Příjemci podpory: ***Poskytovatel:***

Analýza šifrovaného provozu pomocí síťových toků
s identifikačním kódem VJ02010024

Název předkládaného výsledku:

Sada zásuvných modulů pro systém QRadar

Typ výsledku dle UV č. 837/2017	Evidenční číslo (příjemce)	Rok vzniku
R-software	*V2*	2024stup
ISBN-ISSN	**Webový odkaz na výsledek**	**Kde a kdy publikováno**
	https://nesfit.github.io/feta-qradar-modules/	

Stručná anotace k výsledku: *(max. 8 řádků)*

Výsledek V2: „Sada zásuvných modulů pro systém QRadar“ poskytuje pokročilou ochranu proti nežádoucím jevům v šifrované komunikaci jako je výskyt malware, či phishing. Výstup tvoří dva moduly: DomainRadar a Malware Radar. Modul DomainRadar je schopen detekovat maligní doménová jména, která souvisí s podvodnými weby, šířením škodlivého kódu, či botnetovou komunikací. Modul Malware Radar je schopen v síťové komunikaci rozpoznat výskyt škodlivého software.

Řešitelský tým:

Radek Hranický, Ondřej Ryšavý, Ondřej Ondryáš, Ondrej Lichtner, Adam Horák, Jan Polišenský, Petr Pouč, Peter Polóni, Filip Bučko, Dominik Soukup, Petr Matoušek

Obsah

Úvod

Výsledkem projektu VJ02010024-V2 je „Sada zásuvných modulů do QRadar“, která zahrnuje nástroje DomainRadar a MalwareRadar. Tyto moduly jsou určeny k detekci rizik spojených se šifrovanou síťovou komunikací, jako je výskyt malwaru nebo phishingových domén. DomainRadar se zaměřuje na identifikaci podezřelých / škodlivých doménových jmen, zatímco MalwareRadar detekuje přítomnost škodlivého softwaru na základě analýzy síťových toků. Moduly využívají pokročilé techniky strojového učení a analýzy indikátorů kompromitace (IoC). Systém poskytuje flexibilní architekturu s možností nasazení v prostředí Docker, snadnou integraci do existující infrastruktury pomocí IBM QRadar, a nabízí přehledné webové rozhraní pro operátory bezpečnostních týmů. Tento nástroj přispívá k posílení ochrany síťové komunikace a zvyšuje efektivitu boje proti kybernetickým hrozbám.

DomainRadar je nástroj navržený pro identifikaci maligních doménových jmen v síťové komunikaci, včetně domén souvisejících s phishingem, šířením malwaru nebo botnetovými sítěmi. Využívá pokročilé metody strojového učení a heuristické analýzy, aby analyzoval data z DNS záznamů, geolokačních databází, registračních informací (RDAP/Whois), a bezpečnostních certifikátů TLS. Nástroj sbírá informace o doménách prostřednictvím distribuovaných kolektorů, které získávají data z různých zdrojů, jako jsou DNS servery, GeoIP databáze, nebo reputační systémy. DomainRadar poté extrahuje příznaky, které charakterizují doménová jména, a na jejich základě klasifikuje domény podle míry rizika. Výsledky analýz jsou poskytovány prostřednictvím přehledného webového rozhraní, které umožňuje operátorům SOC týmů snadno interpretovat výstupy, zobrazit detailní analýzy jednotlivých domén a navázat na zjištěné bezpečnostní incidenty. Díky integraci s IBM QRadar poskytuje DomainRadar také přímou vazbu na existující bezpečnostní události v síti, což umožňuje komplexní a efektivní reakci na detekované hrozby.

MalwareRadar je specializovaný nástroj určený k detekci škodlivé síťové komunikace. Zaměřuje se na analýzu síťových toků a šifrovaných přenosů pomocí metod strojového učení a fingerprintingu TLS komunikace (např. JA4+ otisky). MalwareRadar využívá indikátory kompromitace (IoC), jako jsou URL, IP adresy, nebo doménová jména, získané z analýzy vzorků malwaru prostřednictvím sandboxové platformy Tria.ge. Detekční mechanismy analyzují metadata komunikace, včetně velikostí paketů, sekvencí TLS segmentů a dalších atributů, čímž identifikují typické vzorce komunikace malwaru. Nástroj podporuje snadné nasazení v prostředí Docker, což usnadňuje instalaci a provoz. MalwareRadar také poskytuje možnost flexibilní integrace s dalšími systémy díky výstupním formátům jako JSON nebo syslog. Tento modul nejen detekuje malware komunikaci, ale také pomáhá klasifikovat konkrétní rodiny malwaru, což umožňuje organizacím lépe pochopit povahu a rozsah detekovaných hrozeb. Výsledky jsou hlášeny do IBM QRadar, kde přispívají k ucelenému pohledu na bezpečnostní situaci v síti.

Struktura výsledku

Výsledek je dostupný jako archiv — balík obsahující zdrojové kódy spolu se skripty a ukázkovou konfigurací pro spuštění v testovacím prostředí. Struktura archivu je naznačena ve Výpisu 1. Archiv je rozdělen na dva hlavní adresáře obsahující nástroj DomainRadar a nástroj MalwareRadar.

```
.
├── LICENSE ..... Licenční soubor
├── README.md ..... Readme soubor
├── DomainRadar
│   ├── data-pipeline ..... Subsystém pro sběr a zpracování dat
│   ├── infra ..... Předloha prostředí pro spuštění
│   ├── input ..... Subsystém pro načítání vstupu
│   ├── setup ..... Skripty pro přípravu prostředí na spuštění
│   └── webui ..... Uživatelské rozhraní
└── MalwareRadar
    ├── Deploy ..... Skripty a konfigurace pro nasazení komponenty
    ├── Scripts ..... Skripty pro přípravu dat
    ├── Source ..... Zdrojové kódy komponenty
    └── Readme.md ..... Popis implementace a instalace
```

Výpis 1: Struktura výsledku.

Nástroj DomainRadar se skládá z řady komponent (viz sekci Sada zásuvných modulů pro systém QRadar -– DomainRadar — **Architektura modulu**), které jsou členěny do podadresářů. Pro všechny komponenty jsou připraveny soubory typu `Containerfile`, pomocí kterých je možné sestavit obrazy kontejnerů a spustit je např. pomocí platformy Docker. K dispozici jsou instalační skripty, které zajistí přípravu ukázkového běhového prostředí na lokálním stroji (viz sekci Instalační příručka — DomainRadar — **Postup instalace**).

Zdrojový kód jednotlivých mikroslužeb, které představují páteřní část nástroje, je umístěn v adresáři `data-pipeline`, jak naznačuje Výpis 2.

```
data-pipeline
├── java
│   ├── common ..... Modul se sdílenými funkcemi
│   ├── connect ..... Zásuvné moduly do systému Kafka Connect
│   ├── merger-flink ..... Subsystém pro agregaci dat „Data Merger“
│   ├── serialization ..... Knihovna se serializačními prostředky
│   └── standalone-collectors ..... Sběrové mikroslužby
└── python
    ├── classifier_unit ..... Klasifikační mikroslužba
    ├── classifiers ..... Klasifikátory a modely
    ├── collector ..... Sběrové mikroslužby
    ├── common ..... Modul se sdílenými funkcemi
    ├── config_manager ..... Správce konfigurace
    └── extractor ..... Extraktor příznaků
```

Výpis 2: Struktura adresáře `data-pipeline` s komponentami nástroje DomainRadar.

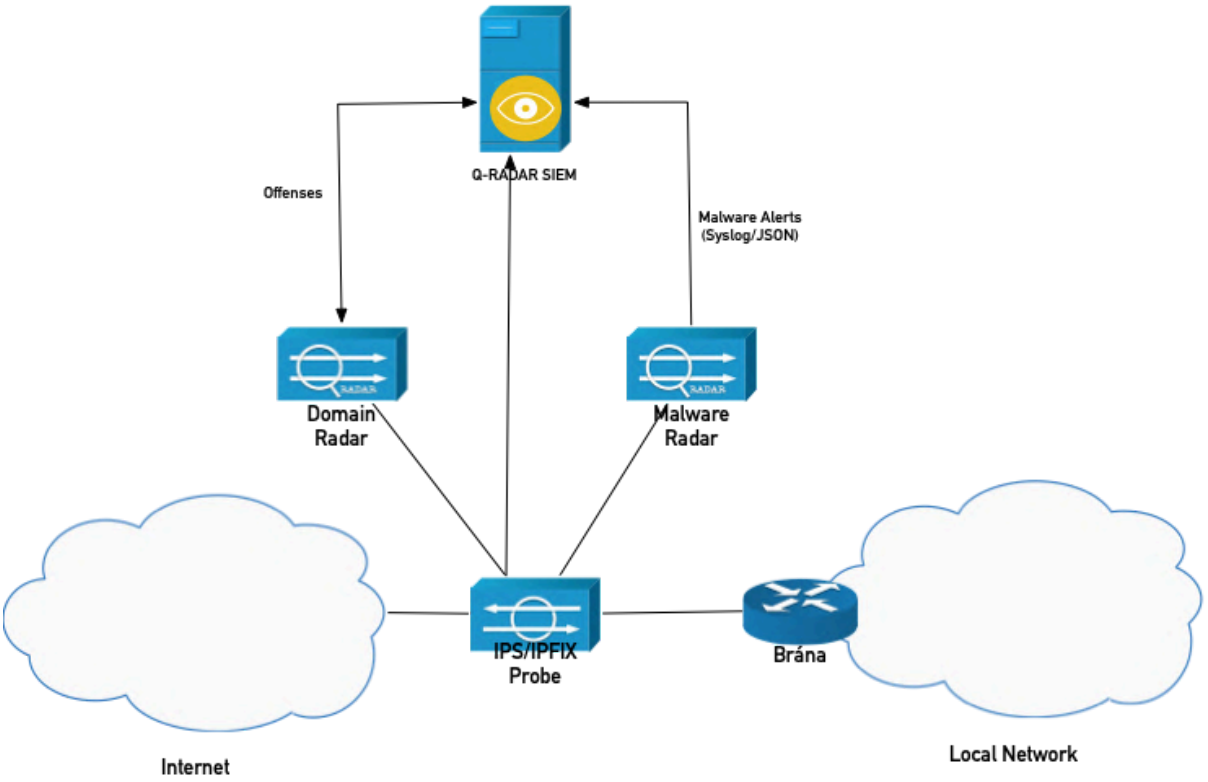
Architektura systému

IBM QRadar je platforma pro správu bezpečnostních informací a událostí (SIEM), která zajišťuje centralizovaný přehled o bezpečnostních událostech v síti a jejich efektivní analýzu. QRadar integruje data z různých zdrojů, jako jsou firewall logy, IDS/IPS systémy, aplikační logy, síťové toky, a další bezpečnostní nástroje, čímž umožňuje organizacím monitorovat a vyhodnocovat bezpečnostní incidenty v reálném čase. QRadar využívá pokročilé analytické metody, včetně strojového učení a korelace událostí, k odhalování hrozeb a jejich prioritizaci na základě vypočteného rizika. Díky své flexibilitě a rozšiřitelnosti podporuje integraci vlastních modulů, což umožňuje adaptaci na specifické potřeby organizace. Schéma integrace vytvořených modulů je na Obrázku 1.

DomainRadar byl navržen pro hladkou integraci s platformou IBM QRadar, což umožňuje jeho nasazení ve dvou režimech: jako plnohodnotný modul QRadar nebo jako samostatný nástroj pro analýzu rizikových domén. Propojení se systémem QRadar je realizováno prostřednictvím QRadar API, které poskytuje modulu přístup k informacím o bezpečnostních incidentech (Offenses), souvisejících událostech (Events) a síťových tocích (Flows). DomainRadar poskytuje výsledky analýzy do systému QRadar pomocí protokolu syslog. QRadar konzumuje tyto výsledky a umožňuje s nimi dále pracovat v rámci svého analytického rozhraní.

MalwareRadar je navržen pro volnou integraci se systémy pro správu bezpečnostních informací a událostí (SIEM) včetně IBM QRadar. Tato integrace poskytuje pokročilé schopnosti detekce škodlivé komunikace a rozšiřuje analytické možnosti SIEM platformy. MalwareRadar generuje výstupy ve standardních formátech, jako jsou JSON nebo syslog, které jsou podporovány většinou SIEM nástrojů, včetně QRadar. Výsledky analýzy obsahují detaily o síťových spojeních identifikovaných jako podezřelé, včetně metadat, jako jsou IP adresy, domény, URL a hodnota pravděpodobnosti výskytu malware. Data z MalwareRadar lze propojit s bezpečnostními incidenty (Offenses) detekovanými SIEM systémem. MalwareRadar doplňuje informace o indikátorech kompromitace (IoC), které pomáhají přesněji identifikovat příčinu a rozsah incidentu.

MalwareRadar monitoruje pomocí externího nástroje, například Suricata, síťový provoz a odesílá výsledky analýzy do QRadar prostřednictvím syslog nebo JSON zpráv. Výsledky se zobrazují jako nové události v konzoli QRadar a propojují se s existujícími záznamy o bezpečnostních incidentech. Operátoři mohou využívat dashboard QRadar pro zobrazení podrobných informací o detekované komunikaci a plánování reakce na incidenty.



Obrázek 1: Architektura zapojení komponent systému.

Sada zásuvných modulů pro systém QRadar

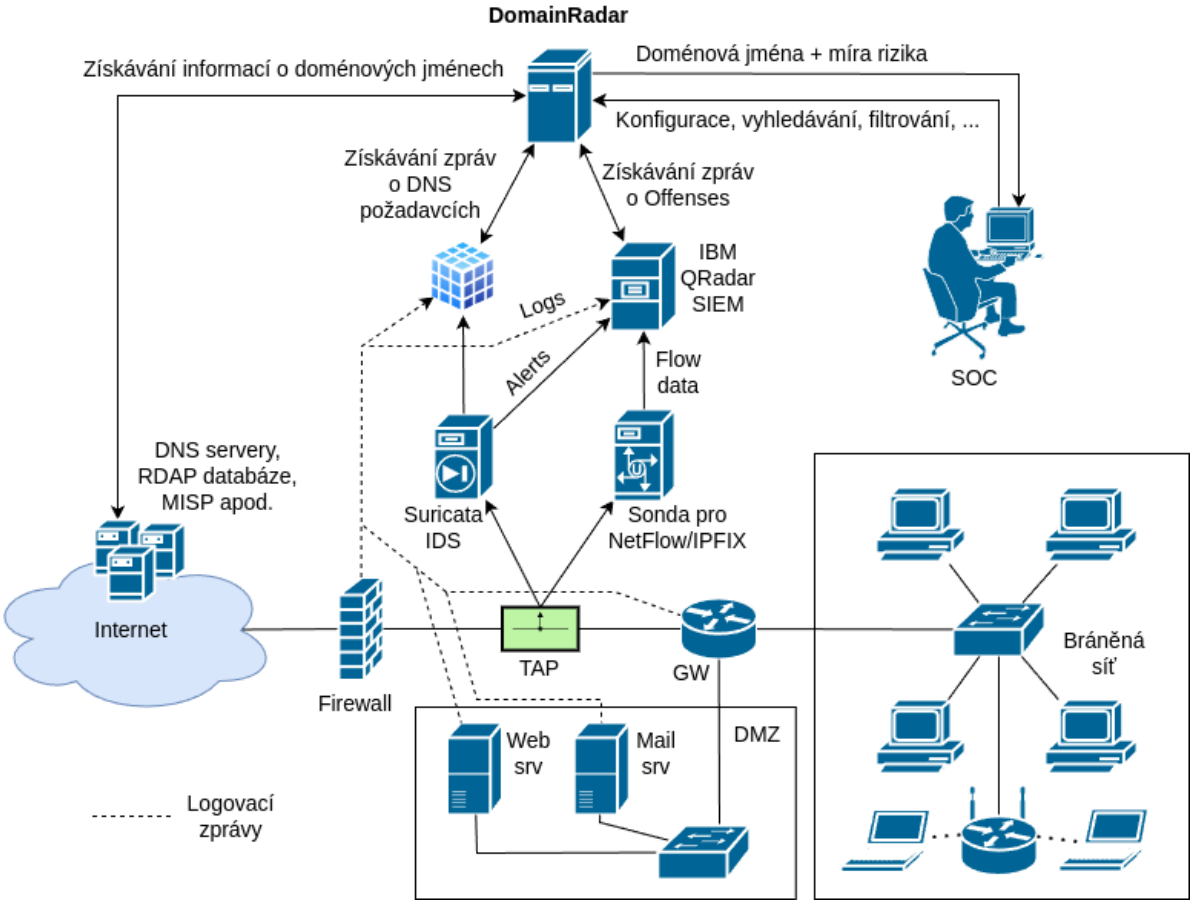
Výstup V2 sestává za dvou zásuvných modulů pro systém IBM QRadar. Jde o moduly:

- **DomainRadar** — Nástroj pro detekci maligních doménových jmen v síťovém provozu
- **MalwareRadar** — Nástroj pro detekci malware ze síťových toků

Následující sekce popisují detailní specifikaci obou výstupů.

DomainRadar

Modul DomainRadar slouží k detekci maligních doménových jmen v síťové komunikaci. Domény jsou získávány primárně z DNS komunikace. Pro každou doménu modul určuje pravděpodobnost rizika jako míru výskytu charakteristik typických pro phishingové domény (pro hosting podvodných webových stránek), malwarové domény (pro šíření škodlivého kódu) a domény generované pomocí algoritmů na DGA (Domain Generation Algorithms — typické pro Command & Control servery botnetových sítí), přičemž komunikace s doménami na bázi DGA je typicky indikátorem, že se v dané síti nachází kompromitované zařízení, jež je součástí botnetu. Se systémem IBM QRadar modul komunikuje přes QRadar API.



Obrázek 2: Modelová situace nasazení nástroje DomainRadar.

Obrázek 2 ukazuje modelovou situaci nasazení modulu DomainRadar pro ochranu sítě. Bráněná síť je přes bránu (směrovač GW s NAT) a firewall připojena k internetu. Oddělená část tvoří demilitarizovanou zónu (DMZ) se servery pro služby, které jsou dostupné z veřejného internetu, zde tedy např. webový server a e-mailový SMTP server. Z těchto serverů a také z firewall a brány jsou získávána logovací hlášení ve formě zpráv protokolu Syslog, které jsou zasílány:

- do systému **IBM QRadar SIEM**,
- do úložiště na bázi **ELK** (Elasticsearch, Logstash, Kibana).

Na lince do internetu je nasazeno zařízení network TAP, které zrcadlí provoz do Suricata IDS a sondy na bázi NetFlow či IPFIX. Informace o síťových tocích (flow data) jsou zasílány do systému IBM QRadar SIEM. IDS systém Suricata je sám o sobě schopen detekovat síťové útoky, přičemž v případě detekce spustí poplach (*alert*), který je zaslán jako zpráva protokolu Syslog do IBM QRadar a je zároveň dojde k jeho uložení do ELK. Na základě logovacích zpráv z monitorovaných zařízení a flow dat o síťové komunikace systém IBM QRadar detekuje bezpečnostní incidenty (*Offenses*).

Primárním vstupem pro modul DomainRadar jsou logovací zprávy ze Suricata o zaznamenaných zprávách protokolu DNS, konkrétně požadavcích na překlad doménových

jmen a odpovědí na tyto požadavky. Tyto zprávy DomainRadar čte z úložiště ELK. Z těchto zpráv následně získává doménová jména, která poté analyzuje. Propojení se systémem IBM QRadar je řešeno přes QRadar API, přes které DomainRadar zjišťuje informace o aktuálně hlášených incidentech a hledá jejich souvislost s analyzovanými doménovými jmény.

DomainRadar následně dohledává a stahuje další informace k analyzovaným doménovým jménům z následujících zdrojů:

- **IBM QRadar** — z tohoto systému DomainRadar získává informace o bezpečnostních incidentech (Offenses), které souvisí s IP adresami zkoumaných domén
- **DNS servery** — zde stahuje dostupné DNS záznamy včetně informací o TTL apod. Z těchto dat jsou také získány IP adresy, které se k dané doméně vážou.
- **Databáze RDAP** — zde získává dostupné informace o a) doméně, b) souvisejících IP adresách. K doméně je dohledán např. registrátor, doba registrace, administrativní kontakt a další položky. K IP adresám je dohledáván např. autonomní systém, prefix sítě, registrátor aj.
- **Certifikáty TLS** — dojde ke stažení celého řetězu certifikátů (certificate chain) včetně informací o certifikačních autoritách, platnosti a bezpečnostních parametrech.
- **Informace z TLS Handshakes** – se vzdáleným serverem je proveden pokus o navázání spojení, ze kterého jsou extrahovány informace o podporovaných šifrovacích sadách aj.
- **ICMP Echo** — doméně je zasláno několik zpráv ICMP Echo Request (ping) a dle odpovědí ICMP Echo Reply je vypočítána průměrná doba odezvy, tedy Round-Trip Time (RTT).
- **Geolokační databáze** — Pro každou IP adresu, která s doménou souvisí dojde k dohledání lokalizačních informací v databázi GeoIP 2. Získávány jsou údaje jako zeměpisná šířka a délka, země, ve které se vzdálený server nachází apod.
- **Obsah HTML/DOM** — běží-li na vzdáleném zařízení HTTP server, dojde k extrakci zdrojového kódu stránky, který slouží k další analýze.

K získání dat o doménách slouží sběrový subsystém, který může běžet jak na stejném stroji jako centrální část nástroje DomainRadar, tak i zcela odděleně na jednom či více sběrových zařízeních, což je výhodné zejména v situacích, kdy není žádoucí, aby identita sběratele dat byla vyzrazena vzdáleným stranám — např. v případě, kdy je analyzována doména hostující podvodné stránky s phishingem, přičemž není žádoucí, aby provozovatel této aktivity znal, kdo se na jeho server doptává. Architektura a princip fungování tohoto subsystému jsou popsány dále v této kapitole.

Po sběru dat dojde k extrakci zájmových příznaků o dané doméně a integrované klasifikátory odhadnou pravděpodobnost hrozby. Tuto informaci, včetně detailních výsledků klasifikace následně uloží do databáze a operátor SOC týmu pak může přes webové rozhraní číst výsledky, jak je vyobrazeno na Obrázku 2. Souvisí-li navíc některá z IP adres asociovaných se zkoumanou doménou s některým z bezpečnostních incidentů (Offense), které detekoval IBM QRadar, nabízí DomainRadar informaci o těchto incidentech, včetně počtu souvisejících síťových toků a logovacích zpráv protokolu Syslog. Na jednotlivé

bezpečnostní incidenty je k dispozici přímý proklik do QRadar Console, kde si operátor SOC může zobrazit detaily o souvisejících hrozbách.

Kromě vstupu domén z DNS se operátor může na konkrétní domény dotázat ručně přes webové rozhraní. Pro odlehčení hardwarových nároků je DomainRadar vybaven také konfigurovatelnou sadou filtrů, které umožňují ignorovat známé domény. DomainRadar nabízí také možnost tyto filtry aktualizovat automaticky na základě informací z platforem Threat Intelligence, přičemž pro demonstraci a ověření funkčnosti byla implementována podpora pro platformu MISP. Tento systém může sloužit jednak jako zdroj filtrovacích pravidel, ale také případně jako další zdroj domén, je-li jejich analýza žádoucí.

Díky skutečnosti, že vstupní subsystém Loader je sám o sobě modulární, lze do něj doplnit případné další doménové vstupy či nové typy filtrů. Kromě domén z DNS, ručně zadaných domén, případně domén z MISP je tak možné přidat podporu pro čtení domén přímo z DNS resolverů a dalších zdrojů dat. Obdobně flexibilně lze implementovat další filtry.

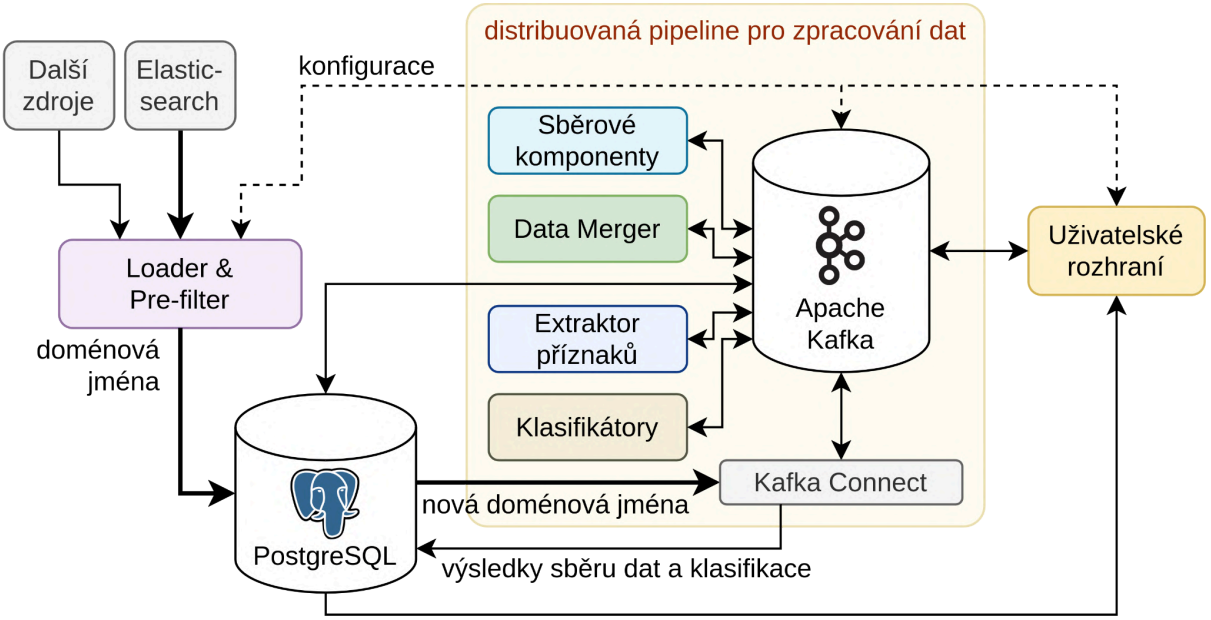
Z pohledu uživatele (operátora SOC) DomainRadar může sloužit:

- k získání přehledu o situaci v bráněné síti,
- k detekci a analýze rizikových domén, které uživatelé bráněné sítě navštěvují,
- k získání doplňující informací při vyšetřování existující hrozby související s doménami,
- k identifikaci kompromitovaných síťových uzlů, které jsou součástí botnetu,
- jako agregátor a filtr informací o nových hrozbách z platform Threat Intelligence, je-li platforma nakonfigurována jako doménový vstup.

Narozdíl od reputačních systémů jako CESNET NERD či VirusTotal modul DomainRadar neklasifikuje na základě hlášení uživatelů či autorit, ale veškerá rizika vyhodnocuje zcela autonomně s využitím strojového učení.

Architektura modulu

Obrázek 3 znázorňuje vysokoúrovňový pohled na architekturu modulu DomainRadar. Ta sestává z několika subsystémů, které mohou běžet jak na společném hardware, tak odděleně. Řešení nativně podporuje paralelní a distribuované zpracování — jak na úrovni jednotlivých subsystémů tak interně v rámci těchto subsystémů.



Obrázek 3: Vysokoúrovňový pohled na architekturu nástroje DomainRadar.

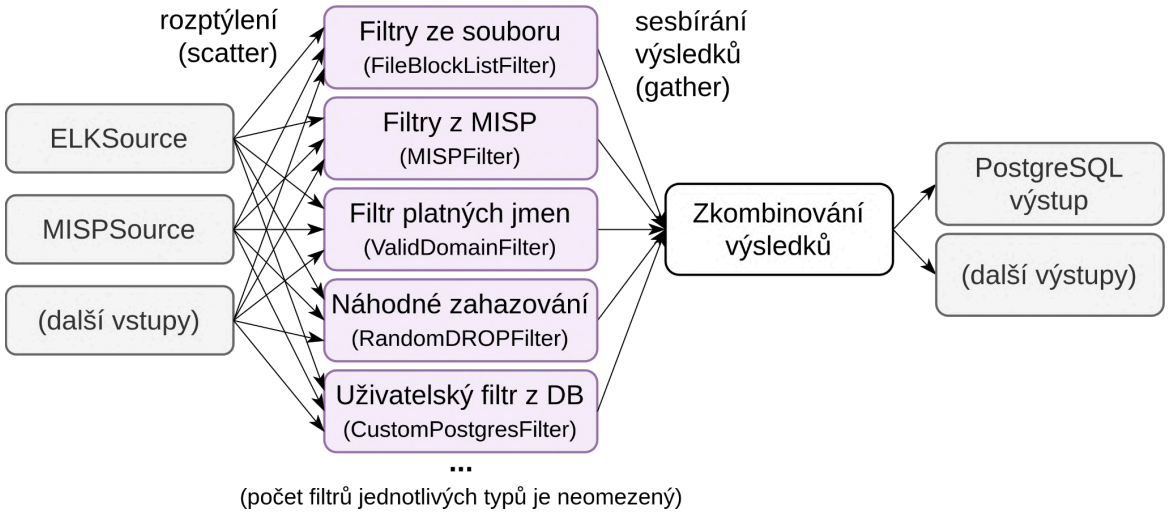
Tyto subsystémy tvoří:

- **Subsystém pro načítání vstupu „Loader & Pre-filter“** – představuje vstupní blok systému, který získává a filtruje vstupní doménová jména, přičemž zdroje i filtry jsou plně nastavitelné.
- **Databáze PostgreSQL** — tvoří úložiště informací o doménových jménech, souvisejících datech, příznacích a výsledcích klasifikace.
- **Apache Kafka®** — je distribuovaná platforma pro přenos a perzistenci zpráv typu „klíč–hodnota“. Zprávy jsou členěny do pojmenovaných proudů, tzv. témat (*topics*). Témata jsou uvnitř platformy reprezentována neomezeným záznamem (*event log*), do kterého jsou nové zprávy zapisovány. Pro každé téma je možné nastavit politiku perzistence, tj. pravidla pro postupné mazání starých zpráv. Klientské aplikace platformy Apache Kafka na tématech naslouchají nebo do nich mohou zasílat zprávy. V nástroji DomainRadar zajišťuje Apache Kafka komunikaci mezi klíčovými komponentami, vč. přenosu konfiguračních zpráv a uložení konfigurace komponent.
- **Apache Kafka Connect** — propojuje systém Kafka s databází PostgreSQL pro účely čtení doménových jmen a ukládání výsledků klasifikace.
- **Sada sběrových mikroslužeb** — sbírají informací o doménových jménech z externích zdrojů.
- **Komponenta „Data Merger“** — provádí agregaci nasbíraných dat o doménách. Je implementována pomocí platformy Apache Flink®.
- **Extraktor příznaků** — ze získaných dat o doménách extrahuje zájmové příznaky a pro každou doménu vytváří vektor těchto příznaků, na základě kterých DomainRadar následně provádí klasifikaci.

- **Klasifikační subsystém** — je tvořen sadou dílčích klasifikátorů na bázi strojového učení a agregační/rozhodovací komponentou, která kombinuje strojové učení se sadou heuristik a statistických odhadů.
- **Uživatelské rozhraní** — webové rozhraní, přes které s modulem komunikuje operátor SOC. Umožňuje zobrazovat výsledky klasifikace, ručně zadávat domény ke klasifikaci, a také konfigurovat jednotlivé funkce modulu DomainRadar.

Následuje detailní popis jednotlivých subsystémů.

Loader & Pre-filter, požadavky na vstupní data



Obrázek 4: Schéma subsystému pro načítání a filtraci doménových jmen ke zpracování.

Loader & Pre-filter je aplikace implementovaná modulárním způsobem, která má za úkol načítat z různých zdrojů doménová jména a po jednoduché filtraci je poskytnout dalším komponentám systému pro zpracování. Jejím cílem je optimalizace rychlosti celého modulu DomainRadar tak, aby zbytečně neanalyzoval domény, které nejsou validní nebo jsou zaručeně benigní.

Celá aplikace pracuje v hlavním cyklu, který má tři části:

1. načtení doménových jmen pomocí vstupních modulů,
2. filtrace doménových jmen pomocí nastavených filtrovacích modulů,
3. výstup doménových jmen pomocí nakonfigurovaných výstupních modulů.

Samotnému hlavnímu cyklu předchází fáze inicializace aplikace podle poskytnuté konfigurace. Hlavní část konfigurace je uložena ve službě Apache Kafka a je možné ji dynamicky měnit za běhu aplikace. Před spuštěním vyžaduje aplikace pouze nastavení několika proměnných prostředí, ze kterých jsou načteny údaje pro připojení ke službě Kafka.

