VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

DIPLOMOVÁ PRÁCE

Brno, 2021 Bc. Ondřej Kupka

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

SOFTWARE HLÁŠENÍ BEZPEČNOSTNÍCH INCIDENTŮ V GPON SÍTI
GPON NETWORK SECURITY INCIDENT REPORTING SOFTWARE

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Ondřej Kupka**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Tomáš Horváth, Ph.D.**
SUPERVISOR

BRNO 2021

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní program **Informační bezpečnost**

Ústav telekomunikací

Student: Bc. Ondřej Kupka ***ID:*** 195158
Ročník: 2 ***Akademický rok:*** 2020/21

NÁZEV TÉMATU:

Software hlášení bezpečnostních incidentů v GPON síti

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je nastudovat a popsat současný stav způsobu elektronické komunikace s CSIRT (Computer Security Incident Response Team) teamy v závislosti na hlášení incidentů v počítačové síti a open source projekty, jako je například TheHive. V praktické části diplomové práce navrhněte a naprogramujte rozšířený modul API rozhraní pro automatické zasílání zpráv při výskytu události v GPON síti s využitím MISP (Malware Information Sharing Platform). API rozhraní bude realizováno v jazyce Python s využitím programového modulu pro zasílání informačních zpráv. Výstupem diplomové práce je funkční aplikace dle navržené specifikace.

DOPORUČENÁ LITERATURA:

[1] Pilgrim, M. Ponořme se do Python(u) 3. CZ.NIC, z.s.p.o., 2010, ISBN: 978-80-904248-2-1

[2] THEHIVE PROJECT [online]. 2017 - 2020 [cit. 2020-09-09]. Dostupné z: https://thehive-project.org/

Termín zadání: 1.2.2021 ***Termín odevzdání:*** 24.5.2021

Vedoucí práce: Ing. Tomáš Horváth, Ph.D.

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

ABSTRAKT

Tato práce je zaměřena na návrh možnosti hlášení bezpečnostních událostí z GPON sítí. Teoretická část představuje principy GPON a poskytuje úvod do problematiky bezpečnostních incidentů. Praktická část je zaměřena na výběr vhodných open-source systémů a návrh aplikace umožňující tvorbu hlášení. Výstupem práce je nasazení systémů TheHive, Cortex a MISP a vytvořená aplikace v jazyce Python3 umožňující tvorbu různých typů hlášení na základě připravené šablony. Práce je zakončena podrobným popisem nasazení, vlastní konfigurace a otestováním.

KLÍČOVÁ SLOVA

GPON, bezpečnostní incident, CERT, CSIRT, projekt TheHive, MISP, Python3

ABSTRACT

This thesis focuses on development of software for security incident reporting from GPON networks. The theoretical part introduces the principles of GPON and provides an introduction to security incidents. The practical part is focused on the selection of suitable open-source systems and the design of an application in Python for the creation of alerts. The output of the work is the deployment of TheHive, Cortex and MISP systems and the creation of an application enabling the creation of various types of alerts based on prepared template. The thesis is finalized by a detailed description of deployment, custom configuration and testing.

KEYWORDS

GPON, security incident, CERT, CSIRT, project TheHive, MISP, Python3

KUPKA, Ondřej. *Vývoj API rozhraní pro hlášení bezpečnostních incidentů*. Brno, 2020, 88 s. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedoucí práce: Ing. Tomáš Horváth, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „Vývoj API rozhraní pro hlášení bezpečnostních incidentů“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

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

Brno ....

podpis autora

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu diplomové práce panu Ing. Tomáši Horváthovi, Ph.D. a panu Ing. Václavu Oujezskému, Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci. Dále bych rád poděkoval panu Ing. Martinu Holíkovi za odborný úvod do problematiky Dockeru a Kafky, konzultace, trpělivost a užitečné tipy. V neposlední řadě pak mé díky patří také mým blízkým, kteří mě při tvorbě práce podpořili a byli ochotni mě nasměrovat správným směrem.

Obsah

Seznam obrázků

Seznam tabulek

Seznam výpisů

Úvod

S rostoucí popularitou pasivních optických sítí, jako je například GPON (Gigabit Passive Optical Network), roste i riziko, že se právě tyto sítě nebo aktivní prvky dané infrastruktury stanou cílem útočníků. Problém možných bezpečnostních incidentů je aktuálně řešen převážně pouze na úrovni Ethernet sítí. Poskytovatelé informačních služeb by však měli mít možnost se těmito bezpečnostními incidenty zabývat již na poli přístupových sítí, které mohou být také ohroženy kybernetickými útoky. V této oblasti jsou ale prozatím možnosti velmi omezené.

Standardy definující GPON sítě slouží pouze jako doporučení a předpokládají určitou úroveň bezpečnosti vycházející z povahy technologie a přenosového média. Tyto sítě ale nejsou výjimkou. Aktuálně již bylo nalezeno a zdokumentováno několik bezpečnostních rizik, které dávají útočníkům prostor k realizaci bezpečnostních incidentů.

Důvod spravovat a analyzovat bezpečnostní události v GPON sítích je tedy opodstatněný. Dostupné nástroje, které by analýzu umožňovaly v reálném čase ale prakticky neexistují.

Hlavním cílem této práce je návrh a realizace automatického hlášení bezpečnostních událostí z GPON sítí, na základě zhodnocení aktuálně dostupných možností hlášení těchto událostí obecně.

V první kapitole je popsána technologie GPON. Tato kapitola popisuje jednotlivé komponenty sítě, princip komunikace a přenášené zprávy. Je zde popsán také aktivační proces koncových jednotek, který jim umožňuje komunikaci v síti a mechanismy pro správu a kontrolu koncových jednotek.

Druhá kapitola je zaměřena na zhodnocení bezpečnosti GPON sítí a aktuální bezpečnostní rizika technologie, jako je zejména odposlech nebo zranitelnost koncových jednotek.

Třetí kapitola se věnuje problematice bezpečnostních incidentů obecně. Pozornost je věnována české právní úpravě, bezpečnostním týmům a aktuálním možnostem hlášení bezpečnostních incidentů.

Čtvrtá kapitola ve zkratce představuje systém využívající FPGA (Field Programmable Gate Array) analyzátor, umožňující zachycení a analýzu síťové komunikace GPON sítí. Na základě této analýzy je možné detekovat bezpečnostní události, které je však nutné dále zpracovat a vyhodnotit.

V páté kapitole jsou představeny jednotlivé dostupné open-source nástroje pro správu a zpracování bezpečnostních událostí v klasických Ethernetových sítích.

V kapitole návrhu řešení jsou porovnány specifikace a výhody jednotlivých nástrojů. Na základě porovnání jsou poté vybrány vhodné nástroje, které je možné využít jejich modifikací i pro GPON sítě. Pro účely testování návrhu je vytvořena

testovací síť skládající se ze systémů TheHive, Cortex a MISP. V rámci návrhu je popsáno i samotné nasazení systémů a jejich nutná dodatečná konfigurace.

Pro hlášení bezpečnostních událostí z GPON sítí pomocí vybraných nástrojů byla navržena vlastní aplikace, která je popsána v sedmé kapitole. Jsou zde uvedeny konkrétní metody, funkcionality a její implementace.

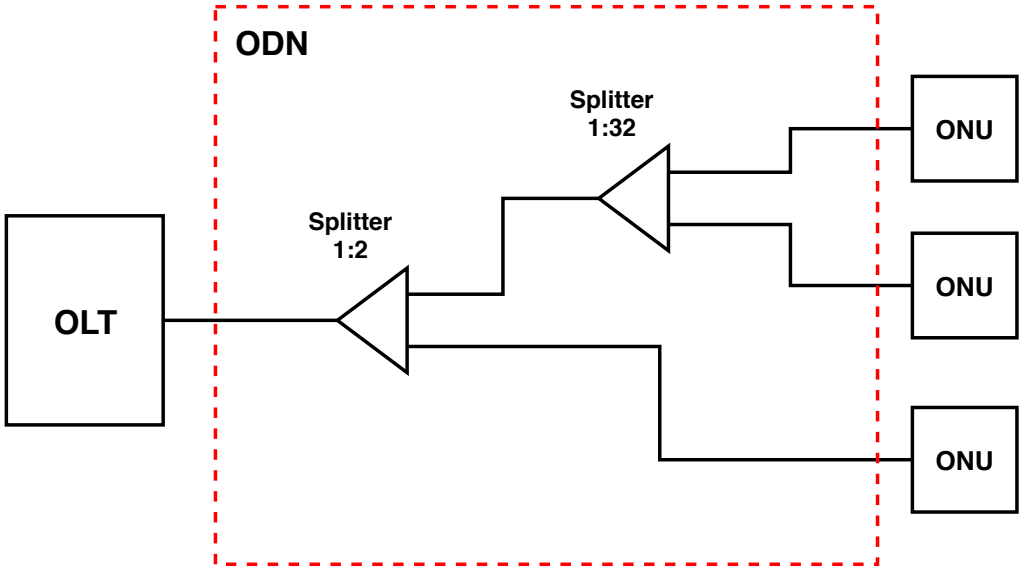
V osmé kapitole je zdokumentován celý proces nasazení a konfigurace testovací sítě a jednotlivých systémů. Pozornost je věnována hlavně nasazení a testování vlastní aplikace pro tvorbu hlášení. Testování proběhlo pomocí simulace bezpečnostních událostí, na základě kterých byla v systému TheHive vytvářena hlášení. Poslední část je pak také věnována následné správě vytvořených hlášení bezpečnostních událostí v rámci jednotlivých systémů.

1 Gigabitová pasivní optická síť - GPON

Technologie gigabitových pasivních optických sítí GPON je souborem standardů definovaným organizací ITU-T (Internacional Telecommunication Union – Telecommunication Standardization Sector). Doporučení ITU-T G.984.1 popisuje obecnou charakteristiku [1], ITU-T G.984.2 se věnuje specifikaci fyzické vrstvy [2], ITU-T G.984.3 specifikuje konvergenční přenosové vrstvy [3] a ITU-T G.984.4 specifikuje řízení koncových jednotek a kontrolu rozhraní [4].

Standard definuje několik základních prvků, které jsou v GPON sítích využívány:

- **Optická distribuční síť – ODN**
  ODN (Optical Distribution network) tvoří pasivní prvky. Patří zde optická vlákna, optické splittery, vlnové filtry a jiné.
- **Optické linkové zakončení – OLT**
  OLT (Optical Line termination) je aktivní kořenový prvek sítě. Zakončuje distribuční síť na straně poskytovatele. Implementuje PON protokol a stará se o správu a řízení komunikace spolu s aktivací a autentizací koncových ONU prvků.
- **Optická síťová jednotka – ONU/ONT**
  ONU (Optical Network Unit) je aktivní prvek sítě na straně koncového uživatele. Převádí optický signál na elektrický a opačně. K této jednotce může být obecně připojeno i více koncových zařízení.



Obr. 1.1: Zjednodušená topologie GPON sítě.

1.1 Základní charakteristiky

Technologie rozlišuje dva směry přenosu a to sestupný (OLT -> ONU) a vzestupný (ONU -> OLT), pro které definuje přenosové rychlosti dle tab.1.1. Nejčastěji používanou kombinací ISP (Internet Service Provider) je 2,4 Gb/s pro sestupný a 1,2 Gb/s pro vzestupný směr.

Tab. 1.1: Přenosové rychlosti GPON.

Směr přenosu	Přenosová rychlost
Sestupný	2,48832 Gb/s
Vzestupný	2,48832 Gb/s
	1,24416 Gb/s

Logický dosah mezi OLT a ONU jednotkami je dle standardu udáván 60 km. Fyzický dosah je poté 20 km. Na trase jsou využívány optické rozbočovače, ty mohou využívat rozbočovacího poměru typicky 1:64, pro krátké vzdálenosti až 1:128 [3].

Díky využití vlnového multiplexu WDM (Wavelength-division multiplexing) je pro přenos použito jedno vlákno. Pro oba směry jsou vyhrazeny rozdílné rozsahy vlnových délek 1480–1500 nm pro sestupný a 1260–1360 nm pro vzestupný směr [5].

1.2 Princip komunikace

Technologie GPON využívá rozdílný způsob komunikace pro oba směry. V sestupném směru je přenos centralizovaný, OLT posílá data všesměrově ke všem jednotkám ONU. Každé ONU poté na základě jedinečného identifikátoru GEM Port-ID filtruje ze zprávy data určené danému ONU [6].

U vzestupného směru je zvolen distribuovaný přístup. Data jsou posílána principem point-to-point. Aby nedocházelo ke kolizím, je z důvodu synchronizace využito TDMA (Time Division Multiple Access). OLT dynamicky přiděluje jednotlivým ONU časová okna, ve kterých mohou vysílat data. Šířka pásma je dynamicky přidělována na základě přenosových kontejnerů T-CONT (Transmission container). Pro zavedení kvality služeb je definováno 5 typů kontejnerů. Každé ONU pak posílá data pomocí jednoho nebo více T-CONT [6].

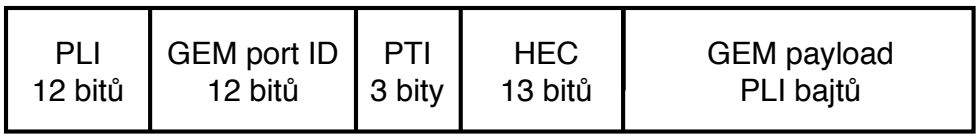
1.2.1 Sestupný směr přenosu

Při komunikaci v sestupném směru je využito dvojitého zapouzdření. Nejdříve jsou data zapouzdřována do rámce GEM (GPON Encapsulation Method). Ty jsou tvořeny 5 bajtovou hlavičkou a samotnými daty různé délky.

GEM rámce následně OLT multiplexuje do rámce GTC (GPON Transmission Convergence Layer). Tyto rámce jsou OLT všesměrově vysílány k ONU jednotkám. Koncové ONU jednotky filtrují jim příslušná data pomocí identifikátoru GEM Port-ID obsaženého v GEM hlavičce. Irelevantní GEM rámce jsou zahozeny [3].

Struktura GEM rámce

Hlavička GEM rámce, viz obr. 1.3, obsahuje pole **PLI** (Payload Length Indicator), které označuje délku přenášených dat, payload. Další pole obsahuje **GEM Port-ID**. Tento identifikátor nabývá hodnot 0–4095 a označuje jednotlivá logická spojení. Jeho přidělení zajišťuje OLT pomocí PLOAM (Physical Layer Operation, Administration and Maintenance) zprávy Configure port-ID. Pole **PTI** (Payload Type Indicator) označuje typ přenášených dat. Poslední pole **HEC** (Header Error Control) pak slouží k detekci a korekci chyb hlavičky [6].

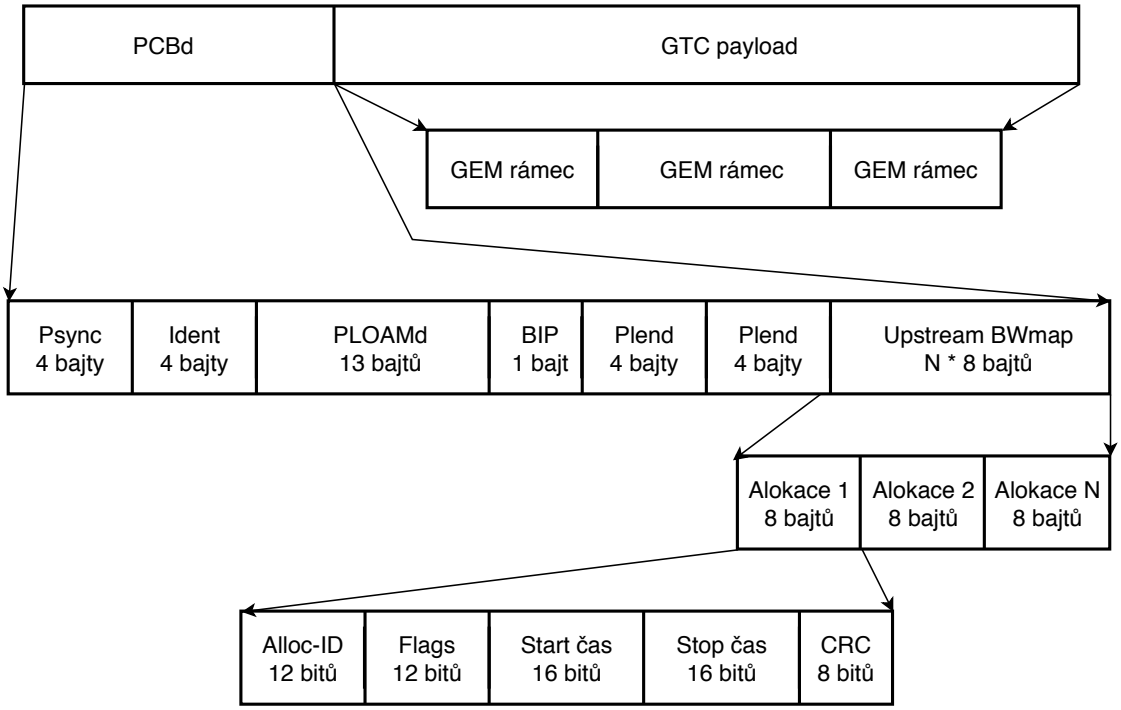


Obr. 1.2: Struktura GEM rámce [3].

Přenášená data jsou obsažena v Payload GEM rámce. Do payload jsou přímo mapovány Ethernet rámce, IP (Internet Protocol) pakety nebo MPLS (Multi-protocol Label Switching) pakety. V případě potřeby mohou být data rozdělena do více GEM rámců, zde jsou pak uplatněna fragmentační pravidla. GEM rámce, ale nikdy nepřenášejí více než jeden ethernet rámec nebo paket [3].

Struktura GTC rámce

GTC rámce mají pevně definovaný časový interval 125 μs a délku 38880 bajtů, odpovídající přenosové rychlosti 2,4832 Gb/s. Rámce jsou skramblovány pomocí polynomu $x^7 + x^6 + 1$ s použitím posuvného registru modulo 2 [3].

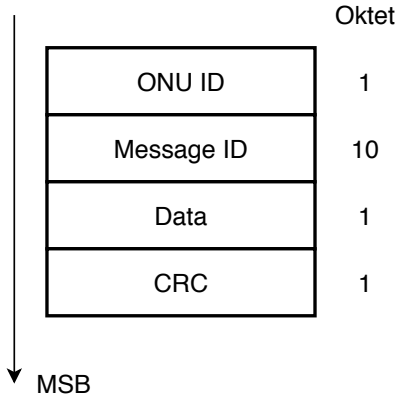


Obr. 1.3: Struktura GTC rámce [3].

Rámec GTC, viz obr. 1.3, je složen ze záhlaví **PCBd** (Physical Control Block Downstream) a payload nesoucího GEM rámce. Délka záhlaví závisí na počtu připojených ONU jednotek. Hlavička začíná 32-bitovým synchronizačním blokem **Psync** (Physical Synchronization), ten slouží k označení začátku GTC rámce a není skramblován. Pole **Ident** používá první bit pro identifikaci použité chybové korekce FEC (Forward Error Correction), následující bit je rezervován a zbývající bity slouží jako čítač rámce. Část **PLOAMd** nese PLOAM zprávu, dále popsáno v následující části. Pole **BIP** (Bit Interleaved Parity) obsahuje paritní bity, sloužící k nalezení chyb v záhlaví. Pole **Plend** udává délku části BWmap. Je posíláno dvakrát pro větší odolnost proti chybám. Poslední část záhlaví **Upstream BWmap** (Bandwidth map) nese informace o přidělení šířky pásma pro vzestupný směr přenosu. Skládá se z 8 bajtových sekcí pro každé ONU. Tyto sekce poté obsahují identifikaci ONU, identifikaci T-CONT kontejneru a přidělený vysílací interval [3].

Zprávy PLOAM

Zprávy PLOAM umožňují OLT správu komunikace. Jsou využívány při aktivaci ONU jednotek, konfiguraci šifrování, správě klíčů a signalizaci. Délka zprávy je 13 bajtů a je složena ze 4 prvků viz obr. 1.4.



Obr. 1.4: Struktura PLOAM zprávy [3].

ONU ID	Identifikátor příslušného ONU. Během aktivace je ONU přidělen OLT. Nabývá hodnot od 0 do 253.
Message ID	Identifikuje typ zprávy
Data	Nese data samotné zprávy (payload)
CRC	Kontrolní součet

Doporučení definuje 18 typů PLOAMd zpráv pro sestupný přenos a 9 typů PLOAMu zpráv pro vzestupný přenos. Níže je uvedeno několik běžně využívaných zpráv. Kompletní výčet lze nalézt v kapitole 9 doporučení ITU-T G.984.3 viz [3].

Sestupný směr

1	**Upstream Overhead**	používána při aktivačním procesu, obsahuje instrukce pro ONU k přidělení globálních parametrů pro vzestupný přenos
2	**Assign ONU-ID**	přidělení jedinečného ONU-ID na základě sériového čísla
3	**Ranging Time**	používána při aktivačním procesu pro přidělení ekvalizačního času, který vyrovnává logickou vzdálenost
4	**Deactivate ONU-ID**	po obdržení ONU přestává vysílat, resetuje své nastavení a přechází do Standby stavu (O2)

5	**Disable Serial Number**	po obdržení ONU přeruší vysílání a přechází do Emergency state stavu (O7)
7	**Request Password**	používána při autentizaci, žádost na zaslání registration ID
8	**Assign Alloc ID**	přiděluje ONU specifické Alloc ID, pro aktivaci T-CONT
9	**No message**	prázdná zpráva
10	**POPUP**	po obdržení ONU v POPU stavu (O6) přechází zpět Operation state stavu (O5), případně do Ranging state stavu (O4)
11	**Request Key**	po obdržení ONU generuje nový šifrovací klíč, který posílá OLT
12	**Configure Port-ID**	přiděluje OMCI kanálu 12-bitový identifikátor
17	**Key Switching Time**	po obdržení ONU generuje nový šifrovací klíč, který posílá OLT

Vzestupný směr

1	**Serial Number ONU**	obsahuje sériové číslo ONU jednotky
2	**Password**	obsahuje registration ID, používána při autentizaci
4	**No message**	prázdná zpráva
5	**Encryption key**	obsahuje fragment nového šifrovacího klíče
9	**Acknowledge**	slouží k potvrzení přijetí sestupných zpráv

1.2.2 Vzestupný směr přenosu

Velikost i časový interval rámce pro vzestupný směr jsou stejné jako pro směr sestupný. Pro zajištění synchronizace přenosu vysílají jednotlivé ONU v přidělených časových intervalech obdržených v poli Upstream BWmap rámce sestupného směru.

O přidělení šířky pásma se stará OLT. Šířka pásma může být přidělena staticky nebo dynamicky pomocí techniky dynamické alokace DBA (Dynamic Bandwidth Assignment) [7].

Statické přidělení šířky pásma může být výhodné u služeb s konstantním přenosem dat jako VoIP (Voice over Internet Protocol). Navzdory tomu je metoda dosti

neefektivní, jednotky ONU, které v danou chvíli nevysílají žádná data stále blokují přiřazenou šířku pásma [7].

