VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ANALÝZA MALWARE NA ÚROVNI SÍŤOVÝCH TOKŮ

MALWARE ANALYSIS AT THE NETWORK FLOW LEVEL

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **ŠIMON BRÁZDA**
AUTHOR

VEDOUCÍ PRÁCE **Ing. DANIEL POLIAKOV,**
SUPERVISOR

BRNO 2024

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Zadání bakalářské práce

156963

Ústav: Ústav informačních systémů (UIFS)
Student: **Brázda Šimon**
Program: Informační technologie
Název: **Analýza malware na úrovni síťových toků**
Kategorie: Počítačové sítě
Akademický rok: 2023/24

Zadání:

1. Seznamte se s metodami analýzy síťového provozu na úrovni síťových toků. Soustřeďte se zejména na využití statistických informací aplikovatelných i na šifrovaný síťový provoz.
2. Prozkoumejte existující datasety síťového provozu malwaru. Vyberte vhodný dataset pro klasifikační experimenty.
3. Prozkoumejte charakteristiky a modely použitelné pro klasifikaci. Nad vybraným datasetem navrhněte experiment, který by měl validovat použitelnost metod.
4. Proveďte experimenty. V rámci experimentů natrénujte model a vyhodnoťte klasifikační metriky.
5. Diskutujte využitelnost výsledků.

Literatura:

- Sikorski, M.; Honig, A.: Practical Malware Analysis: The Hands-on Guide to Dissecting Malicious Software. No Starch Press. 2012. ISBN: 978-1593272906
- J. Svoboda, I. Ghafir, V. Prenosil. Network Monitoring Approaches: An Overview. In Proceedings of International Conference on Advances in Computing, Communication and Information Technology. Birmingham, UK, 2015. ISBN: 978-1-63248-061-3
- Uhříček, D. Detection of IoT Malware in Computer Networks. Master's thesis, Czech Technical University in Prague, Faculty of Information Technology, 2021. Available from: https://dspace.cvut.cz/handle/10467/94501

Při obhajobě semestrální části projektu je požadováno:
Splnění bodů 1 až 3 zadání. Rozpracování bodu 4.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Poliakov Daniel, Ing.**
Vedoucí ústavu: Kolář Dušan, doc. Dr. Ing.
Datum zadání: 1.11.2023
Termín pro odevzdání: 9.5.2024
Datum schválení: 25.10.2023

Abstrakt

Tato práce se zabývá průzkumem volně dostupných datových sad a zkoumání jejich použitelnosti ke trénování modelů strojového učení. K extrahování dat z datové sady byl využit nástroj ipfixprobe a pro další implementaci jazyk Python. V teoretické části jsou probrány základní aplikační protokoly, možnosti sledování sítě na úrovni toků. Dále byly rozebrány různé druhy malware a typy modelů strojového učení použitelné ke klasifikaci síťových toků. Následně byly tyto modely použity k otestování použitelnosti vybrané datové sady, která tímto byla potvrzena.

Abstract

This thesis explores freely available datasets and investigates their applicability to training machine learning models. The ipfixprobe tool was used to extract data from the dataset and the Python language was used for further implementation. In the theoretical part, basic application protocols, network monitoring capabilities at the flow level are discussed. Furthermore, different types of malware and types of machine learning models applicable to network flow classification were discussed. Subsequently, these models were used to test the applicability of the selected dataset, which was thus validated.

Klíčová slova

malware, síťový tok, IPFIX, model strojového učení, datová sada

Keywords

malware, network flow, IPFIX, machine learning model, dataset

Citace

BRÁZDA, Šimon. *Analýza malware na úrovni síťových toků*. Brno, 2024. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Daniel Poliakov,

Analýza malware na úrovni síťových toků

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana inženýra Daniela Poliakova. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Šimon Brázda
8. května 2024

Poděkování

Rád bych poděkoval svému vedoucímu Ing. Danielovi Poliakovi za poskytnuté rady a rodině a blízkým za jejich trpělivost.

Obsah

Seznam obrázků

Úvod

Kybernetická bezpečnost je v dnešní digitální době jednou z největších výzev, kdy kybernetickým útokům musí denně čelit jednotlivci i organizace. Jednou z klíčových hrozeb je malware, neboli škodlivý software, který může způsobit vážné škody, pokud není včasně detekován. Efektivní detekce malwaru je zásadní pro ochranu dat, správné fungování informačních systémů i ochranu integrity sítí.

Tato práce se zaměřuje na analýzu malwaru na úrovni síťových toků v šifrovaném provozu. U škodlivého softwaru není zkoumán jeho kód, ale analyzují se způsoby, jakými se šíří a jak se chová v rámci síťové komunikace, jelikož malware často zanechává specifické stopy v síťovém provozu, které mohou být využity k jeho efektivní a velmi brzké detekci.

