VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ
DEPARTMENT OF COMPUTER SYSTEMS

PŘEVOD MEZI FORMÁTY PRO SDÍLENÍ SÍŤOVÝCH BEZPEČNOSTNÍCH HLÁŠENÍ

CONVERSION BETWEEN FORMATS FOR SHARING OF NETWORK SECURITY ALERTS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **PAVEL EIS**
AUTHOR

VEDOUCÍ PRÁCE **Ing. MARTIN ŽÁDNÍK, Ph.D.**
SUPERVISOR

BRNO 2019

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačových systémů (UPSY) Akademický rok 2018/2019

Zadání bakalářské práce

Student: **Eis Pavel**
Program: Informační technologie
Název: **Převod mezi formáty pro sdílení síťových bezpečnostních hlášení**
Conversion between Formats for Sharing of Network Security Alerts
Kategorie: Algoritmy a datové struktury

Zadání:

1. Nastudujte oblast zabývající se síťovými bezpečnostními hlášeními. Zaměřte se na formáty sdílení jako jsou STIX, MISP, IDEA.
2. Analyzujte možnosti převodu mezi formáty.
3. Navrhněte postupy pro převod mezi formáty.
4. Navržené řešení implementujte.
5. Funkčnost implementace ověřte na datech ze systému Warden.
6. Zhodnoťte dosažené výsledky a navrhněte možná rozšíření.

Literatura:

- Dle pokynů vedoucího práce.

Pro udělení zápočtu za první semestr je požadováno:

- Splnění bodů 1 až 3 zadání.

Podrobné závazné pokyny pro vypracování práce viz http://www.fit.vutbr.cz/info/szz/

Vedoucí práce: **Žádník Martin, Ing., Ph.D.**
Vedoucí ústavu: Sekanina Lukáš, prof. Ing., Ph.D.
Datum zadání: 1. listopadu 2018
Datum odevzdání: 15. května 2019
Datum schválení: 26. října 2018

Abstrakt

Existuje celá řada platforem a systémů určených ke sdílení kybernetických bezpečnostních incidentů a událostí, které často používají rozdílné bezpečnostní formáty. Tímto způsobem dochází ke ztížení nebo přímo nemožnosti sdílení bezpečnostních incidentů a událostí mezi organizacemi, které využívají rozdílné platformy. Řešením tohoto problému může být vznik konvertorů, které jsou schopné převádět používané bezpečnostní formáty mezi sebou. Tato práce se zabývá převodem mezi bezpečnostními formáty IDEA, MISP a STIX. Při konverzi je důležité dbát na postup, aby docházelo k zachování informací, nebo aby při chybném převodu nevznikl jiný druh události, než byl reprezentován původní událostí. Pokud je převod dostatečně přesný, může být jednodušeji dosaženo přesnější a širší analýzy kybernetických bezpečnostních incidentů.

Abstract

There are many platforms and systems designed for sharing cyber security incidents and events, which often use different security formats. This way it gets harder or even not possible to share security incidents and events between organizations, which are using these platforms. Solution of this problem may be creation of converters, which are capable of converting used security formats between each other. This work solves conversion between security formats IDEA, MISP and STIX. In the process of conversion, it is important to care about conversion flow, to prevent information loss or different category of event assignment, than which it was originally represented by. If the conversion is accurate enough, it can be easier achieved more precise and broader analysis of cyber security incidents.

Klíčová slova

bezpečnostní událost, bezpečnostní incident, kybernetická bezpečnost, konverze datových formátů, platformní konektory

Keywords

security event, security incident, cyber security, conversion of data formats, platform connectors

Citace

EIS, Pavel. *Převod mezi formáty pro sdílení síťových bezpečnostních hlášení.* Brno, 2019. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Martin Žádník, Ph.D.

Převod mezi formáty pro sdílení síťových bezpečnostních hlášení

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Martina Žádníka, Ph.D. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Pavel Eis
10. května 2019

Poděkování

Rád bych poděkoval Ing. Martinu Žádníkovi, Ph.D. za velmi cenné rady v průběhu vývoje a odborné konzultace, jeho ochotu a čas, který mi věnoval. Dále bych rád poděkoval Pavlu Káchovi a Václavu Bartošovi ze sdružení Cesnet za příležitostné konzultace. Poslední, ale rozhodně ne nejmenší, poděkování patří mé rodině, která mě při studiu vždy podporovala.

Obsah

Kapitola 1

Úvod

Během posledních let se téměř veškerý přenos informací odehrává přes počítačovou síť. Snad všechny organizace používají síť pro sdílení dat a také pro přístup k internetu. Tohoto často může zneužít útočník, ať už za účelem odcizení citlivých dat přenášených přes počítačovou síť organizace, znemožnění jejího používání nebo dalších škodlivých úmyslů. Proto je nutné ji zabezpečit a takové pokusy útočníka nahlásit administrátorovi dané počítačové sítě. Ne všechny informace jsou ale pro administrátora relevantní a je nutné rozhodnout, které události má smysl reportovat. Je velice přínosné tyto nahlášené bezpečnostní události a incidenty nějakým způsobem obohatit pro lepší porozumění útoku a následné zajištění správných protiakcí.

Touto problematikou se zabývají systémy CTI (Cyber Threat Intelligence). CTI má mnoho definic a podob, ale zpravidla se jedná o *"organizovanou, analyzovanou a obohacenou informaci o potenciálním, právě probíhajícím nebo v minulosti uskutečněném útoku, který ohrožuje organizaci"* [14]. Tato informace je reprezentována určitou podobou, zpravidla se jedná o formát popisující informace vztahující se k hrozbě či útoku (pro účely této práce pojmenovaný kyberbezpečnostní formát). Každá organizace (skupina organizací) může mít jiné důvody pro zavedení CTI a proto se mohou jednotlivé formáty lišit. Jednou z podmínek pro volbu kyberbezpečnostního formátu je tedy možnost přenosu relevantních bezpečnostních informací pro danou organizaci. Existuje celá řada těchto formátů. Některé jsou minimalistické a zaměřují se na přenos pouze těch nejdůležitějších informací, které se mohou hodit více pro automatické zpracování. Jiné formáty zase umožňují popsat událost nebo incident více detailně, čímž může být více srozumitelná pro administrátora. Volba kyberbezpečnostního formátu, popř. jeho návrh, je velmi důležitá a může mít obrovský vliv na celý běh CTI. Špatně zvolený formát, který neobsahuje užitečná data, se těžko zpracovává, analyzuje a interpretuje do využitelné podoby. Analýza a porovnání některých kyberbezpečnostních formátů je rozebrána v kapitole 2.

Každá detekovaná kybernetická bezpečnostní událost nebo incident může mít jiný přínos pro odlišné organizace. Detekovaná událost, která nebyla hrozbou pro jednu organizaci a byla proto zahozena, se může hodit bezpečnostnímu administrátorovi druhé organizace. Díky této informaci může vykonat určité akce, které předejdou detekované hrozbě, která se šíří po síti dalších organizací. Proto je důležité detekované události sdílet s ostatními pro dosažení jejich lepší analýzy a zpracování. Informace obsažené v záznamu o detekované události mohou být často velmi citlivé pro organizaci a je nutné se zaměřit i na jejich spolehlivý a bezpečný přenos.

Dalším důležitým prvkem dobrého CTI je tedy sdílení detekovaných kybernetických bezpečnostních událostí a incidentů. Každá organizace ale může používat jiný kyberbez-

pečnostní formát a proto je nutné umět převádět jednotlivé formáty mezi sebou. Přínosem této bakalářské práce je analýza a převod některých reprezentací (formátů) bezpečnostních událostí a incidentů a byla zhotovena na základě smluvního výzkumu se sdružením Cesnet. Mezi převáděné formáty patří IDEA (Intrusion Detection Extensible Alert), formát využívaný platformou MISP (Malware Information Sharing Project) a modifikovaná verze formátu STIX, který využívá platforma C3ISP (Collaborative and Confidential Information Sharing and Analysis for Cyber Protection). Každý formát je určen k přenosu trochu jiného druhu informací, a proto je v kapitole 3 nejprve analyzováno, zda je možné převádět tyto formáty mezi sebou. Do analýzy jsou zahrnuty možné problémy, kterou mohou nastat v průběhu konverze. Správná analýza je klíčová, aby se části přenášených informací neztrácely nebo nepřeváděly chybně, kdy může vzniknout úplně jiný význam události, než byl ten původní. Kapitola 3 pokračuje kompletním návrhem převodu formátů. V následující kapitole (kapitola 4) je popsán způsob implementace konverze (konverzní knihovny) a obslužných konektorů, které mají za úkol načítání dat a obsluhu konverzních knihoven. Špatná konverze kybernetických bezpečnostních událostí a incidentů má velice špatný dopad na běh CTI, pokud jsou překonvertované události a incidenty dále využity. Proto je v kapitole 5 popsán způsob testování vzniklých konektorů a konvertorů.

Kapitola 2

Analýza a porovnání kyberbezpečnostních formátů

V této kapitole je rozebrána struktura jednotlivých kyberbezpečnostních formátů. Každý formát by měl mít možnost popsat základní atributy události, které dělají událost užitečnou pro budoucí zpracování, ať už člověkem nebo strojem. Mezi tyto atributy patří čas detekce události a kategorie události. Pokud není uvedena časová informace, správce neví, jestli se událost stala před rokem, čili už nejsou nutné žádné protiakce, nebo jestli se stala před půl hodinou a je vhodné na tuto událost nějakým způsobem zareagovat. Stala-li se událost před půl hodinou a správce na ní chce zareagovat, nejdříve musí zjistit, co se vlastně stalo. Na základě této informace (kategorie události) může provést nutné kroky, kterými se pokusí zamezit opakování události.

Dalšími společnými atributy se široké spektrum kyberbezpečnostních formátů rozchází. Některé se zaměřují na možný dopad události, některé se více zaměřují na popis síťového toku v průběhu události, další se zaměřují na jiné hodnoty. Jedním z hlavních formátů, kterými se tato práce zabývá, je IDEA, která se zaměřuje na popis síťových informací události. Proto se pro lepší srovnání formátů následující rozbory více soustředí na možnost formátu popsat síťové informace, čas a kategorii události. Ostatní možnosti formátů budou také rozebrány, ale ne tak detailně. Struktury formátů IDEA, STIX a MISP core format jsou rozebrány více do hloubky, jelikož uvedené informace jsou prerekvizitou pro kapitolu 3, která se zabývá konverzí mezi těmito formáty.

2.1 Intrusion Detection Extensible Alert – IDEA

IDEA [8] (Intrusion Detection Extensible Alert) je kyberbezpečnostní formát vytvořený sdružením Cesnet[1]. Byl vytvořen pro přenos událostí automatických detekčních systémů, často nazývaných IDS[2] (Intrusion Detection System). Vzhledem k tomuto hlavnímu účelu formátu byly navrhnuty požadavky na strukturu formátu IDEA. Důležitým požadavkem formátu byla jeho jednoduchost, protože komplexnost formátu není žádoucí při automatickém zpracování velkého množství událostí, které je generováno systémy IDS. Bezpečnostní zprávy formátu IDEA jsou také generovány velmi blízko samotným detekčním zařízením, které mohou být umístěny na platformách s omezeným výkonem. Proto je důležitá nezávislost na pokročilých knihovnách a nástrojích, které mohou vyžadovat vysoký výkon.

[1] https://www.cesnet.cz/

[2] https://searchsecurity.techtarget.com/definition/intrusion-detection-system

2.1.1 Struktura formátu

```
{
    "Format": "IDEA0",
    "ID": "4390fc3f-c753-4a3e-bc83-1b44f24baf75",
    "CreateTime": "2018-12-03T10:58:35",
    "DetectTime": "2018-12-03T10:58:30",
    "Category": ["Attempt.Exploit"],
    "Note": "Example of IDEA message",
    "Source": [
        {
            "IP4": ["192.168.0.1", "192.168.0.2"],
            "Hostname": ["example.com"],
            "Proto": ["tcp"]
        }
    ],
    "Target": [
        {
            "IP4": ["10.2.2.1", "10.2.2.2"],
            "Proto": ["tcp"]
        },
        {
            "IP4": ["10.3.2.1", "10.3.2.2"],
            "Proto": ["tcp"]
        }
    ],
    "Node": [
        {
            "Name": "com.hostname.report_machine",
            "SW": ["SoftwareName"]
        }
    ]
}
```

Výpis 2.1: Ukázková událost formátu IDEA

Následující informace o struktuře jsou převzaty z validačního schéma formátu IDEA (příloha B). Každá zpráva formátu IDEA musí obsahovat alespoň 4 atributy, kterými jsou *Format*, *ID*, *DetectTime* a *Category*. Jediná verze formátu, která zatím existuje, je IDEA0 (atribut *Format*) a *ID* je jednoznačný identifikátor zprávy. *DetectTime* sděluje přesný čas, kdy byla událost detekována. Formát IDEA běžně využívá i několik dalších časových značek (např. *CreateTime* popisující čas vzniku IDEA zprávy). Kategorie událostí formátu IDEA [9] jsou uloženy v atributu *Category*. Těchto kategorií může být více, kdy nejvyšší prioritu má první uvedená kategorie. Kategorie se zpravidla skládají z dvojic slov popisujících druh události, která jsou oddělena tečkou. První slovo je obecný popis, druhé pomáhá upřesnit typ útoku (např. *Malware.Trojan*). IDEA má předdefinovaný seznam kategorií, který je doporučen používat. V případě nemožnosti popsat daný typ útoku pomocí již definovaných kategorií je možné použít vlastní dvojici "kategorie.subkategorie" nebo kategorii *Other*.

Mezi dalšími atributy, jako jsou ostatní časové značky, popisky (*Note*, *Description*) zjednodušující porozumění zprávě člověku a další, jsou velice důležité atributy *Source* a *Target*. Informace uvnitř atributů *Source* a *Target* mají úplně stejnou strukturu, hlavní rozdíl je ale v jejich celkovém významu. *Source* nese informace o útočníkovi a *Target* nese informace o oběti události. Atributy *Source* i *Target* jsou uvnitř samostatných objektů datové struktury pole. Tímto je umožněn výčet a odlišení více útočníků a obětí v jedné IDEA zprávě. Tuto vlastnost lze využít v běžném případě, kdy obětí bylo několik nejen v jedné, ale více organizacích. Tímto mechanismem od sebe můžeme odlišit jednotlivé organizace a seskupit informace o nich. Objekty nesoucí tyto informace obsahují výčet IP adres, využité síťové protokoly při komunikaci, použité porty, MAC adresy a další informace podobného typu.

Mezi další často využívané atributy (objekty) formátu IDEA patří *Node* a *Attach*. Objekty *Node* (*Node* je datová struktura pole stejně jako *Attach*) obsahují informace o detekčním zařízení, kde důležitý je název detekčního zařízení (*Name*) a software (*SW*), který událost nebo incident detekoval. Objekty *Attach* obsahují různé přílohy od hodnot hashovacích algoritmů přes vzorky malwaru po různé pomocné nebo vysvětlující odkazy.

2.2 Malware Information Sharing Project – MISP

Platforma MISP [16] (Malware Information Sharing Project) používá jako výchozí formát MISP core format [4] (dále uváděn jako MISP formát). MISP formát je využíván pro popis bezpečnostních událostí nebo incidentů, ale také např. pro sdílení analýz různého malware. MISP formát je strukturalizován do podoby formátu JSON. Ovšem MISP platforma umožňuje export událostí i do několika dalších datových formátů používaných pro ukládání informací, např. XML a CSV, tak i přímo do bezpečnostního formátu STIX (verze 1 i 2).

2.2.1 Struktura formátu

V této sekci je popsána struktura MISP formátu v podobě nosného formátu JSON. Celá MISP událost je obalena atributem *Event* (hlavní objekt). Uvnitř objektu jsou obecné informace o události, které jsou uvedeny způsobem klíč a hodnota datového typu řetězec. Dále se uvnitř objektu *Event* nacházejí atributy, které místo přímé hodnoty obsahují pole objektů, které většinou obsahují specifičtější informace o události.

V obecných informacích události jsou důležité hodnoty uložené v atributech *info* a *date*. Atribut *info* slouží k základnímu textovému popisu události. Tento popis by měl být dostatečně jednoduchý, ale zároveň výstižný, aby každý uživatel, který přijde s událostí do styku, věděl, o jaký typ události se jedná. V atributu *date* je uloženo datum výskytu události. Je vhodné upozornit, že hodnota uložená v atributu *date* je samotné datum a nespecifikuje přesný čas. Přesný čas je obsažen v atributu *timestamp*, která informuje o tom, kdy byla provedena poslední aktualizace události nebo atributu obsaženém uvnitř události. Ihned po vytvoření události je tedy uložen čas jejího vytvoření. Pokud ale dojde k editaci události (např. je přidán nový atribut), je tato časová značka timestamp přepsána časem, kdy byla provedena tato editace. Poslední časovou informací, která je použita v MISP události, je *publish_timestamp*. Kdykoliv je vytvořena nebo editována událost, není ihned publikována. Při vkládání je možné, že uživatel, který vkládá událost, může čekat na nějaké dodatečné informace. Proto danou událost může publikovat, až když jsou uvedeny všechny podrobnosti. Poté je do *publish_timestamp* uložen přesný čas publikace. Kdykoliv je událost publikována, je tato informace oznámena dalším uživatelům pomocí knihovny ZeroMQ[3]. Proto je vhodné

[3] http://zeromq.org/

počkat na vložení všech atributů a objektů do události a publikovat až finální verzi, aby nedocházelo ke zbytečným upozorněním. Ne všechny MISP instance mají spuštěné upozornění pomocí ZeroMQ, ale i přesto je vhodné událost publikovat až jako hotovou. Jestli je událost publikována nebo ne, poznáme podle hodnoty boolean uvnitř atributu *published*. Je tedy možné, že událost není publikována (published je false), ale *published_timestamp* obsahuje časovou značku. Tato časová značka tedy sděluje čas poslední publikace události.

Každá událost formátu MISP obsahuje informace popisující úroveň sdílení (distribuce) události. Úroveň může být nastavena dvěmi způsoby. Uvnitř atributu *distribution*, který definuje úroveň pomocí číselných identifikátorů (např. 1 = This community only), nebo pomocí dalšího mechanismu platformy MISP zvaného MISP taxonomie, tzv. tagy, které jsou použity u téměř všech MISP událostí. MISP platforma obsahuje rozsáhlý seznam předem definovaných tagů[4]. Skládají se z jmenného prostoru, predikátu a hodnoty. Jmenný prostor a predikát jsou povinné a hodnota je volitelná (nemusí být použita). Příkladem tagu může být *ecsirt:abusive-content="spam"* (ecsirt je jmenný prostor, abusive-content je predikát a spam je hodnotou). Jedním z jmenných prostorů je také tzv. TLP[5] (Traffic Light Protocol), které je používané jako druhý mechanismus pro nastavení úrovně distribuce události. Samotná konverze událostí (bude rozebrána v kapitole 3) je na úrovni nezávislá, ale obslužný konektor by měl zajistit dodržení stanovené úrovně distribuce. Tagy slouží také pro určení druhu události a jsou uloženy uvnitř atributu *Tag*. Jelikož každá událost může být označena více tagy, atribut *Tag* obsahuje pole objektů, které kromě samotného tagu obsahují doplňující informace (např. *id*).