U metody DBA jednotlivé ONU jednotky informují OLT o svém stavu. Spolu s daty zasílají požadavky na přidělení vysílacího okna. Pro tyto požadavky je využito explicitní hodnoty zaplnění vysílacího bufferu pro daný kontejner T-CONT. Pokud jednotka nemá žádná data k vyslání, posílá nulový rámec s údaji o prázdném bufferu. Šířka pásma tak může být přidělena jiným jednotkám [3].

Přenosové kontejnery T-CONT

Pro přenos ve vzestupném směru je využito přenosových kontejnerů. Tyto kontejnery slouží k přidělení šířky pásma a zajištění kvality služeb. Reprezentují skupiny logických spojení. ONU jednotky mohou posílat data pomocí jednoho nebo více těchto kontejnerů. Pro každé ONU je počet podporovaných kontejnerů pevně daný. Tyto kontejnery jsou vytvořeny automaticky při aktivaci ONU jednotky. Každá ONU jednotka má implicitně přiřazeno výchozí Alloc-ID číselně shodné s ONU-ID. OLT následně aktivuje jednotlivé kontejnery přiřazením dodatečného Alloc-ID identifikátoru pomocí PLOAM zprávy Assign Alloc ID [3].

Na základě několika parametrů šířky pásma definuje doporučení 5 typů T-CONT kontejnerů [3]:

T-CONT typ 1 přiděluje pevnou šířku pásma, vhodné pro služby pracující v reálném čase citlivé na zpoždění, např. VoIP (Voice over Internet Protocol)

T-CONT typ 2 zajišťuje garantovanou šířku pásma, pro aplikace nepracující v reálném čase

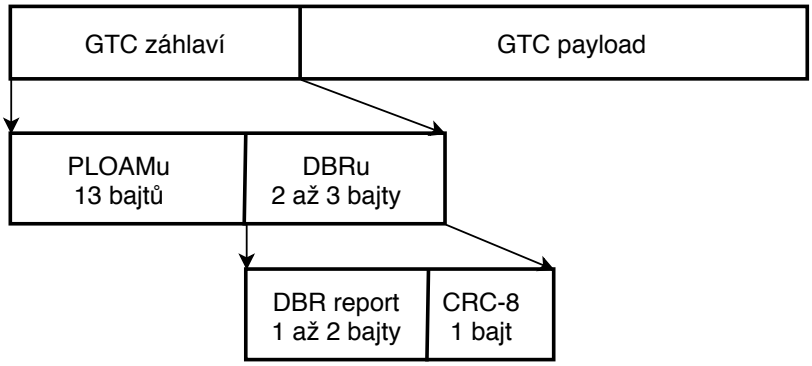
T-CONT typ 3 na základě volné kapacity umožňuje rozšíření garantované šířky pásma o negarantovanou

T-CONT typ 4 dynamicky přiděluje šířku pásma formou best-effort, šířka pásma je negarantovaná, vhodné pro aplikace komunikující v různě velkých rámcových shlucích

T-CONT typ 5 kombinuje přístup všech předchozích typů, použitelný pro většinu obecného datového toku

Struktura rámce

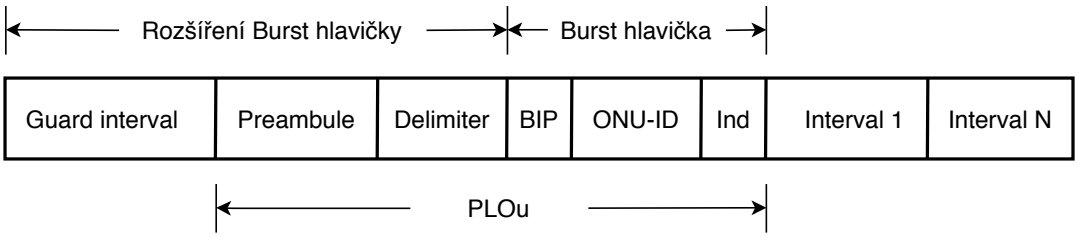
Vzestupné rámce, viz obr.1.5, jsou složeny z GTC záhlaví a samotného payload nesoucího GEM rámce. Pro přenos těchto rámců je využito předem obdrženého časového intervalu. Začátek a konec rámce je roven start a stop časům.



Obr. 1.5: Struktura vzestupného rámce [6].

Záhlaví, viz obr. 1.5, může obsahovat PLOAMu a DBRu (Dynamic Bandwidth Report upstream). Zpráva **PLOAMu** je přenášena pouze je-li časový interval dedikován výchozímu Alloc-ID. Pole **DBRu** obsahuje informace o zaplnění bufferu pro daný T-CONT kontejner [6].

Jednotlivé intervaly GTC rámců jsou následně skládány do tzv. burst struktury viz obr. 1.6. Součástí této burst struktury je PLOu záhlaví. Mimo PLOu záhlaví je však přítomen i **Guard interval** reprezentující čas sepnutí laseru. Tento interval je spolu s **Preambulí** a **Delimiterem** součástí rozšiřující burst hlavičky. Tyto parametry jsou nastaveny OLT a obdrženy při aktivaci ONU jednotky pomocí PLOAM zprávy Upstream overhead [6].



Obr. 1.6: Struktura Burst [6].

Samotná burst hlavička je tvořena 3 bajty. Pole **BIP** nese 8 paritních bitů. Následujících 8 bitů tvoří identifikátor **ONU-ID**. Indikační pole **Ind** pak umožňuje hlášení OLT, nastavením příslušných bitů je možné indikovat čekající PLOAM zprávu k zaslání (bit 7), použití proti-chybového kódování FEC (bit 6) nebo selhání

signálu sestupného směru (bit 5). Zbylé bity pole Ind jsou rezervovány [6].

1.3 Aktivační proces

Po připojení ONU jednotky do sítě není umožněna okamžitá komunikace z důvodu možného vzniku kolizí. Nově připojenou jednotku ONU je nutné nejdříve aktivovat. Proces aktivace řídí OLT a je složen z několika kroků, během kterých ONU přechází mezi 7 operačními stavy [3].

- O1 Initial state
- O2 Standby state
- O3 Serial Number state
- O4 Ranging state
- O5 Operation state
- O6 POPUP state
- O7 Emergency stop state

Během výchozího stavu **Initial state (O1)** nastaví ONU jednotka parametry ztráty rámce LoF (Loss of Frame) a signálu LoS (Loss of Signal). ONU následně naslouchá sestupné komunikaci a zachytává synchronizační blok Psync, který není skramblován. Po přijetí definovaného počtu synchronizačních bloků ONU nuluje parametry LoF a LoS a přechází do stavu O2.

Během **Standby state (O2)** stavu čeká ONU jednotka na přidělení globálních parametrů jako úroveň výkonu, před-přidělené zpoždění a hodnotu delimiteru. Tyto parametry jsou obsaženy v PLOAM zprávě Upstream overhead (1). Po nastavení obdržených parametrů přechází jednotka do stavu O3.

Stav **Serial number state (O3)** slouží ONU jednotce k ohlášení svého sériového čísla OLT. Nejprve je vytvořeno tiché okno, to je umožněno OLT pomocí zaslání zprávy s prázdným polem BWmap. Následně je zaslána žádost o sériové číslo. Jednotka ONU na žádost odpovídá a posílá své sériové číslo pomocí PLOAM zprávy Serial number. Na základě obdrženého sériového čísla připraví OLT jedinečný identifikátor ONU-ID v rozsahu 0–253, který je zaslán ONU pomocí PLOAM zprávy Assign ONU-ID (2). Po úspěšném obdržení ONU-ID si jednotka identifikátor ukládá a přechází do stavu O4 [8].

Ranging state (O4) stav slouží k synchronizaci logické vzdálenosti mezi OLT a ONU jednotkami. Během tohoto stavu OLT vypočítá vyrovnávací zpoždění, které zasílá ONU pomocí PLOAM zprávy Ranging Time (3). Po obdržení zprávy ONU nahradí před-přidělené zpoždění nově obdrženým vyrovnávacím zpožděním a přechází do stavu O5 [3]. Ke zrušení neúspěšných pokusů o aktivaci je využit časovač TO1. Tento časovač je spuštěn po vstupu ONU do stavu O4 a udává dobu, po kterou může

ONU v tomto stavu setrvat. Doporučená hodnota je 10 s. Pokud není synchronizace úspěšná před uplynutím času, přechází ONU zpět do stavu O2.

Dosažením stavu **Operation state (O5)** je ONU jednotka plně schopna zasílat data a PLOAM zprávy vzestupným směrem k OLT, a je tak ONU umožněna obousměrná komunikace. V tomto stavu pak jednotka ONU setrvává dokud nenastane ztráta synchronizace či signálu, v tomto případě přechází ONU do stavu O6. Obdržením žádosti o deaktivaci Deactivate ONU ID (4) přechází ONU zpět do stavu O2. V případě poruchy ONU je OLT umožněno vzdáleně přerušit vzestupnou komunikaci pomocí PLOAM zprávy Disable serial number (5), po obdržení této zprávy přechází ONU do stavu O7.

Detekováním ztráty synchronizace nebo signálu pomocí LoF a LoS alarmů přechází jednotka ONU do stavu **POPUP state (O6)**. Jednotka přestává vysílat a je spuštěn časovač TO2 (s počáteční hodnotou 100 ms). Během tohoto intervalu se ONU snaží obnovit synchronizaci zachycením PSync bloku. Pokud je synchronizace úspěšně obnovena a následně obdržena směrovaná/všesměrová PLOAM zpráva POPUP (10) přechází ONU do stavu O5/O4. V opačném případě přechází jednotka ONU do stavu O1.

Obdržení PLOAM zprávy Disable serial number (5) přechází ONU do stavu E**mergency State (O7)**, okamžitě přeruší vysílání a vypíná laser. Zpráva je posílána třikrát, pokud ani po třetí zprávě ONU nevypíná laser, nejspíše vznikla hardwarová porucha a jednotku ONU je nutné manuálně odpojit. Jednotky ONU ve stavu O7 mohou být znovu vzdáleně aktivovány pomocí stejné PLOAM zprávy s příznakem enable, obdržením této zprávy je ONU resetováno a přechází do stavu O2 [8].

1.4 Správa a Kontrola ONU

Doporučení definuje mechanismy a zprávy protokolu OMCI (ONU Management Control Interface), které OLT jednotky využívají ke správě, konfiguraci a kontrole ONU jednotek. Zprávy protokolu OMCI jsou zapouzdřeny v GEM rámci a jsou přenášeny pomocí OMCC (Onu Management and Control Channel) kanálu. Kontrolní mechanismy OMCI umožňují OLT:

- Navázat a ukončit spojení s ONU jednotkami,
- Správu rozhraní UNI (User Network Interface) jednotek ONU,
- Vyžádat si konfigurační informace a výkonnostní statistiky.
- Monitorovat systémová selhání a události jako selhání linek pomocí hlášení.

Dále pak mechanismy správy OMCI umožňují OLT:

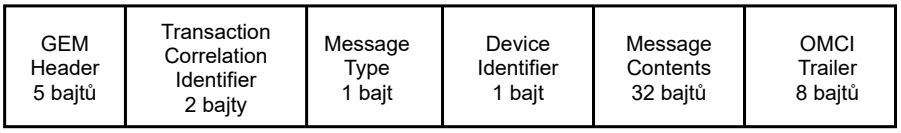
- **Správu konfigurace** – Definuje a podporuje základní konfiguraci jako restart systému, spuštění vlastních testů a konfiguraci rozhraní a portů.

- **Chybovou správu** – Podporovaná chybová správa je omezená, jedná se zejména o indikace systémového selhání a selhání rozhraní.
- **Výkonnostní správu** – Slouží ke sběru a dotazování se na výkonnostní statistiky a monitorování datového toku.
- **Správu zabezpečení** – Umožňuje povolit nebo zakázat šifrování sestupného směru.

Jednotky ONU pomocí těchto zpráv informují OLT o svém stavu, chybových hlášení a hodnotě výkonu.

1.4.1 Formát OMCI zprávy

Zprávy OMCI mají fixní délku 53 bajtů, tedy 5 bajtů GEM hlavička a 48 bajtů datové jednotky. Celé zprávy jsou zapouzdřeny v GEM rámci. Struktura zpráv je znázorněna viz obr.



Obr. 1.7: Struktura OMCI zprávy [4].

Pole hlavičky **GEM Header** obsahuje Port-ID. Následující identifikátor **Transaction correlation identifier** slouží ke spojení dotazu a odpovědi. Dotazu je identifikátor přiřazen, odpověď pak musí obsahovat stejnou hodnotu. Pro označení události použita hodnota 0x0000. Nejvýznamnější bit může být využit k ručení priority, kdy 0 značí nízkou a 1 vysokou prioritu. Pole **Message Type** označuje typ zprávy a je rozděleno na 4 části:

- **DB** (Destination Bit)– 8-bit: rezervován, nastaven na hodnotu 0,
- **AR** (Acknowledge Request)– 7-bit: značí zda zpráva vyžaduje potvrzení (1=ano, 0=ne),
- **AK** (Acknowledgement)– 6-bit: značí zda se jedná o potvrzení (1=ano, 0=ne),
- **MT** (Message Type)– bity 5-1: značí typ přenášené zprávy, doporučení definuje zprávy s hodnotami od 4 do 28, ostatní hodnoty jsou rezervovány pro budoucí použití.

Mezi doporučením definované typy zpráv patří například:

16	**Alarm**	oznámení hlášení
18	**Test**	požadavek testu specifické entity
19	**Start software download**	spuštění akce stažení software

24	**Synchronize Time**	synchronizace času mezi OLT a ONU
25	**Reboot**	restart ONU
27	**Test result**	oznámení výsledku testu (odpověď na typ 18)

Hodnota pole **Device identifier** je definována a vždy nastavena na 0x0A. Pole **Message identifier** pak slouží k identifikaci cílené entity a instance zprávy. Jednotlivé entity jsou definovány doporučením, některé hodnoty ponechány pro potřeby výrobce. Samotná data zprávy jsou obsažena v poli **Message content**. Strukturu jednotlivých typů zpráv definuje doporučení. Poslední pole **OMCI trailer** je rozděleno na 3 části kdy první 2 bajty jsou nastaveny na 0, další 2 bajty značí délku zprávy 0x28 a poslední 4 bajty slouží jako kontrolní součet CRC (Cyclic Redundancy Check).

Pokud se jedná o oznámení události, kdy byl parametr Transaction correlation identifier nastaven na hodnotu 0x0000, pak nejvýznamnější 4 bity pole Message content značí chybové hlášení a zbylé bity jsou vyplněny nulami. Typy možných chyb jsou:

0001 Command processing error	– chyba zpracování
0010 Command not supported	– nepodporovaný typ zprávy
0011 Parameter error	– chybná zpráva
0100 Unknown managed entity	– nepodporovaná entita
0101 Unknown managed entity instance	– nepodporovaná instance
0110 Device busy	– zpráva nebyla zpracována
0111 Attributes failed or unknown	– ONU nepodporuje dané atributy

1.4.2 Aktivace OMCC kanálu

Kanál OMCC k přenosu OMCI zpráv je aktivován v rámci aktivace ONU jednotky. Využívá virtuální přenosový kontejner OMCI T-CONT. Během procesu aktivace ONU jednotky je po obdržení PLOAM zprávy Assign ONU-ID (2) přiřazeno virtuálnímu kontejneru alloc-ID shodné s obdrženým ONU-ID. Není tak nutné zaslání další PLOAM zprávy Assign Alloc ID (8). Nicméně je toto výchozí alloc-ID možné změnit. Následuje standardní průběh aktivace ONU jednotky. Po úspěšné aktivaci je pro OMCI zprávy přiřazeno ONU jednotce GEM Port-ID. Jedná se o 12-bitový identifikátor, který přiřazuje OLT pomocí PLOAM zprávy Configure Port-ID (12). Jednotka ONU přiřadí Port-ID OMCC kanálu a přidělení OLT potvrdí pomocí PLOAM zprávy Acknowledge, čímž je OMCC kanál úspěšně navázán.

2 Bezpečnost a možná rizika GPON sítí

2.1 Zabezpečení

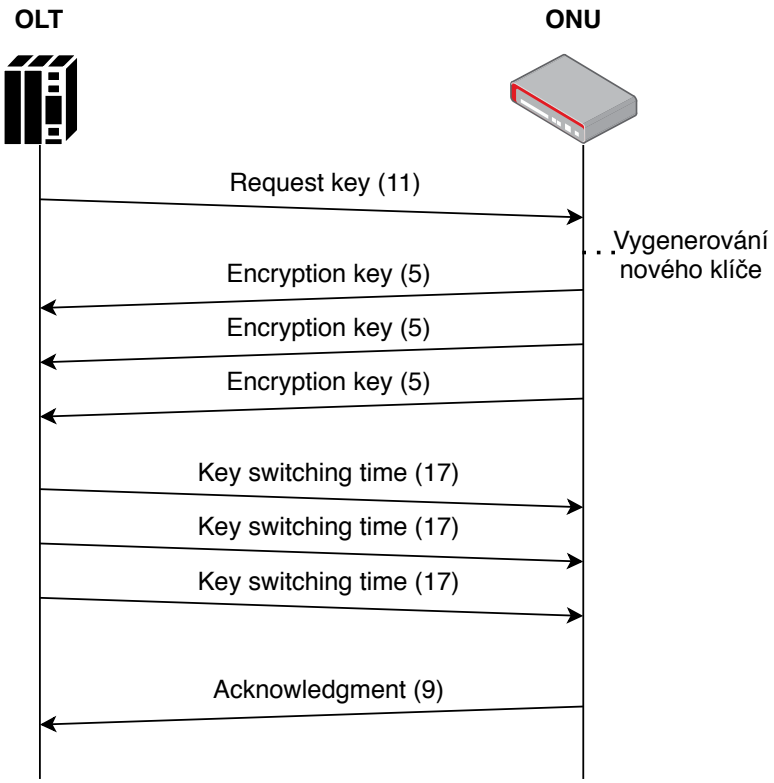
Zabezpečení GPON sítí bylo navrženo s ohledem na využité technologie. GPON síť využívá sdílené médium a data jsou tak vysílána všesměrově všem koncovým jednotkám. Úprava dané ONU, by tak umožnila odposlech komunikace sestupného směru všech ostatních ONU. Doporučení proto definuje využití šifrování. Šifrování je nepovinné a využito pouze na sestupný přenos z důvodu předpokladu náročnosti přístupu k fyzickému médiu [3]. Reálně však může být přístup k médiu relativně snadný, například pomocí volného portu ve splitteru. To dává útočníkům prostor k realizaci několika typů útoků popsaných dále.

2.1.1 Autentizace

Autentizace slouží k ověření odběratele poskytovaných služeb. Autentizace ONU je možná pomocí 2 metod. První metoda je založena pouze na důvěře sériového čísla. U této metody OLT porovnává obdržené sériové číslo s přednastavenými záznamy, případně kontroluje, zda již není aktivní jiná ONU se stejným sériovým číslem. Druhá metoda je založena na porovnání sériového čísla a parametru registration ID (Identifier). Parametr registration ID je přednastaven na OLT i ONU. OLT zahajuje proces autentizace zasláním PLOAM zprávy Request password (7), ONU odpovídá zasláním PLOAM zprávy Password (2). OLT následně porovnává obdržený registration ID s lokálním záznamem a v případě shody je autentizace úspěšná [6].

2.1.2 Výměna šifrovacích klíčů

Před zahájením šifrování je nutné ustanovit šifrovací klíče. Aby mohl začít proces výměny klíčů, musí daná ONU dosáhnout Operation state (O5) a být tak schopna obousměrného přenosu [9]. Tento proces viz obr. 2.1, je řízen OLT a začíná zasláním PLOAM zprávy Request key (11). Poté co ONU obdrží tuto zprávu, generuje šifrovací klíč na základě jedinečného parametru, tedy sériového čísla. Šifrovací klíč je následně zasílán OLT pomocí PLOAM zprávy Encryption key (5). Z důvodu délky je klíč rozdělen do několika zpráv, hlavička PLOAM zprávy pak obsahuje flag značící fragmentaci a sekvenční číslo fragmentu. Celý klíč je zaslán třikrát. Pokud se obdržený klíč na straně OLT ve všech třech případech shoduje OLT si jej ukládá [10]. OLT určí číslo rámce, od kterého se má nový klíč použít. Následně je číslo rámce zasláno ONU pomocí PLOAM zprávy Key switching time (17). Tato zpráva je zaslána třikrát, ONU stačí zachytit tuto zprávu pouze jednou. ONU musí potvrdit



Obr. 2.1: Proces ustanovení nového šiforovacího klíče [6].

přijetí zprávy (17) pomocí PLOAM zprávy Acknowledgment (9). Pokud po vyslání všech 3 zpráv ONU nepotvrdí přijetí, mělo by být deaktivováno. Pokud je proces ustanovení klíče úspěšný je nově zvolený klíč využit k šifrování rámce se zvoleným pořadovým číslem [3].

Vzhledem k možnostem odposlechu komunikace není přenos klíče v otevřeném textu bezpečný. Příloha standardu [11], proto doporučuje implementaci zabezpečení přenosu šifrovacího klíče. Doporučená metoda je závislá na autentizaci. V rámci procesu autentizace si obě jednotky spočítají klíč MSK (Master Session Key), který je následně použit pro zašifrování šifrovacího klíče pomocí AES (Advanced Encryption Standard) v módu ECB (Electronic Codebook mode) [11].

V průběhu let bylo rovněž představeno několik dalších návrhů pro zvýšení bezpečnosti procesu ustanovení klíče. Jedním z návrhů je například využití kvantové kryptografie [12]. Zařízení umožňující kvantovou distribuci klíče jsou v současnosti stále velmi drahá a jejich komerční využití zatím není příliš reálné. Jiným návrhem se zabývá práce [13], která představuje protokol na ustanovení klíče založený na protokolu Diffie-Hellman a časové hodnotě propagace signálu. Využití parametru RTT (Round Trip Time) v procesu ustanovení klíče se dále věnuje i práce [10].

Autoři práce [10] poukazují na fakt, že je klíč generován na základě jediného parametru, tedy sériového čísla. V rámci práce je představen návrh na ustanovení klíče s využitím dalšího unikátního parametru. Tento parametr představuje čas přenosu zprávy mezi OLT a ONU v μs. Parametr RTT by měl být OLT i ONU znám a jeho stanovení by nemělo představovat větší modifikace v rámci komunikace, autoři však uvažují i o jeho přenosu prostřednictvím PLOAM zprávy. V rámci testování byla ověřena jeho unikátnost s výslednou přesností na 126 m. Práce tak představuje zajímavý model ke zvýšení bezpečnosti procesu ustanovení klíče [10].

2.1.3 Šifrování

Standard definuje pro šifrování sestupného přenosu použití algoritmu AES s délkou klíče 128 bitů. Delší 192 a 265 bitové klíče by měly být v budoucnu podporovány, momentálně však jejich použití doporučení nespecifikuje. Šifrovací protokol AES je dle doporučení možné využít pouze v CTR (Counter) módu. Šifrována jsou pouze přenášená data, tedy payload GEM rámce. Využití šifrování je dle standardu nepovinné a ve výchozím stavu je neaktivní [3].

