FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání bakalářské práce

Název:	Analýza časových řad v exportéru síťových toků
Student:	David Kežlínek
Vedoucí:	Ing. Josef Koumar
Studijní program:	Informatika
Obor / specializace:	Bezpečnost a informační technologie
Katedra:	Katedra počítačových systémů
Platnost zadání:	do konce letního semestru 2023/2024

Pokyny pro vypracování

Nastudujte problematiku monitorování počítačových sítí pomocí síťových toků a problematiku analýzy časových řad se zaměřením na získávání jejich atributů. Navrhněte a implementujte prototyp pluginu pro open source exportér síťových toků ipfixprobe [1] pro rozšíření IP flow záznamů o atributy získané analýzou časové řady paketů. Seznam atributů, které by měl plugin exportovat, dodá vedoucí práce. Otestujte funkcionalitu vytvořeného pluginu a jeho výkonnostní limity. Výsledný exportér síťových toků s pluginem otestujte na reálné síti a vyhodnoťte jeho nasaditelnost do vysokorychlostních sítí.

[1] https://github.com/CESNET/ipfixprobe

Elektronicky schválil/a prof. Ing. Pavel Tvrdík, CSc. dne 30. listopadu 2022 v Praze.

Bakalářská práce

ANALÝZA ČASOVÝCH ŘAD V EXPORTÉRU SÍŤOVÝCH TOKŮ

David Kežlínek

Fakulta informačních technologií
Katedra počítačových systémů
Vedoucí: Ing. Josef Koumar
11. května 2023

Obsah

Seznam obrázků

Seznam tabulek

Děkuji vedoucímu práce Ing. Josefovi Koumarovi za pomoc, odborné rady a připomínky v průběhu vypracovaní práce.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 2373 odst. 2 zákona č. 89/2012 Sb., občanský zákoník, ve znění pozdějších předpisů, tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně imnožstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu) licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 11. května 2023 .....

Abstrakt

Tato práce se v první části zabývá problematikou monitorování počítačových sítí pomocí síťových toků. Dále se práce zabývá analýzou časových řad získaných z těchto síťových toků se zaměřením na získávání jejich atributů.

Výsledkem této práce je nový modul pro open source exportér síťových toků ipfixprobe. Tento nový modul umožňuje rozšířit záznamy o síťových tocích o atributy získané z analýzy časových řad. Tyto atributy mohou v budoucnu sloužit jako vstupní data pro detekci hrozeb na základě strojového učení.

Klíčová slova ipfixprobe, prototyp pluginu pro open source exportér síťových toků, IPFIX, sítové toky, nerovnoměrně vzorkované časové řady, NFFT, Lomb-Scarlge periodogram

Abstract

The first part of this thesis deals with the problem of monitoring computer networks using IP flows. Then it deals with the analysis of time series obtained from these IP flows, focusing on the extraction of their attributes.

The result of this work is a new module for the open source IP flow exporter ipfixprobe. This new module allows to extend IP flow records with attributes extracted from time series analysis. These attributes can be used as input data for machine learning based threat detection in the future.

Keywords ipfixprobe, prototype of plugin for open source flow exporter, IPFIX, flows, unevenly spaced time series, NFFT, Lomb-Scarlge periodogram

Seznam zkratek

CESNET	Czech Education and Scientific NETwork
DNS	Domain Name System
DPI	Deep packet inspection
FFT	Fast Fourier Transform
HTTP	Hypertext Transfer Protocol
IP adress	Internet Protocol address
IPFIX	IP Flow Information Export
MTU	Maximum transmission unit
NFFT	Nonequispaced Fast Fourier Transform

Úvod

Monitorování počítačových sítí je důležitou a nutnou činností, která umožňuje správně provozovat a řídit síťové infrastruktury. V dnešní době jsou počítačové sítě nedílnou součástí běžného života. Monitorování umožňuje odhalit útoky na síťovou infrastrukturu tím, že umožňuje sledovat síťový provoz a identifikovat neobvyklé vzorce chování, které mohou naznačovat přítomnost útočníka
v síti. Pomocí specializovaných nástrojů lze také sledovat aktivitu konkrétních uživatelů a detekovat potenciálně škodlivé činnosti. Monitorování sítí také umožňuje sledovat vytížení infrastruktury a podle toho plánovat její rozšiřování.

Díky monitorování lze také provádět analýzu záznamů a zjistit, jak se útoky odehrály a jaké škody způsobily. Toto může pomoci při zajištění bezpečnosti sítě a při navrhování opatření pro zvýšení ochrany proti budoucím útokům.

Nevýhodou je, že některé metody mohou mít až moc velký zásah do soukromí uživatelů, proto je třeba volit méně invazivní metody monitorování, které nezasahují tolik do soukromí uživatelů a zároveň umožňují detekci útoků na síťovou infrastrukturu.

Cíl práce

Hlavním cílem této práce je vytvořit prototyp nového pluginu do síťového exportéru ipfixprobe. Tento plugin bude získávat a zpracovávat časové řady ze sítových toků. Plugin bude rozšiřovat záznamy síťových toků o nové atributy získané analýzou časové řady paketů v jednom síťovém toku. Atributy, které bude plugin získávat, jsou popsány v kapitole 2. Tyto atributy budou rozděleny do čtyř kategorií a plugin bude počítat pouze atributy v kategoriích, které byly zvoleny při spuštění exportéru ipfixprobe.

Další částí práce je také testování vytvořeného pluginu a jeho výkonnostních limitů. V práci se následně zhodnotí nasaditelnosti do reálných sítí.

Kapitola 1

Monitorování síťových toků

Monitorování sítě je nezbytnou součástí správy moderní sítě. V průběhu let byly navrženy a vyvinuty přístupy k monitorování sítí, přičemž každý z nich slouží jinému účelu. Obecně lze tyto přístupy rozdělit do dvou kategorií: aktivní a pasivní.

Aktivní monitorování sítě zahrnuje aktivní testování sítě pomocí různých nástrojů, jako jsou například nástroje ping a traceroute. Tento typ monitorování umožňuje detekovat a diagnostikovat problémy v síti, jako jsou výpadky, pomalý provoz a další.

Jedním z pasivních přístupů k monitorování sítě je monitorování pomocí exportu síťových toků (Flow-based monitoring). Tato metoda agreguje pakety do síťových toků. Sítový tok (IP flow) je v specifikaci protokolu IPFIX [1] definován jako sada paketů nebo rámců procházejících pozorovacím bodem v síti během určitého časového intervalu. Všechny pakety patřící ke konkrétnímu toku mají sadu společných vlastností, například stejnou zdrojovou a stejnou cílovou IP adresu a stejné cílové a odchozí porty. Sítový tok složený jenom z těchto paketů se označuje jako jednosměrný tok (uniflow), protože obsahuje pouze informace o paketech, které byly odeslány z jednoho koncového bodu do jiného koncového bodu. Obousměrný záznam toku (biflow) obsahuje paketech z obou směrů komunikace mezi dvěma koncovými body.

Monitorování sítě pomocí exportu síťových toků sbírá data o síťových tocích v reálném čase a ty například pomocí protokolu NetFlow nebo IPFIX posílá na kolektor, který data uchovává a dále analyzuje. Tento přístup může využívat i síťových prvků, jako jsou například routery, aby sbíraly data o síťových tocích, které procházejí přes síť. Díky tomu jde výrazně ušetřit za hardware, jelikož je potřeba pouze kolektor Ostatní práci mohou zajistit stávající sítové prvky.

V některých státech (včetně České republiky) mají poskytovatelé internetového připojení povinnost uchovávat data o síťových spojeních v jejich síti, takže dodatečná analýza pro ně nepředstavuje téměř žádné další náklady.

Hloubková kontrola paketů (DPI) na druhé straně zachytává kompletní pakety, které prochází přes síť, a dále tyto kompletně zachycené pakety analyzuje. Tento přístup poskytuje mnohem podrobnější informace o tom, jaká data jsou přenášena v síti a jakým způsobem. To umožňuje identifikovat problémy a provádět diagnostiku, která by jinak mohla být značně obtížná.[2]

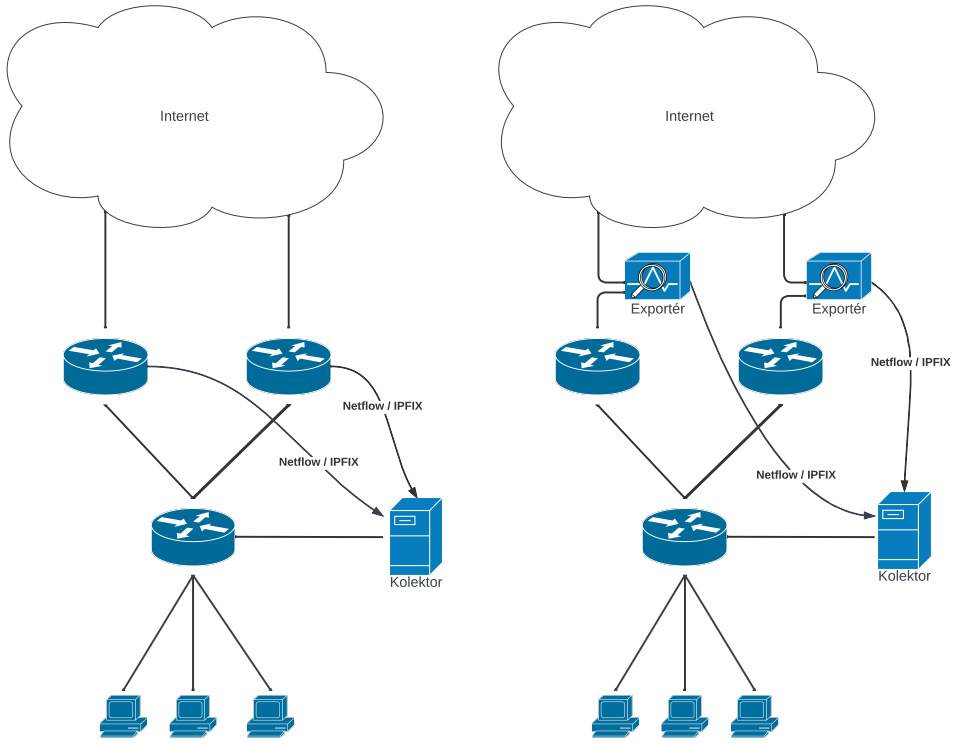
Vzhledem k nárůstu procent šifrované komunikace jsou výhody DPI čím dál menší, protože velká část internetové komunikace je již zašifrovaná. Například Google uvádí, že 95 % webového provozu v Googlu je již šifrováno [3].

1.1 Nástroje pro monitorování síťových toků

Pro monitorování síťových toků se typicky používá několik exportérů síťových toků, které jsou umístěny na okrajích sítě tak, aby mohly zachytit veškerý provoz mezi zařízeními v síti a mimo

ní. Exportéry většinou posílají informace o síťových tocích na jeden centrální kolektor. Kolektor je většinou zařízení s velkou úložnou kapacitou, které sbírá záznamy sítových toků z většího počtu exportérů síťových toků, analyzuje přijatá data a ukládá je do dlouhodobé databáze. Kolektor může také přeposílat data na další zařízení, která je podrobněji analyzují.

Existují dva přístupy jak monitorovat síť. První možností je pro monitoring využít síťové prvky, které jsou již v síti. Některé routery a switche, například od společností Cisco nebo MikroTik, podporují export síťových toků. Druhou možností je umístit na okraje sítě dedikovaná zařízení, na kterých poběží software pro export síťových toků. Výhodou této možnosti je, že na exportérech se může provádět DPI a pomocí rozšířených záznamů o síťových tocích lze posílat získané informace na kolektor. [2] V praxi se ale exportér nenachází přímo páteřním spoji, ale na spoj se umístí zařízení TAP. TAP slouží k duplikování paketů a jejich přeposílání na exportér sítových toků. Díky tomu výpadek exportéru nemůže ohrozit stabilitu sítě.[4]



Obrázek 1.1 Porovnání dvou přístupů monitorování sítí. První s využitím stávajících síťových prvků (nalevo) a druhý s využitím dedikovaných exportérů (napravo)

Pokud jsou exportéry správně nakonfigurovány, mohou být velmi užitečné při detekci anomálií v síťovém provozu, při zjišťování přetížení sítě nebo při analýze chování uživatelů. Díky exportérům mohou administrátoři sítí získat důležité informace o síťovém provozu a včas reagovat na problémy, jako jsou například sítové útoky, které by jinak mohly způsobit výpadek sítě.

Je třeba však mít na paměti, že monitorování síťových toků může být velmi náročné na výpočetní výkon a úložnou kapacitu. Pokud je síť velká a generuje velké množství provozu, mohou být potřeba výkonnější exportéry a kolektory. Navíc je třeba dbát na to, aby exportéry a kolektory byly řádně zabezpečeny, protože záznamy o síťových tocích jsou velmi citlivá data, například některá nezašifrovaná data mouhou obsahovat i hesla uživatelů.