1. inicializace spojení se službou Kafka,
2. načtení konfigurace modulu z příslušného tématu služby Kafka a samotné vytvoření instancí vstupních, filtrovacích a výstupních bloků aplikace.

Příklad hlavní konfigurace, která je uložena ve službě Apache Kafka, je dostupný v archivu s dosaženým výsledkem v souboru `DomainRadar/input/config.example.json`.

Vstupní moduly

Vstupní moduly jsou uzavřené bloky, které poskytují funkci pro získání seznamu doménových jmen. Aktuálně jsou implementovány následující vstupní moduly:

- **ELKSource**: Implementuje možnost načítání doménových jmen ze systému ELK, který je naplněn pomocí sond **suricata** ve standardním datovém formátu. Modul vyhledává záznamy, které jsou maximálně jeden den staré, a v dalších iteracích pokračuje s nově dostupnými záznamy. Jsou vyhledávány pouze DNS dotazy s otázkami na záznamy typu **A** a **AAAA**.

  Modul musí být inicializován s následující konfigurací:

  - elk_url — řetězec ukazující na dostupnou ELK databázi, která nevyžaduje přihlášení pro vyhledávání.
- **MISPSource**: Implementuje možnost načítání doménových jmen ze systému MISP. Vyhledávány jsou všechny atributy typu **domain** nebo **url**, které spadají pod konkrétní **feed**. Podobně jako ELKSource vyhledávání začíná pro záznamy staré nejvýše jeden den, v dalších iteracích jsou přidávány nové záznamy.

  Modul musí být inicializován s následující konfigurací:

  - misp_url: URL pro připojení na instanci MISP,
  - misp_key: autentizační token pro API MISP,
  - misp_feed_eventids: seznam ID feedů, které mají být použity jako zdroj doménových jmen.
- **SimpleFileSource:** Tento modul je určený především pro testování aplikace. Načítá doménová jména z lokálního souboru, a ty v každé iteraci opakovaně vrací.

  Modul musí být inicializován s následující konfigurací:

  - filename: řetězec s cestou k lokálnímu textovému souboru s doménami.

Filtry doménových jmen

Filtrovací moduly jsou uzavřené bloky, které dostávají na vstupu seznam doménových jmen a jejich výstupem je seznam **filtračních akcí** pro jednotlivé domény ze vstupu. Každá instance filtru je identifikována unikátním jménem.

Filtrační akce jsou pokyny, které určují, jak má být s doménovým jménem dál naloženo:

1. **PASS**: jméno nebylo odfiltrováno (má tedy být dále zpracována systémem DomainRadar),
2. **DROP**: jméno bylo odfiltrováno a bude zcela zahozeno,
3. **STORE**: jméno bylo odfiltrováno, ale má být zpracováno výstupními moduly, které jej označí jako filtrované. Tato jména nebudou dále systémem DomainRadar zpracována a klasifikována, nicméně budou uložena v jeho databázi.

Aktuálně implementované filtry využívají suffixové stromy **trie**, do kterých filtry při své inicializaci uloží seznam doménových jmen k vyfiltrování. Pro každou instanci filtru je nakonfigurována jedna společná filtrační akce, která je použita pro všechna jména, která jsou v sufixovém stromu nalezena. Nenalezená jména jsou označena akcí PASS.

V projektu jsou aktuálně implementovány následující filtrační moduly:

- **FileBlockListFilter**: seznam sufixů doménových jmen k vyfiltrování načítá z lokálního souboru. Modul může být použit pro testování aplikace v lokálním režimu nebo pro trvalé nastavení „zaručeně důvěryhodných“ domén.

  Modul musí být inicializován s následující konfigurací:

  - filename: řetězec s cestou k lokálnímu textovému souboru s doménami.
- **MISPFilter**: seznam sufixů doménových jmen k vyfiltrování načítá z feedu platformy MISP.

  Modul musí být inicializován s následující konfigurací:

  - misp_url: URL pro připojení na instanci MISP,
  - misp_key: autentizační token pro API MISP,
  - misp_feed_eventids: seznam ID feedů, které mají být použity jako zdroj doménových jmen k odfiltrování.
- **ValidDomainFilter**: akceptuje jakoukoliv validní doménu a odfiltruje jakoukoliv nevalidní doménu. Validita doménových jmen je ověřena pomocí regulárních výrazů knihovnou *validators*[1].
- **RandomDROPFilter**: modul náhodně filtruje doménová jména s nastavenou pravděpodobností. Je vhodný pro testování aplikace v lokálním režimu nebo ke snížení celkového objemu doménových jmen zaslaných na výstup v případě příliš objemného vstupu.

  Modul musí být inicializován s následující konfigurací:

  - drop_rate: pravděpodobnost vyfiltrování (v procentech).
- **CustomPostgresFilter**: načítá doménová jména k filtraci z databáze PostgreSQL. Modul je určen pro uživatelsky přívětivou dynamickou správu vlastních filtrů prostřednictvím webového rozhraní DomainRadar.

  Modul musí být inicializován s následující konfigurací:

  - host: adresa databázového serveru,
  - port: port databázového serveru,
  - username: uživatelské jméno pro autentizaci k databázi,
  - password: heslo pro autentizaci k databázi,
  - database: jméno databáze,

[1] Zdrojový kód funkce pro ověření validity doménového jména: https://github.com/python-validators/validators/blob/0a791b6a8a0a8e448b2de5148a03b356aabb891f/src/validators/domain.py

- filter_table_name: jméno tabulky, která obsahuje informace o dostupných filtrech,
- domains_table_name: jméno tabulky, která obsahuje domény asociované s konkrétními filtry.

Výstupní moduly

Výstupní moduly jsou uzavřené bloky, které akceptují profiltrovaný seznam párů „doménové jméno – provedená filtrační akce“ a zasílají jej dalším systémů.

V projektu jsou aktuálně implementovány následující výstupní moduly:

- **StdOutput**: vypisuje všechna doménová akce a provedené akce na standardní výstup. Modul je primárně určen pro testování aplikace.
- **PostgresOutput**: výstupní data o doménách zasílá do databáze PostgreSQL. Doménová jména jsou do cílové tabulky vkládána metodou „upsert“, tedy pokud bylo v minulosti viděno, nebude přidán nový řádek, pouze se nastaví časové razítko posledního spatření. Této vlastnosti následně využívá subsystém pro sběr a zpracování dat (viz následující sekci), který z databáze pravidelně načítá pouze nové řádky.

  Modul musí být inicializován s následující konfigurací:

  - host: adresa databázového serveru,
  - port: port databázového serveru,
  - username: uživatelské jméno pro autentizaci k databázi,
  - password: heslo pro autentizaci k databázi,
  - database: jméno databáze.

Subsystém pro sběr a zpracování dat

Základem architektury nástroje DomainRadar je pipeline pro sběr, výměnu a zpracování dat. Jejím vstupem jsou doménová jména načtená subsystémem Loader & Pre-filter. Jednotlivé komponenty pipeline sbírají data z externích zdrojů, další komponenty data agregují, extrahují z nich klasifikační příznaky, předávají je klasifikačnímu subsystému a jeho výsledky zasílají do databáze.

Celá pipeline je navržena jako řada logicky nezávislých komponent typu „vstup–transformace–výstup“ s přesně definovaným chováním a obsahem vyměňovaných zpráv. Komponenty mezi sebou komunikují prostřednictvím platformy Apache Kafka®, která řídí tok dat.

Subsystém byl navržen s ohledem na možnost horizontálního škálování. Kolektory a extraktor příznaků je možné jednoduše spustit ve více instancích, platforma Apache Kafka pak zajišťuje rozdělení práce. Pro agregace dat je využita platforma Apache Flink®, která poskytuje efektivní prostředí pro distribuované stavové výpočty nad proudy dat. V obou případech jde o software s otevřeným zdrojovým kódem. Obě platformy poskytují široké možnosti produkčního nasazení sahající od jedné samostatné instance po rozsáhlé clustery

zajišťující vysokou dostupnost, spolehlivost a odolnost proti výpadkům. Alternativně by bylo možné je nasadit v cloudovém prostředí např. s využitím řešení Confluent Cloud.

Součástí pipeline jsou následující komponenty:

- **Kolektory**, které sbírají data z externích zdrojů.
- **Komponenta pro agregaci dat „Data Merger“**, která pro každé doménové jméno agreguje nasbíraná data z různých kolektorů do jednoho datového objektu.
- **Extraktor příznaků**, který nad datovým objektem popisujícím doménové jméno provádí sérii transformaci, čímž získá vektor příznaků.
- **Klasifikační mikroslužba**, která představuje rozhraní mezi pipeline a klasifikačním subsystémem popsaným v další kapitole.
- **Apache Kafka Connect**, systém, který předává data mezi platformou Apache Kafka a databázovým systémem PostgreSQL.
- **Správce konfigurace**, který umožňuje na základě požadavků uživatele upravovat konfigurační soubory jednotlivých komponent.

Každé téma platformy Apache Kafka může být členěno na libovolný počet oddílů (*partitions*). Skupině instancí jedné klientské aplikace server dynamicky přiřazuje oddíly tak, že z jednoho oddílu čte vždy právě jedna instance (ale jedna instance může obsluhovat více oddílů). V případě, že je spuštěno více instancí, než má téma oddílů, jsou přebývající instance neaktivní a čekají jako záloha pro případ výpadku jiné, aktivní instance.

Pipeline nástroje DomainRadar využívá témata jako vstupní a výstupní kanály pro jednotlivé komponenty. Kolektory vždy naslouchají na jednom tématu, do kterého jsou publikovány požadavky na sběr. Po provedení sběru kolektor zašle výsledek do dalšího tématu, kde čeká na zpracování komponentou Data Merger. Zároveň může také zaslat požadavky pro sběr do závislých kolektorů (např. požadavek na sběr dat o IP adresách vytváří DNS kolektor, neboť právě ten z DNS získá cílové IP adresy). Na Obrázku 5 je naznačeno schéma celé pipeline — jednotlivých komponent, typů přenášených zpráv a témat, která představují komunikační kanály mezi komponentami.

Zdrojové kódy kolektorů jsou dostupné v adresáři `DomainRadar/data-pipeline` a jeho podadresářích. U každého projektu je k dispozici ukázkový konfigurační soubor s komentáři vysvětlujícími jednotlivé položky konfigurace. Konfigurace použitá v ukázkovém běhovém prostředí je po spuštění instalačního skriptu (viz kapitolu **Instalační příručka**) dostupná v adresáři `DomainRadar/infra/client_properties`.

Zone Collector

Kolektor zón přijímá název domény a určuje doménové jméno DNS zóny, která obsahuje vstupní doménové jméno. Využívá k tomu algoritmus, který se postupně v systému DNS dotazuje na SOA záznamy pro jméno, které se postupně rozšiřuje o komponenty zprava: např. pro vstup „server.fit.vut.cz“ se provede dotaz na SOA záznam pro „vut.cz.“, „fit.vut.cz.“ a „server.fit.vut.cz.“. Po prvním neúspěchu je vrácen poslední úspěšně nalezený SOA záznam.

Pokud je doménovým jménem přímo veřejně registrovatelný suffix podle seznamu Mozilla Public Suffix List[2] (např. „cz“, „co.uk“ nebo „hakodate.hokkaido.jp“), provede se rezoluce tak, že výsledkem je záznam SOA sufixu samotného. V ostatních případech se veřejně registrovatelný suffix naopak přeskočí (např. pro „fit.vut.cz“ se neprovede dotaz pouze na „cz.“).

Pro všechny DNS dotazy je použit předem nakonfigurovaný rekurzivní DNS resolver. Pokud je nalezen SOA záznam, provádí se další dotazy pro ověření přítomnosti záznamu typu DNSKEY a získání:

- IP adres (záznamů typu A a AAAA) pro primární nameserver,
- záznamů typu NS pro zónu,
- IP adres (záznamů typu A a AAAA) pro všechny sekundární nameservery ze záznamů typu NS.

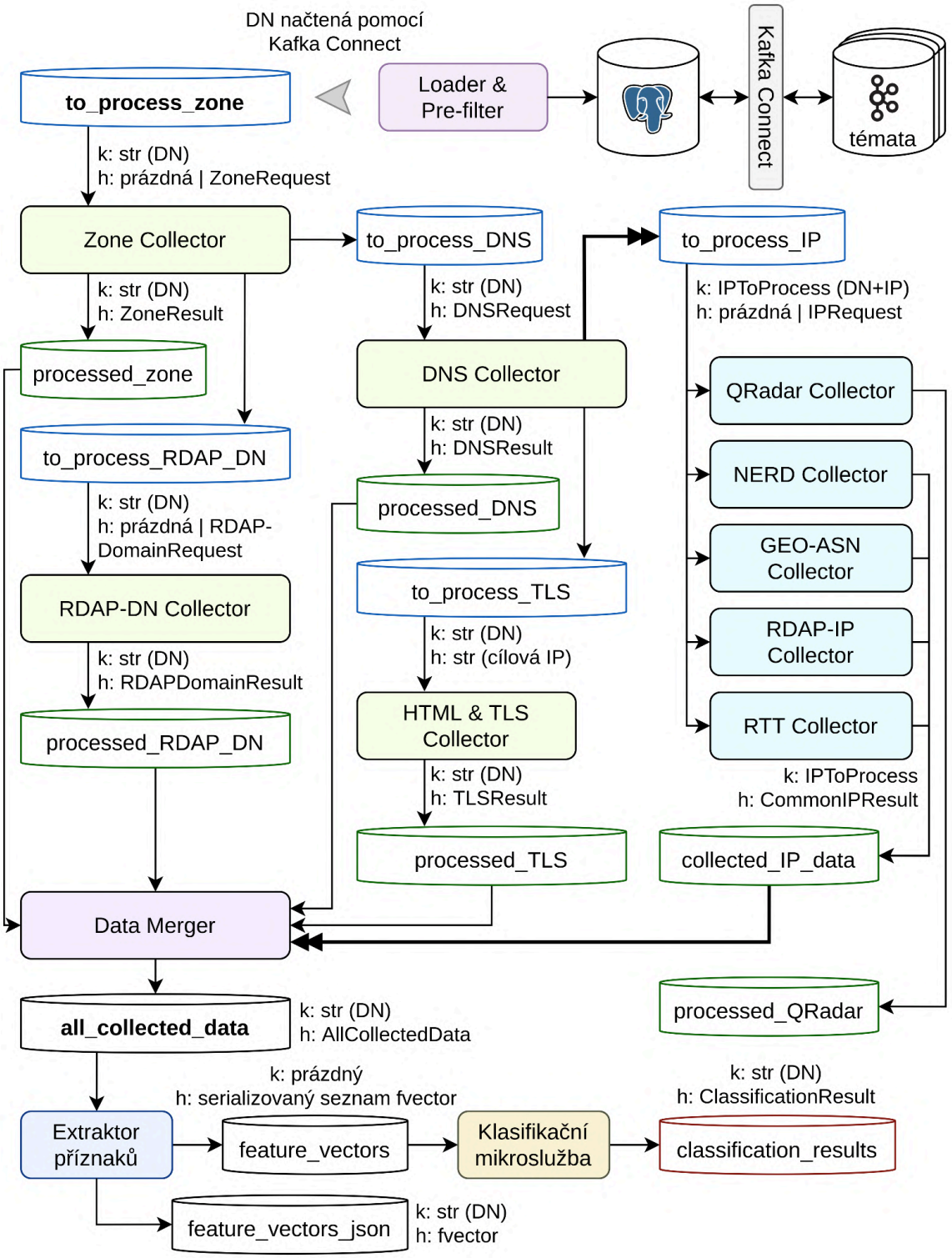
Vstupem kolektoru zón jsou obvykle přímo doménová jména načtená z databáze PostgreSQL prostřednictvím služby Kafka Connect. Jména mohou být zaslána ke zpracování také rovnou z webového rozhraní. **Výstupem** kolektoru je zpráva se zjištěnými daty, dále požadavek pro DNS kolektor a požadavek pro RDAP-DN kolektor.

DNS Collector

DNS kolektor se pro každé vstupní jméno dotazuje jeho autoritativních nameserverů (určených kolektorem zón) na DNS záznamy požadovaných nebo předem nakonfigurovaných typů a sbírá jejich hodnoty. Pro typy záznamů obsahující další doménová jména (CNAME, MX, NS) zjišťuje cílové IP adresy pomocí předem nakonfigurovaného rekurzivního resolveru.

Vstupem DNS kolektoru jsou požadavky na sběr z kolektoru zón. **Výstupem** kolektoru je zpráva se zjištěnými daty, dále požadavek pro HTML & TLS kolektor a sada požadavků pro IP kolektory.

[2] Mozilla Public Suffix List: https://publicsuffix.org/



Obrázek 5: Schéma pipeline pro sběr, výměnu a zpracování dat. Obdélníky představují jednotlivé komponenty pipeline. Barevné válce představují témata platformy Apache Kafka, která slouží jako kanály pro výměnu zpráv mezi komponentami. DN = doménové jméno, k = klíč, h = hodnota, str = string – textový řetězec, fvector = vektor příznaků.

RDAP-DN Collector

RDAP-DN kolektor používá protokol RDAP k vyhledání registračních údajů domén. Mapování TLD na RDAP endpointy je určeno podle registru organizace IANA[3]. Kolektor alternativně využívá službu WHOIS, a to v případě, že dojde k chybě, nebo když provozovatel TLD neposkytuje přístup k registračním údajům pomocí RDAP.

Kolektor implementuje mechanismus pro lokální omezení frekvence požadavků pro prevenci problému „rate limiting“ na vzdálených RDAP serverech. Dotazy na jednotlivé RDAP servery jsou řízeny nezávislými lokálními omezovači. Ty jsou implementovány s využitím algoritmu „leaky bucket“. Parametry omezovače jsou konfigurovatelné nezávisle pro jednotlivé RDAP servery. Místní omezování frekvence požadavků může být vynucováno ve dvou režimech:

- Režim fronty: Pokud je překročena kapacita omezovače, kolektor zařadí požadavek do fronty a po jisté době se je pokusí odbavit. Kolektor umožňuje také nastavit časové omezení pro požadavky zařazené do fronty. Pak kolektor vstup zahodí a vrátí pro něj chybovou zprávu, pokud nemohl být zpracován v nastaveném časovém rámci.
- Okamžitý režim: Pokud je překročena kapacita omezovače, kolektor vstup nezpracovává a okamžitě vrátí chybovou zprávu.

Nastavení režimu je součástí konfigurace kolektoru.

Vstupem RDAP-DN kolektoru jsou požadavky na sběr ze kolektoru zón. **Výstupem** kolektoru je zpráva se zjištěnými daty (nepozměněný obsah odpovědi RDAP nebo WHOIS serveru).

HTML & TLS Collector

HTML & TLS kolektor naváže TCP spojení na port 443 k IP adrese, kterou určí DNS kolektor. Pokusí se navázat TLS tunel s použitím doménového jména jako SNI. Pokud uspěje, pošle „HTTP GET“ na URL / a přečte odpověď. Při přesměrování následuje cílovou URL (počet přesměrování je nastavitelný).

Vstupem HTML & TLS kolektoru jsou požadavky na sběr z DNS kolektoru. **Výstupem** kolektoru je zpráva, která obsahuje:

- použitou verzi SSL/TLS (z prvního spojení),
- identifikátor šifrovací sady, která byla pro spojení zvolena (z prvního spojení),
- seznam certifikátů prezentovaných serverem ve formátu DER (z prvního spojení),
- obsah získaný v HTTP odpovědi (z posledního spojení).

GEO-ASN Collector

GEO-ASN kolektor vyhledává informace o geografické poloze a autonomním systému vstupní IP adresy v databázích GeoLite2 od společnosti MaxMind[4].

[3] IANA registr RDAP serverů pro TLD: https://www.iana.org/assignments/rdap-dns/rdap-dns.xhtml
[4] Databáze GeoLite2: https://dev.maxmind.com/geoip/geolite2-free-geolocation-data/

RDAP-IP Collector

RDAP-IP kolektor používá protokol RDAP k vyhledání registračních údajů o IP adresách. Mapování IP síťových prefixů na RDAP endpointy je určeno podle registru organizace IANA[5]. Výstupem RDAP-IP kolektoru je zpráva, která obsahuje nepozměněný obsah odpovědi RDAP serveru.

NERD Collector

NERD kolektor získává reputační skóre pro vstupní IP adresu z reputačního systému NERD[6] provozovaného organizací CESNET.

RTT Collector

RTT (round-trip time) kolektor provádí „ping“: odesílá několik zpráv typu ICMP Echo na vstupní IP adresu a čeká na odpovědi typu ICMP Echo Reply. Výstupem RTT kolektoru je zpráva, která obsahuje základní statistiky o proběhlém procesu: minimum RTT, průměr RTT, maximum RTT, počet odeslaných zpráv a přijatých odpovědí, jitter.

QRadar Collector

QRadar kolektor vyhledává v systému QRadar vstupní IP adresu a získává informace o případných bezpečnostních incidentech (*offenses*), které s touto adresou souvisí.

Komponenta pro agregaci dat „Data Merger“

Nasbíraná data z různých zdrojů je nutné sloučit do jednoho uceleného datového objektu, který bude předán extraktoru příznaků. **Vstupem** komponenty jsou výstupy z kolektorů, **výstupem** je pak hodnota, která zahrnuje všechna shromážděná data pro konkrétní doménové jméno.

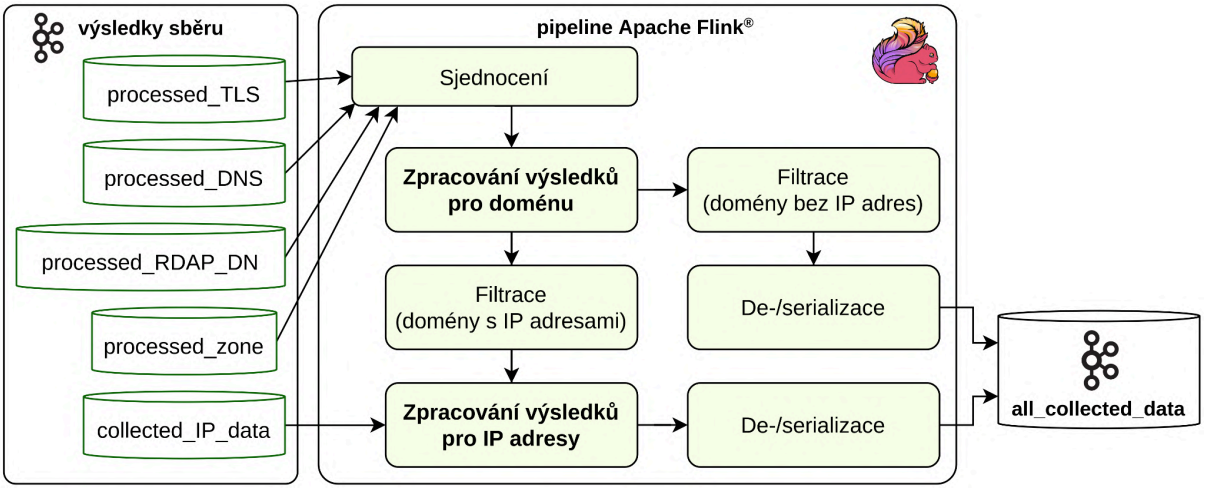
Z konceptuálního pohledu komponenta udržuje „úložiště nejlepšího stavu“ pro každý vstupní kanál (tj. výstup kolektoru). Takové úložiště je mapováním „klíč-hodnota“, kde klíče jsou doménová jména a hodnoty jsou kontejnery naplněné nejužitečnějším výsledkem z příslušného kolektoru pro dané jméno. Vztah „být užitečnější“ porovnává dva kandidátní výsledky na základě časového razítka a podle toho, zda byly úspěšné.

Kdykoli je ze vstupu načtena zpráva a její hodnota je užitečnější, aktualizuje se příslušný kontejner. Úložiště se pravidelně čistí, aby se odstranily záznamy, které nebyly aktualizovány po delší dobu.

Komponenta je realizována jako samostatná proudová ETL (*extract–transform–load*) pipeline s využitím platformy Apache Flink. Obrázek 6 demonstruje jednotlivé části této pipeline. Vstupem jsou zprávy s výsledky sběru načtené z jednotlivých témat Apache Kafka. Základem pipeline jsou vlastní stavové operátory `DomainEntriesProcessFunction`

[5] IANA registry RDAP serverů pro IPv4 a IPv6 adresy: https://www.iana.org/assignments/rdap-ipv4/rdap-ipv4.xhtml, https://www.iana.org/assignments/rdap-ipv6/rdap-ipv6.xhtml
[6] Systém NERD: https://nerd.cesnet.cz/

a `IPEntriesProcessFunction`, které implementují úložiště nejlepšího stavu a další logiku pro vyřazování příliš starých položek (zvlášť pro výsledky závislé pouze na doménovém jméně a pro výsledky závislé také na IP adrese). Výstupem je zpráva, která obsahuje kompletní datový objekt pro jedno doménové jméno a všechny IP adresy, které pro něj byly získány.



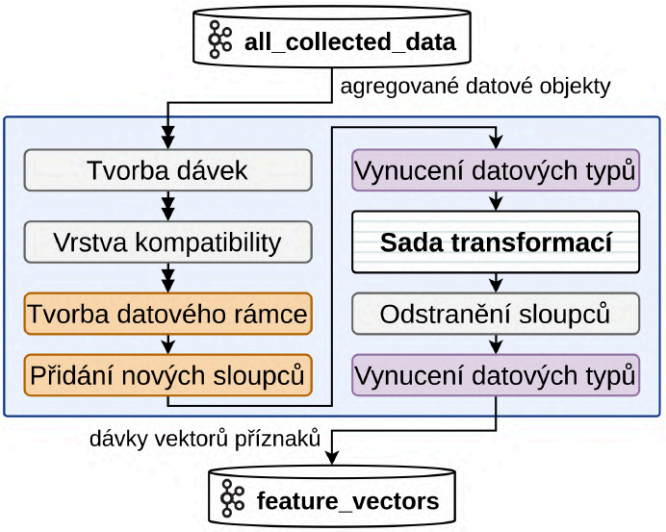
Obrázek 6: Pipeline komponenty pro agregaci dat založená na platformě Apache Flink®.

Extraktor příznaků

Extraktor příznaků čte agregované datové objekty a zpracovává data nasbíraná pro doménové jméno do podoby vektoru číselných příznaků, která je vhodná pro použití v klasifikaci. Samotné příznaky a jejich význam jsou popsány níže v sekci Klasifikační subsystém, tato sekce se zabývá především strukturou mikroslužby, která je součástí pipeline pro sběr a zpracování dat.

Extraktor je navržen jako sada transformačních modulů, jejichž vstupem je datový rámec s dávkou zpracovávaných záznamů (tabulka, kde každý řádek odpovídá jednomu doménovému jménu a sloupce označují vstupní atributy datových objektů nebo cílové příznaky) a výstupem je pozměněný datový rámec, který je obohacený o nově vypočítané příznaky. Extraktor čte vstupní zprávy a ukládá je do mezipaměti nakonfigurované velikosti. Po naplnění mezipaměti nebo uplynutí nakonfigurovaného času od poslední zprávy je obsah mezipaměti převeden na datový rámec, na který jsou postupně aplikovány všechny transformace. Poslední transformací je zahození nepotřebných sloupců (např. s původními atributy datového objektu). Výsledné datové rámce jsou serializovány v binárním formátu Feather V2[7] a zaslány do výstupního tématu. Pomocnými částmi procesu jsou také kontroly datových typů a vrstva kompatibility, která provádí předzpracování datového objektu do formátu, se kterým transformace pracují. Schéma extraktoru je zobrazeno na Obrázku 7.

[7] Feather V2: https://arrow.apache.org/docs/python/feather.html



Obrázek 7: Architektura extraktoru příznaků.

Extraktor je implementován v jazyce Python, a to s ohledem na snadnou rozšiřitelnost. Transformace jsou umístěny v samostatných modulech[8]:

- **dns.py** (39 příznaků): extrakce z dat získaných z DNS (např. počty záznamů jednotlivých typů, histogram hodnot TTL, příznaky popisující textové položky ze SOA a TXT záznamů).
- **geo.py** (18 příznaků): agregace geolokačních dat získaných pro IP adresy z databáze GeoLite2 City (např. počty zemí, unikátní identifikátor kombinace zemí).
- **html.py** (87 příznaků): extrakce z HTML obsahu stránky (např. délka slov, počet odkazů nebo využití skriptů). Příznaky se získávají s využitím regulárních výrazů a knihovny BeautifulSoup.
- **ip.py** (8 příznaků): agregace informací o IP adresách vč. autonomních systémů získaných z databáze GeoLite2 ASN (např. počet, počet unikátních autonomních systémů).
- **lexical.py** (62 příznaků): příznaky získané pouze z doménového jména samotného na základě analýzy řetězce a podřetězců.
- **rdap_dn.py** (14 příznaků): extrakce z registračních informací pro doménové jméno získaných pomocí protokolů RDAP nebo WHOIS (např. délka života domény, příznaky popisující charakter řetězců označujících kontakty).
- **rdap_ip.py** (10 příznaků): extrakce z registračních informací pro IP adresy získaných pomocí protokolu RDAP (např. nejkratší a nejdelší IPv4 a IPv6 prefixy).
- **tls.py** (24 příznaků): extrakce z TLS handshake a certifikátů prezentovaných serverem (např. počet jmen v SAN atributech).
- **drop_columns.py**: zajišťuje zahození nepotřebných sloupců datového rámce.

[8] Viz `DomainRadar/data-pipeline/python/extractor/extractor/transformations`.

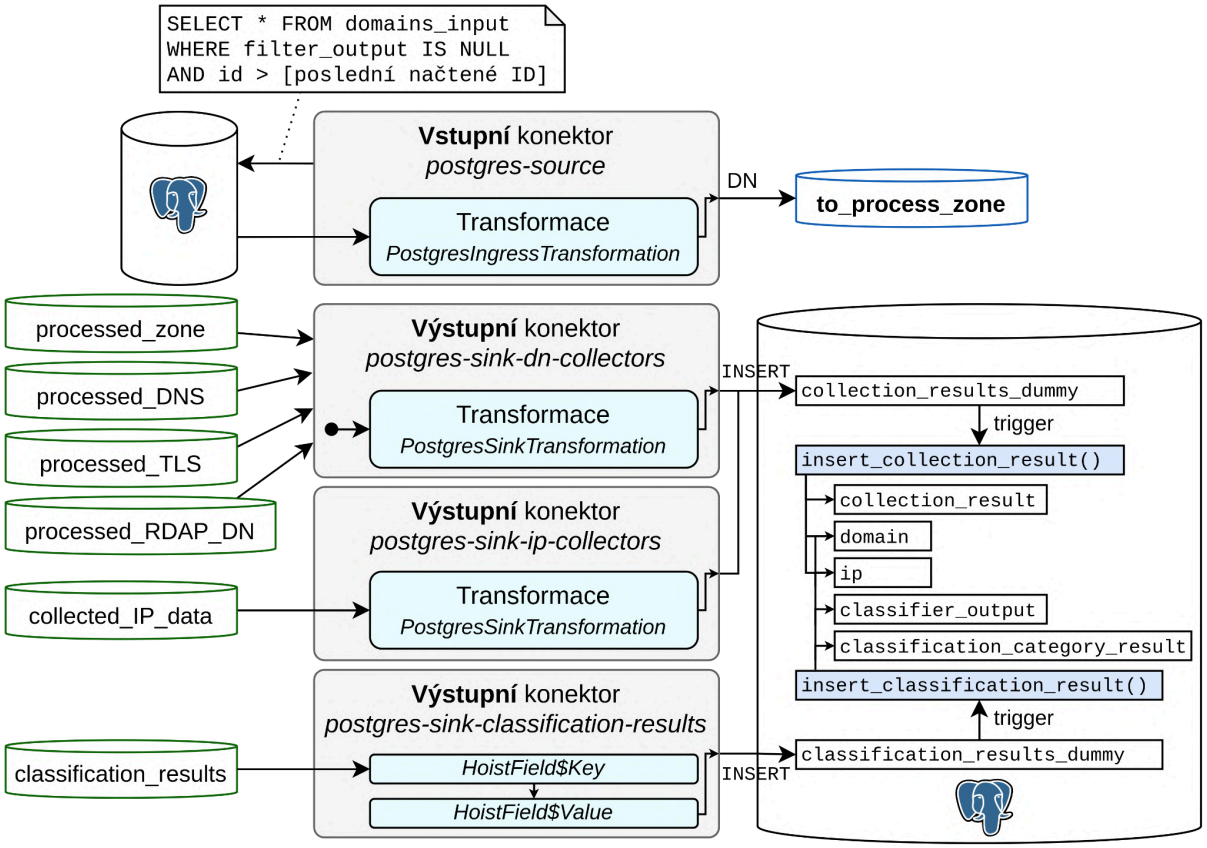
Klasifikační mikroslužba

Klasifikační mikroslužba je program implementovaný v jazyce Python, který čte ze služby Apache Kafka serializované datové rámce produkované extraktorem příznaků a předává je klasifikačnímu subsystému (viz následující kapitolu). Výslednou zprávu o výsledcích klasifikace zasílá do příslušného tématu.

