Reputační systém IP adres s využitím dat honeypotů

Bc. Ester Rei

Diplomová práce
2023

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav informatiky a umělé inteligence

Akademický rok: 2022/2023

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Ester Rei**
Osobní číslo:	**A21452**
Studijní program:	**N0613A140022 Informační technologie**
Specializace:	**Kybernetická bezpečnost**
Forma studia:	**Prezenční**
Téma práce:	**Reputační systém IP adres s využitím dat honeypotů**
Téma práce anglicky:	**IP Address Reputation System Using Honeypot Data**

Zásady pro vypracování

1. Charakterizujte jak se vyvíjí kybernetické útoky v posledních 5-ti letech.
2. Stanovte bezpečnostní nástroje pro včasnou detekci útočníků.
3. Specifikujte možné zdroje dat pro reputační databázi IP adres.
4. Navrhněte aplikaci pro sběr informací z OSINT zdroju pro Vaši reputační databázi IP adres.
5. Navržené řešení implementujte a ověřte jeho funkčnost v testovacím prostředí.

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam doporučené literatury:

1. SPITZNER, Lance. Honeypots tracking hackers. Boston: Addison-Wesley, 2003, xxvi, 452 s. : il. ISBN 0-321-10895-7.
2. BEJTLICH, Richard. The practice of network security monitoring: understanding incident detection and response. San Francisco, CA: No starch press, 2013, xxx, 341 stran : tabulky, grafy ; 24 cm. ISBN 9781593275099.
3. ROGER A. GRIMES. Hacking the Hacker. Wiley, 2017. ISBN 9781119396215.
4. ANSON, Steve. Applied incident response. Indianapolis: Wiley, 2020, 1 online resource (464 pages). ISBN 1119560284. Dostupné také z: [https://proxy.k.utb.cz/login?url=https://onlinelibrary.wiley.com/doi/book/10.1002/9781119560302](https://proxy.k.utb.cz/login?"https://proxy.k.utb.cz/login?url=https://onlinelibrary.wiley.com/doi/book/10.1002/9781119560302")
5. HUANG, Scott Ch, David MACCALLUM a Dingzhu DU, ed. Network security. New York: Springer, 2010, xi, 280 s. Dostupné z: doi:9780387738215
6. STALLINGS, William a Lawrie BROWN. Computer security: principles and practice. Fourth edition. Chennai: Pearson, [2020], 800 atrn. ISBN 978-93-534-3886-9.
7. SANDERS, Chris a Jason McC SMITH, BIANCO, David J., ed. Applied network security monitoring: collection, detection, and analysis. Amterdam: Syngress, Elsevier, [2014], xxiv, 472 s. ISBN 9780124172081.
8. ORIYANO, Sean-Philip. Penetration testing essentials. Hoboken, NJ: Sybex, 2017, 1 online resource. ISBN 9781119235330. Dostupné také z: [https://proxy.k.utb.cz/login?url=https://onlinelibrary.wiley.com/doi/book/10.1002/9781119419358](https://proxy.k.utb.cz/login?"https://proxy.k.utb.cz/login?url=https://onlinelibrary.wiley.com/doi/book/10.1002/9781119419358")

Vedoucí diplomové práce: **Ing. David Malaník, Ph.D.**
Ústav informatiky a umělé inteligence

Datum zadání diplomové práce: **2. prosince 2022**
Termín odevzdání diplomové práce: **26. května 2023**

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky
L.S.

doc. Ing. Jiří Vojtěšek, Ph.D. v.r.
děkan

prof. Mgr. Roman Jašek, Ph.D., DBA v.r.
ředitel ústavu

Ve Zlíně dne 7. prosince 2022

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomové práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové práce bude uložen v příruční knihovně Fakulty aplikované informatiky. Univerzity Tomáše Bati ve Zlíně;
- byl/a jsem seznámen/a s tím, že na moji diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má Univerzita Tomáše Bati ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování diplomové práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně, dne .....

podpis studenta

ABSTRAKT

Diplomová práce se specializuje na tvorbu reputační databáze pro IP adresy s využitím dat honeypotů a dat z volně dostupných zdrojů. Cílem práce je propojení zmíněných zdrojů dat do jednoho funkčního celku a vytvoření jednotné aplikace umožňující vyhledávání v těchto datech. Teoretická část práce se zaměřuje na popis vývoje kybernetických útoků v posledních pěti letech a na popis bezpečnostních nástrojů detekující útočníky. Praktická část obsahuje výběr vhodných zdrojů pro tvorbu reputační databáze, kroky vedoucí k návrhu aplikace, praktickou implementaci a v neposlední řadě ověření její funkčnosti.

Klíčová slova: kybernetické útoky, honeypoty, monitoring, OSINT zdroje, IP adresa, reputační databáze

ABSTRACT

The focus of the master thesis is on the creation of a reputation database for IP addresses by using the collected data by honeypots and also data from freely accessible sources. The main aim of the thesis is to interconnect this data from mentioned sources and create functional application which enables searching in these data. The theoretical part contains description of the five-year evolution of the cyber attacks and characterization of the threat detection tools. The practical part consists of the selection of the valuable sources for a creation of the reputation database, also contains design of an application, its implementation and finally validation of the application functionality.

Keywords: cyber attacks, honeypots, monitoring, OSINT, IP address, reputation database

Poděkování

Ráda bych poděkovala svému vedoucímu práce, panu Ing. Malaníkovi, Ph.D. za pomoc a cenné rady.

OBSAH

ÚVOD

Neustálý vývoj kybernetických útoků nutí aktualizovat a vylepšovat zabezpečení infrastruktury a chod celé počítačové sítě. Pro zefektivnění zabezpečení, je potřebné aktualizovat několik dílčích částí.

Mezi tyto části patří nastavení zabezpečení infrastruktury na základě aktuálních informací, nastavení mechanismů pro sběr, zpracovávání, vhodné vyhodnocování a vizualizaci dat. Dále sem patří mechanismy pro detekci, jakými například jsou intrusion detection system a honeypoty.

Do vyjmenovaných kroků zabezpečení infrastruktury a chodu celé sítě také patří ověřování reputace IP adres, které do sítě přistupovaly. Tyto informace lze získat v průběhu monitoringu sítě, během detekování realizovaného útoku, sběrem dat pomocí honeypotů nebo za využití volně dostupných zdrojů.

Cílem práce je zefektivnění využití sesbíraných dat, a to pomocí tvorby reputační databáze pro IP adresy, které primárně pochází z dat honeypotů.

Teoretická část diplomové práce se bude v první kapitole zaobírat vývojem kybernetických útoků v posledních pěti letech. V druhé kapitole, *Nástroje pro detekci útočníků*, pak budou popsány možné způsoby detekce útočníků. V této kapitole pak například budou přiblíženy honeypoty nebo detekční systémy.

Praktická část práce bude obsahovat možné zdroje dat pro tvorbu reputační databáze IP adres. Součástí této části bude i kapitola *Tvorba reputační databáze pro IP adresy*, která bude obsahovat dílčí kroky návrhu vedoucí k tvorbě finální aplikace, a která bude také obsahovat popis implementace návrhu. Poslední kapitola, *Ověření funkčnosti aplikace*, bude obsahovat snímky z testování finální aplikace.

I. TEORETICKÁ ČÁST

1 VÝVOJ KYBERNETICKÝCH ÚTOKŮ V POSLEDNÍCH 5-TI LETECH

Tato kapitola obsahuje informace o vývoji kybernetických útoků za posledních pět let. Jsou popsány kybernetické útoky z oblasti malwaru, phishingu a sociálního inženýrství, zranitelností webových stránek a webových aplikací, využití zranitelností pomocí předem připravených nástrojů, DDoS útoků a spamů. Informace byly čerpány z veřejných zdrojů Agentury Evropské unie pro kybernetickou bezpečnost, dále jen ENISA. Agentura slouží jako centrum pro síťovou a informační bezpečnost pro území Evropské unie. Nicméně v oblasti sběru dat nezaměřuje pouze na oblast Evropy, ale taky na oblast celého světa.

Mezi její činnosti lze zařadit tvorbu doporučení pro informační bezpečnost, pomoc členským státům při nasazování evropských předpisů, zlepšování bezpečnosti evropské kritické informační infrastruktury a sítí. Její činnost také zahrnuje sledování aktuálních hrozeb a nastalých incidentů během sledovaného období. Na základě těchto výsledků jsou vytvořeny statistiky významných hrozeb, častých útoků, bezpečnostních incidentů a doporučení pro obranu pro jednotlivé sledované časové intervaly.

Jedním z důvodů k výběru statistik od ENISy byl ucelený a průběžný přehled kybernetických útoků v jednotlivých obdobích. Druhým, také významným, důvodem pro výběr bylo technologické zaměření popisu útoků.

Dle informací o často realizovaných útocích během sledovaných období, pak bude provedeno srovnání vývoje kybernetických útoků za posledních pět let.

Vývoj kybernetických útoků byl rozdělen na několik částí, které jsou shrnuty v následujících podkapitolách:

- **Kybernetické útoky pomocí škodlivého kódu**, kde budou například popsány malwarové a ransomwarové útoky, činnosti a tvorba botnetů.
- Podkapitola s názvem **Kybernetické útoky mířící na dostupnost služeb**, která obsahuje popis vývoje spamových kampaní a DDoS útoků.
- Podkapitola **Kybernetické útoky využívající zranitelností** obsahuje popis útoků na webové stránky, webové aplikace a využití zranitelností pomocí předem připravených nástrojů, tzv. exploit kitů.
- Poslední podkapitola **Kybernetické útoky zneužívající lidského faktoru** popisuje vývoj phishingu a sociálního inženýrství.

1.1 Kybernetické útoky pomocí škodlivého kódu

Podkapitola se zaměřuje na kybernetické útoky pomocí malwaru a ransomwaru a na činnost a tvorbu botnetů.

Výskyt malwaru v roce **2017** byl převažující vůči jiným typům útoků. Ve vývoji malwaru vůči předešlému roku byl také zaznamenán jeho nárůst.[1]

Z pohledu struktury malwaru byla celkově zaznamenána menší četnost unikátnosti jednotlivých vzorků a převládající trend využívání již existujícího malwaru s úpravami. Převládající formou malwaru byl skript.[1]

Nicméně ve směru složitosti, sofistikovanosti a v široké škále různých a dodatečných funkcionalit byl zaznamenán nárůst. V případě dodatečných funkcionalit lze zmínit přidané funkce pro sběr přihlašovacích údajů a využití exploitů u ransomwaru.[1]

Primárním cílem útoků na začátku roku byly operační systémy Windows, na které byl zejména šířeny trojské koně a značné množství ransomwaru. Celkově v oblasti operačních systémů byl zjištěn nárůst malwaru, a to pro operační systémy MacOS a Linux.[1]

V oblasti mobilních telefonů byl taktéž nárůst, ale pouze v oblasti distribuce ransomwaru, který cílil na různé typy zařízení nebo na různé operační systémy.[1]

Byla zaznamenána větší míra použití tzv. click-less infekcí a fileless útoků.

Dobrým příkladem je ransomware WannaCry, který se sířil tímto způsobem. Po zneužití zranitelnosti a získání přihlašovacích údajů hrubou silou, infikoval oběti přes RDP protokol. Stejný způsob infekce jako u WannaCry byl použit i u dvou třetin ransomwarů v roce 2017.[1]

Fileless útoky byly v oblibě kvůli menší možnosti detekce a složitějšímu provedení forenzní analýzy nebo menší šance pro zrušení probíhajícího útoku. K útokům byly použity již zabudované nástroje pro administraci systému, pomocí kterých byly v paměti spouštěny škodlivé skripty. V některých případech žádné nástroje použity nebyly a skripty byly spouštěny přímo v paměti počítače. Mezi tyto nástroje patřily Power Shell, PSExec pro vzdálené připojení a WMI pro synchronizaci systému.[1]

Kromě výše zmíněných metod byl malware, přesněji červy, šířen i přes počítačové sítě. Pro zamaskování stop po kyber špionáži byly používány takzvané stěrače nebo mazače, wipers.

Významný vývoj nové generace mazačů byl objeven během Shamoonových útoků, kde malware Shamoon smazal boot sektor na disku, avšak před smazáním boot sektoru byl smazán celý obsah disku, a to právě pomocí mazače se jménem *Trojan.Filerase*[15].[1]

Vyjma sítě nebo konkrétních cílů byl malware ještě šířen pomocí tzv. PUPs přes prohlížeče bez bezpečnostních prvků. Ve většině případů šlo o prohlížeče od tvůrců adwaru,

které nahradili výchozí, legitimní, prohlížeč svým vlastním a tím také sebou dovlekly potenciálně nechtěný program.[1]

Také vzrostl zájem o útoky na hardware a firmware ze strany útočníků, ale zůstal v přijatelných intervalech. V případě útoků za využití konkrétního hardwaru by byl nutný výběr hodně specifického cíle.[1]

Místo toho byly spíše realizovány hybridní útoky s větší šancí úspěšnosti. Hybridní útoky zahrnují kombinaci více typů útoků dohromady. Například pro odlákání pozornosti byl proveden DDoS útok a pro získání informací byl použit útok pro obejití bezpečnostních mechanismů. Skutečným příkladem byl malware BrickerBot, který pro získání přihlašovacích údajů použil útok hrubou silou a následně po získání přístupu do systému z příslušných disků smazal všechna data.[1]

Pro znemožnění blokace útoků z různých domén útočníci s oblibou využívali algoritmy pro generování doménových jmen. Šlo o algoritmy pro rychlou změnu jména domény pro znesnadnění blokace probíhajícího útoku.[1]

Častým útokem byl také útok zaměřený na dodavatelský řetězec, supply chain attack.

Tento útok byl většinou realizován umístěním škodlivého kódu do legitimních aplikací pro administrátory s možností jejich stažení přímo na stránkách vývojáře.[1]

Ransomware měl stále se zvyšující tendenci, a to kvůli jednoduchosti získání zisku za pomocí větší míry automatizace útoku než u jiných typů malwaru. Zvyšující se výskyt také zapříčinil největší podíl obětí (60%) vůči jiným malwarům.[1]

Ransomwarové útoky byly také více zaměřené na konkrétní cíle, jako jsou finanční organizace nebo podniky pracující s financemi. Vyjma finančních organizací byly cílem i serverové technologie, jako MongoDB databáze, Hadoop, Cassandra nebo MySQL servery, kde byl obsah databáze nahrazen ransomwarem. Vyjma útoků na výše zmíněné služby a technologie byly také do budoucna předpovězeny možné ransomwarové útoky na zdravotnická zařízení.[1]

Ulehčením pro získání výdělku byla také služba Ransomware-as-a-service. Příkladem je ransomware Philadelphia od skupiny The Rainmaker Labs.[1]

Kromě ransomwaru byly na popředí i spamové kampaně nebo útoky využívající zranitelnosti cílového systému. Spamové kampaně byly realizovány ze strany botnetů. Mezi nejaktivnější botnety patřil botnet Necrus, který se podílel na zasílání spamů v Asii a Evropě.[1]

Botnety se současně podíleli i na možnosti zveřejnění reklamy ze strany uživatelů. Po zaplacení však reklama zveřejněna nebyla a platba také nebyla navrácena. Činnost botnetů byla také identifikována na stránkách Twitteru, kde bylo nalezeno více

než 350 tisíc falešných účtů, které byly součástí botnetu. Vyjma vyjmenovaných činností botnetů však převládala realizace pulzních DDoS útoků. Z toho nejvýkonnějším botnetem byl botnet Leet, který na začátku roku 2017 realizoval DDoS útok o síle 650 Gb za sekundu.[1]

Do činnosti botnetů byly také zahrnuty IoT zařízení nebo virtuální počítače v cloudovém prostředí. Příkladem je botnet IoTroop, který byl vytvořen infikací IoT zařízení. Současně byl zaznamenán nárůst v počtu botnetů vytvořených na základě zveřejněného kódu botnetu Mirai. Tyto botnety se také zaměřovali na IoT zařízení. V jiných případech byly botnety vytvořeny i za použití malwaru jako Ursnif nebo DELoader, přes phishingové kampaně.[1]

Nejvíce zasaženými zeměmi po útocích botnetů byly Čína, Indie, Rusko, Brazílie, Írán nebo USA.[1]

Pro rok **2018**, malware zůstal stejně často zmiňovanou hrozbou, jak tomu bylo v předešlém roce. Vůči předešlému roku, ale nebyl výskyt výrazných změn ve vývoji malwaru. Převládal trend mírné úpravy postupů a chování z důvodu maximalizace zisku.[2]

V oblasti úniku dat až 30% úniků dat bylo způsobené malwarem. Oblastem zájmu malwaru byly finanční instituce a maloobchody. Nejvíce zasaženými operačními systémy byly Windows (79%), Linux (18%) a nejméně zasaženými byly operační systémy Mac (3%).[2]

Dle Verizon DBIR nejčastějšími přenašeči malwaru byly soubory typu: .js (37,2%), .vbs (20,8%), Windows executable (14,8%), MS Office (14,4%), .pdf (3,3%) other (7,0%). Nejčastějším malwarem v tomto období byly: malware na těžbu kryptoměn, RAT, spyware, botnet, ransomware, banking trojan, backdoor, trojský kůň nebo červ.[2]

Místo ransomwaru se začal využívat cryptojacking, který je spolehlivější a jednodušší na získání většího výdělku než ransomware. Kód pro těžbu kryptoměn byl ve většině případů distribuován přes existující malware, botnety, kompromitované webové stránky, spam, sociální sítě, mobilní aplikace, exploity, reklamy nebo vyměnitelná média.[2]

Výměna ransomwaru za cryptojacking neznamenala nečinnost ransomwaru. Za 85% útoků ve zdravotnictví bylo možné hledat ransomware. Stejná míra byla i u vzdělávacích institucí. Více než 60% z celkové počtu ransomwarových útoků bylo zaměřených na průmyslové kontrolní systémy a více než 60% ze všech detekovaných ransomwarů bylo doručeno přes mail. Nejčastějším důvodem úspěšnosti ransomwaru byly zranitelnosti nebo chybějící záplaty nebo aktualizace na operačních systémech.[2]

Vůči předešlému roku se pokračovalo v používání fileless útoků, které vykazovali vysokou efektivitu. 77% úspěšných útoků použila právě tuto techniku. Tyto útoky zahrnovaly umístění škodlivého kódu do souborů jako Microsoft Office, PDF, nebo

powerShell, VBS, JavaScript.[2]

Pokračujícím cílem útoků zůstal RDP protokol. Naopak kleslo používání exploitů z důvodu zvýšené složitosti dovlečení malwaru. Místo exploitů jako přenašeče malwaru, byl použit adware.[2]

V oblasti finančních institucí klesla míra útoků pomocí trojského koně. Důsledkem bylo vylepšení bezpečnostních kontrol, zákonné omezení a jiné zájmové oblasti útočníků. V některých případech, ale trojské koně realizovali krádež přihlašovacích údajů kryptoměnových peněženek nebo některé přidali funkcionality pro těžbu kryptoměn. Významnými zástupci trojských koní mířící na finance byly Zeus, Emotet, Trickbot, Ursnif. Avšak v oblasti rozšířených funkcionalit vzrostl počet trojských koní poskytující vzdálený přístup.[2]

Malware pro mobilní telefony vůči ostatním typům malwaru neklesl, ale mírně vzrostl a rozšířil své funkcionality. Většinou byl obsažen v aplikacích od třetích stran. Převládajícími aplikacemi obsahující malwaru byly aplikace pro životní styl nebo hudbu. Hlavní cílem mobilních malwarů byla krádež přihlašovacích údajů, instalace trojských koní za účelem vzdáleného přístupu nebo zneužití SIM karty.[2]

Objevili se první malwary zaměřené na poškození kritické infrastruktury. Takovým malwarem byl Triton, který se zaměřil na systémy hlídající bezpečnost průmyslových procesů. Dále se zvýšil počet útoků na IoT zařízení a byl objeven první malware určený pro UEFI rozhraní.[2]

Současně pokračovalo využití volně dostupného malwaru, ke kterému útočníci využili zpřístupněné nástroje, jakými byly Metasploit, Mimikatz nebo PowerShell.[2]

V oblasti botnetů byl zaznamenán vývoj v komunikaci command&control serverů: extrémní zvýšení až o 300% v používání šifrované komunikace, zvýšené využití legitimních komunikačních kanálů.[2]

Vůči původnímu roku se v oblasti botnetů se útočníci spíše zaměřili na udržování stavu současných botnetů. Byly provedeny updaty, instalovány záplaty a dodány nové funkcionality pro lepší fungování. Mezi nové funkcionality lze zařadit: odstranění již dovlečeného malwaru nebo botnetů zaměřených na těžbu kryptoměn, zjišťování nových sítí, exploitace koncových bodů, obfuskace zdroje útoku a DNS využívající blockchain pro komunikaci s C&C servery. Některé botnety se zaměřovali pouze na IP kamery a nastavovaly si rozšíření pro útok na ADB, který je součástí Android zařízení a slouží k ladění a konfiguraci.[2]

Již existující bontet Necrus byl využit pro vydírání uživatelů sledující stránky pro dospělé a také rozeslal vyděračské skamy na bankovní domény. Zajímavým botnetem byl Torii IoT botnet, který se zaměřil na různé počítačové architektury, jako jsou x86_64, x86 nebo ARM. Mezi jeho činnost bylo možné zařadit načítání a realizaci příkazů

na dálku, exfiltrace, mechanismus pro šifrování komunikace na více vrstvách.[2]

Během roku byl objeven další botnet pro operační systém Linux, Chalubo, který zkoušel získat SSH přístup pomocí útoku hrubou silou. Po získání přístupu se zaměřil na záplavové DDoS útoky. Naopak některé botnety se zaměřili na sběr přihlašovacích údajů, identit nebo peněz. Stejně jako v případě ransomwaru i u botnetů byla nabízena služba pr pronájem botnetu.[2]

V dalším sledovaném období,**2019** až **2020**, úroveň malwaru zůstala na stejné úrovni jak tomu bylo předtím. Nicméně míra útoků mířící na podniky, služby a vzdělávání vzrostla. V případě bankovních útoků, útoky mířily na firemní uživatele s cílem poškodit danou firmu.[11]

Stejně jako v předešlém roce, pokračoval trend ve využívání malwarů pro těžbu kryptoměn. Pro těžbu byl například využit javaScript v prohlížeči nebo již nainstalovaný malware na zařízení oběti. Míra distribuce malwaru pro těžbu kryptoměn, ale zůstala stejná jako v předcházejícím roce. Těžba u obětí způsobovala vysokou spotřebu elektřiny a snížení produktivity zaměstnanců. Obdobně pokračovalo používání RDP protokolu jako prostředku útoku. [9]

V oblasti malwaru bylo v první polovině roku **2019** zaregistrováno 265% navýšení útoků pomocí fileless malwaru. Tento typ malwaru měl mnohem větší pravděpodobnost útoku něž jiné typy. Většina útoků byla realizována pomocí skriptů, spuštěním v paměti nebo využitím vestavěných nástrojů.[11]

Pro šíření malwaru byly použity webové a emailové protokoly, které obsahovali malware v souborech typu *docx*. Pro větší možnost šíření do interní sítě se, ale dál používali exploity nebo útok hrubou sílou. Také se objevili tzv. multistage útoky. Na mobilních zařízeních byl objeven velký počet předem nainstalovaných spywarů a adwarů.[11]

Až o polovinu vzrostla míra malwaru určeného pro krádež osobních dat přes bankovní konto. Pro získání dat byl použit phishingový útok s přihlašovací stránkou banky nebo použití falešné aplikace dané banky. Zajímavou novinkou byla detekce pohybu ze strany malwaru, tím se malware zaktivoval pouze při pohybu. Populárními bankovními malwary pro rok 2019 byly: Asacub, Svpeng, Agent nebo Faketoken.[11]

Značnou oblíbenost si získala služba Malware-as-a-service, kde bylo možné získat nástroje pro útok a pro vzdálenou správu zařízení. Potřebné nástroje zahrnovali část pro načtení, C&C servery a backdoor pro ovládání napadeného zařízení.[11]

Obfuskace kódu nebo používání podobných nástrojů jiných skupin byla častá a tím znesnadňovala určení původce útoku.[11]

Vůči roku 2018 oblast ransomwaru zaznamenala nárůst. Ransomware byl druhým nejčastěji využívaným typem malwaru. Ransomwarové útoky vzrostly s cílem získat větší zisk. Místo většího počtu cílů se útočníci zaměřili na menší množství cílů, ale s větší

možností finančního zisku. Z toho důvodu byly vytvořeny malwary, které se šířili uvnitř sítě a ne internetem.[10]

Ze strany organizací vzrostl počet zájemců o pojištění proti ransomwarovému útoku. Tuto možnost například využili vládní organizace, zdravotnické organizace nebo dokonce celá města. Nevýhodou mohl být případ znalosti částky výkupného ze strany útočníka, a tím i nárůst pravděpodobnosti útoku.[10]

Důležité je také zmínit, že ransomwarový útok utrpěla více než polovina zdravotnických organizací a cílem byly také poskytovatelé v oblasti cloudových služeb, poskytovatelé služeb pro podniky, vzdělávací instituce a průmysl.[10]

Například ransomware LockerGoga poškodil kontrolní zařízení ve výrobních závodech. V jiných případech v oblasti zdravotnictví po útoku některé zdravotnické organizace a nemocnice byly nuceny omezit, pozastavit nebo ukončit svůj provoz. V některých případech došlo i k úniku citlivých dat pacientů.[10]

Vůči roku 2018 se zvýšil provoz botnetů o 71,5%. Hlavními oblastmi pro umístění botnetů se stali: Rusko, USA, Nizozemsko, Čína a Francie. Největšími cíli botnetů byly finanční služby a jejich zákazníci. 97,4% útoků bylo realizováno ze strany botnetů běžících na Linuxu. Významnými zástupci byly Emotet, Trickbot a DanaBot.[7]

Mezi hlavní činnosti botnetů patřily: krádeže a opětovné použití přihlašovacích údajů, malwarové nebo spamové kampaně, těžba kryptoměn, DDoS. V některých případech byl použit botnet, který se sebou dovlekl trojského koně, který pak po získání potřebných informací v systému zanechal ransomware.[7]

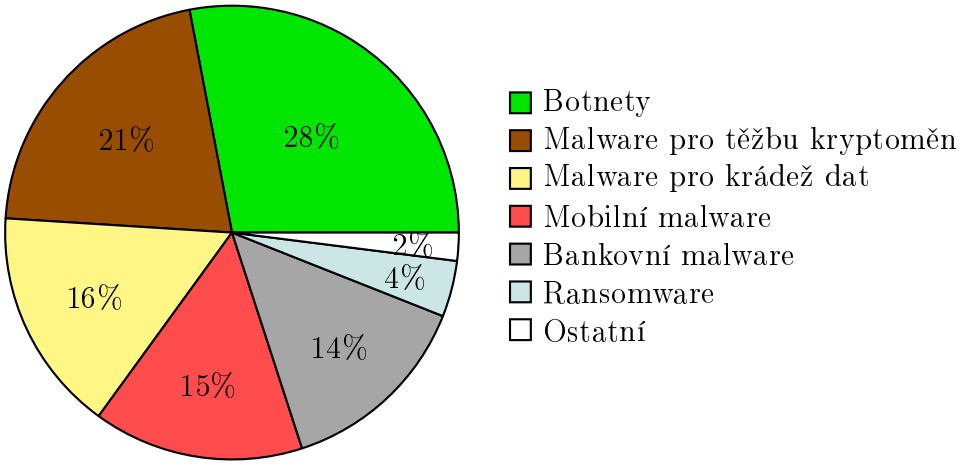
V některých případech byly malwary pro tvorbu botu distribuovány přes emaily a pro řízení C&C serverů byly nejvíce použity RAT.[7]

Stejně jako tomu bylo i v předešlých letech, útočníci využívali algoritmy pro generování doménových jmen, které použili pro zakrytí běžících C&C serverů. Současně pro komunikaci s botnety začali využívat jiné komunikační protokoly jako například point-to-point protokol (P2P), a tím dosáhly větší synchronizace informací mezi boty a menší šanci na odhalení.[7]

Během tohoto období botnet Mirai rozšířil svou činnost. Kompromitoval IoT zařízení pomocí útoku hrubou silou, zkoušením výchozích hesel nebo exploitů. Dále se zaměřil na různé typy architektur.[7]

Zajímavým objevem byl vzorek využívající výchozí hesla pro IoT zařízení. Pomocí těchto údajů se bot přihlásil na zařízení, které úplně vymazal a ponechal tam jen sám sebe. Tento útok byl realizován jako upozornění na možnost zneužití špatně zabezpečených IoT zařízení.[7]

Byl objeven botnet Roboto využívající již zmíněné P2P spojení, tj. boty mezi sebou



Obrázek 1.1 Výskyt malwaru v roce 2020 [12]

interagují a sdílí nashromážděné informace. Tento program obsahoval možnosti reverse shell útoku, sebe-deaktivace, sběru dat o síti a o jiných botech, DDoS útoky, útoky na operační systém nebo spouštění souborů z URL adres.[7]

Dalším botnetem byl botnet Mozi, který se šířil přes telnet pomocí útoku na slabá hesla. Stejně jako botnet Roboto měl několik různých funkcionalit: DDoS útoky, sběr informací, update a spouštění poayloadu z URL adresy, spouštění příkazů.[7]

Novým trendem během roku **2020** v oblasti šíření ransomwaru bylo získávání interních spolupracovníků pro spuštění ransomwaru v cílovém systému. Dále bylo zaznamenáno postupné navyšování výkupného z 15 milionu dolarů v roce 2019 na 30 milionů dolarů v roce 2020. V případě možného nezaplacení byly oběti vydírány zveřejněním citlivých dat.[12]

Nejčastěji detekovanými malwary v roce 2020 byly: botnety (28%), malwary pro těžbu kryptoměn (21%), malwary pro krádež dat (16%), mobilní malware (15%), bankovní malware (14%) a ransomware (4%), viz obrázek 1.1.[12]

Rok 2020 zaznamenal nárůst mobilního malwaru mířícího na bankovnictví. Tento typ malwaru se pokoušel ukrást bankovní spojení spolu s informací o platbě přes falešnou přihlašovací obrazovku a pak realizovat transakce bez vědomí uživatele. Byly také rozeslány falešné SMS zprávy od státních institucí s tematikou covidu-19.[12]

V tomto období byly také realizovány útoky zneužitím zero-day zranitelností na mobilní platformy. Nicméně cílem těchto útoků byly specifické skupiny, jako v případě spywaru Pegasus. Mezi specifické skupiny patřili novináři, aktivisti nebo politici.[12]

Na konci roku 2020 byl zpozorován pokles malwaru, který pokračoval i v roce 2021. Pokles byl pozorován v Evropě a Asii. Důvodem poklesu mohla být pandemie covidu-19 a přesun zaměstnanců domů, kde je menší úroveň detekce útoků než v podnikových sítích. Naopak se útoky ve srovnání s rokem 2019 zvýšili v Severní Americe o až 43%.[12]

Na přelomu let **2020** a **2021** byly objeveny vážné zranitelnosti na některých zařízeních od různých výrobců jako jsou Microsoft, Apple, Google, Adobe, Dell nebo VMware. Také byly objeveny zranitelnosti v softwaru od Applu, které byly útočníky hojně zneužity. Příkladem je malware Silver Sparrow.[12]

Pokles infekce malwarem pokračoval i v roce 2021. Útoky malwarem byly již více zaměřeny na kvalitu a ne na kvantitu jako v předešlých letech.[12]

Vývoj infikovanosti PDF a Microsoft Office souborů byla v roce 2019 na stejné úrovni. V roce 2020 převládala infekce Microsoft Office souborů až o 150% vůči PDF souborům. V roce 2021 naopak nastal pokles obou typů souborů a vzrostla infikovanost spustitelných souborů.[12]

V roce 2021 malware nově cílil i na prostředí běžící na kontejnerech. Byly vytvářeny škodlivé kontejnerové obrazy a objeveny zranitelnosti prostředí Kubernetes. Byl objeven malware, který mířil i na Windows kontejnery. Dalšími možnými vstupy pro bezpečnostní incident byly například špatná konfigurace nebo bezpečnostní problémy přímo v aplikaci, která běžela v kontejneru. Hlavním cílem malwaru bylo vyvlečení se z kontejneru a nakažení dalších aplikací v jiných kontejnerech a získání citlivých informací.[12]

Pro zamezení detekce malwaru byly použity nové nebo méně často používané programovací jazyky. Například jazyky Nim, DLang a Go. Tímto byla zamezena nebo znemožněna detekce po určitý časový interval až do doby vytvoření nových pravidel detekce na základě signatur pro nově objevené vzory.[12]

Dále pak téměř polovina mobilního malwaru byla klasifikována jako adware. Škodlivé aplikace byly většinou nainstalovány přes služby třetích stran nebo přes online fóra. Příkladem může být útok na operační systém Android, kde se škodlivý software vydával za falešný adblocker s tím, že pro svůj běh žádal navýšená oprávnění k systému a následně používal upozornění pro zveřejňování reklam a přesměrovával prohlížeče na stránky s reklamou. Příkladem může být mobilní malware hiddenAds.[12]

Na konci roku **2021** byl opět zaznamenán nárůst v šíření malwaru. Jeho důvodem částečně byl ústup pandemie a návrat zaměstnanců z home office a také nárůst počtu některých druhů malwaru. Nicméně nárůst byl pouze detekován u cryptojackingu a malwaru pro IoT.[12]

Celkově v roce 2021 k nejčastějším typům malwarů patřili: RAT, banking Trojans, info stealers a ransomware. Mezi nejčastější představitele patřili: Agent Tesla, Ursnif, LokiBot, MOUSEISLAND, NanoCore, TrickBot, kde některé představitelé byli aktivní i po dobu několika let.[12]

Jak již bylo napsáno výše, počet malwaru pro IoT zařízení vzrostl. Mezi cílová zařízení patřili zařízení Netgear (DGN), D-Link (HNAP) a Dasan (GPON).[12]

Častým a jednoduchým způsobem distribuce malwaru bylo klonování volně dostup-

ných frameworků s cílem infikovat kohokoli, kdo tyto frameworky použije. Po vložení malwaru do příslušného frameworku byl framework distribuován pod nápadně podobným jménem jak originál. Mezi ty patřili například škodlivé pythonové balíčky, kde některé z nich byly přímo zaměřeny na krádež dat nebo citlivých údajů. Balíčkem zaměřeném na krádež citlivých dat byl Ascil2text, který po vyhledání všech hesel v daném systému odeslal útočníkovi.[12]

Dalším typem souborů pro umístění malwaru a získání přístupu do systému nebo získání citlivých dat byly soubory Microsoft Office. Přesněji umístění malwaru do maker, do VBS souborů. Po implementaci bezpečnostních opatření, blokování maker z internetu, byl zaznamenán pokles tohoto typu malwaru. Na to byla distribuce malwaru přesunuta k souborům zajišťující kompresi dat, jako jsou ZIP, ISO a RAR.[12]

V období let **2020** a **2021** zůstal nadále aktuálním útok na RDP protokol. Z důvodu levnosti a většího zisku vzrostly phishingové útoky a ransomware jako služba zůstala stejně vyhledávanou jako v předešlém roce.[12]

Během let 2021 a 2022 pokračovalo zaměření se na mobilní zařízení. Jak z pohledu distribuce přes google apps, tak z pohledu cílených útoků pouze na konkrétní skupiny jako jsou novináři, politická opozice, atd. Během roku **2022** se malwarové útoky nejvíce zaměřovali na evropské státy.[13]

Objevily se útoky zaměřené na instituce a podniky na Ukrajině. Na začátku roku 2022 byl detekován mazací virus s názvem Wipers, který smazal boot sektor na disku a tím znepřístupnil data i celkový systém. Spolu s ním se objevili i viry podobného typu, které byly taktéž zacíleny na Ukrajinu. Mezi ně patří: WhisperGate nebo IsaacWiper.[12]

Na území EU byl zaznamenán nárůst počtu ransomwarových útoků. Stejně jako v roce 2019 útočníci využívali kódů jiných skupin, buď originál nebo jeho modifikaci.[12]

Nejčastějším inicializačním vektorem útoků vůči minulým letem byl phishing. Dalšími možnostmi byla inicializace útoků přes RDP protokol a nověji i pomocí sociálního inženýrství, mailu nebo interních osob. Každá z metod má určité výhody. Sociální inženýrství cílí na užší oblast než například maily. Metoda přes RDP protokol využívá již znalosti přihlašovacích údajů. Získání přihlašovacích údajů je rychleji detekovatelné, ale v případě jejich získání je pro útočníka mnohem pravděpodobnější, že zůstane neodhalen.[12]

Byla vytvořena veřejná stránka ze strany útočníků s informací o napadnutí nebo nenapadnutí zaměstnanců ransomwarem. Každý zaměstnanec si tento údaj mohl zkontrolovat. Nevýhodou byla možnost zjednodušeného vyhledávání potencionálních obětí, kvůli možné evidenci přístupů na stránku.[12]

Také se objevil nový prostředek pro vymáhání výkupného: upozornění oběti na útok a nabádání ji k odkoupení svých dat dříve, než koupu realizuje konkurence. Dalším

trendem bylo zveřejnění dat oběti bez zveřejnění jména nebo organizace do doby dokud oběť nezaplatí. Při odmítnutí by došlo ke zveřejnění jména.[12]

1.2 Kybernetické útoky mířící na dostupnost služeb

Tato podkapitola se zaměřuje na kybernetické útoky mířící na dostupnost zdrojů, paměti nebo úložiště. Bude popsán vývoj spamových a DDoS útoků.