1.1.1 Exportéry

Zde je popsáno několik exportérů, které lze použít pro monitoring sítě.

nProbe [5] je komerční softwarový nástroj, který je vyvíjen společností ntop. nProbe může sloužit jako exportér síťového provozu nebo jako kolektor a analyzovat přijaté záznamy o síťových tocích.

nProbe je rozšiřitelný a obsahuje několik pluginů, které jsou dostupné na základě verze licence. Ty umožňují provádět DPI síťového provozu a analyzovat vybrané protokoly, jako je například HTTP nebo DNS. nProbe pracuje s různými datovými formáty a protokoly, jako jsou například NetFlow a IPFIX a umožňuje exportovat síťové toky do Apache Kafka. Tento nástroj také podporuje různé platformy, včetně Linuxu, Windows a MacOS.

fprobe [6] je open source nástroj založený na libpcap, který shromažďuje data o síťovém provozu a vysílá je jako toky NetFlow nebo IPFIX směrem k zadanému kolektoru. Fprobe dokáže filtrovat pakety, které zpracovává, ale neobsahuje žádné pokročilé nástroje pro DPI. FProbe podporuje export síťových záznamů pouze protokolem NetFlow v1/5/9.

FlowMon Probe [7] je komerční síťový nástroj, určený pro sběr a analýzu síťového provozu v reálném čase. Flowmon Probe podporuje různé formáty sběru dat, včetně NetFlow a IPFIX. Dále například umožňuje analýzu zašifrovaného provozu nebo analýzu paketů na aplikační vrstvě. FlowMon Probe dokáže pracovat na sítích s rychlostmi od 10 Mb/s až do 100 Gb/s.

ipfixprobe [8] je opensource exportér síťových toků. Exportér obsahuje několik pluginů, které umožňují DPI, a protože je opensource, může si uživatel vytvořit vlastní plugin. Tento exportér bude dále detailněji popsán v podkapitole 1.2.

1.1.2 Protokoly pro export síťových toků

Protokoly pro export síťových toků jsou klíčovým prvkem v moderních sítích. Tyto protokoly umožňují exportovat informace o toku dat v síti z exportéru na kolektor, což může být užitečné pro zjištění případných problémů a optimalizaci výkonu sítě.

Existuje několik různých protokolů pro export síťových toků, každý z nich má své výhody a nevýhody.

NetFlow v5 je jeden z prvních protokolů pro sběr informací o síťovém provozu. Jedná se o standardní protokol, který je používán pro monitorování síťového provozu a analýzu toku dat v reálném čase. NetFlow v5 umožňuje získat informace o tom, jaký provoz prochází síťovými zařízeními.

NetFlow v5 je však omezen v tom, že neumožňuje sběr informací o paketech s IPv6 hlavičkami, což může být v dnešní době problém. Dále také neumožňuje exportovat rozšířené síťové toky.

NetFlow v9 [9] je rozšíření protokolu NetFlow, který umožňuje sběr a analýzu informací o síťovém provozu. Oproti NetFlow v5 přináší mnoho vylepšení a nových funkcí, které umožňují efektivnější a detailnější sběr dat o síťovém provozu.

Jedním z hlavních vylepšení v NetFlow v9 je podpora pro IPv6. Toto rozšíření umožňuje sbírat informace o IPv6 paketech, což je důležité v době, kdy se IPv6 pomalu stává standardem pro síťovou komunikaci.

NetFlow v9 také umožňuje dynamické definování šablon, které obsahují dodatečné atributy. Základní záznam tedy může být exportérem pomocí šablony rozšířen o nové informace, například z aplikačních protokolů. Rozšířené záznamy o síťovém toku umožňují sbírat na síti specifické informace, které jsou pro monitoring dané sítě důležité.

IPFIX [1] (Internet Protocol Flow Information Export) je protokol pro sběr a export informací o síťovém provozu. Podobně jako NetFlow, IPFIX umožňuje sledovat síťový provoz. IPFIX, umožňuje definovat několik šablon pro flexibilní definování atributů pro export, a ty pak během exportu libovolně střídat. V literatuře se IPFIX občas označuje jako NetFlow v10.

UniRec [10] je opensource protokol pro efektivní přenos dat. UniRec umožňuje definovat libovolný počet vlastních atributů pomocí šablony. Na rozdíl od IPFIX však podporuje export pouze jedné šablony v rámci jednoho síťového spojení.

1.1.3 Kolektory

Zde jsou popsány kolektory, které lze využít pro sběr dat o síťovém provozu.

IPFIXcol2 [11] je opensource software pro sběr, zpracování a ukládání záznamů síťových toků. IPFIXcol2 je flexibilní a výkonný kolektor s modulární architekturou, která umožňuje snadné přidávání nových modulů. Je navržen tak, aby byl škálovatelný a zvládal velké objemy dat o síťovém provozu.

Jednou z klíčových funkcí IPFIXcol2 je schopnost exportovat data do různých formátů, včetně JSON, UniRec a Apache Kafka. To usnadňuje integraci dat IPFIX do jiných nástrojů a systémů, například nástrojů pro analýzu sítě.

nfcapd [12] je démon pro zachytávání záznamů síťových toků. Patří do sady opensource nástrojů pro práci se síťovými toky nfdump. Nfcapd umožňuje ukládání dat do různých formátů, včetně binárního formátu nfcapd nebo textového formátu na standardním výstupu.

1.2 Ipfixprobe

Ipfixprobe je modulární open source síťový exportér. Tento exportér zpracovává pakety do obousměrných záznamů o síťových tocích. Dále tyto záznamy pomocí výstupních modulů posílá na další zařízení. Tento exportér se skládá ze třech částí a každá část obsahuje několik modulů. [8]

Vnitřní struktura exportéru je zobrazena na obrázku 1.2.

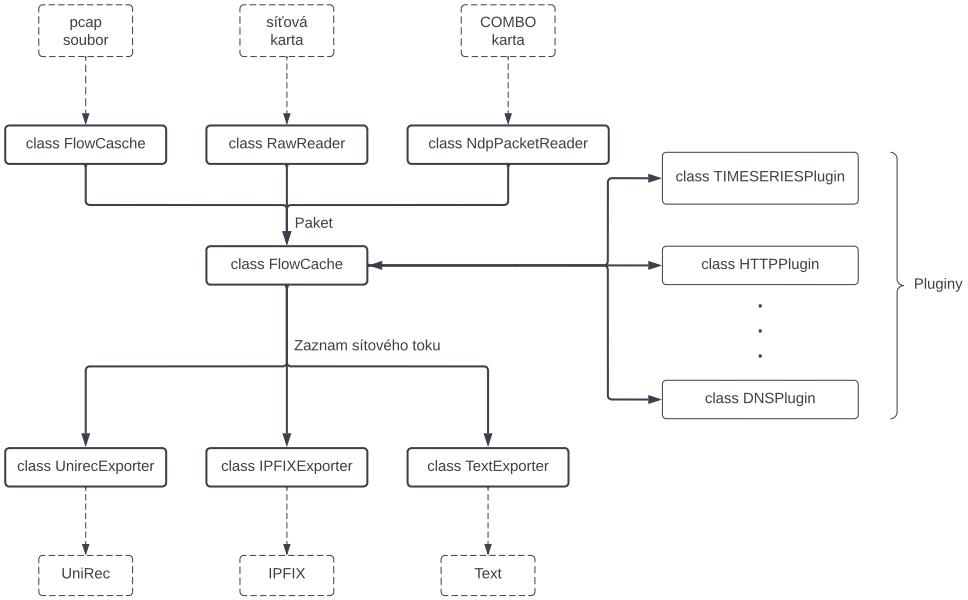
1.2.1 Vstup

Ipfixprobe dokáže přijímat data ze tří různých zdrojů. V závislosti na konfiguraci při spuštění lze vybrat jeden zdroj dat.

Raw - Ipfixprobe dokáže číst pakety ze socketu. Při spuštění může být definováno i více síťových rozhraní, ze kterých ipfixprobe čte současně.

Ndp - Kromě síťových rozhraní dokáže ipfixprobe zpracovávat pakety, které procházejí přes COMBO karty. Tyto karty jsou vývojové desky určené pro zpracování síťových dat a jsou připojeny k počítači pomocí rozhraní PCI. Každá karta obsahuje FPGA (Field-programmable gate array) čip doplněný o paměti, konektory PCI Express, rozhraní, napájecí zdroje a další prvky.

Pcap - Pokud je ipfixprobe zkompilován s parametrem –with pcap, dokáže zpracovávat vstupní data čtením ze souborů typu pcap. K tomu využívá knihovnu libpcap.



Obrázek 1.2 Vnitřní struktura exportéru ipfixprobe.

1.2.2 Procesy

Ipfixprobe dokáže analyzovat zachycené pakety pomocí DPI, což umožňují jednotlivé pluginy, které ipfixprobe obsahuje. Tyto pluginy rozšiřují výsledné záznamy o síťových tocích. Ipfixprobe již obsahuje několik pluginů, jako jsou například DNS, HTTP.

Do této kategorie také spadá plugin, který byl vyvinut v rámci této práce.

1.2.3 Výstup

Získané informace o síťových tocích lze s ipfixprobe dále zpracovávat pomocí pluginů a následně poslat třemi různými způsoby.

IPFIX - Ipfixprobe umožňuje exportovat záznamy o síťovém provozu pomocí protokolu IPFIX na jeden nebo více kolektorů. Cílové kolektory lze nastavit pomocí parametru, zadaného při spuštění ipfixprobe.

UniRec - Pomocí modulu unirec lze ipfixprobe nastavit tak, aby data poslala přímo do systému NEMEA pro další analýzu.

text - Ipfixprobe může vypisovat získané informace o síťovém provozu v textové formě na standardní výstup. Tento výstup lze dále zpracovávat pomocí jiných nástrojů.

Kapitola 2

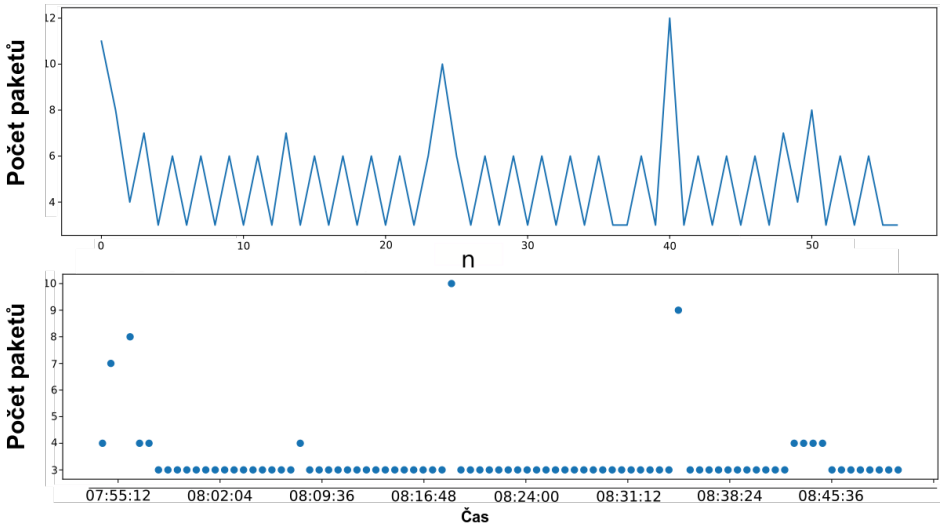
Analýza časových řad

Časové řady jsou soubory dat získaných z pozorování, která jsou uspořádána v časovém pořadí. Tyto řady se používají v různých oblastech, například v ekonomii, astronomii, meteorologii nebo výrobním průmyslu, kde je třeba analyzovat data v čase.

Definice časové řady [13]:

▶ **Definice 2.1.** *Časová řada je soubor pozorování (označených například jako x_t) získaných v konkrétních časových okamžicích $t := 1, 2, ..., n$.*

V rámci časových řad můžeme rozlišovat i nerovnoměrně vzorkované časové řady [14], což jsou řady, kde jsou data zachycena v nepravidelných intervalech. Nerovnoměrně vzorkovanou časovou řadu lze definovat jako soubor pozorování (označených například jako x_i), získaných v konkrétních časových okamžicích t_i, pro které platí $t_i \leq t_{i+1}$ pro $i := 1, 2, ..., n$.



Obrázek 2.1 Porovnání rovnoměrně vzorkované (horní graf) a nerovnoměrně vzorkované časové řady (spodní graf) ze síťového provozu

2.1 Volba typu časové řady

Existují dvě možnosti, jak získat časovou řadu ze síťového toku. První možností je zvolit agregační interval a během tohoto intervalu sčítat velikosti všech přijatých paketů do jednoho datového bodu. Druhou možností je použít nerovnoměrně vzorkovanou časovou řadu a každou velikost příchozího paketu vložit do této nerovnoměrně vzorkované časové řady.