Pro lepší využití prostředků mikroslužba umožňuje využít paralelismus na úrovni procesů. Klasifikační subsystém je pak spuštěn v mnoha instancích, mezi které jsou jednotlivé dávky vektorů příznaků ke zpracování distribuovány.

Výměna dat mezi službou Apache Kafka a databází

Jak naznačují Obrázky 3 a 5, rozhraním mezi tématy služby Apache Kafka a databázovým serverem PostgreSQL je služba Kafka Connect, která poskytuje univerzální rámec pro výměnu dat mezi službou Kafka a dalšími systémy.



Obrázek 8: Architektura výměny zpráv s využitím Apache Kafka Connect.

Výměnu dat poskytují konektory (*connectors*), kterých je možné pro jednu instanci Kafka Connect nakonfigurovat libovolné množství. Konektor buď čte z určeného tématu (resp. témat) a zapisuje do cílového systému (*sink connector*), nebo naopak čte ze zdrojového systému a zapisuje do tématu (*source connector*). Součástí konfigurace konektoru jsou konvertory (*converters*), které data de-/serializují a převádějí do podoby vhodné pro cílový

systém. Před předáním zprávy je možné v konektoru nastavit také sadu transformací, které běží uvnitř služby Kafka Connect a předzpracovávají zprávu před předáním do cílového systému. Služba je modulární: vlastní typy konektorů, konvertory i transformace lze dodávat ve formě zásuvných modulů implementovaných v jazyce Java.

Obrázek 8 naznačuje, jak je v nástroji DomainRadar platforma Kafka Connect využita. Vstupní konektor pravidelně získává z databázové tabulky nová doménová jména a s využitím vlastní transformace je vkládá do vstupního tématu pipeline. Zprávy z výstupních témat kolektorů a klasifikační mikroslužby jsou čteny výstupními konektory, upraveny pomocí transformací a vkládány pomocí dotazu INSERT do „dummy“ tabulek. Na ty je navázaný databázový trigger, který zprávy zachytí a vytvoří podle nich skutečné záznamy v řadě tabulek.

Klasifikační subsystém

Cílem klasifikačního subsystému je určit míru rizika dané domény a vygenerovat klasifikační report. Vstupem jsou klasifikované domény a s nimi související vektory příznaků. Samotnou klasifikaci provádí klasifikační pipeline, která provádí sadu operací rozdělenou do několika fází. Vektor příznaků, pipeline a jednotlivé klasifikátory budou popsány v následujících sekcích.

Vstupní vektor příznaků

Vektor příznaků pro klasifikaci sestává ze příznaků, které jsou rozděleny do několika kategorií:

- **Lexikální příznaky** (`lex_*`) — Popisují samotné doménové jméno a jeho strukturu. Obsahují např. délku doménového jména, počet subdomén, normalizovanou entropii, či přítomnost specifických klíčových slov souvisejících s phishingem. Jsou užitečné při identifikaci podezřelých názvů domén, které často využívají netradiční strukturu nebo specifická slova jako návnadu uživatele. Přesný výčet příznaků ukazuje sekce „Lexikální příznaky doménového jména“ na Obrázku 9.
- **Příznaky založené na DNS** (`dns_*`) — Tato kategorie analyzuje informace získané ze záznamů DNS. Mezi příznaky patří počty různých typů záznamů (A, MX, NS, TXT), hodnoty TTL, nebo přítomnost DNSSEC. Tyto příznaky pomáhají rozlišit legitimní domény, které mají mnohem častěji robustní DNS konfiguraci, od podezřelých. Např. u rychle vznikajících (tzv. „hi-flux“) domén jsou častým rysem nízké hodnoty TTL, zatímco u domén pro nadnárodní služby bývá často vícero záznamů typu A, MX apod. Přesný výčet příznaků popisuje sekce „Příznaky založené na DNS“ na Obrázku 9.
- **Příznaky založené na IP** (`ip_*`) — Tyto analyzují vlastnosti IP adres asociovaných s doménou. Popisují např. celkový počet IP adres, poměru IPv4/IPv6, entropii prefixů apod. Malá rozmanitost IP adres mnohdy ukazuje škodlivé domény, zatímco legitimní služby mívají vyšší diverzitu. Přesný výčet příznaků je popsán v sekci „Příznaky založené na IP“ na Obrázku 9.

- **Příznaky založené na RDAP/Whois** (`rdap_*`) — Tato kategorie zahrnuje příznaky, které se extrahují z dat získaných službou RDAP či WHOIS (není-li RDAP pro danou doménu k dispozici). Zahrnují údaje o registraci domény, jako je stáří domény, délka registrace nebo jméno registrátora. Mezi staršími doménami a doménami registrovanými na delší dobu bývají spolehlivější služby, zatímco krátce registrované mohou ukazovat na phishing či šíření malware. Přesný výčet příznaků popisuje sekce „Příznaky založené na RDAP“ na Obrázku 9.
- **Příznaky založené na TLS** (`tls_*`) — Tyto příznaky popisují data získaná z certifikátů a handshakes protokolu TLS. Popisují např. délku řetězu certifikátů, dobu jejich platnosti a přítomnost různých bezpečnostních rozšíření protokolu TLS. Škodlivé domény často používají jednodušší nebo méně obvyklé konfigurace, případně vůbec nemusí podporovat šifrování přes TLS, zatímco legitimní weby mají častěji certifikáty od uznávaných autorit. Přesný výčet popisuje sekce „Příznaky založené na TLS” na Obrázku 9.
- **Geolokační příznaky** (`geo_*`) — Tato kategorie obsahuje příznaky popisující geografické informace spojené s IP adresami serverů asociovaných s doménou. Popisují např. počet unikátních zemí a světových kontinentů, kde se servery nacházejí, průměrnou a centrální zeměpisnou šířku, resp. délku a jejich rozptyl. Samotné geolokační příznaky typicky nestačí ke spolehlivému rozhodnutí o tom, zda doména je či není maligní, nicméně k tomuto rozhodování přispívají. Zatímco domény nadnárodních služeb jsou rozptýleny ve vícero lokalitách ve světě, domény hostující závadné služby jsou často koncentrovány v určitých regionech. Přesný výčet příznaků popisuje sekce „Příznaky geolokace“ v Obrázku 9.
- **Příznaky založené na HTML** (`html_*`) — Tyto příznaky popisují obsah HTML/DOM webové stránky (běží-li na dané doméně). Zaměřují se konkrétně na výskyt konkrétních tagů jazyka HTML, metod jazyka JavaScript (JS), či specifických klíčových slov. Závadné weby např. často obsahují značný výskyt elementů `<iframe>`, odkazů na weby třetích stran, či přesměrování přes JS. Přesný výčet těchto příznaků popisuje sekce „Příznaky založené na HTML“ na Obrázku 10.

Klasifikační pipeline

Proces klasifikace konkrétního doménového jména provádí na základě vektoru příznaků Klasifikační pipeline, která je jádrem klasifikačního subsystému. Tento proces má několik fází, které znázorňuje Obrázek 11. Zde je vyobrazena celá klasifikační pipeline od vstupu domény s vektorem příznaků až po vytvoření reportu.

Lexikální příznaky doménového jména (lex_)	
Název	**Popis**
name_len	Délka názvu domény
has_digit	Příznak, zda název domény obsahuje číslici
phishing_kw_count	Počet výskytů 47 phishingových klíčových slov
consecutive_chars	Nejdelší sekvence po sobě jdoucích znaků
tld_len	Délka nejvyšší domény (TLD)
tld_abuse_score	Skóre nejzneužívanější TLD
tld_hash	Hash nejvyšší domény
sld_len	Délka druhé úrovně domény (SLD)
sld_norm_entropy	Normalizovaná entropie SLD
sld_phishing_kw_count	Počet výskytů 47 phishingových klíčových slov v SLD
sub_count	Počet subdomén (úrovní)
stld_unique_char_cnt	Počet unikátních znaků v TLD a SLD
begins_with_digit	Příznak, zda název začíná číslem
www_flag	Příznak, zda název začíná www“
sub_max_conson_len	Nejdelší sekvence souhlásek v subdoménách
sub_norm_entropy	Normalizovaná entropie subdomén
{sub,sld}_digit_count	Počet číslic v subdoménách a SLD
{sub,sld}_digit_ratio	Podíl číslic v subdoménách a SLD
{sub,sld}_vowel_count	Počet samohlásek v subdoménách a SLD
{sub,sld}_vowel_ratio	Podíl samohlásek v subdoménách a SLD
{sub,sld}_consonant_count	Počet souhlásek v subdoménách a SLD
{sub,sld}_consonant_ratio	Podíl souhlásek v subdoménách a SLD
{sub,sld}_nonalnum_count	Celkový počet pomlček v subdoménách a SLD
{sub,sld}_nonalnum_ratio	Podíl nepísmenných znaků v subdoménách a SLD
{sub,sld}_hex_count	Počet hexadecimálních znaků v subdoménách a SLD
{sub,sld}_hex_ratio	Podíl hexadecimálních znaků v subdoménách a SLD
bigram_matches	Počet shod phishingových bigramů
trigram_matches	Počet shod phishingových trigramů
tetragram_matches	Počet shod phishingových tetragramů
pentagram_matches	Počet shod phishingových pentagramů
avg_part_len	Průměrná délka částí názvu domény
stdev_part_lens	Směrodatná odchylka délek částí názvu domény
longest_part_len	Délka nejdelší části názvu domény
shortest_sub_len	Délka nejkratší subdomény

Příznaky založené na DNS (dns_)	
Název	**Popis**
A_count	Počet A záznamů
AAAA_count	Počet AAAA záznamů
MX_count	Počet MX záznamů
NS_count	Počet NS záznamů
TXT_count	Počet TXT záznamů
CNAME_count	Počet CNAME záznamů
resolved_rec_types	Počet nalezených RRsetů
has_dnskey	Příznak, zda je v zóně DNSKEY RRset
dnssec_score	Skóre DNSSEC
ttl_avg	Průměrná hodnota TTL napříč RRsety
ttl_stdev	Směrodatná odchylka TTL napříč RRsety
ttl_low	Počet RRsetů s TTL $\in [0, 100]$
ttl_mid	Počet RRsetů s TTL $\in [101, 500]$
ttl_distinct_count	Počet různých TTL hodnot napříč RRsety
soa_refresh	Parametr obnovy SOA
soa_retry	Parametr opakování SOA
soa_expire	Parametr expirace SOA
soa_min_ttl	Minimální TTL SOA
dn_in_mx	Příznak, zda je mailserver subdoménou DN
txt_ext_verif_score	Počet ověřovacích řetězců v TXT záznamech
txt_spf_exists	Příznak, zda existuje SPF záznam v TXT
txt_dkim_exists	Příznak, zda existuje DKIM záznam v TXT
txt_dmarc_exists	Příznak, zda existuje DMARC záznam v TXT
Lexikální příznaky založené na DNS	
zone_level	Počet subdomén v zóně DN
zone_digits	Počet číslic v zóně DN
zone_len	Počet znaků v zóně DN
zone_entropy	Normalizovaná entropie zóny DN
soa_pri_ns_level	Počet subdomén v primárním NS DN
soa_pri_ns_digits	Počet číslic v primárním NS DN
soa_pri_ns_len	Počet znaků v primárním NS DN
soa_pri_ns_entropy	Normalizovaná entropie primárního NS DN
soa_email_level	Počet subdomén v e-mailu správce
soa_email_digits	Počet číslic v e-mailu správce
soa_email_len	Počet znaků v e-mailu správce
soa_email_entropy	Normalizovaná entropie e-mailu správce
mx_avg_len	Průměrný počet znaků DN v MX záznamech
mx_avg_entropy	Průměrná normalizovaná entropie DN v MX záznamech
txt_avg_entropy	Průměrná normalizovaná entropie hodnot TXT záznamů

Příznaky založené na IP (ip_)	
Název	**Popis**
count	Počet IP adres
mean_average_rtt	Průměrná RTT všech ICMP Echo požadavků
ip_v4_ratio	Podíl IPv4 adres vůči všem IP adresám
entropy	Celková entropie všech IP prefixů /16 (/64 pro IPv6)
as_address_entropy	Entropie IP prefixů autonomních systémů (AS)
asn_entropy	Entropie čísel autonomních systémů (ASN)
distinct_as_count	Počet jedinečných čísel AS

Příznaky založené na RDAP (rdap_)	
Název	**Popis**
Související s doménovým jménem	
registration_period	Rozdíl mezi datem expirace a registrace
domain_age	Počet dnů od registrace domény
time_from_last_change	Počet dnů od poslední změny
domain_active_time	min(dnešek, expirace) - datum registrace
has_dnssec	Příznak, zda doména používá DNSSEC
registrar_name_len	Délka názvu registrátora
registrar_name_entropy	Entropie názvu registrátora
registrar_name_hash	Hash názvu registrátora
registrant_name_len	Délka názvu držitele domény
registrant_name_entropy	Entropie názvu držitele domény
admin_name_len	Délka jména administrativního kontaktu
admin_name_entropy	Entropie jména administrativního kontaktu
admin_email_len	Délka e-mailu administrativního kontaktu
admin_email_entropy	Entropie e-mailu administrativního kontaktu
Související s IP adresami domény	
ip_v4_count	Počet IP adres rozpoznaných jako IPv4
ip_v6_count	Počet IP adres rozpoznaných jako IPv6
ip_shortest_v4_prefix_len	Délka nejkratšího IPv4 prefixu
ip_longest_v4_prefix_len	Délka nejdelšího IPv4 prefixu
ip_shortest_v6_prefix_len	Délka nejkratšího IPv6 prefixu
ip_longest_v6_prefix_len	Délka nejdelšího IPv6 prefixu
ip_avg_admin_name_len	Průměrná délka jména správce IP adres
ip_avg_admin_name_ent	Průměrná entropie jména správce IP adres
ip_avg_admin_email_len	Průměrná délka e-mailu správce IP adres
ip_avg_admin_email_ent	Průměrná entropie e-mailu správce IP adres

Příznaky založené na TLS (tls_)	
Název	**Popis**
chain_len	Délka certifikačního řetězce
is_self_signed	Příznak, zda jde o tzv. self-signed certifikát
root_authority_hash	Hash názvu kořenové certifikační autority
leaf_authority_hash	Hash názvu koncové certifikační autority
leaf_cert_validity_len	Délka období platnosti koncového certifikátu
negotiated_version_id	Vyjednaná verze TLS (TLSv1.x)
negotiated_cipher_id	Identifikátor vyjednaného šifrovacího algoritmu TLS
root_cert_validity_len	Délka období platnosti kořenového certifikátu
broken_chain	Příznak, zda existuje certifikát, který nebyl nikdy platný
expired_chain	Příznak, zda existuje expirovaný certifikát v řetězci
total_extension_count	Celkový počet rozšíření ve všech certifikátech v řetězci
critical_extensions	Počet rozšíření označených jako “kritická”
with_policies_crt_count	Počet certifikátů obsahujících rozšíření *policies*
percentage_with_policies	Podíl certifikátů obsahujících rozšíření *policies*
x509_anypol_crt_count	Počet certifikátů v řetězu, které přímo nevynucují žádnou konkrétní bezpečnostní politiku
iso_pol_crt_count	Počet bezpečnostních politik zjištěných z prostoru OID 1.*
isoitu_pol_crt_count	Počet bezpečnostních politik zjištěných z prostoru OID 2.*
subject_count	Počet alternativních názvů subjektů (SAN) v koncovém certifikátu
unique_SLD_count	Počet unikátních doménových jmen v SAN
server_auth_crt_count	Počet certifikátů s “Autentizací webového serveru”
client_auth_crt_count	Počet certifikátů s “Autentizací webového klienta”
CA_certs_in_chain_ratio	Podíl certifikátů CA v řetězci
common_name_count	Počet běžných názvů v certifikačním řetězci

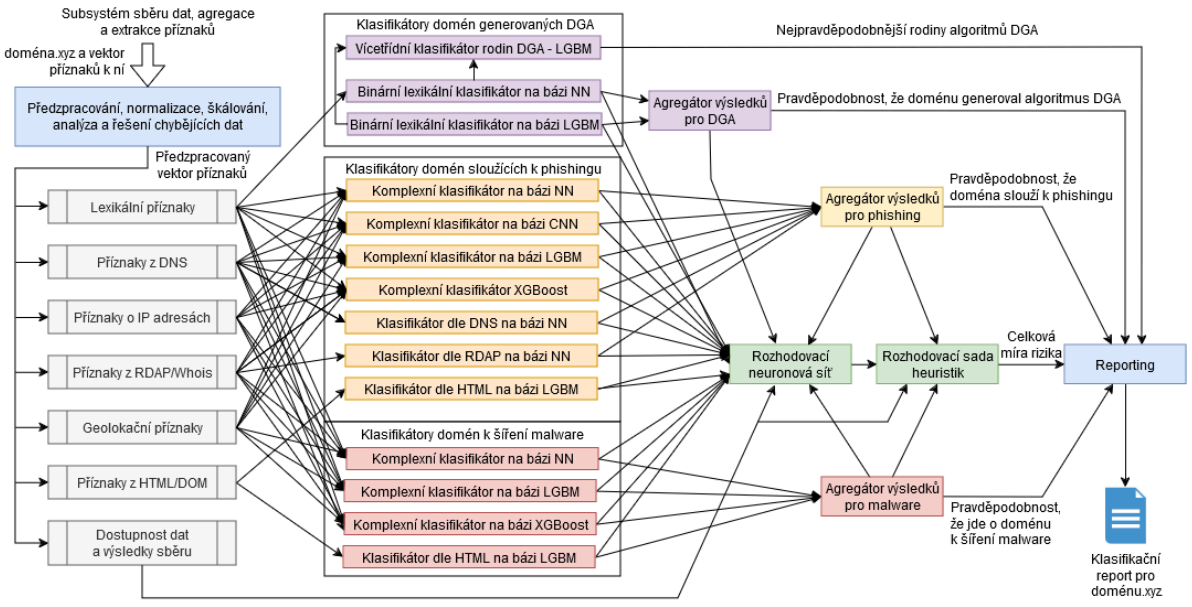
Příznaky geolokace (geo_)	
Název	**Popis**
countries_count	Počet různých zemí
countries_hash	Unikátní hash pro každou kombinaci zemí
continent_hash	Unikátní hash pro každou kombinaci kontinentů
lat_stdev	Směrodatná odchylka zeměpisné šířky IP adres
lon_stdev	Směrodatná odchylka zeměpisné délky IP adres
mean_lat	Průměrná zeměpisná šířka IP adres
mean_lon	Průměrná zeměpisná délka IP adres
centroid_lat	Středová zeměpisná šířka IP adres
centroid_lon	Středová zeměpisná délka IP adres

Obrázek 9: Vektor příznaků pro klasifikací – 1. část: lexikální příznaky, příznaky z DNS, IP, RDAP/Whois, TLS a geolokačních informací.

Příznaky založené na HTML (html_)	
Název	**Popis**
num_of_words	Počet slov v HTML
num_of_lines	Počet řádků HTML
unique_words	Počet unikátních slov
average_word_len	Průměrná délka slov
blocked_keywords_label	Příznak blokovaných klíčových slov
num_of_blank_spaces	Počet prázdných míst
num_of_tags	Celkový počet HTML tagů
num_of_paragraphs	Počet odstavců <p>
num_of_divs	Počet tagů
num_of_titles	Počet nadpisů
num_of_external_js	Počet externích skriptů
num_of_links	Počet tagů <link>
num_of_scripts	Počet skriptů <script>
num_of_scripts_async	Počet asynchronních skriptů
num_of_scripts_type	Počet skriptů s atributem typu
num_of_anchors	Počet odkazů <a>
num_of_anchors_to_hash	Počet odkazů <a> na "#"
num_of_anchors_to_https	Počet odkazů <a> na HTTPS
num_of_anchors_to_com	Počet odkazů <a> na domény "*.com"
num_of_inputs	Počet vstupních polí <input>
num_of_input_password	Počet vstupních polí na hesla
num_of_hidden_elements	Počet skrytých elementů
num_of_input_hidden	Počet skrytých vstupních polí
num_of_objects	Počet externích objektů <object>
num_of_embeds	Počet vložených objektů <embed>
num_of_frame	Počet rámů <frame>
num_of_iframe	Počet rámů inline <iframe>
num_of_iframe_src	Počet rámů <iframe> s atributem "src"
num_of_iframe_src_https	Počet rámů <iframe>, kde "src" vede na HTTPS
num_of_center	Počet tagů pro centrování
num_of_imgs	Počet obrázků <img>
num_of_imgs_src	Počet obrázků s atributem <src>
num_of_meta	Počet meta značek <meta>
num_of_links_href	Počet odkazů <link> s atributem "href"
num_of_links_href_https	Počet odkazů <link> na HTTPS
num_of_links_href_css	Počet odkazů <link> na CSS
num_of_links_type	Počet odkazů <link> s atributem "type"
num_of_link_type_app	Počet odkazů <link> na RSS
num_of_link_rel	Počet odkazů <link> s atributem "rel"
num_of_all_hrefs	Celkový počet odkazů s atributem "href"
num_of_form_action	Počet formulářů s atributem "action"
num_of_form_http	Počet formulářů směřujících na HTTP
num_of_strong	Počet zvýrazněních tučným písmem pomocí

Název	Popis
no_hrefs	Příznak absence odkazů
internal_href_ratio	Podíl interních odkazů "href"
num_of_internal_hrefs	Počet interních odkazů "href"
external_href_ratio	Podíl externích odkazů "href"
num_of_external_href	Počet externích odkazů "href"
num_of_icon	Počet ikon
icon_external	Příznak, že ikona je externí
num_of_form_php	Počet formulářů odkazujících PHP
num_of_form_hash	Počet formulářů odkazujících na "#"
num_of_form_js	Počet JavaScriptových formulářů
malicious_form	Příznak škodlivého formuláře
most_common	Poměr výskytu nejčastějšího odkazu přes "href"
num_of_css_internal	Počet odkazů "href" na interní CSS
num_of_css_external	Počet externích "href" na externí CSS (mimo web)
num_of_anchors_to_content	Počet odkazů <a> na "#content"
num_of_anchors_to_void	Počet odkazů <a> na "javascript:void(0)"
create_element	Počet výskytů metody createElement()
write	Počet výskytů metody write()
char_code_at	Počet výskytů metody charCodeAt()
concat	Počet výskytů metody concat()
escape	Počet výskytů metody escape()
eval	Počet výskytů metody eval()
exec	Počet výskytů metody exec()
from_char_code	Počet výskytů metody fromCharCode()
link	Počet výskytů metody link()
parse_int	Počet výskytů metody parseInt()
replace	Počet výskytů metody replace()
search	Počet výskytů metody search()
substring	Počet výskytů metody substring()
unescape	Počet výskytů metody unescape()
add_event_listener	Počet výskytů metody addEventListener()
set_interval	Počet výskytů metody setInterval()
set_timeout	Počet výskytů metody setTimeout()
push	Počet výskytů metody push()
index_of	Počet výskytů metody indexOf()
document_write	Počet výskytů metody document.write()
get	Počet výskytů metody get()
find	Počet výskytů metody find()
document_create_element	Počet výskytů metody document.createElement()
window_set_timeout	Počet výskytů metody window.setTimeout()
window_set_interval	Počet výskytů metody window.setInterval()
hex_encoding	Počet sekvencí escape pro hexadecimální znaky
unicode_encoding	Počet sekvencí escape pro znaky v unicode
long_variable_name	Počet proměnných v JS s názvem o alespoň 20 znacích

Obrázek 10: Vektor příznaků pro klasifikací – 2. část: příznaky popisující HTML/DOM a JS.



Obrázek 11: Architektura klasifikační pipeline

Vstupem klasifikační pipeline je doménové jméno a odpovídající vektor příznaků, který je detailněji popsán dále v tomto dokumentu. Konkrétně tento vektor tvoří lexikální příznaky, příznaky z DNS, příznaky o IP adresách, příznaky z RDAP/Whois, geolokační příznaky a příznaky z HTML/DOM. Tento vektor je dále doplněn informacemi o dostupnosti jednotlivých dat a výsledcích sběru.

Proces zpracování konkrétní domény má několik fází:

1. **Předzpracování:** Provede se normalizace dat, škálování a také řešení chybějících dat, které jsou nahrazeny vhodnou hodnotou (v naší implementaci -1).
2. **Spuštění dílčích klasifikátorů:** Doména je klasifikována klasifikátory, které určují:
    a. míru podobnosti s algoritmicky generovanými doménami pomocí algoritmů DGA,
    b. míru podobnosti s doménami, které slouží pro hosting phishingu a
    c. míru podobnosti s doménami, které slouží k šíření malware.
3. **Agregace výsledků a rozhodování:** Pro dílčí skupiny klasifikátorů se vytvoří agregované výsledky, které slouží jednak k rozhodování, jednak pro generování reportu. Určení celkové míry rizika (rozhodování) provádí dva subsystémy:
    a. rozhodovací neuronová síť,
    b. sada rozhodovacích heuristik.
4. **Generování reportu:** V poslední fázi je vygenerován výsledek klasifikace (report) ve formátu JSON, který obsahuje celkovou míru hrozby, výsledky dílčích klasifikátorů a doprovodný textový popis výsledku.

Poznámka: Tento proces je popsán zjednodušeně jako zpracování jedné konkrétní domény. Pro dosažení maximální rychlosti výpočtu a efektivity využití hardwarových prostředků implementace reálně používá ještě dvě optimalizace:

- **Dávkové zpracování** — vstupem může být (a typicky je) více domén a jejich vektorů příznaků, jde tedy ve skutečnosti o 2D matici dat, která je implementována jako Pandas DataFrame. Domény jsou zpracovávány a klasifikovány souběžně, přičemž výstupem je vektor výsledků.
- **Paralelizace pipeline** — subsystém je navržen tak, aby byl spustitelný ve více souběžných instancích. Výše uvedená klasifikační mikroslužba tedy reálně vytvoří několik paralelně běžících instancí této pipeline, přičemž v každé jsou v jeden moment zpracovávány různé dávky domén a jejich vektorů příznaků.

Typy klasifikátorů

Jak ukazuje Obrázek 11, klasifikátory lze dělit:

- dle **typu vstupních příznaků** na:
    - **komplexní klasifikátory** — využívají příznaky z vícero různých kategorií,
    - **klasifikátory na základě konkrétního zdroje dat** — např. klasifikace na základě příznaků pouze z DNS, či pouze z RDAP/Whois apod.,
- dle **určení** na:
    - klasifikátory **domén generovaných algoritmy DGA**,

- klasifikátory **phishingových domén**,
- klasifikátory **malwarových domén**.

Jednotlivé klasifikátory budou nyní detailněji vysvětleny.

Klasifikátory phishingových domén

DomainRadar implementuje následující klasifikátory phishingových domén:

- **Komplexní klasifikátor na bázi NN** — Tento klasifikátor využívá všechny příznaky kromě těch, které jsou extrahovány z kódu HTML. Jedná se o binární klasifikátor, který klasifikuje do dvou tříd: a) benigní doména, b) phishingová doména. Architektura staví na principu dopředné (feedforward) neuronové sítě, která obsahuje vstupní vrstvu, 12 skrytých vrstev a výstupní vrstvu. Počet neuronů vstupní vrstvy odpovídá počtu příznaků. Výstupní vrstva obsahuje jediný neuron, jehož výstupem je pravděpodobnost, že doména hostuje phishing. Skryté vrstvy se postupně rozšiřují až na 256 neuronů a následně zužují až do jednoho neuronu výstupní vrstvy. Mezi skrytými vrstvami jsou také dvě dávkové normalizační vrstvy a dvě vrstvy typu *Dropout*, které spolu s technikou *Early Stopping* slouží jako ochrana proti přetrénování modelu.

- **Komplexní klasifikátor na bázi CNN** — Jde o binární klasifikátor, který využívá *konvolučních neuronových sítí (Convolutional Neural Networks — CNN)*. Tyto sítě jsou široce využívány pro úkoly detekce a klasifikace obrazových dat. V případě zpracování rozsáhlého množství příznaků lze vstupní doménová data interpretovat jako obrazovou matici. Jednotlivé příznaky jsou transformovány do šedotónového formátu, což umožňuje jejich následné zpracování konvoluční neuronovou sítí. V tomto konkrétním případě je použitý klasifikátor složen z kombinace dvou konvolučních vrstev a hluboké neuronové sítě. Konvoluční neuronové sítě, aplikované v kontextu klasifikace maligních domén, představují experimentální přístup pro detekci, přičemž dosažené skóre F1 překračuje hodnotu 0,98.

- **Komplexní klasifikátor na bázi LGBM** — Tento binární klasifikátor využívá metodu Light Gradient-Boosting Machine (LightGBM), která vytváří silný model postupným přidáváním slabých modelů (rozhodovacích stromů) způsobem gradientního sestupu. Metoda dále optimalizuje využití paměti a dobu trénování pomocí technik, jako je Gradient-based One-Side Sampling (GOSS). Náš klasifikátor tvoří celkem 230 rozhodovacích stromů s maximální výškou 10 uzlů.

- **Komplexní klasifikátor na bázi XGBoost** — Zde binární klasifikátor využívá metody eXtreme Gradient Boosting (XGBoost), která využívá algoritmu gradientního boostingu. Podobně jako předchozí metoda vytváří silný klasifikátor složený z několika slabších (rozhodovacích stromů). Na rozdíl od LightGBM používá techniku náhodného dělení dat na bloky a stromovou strukturu založenou na hloubce. Náš

klasifikátor využívá celkem 150 rozhodovacích stromů o maximální hloubce 10 uzlů.

- **Klasifikátor dle DNS na bázi NN** — Jedná se o dopřednou neuronovou síť se vstupní vrstvou, která odpovídá počtu příznaků, 11 skrytými vrstvami, včetně dvou vrstev pro dávkovou normalizace, dvou vrstev typu dropout a výstupní vrstvy s jediným neuronem, který určuje pravděpodobnost, že doména je phishingová — jedná se tedy také o binární klasifikátor. S ohledem na menší počet příznaků je i celková architektura sítě jednodušší než v případě komplexního klasifikátoru.

- **Klasifikátor dle RDAP na bázi NN** — Opět jde o binární klasifikátor využívající dopředné neuronové sítě s architekturou podobnou předchozímu. I zde je využito 11 skrytých vrstev, z nichž dvě provádí dávkovou normalizaci a dvě jsou vrstvy typu *Dropout*. Rozdíl oproti klasifikátoru dle DNS je především v menším počtu neuronů v důsledku nižšího počtu příznaků.

- **Klasifikátor na bázi HTML/DOM** — Pro analýzu HTML/DOM se využívá binární klasifikátor na principu metody LightGBM. Tento klasifikátor zkoumá příkazy, které se týkají elementů HTML jako jsou například počty jednotlivých tagů či počty odkazů. Dále se analyzuje JavaScript využití v kódu HTML. Stejně tak klasifikátor zkoumá i příznaky na úrovni dokumentu jako je počet slov, řádků nebo průměrná délka slova.

Klasifikátory malwarových domén

DomainRadar implementuje následující klasifikátory malwarových domén:

- **Komplexní klasifikátor na bázi NN** — Klasifikátor využívá všechny příznaky kromě těch, které jsou extrahovány z kódu HTML. Klasifikace je binární, tj. klasifikuje do dvou tříd, konkrétně: a) benigní doména, b) doména k šíření malware. Architektura staví na bázi dopředné (feedforward) neuronové sítě. Síť obsahuje vstupní vrstvu, kde počet neuronů je ekvivalentní počtu příznaků, 12 skrytých vrstev a výstupní vrstvu s jediným neuronem, jehož výstupem je pravděpodobnost, že doména hostuje phishing. Skryté vrstvy se postupně rozšiřují až na 256 neuronů a následně zužují až do jednoho neuronu výstupní vrstvy. Mezi skrytými vrstvami jsou také dvě dávkové normalizační vrstvy a dvě vrstvy typu *Dropout*, které spolu s technikou *Early Stopping* slouží jako ochrana proti přetrénování modelu.