MISP platforma kromě taxonimií používá ještě tzv. Galaxies. Někdy samotný tag nestačí ke klasifikaci události a je proto dobré uvést dodatečné informace. Uvnitř atributu *Galaxy* tedy můžeme nalézt objekty, které nesou podrobný popis např. o útočníkovi, pokud je známý (může se jednat o známou škodlivou organizaci). Do tohoto objektu se poté může uvést země jejího původu, její motiv, různá synonyma, která s organizací souvisí a další užitečné informace, které by se jinak musely složitě dohledávat pokaždé, když je tento útočník zahrnut v nějaké události. MISP opět nabízí seznam již známých galaxies[6], které definují právě útočníky, ale i skenovací nástroje, vzory útoků nebo třeba obecné nástroje, které jsou využity při útocích.

Dalšími základními atributy uvnitř MISP události jsou *org_id* a *orgc_id*, které se odkazují na organizace dané MISP instance, které událost detekovaly a vytvořily. Za obě tyto akce může zodpovídat i jedna organizace. Stejné hodnoty id jsou poté uvedeny v objektech MISP události *Org* a *Orgc*, které ještě obsahují jméno organizace a její UUID (univerzální unikátní identifikátor organizace).

Pokud chceme zjistit podrobné informace o události, které jsou podstatnou částí bezpečnostní informace, je nutné nahlédnout do atributů události *Attribute* a *Object*. Atribut *Attribute* obsahuje pole tzv. MISP atributů. Každý takový atribut je objekt (viz výpis 2.2), který nese několik informací. Mezi ty nejpodstatnější patří *category* a *type*, díky kterým se pozná, jaký má hodnota uvnitř atributu význam. MISP platforma nabízí širokou škálu předdefinovaných atributů (seznam dostupný v [4], sekce 2.4.2.3). Jsou rozděleny do kategorií (např. Network activity nebo Payload delivery) a každá kategorie má poté definovány vlastní typy (např. ip-src). Takto je určen druh atributu a poté už je jasné, jaký má význam hodnota uvnitř samotného atributu *value*. Každý takový atribut má také vlastní *event_id*, které odkazuje na událost, ke které atribut patří. Toto id je využito v případě hledání ur-

[4] https://www.misp-project.org/taxonomies.html
[5] https://www.us-cert.gov/tlp
[6] https://www.misp-project.org/galaxy.html

čitých atributů pomocí API, které platforma MISP nabízí, k dohledání událostí, ke které hledaný atribut patří. Uživatel se tedy může např. dotázat na všechny atributy, které jsou kategorie Network activity, typu ip-src a hodnoty 192.168.0.1 a instance mu poskytne list všech atributů, které splňují tyto filtrační podmínky. Pomocí event_id si uživatel může jednoduše zobrazit událost, u které byla tato hodnota uvedena. U každého atributu je možné zanechat i vlastní komentář. Atribut také obsahuje svojí vlastní časovou značku *timestamp*, která udává čas vytvoření nebo poslední editace atributu. MISP atribut obsahuje i několik dalších atributů, jejichž hodnoty neposkytují důležité informace, jelikož se jedná převážně a nastavení samotného atributu vůči MISP instanci.

```
{
    "id": "1564223",
    "type": "ip-src",
    "category": "Network activity",
    "to_ids": true,
    "uuid": "5a0bef7e-3874-4309-b6f1-00f8ac101c39",
    "event_id": "13775",
    "distribution": "5",
    "timestamp": "1510731646",
    "comment": "",
    "sharing_group_id": "0",
    "deleted": false,
    "disable_correlation": false,
    "object_id": "0",
    "object_relation": null,
    "value": "192.168.0.1",
    "Galaxy": [],
    "ShadowAttribute": []
}
```

Výpis 2.2: Ukázka MISP atributu

Jedním ze základních principů ukládání informací do MISP události jsou tedy tzv. MISP atributy. Pomocí těchto atributů je možné do události vložit široké spektrum hodnot, není ale možné mezi těmito hodnotami vytvářet žádné vazby. Např. u dvou nebo více IP adres se může hodit sdělit informace, že jsou dané IP adresy z jedné organizace nebo, že k sobě logicky, vzhledem k dané události, patří. MISP platforma právě proto nabízí řešení v podobě MISP objektů[7]. MISP objekt seskupuje jednotlivé MISP atributy k sobě. Jedná se o naprosto stejné MISP atributy, které jsou samostatně uvedeny uvnitř pole objektů *Attribute*. Každý takový objekt má vlastní šablonu, podle které se objekt při vytváření naplní požadovanými atributy. Každý objekt má své jméno a popis, aby bylo jasné k čemu slouží. Má vlastní časovou značku *timestamp*, která má stejný účel, jako u samotného MISP atributu. Dále obsahuje atribut *template_uuid* pro identifikaci šablony a *template_version*, jelikož šablona se může aktualizovat. Poté už MISP objekt obsahuje atribut *Attribute*, uvnitř kterého je pole MISP atributů, které jsou takto uskupeny k sobě. MISP platforma opět nabízí předdefinované šablony a je dokonce možné vytvořit si vlastní šablony.

[7] https://www.misp-project.org/objects.html

2.3 Common Event Format – CEF

CEF [1] (Common Event Format) je textově založený formát navrhnut pro podporu mnoha typů zařízení nabídnutím nejvíce relevantních informací. Přenos událostí formátu CEF je provozován pomocí standardu syslog[8]. Pokud provozovatel nemůže používat syslog, může zvolit alternativu zápisu událostí do souboru. V tomto případě je samozřejmě vynechán prefix syslogu a zpráva začíná verzí formátu CEF.

2.3.1 Struktura formátu

```
CEF:Version|Device Vendor|Device Product|Device Version|Signature
ID|Name|Severity|Extension
```

Výpis 2.3: Základní struktura formátu CEF

```
Sep 19 08:26:10 host CEF:0|Security|threatmanager|1.0|100|worm
successfully stopped|10|src=10.0.0.1 dst=2.1.2.2 spt=1232
```

Výpis 2.4: Ukázka bezpečnostní události ve formátu CEF

Zpráva je rozdělena do tří částí: prefix standardu syslog, hlavička a rozšíření, kdy prefix syslogu a rozšíření může být vynecháno (výpis 2.3 ukazuje strukturu formátu bez prefixu syslogu). Jednotlivé položky hlavičky jsou od sebe odděleny znakem roury ('|'). Volitelný prefix syslogu obsahuje datum a čas, kdy byla zpráva vytvořena a název hosta, který zprávu vytvořil. V hlavičce se nachází verze formátu CEF, následuje název poskytovatele zařízení, označení zařízení a verze zařízení. Další položkou hlavičky je tzv. DeviceEventClass ID, které je unikátní podle druhu události. Nese hodnotu celého čísla nebo řetězce identifikující druh detekované události. Předposlední položka hlavičky je textový popis události pro lepší porozumění typu události. Jedná se pouze o jednoduchý popis, zpravidla jedna věta, která by neměla obsahovat žádné informace, které jsou vyhledatelné v rozšíření události, aby nedocházelo k duplikaci informací. Poslední políčko hlavičky popisuje závažnost události s rozsahem od 0 do 10. Čím vyšší je číslo, tím je závažnost vyšší.

Rozšíření zprávy je popsané formou dvojic klíč a hodnota, které jsou od sebe odděleny mezerou (ukázka ve výpisu 2.4). Názvy klíčů bývají zkratkou jejich opravdového významu, např. klíč *dst* znamená destination address (cílová adresa). CEF podporuje velké množství předdefinovaných klíčů, kde důležité je oddělení *source* a *destination* klíčů pomocí jejich prefixu 's' a 'd', představující informace o útočníkovi a oběti události. Tyto klíče umožňují popsat podrobnosti o síťových informacích události, např. IP adresy, protokoly, porty, ale i třeba název cílového procesu, na který mohl být útok zacílen. Klíč *rt* popisuje čas detekce události, další klíče umožňují popsat začátek a konec události. Někdy je použit i klíč *cat*, který přesněji specifikuje kategorii události.

Většina klíčů, které se mohou nacházet v rozšíření, je předem definovaná. Je možné ale zavést vlastní klíče pro vlastní hodnoty. Aby byly vlastní klíče rozlišitelné od těch předem definovaných, jsou definovány speciálním mechanismem. Název nového atributu (klíče) je uveden jako hodnota atributu *cs1Label* (např. dest_os). Hodnota tohoto nového atributu je poté k nalezení v atributu *cs1* (např. Windows 10). Takto definovaných vlastních klíčů může být i více, pro každý nový se pouze inkrementuje počítadlo (cs2Label, cs2 . . .).

[8] https://www.networkmanagementsoftware.com/what-is-syslog/

2.4 Structured Threat Information eXpression – STIX

Formát STIX [2] (Structured Threat Information eXpression) byl vytvořen pro přenos strukturovaných informací o kybernetických incidentech. Důležité bylo zachovat lidskou čitelnost, ale zároveň vytvořit strukturu formátu, kterou může být lehce zpracována automaticky. Důraz byl také kladen na dostatečnou komplexitu formátu, která umožňuje důkladně popsat kybernetický incident se všemi detaily, které přidají automatizované nástroje, ale i správci samotní.

2.4.1 Struktura formátu

Událost formátu STIX používá formát JSON jako nosný formát. Ten se hodí pro automatické zpracování událostí a je i dobře čitelný pro samotné správce. Celá událost je složena pomocí tzv. doménových objektů [13]. Každý objekt slouží pro přenos různých druhů informací. Hlavním objektem STIX události je objekt *bundle* [12], který slouží pro seskupení všech ostatních objektů, které přenáší informace o události. Objekt *bundle* samotný nemá žádný význam.

Pomocí doménových objektů je složena celá událost formátu STIX. Každý doménový objekt je vytvořen určitou indentitou, která je vyjádřena pomocí doménového objektu *identity* (tento objekt samotný je výjimkou). Objekt *identity* nejčastěji reprezentuje organizaci, která událost detekovala, ale může reprezentovat i samostatné osoby. Síťové informace jsou přenášeny v objektu *observed-data*. Uvnitř tohoto objektu jsou uvedeny informace, které byly zpozorovány. Mezi takové objekty patří IP adresy, MAC adresy, ale je možné popsat i klíč registrů. Zpozorované informace jsou poté k nalezení uvnitř atributu *objects* objektu *observed-data*, který je datového typu pole, čili jich může být uloženo několik. Doménových objektů formátu STIX je několik a každý z nich přenáší unikátní informace. Zajímavým mechanismem formátu STIX jsou tzv. STIX vzory, pomocí kterých je možné uvnitř doménového objektu *indicator* popsat podezřelou aktivitu, která může být později využita k detekci událostí.

Tato práce se zabývá konverzí proprietární verze formátu STIX, kterou pro svoje potřeby vyvíjí platforma C3ISP[9]. Platforma C3ISP vytvořila vlastní STIX objekty nebo pro svoje použití upravila již existující objekty, které používá pro přenos informacích o kybernetických incidentech (proprietární dokumentace uvedena v příloze C). Jedním z hlavních využívaných objektů formátu STIX, které využívá platforma C3ISP, je doménový objekt *observed-data*. Uvnitř objektu už ale nejsou informace v běžné podobě. Platforma C3ISP místo běžně užívaných objektů uvnitř atributu *objects* přenáší seznam kybernetických událostí, které byly detekovány systémy IDS a exportovány pomocí bezpečnostního formátu CEF (viz kapitola 2.3). Platforma C3ISP tak využívá minimalismu formátu CEF, který je jedním z vhodných formátů pro nekomplexní události, pro přenos základních informací. Několik spolu souvisejících událostí formátu CEF poté obalí formátem STIX, který poskytne dodatečnou komplexitu.

```
{
  "type": "stix-bundle",
  "id": "stix-bundle--9b4f417f6e3e42fa4c18ca2323f00310bf553712",
  "objects": [
    {
```

[9] https://c3isp.eu/

```
"type": "observed-data",
"id": "observed-data--afd9be8acc20763f759a821ff03e5336f2dfb66d",
 ⋮
"cybox": {
  "spec_version": "3.0",
  "objects": [
    {
      "items": [
          "CEF:0|Security|Log_Analyzer|1.0|100|
          ssh-2018-12-03T10:30:54.000+0100|5|src=192.168.0.1 app=ssh2
          user=root type=0 reason=bad_password outcome=failure
          date=1543828854000 dtz=Europe/Rome",
          ⋮
```

Výpis 2.5: Základní struktura formátu STIX platformy C3ISP

Některé události platformy C3ISP ve formátu CEF obsahují kategorii události uvnitř atributu *cat* (např. *cat=Trojan*) v tzv. rozšíření události (část události za posledním oddělovačem '|'). V ukázce události exportované ve formátu CEF (výpis 2.5) si lze ale všimnout, že některé druhy událostí mohou být odlišeny různými způsoby. V tomto případě napoví proprietární dokumentace platformy C3ISP. Jedná se o neúspěšný útok hrubou silou hádáním hesla. Síťové informace jsou uchovány uvnitř jednotlivých CEF událostí, kterých může být v jednom objektu *observed-data* vloženo několik.

2.5 Log Event Extended Format – LEEF

LEEF [6] (Log Event Extended Format) je formát přizpůsobený pro systém detekce hrozeb IBM Security QRadar[10] (platforma pro řízení síťového provozu vyvíjená firmou IBM).

2.5.1 Struktura formátu

```
Date IPAddress LEEF:Version|Vendor|Product|Version|EventID|
key=value<tab>key=value
```

Výpis 2.6: Základní struktura formátu LEEF

Formát LEEF má velice podobnou strukturu jako formát CEF (viz kapitola 2.3). Může být uvozen hlavičkou standardu syslog, následuje hlavička, která obsahuje základní informace o detekčním zařízení. Poslední hodnotou hlavičky je EventID, které identifikuje druh události a musí být konzistentní mezi produkty. Může nabývat celého čísla nebo textového řetězce. Jednotlivé hodnoty hlavičky jsou od sebe oddělené znakem roury ('|').

Za hlavičkou následuje popis atributů detekované události pomocí dvojic klíč a hodnota. Klíče formátu LEEF jsou opět předdefinovány a soustředí se na popis síťového toku a popis útočníka. Pomocí prefixů 'src' a 'dst' je rozlišen útočník a oběť události. Klíč *devTime* popisuje čas detekce události. Pokud je událost uvozena hlavičkou syslogu, časová hodnota uvedena v atributu *devTime* je upřednostněna před hlavičkou syslogu. Pokud v předdefinované množině klíčů není žádný, který by vyhovoval popisu hodnoty, která má být vložena do

[10] https://www.ibm.com/cz-cs/security/security-intelligence/qradar

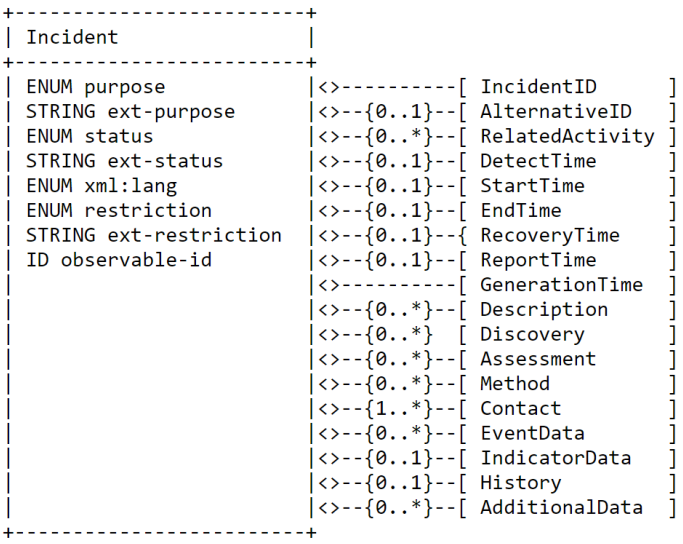
události, je možné si vytvořit vlastní klíč. Ovšem tento mechanismus by měl být využíván pouze v krajních případech. Po prozkoumání předdefinovaných klíčů formátu lze říci, že formát LEEF je podobný množině předdefinovaných klíčů formátu CEF. Jádro formátu je skoro stejné a obsahuje pouze několik klíčů navíc oproti formátu CEF.

2.6 Incident Object Description Exchange Format – IODEF

IODEF [3] (Incident Object Description Exchange Format) je určen pro přenos bezpečnostních incidentů a indikátorů. Formát IODEF využívá datové reprezentace XML a je přímým následníkem jeho předchozí verze IODEF. Předchozí verze se zaměřovala pouze na přenos a reprezentaci bezpečnostních incidentů. Nová verze přidává možnost popisu indikátorů hrozeb.

2.6.1 Struktura formátu

Celý formát je založen na rozdělení do tříd, kdy každá třída obsahuje specifické atributy. Celá událost, která je uvedena ve formátu IODEFv2, je obalena hlavní třídou *IODEF-Document*, do které se poté vkládají všechny ostatní třídy. Musí obsahovat minimálně jednu třídu *Incident*, která slouží k popisu běžných informací události a navíc je možné vkládat další třídy podrobněji popisující incident a síťový tok v průběhu incidentu. Třídy *Incident* můžeme najít v hlavním těle i více než jednu. Dále můžeme v hlavním těle doplnit dodatečné informace pomocí elementu *AdditionalData*.



Obrázek 2.1: Třída *Incident* formátu IODEFv2 (převzato z [3])

Většina informací se tedy vyskytuje v třídě *Incident* (viz obrázek 2.1). Na začátku třídy bývají uvedeny standardní časové značky, kde nejdůležitější je *DetectTime*, ale pro lepší popis časového průběhu incidentu IODEFv2 nabízí i *StartTime*, *EndTime* a další. Dále už se mohou objevit (minimální počet instancí těchto tříd je 0) více specifické informace v podobě třídy *Discovery*, popisující jak byl incident odhalen. Poté mohou následovat dvě důležité třídy *IndicatorData* a *EventData*.