Níže jsou uvedeny výhody a nevýhody použití nerovnoměrně vzorkované časové řady oproti rovnoměrně vzorkované časové řadě.

- Výhody nerovnoměrně vzorkované časové řady:
  - Pakety se po síti většinou neposílají v rovnoměrných intervalech a mezi jednotlivými pakety mohou být dlouhé časové intervaly. Tyto intervaly by u rovnoměrně vzorkované časové řady byly zaplňovány nulami.
  - Při použití nerovnoměrně vzorkované časové řady není potřeba volit agregační interval. Volba univerzálního intervalu pro všechny časové řady by byla značně náročná, či dokonce nemožná.
  - Maximální hodnota y_i v nerovnoměrně vzorkované časové řadě se bude rovnat maximální velikosti datové části paketu.
  - Detekce periodicky opakujícího paketu by byla kvůli agregační metodě hůře vysvětlitelná z pohledu výsledků a nutně by docházelo ke ztrátě dat či špatné klasifikaci.
- Nevýhody nerovnoměrně vzorkované časové řady:
  - Ipfixprobe zachytává pakety na síti s MTU 1500, ale pakety na aplikační vrstvě mohou být větší. Použití agregační metody by umožnilo pracovat s reálnějšími daty.
  - Při použití nerovnoměrně vzorkované časové řady musí být uchovány časy, kdy byly pakety přijaty.
  - Rovnoměrně vzorkované časové řady jsou snáze zpracovatelné a existuje více metod pro jejich analýzu.
  - Pokud dojde k fragmentaci paketu během jeho přenosu po síti, mohou v různých místech měření vzniknout různé časové řady ze stejného datového toku.

Z analýzy výše uvedených výhod a nevýhod nakonec vyplynulo, že nerovnoměrně vzorkované časové řady jsou vhodnější pro využití v pluginu.

2.2 Rozdělení atributů do skupin

V následujících podkapitolách jsou popsány atributy, které plugin exportuje. Atributy jsou rozděleny do čtyř skupin na základě dat, ze kterých se dají vypočítat. Například pro výpočet atributů v podkapitole Statistické atributy stačí pouze histogram, zatímco pro výpočet atributů v podkapitole Frekvenční atributy je potřeba kompletní nerovnoměrně vzorkovaná časová řada. Díky tomuto rozdělení je zajištěno, že plugin je efektivní i když nejdou spuštěny všechny jeho části.

2.3 Statistické atributy

V této kapitole jsou popsány převážně statistické atributy, které plugin exportuje. Atributy v této skupině lze vypočítat pouze z histogramu časové řady. V tabulce 2.1 je seznam těchto atributů, které jsou dále v této kapitole detailněji popsány. Všechny atributy lze spočítat z nerovnoměrně vzorkované časové řady o délce n a hodnotách x_i pro $i := 1, 2, ..., n$. Časy datových bodů pro žádný atribut nejsou potřeba.

Název atributu v pluginu	Český název
TS_MEAN	Průměr
TS_STDEV	Směrodatná odchylka
TS_VAR	Variance
TS_BURSTINESS	Burstiness
TS_Q1	První kvartil
TS_MEDIAN	Medián
TS_Q3	Třetí kvartil
TS_MIN	Minimální hodnota
TS_MAX	Maximální hodnota
TS_MODE	Modus
TS_COEFFICIENT_OF_VARIATION	Koeficient variace
TS_AVERAGE_DISPERSION	Průměrný rozptyl
TS_PERCENT_DEVIATION	Procentuální odchylka
TS_ROOT_MEAN_SQUARE	Kvadratický průměr
TS_PERCENT_BELOW_MEAN	Kolik procent prvků je větší než průměr
TS_PERCENT_ABOVE_MEAN	Kolik procent prvků je menší než průměr
TS_PEARSON_SK1_SKEWNESS	Pearsonův koeficient sklonu 1. řádu
TS_PEARSON_SK2_SKEWNESS	Pearsonův koeficient sklonu 2. řádu
TS_FISHER_MI_3_SKEWNESS	Fisherův koeficient sklonu 3. řádu
TS_GALTON_SKEWNESS	Galtonův koeficient sklonu
TS_KURTOSIS	Koeficient špičatosti
TS_ENTROPY	Entropie časové řady
TS_SCALED_ENTROPY	Normalizovaná entropie
TS_P_BENFORD	Benfordův test

Tabulka 2.1 Seznam statistických atributů

Mean

Průměr je statistický ukazatel, který se používá k vyjádření střední hodnoty datové sady. Vypočítá se jako součet všech hodnot v časové řadě dělený počtem hodnot.

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{2.1}$$

Var, STDEV

Rozptyl (σ^2) a směrodatná odchylka (σ) se počítají pomocí následujícího vzorce.

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n} (x_i - \mu)^2 \tag{2.2}$$

Burstiness

Burstiness [15] je statistický koncept, tendence nějakého jevu k výskytům ve shlucích nebo obdobích vysoké aktivity, následovaných obdobími nízké aktivity. Burstiness je tedy charakteristikou, která popisuje, zda jsou výskyty daného jevu náhodné, nebo zda se vyskytují ve shlucích. Burstiness je v intervalu (-1, 1).

B = 1 – výskyty daného jevu jsou ve shlucích.

B = 0 – výskyty jevu jsou neutrální.

B = -1 – výskyty daného jevu jsou pravidelné.

Q1, Median, Q3 Kvantily jsou statistické ukazatele, které rozdělují uspořádanou množinu hodnot na určený počet stejně velkých částí. Například medián ($\widetilde{x}$), což je speciální druh kvantilu, rozděluje uspořádanou množinu hodnot na dvě stejně velké části, takže 50 % hodnot je menších než medián a 50 % hodnot je větších než medián.

Existují různé kvantily, které se liší podle počtu částí, na které se množina hodnot rozděluje. Například kvartily rozdělují množinu hodnot na čtyři stejně velké části (každá o velikosti 25 %) a označují se Q_1,Q_2 a Q_3, tercily na tři stejně velké části (každá o velikosti 33,33 %).

$$\widetilde{x} = Q_2 = x'_{\frac{n+1}{2}} \tag{2.3}$$

$$Q_1 = x'_{\frac{n+1}{4}} \tag{2.4}$$

$$Q_3 = x'_{\frac{3(n+1)}{4}} \tag{2.5}$$

kde x' je seřazená posloupnost datových bodu. [16]

Min, Max
Min je nejmenší hodnota časové řady a max je největší hodnota v časové řadě.

Mode
Mode je nejčastější prvek v časové řadě, dále je označován pomocí symbolu η.

Coefficirnt of variation
Poměr směrodatné odchylky a průměru časové řady.

$$cv = \frac{\sigma}{\mu} \tag{2.6}$$

Average dispertion
Průměrná odchylka od průměru časové řady.

$$ad = \frac{1}{n}\sum_{i=1}^{n}|x_i - \mu| \tag{2.7}$$

Percent deviation
Procentuální odchylka od průměru časové řady.

$$pd = \frac{ad}{\mu} \tag{2.8}$$

Root mean square
Kvadratický průměr je statistický ukazatel, který se dá spočítat pomocí následujícího vzorce [17].

$$rms = \sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2} \tag{2.9}$$

Percent below mean
Tento atribut značí, kolik procent datových bodů má velikost nižší, než je průměr.

Percent above mean
Tento atribut značí, kolik procent datových bodů má velikost vyšší, než je průměr.

Pearson SK1 skewness

Šikmost je mírou asymetrie rozdělení. Rozdělení je asymetrické, pokud jeho levá a pravá strana je symetrická. Rozdělení může mít pravou (kladnou), levou (zápornou) nebo nulovou šikmost. Existuje více způsobů, jak šikmost spočítat. Jeden ze způsobů je Pearsonův první koeficient šikmosti, ten je definován vztahem [18]:

$$sk_1 = \frac{\mu - M_o}{\sigma} \tag{2.10}$$

Pearson SK2 skewness

Pearsonův druhý koeficient šikmosti je definován vztahem [18]:

$$sk_2 = \frac{3\mu - \widetilde{x}}{\sigma} \tag{2.11}$$

Fisher MI 3 skewness

Fisherů koeficient šikmosti:

$$mi_3 = E\left[\left(\frac{\{x_n\} - \mu}{\sigma}\right)^3\right] = \frac{\sum_{i=1}^{k}(p_i h_i^i) - 3\mu\sigma^2 - \mu^3}{\sigma^3} \tag{2.12}$$

kde h_i je četnost datového bodu i v časové řadě a $p_i = h_i/n$

Galton skewness

Galtonova míra šikmosti:

$$G_s = \frac{Q_1 + Q_3 - 2\mu}{Q_3 - Q_1} \tag{2.13}$$

Kurtosis

Koeficient špičatosti je charakteristika, která porovnává rozdělení dat s normálním rozdělením. Špičatost K udává špičatost/plochost rozdělení.[19]

- K $>$ 3 - rozdělení je špičatější než normální rozdělení.
- K $=$ 3 - rozdělení je stejné jako normální rozdělení.
- K $<$ 3 - rozdělení je plošší než normální rozdělení.

$$Kurt = \frac{1}{n\sigma^4}\sum_{i=1}^{n}(x_i - \mu)^4 \tag{2.14}$$

Entropy

Atribut entropie udává míru náhodnosti velikosti datových bodů.

$$H(x_n) = -\sum_{i=1}^{n} p_i log_2 p_i \tag{2.15}$$

kde h_i je četnost datového bodu i v časové řadě a $p_i = h_i/n$

Scaled entropy

Tento atribut lze spočítat pomocí:

$$H_s(x_n) = \frac{H(x_n)}{-log_2 \frac{1}{n}} \tag{2.16}$$

Tabulka 2.2 Časové atributy

Název atributu v pluginu	Český název
TS_MEAN_SCALED_TIME	Průměrný čas seškálovaný
TS_MEDIAN_SCALED_TIME	Medián času seškálovaný
TS_Q1_SCALED_TIME	1. kvartil času seškálovaný
TS_Q3_SCALED_TIME	3. kvartil času seškálovaný
TS_DURATION	Délka časové řady
TS_MIN_DIFFTIMES	Minimální rozdíl časů
TS_MAX_DIFFTIMES	Maximální rozdíl časů
TS_MEAN_DIFFTIMES	Průměrný rozdíl časů
TS_MEDIAN_DIFFTIMES	Medián rozdílů časů
TS_DIFFTIMES_SKEWNESS	Šikmost rozdělení rozdílů časů
TS_DIFFTIMES_KURTOSIS	Klasický koeficient špičatosti
TS_TIME_DISTRIBUTION	Časové rozložení

P Benford

Tento atribut popisuje pravděpodobnost, že počty výskytů prvních 9 nejčastějších datových bodů odpovídají Benfordovu zákonu [20]. Počítá se pomocí:

$$P_{\text{BENFORD}} = 1 - \frac{1}{2}\sum_{i=1}^{9}\left(log_{10}\left(1+\frac{1}{d}\right) - \frac{h_i}{n}\right) \tag{2.17}$$

kde h_i je počet výskytů daného datového bodu, n je počet všech datových bodů a d je hodnota datového bodu.

2.4 Časové atributy

V tabulce 2.2 je seznam atributů v této části. Atributy jsou zde následně popsány.

Mean scaled time

Průměrný čas datového bodu, který je upraven (scaled) na určitou škálu.

$$\mu_T = \frac{1}{nT_n}\sum_{i=1}^{n} T_i) \tag{2.18}$$

kde $T_i = t_i - t_0$.

Median scaled time

Medián ($\widetilde{x}$) rozděluje časy datových bodů na dvě stejně velké části, takže 50 % časů je menších než medián a 50 % časů je větších než medián.

$$\widetilde{t} = Q_{t2} = t_{\frac{n+1}{2}}/T_n \tag{2.19}$$

Q1 scaled time

První kvartil rozděluje množinu hodnot časů na dvě části. První část o velikosti 25 %, která obsahuje prvky menší než první kvartil. Kvartil byl škálován.

$$Q_{t1} = t_{\frac{n+1}{4}}/T_n \tag{2.20}$$

Q3 scaled time

Třetí kvartil rozděluje množinu hodnot časů na dvě části. První část o velikosti 75 %, která obsahuje prvky menší než třetí kvartil. Kvartil byl škálován.

$$Q_{t3} = t_{\frac{3(n+1)}{4}}/T_n \tag{2.21}$$

Duration

Délka časové řady.

$$Duration = t_n - t_0 \tag{2.22}$$

Min difftimes

Minimální čas mezi jednotlivými datovými body v časové řadě.

$$min_{dt} = min(dt_1, dt_2, ..., dt_n) \tag{2.23}$$

kde $dt_i = t_i - t_{i-1}$,

Max difftimes

Maximální čas mezi jednotlivými datovými body v časové řadě.

$$max_{dt} = max(dt_1, dt_2, ..., dt_n) \tag{2.24}$$

kde $dt_i = t_i - t_{i-1}$,

Mean difftimes

Průměrný čas mezi jednotlivými datovými body v časové řadě.

$$\mu_T = \frac{1}{n}\sum_{i=1}^{n} dt_i \tag{2.25}$$

kde $dt_i = t_i - t_{i-1}$,

Median difftimes

Median časů mezi jednotlivými datovými body v časové řadě.

$$\widetilde{dt} = dt'_{\frac{n+1}{2}} \tag{2.26}$$

kde dt' je seřazená posloupnost.