Proces šifrování je v režii OLT a ONU jednotek, práce [14], pak představuje vysokorychlostní bezpečnostní modul implementován pomocí FPGA. Tento modul, oproti řešením implementovaným některými výrobci, zefektivňuje některé dílčí operace protokolu AES a umožňuje dosažení propustnosti až 30 Gb/s.

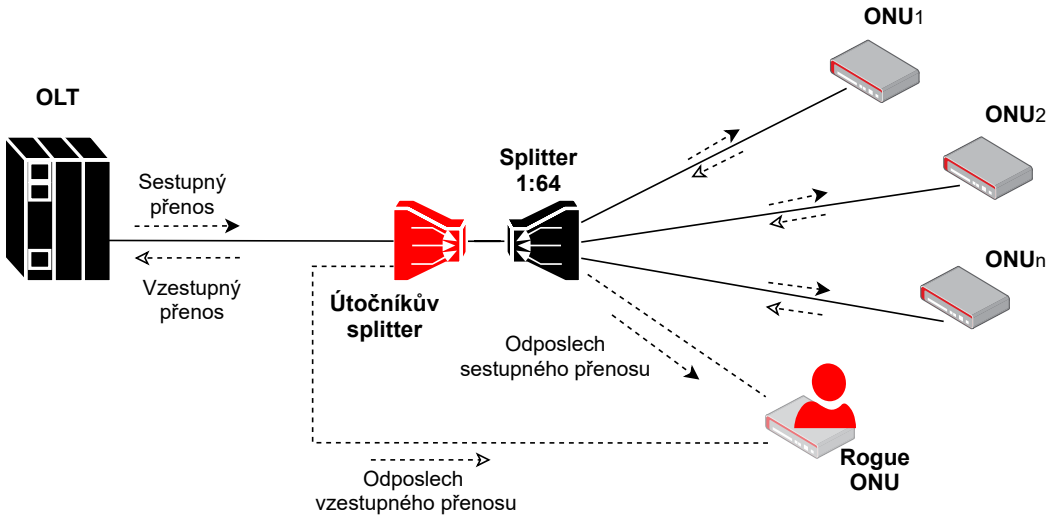
2.2 Bezpečnostní rizika

Prvky pro zajištění bezpečnosti specifikované standardy vycházejí z předpokladu, že fyzický přístup k médiu bude pro útočníka náročný. Vzestupný přenos je tedy považován za bezpečný. To, ale nemusí být v reálném prostředí vždy pravdou. Splittery jsou běžně instalovány například do sklepních prostor, kde nejsou nijak dále zabezpečeny [15]. Pokud jsou některé z portů volné může se útočník snadno připojit k síti. V opačném případě stačí od sítě odpojit legitimního uživatele a připojit vlastní zařízení.

V rámci GPON sítí je známo několik typů bezpečnostních rizik, která by neměla být ignorována. I tyto typy sítí se stávají cílem útočníků. Je proto potřeba dokázat tyto rizika identifikovat. Následující části se tak věnují krátkému popisu některých známých bezpečnostních rizik.

2.2.1 Odposlech

Pasivní odposlech v GPON sítích je umožněn díky charakteru komunikace. Data jsou v sestupném směru vysílána všesměrově. Upravená ONU v promiskuitním režimu, tak dokáže zachytávat veškerou komunikaci v sestupném směru. Tuto komunikaci je možné zachytit i jinými optickými detektory, takto zachycený signál je však nutné dále zpracovat [16]. Samotná realizace odposlechu je pak závislá pouze na přístupu k dané síti, například pomocí volného konektoru ve splitteru.



Obr. 2.2: Obecné schéma možného odposlechu [10].

Jako obrana proti odposlechu sestupného směru komunikace je využito šifrování, to však není povinné. Další nevýhodou je zasílání klíčů v otevřené podobě. Útočník, který tyto klíče zachytí, tak dokáže komunikaci dešifrovat. Toto zabezpečení, proto závisí na bezpečnosti vzestupného směru.

V normálním stavu není ONU schopna zachytit vzestupný přenos jiných ONU stanic. Jednoduchým řešením pro útočníka, viz obr. 2.2, by bylo připojení vlastního splitteru těsně za splitter legitimní [10]. Tímto by, ale došlo k narušení komunikace. Dalším rušivým variantám odposlechu optických vláken se věnuje práce [17]. Odposlechem pomocí makro-ohybů se dále zabývá práce [18], v rámci této práce je pomocí simulace testováno možné využití této metody.

Dalšímu způsobu odposlechu vzestupné komunikace založeném na zpětném odrazu optického signálu se věnuje práce [19]. Tyto odrazy mohou být způsobeny běžně používanými optickými komponenty, například splittery nebo různými optickými konektory. Testování proběhlo na různých konfiguracích sítě. Úspěšnost odposlechu pak závisí na využitých konektorech a typu fotodetektoru.

2.2.2 Denial of service – DoS

Útoky typu DoS (Denial of Service) cílí na znepřístupnění síťové služby pro legitimní uživatele. K zablokování komunikace vzestupného směru provozu dochází v případě, že některá z ONU vysílá mimo přidělený časový interval. Příčinnou DoS útoku může být i hardwarová nebo softwarová porucha ONU. Útočník, ale může ONU záměrně modifikovat, aby nepřetržitě vysílal na dané vlnové délce a dostatečným výkonem vysílacího signálu, tak zablokovat komunikaci ostatních ONU. Útok je možné realizovat i dostatečně výkonným laserovým zdrojem [20].

V rámci práce [21], bylo provedeno testování útoku na OLT firmy Huawei. Výsledky testování dokazují, že s dostatečně výkonným laserovým zdrojem, schopným vysílat na přesných vlnových délkách, je útočník schopen úspěšně zablokovat komunikaci.

Rušivý signál dokáže OLT detekovat, proti útoku však neexistuje efektivní ochrana. Díky pasivní síti není možné přesně identifikovat zdroj rušení a rozlišit poruchu od cíleného útoku. Základní ochranou by měl být zabezpečený přístup k médiu. Tématu se dále věnuje několik prací, jejich autoři se věnují návrhům různých metod detekce a mitigace. Příkladem pak může být metoda detekce založená na periodickém testovacím signálu [22].

2.2.3 Modifikované ONU

Jedním z největších rizik GPON sítí jsou upravené ONU tzv. rogue jednotky. Jedná se o jednotky s neoprávněně upraveným firmware pro potřeby útočníka. Tyto jednotky většinou využívají pasivního odposlechu pro získání informací k realizaci dalších aktivních útoků. Mezi tyto útoky může patřit:

- **Masquerade**:
  Pokud je útočník schopen zachytit a dešifrovat sestupnou i vzestupnou komunikaci, je schopen zachytit citlivé informace jiných ONU, jako jsou identifikátory, hesla a klíče. Odcizením těchto informací je útočník schopen efektivně vydávat své ONU za jiné ONU v síti [23].
- **TOS** (Theft of Service):
  Odcizením identity jiného ONU může útočník získat přístup ke službám či vyšší přenosové rychlosti. Za tyto služby však útočník neplatí a jsou účtovány uživateli ONU jehož identita byla odcizena [6].
- **Reply attack**:
  Pokud útočník zachytává komunikaci, může si zachycené rámce ukládat k jejich pozdějšímu využití. Tento typ útoku většinou cílí na PLOAM zprávy, ty pak mohou být i upraveny a přeposlány. Přeposílání těchto zpráv pak může narušit komunikaci, což pro síť působí určité riziko [6].

Detekce těchto modifikovaných jednotek v síti je náročná. Standard [24], se obecně věnuje principům a technikám, které by měly být zavedeny pro detekci, izolaci a mitigaci těchto jednotek. V rámci patentů a výzkumu bylo publikováno několik návrhů metod detekce. Patent [25], navrhuje metodu založenou na signalizaci a identifikačním kódu. Autoři práce [26], pak navrhují metodu detekce pomocí monitorování parametru FER (Frame Error Rate) každé ONU. V případě velkého množství kolizí je jednotka s nejmenší hodnotou FER následně považována za modifikovanou.

2.2.4 Zranitelnosti ONU jednotek

Cílem útoků se stávají i konkrétní ONU a OLT. Implementace protokolu PON není sjednocená a firmware zařízení různých výrobců může být naprogramován rozdílně. Tyto zařízení tak mohou obsahovat bezpečnostní slabiny. Například server vpnMentor publikoval v roce 2018 dvě kritické zranitelnosti domácích GPON routerů firmy DASAN [27]. Jedná se o zranitelnosti CVE-2018-10561 a CVE-2018-10562. Využití kombinace těchto zranitelností dává útočníkovi možnost naprosto obejít autentizaci přístupu ke správě zařízení a následné spuštění škodlivého kódu pomocí RCE (Remote Code Execution). Podle databáze Shodan bylo k datu vydání na internetu zranitelných až 1 milión zařízení [28]. K datu psaní práce se podle databáze Shodan, stále jedná o 169 tisíc zařízení [29]. Tyto exploity jsou na internetu snadno dostupné. Útoky byly využity převážně pro tvorbu botnetů, v rámci výzkumu bylo zaznamenáno až 5 různých botnet rodin, mimo jiné i variace známého Mirai Botnetu [30].

Podrobnějšímu testování ONU, převážně zařízení Alcatel od poskytovatele Orange, se věnuje autor, Pierre Kim, v práci [15]. V rámci práce byly například nalezeny přístupové údaje umožňující backdoor přístup k jednotce nebo možnost spuštění RCE. V další práci se autor věnuje také testování bezpečnosti OLT různých výrobců, u kterých byly rovněž nalezeny bezpečnostní zranitelnosti. U OLT se může jednat o backdoor přístup pomocí telnet nebo špatnou správu hesel [31].

Řada nalezených zranitelností je poté dostupná například v databázi VulDB. Nejnovější nálezy byly publikovány v roce 2019. Jedná se o potencionální zranitelnosti zařízení výrobce HiNet. Některé ze zranitelností již byly výrobci opraveny, jiné zůstávají potencionálně nebezpečné [32]. Konkrétní exploity jsou poté dostupné například v databázi Exploit Database. Zde je nejnovějším příspěvkem kód umožňující RCE cílený na Netlink routery, publikovaný v roce 2020 [33].

3 Bezpečnostní incidenty

Běžní uživatelé, ale i větší či menší společnosti, denně čelí kybernetickým útokům prostřednictvím internetových sítí. Útočníci necílí pouze na konkrétní subjekty, ale hlavně také na různé bezpečnostní slabiny jednotlivých technologií. Těchto slabin využívají k vyhledání zranitelných systémů a následným obecným útokům na sítě s cílem zamezení přístupu ke službám, získání přístupu či krádeži nebo zničení dat s primárním cílem výdělku. Konkrétním případem může být kybernetický útok na Brněnské nemocnice z března roku 2020 [34].

Úkolem správce sítě je provoz v určité míře monitorovat a v případě možného narušení bezpečnosti situaci patřičným způsobem analyzovat a v co nejkratším čase na ni reagovat. Co považovat za bezpečnostní incident, kdo je povinen jej hlásit a komu, definuje Česká legislativa viz dále.

3.1 Právní úprava

Oblast bezpečnostních incidentů upravuje Zákon č.181/204 Sb., o kybernetické bezpečnosti (ZoKB) [35], a Vyhláška 82/2018 Sb., o kybernetické bezpečnosti [36]. Podle § 3, Zákona č.181/204 Sb. je povinnost v oblasti kybernetické bezpečnosti uložena orgánům a osobám, kterými jsou [35]:

a) „*poskytovatel služby elektronických komunikací a subjekt zajišťující síť elektronických komunikací1), pokud není orgánem nebo osobou podle písmene b)*“,
b) „*orgán nebo osoba zajišťující významnou síť, pokud nejsou správcem nebo provozovatelem komunikačního systému podle písmene d)*“,
c) „*správce a provozovatel informačního systému kritické informační infrastruktury*“,
d) „*správce a provozovatel komunikačního systému kritické informační infrastruktury*“,
e) „*správce a provozovatel významného informačního systému*“,
f) „*správce a provozovatel informačního systémus základní služby, pokud nejsou správcem nebo provozovatelem podle písmene c) nebo d)*“,
g) „*provozovatel základní služby, pokud není správcem nebo provozovatelem podle písmene f), a*“
h) „*poskytovatel digitální služby*“.

Orgány a osoby b) až f) jsou podle § 7, Zákona č.181/204 Sb. „*povinny detekovat kybernetické bezpečnostní události v jejich významné síti, informačním systému kritické informační infrastruktury, komunikačním systému kritické informační*

infrastruktury, informačním systému základní služby nebo významném informačním systému“ [35]. V rámci legislativy je důležité rozlišovat pojmy události a incidentu. Dle § 7, Zákona č.181/204 Sb. jsou tyto pojmy definovány takto [35]:

- **Kybernetická bezpečnostní událost:** „*je událost, která může způsobit narušení bezpečnosti informací v informačních systémech nebo narušení bezpečnosti služeb anebo bezpečnosti a integrity sítí elektronických komunikací.*“
- **Kybernetický bezpečnostní incident:** „*je narušení bezpečnosti informací v informačních systémech nebo narušení bezpečnosti služeb anebo bezpečnosti a integrity sítí elektronických komunikací v důsledku kybernetické bezpečnostní události.*“

Incidentem je tedy událost, při které přímo dochází k narušení bezpečnosti. Osoby a orgány b) až f) podle § 8, Zákona č.181/204 Sb. „*jsou povinny hlásit kybernetické bezpečnostní incidenty v jejich významné síti, informačním systému kritické informační infrastruktury, komunikačním systému kritické informační infrastruktury, informačním systému základní služby nebo významném informačním systému, a to bezodkladně po jejich detekci;...*“ [35]. Dále dle § 8, Zákona č.181/204 Sb. hlásí orgány a osoby b) a h) kybernetické bezpečnostní incidenty provozovateli národního CERT (Computer Emergency Response Team), orgány a osoby c) až g) hlásí kybernetické bezpečnostní incidenty incidenty Úřadu a orgány a osoby neuvedené v § 3 mohou hlásit incidenty provozovateli národního CERT nebo Úřadu [35].

3.2 Bezpečnostní týmy

V oblasti bezpečnostních incidentů působí různé bezpečnostní týmy. Tyto týmy mohou být součástí organizací nebo působit na národní či mezinárodní úrovni. Existuje několik typů těchto týmů z nichž každý má rozdílnou roli.

3.2.1 SOC (Security Operations Centers)

Bezpečnostní dohledové centrum je širším pojmem pro bezpečnostní týmy. Obecně však mají tyto týmy na starost především detekci potencionálních incidentů, monitorizaci síťového provozu, systémů a uživatelů. Dále se starají o sběr a analýzu dat z různých zdrojů jako jsou například systémové logy. Tyto týmy mohou také sledovat aktuální hrozby šířící se mimo interní systémy [37].

3.2.2 Týmy CERT a CSIRT

Oba pojmy CERT a CSIRT (Computer Security Incident Response Teams) mají trochu rozlišný význam, obecně jsou ale zaměňovány a vnímány jako synonyma. Tyto

týmy se zabývají reakcí na incidenty a koordinací řešení incidentů. Mohou působit na úrovni organizací, kde mají přímou možnost zasáhnout a například eliminovat probíhají útoky. Koordinačně zaměřené týmy obvykle nemají možnost přímého zásahu. Tyto týmy se zaměřují na spolupráci, komunikaci a zprostředkování informací. Týmy obecně incidenty komplexně vyhodnocují a koordinují řešení daných incidentů jak uvnitř organizací, tak i ostatními CERT/CSIRT týmy, orgány činnými v trestném řízení, s právníky či médii. Na úrovni státu České republiky působí rovněž i speciální národní a vládní CERT týmy [38].

3.2.3 Národní CERT

Národní týmy plní funkci poslední instance, kam je možné se obrátit o zásah a pomoc. Obvykle nemají možnost přímého zásahu do infrastruktury. Zabývají se hlavně zprostředkováním kontaktu, koordinací postupu řešitelů či spolupráci s více složkami. Tyto týmy mají zároveň na starost osvětu v oblasti kybernetické bezpečnosti [38].

Národní tým České republiky CSIRT.CZ je provozován sdružením CZ.NIC. Polem působnosti týmu jsou uživatelé a sítě provozované na území České republiky. Tým CSIRT.CZ v prostředí České republiky zajišťuje udržování zahraničních vztahů, spolupráci se subjekty v rámci České republiky, vzdělávací a školící činnosti a zejména poté řešení a koordinaci bezpečnostních incidentů spolu s proaktivními službami v oblasti bezpečnosti [39].

3.2.4 Vládní CERT

Vládní týmy chrání vládní sítě, kritické informační infrastruktury a významné informační systémy. Tyto týmy mají obvykle přístup k infrastruktuře a podporu v legislativě [38]. Týmy zároveň působí jako prvotní zdroj bezpečnostních informací a pomoc pro orgány státu, organizace i občany. Zároveň hrají při zvyšování vzdělanosti v oblasti bezpečnosti na internetu [40].

Vládní tým České republiky GovCERT.CZ je součástí Národního úřadu pro kybernetickou a informační bezpečnost. Orgány a osoby c) až g), uvedeny v kapitole 3.1 jsou podle § 8, Zákona č.181/204 Sb. povinny tomuto týmu bezodkladně hlásit bezpečnostní incidenty. Tým GovCERT.CZ v rámci dalších služeb poskytuje například penetrační testování, forenzní laboratoř a pořádá řadu vzdělávacích a výzkumných činností [40].

3.3 Možnosti hlášení

Forma a náležitosti hlášení kybernetických incidentů jsou upraveny vyhláškou 82/2018 Sb., o kybernetické bezpečnosti. Hlášení kybernetických incidentů se podává pomocí elektronických formulářů uvedených na internetových stránkách vládního nebo národního CERT týmu. Náležitostmi hlášení jsou podle § 32, 82/2018 Sb. ods. 4 [36]:

a) „*identifikace odesilatele*“,
b) „*identifikace informačního a komunikačního systému*“,
c) „*datum a čas zjištění incidentu a*“
d) „*popis incidentu*“.

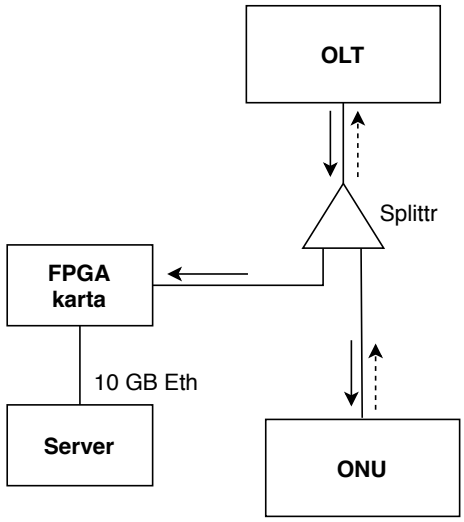
Hlášení bezpečnostních incidentů je aktuálně možné pouze e-mailem, datovou schránkou nebo prostřednictvím datového rozhraní na internetových stránkách [36]. Do budoucna připravuje tým GovCERT.CZ i možnost hlášení incidentů pomocí platformy MISP (Malware Information Sharing Platform). Tato možnost hlášení by mohla být spuštěna v průběhu roku 2021.

V rámci organizací mohou také interní týmy využívat k detekci, hlášení a analýze incidentů některé z dostupných nástrojů. Většina těchto nástrojů je však určena pro Ethernetové sítě. Práce je proto zaměřena na návrh možnosti hlášení bezpečnostních událostí v GPON sítích. Pro které jsou aktuální dostupné nástroje velmi omezené.

4 Analýza GPON komunikace

K detekci bezpečnostních událostí v GPON sítích je potřeba komunikaci nejdříve analyzovat. Jednou z možností analýzy GPON síťového provozu je využití FPGA analyzátoru navrženého v rámci projektu Redukce bezpečnostních hrozeb v optických sítích [41].

Systém je složen z několika částí, síťové FPGA karty, serveru, parseru a webového rozhraní. Pomocí karty je zachycena komunikace a GPON rámce jsou následně přesměrovány k serveru. Na serveru jsou rámce zachyceny pomocí síťového socketu a zpracovány parserem. Zpracovaná data jsou nakonec uložena do databáze. Webové rozhraní poskytuje správu systému a analýzu zachyceného provozu [42].



Obr. 4.1: Topologie připojení FPGA karty [42].

FPGA karta je přímo připojena do GPON sítě. Karta může být přímou součástí serverového řešení a připojena pomocí PCIe (Peripheral Component Interconnect Express) nebo může být připojena k serveru pomocí 10 Gbps Ethernetu. Aktuálně dokáže karta zachytit pouze sestupný provoz, rozšíření o možnost zachycení vzestupného provozu je předmětem budoucího rozvoje projektu. Zachycené rámce jsou deskramblovány a surová, „čitelná“, data jsou pomocí UDP (User Datagram Protocol) datagramů zaslány směrem k serveru [42].

Na serveru zachytává síťový socket obdržené UDP datagramy. Parser skládá GPON rámce a ukládá hodnoty jednotlivých polí do SQL (Structured Query Language) databáze. Data jsou uložena v binární podobě pro potřeby následné analýzy. Do databáze jsou ukládány pouze hlavičky GPON rámců a paketů. V rámci optimalizace datového úložiště jsou GPON hlavičky ukládány bez pole BWmap [42].

Zachycený přenos je analyzován pomocí strojového učení využívající TensorFlow. Data jsou pro potřebu analýzy převáděna do souborů JSON (JavaScript Object Notation) formátu. Předmětem analýzy jsou zejména data z hlaviček GPON rámců. V rámci výzkumu byly v zachycené komunikaci zjištěny nestandardní zprávy. Zejména v poli PLOAMd se hodnoty neshodují s žádnými hodnotami definovanými standardem. Tyto anomálie mohou mít určitý vliv na bezpečnost sítě. Proto je žádoucí anomálie monitorovat a dále vyhodnocovat [43].

V současné době neexistuje řešení pro hlášení bezpečnostních událostí GPON sítí. Dostupné nástroje pracují převážně s incidenty Ethernetových sítí. S rostoucí popularitou GPON sítí, nejen v České republice, je v zájmu tyto bezpečnostní rizika detekovat a dále je předat bezpečnostním týmům.

Cílem této práce je návrh a tvorba aplikace, která by umožnila detekované události automaticky hlásit bezpečnostním týmům pomocí některého z dostupných nástrojů. Případně možnost automaticky hlásit detekované incidenty patřičným úřadům.