Třída *IndicatorData* by měla obsahovat informace, které mohly pomoci včasné detekci problému a zabránění před jeho vznikem. Může poukazovat na určité slabiny v systému,

které byly zpozorovány v průběhu času. Tyto indikátory jsou zpravidla detekovány až po útoku, kdy se zkoumají možné příčiny, ale někdy mohou být detekovány dostatečně brzo a pouze na ně nebyl brán dostatečný ohled. V třídě *EventData* jsou uvedeny informace, které se přímo týkají průběhu bezpečnostního incidentu. V této třídě je podstatná podtřída *Flow*, obsahující jednu nebo více instancí třídy *System*. Třída *System* už obsahuje požadované dělení informací na útočníka a oběť. Poskytuje popis síťového toku, kde najdeme standardní atributy jako IP adresa, protokol, port a další. Můžeme si ale všimnout, že abychom se dostali k této informaci, museli jsme se poměrně hodně zanořit v struktuře. Ve formátu IODEF je toto velice běžné vzhledem k strukturalizaci do tříd. Pro zkušeného uživatele tohoto formátu to může být přehledné díky jednoznačnému dělení atributů do tříd popsaných v dokumentaci, ale pokud se na událost ve formátu IODEF koukne třeba i zkušený správce, který se s tímto formátem dříve nesetkal, může mít v první chvíli problém s nalezením požadované informace.

K popisu kategorie události slouží třída *Assesment* umístěná v třídě *Incident*. Umožňuje široce popsat důsledek události od dopadu na systém, společnost po peníze, které byly ztraceny kvůli aktivitě události. Správce pravděpodobně nejvíce zajímá podtřída *SystemImpact*, která se ze všech tříd dopadu nejvíce přibližuje popisu kategorie. Obsahuje popis úrovně závažnosti události, popis, zda událost uspěla nebo se pokus útočníka nezdařil. Jedním z atributů třídy *SystemImpact* je atribut *type*, který už přímo určuje druh události (např. availability-system může indikovat útok typu DoS - Denial of Service).

2.7 Intrusion Detection Message Exchange Format – IDMEF

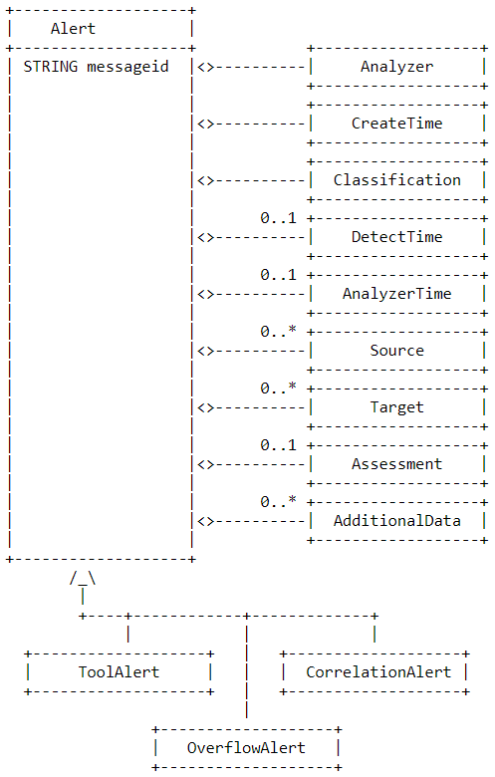
IDMEF [5] (Intrusion Detection Message Exchange Format) byl vytvořen pro výměnu zpráv systémů IDS mezi systémy řízení, které se systémy IDS mohou spolupracovat. Jako nosný formát je použita reprezentace XML.

2.7.1 Struktura formátu

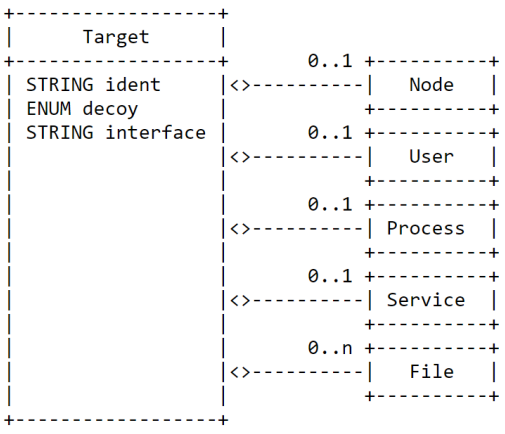
Formát IDMEF je svojí strukturou podobný formátu IODEFv2 (viz kapitola 2.6). Celá událost je strukturována do tříd a je obalena hlavní třídou *IDMEF-Message*, která může obsahovat třídu *Alert* nebo třídu *HeartBeat*. Třída *Alert* nese informace o události a třída *HeartBeat* slouží spíše pro ověření stavu detekčních zařízení. Zprávy třídy *HeartBeat* jsou odesílány v určitých intervalech, kdy delší doba bez obdržení zprávy třídy *HeartBeat* naznačuje chybu detekčního zařízení (např. ztracené připojení).

Třída *Alert* (viz obrázek 2.2a) obsahuje informace o události, mezi které patří časové značky jako *CreateTime* a *DetectTime*. Kategorie události je popsána pomocí třídy *Classification*. Tato třída obsahuje 2 jednoduché atributy pro popis druhu události a to *ident* a *text*. Atribut *ident* je jednoznačný identifikátor klasifikace, který není povinný a *text* je řetězec popisující druh události, který je určen poskytovatelem detekčního zařízení a je povinný. Dále poskytuje třídy *Source* a *Target*, popisující útočníka a oběť. Třídy *Source* a *Target* jsou svojí strukturou identické až na možnost popisu souboru (třída *File*), kterou umožňuje použít pouze třída *Target* (viz obrázek 2.2b).

Pro popis síťových informací slouží třída *Node*, která se nachází v obou třídách *Source* i *Target*. Popisuje lokaci uzlu sítě a umožňuje použít vícenásobně třídu její podtřídu *Address*, v které lze definovat adresu, ať už se jedná o IP adresu, MAC adresu, emailovou adresu a další. Třída *Address* umožňuje popsat i podrobnosti jako číslo a jméno VLAN, do které je adresa umístěna. Třída *User* slouží k popisu uživatele a třída *Process* lze použít



(a) Třída *Alert*



(b) Třída *Target*

Obrázek 2.2: Třídy *Alert* a *Target* formátu IDMEF (převzato z [5])

k popisu spuštěného procesu v průběhu události. Třída *Service* umožňuje popsat zbytek důležitých informací o síťovém toku jako použité porty a protokoly a poslední třída *File* popisuje specifické informace o souborech, které byly nějakým způsobem v průběhu události modifikovány a jak už bylo uvedeno dříve, je možné popsat pouze soubory oběti.

2.8 Porovnání kyberbezpečnostních formátů

Struktura každého formátu byla stručně popsána (pro více podrobností se nabízejí jejich vlastní dokumentace), aby bylo možné formáty následně porovnat mezi sebou podle určitých kritérií. Některá kritéria byla vybrána na základě použitých hodnotících kritérií v článku popisujícím zhodnocení bezpečnostních formátů a protokolů [15].
Hodnotící kritéria a jejich krátký popis:

- **Interoperabilita** – schopnost formátu popsat síťový útok
- **Rozšiřitelnost** – možnosti formátu o rozšíření vlastními atributy
- **Škálovatelnost** – úroveň schopnosti zacházení s velkým objemem dat (objemné bezpečnostní události)
- **Agregovatelnost** – schopnost agregace událostí
- **Závislost na transportním protokolu** – vazba formátu na určitý protokol přenosu událostí

- **Lidská čitelnost** – určuje úroveň čitelnosti událostí pro člověka
- **Strojová čitelnost** – definuje úroveň strojové zpracovatelnosti formátu
- **Jednoznačný popis události** – určuje zda je možné popsat událost více způsoby

2.8.1 Interoperabilita

Formáty IDEA, MISP, STIX, IODEFv2 a IDMEF mají vysokou interoperabilitu. Všechny tyto formáty obsahují několik atributů, tříd nebo objektů, které umožňují popsat průběh síťového toku při detekované události. Formáty LEEF a CEF umožňují díky předdefinovaným klíčům podrobně popsat síťové informace.

2.8.2 Rozšiřitelnost

Ne vždy formáty obsahují dostatek předdefinovaných atributů pro popis události a každý správce nebo detektor může chtít přidat nějakou vlastní hodnotu. Proto je dobré, když formáty podporují alespoň minimální formu rozšiřitelnosti. Formát IDEA umožňuje přidat jakýkoliv atribut podle potřeby. Formáty CEF a LEEF umožňují rozšířit své události o vlastní klíče, které by se ale měly používat pouze omezeně. I přesto je to způsob rozšíření formátu. Formáty IODEFv2 a IDMEF jsou rozšiřitelné s určitými omezeními. Rozšíření je možné přidat pouze pomocí elementů *AdditionalData* nebo v případě formátu IODEF u atributů uvozených předponou 'ext'. MISP platforma nabízí definici vlastních šablon objektů, které mohou být následně použity v událostech. Tyto šablony používají MISP atributy, které je možné také libovolně přidávat. Tímto je MISP formát dobře rozšiřitelný. Formát STIX umožňuje přidávat vlastní atributy k již existujícím objektům. Nabízí i možnost vytváření vlastních objektů. Tyto vlastnosti dělají formát STIX dobře rozšiřitelným.

2.8.3 Škálovatelnost

Některé události mohou obsahovat velké množství hodnot jednoho typu (např. čísla portů při skenování zařízení), které je vhodné držet u sebe. V kritériu interoperability bylo už lehce načnuto, že formáty CEF a LEEF nejsou příliš škálovatelné kvůli jejich struktuře rozšiřujících informací. Pokud by pod jedním klíčem byla vložena např. druhá IP adresa, dojde k přepsání předchozí hodnoty. Ke každé události formátu CEF i LEEF je tedy možné přiřadit pouze jednu hodnotu atributu. Zlepšení lze dosáhnout pomocí vlastních klíčů, ale jen ve velice omezené míře. Naopak formáty IDEA, MISP a STIX umožňují vysoké škálování díky nosnému formátu JSON a jeho datové struktuře pole. Formáty reprezentované pomocí XML jako IODEFv2 a IDMEF dosáhnou škálovatelnosti mnohonásobným vytvořením stejného elementu (třídy).

2.8.4 Agregovatelnost

Detekovaná událost nemusí být vždy samostatná, ale může se jednat o posloupnost malých událostí, které dohromady tvoří jednu velkou událost. Např. DoS útok může být v krátkém časovém okně detekován několika různými detektory nebo se může opakovat útok na jedno zařízení. Tento sled událostí se poté agreguje (seskupuje) do jedné události, která popisuje celý průběh DoS útoku a správce získá širší přehled o útoku na jeho síť. Formát IDEA podporuje agregovatelnost v podobě atributu *AggrID*, do kterého se vkládají *ID* IDEA zpráv, které spolu určitým způsobem souvisí, jsou agregovány. Tato událost je

poté seskupuje. MISP formát umožňuje vložit do atributu *RelatedEvent* neomezené množství objektů *Event*, které obsahují obecné informace o související události. Formáty STIX, CEF a LEEF nepodporují žádnou formu agregace. Události formátu IODEFv2 umožňují agregaci vytvořením více instancí třídy *Incident*. Formát IDMEF nabízí agregaci v podobě třídy *CorrelationAlert*, která slouží k výčtu všech ID událostí, které jsou z určitého důvodu agregovány k sobě.

2.8.5 Závislost na transportním protokolu

Pokud organizace nově zavádí CTI, mohou při výběru bezpečnostního formátu opomenout způsob jeho přenosu. Postupem času ale může chtít organizace svoji infrastrukturu rozšířit o další formát a je výhodné, pokud nemusí implementovat další transportní metodu. Pro přenos událostí formátu IDEA je využíván aplikační protokol HTTPS. MISP události je možné přijímat skrze technologii ZeroMQ, přes kterou instance posílají aktualizace událostí (i nové události). S MISP instancí je také možné komunikovat pomocí MISP API. Toto API ke svému chodu využívá aplikační protokol HTTP(S). Formáty CEF i LEEF používají jako transportní mechanismus standard syslog. Formáty IODEFv2 i IDMEF doporučují využití protokolu RID[11] (Real-time Inter-network Defense) pro transport událostí, ale nejsou na něj striktně vázány. Formát STIX je spojován s transportním mechanismem TAXII (využívá aplikační protokol HTTPS), který byl speciálně navržen pro transport událostí, které jsou reprezentovány formátem STIX.

2.8.6 Lidská čitelnost

Většina bezpečnostních událostí a incidentů je zpracována automaticky, ale některé události (většinou závažnějšího charakteru) si někdy vyžadují zvýšenou pozornost správce, který musí sám analyzovat událost a čitelnost formátu mu analýzu urychlí. Lidská čitelnost je i velice důležitá v případě testovacího procesu, kdy dochází k aktualizacím nebo vývoji nástrojů používajících daný formát. Formáty IDEA a STIX jsou dobře čitelné. Tento fakt je zajištěn nosným formátem JSON. Formátu IDEA tento pomáhá také minimalismus samotného formátu. MISP formát je čitelný, ale s přibývajícím množstvím dat se přehlednost snižuje, jelikož pouze jeden MISP atribut události obsahuje přibližně 16 atributů (*id*, *type*, *category*, *uuid*, *event_id*, *distribution* atd.), který popisují daný atribut. V případě rozsáhlé události, která obsahuje desítky MISP atributů se čitelnost snižuje. Formáty CEF a LEEF jsou dobře čitelné díky slovníkové struktuře, která není nijak zanořená. Čitelnost formátů IODEFv2 a IDMEF je nižší, kvůli jejich zanořující se třídní struktuře. Kvůli popisu informace o síťovém toku, jako např. IP adresa, se struktura zanoří do 7. úrovně v případě IODEFv2 (uvažujeme-li třídu *IODEF-Document* jako 1. úroveň) a do 6. úrovně v případě IDMEF (uvažujeme-li třídu *IDMEF-Message* jako 1. úroveň).

2.8.7 Strojová čitelnost

Detektory generují obrovské množství událostí, které není v lidských silách všechny zkontrolovat. Proto je většina událostí a incidentů zpracována automaticky. Dobře navrhnutý formát pro strojové zpracování urychlí analýzu. Všechny formáty jsou dobře strojově čitelné, běžný analyzátor by neměl mít problém s načtením a zpracováním těchto formátů. Formáty JSON i XML jsou tomu dobře uzpůsobeny, slovníková struktura formátů CEF a LEEF je také dobře zpracovatelná.

[11] https://tools.ietf.org/html/rfc6545

2.8.8 Jednoznačný popis události

Pro strojové zpracování je velmi důležité, aby událost mohla být popsána co nejméně možnými způsoby, ideálně pouze jedním. Každým dalším možným způsobem popisu se velice znesnadňuje strojová analýza události (je nutné ošetřit všechny možné scénáře), která je naprosto nezbytná při větším množství generovaných událostí. Pomocí formátů IDEA, CEF a LEEF není možné popsat událost více než jedním způsobem. MISP formát může událost popsat více způsoby, jelikož některé atributy mohou, ale nemusí být seskupeny do MISP objektu. Seskupení atributu do jednoho MISP objektu ještě neznačí mnoho cest popisu. Ovšem MISP platforma definuje mnoho druhů objektů a některé mají podobné atributy a mohou být použity pro popis stejných informací. Formát STIX je velice komplexní a podporuje také mnoho různých způsobů zápisu události. Formáty IODEFv2 i IDMEF také neposkytují jednocestný popis. Tato informace je sdělena přímo v dokumentaci těchto formátů (IODEF [3] sekce 7 Examples a IDMEF [5] sekce 7 Examples).

Kapitola 3

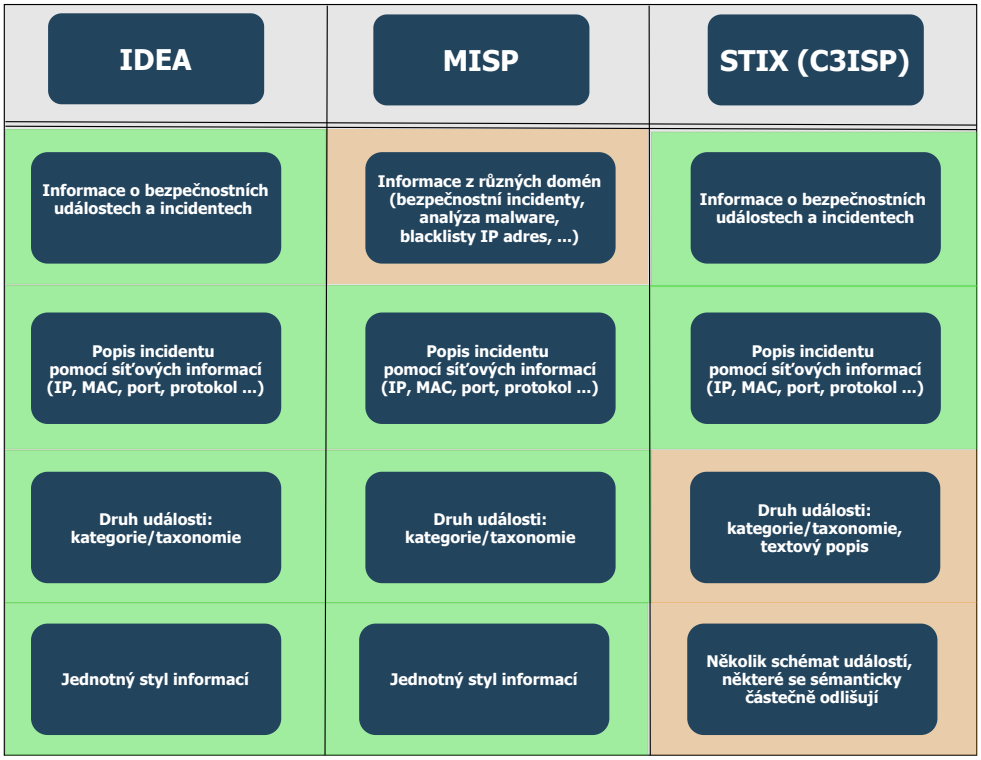
Návrh konverze kyberbezpečnostních formátů

Před začátkem návrhu konverze je nutné důkladně analyzovat konvertované formáty. Krátká analýza byla uvedena v kapitole 2, ale je nutné konvertované formáty prozkoumat více do detailů. S tím už ale pomůže oficiální dokumentace jednotlivých formátů. Po analýze je vhodné vyhodnotit, zda je možné formáty konvertovat mezi sebou, zda přenášejí stejný druh informací. Informace v této kapitole týkající se struktury formátu IDEA jsou převzaty z validačního schéma formátu IDEA (příloha B), informace o MISP formátu jsou převzaty z dokumentace formátu [4] a informace o formátu STIX jsou převzaty z proprietární dokumentace platformy C3ISP (příloha C).

3.1 Analýza možnosti převodu mezi formáty

Tato práce se zabývá konverzí mezi formáty IDEA a MISP a konverzí mezi formáty IDEA a proprietární verzí formátu STIX. První možností řešení problému konverze událostí je, pokud to přinese výhody, nahradit stávající formát IDEA formátem, který je určen ke konverzi. Ovšem formát IDEA byl navrhnut a vytvořen podle několika vlastních a speciálních požadavků (viz kapitola 2.1). Sdílení dat s formátem STIX nebo MISP je pouze malá část infrastruktury, kterou jednoduše upraví konvertor formátů. Ostatní části infrastruktury pracují efektivněji s formátem IDEA, jeho nahrazení tedy není vhodné pro řešení problému.