Difftimes skewness

Šikmost rozdělení časů mezi jednotlivými datovými body v časové řadě. V pluginu se používá stejný vzorec jako pro Pearson SK2 skewness 2.11.

Difftimes kurtosis

Koeficient špičatosti rozdělení časů mezi jednotlivými datovými body v časové řadě. Pro výpočet se používá stejný vzorec jako pro Kurtosis v statistických atributech 2.14.

Time distribution

Tento atribut popisuje rozložení časových rozdílů mezi jednotlivými pakety. Čím je *tdist* nižší tím jsou časové rozdíly lépe rozloženy v čase.

$$tdist = \frac{Y_{DT}}{S_{DT}}, \qquad Y_{DT} = \frac{\sum_{i=1}^{n} |\mu - dt_i|}{n}, \qquad S_{DT} = \frac{max_{DT} - min_{DT}}{2} \tag{2.27}$$

Tabulka 2.3 Behaviorální atributy

Název atributu v pluginu	Český název
TS_HURST_EXPONENT	Hurstův exponent
TS_SWITCHING_METRIC	Metrika pro detekci změn
TS_DIRECTIONS	Směry
TS_PERIODICITY_TIME	Doba periodicity
TS_PERIODICITY_VAL	Hodnota datového bodu který se periodicky opakuje

2.5 Behaviorální atributy

V následující tabulce je seznam atributů, které bude tato část pluginu exportovat 2.3. Jednotlivé atributy jsou zde popsány.

Hurst exponent

Hurstův exponent H umožňuje zjistit tendenci časové řady k regresi. Pokud je H v intervalu $\langle 0; 0,5)$, pak pravděpodobně dochází k přepínání mezi vysokými a nízkými hodnotami v sousedních dvojicích. To znamená, že pokud je hodnota jednoho datového bodu nízká, tak následující datový bod má tendenci mít vysokou hodnotu. Pokud $H \sim 0,5$, pak to znamená že se jedná o náhodnou (nekorelovanou) časovou řadu. Dále pokud $H \in (0,5; 1>$, to ukazuje na dlouhodobou kladnou autokorelaci v časové řadě. Časová řada je pak označována jako perzistentní. Jedna z možností jak odhadnout Hurstův exponent je použít metodu Rescaled range (R/S).

Tuto metodu lze rozdělit na několik kroků. Nejprve se časová řada o délce N rozdělí na několik kratších časových řad o velikostech $n \in \{N, N/2, N/4, N/8, N/16, ...\}$. Pro každé n rozdělíme časovou řadu X na intervaly $[X_1, X_n], [X_{n+1}, X_{2n}], ..., [X_{m-1}n + 1, X_{mn}]$ kde $m = \lfloor N/n \rfloor$.

1. Spočítat průměr intervalu.
$$m = \frac{1}{n} \sum_{i=1}^{n} X_i$$
2. Spočítat časové řady Y_i
$$Y_i = X_i - m$$
kde $i := 1, 2, ..., n$.
3. Vypočítat kumulativní řady Z_i.
$$Z_i = \sum_{j=1}^{i} Y_j$$
kde $i := 1, 2, ..., n$.
4. Spočítat řady R_i.
$$R_i = max(Z_1, Z_2, ...Z_i) - min(Z_1, Z_2, ...Z_i)$$
kde $i := 1, 2, ..., n$.
5. Spočítat řady S_i.
$$S_i = \sqrt{\frac{1}{i} \sum_{j=1}^{i} (X_i - \mu)^2}$$
kde μ je průměr z intervalu $[X_1, X_i]$ a $i := 1, 2, ..., n$.

6. Spočítat průměry $(R/S)_n$ pro tento interval.

$$(R/S)_n = \frac{1}{t}\sum_{i=1}^{t} R_i/S_i$$

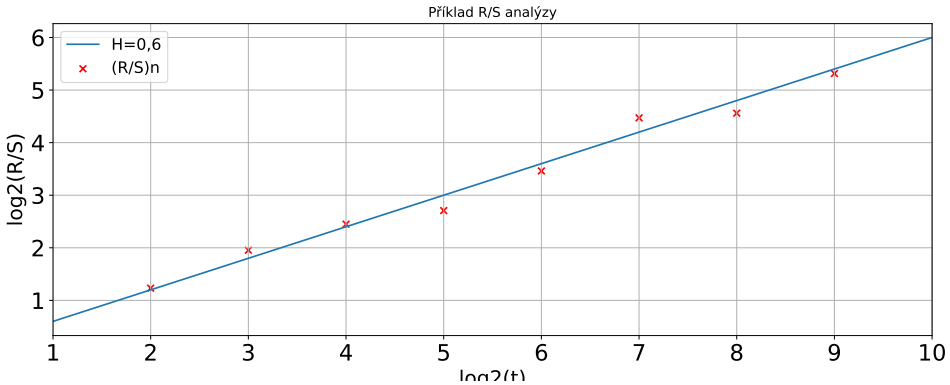
Výsledný $(R/S)_t$ a $t = n$ je průměr ze všech $(R/S)_n$ z intervalů délky n.

Hurstův exponent poté vypočítáme z následující rovnice:

$$(R/S)_t = c * t^H$$

kde c je konstanta a H je hledaný Hurstův exponent.

Hurstův exponent odhadneme tak, že na logaritmické ose vykreslíme graf závislosti (R/S) na t. Sklon přímky aproximuje Hurstův exponent viz obrázek 2.2. Pro t < 10 nemusí být $(R/S)_t$ přesný.[21] [22]



Obrázek 2.2 Příklad výsledné přímky získané z R/S analýzy

Tento přístup nelze použít pro nerovnoměrně vzorkované časové řady. Jak je zmíněno v článku [23] lze tuto metodu upravit tak, aby mohla zpracovat i nerovnoměrně vzorkovanou časovou řadu.

Stejně jako v neupravené R/S metodě si rozdělíme časovou řadu (o délce N, hodnotách X_N a časech T_N) na intervaly o velikostech $n \in \{N, N/2, N/4, N/8, N/16, ...\}$

1. Spočítat průměr intervalu o délce n.

$$m = \frac{1}{T_n - T_0}\sum_{i=1}^{n}(X_i(T_i - T_{i-1}))$$

2. Spočítat kumulativní řadu Z.

$$Z_i = \sum_{j=1}^{i}((X_j - m)(T_j - T_{j-1}))$$

kde $i := 1, 2, ..., n$.

3. Spočítat R.

$$R = max(Z_1, Z_2, ...Z_i) - min(Z_1, Z_2, ...Z_i)$$

kde $i := 1, 2, ..., n$.

4. Spočítat S.

$$S = \sqrt{\frac{1}{T_n - T_0} \sum_{j=1}^{n} ((X_i - \mu)^2 (T_i - T_{i-1}))}$$

kde μ je průměr z intervalu $[X_1, X_j]$ a $i := 1, 2, ..., t$.

5. Spočítat $(R/S)_n$ pro tento interval.

$$(R/S)_n = R/S$$

Hurstův exponent získáme z následující rovnice:

$$(R/S)_n = c * (t_n - t_0)^H$$

kde c je konstanta a H je hledaný Hurstův exponent. [23]

Switching metric

Tento parametr znázorňuje poměr počtu přepnutí mezi různými hodnotami počtu bajtů v paketu.

$$Switching\_metrika := \frac{s_n}{S} \tag{2.28}$$

kde $s_n :=$ počet přepnutí z nějaké hodnoty do jiné a $S := \frac{n-1}{2}$

Directions

Ipfixprobe v základním nastavení do jednoho síťového toku zahrnuje pakety v obou směrech, je třeba nějak určit, kolik paketů bylo po síti posláno jakým směrem. K tomu slouží tento atribut.

$$DIRECTIONS := \frac{dir_1}{n} \tag{2.29}$$

kde dir_1 je počet paketů ve směru prvního zachyceného paketu z oboustranného síťového toku a n je počet všech paketů v oboustranném síťovém toku.

Periodicity time, Periodicity val

Periodicita je zde označována jako chování, kde se opakuje paket o stejné velikosti, v přesně stejném intervalu neustále dokola. Pokud časová řada neobsahuje příliš šumu, lze tento opakující se paket nalézt pomocí histogramu časové řady.

Zde je primitivní algoritmus 1 pro nalezení periodicity. Tento algoritmus je záměrně jednoduchý, protože lepší způsob pro hledání periodicity je Lomb-Scarlge periodogram, a ten je implementovaný v následující části.

2.6 Frekvenční atributy

Využití frekvenční analýzy časových řad umožňuje zjištění řady závislostí, které nejsou na klasické časové ose zjevné. Nejprve je nutné převést časovou řadu do frekvenční domény. Jedna z možností, jak toho docílit, je použít Fourierovu transformaci.

Další variantou je použít metodu Minimalizace fázového rozptylu (PDM). To je technika analýzy dat, která hledá periodické složky souboru dat časových řad. Je užitečná pro datové soubory s mezerami, nesinusovými změnami, špatným časovým pokrytím nebo jinými problémy, kvůli kterým by Fourierovy techniky nebyly použitelné. Jádrem metody je opakovaný pokus a omyl, nebo odhad periody a následným porovnáváním odhadnutých a skutečných dat.[24]

Metoda PDM je ale zbytečně výpočetně náročná a místo ní lze využít rychlejší a více používanou metodu založenou na FFT.

Algorithm 1 Algoritmus pro detekci periodicity

Require: sizeof(TS)≥ 4
histogram = $getHistogram(TS)$
for `Pro každý prvek v histogramu` **do**
  if $\frac{poret\_vyskytu\_prvku}{pocet\_vsech\_prvku} \geq 0{,}95$ **then**
    nalezený prvek = i;
    break;
  end if
end for
if Nebyl nalezen řádný prvek **then**
  return Prvek nenalezen;
end if
Mapa ČasovéMezery;
for `Pro každý prvek časové řady který se rovná nalezenému prvku` **do**
  ČasovéMezery.add(čas, který uplynul mezi tímto a předchozím prvkem stejné velikosti)
end for
$nalezenyprvek = i$;
if Počet nejčastěji nalezených časových mezer mezi pakety o velikosti nalezeného prvku ≤ 1 **then**
  return Prvek nenalezen;
end if
PERIODICITY_TIME = Nejčastěji nalezená časová mezera mezi pakety o velikosti nalezeného prvku
PERIODICITY_VAL = nalezený prvek

2.6.1 Fourierova transformace

V roce 1822 Joseph Fourier [25] přišel s tvrzením, že jakákoliv funkce, spojitá či nespojitá, lze rozepsat do řady sínů. Toto tvrzení využívá Fourierova transformace, která umožňuje rozkládat signály na jejich sinusové složky s různými frekvencemi a amplitudami. Tento rozklad umožňuje snadnou analýzu signálů a umožňuje zjistit, jaké dominantní frekvence obsahuje.

2.6.2 Lomb-Scarlge periodogram

Furierovu tranformaci ale nelze použít pro nerovnoměrně vzorkovanou časovou řadu, protože ta musí mít na vstupu rovnoměrně vzorkovanou časovou řadu. Pokud by na vstupu byla nerovnoměrně vzorkovaná časová řada, tak by výsledek téměř jistě nedával smysl.

Právě z tohoto důvody byl vyvinut Lomb-Scargle periodogram (LS periodogram). Základ tohoto periodogramu byl položen v článcích od N. R. Lomba [26] a J. Scargla [27]. Tato metoda umožňuje efektivní výpočet odhadu Fourierova výkonového spektra z takto nerovnoměrně vzorkovaných dat, což vede k určení periody oscilací. Díky tomu lze najít dominantní sinusové signály v datech.