Spam

Spamové kampaně byly realizovány již od začátku existence internetu, kde primárně sloužili pro šíření reklam. V současnosti spíš jde o obtěžující a nežádoucí zprávy za účelem zahlcení cíle, přenosu malwaru nebo nalákání oběti na škodlivou stránku. Výhodou rozesílání spamů je jejich levné šíření. Na druhou stranu, přenos a zpracování jsou výpočetně náročné.[1]

Vůči předešlým letem jejich míra šíření klesla, ale nadále zůstaly hlavním přenašečem malwaru, který byl zasílán v přílohách nebo přes škodlivé odkazy. Pokles dokazuje i snížená činnost největšího spamového botnetu Necrus vůči roku 2016.[1]

Většina spamu v tomto období byla rozeslána ze strany botnetů (88%). Zprávy většinou nabízely zdravotnické produkty a také rozšířily ransomware Jaff, podobný ransomwaru WannaCry. Napodobeniny mailů známých výrobců sebou v mnoha případech nesly trojského koně skrytého v ZIP souboru. Šíření spamu se přesunulo také do oblasti sociálních sítí. Podobně jak v případě malwarů, se u spamů také zvýšila míra sofistikovanosti pro obcházení spamových filtrů.[1]

Například pro obejití spamových filtrů, útočníci začali rozesílat spamy, které jako přílohu měly heslem chráněný archiv. Tento archiv pak obsahoval JavaScriptem nebo makrem infikovaný soubor typu Microsoft Word nebo Excel. Některé skupiny malwarů dokonce začaly šířit své kopie dál.[1]

Rozesílání počtu spamů rostl lineárně dle velikosti organizace. Z pohledu denní míry obdržení spamových zpráv vůči legitimním emailům, spamy zahrnovali 85% z celku. Celková míra obdržených spamů za celé sledované období byla 55,9%. Největším zdrojem spamů bylo zaznamenáno z území států: Vietnam, USA, Čína a Indie.[1]

Během roku **2018** botnet Necrus zůstal hlavním aktérem v rozesílání spamů, a to 75% z celkových rozeslaných spamů. Jinými aktivními botnety v tomto období byly Gamut a Cutwail.[2]

K šíření spamu přes sociální sítě byla například použita aplikace WhatsApp. Falešné zprávy zahrnovali informace o vymyšlených loteriích, populárních obchodech nabízející zboží nebo v té době aktuální tematiku GDPR, kde byly nabízeny online semináře nebo workshopy. K šíření byly použity falešné účty celebrit, podnikové účty nebo reklamy nabízené na sociální síti. Obsah spamových zpráv se také začal více přizpůsobovat.

Nově text zpráv začal být psán i v jiných jazycích než v angličtině.[2]

Pro snížení možnosti detekce, útočníci začali používat dvě zdrojové adresy. První byla legitimní, veřejně známá a tím i důvěryhodná. Druhá byla původní zdrojová adresa útočníků.[2]

Dalším způsobem pro obejití detekce bylo využití odpovědních formulářů pro předplatné na veřejných webových stránkách. Maily s informacemi o předplatném byly spamovými filtry povoleny a nebyly kontrolovány.[2]

Nejčastěji byly rozesílané spamy: z oblasti zdravotnictví (26,6%), spamy s malwarem (25,7%) a spamy nabízející seznamky (21,4%). 80% rozeslaných spamů pocházelo od spamových gangů, jako například Canadian Pharmacy, Blaze Media Solutions nebo RR Media. Denní průměr rozeslaných spamů zůstal stejný vůči předešlému roku (kolem 85%). Bylo zjištěno, že 40% spamů mělo průměrnou velikost 2kB. Nejpravděpodobnějšími na rozesílání spamů v rámci top level domény byly: .gq, .cf , .loan , .tk , .ml, .ga, .men, .faith, .top a .racing. V tomto období bylo také zveřejněno 80 milionů uživatelských dat ze spamových kampaní.[2]

Na rozdíl od roku 2017 nastala změna ve jménech největších původců spamů. Prvním zdrojem se stala Čína, pak USA a Německo. Naopak umístění spamových botnetů více korespondovali s nejčastějšími zdroji spamu z předešlého roku. Byly jimi Čína, Indie a Vietnam.[2]

V období od července **2019** až do července **2020** byly realizovány spamové kampaně v širším rozsahu a byly více zaměřené na vybrané cíle. Například vybranými cíli byly emailové účty zaměstnanců těžebního průmyslu. Dalším příkladem byly účty v Německu, kam mířilo 10% spamových zpráv.[8]

Spamové kampaně v některých případech kombinovali i phishing. Příkladem byly spamy, které nabízely výhru chytrého zařízení. Podmínkou pro zařazení se do outěže, ale bylo zasálání skenu průkazu totožnosti a fotografie oběti. Jiná spamová kampaň žádala o zaslání fotografie a následně využila emailové účty obětí pro registraci předplatného na televizi nebo na živé vysílání.[8]

Odkazy na phishingové stránky nebo malware se objevily i ve spamech rozeslané na téma probíhající pandemie covidu-19. Častými výrazy v těchto mailech byly covid-19 nebo koronavirus.[8]

Stejně jak v minulých letech, spamy nadále doručovaly malware nebo ransomware. Mezi rozesílaný malware patřil LokiBot, který se přenášel v souboru typu ISO. Přenášenými ransomwary byly například Dharma, Crysis a Ryuk.[8]

Kromě zasílání ransomwaru v příloze, vydírání bylo použito i jinak. Útočníci ve zprávách žádali zaměstnance o zaslání peněz. V případě nesouhlasu hrozilo použití jejich emailové adresy ve spamové kampani, a tím by se jejich adresa dostala na seznam

zakázaných adres.[8]

Ve sledovaném období let **2019** až **2020** byly zaznamenány nové způsoby pro obcházení spamových filtrů. V některých případech, podobně jako v roce 2018, útočníci k rozesílání spamu využili formuláře na webových stránkách. V tomto případě to byly formuláře pro zasílání zpětné vazby zákazníkům. V dalších případech spamové kampaně začali využívat legitimní mailové šablony organizací.[8]

Spamy se také zaměřovali na mobilní telefony, přesněji na rozesílání SMS zpráv s nabídkou peněz nebo s odkazem vedoucím na falešné stránky s možností krádeží citlivých informací.[8]

Během roku **2021** stejně jak v minulém období převládali spamové kampaně. Hlavním prostředkem manipulace bylo šíření informací nebo nabídek spojené s covidem-19. Mnohdy byly nabízeny respirátory nebo vakcíny. Pro jejich objednání bylo potřeba navštívit falešné webové stránky, které obsahovali škodlivý obsah.[12]

Stejně jako v předešlém roce, pokračovalo rozesílání SMS zpráv. V tomto případě se zaměřili na operační systém Android a rozesílaly malware Flubot. Nejvíce zasaženou oblasti v tomto případě byla Evropa.[12]

Během roku **2022** spamové kampaně využili válku mezi Ukrajinou a Ruskem a na základě toho také rozesílaly maily s informacemi o této válce, spolu s přiloženým malwarem. [13]

Prominentním útokem ve všech cílových oblastech nadále zůstaly útoky DDoS. Až 33% organizací bylo obětí DDoS útoků. Nejčastějším cílem útoků byl herní průmysl. Dalšími častými cíli byly finanční a bankovní služby ve střední Evropě a v severských státech.[13]

DDoS

Na konci roku 2016 byly botnetem Mirai pomocí infikovaných IoT zařízení realizovány největší DDoS útoky, až 1Tb za sekundu. V důsledku zrušení několika škodlivých skupin v roce **2017**, celková míra útoků na určitou dobu klesla. Stejně jak u malwaru byl zpozorován nárůst sofistikovanosti útoků. Hlavně v oblasti mobilních telefonů.[1]

Permanentními cíli nadále zůstali datová centra a IoT zařízení. Vzrostly segmentované DDoS útoky v kombinaci s ransomwarem, útoky pomocí odrazu, tzv. reflection attack, a pulzní DDoS útoky, které místo jednoho cíle útočili hned na několik cílů za sebou. Příkladem je pulzní útok trvající několik dní o síle 350 Gb za sekundu.[1]

K útokům byly často využity různé typy DDoS útoků najednou. Například záplavové útoky přes UDP a TCP protokol. V některých případech byly k útoku použity webové stránky nabízející zátěžové testy. Jinými obětmi využitými k útokům byly DNS servery. V tomto období byly také s oblibou realizovány útoky v dávkách, tzv. útok trval pouze určitý časový interval.[1]

Útoky DDoS začaly být využívány jako maskování pro provedení jiného, významnějšího, útoku. Například zakrývali: útok malwarem (50%), únik nebo krádež dat (42%) nebo ztrátu financí (26%).[1]

Nejvíce DDoS útoků během roku 2017 bylo realizováno ze strany Číny (60%). Nejvíce botnetů pak bylo umístěno na území Jižní Koreje. Nejčastějším cílem pak bylo USA, a to více než 90%.[1]

Nejvíce obávanými útoky v roce 2017 byly: ADPoS, DNS water torture attacks, SSL-Based Cyber Attacks, PDoS a IoT botnety. Útoky ADPoS realizovaly útoky pomocí protokolů síťové a aplikační vrstvy. Například pomocí HTTP protokolu nebo používali útoků jako jsou XSS nebo SQL injection. DNS water torture útoky cílili na DNS servery, na které byly záplavově zaslány poškozené požadavky. Útoky založené na protokolu SSL využívali zdlouhavé šifrování a dešifrování zpráv během komunikace. Permanentní DoS útoky způsobovali trvalé škody, po kterých byla nutná přeinstalace nebo výměna hardwaru. Botnety z IoT zařízení pak dokázali realizovat útok o síle až 1Tb za sekundu, a tím si vysloužili, že se staly nejrychleji rostoucí se hrozbou.[1]

Celkově nejčastějšími DDoS byly útoky: UDP fragments, DNS flood a NTP flood. Naopak bylo zaznamenáno, že míra útoků přes protokol HTTP vůči předešlému roku klesla. Útočníci také začali používat tzv. multivektorové útoky. Současně se navýšil zájem o DDoS jako službu, která byla dostupná v průměru za 5 amerických dolarů.[1]

Během roku **2018** byl zaznamenán nárůst útoků DDoS. V tomto období se útočníci nejčastěji zaměřovali na realizaci jednorázových záplavových útoků pomocí protokolů UDP, TCP a ICMP. Útoky byly realizovány v době největšího provozu.[2]

Z pohledu typů útoků byly stejně jak v minulém roce používány jejich kombinace. Byl zaznamenán mírný nárůst útoků v oblasti útoků pomocí odrazu, útoků využívajících protokoly síťové a transportní vrstvy. Také vzrostly útoky z botnetů založených na operačním systému Linux.[2]

Příkladem reflection DDoS útoku byl reflection-amplification DDoS útok na GitHub, který měl sílu 1,35Tb/s. Podobným útokem byl také zasažen americký poskytovatel služeb. V obou případech útoky mířili na výpočetně náročné zpracování požadavku. DDoS útoky byl také zasažen vlakový dopravce v Německu.[2]

V roce 2018 nejčastějšími typy DDoS útoků byly: Memcached (reflected) amplification útoky, multi-target útoky, útoky na zpracování nezpracovatelného nebo poškozeného požadavku (Cache Busting DDoS), PDDoS útoky, útoky používající legitimní šifrovaný provoz těžce rozeznatelný od legitimního. Nejčastějšími cíli byly Čína, USA a Hong Kong.[2]

Odražené, refleted, DDoS útoky pro odraz využívali legitimní služby, pomocí kterých pak bylo možné vytvořit útok až 50-krát silnější než původní. Multi-target útoky

uskutečňovaly útoky na širší IP rozsah. Často používaly SYN flood nebo útoky pomocí protokolů aplikační vrstvy. V případě PDDoS šlo o útoky, které v první řadě z obětí vytvořily botnet a pak realizovaly samotný DDoS útok, který mohl trvat i několik dní.[2]

V období let **2019** až **2020** na výsluní zůstaly útoky pomocí odrazu a posílení. Vzrostly útoky z botnetů, kde nejčastějšími členy byly infikované IoT zařízení. Více než tři čtvrtiny DDoS útoků byly útoky SYN flood. Přitom značná část útoků trvala méně než 10 minut. Nadále však přetrvávali multi-vektorové útoky. Za nárůstem útoků mohla být i pandemie covidu-19.[6]

Útočníci se začali zaměřovat na navzájem různé oblasti, se kterým vzrostla i větší míra průzkumu ze strany útočníků. DDoS jako služba nově začala být propagována i na YouTube a Redditu. Nejvíce botů realizujících DDoS útoky se nacházelo na území Číny (24%), Brazílie (9%) a Íránu (6%).[6]

Nejčastějšími útoky byly: UDP flood, DNS amplification, HTTP flood a TCP-SYN flood. Za oblíbeností UDP protokolu mohly být důvod nenavazování spojení a nepotvrzování přijatých paketů. UDP protokol pak byl nejčastěji používán ve spojení s amplified a reflected DDoS útoky, které nejvíce cílili na strojírenský a herní průmysl. Oblíbenost SYN floodu se mohl přisuzovat k rychlejšímu vyčerpání výpočetních zdrojů oběti. Během sledovaného období byl detekován SYN flood o síle 500-580 milionů paketů za sekundu.[6]

K amplifying DDoS útokům byl využit i WS-Discovery protokol, byl využíván IoT zařízeními pro nalezení uzlů v lokální síti. V oblasti telekomunikací a poskytování služeb se používaly útoky bit-and-piece nebo carpet bombing DDoS. Příkladem je realizace DDoS útoku na router poskytovatele z IP rozsahu samotného poskytovatele.[6]

Stejně tak byl zaznamenán nárůst útoků typu RansomDoS, tj. realizace DDoS útoku, umístěním ransomwaru do systému a žádost výkupného nebo vydírání zranitelného systému pro zaplacení výkupného pro nenapadnutí.[12]

Na základě výše zmíněných byl útok RDDoS považován za nebezpečný, a to z důvodu zaručeného úspěchu, i když útočník neměl dostatečné zdroje pro celou jeho realizaci. Průměrná síla útoků typu RansomDoS dosahovala až 500Gb/s.[12]

V období covidu-19 vzrostla pravděpodobnost útoků na šířku pásma pro komunikační software. Ve stejné době vzrostly DDoS útoky proti RDP a VPN připojením. Také se zvýšili útoky na domácí sítě, které pak vedly k jiným útokům. Například k útoku na DNS záznamy.[12]

Roky **2020** i **2021** zaznamenaly nárůst ve složitosti realizovaných DDoS útoků. Během roku 2021 byl zaznamenán prudký nárůst útoků na oblast vzdělávání, obchodu, finančních služeb, softwaru, technologií a maloobchodu. V některých případech

i o 960% nebo 445%. Nicméně v oblasti rozsahu útoků bylo nejvíce zasaženo zdravotnictví. Také se zvýšily útoky na menší podniky s nižší úrovní zabezpečení. Avšak kritická infrastruktura a veřejné instituce zůstali hlavním cílem.[12]

Zvýšili se útoky na dodavatelský řetězec, tzv. supply chain attack, které byly také nejčastějším inicializačním vektorem. Z toho celkově 58% těchto útoků mířila na získání dat a 16% na získání přístupu k lidem. 20% mířilo na získání dat dodavatelů a 66% na získání jejich zdrojového kódu.[12]

Ve sledovaném období se také zvýšily dlouho trvající útoky. Dle použitých typů tomu bylo podobně jako v předešlém roce. Převládali multi-vektorové útoky (65%). K útokům byly nadále využívány tradiční komunikační protokoly TCP, UDP nebo ICMP. V některých případech se útočníci spoléhaly na využití tzv. vestavěných protokolů jako například: Apple Remote Management Service nebo Web services dynamic Discovery.[12]

DDoS jako služba nadále přetrvávala a nabízela velké množství nástrojů pro realizaci útoku nebo možnost zakoupení útoku na konkrétní cíl. Pro zesílení útoků mohly být využity virtualizované zdroje.[12]

Z důvodu omezených zdrojů, špatné konfigurace nebo nízkého zabezpečení se IoT zařízení často stávaly obětí útoků. Na druhé straně, IoT zařízení byly využitelné jako součást botnetů, které pak realizovali další nebo daný útok. Kritickou oblastí byly IoT zařízení v průmyslu nebo ve zdravotnictví, kde nefunkčnost těchto zařízení mohla způsobit značné škody.[12]

Do DDoS útoků byly zahrnuty i webové aplikace. 43% úniků dat bylo realizováno přes webové aplikace a přibližně 90% webových aplikací bylo cílem útočníků. Dle výzkumů bylo zjištěno, že přibližně polovina firem byla ovlivněna útokem realizovaným na firmu nacházející se v dodavatelském řetězci. Tyto útoky zasahovaly data zasažených podniků, oblast cloudů a dodavatele spojené s obětí. Výhodou pro útočníky bylo i to, že podniky ve většině případů neměly možnost nahlédnout do infrastruktury svých dodavatelů, a tím ani nemohly zajistit, aby se nestaly také obětí možného útoku.[12]

Příkladem supply chain útoku, přesněji na území Číny, bylo vložení backdooru do instalačního balíčku pro software na skenování otisku prstů, který byl používán i státními úřady. Díky tomu útočníci získali vzdálený přístup do chráněných prostředí. Jiným možným supply chain útokem mohly být cloudy využívající šablony třetích stran se špatnou bezpečnostní konfigurací nebo kontejnerové aplikace běžící v cloudu se známou zranitelností.[12]

Zajímavým způsobem supply chain útoku byla tzv. změna ve zdrojovém kódu, která byla provedena pomocí kódu Trojan Source. Šlo o podobný princip jak u útoku na skener otisků prstů v Číně, nicméně jeho způsob distribuce byl jiný. Pomocí Unicode znaků ve zdrojovém kódu byla provedena změna pořadí tokenů. Útok svou úspěšnost zaklá-

dal na tom, že v některých případech vývojáři kopírují kód z online příspěvků do kódu vyvíjeného softwaru. Nakažený kód pak mohl být součástí finální verze softwaru. Jiným známějším supply chain útokem bylo vytvoření zdánlivě legitimních knihoven, které ale obsahovali kód pro těžbu kryptoměn.[12]

Nově útočníci před samotným útokem realizovali průzkum z veřejných zdrojů o použitých technologiích a zranitelnostech potenciálního cíle. Také se začali zaměřovat přímo na pracovníky nebo organizace, které hledají zranitelnosti, a to s cílem tyto informace získat a použít je k útoku co nejdříve.[12]

Nejčastějšími cíli, ale i zdroji DDoS útoků nadále zůstaly Čína a USA. Dalšími častými zdroji i cíli na území EU byly Nizozemsko, Německo a Francie.[12]

Během roku **2022** nadále přetrvávalo používání útoku typu RansomDDoS, UDP a TCP-SYN flood. V některých případech byl RDoS použit pro umístění malwaru do systému oběti. Vzrostla míra výskytu tzv. quadruple-extortion útoků, které rozšiřují oblast vydírání i na obchodní partnery a klienty oběti.[13]

Oblasti útoků přetrvávali, tak jako v minulých letech. Patřily sem bankovnictví, pojišťovnictví, finanční služby, telekomunikace, technologie, zdravotnictví, VoIP služby a vzdělávání.[13]

Útoky byly zejména realizovány na zranitelné systémy a stejně jak v minulém roce: na mobilní a IoT zařízení. Díky službě Cybercrime-as-a-service se zjednodušila realizace útoků, avšak vzrostla jejich složitost a objevili se nové techniky útoku.[13]

Převládali menší útoky, tj. o síle 1 a 3GB/s. Nicméně útoky o síle 10 a 30 GB/s se také objevovaly. Také se zvýšila míra útoků na aplikace a pomocí protokolů naopak poklesly volumetrické útoky.[13]

Vyjma webových aplikací jako možných zdrojů nebo cílů se možnosti rozšířily i na samotná cloudová úložiště nebo cloudová úložiště, na kterých hostovala napadená webová aplikace. Díky vysoce výkonným botnetům vzrostl počet kompromitovaných IoT zařízení. Ke kompromitaci také dopomohly nejčastěji nastavené přihlašovací údaje: login – admin a heslo – 1234. Útoky se také zaměřili na chytrá města nebo automobily a přetrvávaly vůči webovým stránkám bojujícími proti covidu-19.[13]

Významná většina DDoS útoků byla realizována z důvodu Rusko-Ukrajinské války. Cíle i zdroje útoků zahrnovali účastníci z více vrstev a z více států. V některých případech byly státem sponzorované útoky použity ještě před vypuknutím války. Například během července roku 2021 byly z neznámých zdrojů realizovány útoky na bezpečnostní agentury Ruska a Ukrajiny. Během války byly nejvíce zasaženými oblastmi: média, internet, kryptoměny a maloobchod.[13]

Vyděračské DDoS útoky začaly být masivně využívány státem sponzorovanými útočníky a cílili na finanční instituce, poskytovatele internetového připojení, malé a střední

firmy. Během července 2022 byl zaznamenán nárůst DDoS útoků.[13]

Největším zaznamenaným útokem během roku 2022 byl útok proti evropské zákaznické platformě Prolexic. K útoku byly použity zařízení z celého světa. Trval 14 hodin a měl sílu 853,7 GB/s a 659,6 Mb/s.[13]

Častými zdroji útoků byly kompromitované servery, routery nebo koncová zařízení. V některých případech ke kompromitaci byly zneužity i několik let staré zranitelnosti. Největšími cíli a zdroji útoků byly USA a Čína a největší počet C&C serverů byl na území USA, Nizozemska a Německa.[13]

Jiným druhem znepřístupnění v roce 2022 byly i převzetí nebo destrukce síťové nebo telekomunikační infrastruktury, cenzura nebo BGP hijacking.[13]

Převzetí nebo destrukce byly spojeny s válkou na Ukrajině. Ze strany Ruska byl veden nátlak na ukrajinské síťové poskytovatele, aby se vzdali kontroly nad svou infrastrukturou a přesměrovali provoz přes Rusko. Tím by byla umožněna blokace provozu nebo úniku dat. Z pohledu destrukce byla zničena ukrajinská telekomunikační síť.[13]

Na území Ruska od února bylo zablokováno několik tisíc web stránek, včetně Instagramu, Facebooku, Twitteru, BBC News a dalších stránek poskytující nebo umožňující šíření zpráv. Pro zamezení šíření aktuálních zpráv byly rozesílány TCP pakety s příznakem RST ihned po navázání TLS handshaku.[13]

Pro přesměrování síťového provozu byl realizován tzv. BGP hijacking, co znamená převzetí vlastnictví nebo vlády nad určitým IP rozsahem a následné přesměrování provozu. Vyjma přesměrování provozu byl možný i odposlech.[13]

1.3 Kybernetické útoky využívající zranitelností

Tato podkapitola se zaměřuje na kybernetické útoky využívající zranitelností. Například zranitelností systému, softwaru nebo aplikací. Bude popsán vývoj útoků na webové stránky, webové aplikace a vývoj tzv. exploit kitů.

V roce **2017** se útoky na webové stránky nejvíce zaměřovaly na: využití zranitelnosti prohlížeče nebo jeho rozšíření, využití zranitelnosti webového serveru nebo webové služby, drive-by útoky, water-holing útoky, přesměrování, šíření malwaru přes škodlivé URL odkazy a man in the middle. Zmíněné typy útoků se často využívali v kombinaci se šířením malwaru. Například zneužitím zranitelnosti prohlížečů byl šířen finanční malware Zeus.[1]

Více než polovina všech kybernetických útoků byla uskutečněna právě přes webové stránky. Z toho, v některých případech byly použity Adobe Flash nebo Oracle Java. Oblíbeným útokem byl drive-dy útok, který nepotřeboval interakci uživatele. Téměř

polovina útoků byla realizována právě tímto způsobem. Nejčastějším malwarem šířícím se přes webové stránky byl malware napsaný v jazyce JavaScript.[1]

Zranitelnosti rozšíření byly zneužity u prohlížeče Google Chrome. Přesněji rozšíření pro vývojáře v sobě neslo malware, pomocí kterého útočníci mohli spouštět kód JavaScriptu, popřípadě získat v prohlížeči uložená hesla.[1]

Častými byly i drive-by download útoky, které zahrnovali stažení škodlivého HTTP nebo PHP skriptu z nezabezpečených stránek. Skript pak nainstaloval malware přímo na počítač nebo přesměroval oběť na nebezpečnou stránku.[1]

Na vzestupu také byly tzv. water holing útoky. V některých případech se útoky realizovali stažením malwaru do prohlížeče bez vědomí oběti. V jiných případech oběti byly přesměrovány na jiné stránky. Společným znakem těchto verzí útoků bylo to, že se specializovaly na užší okruh obětí. Byly případy, kdy se útok realizoval pouze u konkrétní verze prohlížeče nebo v případě, že počítač patřil do konkrétního rozsahu IP adres.[1]

Největšími zdroji útoků na webové stránky byly USA a Nizozemsko.[1]

Dalším typem útoků spojených s weby byly útoky na webové aplikace, služby a mobilní aplikace. Tyto útoky mířily na zneužití API rozhraní. Zejména se špatným zabezpečením nebo zranitelností. Častými cíli například byly API využívající SOAP/XML, REST/JSON, RPC nebo GWT. Mezi webové aplikace patřili WordPress nebo Magento.[1]

Většinovými cíli byly nadále státní sféra, finanční a IT organizace. Největší počet útoků byl realizován proti webovým aplikacím v oblasti IT (1 346) a ve státní sféře (1 184). Vysoký počet útoků zaznamenalo i zdravotnictví (610). Nicméně počet těchto útoků vůči minulému rok klesl.[1]

Útok pomocí SQL injection zůstal nadále populárním a na seznamu top 10 hrozeb od OWASPu se umístil na prvním místě. Převládali i útoky Local File Inclusion, Remote File Inclusion nebo PHP injection. Bylo zaznamenáno zvýšení Cross-site scriptingu. V některých případech útočníci jako zdroj útoku využily i Content Management Systém. Více než čtvrtina útoků byla mířena na webové aplikace a 93% z nich bylo realizováno organizovanou skupinou za účelem zisku.[1]

Rok **2017** znamenal snížení míry výskytu exploit kitů. Hlavním důvodem poklesu byla nižší zaručitelnost úspěchu než u jiných útoku, i když významnou výhodou exploit kitů je nepotřebnost interakce s uživatelem. Jeho úspěšnost záležela na několika faktorech: úspěšném skrytí činnosti, obejití bezpečnostních mechanismů a využití zranitelnosti.[1]

Navzdory malému výskytu byl objeven nový exploit kit Terror, který využíval některé schopnosti Metasploitu. Dlašími významnými exploit kity v tomto období byly Angler, Neutrio, RIG a Nuclear. Exploit kit Neutrino se již objevil během roku 2016

a přetrvával i v roce 2017. Pro nakažení svých obětí využíval water-holing útok a dovlečení malwaru do systému oběti. RIG se šířil přes podezřelé reklamy na kompromitovaných webových stránkách.[1]

Ve většině případů, exploit kity byly distribuovány během spamových kampaní nebo pomocí sociálního inženýrství. V některých případech bylo možné exploit kit i pronajmout. Buď na dobu dnů, týdnů nebo měsíců.[1]

Hlavní metodou infekce byly drive-by download, co nevyžadovalo interakci s uživatelem. Další metodou infekce bylo využití zranitelností prohlížeče během prohlížení dané stránky. Většina exploit kitů byla zaznamenána na území Asie. Důvodem mohlo být rozšíření Internet Exploreru s veřejně známými zranitelnostmi.[1]

Za rok **2018** byl zaznamenán nárůst útoků na weby i na webové aplikace. Útočníci využívali zranitelností samotných prohlížečů, přes HTML, Javu, JavaScript nebo přes jiné technologie. Nejčastějšími cíli byly Internet Explorer a Adobe Flash.[2]

Útoky pře weby se staly významnou součástí větších útoků. Například součástí spamových kampaní šířící bankovní trojské koně a šíření malwaru pomocí exploitů. Distribuce malwaru, stejně jako v minulém roce, byla realizována přes nezabezpečené webové stránky pomocí škodlivého PHP nebo HTML skriptu. Také převládali odkazy na škodlivé URL stránky, water-holing útoky a kompromitace pluginů nebo rozšíření. Převládajícím útokem na webové aplikace, ale nadále zůstal SQL injection. Následovaly LFI a XSS.[2]

Nově, například ke krádeži financí byl použit WinAPI, pomocí kterého byly vkládány škodlivé skripty do prohlížeče nebo webové stránky. Příkladem pro šíření exploit kitů byly webové stránky Hong Kongské telekomunikační firmy, kde byl nalezen exploit kit. V dalších případech bylo rozšíření v Google Chromu využito pro sběr přihlašovacích údajů a pro těžbu kryptoměn.[2]

Hlavními důvody úspěchu útoků na webové aplikace byly: zranitelnosti a špatná konfigurace nebo nasazení (29%), neošetřený cross-site scripting (24%), již předešlý únik informací (20%), injekce škodlivého kódu (12%) nebo autentizace (6%). Také byly využita tzv. slepá místa na webech a nepoužívaný, ale dostupný kód.[2]

Nejčastějším výsledkem útoků na webové aplikace byl únik dat. Nejvíce zasaženými oblastmi na webové útoky byly USA, Německo a Francie.[2]

Výskyt exploit kitů poklesl pod úroveň hlavních kybernetických hrozeb.[2]

Během roku **2019** vzrostly útoky na webové aplikace o více než 50% vůči roku 2018. V téměř 90% případů úspěšných útoků byla špatná bezpečnostní konfigurace nebo prolomení autentizace (45%).[5]

SQL injection nadále zůstal primární metodou útoku a byl využit až ve dvou třeti-

nách všech útoků. Local file inclusion byl použit v oblasti finančnictví. Dalším častým útokem byl také directory traversal.[5]

Významnými útoky na webové aplikace byly: buffer overflow (24%), DDoS v 20% případů, HTTP flood (23%), resource reduction (23%), HTTPS flood (21%) a Low Slow (21%). Útoky přes PHP API a Cross site scripting také nezaostávaly.[5]

V roce **2021** až 90% hackerských útoků mířilo na webové aplikace. 43% všech dat byla získána přes webové aplikace.[12]

Byly uskutečňovány DDoS útoky za pomocí získané nadvlády nad weby pomocí technik zmíněných výše. Stejně jak u spamu nebo u DDoS útoků, útočníci začali používat legitimní zabezpečené kanály.[12]

Podobně jak v minulých letech, hlavními důvody úspěšnosti útoků byly: špatné zabezpečení, tj. používání výchozích hesel a účtů, ponechání nepoužívaných webových stránek a neaktuálnost záplat. Nejvíce úspěšnými metodami útoků na webové aplikace byly: útok hrubou silou, slovníkový útok a session management attack.[12]

V některých případech webové aplikace byly složeny z více služeb od různých poskytovatelů, tj. webová služba nebo Java služby. Napadením nejslabšího článku byla získána nadvláda nad webovou aplikací.[12]

Útočníci se podíleli na šíření falešných zpráv ve spojení s covidem-19 na oficiálních stránkách zabývajícími se covidem-19 a také se podíleli na šíření nástrojů pro poškození webových stránek.[12]

Během roku **2022** byla více než polovina webových aplikací využita k datovým únikům. Pro útoky na zdravotnictví byly zahrnuty i webové aplikace. Významná většina technik na weby a webové aplikace zůstala stejná jak v předešlých letech.[13]

1.4 Kybernetické útoky zneužívající lidského faktoru

Podkapitola popisuje vývoj phishingových útoků a útoků pomocí sociálního inženýrství.