Formát IDEA je používán pro přenos událostí, které jsou generované systémy IDS (Intrusion Detection System). V průběhu vývoje byl umožněn přístup k MISP instanci organizace CIRCL (Computer Incident Response Center Luxembourg). Po prozkoumání a analýze sdílených událostí touto instancí byly nalezeny mimo jiné i události, které jsou právě generovány systémy IDS. Bylo usouzeno, že některé události přenášené formáty IDEA a MISP jsou vzájemně kompatibilní a jejich konverze by mohla přinést užitečné výsledky. Proprietární verze formátu STIX platformy C3ISP slouží pro přenos mnoha druhů událostí, mezi které patří také události systémů IDS. Hlavní rozdíly vlastností konvertovaných formátů jsou znázorněny na obrázku 3.1. U proprietární verze formátu STIX není vždy jednoduché správně určit kategorii události. Toho je ale dosaženo pomocí predikce možných kategorií a na základě dokumentace formátu. Navíc proprietární verze formátu STIX obsahuje několik schémat objektů, které si nejsou vždy sémanticky podobné. Tento jev ztěžuje jejich analýzu a automatickou konverzi. V kapitole 3.4 je popis řešení tohoto problému. Jde o výběr vhodných objektů určených ke konverzi.



Obrázek 3.1: Základní vlastnosti konvertovaných formátů (zelené podbarvení daného řádku značí stejné vlastnosti, oranžové podbarvení řádku značí pouze podobné vlastnosti)

U platformy MISP je při konverzi do formátu IDEA nutné správně vyfiltrovat druhy událostí, které korespondují k událostem, které jsou přenášeny formátem IDEA. Při vývoji konvertoru mezi formáty IDEA a MISP byl umožněn přístup k reálně využívané instanci MISP platformy, kterou provozuje centrum CIRCL (The Computer Incident Response Center Luxembourg). Tato instance hojně využívá sdílení externích analýz různého malware a událostí, které ale nejsou relevantní pro konverzi do formátu IDEA. Až po detailním průzkumu instance byly nalezeny i události, které svým obsahem a myšlenkou korespondovaly s běžným obsahem událostí formátu IDEA. V takovém případě bude tedy konvertovaná pouze předem definováná část událostí. Ovšem takové události není zas tak jed-noduché vyfiltrovat. V kapitole 2.2 bylo zmíněno, že platforma MISP umožňuje události kategorizovat pomocí tzv. taxonomií neboli tagů. Po analýze dostupných taxonomií jsou vybrány ty, kterými jsou označované námi chtěné události nebo které podle svého popisu slouží tomuto účelu.

Díky vzniklému filtračnímu listu tagů se podařilo odstranit značnou část událostí, které nejsou relevantní. Nicméně některé zbývající události jsou nejednoznačné. Např. událost označena tagem *circl:incident-classification="malware"* může označovat událost generovanou systémem IDS, kdy došlo k přenosu škodlivého malware nebo může jít pouze o analýzu daného malware, který byl detekován při nějakém incidentu. Pod jedním tagem se tak mohou skrývat dva odlišné koncepty událostí, kdy jeden z nich by neměl být konvertován. Další heuristikou, zda je možné provést konverzi, je analýza časových značek atributů. Každý atribut MISP události má vlastní časovou značku a při manuální analýze těchto událostí byl vypozorován vzor, kdy atributy externí analýzy některých událostí byly přidány k události až po delší době, než atributy nesoucí ostatní informace. Když byly atributy externí ana-

lýzy přidány s dlouhou prodlevou, navíc až těsně před publikací události, jedná se často o událost, která byla generována systémem IDS a pouze se čekalo na provedení dodatečné externí analýzy. U takových událostí stačí tedy pouze ignorovat atributy externí analýzy a překonvertovat všechny ostatní užitečné hodnoty události. Pokud ale byly vloženy atributy externí analýzy ihned, jedná se pouze o analýzu události. Nicméně ani tato heuristika nepřináší uspokojivé výsledky.

Na základě konzultací s vývojáři MISP platformy bylo doporučeno nepoužívat výše uvedenou heuristiku, ale vytvořit si vlastní list taxonomií (jmenný prostor), který by byl kopií 1:1 kategorií využívaných formátem IDEA a byl následně využíván pro označování takových událostí. Velmi podobné taxonomie jsou už ale definovány a formát IDEA z nich dokonce vychází. Těmito taxonomiemi jsou jmenné prostory ECSIRT (Incident Classification by the ecsirt.net) a RSIT (Reference Security Incident Classification Taxonomy). Po bližší analýze těchto dvou jmenných prostorů vyplývá, že pokud je událost označena takovýmto tagem, jedná se téměř vždy o událost generovanou systémem IDS. Jmenné prostory ECSIRT a RSIT jsou proto zvoleny jako vhodné pro konverzi mezi formáty MISP a IDEA. Většina tagů z těchto jmenných prostorů byla již dříve umístěna ve filtračním listu, došlo v podstatě k odebrání ostatních tagů, které nezaručovaly korektní konverzi.

Potenciální výsledky konverze vzorových událostí platforem C3ISP i MISP byly vyhodnoceny jako užitečné. Po těchto vyhodnoceních došlo k samotnému návrhu konverze, ve kterém jsou řešeny výše zmíněné problémy a podle kterého byly poté implementovány konverzní knihovny.

3.2 Návrh konverze formátu IDEA do formátu MISP

Konverze začíná s převodem základních informací o události formátu IDEA. Formát IDEA obsahuje několik časových značek. MISP událost sice také, ale při vytváření nové události umožňuje spravovat pouze jednu a tou je atribut *date* (datum detekce události). Do tohoto atributu je při konverzi vložena časová značka z události IDEA atributu *DetectTime* (čas je useknut, do tohoto pole události se vkládá pouze datum). Ostatní časové značky formátu IDEA nejsou využity, jelikož nejsou přímo převeditelné do MISP formátu. Ostatní časové atributy MISP formátu jsou zpracovány automaticky MISP instancí (*publish_timestamp* se nastaví při publikaci události, *timestamp* sděluje poslední změnu události atd.).

Mezi další atributy MISP události patří *threat_level_id* (nabývající hodnot *1* → Low, *2* → Medium, *3* → High). Oficiální specifikace formátu IDEA neobsahuje podobnou informaci, ale někdy bývají použity proprietární atributy v objektu *_Cesnet*. Jedním z těchto atributů je *EventSeverity* (nabývá hodnot *low*/*medium*/*high*/*critical*), který obsahuje stejnou informaci jako právě MISP atribut *threat_level_id* (jediný rozdíl je ve formátu, IDEA slovně, MISP numericky). Pokud se objekt *_Cesnet* s atributem *EventSeverity* nachází v události IDEA, je převeden (IDEA hodnota *critical* je převedena jako *high*). Pokud atribut *EventSeverity* není použit v události IDEA, je do MISP atributu *threat_level_id* vložena hodnota *0*, která znamená nedefinováno. MISP atribut *analysis* je irelevantní v případě konverze z formátu IDEA, proto je vložena hodnota *2* (znamená, že analýza je dokončena, vkládání atributů analýzy ve webovém rozhraní MISP instance bude i přesto možné v případě potřeby). IDEA aribut *Description* je vložen do MISP atributu *info*. Pokud IDEA událost obsahuje dodatečné informace v atributu *Note*, jsou vloženy do samostatného MISP atributu typu *comment*. Událost formátu IDEA může obsahovat pouze atribut *Note* a atribut *Description* nemusí být uveden. Atribut *Note* je v takovém případě vložen přímo do atributu *info*, jelikož nemůže zůstat nevyplněn. Posledním povinným atributem při vytvá-

ření MISP události je vyplnění úrovně distribuce. Ta záleží na konkrétním použití konverze a MISP instanci, v našem případě je použita hodnota *1* (distribuce pouze pro tuto komunitu) a je vložena do atributu MISP události *distribution*. Celkový postup je znázorněn v tabulce 3.1.

Událost formátu IDEA	**Událost formátu MISP**
"**__Cesnet**": {"EventSeverity": "low"}	"**thread__level__id**": "1"
"**DetectTime**": "2018-12-17T12:28:59Z"	"**date**": "2018-12-17"
"**Description**": "Synthetic example"	"**info**": "Synthetic example"
	"**analysis**": "2"
	"**distribution**": "1"

Tabulka 3.1: Konverzní tabulka základních informací formátů IDEA a MISP

3.2.1 Konverze kategorie formátu IDEA do MISP formátu

U tohoto převodu je také důležitá správná konverze kategorie události formátu IDEA do MISP formátu. Již dříve bylo zmíněno, že formát IDEA je orientován na popis síťových událostí nebo incidentů, které jsou generovány systémy IDS. MISP platforma dokáže definovat několik druhů událostí, mezi které patří právě síťové události a incidenty, ale i mnohé další události jako např. různé analýzy malwaru a hrozeb obecně a mnohé další. Je důležité nemíchat tyto druhy událostí mezi sebou.

Tato část konverze je řešena převodem kategorie formátu IDEA do MISP taxonomií – tagů. V kapitole 3.1 byl zdůvodněn výběr taxonomií jmenných prostorů ECSIRT a RSIT. Na základě analýzy jmenných prostorů ECSIRT a RSIT je vytvořena konverzní tabulka kategorií (tabulka 3.2), podle které jsou převáděny kategorie formátu IDEA do MISP tagů. Je dobré zdůraznit, že MISP událost může obsahovat více tagů najednou (stejně tak IDEA může obsahovat více kategorií), proto budou při konverzi jedné IDEA kategorie využity oba jmenné prostory, pro jasnou kategorizaci události.

Takto by konverzní tabulka 3.2 mohla pokračovala dále. Mapování těchto dvou jmenných prostorů na klasifikaci formátu IDEA je poměrně přímočaré a proto není nutné uvádět celou tabulku tabulku. Nicméně i v této části konverze jsou menší nejasnosti. Např. poslední uvedený řádek konverzní tabulky ukazuje, že ani jmenný prostor ECSIRT ani RSIT neobsahuje hodnotu podobnou výrazu *searching* v případě predikátu *information-gathering*. V takovém případě lze ale využít volitelnosti poslední části taxonomie (hodnoty) a její místo zůstane prázdné. Poté ani jeden ze dvou použitých jmenných prostorů neobsahuje predikát podobný IDEA kategorii *Anomaly*. Jelikož predikát musí být použit, nezbývá nic jiného než využít predikátu *other* a jeho hodnotu *other*. Takto sice není specifikováno, o jaký typ události se jedná, ale použití alespoň jmenného prostoru nám jasně sdělí, že se jedná o druh události, který je definován těmito jmennými prostory.

Kategorie formátu IDEA	**Tag formátu MISP**
Abusive.Spam	**ecsirt:abusive-content**="spam" **rsit:abusive-content**="spam"
Abusive.Harassment	ecsirt:abusive-content="harmful-speech" rsit:abusive-content="harmful-speech"
Abusive.Child	ecsirt:abusive-content="violence" rsit:abusive-content="violence"
Abusive.Sexual	ecsirt:abusive-content="violence" rsit:abusive-content="violence"
Abusive.Violence	ecsirt:abusive-content="violence" rsit:abusive-content="violence"
Malware.Virus	**ecsirt:malicious-code**="virus" **rsit:malicious-code**="virus"
Malware.Worm	ecsirt:malicious-code="worm" rsit:malicious-code="worm"
Malware.Trojan ⋮ **Recon**.Searching ⋮	ecsirt:malicious-code="trojan" rsit:malicious-code="trojan" ⋮ **ecsirt:information-gathering** **rsit:information-gathering** ⋮

Tabulka 3.2: Konverzní tabulka IDEA kategorií do MISP taxonomií

3.2.2 Konverze IDEA objektů *Source*, *Target* a *Attach*

Po převedení základních informací o IDEA události následuje konverze objektů uvnitř atributů *Source*, *Target* a *Attach* (pokud jsou v IDEA události použity). Nabízí se jednoduchý přístup, který spočívá v postupném procházení atributů těchto IDEA objektů a nalezení souvisejícího MISP atributu, do kterého by byla hodnota pouze překopírována. Avšak tímto postupem by se ztratily určité vazby mezi informacemi. Atributy *Source*, *Target* a *Attach* obsahují pole objektů, které sdružují určité informace. Např. v prvním objektu uvnitř pole *Source* mohou být uchovány společně informace o jednom zdroji události (několik IP adres, portů a další informace) a v druhém objektu uvnitř pole *Source* mohou být uchovány informace o druhém zdroji (jiné IP adresy a porty). Takto je jasné, že první skupina IP adres a portů spolu nějak souvis (např. mohou být z jedné organizace) a druhá skupina IP adres a portů mohou být z druhé organizace. Pokud by ale tyto adresy byly převedeny do

samostatných MISP atributů, tato informace o jejich spojitosti by se ztratila. Nově překonvertovaná MISP událost by tak obsahovala pouze množinu IP adres bez jakýchkoliv vazeb. Stejný efekt by se projevil i na ostatních atributech objektů v polích objektů *Source*, *Target* a *Attach*.

Tento problém je vyřešen díky otevřenosti platformy MISP k rozšířením. MISP platforma používá mimo MISP atributy také MISP objekty. Každý MISP objekt má definovanou šablonu, podle které může být naplněn konkrétními daty. A MISP také umožňuje vytvořit a importovat vlastní šablony objektů, které mohou nabývat mnoha podob. Pro účely převodu IDEA objektů uvnitř polí *Source* a *Target* jsou vytvořeny dvě šablony MISP objektů. I přestože objekty polí *Source* a *Target* mají stejnou definici a obsahují stejné atributy, musí být pro každý z nich vytvořena samostatná šablona. Protože jednotlivé atributy, které jsou používány ve vytvářených objektech musí přímo referovat na již předem definovaný MISP atribut. V případě třeba IP adresy v objektu *Source* se bude takový atribut odkazovat na MISP atribut *ip-src*. A poté IP adresa v objektu *Target* se bude odkazovat na MISP atribut *ip-dst*. Obě šablony budou tedy velice podobné až na pár výjimek.

Následující popis MISP objektů byl převzat z [7]. Každá šablona MISP objektu (viz výpis 3.1) má určité jméno (*name*) a je zařazena do určité kategorie objektů (*meta-category*). Aby každý porozuměl jejímu účelu, obsahuje i jednoduchý popisek (*description*). Pro možnost reference na určitou šablonu je vygenerován její identifikátor (*uuid*) a jelikož podoba šablony může být z mnoha důvodů v budoucnosti změněna (např. rozšíření o další atribut), je uchovávána i její verze (*version*).

Poté už následuje výčet MISP atributů uvnitř pole *attributes*, který daná šablona umožňuje použít. Každý atribut má své vlastní zástupné jméno, které slouží pro rozmanitější definici atributů. Všechny tyto atributy mohou nabýt hodnoty, kterou umožňuje nést jakýkoliv MISP atribut. Každý atribut objektu tedy obsahuje přímou referenci na již existující definici MISP atributu, aby bylo jasné, které hodnoty může nabýt. Tento vztah je definován klíčem *misp-attribute*, jehož hodnotou je název MISP atributu, ke kterému se atribut objektu odkazuje (udává jaké hodnoty může nabýt). Další hodnotou definice atributu objektu je atribut *categories*, který udává, do které kategorie spadá MISP atribut, na který se atribut MISP objektu odkazuje (např. typ atributu *ip-src* je zařazen v kategorii *Payload delivery*, ale také i v kategorii *Network activity*). Definice atributu objektu obsahuje dále popis atributu (aby bylo jasné, k čemu slouží), klíč *multiple*, který pokud nese hodnotu *True*, umožňuje při vytváření objektu vytvořit více atributů stejného jména (běžně lze pouze jeden atribut stejného jména, tento mechanismus zastupuje vlastnosti pole hodnot) a *ui-priority*, který definuje pořadí řazení atributů ve webovém rozhraní MISP instance (atribut s nižší *ui-priority* se zobrazí dříve).

```
{
    "name": "network-source",
    "meta-category": "network",
    "description": "Description of the source of the event",
    "uuid": "63cf1c78-4afe-49be-baff-2c101a942000",
    "version": 1,
    "attributes": {
        "Hostname": {
            "misp-attribute": "hostname",
            "description": "Participating hostname",
            "ui-priority": 0,
```

```
        "categories": ["Network activity"],
        "multiple": true
    },
    "IP4": {
        "misp-attribute": "ip-src",
        "description": "Source IPv4 address",
        "ui-priority": 1,
        "categories": ["Network activity"],
        "multiple": true
    },
    "MAC": {
        "misp-attribute": "mac-address",
        "description": "Source MAC adress",
        "ui-priority": 0,
        "categories": ["Network activity"],
        "multiple": true
    },
    ⋮
```

Výpis 3.1: Ukázka šablony MISP objektu

Šablona MISP objektu (výpis 3.1) by pokračovala dále s ostatními atributy. Už bylo řečeno, že vytvořená šablona tedy obsahuje všechny atributy objektů *Source* až na pár výjimek. Mezi tyto výjimky patří atribut *AttachHand*, který ve formátu IDEA slouží jako reference na objekt *Attach*. Pro objekt *Attach* je sice taky vytvořena šablona, ale tato reference by musela být uvedena v samostatném atributu. Hlavním cílem ovšem bylo zachovat nově definované šablony co nejjednodušší, aby bylo jejich použití naprosto jasné i ostatním uživatelům MISP instance.

Další vynechané atributy, které nejsou zahrnuty do šablony, jsou *Spoofed*, *Imprecise* a *Anonymised*. Tyto atributy nesou hodnotu typu boolean, která není definovaná mezi atributy MISP formátu. Hodnota by se i přesto mohla vložit do atributu text, ale význam těchto atributů nemusí být všem ihned zřejmý, a proto nebyly zařazené do šablony. Navíc aktuálně nejsou ve formátu IDEA příliš využívány. Vynechány byly taky atributy *Router* a *Netname* z podobného důvodu (v úvahu připadá pouze vložení do atributu text a jejich použití může být opět nejasné). Poslední změnou vůči objektům *Source* a *Target* je rozdělení atributu *Ref* do dvou atributů *Vulnerability* a *Reference*. Hodnoty, které se mohou vyskytnout v objektu *Ref*, mohou nabývat jak odkazů na různé zranitelnosti systému, tak odkazy na samotné dodatečné informace. Aby bylo dosaženo srozumitelnější převeditelnosti, byl tento atribut rozdělen do dvou atributů, kdy alespoň jeden z nich je přímo mapovatelný na MISP atribut (*vulnerability*) a dochází tedy k čistčímu převodu. Šablona definující objekt *Target* vypadá totožně až na pár výjimek. Kdykoliv je u atributu uvedeno, že se jedná o zdroj, tak v případě objektů *Target* se jedná o oběť (takže např. typ atributu *ip-src* byl nahrazen za *ip-dst* a byl přepsán popis tohoto atributu).