Lomb-Scarlge periodogram byl vyvinut pro použití v astronomii. Tam se používá pro analýzu periodických signálů v hvězdných spektrech. Ty nelze analyzovat pomocí Furierových transformací, protože pozorování nejde kvůli počasí či jiným vlivům provádět nepřetržitě.[28]

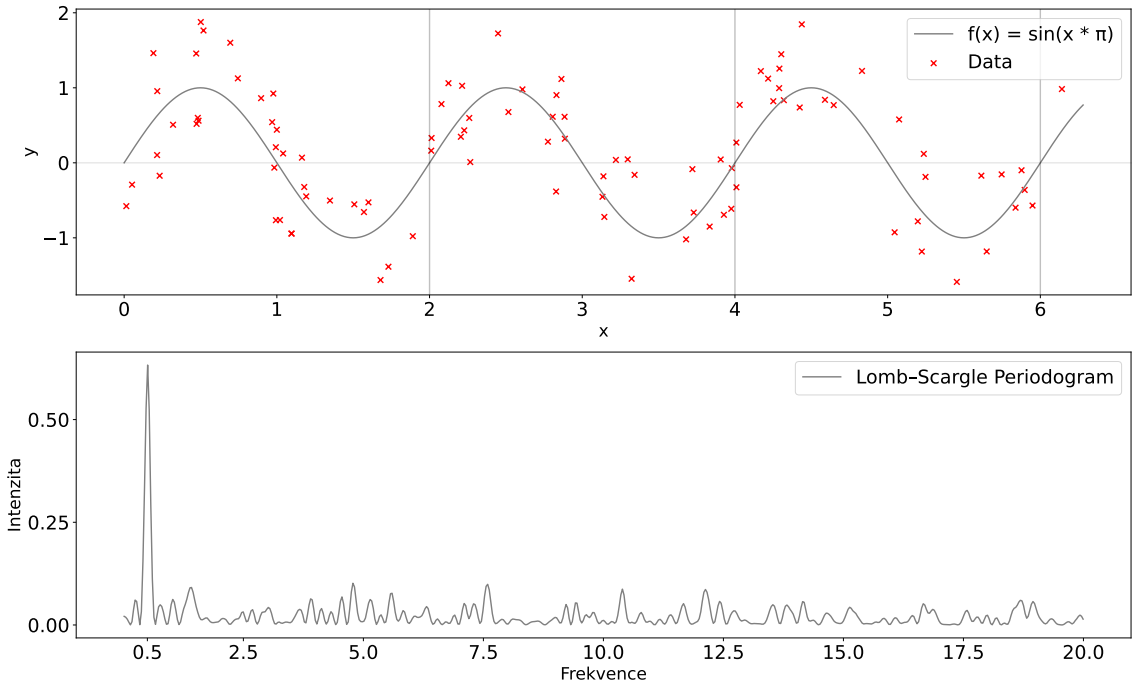
Základní algoritmus získání Lomb-Scarlge periodogramu je dle vzorce:

$$P_{LS}(f) = \frac{1}{2}\frac{\left[\sum_{j=1}^{N} y_j \cos(2\pi f(t_j-\tau))\right]^2}{\sum_{j=1}^{N}\cos^2(2\pi f(t_j-\tau))} + \frac{\left[\sum_{j=1}^{N} y_j \sin(2\pi f(t_j-\tau))\right]^2}{\sum_{j=1}^{N}\sin^2(2\pi f(t_j-\tau))} \tag{2.30}$$

kde y_i je velikost datového bodu i a t_i je čas datového bodu i, τ je určeno pro každé f zvlášť pomocí rovnice:

$$\tau = \frac{1}{4\pi f} \tan^{-1}\left(\frac{\sum_{j=1}^{N} \sin 4\pi f t_j}{\sum_{j=1}^{N} \cos 4\pi f t_j}\right) \tag{2.31}$$

Na obrázku 2.3 je příklad použití Lomb–Scargle Periodogramu. Vstupní data byla získána z funkce $f(x) = \sin(x\pi)$ a byl do nich přidán náhodný šum. Ve spodním grafu je vidět, že největší vrchol v Lomb–Scargle Periodogramu je na frekvenci $0,5\ Hz$, a to přesně odpovídá funkci $f(x)$.



Obrázek 2.3 Vstupní data do Lomb–Scargle Periodogramu (horní graf) a Lomb–Scargle Periodogram (spodní graf).

2.6.3 Atributy

V tabulce 2.4 jsou uvedeny všechny atributy v táto skupině. Atributy se počítají z předem vypočítaného Lomb-Scargle periodogramu $P_{LS}(f)$ kde $f \in \{f_0, \ldots, f_n\}$.

Min power, Max power
Tyto atributy reprezentují minimální a maximální nalezený výkon v Lomb-Scarlge periodogramu.

Min power freq, Max power freq
Tyto atributy reprezentují frekvence, na kterých byl nalezen minimální a maximální výkon v Lomb-Scarlge periodogramu.

Název atributu v pluginu	Český název
TS_MIN_POWER	Minimální výkon
TS_MAX_POWER	Maximální výkon
TS_MIN_POWER_FREQ	Frekvence minimálního výkonu
TS_MAX_POWER_FREQ	Frekvence maximálního výkonu
TS_POWER_MEAN	Průměrný výkon
TS_POWER_STDEV	Standardní odchylka výkonu
TS_SPECTRAL_ENERGY	Spektrální energie
TS_SPECTRAL_ENTROPY	Spektrální entropie
TS_SPECTRAL_KURTOSIS	Spektrální kurtóza
TS_SPECTRAL_SKEWNESS	Spektrální šikmost
TS_SPECTRAL_ROLLOFF	Spektrální rolloff
TS_SPECTRAL_CENTROID	Spektrální střed
TS_SPECTRAL_SPREAD	Spektrální rozptyl
TS_SPECTRAL_SLOPE	Spektrální sklon
TS_SPECTRAL_CREST	Spektrální crest
TS_SPECTRAL_FLUX	Spektrální flux
TS_SPECTRAL_BANDWIDTH	Spektrální šířka pásma
TS_PERIODICITY_SCDF	Pravděpodobnost periodicity SCDF

Tabulka 2.4 Seznam frekvenčních atributů

Mean power

Průměrný výkon v Lomb-Scarlge periodogramu.

$$\mu_p = \frac{1}{n}\sum_{i=1}^{n} P_{LS}(f_i)$$

STDEV power

Standardní odchylka (σ) výkonu v Lomb-Scarlge periodogramu.

$$\sigma_p = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(P_{LS}(f_i) - \mu_f\right)^2}$$

Spectral energy

Tento atribut představuje celkovou energii v celém Lomb-Scarlge periodogramu. Lze ho spočítat sečtením výkonu na všech frekvencích.

$$S_e = \sum_{i=1}^{n} P_{LS}(f_i)$$

Spectral entropy

Spektrální entropie je stupeň náhodnosti v Lomb-Scarlge periodogramu. Lze ji spočítat pomocí následujícího vzorce [29]:

$$S_E = -\sum_{i=1}^{n}(P_{LS}(f_i) * \ln P_{LS}(f_i))$$

Spectral kurtosis

Spektrální koeficient špičatosti udává špičatost/plochost rozdělení. Pro výpočet se používá stejný vzorec jako pro Kurtosis v statistických atributech 2.14.

Spectral skewness

Spektrální šikmost je statistický ukazatel, který popisuje asymetrii spektra kolem jeho středu. Spektrální šikmost se dá spočítat stejným vzorcem jako skewness ve statistických atributech. V pluginu se používá vzorec pro Pearson SK2 skewness 2.11.

Spectral rolloff

Atribut spektrální útlum je definován jako frekvence, pod kterou se koncentruje 95 % distribučního výkonu. Tedy součet výkonu na všech frekvencích menších než Spectral rolloff se rovná 95 % celkové energii v celém Lomb-Scarlge periodogramu [30]. Pro účely monitorování sítí byla ale nakonec zvolena hranice 85 %.

Je definován následujícím vztahem:

$$\sum_{i=1}^{S_r} P_{LS}(f_i) = 0.85 * \sum_{j=1}^{n} P_{LS}(f_j) \tag{2.32}$$

Spectral centroid

Spektrální střed je vážený průměr frekvencí výkonového spektra a indikuje frekvenci, na které je energie spektra soustředěna [31].

$$S_c = \frac{\sum_{i=1}^{n}(f_i P_{LS}(f_i))}{\sum_{i=1}^{n} P_{LS}(f_i)} \tag{2.33}$$

Spectral spread

Spektrální rozptyl je rozdíl mezi nejvyšší a nejnižší frekvencí ve výkonovém spektru. Vypočítat lze pomocí následujícího vzorce [30]:

$$S_{sp} = \sqrt{\frac{\sum_{i=1}^{n}(f_i - S_c)^2 P_{LS}(f_i)}{\sum_{i=1}^{n} P_{LS}(f_i)}} \tag{2.34}$$

Spectral slope

Spektrální sklon představuje sklon trendu výkonového spektra. Lze ho spočítat pomocí následující rovnice [30].

$$S_{sl} = \frac{\sum_{i=1}^{n}(f_i - \mu_f)(f_i - \mu_p)}{\sum_{i=1}^{n}(f_i - \mu_f)^2} \tag{2.35}$$

kde μ_f je průměr frekvencí a μ_p je průměr výkonového spektra. Spektrální sklon lze zobrazit v grafu 2.4 pomocí následující funkce:

$$a(f) = S_{sl} * f + const$$

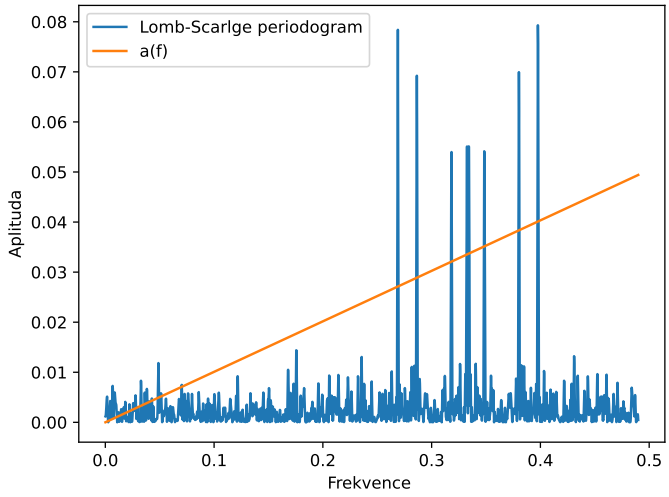
Spectral crest

Výkyv spekra je atribut, který představuje poměr mezi maximálním výkonem Lomb-Scarlge periodogramu a průměrným výkonem.

$$S_{cr} = \frac{MAX\_POWER}{\mu_p} \tag{2.36}$$

Spectral flux

Spektrální flux je součet všech změn výkonu mezi jednotlivými frekvencemi. Lze spočítat pomocí následující rovnice [31]:



Obrázek 2.4 Sklon trendu výkonového spektra

$$S_F = \sum_{i=2}^{n} |P_{LS}(f_n) - P_{LS}(f_{n-1})| \tag{2.37}$$

Spectral bandwidth

Tento atribut popisuje spektrální šířku pásma řádu $p = 2$. Cílem tohoto atributu je popsat rozdíl mezi horní a dolní frekvencí, při níž je spektrální energie poloviční než její maximální hodnota [32].

$$S_b = \sum_{i=1}^{n} P_{LS}(f_i) \, (f_i - S_c)^{\frac{1}{p}} \tag{2.38}$$

Periodicity SCDF

Tento atribut popisuje jak je pravděpodobné, že maximální nalezený výkon v periodogramu je statisticky významnější než ostatní vrcholy, čímž by indikoval přítomnost periodického signálu v původní časové řadě. Jako statistický test je zde použit test SCDF [33].

$$scdf = 1 - e^{-\frac{0{,}1 * MAX\_POWER}{\sigma^2}} \tag{2.39}$$

Kapitola 3
Implementace pluginu

Celý plugin je navržen tak, aby byl konfigurovatelný parametry při spuštění ipfixprobe. Celkem je možné zadat čtyři různé parametry. Plugin byl implementován v jazyce C++ a dokáže zpracovat maximálně 2^{16} paketů v jednom oboustranném síťovém toku. Tento limit byl stanoven během vývoje volbou typů proměnných.

3.1 Implementace do Ipfixprobe

Ipfixprobe obsahuje skript, který vytvoří základ nového pluginu. Dále je potřeba upravit několik souborů, jako například soubor Makefile.am, aby byl plugin zahrnut v kompilaci. Pro správné fungování pluginu je potřeba implementovat několik objektů. Tyto objekty jsou popsány v následující části.

3.1.1 Třída TIMESERIESPlugin

Tato třída je základem pluginu a zajišťuje interakci mezi pluginem a ipfixprobe. Zde je seznam nejdůležitějších metod, které třída může implementovat. Následuje stručný popis, k čemu tyto metody slouží:

- **init(const char *params)**
  Tato metoda se zavolá jednou při spuštění ipfixprobe. Parametrem *params* jsou pluginu předány parametry, které uživatel zadal při spuštění ipfixprobe.