5 Nástroje pro zpracování bezpečnostních incidentů

Bezpečnostní týmy využívají k detekci a řešení incidentů řadu nástrojů. Neexistuje jednotlivý standard, který by nařizoval použití konkrétních nástrojů, záleží na rozhodnutí jednotlivých týmů. Organizace jako ENISA (European Union Agency for Cybersecurity) nebo NIST (National Institute of Standards and Technology) však poskytují týmům určitá doporučení a postupy. Nástroje je možné rozdělit do několika obecných kategorií:

- **IDS** (Intrusion Detection Systems) – systémy detekce průniku, na základě známých signatur nebo anomálií (odchylek od normálního provozu) analyzují a monitorují síťový provoz.
- **IPS** (Intrusion Prevention Systems) – systémy prevence průniku rozšiřují funkci IDS o možnost aktivní blokace a filtrace provozu. Oba systémy IDS a IPS pak generují oznámení pro potřeby další analýzy.
- **SIEM** (Security Information and Event Management) – tyto systémy agregují a analyzují data získaná z různých síťových či systémových logů většinou s pomocí strojového učení. Na základě těchto dat dokáží rozpoznat potencionální hrozby a automaticky generovat alarmy.
- **SOAR** (Security Orchestration, Automation and Response) – systémy řízení, automatizace a řešení shromažďují a centralizují data a alarmy z různých zdrojů. Umožňují tak ucelený pohled na probíhající událost. Většinou integrují sadu dalších nástrojů pro zefektivnění procesu analýzy a řešení incidentů. Zároveň umožňují některé z procesů zautomatizovat.

Jednotlivé nástroje pracují s **IoC** (Indicators of Compromise). Indikátory kompromitace jsou forenzní data objevená při monitorizaci sítě nebo systému, která indikují potenciální průnik nebo škodlivou činnost. Obecně může jít například o IP adresu, malware singnatury, doménová jména, hash malware souborů a další [45]. Pro GPON sítě nejsou IoC definovány. Na základě detekce nestandardních PLOAM zpráv by mohly být právě tyto zprávy považovány za IoC.

5.1 Představení dostupných nástrojů

Práce je také mimo návrh samotného API rozhraní zaměřena na výběr vhodného systému, který by umožnil příjem a následnou analýzu hlášení z GPON sítí. Dostupné nástroje jsou určeny především pro Ethernet sítě, je tedy předpokládaná určitá nutnost modifikace systému pro příjem specifických identifikátorů kompromitace GPON sítí. Komerční nástroje proto nevyhovují účelu a práce je dále zaměřena

na nástroje typu open-source. V následující části je představeno několik dostupných nástrojů.

FIR (Fast Incident Response)

FIR je platforma pro správu incidentů navržena s ohledem na rychlost. Umožňuje jednoduchou tvorbu, sledování a hlášení bezpečnostních incidentů. Díky své jednoduchosti umožňuje snadnou úpravu podle potřeb. FIR je napsán pomocí programovacího jazyka Python a využívá Django 1.9. Systém není náročný na výkon, dle specifikace vyžaduje minimálně 1 jádro, a 40 GB disk a 1 GB RAM [46].

Wazuh

Wazuh je systém schopný monitorovat infrastrukturu, detekovat hrozby, pokusy průniku a anomálie v systému. Nenáročný agent systému umožňuje vykonávat řadu úkonů. Nástroj je schopný shromažďovat data z logů a událostí, monitorovat soubory a integritu systémových klíčů, monitorovat otevřené porty a síťová nastavení, detekovat malware a vykonávat proaktivní řešení. Systém je podporován více platformami a server tak může být naistalován na Windows, Linux, Mac OS a další. Hlášení jsou generována pomocí Elastic Stack a systém integruje webové rozhraní Kibana [47].

Cyphon

Cyphon je platformou pro správu incidentů. Realizuje sběr, zpracování a třídění událostí. Dokáže agregovat data z různých zdrojů a poskytuje možnost analýzy v rámci jednotné platformy. Proces analýzy usnadňuje díky možné eskalaci a sdílení hlášení mezi členy týmu a přiřazení výsledků analýzy daným událostem. Vizualizaci hlášení poskytuje pomocí uživatelského rozhraní. Hlášení jsou tříděna podle závažnosti [48].

Projekt TheHive

Projekt TheHive je systém umožňující spojení několika dílčích procesů pod jednu platformu. Bezpečnostní týmy mohou vzájemně pracovat na jednotlivých incidentech, systém také umožňuje rozdělení na organizace. Hlášení je možné zasílat z různých zdrojů pomocí thehive4py Python rozhraní. Obsah hlášení může být přizpůsoben. Hlášení je následně možno přidělit k analýze členům týmu díky přizpůsobitelným šablonám případů. Systém integruje nástroje jako Cortex a MISP, čímž usnadňuje procesy analýzy a reakce na incidenty. Díky integraci systému Cortex je umožněna automatizace [49].

Cortex

Systém Cortex je doplňujícím nástrojem projektu TheHive. Je zaměřen na analýzu pozorovaných identifikátorů a aktivní odpověď. Tyto procesy mohou být automatizovány a je tak usnadněno zpracování velkého množství pozorovaných identifikátorů. Pro analýzu a odpověď je možné využít některé z dostupných analyzačních či response modulů. Jednotlivé moduly mohou být upraveny nebo zcela vytvořeny podle vlastních potřeb. Systém rovněž poskytuje uživatelské webové rozhraní [50].

IntelMQ

IntelMQ je modulární nástroj pro plně automatický sběr, normalizaci, obohacování a distribuci datových kanálů, například systémových a síťových logů. Může být využit pro automatické zpracování incidentů, automatická oznámení nebo jako sběrač dat pro další nástroje. Poskytuje jednoduchou komunikaci s jinými systémy díky HTTP (Hypertext Transfer Protocol) text REST (Representational State Transfer) API. Pro všechny zprávy využívá formát JSON [51].

Elasticsearch

Elasticsearch je distribuovaný vyhledávací a analytický nástroj navržený pro rychlé vyhledávání plných textů a struktur. Nástroj využívá REST API rozhraní, ke kterému lze přistoupit pomocí nástroje curl nebo programovacím jazykem. Elasticsearch lze použít k ukládání velkého množství strukturovaných dat, na která je umožněno efektivní dotazovaní. Dále může být využit k interaktivní analýze a automatické detekci bezpečnostních hrozeb [52].

Spolu s nástrojem Elasticsearch lze využít nástroj Kibana, který poskytuje uživatelské webové rozhraní umožňující vizualizaci a analýzu sesbíraných dat.

Webhooks

Webhooks jsou uživatelsky definována HTTP zpětná volání. Jsou využívány pro předávání informací mezi jednotlivými systémy. Pomocí HTTP požadavků je možné vyvolat hlášení. Obsah požadavků je většinou v JSON nebo XML (Extensible Markup Language) formátu. Mohou být využity pro automatizaci bezpečnostních hlášení a reakcí na vzniklé incidenty [53].

Projekt MISP

MISP (Malware Information Sharing Platform) je platforma pro sběr, sdílení, ukládání a korelaci bezpečnostních indikátorů kompromitace, informací o zranitelnostech

a analýzu bezpečnostních incidentů a malware. Cílem projektu je jednoduché sdílení strukturovaných informací mezi bezpečnostními komunitami. Tyto informace pak mohou být využity pro detekci či analýzu dalšími systémy. Systém využívá pro sdílení dat stejnojmenný datový model MISP, založený na formátu JSON [54].

6 Návrh řešení

Tato část je zaměřena na porovnání dostupných nástrojů pro správu bezpečnostních incidentů a následný výběr vhodného kandidáta pro návrh řešení hlášení incidentů z GPON sítě. Následně je navržena testovací síť a otestována možnost zasílání hlášení.

6.1 Porovnání nástrojů

V předchozí kapitole bylo představeno několik dostupných nástrojů pro zpracování bezpečnostních incidentů. Ne všechny zmíněné nástroje spadají pod stejnou kategorii a řada z nich byla navržena pro odlišné účely. Nástroje se ale vzájemně doplňují. Některé z nástrojů vzájemně integrují další nástroje a umožňují spojení více procesů pod jednu platformu. Bezpečnostní týmy běžně využívají větší množství nástrojů souběžně.

Pro potřeby dalšího porovnání byly nástroje rozděleny do následujících kategorií.

Tab. 6.1: Kategorizace dostupných nástrojů.

Kategorie	**Nástroje**
Zpracování incidentů	FIR Wazuh Cyphon TheHive
Automatizace analýzy a odpovědí	Cortex
Sdílení	Webhooks MISP
Sběr a normalizace	IntelMQ
Databáze a vyhledávání	Elasticsearch

Nástroje mimo kategorii pro zpracování incidentů nejsou většinou využívány samostatně, ale tvoří část uceleného systému. Tyto nástroje nepatří mezi přímé kandidáty pro účely této práce, jsou však některými nástroji integrovány nebo se vzájemně doplňují.

6.1.1 Doplňující nástroje

Cortex je doplňkový nástroj systému TheHive, který umožňuje automatickou analýzu a odpovědi na incidenty. V roce 2018 poskytoval až 39 veřejných analyzátorů.

Příkladem může být analyzátor VirusTotal umožňující nahrání souboru a kontrolu dostupných hlášení spojených s hash tohoto souboru. Respondery pak slouží k automatizaci odpovědí. Zde se může jednat o automatické zaslaní emailové zprávy s výsledky analýzy potencionálních phishingových emailů hlášených uživateli. Jak analyzátory tak respondery mohou být vytvořeny vlastní [50].

Nástroj IntelMQ slouží ke sběru dat a následnému zpracování či normalizaci daných dat. Různé soubory, například výstupy detektorů, mohou být předány IntelMQ. Pomocí bot modulů může nástroj získat ze souborů důležité hodnoty. Z hodnot mohou být například odstraněny duplicity. Data je pak možné normalizovat a předat v definovaném výstupním formátu, např. JSON, dalším systémům. IntelMQ může být využit v kombinaci s Elasticsearch, kdy jsou výstupy ukládány do databáze.

Elasticsearch je typ databáze navržený pro ukládání velkého množství dat a rychlé vyhledávání. Jako datový typ využívá formátu JSON. K nástroji je umožněn přístup pomocí REST API nebo některého z podporovaných programovacích jazyků, Java, Python, PHP a další. Pro práci s nástrojem je možné využít webové rozhraní Kibana od stejné společnosti. V rámci produktů společnost Elastic nabízí například i SIEM pro detekci síťového provozu. Elasticsearch je využíván jako databáze v jádru systému TheHive.

Mezi bezpečnostními týmy je důležitá výměna informací, převážně IoC. Útoky cílené na jiné organizace totiž mohou být v budoucnosti cíleny i na danou interní organizaci. Je důležité mít informační povědomí o aktuálních hrozbách v podobných systémech či sítích. Díky těmto datům je možné útoky detekovat a včasně mitigovat. Menší organizace mohou ke sdílení IoC využívat například Webhooks. Ty umožní pomocí HTTP requestů podávat hlášení, ke kterým mohou přistoupit členové bezpečnostního týmu jiného oddělení organizace. K efektivnějšímu sdílení dat mezi týmy i mezi jednotlivými státy, je možné využít platformu MISP. Podle informací projektu využívá systém celosvětově více než 6000 organizací. MISP využívá například i komunita NATO (North Atlantic Treaty Organization) [54]. Systém MISP by měl být zprovozněn během roku 2021 i českým týmem GovCert.

6.1.2 Porovnání nástrojů pro zpracování incidentů

K porovnání byly vybrány 4 open source nástroje FIR, Wazuh, Cyphon a TheHive. Každý z nástrojů poskytuje trochu odlišné služby a vyniká v odlišných oblastech. Jednotlivé specifikace byly zapsány do tabulky, viz tab. 6.2.

Tab. 6.2: Porovnání nástrojů pro zpracování incidentů.

	FIR	**Wazuh**	**Cyphon**	**TheHive**
Vyvíjeno v	Python	Python	Python	Python
Instalace	Linux Docker	Linux Cloud Kubernetes Docker	Linux Widows Mac OS Docker	Linux Docker
GUI	Web UI Dashboard	Web UI Dashboard	Web UI Dashboard	Web UI Dashboard
Databáze	MySQL	Elasticsearch	PostgreSQL	Elasticsearch
Integrace nástrojů		Elastic Stack	Splunk VirusTotal Snort	Cortex MISP Webhooks
API	Python	Python Curl PowerShell	Django Logstash	thehive4py (Python) Curl
Výhody	nenáročný jednoduchý	oficiální podpora	Twitter API IoT data	šablony postupů custom indikátory dělení organizací aktivní komunita

Systém FIR byl navržen s ohledem na rychlost a jednoduchost. V porovnání s ostatními nástroji má nejnižší hardwarové nároky. Díky jeho obecnosti si jej může každý tým upravit podle vlastních potřeb. Je poskytnuto jednoduché webové UI (User Interface) a umožněna správa uživatelů. Systém je možné spravovat pomocí API v jazyce Python. Určitou nevýhodou může být menší aktivita vývojářů, systém není pravidelně aktualizován.

Systém Wazuh je složen z agentů a serveru. Agenti sbírají data z různých zdrojů a vykonávají automatické odpovědi, server pak analyzuje získaná data a provádí správu agentů. Umožňuje implementaci pomocí cloudového řešení. V základu integruje nástroj Elastic Stack, data jsou tak ukládána do databáze Elasticsearch. Webové UI je propojeno s rozhraním Kibana, jednou z komponent Elastic Stack. K systému je možné přistoupit i přímo pomocí Curl. Systém je oficiálně podporován a vývojáři jsou nabízena i tréninková sezení.

Platforma Cyphon umožňuje instalaci na většinu operačních systémů. Pro vizualizaci dat poskytuje odděleně vyvíjené webové rozhraní Cyclops. Jednotlivé incidenty

jsou přehledně tříděny podle závažnosti. Díky automatické analýze a integraci dalších nástrojů dokáže sbírat například i geolokační data. Rovněž podporuje sběr dat ze sociálních médií (např. pomocí Twitter API) a různých senzorů a zařízení IoT (Internet of Things). Platforma podporuje a integruje řadu nástrojů jako VirusTotal, Splunk, Snort, Sophos a dalších. Systém je možné spravovat pomocí Django API.

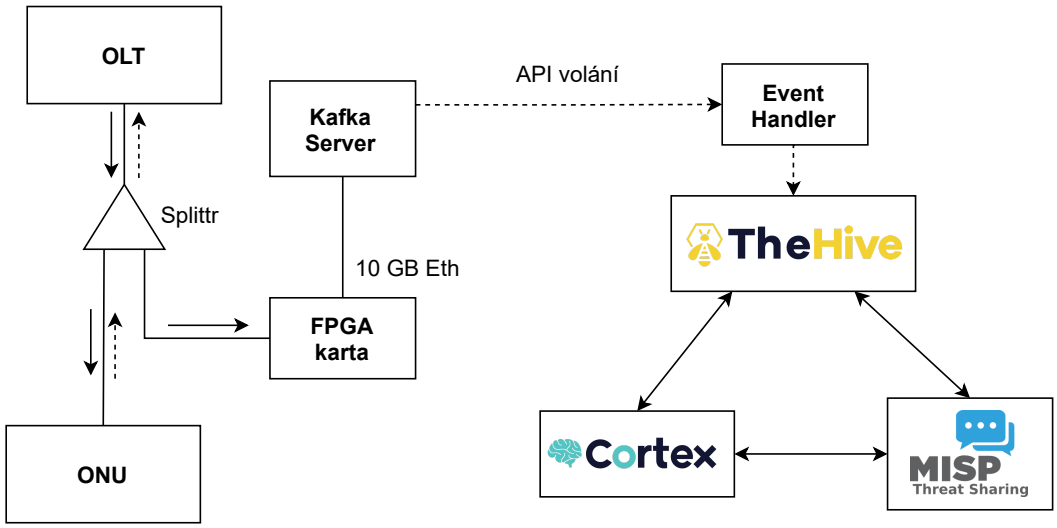
Projekt TheHive je škálovatelnou platformou. Hlavní výhodou je jeho integrace a pevné propojení se systémy MISP a Cortex. Spojení těchto tří nástrojů pak tvoří ucelený systém. TheHive umožňuje rozdělení systému na „organizace“, pod které jsou pak přiřazeni jednotliví uživatelé. V rámci správy uživatelů je možná autorizace a omezení přístupu uživatelů k určitým službám. Systém pracuje v reálném čase a umožňuje více uživatelům současně pracovat na jedné události. Pro jednotlivé incidenty je možné definovat šablony postupu, pomocí nichž je možné jednoduše rozdělit úkony členům týmu. Šablony jsou použity na přijatá hlášení pro tvorbu událostí. Do událostí je možné spojit více souvisejících hlášení. Díky integraci systému Cortex je usnadněna automatická analýza a odpověď na incidenty. Propojení se systémem MISP pak umožní sdílet detekované IoC a zároveň využít sdílené IoC jiných organizací k detekci či korelaci ve vlastní síti. V základu je definováno několik datových typů „observables“, tedy IoC, je však možné definovat i vlastní. Se systémem je možné pracovat pomocí webového rozhraní. Přístup je umožněn pomocí thehive4py API v jazyce Python nebo přímo pomocí Curl. Velkou výhodou je aktivní komunita a pravidelné aktualizace.

6.2 Výběr systémů

Po zhodnocení dostupných možností byl vybrán pro následnou realizaci práce projekt TheHive. Konkrétně verze systému označená jako TheHive4. Hlavními kritérii výběru byla možnost tvorby vlastních indikátorů a pevné propojení se systémy MISP a Cortex. Díky možnosti vytvoření vlastních indikátorů mohou být definovány datové typy pro jednotlivá pole PLOAM zpráv. Výhodou propojení se systémem MISP je jednoduché sdílení IoC, jak interně, tak i mezi dalšími bezpečnostními týmy. Systém je aktuálně zejména v Evropě hojně využíván řadou organizací a budoucí umožnění hlášení incidentů vládnímu CERT České republiky pomocí MISP představuje zajímavé možnosti.

6.2.1 Návrh testovací sítě

Pro otestování možnosti hlášení bezpečnostních incidentů z GPON sítě byla navržena jednoduchá testovací síť, viz obr. 6.1.



Obr. 6.1: Topologie testovací sítě.

Na základě předchozích testů byla klasická databáze, do které byly ukládány zachycená data z FPGA analyzátoru, nahrazena systémem Apache Kafka. Samotná aplikace je pak spuštěna na stejném serveru jako systémy TheHive, Cortex a MISP.

6.3 Návrh implementace testovací sítě

Pro snadné nasazení systémů byla zvolena implementace pomocí Docker kontejnerů. Samotná aplikace pro tvorbu hlášení v systému TheHive je poté rovněž nasazena v rámci Docker kontejneru. Hlášení jsou vyvolávána na základě detekce bezpečnostních událostí pomocí analyzátoru na FPGA kartě. Aplikaci jsou tato data předávána pomocí systému Apache Kafka. Správa systému Apache Kafka není součástí této práce, aplikace však využívá metody pro zpracování zpráv a je na systém napojena.

6.3.1 Využité nástroje

Apache Kafka

Apache Kafka je open-source distribuovaná streamovací platforma, neboli „message broker“. Systém je založen na principu server–klient. Server je zapojen v rámci klastru jednoho nebo více instancí brokerů, které mohou být rozprostřeny v rámci několika datových center. Klienti jsou speciálně rozděleni do dvou funkcí producenta a konzumenta. Producenti produkují zprávy, které jsou ukládány na server v rámci témat, neboli /uvtopics. Konzumenti poté mohou ze serveru tyto zprávy

číst a dále zpracovávat. Oproti klasické frontě zpráv zde tyto zprávy přečtením nezanikají a je tak možné k nim přistupovat i zpětně. Oba klienti jsou na sobě plně nezávislí, a producenti tak nemusejí čekat na konzumenty. Jednotlivé témata na serveru pak podporují přístup více producentů i konzumentů zároveň [56].

Docker

Docker je open-source platforma umožňující vývoj, přenos a spouštění aplikací. Poskytuje možnost zabalit a spustit aplikace ve volně izolovaném prostředí zvaném kontejnery. Oproti klasické virtualizaci obsahují kontejnery aplikace a všechny jejich závislosti, ale sdílejí jádro hostitelského operačního systému. Kontejnery tak vyžadují mnohem méně prostředků, jelikož nepotřebují virtualizaci kompletního operačního systému. Aplikace je pak snadné přenášet a spouštět v jakémkoli podporovaném Docker prostředí [57].

6.3.2 Nasazení systémů

Systémy TheHive, Cortex a MISP byly na server implementovány v rámci Docker kontejnerů. Každý ze systémů vyžaduje pro svou funkci různý databázový systém a manuální instalace kompletu by v případě potřeby byla značně zdlouhavá. Z tohoto důvodu byl vytvořen jednoduchý gitlab repositář obsahující potřebné konfigurační soubory pro TheHive a Cortex a soubor *docker-compose.yaml* viz obr. 6.2. Navržená implementace přebírá konfigurační soubory z šablony projektu [58].

```
TheHive_Stack
├── TheHive
│   └── application.conf
├── Cortex
│   └── application.conf
└── docker-compose.yaml
```

Obr. 6.2: Adresářová struktura repositáře pro nasazení kompletu systémů.

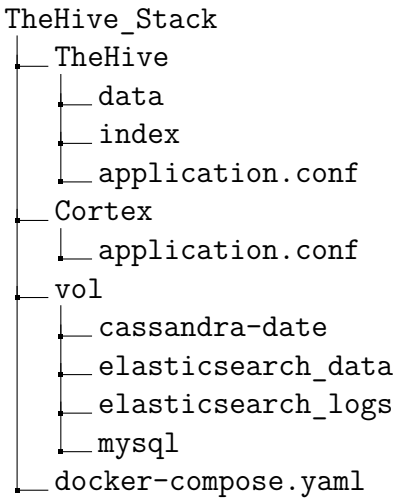
Díky nástroji Docker Compose je možné automaticky spustit několik kontejnerů najednou v rámci jednotné sítě (stacku). V rámci souboru docker-compose.yaml je možné rovněž definovat dodatečnou konfiguraci spuštěných kontejnerů. Bez dodatečné konfigurace ukládají aplikace spuštěné v prostředí kontejneru data pouze v rámci virtuálního prostředí. Data tak nejsou perzistentní a v případě výpadku kontejneru jsou smazána. Pro zachování persistence dat je pomocí souboru *docker-compose.yaml* namapováno lokální úložiště pro jednotlivé systémy.

6.3.3 Spuštění kompletu systémů

Po nakopírování repositáře a přesunutí se do hlavní složky je celý komplet spuštěn pomocí příkazu:

```
$ docker-compose up
```

Přepínač *–d* umožňuje spuštění na pozadí a potlačení debug zpráv. V případě prvního spuštění je automaticky vytvořena adresářová struktura viz obr. 6.3.



Obr. 6.3: Adresářová struktura kompletu systémů.

Díky definovanému mapování jsou data ukládána i lokálně v rámci tohoto adresáře. V případě migrace kompletu systémů nebo pro vytvoření zálohy je možné celou složku *TheHive_Stack* archivovat a exportovat. Po přesunu na nový systém nebo pro obnovení zálohy stačí archiv dekomprimovat a opětovným spuštěním pomocí příkazu *docker-compose up* budou data a konfigurace automaticky nahrána.