Stejným způsobem byla vytvořena i šablona pro IDEA objekty *Attach*. U této šablony bylo potřeba navíc vyřešit postup převodu atributu *Hash*. Jeho hodnota totiž obsahuje jak název hešovací funkce, tak i výsledný řetězec. MISP ale pro každou hešovací funkci (kterou podporuje) vytvořil vlastní MISP atribut. Proto výsledná šablona pro objekt *Attach* obsahuje, místo jednoho samotného atributu *Hash*, několik atributů pojmenovaných podle

hashovací metody, které referují ke každé hashovací metodě, kterou MISP platforma podporuje (podporuje všechny nejpoužívanější). Každá hodnota umístěna v poli *Hash* je při konverzi vyjmuta, je odebrán název hešovací metody a dvojtečka a samotná hodnota je poté vložena do správného MISP atributu. Jistě by šlo v šabloně vytvořit atribut *Hash* a vkládat hodnoty stejným způsobem jako se to dělá ve formátu IDEA, ale při konverzi je vhodné se co nejvíce přiblížit cílovému formátu. Dále atribut *type* uvnitř objektů *Attach* není konvertován do samostatného atributu, jelikož samotný takový tag bez dokumentace jednotlivých hodnot (u formátu IDEA je uvedena) nesdělí mnoho, proto pokud je uveden, je přidán na začátek atributu *Note*. Ostatní atributy objektů *Attach* jsou použity ve stejné podobě, jako ve formátu IDEA.

3.2.3 Finalizace konverze

Poslední částí IDEA události, která zbývá překonvertovat, jsou objekty *Node* sdělující informace o detekčním zařízení. Z objektů *Node* jsou tedy vybrány informace uvedeny v atributech *Name* a *SW*. Obě tyto informace jsou převedeny do samostatných MISP atributů z kategorie *Internal reference* a typu *comment*, jelikož mezi MISP atributy není korespondující typ atributu. Jména detektorů jsou uvozeny v komentáři jako „Detector name(s)“ a detekční software je poté uvozen jako „Detection software(s)“.

3.3 Návrh konverze formátu MISP do formátu IDEA

Dříve bylo zmíněno, že MISP platforma umožňuje vytvářet několik druhů událostí. Pokud má být určitá událost převedena do formátu IDEA, je důležité zaručit, že se jedná o druh události, pro který je formát IDEA určen. Takovou událostí je událost, která je generována systémy IDS. Platforma MISP kategorizuje události pomocí tagů a v kapitole 3.1 byly pro převod kategorie IDEA do tagů vybrány jmenné prostory ECSIRT a RSIT. Tyto jmenné prostory byly zvoleny, jelikož jsou dobře mapovatelné na kategorie formátu IDEA, ale také jsou jimi v MISP platformě označovány události generované systémy IDS, které jsou určeny pro převod do formátu IDEA. Pokud tedy má být některá událost pocházející z platformy MISP překonvertována, je prvně zkontrolováno, jestli je otagována alespoň jedním tagem, který pochází z těchto dvou jmenných prostorů. Některé ostatní jmenné prostory mohou být také občas využity pro klasifikaci síťové události nebo incidentu. Jelikož je jejich primární využití určeno pro jiný druh událostí, musí být tyto jmenné prostory vyloučeny z konverze, protože nelze přesně určit tyto výjimečné případy, kdy je daná událost síťovým incidentem nebo událostí a to ani v případě, kdy by došlo k filtraci pouze části jmenných prostorů (predikátů).

Pokud alespoň jeden z tagů uvedených v události patří do vybraných jmenných prostorů, dojde ke konverzi. Pokud je na základě použitých tagů rozhodnuto, že je MISP událost převeditelná do formátu IDEA, je nutné převést i ostatní atributy události. Při tvorbě nové IDEA události je nutné vyplnit povinné atributy. Atributy *ID* (uuid verze 4) a *Format* (zatím jediný *IDEA0*) lehce vyplněny. Poté jsou po jednom převedeny všechny tagy použité ke klasifikaci události, pokud patří do vyčleněných jmenných prostorů (konverze probíhá na základě konverzní tabulky 3.2). Jsou postupně vkládány do pole kategorií formátu IDEA *Category*. Pokud některý z použitých tagů nepatří do vyčleněných jmenných prostorů, je ignorován.

Posledním povinným atributem IDEA události je atribut *DetectTime*. Každá MISP událost obsahuje atribut *date*, který koresponduje k datu detekce události. Ovšem IDEA

atribut *DetectTime* vyžaduje kromě přesného data i přesný čas detekce události, který už ale v MISP atributu *date* ani v žádném jiném nenalezneme (MISP událost obsahuje více časových značek, ale ty přímo nesouvisí s časem detekce události). Nabízí se dvě možnosti řešení tohoto problému:

- Nastavit čas uvnitř atributu *DetectTime* na 00:00:00.
- Iterovat přes všechny časové značky MISP události (každý atribut i objekt mají svojí vlastní časovou značku) a pokud je časová značka ze stejného dne jako date, poté nejstarší časová značka bude vložena do atributu *DetectTime*. Nicméně je nutné brát v potaz, že tyto časové značky u atributů i objektů přímo nedefinují, kdy byl tento atribut (nebo událost celkově) detekován, ale čas, kdy byl vytvořen nebo naposledy upraven. I přesto je ale velice pravděpodobné, že nejstarší časová značka bude svou hodnotou blíž k detekci, než prostý čas 00:00:00, protože událost je do MISP instance často vložena co nejdříve po detekci a její atributy ihned poté.

Událost formátu MISP	**Událost formátu IDEA**
"**date**": "<hodnota>"	"**DetectTime**": "<hodnota + čas 00:00:00 nebo čas z jiné časové značky stejného data>"
"**Tag**": {'name': <hodnota>}	"**Category**": [<hodnoty kategorií, které byly převedeny pomocí konverzní tabulky kategorií (tabulka 3.2)>]
"**info**": "<hodnota>"	"**Description**": "<hodnota>"
	"**Note**": "This event was converted from MISP instance <address>."

Tabulka 3.3: Základní konverzní tabulka MISP formátu do formátu IDEA

Primárně je aplikována druhá možnost (iterace přes všechny časové značky). V některých výjimečných případech nemusí událost obsahovat žádný atribut (i taková událost se konvertuje, alespoň informuje o detekci události) nebo jsou atributy vloženy později (třeba až další den). Tehdy je využita první možnost, vložení času 00:00:00. Ostatní časové značky MISP události nejsou použity, protože se nedají využít pro konverzi. Atribut MISP události *info* je překonvertován do IDEA atributu *Description*. Do atributu *Note* je vložen popisek informující o tom, z jaké instance byla MISP událost konvertována a pokud MISP událost obsahuje nějaký objekt uvnitř atributu *Galaxy*, je z něj vyjmuto jméno a vloženo do atributu *Note* s popiskem "Event additional keywords". Tato část konverze je znázorněna pomocí tabulky 3.3.

Po vyplnění obecných atributů události IDEA jsou konvertovány konkrétní MISP atributy a objekty vložené do události. Každý MISP atribut je definován vlastní kategorií a typem, které dohromady určují druh hodnoty, kterou atribut nese. Vzhledem k převodu do formátu IDEA hledá konverze převážně atributy, které popisují síťové informace. To jsou všechny atributy obsahující hodnoty IP adres, portů, doménových jmen, protokolů atd. Kdykoliv je detekován atribut nesoucí jednu z těchto informací, je tato informace vložena do IDEA objektů *Source* nebo *Target* IDEA zprávy. Rozdíl mezí útočníkem (*Source*)

MISP atribut	**Událost formátu IDEA**
"**ip-src**": "<hodnota>"	"**Source**": [{"IP4/6": ["<hodnota>"]}]
"**ip-dst**": "<hodnota>"	"**Target**": [{"IP4/6": ["<hodnota>"]}]
"**ip-src|port**": "<IP|port>"	"**Source**": [{ "IP4/6": "<IP>", "Port": [<port>] }]
"**ip-dst|port**": "<IP|port>"	"**Target**": [{ "IP4/6": "<IP>", "Port": [<port>] }]
"**src-port**": "<hodnota>"	"**Source**": [{"Port": [<hodnota>]}]
"**dst-port**": "<hodnota>"	"**Target**": "[{"Port": [<hodnota>]}]"
"**protocol**": "<hodnota>"	"**Source**": [{"Proto": ["<hodnota>"]}], "**Target**": [{"Proto": ["<hodnota>"]}]
"**src-as**": "<hodnota>"	"**Source**" [{"ASN": [<hodnota>]}]
"**dst-as**": "<hodnota>"	"**Target**" [{"ASN": [<hodnota>]}]
"**hostname-src**": "<hodnota>"	"**Source**": [{"Hostname": ["<hodnota>"]}]
"**hostname-dst**": "<hodnota>"	"**Target**": [{"Hostname": ["<hodnota>"]}]
"**domain**": "<hodnota>"	"**Target**": [{"Hostname": ["<hodnota>"]}]
"**email-src**": "<hodnota>"	"**Source**": [{"Email": ["<hodnota>"]}]
"**email-dst**": "<hodnota>"	"**Target**": [{"Email": ["<hodnota>"]}]

Tabulka 3.4: Konverze MISP atributů obsahujících síťové informace

a obětí (*Target*) je v MISP formátu definován převážně uvnitř typu atributu pomocí prefixů "src"(source) a "dst"(destination = target).

Ve formátu IDEA lze seskupovat informace o jednotlivých útočnících a obětech. Samotné MISP atributy tento druh seskupování nepodporují, a proto informace extrahované z těchto atributů nejsou seskupovány ani ve výsledné IDEA zprávě. Toto seskupení podporují až MISP objekty. Tyto objekty obsahují mnohdy více specifické informace o síťovém provozu. Každý objekt obsahuje atributy, které jsou totožné s obyčejnými MISP atributy, akorát je mezi nimi uchována vazba pomocí daného objektu, který je seskupuje dohromady a dodává těmto atributům dodatečnou informaci. MISP nabízí předdefinované šablony pro tvorbu objektů, kdy po jejich manuální analýze byly vybrány ty, které jsou určeny pro přenos síťových informací. Mezi tyto objekty patří *ip-port*, *domain-ip*, *netflow*, *network-connection*, *network-socket*. V tabulce 3.4 je znázorněna konverze většiny atributů přenášejících síťové informace. Jako MISP atributy jsou uváděny i zástupná jména atributů (*object_relation*) některých převáděných MISP objektů, aby byla ukázka bohatší.

MISP atribut	**Událost formátu IDEA**
"**filename**": "<hodnota>"	"**Attach**": [{"Handle": "att1", "Filename": "<hodnota>" }]
"**filename|<hash_metoda>**": "<jmeno>|<hash_hodnota>"	"**Attach**": [{ "Handle": "att1", "Filename": "<jmeno>", "Hash": ["<hash_metoda>:<hash_hodnota>"] }]
"**<hash_metoda>**": "<hodnota>"	"**Attach**": [{ "Handle": "att1", "Hash": ["<hash_metoda>:hash_hodnota"] }]
"**url**": "<hodnota>"	"**Attach**": [{ "Handle": "att1", "Ref": ["<hodnota>"] }]

Tabulka 3.5: Konverze MISP atributů obsahujících informace o přílohách

Pro účely konverze formátu IDEA do formátu MISP byly vytvořeny šablony objektů *Source*, *Target* a *Attach* formátu IDEA. Tyto objekty je ale vhodné umožnit konvertovat i zpět. Jejich konverze je díky jejich skoro totožné definici přímočará. U takového objektu stačí u každého použitého MISP atributu zkontrolovat zástupné jméno (*object-relation*), které je, až na pár výjimečných případů popsaných u tvorby těchto šablon (kapitola 3.2), stejné jako název atributu, pod kterým má být zpracovávaný atribut být vložen. Ovšem ne všechny organizace budou tyto objekty v MISP instanci využívat, proto je stále nutné kontrolovat i samostatné atributy a jiné typy objektů, pro docílení převodu co největšího možného počtu relevantních informací. Pro každý samostatný převáděný MISP atribut je ve formátu IDEA vytvořen samostatný objekt (převádí-li se tyto atributy do objektů *Source*, *Target* nebo *Attach*), jelikož mezi samostatnými MISP atributy nelze určit žádná vazba a proto je nutné je takto samostatně i převádět, aby nedošlo k špatné interpretaci. Výjimkou jsou složené MISP atributy jako např. *ip-src/port*, který obsahuje dvě hodnoty, mezi kterými je jasná vazba, a proto jsou obě tyto hodnoty vložené do jednoho IDEA objektu.

Mezi další konvertovatelné hodnoty patří informace o souborech a hodnoty hashovacích algoritmů. Uvažujeme-li soubory, můžeme kontrolovat atributy nebo objekt vytvořený pomocí šablony MISP objektu *file*. V případě hashovacích algoritmů nás zajímají atributy, které nesou název nějaké hashovací funkce (sha256, md5 atd., viz tabulka 3.5). Tyto informace se ve formátu IDEA ukládají do objektů *Attach*. Do tohoto objektu se ukládají i hodnoty URL, pokud jsou nějaké v MISP události obsaženy v atributech typu *url*.

Informace o detektoru a autorovi zveřejněné události je nutné také konvertovat. Atribut MISP události *orgc* obsahuje informace o organizaci, která vytvořila událost. Jméno orga-

Událost formátu MISP	Událost formátu IDEA

"**Orgc**": {
...
"name": "CIRCL",
"id": "<id>",
...
}
	"**Node**": [{
	"Name": "ext.misp.circl",
	"SW": ["MISP", "MISP-to-IDEA"],
	"Type": ["Relay"],
	"Note": "MISP organization name (UUID):
	CIRCL (<id>)"
	}

Tabulka 3.6: Konverze atributu *orgc* události formátu MISP do formátu IDEA

nizace a její *id* (může sloužit pro zpětné dohledání organizace) je vloženo do objektu *Node*. Jako *Name* objektu *Node* je zvoleno *„ext.mixp.circl“*, jelikož konektor je zatím testován s organizací CIRCL. Doplněny jsou i Z těchto dvou atributů jsou překonvertovány vždy dvě informace, a to jméno organizace a id organizace, pro jejich možné zpětné dohledání. Tyto informace se ve formátu IDEA vkládají do objektů *Node*. Dále jsou staticky vyplněny atributy *SW*, definující typ software, který detekoval (v tomto případě konvertoval) událost a atribut *Type* (*„Relay“* označuje, že se jedná o zprostředkovatele a ne o vlastní detektor). Výsledek konverze je znázorněn v tabulce 3.6.

3.4 Návrh konverze formátu STIX do formátu IDEA

Objekty formátu STIX konverzi výrazně neovlivňují, slouží pouze jako obálka. V budoucích verzích platformy C3ISP je v plánu formát STIX více obohatit dodatečnými informacemi, které doplní dodatečné komponenty platformy. Nyní se ale většina užitečných a konvertovatelných informací nachází uvnitř událostí formátu CEF, které jsou zanořeny uvnitř objektu formátu STIX *observed-data*. Jednotlivé události spolu ne vždy úplně souvisí, proto je každá událost formátu CEF překonvertována do samostatné události formátu IDEA.

Do každé události formátu IDEA je nutné překonvertovat kategorii události a čas její detekce. Čas detekce se nachází uvnitř atributu *rt* rozšíření formátu CEF. V některých vzorových událostech platformy C3ISP byl ale tento atribut nahrazen atributem *date*, který je uveden ještě s atributem *dtz*, který přenáší informaci o časové zóně atributu *date*. Tyto atributy jsou konvertovány do atributu *DetectTime* formátu IDEA. Ve většině případů je kategorie události konvertována z atributu *cat*. Kategorie, které se mohou objevit v atributu *cat* jsou rozdílné podle detekčního systému, který detekuje události a exportuje je do formátu CEF. Bohužel v průběhu vývoje nebyly poskytnuty informace o detekčních zařízeních, které využívá platforma C3ISP. V jedné z pěti dostupných událostí platformy C3ISP byla použita kategorie *Trojan*. Dále jsem se pokusil dohledat různé příklady[1] kategorií, které mohou být používány detekčními zařízeními, které exportují události do formátu CEF. Povedlo se mi najít pouze několik běžných výrazů vyjadřující kategorii události, např. *malware* nebo *bot*. Na základě těchto tří výrazů jsem se pokusil navrhnout několik dalších výrazů, které by mohly určovat kategorii události, která je konvertovatelná do kategorií

[1] http://docs.trendmicro.com/all/ent/ddi/v5.1/en-us/ddi_5.1_sg.pdf

formátu IDEA. Seznam těchto výrazů a jejich konverze do kategorií formátu IDEA ukazuje tabulka 3.7. Formát IDEA definuje více kategorií, které nejsou uvedeny v tabulce 3.7. Jedná se o pouhý návrh, který může být lépe prozkoumán až při analýze většího množství událostí platformy C3ISP. V některých případech je kategorii nutné určit na základně proprietární dokumentace. Příkladem takové kategorie je hádání hesel hrubou silou, která je rozpoznána pomocí prefixu 'ssh-' v části *name* hlavičky formátu CEF (např. ssh-2018-12-03T10:30:54). Taková událost je vytvořena ze záznamů systémového souboru *„/var/log/auth.log“*. Pokud atribut *outcome* nese hodnotu *failure*, lze určit kategorii *Attempt.Login*. V případě hodnoty *accepted* by se dalo uvažovat o kategorii *Intrusion.UserCompromise*, ale z logu souboru tohoto typu nelze přesně říci, zda šlo opravdu o narušení uživatelského účtu.

Kategorie formátu CEF	**Kategorie formátu IDEA**
trojan	**Malware**.Trojan
malware	**Malware**
ransom(ware)	**Malware**.Ransomware
virus	**Malware**.Virus
worm	**Malware**.Worm
spyware	**Malware**.Spyware
scan	**Recon**.Scanning
exploit	**Attempt**.Exploit
bot	**Intrusion**.Botnet
dos	**Availability**.DoS
ddos	**Availability**.DDoS

Tabulka 3.7: Konverzní tabulka kategorií formátu CEF do kategorií formátu IDEA

Po konverzi povinných atributů formátu IDEA jsou zpracovány všechny ostatní atributy rozšíření formátu CEF, které jsou konvertovatelné do formátu IDEA. Vzorové události platformy C3ISP využívaly ve většině případů pouze několik atributů určených pro přenos síťových informací, ale navrženy k převodu byly všechny relevantní atributy rozšíření formátu CEF (viz tabulka 3.8) na základě dokumentace tohoto formátu [1].

U atributů, které přenáší časové informace je nutné kontrolovat jejich formát. Časové informace formátu CEF mohou nabývat 2 verzí. První verze je tzv. unixový čas[2]. Jedná se o číslo, které indikuje počet sekund od 00:00:00 1. ledna 1970. Formát IDEA ale všechny časové informace uchovává v podobě textového řetězce data a času (navíc v podobě UTC[3]). Tuto první verzi zápisu je nutné překonvertovat do formátu, který používá formát IDEA. Druhá verze zápisu formátu CEF už správně koresponduje k formátu IDEA, není potřeba dodatečná konverze.