- **pre_create(Packet pkt)**
  Při detekci nového datového toku zavolá ipfixprobe nejprve tuto metodu a parametrem *pkt* předá pluginu všechny informace o daném paketu. To umožňuje pluginu analyzovat první paket.

- **post_create(Flow rec, const Packet pkt)**
  Po přiřazení struktury *Flow* k nově detekovanému síťovému toku zavolá ipfixprobe tuto metodu.

- **pre_update(Flow rec, Packet pkt)**
  Ihned po přijetí dalšího paketu patřícímu k danému síťovému toku je zavolána tato metoda. Tato metoda je zavolána ještě před aktualizací dat v *rec*, takže například v proměnné *rec.time_last* je stále uložen čas přijetí předchozího paketu.

- **post_update(Flow rec, const Packet pkt)**
  Tato metoda je téměř stejná jako předchozí, ale struktura *rec* je již aktualizována.
- **pre_export(Flow rec)**
  Když ipfixprobe vyhodnotí skončení síťového toku, zavolá tuto funkci. To umožní pluginu dopočítat atributy před exportem.

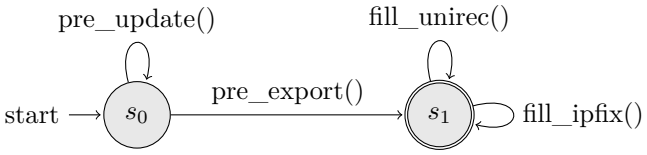
Jak jsou tyto metody implementovány, je popsáno v následující podkapitole.

3.1.2 Struktura RecordExtTIMESERIES

Plugin využívá tuto strukturu pro uchovávání dat o jednotlivých síťových tocích.

Následující stavový automat znázorňuje životní cyklus struktury *RecordExtTIMESERIES* v ipfixprobe. Tato struktura slouží k ukládání dat ze síťového toku. Při detekci nového síťového toku zavolá ipfixprobe metodu pluginu *TIMESERIESPlugin::post_create()*, ta vytvoří novou strukturu *RecordExtTIMESERIES* a přiřadí ji ke korespondujícímu síťovému toku (flowu). Ipfixprobe při každé další detekci paketu patřícího ke již detekovanému síťovému toku zavolá metodu *TIMESERIESPlugin::pre_update()*, ta uloží velikost datové části paketu a čas jeho přijetí do datových struktur v struktuře *RecordExtTIMESERIES* přiřazené k danému síťovému toku. Metoda také kontroluje, zda velikost síťového toku nepřekročila maximální povolenou velikost. Pokud byla velikost překročena, metoda neukládá žádná nová data.

Konec síťového toku detekuje ipfixprobe vypršením timeoutu nebo jiným mechanismem, poté zavolá metodu TIMESERIESPlugin::pre_export(). Tato metoda vypočítá všechny potřebné atributy pro export. Ipfixprobe následně získá vypočítané hodnoty atributů pro export pomocí metod fill_unirec() a fill_ipfix().



3.1.3 Konfigurace pluginu

Při spuštění ipfixprobe jde pomocí parametrů zvolit, jaké typy atributů mají být spočítány. Seznam parametrů je pluginu při spuštění předán v podobě jednoho textového řetězce. Plugin si pomocí třídy *TIMESERIESParser* tento řetězec rozdělí na jednotlivé parametry. Mezi dostupné typy patří statistické atributy, časové atributy, behaviorální atributy a frekvenční atributy.

- statistics (zkratka: s) – spočítá statistické atributy
- time (zkratka: t) – spočítá časové atributy
- behavior (zkratka: b) – spočítá behavioární atributy
- frequency (zkratka: f) – spočítá frekvenční atributy

Pokud není při spuštění zadán žádný parametr, tak plugin v základu počítá pouze statistické atributy.

3.2 Ukládání Dat

Při spuštění pluginu lze zvolit kategorie atributů, které bude plugin počítat, musí tomu být přizpůsobeny i datové struktury. Datové struktury v pluginu byly vybrány tak, aby při každé konfiguraci byl plugin efektivní a neukládal zbytečná data navíc. Jedním ze způsobů jak toho plugin docílil, je použití polymorfní třídy pro ukládání velikostí paketů.

```
class PacketLengths{
public:
    virtual ~PacketLengths(){};
    virtual bool add(uint16_t PacketLength) = 0;
    virtual std::vector<std::pair<uint16_t, uint16_t>> getHistogram() = 0;
    virtual uint32_t getFlowSize() const = 0;
    virtual uint16_t getPacketCount() const = 0;
};
```

Pokud není potřeba pro výpočet kompletní časová řada, používá se pro ukládání dat histogram. Při analýze zachycených záznamů sítě CESNET2 bylo zjištěno, že přibližně 80 % síťových toků na této síti obsahuje pouze pakety o dvou či jedné velikosti datové části. Proto se pro ukládání dat nových síťových toků používá třída *PacketLengthsSmall*, která obsahuje pouze 2 políčka pro ukládání četnosti velikosti paketu a šetří tak paměť. Pokud plugin zachytí paket o jiné velikosti než ty, které byly předtím detekovány, třída *PacketLengthsSmall* se nahradí třídou *PacketLengthsHistogram*. Tato třída obsahuje statické pole o velikosti 1501, ve kterém je velikost paketu rovna indexu a četnost této velikosti je reprezentována obsahem buňky. Velikost 1501 byla zvolena tak, aby pole mohlo obsahovat všechny možné velikosti ethernetových paketů na síti s MTU 1500. Pokud by byl plugin spuštěn na síti s větším MTU než 1500, jsou velikosti všech paketů modulovány touto hodnotou, což umožňuje zpracování jednoho paketu s konstantní paměťovou náročností a složitostí $\mathcal{O}(1)$.

Pokud je potřeba pro další výpočty kompletní časová řada paketů, používá se třída *PacketLengthsArray*. Ta ukládá velikosti paketů do klasického kontejneru typu vector pomocí funkce push_back(). Vector je vytvořen s rezervovanou kapacitou 100, ale optimální by bylo, kdyby kapacita byla nastavena na maximální počet přijatých paketů. Tato varianta je však neúměrně náročná na paměť a zrychlení je zanedbatelné. Nabízí se také použití jiné datové struktury, jako například unrolled linked list. Tato datová struktura by teoreticky mohla být rychlejší, ale v praxi to není, kvůli amortizaci rozšiřování vectoru ve funkci push_back(). Unrolled linked list má větší paměťovou náročnost v podobě pointerů na další node a dalších režijních dat pro každou alokovanou node na haldě. Kvůli tomu, že data jsou rozdělena do několika polí, má také horší využití prostorové lokality v paměti.

Informace o časech, kdy byly pakety přijaty se ukládá do třídy *PacketTimes*. Ta používá pro ukládání také vector. Použití histogramu nedává v tomto případě žádný smysl, protože většina uložených časů je unikátních. Časy přijetí jednotlivých paketů se ukládají v mikrosekundách.

3.3 NFFT3

Pro výpočet Lomb-Scargle periodogramu je v pluginu použita knihovna NFFT3 [34]. Tato knihovna zajišťuje rychlý výpočet Lomb-Scargle periodogramu pomocí NFFT (nonequispaced fast Fourier transform). Díky této knihovně lze vypočítat Lomb-Scargle periodogram s časovou složitostí $\mathcal{O}(n \log n)$.

3.3.1 Příprava dat

Samotný Lomb-Scargle periodogram nelze získat jedním voláním funkce knihovny NFFT3, protože je nutné nejdříve připravit data. Celkově lze výpočet Lomb-Scargle periodogramu rozdělit na následující kroky:

1. **Určit parametry pro výpočet NFFT**

   Nejdřív je potřeba určit hodnoty parametrů oversampling_factor (C_{over}) a highest_freq_factor (C_{high}), ty lze stanovit v závislosti na datech, ale v pluginu jsou určeny staticky. Zde je popsáno co znamenají a jak jejich volba ovlivní výsledná data.

   Výstupem periodogramu je rovnoměrná frekvenční mřížka.

   $$f_k = k\Delta f (k = 1, ..., N_f) \tag{3.1}$$

   kde odstup mezi frekvencemi je dán vztahem

   $$\Delta f = \frac{1}{C_{over}(t_{N_t} - t_1)} \tag{3.2}$$

   kde N_t je počet všech prvků v časové řadě.

   Celkový počet výsledných frekvencí v periodogramu je dán vztahem

   $$N_f = \frac{1}{2} C_{over} C_{high} N_t \tag{3.3}$$

   (Pro N_f je použita proměnná m)

   Parametr Chigh je poměr nejvyššího požadované frekvence (f_{high}) k Nyquistově frekvenci.

   Nyquistova frekvence je frekvence, jejíž perioda je rovna dvěma vzorkovacím intervalům. Časová řada je ale nerovnoměrně vzorkovaná, proto se pro výpočet této frekvence použije frekvence, která by byla získána pokud by N_t datových bodů bylo získáno rovnoměrně v čase $T = t_{N_t} - t_1$. Nyquistova frekvence je zde určená následujícím vztahem:

   $$f_c = \frac{N_t}{2T} \tag{3.4}$$

2. **Výpočet průměru a rozptylu dat**

   Nejdřív je potřeba získat průměr a rozptyl z časové řady.

   Tyto údaje plugin vypočítal už v statistických atributech, takže je nemusí počítat podruhé.

3. **Vycentrovat data kolem jejich průměru**

   Dále je nutné vycentrovat data okolo průměru. Toho jde docílit jednoduchým for cyklem:

   ```
   for (int i = 0; i < npts; i++){
       value[i] = value[i] - mean_value;
   }
   ```

4. **Zmenšit časové rozpětí na interval** $[-1/2, 1/2)$

   Knihovna NFFT předpokládá, že data časů na vstupu jsou v intervalu $[-1/2, 1/2)$, proto je třeba před výpočtem NFFT zobrazit časy datových bodů z časové řady na interval $[-1/2, 1/2)$. Toho lze docílit použitím následujícího zobrazení.

   $$t_i \to x_i = 2a(t_i - t_1)\Delta f - a \tag{3.5}$$

   kde $a = 1/2 - \epsilon$ a ϵ je nějaké malé číslo například 10^{-5} [35].

Zobrazení realizuje následující kód:

```
        double t1 = (double)time[0];
        double tNt = (double)time[npts - 1];
        double delta_f = 1.0 / (oversampling_factor * (tNt - t1));
        double a = 0.5 - 0.00001;
        std::vector<double> x(npts);
        for (int i = 0; i < npts; i++){
            x[i] = (2.0 * a * ((double)time[i] - t1) * delta_f) - a;
        }
```

5. **Spočítat Furierovu transformaci pro data**

Furierovu transformaci lze spočítat pomocí následující funkce:

```
/* Computation of the positive frequency
 part of the (unnormalised ) Fourier
 transform of a times -series (t, y).

 Input:
 t the times reduced to [1/2, 1/2)
* y the measurements (NULL, for
* computing the FT of the window)
* n the number of measurements
* m the number of positive frequencies
* Output:
* d the Fourier coefficients
* (preallocated array for (m+1)
* elements)
*/

void nfft(const double *t, const double *y,
          int n, int m, double _Complex *d){
    // Creates NFFT plan for 2*m Fourier
    // coefficients (positive and negative
    // frequencies ) and n data samples.
    nfft_plan p;
    nfft_init_1d(&p, 2 * m, n);

    if (y != NULL){ // data spectrum
        for (int i = 0; i < n; i++){
            p.x[i] = t[i];
            p.f[i][0] = y[i];
            p.f[i][1] = 0.0;
        }
    }
    else{ // window spectrum
        for (int i = 0; i < n; i++){
            p.x[i] = t[i];
            p.f[i][0] = 1.0;
            p.f[i][1] = 0.0;
        }
    }
    // Possibly optimises.
    if (p.flags & PRE_ONE_PSI)
        nfft_precompute_one_psi(&p);
```

```
    // Computes the adjoint transform.
    nfft_adjoint(&p);

    // Outputs the positive frequency
    // Fourier coefficients .
    for (int i = 0; i < m; i++){
        d[i] = p.f_hat[i][0] + p.f_hat[i][1] * I;
    }
    d[m] = p.f_hat[0][0] - p.f_hat[0][1] * I;
    nfft_finalize(&p);
}
```

Tato funkce byla převzata z [35] a upravena tak, aby fungovala v C++.

6. **Spočítat Furierovu transformaci pro dvojnásobně veliké okno**

   Dále je potřeba použít znovu funkci z předchozího kroku, ale tentokrát jí předat pouze časové údaje a zvolit dvojnásobnou hodnotu m.