Stejně jak u jiných kybernetických hrozeb, v roce **2017** byl zpozorován trend nárůstu ve složitosti. Současně si útočníci začali vybírat konkrétní cíle útoku, tj. snížili rozsah svých útoků. Primárními cíli byly uživatelé, avšak nezaostávali ani cíle jako dodavatelé elektrické energie, výrobní a konstrukční závody nebo strojírenské firmy. Důvodem útoků byl přímý finanční zisk za pomocí dovlečení ransomwaru, spywaru do systému nebo kompromitace uživatelských účtů.[1]

Vyjma získání přihlašovacích údajů uživatelů pro kompromitaci jejich účtů se phishingové útoky používaly i jako inicializační vektor většího útoku. Například zavlečení malwaru, trojského koně, ransomwaru nebo otevření zadních vrátek. Na základě výše zmíněných až 74% kybernetických útoků využilo tento typ útoku ke kompromitaci

systému.[1]

Nejčastěji byly použity techniky spear-phishingu, business email compromise útoků. Většina phishingových emailů byla rozeslána z botnetů. Maily jako zdrojové emailové adresy používaly adresy důvěryhodných organizací. V obsahu mailů se také začaly objevovat legitimní loga organizací.[1]

Vůči předešlým letem, phisingové stránky začali mít mnohem kratší životnost a byly rozloženy na více různých webech, a tím byly útoky mnohem hůře zastavitelné. Například umístěním do blacklistů. Pro obejití bezpečnostních kontrol se například začaly jako přílohy používat heslem chráněné soubory, které se před kontrolou musely rozbalit, ale ještě před tím musely být staženy na zařízení oběti. V tělech mailů se začaly objevovat odkazy na legitimní nebo populární stránky, a to ztížilo detekci phishingových útoků.[1]

Během roku **2017** byl zaznamenán nárůst phishingových stránek v průměru milionu stránek za měsíc a zkrácení jejich životnosti na 4-8 hodin. Nejčastějším obsahem phisingových mailů byly: oznámení o úniku dat (12%), expirace hesla za méně než 24 hodin (12%), okamžitá změna hesla (10%) nebo důležitá informace z oddělení lidských zdrojů (10%).[1]

U mobilních aplikací se phishingové útoky objevili ve formě upozornění, přes které se útočníci snažili ukrást přihlašovací údaje.[1]

Větší zacílení útoků v podobě konkrétních obětí pokračovalo i v roce **2018**. Mezi takové cíle kromě společností patřili bohatí jednotlivci, státní orgány, tj. osoby nebo instituce, které měly přístup k citlivým údajům.[2]

Spear-phishingové útoky se zaměřily na vysoký management, tj. nejvyšší vrstvu řízení. Většina takových útoků mířila na organizace s nemovitostmi. Jindy spear-phishingové útoky mířily na oddělení lidských zdrojů. V jiných případech byl spear-phishing pouze inicializačním vektorem pro možnost dovlečení malwaru nebo realizace špionáže. Vishingové útoky se pak vyskytovali u finančních organizací a samotné phishingové útoky se zaměřily na sociální média.[2]

Kromě výše zmíněných cílů největšími cíli byly také emailové služby jako Microsoft Office 365, online služby jako Dropbox, Google Drive a služby s financemi jako stránky finančních institucí nebo Paypal.[2]

Byly používány techniky jako typosquatting, domain shadowing, propagace škodlivých domén a URL a subdomain services. Technika typosquattingu byla využita několika různými způsoby. Nejčastěji zahrnovala: výměnu jednoho znaku v názvu webu, přidání dalšího znaku do názvu, smazání nebo přidání začátečního nebo konečného znaku. Pro navození přesvědčení o bezpečné stránce útočníci začali využívat volně dostupné certifikáty. Nejčastěji používanými slovy pak byly: platba, urgentní, žádost,

pozornost, důležitý, důvěrný, okamžitá odpověď, převod nebo důležitý update. Vyjma slov se také často vyskytovali přílohy s názvy spojenými s financemi. Například objednávka, platba, faktura nebo účtenka. Nejvíce se vyskytovaly přílohy typu: Microsoft Office, pdf, archivy, JavaScript, visual basic scripty.[2]

Pro jednu legitimní stránku bylo založených 20 stránek s podobným názvem. V období posledních sedmy let bylo zaznamenáno postupné zvyšování mobilních phishingových útoků. Z důvodu větší efektivity se také navýšila míra phishingu přes SMS, WhatsApp, Facebbok Messenger a sociální média. Častým tématem phishingu byl i covid-19.[2, 3]

Na začátku roku **2019** byl zaznamenán nárůst výskytů virů o více než 600% z pouze ze 2% celkových phishing scamů. Také se zvýšily útoky na zdravotnictví. K útokům byl využit i spear-phishing. Například u ukrajinské vlády.[3]

Stejně jako v předešlých letech byl phishing použit jako součást dalšího významnějšího útoku. Například phishingový útok na Lancasterskou univerzitu způsobil únik dat. V jiném případě byly získány přihlašovací údaje Discordu.[3]

Nově, vzhled phishingových emailů byl vytvořen dle vzorů zdravotnických společností WHO a CDC. Maily rozesílaly informace o falešné míře infekce v okolí oběti nebo informace o vývoji pandemie, které vedly na škodlivé stránky.[3]

Rok **2020** také zaznamenal nárůst phishingu. Stejně jak v roce 2017, některé maily používaly slovo platba. Nejvíce byly cíleny oblasti software-as-a-service, webové služby, cloudová úložiště, finanční služby a telekomunikace. 88% organizací zaznamenalo spear phishingové útoky, které měly za cíl získat přihlašovací údaje. Během pandemie byl zaznamenán nárůst phishingových útoků o 667%, a to pouze v průběhu jednoho měsíce.[3]

Více než dvě třetiny phishingových stránek začaly využívat HTTPS připojení. V některých případech útočníci mohly použít i již nabourané stránky pro svůj útok. Téměř polovina příloh byla typu Microsoft Office.[3]

Současně se zvýšila míra využití phishingu jako služby. Byly nabízeny nástroje za poměrně nízkou cenu, například za 50-80 amerických dolarů měsíčně. V některých případech byly tyto služby hostovány na DNS serverech nebo veřejně dostupných legitimních cloudech. Útočníkům se za využití těchto služeb podařilo zkompromitovat kolem 65 tisíc účtů.[3]

Nejčastějším cílem byl Microsoft Office 365. Až 99% případů v případě distribuce malwaru potřebovala interakci oběti.[3]

I v roku **2021** byl hlavním důvodem phishingu finanční zisk. Pro většinu útoků byla využita služba Phishing-as-a-service. Například byla prodávána sada pro phishing s tématem covidu-19. Naopak se snížila míra phishingu jako inicializačního vektoru. Většina phishingových útoků byla použita pro umístění ransomwaru na zařízení oběti.[12]

Dále se pomocí falešných zpráv šířily dezinformace. Ve spojení s covidem-19 byl použit spear-phishing i phishing. Spear-phishing byl použit i na firmy s nemovitostmi, auty a s kryptoměnovými směnárnami nebo ve výjimečných případech i na státní instituce. Příkladem je útok skupiny The Dukes, která na konci roku 2021 realizovala spear-phishingový útok na některá evropská velvyslanectví a na Ministerstva zahraničních věcí. Přesvědčili své oběti k otevření HTML stránky, která pak stáhla škodlivý ISO nebo VHDX soubor obsahující malware. V oblasti politiky byly použity SMShing a vishing. V případě vishingu jedna skupina útočníků použila software, který umožňoval pozměnění jejich mužského hlasu na ženský. Jiná skupina pomocí SMS zpráv a emailů rozesílala falešná bezpečnostní upozornění od Googlu.[12]

Útočníci se zaměřovali na uživatele vlastnící kryptoměny nebo na kryptoměnové směnárny. Také byl realizován útok na uživatele Coinbase.[13]

Rozesílaly se falešné maily od poštovních služeb s cílem získat bankovní informace obětí. Během pandemie byly rozesílány emaily ve jménu bank, spolu s falešným QR kódem, který vedl buď na stránky vyžadující zadání informací o kreditní kartě, na škodlivé stránky nebo na stránky, přes které byl stažen malware.[12]

Z oblasti mobilních technologií se v Evropě objevil bankovní malware FluBot. Byl určen pouze pro operační systém Andriod. Na samotném začátku útoku rozesílal SMS zprávy, které obsahovaly informace o falešné zásilce nebo softwarem. Zpráva dále obsahovala odkaz pro instalaci potřebné aplikace, která však pro svůj běh požadovala největší oprávnění. Po instalaci pak aplikace realizovala sběr dat. Mezi sbíraná data patřily: čísla kreditních karet, přihlašovací údaje, SMS zprávy, snímky obrazovky.[12]

Dle statistik se náklady na phishing od roku 2015 do roku **2021** více než ztrojnásobily. Mezi nákladné úkony například patřily identifikace přesného zdroje infekce, následné čištění a oprava.[12]

Rok **2022** přinesl převládající trend v používání sociálního inženýrství místo phishingových zpráv a finanční zisk zůstal nadále primárním cílem těchto útoků. Bylo zaznamenáno, že až 82% datových úniků bylo z důvodu zneužití slabiny lidského faktoru. Z toho alespoň 60% bylo realizováno pomocí sociálního inženýrství. Pomocí vishingu nebo spear phishingu byl do systému zavlečen ransomware.[13]

Hlavním cílem útočníků byla krádež přihlašovacích údajů, které pak mohli použít k jinému, více zaměřenému, útoku. Ke krádeži údajů byly většinou použity phishingové kampaně.[13]

Nejvíce zasaženými oblastmi byly finanční organizace a technologické firmy jako Microsoft, Google a Apple. Nadále bylo využíváno téma covidu-19. Stejně jak v minulých letech pokračovalo používání QR kódů vedoucích na nebezpečné stránky. V některém z případů phishingových kampaní se útočníci vydávali za vědce jedné z lon-

dýnských univerzit a rozeslali falešné pozvánky profesorům, novinářům na neexistující konference. V jiném případě se útočníci vydávali za atraktivní ženu, která oběti zaslala infikované dokumenty.[13]

U vishingových kampaní se například útočníci vydávaly za členy policie nebo zaměstnance finančních institucí a přesvědčovali své oběti, aby převedly své peníze na jiný, zdánlivě bezpečnější, účet z důvodu kompromitování aktuálního bankovního účtu. Zajímavý způsob zaslání škodlivého kódu na zařízení oběti bylo zpozorováno na území Asie. Škodlivý kód byl zaslán až v odpovědi na předchozí mailovou komunikaci. Byly situace, kdy útočníci ukradli mailovou konverzaci a použili ji v írně pozměněné podobě.[13]

Stále více častým se stalo používání phishingových sad. Vysoká míra sofistikovanosti umožnila maskování se za legitimní obsah. Sady například umožňovaly správné používání pravopisu daného jazyka. V některých případech bylo zaznamenáno, že tyto sady získané informace nezasílají pouze aktuálním útočníkům, ale i autorovy dané sady. Mezi nabídky phishingu jako služba byla nově zařazena i služba zvaná initial access brokers, co znamená odkoupení již předešle získaných přihlašovacích údajů dané organizace.[13]

Ve více než polovině případů útočníci volili kompromitaci zaměstnaneckých emailových účtů. Útok byl snadno realizovatelný a poměrně levný, protože nebyly potřebné žádné další nástroje, například pro obcházení mechanismů dané infrastruktury. Z pohledu organizací finančně nejzávažnějším útokem bylo kompromitace zaměstnaneckého účtu. Pro zvýšení důvěryhodnosti se zakládaly emailové adresy na legitimních emailových stránkách, jako například Office 365.[13]

Válka na Ukrajině byla využita různými útočníky. V tomto případě se zejména zaměřovali na vládní instituce, obranné síly, politiky, novináře a nevládní organizace. Příkladem byla kampaň zvaná COLDRIVER. V souvislosti s válkou pak byl na začátku roku 2022 uskutečněna phishingová kampaň na evropskou vládu zneužitím emailu ukrajinského člena ozbrojených sil.[13]

V budoucnosti s rozvojem technologií hrozí větší míra použití automatizace útoků. Například za využití umělé inteligence. Také je očekávána větší zaměření na vybraný rozsah osob.[13]

1.5 Shrnutí vývoje kybernetických útoků

U všech typů útoků se vyskytl podobný vývoj ve složitosti, rozsahu a zaměření. U všech typů se zvyšovala složitost z pohledu širší nabídky funkcionalit. Útoky se v průměru postupně začaly zaměřovat na užší rozsah cílů s pravděpodobností většího finančního zisku, jako například finanční instituce, IT firmy, zdravotnictví nebo průmysl.

Pro realizaci útoků začaly být nabízeny hotové nástroje nebo bylo možné si zakoupit pouze samotný útok. Útočníci si také zjednodušili způsob k získávání dat, a to napří-

klad pomocí phishingu nebo sociálního inženýrství. Nicméně v některých případech hlavními vektory útoku zůstaly již tradiční útoky. Například u DDoS útoků téměř ve všech obdobích převládal UDP a TCP-SYN flood. V některých případech k němu byly připojeni HTTP, DNS a ICMP flood.

U phishingu hrály roli zprávy s důležitou informací nebo urgentním obsahem. Tyto zprávy nebo jiné techniky se velmi rychle přizpůsobovaly aktuální situaci. U webových aplikací byly zejména využity špatné konfigurace a špatné zabezpečení.

Dle statistik se během pandemie zvýšila míra DDoS útoků, spamových útoků a phishingových útoků, které byly v tomto případě určeny koncovým uživatelům. Ke zvýšení DDoS útoků přispěla i válka na Ukrajině. Botnety chytře využily zranitelné IoT zařízení, aby je zařadily do své sítě nebo na ně uskutečnily útok. Pro zamezení provozu nebo vydírání některých podniků byly použity supply chain útoky, které byly čím dál častěji používány.

2 NÁSTROJE PRO DETEKCI ÚTOČNÍKŮ

Kapitola popisuje způsoby detekce útočníků za použití honeypotů, monitoringu a pomocí detekčních systémů, přesněji IDS. V jednotlivých podkapitolách budou tyto metody popsány blíže.

Detekovat útočníky lze několika způsoby: za pomocí nasazení zranitelného systému, *honeypotu*, nebo sledováním a zjišťováním nesrovnalostí v provozu sítě nebo na zařízeních, tj. *monitoring a detekce*. Nicméně pro možnost detekci je také nutný sběr dat, který lze přímo získat z honeypotu nebo z monitoringu. Je třeba poznamenat, že předešle zmíněné přístupy nelze použít samostatně, protože mezi sebou vzájemně souvisí a doplňují se.

První podkapitola popisuje honeypoty a jejich účel a jednotlivé typy. Druhá podkapitola se zabývá důležitostí monitoringu sítě. Poslední podkapitola se zabývá IDS systémy, které lze použít pro detekci útočníků na síti a na koncových zařízeních.

2.1 Honeypoty

Honeypot je zařízení, které simuluje skutečný reálný systém nebo jeho část. Pro větší pravděpodobnost nalákání útočníka je honeypot záměrně nastaven jako lehce přístupné zařízení s cennými informacemi. Nicméně slouží pro sledování, sběr a zaznamenávání veškerých činností, technik, použitých nástrojů a postupů útočníka, snížení šance útoku na skutečný systém a současně pro detekci případného útoku. Mezi zaznamenávané činnosti lze zařadit: přístupy k souborům a složkám, pokusy o přístup nebo spuštění procesů. [16, 17, 18]

Důležité je poznamenat, že při nasazování je potřeba myslet na maskování, které musí simulovat skutečné zařízení v systému, tj. musí obsahovat realisticky vypadající soubory a složky.[19]

Výhodou nasazení honeypotu je, že každý přístup je považován za neautorizovaný, takže každý přístup je považován za škodlivý. To je také důvodem malého množství falešně pozitivních záznamů a přehlednějších, srozumitelnějších a stručnějších záznamů.[16, 17, 18]

Vyjma detekce útoků, mohou mít i schopnost potlačit útoky. Obranou ze strany honeypotů je zpomalení skenování při navázání spojení. Zpomalování je docíleno snižováním velikosti okna u TCP paketů až postupně na nulu. [17]

Z důvodu, že zachytávají informace, které jsou adresovány pouze jim, tak trpí menším zahlcením a zároveň při jejich nasazování není potřebný velký výpočetní výkon a paměťové úložiště. Na novost nebo aktuálnost systému honeypotu také není nutné dát takový důraz jak u systémů určených přímo pro monitoring nebo detekci.[17, 18]

Další výhodou honeypotů vůči metodám analyzujících síťový provoz je to, že jsou schopny pracovat i v šifrovaném prostředí, protože zaznamenávají činnosti útočníka.[16, 17]

Významnou předností honeypotů je schopnost detekce nových, neznámých útoků nebo útoků, na které ještě nebyly vytvořeny žádné vzory, signatury. Například mohou detekovat pokus o zneužití zero-day zranitelností. Z těchto záznamů pak lze vytvořit signatury pro detekční systémy. Stejně tak je velká šance včasné detekce útoku nebo pokusu o útok.[16, 17]

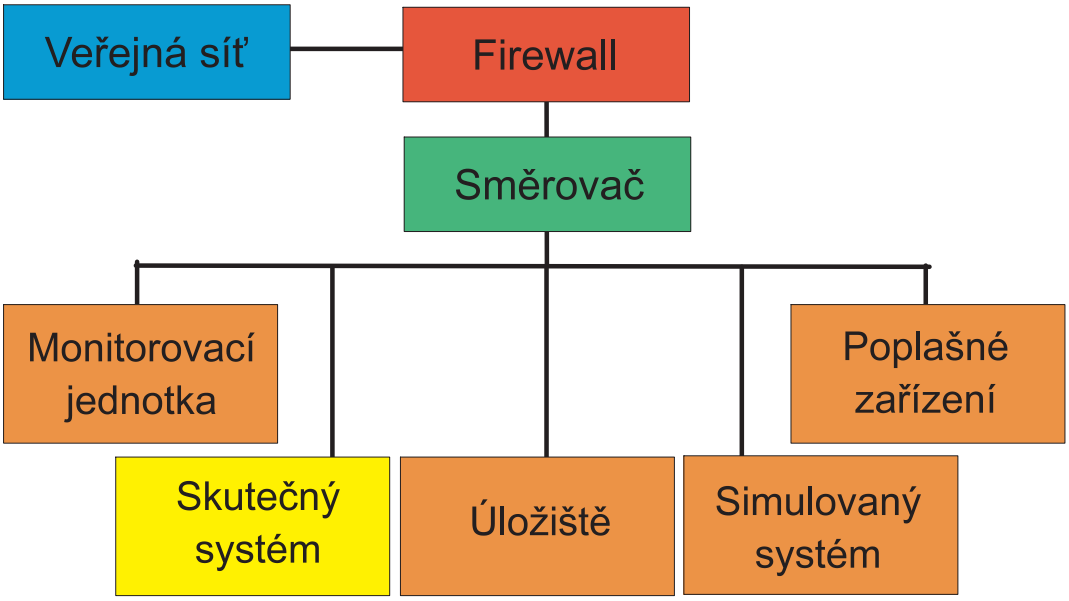
Výše zmíněné klady také závisí na způsobu sestavení, nasazení a použití honeypotů [18].

V návaznosti na předešle zmíněné, špatné nastavení, nedostatečná interakce, nedostatečné maskování nebo charakteristické vzory chování honeypotu mohou vést k odhalení honeypotu, který pak může být zneužit ve prospěch útočníka. Může se stát prostředkem útoku na samotný systém, kde se nachází, nebo součástí útoku na jiné infrastruktury. V jiném případě může být zneužit pro maskování jiného útoku na stejnou infrastrukturu, kde se nachází. [16, 17]

Vůči detekčním systémům, které analyzují provoz na síti, je nedostatkem honeypotu to, že zaznamenává pouze to, co je určeno pouze pro něj. To také znamená, že honeypot nemůže zastoupit ostatní systémy zajišťující bezpečnost. [16, 17]

Nevýhodou může být složité nastavení a údržba ve formě co nejlepšího maskování [17]. Pro větší šanci využití honeypotu a snížení rizika útoku na skutečný systém, honeypot musí zabírat větší IP rozsah a také musí být dostupný z vnější sítě [17, 18].

Honeypoty se většinou skládají z: *firewallu, napodobeniny skutečného systému, úložiště, monitorovacího a poplašného zařízení*, viz obrázek 2.1. Hned na počátku přístupu na simulovaný a zranitelný systém s falešnými soubory je útočník detekován firewallem, který eviduje veškerou příchozí komunikaci. Současně je činnost útočníka zaznamenávána monitorovací jednotkou, která sleduje aktivitu na síti, v oblasti služeb a v oblasti operačního systému. Záznamy jsou ukládány na úložiště a detekovaný útok v simulovaném prostředí je odesílán na řídící stanici přes poplašné zařízení. Mezi tím simulovaný systém komunikuje s útočníkem a snaží se jeho pozornost udržet co nejdéle. Zpráva o útoku může obsahovat: pořadí, časové razítka a typ paketů, změněné soubory nebo zadané příkazy.[17]



Obrázek 2.1 Základní model honeypotu [17]

Honeypoty lze rozdělit z pohledu interakce s útočníkem, dle účelu použití, dle role, dle umístění a dle hardwarového nasazení. Zmíněná rozdělení budou popsána níže.

2.1.1 Honeypoty dle interakce s útočníkem

Honeypoty dle inerakce se dělí do tří skupin: s nízkou interakcí, střední a vysokou interakcí. Současně úroveň interakce určuje množství zachycených dat [21, 20].

Na úrovni nízké interakce, **low-interaction**, se simulují pouze určité služby nebo procesy, se kterými může útočník komunikovat. Většinou jde o webové služby, přes HTTP spojení nebo služby pro práci se soubory nebo pro jejich ukládání. Je potřeba zmínit, že honeypot pouze zachytává činnost útočníka, ale již neodpovídá na jeho dotazy nebo požadavky. [17]

Běží nad úrovní operačního systému, ke kterému útočník nemá přístup, tj. při útoku je nižší míra rizika zneužití. Vůči ostatním typům jsou jednodušší na nasazení a údržbu. Současně, ale zaznamenávají pouze omezené množství dat, avšak v případě útoku je zasažena pouze simulovaná služba. [17, 20] Může se stát, že všechny funkce nebo příkazy nebudou fungovat, a tím je také vyšší riziko odhalení [19].

Jsou vhodné pro detekci IP adresy útočníka, identifikaci skenů a automatizovaných útoků, pro sběr vzorků a zjišťování trendů, pro oklamání tzv. script kiddies a odlákání útočníků od skutečného systému.[17]

Honeypoty se střední úrovní interakce, **medium-interaction**, kombinují přednosti honeypotů s nízkou a s vysokou úrovní interakce. Stejně jak honeypoty u honeypotů s nízkou úrovní interakce operační systém pro útočníka není přístupný. Služba, aplikace

nebo některé procesy operačního systému běží na virtualizované vrstvě. Tyto honeypoty útočníkovi nabízejí větší množství funkcionalit, využití protokolů a reakce na dotazy. V některých případech je i omezená možnost komunikace s operačním systémem [18]. [17, 18, 20]

Jejich nasazení je časově náročné a vyžaduje rozsáhlou znalost služeb, zabezpečení a protokolů. V případě útoku je možné očekávat větší míru rizika než u předešle popsaného honeypotu.[17]

Vysoká míra interakce, **high-interaction**, již zahrnuje kompletní přístup k službám, operačnímu systému a jeho zdrojům a také kompletní interakci s útočníkem. Pro nasazení a údržbu jsou náročné, buď z pohledu znalostí a zkušeností nebo z pohledu času. Nicméně mají detailnější záznamy a jsou schopné identifikovat nové exploity, zranitelnosti a malware nebo zaznamenat celý postup kompromitace systému.[17, 21, 20]

Jsou vhodné pro výzkumné účely, pro sběr dat, pro analýzu, pro identifikaci zranitelností, hrozeb, zjišťování technik a trendů. Oproti předešlým honeypotům jsou nejčastěji nasazovány ve skutečných systémech [18] nebo v případech výzkumu a zjišťování technik útočníků u společností z oblasti bezpečnosti. Vůči předchozím typům jsou schopny posbírat více dat a také přinášejí větší riziko v případě útoku, protože mohou být ovládnuty útočníkem.[17, 18, 21]

2.1.2 Honeypoty dle účelu použití

Dalším možným dělením jsou tzv. *produkční* a *výzkumné* honeypoty. Jak již název napovídá, produkční honeypoty jsou používány v systémech zajišťujících například výrobu a nejsou využívány v oblasti výzkumu. Slouží pro ochranu a zmírnění případného útoku na organizaci nebo společnosti. [18, 17, 20]

Produkční honeypoty jsou obvykle lehké na nasazení a údržbu a také s nízkým počtem funkcí. Většinou jde o honeypoty s nízkou interakcí. Sbírají údaje pouze o napadených systémech, ale ne o samotném postupu. Také mohou objevit nedostatky v zabezpečení nebo neobjevené zranitelnosti [17]. [18, 20]

Honeypoty určeny pro výzkum, oproti produkčním honeypotům, slouží primárně pro zjištění informací o aktuálních postupech, hrozbách a používaných nástrojích a mají vlastní operační systém. Většinou jde o honeypoty s vysokou mírou interakce. Získané údaje jsou použity k dalšímu výzkumu pro zlepšení obrany proti hrozbám. Jsou používány univerzitami, armádou nebo společnostmi zabývající se bezpečností. [17, 18, 20]

2.1.3 Honeypoty dle role v komunikaci

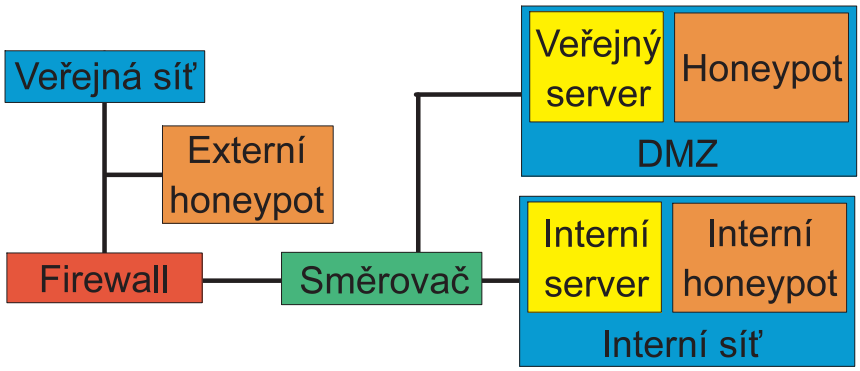
Honeypot s útočníkem interaguje jako server dle typu komunikace klient-server, tj. začátek spojení očekává ze strany útočníka, se kterým pak následně komunikuje. Je vhodný

pro detekci nových exploitů nebo pro sběr malwaru.[17]

V druhém případě honeypot může zastávat roli klienta. To pak znamená, že honeypot aktivně vyhledává škodlivé servery, ze kterých realizuje sběr dat. Například stažení malwaru. Jeho účelem je sbírat informace o útocích, které míří na koncové stanice nebo aplikace. [17, 19]

2.1.4 Honeypoty dle umístění

Dále je možné rozdělit honeypoty dle umístění: externě, hned na hranici sítě před firewall, interně, do interní sítě, a do DMZ zóny mezi veřejně dostupné servery a aplikace. Tyto možnosti jsou zobrazeny na obrázku 2.2.



Obrázek 2.2 Rozmístění honeypotů [17]

Umístění před samotnou síť snižuje riziko útoku na interní síť, nicméně zvyšuje množství falešně pozitivních záznamů a nezachycuje útoky mířící na interní síť. Další nevýhodou je, že zachycuje velké množství provozu a je vyšší šance pro kompromitaci honeypotu.[17]

Většinou používá stejnou veřejnou IP adresu jako samotná síť a slouží pro sběr nejškodlivějších a nejnebezpečnějších útoků [17].

V DMZ zóně je honeypot stejně dobře dostupný jako při externím umístění. Má přidělenou IP adresu ze stejného rozsahu jako servery v DMZ zóně. V tomto případě již detekuje menší množství falešně pozitivní záznamů, takže umožňuje včasné zachycení útoku. Útoky jsou zachytávány pomocí zrcadlení portu. Stejně jako v případě externích honeypotů neumožňuje zachytávání útoků na interní síť. [17]

Interní honeypot je umístěn přímo do interní sítě a zachycuje příchozí škodlivý provoz, ale i možné vnitřní hrozby. V případě kompromitace honeypotu je možné, že by se mohl stát prostředkem útoku na vnitřní síť. Z toho důvodu je doporučeno použít i jiné bezpečnostní mechanismy, jako například další firewall nebo zachytávání provozu pomocí zrcadlení portů. [17]

2.1.5 Honeypoty dle hardwarového nasazení

Honeypoty mohou být nasazeny jako *fyzické* zařízení nebo jako *virtuální* stroj.

V případě fyzického nasazení, jde o skutečný počítač s operačním systémem a dalšími službami s připojením do sítě se svou IP adresou. Tato možnost se volí u honeypotů s vysokou interakcí. Nevýhodou je náročná údržba a omezená viditelnost IP adresy, protože jde pouze o jednu konkrétní IP adresu.[17, 18]

U virtuálních honeypotů jde o virtuální stroje, které běží na fyzickém zařízení. Vůči fyzickým honeypotům jsou schopny zastoupit větší rozsah IP adres, co zvyšuje šanci na nalákání útočníka. [17, 18]

2.1.6 Jiné možnosti nasazení honeypotů

Jednotlivé honeypoty pak lze propojit a získat tak síť honeypotů, **honeynet**, které pak zlepšují sběr dat i obranyschopnost sítě [16]. Pro tyto účely jsou většinou použity fyzické honeypoty s vysokou interakcí [17]. Výhodou je těžká detekce ze strany útočníka.[18]

V jiných případech honeypotem nemusí být samotné zařízení, ale pouze jeden nebo více souborů. V takovém případě jsou nazývány jako **honeytokeny**. Jde o soubory s potenciálně zajímavým obsahem pro útočníka, ale současně s falešnými informacemi. Mohou mít například název *passwords.txt*. Vyjma umísťování na samotné honeypoty, mohou být použity pro detekci podvodů [53].[19]

Honeypoty nemusí stát samostatně, ale mohou být například součástí systému pro detekci anomálií, v takovém případě je lze nazývat stínovými honeypoty, **shadow honeypot** analyzující provoz, který je považován za odchylku v síti.[16]

2.2 Monitoring

Pro detekci útočníků se vyjma honeypotů používá i sledování provozu sítě. Zachycený provoz je ověřován a klasifikován pomocí detekčního (IDS) nebo prevenčního (IPS) systému v reálném čase. Současně, ale monitoring také slouží pro sběr dat, které jsou použity pro zmírnění zátěže sítě, plánování vývoje, řešení potíží, zajištění bezpečnosti, také ke sběru, uchování a analýze užitečných a významných dat pro tvorbu záznamů o útocích nebo pro tvorbu modelu standardního chování sítě nebo pro detekci incidentů.[22, 23, 24]

Sledování sítě napomáhá k přehledu nad celým systémem z pohledu hardwaru i z pohledu softwaru, k optimalizaci spolehlivosti sítě ve formě identifikace nefunkčních zařízení nebo upozornění na zpoždění. Současně může včasně zamezit výpadky, snížit prodlevy, detekovat místa nebo původce potíží. Díky sledování provozu a zařízení lze odhalit neobvyklé chování, malware nebo jiné bezpečnostní hrozby.[23]

Jednotně: hlavním cílem je zajištění neustálého běhu a funkčnosti služeb. To je také důvod pro sledování, analýzu a klasifikaci síťového provozu a funkčnosti samotné sítě. Mezi sledovanou komunikaci lze například zařadit komunikaci mezi síťovými prvky, komunikaci na serverech, na virtuálních strojích. Je sledován, jak odchozí, tak příchozí provoz. [24, 25]

Jak bylo popsáno výše nasbíraná data lze použít pro tvorbu signatur v detekčních a prevenčních systémech, pro tvorbu modelu obvyklého chování sítě, řešení chyb nebo pro ochranu sítě.[25]

Pro tyto účely slouží tzv. **historická**, průběžně zachycovaná, data, která jsou podrobena další analýze a zpracování. Provoz zachycený v reálném čase pak může být ověřován právě vůči zpracovaným historickým datům ve formě signatur nebo modelu chování. Reálná data mohou být také využita ke klasifikaci provozu.[25]

Vyjma sledování a sběru dat je nutné dbát i na umístění sledovacích nástrojů nebo zařízení. Je potřebné určit místa, která jsou nejefektivnější pro sledování provozu a současně monitorují pouze nezbytné, tj. vybrat zařízení dle výkonu a důležitosti v hierarchii komunikace. Zejména jde o zařízení pracující s kritickými nebo důležitými informacemi. Mohou to být síťové prvky, servery nebo v některých případech i koncová zařízení jako například počítače. [25]

Důležitým faktorem je také nastavení intervalu sledování. Například využití úložiště má být kontrolováno každých 15 minut nebo dostupnost má být ověřována každou minutu. Dalším významným parametrem pro nastavení je práh provozu sítě, který hraje významnou roli, například při detekci DoS útoků.[25]

2.2.1 Způsoby sledování sítě

Podkapitola popisuje možné způsoby sledování síťového provozu.

Komunikaci lze sledovat z více pohledů. Jedním z nich je **interní** a **externí** monitoring. Pro **interní** monitoring jsou používány nástroje přímo dostupné na již existujícím hardwaru, tj. nejsou použity žádné dodatečná zařízení. U **externího** monitoringu jsou již nasazeny další externí zařízení, které mají pouze za cíl sledování provozu.[25]

Vyjma používaných zařízení ještě lze monitorovací systém umístit **přímo do sítě** nebo **na vzdálený**, externí nebo cloud computingový server. V některých případech lze také sledovat součást sítě, které má jiné geografické umístění. [25]

Provoz lze zachycovat **pasivním** nebo **aktivním** způsobem. Jak už název napovídá, **pasivní** sběr dat je realizován neinvazivním způsobem, tj. jsou přímo zachycována data, které procházejí sítí. Naopak, **aktivní** přístup do sítě zasílá různé zprávy. Většinou jde o zprávy zjišťující dostupnost zařízení, procesu, rychlost sítě nebo rychlost odezvy.[25, 24]

Zachycené informace lze zpracovávat jako exportovaný souhrn, **flow-based monitoring**, nebo celkové pakety zachycené během provozu, **packet-based monitoring**. K poslední zmíněné metodě patří i **deep packet inspection (DPI)**, který kontroluje celý obsah paketu a je hlavně využíván pro klasifikaci provozu. Výhodou **packet-based monitoringu** je zjištění a zamezení úzkých hrdel sítě. [24]

Zachycená data jsou pak zpracována a převedena na posloupnost vlastností, která jsou uložena v databázi a používána dále.

2.3 Detekce

Přímá detekce útočníků je realizována pomocí detekčních systémů, IDS. Tento systém pomocí ověřování veškerého příchozího a odchozího provozu detekuje a oznamuje možné průniky do systému nebo počítačové sítě [17, 21, 27].

Detekční systém slouží pro identifikaci neautorizované činnosti, zneužití nebo zneužívání počítačového systému ze strany externích nebo i interních útočníků [16]. Dokáží detekovat napadení, chyby v konfiguraci systému. Kontrolují a hlásí změny dat a automatizují monitoring.[52]

Pro možnost nasazení, IDS systém musí mít absolutní náhled do celé sítě. Vysoká rychlost sítě může vést ke ztížení zpracování dat provozu. Nedostatkem je velké množství generovaných upozornění, jak z pohledu zatížení sítě. Vážným nedostatkem IDS systému je nemožnost zpracování šifrovaného provozu.[17]

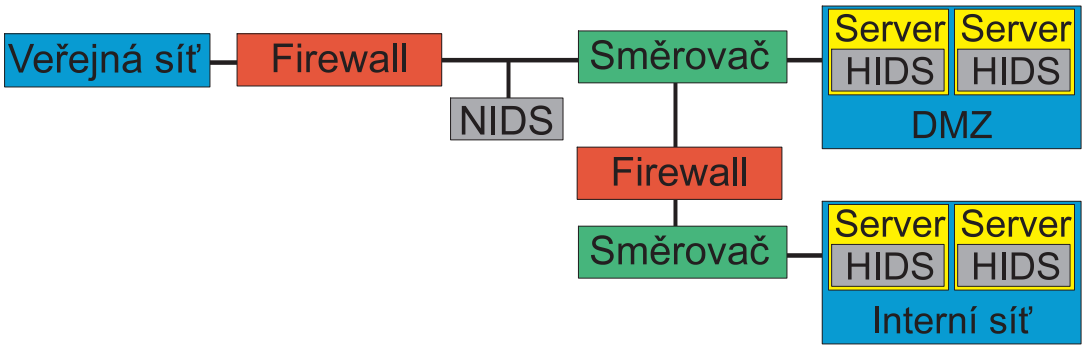
Současně detekčním systémem nelze nahradit autentizační mechanismy, prostředky pro zajištění integrity informací a také je nelze použít pro zabezpečení slabin síťových protokolů.[52]

Podkapitoly níže popisují možnosti umístění detekčního systému, způsoby detekce a možnosti zpracování zachycených dat.

2.3.1 Detekce dle umístění

IDS systémy lze umístit do sítě, **network-based** (NIDS), na koncová zařízení, **host-based**(HIDS). Příslušné rozmístění je uvedeno na obrázku 2.3. Taktéž lze odposlouchávat provoz na bezdrátové síti, **wireless**.[26]

V prvním případě je zařízení nebo software zachycující veškerý provoz umístěno do sítě. Provoz může zachycovat hned na hranicích sítě nebo již uvnitř. Hledá vzory, chyby nebo jiné aktivity, které by napovídali útoku. Během své činnosti dokáže detekovat skenování portů, výskyt virů, červů nebo spamu, průnik do sítě, útoky na síťové prvky nebo DoS útoky. Pokud je detekován možný útok, tak je k němu vytvořen záznam a posláno upozornění. [26, 21]



Obrázek 2.3 Network a host-based IDS [21]

Network-based se obvykle skládají z několika částí [26] :

- ze senzoru, který zachytává veškerý provoz na síti,
- ze zařízení, které porovnává zachycený provoz se svou databází pravidel a současně je schopno učit se a přidávat, tak nové záznamy do své databáze,
- a ze zařízení, které zobrazuje upozornění o sledování a podezřelém chování.