- **Komplexní klasifikátor na bázi LGBM** — Tento binární klasifikátor využívá metodu Light Gradient-Boosting Machine (LightGBM), obdobně jako jeden z klasifikátorů pro phishing. Náš klasifikátor tvoří celkem 220 rozhodovacích stromů s maximální výškou 7 uzlů.

- **Komplexní klasifikátor na bázi XGBoost** — Binární klasifikátor staví na metodě XGBoost, která využívá principu gradientního boostingu a vytváření množství

rozhodovacích stromů, které dohromady tvoří celý klasifikátor. Náš klasifikátor využívá celkem 150 rozhodovacích stromů o maximální hloubce 10 uzlů.

- **Klasifikátor na bázi HTML/DOM** — Pro detekci malware na základě příznaků z HTML se opět využívá binární klasifikátor na bázi metody LightGBM, který funguje na podobném principu jako klasifikátor phishingu dle HTML. I zde dochází k analýze elementů kódu HTML, kódu jazyka JavaScript a příznaků na úrovni dokumentu.

Klasifikátory algoritmicky generovaných domén

Pro detekci domén generovaných algoritmy DGA slouží následující klasifikátory:

- **Binární klasifikátor na bázi NN** — Tento klasifikátor využívá dopředné (feedforward) neuronové sítě. Počet neuronů vstupní vrstvy odpovídá počtu příznaků. Skrytých vrstev je celkem 6, přičemž dvě vrstvy jsou typu *Dropout* a slouží jako ochrana proti přetrénování klasifikátoru a pro lepší generalizaci rozhodování. Klasifikace je binární a výstupní vrstva tedy obsahuje jediný neuron, který určuje pravděpodobnost, že doména je algoritmicky generována. Klasifikujeme tedy do dvou tříd: a) doménu je algoritmicky generována, b) doména není algoritmicky generována.

- **Binární klasifikátor na bázi LGBM** — Jde také o binární klasifikátor, který klasifikuje do stejných tříd. Princip je postaven na metodě LightGBM, přičemž náš klasifikátor využívá 190 rozhodovacích stromů o maximální výšce 9 uzlů.

- **Vícetřídní klasifikátor na bázi LGBM** — Cílem tohoto klasifikátoru je pro algoritmicky vygenerovanou doménu určit pravděpodobnou rodinu algoritmu DGA. Tento výsledek slouží jako doplňující informace do klasifikačního reportu a výsledek nemá na celkovou míru rizika vliv.

Agregace výsledků a určení celkové míry hrozby

Pro každou z kategorií: a) DGA, b) phishing, c) malware dochází je dále provedena agregace výsledků. Zde se počítá jednak průměrná hodnota pravděpodobnosti hrozby z výsledků napříč klasifikátory, dále maxima a další odvozené hodnoty. Průměrné hodnoty pro jednotlivé kategorie pak slouží také jako součást výsledného reportu.

Pro určení celkové „míry rizikovosti“ dané domény se využívá kombinace rozhodovací neuronové sítě a sady heuristik:

- **Rozhodovací neuronová síť** — jedná se o dopřednou (feedforward) neuronovou síť, jejímž vstupem jsou: a) výsledky jednotlivých klasifikátorů, b) výsledky agregátorů, c) informace o chybějících hodnotách a výsledcích sběru, pomocí kterých může neuronová síť určit, jakou váhu dá výsledkům dílčích klasifikátorů s ohledem na aktuálně dostupná data. Např. nepodařilo-li se k dané doméně získat data ze serverů RDAP či WHOIS, může být výsledek klasifikace dle RDAP ignorován apod.

- **Rozhodovací sada heuristik** — několik dalších heuristik následně provede dodatečné korekce výsledků pomocí sady matematických operací jako násobení a prahování s využitím několika konfigurovatelných koeficientů, pomocí kterých lze také přizpůsobit celkovou citlivost systému na hrozby a přizpůsobit tak chování klasifikačního subsystému.

Následně je vygenerován klasifikační report, který obsahuje:

- celkovou míru rizika na škále 0 (žádné riziko) až 1 (maximální riziko),
- slovní komentář k celkové míře rizika,
- míry podobnosti s doménami generovanými pomocí DGA, výsledky dílčích klasifikátorů a slovní komentář s vysvětlením,
- míra podobnosti s doménami, které hostují phishing, výsledky dílčích klasifikátorů a slovní komentář s vysvětlením,
- míra podobnosti s doménami pro šíření malware, výsledky dílčích klasifikátorů a slovní komentář s vysvětlením,
- výsledky dílčích klasifikátorů.

Webové rozhraní a výstup modulu

Grafické uživatelské rozhraní systému DomainRadar je webová aplikace, která poskytuje pohled na výstupní informace a konfigurací dalších součástí systému. Aplikace sestává ze serveru a klientské části, které jsou těsně spjaté jako full stack řešení nad meta frameworkem Nuxt.

Server komunikuje s databázovou vrstvou PostgreSQL a MQ systémem Kafka, poskytuje API pro klientskou část i staticky generované stránky. Umožňuje také ukládání jednoduchých konfigurací pro webové rozhraní a zprostředkovává přihlašování uživatelů.

Komunikace s databází probíhá skrze ORM Prisma k získání vyžádaných výstupních dat a nastavení vlastních vstupních filtrů. Kafka slouží webovému rozhraní primárně ke čtení a zápisu konfigurací ostatních komponent systému, což je vypořádáváno bez využití komunikace s klientem v reálném čase za pomoci perzistentního abstrahujícího modulu na straně serveru. Kafka také umožňuje přímé vkládání vlastních domén ke kontrole do vstupní fronty systému na přání uživatele, a to bez průchodu vstupním filtrem.

Celkový přehled domén s interaktivní mapou

Přehled domén umožňuje řadit a filtrovat domény na výstupu systému na základě různých kritérií. Výsledné domény se zobrazují ve stránkovaném seznamu nad mapou světa, která pro každý záznam zobrazuje polohy přidružených IP adres. Náhledy domén obsahují pravděpodobnosti hrozeb odhadované systémem a jejich výběrem může uživatel otevřít detail dané domény.

Detail domény

Detailní pohled na doménu obsahuje všechna výstupní a doplňková data, která se k ní vážou. Konkrétně to jsou:

- agregace zhodnocení typů hrozeb,
- výstupy všech dílčích klasifikátorů,
- seskupené IP adresy a detaily o nich, včetně QRadar incidentů,
- časová osa domény v systému.

V detailu je také možné nahlédnout na informace o sběru dat k doméně a jednotlivým IP adresám. Navíc poskytuje odkazy na externí služby, které mohou nabízet další informace o doméně. Tyto odkazy jsou plně konfigurovatelné v rámci konkrétní instance prostřednictvím nastavení systému.

Předfiltrované domény

Tato stránka zobrazuje domény zachycené vstupními filtry zanesené přímo vstupním modulem do databáze. Domény jsou seskupené podle jednotlivých filtrů a v nastavení je možné si vytvořit pravidla pro jejich zvýraznění libovolnou barvou na základě regulárních výrazů.

Klasifikace vlastní domény

Pro kontrolu domény ručně, tj. bez jejího načtení z vnějších vstupních systémů, lze využít tento formulář. Domény je možné zadat do vstupního pole ručně a nebo z textového souboru. Zadané domény se posílají přímo na Kafka topic určený jako vstupní fronta systému pro sběr dat a následnou klasifikaci.

Domény tak v podstatě obejdou vstupní filtry a jejich kontrola je vynucena, ačkoliv ne upřednostněna. Výběr domén z fronty již není v moci webového rozhraní, a tak nelze předpovědět čas doručení výsledků vlastní kontroly do přehledu domén.

Nastavení systému

Nastavení se dělí na vícero sekcí. Předvolby pro webové rozhraní umožňují nastavit například jazyk a vzhled aplikace. Tyto volby platí pro každého klienta zvlášť, tedy jsou lokální pro daný prohlížeč.

Možnosti zvýrazňování předfiltrovaných domén a konfigurace odkazů na externí služby v detailu domény jsou nastavení platící pro daný server a tedy konkrétní instanci systému DomainRadar. Jsou ukládané do key-value úložiště na serveru a sdílené pro všechny uživatele.

Komponenty si své konfigurace spravují individuálně. Aplikace čte z MQ nejnovější konfiguraci pro každou z nich a umožňuje uživateli je upravovat a poslat požadavek na změnu. Pokud konfigurace není platná, změny se neuloží a chyba je uživateli oznámena. Platnost konfigurace rozhoduje komponenta po přijetí požadavku.

Pilotní provoz na síti CESNET

Pilotní provoz byl realizován na národní akademické síti sdružení CESNET, přičemž modul DomainRadar byl použit k detekci rizikových domén se kterými zařízení v této síti komunikovala. Pro pilotní nasazení sloužily následující virtuální servery:

- **qradar.liberouter.org** — virtuální stroj se systémem IBM QRadar, přičemž vstupem byly jednak poplachy (alerts) ze systému Suricata IDS, jednak logovací zprávy z virtuálního stroje scanner (viz níže),
  hardwarová specifikace: 4 vCPU na bázi Intel(R) Xeon(R) Gold 6226R CPU @ 2,90 GHz, 24 GB RAM, disk 250 GB,
- **ct-elk.liberouter.org** — virtuální stroj s úložištěm ELK (Elasticsearch, Logstash, Kibana), do kterého prostřednictvím protokolu Syslog systém Suricata IDS zasílal zprávy o spatřených DNS požadavcích a odpovědích, ze kterých modul DomainRadar čerpal doménová jména k jejich následné klasifikaci,
  hardwarová specifikace: 8 vCPU na bázi Intel(R) Xeon(R) Gold 6226R CPU @ 2,90 GHz, 16 GB RAM, disk 500 GB,
- **domain-radar.liberouter.org** — virtuální stroj, na kterém běžely infrastrukturní služby modulu DomainRadar — databáze PostgreSQL, server Apache Kafka a Apache Kafka Connect, komponenta Loader & Pre-filter, komponenta Data Merger a webové rozhraní pro obsluhu, konfiguraci a čtení klasifikačních reportů,
  hardwarová specifikace: 8 vCPU na bázi Intel(R) Xeon(R) Gold 6226R CPU @ 2,90 GHz, 24 GB RAM, disk 500 GB,
- **domain-radar2.liberouter.org** — virtuální stroj, na kterém v jednotlivých vláknech běžely instance klasifikační pipeline,
  hardwarová specifikace: 8 vCPU na bázi Intel(R) Xeon(R) Gold 6226R CPU @ 2,90 GHz, 24 GB RAM, disk 128 GB,
- **scanner.liberouter.org** — virtuální stroj, který sloužil ke sběru dat o doménách, který byl jako jediný z uvedených přímo dostupný z internetu. Na stroji také běžel webový server s úvodní stránkou, která vysvětlovala, že jde o zařízení ke sběru dat pro výzkumné účely,
  hardwarová specifikace: 8 vCPU na bázi Intel(R) Xeon(R) Gold 6226R CPU @ 2,90 GHz, 24 GB RAM, disk 100 GB.

V první fázi byl jako zdroj dat použit stroj meter1, který exportoval data z jedné reálné peeringové linky síťové infrastruktury CESNET3.

- **meter1** — fyzický server monitorovací sondy na lince do NIX.CZ, který sloužil jako zdroj reálných síťových dat. Tento server zajišťoval provoz detekčního systému Suricata, který exportoval metadata o DNS dotazech a zároveň pomocí nastavené sady detekčních pravidel detekoval škodlivý provoz. Data z tohoto serveru byla odesílána ve formátu JSON do systému ELK na stroji *ct-elk.liberouter.org*.
  hardwarová specifikace: 112 CPU jader (v rámci 2 NUMA uzlů) Intel(R) Xeon(R) Gold 6348 CPU @ 2.60GHz, 128 GB RAM, disk 500 GB.

Ve druhé fázi reálného nasazení a pilotního provozu byl místo přímého exportu dat z jedné sondy upravena konfigurace centrálního kolektoru síťových toků (collector-nemea), na kterém se sbírají IPFIX data z hraničních linek infrastruktury pomocí 7 měřicích bodů. Pro účely zpracování

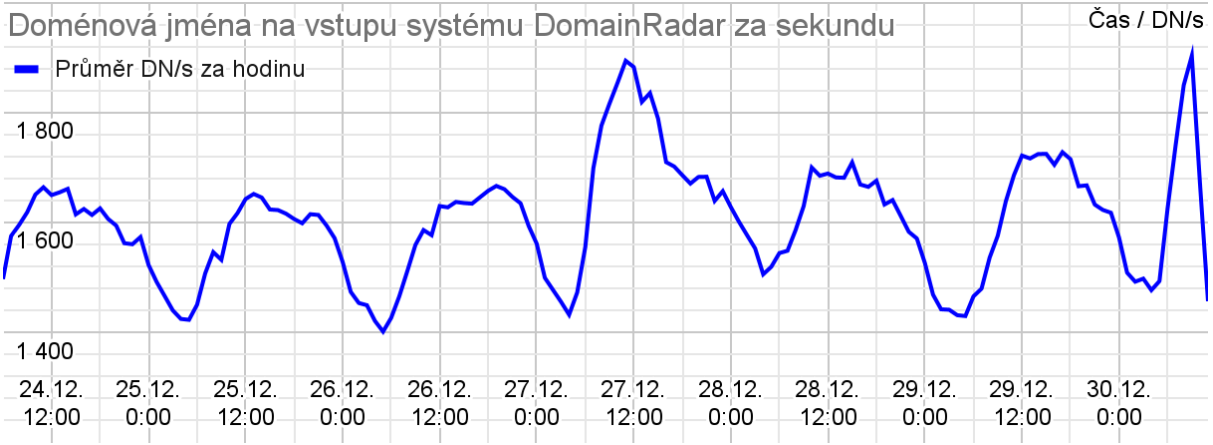
v systému DomainRadar byla informace o pozorovaných doménových jménech převedena do formátu JSON a provoz DNS známých serverů (8.8.8.8, 8.8.4.4, 1.1.1.1, 195.113.144.228, 2001:718:1:101::144:228) byl odesílán na server *ct-elk.liberouter.org*.

Ukázka JSON dokumentu odeslaného z kolektoru do ct-elk.liberouter.org:

```
{
  "DNS_RR_TTL": 0,
  "FME_DNS_RR_CLASS": 1,
  "FME_DNS_RR_TYPE": 1,
  "DNS_Q_NAME": "api.pinterest.com"
}
```

Měření výkonu

DNS záznamy typu A a AAAA byly ze systému ELK načítány komponentou Loader & Pre-filter, průměrná frekvence záznamů na vstupu byla přes **2 200 DN** (doménových jmen) **za sekundu**. Na úrovni jednotlivých dní provoz vykazoval špičky kolem 12. hodiny (zejm. během pracovních dní) a propady v noci. Globálně byla patrná pozměněná charakteristika provozu během víkendů a také větší propad během období vánočních svátků, jak naznačuje Obrázek 12.



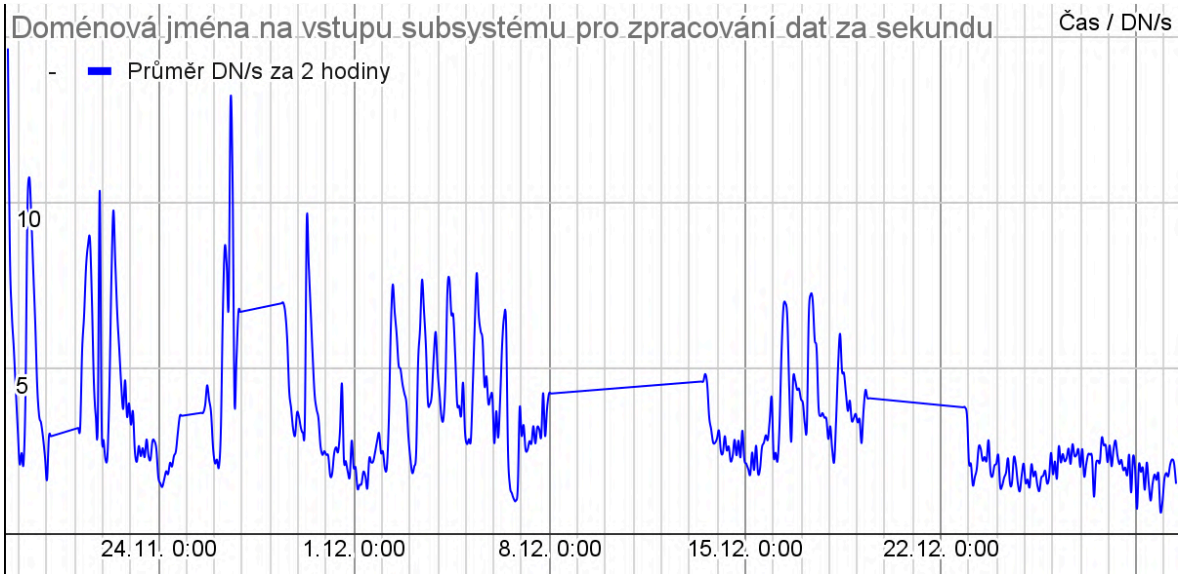
Obrázek 12: Graf zobrazující frekvenci přibývání DNS záznamů typů A a AAAA v úložišti ELK v období 24. 12. – 30. 12. 2024. Průměrná frekvence na vstupu je zde 1 604 DN/s.

Komponenta Loader & Pre-filter aplikovala na vstup následující filtry:

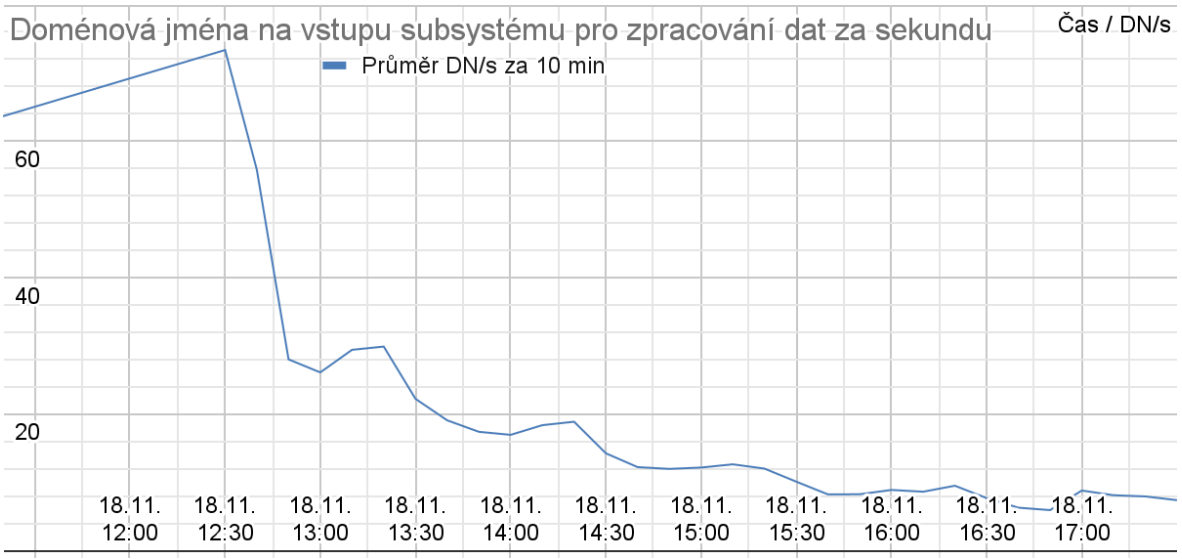
- ValidDomainFilter filtrující nevalidní doménová jména,
- FileBlockListFilter filtrující 100 nejčastěji navštěvovaných domén[9] **a všechny jejich subdomény**,
- FileBlockListFilter filtrující všechny subdomény in-addr.arpa a ip6.arpa,
- RandomDROPFilter, který náhodně zahazoval domény s pravděpodobností 0,02.

[9] Viz `DomainRadar/input/top100.blocklist`. Zdroj: Cloudflare Radar – seznam Top 100 domains Worldwide z 22. 6. 2024 – https://radar.cloudflare.com/domains.

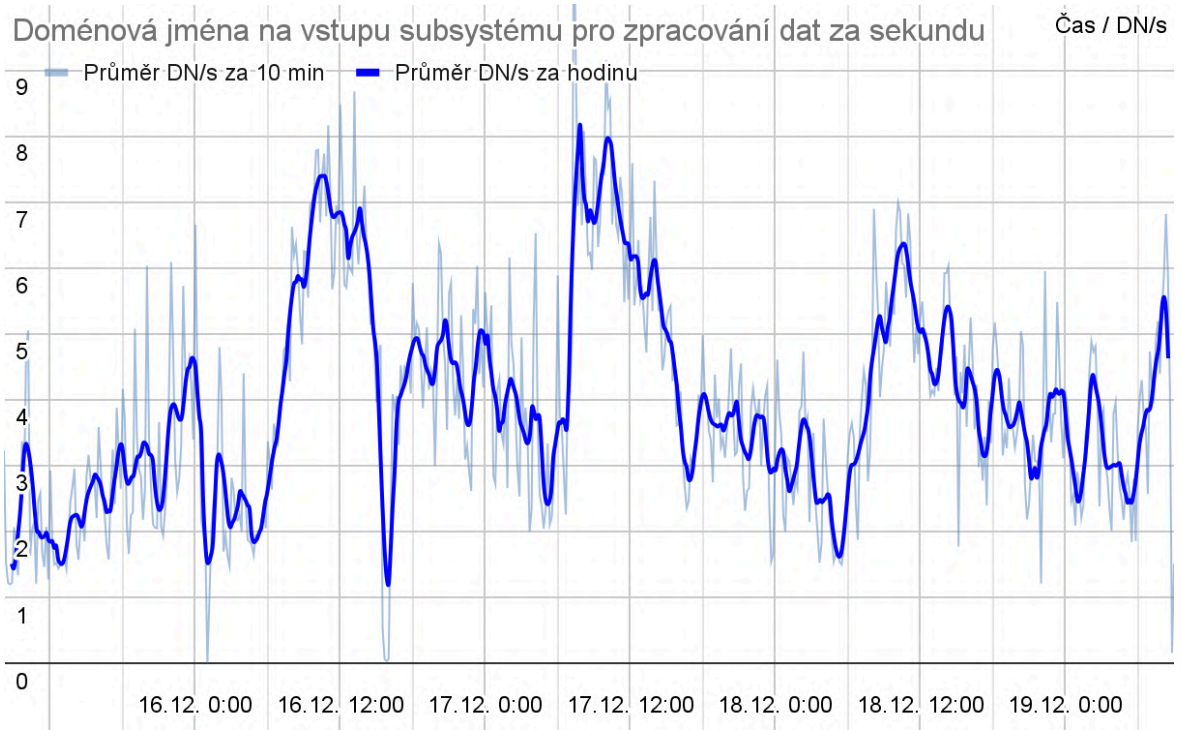
Obrázek 13 ukazuje průměrnou frekvenci přírůstku nově viděných doménových jmen, tj. frekvenci vstupu subsystému pro sběr a zpracování dat. Na začátku sledovaného období (18. 11. 2024 v 11:20) byla databáze viděných jmen vyprázdněna, dále do ní položky pouze přibývaly. V několika prvních hodinách provozu, které jsou zobrazeny na obrázku 14, bylo filtrem propuštěno i přes 70 DN/s, která byla systémem bez problémů zpracována. Systém se v této síti postupně ustálil na zpracovávání cca 7–11 DN/s ve špičkách provozu a celkově asi **400 000–500 000** zpracovaných DN za den. Z grafu na obrázku 13 lze nicméně rozpoznat tendenci k postupnému snižování frekvence v čase, patrné je také snížení datového toku v období vánočních svátků, které koresponduje se snížením celkového objemu vstupu před filtrací.



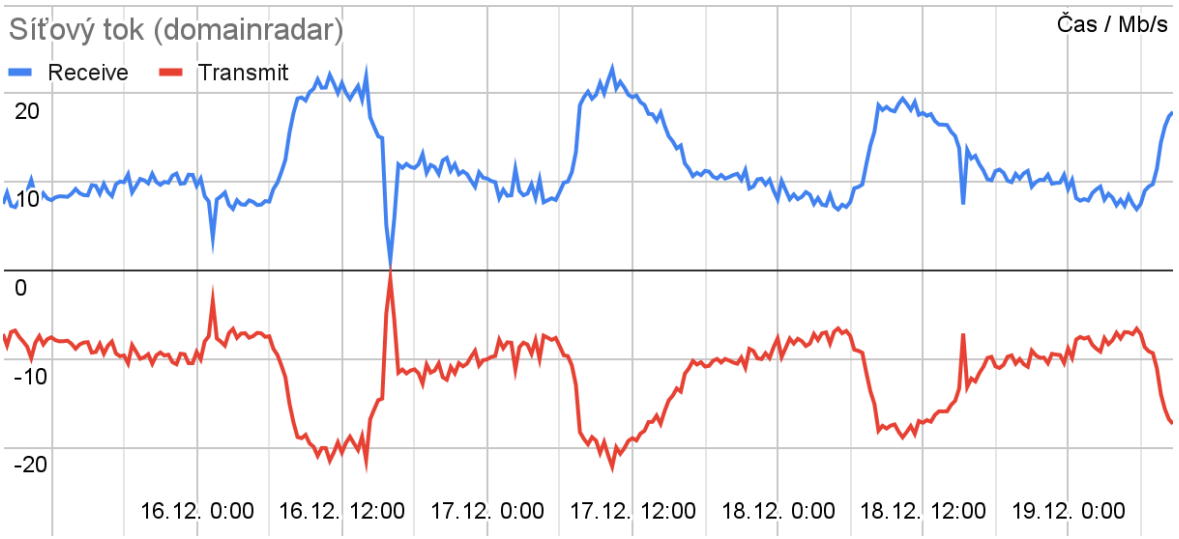
Obrázek 13: Graf zobrazující frekvenci nově viděných doménových jmen propuštěných vstupní komponentou ke zpracování v pipeline v období pondělí 18. 11. (11:20) – pondělí 30. 12. 2024 (12:00). Hluchá místa (rovné čáry) byla zapříčiněna vypnutím vstupní komponenty z vývojových důvodů.



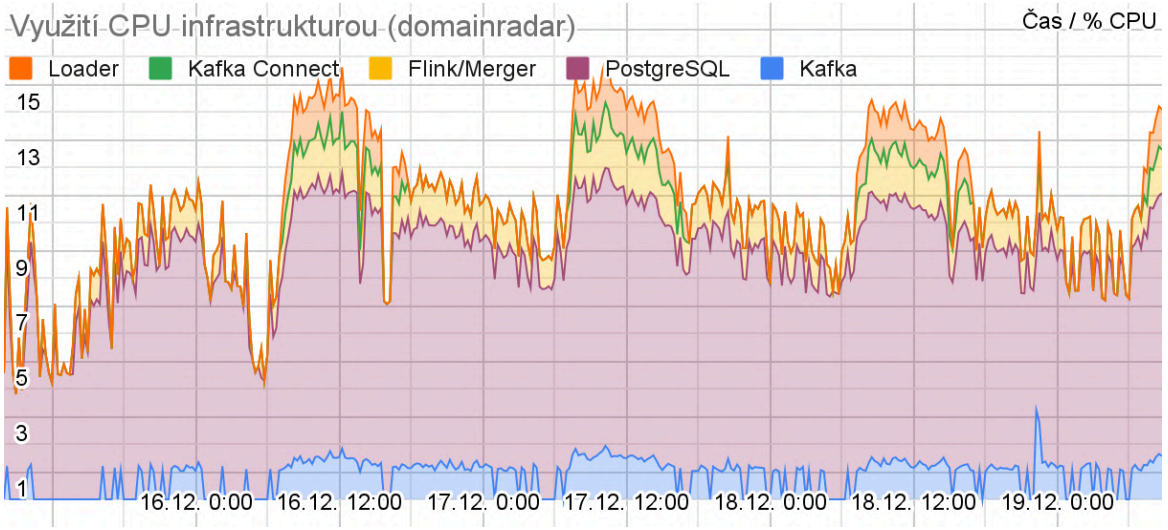
Obrázek 14: Detail grafu z obrázku 13 v prvních šesti hodinách provozu.

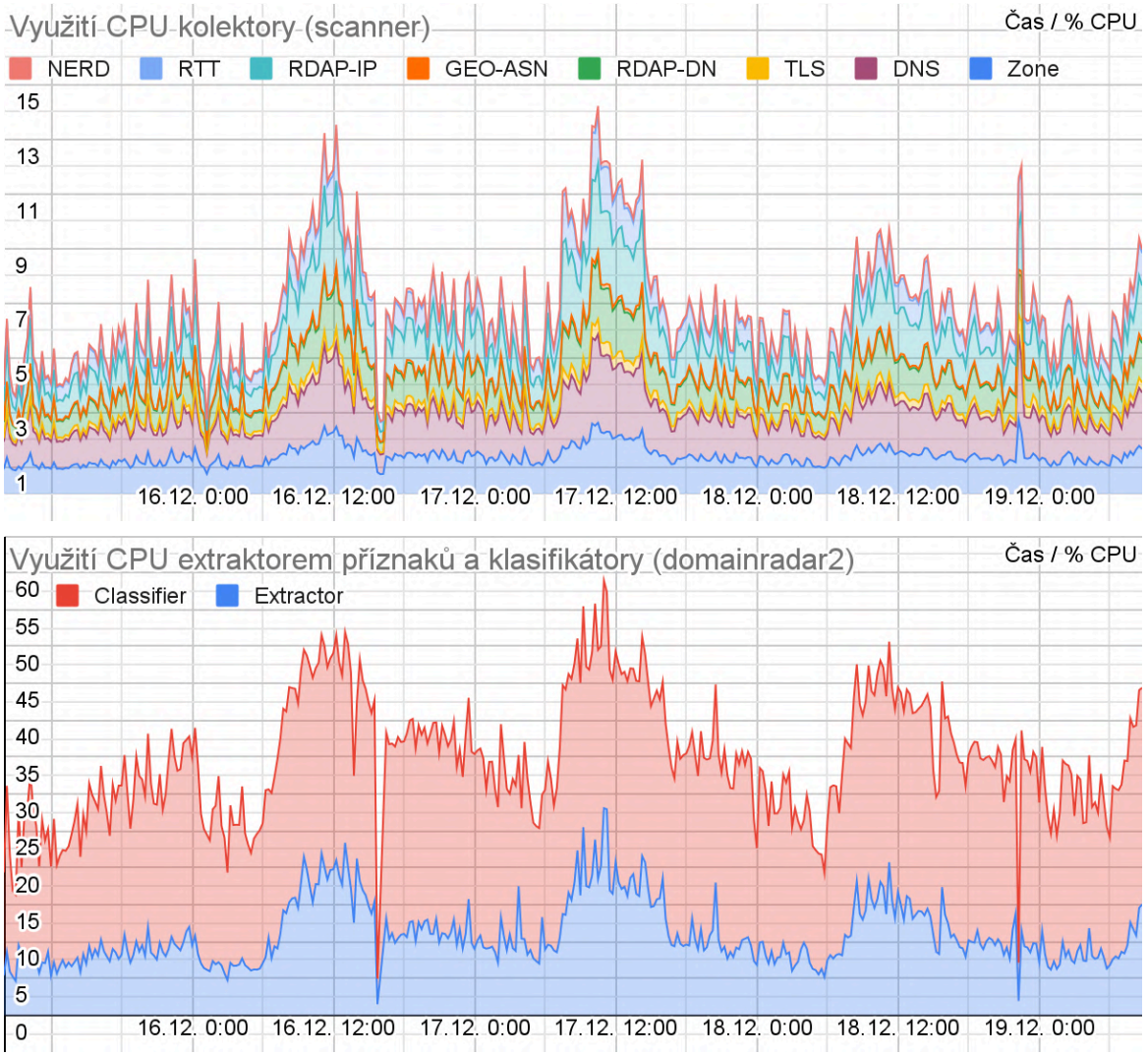


Obrázek 15: Detail grafu z obrázku 13 v období neděle 15. 12. (08:00) – čtvrtek 19. 12. 2024 (09:00).