[2] https://en.wikipedia.org/wiki/Unix_time
[3] https://en.wikipedia.org/wiki/Coordinated_Universal_Time

Atribut rozšíření formátu CEF	**Událost formátu IDEA**
src=<hodnota>	"**Source**": [{"IP4": ["<hodnota>"]}]
dst=<hodnota>	"**Target**": [{"IP4": ["<hodnota>"]}]
spt=<hodnota>	"**Source**": [{"Port": [<hodnota>]}]
dpt=<hodnota>	"**Target**": [{"Port": [<hodnota>]}]
cnt=<hodnota>	"**ConnCount**": <hodnota>
shost=<hodnota>	"**Source**": [{"Hostname": ["<hodnota>"]}]
dhost=<hodnota>	"**Target**": [{"Hostname": ["<hodnota>"]}]
smac=<hodnota>	"**Source**": [{"MAC": ["<hodnota>"]}]
dmac=<hodnota>	"**Target**": [{"MAC": ["<hodnota>"]}]
"**fname**"=<jmeno_souboru> "**fsize**"=<velikost_souboru>	"**Attach**": [{ "Handle": "att1", "Filename": ["<jmeno_souboru>"] "Size": "<velikost_souboru>" }]
in=<hodnota>	"**ByteCount**": <hodnota>
out=<hodnota>	"**ByteCount**": <hodnota>
msg=<hodnota>	"**Description**": <hodnota>
start=<hodnota>	"**EventTime**": "<hodnota>" (+ případná konverze formátu času)
end=<hodnota>	"**CeaseTime**": "<hodnota>" (+ případná konverze formátu času)

Tabulka 3.8: Konvere atributů rozšíření formátu CEF do formátu IDEA

Poslední částí konverze je vyplnění objektu *Node* (viz výpis 3.2), který je vyplněn téměř staticky. Pouze v případě detekce hodnoty *log_analyzer* v hlavičce formátu CEF je do atributu *Type* přidána hodnota *Log*.

Většina bezpečnostních událostí platformy C3ISP je uchována ve formátu STIX, který obaluje jednotlivé události formátu CEF. Některé události jsou ale uloženy ve vlastních objektech formátu STIX, které vytvořila platforma C3ISP. Mezi tyto objekty patří objekt *report* a objekt *email*. Objekt *report* je používán pro reprezentaci událostí, které obsahují

```
"Node": [{
    "Name": "cz.cesnet.ext.c3isp",
    "SW": ["C3ISP", "STIX-to-IDEA"],
    "Type": ["External", "Relay"]
}]
```

Výpis 3.2: Objekt Node překonvertované události formátu IDEA

převážně textový popis události. Může obsahovat informace, které jsou konvertovatelné do formátu IDEA, ale často by musely být vyjmuty z textových řetězců. Není příliš jednoduché určit pozici užitečných hodnot uvnitř textu a stejný problém nastává s určením kategorie události. Objekt *email* slouží pro přenos informací o potenciálně škodlivém emailu. Opět ale není možné určit o jaký druh nebezpečí se jedná. Tato informace není uvedena ani v textovém popisu, protože platforma C3ISP plánuje tento objekt obohatit více informacemi dodatečným zpracováním, které zařídí komponenty platformy, které budou v budoucnu implementovány.

Kapitola 4

Implementace konverzních knihoven a konektorů

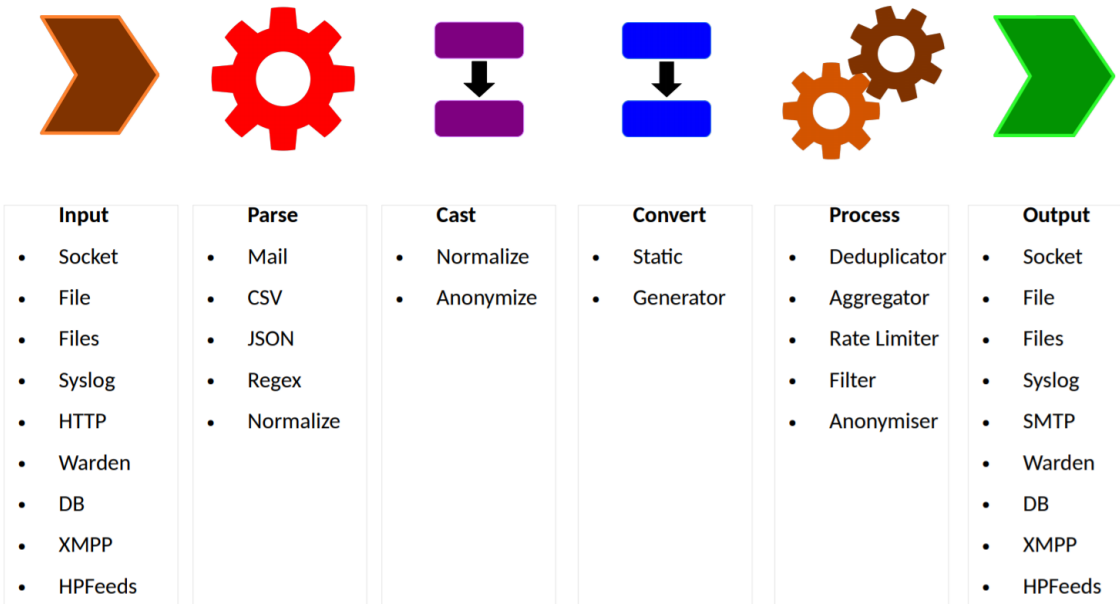
V kapitole 3 byl popsán návrh konverze, podle kterého jsou implementovány konverzní knihovny. Ovšem implementace konverzní knihovny je pouze jedna část celého procesu konverze událostí. Konverzní knihovna umí načtenou událost pouze překonvertovat a na výstupu vrátit výsledek konverze. Ještě je ale potřeba implementovat obslužné konektory, které umí události správně načítat ze zdroje a rozebrat nosný formát (např. JSON). Někdy je navíc potřeba určité předzpracování, např. anonymizace dat, pokud organizace nechce zveřejňovat celý obsah události (některá data mohou být pro organizaci citlivá, ale byla by škoda kvůli tomu zahodit celý obsah události). Až po tomto bodě proběhne proces konverze. Poté je možné na události udělat finální drobné úpravy, například agregace a deduplikace událostí. Tento celý proces zpracování událostí znázorňuje obrázek 4.1. V této kapitole je proto popsána nejen implementace konverzních knihoven, ale také způsob implementace konverzních konektorů, které obsluhují konverzi událostí použitím konverzních knihoven. Konverzní knihovny i konektory jsou implementovány v programovacím jazyce Python, který je vhodný pro zpracování textových dat, kterými jsou např. bezpečnostní události.

4.1 Implementace konverze mezi formáty IDEA a MISP

4.1.1 Implementace konverzní knihovny

Centrálním bodem konverze je knihovna ***misp.py***, která obsahuje třídu **IdeaToMisp** určenou pro konverzi z formátu IDEA do MISP formátu a třídu **MispToIdea** určenou pro opačnou konverzi. Obě třídy jsou umístěné v jednom souboru, jelikož využívají společný konverzní slovník určený pro konverzi IDEA kategorií a MISP taxonomií a zároveň takto vzniká jednotná knihovna sloužící pro konverzi mezi formátem IDEA a MISP formátem. Konverzní slovník kategorií a taxonomií je důležitý, bez něho se konverze neobejde. Jeho přímé využití slouží pro konverzi IDEA kategorií do MISP taxonomií, ale po lehké úpravě je možné ho využít i pro konverzi MISP taxonomií do IDEA kategorií. Dále už je veškerý kód umístěn a rozdělen do dříve jmenovaných dvou tříd (**IdeaToMisp** a **MispToIdea**).

Obě tyto třídy jsou implementovány způsobem, aby se mohly jejich instance použít k opětovné konverzi bezpečnostních událostí a nebylo nutné pro každou konverzi vytvářet novou instanci dané třídy. Konverze se vždy spustí voláním jediné instanční metody. Název této metody se odvíjí od druhu konvertoru a sděluje výsledek konverze, tedy do jakého



Obrázek 4.1: Architektura konektorů (postupuje zleva doprava) (převzato z [10])

formátu bude událost konvertována. V případě konverze z formátu IDEA do MISP formátu je uvnitř korespondující třídy implementována metoda **to_idea(...)**, která převede IDEA událost do MISP formátu. Druhá třída pro převod umožňuje použít instanční metodu **to_misp(...)**. Obě tyto instanční metody převezmou jako argument událost, která je určena ke konverzi a je možné předat i několik konfiguračních argumentů, které mohou lehce pozměnit předem definovaný proces konverze. Příkladem takového konfiguračního parametru metody je parametr *test*, který umožňuje u obou směrů konverze přidat kategorii nebo taxonomii výsledné události, která indikuje, že se jedná o testovací událost.

Instance těchto tříd také nabízejí možnost logování použitím instanční proměnné *_logger* (vyžívá python knihovnu *logging*). Konvertor využívá pouze úrovně logování ERROR a DEBUG. Hlášení úrovně ERROR jsou použity v případě neočekávaného stavu a úroveň DEBUG nabízí podrobnější výpis průběhu konverze. Výchozí úroveň logovacího modulu je úroveň INFO (logovací modul loguje všechny zprávy kromě úrovně DEBUG), konektory používající konverzní knihovnu umožňují svým parametrem *--verbose* snížit úroveň na DEBUG (logovací modul poté loguje všechny zprávy).

4.1.2 Konverze událostí IDEA do formátu MISP

Konverze třídy **IdeaToMisp** začíná vytvořením MISP události pomocí knihovny PyMISP[1], která umožňuje jednodušší zpracování MISP událostí. Poté jsou nastaveny základní informace o události, které zahrnují úroveň distribuce, úroveň hrozby, datum události, info o události a především vložení správných tagů na základně konverzní tabulky 3.2. Všechny tyto informace jsou konvertovány z kořene IDEA události.

Poté jsou zpracovány všechny objekty události IDEA, které se nachází uvnitř atributů *Source*, *Target*, *Attach* a *Node*. Zpracování je docíleno využitím knihovny PyMISP a šablon MISP objektů, které byly vytvořeny v kapitole 3.2. V případě objektů *Source* a *Target* objektů jsou podle šablon vytvořeny MISP objekty *network-source* a *network-*

[1] https://github.com/MISP/PyMISP

target pomocí třídy **MISPObject** knihovny PyMISP. Instance této třídy nabízí metodu **add_attribute(...)**, které stačí předat klíč nebo-li jméno atributu, který bude vložen do události a jeho hodnotu. Jelikož jsou skoro všechna zástupná jména atributů v použitých šablonách stejná, dojde k přímočarému zpracování těchto objektů a stačí ošetřit pouze atributy těchto objektů, jejichž jména uvnitř šablony se liší oproti názvům formátu IDEA. V tomto případě se jedná pouze o IDEA atribut *Ref* a je překonvertován podle obsažené hodnoty buď na atribut typu *vulnerability* nebo *link*.

Stejným způsobem jsou zpracovány i IDEA objekty *Attach*, speciálně ošetřen musí být atribut *Ref* a navíc ještě hodnoty uložené v atributu *Hash*. V MISP formátu jsou hodnoty hashovacích funkcí ukládány do MISP atributů, jejichž typ je určen podle názvu hashovací funkce. Kdykoliv je tedy zpracována hodnota IDEA atributu *Hash*, je nutné její hodnotu rozdělit na dvě části, kdy první část určuje název hashovací funkce a druhá část nese její hodnotu. Podle názvu hashovací funkce se vytvoří MISP atribut, který je poté i s hodnotou přidán do korespondujího MISP objektu. IDEA objekty *Node* jsou konvertovány do samostatných MISP atributů typu *comment* a jsou přidány do nově vznikající MISP události.

4.1.3 Konverze událostí MISP platformy do formátu IDEA

Třída **MispToIdea** knihovnu PyMISP nevyužívá, jelikož by práci s MISP událostmi výrazně neulehčila. Na začátku konverze dojde ke kontrole tagů uvnitř převáděné MISP události. Pokud MISP událost neobsahuje ani jeden tag z jmenných prostorů ECSIRT nebo RSIT, je konverze u konce, jelikož taková událost není převáděna (viz kapitola 3.3). Pokud obsahuje alespoň jeden tag z požadovaných jmenných prostorů, jsou všechny takové tagy překonvertovány do IDEA kategorií na základě konverzní tabulky 3.2. Konverze pokračuje zpracováním obecných informací, mezi které patří vyplnění IDEA atributů *CreateTime*, *DetectTime*, *Note* a objektů *Node*. Konverze umožňuje přidat původní MISP událost jako přílohu v IDEA objektu *Attach*.

Následuje zpracování samostatných MISP atributů převáděné události. Podle jejich typu jsou překonvertovány do formátu IDEA podle konverzní tabulky 3.4. Je nutné kontrolovat atributy IP adres, jelikož MISP formát používá jeden typ atributů pro IP adresy verze 4 i 6, ale formát IDEA jejich verzi rozlišuje.

Zpracování objektů MISP události se dělí podle jména konvertovaných objektů. Pokud se jedná o objekt, který je vytvořen podle již vlastní šablony určené přímo pro IDEA objekty *Source*, *Target* a *Attach*, je konverze snazší. Atributy uvnitř objektu obsahují speciální atribut *object_relation*, který nese zástupné jméno atributu MISP objektu, které byly navrhnuty stejně jako jména atributů objektů formátu IDEA. Po ošetření několika výjimek v podobě atributů MISP objektu *Vulnerability* a *Reference*, které jsou převedeny do IDEA atributu *Ref*, je konverze těchto objektů hotová. V případě konverze objektu *Attach* je důležité ošetřit hodnoty hashovacích funkcí. Všechny takové hodnoty se konvertují do formátu IDEA jako pole hodnot uložené pod klíčem *Hash*. Je potřeba spojit název hashovací funkce a výslednou hodnotu do jednoho řetězce a až tuto podobu uložit. Atributy MISP objektu *Vulnerability* a *Reference* jsou vkládány do pole referencí *Ref*. U ostatních MISP objektů (pouze které jsou převáděny) se kontrolují všechny atributy a pokud atribut přenáší užitečnou hodnotu (podle konverzní tabulky 3.4), je tato hodnota překonvertována do korespondujícího objektu formátu IDEA.

4.1.4 Implementace konverzních konektorů

Implementovanou konverzní knihovnu využívají dva konektory, kdy každý je určen pro jeden směr konverze. Konektor obsluhující konverzi IDEA formátu do MISP formátu (***idea_to_misp.py***) načítá IDEA události ze složky, v které je každá událost umístěna v samostatném souboru. Události určené ke konverzi jsou přijímány ze systému Warden. Konektor ***idea_to_misp.py*** sleduje obsah složky a pokud přibyde nová IDEA událost, konektor ji načte a překonvertuje ji pomocí konverzní knihovny do MISP události. Překonvertovaná událost je ihned po konverzi vložena do MISP instance, ke které je konektor připojen. Vkládání nových událostí do MISP instance je realizováno pomocí knihovny PyMISP, která umožňuje udržet spojení s MISP instancí a je uzpůsobena k jednoduché správě MISP instance.

Běh konektoru je konfigurovatelný spouštěcími parametry skriptu. Povinnými parametry jsou cesta k adresáři, z kterého bude konektor načítat nové IDEA zprávy a poté URL adresa MISP instance, do které budou překonvertované události vkládány. Posledním povinným parametrem je tzv. API klíč, který slouží pro autentizaci uživatele, pod kterým budou nové události vkládány. Tento klíč je dohledatelný v uživatelském rozhraní MISP instance. Po přihlášení uživatele do MISP instance je k nalezení v hlavním menu pod položkou **Event Actions** a její možností **Automation**. Předpokladem takového konektoru je také konstantní běh, proto je možné spustit tento konektor jako proces na pozadí (parametr *--daemonize*). Ve výchozím nastavení konektor načítá nově vzniklé události ze složky, ale je možné zpracovat i události, které se ve složce nacházely ještě před spuštěním konektoru (parametr *--include*). Každá zpracovaná událost je po konverzi ze složky smazána. Konektor podporuje základní úroveň logování, aby bylo možné kontrolovat běh konektoru a také jednodušeji objasnit neobvyklé chování konektoru. Logovací zprávy jsou uloženy ve složce, v které byl spuštěn konektor v souboru ***idea_to_misp.log***. Je možné konektoru předat i certifikát MISP instance, který umožňuje připojovat se k MISP instanci pomocí protokolu HTTPS. Bez něj jsou všechny operace nad MISP instancí prováděny pomocí protokolu HTTP.

Druhý konektor (***misp_to_idea.py***) obsluhuje opačný směr konverze z formátu MISP do formátu IDEA. Tento konektor se podle zvoleného intervalu (1-59 minut, implicitně 20 minut) dotazuje MISP instance na všechny bezpečnostní události, které byly do MISP instance přidány v aktuální den. Jelikož MISP API umožňuje se dotazovat pouze na události z určitého data (čas nemůže být specifikován), konektor se pokaždé dotáže na všechny události daného dne a vyfiltruje ty události, které ještě nebyly překonvertovány. Kdyby nedocházelo k takové filtraci, některé události by byly překonvertovány několikrát a docházelo by k duplicitě událostí. To je určitě nežádoucí chování konektoru. Pokaždé když konektor překonvertuje MISP událost, uloží si její *id*, aby byl při další iteraci (dotaz na nové události MISP instance) schopný rozeznat události, které už byly překonvertovány. Problém by mohl nastat, pokud by se ukončila činnost konektoru a ve stejný den se konektor opět spustil. Pro takové případy je nutné si *id* překonvertovaných událostí uchovat i mimo běh konektoru. Kdykoliv konektor dostane signál k ukončení, před opravdovým ukončením si uloží *id* všech překonvertovaných událostí do souboru, který se jmenuje ***misp_to_idea_converted_YYYY_MM_DD.txt***. Když je konektor spuštěn, nejprve se podívá po takovémto souboru ve své složce. Pokud nalezne nějaký aktuální soubor(podle data uvedeného v názvu souboru), načte si z něho všechny identifikátory do paměti, aby mohl opět správně rozhodovat o tom, zda již byla určitá událost překonvertována. Při spuštění také smaže všechny pomocné soubory, které jsou již neaktuální, např. z předchozího dne

nebo starší. Pokud by z jakéhokoliv důvodu byla vyžadována opětovná konverze již konvertovaných událostí (např. při testování), je nutné přerušit běh konektoru, smazat pomocný soubor, který uchovává *id* překonvertovaných událostí, a znovu spustit konektor. Konektor poté bude obsahovat prázdný list překonvertovaných událostí a může tedy dojít k opětovné konverzi. Je nutné mít na paměti, že tento mechanismus uchování seznamu identifikátorů v souboru nebude fungovat, pokud bude konektor ukončen jiným, nežli běžným způsobem.