Celkově NIDS detekují cizí nebo jiné sondy a služby na síti, několikanásobné pokusy o vzdálené přihlášení, neobvyklý provoz, skenování IP adres a portů, vysoký provoz na DNS serverech nebo zvýšené a neodůvodněné využití šířky pásma.[26]

Naopak **host-based** IDS zaznamenávají události na konkrétním zařízení, tím že monitorují každou systémovou událost. Využívají stejné detekční metody jako NIDS, s tím rozdílem, že detekují hlavně zneužití systému, jako například neplánované restarty, podezřelý příchozí provoz nebo procesy, zneužití oprávnění, změny v souborech, nové soubory nebo vypnutí antiviru nebo firewallu. Současně sledují stav systému, činnosti osob, oprávnění běhu programu v daný časový okamžik. [26, 21, 27]

Wireless IDS systémy sledují provoz a detekují události na bezdrátové síti. Současně mohou analyzovat i používání protokolů pro bezdrátovou komunikaci.[28]

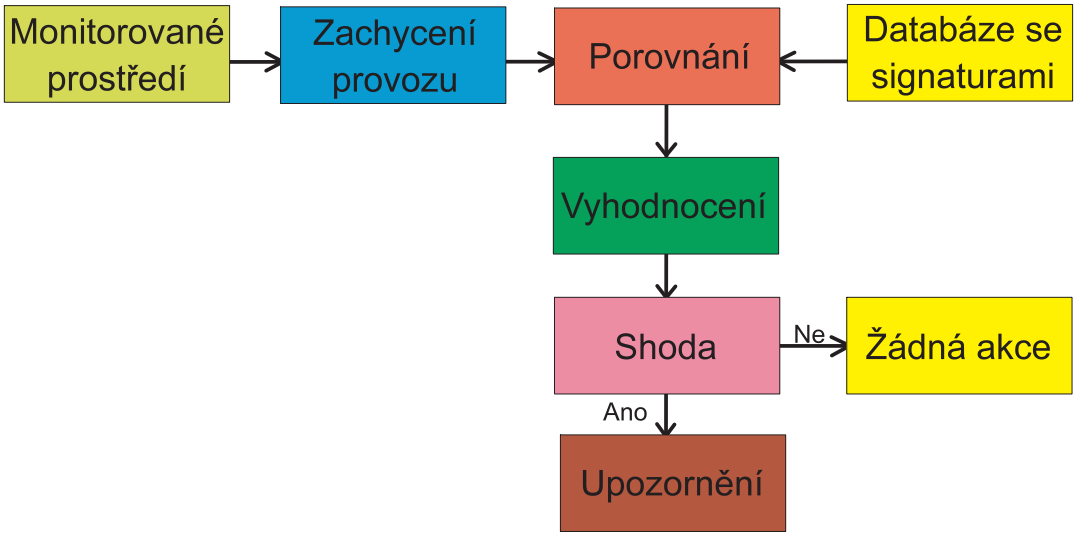
Nejlepším způsobem je kombinace několika nebo všech zmíněných způsobů detekce.

2.3.2 Detekce dle způsobu porovnávání zachycených dat

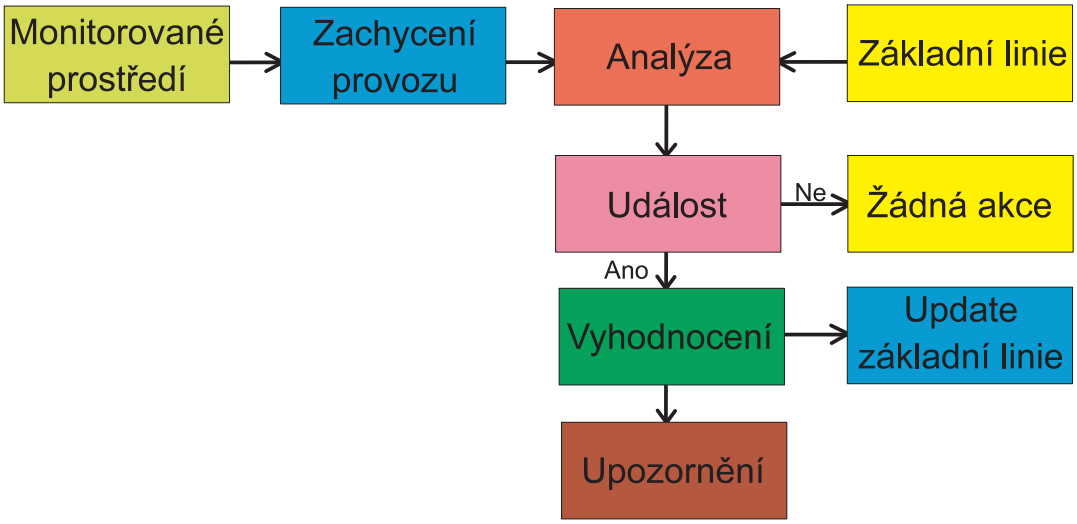
Útoky lze detekovat na **základě signatur**, pomocí předem nastavených vzorů. Dalším způsobem detekce je **detekce anomálií**, kde jsou detekovány odchylky vůči standardnímu chování sítě nebo vůči standardní struktuře paketů daného protokolu.[21]

Obecným postupem pro detekci útoků na základě signatur je v prví řadě zachytávání provozu v monitorovacím prostředí pomocí příslušných nástrojů. Jeho obsah je pak porovnáván s vytvořenou databází signatur. V případě shody s některým záznamem je vysláno upozornění. Zmíněný postup je zobrazen na obrázku 2.4.

Pro detekci anomálií prvotní krok probíhá stejným způsobem, avšak je srovnáván s vytvořenou základní linií chování sítě. Pokud zachycený provoz se od této linie odchyluje, tak je vysláno upozornění a současně je tato linie aktualizovana, viz obrázek 2.5.



Obrázek 2.4 Detekce dle signatur [30]



Obrázek 2.5 Detekce na základě anomálií [30]

Přesněji, detekční systémy mohou provádět detekci pomocí již zmíněných signatur, tj. zachycený provoz porovnávají se známými a nastavenými indikátory, mezi které mohou například patřit specifické hashe souborů, škodlivé domény, škodlivé IP adresy, známé sekvence, známé hlavičky mailů nebo charakteristické chování předcházející útoku [27]. Jde o popsané události, které by mohly vést ke zneužití systému [21]. Po nalezení shody je vytvořeno upozornění pro správce sítě a událost je zapsána do záznamu. [21, 26]

Rozpoznávání provozu pomocí signatur je vhodné v případě detekování známých útoků a vzorů chování. Nevýhodou je větší míra falešně pozitivních upozornění a vysoké nároky na výpočetní výkon při zpracování dat. Současně vysoké nároky na paměť a rozsáhlost databáze, která v některých případech může obsahovat i několik variant stejného útoku.[26, 21]

Další možností pro detekci, je prvotní realizace monitoringu sítě, kde je zjištěno obvyklé chování sítě a nastavena tzv. základní linie, která pak slouží jako výchozí stav pro detekci odchylek, anomálií a neobvyklého chování. Po vytvoření modelu chování je ještě potřebné natrénovaní modelu pro snížení falešně pozitivních detekcí. V tomto případě pak každý předešle zmíněný stav je detekován jako hrozba a je odesíláno upozornění. [26, 21]

Anomálie mohou být detekovány pomocí statistických, data-miningových metod nebo pomocí strojového učení [30].

Třetí možností je detekce anomálií v protokolech, tj. ve způsobu použití nebo zneužití. Rozpoznávání je založeno na znalosti protokolu, například známé specifikace. Výhodou této metody je, že nepotřebuje aktualizaci databáze.[26]

Rozeznávání pomocí anomálií detekují neznámé nebo nové útoky [26, 21]. Místo databáze, pro rozeznání útoku používají strojové učení. Mezi anomálie lze zařadit: pokus o přihlášení uživatele mimo pracovní dobu nebo nové neregistrované zařízení v síti. Jejich nevýhodou je časové a výpočetní nároky na natrénování modelu pro základní linii sítě. [27]

2.3.3 Detekce dle zpracování dat

Provoz lze rozpoznávat pomocí: signatur, natrénovaných modelů pomocí strojového učení, neuronových sítí nebo pomocí fuzzy logiky [28, 29].

V případě signatur je zachycený provoz porovnáván s pevně nastavenými pravidly, které jsou nadefinovány na základě znalosti známých útoků. Toto zpracování odpovídá metodě **signature-based**.

Modely založené na strojovém učení jsou natrénované pomocí speciálních datasetů nebo předem zachyceného provozu, které pak slouží k nastavení základního modelu [28].

Modely založené na neuronových sítích jsou vytvořené obdobným způsobem, jak modely založené na strojovém učení, avšak jsou určeny pro tvorbu sofistikovanějších IDS systémů [28].

Posledně zmíněné modely nebo signatury využívají fuzzy logiky, tj. vzory útoků nejsou jednoznačně popsány, ale místo toho využívají volnějších popisů útoků. Z toho

důvodu také mají schopnost detekovat narušení a škodlivé chování i v přítomnosti nejistých údajů [29].

Vyjma detekce dle signatur jsou ostatní metody používány při detekci anomálií.

II. PRAKTICKÁ ČÁST

3 MOŽNÉ ZDROJE DAT PRO TVORBU REPUTAČNÍ DATABÁZE

V této kapitole budou uvedeny nejvhodnější volně dostupné zdroje z nalezených zdrojů, které mohou posloužit pro tvorbu reputační databáze IP adres[1)]. Z těchto zdrojů pak byly pro tvorbu reputační databáze vybrány, ty které umožňovaly automatické stažení dat z webové stránky, buď přes API nebo jiným vhodným způsobem[2)].

Níže uvedené zdroje lze rozdělit do tří kategorií:

- IP adresy získané pomocí honeypotů,
- IP adresy získané nahlášením,
- IP adresy zařazené na blacklisty nebo blocklisty.

3.1 CZ.NIC

Mezi volně dostupné zdroje dat vhodné pro tvorbu reputační databáze IP adres lze také zařadit data z honeypotů, přesněji data získaná pomocí služby Honeypot as a Service od *CZ.NIC*.

Jak již bylo napsáno v teoretické části, každý přístup na honeypot je považován za nelegitimní, tj. IP adresu lze okamžitě považovat za škodlivou.[31]

Sběr dat je realizován od roku 2017 až po současnost. Data jsou dostupná na stránce *https://haas.nic.cz/stats/export/* pro každý jednotlivý rok, měsíc a den ode dne sběru. Pro každý den je vytvořen soubor typu *.gz*, který po rozbalení obsahuje soubor typu *.json*. Tento soubor pak přímo obsahuje sesbíraná data. Jako příklad lze uvést: IP adresu, která přistupovala na honeypot, datum a čas záznamu, geografické umístění adresy a spuštěné příkazy.[31]

Data je možné stáhnout pomocí nástroje *wget* nebo *curl*. Pro stažení dat pomocí nástroje *wget* je na stránkách *CZ.NIC* uveden příslušný příkaz.

3.2 AbuseIPDB

Stránka *AbuseIPDB* slouží pro sběr informací o IP adresách, které vykazují podezřelé nebo škodlivé chování. Stránka na jednu stranu umožňuje nahlášení takových IP adres, na druhou pak umožňuje získání informací o nich. Mezi to patří kontrola IP adresy vůči databázi nebo prohlížení statistik o nejčastěji nahlašovaných IP adresách.[32]

Zmíněné funkcionality lze využít přímo na webové stránce nebo přes volně dostupné API, které je dostupné v různých formátech pro programovací jazyky: Python, PHP

[1)] Pro lepší srovnání získaných informací z různých zdrojů bude použita jedna konkrétní IP adresa.
[2)] Vybrané zdroje dat využité v aplikaci budou uvedeny v další kapitole.

a VBScript. Rozhraní je možné využívat až po registraci na stránkách *abuseipdb.com*, po obdržení klíče a po stažení tzv. *api-endpoint*.

Vyjma výše uvedených funkcionalit jsou dostupné i jiné možnosti, které závisí od výběru dostupných verzí pro vyhledávání dat. Primární verzí pro práci s databází *AbuseIPDB* je *Individuální* verze, která je přidělena ihned po registraci. Dále pak jsou dostupné další tři varianty: *Základní, Prémiová* a *Podniková*. Následující rozdělení obsahuje bližší popis zmíněných verzí:

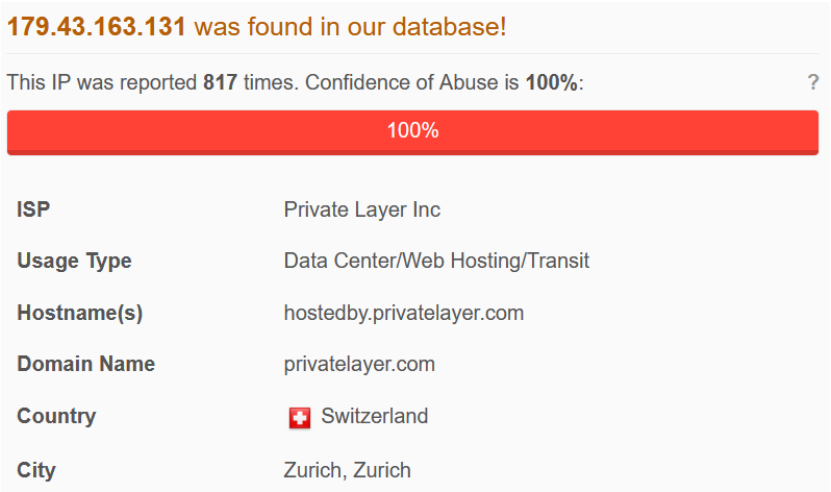
- **Individuální**, bezplatná, verze umožňuje 1000 dotazů na databázi za jeden den, kontrolu 100 IP rozsahů za den nebo 5 dotazů na blacklist za den.
- **Základní** verze již umožňuje realizovat 10 000 dotazů za stejný časový interval jako *individuální* varianta. Vůči bezplatné verzi, lze také zkontrolovat více IP rozsahů. Varianta také umožňuje trasování vymezeného počtu vybraných IP rozsahů.
- **Prémiová** verze již nabízí možnost vyhledat 50 tisíc IP adres a zkontrolovat 5 000 rozsahů. Současně nabízí trasování většího množství IP rozsahů.
- Poslední verzí je **Podniková** varianta, která je určena pro poskytovatele internetového připojení nebo pro jiné podniky. Umožňuje přímý přístup k datům v databázi a je dostupná až po domluvě.

Obrázek níže 3.1 zobrazuje maximální denní limity při používání API rozhraní pro *individuální* variantu. Druhý obrázek 3.2 zobrazuje informace o hledané IP adrese *179.43.163.131*. Sem patří informace o dodavateli internetového připojení, typu využití, krajiny původu nebo doménového jména přidruženého k IP adrese. [33]

Your APIv2 Daily Request Limits

Endpoint	Usage / Daily Limit	Utilization Rate
check	0 / 1,000	0%
reports	0 / 100	0%
blacklist	0 / 5	0%
report	0 / 1,000	0%
check-block	0 / 100	0%
bulk-report	0 / 5	0%
clear-address	0 / 5	0%

Obrázek 3.1 Maximální denní limity (individual) - stránka abuseIPDB.com



Obrázek 3.2 Základní informace o hledané IP adrese - stránka abuseIPDB.com

3.3 AlienVault

Dalším možným zdrojem informací je stránka *AlienVault* - *https://otx.alienvault.com/*. Vyjma základních informací o IP adrese, jsou navíc uvedeny:

- otevřené porty zařízení s touto IP adresou,
- další zdroje, kde byla adresa zmíněna,
- nebo jestli byla detekována jako hrozba.

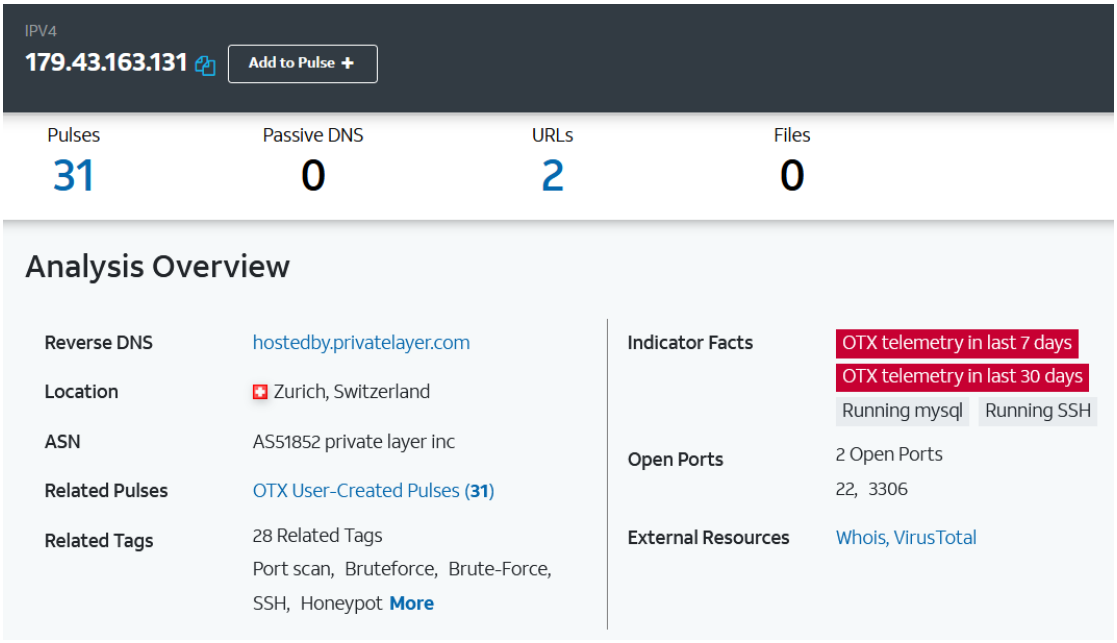
Dále jsou k ní přiřazeny tzv. pulzy, které byly vytvořeny uživateli a blíže popisují činnosti IP adresy. Obrázek 3.3 níže zobrazuje dostupné informace v databázi *AlienVaultu* pro hledanou IP adresu.[34]

Pro externí připojení k databázi, *AlienVault* nabízí připojení přes IDS systémy nebo pomocí API rozhraní pro programovací jazyky Java, Python a Golang. Po registraci je přidělený klíč, který lze použít k API rozhraní.[35]

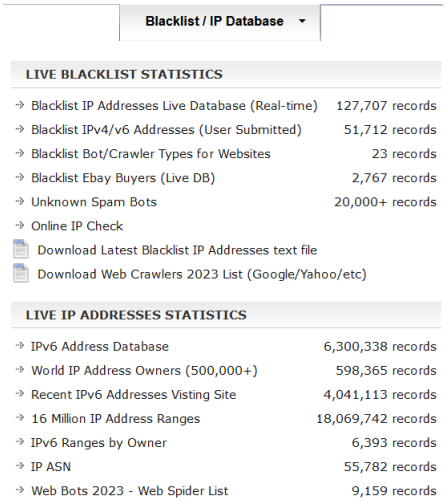
3.4 Myip.ms

Následujícím zdrojem dat je databáze webové stránky *Myip.ms*. Databáze obsahuje až 9 bilionů IP adres pro IPv4 a IPv6. Také obsahuje 377 milionů doménových jmen. Pro IPv6 existují rozšiřující kritéria pro vyhledávání. Vyjma informací o IP adresách a lokálním blacklistu, stránka eviduje i adresy botů, které jsou rozděleny dle vyhledávače.

Po zadání příslušné IP adresy je možné nalézt základní informace: o krajině původu spolu s mapou, o majiteli, o hostované webové stránce, o datu vytvoření záznamu nebo o rozsahu IP adres, do kterého hledaná adresa patří. Zmíněné informace jsou



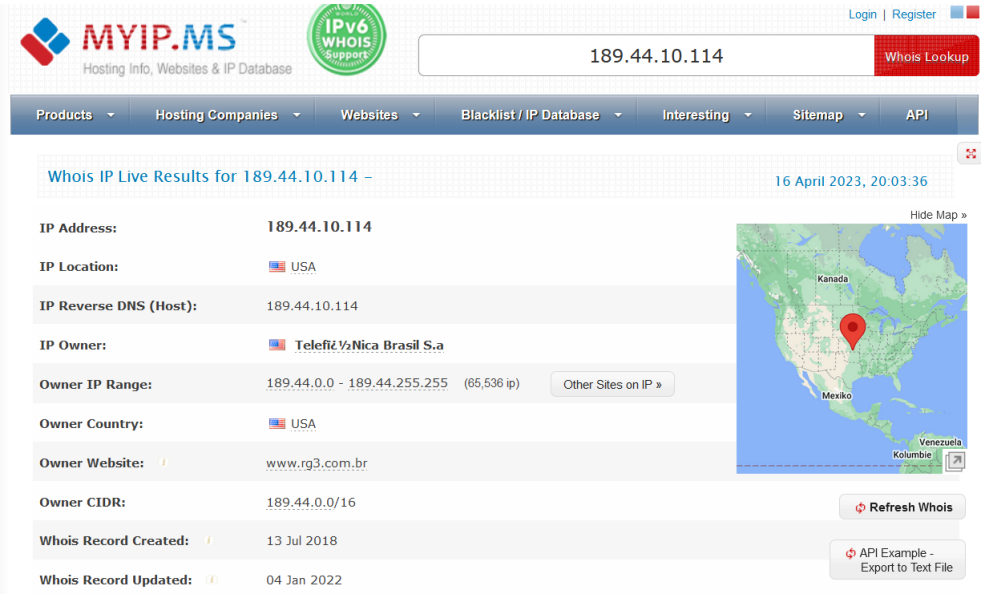
Obrázek 3.3 Základní informace o hledané IP adrese - stránka otx.alienvault.com



Obrázek 3.4 Vyhledatelné informace o IP adresách - stránka myip.ms

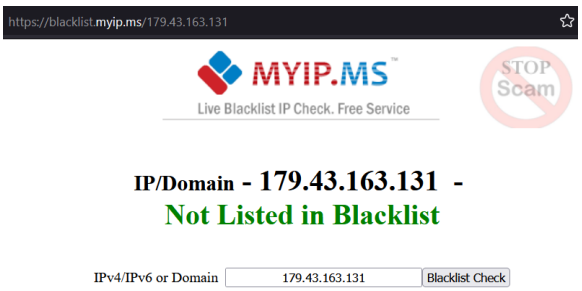
zobrazeny na obrázku 3.5. Dále je možné nalézt dodatečné informace, jako například: umístění IP adresy na blacklistu, přesnější informace o geografickém umístění IP adresy, kontaktní osoba a její kontaktní údaje nebo zdroje odkud byly získány informace o hledané IP adrese. Současně je ověřovaná IP adresa je současně kontrolovaná i vůči blacklistům, které tato stránka využívá.[36]

Stránka navíc nabízí dostupný blacklist, který je denně aktualizovaný a volně stažitelný, buď přímo ze stránky pro vyhledávání v blacklistu, pomocí nástroje *wget* nebo pomocí vestavěných knihoven jazyka Python.[36] Ve výchozím stavu je v blacklistu možné vyhledávat rozsah IP adres, ke kterým lze přidat další kritéria, jako typ hrozby, krajinu nebo vymezený časový interval hledání. Pro zkontrolování samostatné



Obrázek 3.5 Základní informace o hledané IP adrese - stránka myip.ms

IP adresy stránka nabízí samostatný formulář, který je dostupný na stránce pro vyhledávání v blacklistu. Tato možnost vyhledávání je zobrazena na obrázku 3.6.[36]



Obrázek 3.6 Vyhledávání v blacklistu - stránka myip.ms

Vyjma vyhledávání IP adres, stránka poskytuje statistické informace o webových stránkách, které jsou registrované u webhostingových serverů [37]. Dostupná data nebo statistické informace si lze stáhnout ve formě Excel tabulek nebo je možné si vyžádat vlastní sestavu informací.

Mezi statistické údaje patří například: seznam webových stránek, které si v období 2014 až 2023 změnili svou IP adresu, informace o DNS serverech ve světě nebo seznam IP adres na blacklistu. Některé tabulky jsou volně stažitelné, jiné jsou dostupné za určitý finanční obnos.[38]

Stránka dále poskytuje informace: o posledních návštěvnících, o jejich geografickém umístění, o operačním systému, o prohlížeči a o časovém pásmu.[39, 40, 41, 42]

3.5 Databáze Whois

Dalším zdrojem dat je databáze *Whois*, která je po instalaci stejnojmenného terminálového nástroje dostupná na operačním systému Linux a Windows. Poskytuje obdobné informace jako služba *WhoisXML API*, která bude popsána níže.

Vyjma terminálového nástroje *whois* je možné IP adresy vyhledávat i pomocí obdobné služby. Je jím webová stránka *WhoisXML API*, která přímo nabízí dostupné API rozhraní pro získání informací o IP adrese. Stejně jak u databáze *AbuseIPDB* uživatel obdrží klíč až po registraci, který je v tomto případě vložen do https odkazu spolu s příkladem pro použití[3)]. Vyjma IP adresy verze 4 a verze 6 je možné vyhledat i organizaci nebo údaje o autonomním systému.

Odkaz lze použít v prohlížeči, kde je možné informace stáhnout ručně nebo ho lze použít přímo v kódu a stáhnout data pomocí knihoven daného jazyka. V obou případech jsou data stažena ve formátu *json*.

WhoisXML API nabízí 4 základní možnosti pro vyhledávání:

- *Bezplatná* - nabídka 1000 dotazů za měsíc,
- *Jednorázová* - poplatek je určen na základě počtu dotazů,
- *Měsíční* - na základě měsíčního poplatku je přidělen vymezený počet možných dotazů za měsíc,
- *Roční* - poplatek je účtován stejný způsobem jak u měsíční možnosti s tím, že je také vymezen počet dotazů na jeden měsíc.

V každém případě nevyužité dotazy nelze přenášet do dalšího období. Veškeré informace byly čerpány ze zdroje [50].

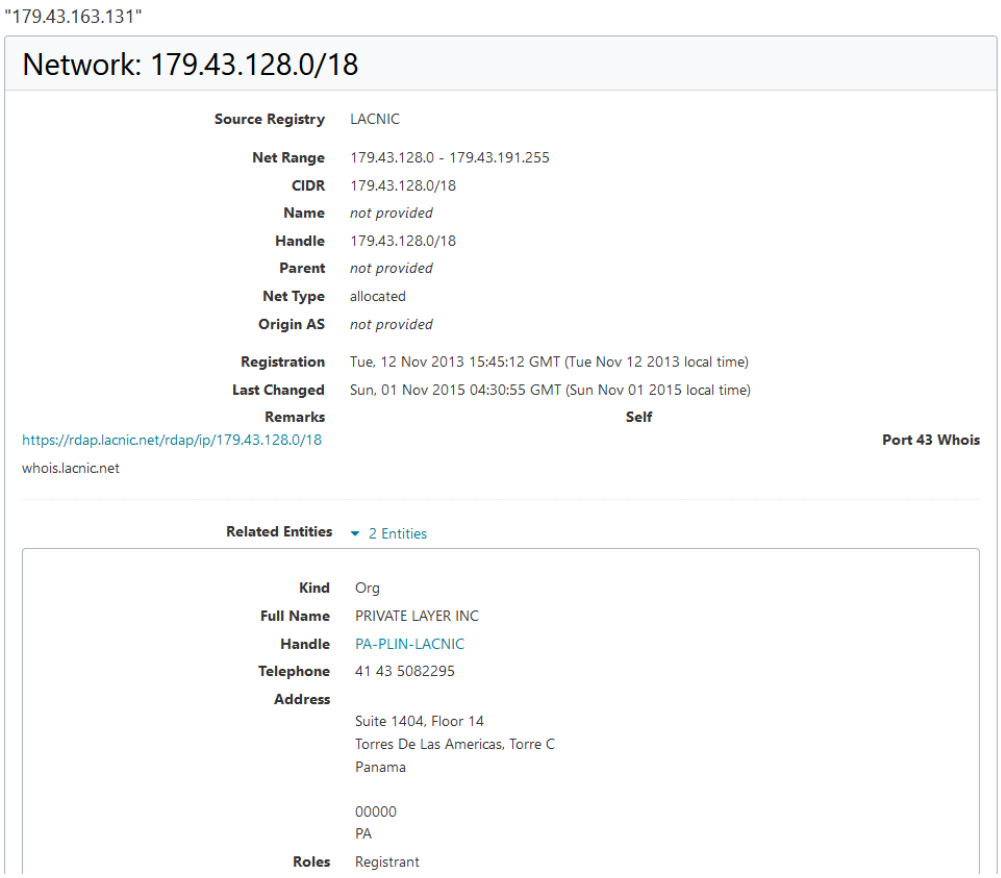
Součástí databáze *Whois* a také dalším možným zdrojem dat je i regionální internetový registr ARIN, který se zaměřuje na přidělování adresního prostoru a přidělování čísel pro autonomní systémy v oblasti Severní Ameriky.[46]

Vyjma adresních rozsahů, doménových jmen, IP adres, stránka umožňuje vyhledávat údaje o číslech autonomních systémů a o organizacích.

Například pro IP adresu lze získat podrobné informace z pohledu její registrace, tj. jménu regionálního internetového registru, IP rozsah, do které adresa patří, datum registrace a kontaktní osoba. Tyto informace jsou zobrazeny na obrázku 3.7.[46]

Po registraci je možné získat API klíč nebo pro vyhledávání lze použít globální databázi *Whois*, pro kterou je na stránkách *ARINu* uveden postup pro vyhledávání.

[3)] Obdržený https spolu s klíčem: *https://ip-netblocks.whoisxmlapi.com/api/v2?apiKey=<API klíč>&ip=8.8.8.8*



Obrázek 3.7 Získané informace o IP adrese - stránka arin.net

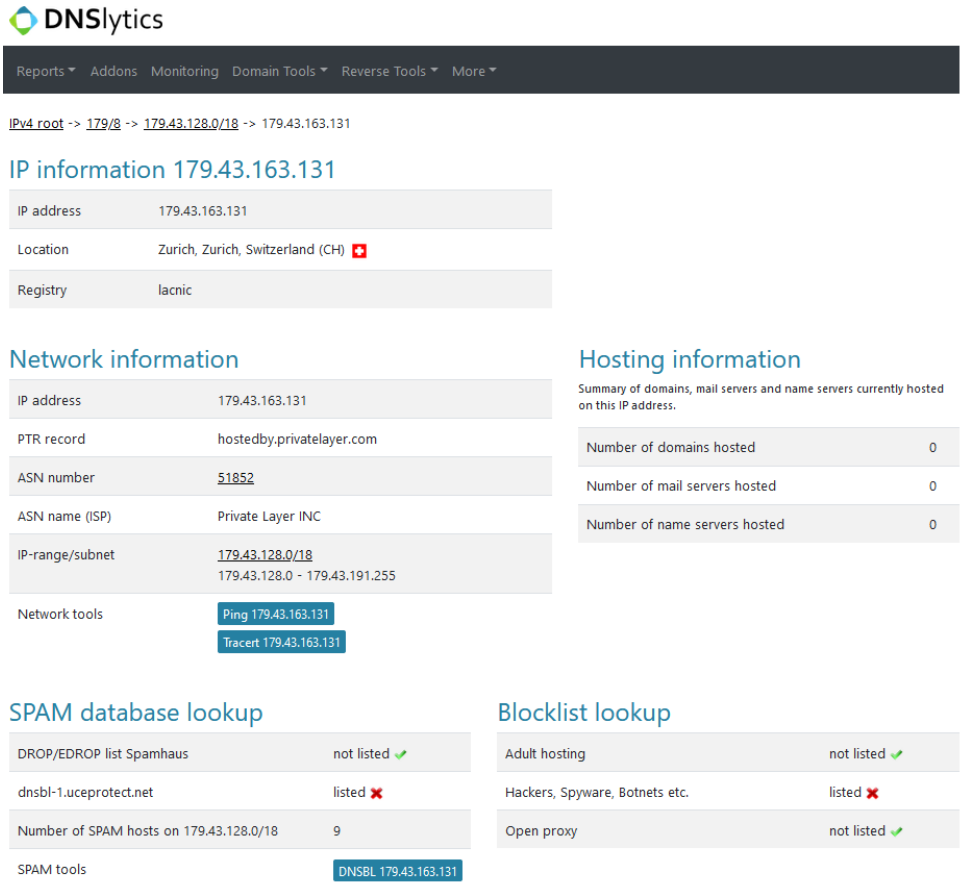
3.6 DNSlytics

Webová stránka *DNSlytics.com* nabízí možnost vyhledávání údajů: o IP adresách pro verzi 4 i pro verzi 6, o doménových jménech, o autonomních systémech a o mailových serverech. Dále je nabízena možnost vyhledávání registrovaných i smazaných domén, podobně znějících nebo podobně napsaných domén nebo historických záznamu pro konkrétní IP adresu nebo pro DNS server. Stránka nabízí poskytuje i informace pro vyšetřování a proti podvodům. Dalšími nástroji pro vyhledávání jsou: DNS lookup, test emailu, IP Geo lookup, Ping, SPF Lookup, DNSBL Lookup a trasování.[43]

Informace lze vyhledávat přímo na webové stránce nebo v případě potřeby lze využít placené API na 30 dní nebo na celý rok. U API na 30 dní je možný přístup pouze ke 100 stránkám. API na celý rok ještě nabízí dvě různé varianty, které se navzájem liší v počtu přístupů. Omezenější verze umožňuje přístup k 200 stránkám za den a plná verze umožňuje přístup k 1000 stránkám za den. Celoroční varianty nabízí i funkce: reverzní vyhledávání, vyhledávání domén nebo monitoring.[44]

Za využití možnosti vyhledávání IP adres byl získán následující obrázek 3.8, který zobrazuje informace o geografickém umístění IP adresy, místo registrace IP adresy, umístění v blacklistech nebo blocklistech nebo číslo autonomního systému, kde se IP ad-

resa nachází.[43]