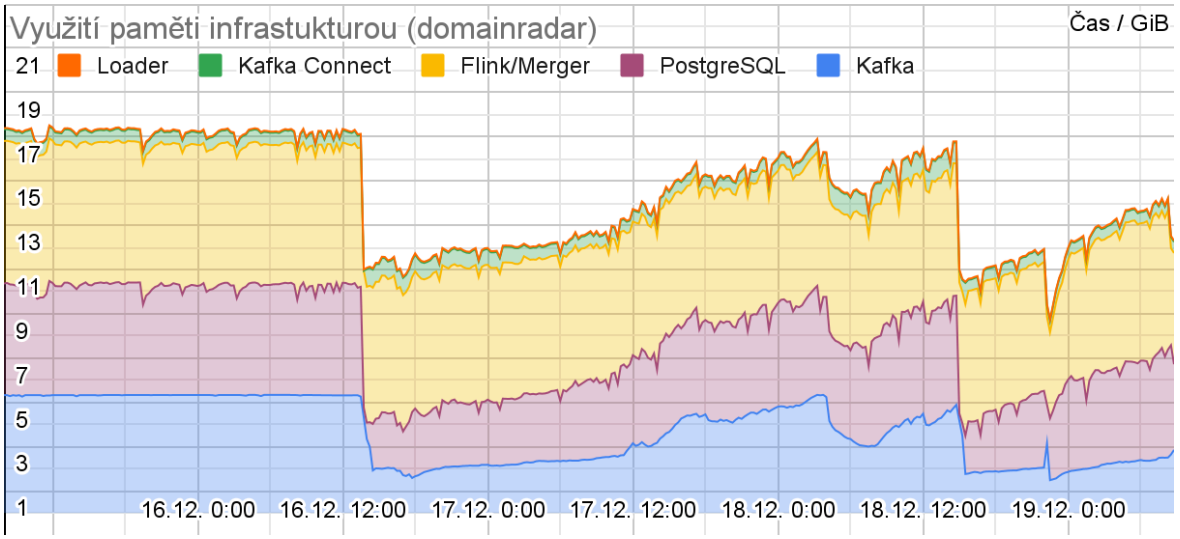


Obrázek 16: Objem dat přenášený na VM *domainradar* ve sledovaném období.





Obrázek 17: Spotřeba CPU služeb běžících na jednotlivých VM ve sledovaném období.



Obrázek 18: Využití paměti na serveru *domainradar* ve sledovaném období.

Grafy na obrázcích 15–18 nabízí detailnější pohled na charakter využití systémových prostředků službami modulu DomainRadar během několika běžných dní provozu. V tomto období bylo na vstupu modulu v průměru 2 080 doménových jmen za sekundu, po filtraci na vstupu subsystému pro zpracování dat pak v průměru 3,96 ± 1,71 DN/s, ve špičkách však dosahoval propustnosti až 11,58 DN/s. Špičky provozu (vždy okolo 12. hodiny) se očekávatelně odrážejí zejména na celkovém objemu přenášených dat (Obr. 16) a spotřebě CPU (Obr. 17). Z grafů je zřejmé, že v kontextu sledovaného datového toku byly stroje *domainradar* a *scanner* značně naddimenzované. Výpočetně náročná je zejména extrakce příznaků a klasifikace, které ve špičce dohromady dosahovaly cca 60% využití CPU – zde se objem prostředků jeví přiměřený. V infrastrukturní VM lze pozorovat zdánlivě vyšší využití paměti (Obr. 18), to je však způsobeno alokační strategií běžících služeb, které si mohou předalokovat paměť až do výše nakonfigurovaných limitů. V této konfiguraci nebyly na službách infrastruktury sledovány žádné problémy s propustností.

Příklady nalezených škodlivých domén

V rámci pilotního provozu modulu na síti CESNET modul DomainRadar odhalil množství škodlivých doménových jmen. Část z nich byla podrobena manuální kontrole, aby došlo k ověření jejich závadnosti. Ověření proběhlo zejména:

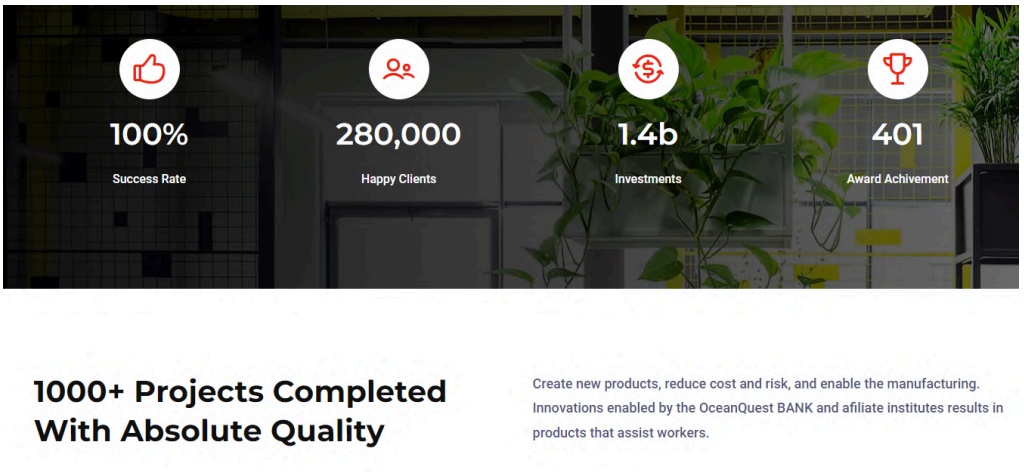
1. návštěvou dané stránky přes prohlížeč izolované pracovní stanice,
2. sledováním hlášení aplikace BitDefender[10],
3. sledováním upozornění služby Google SafeBrowsing[11],
4. využitím služby URL Scan provozovatele VirusTotal[12], a také
5. využitím webových rozšíření jako Avast Online security, Traffic Light a dalších,
6. manuálním hledáním informací o dané stránce (černé listiny aj.).

Tato sekce popisuje vybrané hrozby, které odhalil modul DomainRadar a které byly následně touto manuální kontrolou potvrzeny.

Doména oceanquestb.com s podvodným webem

Stránka *oceanquestb.com* představuje podvodný web falešné banky. Snímek obrazovky ukazuje Obrázek 19. Na první pohled se jedná o uživatelsky přívětivou prezentaci bankovní instituce. Provozovatelé zmiňují velký počet spokojených klientů, řadu uzavřených investic a množství získaných ocenění. Již při prvotním dohledání vývojáři modulu DomainRadar zjistili, že tato čísla web generuje zcela náhodně pro každého návštěvníka. Různí návštěvníci tedy vidí různé počty spokojených klientů apod. Dalším podezřelým znakem je skutečnost, že množství hypertextových odkazů je nefunkčních. K závěru, že se skutečně jedná o podvodný web vývojáři došli nalezením tohoto webu na seznamu falešných bank AA419[13]. Mimo to, 15 z 96 dostupných autorit služby VirusTotal web hodnotilo jako závadný.

[10] Viz Viz https://www.bitdefender.com/.
[11] Viz https://safebrowsing.google.com/.
[12] Viz https://www.virustotal.com/gui/home/url.
[13] Viz https://db.aa419.org/fakebankslist.php.



Obrázek 19: Snímek falešné bankovní instituce oceanquestb.com

Doména greenfastline.com pro šíření malware

Po návštěvě webu *greenfastline.com*, na který upozornil modul DomainRadar, došlo k okamžitému zahájení stahování malware, konkrétně **Trojan: Kimsuky.Gen/VBS!8.13D95**. Jedná se o škodlivý kód vytvořený skupinou Kimsuky, která působobí na území Korejské lidově demokratické republiky a která je americkou Agenturou pro kybernetickou a infrastrukturní bezpečnost (CISA) klasifikována[14] jako Advanced Persistent Threat (APT). Navíc bylo zjištěno, že pod doménou existuje také množství subdomén třetí úrovně jako *reset.greenfastline.com*, či *step.greenfastline.com*, které vykazují podobné, závadné chování. Jak ukazuje Obrázek 20, závadnost domény i jejích subdomén potvrdila také služba VirusTotal. Jedná se tedy zcela nepochybně o doménu určenou k šíření malware, navíc v režii skupiny, která je kategorizována jako APT.

Subdomains (11)

reset.greenfastline.com	10 / 93	80.66.79.251		
reste.greenfastline.com	9 / 93	80.66.79.251		
2frest.greenfastline.com	9 / 93	80.66.79.251		
step.greenfastline.com	9 / 93	80.66.79.251		
dominiorest.greenfastline.com	7 / 93	80.66.79.251		
goto.greenfastline.com	9 / 93	80.66.79.251		
away.greenfastline.com	4 / 93	45.140.146.101		
www.greenfastline.com	0 / 93	104.21.83.35	172.67.211.36	
fine.greenfastline.com	9 / 93	80.66.79.251	80.66.79.248	104.21.83.35
greenfastline.com	11 / 93	80.66.79.251	80.66.79.248	104.21.83.35

Obrázek 20: Výsledky hodnocení domény greenfastline.com službou VirusTotal

[14] Viz https://www.cisa.gov/news-events/cybersecurity-advisories/aa20-301a.

Algoritmicky generované domény *.hearing-aid-101.xyz

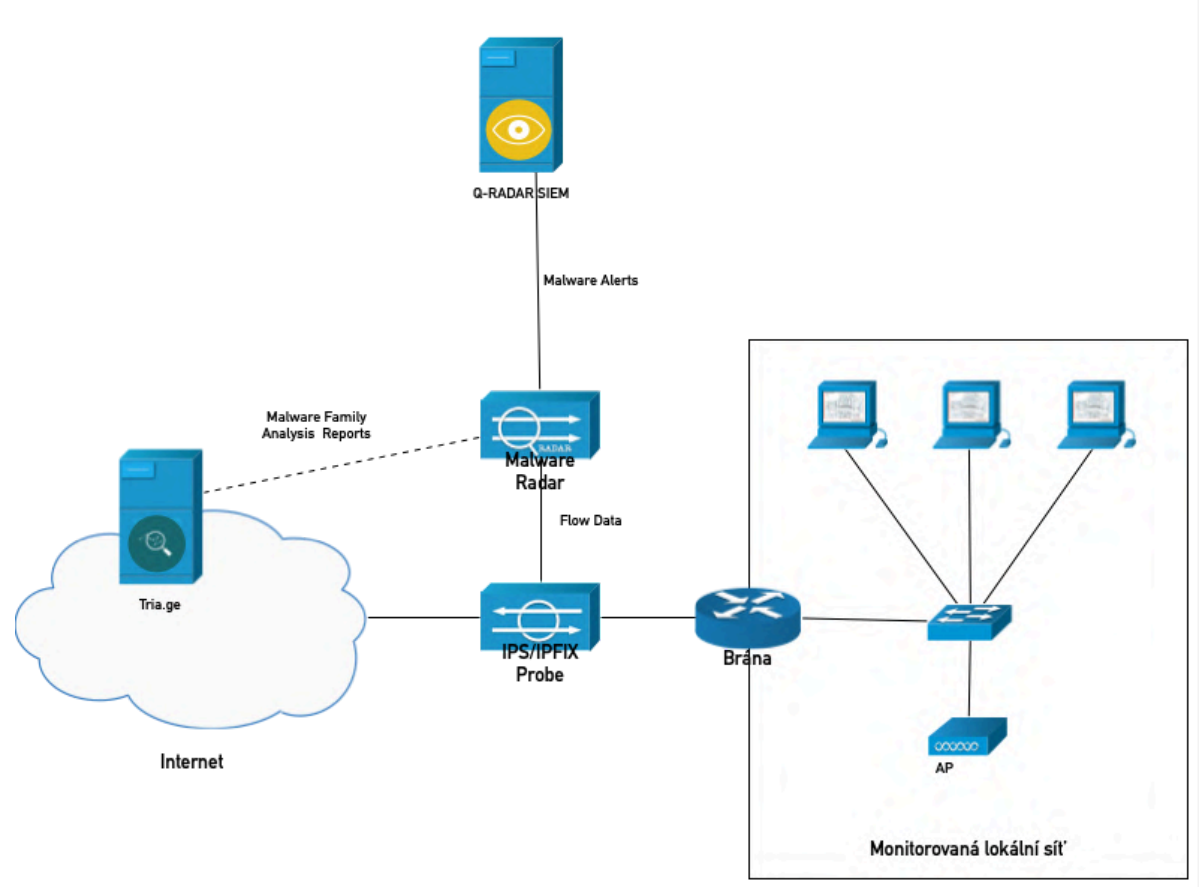
Modul DomainRadar jako závadnou označil také řadu doménových jmen spadajících pod doménu druhé úrovně *hearing-aid-101.xyz*. Jednalo se například o domény:

- *sberbank.pay.www.sberbank.sber.pay.pay.hearing-aid-101.xyz*,
- *sberbank.avito.www.avito.yandex.sberbank.pay.pay.hearing-aid-101.xyz*,
- *sberbank.avito.sber.avito.sberbank.www.pay.pay.hearing-aid-101.xyz* a
- *www.pay.www.yandex.www.pay.pay.pay.pay.www.hearing-aid-101.xyz*.

Nutno podotknout, že podobných subdomén bylo několik desítek. Zcela očividně se jedná o generované řetězce různých slov, která souvisejí zejména s bankovnictvím, např. bankou Sberbank, mobilní platební aplikací Avito, ale také s vyhledávačem Yandex. Všechny tyto produkty pojí skutečnost, že mají sídlo na území Ruské federace. Jejich závadnost byla následně potvrzena mimo jiné nástrojem BitDefender, který zablokoval podezřelá spojení.

MalwareRadar

MalwareRadar je nástroj (modul) pro analýzu síťového provozu a detekci spojení, které mohly být vytvořeny malware. Tento modul analyzuje metadata síťových spojení z lokální monitorované sítě (viz obrázek 21) a na základě naučených modelů určuje pravděpodobnost, že dané síťové spojení představuje malware komunikaci. Výsledky analýzy jsou reportovány do SIEM szstemu, například QRadar vybraným způsobem (syslog, JSON události, atd.) Nástroj je modulární a umožňuje zapojení různých detekčních metod. Nástroj je spustitelný v prostředí Docker, čímž se zjednodušuje jeho instalace, nasazení a provoz.



Obrázek 21: Nasazení nástroje MalwareRadar

Principy detekce

Implementované metody detekce zahrnují identifikaci malware pomocí naučených indikátorů kompromitace (IoC), použití metod porovnávání otisků komunikace (fingerinting) a použití klasifikátorů-detektorů založených na modelech strojového učení.

Malware IoC

Indikátory kompromitace (IoC) jsou klíčovými prvky pro identifikaci a analýzu malwaru. Jedná se o artefakty, které signalizují přítomnost nebo aktivitu malwaru v napadeném systému. IoC se běžně získávají z analýzy vzorků malware, například prostřednictvím prostředí Tria.ge. Tyto indikátory typicky zahrnují:

- URL adresy používané malwarem k přístupu k serverům nebo ke stažení dalších komponent.
- Doménová jména, která slouží k přesměrování komunikace nebo hostování škodlivého obsahu.
- IP adresy, které zajišťují konektivitu mezi malwarem a jeho kontrolním serverem.

Proces získávání a zpracování indikátorů kompromitace (IoC) zahrnuje několik kroků, počínaje analýzou vzorků malwaru v sandboxových prostředích, jako je Tria.ge. Z každého vzorku se extrahují relevantní IoC, například URL adresy, domény a IP adresy, které malware využívá pro komunikaci s kontrolními servery nebo distribuci škodlivého obsahu. Tyto indikátory jsou následně seskupeny podle kategorií a porovnány napříč více vzorky téže malware rodiny, aby bylo možné identifikovat společné i unikátní hodnoty. Pomocí frekvenční analýzy se určují míry výskytu jednotlivých IoC, což umožňuje vytvořit fuzzy množiny, které reprezentují pravděpodobnost, že daný indikátor patří do určité malware rodiny. Následně se pro každou kategorii vypočítává skóre na základě hodnot příslušnosti v fuzzy množinách, přičemž prahové hodnoty se stanovují jako střední hodnota a směrodatná odchylka skóre analyzovaných vzorků. Tyto prahové hodnoty slouží k normalizaci a interpretaci detekčních skóre, aby bylo možné rozhodnout, zda je kontext analyzovaný modelem dostatečně podobný známým vzorkům malwaru. Tento proces zajišťuje, že výsledný malware model je schopen přesně detekovat nové vzorky malwaru, a současně minimalizuje riziko falešných poplachů tím, že diferencuje mezi běžně používanými IoC (např. veřejné DNS servery) a indikátory charakteristickými pro škodlivé aktivity.

Následuje příklad analýzy malware rodiny Amadey. Při analýze 10 vzorků malware z rodiny Amadey byly identifikovány IoC a uspořádány do tří kategorií: URL, domény a IP adresy. Pro každý vzorek byly získány příslušné množiny, např.:

Vzorek 1:

- IoC-URL: {https://j.ffbbjjkk.com/2701.html, https://j.ffbbjjkk.com/logo.png, http://77.73.134.27/8bmdh3Slb2/index.php?scr=1}
- IoC-DOM: {j.ffbbjjkk.com, y1.ffbbyykk.com, rpraqmrigh.com}
- IoC-IPS: {8.8.8.8, 104.21.8.227, 77.73.134.27}

Vzorek 2:

- IoC-URL: {http://nestlehosts.xyz/new/837, http://nestlehosts.xyz/so57Nst/index.php, http://77.73.134.27/8bmdh3Slb2/index.php?scr=1}
- IoC-DOM: {nestlehosts.xyz, nestlecareers.cf, nestleservers.xyz}
- IoC-IPS: {8.8.8.8, 178.62.77.44, 77.73.134.27}

Pro získané vzorky se provede frekvenční analýza, tak aby se zohlednila četnost jednotlivých indikátorů mezi vzorky. Například:

URL:

- https://j.ffbbjjkk.com/2701.html – četnost 0,3
- http://77.73.134.27/8bmdh3Slb2/index.php?scr=1 – četnost 0,3

IP adresy:

- 8.8.8.8 – četnost 1 (ve všech vzorcích)
- 77.73.134.27 – četnost 0,3

IoC s vysokou četností, jako je 8.8.8.8, mohou být společné pro více rodin malware, což ukazuje na jejich význam při detekci. Unikátní IoC naproti tomu signalizují specifické vzorce chování. Pro detekci malwaru se vypočítává skóre na základě hodnot příslušnosti indikátorů v fuzzy množinách. Skóre umožňuje stanovit prahové hodnoty pro identifikaci malwaru, například:

URL: [3.19, 0.4, 5.69, ...]

Domény: [6.3, 11.19, ...]

IP adresy: [5.39, 2.7, ...]

Prahová hodnota se stanoví jako střední hodnota rozložení skóre ± směrodatná odchylka. Citlivější detekci lze dosáhnout nižší prahovou hodnotou, avšak za cenu vyšší pravděpodobnosti falešných poplachů.

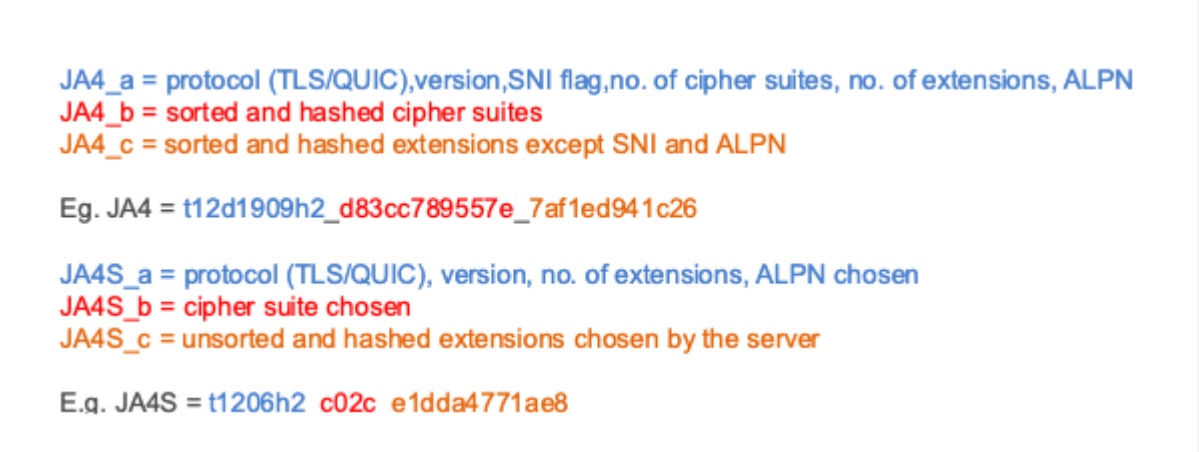
Malware Fingerprinting

Detekce komunikace škodlivého softwaru pomocí klasických technik detekce vzorů v komunikaci je omezená kvůli stále častějšímu použití šifrování. Komunikace malware používá TLS spojení, což ztěžuje rozlišení mezi škodlivým a neškodným přenosem. Jednou z možností, jak odhalit komunikaci malwaru, je analyzovat handshake TLS a získat otisky JA4+. JA4+ je kolekce otisků TLS vyvinutá Johnem Althousem a dalšími v roce 2023, která má nahradit otisky JA3. Otisk TLS zahrnuje extrakci specifických atributů z handshake TLS a jejich hashování za účelem identifikace aplikací nebo malware v šifrovaném provozu.

TLS fingerprinting identifikuje aplikace analýzou bezpečnostních parametrů, které si každý klient a server vyjednají během TLS handshake. Tyto parametry mohou vytvořit jedinečné otisky pro konkrétní aplikace a verze. Porovnáním pozorovaných přenosů TLS s databází známých otisků, včetně otisků běžných aplikací a malwaru, můžeme identifikovat aplikace v šifrovaném provozu. Protože však některé aplikace sdílejí otisky prstů, jsou pro přesnější identifikaci zapotřebí další atributy, jako je například SNI (Server Name Indication).

Otisky JA4 a JA4S se skládají ze tří částí (viz Obrázek 22). Otisk JA4 obsahuje následující atributy TLS: typ protokolu (TLS nebo QUIC), verzi protokolu TLS handshake, seřazený seznam sad šifer nabízených klientem, seřazený seznam rozšíření, příznak SNI (Server Name Indicator), ALPN (Application Layer Protocol Negotiation), podporované verze a podpisové algoritmy. Některé z těchto hodnot jsou vloženy přímo do otisků JA4, jiné jsou setříděny nebo zaheslovány. Otisk JA4S představuje konfiguraci TLS na straně serveru a obsahuje atributy TLS podobné otisku JA4, s výjimkou algoritmů SNI a podpisu.

Otisky JA4X se používají k detekci škodlivého softwaru analýzou vybraných atributů certifikátů X.509 odeslaných serverem TLS během handshake. Na rozdíl od JA4 a JA4S se otisky JA4X zaměřují spíše na způsob generování certifikátu než na jeho konkrétní hodnoty. Autoři škodlivého softwaru často používají k vytváření podvržených certifikátů stejný nástroj, takže tyto certifikáty sdílejí společný otisk JA4X, který odráží generátor bez ohledu na změny v názvech vydavatele nebo subjektu. Otisky JA4X zahrnují sledování formátu tří atributů certifikátu: jména vydavatele, jména subjektu a seznamu rozšíření certifikátu. Jméno vydavatele nebo subjektu je formálně posloupnost objektů X.500 nazývaná Relative Distinguished Name (RDN), např. common name = „GlobalSign Organization“, organization = „GlobalSign“, country = „BE“. Otisk JA4X přebírá posloupnost prvků RDN, tj. společný název (cn), organizace (o), země (c), bez hodnot. Prvky (jejich OID) jsou spojeny a zaheslovány pomocí SHA256. Výsledek je součástí otisku JA4X. Otisk JA4X se také skládá ze tří částí, jak je znázorněno na obrázku 23. Všimněte si, že server TLS obvykle odesílá nejen svůj vlastní certifikát, ale také certifikáty nadřazených certifikačních autorit, což vede k tomu, že na jedno spojení TLS připadá více otisků prstů JA4X. V TLS 1.3 a vyšších verzích jsou však certifikáty šifrovány, takže otisky JA4X nejsou k dispozici.



Obrázek 22: Formát otisku JA4 a JA4S

Obrázek 23: Formát otisku JA4 a JA4S

Pro účely analýzy přesnosti použití techniky otisků pro určení malware komunikace jsme provedli několik experimentů. Sourčástí experimentů bylo také vytvoření datové sady s malware komunikací a datové sady s komunikací běžných aplikací. Na těchto datových sadách bylo vyhodnoceno jak se otisky jednotlivých kategorií komunikace (běžné aplikace, mobilní aplikace, a malware) liší z pohledu různých typů otisků.

Soubor dat pro experimenty byl vytvořen pomocí sandboxu pro analýzu malwaru pro desktopový malware a emulátoru virtuálního zařízení Android (AVD) pro mobilní malware. Tyto nástroje umí zachytit síťovou komunikaci malwaru. Z takto zachycené komunikace byly extrahovány otisky pro jednotlivé TSL komunikace.

Id	Desktop Malware	Id	Mobile Malware
0	agenttesla	33	3d-skin-editor-for-minecraft
1	asyncrat	34	3d-skin-editor-for-minecraft
2	azorult	35	alaa-win-play
3	bazarbackdoor	36	auto-click-repeater
4	darkcomet	37	bank-bingo
5	dridex	38	bitcoin-master
6	emotet	39	cashzine
7	formbook	40	colo-chess
8	gozi-ifsb	41	crazy-magic-ball
9	hawkeye	42	daily-step
10	hawkeye-reborn	43	get-rich-scanner
11	icedid	44	happy-2048
12	lokibot	45	hexa-pop-link-2048
13	masslogger	46	jelly-connect
14	matiex	47	letter-link
15	metasploit	48	macaron-boom
16	modiloader	49	macaron-match
17	nanocore	50	mega-coin-dozer
18	netwire	51	mega-wins-slot
19	njrat	52	picpro
20	pony	53	play-box
21	qakbot	54	royal-dice-party
22	qnodeservice	55	smart-walk
23	raccoon	56	star-quiz
24	remcos	57	tiler-master
25	revengerat	58	track-your-sleep
26	smokeloader	59	treasure-scanner
27	sodinokibi	60	vfly
28	trickbot	61	vibetik
29	upatre		
30	wannacry		
31	yunsip		
32	zloader		

a) Analyzované malware rodiny

Id	Application	Id	Application	Id	Application
0	amazon-music	21	tor	42	netflix
1	amplibraryagent	22	trello	43	packeta
2	appletv	23	whatsapp	44	reddit
3	chrome	24	zoom	45	regiojet
4	firefox	25	accuweather	46	seznam
5	fournet	26	alza	47	shazam
6	gamebar	27	cestovne-poriadky	48	signal
7	hp	28	discord	49	snapchat
8	itunes	29	disneyplus	50	spotify
9	maps	30	duolingo	51	tiktok
10	messenger	31	facebook	52	tmobile
11	ms-teams	32	foodora	53	tor
12	msedge	33	gmail	54	twitter
13	msmpeng	34	idos	55	uc-browser
14	omencmdcenter	35	instagram	56	viber
15	primevideo	36	linkedin	57	whatsup
16	runtimebroker	37	mapycz	58	wolt
17	sky-go	38	messenger	59	youtube
18	skype	39	moje-vut		
19	spotify	40	mujvlak		
20	telegram	41	navlak		

b) Analzyované mobilní a desktopové aplikace

Tabulka 24: Vzorky komunikace použité v experimentech

Soubor dat o komunikaci malwaru byl vytvořen pomocí sandboxu pro analýzu malwaru Tria.ge (popsáno detailně v následující kapitole). Tento nástroj analyzuje vzorky malwaru a poskytuje výsledky z dříve nahraných vzorků, přičemž každý pozitivní vzorek kategorizuje podle rodiny malwaru. Použili jsme veřejné rozhraní API Tria.ge ke shromáždění vzorků z 33 různých rodin malwaru, přičemž pro každou rodinu bylo požadováno 50 vzorků. Každá zpráva o analýze vzorku ve formátu JSON obsahovala indikátory kompromitace, jako jsou názvy domén, IP adresy a adresy URL. Byly shromážděny komunikační stopy po spuštění malwaru, ale tyto stopy zahrnovaly veškerou komunikaci hostitele. Aby bylo možné izolovat komunikaci malwaru, byly stopy filtrovány pomocí nahlášených IP adres. Konečný soubor dat, uspořádaný podle rodiny malwaru, obsahuje komunikační stopy v souborech PCAP s anotací názvů rodiny malwaru jako označení síťových spojení.

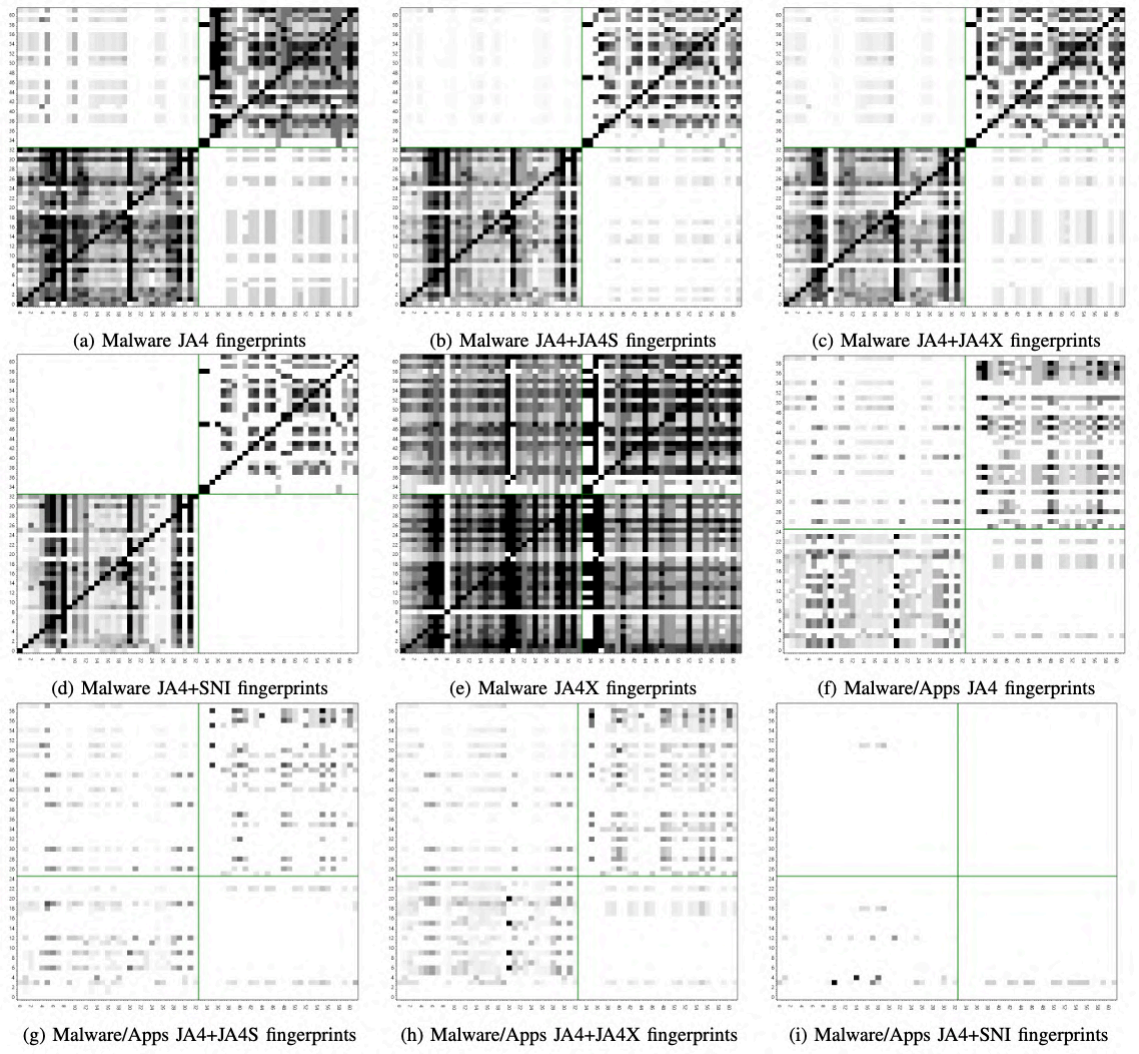
Při analýze mobilního malwaru jsme se zaměřili na nebezpečné a infikované aplikace nahlášené zdroji, jako jsou McAfee a Doctor Web. Získali jsme soubory APK s infikovanými aplikacemi, nahráli je do virtuálního zařízení Android (AVD) pro testování a zachytili síťovou komunikaci v souborech PCAP. Extrahovali jsme komunikaci TLS a odfiltrovali nesouvisející relace TLS na základě SNI. Zbývající spojení TLS byla označena názvem malwaru. Celkem jsme analyzovali 31 různých mobilních malwarových aplikací, jak ukazuje Obrázek 23, pravý sloupec.