7. **Dopočítat Lomb-Scarlge periodogram**

   Následující kód vypočítá Lomb-Scarlge periodogram. Pole sp[m+1] je výstup funkce nfft z kroku 5. a pole win[(2*m)+1] je výstup funkce nfft z kroku 6.

```
LS *ls = new LS(m);
// Computes the periodogram ordinates,
// and store the results in the LS structure.
for (int j = 1; j <= m; j++){
    double _Complex z1 = sp[j];       // FT of data at \omega
    double _Complex z2 = win[2 * j]; // FT of window at 2\omega
    double absz2 = cabs(z2);
    double hc2wt = 0.5 * cimag(z2) / absz2;
    double hs2wt = 0.5 * creal(z2) / absz2;
    double cwt = sqrt(0.5 + hc2wt);
    double swt = sign(sqrt(0.5 - hc2wt), hs2wt);
    double den = 0.5 * npts + hc2wt * creal(z2) + hs2wt * cimag(z2);
    double cterm = square(cwt * creal(z1) + swt * cimag(z1)) / den;
    double sterm = square(cwt * cimag(z1) - swt * creal(z1)) / (npts-den);
    ls->freqs[m - j] = (j - 1) * df * df;
    ls->Pn[j - 1] = (cterm + sterm) / var / npts;
    }
```

Tato funkce byla převzata z [35] a upravena tak, aby fungovala v C++.

3.3.2 CUNFFT

Výpočet Lomb-Scarlge periodogramu stále zůstává výpočetně nejnáročnější operací v celém pluginu. Jednou z možností, jak tento výpočet zrychlit, je použít jinou knihovnu než NFFT3. Zdánlivě lepší možností je knihovna CUNFFT [36], ta je založena právě na knihovně NFFT3, ale na rozdíl od knihovny napsané v jazyce C, je napsána v jazyce CUDA. Díky tomu knihovna využívá zdroje GPU, a tím urychluje výpočet NFFT.

Hlavním omezením této knihovny je, že ji lze využít pouze na kartách společnosti Nvidia. Tato knihovna nakonec nebyla využita, protože exportéry, na kterých má být vytvořený plugin nasazen, neobsahují grafickou kartu.

Kapitola 4
Testování

4.1 Správnost implementace

Správnost implementace výpočtu atributů byla ověřena porovnáním výsledků pluginu s výsledky implementace v jazyce Python. Implementace v Pythonu má výhodu, že využívá více knihoven, a tudíž existuje menší prostor pro chyby. Plugin byl testován jak na náhodných datech, tak na datech z reálného síťového provozu.

Tento srovnávací proces umožnil ověřit, že plugin funguje správně a poskytuje přesné výsledky.

4.1.1 Lomb-Scargle periodogram

Nejdůležitější bylo otestovat správnost implementace NFFT3, která počítá Lomb-Scargle periodogram. Data z pluginu byla porovnána s daty z Python knihovny Astropy.

Jak je vidět na obrázku 4.1 data se velmi mírně liší. To je zřejmě způsobeno jiným vzorkováním v obou implementacích, dále se liší počet frekvencí, v Pythonu je graf kratší. To způsobil jinak nastavený oversampling_factor v obou implementacích. Tyto drobné odchylky jsou naprosto zanedbatelné a je vidět, že Lomb-Scarlge periodogram je v pluginu implementován správně a dokáže najít všechny důležité vrcholy.

4.2 Měření výkonu pluginu

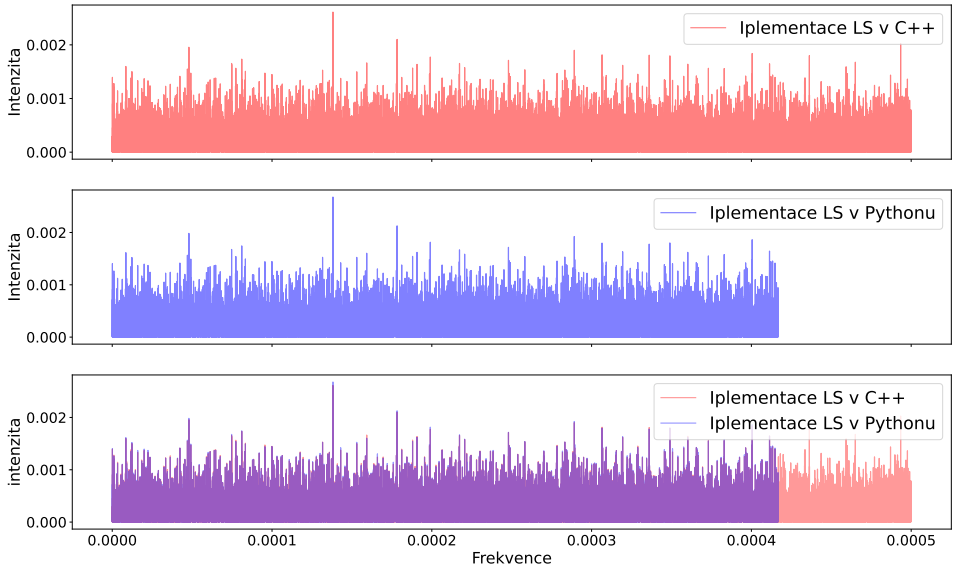
V této sekci je popsáno testování výsledného pluginu. Během testů pluginu přijímal ipfixprobe pakety z pcap souborů a výsledné záznamy síťových toků posílal modulem ipfix na kolektor. Měření probíhalo na stroji s operačním systémem Ubuntu 20, procesorem AMD Ryzen 5 3400G 3,7 GHz a 16 GB RAM a používaly se dvě datové sady, které byly získány zachycením komunikace na síti CESNET2.

- Datová sada 1. - pcap soubor o velikosti 7,1 GB, obsahuje 8 244 588 paketů a 386253 bitflow.
- Datová sada 2. - pcap soubor o velikosti 48 GB, obsahuje 46 790 172 paketů a 181879 bitflow.

Plugin se testoval s následujícími limity, které jsou popsány v tabulce 4.1. Limit 0 byl zvolen čistě pro testování výkonu, ale nelze ho použít do exportéru, protože hrozí přetečení proměnných.

V následující tabulce 4.2 jsou uvedeny výsledky jednotlivých testů. Testoval se plugin, který počítal všechny atributy a každý výsledek je průměr z několika testů.

Jak je vidět, tak časy jsou pro každý limit téměř identické. To je zaviněno pcap modulem, který nestíhá číst tak rychle data z disku. Podařilo se dosáhnout rychlosti $\sim$ 1.2 GB/s.



Obrázek 4.1 Porovnání Lomb-Scarlge periodogramu: 1. implementovaný knihovnou NFFT3 v C++ (horní graf), 2. implementovaný knihovnou Astropy v Pythonu (prostřední graf), 3. porovnání obou implementací (spodní graf)

Tabulka 4.1 Seznam různých testovacích konfigurací

Označení	Limit paketů v poli	Limit paketů v histogramu
0	Neomezeno	Neomezeno
1	$2^{16} - 10$	$2^{16} - 10$
2	50000	50000
3	1000	1000
4	500	500
5	200	200

Díky těmto testům lze usoudit, že ipfixprobe s pluginem dokáže zpracovat minimálně milion paketů/s. Maximální rychlost v tomto testu byla omezena rychlostí disku. Po nasazení do reálné sítě by plugin dosahoval lepších výsledků.

Na tomto počítači byl také plugin otestován v malé síti a tam se choval dle očekávání. Plugin zpracoval všechen přicházející síťový provoz bez problému.

Dále byl plugin otestován pomocí programu *Colasoft Packet Player* na druhém počítači. Tento program přeposílal obsah první datové sady na exportér. I při maximální rychlosti, kterou dokázal *Colasoft Packet Player* přeposílat pakety, zvládnul ipfixprobe s pluginem zpracovat všechny příchozí pakety.

4.2.1 Testování na virtuálním počítači

Dále byl plugin testován na virtuálním počítači s typem virtualizace KVM. Na virtuálním počítači byl operačním systém Debian 11, měl 3 jádra z procesoru AMD EPYC 7543 se základní frekvencí 2.8 GHz. Virtuální stroj měl přiděleno 32 GB RAM.

Tabulka 4.2 Výsledky prvního testování

Limity:	Datová sada 1	Datová sada 2
Limit 0	4,7 s	40s
Limit 1	4,8 s	41,1s
Limit 2	4,7 s	40.4s
Limit 3	5 s	40,4s
Limit 4	4,5 s	40,8s
Limit 5	4,6 s	41.5s

Výhodou této možnosti bylo že stroj měl diskové úložiště složené z NVMe disků v hardwarovém RAIDu. Díky tomu dokázal číst z disku cca dvojnásobnou rychlostí než první testovací stroj. Nevýhodou testování na virtuálním stroji je, že nelze přesně určit kolik výkonu procesoru zabere virtualizace a jiné virtuální stroje.

Tabulka 4.3 Výsledky druhého testování

Limity:	Datová sada 1	Datová sada 2
Bez pluginu	3 s	25s
Limit 0	3,5 s	27s
Limit 1	3,2 s	26s
Limit 5	3,1 s	26s

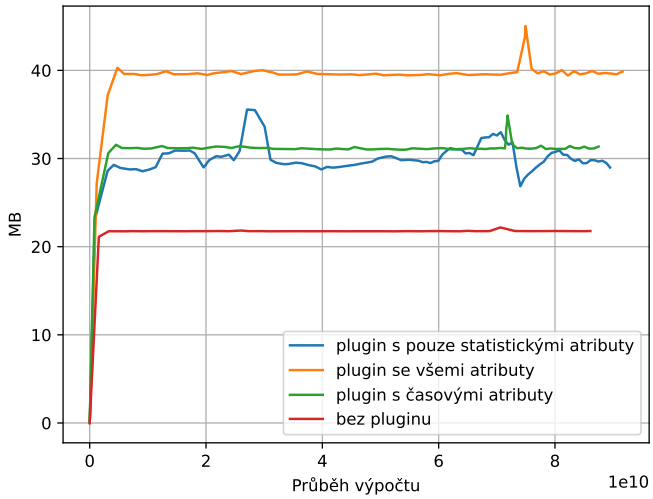
Jak je vidět v tabulce 4.3, tak ani na tomto stroji se nepodařilo dosáhnou limitu procesoru, opět nestíhal modul pcap. Díky rychlejšímu disku se zde podařilo dosáhnout rychlosti ~ 1.75 GB/s. Z těchto výsledků lze usoudit, že exportér s tímto pluginem by šel nasadit na spoji s rychlostí $\sim$ 14 Gb/s. S vhodně zvolenou filtrací příchozích paketů by plugin mohl pracovat na vysokorychlostní síti.

4.3 Testování náročnosti na paměť

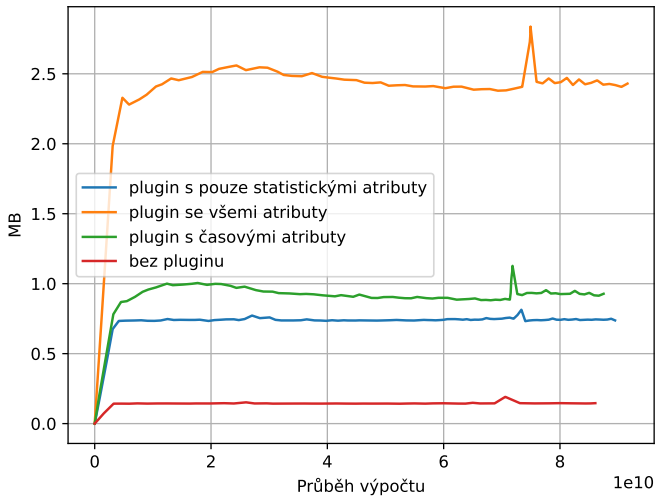
Kromě měření výkonu nového pluginu bylo také nutné otestovat, zda byly zvoleny správné datové struktury a plugin není neúměrně náročný na paměť. K tomu byl použit profilovaní nástroj *massif*. Massif je součástí programu *valgrind* a slouží k měření kolik paměti na haldě program využívá. Massif dokáže změřit velikost užitečné paměti na haldě, ale také kolik další paměti je na haldě alokováno pro účely alokací a zarovnání.

V následujícím grafu 4.2 je vidět, kolik MB na haldě zabírá ipfixprobe s pluginem v různých konfiguracích při analýze 1. pcapu.

Všechna tato paměť ale není programem využita pro data. Část paměti je využita pro správu alokací v paměti, a také může být využita pro zajištění vhodného zarovnání prvků do bloků. Kolik paměti bylo zabráno režijními daty je vidět na grafu 4.3.



Obrázek 4.2 Graf zobrazující průběh využití paměti



Obrázek 4.3 Graf zobrazující velikost režijní dat v paměti

Závěr

Cílem práce bylo vytvořit prototyp pluginu pro open source exportér síťových toků ipfixprobe. Tento plugin získává časové řady ze síťových toků. Tyto časové řady následně analyzuje a získává z nich atributy. Plugin následně rozšiřuje záznamy síťových toků, které ipfixprobe exportuje.