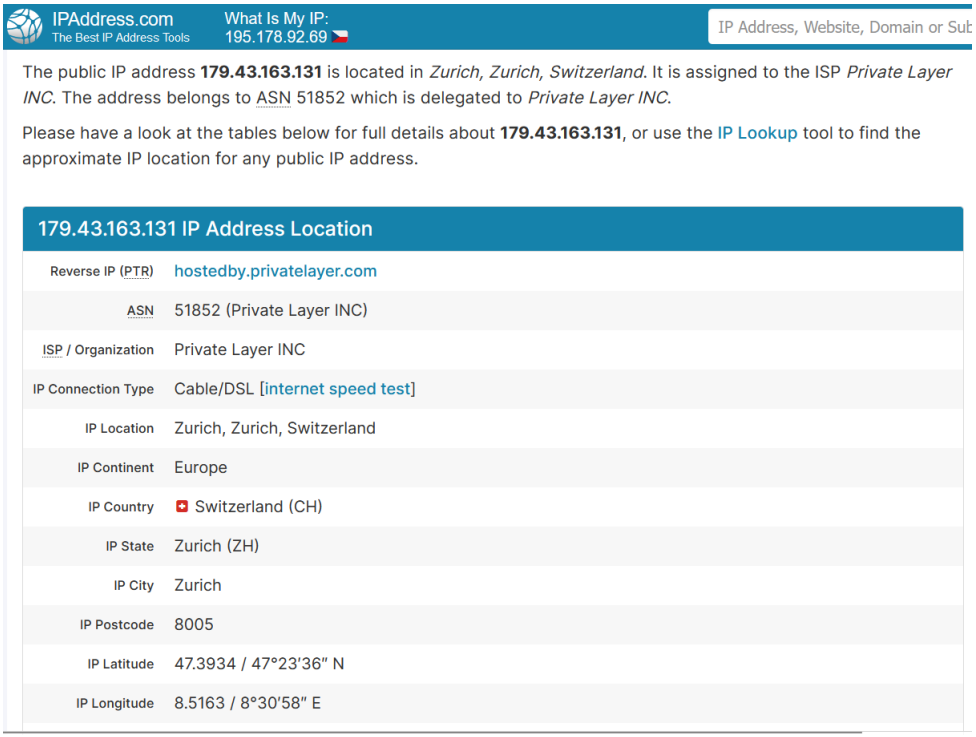


Obrázek 3.8 Získané informace o IP adrese - stránka dnslytics.com

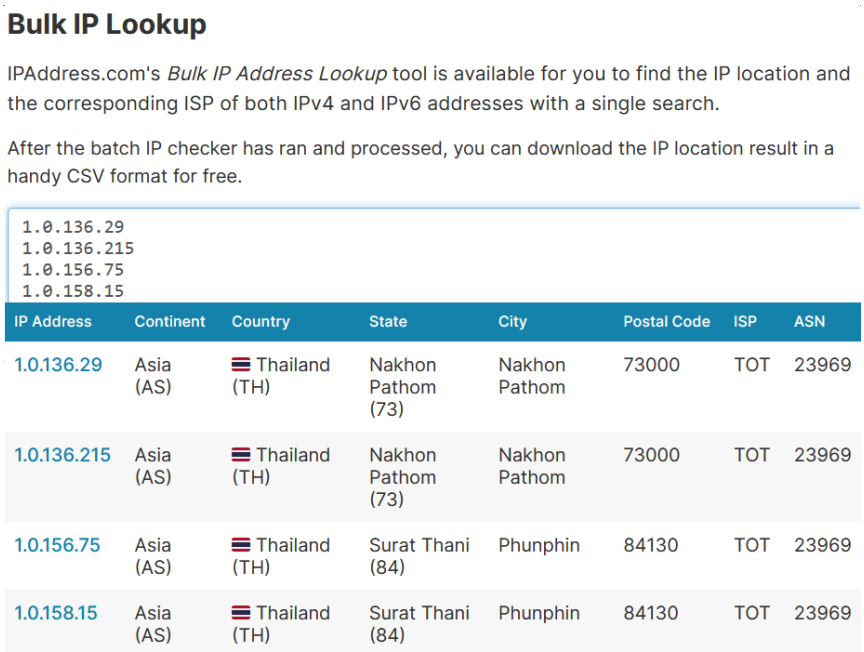
3.7 IPAddress

Databáze webové stránky *IPAddress.com* poskytuje obdobné informace o hledané IP adrese jak předešle zmíněné zdroje, viz obrázek 3.9. Navíc, ale poskytuje možnost vyhledávání údajů o více IP adresách najednou, obrázek 3.10. V oblasti dalších služeb jsou poskytovány podobné služby jak u stránky *DNSlytics*. [45]

Stránka poskytuje bezplatné API. Nicméně API je určeno pro získávání údajů o IP adrese účastníka nebo k zjištění vlastní IP adresy. Přesněji, údaje jsou spojeny s přesným nebo přibližným geografickým umístěním.[45]



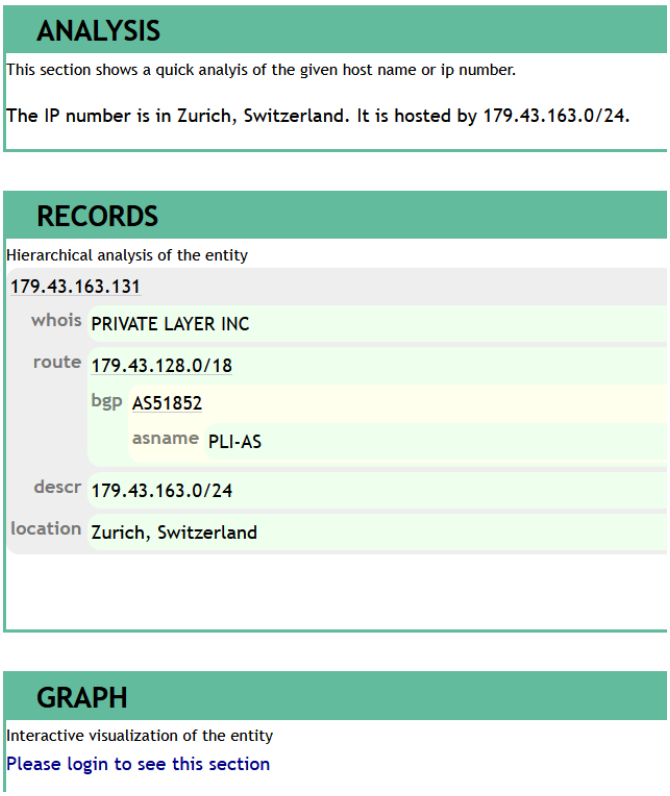
Obrázek 3.9 Získané informace o IP adrese - stránka ipaddress.com



Obrázek 3.10 Získané informace o více IP adresách - stránka ipaddress.com

3.8 Robtex

Předposlední webová stránka *Robtex.com*, ve své základní verzi poskytuje omezené informace o vyhledávané IP adrese, doménovém jménu nebo číslu autonomního systému. Po přihlášení přes Google účet jsou zobrazeny další informace IP adrese, viz obrázek 3.11. Dále je dostupné bezplatné API s omezeným počtem požadavků.[47]



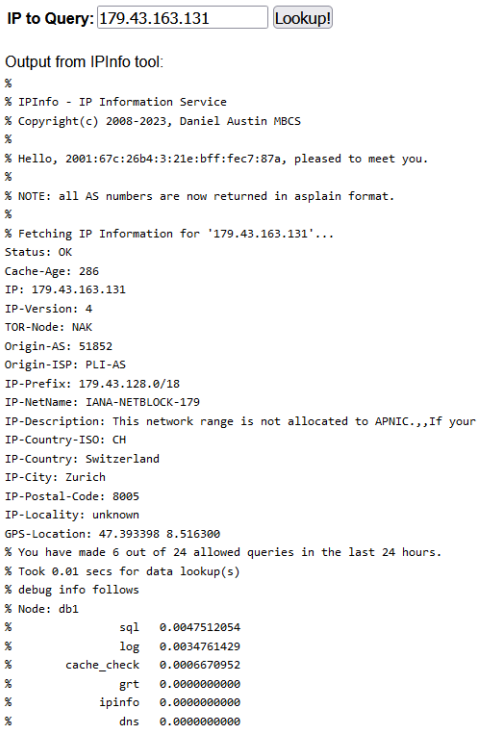
Obrázek 3.11 Získané informace o IP adrese - stránka robtex.com

3.9 Dan.me.uk

Webová stránka nabízí služby v oblasti IP adres. Vyjma záznamů o IP adresách, poskytuje údaje o:

- přiřazených IP adresách v jednotlivých autonomních systémech,
- zařazených IP adresách do blacklistů,
- informace o účasti IP adresy jako TOR uzlu,
- kalkulačku pro rozdělení IP rozsahu,
- seznam uzlů zařazených do TOR sítě,
- kontrolu webových serverů.[51]

Následující obrázek 3.12 zobrazuje nalezené informace o hledané IP adrese. Hledaná IP adresa je patří do verze 4, nachází se v autonomním systému *51852* a patří do sítě s názvem *IANA-NETBLOCK-179*. Dále je uvedena krajina, město nebo přesná geolokace.[51]



Obrázek 3.12 Získané informace o IP adrese - stránka dan.me.uk

3.10 CSIRT.CZ

Podkapitola obsahuje vybrané zdroje blacklistů a blocklistů ze stránek *CSIRT.CZ*. Zdroje byly vybrány dle vhodnosti využití při tvorbě praktické části.

Seznamy lze stáhnout pomocí nástroje *wget* nebo pomocí knihoven v jazyce Python.

- **AlienVault** - jde o blacklist, který je součástí Open Threat Exchange a obsahuje škodlivé IP adresy nahlášené z různých zdrojů, například od uživatelů AlienVaultu nebo ze strany výzkumníků.
- **CI Army** - IP adresy v blacklistu pochází z databáze CINS, která získává informace z příslušné sítě a jiných důvěryhodných zdrojů. IP adresy mohou být do seznamu zařazeny například na základě špatného hodnocení tzv. Rogue Packet.
- **Danger Rulez** - seznam obsahuje IP adresy, které přesáhly určitý počet pokusů o přihlášení přes SSH.
- **BlockList.DE** - zdroj pro několik blocklistů na různé služby. Mezi tyto služby patří: Mail a Postfix, SSH, Apache, Apache-DDOS a RFI-Attacks, IMAP, FTP, RFI-Attacks, REG-Bots, IRC-Bots nebo BadBots, SIP-, VOIP- a Asterisk-. Tento zdroj také obsahuje blocklist pro IP adresy, které použily metodu útoku pomocí hrubé síly pro získání vstupu na CMS Joomla, Wordpress nebo login na jiné

webové služby. Současně nabízí seznam IP adres, které realizovaly více než 5000 útoků před 2 měsíci.

3.11 Binary defense

Společnost *Binary defense* poskytuje služby v oblasti kybernetické bezpečnosti a na svých stránkách umožňuje přístup k volně stažitelnému blocklistu, který nesmí být použit komerčně. [49]

4 TVORBA REPUTAČNÍ DATABÁZE PRO IP ADRESY

Kapitola se v první řadě zabývá návrhem aplikace pro sběr informací z OSINT zdrojů. Druhá část kapitole již popisuje praktickou implementaci samotného návrhu.

4.1 Návrh aplikace pro sběr informací z OSINT zdrojů

Podkapitola obsahuje popis dílčích kroků, které vedly k vytvoření konečného návrhu aplikace. Dále pak obsahuje finální vývojový diagram návrhu, na základě kterého byly vytvořeny jednotlivé součásti a funkce celé aplikace.

4.1.1 Stručný popis návrhu aplikace

Cílem práce bylo vytvoření reputační databáze pro IP adresy, která bude využívat data z volně dostupných zdrojů. Pro výběr těchto zdrojů byly vybrány zdroje uvedené v předešlé kapitole, na základě kterých bylo potřebné navrhnout způsob pro jejich stažení, zpracování, vhodného uložení a pro průběžnou aktualizaci. Do návrhu bylo také zahrnuto vyhledávání v databázi a vhodná reprezentace získaných informací.

Před samotnou implementací návrhu byly vybrány konkrétní cílové platformy a stejně tak byl vybrán vhodný programovací jazyk. Primárním cílovým operačním systémem byl určen operační systém Ubuntu 22.04 a sekundárním operačním systémem byl vybrán operační systém Windows 11. Pro tvorbu programu byl zvolen programovací jazyk Python verze 3.10+. Jako úložiště zpracovaných dat byla zvolena databáze *sqlite3*.

4.1.2 Výběr zdrojů dat pro tvorbu reputační databáze

Primárním zdrojem dat byla určena data z honeypotů, které aplikace získává ze stránek CZ.NIC. Druhým zdrojem dat byly vybrány blacklisty a blocklisty (dále jen externí data) ze stránky *myIP.ms* a ze zdrojů uvedených v podkapitole *CSIRT.CZ*.

Třetím zdrojem dat byly zdroje pro získání dodatečných, zpřesňujících, informací o IP adresách. Byly zvoleny databáze *AbuseIPDB*, *whois*, *ipinfo.dan.me.uk* a *WhoisXMLAPI*.

Výše zmíněné zdroje byly zvoleny na základě nejvhodnější možnosti implementace v jazyce Python, a to dle dvou nezávislých kritérií:

- Prvním kritériem byla možnost stažení dat pomocí dostupných nástrojů. Například pomocí terminálových nástrojů *wget* nebo *curl*, nebo pomocí knihoven jazyka Python.
- Druhým kritériem byla dostupnost API pro jazyk Python.

Stažení a průběžná aktualizace dat

Dalším krokem bylo určení způsobu stažení dat pro vybrané zdroje.

Pro stažení dat z honeypotů byly zvoleny nástroje používané přes příkazovou řádku. Pro operační systém Ubuntu 22.04 byl zvolen způsob, který je uveden na stránkách CZ.NIC, tj. pomocí nástroje *wget*. Naopak pro operační systém Windows byl vybrán obdobný způsob stažení dat, ale za pomocí nástroje *curl*. Nástroj byl vybrán, protože je součástí operačního systému a není potřebné ho dodatečně stahovat a instalovat, případně nastavovat.

Jelikož jde o data, která jsou opakovaně rozšiřována, tak bylo nutné nastavit časovače pro jejich aktualizaci a mechanismus pro opětovné stažení a případné smazání již zpracovaných dat.

V obou případech, pro oba operační systémy, jsou data vždy pro aktuální rok stažena jako celek a následně jsou promazána dle stavu jejich zpracování. Například, pokud již byly zpracovány první tři měsíce z celkového počtu měsíců pro daný rok, tak jsou před zpracováním dalších dat tyto tři měsíce smazány.

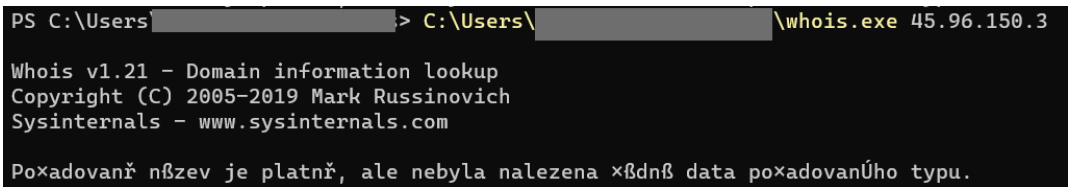
Pro druhou kategorii zdrojů, externí data, byla zvolena možnost stažení pomocí knihoven jazyka Python. Stejně jak u honeypotů byl nastaven časovač pro aktualizaci.

Časovače pro aktualizaci dat z honeypotů i pro externí data byly nastaveny na jeden den, tj. data jsou aktualizována každý den.

Stažení dodatečných informací z *AbuseIPDB* a *whoisXML API* jsou realizovány přes nabízená API rozhraní. Pro zbylá data, *whois* a *ipinfo.dan.me.uk*, je používán nástroj *whois*.

Je nutné zmínit, že zdroje *Whois* a *ipinfo.dan.me.uk* jsou dostupná pouze pro operační systému Ubuntu, z důvodu nefunkčnosti nástroje *whois* na operačním systému Windows, který je zobrazen na obrázku 4.1.

Stažení dat ze zdrojů pro dodatečné informace je uskutečňováno příležitostně, tj. v případě vyhledávání informací o konkrétní IP adrese zadané uživatelem.



Obrázek 4.1 Nástroj whois na operačním systému Windows

4.1.3 Zpracování a uložení dat

Po navržených mechanismech pro stažení byly vytvořeny mechanismy i pro zpracování dat.

Z důvodu velkého množství dat z honeypotů, byl zvolen dávkový způsob zpracování[1)].

První úrovní bylo nastavení zpracovávání dat po jednotlivých měsících. Výjimka byla vytvořena pouze pro rok 2017 s nejmenším množstvím dat, které je možné zpracovat najednou. Druhou úrovní byl návrh časovače, který automaticky spouští zpracování dat po osmy hodinách.

Před zpracováním byly zvoleny vhodné mechanismy pro extrakci souborů typu *.gz* na formát *.json* pomocí knihovny *gunzip*. Soubor JSON je následně jako vstupní parametr vložen do vláken, které pak zodpovídají za extrakci relevantních informací a za uložení těchto dat do tabulky *honeypotData* ve vytvořené databázi. Počet vláken a délka běhu záleží od velikosti zpracovávaného souboru.

Během návrhu se také počítalo s průběžným mazáním již zpracovaných souborů. Například po zpracování příslušného měsíce byla složka spolu s již zpracovanými soubory smazána.

Pro zpracování externích dat bylo zvoleno okamžité zpracování ihned po jejich stažení. Pro jednotlivé typy souborů byly vytvořeny funkce pro extrakci dat. Například proces zpracování pro typ *.php* se liší od procesu zpracování pro typ *.txt*.

Ukládání samotných IP adres je vkládáno do tabulky *externalData*. Pro případy existence i jiných údajů vyjma samotných IP adres je pro tyto údaje volena tabulka *additionalData*.

V tomto případě také probíhá smazání dat po uložení.

Dodatečná data jsou zpracována obdobně jako data z externích zdrojů. Výjimka platí pro data z *WhoisXMLAPI*, kde je odpověď zpracována jak u dat z honepoytů.

Na základě různorodé struktury odpovědí byl navržen jiný způsob extrakce dat pro každý zdroj. Získané informace jsou v následujícím kroku uloženy do tabulky *additionalData*.

4.1.4 Vyhledávání v datech a jejich reprezentace

Pro možnost vyhledávání v databázi a reprezentaci dat bylo vytvořeno společné rozhraní pro oba operační systémy: přes terminál a přes grafické uživatelské rozhraní.

Obě možnosti umožňují vyhledávat IP adresy ve formátu verze 4 a také ve verzi 6. Během vyhledávání jsou prohledány všechny tabulky v databázi a současně je vyslán dotaz pro získání dodatečných informací na zmíněné databáze. Vyslání dotazu je podmíněno neexistencí dodatečného záznamu pro hledanou IP adresu, tj. pokud ještě

[1)] Data mají stromovou strukturu. Jejich sběr je realizován od roku 2017 až do současnosti, tj. data jsou v první řadě rozdělena do složek podle let. Druhou úrovní je rozdělení dle měsíců, ve kterých jsou již umístěny soubory pro jednotlivé dny v měsíci ve formátu *.json.gz*

dotaz pro tuto adresu na tyto databáze proveden nebyl. To znamená, že doplňující informace o hledané IP adrese v lokální databázi prozatím nejsou uloženy.

Po získání výsledků jsou informace zobrazeny:

- pro terminál - jako po sobě jdoucí řádky ve formátu - *parametr: detail*,
- pro GUI - v podobě tabulky se dvěma sloupci, kde první sloupec zobrazuje *parametry* a druhý obsahuje *detaily* k příslušným parametrům.

4.1.5 Pomocné funkcionality

Návrh pomocných funkcí byl realizován z důvodu větší automatizace celkového programu.

První funkcionalitou je kontrola internetového připojení, která byla navržena pro ověření spojení před stažením dat, dotazováním se na externí databáze nebo při instalaci balíčků jazyka Python.

Druhou navazující funkcionalitou bylo navržení mechanismu pro kontrolu nainstalovaných balíčků jazyka Python pro oba operační systémy. Pro operační systém Ubuntu se současně kontroluje i instalace terminálového nástroje *whois*.

Do této části bylo zahrnuto automatizované nastavení časovačů pro oba operační systémy. I když, tvorba časovačů pro oba operační systémy byl navržena pomocí jiných nástrojů, tak obecný postup je stejný.

V obou případech jsou vytvořeny soubory, které slouží pro spuštění jednotlivých bloků programu. Následně jsou tyto soubory provázány se službou pro spouštění úkolů.

U operačního systému Ubuntu byla vybrána služba *systemd*. Pro tvorbu časovačů byla vytvořena šablona, viz 1.

```
[Unit]
Description=Service - {{description}}
After=multi-user.target
[Install]
WantedBy=multi-user.target
After=network.target
[Service]
Type=Simple
User={{login}}
ExecStart=/usr/bin/python3 {{path}}
StandardOutput=syslog
StandardError=syslog
SyslogIdentifier=%n
```

Listing 1 Šablona pro tvorbu časovače

Klíčové slovo *Unit* určuje samotnou službu - její popis a možnosti běhu na účtu každého uživatele v systému. Řádek *After=network.target* určuje zapnutí služby v případě, že je síťové připojení dostupné. *ExecStart* určuje spuštění vybraného modulu. Klíčové

slovo *RestartSec* určuje za jakou dobu se má služba restartovat a řádek *Restart=always* stanovuje, že služba má být restartována vždy. Poslední tři řádky určují místo výpisu stavu a chyb do systémového logu.

U operačního systému Windows byl nastaven obdobný způsob zpracování.

Poslední důležitou pomocnou funkcionalitou byl návrh dvou souborů uchovávající podstatné informace. Prvním z nich je *managerFile.txt* a obsahuje souhrnné informace, jakým je například stav celkového zpracování dat, jejich poslední aktualizace, využitý počet dotazů pro získání dodatečných informací a také uložené klíče k API, viz tabulka 4.1.

Struktura navrženého souboru *managerFile.txt* je uvedena v tabulce 4.1. První sloupec tabulky obsahuje indexy jednotlivých úkolů, které jsou popsány ve druhém sloupci tabulky. Třetí sloupec obsahuje výchozí hodnoty úkolů, které jsou nastaveny ihned po vytvoření souboru. Čtvrtý sloupec, již obsahuje popis hodnot, které mají být postupně nastavovány dle proběhlých činností.

Příkladem je například činnost instalace balíčků a nastavení časovačů, které probíhá najednou. Naopak přidání API klíčů a zapsání počtu vyčerpaných dotazů je zapsáno do souboru až v případě vyhledávání v databázi.

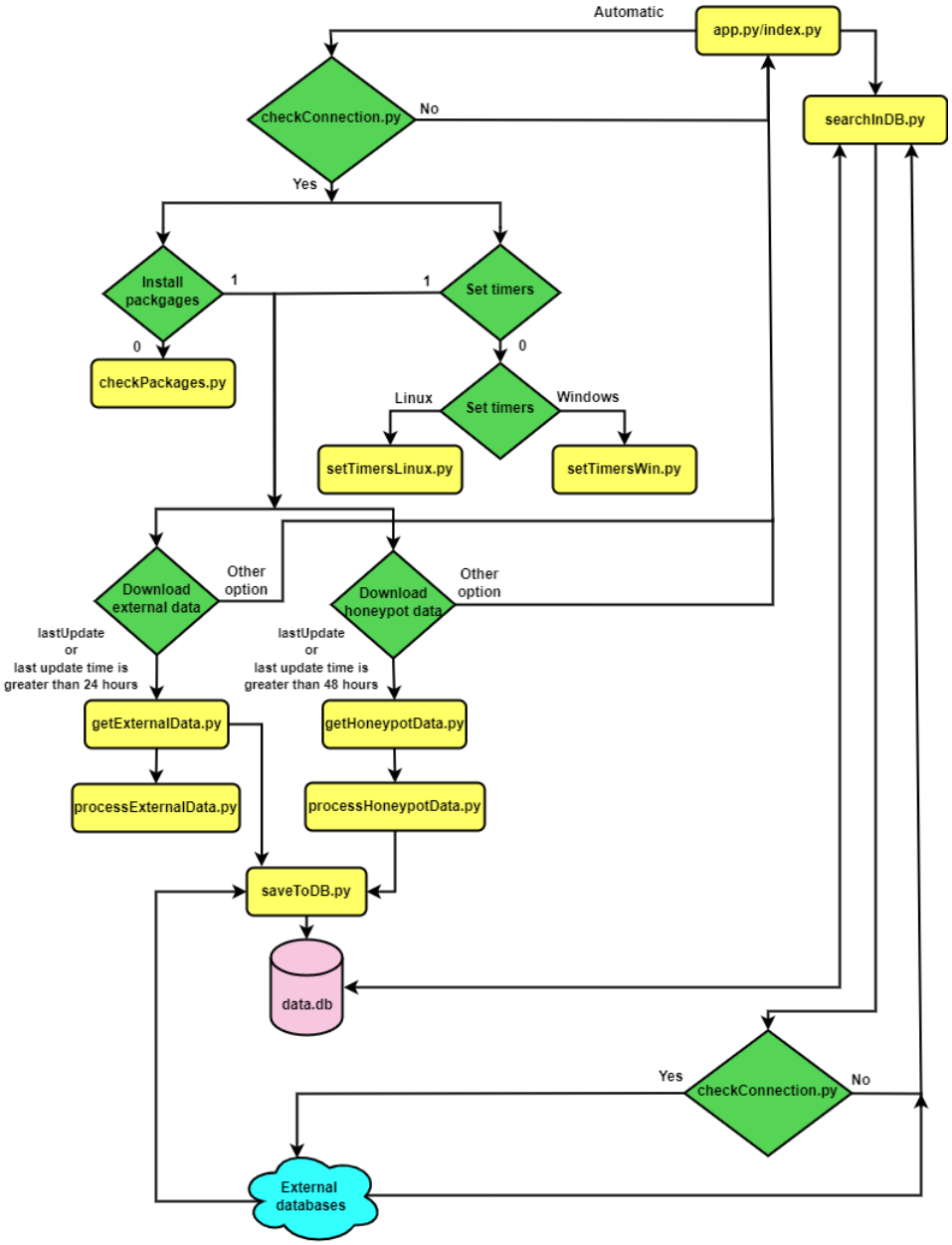
Druhý soubor, *logs.txt*, obsahuje název posledně zpracovaného souboru z dat honeypotů.

Index	Jméno úkolu	Výchozí hodnoty	Nastavené hodnoty
0	Instalace balíčků	0	1
1	První běh	0	1
2	Nastavení časovačů	0	1
3	Externí data - poslední stažení	lastUpdate	Datum a čas
4	Honeypot - poslední stažení dat	lastUpdate	Datum a čas
5	Honeypot - stažení dat předchozí léta	0	1
6	Honeypot - poslední zpracování dat	lastUpdate	Datum a čas
7	Honeypot - nastavení tabulky	0	1
8	Externí data - nastavení tabulky	0	1
9	Doplňující informace - nastavení tabulky	0	1
10	AbuseIPDB - API klíč	0	1
11	Počet dotazů za aktuální den	0	Rozsah 0 až 1000
12	Posledně provedený dotaz	lastUpdate	Datum a čas
13	WhoisXML API - API klíč	0	1
14	Počet dotazů za aktuální měsíc	0	Rozsah 0 až 1000
15	Posledně provedený dotaz	lastUpdate	Datum a čas
16	Honeypot - zpracování dat - aktuální rok	lastUpdate	Datum a čas
17	Již zpracované rok (uložené v DB)	0	2017, 2018,..

Tabulka 4.1 Struktura souboru *managerFile.txt*

4.2 Konečný návrh aplikace

Složením výše zmíněných kroků vznikl finální návrh aplikace, který je zobrazen na obrázku 4.2, kde jsou pomocné funkce zobrazeny zelenou barvou. Ostatní části mají žlutou barvu a patří mezi hlavní části aplikace.



Obrázek 4.2 Vývojový diagram pro tvorbu aplikace

4.3 Implementace návrhu

Podkapitola již obsahuje samotný popis implementace výše uvedeného návrhu. Pro začátek bude uvedeno shrnutí celkového programu, tj. krátkou charakteristiku modulů programu. Dále pak budou moduly popsány jednotlivě.

4.3.1 Souhrnný popis

Jak bylo popsáno v předešlé podkapitole, program sestává z několika bloků, tj. existuje více vzájemně oddělených vstupních bodů.

První část zahrnuje pomocné funkce nastavující časovače pro řízení nebo kontrolující určitý stav. Sem také patří funkce, které jsou současně využívány dalšími dvěma částmi programu. Například jde o funkci pro konverzi dat z jednoho datového typu na druhý.

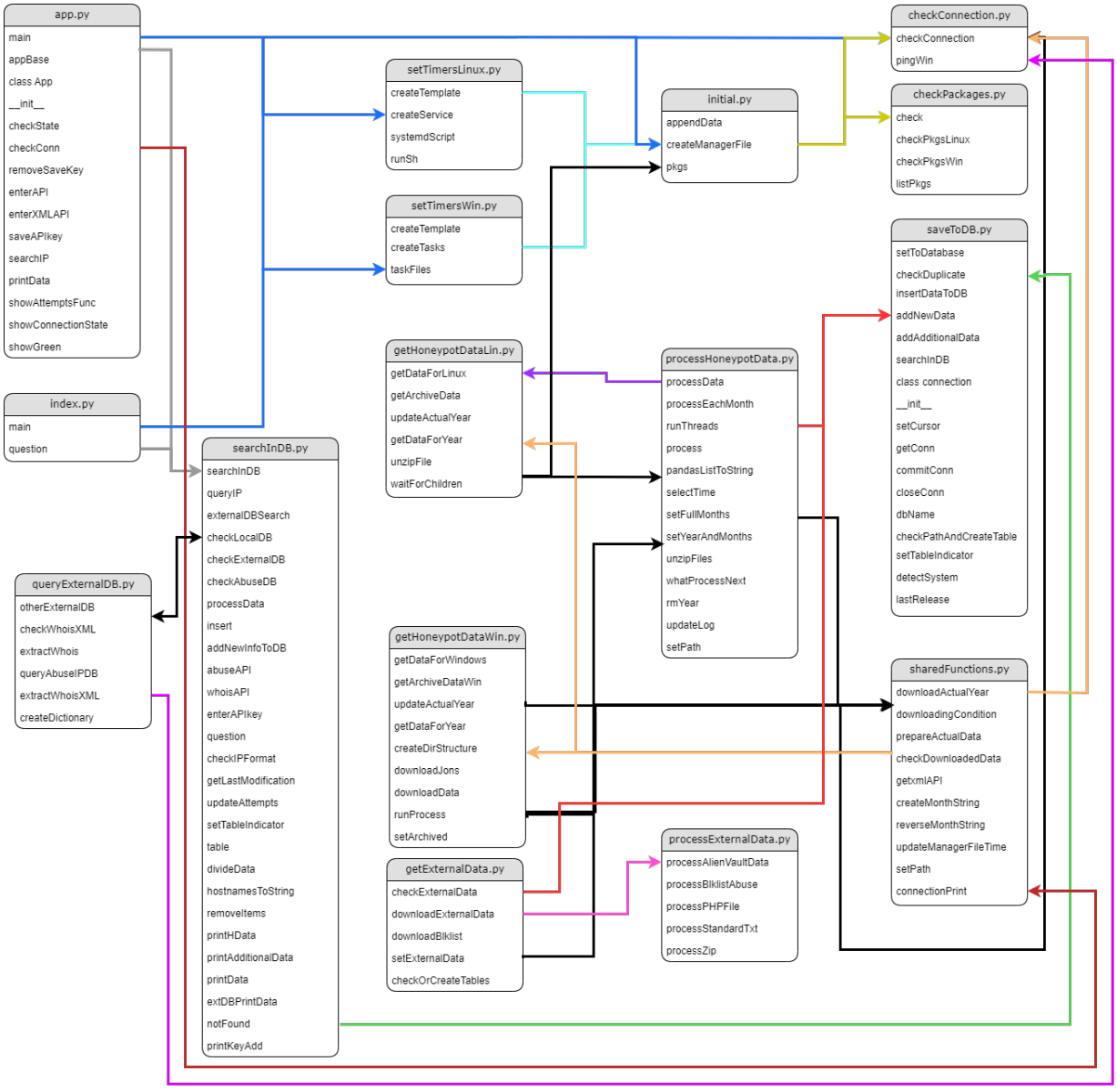
Druhá část obsahuje funkce, které jsou zodpovědné za: stažení, zpracováváni a ukládání získaných informací do tabulek v databázi. Tato část je oddělena od třetí části. Oddělení slouží hlavně pro souvislý běh obou částí.

Třetí část pouze umožňuje vyhledávání, zobrazování informací a případné rozšiřování informací o IP adresách, která je však realizována bez vědomí uživatele a probíhá odděleně od druhé části. Do druhé části není nijak zasahováno pouze je zobrazen aktuální stav stažených a zpracovaných dat, která jsou získána ze souborů: *managerFile.txt* a *logs.txt*.

Obrázek 4.3 níže zobrazuje celkovou strukturu a propojení modulů.

První část obsahuje moduly:

- **initial.py** - modul je spuštěn pouze v případě prvního spuštění a zajišťuje vytvoření souboru *managerFile.txt*, kontrolu nebo případnou instalaci potřebných balíčků.
- **checkConnection.py** - modul slouží pro kontrolu internetového připojení před začátkem samotné činnosti jiného modulu. Například je využíván při stažení dat nebo při dotazování se na externí databáze.
- **checkPackages.py** - modul přímo kontroluje instalované balíčky pro oba operační systémy. V případě potřeby jsou tyto balíčky doinstalovány.
- **setTimersLinux.py** - modul pro nastavení časovačů pro službu *systemd* pro operační systém Ubuntu.
- **setTimersWin.py** - zajišťuje nastavení časovačů pro operační systém Windows.
- **sharedFunctions.py** - obsahuje funkce, které jsou společné pro oba operační systémy nebo jsou využívány různými funkcemi z více modulů.



Obrázek 4.3 Diagram funkcí pro vytvořenou aplikaci

Druhá část sestává z modulů:

- **getHoneypotDataLin.py** - modul pro stažení dat z honeypotů pro operační systém Ubuntu 22.04. Obsahuje funkce pro stažení všech dat a také pro jejich aktualizaci.
- **getHoneypotDataWin.py** - modul pro stažení dat z honeypotů pro operační systém Windows, který obsahuje obdobné funkce jak předcházející modul.
- **getExternalData.py** - modul pro stažení dat z blacklistů a blocklistů ze zvolených zdrojů dat. Obsahuje funkce pro stažení dat a využívá **processExternalData.py** pro okamžité zpracování dat a modulu **saveToDB.py** pro okamžité uložení dat.

- **processHoneypotData.py** - modul pro zpracování dat z honeypotů, který je společný pro oba operační systémy. Extrahuje relevantní informace z dat honeypotů a za využití modulu **saveToDB.py** je ukládá do databáze.
- **processExternalData.py** - modul pro zpracování externích dat, který je také společný pro oba operační systémy.
- **saveToDB.py** - obsahuje funkce pro ukládání dat do lokální databáze. Stejně tak má na starost kontrolu duplikátů v tabulkách. Je využíván i třetí částí, pokud jsou dohledávány dodatečné informace k hledané IP adrese.

Stažení a zpracování dat z honeypotů fungují odděleně. Jsou spuštěny pomocí nastavených časovačů. Naopak stažení a zpracování externích dat je vzájemně propojeno a spuštěno pouze pomocí jednoho časovače.

Poslední, třetí, část obsahuje moduly:

- **app.py** - obsahuje grafické uživatelské rozhraní a příslušné funkce pro jeho obsluhu. Například funkce pro možnost zadávání API klíčů prostřednictvím GUI a barevné indikátory se zbývajícím počtem možných dotazů z celkového počtu za daný časový interval. Většinu funkcí čerpá z modulu **searchInDB.py**.
- **index.py** - ekvivalentní verze **app.py**, ale bez grafického rozhraní, která je přímo propojena s modulem **searchInDB.py**, která již umožňuje vyhledávání v databázi a přidávání nových dat.
- **searchInDB.py** - modul, který slouží jako rozhraní pro vyhledávání informací v lokální databázi nebo externích zdrojích a pro ukládání do dodatečných informací. V případě vyhledávání přes terminál také zajišťuje reprezentaci dat. U GUI jsou data zobrazena pomocí tabulky, pro kterou je vytvořena funkce v modulu **app.py**.
- **queryExternalDB.py** - slouží pro realizaci dotazů na externí databáze a také pro extrakci potřebných informací během vyhledávání v databázi.

Níže budou detailněji popsány výše zmíněné moduly.

4.3.2 Initial.py

Modul využívá knihoven *os* a *sys*, která jsou využita v průběhu ověřování existence a tvorbu souborů nebo jen získání k tvorbě cest pro uložení souborů.

Obsahuje tři funkce: *pkgs(), createManagerFile()* a *appendData(l, path)*. První a druhá jsou z příslušných modulů volány vždy odděleně. Jako první je vždy volána funkce *createManagerFile()* pro vytvoření souboru *managerFile.txt*, dále jen soubor pro řízení.

V první řadě je vždy provedena kontrola existence souboru. Pokud soubor zatím vytvořen nebyl nebo je prázdný, tak je zavolána funkce *appendData(l, path)*, která obsahuje předem nadefinované stavy uvedené v tabulce 4.1, které jsou zapsány do souboru.