Pro srovnání jsme použili anotovanou sadu dat běžných desktopových a mobilních aplikací. Datová sada obsahovala 25 desktopových a 35 mobilních aplikací, viz Obrázek 24. Pro další analýzu jsme extrahovali spojení TLS, vypočítali otisky JA4+ a přidali anotace.

Pro každou rodinu malwaru jsme shromáždili otisky (tzv. fingerprints) a identifikovali jejich případné překrývání, přičemž jsme analyzovali otisky JA4 a jejich kombinace, například JA4+JA4S, JA4+JA4X a JA4+SNI. Obrázky 25(a-e) ukazují vztahy mezi rodinami malwaru na základě sdílených otisků, přičemž sloupce a řádky představují různé rodiny a tmavší barvy označují vyšší poměr sdílených otisků. Levý dolní kvadrant znázorňuje vztahy mezi malwarem pro stolní počítače, pravý horní kvadrant znázorňuje mobilní malware a ostatní kvadranty znázorňují vztahy mezi platformami. Údaje ukazují významné překrývání otisků prstů mezi rodinami malwaru na platformě stolních počítačů i mobilních zařízení, ale minimální překrývání mezi těmito dvěma platformami. Samotné otisky JA4 mohou jednoznačně identifikovat pouze několik rodin (viz Obr. 25a). Přidání JA4S (Obr. 25b) nebo JA4X výsledky mírně zlepšuje. Nejlepší zlepšení přinese přidání SNI (Obr. 25d), ale mnoho otisků je stále společných pro různé rodiny malwaru. Obr. 25e) znázorňuje otisk JA4X, který je určen k detekci malwaru pomocí informací z certifikátů. Obrázek ukazuje, že mnoho rodin malwaru sdílí stejné otisky JA4X, což potvrzuje hypotézu autorů. Ukazuje však také, že bez dalších informací je obtížné rozlišit různé rodiny malwaru pouze pomocí této metody.

Zkoumali jsme také použití otisků k rozlišení škodlivé a neškodné komunikace aplikací. Vypočítali jsme otisky prstů pro desktopové a mobilní aplikace a analyzovali překrývání s otisky prstů malwaru. Výsledky jsou uvedeny na Obrázcích 25 (f-i), kde sloupce představují 62 rodin malwaru (0-61), viz tabulka 24a, a řádky představují 25 desktopových a 35 mobilních aplikací (0-59), viz tabulka 24b. V analýze otisků JA4 (Obrázek 25f) má mnoho rodin malwaru pro stolní počítače společné otisky s aplikacemi pro stolní počítače a mobilní malware sdílí otisky s mobilními aplikacemi, přičemž překryv mezi platformami je omezený. Kombinace otisků klientů a serverů (Obrázek 25g) nebo použití JA4 s hashi JA4X (obrázek 25h) zlepšuje výsledky. Přidání SNI k otisku JA4 výrazně snižuje počet kolizí (obrázek 25i).

Metoda detekce malware založená na JA4 otiscích je vhodná, pokud máme k dispozici spolehlivou datovou sadu vzorků malwaru. Výslednou databázi otisků JA4+ malwaru mohou využívat zařízení pro monitorování sítě k detekci komunikace malwaru v reálném čase. Na základě našich experimentů jsme také prokázali, že otisky JA4X mají omezené využití, protože se vyskytují v méně než třetině spojení TLS a stejné otisky JA4X jsou sdíleny mezi malwarem a neškodnými aplikacemi. Experimenty také ukázaly, že stejné otisky JA4 jsou sdíleny mezi různými rodinami malwaru, takže není snadné rozlišit rodiny malwaru pouze na základě otisků JA4. Další zkoumání ukázalo, že použití kombinovaných otisků prstů by mohlo zlepšit schopnost rozlišovat mezi šifrovaným malwarem a komunikací aplikací. Zahrnutí JA4S a SNI do otisků zlepšuje detekci, ale zcela neeliminuje sdílení otisků mezi různými rodinami malwaru. Další zkoumání ukázalo, že použití kombinovaných otisků prstů může poskytnout schopnost rozlišit mezi šifrovaným malwarem a komunikací aplikací.



Obrázek 25: Kolize otisků

Malware ML model

Krom použití otisků je možné identifikovat malware komunikaci pomocí charakteristický komunikačních vzorů. Zde použitá detekce je založena na vytvoření ML modelu pro TLS komunikaci spočívá ve sledování velikosti síťových paketů a segmentů. Toto je často používaný přístup pro tvorbu ML modelů provozu[15]. Relevantní práce navíc používají Interarrival time (mezi paketový čas), se kterým sice dosahují modely lepších výsledků, ale z hlediska praktické aplikace je tento přístup problematický:

[15] Anderson, B., Paul, S., McGrew, D.: Deciphering malware's use of TLS (without decryption). J. Comput. Virol. Hacking Tech. 14, 195–211 (2018).

- Interarrival time může být ovlivněn faktory, které nesouvisí s chováním sledované aplikace nebo malwaru, například latencí na síťových uzlech, přetížením sítě, nebo zpožděním způsobeným retransmisemi na nižších vrstvách síťového stacku (např. TCP).
- Síťové podmínky, jako jsou vysoká latence, kolísající šířka pásma nebo přítomnost QoS mechanismů, mohou interarrival time výrazně zkreslit, což vede k nepřesnostem při identifikaci nebo klasifikaci provozu.
- Interarrival time paketů může být ovlivněn plánováním procesů v operačních systémech, výkonem zařízení, či implementací síťového stacku. Tato nepředvídatelnost snižuje kvalitu atributu jako prediktoru.
- Hodnota interarrival time může být závislá na tom, kde v síti se měření provádí. Rozdílné podmínky v těchto bodech mohou vést ke zkreslení hodnot.

Kvůli těmto faktorům může být vhodnější spoléhat na atributy jako velikosti paketů nebo sekvence TLS segmentů, které jsou stabilnější, méně ovlivněné vnějšími faktory, a více reflektují strukturu a účel komunikace, nikoliv náhodné nebo dynamické vlivy prostředí.

Použitý přístup zahrnuje pouze použití dvou klíčových typů atributů (features): N-prvních velikostí paketů a M-prvních TLS segmentů (v případě TLS komunikace). Postup tvorby modelu pro anotovaná data malware komunikace (každé spojení má přiřazen label malware family) je následující.

1. **Extrahování atributů:** V první fázi se ze síťového spojení extrahují velikosti N prvních paketů a délky M prvních TLS segmentů. Tyto atributy zachycují vzory komunikace mezi klientem a serverem, například délku přenosu handshake zpráv, přenos certifikátů, nebo výměnu šifrovacích klíčů.
2. **Předzpracování dat:** Data získaná z velikostí paketů a TLS segmentů jsou normalizována tak, aby byla vhodná pro strojové učení. Jelikož chceme rozlišit odchozí od příchozí komunikace, negujeme hodnoty velikosti paketů příchozí komunikace.
3. **Trénování modelu:** Na základě předzpracovaných atributů je vytvořen ML model. Pro trénování modelu, který je reprezentován binárním klasifikátorem používáme jednak malware vzorky, ale také ostatní vzorky, včetně normální komunikace. Při trénování v závislosti na použitém ML algoritmu je nutné kompenzovat také nevyváženost tříd (množina spojení konkrétní malware rodiny je typicky řádově menší než ostatní provoz).

V experimentech jsme zkoušeli různé ML algoritmy a přístupy pro tvorbu klasifikátorů. Jako vhodný se zdál přístup, kdy jsou trénovány klasifikátory pro odlišení normálního a malware provozu a posléze je identifikovaný malware provoz analyzován klasifikátory pro identifikaci konkrétní malware family.

Trénování klasifikátoru pro identifikaci malware tak umožňuje použít celou datovou sadu malware a datovou sadu normálního provozu. V případě identifikace malware family, pak můžeme natrénovat vícetřídní klasifikátor pro identifikaci rodiny nebo skupinu binárních klasifikátorů pro každou z rodin.

Malware vzorky

Pro vytvoření modelů detekce je nutné nejprve analyzovat dostupné známé vzorky malware. Pro tento účel se používá systém, který zahrnuje virtuální izolované prostředí pro statickou a dynamickou analýzu malware, které umožňuje monitorovat síťovou komunikaci v prostředí a monitorovací nástroje pro získání informací o síťové komunikaci. Z dostupných možností bylo vybráno prostředí Tria.ge, které nabízí komunitně orientovaný přístup k analýze nových podezřelých vzorků.

Tria.ge

Systém Hatching Tria.ge je moderní sandboxová platforma navržená pro komunitní analýzu malwaru, která umožňuje automatizované zpracování podezřelých souborů nebo odkazů v izolovaném virtuálním prostředí a nabízí kombinaci statické i dynamické analýzy. Statická analýza dekompiluje kód a identifikuje podezřelé vzorce, zatímco dynamická analýza spouští malware v reálném čase, aby sledovala jeho chování, jako je manipulace s registry, komunikace s C2 servery nebo změny v systému. Platforma automaticky generuje indikátory kompromitace (IoC), včetně URL, doménových jmen, IP adres či hashů, které jsou klasifikovány a ukládány pro budoucí vyhledávání a analýzu. Tria.ge podporuje komunitní sdílení výsledků mezi uživateli, čímž zvyšuje povědomí o aktuálních hrozbách a umožňuje spolupráci mezi jednotlivci i organizacemi. Využívá strojové učení a heuristické techniky k detekci neznámého malwaru, analyzuje vzory chování a porovnává je s databází známých hrozeb. Systém podporuje analýzu na různých operačních systémech, jako jsou Windows, Linux a Android, a emuluje běžné konfigurace pro co nejpřesnější výsledky. Výsledky analýzy jsou prezentovány pomocí přehledných vizualizací, zahrnujících grafy, časové osy a podrobné výpisy aktivit, což usnadňuje interpretaci dat. Analýza probíhá v izolovaném prostředí, které zajišťuje bezpečnost a zabraňuje šíření malwaru.

Analýza vzorků

Platforma Hatching Triage provádí analýzu vzorků malwaru pomocí kombinace automatizovaných metod a virtualizovaných prostředí, která umožňují detailní sledování chování vzorků v reálném čase. Uživatelé mohou nahrát podezřelé soubory, odkazy nebo emailové přílohy prostřednictvím uživatelského rozhraní nebo API. Systém podporuje různé formáty souborů, včetně spustitelných souborů, dokumentů, skriptů a dalších typů potenciálně škodlivého obsahu. Každý vzorek je spuštěn v izolovaném virtuálním prostředí, které simuluje reálný operační systém. Toto prostředí emuluje běžné uživatelské chování, jako je otevírání souborů, spuštění programů nebo interakce se systémem, aby byl malware přiměn k aktivaci svých funkcí. Hatching Triage podporuje analýzu na různých operačních systémech, jako jsou Windows, Linux a Android. Nabízí možnost výběru konkrétních konfigurací systému, aby se zvýšila pravděpodobnost aktivace malwaru, například simulací různých verzí OS, jazykového prostředí nebo specifických softwarových nastavení. Během spuštění vzorku systém sleduje jeho aktivity, jako je manipulace se soubory, registry, pamětí, síťová komunikace, injektování kódu do jiných procesů nebo šifrovací operace. Tyto aktivity jsou zaznamenávány a analyzovány v reálném čase. Pokud vzorek nelze spustit, nebo jako

doplňková metoda, Hatching Triage provádí statickou analýzu. Ta zahrnuje dekompilaci kódu, analýzu struktur binárního souboru, identifikaci podezřelých vzorců, jako jsou známé řetězce, hash hodnoty nebo použité knihovny. Systém automaticky extrahuje indikátory kompromitace, jako jsou URL, IP adresy, doménová jména, hashe souborů, a další artefakty, které mohou být použity k detekci podobného malwaru nebo při budování detekčních pravidel.

Získání IoC

Pro získání informací k analyzovaným vzorkům je k dispozici API. MalwareRadar používá již analyzované vzorky pro rodiny malware. Byl vytvořen automatizovaný nástroj, který umožňuje získat informace k vybraným malware rodinám stažením informací k analyzovaným vzorkům. Tento PowerShell skript slouží k automatizovanému získávání reportů o malware z platformy Triage prostřednictvím jejího API. Jeho hlavním cílem je usnadnit sběr metadat, detailních analýz a souvisejících souborů pro vzorky malwaru konkrétní rodiny. Na základě těchto informací je pak možné vytvořit různé modely pro detekční metody malware. Skript poskytuje efektivní způsob, jak získat informace o malwaru, které obashují přehledovou zprávy z analýzy, detailní zprávy z statických a dynamických analýz a síťovou komunikaci ve formě PCAP souborů. Příklad použití nástroje je následující:

```
.\Triage.Collect-Reports.ps1 -ApiKeyFile "C:\apikeys\triage_api.txt" `
                             -FamilyName "agenttesla" `
                             -Samples 5 `
                             -IncludePcap `
                             -IncludeDynamic `
                             -OutputFolder "C:\malware_samples"
```

Skript vyžaduje platný API klíč pro přístup k platformě Triage a specifikaci několika parametrů:

-ApiKeyFile	Povinný parametr obsahující cestu k souboru s API klíčem.
-FamilyName	Povinný parametr určující název malware rodiny.
-Samples	Volitelný parametr určující počet vzorků ke stažení (výchozí hodnota je 10).

-IncludePcap	Volitelný parametr, který povoluje stahování PCAP souborů.
-IncludeDynamic	Volitelný parametr, který povoluje stahování dynamických analýz.
-LeaveIncomplete	Volitelný parametr, který rozhoduje, zda mají být ponechány neúplné stažené soubory (výchozí hodnota je true).

Pro každý vzorek zadané malware rodiny se během analýzy získávají následující čtyři soubory obsahující klíčové informace:

- PCAP soubor: Obsahuje kompletní záznam síťové komunikace, která proběhla během analýzy vzorku. Tento soubor slouží jako základ pro detailní zkoumání síťového chování malwaru a pro extrakci specifických síťových atributů.
- Overview JSON: Tento soubor kombinuje výsledky statické a dynamické analýzy vzorku. Obsahuje především seznam indikátorů kompromitace (IoC), jako jsou IP adresy, domény a URL, které jsou pro další analýzu kriticky důležité. IoC z tohoto souboru se přímo využívají při tvorbě modelů zaměřených na detekci malwaru pomocí identifikace podezřelých entit.
- Static JSON: Obsahuje podrobné informace získané čistě statickou analýzou, včetně aplikace pravidel YARA, detekce známých vzorců a dalších charakteristik vzorku. Tyto informace jsou užitečné pro statické modely detekce a klasifikace malwaru.
- Summary JSON: Poskytuje přehled o vzorku a shrnuje výsledky analýzy. Obsahuje obecné informace o malwaru, jeho chování a klasifikaci. Tento soubor slouží jako přehledná referenční informace.

Získané soubory jsou dále zpracovány podle požadavků jednotlivých modelů:

- IoC modely: Přímá extrakce IoC informací (IP adresy, domény, URL, SNI, JA3) se provádí z overview JSON souboru. Tyto informace se používají pro detekční mechanismy založené na identifikaci známých indikátorů kompromitace.
- Síťové otisky a ML modely: Pro práci s atributy síťové komunikace, jako jsou sekvence paketů nebo vlastnosti TLS segmentů, je použit nástroj *Joy*. Ten extrahuje potřebné informace přímo z PCAP souborů a připravuje je pro další analýzu nebo trénink strojového učení. Byly také vytvořeny nástroje pro předzpracování těchto dat pro účely další analýzy. Příklad zpracovaného vzorku malware do podoby IoC je na Obrázku 26:

```
{
 "domains": [
   "api.ipify.org",
   "8.8.8.8.in-addr.arpa",
   "138.32.126.40.in-addr.arpa",
   "172.214.232.199.in-addr.arpa",
   "205.13.26.104.in-addr.arpa",
   "15.164.165.52.in-addr.arpa",
   "183.59.114.20.in-addr.arpa",
   "192.142.123.92.in-addr.arpa",
   "88.210.23.2.in-addr.arpa",
   "81.144.22.2.in-addr.arpa",
   "29.243.111.52.in-addr.arpa"
 ],
 "sni": [
   "api.ipify.org"
 ],
 "sample": "240716-b2a7bssale",
 "ips": [
   "8.8.8.8",
   "104.26.13.205"
 ],
 "urls": [
   "https://api.ipify.org/"
 ],
 "ja3": [
   "4d7a28d6f2263ed61de88ca66eb011e3",
   "3b5074b1b5d032e5620f69f9f700ff0e"
 ],
 "family": [
   "agenttesla"
 ]
}
```


Obrázek 26: Příklad IoC souboru pro vzorek malware

Tímto způsobem se zajišťuje komplexní zpracování informací z analýzy vzorků dané malware rodiny, přičemž získaná data slouží jako vstup pro různé analytické nástroje a modely detekce malwaru. Tyto modely je možné automaticky aktualizovat pomocí periodického získávání nových vzorků malware z prostředí Tria.ge.

Architektura

MalwareRadar je nástroj pro detekci malwaru založený na kontextové analýze síťového provozu. Díky modulární architektuře umožňuje flexibilní nasazení a integraci s dalšími řešeními monitorování sítě, jako jsou IPFIX sondy a kolektory, SIEM systémy (např. QRadar) a další bezpečnostní nástroje. Tato modulární architektura podporuje flexibilitu, škálovatelnost a snadné nasazení v různých prostředích prostřednictvím kontejnerizace v Dockeru.

MalwareRadar také kombinuje různé metody detekce malware (popsané v podkapitole Principy detekce). Dále pak obsahuje komponenty nutné pro získání vstupních informací o síťových tocích, jejich následné zpracování — agregaci pro jednotlivé zdrojové komunikující stanice, sběr výsledků analýzy a reportování výsledků do externích nástrojů či SIEM QRadar. Tyto jednotlivé komponenty jsou tvořeny Docker kontejnery, které mezi sebou komunikují pomocí gRPC protokolu.

Hlavní komponenty architektury systému jsou (viz Obrázek 27):

FlowReader je první komponenta pipeline, která čte síťové záznamy (IPFIX) reprezentované ve formátu JSON a převádí je do uniformního formátu vhodného pro další zpracování. Tento nástroj je navržen jako univerzální a efektivní řešení pro čtení a konverzi síťových dat, přičemž podporuje různorodé vstupní formáty, jako jsou záznamy ze systémů Flowmon, IPFIXCOL nebo Suricata. FlowReader následně posílá převedená data do následujícího modulu ContextCollector, čímž zajišťuje plynulý tok dat v rámci pipeline. Klíčové vlastnosti FlowReaderu:

- Podpora různých vstupních formátů: FlowReader dokáže zpracovávat síťové toky ve více podporovaných formátech, což z něj činí flexibilní nástroj pro různé prostředí a požadavky.
- Uniformní datové objekty: Bez ohledu na formát vstupních dat FlowReader převádí informace do jednotných datových objektů. Tím zajišťuje konzistenci a usnadňuje další manipulaci a analýzu dat v následujících modulech.
- Flexibilní výstupní formáty: Uživatelé mohou zvolit výstupní formát zpracovaných dat podle svých potřeb. FlowReader podporuje jak binární formát (Protocol Buffers), který je efektivní a vhodný pro vysoký výkon, tak formát JSON, který je snadno čitelný a vhodný pro analýzu či integraci s dalšími nástroji.
- Jednoduchost a efektivita: FlowReader je navržen s důrazem na snadné použití a vysokou výkonnost. Umožňuje rychlé zpracování síťových dat z různých zdrojů, což z něj činí klíčovou komponentu v analytických nástrojích zaměřených na síťový provoz.

ContextCollector je klíčový modul, který agreguje a obohacuje data získaná z modulu FlowReader, aby vytvořil komplexní kontext síťových aktivit pro jednotlivé monitorované stanice. Na základě vstupních záznamů IPFIX generuje host-based temporální kontexty, což znamená, že shromažďuje a strukturuje síťovou komunikaci podle jednotlivých hostitelů (např. IP adres) v definovaných časových intervalech. Takto vytvořený kontext zahrnuje podrobné informace o síťových aktivitách konkrétního hostitele během určitého časového okna, což umožňuje přesnější analýzu a identifikaci škodlivé aktivity. Klíčové vlastnosti ContextCollectoru:

- Agregace komunikace podle hostitelů: Kontexty jsou generovány na základě komunikace jednotlivých hostitelů. Klíčovým identifikátorem je IP adresa hostitele. Agregovaná data zahrnují všechny relevantní aktivity spojené s daným hostitelem.
- Obohacení síťových dat: Modul extrahuje a zpracovává detailní informace, jako jsou Identifikace všech síťových spojení, která hostitel vytvořil, informace o doménách, ke

kterým hostitel přistupoval, detailní údaje o šifrované komunikaci, včetně verze protokolu a certifikátů.

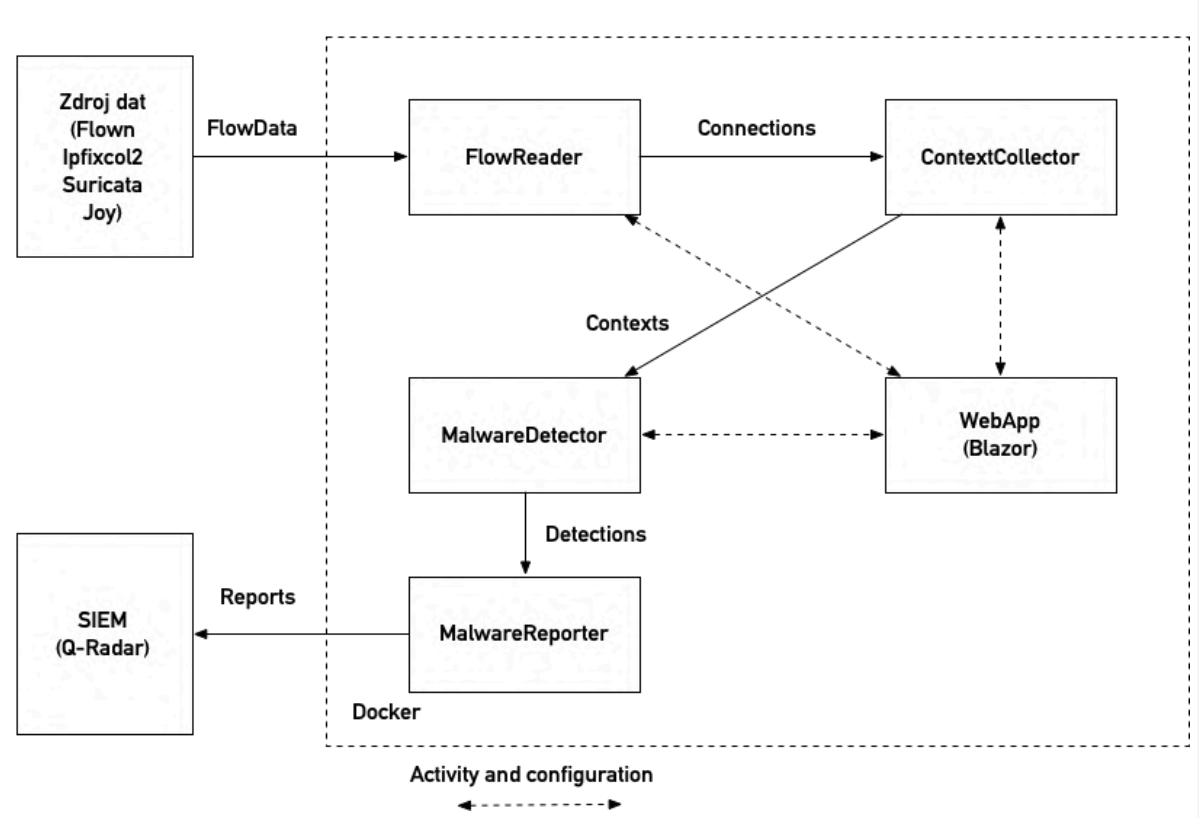
- Definice parametrů kontextu: ContextCollector umožňuje přizpůsobení výpočtu kontextu na základě vlastní konfigurace, která definuje specifické parametry (např. metriky toků nebo vztahy mezi přenosy). Tato flexibilita umožňuje nasazení v různých prostředích s odlišnými analytickými potřebami. Kontexty jsou generovány na základě definovaných časových oken, což umožňuje detailní analýzu aktivit hostitelů v průběhu času. Každé okno obsahuje ucelený pohled na síťové aktivity konkrétního hostitele.
- Předání obohacených dat: Modul posílá vytvořené kontexty do následujícího modulu MalwareDetector, který je dále analyzuje za účelem identifikace potenciálně škodlivého chování.

MalwareDetector je klíčový modul pro detekci malwaru v síťové komunikaci, který analyzuje obohacený kontext poskytovaný modulem ContextCollector. Tento modul je navržen tak, aby identifikoval výskyt malwaru na základě sledování síťových toků a detekce indikátorů kompromitace (IoC). Využívá předdefinované profily malwaru, které jsou vytvořeny na základě IoC poskytovaných externím rámcem pro analýzu malwaru. Modul kombinuje různé metody detekce, jako jsou porovnávání známých signatur, detekce anomálií a fuzzy přístup k vyhodnocování dat, což umožňuje flexibilní a přesnou analýzu. Klíčové vlastnosti MalwareDetectoru jsou:

- Monitorování síťové komunikace: Sleduje a analyzuje síťový provoz hostitelských stanic s cílem identifikovat podezřelé aktivity a zaměřuje se na komunikaci spojenou s malwarem, například připojení k nebezpečným IP adresám nebo přenos šifrovaných dat s podezřelými TLS parametry.
- Detekce založená na IoC: Používá indikátory kompromitace (IoC), jako jsou IP adresy, doménová jména, URL, TLS fingerprinty JA3 a SNI. Detekční mechanismy jsou postaveny na známých IoC malwaru, což zvyšuje spolehlivost identifikace.
- Integrace s externí analýzou malwaru: Malware profily jsou sestavovány na základě IoC poskytnutých externím rámcem pro analýzu malwaru. Tato integrace zajišťuje přesnost a aktuálnost detekčních modelů.
- Struktura modelu: Každý model se skládá z kolekcí FuzzySet pro jednotlivé typy IoC, zejména: IP Address, Domain Name, URL, TLS JA3, TLS SNI.
- Skórování a detekce: Výstupem modulu je skóre, které reprezentuje pravděpodobnost přítomnosti malwaru na základě počtu a typu detekovaných IoC. Toto skóre poskytuje kvantitativní měřítko, které pomáhá vyhodnotit bezpečnostní stav analyzovaného hostitele.
- Flexibilita detekčních metod: MalwareDetector umožňuje použití různých detekčních přístupů, od statických signatur až po pokročilé algoritmy pro detekci anomálií. Výstupy mohou být generovány ve formátu JSON, což usnadňuje další zpracování a integraci do jiných nástrojů.

MalwareDetector používá FuzzySet přístup k vyhodnocování IoC, což umožňuje flexibilní porovnávání a skórování na základě pozorovaných vzorců v síťové komunikaci. Tento přístup je obzvláště vhodný pro detekci malwaru v dynamických prostředích, kde mohou být jednotlivé ukazatele kompromitace variabilní.

MalwareDetector představuje jádro detekčního procesu zaměřeného na odhalení škodlivých aktivit na základě síťové komunikace. Jeho modulární a otevřená struktura umožňuje přizpůsobení různým prostředím a scénářům, což z něj činí univerzální nástroj pro moderní kybernetickou bezpečnost. Kombinace síťového monitorování, analýzy IoC a fuzzy přístupu zajišťuje efektivní detekci i u pokročilých typů malwaru.



Obrázek 27: Architektura systému MalwareRadar

Dále architektura obsahuje další podpůrné moduly pro vizualizace, sledování funkčnosti systému a a další integraci s ostatními bezpečnostními nástroji:

Webové rozhraní (Web App): Je postavené na technologiích Blazor a Telerik a slouží jako uživatelsky přívětivý portál pro administrátory, kteří mohou konfigurovat, monitorovat a analyzovat bezpečnostní data.

SIEM integrace (QRadar): Analyzovaná data jsou integrována do SIEM systému QRadar, což umožňuje centralizované zpracování a lepší přehled o bezpečnostních incidentech. QRadar zajišťuje přehledné zobrazení, korelaci událostí a podporu při vyšetřování hrozeb.

Komunikace mezi moduly je možná pomocí:

Standard Input/Output Streams: Jednoduchý způsob propojení komponent, který umožňuje spouštění například jako UNIX pipeline.

GRPC: Vhodné pro robustní a škálovatelné integrace, zejména v distribuovaných prostředích nebos s použitím technologie kontejnerů (Docker Compose).

Vstupní data

Vstupní data lze získat z různých nástrojů, například Suricata, IPFIX sond, či extrakcí z PCAP souborů, či živého provozu pomocí například nástroje tshark. Nástroj MalwareRadar podporuje vstup z nástrojů Suricata, Joy, a ipfixcol, případně pro offline experimenty také informace generované nástrojem tshark.

Vstupní data z různých nástrojů a formátů jsou čtena komponentou FlowReader, která tyto data unifikuje pro další použití v systému. Tento výstup reprezentuje strukturovaný popis síťového toku, zahrnující identifikaci toku, statistiky přenosu a podrobnosti o šifrované komunikaci pomocí TLS. Sekce flowKey obsahuje klíčové vlastnosti toku, jako jsou zdrojová a cílová IP adresa, porty a použitý protokol, což umožňuje jednoznačnou identifikaci komunikace. Sekce flowData zahrnuje informace o průběhu toku, včetně počtu přenesených paketů, bajtů, času zahájení a trvání toku, a také identifikaci aplikace (například SSL). Podsekce tlsFlow poskytuje detailní informace o TLS komunikaci, jako je název serveru, verze TLS, a údaje o vydavateli certifikátu, včetně jeho platnosti, což je klíčové pro analýzu bezpečnosti přenosu. Tato struktura je užitečná pro detekci anomálií, monitorování síťového provozu a forenzní analýzu, zejména při zkoumání šifrované komunikace a jejich bezpečnostních aspektů. Příklad takového toku je na Obrázku 28.

```
{
  "flowKey": {
    "version": "IPv4",
    "protocol": "TCP",
    "sourceAddress": {
      "version": "IPv4",
      "address": "wKhvIA=="
    },
    "sourcePort": 62117,
    "destinationAddress": {
      "version": "IPv4",
      "address": "mMcToQ=="
    },
    "destinationPort": 443
  },
  "flowData": {
    "flowType": "BIDIRECTIONAL",
    "timeStart": "2024-04-08T07:36:46.052420Z",
    "timeDuration": "75.030138s",
    "applicationTag": "SSL",
    "sentPackets": 14,
    "sentOctets": "1568",
    "recvPackets": 39,
    "recvOctets": "49667",
    "tlsFlow": {
      "issuerCommonName": "DigiCert SHA2 Secure Server CA",
      "subjectCommonName": "*.vo.msecnd.net",
      "subjectOrganisationName": "Microsoft Corporation",
      "serverNameIndication": "az667904.vo.msecnd.net",
      "serverVersion": "TLS1_2",
      "certificateValidityFrom": "2024-01-29T23:00:00Z",
      "certificateValidityTo": "2025-01-30T22:59:59Z"
    }
  }
}
```


Obrázek 28: Příklad unifikovaného obousměrného toku pro TLS spojení

Výstup modulu

Výstupem MalwareRadar je JSON výstup, který pro analyzovaný kontext poskytuje výsledky detekce malware. Tento JSON výstup poskytuje detailní informace o analýze síťového provozu a detekci potenciální škodlivé aktivity. Obsahuje časovou stopu, identifikátor reportu, informace o zdroji detekce, detaily analyzovaného hostitele, souhrn analyzovaného síťového kontextu a výstupy jednotlivých detekčních mechanismů. Obsah výstupu tvoří následující položky:

Obecné informace	
timestamp	Čas vytvoření reportu v UTC formátu
report_id	Jedinečný identifikátor reportu.
source	Zdroj detekce, v tomto případě DomainRadar,
Informace o hostiteli	
ip_address	IP adresa analyzovaného zařízení.
hostname	Název analyzovaného zařízení, je-li k dispozici.
os	Operační systém je-li znám.
Časové okno analýzy	
start	Čas zahájení analýzy
end	Čas ukončení analýzy
Souhrn informací síťového kontextu	
all_connections	Celkový počet zaznamenaných síťových spojení.
tls_connections	Počet spojení přes protokol TLS.
dns_connections	Počet DNS požadavků.

http_connections	Počet HTTP spojení.
Informace o hrozbě (alert)	
severity	Úroveň závažnosti detekce.
malware_detected	Informace, že byl detekován malware.
confidence_score	Pravděpodobnost detekce malwaru na základě modelu.
description	Stručný popis detekované aktivity
evidence	Seznam podezřelých spojení.
malware_details	Informace o detekovaném malwaru (název, popis, odkaz).
Detailní informace k detekční metodě — Fuzzy detekce	
score	Celkové skóre fuzzy detekce.
matched_indicators	Indikátory kompromitace (IoC) odpovídající známým vzorcům (typ, hodnota, klíč spojení).
Detailní informace k detekční metodě — Detekce sekvencí paketů	
threshold	Prahová hodnota pro vyhodnocení shody.
matched_connections	pojení odpovídající známým sekvencím paketů, tj. přesahující threshold (klíč spojení a skóre).