6.3.4 Správa kompletu systémů

Systémy jsou po spuštění dostupné na definovaných portech:

- TheHive: 9000
- Cortex: 9001
- MISP: 80 a 443

Porty spolu s adresami, na kterých budou systémy dostupné je možné změnit modifikací souboru *docker-compose.yaml* v rámci sekce *ports*. V případě potřeby je pomocí souboru *docker-compose.yaml* možné také definovat například systémová omezení využití paměťových či procesorových prostředků.

Webové rozhraní jednotlivých systémů je následně dostupné zadáním příslušné adresy a portu v rámci webového prohlížeče. Pro přístup k systémům jsou v čisté instalaci definovány následující přihlašovací údaje:

System	Login	Heslo
TheHive	admin	secret
MISP	admin@admin.test	admin

Pokud byl komplet spuštěn poprvé je nutné pro propojení systému TheHive spolu se systémy Cortex a MISP doplnit příslušné API klíče do konfiguračního souboru application.conf ve složce TheHive.

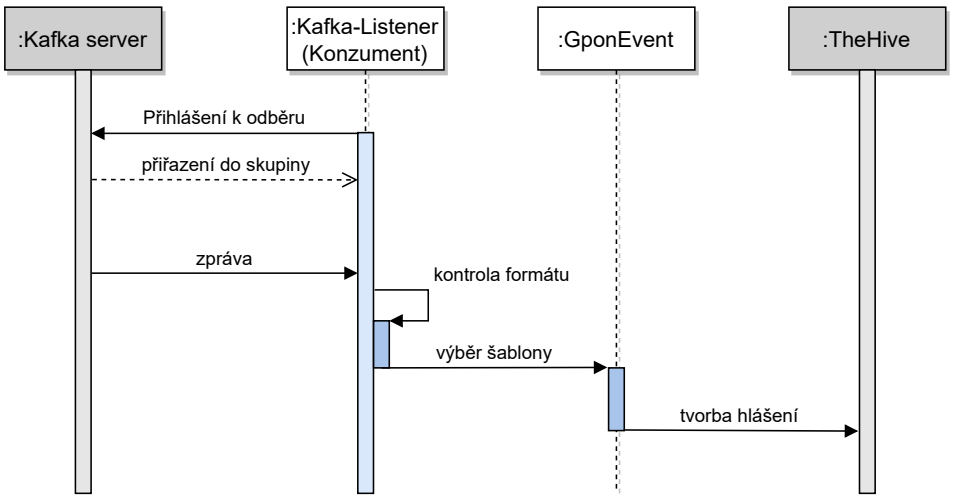
Při prvním přístupu k systému Cortex je nutné odkliknout tlačítko „update database“. Následně je zobrazena výzva k vytvoření účtu super-administrátora. Pomocí tohoto účtu je pak umožněna tvorba organizací a jednotlivých uživatelů. Při vytvoření nového uživatele je možné přiřadit heslo a vygenerovat API klíč, který je nutné přidat do konfiguračního souboru systému TheHive.

V rámci prvního přístupu k systému MISP je možné se přihlásit základními přihlašovacími údaji. Po přihlášení je vyžadována změna hesla. Příslušný API klíč pro integraci se systémem TheHive je pak dostupný pod záložkou *Automation page*.

7 Vlastní aplikace

Tato sekce se věnuje popisu návrhu vytvořené vlastní aplikace, umožňující tvorbu hlášení v systému TheHive. Aplikace se skládá ze dvou hlavních modulů, pomocného core modulu, konfiguračního souboru a souborů umožňující nasazení do Dockeru. Hlavními moduly jsou Kafka klient v roli konzumenta (*KafkaEventListener*) a modul definující třídu šablony hlášení (*GponEvent*). Pomocný core modul pak definuje datové třídy a formátování informačních a debug výpisů (*Logger*).

Základní princip je znázorněn viz obr.7.1. Nejdříve je navázáno spojení se serverem Apache Kafka. Aplikace se jako klient v roli konzumenta přihlašuje k odběru zpráv z daného tématu. Následně jsou přijímány zprávy obsahující data popisující detekované bezpečnostní události. Po obdržení zprávy je nejprve zkontrolován její formát. V případě správnosti formátu je na základě typu zprávy vytvořeno hlášení podle definovaných šablon. Hlášení je vyvoláno v systému TheHive, kde může být analyzováno a podle uvážení operátora mohou být provedeny další akce.

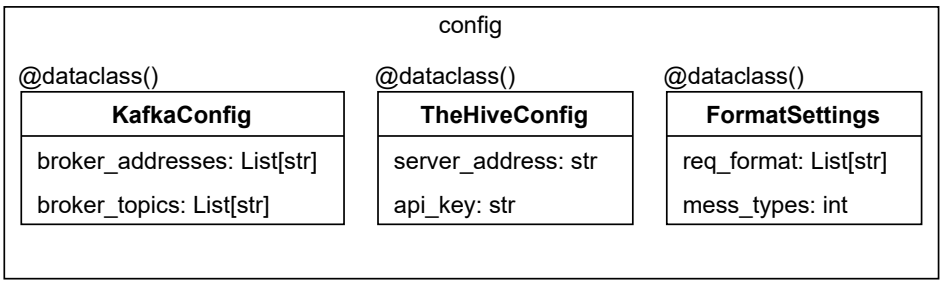


Obr. 7.1: Sekvenční diagram aplikace.

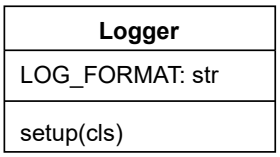
7.1 Pomocný core modul

Pomocný core modul deklaruje v rámci souboru *config.py* datové třídy pro konfigurační soubor. V rámci těchto datových tříd jsou nastaveny jednotlivé atributy nastavení Kafky, systému TheHive a formátu zpráv viz obr. 7.2.

Třída Logger slouží k definici formátu informačních a debug výpisů viz obr. 7.3. Je využito knihovny *logging* pro zvýšení informační hodnoty výpisů oproti klasické funkci *print*.



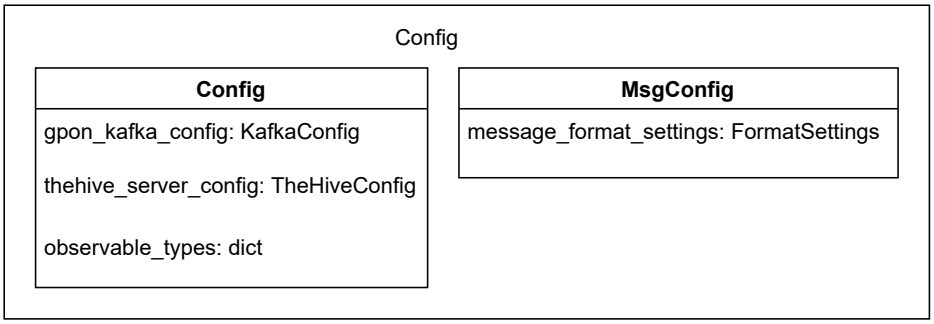
Obr. 7.2: Třídní diagram config.py.



Obr. 7.3: Třídní diagram třídy Logger.

7.2 Konfigurační soubor

Konfigurační soubor *Config.py* slouží pro nastavení parametrů aplikace. Je využito pomocného *core* modulu a deklarovaných datových tříd. Je zde definováno nastavení Kafka serveru, obecného formátu zpráv Kafky a systému TheHive viz obr. 7.4.



Obr. 7.4: Třídní diagram konfiguračního souboru Config.py.

Pro Kafku je zde nutné nastavit IP adresu a port, na kterém je server dostupný a téma, ze kterého mají být přijímány zprávy.

Datová třída *MsgConfig* definuje obecný formát příchozí zprávy z Kafky, tedy požadovaná pole a počet definovaných šablon.

Nastavení systému TheHive je nutné doplnit o IP adresu a port, na které je systém dostupný a API klíč uživatele. Dalším nastavením pro systém TheHive je slovník *observable_types* viz výpis 7.1. Tento slovník svazuje typ hlášení (z definovaných šablon GponEvent) s typem indikátoru kompromitace. Pro každý typ hlášení je zde nutné definovat datový typ, pod kterým budou do hlášení vloženy data události. V případě použití vlastně definovaných datových typů indikátorů je potřeba je nejdříve nadefinovat také přímo v prostředí TheHive.

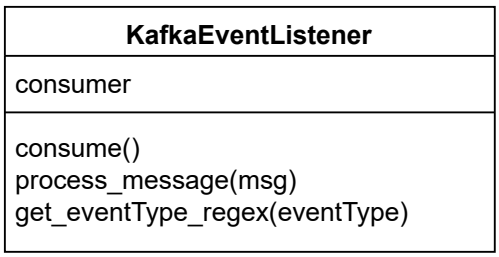
```
        ...
observable_types = {
    "1": "PLOAMd",
    "2": "GPON-Activation",
        ...
```

Výpis 7.1: Definice datových typů indikátorů pro jednotlivé typy hlášení.

Soubor je nutné před nasazením aplikace modifikovat doplněním údajů o Kafka serveru a systému TheHive.

7.3 Modul KafkaEventlistener

Modul KafkaEventListener definuje asynchronního Kafka klienta v roli konzumenta a metody pro zpracování přijatých zpráv viz obr. 7.5. K definici klienta byla využita knihovna *aiokafka* [59]. Knihovna poskytuje tvorbu asynchronního konzumenta, který umožňuje více konzumentům rozložení zátěže témat a umožňuje efektivnější zpracování přijatých zpráv.



Obr. 7.5: Třídní diagram modulu KafkaEventListener.

7.3.1 Inicializace konzumenta

Konstruktor třídy umožňuje inicializovat konzumenta viz výpis 7.2. Na vstupu metody jsou očekávány atributy IP adresa Kafka serveru (*boostrap_servers*), seznam témat (*topics*) a id skupiny (*group_id*).

```
        ...
    self._consumer: "AIOKafkaConsumer" = AIOKafkaConsumer(
        *topics,
        loop=loop,
        bootstrap_servers=bootstrap_servers,
        group_id=group_id,
    )
        ...
```

Výpis 7.2: Konstruktor modulu KafkaEventListener.

7.3.2 Přihlášení k odběru zpráv

Pro přihlášení konzumenta k odběru zpráv z Kafka serveru slouží metoda *consume()*. Pomocí metody konzument naváže spojení s Kafka serverem a v případě úspěšného přihlášení k odběrové skupině začíná přijímat zprávy. Přijaté zprávy jsou následně zpracovávány pomocí metody *process_message(msg)*.

```
        ...
async def consume(self):
    try:
        logging.debug("Listener starting")
        await self._consumer.start()
        logging.debug("Starting consuming")
        async for msg in self._consumer:
            await self.process_message(msg)
    finally:
        await self._consumer.stop()
        ...
```

Výpis 7.3: Metoda consume() modulu KafkaEventListener.

7.3.3 Zpracování přijatých zpráv

Přijaté zprávy jsou zpracovávány metodou *process_message(msg)* viz výpis 7.4. Příchozí zprávy jsou ve formátu JSON, proto jsou nejdříve načteny do datového typu

slovníku. Tímto je umožněna práce s jednotlivými poli původní zprávy.

Následně je ověřeno přijetí validní zprávy korelací obsažených polí s definovanými požadovanými poli (*required_fields*). V průběhu tvorby aplikace bylo přistoupeno k obecnému formátu těchto zpráv. Jednotlivá požadovaná pole jsou definována v konfiguračním souboru Config.py. Navržený obecný formát zprávy obsahuje typ události (EventType), čítač super-rámce (SuperFrameCounter) a samotná data popisující událost (IncidentData). Obsah pole IncidentData se liší pro každý typ události a umožňuje přenos libovolného počtu indikátorů kompromitace ve formátu JSON. K obecnému formátu bylo přistoupeno pro snížení režie na straně producenta.

```
        ...
async def process_message(self, msg) -> None:
    ...
    if all(key in required_fields for key in val):
        if re.search(match, val['EventType']) is None:
            logging.warning('Non specified EventType ID, no
                alert created')
        else:
            new_alert = GponEvent(val['EventType'], val["
                IncidentData"], val["SuperFrameCounter"],
                reference)
            new_alert.createAlert()
            logging.debug("New Alert Created")
    else:
        logging.warning('Received message is missing required
            fields, no alert created')
        ...
```

Výpis 7.4: Metoda process_message(msg).

Pokud je formát zprávy validní, je ověřena existence šablony hlášení pro daný typ události. Ověření je provedeno jednoduchou korelační funkcí *re.search()* s využitím balíčku *re*. Regulární výraz pro vstup do této funkce je vytvořen metodou *get_eventType_regex(eventType)* viz výpis 7.5. Výsledný výraz hledá řetězec o jedné číslici v rozsahu od 1 do počtu definovaných šablon hlášení. Výčet dostupných šablon hlášení spolu s jejich počtem je definován konfiguračním souboru Config.py.

```
        ...
def get_eventType_regex(eventType: int) -> str:

    match = "^[1-" + str(eventType) + "]$"

    return match
        ...
```

Výpis 7.5: Metoda pro tvorbu regulárního výrazu.

Je-li v modulu *GponEvent* definovaná šablona pro přijatý typ hlášení je inicializován objekt šablony *GponEvent*. Šabloně je předán obsah jednotlivých polí přijaté zprávy a reference zdroje. Referenci tvoří kombinace údajů o původu přijaté zprávy z Kafka serveru, tedy téma, oddíl a offset viz výpis 7.6.

```
        ...
reference = msg.topic + ":" + str(msg.partition) + ":" + str(
    msg.offset)
        ...
```

Výpis 7.6: Tvorba hodnoty reference pro hlášení.

Nakonec je na základě vyplněné šablony vytvořeno hlášení v systému TheHive zavoláním funkce *createAlert()*.

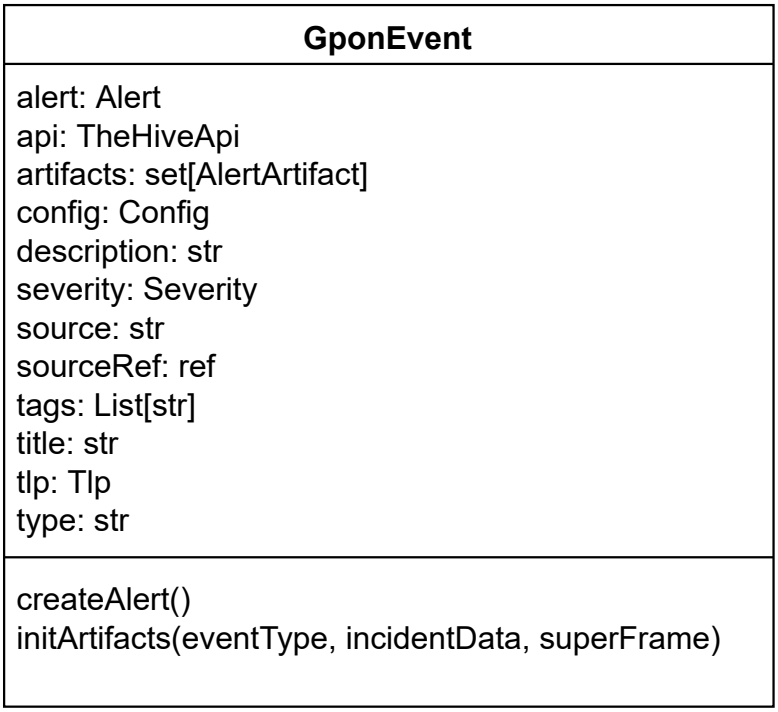
Chybí-li ve zprávě některé z povinných polí nebo není-li dostupná šablona pro daný typ hlášení (nedefinovaná hodnota pole *eventType*) je zobrazena chybová hláška s varováním, není vytvořeno žádné hlášení a aplikace pokračuje v příjmu zpráv.

7.4 Modul šablony GponEvent

Modul GponEvent slouží jako šablona pro tvorbu hlášení. Třída definuje všechny povinné atributy a metody pro vytvoření hlášení viz diagram 7.6.

V rámci šablony je pro komunikaci se systémem TheHive a možnost tvorby hlášení využita knihovna *thehive4py* [60]. Tato knihovna poskytuje přístup k API rozhraní systému TheHive a metody ke správě hlášení. Do šablony jsou importovány pouze vybrané použité metody viz výpis 7.7.

Pro navázání spojení se systémem TheHive je definován objekt API rozhraní viz výpis 7.8. Nastavení je ve formátu *TheHiveApi('IP:port', 'API klíč')*. Objekt rozhraní je doplněn potřebným nastavením z konfiguračního souboru a není tak potřeba modifikovat šablonu.



Obr. 7.6: Třídní diagram modulu GponEvent.

```
            ...
from thehive4py.api import TheHiveApi
from thehive4py.models import Alert, AlertArtifact, Severity, Tlp
            ...
```

Výpis 7.7: Import funkcí knihovny thehive4py.

```
            ...
    self.api = TheHiveApi(
          self.config.thehive_server_config.server_address,
          self.config.thehive_server_config.api_key)
            ...
```

Výpis 7.8: Definice API rozhraní pro systém TheHive.

Pomocí definovaného API rozhraní je možné vytvářet hlášení. Tato hlášení musí obsahovat všechny následující povinné atributy, aby bylo jejich vytvoření v systému TheHive úspěšné [61]:

title (text) : titulek hlášení,

description (text) : popis hlášení,

severity (number) : závažnost (1: nízká, 2: střední, 3: vysoká), výchozí hodnota = 2,

date (date) : datum a čas vyvolání hlášení, výchozí hodnota = now,

tags (multi-string) : štítky, výchozí hodnota=prázdné,

tlp (number) : TLP (TRAFFIC LIGHT PROTOCOL), (0: white; 1: green; 2: amber; 3: red) výchozí hodnota = 2,

status (AlertStatus) : status hlášení (New, Updated, Ignored, Imported), výchozí hodnota = New,

type (string) : typ hlášení (internal, external, human),

source (string) : zdroj hlášení,

sourceRef (string) : reference hlášení, např. ID alramu SIEM systému,

artifacts (multi-artifact) : seznam indikátorů nesoucí jednotlivé atributy.

Pro každé hlášení musí být kombinace atributů type, source a sourceRef jedinečná. Pokud již v systému existuje hlášení se stejnou kombinací těchto tří atributů, je vytvoření nového hlášení zamítnuto. Definovaná hlášení nesou externí typ, jelikož pocházejí z externího zdroje (source) Gpon_Analyzer, tedy FPGA karty. Jako reference je použita kombinace hodnot popisující původ obdržené zprávy z Kafka serveru. Tuto hodnotu předává definovaný konzument viz kapitola 7.3.3.

Jednotlivé atributy jsou definovány pomocí konstruktoru na jehož vstupu jsou očekávány parametry eventType, incidentData, superFrameCounter a reference. Některé z atributů jsou nastaveny obecně pro všechny typy hlášení, některé jsou definovány specificky pro daný typ hlášení na základě přijaté hodnoty *eventType*.

7.4.1 Návrh typů hlášení

Šablona definuje několik typů hlášení. Každému typu je přiřazena číselná hodnota *eventType*. Podle obdržené hodnoty z přijaté zprávy jsou následně naplněny některé z atributů specifickými hodnotami pro daný typ hlášení viz výpis 7.9.

```
        ...
    if eventType == '1': # PLOAM anomaly
        self.title = 'PLOAMd Anomaly'
        self.description = 'Analyzer ... '
        addTags = ["PLAOMd", "Non-standard formatting", "
            Suspicious activity"]
        self.tags.extend(addTags)

    elif eventType == '2': # Activation process anomaly
        self.title = 'Activation process anomaly'
        self.description = 'Analyzer ... '
        self.severity = Severity.LOW.value
        addTags = ['Activation process', 'PLOAM']
        self.tags.extend(addTags)
        ...
```

Výpis 7.9: Definice specifických atributů pro jednotlivé typy hlášení.

Vlastní aplikace definuje 5 základních typů hlášení:

- **PLOAMd Anomaly** – oznámení detekce anomálie v PLOAMd zprávě, tedy odchýlení od standardem specifikovaných zpráv a jejich obsahu,
- **Activation process anomaly** – oznámení detekce anomálie v aktivačním procesu
- **Non-standard Frame structure** – oznámení detekce anomálie ve formátu GTC rámce
- **OMCI Anomaly** – oznámení detekce anomálie ve formátu OMCI (ONU Management and Control Interface) zprávy nebo aktivačním procesu OMCC (ONU Management and Control Channel) kanálu
- **Non-specified error** – prázdné hlášení, slouží jako šablona pro specifikaci nového typu hlášení

Šablonu hlášení je díky definovanému prázdnému hlášení možné snadno rozšířit a doplnit o nové typy hlášení. V novém hlášení lze předefinovat i obecně definované atributy. Po přidání nového typu hlášení je následně rovněž nutné upravit konfigurační soubor. Do slovníku *observable_types* musí být přidán záznam s označením nové hodnoty *eventType* spojen s požadovaným datovým typem indikátoru kompromitace v hlášení.

7.4.2 Definice indikátorů kompromitace

Pro naplnění atributu *artifacts* slouží metoda *initArtifacts()* viz výpis 7.10. Aplikace přidává do hlášení dva indikátory a to detekována data a hodnotu super rámce.

```
        ...
def initArtifacts(self, eventType, incidentData, superFrame):
    d = json.dumps(incidentData)
    self.artifacts = {
        AlertArtifact(dataType=self.config.observable_types[
            eventType], data=d),
        AlertArtifact(dataType="superframe-counter", data=
            superFrame),
    }
        ...
```

Výpis 7.10: Metoda pro inicializaci indikátorů kompromitace initArtifacts().

Je využito metody *AlertArtifact(datatype, data)* z knihovny *thehive4py*. Parametr datatype na vstupu metody *AlertArtifact()* určuje datový typ indikátoru v prostředí TheHive. Pro potřebu vlastní aplikace bylo využito vlastně definovaných datových typů. Výčet definovaných datových typů je uveden v konfiguračním souboru. Nejedná se o datové typy na úrovni programovacího jazyka, jedná se o označení indikátorů v prostředí TheHive pro potřeby další analýzy. Obecně mohou být předány pouze data ve formátu *string*. Je však umožněno předat i více hodnot v rámci formátu JSON. Bylo tedy přistoupeno k definici obecných datových typů pro každý typ hlášení. Jako data je následně předán soubor detekovaných indikátorů ve formátu JSON. Všechny data pro naplnění indikátorů kompromitace jsou šabloně předány definovaným konzumentem z obdržené zprávy.

7.4.3 Vytvoření hlášení

Hlášení jsou tvořena pomocí metody *createAlert* viz výpis 7.12. Zavoláním metody je objekt hlášení typu *Alert* naplněn definovanými atributy. Samotné vytvoření poté realizuje API rozhraní pomocí vlastní metody *api.create_alert()*, které je předán objekt hlášení.