Funkce *pkgs()* má na starost spuštění instalace potřebných balíčků v modulu *checkPackages.py*. Funkce se spouští dle hodnoty stavu v již vytvořeném souboru *managerFile.txt*. Pokud je hodnota na indexu nula nulová, tak je spuštěna instalace. Hodnota jedna určuje opak.

Při hodnotě nula je spuštěna funkce *check()* v modulu *checkPackages*. Po instalaci je hodnota v souboru pro řízení na stejném indexu nastavena na 1.

4.3.3 CheckConnection.py

Obsahuje pouze jednu funkci, *checkConnection(gui)*, kde je vstupním parametrem hodnota *False* (pro terminál) nebo *True* (pro GUI). Následně dle této hodnoty je pak interpretován i výstup z testu síťového připojení.

Tento modul kontroluje síťové připojení pomocí terminálového nástroje *ping*, který je spuštěn jako podproces za využití balíčku *subprocess*.

Ping je spuštěn pro oba operační systémy jiným způsobem, proto je také využita knihovna *platform*, pomocí které je získáno jméno operačního systému.

Pro operační systém Windows je použit klasický ping z důvodu možnosti běhu aplikace i mimo účet administrátora. Zatímco u Ubuntu je využit rozšířený ping, který posílá pouze dva dotazy, tj. *ping -c 2 -n 8.8.8.8*. V obou případech je výstup z pingu zachycen. Pak probíhá porovnání obdrženého výstupu s vybranými klíčovými slovy.

V prvním případě se testuje výstup na řetězec *TTL*. Pokud je v něm obsažen, tak je v případě terminálu poslán výpis *print('—— the internet connection is OK —– data will be downloaded—–')* a příslušná část programu pokračuje ve své činnosti dál. Pro GUI je vrácena hodnota *ok*, která je pak dále zpracována nebo je použita ve větvi pro rozhodování.

V druhém případě je výstup testován na existenci slova *unreachable*. Při kladném výsledku je činnost příslušného modulu zastavena, protože síťové připojení není dostupné. U GUI je v takovém případě zakázáno dotazování se na externí databáze.

4.3.4 CheckPackages.py

Modul obsahuje čtyři funkce: hlavní funkci *check()*, separátní funkce pro oba operační systémy pro zjištění nainstalovaných balíčků - *checkPkgsWin* a *checkPkgsLinux* a poslední funkci *listPkgs*.

Ve funkci *check()* probíhá zjištění operačního systému pomocí balíčku *platform*.

Pro operační systém Windows je pak spuštěna funkce *checkPkgsWin(operSyst)* a pro Ubuntu *checkPkgsLinux(operSyst)*.

V obou případech je pomocí *pip list* nebo *pip3 list* získán seznam již nainstalovaných balíčků a uložen do proměnné, která v obou případech putuje do funkce *listPkgs(operSyst, list)*, která již realizuje instalaci balíčků na základě porovnání získaného výstupu s fixně zadaným seznamem potřebných balíčků. Instalace pro oba systémy probíhá odděleně. Balíčky se v obou případech instalují pomocí nástroje *pip* nebo *pip3*. Výjimkou je balíček *tkinter* u systému Ubuntu, který se instaluje pomocí příkazu: *sudo apt install python3-tk*.

V případě Ubuntu je po instalaci balíčků jazyka Python ještě instalován nástroj *whois*.

4.3.5 SetTimersLinux.py

Stejně jako předešlý modul i *setTimersLinux.py* obsahuje jednu hlavní funkci, *createTemplate()*, na kterou pak navazují ostatní funkce: *createService(descr, serv, toRepeat, path, repeatTime, log), systemdScript(path, service)* a *runSh()*.

Modul nastavuje časovače, které se nastavují hned při prvním spuštění programu. Z toho důvodu hlavní funkce v první řadě kontroluje příslušnou hodnotu v souboru *managerFile.txt*. Přesněji hodnotu na indexu dva. Modul je spuštěn pouze v případě, že ještě časovače nastaveny nebyly, tj. hodnota na indexu dva se rovná nule.

Tehdy jsou v první řadě načteny seznamy obsahující:

- popis činnosti časovače (*descriptions*),
- název služby (*services*),
- jména modulů, které mají být spuštěny (*toRepeat*),
- a čas opakování služby (*repeatTimes*).

Postupným procházením zmíněných seznamů jsou vytvořeny soubory pro časovače za použití již vytvořené šablony a balíčku *jinja2*. Jsou vytvořeny časovače pro stažení dat z honeypotů, pro stažení a zpracování dat z externích zdrojů a pro zpracování dat z honeypotů.

Současně je vytvořen skript *(.sh)*, do kterého jsou vloženy příkazy pro jednotlivé služby, které budou nastaveny jako součást služby *systemd*. Na závěr je do skriptu přidán příkaz pro restartování služby *systemd*.

Současně je do souboru pro řízení místo nuly zapsána hodnota jedna.

Například vytvoření časovače pro stažení dat z honeypotů probíhá následujícím způsobem: do souboru časovače bude zapsán popis *download - honeypot data*, služba se

bude jmenovat *honeypotData*, bude spouštěn soubor *getHoneypotDataLin.py* a interval opakování bude *86400s* (24 hodin).

Po vytvoření příslušných souborů pro časovače je přes funkci *runSh()* spuštěn vytvořený skript.

4.3.6 SetTimersWin.py

Vstupním bodem modulu je funkce *createTemplate()*, která dle hodnoty na indexu 2 v souboru *managerFile.txt* spouští funkci *createTasks(path)*.

Při hodnotě nula jsou pomocí funkce *createTasks(path)* vytvořeny soubory pro automatické spuštění příslušných modulů a úkoly nastavené v *Plánovači úloh*. Jinak je běh modulu ukončen.

Ve funkci *createTasks(path)* jsou postupně pomocí *while* cyklu jsou nastaveny úkoly pro plánovač a také jsou pomocí funkce *taskFiles(task, fileName)* vytvořeny soubory pro spuštění. Nastavení jmen úkolů nebo jmen souborů probíhá načítáním hodnot z předem nastaveného seznamu.

Posledně zmíněná funkce, v první řadě vytvoří BATCH soubor, který je umístěn ve složce s ostatními moduly a který obsahuje jméno spouštěného modulu s koncovkou *.pyw*. Tento modul je spuštěn na pozadí, pomocí příkazu *"START /B /MIN "*.

Následující výpis 2 obsahuje již zmíněnou funkci *createTasks(path)*. Po nastavení úkolů je nastavena hodnota 1 na indexu pro nastavení časovačů a uložena do *managerFile.txt*.

```
def createTasks(path):
    services = ['honeypotData', 'externalData', 'processHoneypotData']
    toRepeat = ['getHoneypotDataWin.pyw', 'getExternalData.pyw',
                                    'processHoneypotData.pyw']
    i = 0
    while i != len(toRepeat):
        pathFile = taskFiles(services[i], toRepeat[i])
        i += 1
    file = open(os.path.join(path, str('managerFile.txt')), 'r');
    r = file.readlines();
    file.close()
    r[2] = '1\n'
    file = open(os.path.join(path, str('managerFile.txt')), 'w');
    file.writelines(r);
    file.close()
    del services, toRepeat, i, path, file
```

Listing 2 Funkce *taskFiles(task, fileName)*

4.3.7 SharedFunctions.py

Modul obsahuje 11 funkcí. Některé jsou využívány oběma operačními systémy, jiné slouží pro několik modulů najednou. Funkce budou popsány dle pořadí v souboru.

První funkcí je *getxmlAPI(xmlapi)*, která extrahuje API klíč z poskytnutého http odkazu od *whoisXMLAPI*. Za využití balíčku *re* je z odkazu smazána IP adresa. Dále pak jsou smazány části se samotným odkazem, a tak je získán samotný API klíč, který je pak přes funkce modulu *searchInDB.py* nebo *app.py* uložen do souboru *managerFile.txt*.

Druhou často používanou funkcí je *updateManagerFileTime(y, idx)*, která se stará o aktualizaci časových položek v souboru *managerFile.txt*. Vstupními parametry jsou samotný obsah souboru pro řízení a číslo indexu položky, která má být aktualizována.

Pomocné funkce *reverseMonthString(month)* a *createMonthString(month)* jsou si navzájem opačné a zodpovídají za převod názvu složky pro vybraný měsíc z typu *str* na typ *int* a naopak. U převodu na číslo je v případě názvu složky *03* odstraněno číslo nula a číslo tři je pak převedeno na datový typ *int*. Tyto funkce slouží pro práci s daty z honeypotů. Například pomáhají při převodu názvu k určování, který měsíc má být stažen jako další, nebo při určování aktuálně zpracovávaných dat.

Funkce *setPath()* je vyžíváno pro nastavení cesty. Například může být využito při nastavení cesty pro stažení nových dat nebo v případě systému Ubuntu pro získání posledně staženého roku z názvů souborů.

ConnectionPrint(year) realizuje výpis dodatečně staženého roku a je využívána funkcí *checkDownloadedData(year)*.

Funkce *downloadActualYear(y)* je již využívána moduly pro stažení dat z honeypotů. Testuje, jestli soubory pro aktuální rok již byly zpracovány. Pokud je aktuální rok obsažen mezi smazanými lety v souboru *managerFile.txt*, tak se spustí funkce *downloadingCondition(year, path)* pro opětovné stažení dat pro aktuální rok. Následně je spuštěno smazání již zpracovaných souborů pomocí funkce *prepareActualData(year, y, path)*.

V případě, že data pro aktuální rok zpracována nebyla, tak je uskutečněno pouze opětovné stažení dat pomocí funkce *downloadingCondition(year, path)*.

Další funkcí je zmíněná funkce *downloadingCondition(year, path)*, která v první řadě testuje existenci složky pro aktuální rok: *os.path.exists(os.path.join(path, str(year)))*. Pokud je výsledek podmínky pravdivý, tak je složka i se soubory smazána. Poté je zjištěn operační systém a na základě toho probíhá stažení dat, viz 3. Při nepravdivosti podmínky je postup stejný, vyjma smazání složky pro současný rok.

```
if 'Windows' in platform.uname().system:
        getHoneypotDataWin.updateActualYear(year, path)
if 'Linux' in platform.uname().system:
        getHoneypotDataLin.updateActualYear(year)
```

Listing 3 Stažení dat pro aktuální rok dle operačního systému

Smazání již zpracovaných souborů nebo případně celých složek, realizuje funkce *pre-*

pareActualData(year, y, path). V první řadě jsou smazány složky předchozích měsíců, tj. je ponechána pouze složka aktuálního měsíce. V dalším kroku je načten posledně zpracovaný soubor z *logs.txt*. Dle něj jsou smazány všechny předchozí, již zpracované, soubory.

Mazání souborů pro oba operační systémy probíhá odděleně, a to z důvodu odlišného ukládání názvů zpracovaných souborů v *logs.txt*. V případě systému Windows je do souboru vložen vždy název posledně zpracovaného souboru, kdežto u Ubuntu jsou do souboru ukládány všechny názvy pro daný měsíc. Výjimka platí pro posledně zpracovaný soubor daného měsíce, která je do souboru vložena samostatně.

Následující výpis 4 zobrazuje smazání souborů pro systém Ubuntu. Po výše uvedených krocích je aktualizován časový záznam na příslušném indexu v souboru pro řízení.

```
if 'Linux' in platform.uname().system:
        file = open(os.path.join(path, 'logs.txt'), 'r')
    p = file.readlines(); file.close()
    if len(p) == 1:
                if p[0:4] == str(year):
                        for rm in range(0,files.index(p)):
                os.remove(files[rm])
        else:
                        if p[-1][0:4] == str(year):
                        for rm in range(0,files.index(p)):
                                os.remove(files[rm])
```

Listing 4 Smazání souborů pro daný měsíc (pro Ubuntu)

Funkce *checkDownloadedData(year)* slouží pro ověření správně stažených dat v případě pádu aplikace nebo přerušení síťového připojení během prvotního stahování veškerých dat. Kontrola probíhá přes všechny složky od roku 2017 až po aktuální rok.

Pro možnost opravy byly vytvořeny seznamy *Repair*, pro chybně stažená data, a *Ok*, pro správně stažená data.

Pro oba operační systémy platí, že pokud při prvotním spuštění neexistuje složka pro rok 2017, tak je do modulu *getHoneypotDataLin.py* nebo *getHoneypotDataWin.py* vrácen seznam *Repair* se stejnou délkou jako rozdíl mezi rokem 2017 a aktuálním rokem. Seznam *Ok* je vrácen jako prázdný, co značí, že data ještě stažena nebyla.

Naopak pokud délka seznamu *Ok* je rovna zmíněnému rozdílu, tak data byla stažena v pořádku.

V ostatních případech, tj. seznam *Repair* je jiný od rozdílu, tak jsou data stažena dle hodnot v seznamu *Repair*. Například pokud seznam obsahuje *[2018,2020]*, tak budou stažena data pro roky 2018 a 2020.

Předešle zmíněné popisuje pouze výsledek kontroly. Detailněji. Pro každou existující složku roku je procházena také složka pro daný měsíc. Pro obsah složky měsíce je získán počet souborů, který je srovnán s počtem dní pro aktuální měsíc v ověřovaném roce.

Pokud je nalezena nesrovnalost, tak je číslo roku vloženo do seznamu *Repair*.

V tomto případě tady platí také výjimka, která platí pro první měsíc roku 2017. Obsahuje pouze pět souborů.

Jestliže hledaný rok není mezi soubory, tak je také přidán do seznamu *Repair*. V opačném případě je přidán do seznamu *Ok*. U operačního systému Windows jsou chybějící data ihned stažena. Na systému Ubuntu je stažení realizováno až v modulu *getHoneypotDataLin.py*.

Poslední funkcí modulu je *waitForChildren(r)*, která pouze realizuje zastavení procesů při stahování dat a také aktualizuje datum a čas posledního stažení v *managerFile.txt* přes funkci *updateManagerFileTime(r, 4)*.

4.3.8 GetHoneypotDataLin.py

Vstupním bodem pro stažení dat je funkce *getDataForLinux()*, která na základě hodnot v *managerFile.txt* testuje, zda již byla stažena data z předešlých let a také jestli je nutné aktualizovat data pro aktuální rok. Výpis 5 zobrazuje část, která je zodpovědná pro získání časového údaje o stažených datech pro předcházející roky (index 4 a index 5) a pro aktualizaci dat pro aktuální rok (index 4).

Pokud je hodnota na indexu 4 rovna řetězci *lastUpdate* nebo ještě veškerá data nebyla stažena, tj. hodnota na pozici 5 je nulová, tak je spuštěna funkce *getArchiveData(mF, path)*. Pokud na pozici 4 je již časový údaj a hodnota na pozici 5 je jedna, tak je časový údaj z pozice 4 převeden na typ *datetime*, pomocí kterého je možné porovnat dvě časové hodnoty. Je spočítán rozdíl hodnot aktuální a získané časové hodnoty. Pokud je rozdíl větší než 1 den, 24 hodin, 86 400 sekund, tak je spuštěna funkce *downloadActualYear(y, path)* z modulu *sharedFunctions*.

```
path = os.path.join(sys.argv[0].replace(os.path.basename
        (sys.argv[0]), ''), str('managerFile.txt'))
if os.path.exists(path) == True:
        file = open(path, 'r'); y = file.readlines(); file.close()
        if y[4] == 'lastUpdate\n' or y[5] == '0\n':
                getArchiveData(y, path)
        else:
                lastUpdate = datetime.datetime.strptime
                        (y[4].replace('\n', ''), '%Y-%m-%d %H:%M:%S')
                d = datetime.datetime.strptime(
                datetime.datetime.strftime(datetime.datetime.now(),
                        '%Y-%m-%d %H:%M:%S'), '%Y-%m-%d %H:%M:%S')
                        - lastUpdate
                if d.total_seconds()/86400 > 1:
                        sharedFunctions.downloadActualYear(y, path)
```

Listing 5 Rozhodnutí pro stažení celkových dat nebo jen jejich aktualizace

Funkce *getArchiveData(mF, path)* slouží pro stažení dat z předchozích let až po aktuální datum, kdy je modul spuštěn. Na začátku je nastaven počáteční rok pro sta-

žení, 2017. V dalším kroku je použita již popsaná funkce *checkDownloadedData(year)* z modulu *sharedFunctions*. Jeho výsledkem jsou dva seznamy *Repair* a *Ok*.

Pokud délka seznamu *Ok* je stejná jako počet let od roku 2017, tak je nastavena hodnota 1 na pozici 5 v *managerFile.txt*, která značí, že všechna data byla stažena. Pokud, ale seznam *Repair* bude mít zmíněnou délku, tak budou staženy všechny všechny soubory od roku 2017 až po aktuální rok.

V případě, pokud délka seznamu *Repair* bude jiná, menší, tak data budou stažena pouze pro roky obsažené v seznamu. Pro poslední dvě zmíněné možnosti je použita funkce *getDataForYear(year, FileNames)*. Současně u obou případů je po stažení nastavena hodnota 1 v souboru *managerFile.txt*, která indikuje, že vše bylo staženo.

Funkce *getDataForYear(year, FileNames)* realizuje stažení dat pro jednotlivé roky pomocí nástroje *wget*, viz výpis 6.

```
command = 'https://haas.nic.cz/stats/export' + '/' + year +'/'
getDataCommand = subprocess.run(['wget', '-r', '-np',
        '--no-verbose', '-nH', '--cut-dirs=3', '-R',
         'index.html',command],
                stdout=subprocess.PIPE, stderr=subprocess.PIPE)
```

Listing 6 Stažení dat pro operační systém Ubuntu

Vstupním bodem pro stažení dat pro aktuální rok je přes zmíněnou funkci *downloadActualYear(y, path)* v modulu *sharedFunctions*. Po ověření a případném promazání dat pomocí funkce *prepareActualData(year, y, path)*, je spuštěna funkce *updateActualYear(year, y)*, která je oddělena pro každý operační systém. Po nastavení cesty k souborům je opět spuštěna funkce *getDataForYear(year, FileNames)*, která využívá dalších funkcí pro stažení dat.

Poslední funkce modulu, *unzipFile(fileName)*, je používána pro extrakci souborů typu *.gz* na soubory typu *.json*, které jsou pak zpracovávány pomocí modulu *processHoneypotData.py*.

4.3.9 GetHoneypotDataWin.py

Modul je v podstatě ekvivalentní k předešle popsanému, pouze s tím, že je určena pro operační systém Windows. První funkce *getDataForWindows(gui)* modulu je založena na stejném principu jako první funkce předešlého modulu. Rozhoduje, jestli má být realizováno stažení veškerých dat nebo jen dat pro aktuální rok.

Kontrola stažených dat probíhá obdobným způsobem jak u funkce *getArchiveData(mF, path)* u předešlého modulu. Je realizováno přes funkci *getArchiveDataWin(r, path)*. Výjimkou je stažení špatně stažených dat, která jsou stažena přes modul *sharedFunctions*, který byl popsán výše. Pokud, ale data zatím stažena nebyla, tak je přímo použita funkce *runProcess(r, year, path)*, která spouští ekvivalentní počet procesů jaký je počet

let pro stažení.

Procesy spouští funkci *getDataForYear(year, path)*, která dále spouští dvě funkce: *createDirStructure(year, month, day, path)* a *downloadJons(year, month, monthDir, day)*.

Z důvodu, že nástroj *curl* nezajišťuje stažení adresářové struktury, tak musela být vytvořena funkce *createDirStructure(year, month, day, path)* která vytvoří požadovanou strukturu složek pro roky a měsíce před stažením dat. Tato struktura byla zachována také z důvodu, že nástroj *wget* soubory stahuje v adresářích a funkce pro zpracování dat byla uzpůsobena pro zpracování dat rozdělených ve složkách.

Přesná nastavení pro stažení souborů probíhá ve funkci *downloadJons(year, month, monthDir, day)*. Před stažením je nastaven přesný název souborů pro stažení pomocí nástroje *curl*. Výpis 7 zobrazuje nastavení jména souboru pro stažení dat prvních devíti dní v prvních devíti měsících.

```
if day < 10 and month < 10:
        fileToDownload = str(year) + '-0' + str(month) + '-0'
        + str(day) + '.json'
    downloadData(year, monthDir, fileToDownload)
```

Listing 7 Nastavení návzu souborů pro stažení (Windows)

Stažení dat již probíhá ve funkci *downloadData(year, month, fileToDownload)*, která používá obdobný příkaz pro stažení jak pro operační systém Ubuntu.

4.3.10 ProcessHoneypotData.py

Modul obsahuje třináct vzájemně propojených funkcí.

Vstupním bodem modulu je funkce *processData()*, která po zjištění operačního systému nastaví cestu, kde se nachází stažená data, dále pak získá cesty:

- pro *managerFile.txt*, kterou uloží do proměnné *path*,
- pro soubor *logs.txt*, který obsahuje posledně zpracovaný soubor. Cesta je uložena do proměnné *logsPath*.

Dále pak je získán obsah složky, kde jsou umístěna data z honeypotů. Obsah je uložen do seznamu *Dirs*, který je postupně procházen.

Po nastavení cest, jsou na základně jména posledně zpracovaného souboru z logovacího souboru *logs.txt* získány rok, měsíc a den zpracován pomocí funkce *setYearAndMonths(r, path)*. Pokud je měsíc v záznamech hodnota *12* a den je *31*, tak je automaticky navýšena hodnota roku a nastaveny všechny měsíce pro zpracování. V ostatních případech jsou dle hodnoty měsíce nastaveny názvy měsíců pro další zpracování.

Pokud, ale logovací soubor neexistuje nebo je prázdný, tak je použita funkce *setFullMonths()*, která nastaví každý měsíc v roce a také zpracování let od roku 2017.

Výpis 4.3.10 zobrazuje nastavení všech měsíců pro zpracování a také všechny roky od 2017 až aktuálnímu datu.

```
FoldersMonths = ['01', '02', '03', '04', '05', '06', '07', '08',
        '09', '10', '11', '12']
Years = []
for i in range(2017, date.today().year+1):
        Years.append(str(i))
return FoldersMonths, Years
```

Obě funkce vrací dva seznamy. Prvním z nich je *FoldersMonths*, která obsahuje složky měsíců, které mají být zpracovány. Pokud ještě daný rok zpracován nebyl, tak seznam obsahuje názvy od *01* až po *12*. V opačném případě seznam neobsahuje úplný výčet těchto hodnot. Druhým seznamem je seznam *Years*, ve kterém jsou uloženy roky, které ještě zpracovány nebyly.

Také je nastavena pomocná proměnná *p* na nulu, která je určena pro výpis hlášení o nestažených datech v i po procházení seznamem *Dirs*.

Dalším krokem je procházení seznamu *Dirs*. Pokud při procházení seznamu není nalezena žádná shoda s hodnotou v seznamu *Years*, tak hodnota pomocné proměnné *p* zůstává stejná (nulová).

Pokud naopak je při procházení seznamu *Dirs* nalezena shoda s hodnotou v seznamu *Years*, tak je spuštěna funkce *processEachMonth(str(name), FoldersMonths, operatingSystem, path, logsPath)* se vstupními hodnotami v pořadí: rok (*name*), seznam se jmény podsložek pro měsíce (*FoldersMonths*), jméno operačního systému (*operatingSystem*), cesta k *managerFile.txt* (*path*) a cesta k logovacímu souboru (*logsPath*).

Funkce má návratovou hodnotu *close*, na základě které je běh celého modulu ukončen. Před samotným ukončením je nastavena hodnota pro další zpracování souboru pomocí funkce *whatProcessNext(operatingSystem, logsPath)* a je aktualizován datum posledního zpracování. Pokud byly zpracovány soubory z předešlých let než aktuální rok, tak je hodnota uložena na pozici 6, jinak je aktualizována hodnota na pozici 12. Dále pak je vypsán výpis celkového času zpracování a běh je ukončen.

Navazující funkcí je *processEachMonth(name, FoldersMonths, operatingSystem, path, logsPath)*, které jsou předány hodnoty z předešlé funkce.

Pomocí příkazu *os.listdir()* pro každý rok jsou získány jména složek pro jednotlivé měsíce, které jsou postupně procházeny. Jejich obsah bude postupně zpracován pomocí funkce *runThreads(FileNames, operatingSystem, logsPath)*. Například ze složky *01* jsou získány příslušné soubory, které jsou pak zpracovány.

Data jsou zpracovávána po jednotlivých měsících. Výjimka platí pro rok 2017, který

je zpracován jako celek, protože jeho obsah je možné zpracovat najednou.

Funkce *runThreads(FileNames, operatingSystem, logsPath)* vrací proměnnou *d*, *year* a *r*, která určuje posledně zpracovaný soubor. Hodnota proměnné *d* může nabývat hodnot *True* nebo *False*.

V případě, že hodnota je *False* a hodnota proměnné *year* se rovná nule, tak je činnost modulu ukončena, viz 4.3.10.

```
if d == False and year == 0:
        return 'close'
```

Složka s daty pro daný rok je ponechána. Pokud má pro proměnná *d* hodnotu *True* a proměnná *r* hodnotu *12*, tak je složka pro celý rok smazána. Níže uvedený výpis 8 uvádí smazání již zpracovaných složek.

```
if 'Linux' in operatingSystem:
        os.chdir('..')
    shutil.rmtree(Folders[j])
    os.chdir('..')
    if Folders[j] == '12' and d == True: rmYear(year, path)
```

Listing 8 Průběžné smazání souborů pro daný měsíc a pro daný rok (pro Ubuntu)

Pro smazání celé složky pro daný rok je použita funkce *rmYear(year, path)*.

Funkce *runThreads(FileNames, operatingSystem, logsPath)* zodpovídá za řízení zpracování jednotlivých souborů ve složce. V níže uvedeném výpisu 9 je uvedená hlavní část funkce, ve které jsou postupně spuštěny vlákna pro zpracování JSON souborů na základě počtu záznamů. Ve všech případech je zpracování realizováno pomocí vláken. Jedno vlákno pro jeden soubor.

Rozdíl je ve spouštění jednotlivých vláken, které je časováno dle počtu záznamů v souboru. Pokud soubor obsahuje více než tři sta tisíc záznamů, tak je spuštění dalšího vlákna zastaveno na 3 minuty. Pro ostatní velikosti to platí obdobným způsobem. Vlákna spouští funkci *process(file, operatingSystem, logsPath)*.

```
d = False; year = 0
for file in FileNames:
        if ".json" not in file:
                continue
    if '.gz' in file:
                file = unzipFiles(operatingSystem, file)
    year = file[0:4]
    if len(pd.read\_json(file, orient='records')) > 300000:
                process(file, operatingSystem, logsPath)
    if len(pd.read\_json(file, orient='records')) > 300000:
                time.sleep(180)
    if len(pd.read\_json(file, orient='records')) <= 300000:
                t = threading.Thread(target=process,
                        args=(file,operatingSystem,logsPath))
        t.start()
    if 100000 < len(pd.read\_json(file, orient='records')) < 300000:
                time.sleep(80)
    if 50000 < len(pd.read\_json(file, orient='records')) < 100000:
                time.sleep(50)
```

```
    if 10000 < len(pd.read\_json(file, orient='records')) < 50000:
                time.sleep(30)
    if 100 < len(pd.read\_json(file, orient='records')) < 10000:
                time.sleep(2)
    if len(pd.read\_json(file, orient='records')) < 100:
                time.sleep(0)
    if file[5:7] == '12' and file[8:10] == '31':
                d = True; year = file[0:4]
    t.join(60)
    t.join()
    if ".json" in file: r = file
return d, year, r
```

Listing 9 Řízení zpracování jednotlivých souborů

Ve funkci *process(file, operatingSystem, logsPath)* je za využití modulu *saveToDB.py* a funkce *checkPathAndCreateTable(pointer, databaseFileName, tableName)* vytvořena databáze a tabulka *honeypotData*, za podmínky, že ještě vytvořena nebyla.

V dalším kroku je zapsán název souboru do logovacího souboru pomocí funkce *updateLog(file, operatingSystem, logsPath)* a také je realizováno připojení k databázi. Dále je přistoupeno k extrakci dat ze souboru - funkce *pandasListToString(process, conn)*. Pokud je počet záznamů v souboru větší než tři sta tisíc, tak je soubor rozdělen na části o velikosti 250 000. Tyto části jsou pak zpracovávány postupně. Menší počet záznamů je zpracováván najednou. Po zpracování je ukončeno spojení s databází.

Výběr podstatných dat ze souborů je extrahován ve funkci *pandasListToString(Readed, conn)*. Každý načtený soubor je procházen řádek po řádku a jsou z něho vybírány informace o: ip adrese, času, zadaných příkazech a krajině původu.

V případě času je provedena kontrola vůči uložené hodnotě v tabulce. Pokud existuje, tak je z databáze získán posledně uložený časový údaj pro hledanou IP adresu. Hodnota je porovnána s hodnotou, která má být vložena do databáze, přes funkci *selectTime(actualTime, lastReleasedTime)*. Funkce porovnává tyto dvě hodnoty. Pokud je nová hodnota novější, tak je hodnota v databázi přepsána. Jinak je ponechána předešlá hodnota.

Po těchto úpravách jsou data vložena do datového typu *tuple*, která je poslána do funkce
setToDatabase(dataToInsert, length, actualRow, conn, tableName) pro modul *saveToDB* pro uložení do databáze.

4.3.11 GetExternalData.py

Modul obsahuje čtyři funkce, kde hlavním z nich *checkExternalData()*, která obsahuje předem nastavené zdroje dat pro stažení blacklistů a blocklistů. Data jsou stažena pomocí časovače každý den. Po jejich stažení a zpracování je nastavena poslední aktualizace.

Funkce *downloadExternalData(urls, filesToDownload, path, origins)* již pomocí vestavěných knihoven jazyka Python stáhne data z příslušných odkazů. Například soubor *reputation.data* ze seznamu *filesToDownload* je stažen z odkazu *https://reputation.alienvault.com/* ze seznamu *urls*.

Data jsou stažena postupně pro každý uvedený zdroj, po kterém jsou okamžitě zpracována a také uložena do databáze přes funkci *setExternalData(filesToDownload, path, origin)*.

Samotná funkce pro získání relevantních dat využívá parsovací funkce z modulu *processExternalData.py*, který bude rozebrán později. Parsovací funkce ve většině případů vrací extrahované informace ve formě jednoho nebo dvou seznamů. Jejich obsah je pak postupně vkládán do tabulek.

Pokud blacklist nebo blocklist obsahoval pouze IP adresy, tak jsou data vložena do tabulky *externalData* pomocí funkce *setToDatabase(dataToInsert, length, actualRow, conn, tableName)* z modulu *saveToDB.py*.

Jestli extrahovaná data obsahovala i jiné údaje, tak jsou vloženy do tabulky *additionalData* přes funkci *addNewData(ip, dataToInsert, columns, conn)* také z modulu pro práci s databází. Pro každý soubor je také měřeno celková doba zpracování.

Následující výpis 10 zobrazuje postoupení již extrahovaných dat ze seznamu *ips* do tabulky *externalData* spolu s připojením a odpojením od databáze.

```
conn = saveToDB.connection()
conn.getConn()
conn = saveToDB.setToDatabase((ips[j]), len(ips), j, conn,
            'externalData')
conn.closeConn()
```

Listing 10 Vkládání dat do tabulky externalData

4.3.12 ProcessExternalData.py

Na základě formátů blacklistů a blocklistů bylo potřebné vytvořit funkce pro extrakci dat. Některé soubory obsahují pouze škodlivé IP adresy, jiné obsahují i informace jako poslední nahlášení, krajinu nebo jméno hostující webové stránky.

Stejně tak, seznamy jsou staženy ve více formátech. Nejčastějším je formát *.txt*. Nicméně například soubor ze stránek *AlienVaultu* je typu *.data*. V jiném případě je seznam uložen v komprimovaném souboru *.zip*.

Rozdílnost ve formátech i tvaru uložení dat vedlo k vytvoření pěti funkcí pro jednotlivé formáty. Společnou vlastností funkcí jsou stejné vstupní parametry: název souboru a cesta pro načtení souboru.

Prvním z nich je formát *.data* (zdroj AlienVault), který obsahuje IP adresy a k nim přiřazené město a krajinu původu. Postupnou iterací jsou z řádků získány tyto infor-

mace a vráceny ve tvaru seznamu do modulu *getExternalData.py*.

Pro typ souboru *.php* byla vytvořena funkce *processPHPFile(fileName, path)*, která ze souboru extrahuje IP adresu a datum posledního hlášení.

V případě blacklistu zabaleného jako ZIP soubor, byla vytvořena funkce *processZip(fileName, path)*, která extrahuje informace pomocí vytvořených regulárních výrazů. Přesněji jsou získány informace: IP adresa, původ, host, datum a čas.

Následující výpis 11 obsahuje otestování podmínky pro existenci regulárního výrazu pro IP adresu. Pokud byl řetězec nalezen, tak je vložen do seznamu *tmp*. Obsah seznamu je pak po nalezení všech řetězců zkopírován do konečného seznamu, který je vrácen do modulu *getExternalData.py* pro uložení do databáze.

```
ipPattern = re.compile(r'(\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})')
if bool(re.search(ipPattern, line)) == True:
        tmp.append(re.search(ipPattern, line)[0])
```

Listing 11 Nalezení IP adresy dle regulárního výrazu

Pro blacklist z abuseIPDB byla vytvořena funkce *processBlklistAbuse(path)*, která také používá regulární výrazy pro výběr důležitých atributů. Přesněji jsou získávány informace: IP adresa, země původu, datum a čas a tzv. *abuseConfidenceScore*, která určuje míru důvěrnosti IP adresy.

Poslední funkcí je funkce *processStandardTxt(fileName, path)*. Pro formát TXT byla vytvořena funkce *processStandardTxt(fileName, path)*, která postupně získává příslušné IP adresy ze souboru.

4.3.13 SaveToDB.py

Posledním modulem druhé části je modul *saveToDB.py*. Obsahuje jedenáct funkcí a jednu třídu pro uskutečnění připojení s databází. Některé funkce v modulu jsou vzájemně propojené. Jiné jsou přímo využívány jinými moduly, například pro získání hodnot. Stejně tak pro vkládání dat a získání dat s databáze existují různé vstupní body.

Primárními vstupními body jsou:

- funkce *checkPathAndCreateTable(pointer, databaseFileName, tableName)*, která má na starost tvorbu databáze a příslušných tabulek,
- a třída *connection()*, pomocí které je vytvořeno spojení s databází.

Funkce *checkPathAndCreateTable(pointer, databaseFileName, tableName)* po vytvoření tabulek využívá funkci *setTableIndicator(idx, path)* pro nastavení hodnot pro nastavení tabulek na indexech 7 až 9. Je nutné zmínit, že tabulky jsou vytvářeny postupně na základě vkládaných hodnot.

Například pokud jsou vkládány data do tabulky *honeypotData*, tak je vytvořena tabulka s tímto jménem. Ostatní tabulky pokud ještě vytvořeny nebyly, tak je v databázi nebude možné nalézt.

Třída *connection()* obsahuje funkci *__init__(self)*, která vytvoří soubor *data.db* přes funkci *detectSystem(workingFolder)*. Dále pak obsahuje funkce: *dbName(self)* - pro získání jména databáze, *getConn(self)* - pro vrácení připojení, *setCursor(self)* - pro získání spojení, *commitConn(self)* - pro potvrzení zadaného příkazu, *closeConn(self)* - pro ukončení spojení s databází.

Vstupním bodem pro vkládání dat z jiným modulů jsou funkce *setToDatabase(dataToInsert, length, actualRow, conn, tableName)* a *addNewData(ip, dataToInsert, columns, conn)*.