JSON výstup je generován pro každý analyzovaný kontext. Tento výstup je možné filtrovat na základě hodnot v sekci informace o hrozbě (severity nebo malware_detected) případně i pomocí confidence_score předtím než je záznam zaslán na další zpracování do SIEM. Případně je možné zasílat všechny generované záznamy a další zpracování provádět v SIEM. Příklad JSON výstupu (zjednodušený):

```
  "timestamp": "2025-01-09T14:30:00Z",
  "report_id": "AFAADBBA-2D1A-4DCF-8A83-128A6E9872E2",
  "source": "DomainRadar",
  "host_details": {
    "ip_address": "192.168.1.101",
```

```
    "hostname": "host1.local",
  },
  “analysis_window” : {
    “start” : "2025-01-09T14:25:00Z",
    “end” : "2025-01-09T14:30:00Z"
  }
  “analysis_context” : {
    “all_connections” : 58,
    “tls_connections” : 39,
    “dns_connections” : 5,
    “http_connections” : 3
  }
```

```
 "alert": {
    "severity": "high",
    "malware_detected": true,
    "confidence_score": 0.92,
    "description": "Network activity consistent with malware communication.",
    "evidence": [
     {
       "src_ip": "192.168.1.101",
       "src_port": 52345,
       "dst_ip": "93.184.216.34",
       "dst_port": 443,
       "protocol": "TLS",
       "timestamp": "2025-01-09T14:28:45Z"
     },
     {
       "src_ip": "192.168.1.101",
       "src_port": 52347,
       "dst_ip": "18.14.126.87",
       "dst_port": 443,
       "protocol": "TLS",
       "timestamp": "2025-01-09T14:29:14Z"
      }
    ],
    "malware_details": {
      "name": "Emotet",
      "description": "Emotet is a sophisticated banking Trojan known for stealing
sensitive data and spreading via phishing emails.",
      "reference": "https://malpedia.caad.fkie.fraunhofer.de/details/win.emotet"
    }
  }
```

```
"fuzzy_detection": {
    "score": 0.90,
    "matched_indicators": [
      {
        "type": "IP",
        "value": "93.184.216.34",
        "connection_key" : “TCP@192.168.1.101:52345<>93.184.216.34:443”
      },
      {
        "type": "JA3",
        "value": "769d5b9e6183c1d74a7ef10154e5cbed",
        "connection_key" : “TCP@192.168.1.101:52347<>18.14.126.87:443”
      }
    ]
  }
```

```
"packet_sequence_detection": {
   “threshold” : 0.8,
   “matched_connections” : [
    {
     "connection_key" : “TCP@192.168.1.101:52347<>18.14.126.87:443”,
     “score” : 0.83
    },
  ]
}
```

Vývojářská dokumentace

Primární složkou vývojářské dokumentace jsou samotné zdrojové kódy obou modulů. Kód je psán tak, aby byl samovysvětlující, a je doplněn dokumentačními komentáři. Klíčové části kódu jsou detailně komentovány, aby byl jasně pochopitelný jejich účel. Komentáře popisují jak rozhraní jednotlivých metod, tak i procesní logiku jejich implementace. Tímto způsobem je zajištěno, že jsou předkládané zdrojové kódy srozumitelné interním a externím vývojářům, což zvyšuje transparentnost a udržitelnost vytvořeného řešení. V podsložkách archivu s dosaženým výsledkem jsou ke každému dílčímu projektu přiloženy soubory „README“, ve kterých jsou projekty popsány, a to včetně principů funkce, členění kódu a kroků nutných k jejich konfiguraci a spuštění. Následující sekce obsahují vysokoúrovňový vývojářský popis některých komponent.

DomainRadar

Nástroj DomainRadar je tvořen řadou dílčích komponent, které jsou založeny na následujících otevřených platformách:

- Apache Kafka[16] a Apache Kafka Connect ve verzi 3.8.1.
- PostgreSQL[17] ve verzi 16.
- Komponenta Loader & Pre-filter, některé kolektory, extraktor příznaků a klasifikační komponenty využívají jazyk Python (verze 3.11).
    - Kolektory a extraktor příznaků jsou postaveny na knihovně pro zpracování dat Faust[18] (verze 0.11.3).
    - Klasifikační mikroslužba používá knihovnu confluent-kafka (verze 2.6.0).
    - Klasifikační modely jsou postaveny na knihovnách tensorflow + keras (verze 2.15.0), xgboost (verze 2.0.3), lightgbm (verze 4.3.0) a torch (verze 2.1.2).
- Komponenta Data Merger využívá platformu Apache Flink[19] verze 1.20 a vlastní kód je napsán v jazyce Java (verze 17).
- Některé kolektory jsou implementovány v jazyce Java (verze 21) a využívají knihovnu pro paralelizované zpracování dat Confluent Parallel Consumer[20] (verze 0.5.3.2).
- Webové rozhraní je postaveno na platformách Node.js (verze 21) a Nuxt[21] (verze 3.11) a implementováno v jazyce TypeScript.

Všechny komponenty implementované v jazyce Python využívají pro správu závislostí software Poetry[22]. Všechny komponenty implementované v jazyce Java využívají pro

[16] https://kafka.apache.org/
[17] https://www.postgresql.org/
[18] https://faust-streaming.github.io/faust/
[19] https://flink.apache.org/
[20] https://github.com/confluentinc/parallel-consumer
[21] https://nuxt.com/
[22] https://python-poetry.org/

správu závislostí a životního cyklu software Apache Maven[23]. Webové rozhraní využívá pro správu závislostí software Yarn[24].

Konfigurace

Každá komponenta obsahuje statickou část konfigurace, kterou načítá z proměnných prostředí nebo z konfiguračního souboru. Některé komponenty podporují také dynamickou konfiguraci, která je uložena ve službě Apache Kafka. Mechanismus dynamické konfigurace je podrobně popsán v souboru `DomainRadar/configuration_exchange.md`.

Pro komponentu Loader & Pre-filter je nutné staticky načíst pouze přístupové údaje k serveru Apache Kafka. Konfigurace upravující jednotlivé vstupní, filtrační a výstupní moduly je dynamická. Bližší informace jsou uvedeny v souboru `DomainRadar/input/README.md`.

Všechny kolektory, komponenta Data Merger, extraktor příznaků a klasifikační služba jsou konfigurovány výhradně staticky. Vzorové konfigurace, ve kterých jsou jednotlivé konfigurační položky a mechanismy detailně popsané, jsou dostupné v souborech:

- `DomainRadar/colext/python/collector/config.example.toml`
- `DomainRadar/colext/java/collector.example.properties`
- `DomainRadar/colext/java/merger.example.properties`
- `DomainRadar/colext/python/extractor/config.example.toml`
- `DomainRadar/colext/python/classifier_unit/config.example.toml`

Při spouštění v kontejnerech (s použitím dodané specifikace Compose, viz sekci Instalační příručka) je možné konfiguraci těchto komponent za běhu měnit s využitím doplňkové experimentální služby `config_manager`. Ta provádí synchronizaci konfiguračních souborů s úložištěm ve službě Apache Kafka a restartuje ostatní Compose služby. Musí tedy být schopna spravovat místního Docker démona – je nutné ji spustit přímo na hostitelském stroji (s příslušnými právy). Alternativou je na hostitelském stroji spustit pouze doplňkový skript `config_manager_daemon.py`, který naslouchá na unixovém socketu a podle příchozích zpráv ze služby config_manager spouští nebo vypíná ostatní Compose služby.

Databáze a její integrace se systémem

Schéma databáze relačního systému PostgreSQL, které bylo naznačeno v Obrázku 8, je formálně popsáno v souborech v adresáři `DomainRadar/infra/postgres/init/sql`. Skript `10_create_domainradar_db.sql` vytvoří požadované tabulky i procedury triggerů v jazyce PL/pgSQL, které zajišťují uložení komplexních vstupních objektů do normalizovaného schématu. Skript `15_seed.sql` vloží do tabulek statická data (typy kolektorů a jejich stavových kódů, typy klasifikátorů). Skript `20_grant_access.sql`

[23] https://maven.apache.org/
[24] https://yarnpkg.com/

nastavuje oprávnění databázovým uživatelům, kteří jsou použití v ukázkovém běhovém prostředí (viz sekci Instalační příručka).

Konfigurace konektorů platformy Kafka Connect, taktéž naznačených v Obrázku 8, je uložena v adresáři `DomainRadar/infra/kafka_connect/properties`. Pro snadnější výměnu komplexních objektů je implementován vlastní zásuvný modul pro Kafka Connect s doménově specifickými transformacemi, a to v adresáři `DomainRadar/colext/java/connect`. Přiložen je také Containerfile `DomainRadar/colext/connect.Dockerfile`, který projekt přeloží a sestaví kontejner, ve kterém poběží samostatná instance Kafka Connect s vlastním zásuvným modulem.

Kolektory

Kolektory HTML & TLS, NERD, GEO-ASN a QRadar jsou implementovány v jazyce Java, jejich zdrojový kód je k dispozici v adresáři `DomainRadar/colext/java/standalone-collectors`. Pro čtení a vysoce paralelizované zpracovávání dat je využita knihovna Confluent Parallel Consumer. Pro rozšíření o nové kolektory postačí po vzoru existujících kolektorů vytvořit novou třídu dědící z `BaseStandaloneCollector` (sběr pouze podle doménového jména) nebo `IPStandaloneCollector` (sběr podle DN a IP adresy) a upravit vstupní třídu `StandaloneCollectorRunner`, ve které metoda `initCollectors` kolektory inicializuje podle argumentů příkazové řádky.

Kolektory Zone, DNS, RDAP-DN a RDAP-IP jsou implementovány v jazyce Python s využitím rámce Faust, který řídí komunikaci se službou Kafka, a knihovny Pydantic pro de-/serializaci a validaci přenášených zpráv. Pro rozšíření o nové kolektory je možné využít jako vzor kolektor DNS (sběr pouze podle DN) nebo kolektor RTT (sběr podle DN a IP adresy). Postačí zkopírovat příslušný podadresář z `DomainRadar/colext/python/collector/collectors` a patřičně upravit funkci `process_entries`. Pro automatické sestavování obrazů kontejnerů je vhodné nový kolektor přidat také do definic v úvodu skriptu `DomainRadar/colext/build_images.sh`.

Přidaným kolektorům je nutné jim přiřadit jedinečný číselný a textový identifikátor a přidat jej do tabulky `Collector` v databázi. Pro přidání kolektoru, který sbírá data pro IP adresy, stačí nový kolektor zaregistrovat ve třídě `TagRegistry` projektu `serialization` (`DomainRadar/colext/java/serialization`). Pro přidání kolektoru, který sbírá data jen pro doménové jméno, by bylo nutné provést komplexnější úpravy sdílených datových struktur (zejm. modelu `AllCollectedData`) a komponenty Data Merger (zejm. třídy `DomainEntriesProcessFunction` a modelu `KafkaDomainAggregate`).

Uživatelská dokumentace

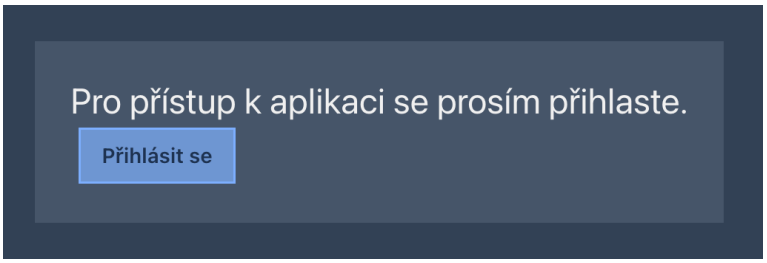
Tato kapitola obsahuje uživatelskou dokumentaci k oběma vytvořeným modulům: DomainRadar i MalwareRadar.

DomainRadar

Návod k webovému rozhraní

Tato sekce dokumentuje interaktivní webové rozhraní modulu DomainRadar a poskytuje uživatelům návod k použití.

Přihlášení a úvodní stránka



Obrázek W1: Výzva k přihlášení uživatele

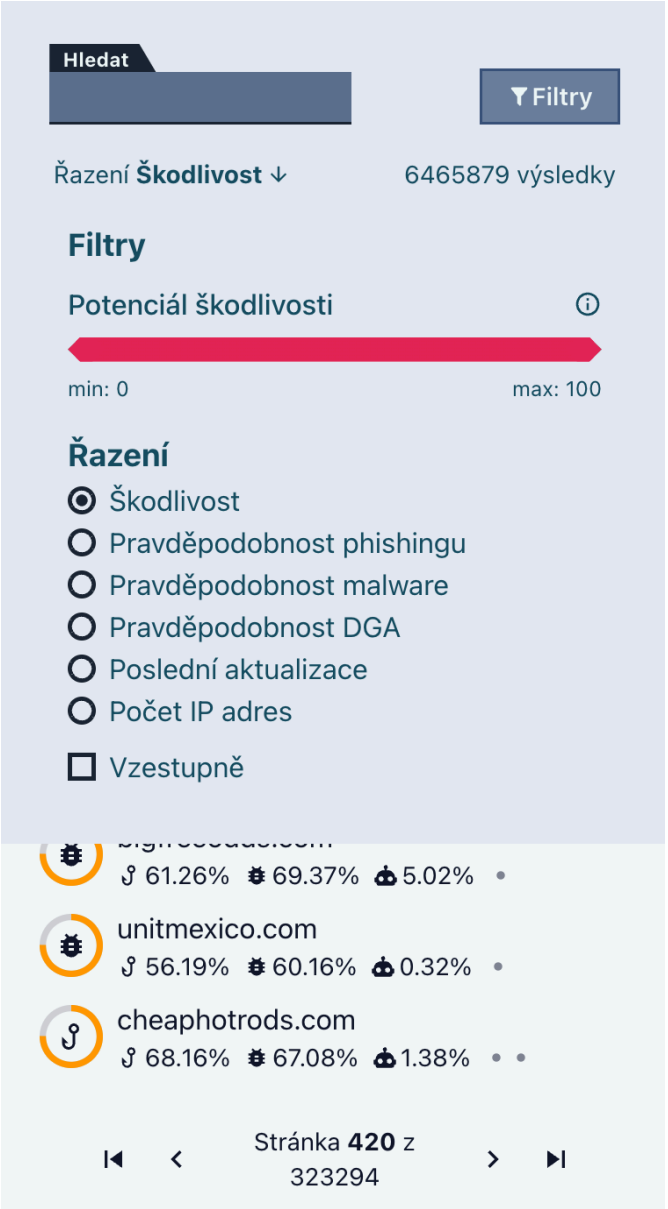
Na úvodní stránce se zobrazuje přihlášený uživatel nebo výzva k přihlášení (Obrázek W1). Nejprve se přihlaste, jinak nebudete mít přístup k ostatním částem aplikace ani API. Po přihlášení můžete používat odkazy v záhlaví nebo kliknutím na tlačítko „Otevřít aplikaci“ přejít na hlavní zobrazení.

Hlavní zobrazení domén

Boční panel v zobrazení domén (na Obrázku W2) vypisuje domény zachycené a klasifikované systémem po stránkách. V horní části je vyhledávací pole, kde můžete domény filtrovat podle shody jmen. Kliknutím na aktuálně platné řazení se zobrazí další možnosti, jak výsledky řadit. Kliknutím na tlačítko „Filtry“ se zobrazí možnosti, jak výsledky omezit na základě některých kritérií. Každý filtr má své ovládací prvky a vysvětlivku. V horní části je také počet výsledků, které filtrům odpovídají. V dolní části pod doménami je pak stránkování. Pomocí šipek lze přecházet na předchozí/další stránku a přeskočit na začátek nebo na konec. Kliknutím na aktuální číslo stránky můžete zadat konkrétní číslo ručně.

Každá doména na stránce znázorňuje celkové (agregované) skóre škodlivosti jako kruhový ukazatel s ikonou nejpravděpodobnějšího typu hrozby. Pod doménovým jménem se nachází skóre všech typů hrozeb a znázornění počtu IP adres v podobě teček. Při najetí kurzorem myši na jednu z domén se na mapě vedle panelu zobrazí přibližné polohy těchto IP adres.

Kliknutím na doménu otevřete detailní pohled, který obsahuje všechna známá data k doméně a další odkazy.

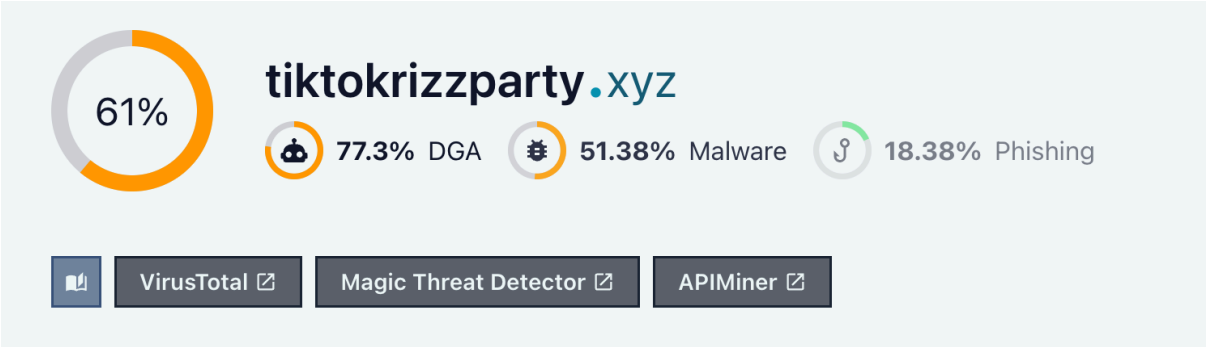


Obrázek W2: Panel přehledu domén

Detail domény

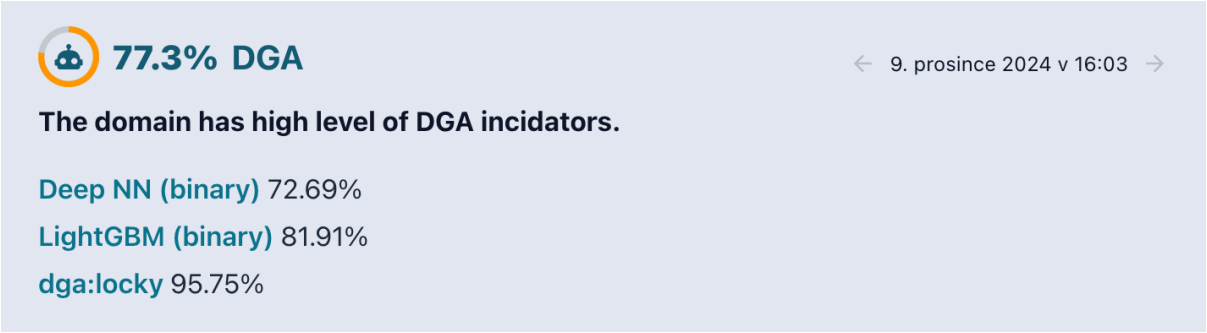
Detail domény se dělí na záhlaví, výsledky klasifikace, IP adresy a časovou osu. V záhlaví se nachází přehled pravděpodobností, podobný náhledu na obrázku W3. Po stisknutí tlačítka s obrázkem otevřené knihy jsou uživateli zobrazena časová razítka a zprávy zaznamenané při sběru dat k doméně. Dále jsou k dispozici odkazy na externí služby pro

analýzu doménových jmen (např. na službu VirusTotal), přičemž jednotlivé odkazy lze přidávat, odebírat a měnit v nastavení (viz dále). Výchozí nastavení je bez odkazů.



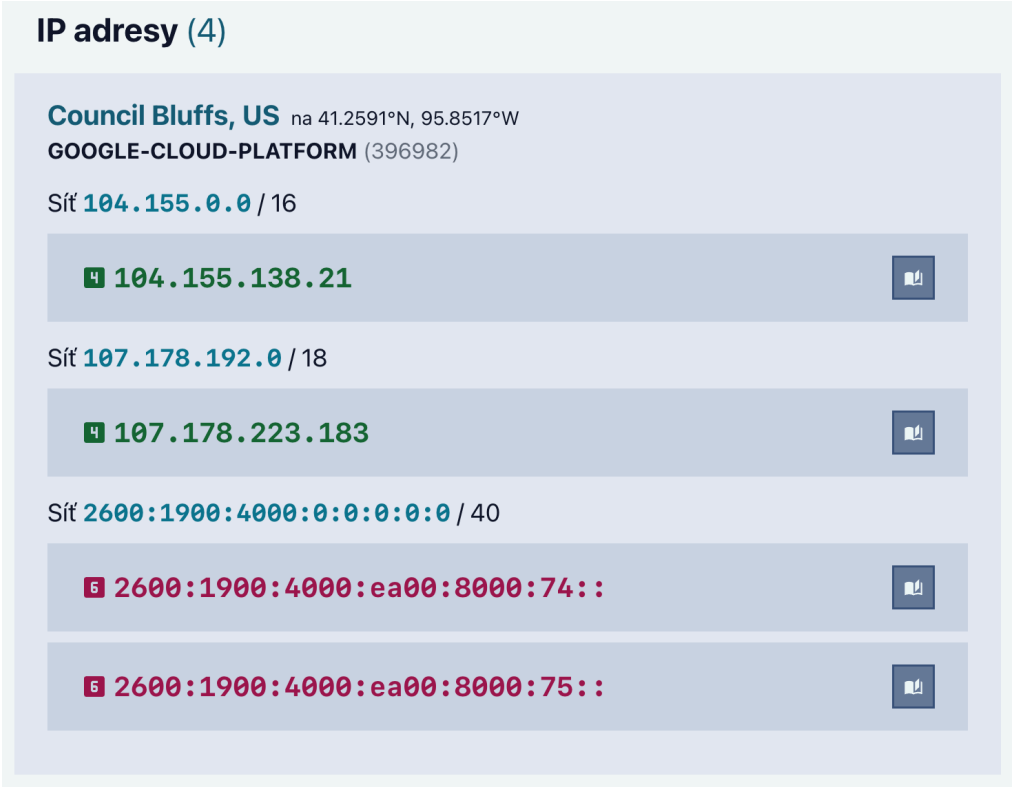
Obrázek W3: Záhlaví detailu domény

V části s výsledky klasifikace se nachází několik karet podle jednotlivých typů hrozeb. Příklad je na Obrázku W4. Zobrazují pravděpodobnost dané hrozby a popis ohrožení vycházející z klasifikace. Níže se nachází výsledky jednotlivých klasifikátorů, které se na zhodnocení podílely. Pokud existují i starší výsledky pro daný typ hrozby, můžete mezi nimi přepínat šipkami podél data a času v pravém horním rohu karty.



Obrázek W4: Karta reportu o typu hrozby

V části s IP adresami (obrázek W5) jsou jednotlivé záznamy seskupené podle podsítí, které jsou samy seskupené na základě autonomních systémů a polohy. Adresy IPv6 jsou červené, zatímco IPv4 používá barvu zelenou. Každá adresa má také k dispozici záznamy o výsledcích sběru dat konkrétních pro IP adresu. Pokud jsou k IP adrese známé incidenty (Offenses) ze SIEM systému QRadar, zobrazí se také tlačítko s možností jejich náhledu a s odkazy na dané záznamy přímo v QRadar konzole.



Obrázek W5: Přehled IP adres domény

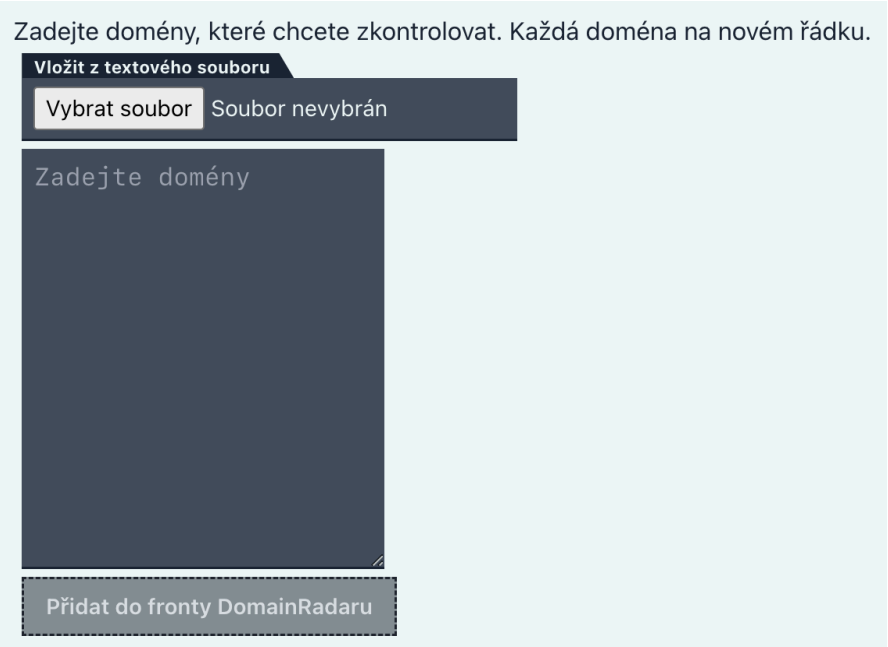
V sekci časové osy najdete datumy a časy, kdy byla doména poprvé systémem zachycena a kdy došlo k poslední aktualizaci dat a výsledků.

Předfiltrované domény

Stránka předfiltrovaných domén ukazuje domény zachycené vstupním filtrovacím modulem systému. Jsou rozdělené podle filtrů, které je zachytily a datum a čas, kdy k tomu došlo.

Kontrola vlastních domén

Na stránce kontroly vlastních domén (Obrázek W6) můžete ručně přidat doménová jména do fronty systému. Zadejte doménová jména na jednotlivé řádky ve vstupním poli a nebo otevřete textový soubor obsahující takto zadaná jména, ze kterého si je aplikace sama načte. Kliknutím na tlačítko přidání do fronty se tento seznam domén odešle ke zpracování.



Obrázek W6: Vstup vlastní kontroly domén

Nastavení

Na stránce nastavení můžete upravit chování webového rozhraní, předfiltrů, a také upravovat konfigurace jednotlivých komponent systému. Obsahuje tyto možnosti:

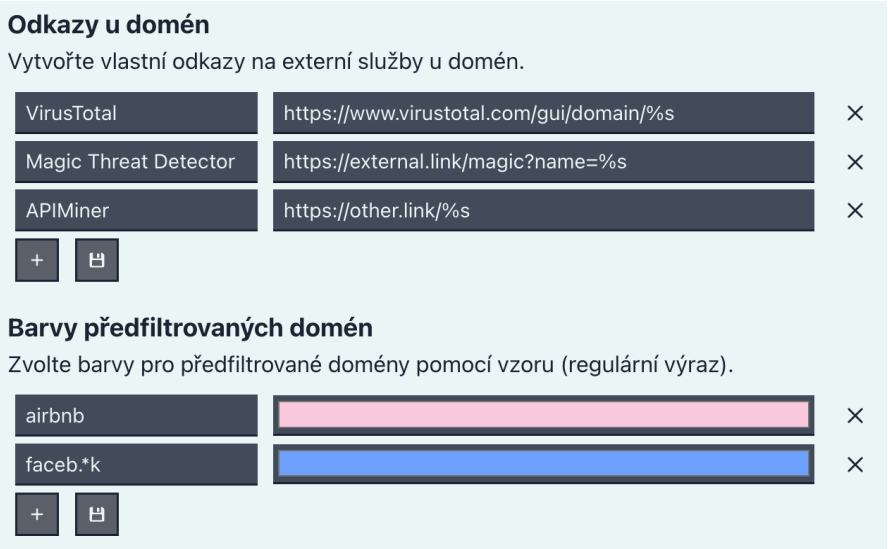
- nastavení jazyka,
- ovládání vzhledu (světlého a tmavého režimu),
- konfiguraci odkazů na externí služby u domén,
- konfiguraci barevného zvýrazňování předfiltrovaných domén,
- vytváření a správu vlastních vstupních filtrů,
- editor konfigurací ostatních komponent.

Výběr jazyka lze provádět i kdekoliv jinde pomocí přepínače v záhlaví aplikace. Vzhled se ve výchozím stavu řídí preferencí z operačního systému. Může se tak měnit dynamicky.

Konfigurace odkazů a barev

Kliknutím na tlačítko „+“ přidejte nový řádek a vyplňte údaje. Smazat jej můžete pomocí „╳.“ Změny vždy nezapomeňte uložit tlačítkem s disketou.

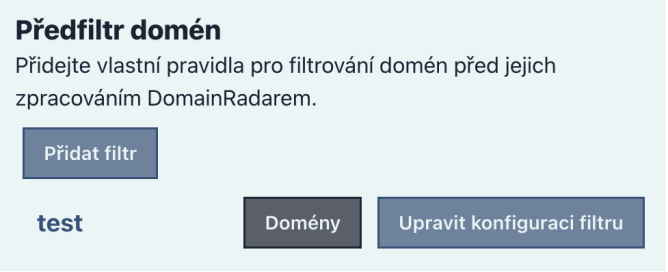
Konfigurace fungují na principu klíčů a hodnot. Pro konfiguraci odkazů je klíčem název zobrazovaný na tlačítku a hodnota je samotný odkaz, kde se za %s dosadí doménové jméno (viz obrázek níže). U barev předfiltrovaných domén je klíč regulární výraz, podle kterého se doménová jména vyberou a hodnota je zvolená barva. Příklad je na Obrázku W7.



Obrázek W7: Konfigurace odkazů a barev předfiltrovaných domén

Vlastní vstupní filtry

Vstupních filtrů lze vytvořit vícero. každý má název, popis, akci, která se provede s doménami a lze jej zapnout nebo vypnout bez nutnosti úplného smazání. Přehled všech vlastních filtrů má vlastní sekci v nastavení, jak ukazuje Obrázek W8.



Obrázek W8: Přehled vlastních filtrů

Dialog vytvoření filtru z Obrázku W9 zobrazíte tlačítkem „Přidat filtr“ a po uložení budete na stránce filtru moci zadat domény nebo vzory. Domény lze zadávat ručně po jedné nebo nahrát ze souboru. Změny je potřeba vždy ručně potvrdit tlačítkem „Uložit“ nad seznamem domén. Do této akce jsou všechny změny pouhý náhled. Domény již uložené ve filtru mají neutrální barvu a pokud jsou označeny ke smazání, zobrazí se červeně s proškrtnutím. Nově zadané domény se zobrazují zelené a pokud je odeberete, zmizí ze seznamu. Uložením se tedy červené domény definitivně odeberou a zelené se přidají. Příklad úprav připravených k uložení je na Obrázku W10.

Obrázek W9: Konfigurace vlastního filtru

Obrázek W10: Správa domén filtru

MalwareRadar

MalwareRadar se skládá z konfigurovatelných komponent, které umožňují sestavit systém různým způsobem. Zde jsou popsány parametry a použití jednotlivých hlavních komponent a příklad sestavení do funkčního systému.

FlowReader

FlowReader je nástroj navržený pro čtení a konverzi síťových toků do jednotného formátu. Podporuje různé vstupní formáty a umožňuje výstup dat ve formě binárních dat (Protocol Buffers) nebo JSON, čímž vyhovuje různým případům použití. Je optimalizován pro jednoduchost a efektivitu, což uživatelům umožňuje rychlé zpracování a analýzu toků z různých zdrojů. Jeho klíčové vlastnosti jsou:

- Podpora více vstupních formátů: FlowReader dokáže zpracovávat síťové toky v různých podporovaných formátech, což ho činí univerzálním nástrojem pro různá prostředí.
- Jednotné datové objekty: Bez ohledu na formát vstupních dat FlowReader generuje jednotné datové objekty, což zajišťuje konzistenci a usnadňuje následné zpracování a analýzu.
- Flexibilní výstupní formáty: Uživatelé mohou zvolit výstup v binárním formátu (Protocol Buffers) nebo v JSON, podle potřeb a zamýšleného použití dat.