7.5 Hlavní spustitelný soubor

Hlavní main soubor slouží k samotnému spuštění aplikace. Je definována asynchronní *main()* funkce, která načítá konfigurační soubor a nastavení formátu informačních

```
            ...
    def createAlert(self):
        self.alert = Alert(title=self.title,
                           description=self.description,
                            ...
                           artifacts=self.artifacts)
        response = self.api.create_alert(self.alert)
            ...
```

Výpis 7.11: Metoda pro vytvoření hlášení createAlert().

výpisů *Logger*. Dále je zde inicializován klient v roli konzumenta, adresa serveru a téma jsou převzaty z konfiguračního souboru. Pole *group_id* je definováno specificky. V případě inicializace více konzumentů musí být tato hodnota jedinečná. Konzument je následně spuštěn pomocí *await listener.consume()*.

Aplikace využívá asynchronního klienta Kafky. Hlavní main metoda tedy spouští definovanou *main()* funkci v asynchronní smyčce událostí viz výpis 7.12. Použití asynchronní smyčky a funkcí umožňuje využitá knihovna *asyncio*.

```
            ...
if __name__ == '__main__':
    loop = asyncio.get_event_loop()
    loop.run_until_complete(main())
    exit()
            ...
```

Výpis 7.12: Metoda pro vytvoření hlášení createAlert().

7.6 Příprava pro spuštění v Docker

Vlastní aplikaci je možné spustit ručně pomocí nástroje poetry viz následující sled příkazů:

```
$ apt update && apt install -y python3-venv
$ pip3 install poetry
$ poetry install
$ poetry run python3 src/theHive_alert_handler.py
```

Výpis 7.13: Manuální spuštění vlastní aplikace.

Aplikaci je doporučeno spustit v rámci kontejneru pomocí Docker. Pro spuštění v Docker bylo připraveno několik souborů.

- **Dockerfile**
  – soubor obsahuje nastavení a příkazy pro tvorbu obrazu aplikace. Jsou zde namapovány zdrojové soubory a pomocí nástroje *poetry* jsou nainstalovány závislosti ze souboru *pyproject.toml*.
- **pyproject.toml**
  – soubor definuje využitou verzi python a závislosti vlastní aplikace. Jedná se zejména o knihovny, které je nutné pro správnou funkci aplikace doinstalovat jako *thehive4py* nebo *aiokafka*.
- **Makefile**
  – soubor slouží k vytvoření docker obrazu aplikace pomocí definovaného příkazu.
- **docker-compose.yaml**
  – soubor slouží k vytvoření a spuštění kontejneru vlastní aplikace.

Pro jednoduché nasazení aplikace byl vytvořen gitlab repositář. Po naklonování adresáře je nejprve nutné upravit konfigurační soubor *Config.py*. Následně je aplikaci možné spustit v rámci Docker kontejneru pomocí sledu příkazů viz výpis 7.14.

```
$ cd path/to/gpon-incident-handling
$ make docker
$ docker-compose up
```

Výpis 7.14: Spuštění vlastní aplikace pomocí Docker.

8 Testování funkcionality

Testování proběhlo implementací vybraných systémů a vlastní aplikace v rámci navrženého testovacího prostředí.

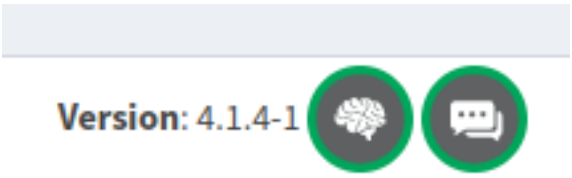
Sada vybraných systémů TheHive, Cortex a MISP byla implementována na virtuální stroj s operačním systémem Linux pomocí vytvořeného repositáře. Repositář byl na server naklonován a celý komplet byl následně spuštěn pomocí příkazu *docker-compose up*. Nástroj automaticky stáhne definované obrazy jednotlivých systémů, vytvoří adresářovou strukturu a spustí jednotlivé kontejnery.

Systém TheHive je dostupný ve dvou hlavních verzích TheHive a TheHive4. Pro testování byla využita verze TheHive4, konkrétně 4.1.4-1. Tato verze se liší oproti klasické řadě širšími možnostmi konfigurace a poskytuje možnosti definice vlastních datových typů indikátorů kompromitace.

Pro otestování byl v síti nakonfigurován Kafka server. Konfiguraci a správu serveru poskytl kolega Ing. Martin Holík. Na serveru byl následně nakonfigurován Kafka producent simulující detekované události pomocí souboru předem definovaných několika zpráv. Producent tak ve smyčce konstantně generoval definované zprávy. Tímto byla zajištěna simulace detekování bezpečnostní události v reálném prostředí.

8.1 Nastavení integrace

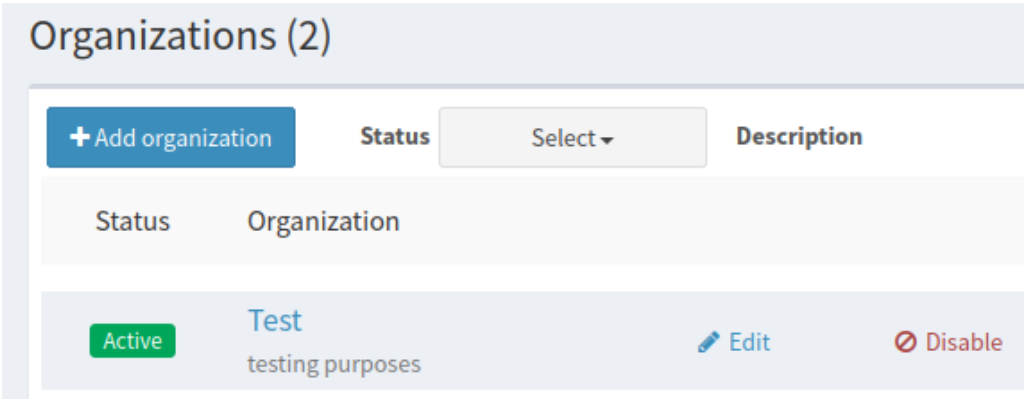
Pro otestování bylo nejdříve nutné provést inicializační konfiguraci. Po prvním spuštění je pro úspěšnou integraci systémů Cortex a MISP se systémem TheHive nutné doplnit konfigurační soubor *TheHive/application.conf* o API klíče obou systémů. Klíče lze získat prostřednictvím webového rozhraní jednotlivých systémů. Po úpravě konfiguračního souboru je potřeba restartovat kontejner systému TheHive. Úspěšná integrace je poté indikována v prostředí webového rozhraní systému TheHive zelenými ikonami v pravém dolním rohu viz obr. 8.1.



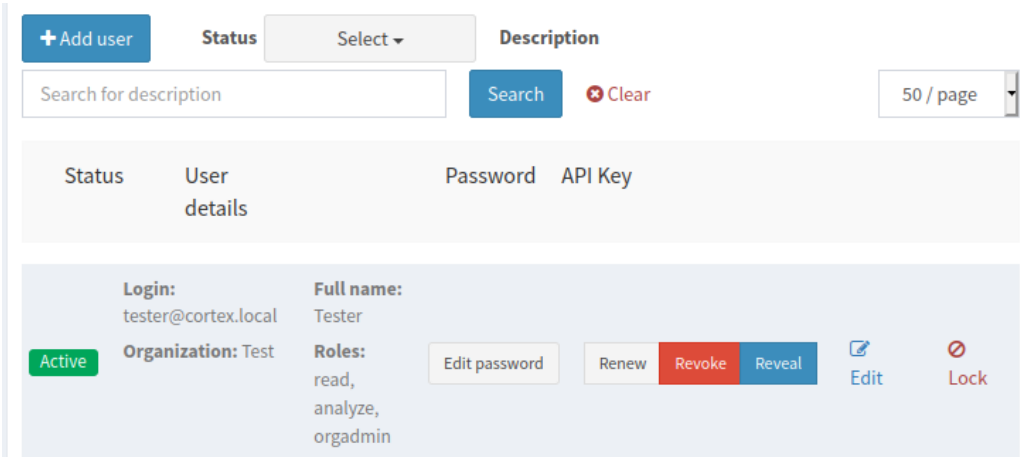
Obr. 8.1: Indikace úspěšné integrace Cortex a MISP.

8.1.1 Cortex API klíč

Webové rozhraní systému Cortex je dostupné skrze webový prohlížeč zadáním adresy *localhost:9001*. Po prvním spuštění je potřeba aktualizovat databázi stiskem tlačítka „update database“. Následně byl vytvořen super-administrátorský účet, který umožňuje vytvářet organizace, uživatele a celkovou správu systému. API klíč je generován pro konkrétní uživatele. Nejprve byla tedy vytvořena testovací organizace *Test* viz obr. 8.2, ve které byl vytvořen testovací uživatel *tester* viz obr. 8.3.



Obr. 8.2: Cortex: tvorba organizace.

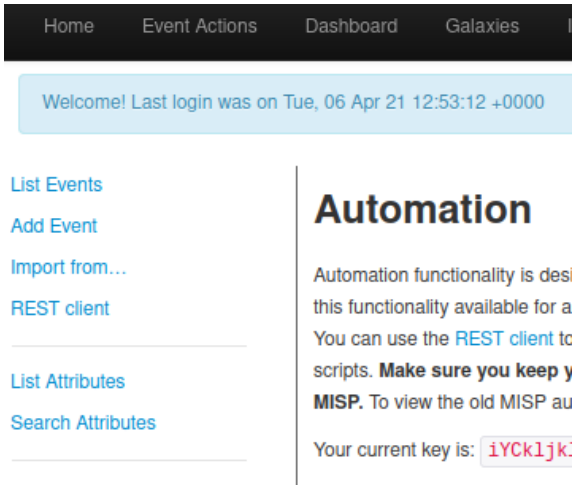


Obr. 8.3: Cortex: tvorba uživatele.

Uživateli je API klíč vygenerován pomocí tlačítka *Create API key*. Tento klíč je potřeba doplnit do konfiguračního souboru systému TheHive do sekce *cortex* k parametru *key*.

8.1.2 MISP API klíč

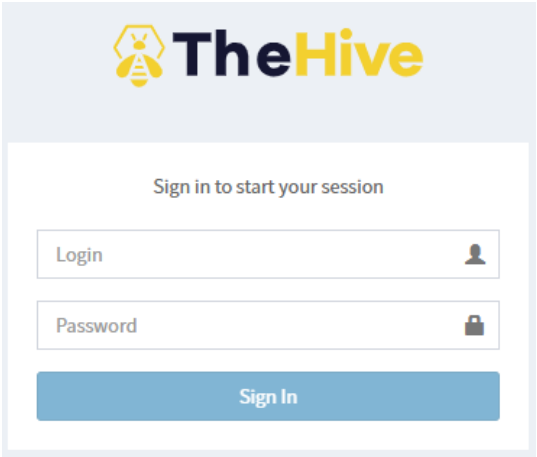
Webové rozhraní systému MISP je dostupné skrze webový prohlížeč zadáním adresy *localhost:80*. Po prvním přihlášení se základními přihlašovacími údaji je nutné změnit heslo. API klíč je následně možné nalézt pod záložkou *Event Actions -> Automation* viz obr. 8.4. Klíč je potřeba doplnit do konfiguračního souboru systému TheHive do sekce *MISP configuration*.



Obr. 8.4: MISP: API klíč.

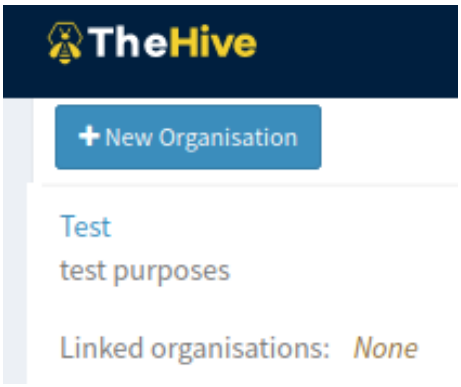
8.2 Vlastní konfigurace systému TheHive

Pro otestování vlastní aplikace bylo potřeba v systému TheHive provést základní konfiguraci. Webové rozhraní viz obr. 8.5 je dostupné skrze webový prohlížeč na adrese *localhost:9000*.



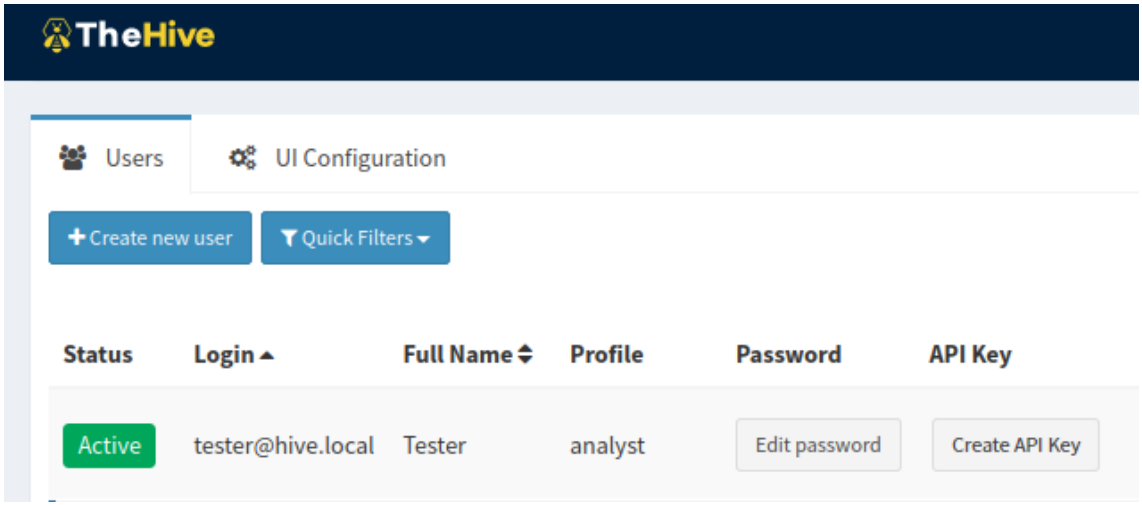
Obr. 8.5: Přihlašení k webovému rozhraní TheHive.

Při prvním spuštění je možné se přihlásit základními údaji *admin:secret*. Základní administrátorský účet má práva super-administrátora a umožňuje správu organizací, tvorbu nových uživatelů, profilů oprávnění a vlastních indikátorů. Po přihlášení byla vytvořena nová organizace s názvem „*Test*“ viz obr. 8.6. Novou organizaci je možné vytvořit pomocí *Admin* > *Organization* > *New Organization*.



Obr. 8.6: Tvorba nové organizace.

Následně byl v rámci nové organizace vytvořen nový uživatel „Tester“ s profilem *org-admin*. Přidělený profil dává uživateli administrátorská práva pro správu organizace. Uživatel tak může vytvářet hlášení, události, šablony, rozdělovat úkony a další. Vytvoření nového uživatele je možné po zvolení dané organizace pomocí *Create new user*, viz obr. 8.7.



Obr. 8.7: Vytvoření nového uživatele.

Nově vytvořenému uživateli je nutné přidělit nové heslo a vygenerovat nový API klíč, který umožňuje přístup k systému pomocí API rozhraní thehive4py. V seznamu

uživatelů je možné přidělit nebo změnit heslo konkrétnímu uživateli pomocí *Create new password/Edit password* a vygenerovat API klíč pomocí *Create API key*.

Nakonec byly v systému definovány vlastní datové typy indikátorů kompromitace. Tvorba vlastních datových typů indikátorů je umožněna pod základním administrátorským účtem v záložce *Admin* > *Observable types* > *Add data type*. Pro potřeby testování bylo vytvořeno 5 nových datových typů:

- PLOAMd
- GPON-Activation
- GTC_Frame
- OMCI
- superframe-counter

Systém samotný byl následně připraven na testování tvorby hlášení pomocí vlastní aplikace.

8.3 Simulace bezpečnostních událostí

Na dostupném Kafka serveru byl vytvořen klient v roli producenta, který pravidelně generoval předem definované zprávy. Správci serveru byl předán textový dokument s definicí 2 typů testovacích zpráv ve formátu JSON viz výpis 8.1. Tyto zprávy pak byly producentem zasílány na server pod testovacím tématem *GPONAnomalies*.

```
 ...
9:{"EventType":"1","SuperFrameCounter":"968107662","IncidentData"
   :{"OnuID":"255", "MessageID":"24", "Data":"71BC0290300"}},
10:{"EventType":"2","SuperFrameCounter":"968107662","IncidentData
   ":{"OnuID":"5","ActivationAnomyly":[ "4", "14", "18", "8", "8"
   , "8" ]}},
...
```

Výpis 8.1: Návrh formátu testovacích zpráv.

Vybrané testovací zprávy reprezentují zachycení události typu 1. tedy anomálie ve formátu PLOAMd zprávy a typu 2. tedy anomálie v pořadí aktivačního procesu. Obsah jednotlivých popisů událostí (*IncidentData*) byl založen na předem zachycených anomáliích v rámci projektu [43].

8.4 Testování vlastní aplikace

Na server byl naklonován repositář s vlastní aplikací a pomocí Kafka serveru, který simuloval detekci bezpečnostních událostí, byla otestována tvorba hlášení v TheHive.

8.4.1 Úprava konfigurace

Před samotným spuštěním aplikace bylo třeba upravit konfigurační soubor *Config.py*. V sekci nastavení Kafka serveru byla doplněna IP adresa na které byl server dostupný, ve formátu *"IP:PORT"* a název testovacího tématu. V sekci nastavení sytému TheHive byla rovněž doplněna IP adresa ve formátu *"http://IP:PORT"*, na které bylo dostupné API rozhraní systému TheHive a předem získaný API klíč nově vytvořeného uživatele. Dále bylo nutné doplnit slovník *observable_types* o nově definované vlastní datové typy indikátorů kompromitace pro jednotlivé typy hlášení.

8.4.2 Vytvořená hlášení

Po doplnění konfigurace byl vytvořen obraz a spuštěn kontejner. Aplikace se spojí s Kafka serverem a přihlásí se k odběru tématu simulujícího detekci bezpečnostních událostí. Následně začíná aplikace přijímat zprávy a vytvářet hlášení v systému TheHive.

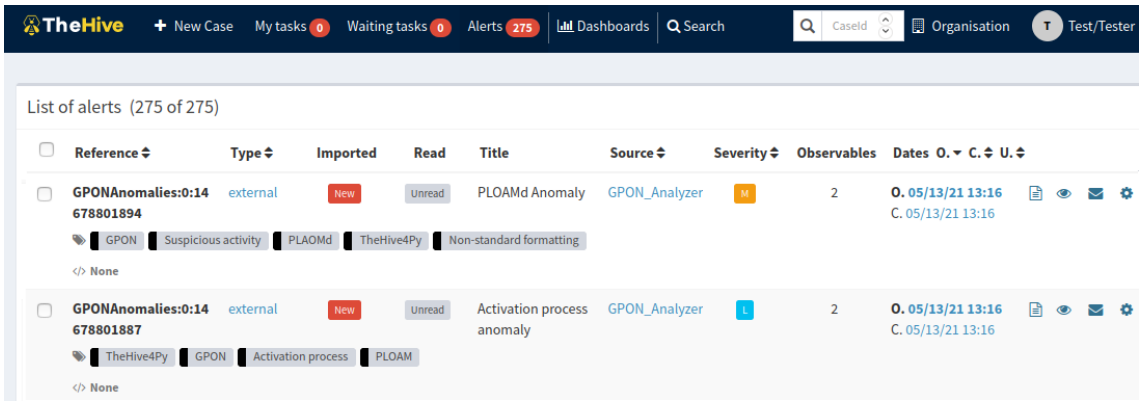
```
hive@hive-virtual-machine:/media/data-hdd/HiveStack/EventHandler$ sudo docker-compose up
[sudo] password for hive:
Starting gpon-event-handler ... done

Attaching to gpon-event-handler
gpon-event-handler    | 2021-05-13 11:15:54,206 - [1|MainThread] - INFO: Updating subscribed topics to: frozen
set({'GPONAnomalies'}) {subscription_state.py:107}
gpon-event-handler    | 2021-05-13 11:15:54,206 - [1|MainThread] - INFO: Consumer with group_id `test-Anomalie
s-group` initialized {Kafka_listener.py:29}
gpon-event-handler    | 2021-05-13 11:15:54,243 - [1|MainThread] - INFO: Discovered coordinator 0 for group te
st-Anomalies-group {group_coordinator.py:550}
gpon-event-handler    | 2021-05-13 11:15:54,243 - [1|MainThread] - INFO: Revoking previously assigned partitio
ns set() for group test-Anomalies-group {group_coordinator.py:384}
gpon-event-handler    | 2021-05-13 11:15:54,243 - [1|MainThread] - INFO: (Re-)joining group test-Anomalies-gro
up {group_coordinator.py:1196}
gpon-event-handler    | 2021-05-13 11:15:54,258 - [1|MainThread] - INFO: Joined group 'test-Anomalies-group' (
generation 1) with member_id aiokafka-0.7.0-b7b21c3a-16c4-4d20-bf4d-05375c5aece8 {group_coordinator.py:1251}
gpon-event-handler    | 2021-05-13 11:15:54,258 - [1|MainThread] - INFO: Elected group leader -- performing pa
rtition assignments using roundrobin {group_coordinator.py:1255}
gpon-event-handler    | 2021-05-13 11:15:54,262 - [1|MainThread] - INFO: Successfully synced group test-Anomal
ies-group with generation 1 {group_coordinator.py:1369}
gpon-event-handler    | 2021-05-13 11:15:54,262 - [1|MainThread] - INFO: Setting newly assigned partitions {To
picPartition(topic='GPONAnomalies', partition=0)} for group test-Anomalies-group {group_coordinator.py:464}
gpon-event-handler    | 2021-05-13 11:15:54,269 - [1|MainThread] - INFO: Creating Alert {Kafka_listener.py:68}
gpon-event-handler    | 2021-05-13 11:15:57,064 - [1|MainThread] - INFO: Creating Alert {Kafka_listener.py:68}
```

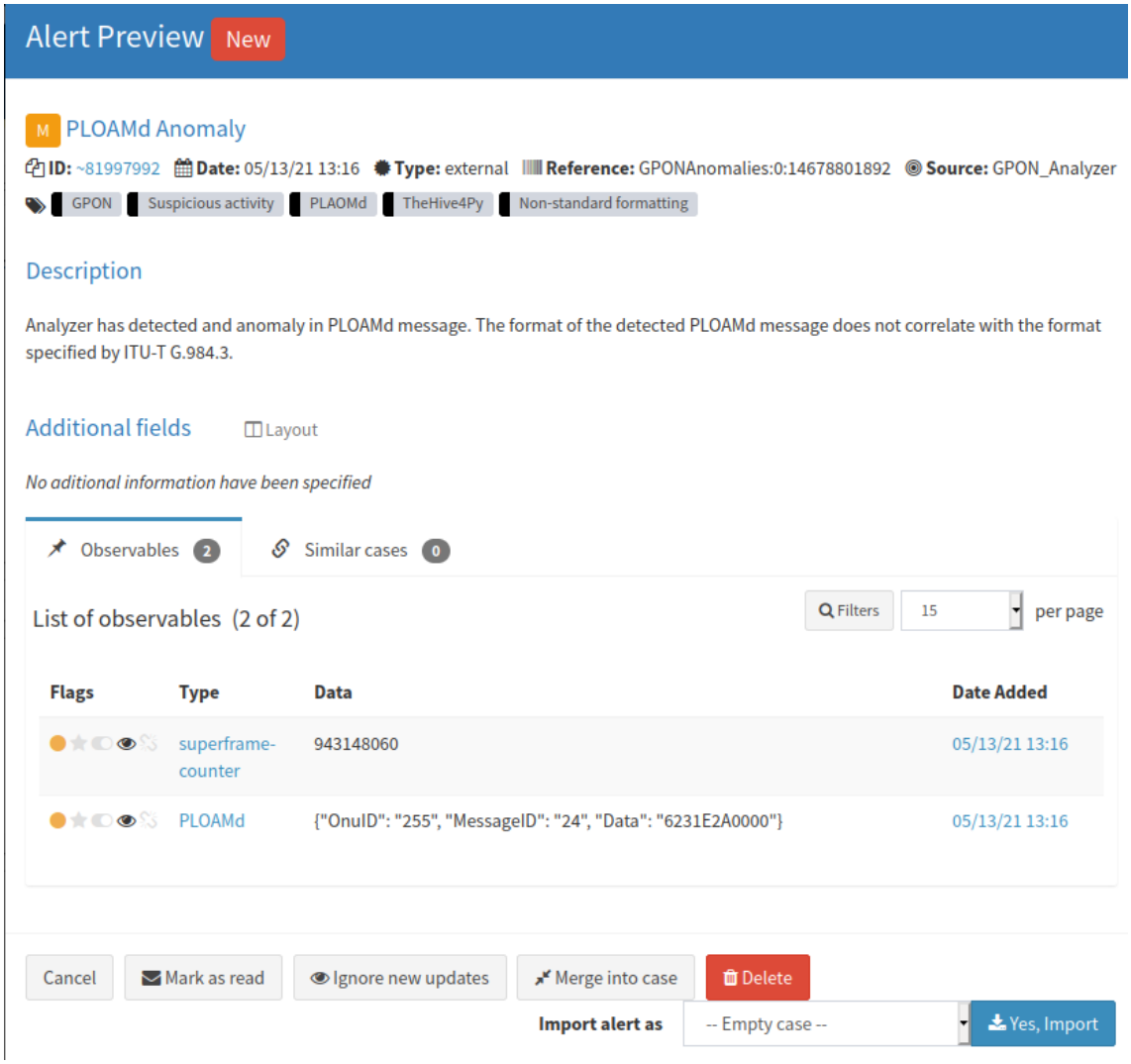
Obr. 8.8: Spuštění vlastní aplikace.