Vstupními parametry funkce *setToDatabase(dataToInsert, length, actualRow, conn, tableName)* jsou:

- řádek dat, *dataToInsert*, který má být vložen do tabulky,
- celková délka dat, využita pro uzavření ukazatele v databázi,
- současný index vkládaného řádku, *actualRow*, který slouží pro vytvoření databáze, pokud ještě vytvořena nebyla,
- již vytvořené spojení s databází, *conn*,
- jméno tabulky, *tableName*, do které mají být data vkládány,

Po nastavení ukazatele v databázi je v případě nulového indexu zkontrolována existence databáze a příslušné tabulky. Pokud je hodnota *actualRow* jiná od nuly tak je přímo spuštěna funkce *checkDuplicate(pointer, dataToInsert, connection, tableName)*, která kontroluje existenci duplikátních hodnot v příslušné tabulce.

Následující výpis 12 zobrazuje kontrolu duplikátních hodnot pro tabulku *honeypotData*. Ne-duplikátní data, příkazy, sloučeny s již existujícími a vloženy do tabulky. V opačném případě je vrácena hodnota 0 a použita funkce *insertDataToDB(pointer, dataToInsert, connection, tableName)*.

```
exist = 0
if tableName == 'honeypotData':
        for row in pointer.execute("SELECT TIME, COMMANDS FROM
honeypotData WHERE IP=?", (dataToInsert[0],)):
                if dataToInsert[2] in row[1]:
                        exist = 2
        else:
            dtI = list(dataToInsert)
            dtI[2] = row[1] + ';' + dtI[2]
            dataToInsert = tuple(dtI)
            exist = 1
```

Listing 12 Kontrola duplikátních hodnot pro tabulku *honeypotData*

Jestliže výslednou hodnotou funkce pro kontrolu duplikátních hodnot je hodnota 0, tak je spuštěna funkce *insertDataToDB(pointer, dataToInsert, connection, tableName)*, která se stará o přímé vkládání dat do jednotlivých tabulek v databázi. Samotná funkce *setToDatabase(dataToInsert, length, actualRow, conn, tableName)* pak vrací samotné připojení do jiného modulu.

Funkce *addNewData(ip, dataToInsert, columns, conn)* slouží pro vkládání dat do tabulky *additionalData*, například z modulů *saveToDB.py* nebo *getExternalData.py*. V tomto případě jsou vstupními hodnotami IP adresa, konkrétní údaje o IP adrese, parametry k dané IP adrese a připojení k databázi. Po získání ukazatele je spuštěna funkce *addAdditionalData(ip, columns, conn, pointer, dataToInsert)* pro vložení dat. Po vložení dat je spojení s databází ukončeno.

Funkce *addAdditionalData(ip, columns, conn, pointer, dataToInsert)* slouží k přidávání dat do tabulky *additionalData*. V první řadě jsou z tabulky získány dostupné informace na základě IP adresy. V dalším kroku probíhá srovnání dat z tabulky s daty, které mají být vloženy. Existující data s danými parametry jsou z potenciálně vkládaných dat smazána.

Pokud je potřebné existující rozšířit, tak je v první řadě smazán příslušný záznam. Následně jsou data rozšířena a vložena do tabulky znova. Data bez rozšíření se do tabulky vkládají standardním způsobem. Pomocí sql příkazu *INSERT*.

Poslední funkcí modulu je funkce *searchInDB(search, tableName)*, která slouží pro vyhledávání dat v databázi na základě jména tabulky. Po získání dat jsou informace vrácena do modulu *searchInDB.py* a *app.py*.

4.3.14 Index.py

Modul slouží jako vstupní bod pro spuštění aplikace v terminálu. Ještě před samotným spuštěním funkce *main()* je vytvořen soubor pro řízení, *managerFile.txt*, a doinstalovány potřebné balíčky na základě hodnoty v předešle zmíněném souboru. Pokud je hodnota v souboru pro řízení na indexu 0 nulová, tak je provedena instalace balíčků. Dále pak jsou na základě operačního systému nastaveny časovače. Rozhodnutí nastavení probíhá stejným způsobem jak u instalace balíčků s výjimkou, že se kontroluje hodnota na indexu dva.

Po spuštění hlavní funkce jsou v načteny API klíče ze souboru pro řízení. Pro načtení klíče pro databázi *abuseIPDB* je použita funkce *abuseAPI(path)* z modulu *searchInDB.py*. Načtení klíč pro databázi *whoisXMLAPI* probíhá pomocí funkce *whoisAPI(path)* ze stejného modulu. Pokud některý z klíčů není v souboru obsažen, tak je uživatel požádán o jeho zadání. V případě klíče pro databázi *whoisXMLAPI* stačí zkopírovat jeden z obdržených odkazů.

Po kontrole klíčů je zkontrolována existence lokální databáze. Při neexistenci je vypsána hláška: —-*local DB is not set - data haven't been downloaded or processed yet.*

Následně je spuštěn cyklus s možností výběru pro vyhledávání v databázi, která vede na modul *searchInDB.py*, nebo pro ukončení aplikace v terminálu. K těmto úkonům je využita návratová hodnota proměnné *choice* z funkce *question(s)*, která pouze načítá vstup z terminálu. Hodnota jedna indikuje spuštění modulu pro vyhledávání v databázi a hodnota nula ukončuje celou aplikaci.

4.3.15 SearchInDB.py

Pro přímé vyhledávání dat v databázi slouží modul *searchInDB.py*. Tyto funkce využívá aplikace s grafickým rozhraním i aplikace v terminálu. Pro vyhledávání v terminálu je použita funkce *searchInDB()*, kde jsou načteny příslušné API klíče pomocí funkcí zmíněných v popisu modulu *index.py*.

Načtení API klíče pro *abuseIPDB* probíhá načtením přes funkci *abuseAPI* získáním hodnoty z indexu 10 ze souboru *managerFile.txt*a vypsáním výpisu o tom, že klíče jsou načtené. Klíč pro databázi *whoisXMLAPI* je načten z indexu 13.

Pro oba případy platí: při chybějícím klíči nebo při nulové délce klíče je spuštěna funkce *enterAPIkey(path, version)*, pomocí které je načten klíč z terminálu nebo vstupního pole a pomocí funkce *setTableIndicator(idx, APIkey, path)* je zapsán do souboru pro řízení.

Po získání API klíčů následuje rozcestník pro vyhledávání nebo pro opuštění části pro vyhledávání. K tomuto účelu je použita již popsaná funkce *question(s)*.

Pokud je nastavena možnost pro vyhledávání, tak je spuštěna funkce *queryIP(APIkey, xmlAPIkey, gui, path)*, která již realizuje vyhledávání v databázi.

Prvně je pomocí funkce *getLastModification(path, extDB)* ze souboru pro řízení na základě jména databáze získán využitý počet dotazů. Následující výpis 13 zobrazuje načtení těchto hodnot z *managerFile.txt* pro databázi *abuseIPDB*. Pokud poslední aktualizace na indexu 12 není rovna řetězci *lastUpdate*, tak je tento časový údaj načten do proměnné *lastModification*, která je srovnána s aktuálním datem. Při rovnosti je načtena hodnota realizovaných dotazů a uložena do proměnných *numA* a *num*.

Nerovnost značí, že realizované dotazy jsou z jiného dne. Pro tuto databázi je limit denně aktualizován, tj. hodnoty proměnných *numA* a *num* jsou nastaveny na nulu a hodnota na indexu 11 v souboru je také nastavena na nulu a současně je také aktualizován datum poslední změny na indexu 12.

```
file = open(path, 'r'); y = file.readlines(); file.close()
if extDB == 'abuseIPDB':
        if y[12] != 'lastUpdate\n':
                modificationTime = datetime.datetime.
                strptime(y[12].replace('\n', ''), '%Y-%m-%d').date()
        if modificationTime == datetime.datetime.now().date():
                        numA = y[11].replace('\n', '');
                        num = int(y[11].replace('\n', ''))
        else:
            numA = str(0); num = 0
            y[11] = str(num)+'\n'
            y[12] = datetime.datetime.strftime
            (datetime.datetime.now().date(), '%Y-%m-%d')+'\n'
            file = open(path, 'w'); file.writelines(y);
            file.close()
    else:
        numA = str(0); num = 0
        y[11] = str(num)+'\n'
        y[12] = datetime.datetime.strftime(datetime.datetime.now()
        .date(), '%Y-%m-%d')+'\n'
        file = open(path, 'w'); file.writelines(y);
        file.close()
```

Listing 13 Načtení počtu provedených dotazů na databázi *AbuseIPdb*

Pro databázi *whoisXMLAPI* je použit obdobný postup, získané nebo aktualizované hodnoty jsou uloženy do proměnných:

- *num* - počet vyčerpaných pokusů pro databázi *abuseIPDB*, jako datový typ *int*,
- *numA* - stejná hodnota jak *num*, ale typu *str*,
- *enableXML* - hodnota typu *bool*, která určuje vyhledávání v databázi *whoisXMLAPI*, kde hodnota *True* umožňuje vyhledávání,
- *att* - počet vyčerpaných pokusů pro databázi *whoisXMLAPI*.

Také je nastavena hodnota pomocné proměnné *query* na True a je spuštěn *while* cyklus pro zadávání IP adres, který může být ukončen zadáním hodnoty nula.

Před samotným zadáním hledané IP adresy je přes funkci *externalDBSearch(num, numA, maxAbuseIPDB, gui)* proveden výpis zbývajícího počtu pokusů pro obě databáze.

Výpis 14 zobrazuje rozhodnutí pro vypsání hodnot pro databázi *abuseIPDB*. U nepřesažené hodnoty u proměnné *num* je realizován výpis zbývajících dotazů. Jinak je vypsáno hlášení, že je denní limit dosažen a na databázi se dotazovat nelze.

```
if isinstance(num, int) == True:
        if num < 1000:
                if gui == True:
                        return 0
        if gui == False:
                        print('\n------(abuseIPDB) you have reached:
 ' +numA+ '/1000 attempts per day------')
                        print('------left attempts for today: '
```

```
                                +str(maxAbuseIPDB-num)+' ----------\n')
    if num == 1000 or num > 1000:
                    if gui == True:
                                return 0
        if gui == False:
                                print('\n-------(abuseIPDB) can
                                not use external database--------')
            print('--------daily limit reached---------\n')
```

Listing 14 Vypsání počtu zbývajících dotazů pro databázi *AbuseIPDB*

Následuje cyklus pro zadaní IP adresy ze strany uživatele spolu s kontrolou formátu IP adresy verze 4 nebo verze 6. Při správnosti je tento cyklus ukončen a je spuštěno prohledávání všech tří tabulek databáze v následujícím pořadí: *honeypotData, externalData* a *additionalData*.

K prohledávání je použita funkce *checkLocalDB(ip, gui, tableName)*, která vrací jeden seznam v případě tabulek *honeypotData* a *externalData*. U tabulky *additionalData* je vrácen jeden seznam se dvěma vnořenými seznamy, kde první z nich obsahuje parametry ze sloupce *ATTRNAME* a druhý obsahuje detaily ze sloupce *DETAILS*.

Z těchto rozdělených dat je získán původ informace, *origin* nebo *source*. Pokud u původu není uveden řetězec *abuseIPDB* nebo *xml*, tj. ještě nebyl proveden dotaz na externí databáze, tak je do pomocné proměnné nastaven prázdný řetězec. V opačném případě je proměnná nastavena na hodnotu *abuseIPDB*, hodnotu *xml* nebo jsou vloženy oba řetězce. Pomocná proměnná je pak použita pro rozhodování stažení dat z příslušných databází.

Před realizací dotazů na externí databáze jsou vypsány informace z dodatečné tabulky pomocí funkce *printAdditionalData(ip, row)*.

Následuje získání dat pro hledanou IP adresu z databáze *abuseIPDB* přes funkci *checkAbuseDB(numAbuse, ip, APIkey)*, která realizuje dotaz pouze pokud není přesažen denní limit 1000 dotazů.

Dotaz je uskutečněn přes funkci *queryAbuseIPDB(ip, APIkey)* z modulu *queryExternalDB.py*, která vrací odpověď v datovém typu *dictionary*. Výsledek je uložen do proměnné *data*.

Po realizaci dotazu je aktualizován počet provedených dotazů na databázi, tj. v souboru pro řízení je navýšena hodnota pokusů o jedna. K tomu slouží funkce *updateAttempts(num, path, extDB)*, která funguje obdobným způsobem jak funkce *updateManagerFileTime(y, idx)* s tím rozdílem, že je najednou ukládána časová hodnota i navýšený počet provedených dotazů.

Dále je obsah proměnné *data* rozdělen na parametry a detaily za využití funkce *processData(data, origin, additional)* a *divideData(data)*. Data jsou pak také průběžně vypsána do terminálu přes funkci *printData(row, data, ip, gui, origin, additional)*.

Funkce *addNewInfoToDB(ip, items, columns, path)* pak na dalším řádku zajišťuje

uložení těchto dat do lokální databáze. Před vložením dat je provedena kontrola existence tabulky přes *table()* a následně pro vkládání je spuštěna funkce *insert(ip, items, columns)*.

Pro získání dat z *whoisXMLAPI* databáze je uskutečněn obdobný postup, s tou výjimkou, že data jsou do databáze uložena ihned po extrakci v modulu *queryExternalDB.py* k výpisu je použita funkce *extDBPrintData(result)*.

Jako poslední jsou vypsána data z honeypotů pomocí funkce *printHData(ip,row)*, ale pouze v případě pokud je tato IP adresa obsažena v tabulce *honeypotData*.

4.3.16 QueryExternalDB.py

Modul slouží pro realizaci dotazů na externí databáze a zpracování obdržených dat. Ve funkci *otherExternalDB(ip, APIkey)* jsou postupně procházeny zdroje dat z databází: *dan.me.uk, whois* a *whoisXMLAPI*, kde je vybrán úplně první záznam, který obsahuje nejvíce specifické hodnoty pro hledanou IP adresu. Ostatní záznamy již obsahují širší rozsahy IP adres, do kterých hledaná IP adresa patří a které jsou vedeny až k rozsahu *0.0.0.0/0*.

První dva zdroje jsou uplatněny pouze u operačního systému Ubuntu z důvodu nefunkčnosti nástroje *whois* na operačním systému Windows. Získané informace pro *dan.me.uk* extrahovány a vloženy do databáze pomocí funkce *extractDanMe(toExtract, ip)*. Navrácená hodnota putuje do modulu *searchInDB.py* nebo do *app.py* pro zobrazení.

Informace z *whois* databáze jsou extrahovány a uloženy do databáze ve funkci *extractWhois(toExtract, ip)*. Dále pak je výsledek zobrazen v terminálu nebo tabulce.

Následující výpis 15 slouží pro přednastavení parametrů pro extrakci dat z odpovědi databáze whois. Pomocí *while* cyklu je vybrán příslušná databáze ze získané odpovědi na základě seznamu *organizations*. Následuje nastavení seznamů pro extrakci klíčových slov. Seznam *params* obsahuje klíčová slova, která jsou určena pro sloupec *ATTRNAME* v tabulce *additionalData*. Druhý seznamem je seznam *patterns*, který obsahuje regulární výrazy pro extrakci hodnot pro ekvivalentní parametry dle předchozí tabulky. Tyto hodnoty jsou pak vloženy do sloupce *DETAILS* ve stejné tabulce.

```
j = 0;
organizations = ['afrinic', 'apnic', 'arin', 'lacnic', 'ripe' ];
org = 5
while j != len(toExtract):
        l = 0
    while l != len(organizations):
                if bool(re.search(organizations[l],
                                  toExtract[j].lower())) == True:
                        org = organizations.index(re.search
                        (organizations[l], toExtract[j].lower())[0])
        if org != 5: break
        l += 1
```

```
    if org != 5: break
    j += 1
    origin = organizations[org].capitalize()
    if org == 0 or org == 1: org = 0 #afrinic, apnic
    if org == 2: org = 1 #arin
    if org == 3: org = 2 #lacnic
    if org == 4: org = 3 #ripe
    params = [['inetnum', 'netname', 'descr', 'country', 'status'],
              ['CIDR', 'Organization', 'StateProv', 'Country'],
              ['inetnum', 'country', 'owner', 'status', 'ownerid'],
              ['inetnum', 'netname', 'descr', 'country',
               'status', 'origin']]
    patterns = [['(\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}-
                                \d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})',
                 '-', '-', '[A-Z][A-Z]', '-'],
                ['(\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}/\d{2})', '-',
                 '[A-Z][A-Z]', '[A-Z][A-Z]'],
                ['(\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}/\d{2})',
                 '[A-Z][A-Z]', '-', '-', '-'],
                ['(\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}
                -\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})',
                 '-', '-', '[A-Z][A-Z]', '-', '-']]
```

Listing 15 Kontrola duplikátních hodnot pro tabulku *honeypotData*

Extrakce dat z odpovědi z databáze *whoisXMLAPI* slouží funkce *checkWhoisXML(url, ip, APIkey)*, který odpověď načte jako tabulku a pomocí funkce *extractWhoisXML(data, ip)* data zpracuje a uloží tabulky *additionalData*. Posledně zmíněná funkce také obsahuje seznam klíčových slov na základě, kterých jsou data extrahována.

4.3.17 App.py

Modul *app.py* slouží pro zobrazení grafického rozhraní pro možnost vyhledávání a zobrazování dat z databáze. Výsledné rozhraní obsahuje vstup pro vyhledávání, tabulku s výslednými daty, indikátory o počtu využitých dotazů na externí databáze, indikátory o stavu aktualizací dat a jejich zpracování. V případě chybějících API klíčů obsahuje i pole pro jejich zadání.

App.py ještě před nastavením GUI využívá modulu *initial.py*, pomocí kterého je vytvořen *managerFile.txt* a pomocí kterého jsou doinstalovány příslušné balíčky. Pokud indikátor pro nainstalované balíčky byl v souboru pro řízení nastaven na 1, tak jsou importovány balíčky pro GUI a nastaveny časovače na základě operačního systému. Jinak je modul ukončen s následujícím výpisem: *The GUI can not be launched\nCheck your internet connection!!\nRun again please.*

Dále je spuštěna hlavní funkce *main()*, ve které jsou pomocí funkce *appBase()* nastaveny základní parametry pro scénu grafického rozhraní. Tyto parametry jsou postoupeny do třídy *App*, která již obsahuje nastavení GUI.

Nastavení objektů scény je realizováno ve funkci *__init__(self, scene)*. Například tady jsou nastaveny komponenty pro rozdělení hlavní scény nebo umístění tlačítek a labelů.

Komponenty této funkce pak využívají další funkce třídy. Vyhledávací tlačítko *checkBtn* například využívá funkci *searchIP(self)* pro vyhledání zadané IP adresy v databázi. Tato funkce je obdoba funkce *queryIP(APIkey, xmlAPIkey, gui, path)* v modulu *searchInDB.py*, který bude popsán níže.

Po zadání IP adresy uživatelem je hodnota postoupena do této funkce, kde je v první řadě zkontrolován formát pomocí funkce *checkIPFormat(ip)* v modulu *searchInDB.py*. Pokud je formát adresy správný, tak je promazán vstup pro zadávání IP adres a také jsou načteny API klíče. V dalších krocích již probíhá vyhledávání v lokální databázi přes všechny tabulky pomocí funkce *checkLocalDB(ip, gui, tableName)*, také z modulu *searchInDB.py*. Pokud je adresa nalezena v databázi, tak je přidána do seznamu *information*. Pokud v lokální databázi nejsou data z *abuseIPDB* nebo *whoisXMLAPI*, tak je zaslán požadavek na tyto databáze a odpověď je uložena do seznamu *information* pro zobrazení a také uložena do lokální databáze.

Výpis níže 16 zobrazuje realizaci dotazu na databázi *abuseIPDB* za podmínky, že nebyl vyčerpán denní limit 1000 dotazů. V dalším kroku je zkontrolováno síťové připojení a je odeslán dotaz na databázi. Po přijetí odpovědi, která obsahuje informace o hledané IP adrese, je odpověď rozdělena na dvě části, seznamy: *items* pro detaily a *columns* pro parametry. Pokud IP adresa v databázi není, tak je do příslušné sekce vypsán výpis: *Requested data are not in abuseIPDB database.*

```
if 1000 - num != 0:
        self.checkConn()
    if self.connectionBtn['background'] != 'red':
                data = searchInDB.checkAbuseDB(num, ip, APIkey)
        result = searchInDB.updateAttempts(num, self.path, 'abuseIPDB')
        num = result[0]
        file = open(self.path, 'r'); r = file.readlines(); file.close()
        self.showAttempts['text'] = str(1000 -
                        int(r[11].replace('\n', '')))+'/1000'
        if 'ipAddress' in str(data):
                        items, columns = searchInDB.divideData(data)
                        searchInDB.addNewInfoToDB(ip, items,
                                columns, self.path)
                        information.append(columns)
                information.append(items)
        if len(data) == 0:
                        self.commenctLabel['text'] =
                                'Requested data are not in
                        abuseIPDB database'
```

Listing 16 Dotaz na databázi abuseIPDB

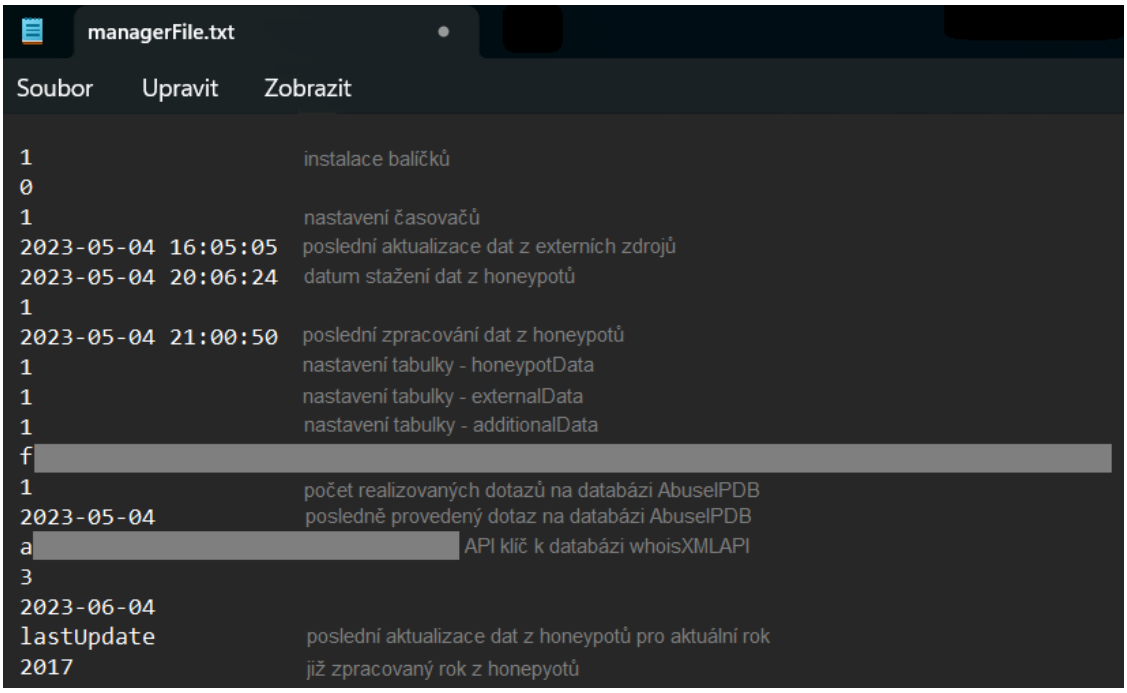
Po získání informací o IP adrese, ať už z lokální nebo externí databáze, je přistoupeno k jejich zobrazení pomocí funkce *printData(self,information, ip)*. Na základě délky seznamu *information* jsou pak příslušné informace, parametry a jejich detaily, zobrazeny do tabulky. Jelikož mohou nastat případy, když data v některé tabulce nebo databázi nebudou, tak pro každou možnost bylo vytvořeno rozdílné ukládání dat do tabulky.

5 OVĚŘENÍ FUNKČNOSTI

Kapitola obsahuje snímky z ověření funkčnosti implementace návrhu z předchozí kapitoly na vybraných operačních systémech.

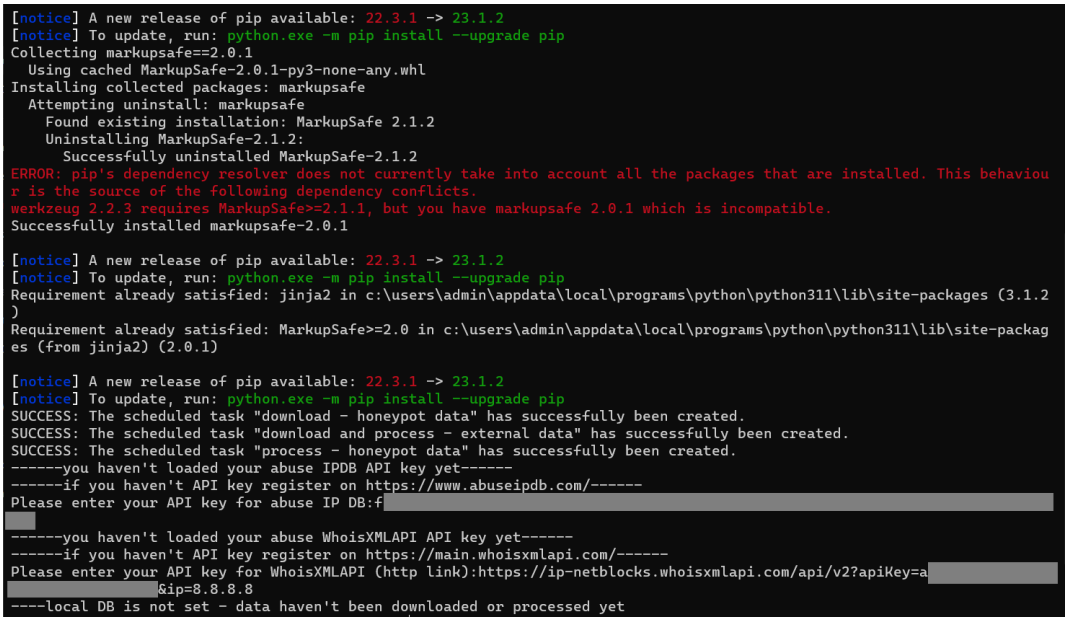
Obrázky budou uvedeny v pořadí dle spouštěných funkcí. Výjimkou je snímek souboru *managerFile.txt*, který je uveden hned na začátku. Dále následují obrázky prvotního spuštění aplikace v terminálu i spuštění aplikace s GUI. Pak následují obrázky vyhledávání v databázích a výpis získaných údajů o hledané IP adrese. Budou uvedeny snímky z nastavení časovačů, stažení, zpracování a ukládání dat. Na závěr budou uvedeny vytvořené tabulky se zpracovanými daty uloženými v tabulce.

Obrázek 5.1 zobrazuje již nastavené hodnoty v souboru pro řízení spolu se stručným slovním popisem.

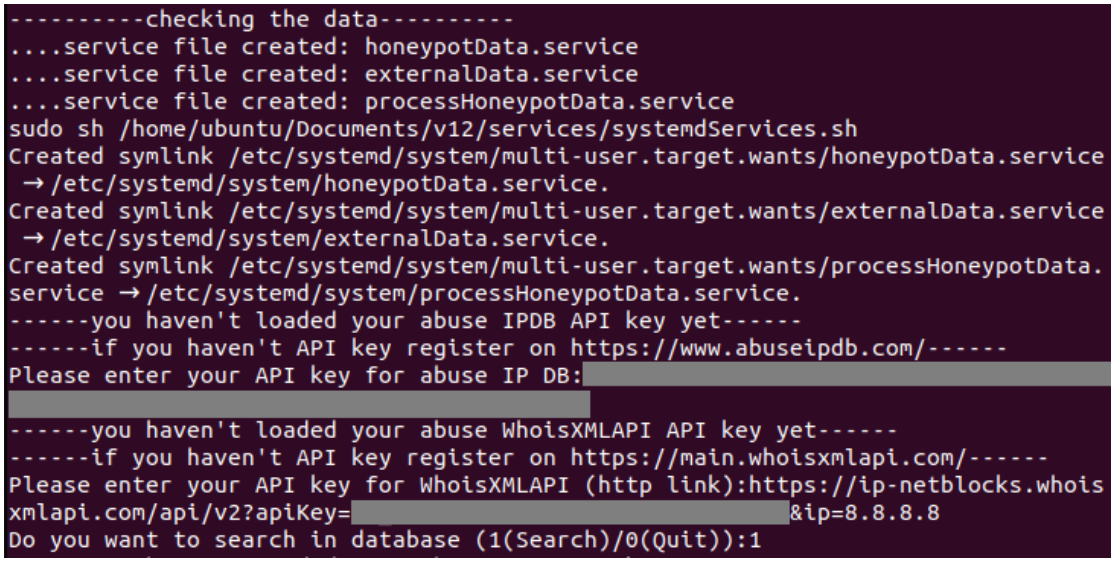


Obrázek 5.1 *ManagerFile.txt* s nastavenými hodnotami

Dalším obrázky 5.2 a 5.3 zobrazují prvotní spuštění aplikace v terminálu pro oba operační systémy, kde jedna z nich obsahuje kontrolu instalovaných balíčků a oba obsahují zadání API klíčů pro jednotlivé databáze. Je nutné podotknout, že při prvním běhu aplikace a v průběhu zpracování dat v databázi vyhledávat nelze.



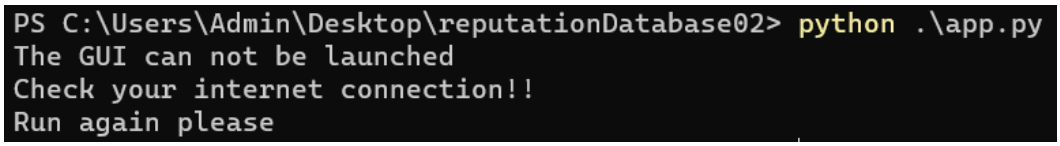
Obrázek 5.2 První spuštění aplikace v terminálu (Windows)



Obrázek 5.3 První spuštění aplikace v terminálu (Ubuntu)

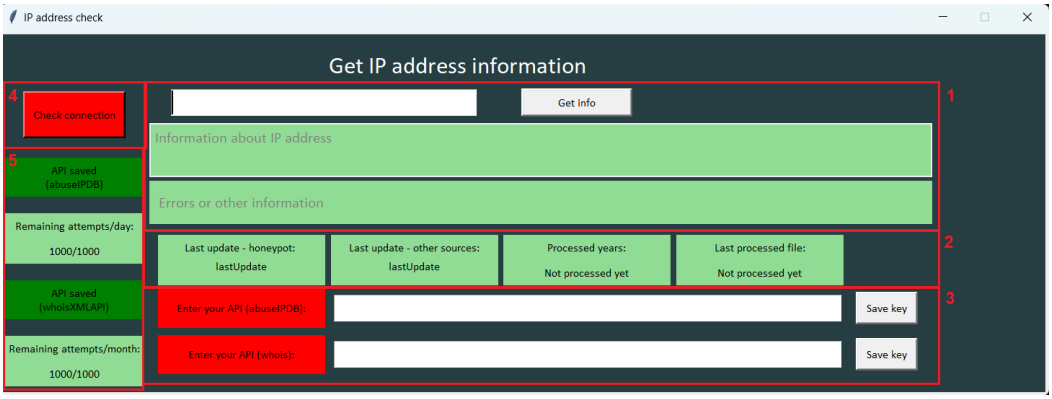
Během testování byla vyzkoušena i vytvořená aplikace, kterou je nutné spoustit příkazem *python app.py* pro operační systém Windows nebo pomocí příkazu *python3 app.py* pro operační systém Ubuntu.

Z důvodu instalace balíčku je potřebné funkční síťové připojení. Jinak aplikace s grafickým rozhraním není spuštěna. Tato událost je zobrazena na obrázku 5.4.



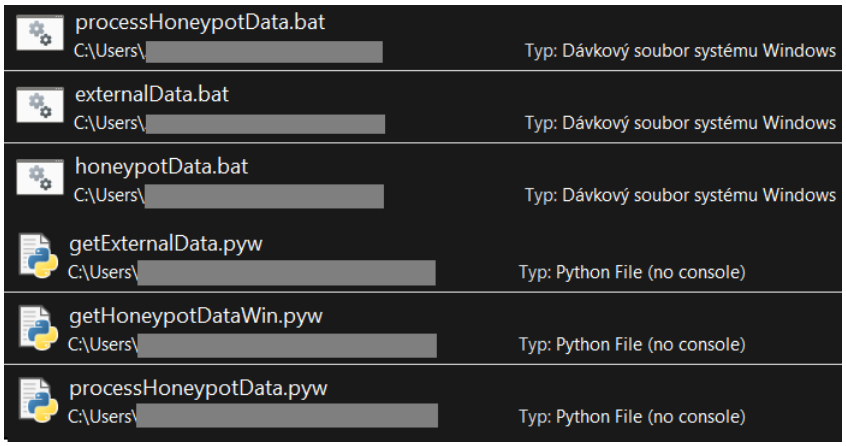
Obrázek 5.4 Spuštění aplikace s GUI bez síťového připojení

Následující snímek 5.5 zobrazuje aplikaci s GUI před nastavením časovačů a zadáním API klíčů. Aplikaci lze rozdělit na čtyři části. První část s hodnotou 1 slouží pro vyhledávání IP adres a zobrazování informací o nich. Část druhá zobrazuje informace o stažených a zpracovaných datech. Zadávání API klíčů patří do třetí části a je zobrazeny pouze v případě chybějících klíčů. Jinak tato část není zobrazena. Aplikace obsahuje tlačítko pro kontrolu síťového připojení, které také indikuje, jestli je síťové připojení funkční. Poslední, pátá, část obsahuje načtené hodnoty zbylých dotazů pro databáze *AbuseIPDB* a *whoisXMLAPI*. Pro první běh aplikace v grafické verzi také není dovoleno vyhledávání v databázi.



Obrázek 5.5 První spuštění aplikace (GUI)

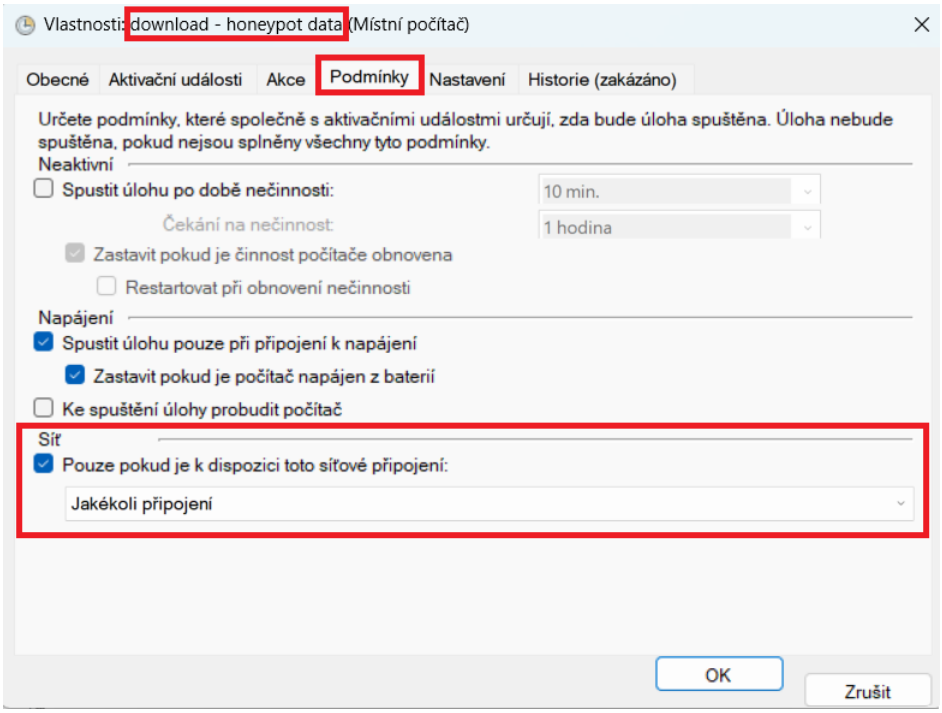
Po prvním spuštění aplikace jsou nastaveny časovače pro oba operační systémy. Obrázek 5.6 zobrazuje zmíněné soubory pro automatickou aktualizaci a zpracování dat. Soubory s koncovkou *.bat* a *.pyw* pro stažení a zpracování dat jsou umístěny pracovní složky aplikace.