Flow Reader umožňuje číst vstupy prostřednictvím URI schémat, což poskytuje flexibilitu v tom, jak jsou data poskytována. Podporované vstupy zahrnují:

- stdin://: Čte data přímo ze standardního vstupu.
- file://: Čte data z konkrétního souboru.
- http(s)://: Čte data z konkrétního umístění pomocí protokolu HTTP(S).

URI vstupu může obsahovat parametry, které specifikují další informace o poskytovaných datech:

- format: Formát vstupních dat (např. JSON). Tento parametr umožňuje Flow Readeru správně parsovat vstupy.
- source: Zdroj dat toků. Flow Reader aktuálně podporuje tyto zdroje:
    - Flowmon
    - IPFIXCOL
    - Suricata

Následující příklad ukazuje, jak použít FlowReader ke čtení dat ze souboru a jejich výstupu ve formátu JSON na standardní výstup:

```
.\FlowReader.exe read-input -i
"file://d:/github/fetanol/testdata/flows/flows.ipfixcol.json?format=json&source=i
pfixcol" -o "stdout:///?format=json"
```

V tomto příkazu:

-i: Specifikuje vstupní URI, kde data pocházejí ze souboru flows.ipfixcol.json ve formátu JSON se zdrojem ipfixcol.

-o: Specifikuje výstupní URI. Data jsou poslána na standardní výstup ve formátu JSON.

FlowReader generuje toky ve standardizovaném formátu. Obsah výstupního toku závisí na vstupních datech, ale výstup obsahuje vždy relevantní informace ve strukturované podobě.


```
{
  "flowKey": {
    "version": "IPv4",
    "protocol": "TCP",
    "sourceAddress": {
      "version": "IPv4",
      "address": "wKhvIA=="
    },
    "sourcePort": 62117,
    "destinationAddress": {
      "version": "IPv4",
      "address": "mMcToQ=="
    },
    "destinationPort": 443
  },
  "flowData": {
    "flowType": "BIDIRECTIONAL",
    "timeStart": "2024-04-08T07:36:46.052420Z",
    "timeDuration": "75.030138s",
    "applicationTag": "SSL",
    "sentPackets": 14,
    "sentOctets": "1568",
    "recvPackets": 39,
    "recvOctets": "49667",
    "tlsFlow": {
      "issuerCommonName": "DigiCert SHA2 Secure Server CA",
      "subjectCommonName": "*.vo.msecnd.net",
      "subjectOrganisationName": "Microsoft Corporation",
      "serverNameIndication": "az667904.vo.msecnd.net",
```

```
      "serverVersion": "TLS1_2",
      "certificateValidityFrom": "2024-01-29T23:00:00Z",
      "certificateValidityTo": "2025-01-30T22:59:59Z"
    }
  }
}
```

ContextCollector

ContextCollector je komponenta určená k výpočtu temporálních kontextů na základě IPFIX reprezentace síťové komunikace. Modul shromažďuje data o aktivitách konkrétních hostitelů během definovaných časových oken a vytváří přehled zahrnující:

- IP spojení,
- Překlady doménových jmen,
- Parametry TLS handshake.

Tento kontext poskytuje komplexní informace o síťových aktivitách hostitele a umožňuje efektivní monitorování i detekci podezřelého chování. Ukázková struktura kontextu je následující:

```
{
  "key": "192.168.111.19",
  "start": "2023-03-10T11:32:00Z",
  "duration": "120s",
  "connections": [ ... ],
  "resolvedDomains": [ ... ],
  "tlsHandshakes": [ ... ]
}
```

Jednotlivé položky mají následující význam:

- key: IP adresa monitorovaného hostitele.
- start: Čas zahájení časového okna.
- duration: Doba trvání časového okna.
- connections: Seznam záznamů o síťových připojeních. Každý záznam představuje souhrnné informace o síťových tocích k vzdálenému bodu (IP adresa a port). Obsahuje údaje o příchozích i odchozích tocích, včetně počtu paketů a bajtů.

```
{
  "remoteHostAddress": "192.168.111.1",
  "remotePort": 53,
  "flows": 121,
  "packetsSent": 121,
```

```
  "octetsSent": "7894",
  "packetsRecv": 121,
  "octetsRecv": "20600"
}
```


- resolvedDomains: Seznam záznamů o přeložených doménách. Obsahuje informace o DNS dotazech a odpovědích.


```
{
  "domainServer": "192.168.111.1",
  "queryString": "googlemail.l.google.com",
  "responseData": "142.251.36.133"
}
```


- tlsHandshakes: Seznam záznamů o TLS handshake. Zahrnuje detailní informace získané z TLS handshake.


```
{
  "remoteHostAddress": "142.251.36.110",
  "remotePort": 443,
  "serverNameIndication": "ogs.google.com",
  "ja3Fingerprint": "24C42D3E77F06F2A8F47F4EBBBFDFAF4",
  "version": "Tls13",
  "certificateValidFrom": "1970-01-01T00:00:00.0000000+00:00",
  "certificateValidTo": "1970-01-01T00:00:00.0000000+00:00"
}
```


Komponentu lze spustit ve dvou režimech:

Zpracování souborů nebo standardního vstupu (batch mode): Příkaz read-input umožňuje načíst data ze souboru nebo stdin a zpracovat je do požadovaného formátu.

Parametry:

-i, --input: URI cesty/umístění vstupu. (Povinné)

-o, --output: URI cesty/umístění výstupu. (Povinné)

--config: Cesta ke konfiguračnímu souboru pro specifikaci parametrů agregace. (Volitelné)

Příklad:

```
ContextCollector.exe read-input -i "file:///path/to/input.json?format=json"
--config ContextCollector.custom.json -o
"file:///path/to/output.json?format=json"
```

Režim serveru (real-time mode): Příkaz run-server spouští GRPC server pro zpracování dat v reálném čase.

Parametry:

-p, --port: Port, na kterém server naslouchá. (Povinné)

-o, --output: URI cesty/umístění výstupu. (Povinné)

--config: Cesta ke konfiguračnímu souboru. (Volitelné)

Příklad:

```
ContextCollector.exe run-server -p 50051 -o
"file:///path/to/output.json?format=json" --config ContextCollector.custom.json
```

Konfigurační soubor slouží k nastavení parametrů tvorby kontextu. Konfigurační soubor (např. ContextCollector.custom.json) umožňuje definovat:

- Časová okna pro agregaci dat.
- Monitorované IP adresy.
- Další parametry kontextové analýzy.

ContextCollector poskytuje detailní a strukturovaný přehled o aktivitách monitorovaných zařízení v síti. Díky podpoře zpracování dat ze souborů nebo v reálném čase je flexibilní a snadno integrovatelný do různých monitorovacích řešení.

MalwareDetector

MalwareDetector je komponenta určená k identifikaci malwaru na klientských zařízeních pomocí analýzy síťové komunikace. Nástroj využívá indikátory kompromitace (IoC) k detekci podezřelé aktivity a umožňuje flexibilní nasazení díky možnosti práce s daty v reálném čase i dávkovým zpracováním. Mezi hlavní funkce patří:

- Monitorování síťové komunikace: Sleduje a analyzuje síťový provoz za účelem detekce podezřelých aktivit.

- Detekce založená na IoC: Identifikuje přítomnost malwaru pomocí známých indikátorů kompromitace, jako jsou IP adresy, domény, URL, fingerprinty JA3 nebo pole SNI v TLS komunikaci.
- Integrace s externí analýzou malwaru: Malware modely jsou vytvořeny na základě IoC získaných externím analytickým nástrojem.

V současné době jsou podporované tyto IoC:

- IP adresy: Detekuje podezřelé adresy spojené s malwarem.
- Doménová jména: Identifikuje škodlivé domény.
- URL: Sleduje a analyzuje podezřelé URL.
- TLS JA3 fingerprinty: Umožňuje detekci na základě fingerprintů TLS komunikace.
- TLS SNI: Analyzuje hodnoty SNI v TLS přenosech pro odhalení podezřelých aktivit.

Malware Detector využívá fuzzy množiny pro reprezentaci každého typu IoC. Tyto fuzzy množiny umožňují flexibilní porovnávání a skórování podle síťové komunikace.

Komponenta je spustitelná z příkazového řádku a má dva režimy činnost a to učení modelu malware a jeho použití pro detekci.

Učení modelu

Pro vytvoření malware modelů postupujte podle následujících kroků:

1. Organizace reportů: Vytvořte složky pro každý typ malwaru a umístěte tam související reporty (např. overview.json, behavioral.pcapng).
2. Stáhnutí reportů: Použijte skript Triage.Collect-Reports.ps1 k automatizaci procesu.
3. Extrahování IoC: Pomocí skriptu Triage.Export-Iocs.ps1 získejte IoC z reportů.
4. Použijte příkaz learn-profile k vytvoření modelů:

```
.\MalwareDetector.exe learn-profile -r /path/to/iocs-folder -o /path/to/output/malware-profiles.json -p *.iocs.json
```

Detekce Malware

Detekční režim analyzuje síťovou komunikaci na základě vytvořených modelů. Detekce může být provedena buďto dávkovým způsobem (pro offline použití a experimenty) nebo v režimu serveru (pro online použití)

Dávkovým zpracováním vstupu (scan-input). Příkaz pro dávkové zpracování:

```
.\MalwareDetector.exe scan-input -m /path/to/malware-profiles.json -i file://path/to/input/data -o file://path/to/output/report.json?format=json -t 1.0
```

Režimem serveru (run-server) pro zpracování v reálném čase. Příkaz pro spuštění jako GRPC server:

```
.\MalwareDetector.exe run-server -m /path/to/malware-profiles.json -p 50051 -o file://path/to/output/report.json?format=json
```

Komponenta podporuje různé formáty vstupní a výstupních dat:

Vstupy:

- stdin://: Standardní vstup.
- file://PATH: Souborová cesta.

Výstupy:

- stdout://: Standardní výstup.
- file://PATH: Souborová cesta.

Přidáním parametru ?format= lze specifikovat formát dat (např. JSON, CSV).

Vytvoření procesní pipeline pro analýzu a detekci malwaru

Tento postup popisuje procesní pipeline složenou z nástrojů FlowReader, ContextCollector a MalwareDetector. Pipeline umožňuje číst IPFIX JSON záznamy, analyzovat síťový kontext a detekovat malware na základě kontextuální analýzy. Jednotlivé nástroje jsou propojeny standardním vstupem a výstupem (Standard I/O), což zajišťuje plynulý tok dat mezi kroky.

Pipeline zahrnuje tři hlavní kroky:

FlowReader: Čtení a normalizace síťových dat.

ContextCollector: Výpočet síťového kontextu pro jednotlivé hostitele.

MalwareDetector: Analýza kontextu a detekce malwaru.

Postup vytvoření pipeline je následující:

1. **Čtení záznamů pomocí FlowReader**
   Příkaz spustí instanci FlowReader, která Čte síťové toky (např. Flowmon IPFIX) z definovaného zdroje a převádí je do jednotného formátu.

```
..\Source\FlowReader\bin\Debug\net8.0\FlowReader.exe read-input -i "${inUri}?format=json&source=flowmon" -o "stdout://"
```

2. **Výpočet kontextu pomocí ContextCollector**

   ContextCollector agreguje síťová data na základě koncových stanic a vypočítává jejich kontext.

```
..\Source\ContextCollector\bin\Debug\net8.0\ContextCollector.exe read-input
-i "stdin://" --config ContextCollector.custom.json -o "stdout://"
```

3. **Detekce malwaru pomocí MalwareDetector**
   Zde probíhá samotná analýza kontextuálních dat a detekce malwaru pomocí předdefinovaných profilů.

```
..\Source\MalwareDetector\bin\Debug\net8.0\MalwareDetector.exe scan-input
-m Malware\Models\triage.mal.zip -t 0.5 -i "stdin://" -o
"${outUri}?format=markdown"
```

Příklad spuštění pipeline

Celý příkaz pro spuštění pipeline je následující:

```
..\Source\FlowReader\bin\Debug\net8.0\FlowReader.exe read-input -i
"${inUri}?format=json&source=flowmon" -o "stdout://" |
..\Source\ContextCollector\bin\Debug\net8.0\ContextCollector.exe read-input -i
"stdin://" --config ContextCollector.custom.json -o "stdout://" |
..\Source\MalwareDetector\bin\Debug\net8.0\MalwareDetector.exe scan-input -m
Malware\Models\triage.mal.zip -t 0.5 -i "stdin://" -o "${outUri}?format=markdown"
```

Jednotlivé komponety jsou propojeny pomocí standardního vstupu a výstupu. Toto je nejjednodušší způsob jak sestavit celou procesní pipeline. Nevýhodou je, že všechny komponenty běží na jedné stanici a jsou na sobě závislé. Selhání komponenty uzavře její výstup a způsobí tak ukončení celé procesní pipeline. Robustnější přístup je použití kontejnerů jak je popsáno v části instalace. FlowReader čte data z ${inUri} a posílá je na výstup. ContextCollector přijímá data z FlowReaderu, vypočítává kontext a posílá jej na výstup. MalwareDetector analyzuje kontext z ContextCollectoru a generuje výsledky detekce ve formátu Markdown na ${outUri}. Nahraďte ${inUri} cestou k souboru s IPFIX daty, např. file:///path/to/input.json. Nahraďte ${outUri} cestou pro uložení výsledků, např. file:///path/to/output.md.

Instalační příručka

V následujících sekcích bude vysvětleno jak nainstalovat a nakonfigurovat oba moduly tak, aby byly použitelné k detekci hrozeb v dané počítačové síti.

DomainRadar

Na základě požadavků aplikačních garanta projektu byl DomainRadar navržen tak, aby byl schopen fungovat ve dvou režimech:

- **Jako plnohodnotný modul systému IBM QRadar**, propojený přes QRadar API. V tomto režimu je schopen dohledávat související bezpečnostní incidenty (*Offenses*), které souvisí s detekovanými hrozbami a pro jednotlivé domény také nabízí informaci o souvisejících událostech (*Events*), síťových tocích (*Flows*) a přímý proklik do QRadar Console na jednotlivé incidenty.
- **Jako samostatný nástroj** pro detekci maligních doménových jmen bez integrace s IBM QRadar, přičemž funkcionalita je omezena pouze na klasifikaci doménových jmen a nejsou využity informace o Offenses z IBM QRadar.

Požadavky

Instalace řešení DomainRadar předpokládá následující požadavky:

- **IBM QRadar** – Jde o nezbytnou prerekvizitu pro nasazení v prvním ze dvou zmíněných režimů (modul pro IBM QRadar). Aby DomainRadar mohl fungovat jako modul pro systém IBM QRadar a klasifikované domény zařazovat k existujícím incidentům (Offenses), je potřeba mít funkční instanci IBM QRadar a přístup na QRadar API. Modul DomainRadar je navržen tak, aby fungoval jak s plnohodnotnou verzí IBM QRadar SIEM, tak i s bezplatnou alternativou IBM QRadar Community Edition (CE). Hardwarové požadavky jsou následující:
  - Pro bezplatnou verzi QRadar CE[25] je potřeba minimálně 24 GB RAM, minimálně 4 jádra CPU (optimálně 6 jader) a diskový prostor o minimální velikosti 250 GB.
  - Požadavky na komerční verzi QRadar SIEM se liší na základě konkrétní varianty řešení a jsou k dispozici[26] na webu IBM.
- **ELK** – Primárním zdrojem doménových jmen je úložiště ELK (Elasticsearch, Logstash, Kibana), kde jsou ukládána hlášení ve formátu Syslog o spatřených doménových jménech. V rámci projektu FETA byla implementována podpora pro zprávy systému Suricata IDS. DomainRadar tedy ve výchozí konfiguraci vyžaduje přístup k běžící instanci ELK, odkud načítá zprávy a extrahuje z nich doménová jména ke klasifikaci. Pro podporu pro další typy vstupů (či vstup domén mimo ELK) je

[25] https://www.ibm.com/community/101/qradar/ce/
[26] https://ibm.com/docs/en/qsip/7.5?topic=installations-prerequisites-installing-qradar-your-hardware

nutné vytvořit nový vstupní modul odvozený od bázové třídy **BaseSource.py** (viz popis subsystému Loader & Pre-filter).

- **Dedikované či virtuální servery pro DomainRadar** – zde záleží na konkrétních výkonových požadavcích dané sítě, ve které je DomainRadar nasazen. Celé řešení může fungovat buď na jediném stroji, nebo distribuovaně s jednotlivými subsystémy na různých zařízeních.

V případě distribuovaného nasazení musí být také splněny mj. následující požadavky:

  - Ze všech strojů musí být dostupné instance systému Apache Kafka.
  - Ze stroje, na kterém běží subsystém Loader & Pre-filter, musí být dostupné také:
    - vstupní zdroj doménových jmen (např. úložiště ELK se hlášeními o DNS zprávách ze systému Suricata IDS),
    - instance databázového systému PostgreSQL.
  - Ze strojů, na kterých běží uživatelské rozhraní a platforma Kafka Connect, musí být dostupné také instance databázového systému PostgreSQL.
  - Ze strojů, na kterých běží kolektory, musí být dostupný internet.
  - Pro plnohodnotné využití nástroje DomainRadar jako modulu pro IBM QRadar je nutné, aby patřičný kolektor měl přístup ke QRadar API.
  - Pozn.: v rámci pilotního provozu bylo nasazení realizováno na třech virtuálních serverech (domain-radar, domain-radar2 a scanner), jejichž specifikace je blíže popsána v sekci **Pilotní provoz na síti CESNET**. V této konfiguraci byl systém schopen zpracovávat průměrně cca 2 200 vstupních doménových jmen za sekundu (před filtrací).

Postup instalace ukázkového prostředí (na jednom stroji)

Minimální HW požadavky na systém: 16 GB RAM, 4 jádra CPU architektury x86_64, 64 GB diskového úložiště.

Doporučené HW požadavky na systém: 32 GB RAM, 8 jader CPU architektury x86_64, 256 GB diskového úložiště.

Podporovaný OS a prostředí:

- operační systém GNU/Linux v distribuci Debian 12.2.0 s jádrem verze 6.1,
- platforma Docker Engine ve verzi 27.0.3 (vč. systému BuildKit),
- nástroj Docker Compose ve verzi 2.29.7.

Pro všechny dílčí části nástroje DomainRadar jsou připraveny soubory typu Containerfile (Dockerfile) pro automatizované sestavení kontejnerů. Dále je připraven soubor typu Compose pro základní souhrnnou správu kontejnerů. Kompatibilita je zaručena pouze s platformou Docker a Docker Compose ve výše uvedených verzích, pro použití s jinými platformami pro správu a orchestraci kontejnerů může být nutné provést úpravy.

Před instalací rozbalte archiv s dosaženým výsledkem do adresáře, který bude dále označen jako `/`. Přejděte do adresáře `/DomainRadar/setup`. Zde je připraven skript **setup.sh**, který připraví prostředí pro spuštění nástroje DomainRadar.

Pozor: Počáteční část skriptu tvoří řada popsaných konfiguračních položek, které je vhodné před spuštěním skriptu upravit.

Součástí konfigurace je také řada interních hesel, která zabezpečují soukromé klíče komponent pipeline a přístup do databáze – pro běžný provoz je není nutné znát, jsou užitečná při vývoji a ladění. Hesla je možné explicitně nastavit (obdobně jako ostatní položky konfigurace). Pokud nastavena nejsou, skript vygeneruje náhodné heslo. Všechna hesla budou uložena v souboru used_passwords.

Skript:

1. Vytvoří zálohu adresáře `infra`, který obsahuje vzor pro přípravu prostředí. (Je tedy možné později skript spustit znovu např. pro ustavení prostředí s jinou konfigurací.)
2. Vygeneruje (chybějící) hesla.
3. Ověří, zda jsou všechny konfigurační položky vyplněny.
4. Upraví konfigurační soubory nástroje DomainRadar v adresáři `infra`.
5. Vytvoří lokální certifikační autoritu a sadu certifikátů, kterými se ověřují klienti při komunikaci se servery Apache Kafka.
6. Sestaví obrazy kontejnerů všech služeb.

Pozor: Skript je interaktivní. Pokud detekuje, že byl dříve spuštěn, nebo pokud detekuje dříve vytvořený soubor used_passwords, ptá se uživatele, zda má dřívější obsah přepsat. Pro vynucení přepisu konfigurace bez potvrzení je možné použít parametr `-y`.

Pozor: Nepřerušujte skript během jeho činnosti. Mohlo by dojít k nekonzistentnímu stavu. V případě přerušení skriptu začněte znovu rozbalením původního archivu.

Po dokončení skriptu je možné **přejít do adresáře `/DomainRadar/infra`** a zde následující sekvencí příkazů inicializovat server Apache Kafka a následně spustit celý nástroj DomainRadar:

```
$ docker compose up -d kafka1 postgres
$ docker compose up --build initializer
$ docker compose up -d
```

Správné spuštění nástroje DomainRadar je možné ověřit návštěvou webového rozhraní na adrese http://localhost:31003/ nebo rozhraní pro správu serveru Kafka na adrese http://localhost:31000/ (ve výchozím nastavení). Přihlašovací údaje do obou rozhraní jsou konfiguračními položkami skriptu setup.sh.

Následně je možné DomainRadar spouštět a vypínat pomocí:

```
$ docker compose up -d  # Spuštění
$ docker compose down   # Vypnutí
```

V ukázkovém prostředí jsou (po provedení počátečního nastavení skriptem setup.sh) k dispozici také následující konfigurační soubory jednotlivých komponent systému:

- `/DomainRadar/infra/client_properties`: konfigurační soubory kolektorů a komponenty Data Merger.
- `/DomainRadar/infra/envs`: soubory s proměnnými prostředí, které nastavují konfiguraci serveru Apache Kafka, prostředí Apache Flink (pro komponentu Data Merger), komponenty Loader & Pre-filter a webového rozhraní DomainRadar.
- `/DomainRadar/infra/.env`: soubor s parametry pro dodaný soubor Compose.
- `/DomainRadar/infra/initializer/prepare_topics.sh`: skript, který slouží k nastavení jednotlivých témat služby Kafka. Pro změnu nastavení po prvním spuštění je nutné v souboru compose.base.yml upravit proměnné prostředí služby initializer:

  `UPDATE_EXISTING_TOPICS=1` nebo `UPDATE_PARTITIONING=1`

  a následně sestavit obraz služby a spustit ji:

  ```
  docker compose up --build initializer
  ```

  Pozor, počet oddílů (partitions) jednotlivých témat je možné pouze zvyšovat, ne snižovat. Nedoporučujeme počet oddílů po prvním spuštění měnit.
- `/DomainRadar/infra/kafka_connect/properties`: konfigurační soubory konektorů služby Kafka Connect.
- `/DomainRadar/infra/postgres/postgres.conf`: konfigurační soubor databázového systému PostgreSQL.

Správa konfigurace

Ukázková konfigurace komponenty Loader & Pre-filter je při inicializaci automaticky vložena do služby Kafka.

Konfigurace kolektorů je uložena ve statických konfiguračních souborech (viz výše) a pro její úpravu je nutné dotčené služby restartovat. Součástí výstupu je také experimentální správce konfigurace, který umožňuje měnit konfiguraci kolektorů za běhu např. prostřednictvím webového rozhraní. Pro spuštění správce konfigurace je nutné na hostitelském stroji spustit v pozadí skript, který zajišťuje správu Compose služeb, a poté spustit Compose službu config-manager (v adresáři `/DomainRadar/infra`):

```
$ python config-manager-daemon.py &
$ docker compose up -d config-manager
```

Pro dlouhodobý provoz by bylo vhodné pro spouštění skriptu config-manager-daemon.py vytvořit například systémovou službu v systemd. Compose služba config-manager je definována s jiným profilem, je proto nutné ji spouštět a vypínat samostatně. Pro spuštění skriptu musí být na hostitelském stroji dostupný Python ve verzi alespoň 3.6.

MalwareRadar

MalwareRadar byl navržen jako samostaný nástroj, který je možné integrovat se systémem QRadar pomocí zasílání zpráv (alertů, logů) ve formátu JSON. Pro podporu zpracování těchto zpráv v nástroji QRadar byl vytvořen vstupní parser. MalwareRadar je závislý na vstupních datech, které poskytují nástroje pro monitorování provozu, například Suricata.

Instalace je možná v rámci jednoho systému, které poskytuje Microsoft.NET SDK 8. Nicméně preferovaná instalace je v prostředí Docker, ve kterém se pomocí Docker Compose provede automatický překlad, instalace a konfigurace.

Postup instalace v prostředí Docker

Hardwarové požadavky závisí na konfiguraci systém a velikosti monitorované lokální sítě, respektive na množství informací o síťovém provozu v čase. Pro většinu scénářů jsou hardwarové požadavky následující:

Minimální HW požadavky na systém: 16 GB RAM, 4 jádra CPU architektury x86_64, 64 GB diskového úložiště.

Doporučené HW požadavky na systém: 32 GB RAM, 8 jader CPU architektury x86_64, 256 GB diskového úložiště.

Níže je popsáno, jak zprovoznit aplikaci definovanou v daném Docker Compose souboru. Aplikace obsahuje tři propojené služby: fetanol-reader-service, fetanol-collector-service a fetanol-detector-service. Tyto služby spolupracují na zpracování dat z různých zdrojů a jsou nakonfigurovány pro běh v propojených kontejnerech. Postup spuštění systému:

1. Příprava prostředí
   Ujistěte se, že máte na svém systému nainstalovaný:
   Docker (verze 20.10 nebo vyšší).
   Docker Compose (verze 1.29 nebo vyšší).
2. Sestavení a spuštění aplikace
   Ve složce Deploy/Docker Spusťte příkaz pro sestavení všech kontejnerů aplikace:
   ```
   docker-compose build
   ```
3. Spuštění aplikace
   Po úspěšném sestavení spusťte všechny služby:
   ```
   docker-compose up
   ```

Použití s nástrojem Suricata

Nástroj je navržen tak, aby spolupracoval s Suricata, což je pokročilý systém pro monitorování a detekci síťového provozu. Integrace zajišťuje získávání relevantních dat o síťovém provozu, která jsou následně zpracovávána a analyzována. Níže je uvedena potřebná konfigurace a postup integrace.

Suricata generuje výstupy ve formátu EVE JSON, které obsahují klíčové informace o síťové komunikaci. Tato konfigurace je nastavena v souboru suricata.yaml, sekce outputs:

```
outputs:
  - eve-log:
      enabled: yes
      filetype: regular
      filename: eve.json
      pcap-file: false
      types:
        - http:
            extended: yes
        - dns:
            enabled: yes
            requests: no
        - tls:
            extended: yes
            custom: [subject, issuer, sni, version, not_before, not_after, ja3,
ja3s]
        # bi-directional flows
        - flow
        # uni-directional flows
        - netflow
```

Tato konfigurace specifikuje jako typ výstupu: eve-log (JSON), výstupní soubor je eve.json. Dále jsou uvedeny volby pro získání rozšířených informací o spojení:

- HTTP: Rozšířený záznam HTTP přenosů.
- DNS: Informace o DNS provozu (bez ukládání požadavků).
- TLS: Rozšířený záznam šifrované komunikace (včetně metadat, jako jsou subject, issuer, sni, ja3).
- Flows: Obousměrné a jednosměrné toky síťového provozu (flow, netflow).

Tato konfigurace umožňuje podrobné sledování provozu, včetně protokolů HTTP, DNS, TLS a síťových toků.

Pro integraci s nástrojem MalwareRadar je výstup eve.json zpracováván pomocí FluentBit[27], což je flexibilní nástroj pro sběr a odesílání logů. FluentBit monitoruje soubor eve.json a zpracovává nové záznamy. Nové záznamy jsou automaticky odesílány do FlowReaderu, který je součástí zpracovací pipeline.

[27] https://fluentbit.io/

FluentBit umožňuje efektivní zpracování velkého objemu logů. Lze přizpůsobit filtry, aby byly odesílány pouze relevantní záznamy (např. jen protokol TLS nebo konkrétní IP adresy).

```
[INPUT]
    Name tail
    Path /path/to/eve.json
    Parser json
    Tag suricata
[OUTPUT]
    Name tcp
    Match suricata
    Host flowreader-service
    Port 7811
```

Pro přímé zasílání dat bez použití FluentBit lze využít volbu unix_stream v konfiguraci Suricata. Tímto způsobem Suricata odesílá data přímo prostřednictvím TCP soketu do FlowReaderu:

```
outputs:
  - eve-log:
      enabled: yes
      filetype: unix_stream
      filename: /var/run/suricata.sock
```

FluentBit lze také nakonfigurovat tak, aby rozkládal zátěž (load balancing) při odesílání dat na více cílů. To je užitečné v prostředích s vysokým objemem síťových dat, kde je třeba distribuovat zatížení mezi více instancemi příjemců (např. více služeb MalwareRadar).

FluentBit podporuje load balancing prostřednictvím round-robin distribuce: Data jsou rozdělena mezi všechny definované cíle. Vzhledem k tomu, že pro správnou funkci MalwareRadar je nutné, aby komunikace jedné koncové stanice byla celá zpracována jednou instancí MalwareRadar je nutné zajistit správné směrování informací s spojeních na příslušné instance. Toho lze dosáhnout například pomocí hash-based směrování, kdy hash je počítán ze zdrojové adresy odesílatele. Tato funkce není ve FluentBit přímo podporována, ale je možné použít Lua skript, který toto zajistí:

```
function route(tag, timestamp, record)
    local ip = record["src_ip"]
    local hash = 0

    for i = 1, #ip do
```

```
        hash = hash + string.byte(ip, i)
    end

    local instance = hash % 2
    if instance == 0 then
        return “suricata_1”, timestamp, record
    else
        return “suricata_2”, timestamp, record
    end
end
```

Odpovídající konfigurace FluentBit pak bude následující:

```
[INPUT]
    Name tail
    Path /path/to/eve.json
    Parser json
    Tag suricata
[FILTER]
    Name lua
    Match suricata
    Script /path/to/hash_router.lua
    Call route
[OUTPUT]
    Name tcp
    Match suricata_1
    Host flowreader-instance1
    Port 7811
[OUTPUT]
    Name tcp
    Match suricata_2
    Host flowreader-instance2
    Port 7811
```

Publikace

Publikace související s vytvořeným výsledkem V2.

[1] HRANICKÝ Radek, HORÁK Adam, POLIŠENSKÝ Jan, JEŘÁBEK Kamil a RYŠAVÝ Ondřej. Unmasking the Phishermen: Phishing Domain Detection with Machine Learning and Multi-Source Intelligence. *Proceedings of IEEE/IFIP Network Operations and Management Symposium (CNSM) 2024*. Soul, Korejská republika, IEEE, 2024, ISBN 979-8-3503-2794-6.

[2] HRANICKÝ Radek, HORÁK Adam, POLIŠENSKÝ Jan, ONDRYÁŠ Ondřej, JEŘÁBEK Kamil a RYŠAVÝ Ondřej. Spotting the Hook: Leveraging Domain Data for Advanced Phishing Detection. *Proceedings of the 20th IEEE/IFIP International Conference on Network and Service Management (CNSM)*. Praha, Česká republika, IEEE, 2024, ISBN 978-3-903176-66-9.

[3] MATOUŠEK Petr, RYŠAVÝ Ondřej and BURGETOVÁ Ivana. Experience Report: Using JA4+ Fingerprints for Malware Detection in Encrypted Traffic. In: Proceedings of 20th International Conference on Network and Service Management. Prague, 2024, pp. 1-5.

[4] BURGETOVÁ Ivana, MATOUŠEK Petr and RYŠAVÝ Ondřej. Towards Identification of Network Applications in Encrypted Traffic. In: The Proceedings of the 8th Cyber Security in Networking Conference (CSNet 2024). IEEE Explore. Paris, 2024, pp. 1-10.

[5] HRANICKÝ Radek, ONDRYÁŠ Ondřej, HORÁK Adam, POUČ Petr, JEŘÁBEK Kamil, EBERT Tomáš, POLIŠENSKÝ Jan. A Multi-Dimensional DNS Domain Intelligence Dataset for Cybersecurity Research. Data in Brief, ISSN 2352-3409 *(v recenzním řízení, odesláno: prosinec 2024)*