Vytvořená hlášení jsou skrze webové rozhraní systému TheHive dostupná uživateli *tester* pod záložkou *Alerts*. Obdržená hlášení jsou zobrazena v seznamu viz obr. 8.9. V seznamu je pak možné filtrovat hlášení podle zadaných kritérií. Hlášení je v seznamu možné také spravovat hromadně označením jednotlivých hlášení. Je tak možné je hromadně mazat, vytvářet události nebo sloučit více hlášení pod jednu událost.

Kliknutím na ikonu *Preview and Import* lze zobrazit obsah hlášení a podrobné informace včetně obsažených indikátorů, viz obr. 8.10.



Obr. 8.9: Seznam hlášení v systému TheHive.



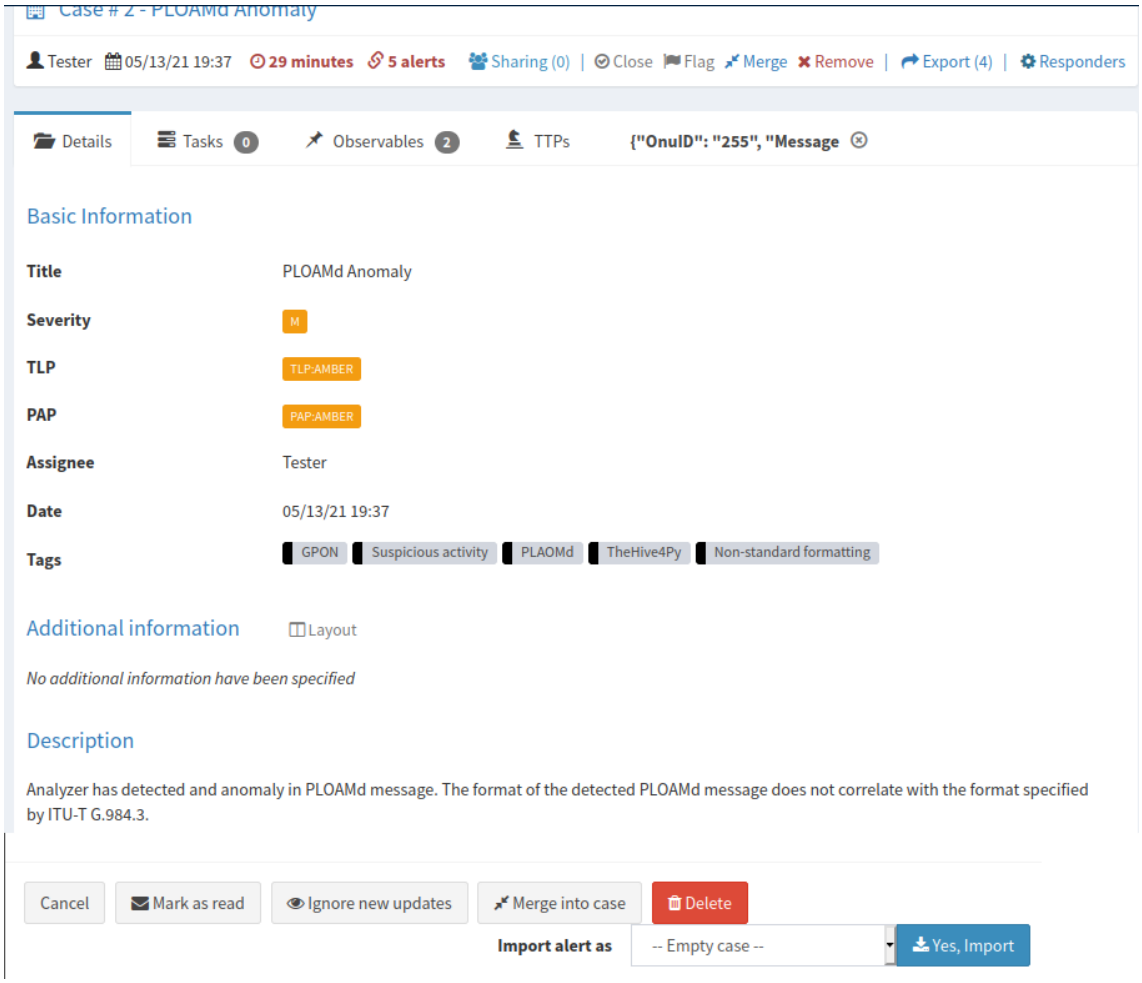
Obr. 8.10: Obsah hlášení v systému TheHive.

V podrobném přehledu hlášení je pak dostupný popis detekované události, zdroj hlášení a zejména pak indikátory kompromitace.

Na základě hlášení lze následně vytvořit událost (case) pomocí tlačítka *Import*. Pod událostí lze rozdělit jednotlivým členům úkony, provádět analýzu indikátorů, sdílet informace nebo spustit responder pro automatickou odpověď. Pokud mezi sebou souvisí více hlášení je možné spojit je pod jednu událost pomocí tlačítka *Merge into case*.

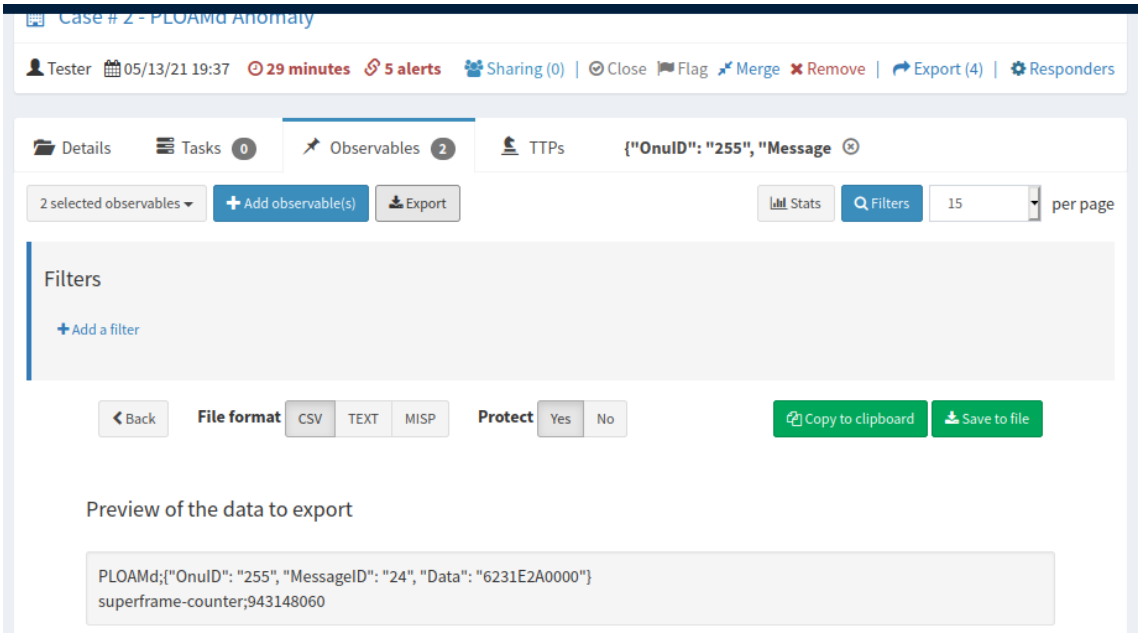
8.4.3 Tvorba události

Na vytvořené události z hlášení v systému TheHive viz obr. 8.11, pak mohou pracovat jednotliví členové týmu. Mimo další analýzy obsažených indikátorů a automatizace odpovědí je možné celou událost jednoduše exportovat do systému MISP pomocí tlačítka *Export*.



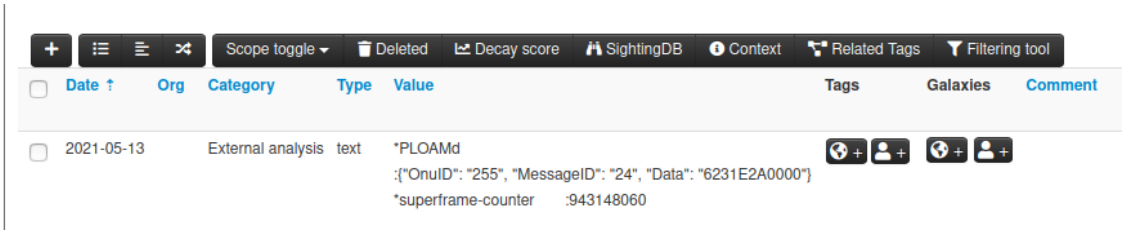
Obr. 8.11: Událost systému TheHive.

V sekci *Observables* lze obsah indikátorů rovněž exportovat do různých datových formátů viz obr. 8.12.



Obr. 8.12: Export zachycených indikátorů.

Při automatickém exportu události do systému MISP může nastat problém s vlastně definovanými datovými typy indikátorů. MISP totiž požaduje standardní datový formát pro indikátory kompromitace a specifikuje standardní kategorie. Je však možné indikátory manuálně vyexportovat pod MISP formátem a manuálně přidat k události v systému MISP pomocí *Add attribute* viz obr. 8.13. Jako typ pak zvolit například *text*.



Obr. 8.13: Přidání atributu v systému MISP.

8.4.4 Obsah navržených hlášení

Navržené typy hlášení jsou založeny na možných scénářích bezpečnostních událostí, které by mohly v GPON síti nastat. Jednotlivé události a obsah definovaných hlášení

vychází z teoretické části této práce a dříve zachycených anomálií. Hodnoty závažnosti a TLP byly navrženy prozatím spíše experimentálně. Po zvážení však mohou být jednoduše jednotlivé hodnoty atributů hlášení předefinovány.

Samotný obsah hlášení a vybrané indikátory kompromitace počítají s technickou znalostí a porozumění principů GPON sítí daného analytika, který bude se systémem pracovat a hlášení vyhodnocovat.

Vlastní aplikace nespecifikuje konkrétní obsah indikátorů. Tvorba hlášení je založena na obecném tvaru zprávy přenášející data, která popisují událost. Na straně producenta je tak možné předat aplikaci libovolné indikátory. Podmínkou je, aby byl obsah parametru přenášející tyto data (*IncidentData*) ve formátu JSON. Do hlášení jsou pak všechny tyto data vložena pod jeden datový typ definovaný pro dané hlášení. V případě potřeby je však možné šablonu modifikovat a přidat jednotlivé indikátory z parametru (*IncidentData*) do hlášení pod vlastními datovými typy.

V rámci testování a simulace bezpečnostních událostí byly navrženy možné indikátory a formát zpráv pro dva typy hlášení viz výpis 8.1. Definované indikátory nejsou pevné a jejich výčet je možné podle potřeby upravit.

8.5 Archivace

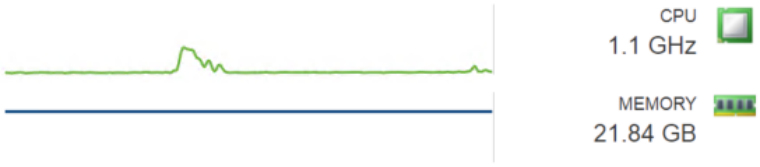
Po úspěšném otestování systémů, vlastní aplikace a tvorby hlášení byl komplet systémů s aktuální konfigurací archivován. Možnost archivace konfigurace a aktuálních dat byla otestována přenesením archivu na jiný systém. Na tomto systému byl archiv dekomprimován a soubor systémů spuštěn. Veškerá konfigurace a data byly zachovány. Přenos systémů byl úspěšný.

Takto vytvořená záloha otestované konfigurace byla přidána do repositáře s kompletem systémů.

8.6 Zhodnocení stability

V rámci testování nasazení jednotlivých systémů a vlastní aplikace byla rovněž posouzena stabilita implementace. Jednotlivé systémy byly dlouhodobě stabilní a bez výpadků. Vzhledem k tomu, že každý z použitých systémů využívá jiný databázový systém je celý komplet dosti paměťově náročný viz obr. 8.14. Komplet je tak nutné implementovat na výkonnější server.

Menší problémy mohou u nasazení systému TheHive pomocí Docker nastat při migraci dat mezi verzemi. Při migraci dat s přechodem na novou verzi je doporučeno



Obr. 8.14: Průměrné zdrojové a paměťové nároky kompletu systémů.

spuštění pomocí příkazu *docker-compose up* bez přepínače *-d* pro zobrazení možných chybových hlášení umožňující identifikaci problému.

Stabilita systému a aplikace byla rovněž testována i v případě vysokého množství událostí. Během testu byly na Kafka server periodicky produkovány zprávy simulující bezpečnostní události. Vlastní aplikace dokázala odebírat a zpracovávat zhruba 400 zpráv za minutu. Vytvořená hlášení v rámci zpracování zpráv byly ve webovém rozhraní systému TheHive zobrazeny v rámci sekund. Malé zpoždění vzniká díky obnovení seznamu po přijetí určitého množství hlášení. Obnovovací frekvence lze případně podle potřeby upravit. Takovéto množství bezpečnostních událostí v jeden okamžik však není obecně předpokládáno. Vlastní aplikace a zvolené systémy jsou nicméně tuto zátěž schopny zpracovat.

Závěr

Cílem této práce byl výběr vhodného open-source systému a návrh vlastní aplikace umožňující hlášení bezpečnostních událostí z GPON sítí. Teoretická část poskytuje nezbytný teoretický podklad k pochopení principů technologie GPON. Byly popsány jednotlivé komponenty GPON sítí, princip komunikace a přenášené zprávy. Pozornost pak byla věnována bezpečnosti technologie a aktuálním možným bezpečnostním hrozbám.

Součástí práce je také představení problematiky bezpečnostních incidentů, tak jak je definována v české právní úpravě. Část je zaměřena zejména na definici důležitých pojmů. Následně jsou představeny různé týmy působící v oblasti bezpečnostních incidentů a aktuální možnosti hlášení bezpečnostních incidentů.

K výběru vhodných systémů pro účely práce bylo třeba provést průzkum dostupných open-source systémů umožňující správu incidentů. V další části tak byly představeny různé nástroje a několik kandidátů k realizaci řešení.

V praktické části bylo provedeno porovnání specifikací jednotlivých nástrojů. Na základě porovnání byl vybrán soubor systémů TheHive, Cortex a MISP. Systémy byly vybrány zejména kvůli jejich vzájemné integraci a poskytovaným možnostem vlastní úpravy. Navržený komplet systémů byl poté použit pro návrh řešení. Následně byla navržena testovací síť a možnost její implementace. Pro usnadnění implementace celého kompletu byl vytvořen gitlab repositář umožňující nasazení kompletu pomocí nástroje Docker.

Hlavním prvkem práce byl návrh a tvorba vlastní aplikace umožňující vytvoření hlášení bezpečnostních událostí v systému TheHive. V rámci práce je GPON komunikace analyzována FPGA analyzátorem, který na základě detekce možné bezpečnostní události předává data Kafka serveru. Vlastní aplikace pak využívá asynchronního Kafka klienta v roli konzumenta, který odebírá z Kafka serveru zprávy obsahující popis zachycených událostí. Pomocí vytvořené knihovny, sloužící jako šablona hlášení pak podle typu události vytváří hlášení v systému TheHive.

Komplet systémů byl v rámci práce implementován podle návrhu pomocí vytvořeného repositáře v prostředí testovací sítě. Na stejný server pak byla nasazena i vlastní aplikace a bylo provedeno testování. Kompletní postup nasazení všech součástí včetně dodatečné konfigurace byl v této práci podrobně zdokumentován.

Pro otestování aplikace byl na straně Kafka serveru vytvořen klient v roli producenta. Tento producent simuloval detekci bezpečnostních událostí pomocí předem připraveného vzorku 2 typů událostí. Test proběhl úspěšně a testovací hlášení byla v systému TheHive úspěšně vytvořena.

Pozornost byla věnována také následné správě systému TheHive a vytvořených hlášení, ale i možnostem propojení se systémem MISP. Otestována byla také možnost

přenosu konfigurace a archivace kompletu systémů. Nakonec byla zohledněna celková stabilita a odolnost na vyšší zátěž. Vlastní aplikace ani využité systémy neměly problém se zpracováním vyššího množství zpráv a jsou tak vhodné i pro nasazení v reálném prostředí.

Výsledkem práce je připravený repositář umožňující nasazení kompletu systémů TheHive, Cortex a MISP pomocí nástroje Docker a vlastní aplikace v jazyce Python3 umožňující automatickou tvorbu bezpečnostních hlášení v systému TheHive.

Literatura

[1] *ITU-T: Recommendation G.984.1 Gigabit-capable Passive Optical Networks (G-PON): General Characteristics* [online], 3/2008 [cit. 2020-10-16]. Dostupné z: <http://www.itu.int/rec/T-REC-G.984.1/en>.

[2] *ITU-T: Recommendation G.984.1 Gigabit-capable Passive Optical Networks (G-PON): Physical Media Dependent (PMD) layer specification* [online], 3/2008 [cit. 2020-10-16]. Dostupné z: <https://www.itu.int/rec/T-REC-G.984.2/en>.

[3] *ITU-T: Recommendation G.984.3 Gigabit-capable Passive Optical Networks (G-PON): Transmission convergence layer specification* [online], 1/2014 [cit. 2020-10-16]. Dostupné z: <http://www.itu.int/rec/T-REC-G.984.3/en>.

[4] *ITU-T: Recommendation G.984.1 Gigabit-capable Passive Optical Networks (G-PON): ONT management and control interface specification* [online], 3/2008 [cit. 2020-10-16]. Dostupné z: <https://www.itu.int/rec/T-REC-G.984.4/en>.

[5] CALE, Ivica, Aida SALIHOVIC a Matija IVEKOVIC. *Gigabit Passive Optical Network - GPON.* In: 2007 29th International Conference on Information Technology Interfaces [online]. IEEE, 2007, 2007, s. 679-684 [cit. 2020-10-16]. ISBN 953-7138-09-7. ISSN 1330-1012. Dostupné z: <https://ieeexplore.ieee.org/document/4283853>.

[6] HOOD, Dave a Elmar TROJER. *Gigabit-capable Passive Optical Networks.* Canada: Wiley, 2012. ISBN 978-0470936870.

[7] WALID, Anwar a Aiyou CHEN. *Self-adaptive dynamic bandwidth allocation for GPON.* Bell Labs Technical Journal [online]. 2010, 15(3), 131-139 [cit. 2020-10-26]. ISSN 10897089. Dostupné z: <https://ieeexplore.ieee.org/document/6767875>.

[8] HORVATH, Tomas, Petr MUNSTER a Ning-Hai BAO. *Lasers in Passive Optical Networks and the Activation Process of an End Unit: A Tutorial.* Electronics [online]. 2020, 9(7) [cit. 2020-11-02]. ISSN 2079-9292. Dostupné z: doi:10.3390/electronics9071114

[9] HORVATH, Tomas, Lukas MALINA a Petr MUNSTER. *On security in gigabit passive optical networks.* In: 2015 International Workshop on Fiber Optics

in Access Network (FOAN) [online]. IEEE, 2015, 2015, s. 51-55 [cit. 2020-11-09]. ISBN 978-1-4673-7625-9. Dostupné z: <https://ieeexplore.ieee.org/document/7320479>.

[10] HORVATH, Tomas, Petr MUNSTER a Miloslav FILKA. *A Novel Unique Parameter for Increasing of Security in GPON Networks.* Journal of Communications Software and Systems [online]. 2017, 12(2), 112-116 [cit. 2020-11-09]. ISSN 1846-6079. Dostupné z: <https://dspace.vutbr.cz/handle/11012/69230>

[11] *ITU-T: Recommendation G.984.3 Gigabit-capable Passive Optical Networks (G-PON): Transmission convergence layer specification Amendment 3* [online], 1/2014 [cit. 2020-10-16]. Dostupné z: <http://www.itu.int/rec/T-REC-G.984.3/en>.