Obrázek 5.6 Soubory pro automatickou aktualizaci a zpracování dat (Windows)

Je potřebné dodat, že pro operační systém Windows nebylo možné zajistit plnou automatizaci nastavení časovačů. Proto bylo potřebné dodatečné manuální nastavení úkolu ihned po prvním spuštění aplikace v *Plánovači úloh*. Po výběru příslušného úkolu

je nutné doplnit spuštění úkolu až v případě existence síťového připojení. Snímek 5.7 zobrazuje příslušnou část nastavení spuštění úkolu.



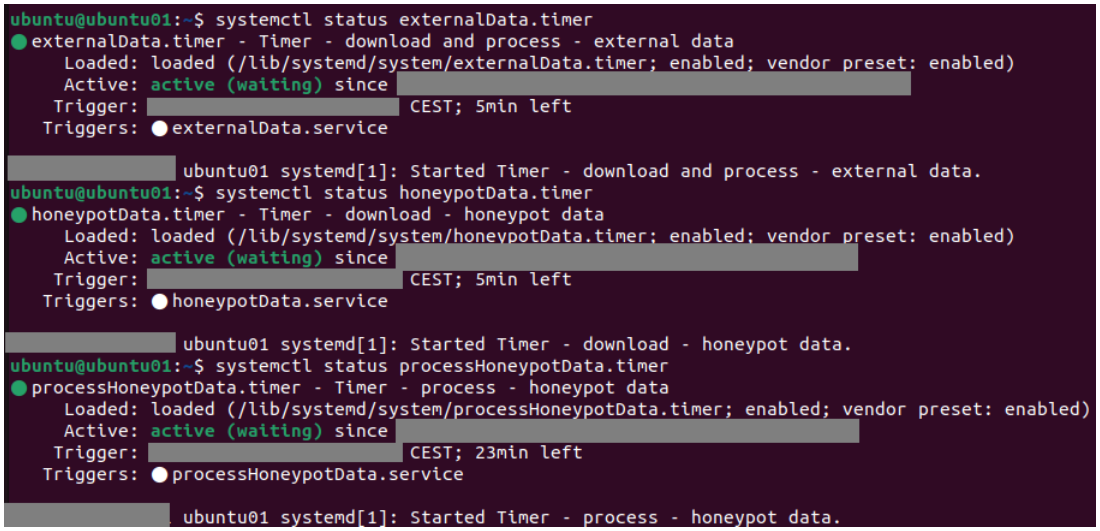
Obrázek 5.7 Nastavení spuštění časovačů až v případě existence síťového připojení (Windows)

Po vytvoření souboru pro řízení, instalaci balíčků byly vytvořeny služby a k nim příslušné časovače, které jsou zobrazeny na obrázku 5.8, který zobrazuje výpis aktuálního stavu pomocí příkazu *systemctl status <název_časovače>.timer*, obrázek 5.9.

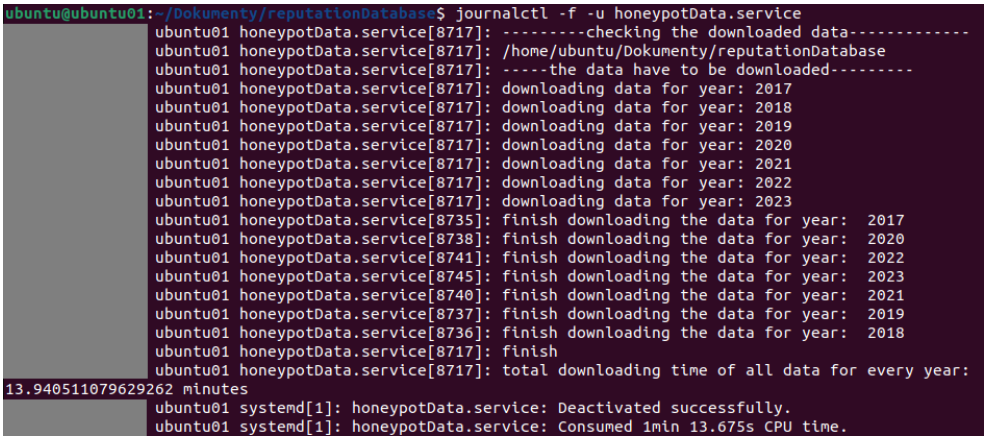
```
ubuntu@ubuntu01:~$ systemctl list-unit-files --type=timer
UNIT FILE                          STATE   VENDOR PRESET
anacron.timer                      enabled enabled
apport-autoreport.timer            enabled enabled
apt-daily-upgrade.timer            enabled enabled
apt-daily.timer                    enabled enabled
dpkg-db-backup.timer               enabled enabled
e2scrub_all.timer                  enabled enabled
externalData.timer                 enabled enabled
fstrim.timer                       enabled enabled
fwupd-refresh.timer                enabled enabled
honeypotData.timer                 enabled enabled
logrotate.timer                    enabled enabled
man-db.timer                       enabled enabled
motd-news.timer                    enabled enabled
processHoneypotData.timer          enabled enabled
snapd.snap-repair.timer            enabled enabled
systemd-tmpfiles-clean.timer       static  -
ua-timer.timer                     enabled enabled
update-notifier-download.timer     enabled enabled
update-notifier-motd.timer         enabled enabled

19 unit files listed.
```

Obrázek 5.8 Nastavené časovače pro operační systém Ubuntu

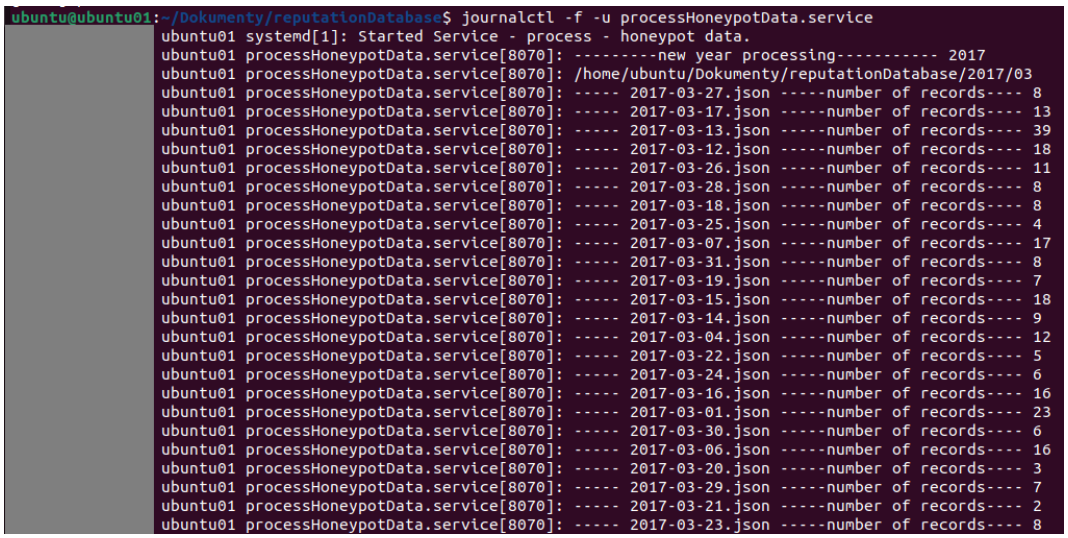
Snímek 5.10 zobrazuje stažení dat z honeypotů pomocí nastavených časovačů a na obrázku 5.11 je vypsán průběh zpracování dat z externích zdrojů.



Obrázek 5.9 Stav časovačů pro operační systém Ubuntu



Obrázek 5.10 Stažení dat z honeypotů (Ubuntu)



Obrázek 5.11 Zpracování a ukládání dat z honeypotů

Následující obrázky zobrazují vytvořené tabulky v lokální databázi. Každý obrázek

je vytvořen ze dvou částí: tabulka pro operační systém Ubuntu a tabulka pro operační systém.

Tabulka *honeypotData* je zobrazena na obrázku 5.12. Dodatečné informace jsou zobrazeny na obrázku 5.13 v tabulce *additionalData*. Obrázek 5.14 zobrazuje IP adresy získané z blacklistů a blocklistů uložených v tabulce *externalData*.

Databázová Struktura | Prohlížet data | Editovat Pragma | Proveďte SQL

Tabulka: honeypotData — Filter in any column

	IP	TIME	COMMANDS	COUNTRY
	Filtr	Filtr	Filtr	Filtr
1	195.178.120.188	2023-01-02T00:00:01.496006+00:00		NULL
2	193.169.255.30	2023-01-02T00:00:02.257140+00:00		NULL
3	107.189.31.234	2023-01-02T00:00:11.444156+00:00	uname -s -m	US
4	37.235.54.63	2023-01-02T00:00:26.110690+00:00	whoami	AT
5	37.44.238.78	2023-01-02T00:01:26.231909+00:00	cd /tmp || cd /var/run || cd /mnt || cd /root ...	RU
6	175.195.119.54	2023-01-02T00:01:33.755522+00:00		KR
7	61.50.139.218	2023-01-02T00:01:53.555784+00:00		CN
8	178.88.161.82	2023-01-02T00:02:02.863664+00:00		KZ
9	103.215.127.5	2023-01-02T00:02:30.491429+00:00	curl -s -L http://download.c3pool.org/...	HK

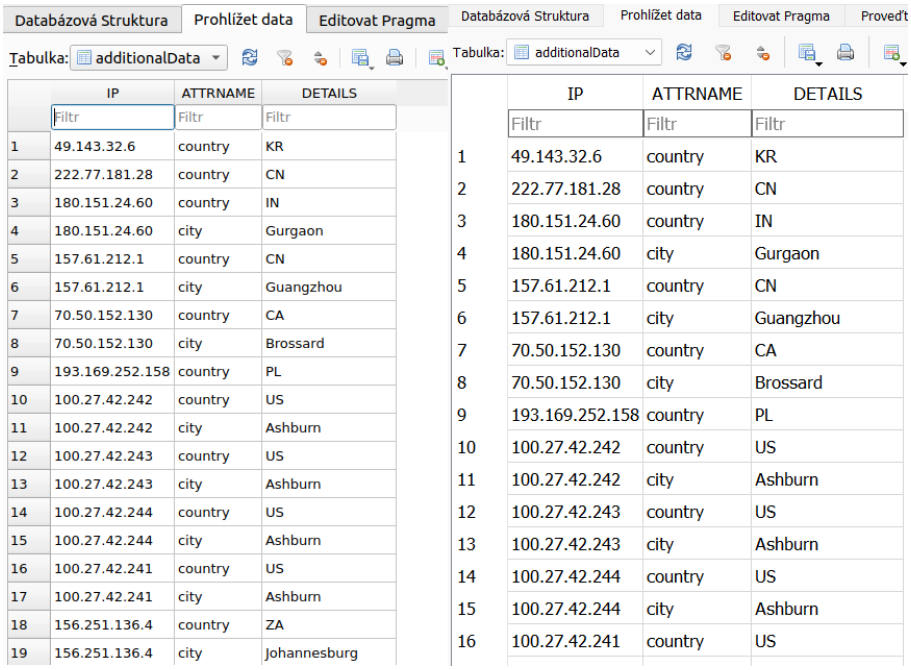
Databázová Struktura | Prohlížet data | Editovat Pragma | Proveďte SQL

Tabulka: honeypotData — Filter in any column

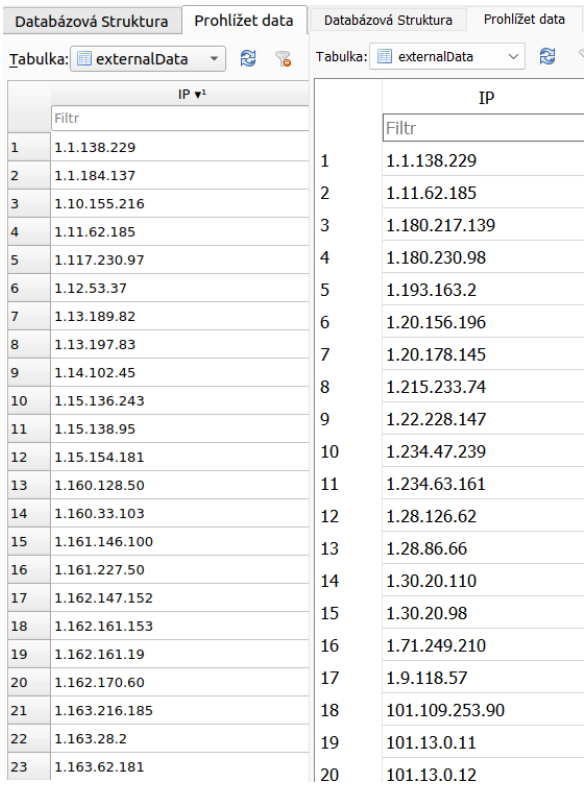
	IP	TIME	COMMANDS	COUNTRY
	Filtr	Filtr	Filtr	Filtr
1	127.0.0.1	2017-10-03T11:49:46+00:00	ls -la;unset HISTORY HISTFILE ...	
2	187.115.73.70	2017-01-28T02:12:57+00:00	unset HISTORY HISTFILE HISTSAVE...	BR
3	109.236.86.63	2017-01-28T03:05:37+00:00	/qweerwe323f', 'sudo /bin/sh', '/...	NL
4	104.251.176.63	2017-02-15T02:52:07+00:00	ls -la /var/run/gcc.pid', 'wget http:/...	US
5	188.166.241.178	2017-01-28T03:53:00+00:00	unset HISTORY HISTFILE HISTSAVE...	SG
6	104.251.177.143	2017-02-04T19:53:19+00:00	wget http://104.251.177.143/s443ls ...	US
7	189.44.10.114	2017-01-28T10:49:32+00:00	unset HISTORY HISTFILE HISTSAVE...	BR
8	129.144.52.191	2017-02-06T17:44:13+00:00	cd /tmp', 'rm -rf *sh', 'cd /tmp', 'cd...	US
9	88.122.101.183	2017-01-28T20:43:04+00:00	w ', 'ps x', 'cat /etc/passwd', 'reboot	FR
10	112.74.76.94	2017-01-29T03:25:41+00:00	unset HISTORY HISTFILE HISTSAVE...	CN

Obrázek 5.12 Tabulka *honeypotData*

Při dalším spuštění je již spuštěna možnost vyhledávání (hodnota jedna), viz obrázky 5.15 a 5.16. Před samotným vyhledáváním jsou načteny již zadané klíče a je proveden výpis zbývajících dotazů na příslušné databáze.

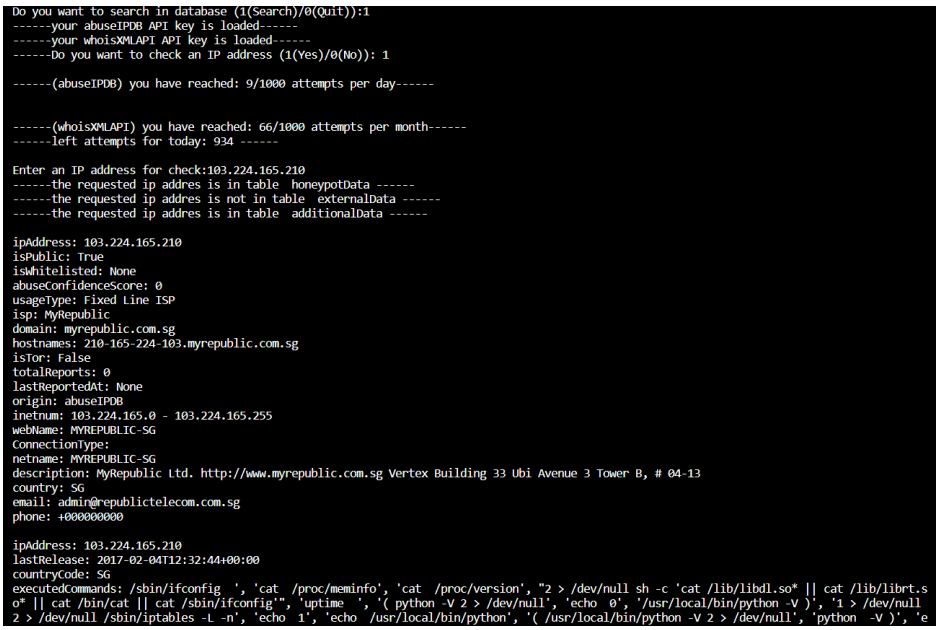


Obrázek 5.13 Tabulka *additionaData*

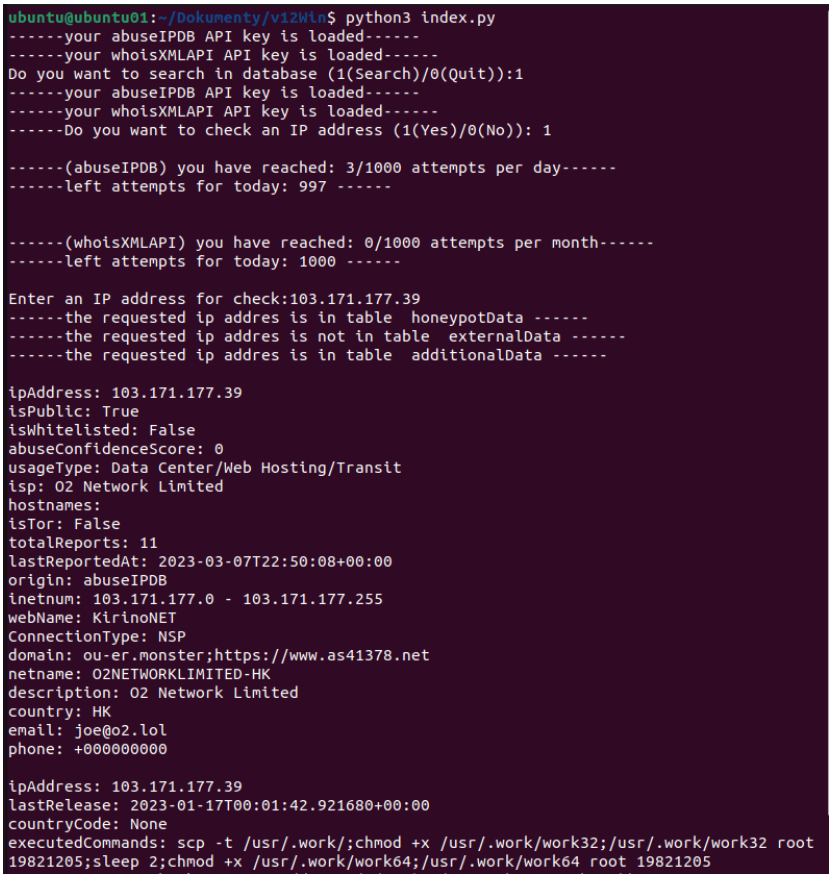


Obrázek 5.14 Tabulka *externalData*

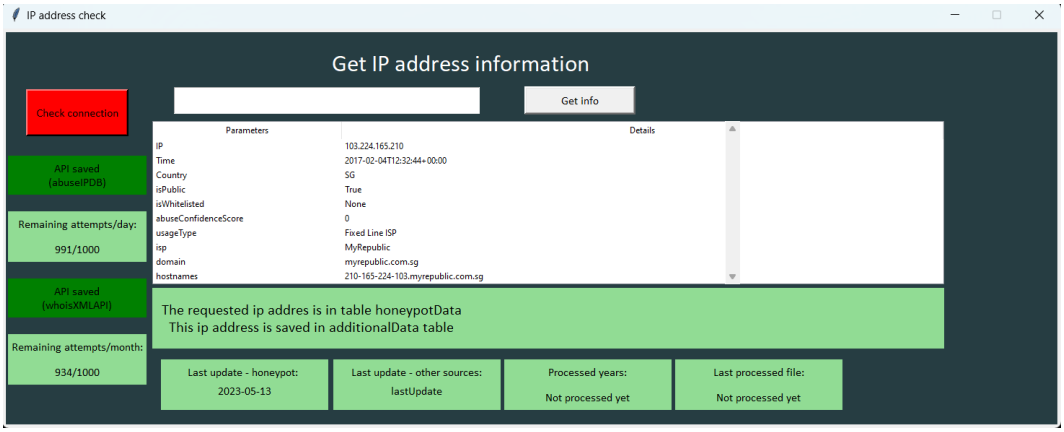
Na obrázku 5.17 je již zobrazena aplikace po zadání IP adresy a obdržení výsledku pomocí tabulky. Část dvě již také obsahuje načtenou hodnotu poslední aktualizace, avšak všechna data zatím nebyla zpracována.



Obrázek 5.15 Vyhledávání v databázi - terminál (Windows)



Obrázek 5.16 Vyhledávání v databázi - terminál (Ubuntu)



Obrázek 5.17 Běh aplikace (GUI)

ZÁVĚR

Cílem diplomové práce bylo navrhnout a implementovat reputační databázi pro IP adresy s využitím dat z dat honeypotů a ověřit její funkčnost v testovacím prostředí.

V teoretické části byl popsán vývoj kybernetických útoků v posledních pěti letech a byly přiblíženy možné nástroje pro detekci útočníků. První kapitola *Vývoj kybernetických útoků v posledních 5-ti letech* obsahuje stručný vývoj kybernetických útoků rozdělených do kategorií a jejich následné srovnání. Druhá kapitola *Nástroje pro detekci útočníků* obsahuje vybrané způsoby detekce útočníků, mezi které lze zařadit monitoring, honeypoty a detekční systémy.

Na začátku praktické části byly stručně charakterizovány vybrané zdroje dat pro tvorbu reputační databáze, kdy byly zmíněny zdroje jako *AbuseIPDB*, *WhoisXML API* nebo zdroje ze stránek *CSIRT.CZ*.

Ve čtvrté kapitole, *Tvorba reputační databáze pro IP adresy*, byl rozepsán návrh pro tvorbu reputační databáze pro jednotlivé funkční bloky aplikace a byla vytvořena implementace návrhu ve formě finální aplikace, která byla zdokumentována v páté kapitole.

Na závěr je k práci přiloženo DVD s elektronickou verzí diplomové práce, videem a s vytvořenou aplikací.

SEZNAM POUŽITÉ LITERATURY

[1] ENISA Threat Landscape Report 2017 [online]. https://www.enisa.europa.eu/publications/enisa-threat-landscape-report-2017, January 15, 2018, [cit. 2023-02-05].

[2] ENISA Threat Landscape Report 2018 [online]. https://www.enisa.europa.eu/publications/enisa-threat-landscape-report-2018, January 28, 2019, [cit. 2023-02-05].

[3] ENISA Threat Landscape 2020 - Phishing [online]. https://www.enisa.europa.eu/topics/cyber-threats/threats-and-trends/etl-review-folder/etl-2020-cryptojacking, October 20, 2020, [cit. 2023-02-05].

[4] ENISA Threat Landscape 2020 - Web-based attacks [online]. https://www.enisa.europa.eu/publications/web-based-attacks, October 20, 2020, [cit. 2023-02-05].

[5] ENISA Threat Landscape 2020 - Web application attacks [online]. https://www.enisa.europa.eu/publications/web-application-attacks, October 20, 2020, [cit. 2023-02-05].

[6] ENISA Threat Landscape 2020 - Distributed denial of service [online]. https://www.enisa.europa.eu/publications/enisa-threat-landscape-2020-distributed-denial-of-service, October 20, 2020, [cit. 2023-02-05].

[7] ENISA Threat Landscape 2020 - Botnet [online]. https://www.enisa.europa.eu/publications/enisa-threat-landscape-2020-botnet, October 20, 2020, [cit. 2023-02-05].

[8] ENISA Threat Landscape 2020 - Spam [online]. https://www.enisa.europa.eu/publications/spam, October 20, 2020, [cit. 2023-02-05].

[9] ENISA Threat Landscape 2020 - Cryptojacking [online]. https://www.enisa.europa.eu/publications/enisa-threat-landscape-2020-cryptojacking, October 20, 2020, [cit. 2023-02-05].

[10] ENISA Threat Landscape 2020 - Ransomware [online]. https://www.enisa.europa.eu/publications/ransomware, October 20, 2020, [cit. 2023-02-05].

[11] ENISA Threat Landscape 2020 - Malware [online]. https://www.enisa.europa.eu/publications/malware, October 20, 2020, [cit. 2023-02-05].

[12] ENISA Threat Landscape 2021 [online].https://www.enisa.europa.eu/publications /enisa-threat-landscape-2021, October 27, 2021, [cit. 2023-02-05].

[13] ENISA Threat Landscape 2022 [online].https://www.enisa.europa.eu/publications /enisa-threat-landscape-2022, November 03, 2022, [cit. 2023-02-05].

[14] Proč je kybernetická bezpečnost pro EU důležitá? A kolik nás stojí kybernetické útoky? [online]. https://www.europarl.europa.eu/news/cs/headlines/society/20211008STO14521 /proc-je-kyberneticka-bezpecnost-dulezita-a-kolik-nas-stoji-kyberneticke-utoky, November 28, 2022, [cit. 2023-02-05].

[15] Shamoon: Destructive Threat Re-Emerges with New Sting in its Tail [online]. https://symantec-enterprise-blogs.security.com/blogs/threat-intelligence/shamoon-destructive-threat-re-emerges-new-sting-its-tail, December 14, 2018, [cit. 2023-02-05].

[16] PETER, Eric a Todd SCHILLER. A Practical Guide to Honeypots [online].https://www.cse.wustl.edu/ jain/cse571-09/ftp/honey/index.html, 2008, 19 [cit. 2023-02-22].

[17] JOSHI, R. C. a Anjali SARDANA. Honeypot : a new paradigm to information security [online]. https://learning.oreilly.com/library/view/honeypots/9781578087082/, CRC Press, 2011 [cit. 2023-02-22]. ISBN 9781578087082.

[18] TITARMARE, Neha, Nayankumar HARGULE a Anand GUPTA. An Overview of Honeypot Systems. INTERNATIONAL JOURNAL OF COMPUTER SCIENCES AND ENGINEERING [online]. 2019, February 2019, 2019, 5 [cit. 2023-02-22]. ISSN 2347-2693. Dostupné z: doi:10.26438/ijcse/v7i2.394397

[19] REGALADO, Daniel, Shon HARRIS, Allen HARPER, Chris EAGLE, Jonathan NESS, Branko SPASOJEVIC, Ryan LINN a Stephen SIMS. Gray Hat Hacking The Ethical Hacker's Handbook, Fifth Edition, 5th Edition [online]. https://learning.oreilly.com/library/view/gray-hat-hacking/9781260108422/, 5th. McGraw-Hill, 2018 [cit. 2023-02-22]. ISBN 9781260108422.

[20] BASER, Melike, Ebu Yusuf GUVEN a Muhammed Ali AYDIN. SSH and Telnet Protocols Attack Analysis Using Honeypot Technique: Analysis of SSH AND

TELNET Honeypot. 2021 6th International Conference on Computer Science and Engineering (UBMK) [online]. IEEE, 2021, 13 October 2021, 806-811 [cit. 2023-03-03]. ISBN 978-1-6654-2908-5. Dostupné z: doi:10.1109/UBMK52708.2021.9558948

[21] Evading IDS, Firewalls, and Honeypots [online]. https://www.academia.edu/19741965/CEHv9_Module_16_Evading_IDS_Firewalls_and_Honeypots_1, 2015, 105 [cit. 2023-02-22].

[22] ANSON, Steve. Applied Incident Response [online]. https://learning.oreilly.com/library/view/applied-incident-response/9781119560265/, Wiley, 2020 [cit. 2023-02-24]. ISBN 9781119560265.

[23] The best tools and why network monitoring is important. Network king [online]. https://network-king.net/best-network-monitoring-tools/, January 30, 2023 [cit. 2023-02-24].

[24] A. D'Alconzo, I. Drago, A. Morichetta, M. Mellia and P. Casas, "A Survey on Big Data for Network Traffic Monitoring and Analysis,"in IEEE Transactions on Network and Service Management, vol. 16, no. 3, pp. 800-813, Sept. 2019, doi: 10.1109/TNSM.2019.2933358.

[25] Basics of Network Monitoring. Manage Engine [online]. https://www.manageengine.com/network-monitoring/basics-of-network-monitoring.html [cit. 2023-02-24].

[26] ORIYANO, Sean-Philip. Penetration Testing / Ethical Hacking [online]. https://learning.oreilly.com/library/view/penetration-testing-essentials/9781119235309/, Sybex, 2016 [cit. 2023-02-27]. ISBN 9781119235309.

[27] Intrusion Detection System (IDS): Signature vs. Anomaly-Based. N-Able [online]. https://www.n-able.com/blog/intrusion-detection-system, 21. březen 2021 [cit. 2023-02-28].

[28] ERNEY, Tristan a MD Minhaz CHOWDHURY. A Survey of Intrusion Detection and Prevention Systems. 2022 IEEE World AI IoT Congress (AIIoT) [online]. IEEE, 2022, 13 July 2022, 578-584 [cit. 2023-03-02]. ISBN 978-1-6654-8453-4. Dostupné z: doi:10.1109/AIIoT54504.2022.9817348

[29] MASDARI, Mohammad a Hemn KHEZRI. A survey and taxonomy of the fuzzy signature-based Intrusion Detection Systems. Applied Soft Computing [online]. 2020, 92 [cit. 2023-03-02]. ISSN 15684946. Dostupné z: doi:10.1016/j.asoc.2020.106301

[30] MUDZINGWA, David a Rajeev AGRAWAL. A study of methodologies used in intrusion detection and prevention systems (IDPS). 2012 Proceedings of IEEE Southeastcon [online]. IEEE, 2012, 2012, 1-6 [cit. 2023-03-03]. ISBN 978-1-4673-1375-9. Dostupné z: doi:10.1109/SECon.2012.6197080

[31] Globální Statistika. HaaS [online]. [cit. 2023-04-18]. Dostupné z: https://haas.nic.cz/stats/

[32] AbuseIPDB making the internet safer, one IP at a time. Www.abuseipdb.com [online]. [cit. 2023-04-18]. Dostupné z: https://www.abuseipdb.com/

[33] Introduction. Www.abuseipdb.com [online]. [cit. 2023-04-18]. Dostupné z: https://docs.abuseipdb.com/#introduction

[34] IPv4 179.43.163.131 [online]. [cit. 2023-04-18]. Dostupné z: https://otx.alienvault.com/indicator/ip/179.43.163.131

[35] DirectConnect API [online]. [cit. 2023-04-18]. Dostupné z: https://otx.alienvault.com/api

[36] My IP Address [online]. 2012 [cit. 2023-04-18]. Dostupné z: https://myip.ms/

[37] Top 35 Largest Most Popular Web Hosting Companies in the World. My IP Address [online]. 2012 [cit. 2023-04-18]. Dostupné z: https://myip.ms/info/top_hosting/TOP_Best_World_Web_Hosting-_Companies_Real_Time_Statistics.html

[38] Reports in Excel. My IP Address [online]. 2012 [cit. 2023-04-18]. Dostupné z: https://myip.ms/browse/reports/Files_Reports.html

[39] Google bot visit time to your website. My IP Address [online]. 2012 [cit. 2023-04-18]. Dostupné z: https://myip.ms/info/bot/Google_Yahoo_Bing_Bots_Last_Visit_Date.html

[40] Free GEO information box for your site. My IP Address [online]. 2012 [cit. 2023-04-18]. Dostupné z: https://myip.ms/info/freebox/Free_GEO_Information_Box_for_Your_Site.html

[41] Latest visitors statistics box for your website. My IP Address [online]. 2012 [cit. 2023-04-18]. Dostupné z: https://myip.ms/info/visitsbox/Latest_Visitors_Statistics_Box_for_Your-_Website.html

[42] Cities SQL database for your site. My IP Address [online]. 2012 [cit. 2023-04-18]. Dostupné z: https://myip.ms/info/cities_sql_database/World_Cities_SQL_Mysql_Database-_2023_year.html

[43] DNSlytics [online]. [cit. 2023-04-18]. Dostupné z: https://dnslytics.com/

[44] Premium website access. DNSlytics [online]. [cit. 2023-04-18]. Dostupné z: https://dnslytics.com/premium-access

[45] IPAddress.com [online]. 2010 [cit. 2023-04-18]. Dostupné z: https://www.ipaddress.com/

[46] American registry for internet numbers [online]. 1997 [cit. 2023-04-18]. Dostupné z: https://www.arin.net/

[47] Robtex [online]. 2020 [cit. 2023-04-18]. Dostupné z: https://www.robtex.com/

[48] Zdroje dat. CSIRT.CZ [online]. 2019 [cit. 2023-04-18]. Dostupné z: https://csirt.cz/cs/zdroje-dat/

[49] Binary Defense [online]. 2022 [cit. 2023-04-18]. Dostupné z: https://www.binarydefense.com/

[50] WhoisXMLAPI [online]. 2022 [cit. 2023-04-18]. Dostupné z: https://main.whoisxmlapi.com/

[51] Dan.me.uk [online]. 2022 [cit. 2023-04-18]. Dostupné z: https://www.dan.me.uk/

[52] BEIGH, Bilal Maqbool. Intrusion Detection and Prevention System: Classification and Quick Review. ARPN Journal of Science and Technology [online]. 2012, 2(7), 15 [cit. 2023-03-02]. ISSN 2225-7217. Dostupné z: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=e66f83a8cea534bcd8b7261ee74b34d18745deac

[53] SHIRSATH, Vaishali. A Survey on Current States of Honeypots and Deception Techniques for Attack Capture. INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) [online]. 2021, 22-02-2021, 2020(Volume 09 – 03), 6 [cit. 2023-02-23]. ISSN 2278-0181. doi:10.17577/IJERTCONV9IS03092

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

ADB	Android Debug Bridge
APDoS	Advanced Persistent DoS
API	Application Programming Interface
ARIN	American Registry for Internet Numbers
CDC	Centers for Disease Control and Prevention
C&C	Command and control server
DB	Database
DDoS	Distributed Denial of Service
DLang	D language
DNS	Domain Name System
DoS	Denial of Service
ENISA	European Union Agency for Network and Information Security
FTP	File Transfer Protocol
GPON	Gigabit Passive Optical Network
HIDS	Host-based Intrusion Detection System
HNAP	Home Network Administration Protocol
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IAB	Initial Access Broker
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
IoT	Internet of things
ISO	Identical Storage Image of optical media
IT	Informační technologie
IPS	Intrusion Prevention System
JSON	JavaScript Object Notation
LFI	Local File Inclusion
NIDS	Network-based Intrusion Detection System
OWASP	The Open Web Application Security Project
RAT	Remote Access Trojan
PDF	Portable Document Format
PDoS	Permanent Denial of Service
PHP	Personal Home Page Tools
PUP	Potentially Unwanted Program
P2P	Peer-to-peer
RAR	Roshal ARchive
RDP	Remote Desktop Protokol
REST	REpresentational State Transfer

RFI	Remote File Inclusion
RPC	Remote Procedure Call
RST	Reset
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
SSH	Secure Socket Shell
SSL	Secure Sockets Layer
SYN	Synchronize
TLS	Transport Layer Security
UDP	User datagram protocol
URL	Uniform Resource Locators
UEFI	Unified Extensible Firmware Interface
VBS	Virtual Basic Script
VHDX	Hyper V virtual hard disk
VPN	Virtual Private Network
WHO	World Health Organization
WMI	Windows Management Instrumentation
WS-Discovery	Web Service-Discovery Protocol
XML	Extensible Markup Language
XSS	Cross-site Scripting
ZIP	ZIP File Format

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

Listings