Konektor po úspěšné konverzi novou IDEA událost uloží do souboru do požadované složky, jejíž systémová cesta byla předána jako povinný spouštěcí parametr konektoru (parametr *--path*). Pravděpodobně bude tato složka pod dohledem tzv. *warden_fileru*, který má za úkol odesílat nové události do systému Warden. Opět je nutné předat konektoru URL adresu MISP instance, z které budou události konvertovány a autentizační API klíč. Konektor je možné spustit jako proces na pozadí (parametr *--daemonize*). Uživatel také může změnit interval dotazování na nové události MISP instance pomocí parametru *--poll <value>* a v případě jakýchkoliv nejasností konektor podporuje mechanismus logování. Pomocí parametru konektoru *--verbose* může být úroveň snížena na úroveň DEBUG. Logovací zprávy jsou uloženy ve složce, v které byl spuštěn konektor v souboru ***misp__to__idea.log***.

4.2 Implementace konverze z formátu STIX do formátu IDEA

Centrálním bodem konverze je knihovna ***stix.py***. Před počátkem konverze musí být vytvořena instance třídy **StixToIdea**, která umožňuje a řídí samotnou konverzi. Tato instance je uzpůsobena k opakovanému použití, není nutné pro každou konverzi vytvářet novou instanci. Instance třídy **StixToIdea** se postupně zanoří do úrovně události STIX, v které může rozhodnout o provedení konverze. Některé události platformy C3ISP nejsou konvertovány, jelikož přímo nekorespondují s formátem IDEA. Konverze je spuštěna voláním instanční metody **to__idea(...)**, kdy je instanci předána událost formátu STIX.

Pokud konvertor rozhodne o provedení konverze, postupně načítá a zpracovává jednotlivé události formátu CEF. Jelikož jednotlivé události formátu CEF jsou konvertovány do samostatných událostí formátu IDEA, instanční metoda **to__idea(...)** je implementována jako generátor, který poustupně generuje nové události formátu IDEA, které jsou zpracovány obslužným konektorem. Každá konverze události formátu CEF začne oddělením jednotlivých částí hlavičky. Následně jsou rozděleny atributy rozšíření formátu CEF. Hlavním oddělovačem atributů v rozšíření formátu CEF je znak mezery, někdy může ale být znak mezery uveden uvnitř hodnoty samotného atributu. Je proto důležité ošetřit takové stavy, aby nedocházelo oddělení atributů na základě špatné mezery a konverze neskončila chybou. V implementaci byl tento problém ošetřen krátkým regulárním výrazem, který vyhledá správné oddělovací mezery a nahradí je za nový oddělovač (znak '|').

Poté jsou jednotlivé hodnoty rozšíření konvertovány do formátu IDEA podle konverzní tabulky 3.8. Některé atributy rozšíření mohou obsahovat prázdný řetězec (prázdná hodnota). Takové atributy je také potřeba odfiltrovat, aby prázdné řetězce nebyly konvertovány do formátu IDEA. Následně je určena kategorie formátu IDEA a podle konfigurace přidána originální událost do přílohy objektu Attach. Konverzní knihovna podporuje logovaní pro podrobný přehled konverze.

4.2.1 Implementace konverzního konektoru

Konverzní knihovnu spouští konektor, který je vstupním bodem konverze. Konektor sleduje zvolenou složku, jejíž systémovou cestu načte pomocí vstupního atributu *--path*. Pokud se

v dané složce objeví nová událost, konektor ji ihned načte a předá konverzní knihovně ke zpracování.

Konektor podporuje řadu konfiguračních argumentů, např. argument *--daemonize* umožňuje spustit konektor jako službu na pozadí. Konektor podporuje i základní logovací hlášení, které je možné rozepsat více detailně, pokud je konektor spušten s argumentem *--verbose*. Logovací zprávy jsou uloženy ve složce, v které byl spuštěn konektor v souboru ***stix_to_idea.log***.

Kapitola 5

Testování konverzních knihoven a konektorů

Každý software je nutné v průběhu implementace testovat a ověřovat jeho správnou funkcionalitu. Implementované konverzní knihovny a konektory nejsou výjimkou. Cílem testování vzniklých konektorů a konvertorů je nalézt možné odchylky od předem navrhnuté analýzy převodu (kapitola 3). Tato kapitola shrnuje způsob testování funkčnosti konvertorů a konektorů, díky kterému bylo možné odhalit a následně opravit co nejvíce chybných částí implementace a dosáhnout požadované přesnosti implementace.

V průběhu implementace byly pro testování využity události formátu IDEA, které pocházejí ze systému Warden [11]. Warden je centrální sběrné místo bezpečnostních událostí provozované sdružením Cesnet. Pomocí skriptu ***warden_filer.py*** je možné se připojit k instanci systému Warden a automaticky přijímat bezpečnostní události ve formátu IDEA. Před samotným připojením k Wardenu je ale nutné zažádat o přístup a správně vyplnit konfigurační soubor, který obsahuje parametry, které při spuštění převezme skript ***warden_filer.py***. Celý postup je možné nalézt na webových stránkách[1] systému Warden.

K testovacím účelům konverze mezi formáty IDEA a MISP byl použit virtuální stroj MISP instance[2], který nabízí běh plnohodnotné MISP instance sloužící pro testovací účely. Do této instance MISP platformy byly pro plnohodnotné testování naimportovány vlastní šablony MISP objektů (popis jejich importu je přiložen v souboru README u konverzní knihovny).

5.1 Testování konverze mezi formáty IDEA a MISP

Po úspěšném propojení s platformou MISP byla vytvořena sada jednoduchých testů (***test_misp.py***) využívajících knihovny jazyka Python zvané *unittest*, která základním způsobem kontroluje správnost konverze. Pro účely testování konverze z formátu MISP do formátu IDEA byla vytvořena vzorová událost, která obsahuje kombinaci většiny MISP atributů a hodnot, které byly navrhnuty ke konverzi. Tato testovací událost byla uložena do souboru. Testovací událost byla poté manuálně vzorově překonvertována do formátu IDEA a výsledná překonvertovaná událost byla také uložena do souboru. Testovací třída **TestMispToIdeaConverter** (uvnitř souboru ***test_misp.py***) zahájí testovací proces načtením obou těchto testovacích událostí formátů IDEA a MISP. Vzorovou událost formátu

[1] https://warden.cesnet.cz/en/participation
[2] https://www.circl.lu/doc/misp/get-your-instance/

MISP poté překonvertuje pomocí implementované konverzní knihovny a výsledek konverze porovná s předem vzorově navrženou konverzí IDEA události (podle kapitoly 3.3), která byla při počátku testování také načtena. U porovnání jsou ignorovány atributy formátu IDEA, které jsou při každé konverzi odlišné (*CreateTime*, *ID* a *Note*). Tento způsob testování pomohl k odhalení a následné opravě několika chyb konverze, ovšem vzorová událost formátu MISP neobsahuje všechny možné kombinace MISP atributů a hodnot a proto je nutné doplnit tento způsob manuálním testováním.

Konverze z formátu IDEA do formátu MISP byla testována podobným způsobem. Ovšem událost ve formátu MISP a její MISP atributy obsahují několik atributů, které se více než samotné události týkají nastavení MISP instance, do které má být tato událost vložena (např. atributy *to_ids*, *disable_correlation* a další). Z celého MISP atributu jsou ale pro účely konverze podstatné pouze hodnoty uvnitř atributů *category*, *type* a value. Proto testovací třída **TestIdeaToMispConverter** testuje pouze korektní převedení těchto částí atributů a nekontroluje výsledek konverze s předem vzorově překonvertovanou událostí formátu MISP.

Velká část testování také probíhala manuálně. Ze systému Warden jsem získal několik desítek událostí formátu IDEA. Pokusil jsem se vybrat co nejvíce různorodé události, které mohly lépe otestovat různé části konverze. Vybrané události jsem poté předal konektoru, který je vkládal do virtuální instance platformy MISP. Události jsem nevložil všechny najednou, ale vkládal jsem je postupně, abych pořádně otestoval i funkčnost konektoru. Konektor byl spouštěn i s většinou kombinací různých spouštěcích parametrů. Konverzi vložených událostí jsem postupně kontroloval, nalezené odchylky od návrhu konverze jsem vždy ihned opravil. Manuální testování je velice důležité, protože automatickými testy nelze podchytit všechny možné kombinace atributů a hodnot formátu IDEA.

Při řešení problému filtrace správných událostí u konverze z platformy MISP do formátu IDEA (kapitola 3.1) byl proveden i jednoduchý experiment, který odhaloval množství přefiltrovaných událostí. V první fázi řešení problému, kdy byl vytvořen první návrh listu konvertovaných tagů a jimi označených událostí, byl proveden pokus o konverzi jeden rok starých událostí z reálné instance, ke které byl umožněn přístup v průběhu vývoje. Z přibližně dvou tisíc událostí bylo pomocí první verze listu tagů vyfiltrováno přibližně 15 % událostí, které byly takto určeny ke konverzi. Určitá část z nich byla ale stále špatného druhu. Při finálním návrhu řešení filtrace pomocí tagů z jmenných prostorů ECSIRT a RSIT byl proveden podobný experiment, který vykázal značné snížení počtu převedených událostí (méně než 5 %). Tento malý objem překonvertovaných událostí lze ale považovat za správně překonvertované, protože nedochází ke konverzi odlišného druhu události. To je velmi důležitý bod konverze. Navíc pomocí vzniklého konvertoru a konektorů je možné se připojit k jakékoliv MISP instanci. Instance která sloužila k testování při vývoji je podle mnoha analýz událostí určena spíše pro sdílení analýz malware a různých blacklistů IP adres. I přestože je v této konkrétní instanci množství překonvertovaných událostí malé, je předpoklad že u jiných MISP instancí, které se zaměřují na vyhledávané síťové bezpečnostní události, bude poměr konvertovaných událostí větší. Platforma MISP je aktuálně rozšířena mezi více než 1 000 organizací. Je velice pravděpodobné, že některé z nich budou MISP platformu používat pro sdílení takových událostí. V blízké budoucnosti je v plánu využít vzniklé konektory a konvertor k propojení s komunitou CIRCL, jejíž některé ostatní propojené komunity používají platformu MISP pro sdílení událostí, jejichž obsah koresponduje s formátem IDEA.

5.2 Testování konverze mezi formáty STIX a IDEA

Testování konverze mezi formáty STIX a IDEA probíhá podobným způsobem jako testování konverze do formátu MISP. Ovšem u tohoto testování byl velký problém s nedostatkem vzorových událostí určených ke konverzi. V průběhu implementace mi byl umožněn přístup pouze k pěti vzorovým událostem, z kterých dvě byly objekty typu *email* a *report*, které ani nejsou navrhnuty ke konverzi. Zbylé události mi poskytly alespoň základní přehled o tvaru událostí a jejich atributech, které jsem zkoušel pro testovací účely různě upravovat, aby vzniklo co největší možné množství kombinací. Vytvořené kombinace jsem předal konektoru, který je poté pomocí konverzní knihovny úspěšně překonvertoval do událostí formátu IDEA. I přesto ale není vhodné prozatím prohlásit konvertor z formátu STIX do formátu IDEA za plně otestovaný. Stavu kompletního otestování většiny možných scénářů lze dosáhnout až v případě poskytnutí většího množství bezpečnostních událostí platformy C3ISP. Celkový proces testování byl také kratší, protože se jedná pouze o jednosměrnou konverzi.

Poskytnuté události formátu STIX byly vzorově překonvertovány a výsledky vzorové konverze jsou uloženy do jednotlivých událostí formátu IDEA. Událostí formátu IDEA je více, jelikož z jedné události formátu STIX, která obaluje více událostí formátu CEF, vznikne několik událostí formátu IDEA. Testovací třída **TestStixToIdeaConversion** poté načte zdrojovou událost formátu STIX a spustí její konverzi. Výstup konverze je porovnán s vzorově překonvertovanými událostmi formátu IDEA. Z testu jsou vynechány atributy formátu IDEA *ID* a *CreateTime*, protože jejich hodnoty jsou pokaždé odlišné. Testy jsou úspěšné, pokud se vzorové a právě překonvertované události shodují.

5.3 Výkonnostní testování konektorů a konvertorů

V poslední fázi vývoje konektorů a konvertorů došlo také k testování jejich výkonu. Nejdůležitějším výkonnostním parametrem konektorů je počet událostí za jednotku času, které dokážou překonvertovat. Výpočtu tohoto parametru jsem se pokusil dosáhnout maximálním zatížením konektorů, při kterém jsem pomocí knihovny jazyka Python zvané *cProfile* měřil čas, který je potřebný k převodu určitého počtu událostí. Využil jsem také knihovny *profilehooks*, která je jednodušší na použití při profilování konkrétních funkcí. Knihovna *cProfile* je vhodnější pro testování skriptu jako celku.

Konektoru, který konvertuje události formátu IDEA do MISP formátu, jsem do složky, kterou sleduje, vložil 100 událostí formátu IDEA. Zpracování všech událostí trvalo 167 sekund. To je poměrně dlouhá doba, proto jsem se pokusil prozkoumat výpis profilovací knihovny *cProfile*. V tabulce 5.1 je vidět, že celkový čas zpracování malé události formátu IDEA (jako malá událost formátu IDEA je myšlena taková událost, která obsahuje jednotky atributů, především IP adres a portů, kterých bývá nejvíce) byl 0.695 sekundy (funkce ***on_created*** spouští konverzi nově detekované události ve složce). Z toho 0.687 sekundy trval průběh funkce ***add_event(...)*** knihovny *PyMISP*, která slouží pro vložení události do instance platformy MISP. Samotná konverze události IDEA do formátu MISP trvala 0.007 sekundy. Většinu času tedy zabere zpracování požadavku protokolu HTTP při vložení nové události do platformy. Přesně 0.645 sekund trvalo zpracování funkce ***send(...)*** knihovny *requests* jazyka Python, která má zá úkol odeslat připravený požadavek protokolu HTTP a zpracovat odpověď od serveru. Za většinovou délku zpracování události tedy není zodpovědná obslužná knihovna platformy MISP *PyMISP* ani konverzní knihovna, ale způsob zpracování požadavku serverem platformy MISP. Nutno podotknout, že zpracování dané události jsem zkoušel vůči reálné instanci MISP platformy, ale také vůči virtuálnímu

stroji, který byl naimportován lokálně. Délka dotazu ale zůstala téměř stejná v obou případech.

ncalls	**cumtime (s)**	**percall (s)**	**filename:lineno(function)**
1	0.695	0.695	idea_to_misp_protective.py:42(on_created)
1	0.687	0.687	aping.py:100(add_event)
1	0.645	0.645	sessions.py:617(send)
1	0.007	0.007	misp.py:708(to_misp)

Tabulka 5.1: Úryvek výpisu profilovací knihovny *cProfile* při konverzi malé události formátu **IDEA** do formátu **MISP** (**ncalls** = počet volání funkce, **cumtime** = celkový čas funkce, **percall** = čas jednoho volání funkce, **filename:lineno(function)** = název souboru:číslo řádku funkce(název funkce))

V tabulce 5.2 si lze všimnout podobné situace. Rozdílem jsou časy potřebné ke zpracování události. Jsou větší, jelikož daná vstupní událost byla rozsáhlejší. Událost obsahovala několik desítek IP adres a portů, které jsou konvertovány do MISP objektů, ale každá hodnota je uložena v samostatném atributu. Proto i překonvertovaná MISP událost je rozsáhlá a její zpracování serverem platformy MISP trvá o to déle. Konverze události konverzní knihovnou byla ale opět velmi rychlá (0.023 sekundy).

ncalls	**cumtime (s)**	**percall (s)**	**filename:lineno(function)**
1	3.442	3.442	idea_to_misp_protective.py:42(on_created)
1	3.418	3.418	aping.py:100(add_event)
1	2.975	2.975	sessions.py:617(send)
1	0.023	0.023	misp.py:708(to_misp)

Tabulka 5.2: Úryvek výpisu profilovací knihovny *cProfile* při konverzi velké události formátu **IDEA** do formátu **MISP** (**ncalls** = počet volání funkce, **cumtime** = celkový čas funkce, **percall** = čas jednoho volání funkce, **filename:lineno(function)** = název souboru:číslo řádku funkce(název funkce))

Celkově lze tedy vyvodit, že konektor ***idea_to_misp.py*** dokáže překonvertovat přibližně 0.6 události formátu IDEA za 1 sekundu. To není velmi vysoký výkon, ale nenabízí se žádné přímé řešení, jelikož většinu času nutného pro zpracování události je vyhrazeno na straně serveru. Hlavním faktorem výkonu je také velikost vstupní události formátu IDEA, podle které se poté odvíjí doba potřebná pro zpracování překonvertované události na straně serveru.

Konektor obsluhující druhý směr konverze (MISP do IDEA) je výrazně rychlejší. Konektor jsem spustil nad MISP instancí organizace CIRCL a zároveň jsem požadoval všechny události dané instance, které obsahují tagy z jmenných prostorů ECSIRT a RSIT (konfigurační argument skriptu ***misp_to_idea.py*** *--all_events*) a TLP úrovně green (výchozí nastavení skriptu). Z platformy bylo tímto způsobem staženo 157 událostí, které byly ihned překonvertovány do samostatných 157 událostí formátu IDEA. Celý tento proces zabral

7.209 sekundy a v tabulce 5.3 si lze všimnout, že opět největší část zpracování zabírá doba potřebná k vyřízení požadavku protokolu HTTP. Při normálním provozu konektoru by byla doba potřebná k vyřízení požadavku pravděpodobně menší, nebylo by nutné prohledávat celou instanci, ale pouze události z daného dne. Pro testovací účely bylo ale potřeba simulovat větší vytížení konektoru a proto jsem se dotázal MISP instance na více událostí. Samotná konverze všech 157 událostí konverzní knihovnou trvala 0.175 sekundy. Lze usoudit, že konektor je schopný zpracovat přibližně 21.8 událostí platformy MISP za 1 sekundu. Hlavním faktorem výkonu je opět zpracování HTTP požadavku serverem (funkce search). Zbytek času zpracování je složen ze samotné konverze (funkce ***to_idea(...)***) a zápis nových událostí do souborů (funkce ***save_event(...)***).

ncalls	**cumtime (s)**	**percall (s)**	**filename:lineno(function)**
1	7.209	7.209	misp_to_idea_protective.py:194(get_events)
1	6.829	6.829	aping.py:213(search)
1	6.713	6.713	sessions.py:617(send)
157	0.175	0.001	misp.py:414(to_idea)
157	0.190	0.001	misp_to_idea_protective.py:135(save_event)

Tabulka 5.3: Úryvek výpisu profilovací knihovny *cProfile* při konverzi desítek událostí formátu **MISP** do formátu **IDEA** (**ncalls** = počet volání funkce, **cumtime** = celkový čas funkce, **percall** = čas jednoho volání funkce, **filename:lineno(function)** = název souboru:číslo řádku funkce(název funkce))

Poslední implementovaný konektor (***stix_to_idea.py***) byl výkonnostně testován podobným způsobem, jako předchozí konektory. Konektoru jsem předložil 20 událostí formátu STIX platformy C3ISP a opět jsem použil knihovnu *cProfile* pro změření výkonu. Předložené události jsem si vytvořil manuálně na základě poskytnutých událostí platformy C3ISP. Strukturou byly velmi podobné, lišily se především v počtu událostí formátu CEF, které byly zanořeny uvnitř. Počet vnořených událostí formátu CEF se pohyboval od 10 do 70 událostí. V tabulce 5.4 lze vidět konverze průměrně veliké události formátu STIX, která trvala celkově 0.06 sekundy a vygenerovala 45 událostí formátu IDEA. Celkové zpracování události formátu STIX konverzní knihovnou trvalo 0.045 sekundy, zbytek času bylo spotřebováno pro zápis nových událostí formátu IDEA do souborů. Konverze všech 20 událostí trvala 0.745 sekundy, konektor tedy zvládne překonvertovat přibližně 26.84 událostí formátu STIX za 1 sekundu, kdy na výstupu může být klidně i několik desítek událostí formátu IDEA. Ovšem počet a variabilita vstupních událostí byly značně omezené a proto nelze výsledek považovat za plně věrohodný a v budoucnu se může lišit. Rozdíl by ale neměl být příliš veliký.