[12] MARTINEZ-MATEO, Jesus, Alex CIURANA a Vicente MARTIN. *Quantum Key Distribution Based on Selective Post-Processing in Passive Optical Networks.* IEEE Photonics Technology Letters [online]. 2014, 26(9), 881-884 [cit. 2020-11-09]. ISSN 1041-1135. Dostupné z: <https://ieeexplore.ieee.org/document/6762880>

[13] MALINA, Lukas, Petr MUNSTER Jan HAJNY a Tomas HORVATH. *Towards secure Gigabit Passive Optical Networks: Signal propagation based key establishment.* 2015 12th International Joint Conference on e-Business and Telecommunications (ICETE) [online]. IEEE, Colmar, 2015 [cit. 2020-11-09]. ISBN 978-1-4673-8532-9 Dostupné z: <https://ieeexplore.ieee.org/document/7518056>

[14] TRUONG QUANG VINH, JU-HYUN PARK, YOUNG-CHUL KIM a KWANG-OK KIM. *An FPGA implementation of 30Gbps security module for GPON systems.* In: 2008 8th IEEE International Conference on Computer and Information Technology [online]. IEEE, 2008, 2008, s. 868-872 [cit. 2020-11-10]. ISBN 978-1-4244-2357-6. Dostupné z: <https://ieeexplore.ieee.org/document/4594788>

[15] PIERRE, Kim. *IT Security Research by Pierre: GPON FTTH networks (in)security* [online]. 2016 [cit. 2020-11-14]. Dostupné z: <https://pierrekim.github.io/blog/2016-11-01-gpon-ftth-networks-insecurity.html>

[16] HORVATH, Tomas, Petr MUNSTER a Josef VOJTECH. *Deployment of PON in Europe and Deep Data Analysis of GPON.* A. ALIMI, Isiaka, Paulo P.

MONTEIRO a António L. TEIXEIRA, ed. Telecommunication Systems - Principles and Applications of Wireless-Optical Technologies [online]. IntechOpen, 2019, 2019-10-30 [cit. 2020-11-15]. ISBN 978-1-78984-293-7. Dostupné z: doi:10.5772/intechopen.82679

[17] SHANEMAN, K. a S. GRAY. *Optical network security: technical analysis of fiber tapping mechanisms and methods for detection and prevention.* In: IEEE MILCOM 2004. Military Communications Conference, 2004 [online]. IEEE, 2004, s. 711-716 [cit. 2020-11-15]. ISBN 0-7803-8847-X. Dostupné z: <https://ieeexplore.ieee.org/document/1494884>

[18] DIAA, M., M. SHALABY, A. A. MOHAMED, Kamel. M. M. HASSAN a Ayman. M. MOKHTAR. *Undetectable Tapping Methods for Gigabit Passive Optical Network (GPON).* In: 2018 14th International Computer Engineering Conference (ICENCO) [online]. IEEE, 2018, 2018, s. 52-57 [cit. 2020-11-15]. ISBN 978-1-5386-5117-9. Dostupné z: <https://ieeexplore.ieee.org/document/8636110>

[19] MENDONCA, Claudia, Mario LIMA a Antonio TEIXEIRA. *Security issues due to reflection in PON physical medium. In: 2012 14th International Conference on Transparent Optical Networks* (ICTON) [online]. IEEE, 2012, 2012, s. 1-4 [cit. 2020-11-15]. ISBN 978-1-4673-2229-4. Dostupné z: <https://ieeexplore.ieee.org/document/6254487>

[20] GUTIERREZ, David, Jinwoo CHO a Leonid G. KAZOVSKY. *TDM-PON Security Issues: Upstream Encryption is Needed.* In: OFC/NFOEC 2007 - 2007 Conference on Optical Fiber Communication and the National Fiber Optic Engineers Conference [online]. IEEE, 2007, 2007, s. 1-3 [cit. 2020-11-16]. ISBN 1-55752-831-4. Dostupné z: <https://ieeexplore.ieee.org/document/4348474>

[21] ŠIMONÍK, Jan a Tomáš HORVÁTH. *GPON sít s modifikovanou koncovou jednotkou.* Elektrorevue [online]. Fakulta elektrotechniky a komunikačních technologií VUT v Brně, 2018, 2018(4) [cit. 2020-11-16]. ISSN 1213-1539. Dostupné z: http://www.elektrorevue.cz/cz/clanky/komunikacni-technologie/0/gpon-sit-s-modifkovanou-koncovou-jednotkou/

[22] BYUNG-TAK LEE a MUN-SEOB LEE. *Remote Fault Detection Method for Time-Slot-Violated Terminal Using Periodic Probing Signal in TDM-PON.* Journal of Lightwave Technology [online]. 2009, 27(16), 3498-3508 [cit. 2020-11-20]. ISSN 0733-8724. Dostupné z: <https://ieeexplore.ieee.org/document/4815443>

[23] YAN, Y., S. YAMASHITA, S.-H. YEN, P.T. AFSHAR, V. GUDLA, L.G. KAZOVSKY a S.-W. WONG. *Invited Paper: Challenges in next-generation optical access networks.* IET Optoelectronics [online]. 2011, 5(4), 133-143 [cit. 2020-11-20]. ISSN 1751-8768. Dostupné z: <https://ieeexplore.ieee.org/document/5984711>

[24] *ITU-T: SERIES G: TRANSMISSION SYSTEMS AND MEDIA, DIGITAL SYSTEMS AND NETWORKS : Rogue optical network unit (ONU)considerations* [online], 2/2011 [cit. 2020-11-21]. Dostupné z: <https://www.itu.int/rec/T-REC-G.Sup49/en>.

[25] HUAWEI TECH CO LTD *METHOD, DEVICE AND SYSTEM FOR DETECTING ROGUE OPTICAL NETWORK UNIT.* EP3220556A1. Dostupné z: <https://worldwide.espacenet.com/patent/search/family/056125549/publication/EP3220556A1?q=GPON%20rogue%20onu>

[26] DRAKULIC, Sanda, Massimo TORNATORE a Giacomo VERTICALE. *Degradation attacks on Passive Optical Networks.* In: 2012 16th International Conference on Optical Network Design and Modelling (ONDM) [online]. IEEE, 2012, 2012, s. 1-6 [cit. 2020-11-21]. ISBN 978-1-4673-1442-8. Dostupné z: <https://ieeexplore.ieee.org/document/6210184>.

[27] *Critical RCE Vulnerability Found in Over a Million GPON Home Routers* [online]. 2018 [cit. 2020-11-21]. Dostupné z: <https://www.vpnmentor.com/blog/critical-vulnerability-gpon-router/>.

[28] *The Hacker News: A Simple Tool Released to Protect Dasan GPON Routers from Remote Hacking.* [online]. May 08 2018 [cit. 2020-11-21]. Dostupné z: <https://thehackernews.com/2018/05/protect-router-hacking.html>.

[29] *Shodan: GPON Home Gateway.* [online]. [cit. 2020-11-21]. Dostupné z: <https://www.shodan.io/search?query=title%3A%22GPON+Home+Gateway%22/>.

[30] *The Hacker News: 5 Powerful Botnets Found Exploiting Unpatched GPON Router Flaws.* [online]. May 08 2018 [cit. 2020-11-21]. Dostupné z: <https://thehackernews.com/2018/05/botnet-malware-hacking.html>.

[31] PIERRE, Kim. *IT Security Research by Pierre: Multiple vulnerabilities found in CDATA OLTs* [online]. 1 11 2016 [cit. 2020-11-21]. Dostupné z: <https://pierrekim.github.io/blog/2016-11-01-gpon-ftth-networks-insecurity.html>.

[32] *VulDB: Search: GPON* [online]. [cit. 2020-11-22]. Dostupné z: <https://vuldb.com/>.

[33] *Exploit Database: Search: GPON* [online]. 2020 [cit. 2020-11-22]. Dostupné z: <https://www.exploit-db.com/>.

[34] Brněnská nemocnice čelí kybernetickému útoku, neoperuje a převáží pacienty. *Idnes.cz/Zpravodajství* [online]. [cit. 2021-5-19]. Dostupné z: <https://www.idnes.cz/brno/zpravy/brno-nemocnice-fakultni-nemocnice-kyberneticky-utok.A200313_071531_brno-zpravy_bur>.

[35] ČESKO. *Předpis č. 181/2014 Sb.: Zákon o kybernetické bezpečnosti a o změně souvisejících zákonů (zákon o kybernetické bezpečnosti).* In: Sbírka zákonů ČR. [Praha], 2014, ročník 2014, 75/2014.

[36] ČESKO. *Předpis č. 82/2018 Sb.: Vyhláška o bezpečnostních opatřeních, kybernetických bezpečnostních incidentech, reaktivních opatřeních, náležitostech podání v oblasti kybernetické bezpečnosti a likvidaci dat (vyhláška o kybernetické bezpečnosti)).* In: Sbírka zákonů ČR. [Praha], 2018, ročník 2018, 43/2018.

[37] CSIRT vs. SOC – co je co a jaké jsou jejich úlohy? *IT SECURITY NETWORK NEWS* [online]. 2019 [cit. 2020-11-25]. Dostupné z: <https://www.itsec-nn.com/csirt-vs-soc-co-je-co-a-jake-jsou-jejich-ulohy/>.

[38] ŠTOREK, Daniel. *Pracoviště typu CSIRT a CERT v Evropě.* Ochrana & Bezpečnost. 2018, 7(1). ISSN 1805-5656.

[39] *CSIRT.cz* [online]. 2019 [cit. 2020-11-25]. Dostupné z: https://csirt.cz/

[40] *Vládní CERT.* NÚKIB [online]. [cit. 2020-11-25]. Dostupné z: <https://nukib.cz/cs/kyberneticka-bezpecnost/vladni-cert/>.

[41] *Redukce bezpečnostních hrozeb v optických sítích.* Network Research Group [online]. [cit. 2020-11-24]. Dostupné z: <http://nsr.utko.feec.vutbr.cz/VI20172020110.php>.

[42] OUJEZSKY, Vaclav, Tomas HORVATH, Michal JURCIK, Vladislav SKORPIL, Martin HOLIK a Marek KVAS. *Fpga Network Card And System For Gpon Frames Analysis At Optical Layer.* In: 2019 42nd International Conference on Telecommunications and Signal Processing (TSP) [online]. IEEE, 2019, 2019, s. 19-23 [cit. 2020-11-24]. ISBN 978-1-7281-1864-2. Dostupné z: <https://ieeexplore.ieee.org/document/8769054>.

[43] OUJEZSKY, Vaclav, Adrian TOMASOV, Martin HOLIK, Vladislav SKORPIL, Tomas HORVATH a Michal JURCIK. *GPON Traffic Analysis with TensorFlow.* In: 2020 43rd International Conference on Telecommunications and Signal Processing (TSP) [online]. IEEE, 2020, 2020, s. 69-72 [cit. 2020-11-24]. ISBN 978-1-7281-6376-5. Dostupné z: doi:<https://ieeexplore.ieee.org/document/9163575>.

[44] *Computer Security Incident Handling Guide.* Special Publication (NIST SP) - 800-61 Rev 2. 2012. Dostupné z: doi:<https://www.nist.gov/publications/computer-security-incident-handling-guide>.

[45] Catakoglu, Balduzzi, a Balzarotti, *Automatic extraction of indicators of compromise for web applications* in Proceedings of the 25th International Conference on World Wide. International World Wide Web Conferences Steering Committee, 2016, pp. 333–343

[46] *FIR* [online]. [cit. 2020-11-29]. Dostupné z: <https://github.com/certsocietegenerale/FIR>.

[47] *Wazuh* [online]. Wazuh, 2020 [cit. 2020-11-29]. Dostupné z: <https://wazuh.com/>.

[48] *Cyphon* [online]. [cit. 2020-11-29]. Dostupné z: <https://www.cyphon.io/>.

[49] *TheHive Project: SECURITY INCIDENT RESPONSE FOR THE MASSES* [online]. 2020 [cit. 2020-11-29]. Dostupné z: <https://thehive-project.org/>.

[50] *Cortex* [online]. 2020 [cit. 2020-11-29]. Dostupné z: <https://github.com/TheHive-Project/Cortex>.

[51] *IntelMQ* [online]. [cit. 2020-11-29]. Dostupné z: <https://github.com/certtools/intelmq>.

[52] *Elasticsearch: The heart of the free and open Elastic Stack* [online]. 2020 [cit. 2020-11-29]. Dostupné z: <https://www.elastic.co/elasticsearch/>.

[53] Simon Fredsted: *What is a Webhook?* [online]. 2018 [cit. 2020-11-29]. Dostupné z: <https://simonfredsted.com/1583>.

[54] *MISP: Open Source Threat Intelligence Platform & Open Standards For Threat Information Sharing* [online]. [cit. 2020-11-29]. Dostupné z: <https://www.misp-project.org/index.html>.

[55] *TheHive Project: Training Material* [online]. 2020 [cit. 2020-11-29]. Dostupné z: <https://github.com/TheHive-Project/TheHiveDocs/blob/master/training-material.md>.

[56] *Apache Kafka: INTRODUCTION* [online]. [cit. 2021-5-4]. Dostupné z: <https://kafka.apache.org/intro>.

[57] *Docker docs: Docker overview* [online]. [cit. 2021-5-4]. Dostupné z: <https://docs.docker.com/get-started/overview/>.

[58] *TheHive-Project / Docker-Templates* [online]. [cit. 2021-5-23]. Dostupné z: <https://github.com/TheHive-Project/Docker-Templates/tree/main/docker/thehive4-cortex3-misp-shuffle>.

[59] *Aiokafka* [online]. [cit. 2021-5-13]. Dostupné z: <https://github.com/aio-libs/aiokafka>.

[60] *Thehive4py* [online]. [cit. 2021-5-13]. Dostupné z: <https://github.com/TheHive-Project/TheHive4py>.

[61] *ThehiveDocs: Alert* [online]. [cit. 2021-5-13]. Dostupné z: <https://github.com/TheHive-Project/TheHiveDocs/blob/master/api/alert.md>.

Seznam symbolů, veličin a zkratek

AES Advanced Encryption Standard – standard pokročilého šifrování

API Application Programming Interface – rozhraní pro programování aplikací

BIP Bit Interleaved Parity – bitová parita

BWmap Bandwidth map – seznam informací o přidělení šířky pásma

CERT Computer Emergency Response Team – chráněná značka týmu zabývajícího se reakcí na kybernetické bezpečnostní incidenty

CSIRT Computer Security Incident Response Teams – tým zabývající se reakcí na kybernetické bezpečnostní incidenty

CRC Cyclic Redundancy Check – cyklický redundantní součet

CTR Counter mod – čítačový režim

DBA Dynamic Bandwidth Assignment – metoda dynamického přidělení šířky pásma

DBRu Dynamic Bandwidth Report upstream – dynamické oznámení šířky pásma

DoS Denial of Service – odepření přístupu ke službě

ECB Electronic Codebook mode – režim elektronické kódové knihy

ENISA European Union Agency for Cybersecurity – Evropská organizace kybernetické bezpečnosti

FEC Forward Error Correction – metoda dopředné korekce chyb

FER Forward Error Rate – chybovost rámců

FIR Fast Incident Response – platforma pro správu incidentů

FPGA Field Programmable Gate Array – programovatelné hradlové pole

GEM GPON Encapsulation Method – metoda zapouzdření GPON sítí

GPON Gigabit Passive Optical Network – gigabitová pasivní optická síť

GTC GPON Transmission Convergence Layer – vrstva přenosové konvergence GPON sítí

GUI	Graphical User Interface– grafické uživatelské rozhraní
HEC	Header Error Control – bity pro chybovou kontrolu hlavičky GEM rámce
HTTP	Hypertext Transfer Protocol – protokol určený k přenosu hypertextových dokumentů
ID	Identification – identifikátor
IDS	Intrusion Detection System – systém detekce průniku
IoC	Indicators of Compromise – indikátory kompromitace
IoT	Internet of Things – internet věcí
IP	Internet Protocol – protokol síťové vrstvy pro směrování paketů
IPS	Intrusion Prevention System – systém prevence průniku
ISP	Internet Service Provider – poskytovatel informačnís služeb
JSON	JavaScript Object Notation – odlehčený formát pro výměnu dat
LoF	Loss of Frame – ztráta rámce
LoS	Loss of Service – ztráta přístupu k službě
MISP	Malware Information Sharing Platform – platforma pro sdílení indikátorů kompromitace
MPLS	Multi-protocol Label Switching – multiprotokolové přepojování podle návěští
MSK	Master Session Key – hlavní klíč relace
NIST	National Institute of Standards and Technology – národní institut standardizace a technologií
NATO	North Atlantic Treaty Organization – Severoatlantická aliance
ODN	Optical Distribution Network – optická distribuční síť
OLT	Optical Line Termination – optické linkové zakončení
OMCC	Onu Management and Control Channel – kanál pro správu a kontrolu ONU

OMCI ONU Management and Control Interface – zpráva pro správu a kontrolu ONU

ONU Optical Network Unit – optická síťová jednotka

PCBd Physical control block downstream – kontrolní blok fyzické vrstvy sestupného rámce

PCIe Peripheral Component Interconnect Express – standard systémové sběrnice

PTI Payload Type Indicator – indikátor typu payload

PLI Payload Length Indicator – indikátor délky payload

PLOAM Physical Layer Operation, Administration and Maintenance – provoz, správa a údržba fyzické vrstvy

Psync Physical Synchronization – fyzická synchronizace, označení začátku rámce

REST APi Representational State Transfer API – architektura API rozhraní

RCE Remote Code Execution – dálkové spuštění kódu

RTT Round Trip Time – obousměrné zpoždění

SIEM Security Information and Event Management – správa bezpečnostních informací a událostí

RTT Security Orchestration, Automation and Response – bezpečnostní řízení, automatizace a reakce

SQL Structured Query Language – standardizovaný strukturovaný dotazovací jazyk

T-CONT Transmission container – přenosový kontejner

TDMA Time Division Multiple Access – deterministická metoda přístupu k médiu pro sítě se sdíleným médiem

ToS Theft of Service – odcizení přístupu ke službě

UDP User Datagram Protocol – transportní protokol bez záruky doručení

UI User Interface – uživatelské rozhraní

VoIP	Voice over Internet Protocol – technologie umožňující přenos digitalizovaného hlasu
WDM	Wavelength-division multiplexing – vlnový multiplex
XML	Extensible Markup Language – rozšiřitelný značkovací jazyk

Seznam příloh

A Obsah elektronické přílohy

Obsahem přílohy jsou dostupné zdrojové LaTex soubory diplomové práce společně se samotným dokumentem ve formátu pdf. V adresáři *HiveStack* se nachází soubory pro nasazení kompletu systému pomocí Docker. Postup nasazení je popsán v souboru *README.md*. V adresáři *GPON-incident-handling* se nachází zdrojové soubory vlastní aplikace. Postup konfigurace a spuštění je popsán v souboru *Readme.rst*. V adresáři Kafka se nachází soubor s testovacími zprávami ve formátu JSON. V posledním souboru *Server-credentials.txt* se nachází postup a přihlašovací údaje k virtuálnímu stroji s testovaným zapojením. Některé soubory jsou ve výpisu vynechány.

- `/` — kořenový adresář elektronické přílohy
  - `latex` — LaTex zdrojový kód práce
  - `thesis.pdf` — Dokument diplomové práce ve formátu pdf
  - `HiveStack` — Zdrojové souboru pro nasazení kompletu systémů
    - `backup` — Archiv otestované konfigurace
      - `backup.tar.gz`
      - `backup-credentials.txt` — Přístupové údaje
    - `Cortex`
      - `application.conf` — Konfigurační soubor
    - `TheHive`
      - `application.conf` — Konfigurační soubor
    - `docker-compose.yaml` — Soubor umožňující spuštění pomocí Docker
    - `README.md` — Popis postupu nasazení
  - `GPON-incident-handling` — Zdrojové soubory vlastní aplikace
    - `src`
      - `core`
        - `__init__.py`
        - `config.py`
        - `Logger.py`
      - `examples`
        - `alert.py` — Skript pro manuální vytvoření hlášení
      - `template`
        - `__init__.py`
        - `GponEvent.py`
      - `__init__.py`
      - `Config.py` — Konfigurační soubor
      - `Kafka_listener.py`
      - `theHive_alert_handler.py` — Hlavní spustitelný soubor
    - `docker-compose.yaml` — Soubor umožňující spuštění pomocí Docker
    - `Dockerfile` — Soubor pro vytvoření Docker obrazu
    - `Makefile` — Soubor pro automatické vytvoření Docker obrazu
    - `Readme.rst` — Popis postupu nastavení a spuštění
  - `Kafka`
    - `GPONEvents.txt` — Soubor s testovacími zprávami ve formátu JSON
  - `Server-credentials.txt` — Údaje pro připojení k testovacímu virtuálnímu stroji

B Testovací Kafka zprávy

Soubor připravených testovacích zpráv ve formátu JSON pro simulaci bezpečnostních událostí využitý v rámci testování Kafka producentem.


```
{1:{"EventType":"1","SuperFrameCounter":"930188273","IncidentData
   ":{"OnuID":"255", "MessageID":"24", "Data":"608502A0000"}},
2:{"EventType":"1","SuperFrameCounter":"932348237","IncidentData"
   :{"OnuID":"255", "MessageID":"24", "Data":"60D460C0000"}},
3:{"EventType":"1","SuperFrameCounter":"936668166","IncidentData"
   :{"OnuID":"255", "MessageID":"24", "Data":"616320C0000"}},
4:{"EventType":"1","SuperFrameCounter":"938828129","IncidentData"
   :{"OnuID":"255", "MessageID":"24", "Data":"61A322A0000"}},
5:{"EventType":"1","SuperFrameCounter":"943148060","IncidentData"
   :{"OnuID":"255", "MessageID":"24", "Data":"6231E2A0000"}},
6:{"EventType":"1","SuperFrameCounter":"947467993","IncidentData"
   :{"OnuID":"255", "MessageID":"24", "Data":"62C142A0000"}},
7:{"EventType":"1","SuperFrameCounter":"949628167","IncidentData"
   :{"OnuID":"255", "MessageID":"24", "Data":"631140C0000"}},
8:{"EventType":"1","SuperFrameCounter":"965947949","IncidentData"
   :{"OnuID":"255", "MessageID":"24", "Data":"717CA0B0300"}},
9:{"EventType":"1","SuperFrameCounter":"968107662","IncidentData"
   :{"OnuID":"255", "MessageID":"24", "Data":"71BC0290300"}},
10:{"EventType":"2","SuperFrameCounter":"968107662","IncidentData
   ":{"OnuID":"5","ActivationAnomaly":[ "4", "14", "18", "8", "8"
   , "8" ]}},
11:{"EventType":"2","SuperFrameCounter":"968864662","IncidentData
   ":{"OnuID":"2","ActivationAnomaly":[ "4", "14", "10", "8", "14
   ", "8", "8", "4", "14", "8", "8" ]}},
12:{"EventType":"2","SuperFrameCounter":"988119432","IncidentData
   ":{"OnuID":"6","ActivationAnomaly":[ "4", "4", "4", "14", "8",
    "8", "5", "8", "18", "18", "10", "14", "14", "8", "8" ]}},
13:{"EventType":"2","SuperFrameCounter":"981747674","IncidentData
   ":{"OnuID":"255","ActivationAnomaly":[ "21", "1", "3", "3", "3
   ", "3", "3", "1", "20", "3", "3", "3", "3", "1", "3" ]}},
14:{"EventType":"2","SuperFrameCounter":"988975435","IncidentData
   ":{"OnuID":"4","ActivationAnomaly":[ "5", "4", "8", "18", "8",
    "8", "5", "4", "14", "18", "10", "4", "14", "18", "10" ]}}
}
```