V praktické části jsou analyzovány reálné, volně dostupné datové sady obsahující zachycený síťový provoz. Hlavním cílem práce je vyvinout a otestovat pokročilé metody pro identifikaci malwarových aktivit na základě analýzy síťových toků. K tomu jsou využity moderní techniky strojového učení a zpracování dat. Práce se zaměřuje na zkoumání charakteristik malwaru, a jak se projevuje v síťové komunikaci.

Kapitola 1

Síťový provoz

1.1 Aplikační protokoly

Aplikační protokoly jsou soubory standardů a pravidel, kterými se řídí komunikace mezi aplikacemi koncových uživatelů přes síť. Poskytují specifické služby a funkce pro podporu různých typů komunikace na úrovni aplikací, včetně přenosu souborů, e-mailu, vzdálených terminálových připojení a prohlížení webu. Mezi příklady protokolů aplikační vrstvy v počítačových sítích patří například HTTP, FTP, SMTP, DNS, Telnet, SSH, NFS, SNMP, DHCP a NTP[58].

Pro zajištění integrity a důvěryhodnosti dat přenášených po síti jsou důležité bezpečnostní protokoly. Ty pracují na různých vrstvách modelu OSI[1], což je referenční model pro komunikaci aplikací v sítích.

Na síťové vrstvě je Internet Protocol Security (IPsec) sada protokolů a algoritmů, které zabezpečují data přenášené přes veřejné sítě. Šifruje a ověřuje síťové pakety a zajišťuje tak bezpečnost na úrovni IP. IPsec původně obsahoval protokoly ESP a AH, které šifrují data a zajišťují ověřování, zatímco autentizační hlavička (AH) nabízí funkce proti reprodukování a chrání integritu dat. Sada je nyní rozšířena o protokol IKE (Internet Key Exchange), který poskytuje sdílené klíče vytvářející bezpečnostní asociace (SA). Ty umožňují šifrování a dešifrování prostřednictvím brány firewall nebo směrovače[12].

Na transportní vrstvě jsou protokoly SSl a TLS protokoly, které zabezpečují data přenášená po síti. Protokol SSL ověřuje server pomocí handshake, vyjednává parametry bezpečnostní relace a generuje klíče relace. Poté může bezpečně přenášet data tím, že ověří jejich původ. TLS je protokol založený na protokolu SSL definovaný organizací IETF, který zajišťuje bezpečnou komunikaci přes internet. Protokoly SSL a TLS se běžně používají při prohlížení webu, v elektronické poště a dalších aplikacích, které vyžadují bezpečný přenos dat[12].

Na aplikační vrstvě jsou protokoly SFTP (Secure File Transfer Protocol), PGP (Pretty Good Privacy) a SMP (Secure Messaging Protocol), které zajišťují bezpečnost a soukromí dat přenášených po síti. SFTP zajišťuje, že si dva počítače mohou bezpečně vyměňovat téměř libovolné soubory, zatímco PGP je bezpečný program, který umožňuje uživatelům šifrovat e-maily, soubory a adresáře. SMP umožňuje vzdáleným uživatelům posílat si navzájem zprávy a soubory bezpečně a bez narušení prostřednictvím šifrované sítě[7].

[1] https://www.iso.org/ics/35.100/x/

1.1.1 HTTP

HTTP[55] (Hypertext Transfer Protocol) je protokol pro přenos dat přes World Wide Web. Jedná se o protokol typu požadavek-odpověď, což znamená, že klient (obvykle webový prohlížeč) odešle serveru požadavek a server odpoví požadovanými daty. HTTP je bezstavový protokol, což znamená, že každý požadavek je zpracováván samostatně, bez znalosti předchozích požadavků nebo odpovědí.

HTTP funguje na aplikační vrstvě modelu OSI a je základem webu. Používá se k přenosu dat mezi klienty a servery, včetně dokumentů HTML, obrázků, videí a dalších typů dat. Jedná se o textový protokol, což znamená, že ke komunikaci mezi klienty a servery používá prostý text. To usnadňuje ladění a odstraňování problémů i vývoj nových aplikací a služeb využívajících protokol HTTP. K zadávání požadavků používá řadu metod, například GET, POST, PUT, DELETE a další. Metoda GET slouží k vyžádání zdroje ze serveru, zatímco metoda POST odesílá data na server ke zpracování, PUT slouží k aktualizaci zdroje na serveru a metoda DELETE k odstranění zdroje ze serveru[55].

HTTP podporuje také řadu hlaviček, které lze použít k poskytnutí dalších informaci o požadavku nebo odpovědi. Například hlavička *User-Agent* může být použita k identifikaci klienta, který požadavek odesílá, zatímco hlavička *Content-Type* může být použita k určení formátu dat odesílaných v požadavku nebo odpovědi[55].