Běžné konektory sdružení Cesnet jsou výkonnostně velmi podobné konektoru ***stix_to_idea.py***, jelikož často zpracovávají textová vstupní data, které není výkonnostně příliš náročné zpracovat. Např. konektor sdružení Cesnet zvaný Labrea zpracovává při aktuálním zatížení přibližně 697 954 událostí za 1 den, to znamená přibližně 8.08 vstupních událostí za 1 sekundu. Ovšem dané zatížení procesoru je pouhých 10 %. Lze tedy předpokládat, že při plném vytížení by daný konektor mohl zvládat zpracovat až 80 vstupních událostí za 1 sekundu, které převádí do formátu IDEA. Daný údaj zahrnuje i činnost odeslání překon-

ncalls	**cumtime (s)**	**percall (s)**	**filename:lineno(function)**
1	0.060	0.060	stix_to_idea.py:35(on_created)
45	0.045	0.001	stix.py:302(to_idea)

Tabulka 5.4: Úryvek výpisu profilovací knihovny *cProfile* při konverzi události formátu **STIX** platformy C3ISP do formátu **IDEA** (**ncalls** = počet volání funkce, **cumtime** = celkový čas funkce, **percall** = čas jednoho volání funkce, **filename:lineno(function)** = název souboru:číslo řádku funkce(název funkce))

vertovaných událostí do systému Warden pomocí skriptu ***warden_filer.py***, který používá také protokol HTTP, ale zvládne odeslat i několik událostí najednou. Proto nemá tak vysoký vliv na výkon.

Rychlost implementovaných konektorů ***misp_to_idea.py*** a ***stix_to_idea.py*** je v pořádku. V porovnání s konektorem Labrea jsou o něco pomalejší, ale u konektoru ***misp_to_idea.py*** s daným parametrem výkonu značně hýbe čekání na vstupní požadavek protokolu HTTP vůči instanci MISP platformy a konektor ***stix_to_idea.py*** zpomaluje generování mnoha událostí formátu IDEA z jedné vstupní události formátu STIX. Pokud by se parametr výkonu konektorů zabýval počtem výstupních událostí, konektor ***stix_to_idea.py*** by se potenciálně mohl rovnat výkonu konektoru Labrea (i pokud by konektoru ***stix_to_idea.py*** byla připočtena režie odesílání překonvertovaných událostí do systému Warden). Konektor ***idea_to_misp.py*** není příliš výkonný, protože je zpomalován zpracováním požadavků protokolu HTTP odesílajících překonvertované události do platformy MISP. Daný problém by se dal vyřešit odesíláním více událostí najednou, ale knihovna *PyMISP* takový způsob odesílání událostí nepodporuje a je možné, že vícenásobné zpracování událostí nepodporuje ani serverová strana platformy MISP.

Dále je vhodné uvažovat výkon strojů, na kterých byly naměřeny výkonnostní údaje. Konektor Labrea běží na serverovém stroji, který je svým výkonem uzpůsoben pro běh podobných konektorů. Mnou implementované konektory byly testovány na lokálním počítači (procesor Intel i5-8250U). Konektor Labrea také zpracovává události z logovacího souboru, které jsou menšího charakteru. Složitost zpracování formátů STIX a MISP není také přímo porovnatelná. Složitost zpracování vstupních formát může být rozdílná a i proto se výkony konektorů mohou lišit. I přesto je ale rozumné porovnat mnou implementované konektory s alespoň přibližnou výkonností konektorů implementovaných sdružením Cesnet.

5.4 Výsledky testování

Po automatických i manuálních testech konvertory konvertují události dle navrženého postupu. Odhalené chyby jsou opraveny. Automaticky otestovat konverzi všech možných podob událostí je velice těžké, proto je nutné konverzi testovat v průběhu vývoje i manuálně a zkoušet co nejvíce možných kombinací. Automatické testy byly velmi užitečné v pozdější fázi vývoje, kdy některé části konvertorů podléhaly změnám ve struktuře implementace.

Konektory a konvertor zajišťující konverzi mezi formáty IDEA a MISP jsou otestovány na velkém množství událostí a konverze nyní nevykazuje odchylky od předem navrženého postupu konverze (kapitola 3). Procesu testování velkým dílem přispěl přístup k systému Warden, odkud bylo možné získat neomezené množství událostí formátu IDEA. Události byly využity primárně pro testovací účely, ale vedly také k lepšímu celkovému porozumění

formátu IDEA. Přístup byl také povolen k jedné reálné instanci platformy MISP. Reálné události této instance výrazně urychlily proces porozumění formátu platformy, ale také pomohly samotnému testování. Celému testování také pomohla možnost provozu vlastní virtuální MISP instance, kde bylo možné vyzkoušet si všechny možné funkce platformy bez obav ze špatné konfigurace. Výkonnostně je konverzní knihovna velmi rychlá. Samotný konektor ***idea_to_misp.py*** už ale nedokáže udržet krok s konverzní knihovnou a zaostává při vkládání překonvertovaných událostí do platformy MISP pomocí protokolu HTTP. Podobného zpomalení si lze všimnout i u konektoru ***misp_to_idea.py***, kterému trvá prohledání platformy a stažení událostí určených ke konverzi. Jakmile jsou události staženy, dojde k jejich velmi rychlé konverzi pomocí konverzní knihovny. Tímto je výkon konektoru uveden do rozumného poměru.

Konektor a konvertor obsluhující konverzi mezi formáty STIX a IDEA je dobře otestovaný na dostupných vstupních událostech. Dostupnost více vstupních událostí by zjednodušila postup vývoje. Navíc při umožnění přístupu k více událostem v budoucnosti může být vyžadována aktualizace konverzní knihovny, protože objekty platformy C3ISP mohou nabývat lehce odlišných struktur. To samé platí pro události formátu CEF, v kterých mohou být použity vlastní klíče a hodnoty, které kvůli nedostatku informací nejsou zahrnuty do konverze. Konverzní knihovna je ale plně funkční a funguje správně podle navrženého postupu konverze (kapitola 3). Výkonnostně jsou na tom konektor i konverzní knihovna dobře, dokážou zpracovat rozumné množství událostí za jednotku času.

Kapitola 6

Závěr

Konverze mezi bezpečnostními formáty nemusí být vždy přímočará, ale je velmi důležitá. Pomocí konvertorů je možné sdílet bezpečnostní události mezi organizacemi, které díky více událostem mohou lépe reagovat na aktuální hrozby. Navíc většina bezpečnostních událostí bývá zpracována automaticky a do hry mnohdy přichází i strojové učení a umělá inteligence. Síla těchto pokročilých metod se ale silně odvíjí od množství informací, na kterých se mohou tyto metody učit. Sdílení bezpečnostních událostí tedy může rozšířit obzory nejen samotným správcům, ale i strojům.

Cílem této práce bylo zanalyzovat a navrhnout možný postup konverze mezi kyberbezpečnostními formáty pro sdílení síťových bezpečnostních hlášení. Výsledkem této práce je úspěšná konverze mezi bezpečnostními formáty IDEA a formátem používaným platformou MISP. Dalším přínosem je konverze modifikované verze formátu STIX verze 2.0, který využívá platforma C3ISP do formátu IDEA. Tato práce byla zhotovena na základě smluvního výzkumu se sdružením Cesnet a již brzy je v plánu nasadit zhotovené konvertory a konektory do provozu v praxi.

V první fázi této práce jsem prozkoumal a zanalyzoval několik bezpečnostních formátů, u kterých jsem se zaměřil především na samotné struktury formátů, které jsou pro zbytek práce velmi důležité. Následně jsem porovnal vlastnosti jednotlivých formátů mezi sebou. Dalším krokem práce byla analýza převeditelnosti formátů, z které vyplynula lehká omezení konverze, které pokud budou vyřešeny, lze prohlásit formáty platforem C3ISP a MISP jako konvertovatelné s formátem IDEA. Finálním bodem teoretického rozboru byl návrh převodu mezi formáty, který kromě podrobného návrhu převodu řeší i některé problémy, které vykazovala analýza převeditelnosti. Podle návrhu převodu byly vytvořeny konverzní knihovny a konektory, které realizují konverzi. V poslední části této práce jsem implementoval několik testů, které ověřili správnost implementace podle návrhu na událostech ze systému Warden.

Tato práce se zabývala konverzí bezpečnostních událostí platformy C3ISP do formátu IDEA. Při získání podrobnějších specifikací platformy C3ISP je možným rozšířením práce konverze formátu IDEA do událostí platformy C3ISP. Dále je možné se zamyslet nad konverzí formátu IDEA do nemodifikované podoby formátu STIX verze 2.1. Tato verze má být dokončena tento rok (2019) a s novou verzí přichází nové objekty formátu, které jsou přizpůsobeny pro přenos událostí generovaných systémy IDS (např. doménový objekt *Event*) a mohlo by být dosaženo efektivnější a přesnější konverze. Specifikace nové verze ještě ale není dokončena a přesné závěry proto zatím nemohou být vyvozeny.

Literatura

[1] ArcSight, Inc.: Common Event Format [online]. 2010, [vid. 30.11.2018].
URL https://kc.mcafee.com/resources/sites/MCAFEE/content/live/CORP_KNOWLEDGEBASE/78000/KB78712/en_US/CEF_White_Paper_20100722.pdf

[2] Barnum, S.: Standardizing cyber threat intelligence information with the Structured Threat Information eXpression (STIX). *Mitre Corporation*, ročník 11, 2012: s. 1–22.

[3] Danyliw, R.: The Incident Object Description Exchange Format Version 2. RFC 7970, Listopad 2016.

[4] Dulaunoy, A.; Iklody, A.: MISP core format. Internet-Draft draft-dulaunoy-misp-core-format-07, Internet Engineering Task Force, Únor 2019, work in Progress.

[5] Feinstein, B.; Curry, D.; Debar, H.: The Intrusion Detection Message Exchange Format (IDMEF). RFC 4765, Březen 2007.

[6] IBM corp.: Log Event Extended Format (LEEF) version 2 [online]. 2013, [vid. 28.11.2018].
URL https://www.ibm.com/support/knowledgecenter/en/SS42VS_DSM/b_Leef_format_guide.pdf

[7] Iklody, A.; Dulaunoy, A.; CIRCL: misp-objects [online]. 2016, [vid. 2.3.2019].
URL https://github.com/MISP/misp-objects

[8] Kácha, P.: IDEA: Designing the data model for security event exchange. In *Proceedings of the 17th International Conference on Computers: Recent Advances in Computer Science*, 2013.

[9] Kácha, P.: IDEA: Security event taxonomy mapping. In *Proceedings of the 18th International Conference on Circuits, Systems, Communications and Computers*, 2014.

[10] Kácha, P.: Framework pro konektory, [online]. 2017, [vid. 25.3.2019].
URL https://sabu.cesnet.cz/_media/cs/2017-08-07_pavel_kacha_warden_a_konektory.pdf

[11] Kácha, P.; Kostěnec, M.; Kropáčová, A.: Warden 3: Security event exchange redesign. In *Proceedings of the 19th International Conference on Computers: Recent Advances in Computer Science*, 2015.

[12] Piazza, R.; Wunder, J.; Jordan, B.: STIX™ Version 2.0. Part 1: STIX Core Concepts [online]. 2017, OASIS Committee Specification 01, [vid. 20.12.2018]. URL https://docs.oasis-open.org/cti/stix/v2.0/stix-v2.0-part2-stix-objects.html

[13] Piazza, R.; Wunder, J.; Jordan, B.: STIX™ Version 2.0. Part 2: STIX Objects [online]. 2017, OASIS Committee Specification 01, [vid. 20.12.2018]. URL https://docs.oasis-open.org/cti/stix/v2.0/stix-v2.0-part2-stix-objects.html

[14] Rouse, M.: threat intelligence (cyber threat intelligence) [online]. 2015, [vid. 10.10.2018]. URL https://whatis.techtarget.com/definition/threat-intelligence-cyber-threat-intelligence

[15] Steinberger, J.; Sperotto, A.; Golling, M.; aj.: How to exchange security events? overview and evaluation of formats and protocols. In *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, IEEE, 2015, s. 261–269.

[16] Wagner, C.; Dulaunoy, A.; Wagener, G.; aj.: MISP: The Design and Implementation of a Collaborative Threat Intelligence Sharing Platform. In *Proceedings of the 2016 ACM on Workshop on Information Sharing and Collaborative Security*, ACM, 2016, s. 49–56.

Příloha A

Obsah přiloženého paměťového média - CD

V kořenovém adresáři se nachází adresáře s následujícím obsahem:

dokumentace: zdrojové soubory tohoto dokumentu

misp: zdrojové soubory implementující konverzi mezi formáty IDEA a MISP

prilohy: proprietární dokumentace formátu STIX platformy C3SIP a validační schéma formátu IDEA

stix: zdrojové soubory implementující konverzi mezi formáty IDEA a STIX

Příloha B

Schéma formátu IDEA

V této příloze je uveden výběr nejdůležitějších informací o struktuře formátu IDEA, které jsou převzaty z validačního schéma formátu. Celé schéma je přiloženo na paměťovém médiu uvnitř složky **\prilohy\idea**.

Schéma popisuje jednotlivé atributy formátu a jejich význam (*description*). Dále jsou popsány datové struktury, kterých informace uložené v daných atributech nabývají. K tomu slouží atribut *ref* odkazující se na striktní definici (která je také uvnitř validačního schéma) nebo atribut *type*.

```
"required": ["Format", "ID", "DetectTime", "Category"],
 ⋮
"Format": {
    "description": "Identifier of the IDEA container.",
    "$ref": "#/definitions/Version"
},
"ID": {
    "description": "Unique message identifier.",
    "$ref": "#/definitions/ID"
},
"DetectTime": {
    "description": "Timestamp of the moment of detection of event (not
    necessarily time of the event taking place). This timestamp is
    mandatory, because every detector is able to know when it detected the
    information - for example when line about event appeared in the logfile,
    or when its information source says the event was detected, or at least
    when it accepted the information from the source.",
    "$ref": "#/definitions/Timestamp"
},
"Category": {
    "description": "Array of event categories.",
    "type": "array",
    "items": {
        "description": "Category of event.",
        "$ref": "#/definitions/EventTag"
    }
```

```
},
"Source": {
    "type": "array",
    "description": "Array of source or target descriptions.",
    "items": {
        "description": "Information concerning particular source or
        target.",
        "type": "object",
        "properties": {
            "IP4": {
                "description": "Array of IPv4 addresses.",
                "type": "array",
                "items": {
                    "description": "IPv4 addresses of this source/target.",
                    "$ref": "#/definitions/Net4"
                }
            },
            "MAC": {
                "description": "Array of MAC addresses.",
                "type": "array",
                "items": {
                    "description": "MAC addresses of this source/target.",
                    "$ref": "#/definitions/MAC"
                }
            },
             ⋮
        }
"Target": {
    "$ref": "#/properties/Source"
},
"Attach": {
    "description": "Array of attachment descriptions.",
    "type": "array",
    "items": {
        "type": "object",
        "description": "Additional attachment information and data.",
        "properties": {
             ⋮
        }
    }
},
"Node": {
    "description": "Array of detector descriptions.",
    "type": "array",
    "items": {
        "description": "Detector or possible intermediary (event
        aggregator, correlator, etc.) description.",
```

```
        "type": "object",
        "properties": {
            "Name": {
                "description": "Name of the detector, which must be
                reasonably unique, however still bear some meaningful sense.
                Usually denotes hierarchy of organisational units which
                detector belongs to and its own name.",
                "$ref": "#/definitions/NSID"
            },
             ⋮
        }
    }
}
```

Výpis B.1: Úryvky validačního schéma formátu IDEA

Příloha C

Proprietární dokumentace platformy C3ISP

V této příloze je uveden výběr důležitých informací o struktuře formátu STIX platformy C3ISP, které jsou převzaty z proprietární dokumentace. Celá dokumentace je přiložena na paměťovém médiu uvnitř složky **\prilohy\c3isp**.

Následující číslování kapitol uvedené v této příloze **nemá žádnou souvislost** s číslováním kapitol této práce. Číslování bylo zanecháno pouze pro snazší dohledání daného úryvku dokumentace, která je poměrně rozsáhlá.

3 Detection of brute force attacks

This document describes a translation of the /var/log/auth.log file. Most of the lines contained in these logs can be dropped, for now we are filtering only some information extracted from the sshd daemon (openssl).

3.1 Input

The first line represents the new metadata that will be used to build the CEF prefix

```
#dataType=ssh; startTime=2018-10-26T17:02:23.000+0200;
endTime=2018-10-26T17:02:43.000+0200; organization=isp@cnr; hostname=
calogero; vendor=ssh-ubuntu; severity=5; year=2018;
```

3.2 Intermediate step (example)

```
CEF:0|vendor|hostname|1.0|100|dataType-endTime|1|src=192.168.0.1 app=ssh2
duser=elastic spt=37239 type=0 outcome=accepted date=1539684793000
dtz=Europe/Rome
```

Last field of the CEF prefix is called *severity* and it can be:
Severity values:

1 – for auth successes
5 – for bad key/password
10 – for invalid user

The other CEF parameters get the following explanation:

Type

type=0 – password access

type=1 – pubkey access

Outcome

outcome=accepted – login successful

outcome=failure – login unsuccessful

Reason

reason=none – no errors (optional)

reason=bad_password – login failed

reason=invalid_user – user does not exist

reason=bad_key – login failed for pubkey access