Tento cíl byl úspěšně splněn a výsledný plugin *timeseries* je dostatečně výkonný na to, aby se dal nasadit do reálné vysokorychlostní sítě. S vhodně zvolenou filtrací příchozích paketů by mohl představovat jen malou zátěž pro exportér. Výsledný kód pluginu byl otestován a je připravený pro integraci do ipfixprobe. Během testování se podařilo otestovat výkon ipfixprobe s vytvořeným pluginem v různých konfiguracích. Plugin byl testován na datových sadách které načítal modulem ipfixprobe pcap. Tento modul lze ho také použít pro čtení dat z interfacu, takže jde odvodit, že by se plugin stejně choval na síti.

Atributy, které plugin počítá, jsou rozděleny do čtyř kategorií a tyto kategorie lze libovolně kombinovat. Díky návrhu je zajištěno, že plugin je při každé možné kombinaci efektivní. Všechny atributy se podařilo implementovat a otestovat, zda byly implementovány správně.

Ipfixprobe s výsledným pluginem je připraven pro nasazení do ISP sítě CESNET2 (Czech national and research network). V rámci budoucího rozvoje práce může být použit za účelem získávání vstupních dat pro detekci hrozeb na základě strojového učení.

Pro výsledný kód byl vytvořen pull request do github repositáře ipfixprobe[1].

[1] `https://github.com/CESNET/ipfixprobe/pull/150`

Bibliografie

1. AITKEN, Paul; CLAISE, Benoît; TRAMMELL, Brian. *Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information* [RFC 7011]. RFC Editor, 2013. Request for Comments, č. 7011. Dostupné z DOI: 10.17487/RFC7011.
2. HOFSTEDE, Rick; ČELEDA, Pavel; TRAMMELL, Brian; DRAGO, Idilio; SADRE, Ramin; SPEROTTO, Anna; PRAS, Aiko. Flow Monitoring Explained: From Packet Capture to Data Analysis With NetFlow and IPFIX. *IEEE Communications Surveys Tutorials*. 2014, roč. 16, č. 4, s. 2037–2064. Dostupné z DOI: 10.1109/COMST.2014.2321898.
3. *HTTPS encryption on the web* [online]. Google. Dostupné také z: https://transparencyreport.google.com/https/overview?hl=en. [Navštíveno 2. 5. 2023].
4. JEONG, Seyeon; YOU, Jae-Hyoung; HONG, James Won-Ki. Design and Implementation of Virtual TAP for SDN-based OpenStack Networking. In: *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. 2019, s. 233–241. ISBN 978-3-903176-15-7.
5. *nProbe documentation* [online]. ntop.org. Dostupné také z: https://www.ntop.org/guides/nprobe/. [Navštíveno 20. 4. 2023].
6. *fprobe manual* [online]. Dostupné také z: https://manpages.ubuntu.com/manpages/xenial/man8/fprobe.8.html. [Navštíveno 22. 4. 2023].
7. *Flowmon Probe* [online]. Progress Software Corporation. Dostupné také z: https://www.flowmon.com/en/products/appliances/probe. [Navštíveno 22. 4. 2023].
8. *ipfixprobe* [online]. CESNET. Dostupné také z: https://github.com/CESNET/ipfixprobe. [Navštíveno 22. 4. 2023].
9. CLAISE, Benoît. *Cisco Systems NetFlow Services Export Version 9* [RFC 3954]. RFC Editor, 2004. Request for Comments, č. 3954. Dostupné z DOI: 10.17487/RFC3954.
10. *UniRec documentation* [online]. CESNET. Dostupné také z: https://nemea.liberouter.org/doc/unirec. [Navštíveno 24. 4. 2023].
11. *IPFIXcol2* [online]. CESNET. Dostupné také z: https://github.com/CESNET/ipfixcol2. [Navštíveno 22. 4. 2023].
12. *nfdump* [online]. Dostupné také z: https://github.com/phaag/nfdump. [Navštíveno 22. 4. 2023].
13. HAMILTON, James Douglas. *Time series analysis*. Princeton university press, 2020. ISBN 9780691218632.
14. FOSTER, Grant. Wavelets for period analysis of unevenly sampled time series. *Astronomical Journal v. 112, p. 1709-1729.* 1996, roč. 112, s. 1709–1729.

15. GOH, K.-I.; BARABÁSI, A.-L. Burstiness and memory in complex systems. *Europhysics Letters*. 2008, roč. 81, č. 4, s. 48002. Dostupné z DOI: 10.1209/0295-5075/81/48002.

16. HURST, H. E. Long-Term Storage Capacity of Reservoirs. *Transactions of the American Society of Civil Engineers*. 1951, roč. 116, č. 1, s. 770–799. Dostupné z DOI: 10.1061/TACEAT.0006518.

17. KENNEY, John F. Mathematics of statistics. *American Mathematical Monthly*. 1940, s. 59–60. Dostupné také z: https://hdl.handle.net/2027/mdp.39015015725339. [Navštíveno 10. 4. 2023].

18. WEISSTEIN, Eric W. *Pearson's Skewness Coefficients: From MathWorld–A Wolfram Web Resource*. Dostupné také z: https://mathworld.wolfram.com/PearsonsSkewnessCoefficients.html [Navštíveno 13. 4. 2023].

19. WESTFALL, Peter. Kurtosis as Peakedness, 1905-2014. RIP. *The American Statistician*. 2014, roč. 68. Dostupné z DOI: 10.1080/00031305.2014.917055.

20. MILLER, Steven. *Benford's Law: Theory and Applications*. 2015. ISBN 9781400866595.

21. HURST, H. E. Long-Term Storage Capacity of Reservoirs. *Transactions of the American Society of Civil Engineers*. 1951, roč. 116, č. 1, s. 770–799. Dostupné z DOI: 10.1061/TACEAT.0006518.

22. QIAN, Bo; RASHEED, Khaled M. HURST EXPONENT AND FINANCIAL MARKET PREDICTABILITY. In: 2005.

23. JI, Li-Jun; ZHOU, Wei-Xing; LIU, Hai-Feng; GONG, Xin; WANG, Fu-Chen; YU, Zun-Hong. R/S method for unevenly sampled time series: Application to detecting long-term temporal dependence of droplets transiting through a fixed spatial point in gas–liquid two-phase turbulent jets. *Physica A: Statistical Mechanics and its Applications*. 2009, roč. 388, č. 17, s. 3345–3354. ISSN 0378-4371. Dostupné z DOI: https://doi.org/10.1016/j.physa.2009.05.006.

24. STELLINGWERF, R. F. Period determination using phase dispersion minimization. 1978, roč. 224, s. 953–960. Dostupné z DOI: 10.1086/156444.

25. GRATTAN-GUINNESS, I. Chapter 26 - Joseph Fourier, Théorie analytique de la chaleur (1822). In: GRATTAN-GUINNESS, I.; COOKE, Roger; CORRY, Leo; CRÉPEL, Pierre; GUICCIARDINI, Niccolo (ed.). *Landmark Writings in Western Mathematics 1640-1940*. Amsterdam: Elsevier Science, 2005, s. 354–365. ISBN 978-0-444-50871-3. Dostupné z DOI: https://doi.org/10.1016/B978-044450871-3/50107-8.

26. LOMB, N. R. Least-Squares Frequency Analysis of Unequally Spaced Data. 1976, roč. 39, č. 2, s. 447–462. Dostupné z DOI: 10.1007/BF00648343.

27. SCARGLE, J. D. Studies in astronomical time series analysis. II. Statistical aspects of spectral analysis of unevenly spaced data. 1982, roč. 263, s. 835–853. Dostupné z DOI: 10.1086/160554.

28. VANDERPLAS, Jacob T. Understanding the Lomb–Scargle Periodogram. *The Astrophysical Journal Supplement Series*. 2018, roč. 236, č. 1, s. 16. Dostupné z DOI: 10.3847/1538-4365/aab766.

29. SHANNON, C. E. A mathematical theory of communication. *The Bell System Technical Journal*. 1948, roč. 27, č. 3, s. 379–423. Dostupné z DOI: 10.1002/j.1538-7305.1948.tb01338.x.

30. PEETERS, Geoffroy. A large set of audio features for sound description (similarity and classification) in the CUIDADO project. *CUIDADO Ist Project Report*. 2004, roč. 54, č. 0, s. 1–25.

31. SCHEIRER, Eric; SLANEY, Malcolm. Construction and evaluation of a robust multifeature speech/music discriminator. In: *1997 IEEE international conference on acoustics, speech, and signal processing*. 1997, sv. 2, s. 1331–1334. Dostupné z DOI: 10.1109/ICASSP.1997.596192.

32. LIANG, Chen; CASTAGNA, John; ZAVALA TORRES, Ricardo. Tutorial: Spectral bandwidth extension — Invention versus harmonic extrapolation. *Geophysics*. 2017, roč. 82, č. 4, s. W1–W16. ISSN 0016-8033. Dostupné z DOI: 10.1190/geo2015-0572.1.

33. KOUMAR, Josef; ČEJKA, Tomáš. Network traffic classification based on periodic behavior detection. In: *2022 18th International Conference on Network and Service Management (CNSM)*. 2022, s. 359–363. Dostupné z DOI: 10.23919/CNSM55787.2022.9964556.

34. KEINER, Jens; KUNIS, Stefan; POTTS, Daniel. Using NFFT3 - a Software Library for Various Nonequispaced Fast Fourier Transforms. *ACM Trans. Math. Software*. 2009, roč. 36, s. Article 19, 1–30. Dostupné z DOI: 10.1145/1555386.1555388.

35. LEROY, B. Fast calculation of the Lomb-Scargle periodogram using nonequispaced fast Fourier transforms. *Astronomy and Astrophysics - A&A*. 2012, roč. 545, s. A50. Dostupné z DOI: 10.1051/0004-6361/201219076.

36. KUNIS, Susanne; KUNIS, Stefan. *CUNFFT - Nonequispaced FFT and it's inversion in CUDA* [online]. Dostupné také z: https://github.com/sukunis/CUNFFT. [Navštíveno 22. 4. 2023].

Příloha A

Instalace ipfixprobe s pluginem

Tento krátký návod popisuje, jak v terminálu nainstalovat a spustit vytvořený plugin. Návod byl vytvořen na operačním systému Ubuntu a pro jiné systémy se může postup lišit. Pro kompilaci je potřeba mít nainstalované tyto balíky *$ sudo apt install git make automake libtool g++ gcc libfftw3-dev libpcap-dev libatomic-ops-dev libssl-dev*

Nejprve je potřeba nainstalovat knihovnu NFFT3.
$ git clone https://github.com/NFFT/nfft
$ cd nfft
$ ./bootstrap.sh
$ ./configure –enable-all –enable-openmp
$ make
$ make check
$ sudo make install

Dále je potřeba nainstalovat samotný exportér[1].
$ git clone –recurse-submodules https://github.com/CESNET/ipfixprobe
$ cd ipfixprobe
$ autoreconf -i
$ ./configure –with-pcap –with-timeseries
$ make
$ sudo make install

Po spuštění *ipfixprobe* může nastat tato chyba:
./ipfixprobe: error while loading shared libraries: libnfft3.so.4: cannot open shared object file: No such file or directory
Tato chyba jde jednoduše opravit 2 příkazy.
$ LD_LIBRARY_PATH=/usr/local/lib
$ export LD_LIBRARY_PATH

Nyní jde ipfixprobe spustit příkazem:
$ ipfixprobe

[1] Ipfixprobe s pluginem je také dostupný na `https://github.com/Dakevid/ipfixprobe`

Příloha B

Spouštění ipfixprobe s pluginem

Zde je pár příkladů jak spustit ipfixprobe s novým pluginem timeseries. Ve všech případech je zvolen jako vstupní moudul *pcap*, ale lze použít jakýkoliv jiný. Pro výstup v ukázkách je ve výchozím nastavení zvolen jako modul ipfix který odesílá data na adresu 127.0.0.1:4739.

Spuštění pluginu s základním nastavením (pouze statistické atributy):
$ ipfixprobe 'pcap;file=pcaps/http.pcap' -p timeseries

Se všemi atributy:
$ ipfixprobe 'pcap;file=pcaps/http.pcap' -p 'timeseries;s;t;b;f'

Pouze se statistickými atributy:
$ ipfixprobe 'pcap;file=pcaps/http.pcap' -p 'timeseries;s'

Pouze s časovými a statistickými atributy:
$ ipfixprobe 'pcap;file=pcaps/http.pcap' -p 'timeseries;s;t'

Obsah přiloženého média

- `src`
  - `readme.txt` ..... stručný popis obsahu
  - `impl` ..... zdrojové kódy implementace pluginu
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
  - `elements` ..... definice IFIX políček ve formátu xml pro knihovnu libfds
- `thesis.pdf` ..... text práce ve formátu PDF