HTTP[27] se v průběhu času vyvíjel a přicházely nové verze a rozšíření, které zlepšují jeho výkon a zabezpečení. Aktuální minimální verzí je HTTP/1.1, který oproti předchozím verzím obsahuje například podporu trvalých připojení, pipeliningu a kódování přenosu po částech. Trvalá připojení umožňují odesílat více požadavků prostřednictvím jediného připojení, čímž se snižuje režie spojená s navazováním a rušením připojení pro každý požadavek. Pipelining umožňuje odeslat více požadavků dříve, než je přijata odpověď na první požadavek, což dále snižuje režii navazování a ukončování spojení. Kódování přenosu po částech ruší nutnost posílat data jako jeden velký blok, což může zvýšit výkon a snížit využití paměti.

HTTP/2[27] je nová verze protokolu, která má zlepšit výkon HTTP. Používá binární formát namísto textového, který se používá v předchozích verzích, což umožňuje efektivnější kódování a dekódování dat. HTTP/2 také podporuje vícenásobné propojení, které umožňuje posílat více požadavků a odpovědí přes jedno spojení, což nadále snižuje režii navazování a ukončování spojení.

Dalším významným rozšířením je HTTPS[9] (HTTP Secure), které přidává další vrstvu zabezpečení komunikace mezi klienty a servery. K šifrování přenášených dat používá protokol SSL/TLS, který útočníkům ztěžuje zachycení a čtení dat. HTTPS se běžně používá v internetovém bankovnictví, elektronickém obchodě a dalších aplikacích, kde je bezpečnost dat kriticky důležitá. HTTPS používá k ověření identity serveru systém ověřování založený na certifikátech. Klient kontroluje certifikát předložený serverem, aby se ujistil, že je platný a že jej vydala důvěryhodná certifikační autorita. Pokud je certifikát platný, může klient navázat zabezpečené spojení se serverem a bezpečně odesílat a přijímat data.

1.1.2 DNS

Systém doménových jmen (DNS) je základní součástí internetu, která poskytuje hierarchický a distribuovaný systém pojmenování počítačů, služeb a dalších zdrojů na internetu a jiných sítích s internetovým protokolem (IP). Slouží jako *telefonní seznam internetu* a přiřazuje různé informace k doménovým jménům (identifikačním řetězcům) přiřazeným jednotlivým entitám. Systém DNS převádí doménová jména, která jsou jednodušeji čitelná

pro lidi, na číselné IP adresy potřebné pro vyhledávání a identifikaci počítačových zdrojů. Každé zařízení připojené k internetu má jedinečnou IP adresu, kterou používají ostatní zařízení k jeho vyhledání. Servery DNS hrají klíčovou roli při překladu doménových jmen na IP adresy, což uživatelům umožňuje zadávat do prohlížeče běžná slova, aniž by si museli pamatovat IP adresu každé webové stránky. [19]

Servery DNS ukládají záznamy DNS pro domény a odpovídají na dotazy klientů tím, že poskytují potřebné informace k vyhledání konkrétních webových stránek. Databáze DNS obsahuje různé typy záznamů, včetně záznamů o začátku autority (SOA), IP adres (A a AAAA), výměníků pošty SMTP (MX), jmenných serverů (NS), ukazatelů pro zpětné vyhledávání DNS (PTR) a aliasů doménových jmen (CNAME). Servery DNS mohou také uchovávat záznamy pro další typy dat, jako jsou záznamy DNSSEC a seznamy černých děr v reálném čase (RBL) používané v boji proti spamu. [40]

Proces překladu DNS zahrnuje několik krok, které začínají dotazem na místní překladač DNS, který obvykle poskytuje poskytovatel internetových služeb (ISP) uživatele. Místní resolver zkontroluje svoji mezipaměť, zda již nemá IP adresu pro požadovaný název domény. Pokud ne, předá dotaz kořenovému serveru DNS, který je zodpovědný za správu domén nejvyšší úrovně. Kořenový server odpoví odkazem na server TLD, který je odpovědný za správu TLD požadovaného doménového jména. Server TLD odpoví odkazem na server DNS odpovědný za dané doménové jméno, který je známý jako autoritativní server DNS. Autoritativní server DNS odpoví IP adresou požadovaného doménového jména, která je poté uložena do mezipaměti místního DNS resolveru pro budoucí použití. [15]

Systém DNS je navržen tak, aby byl maximálně dostupný a odolný vůči poruchám, kde za správu různých částí hierarchie DNS odpovídá více serverů DNS. To zajišťuje, že systém DNS může fungovat i v případě, že některý ze serverů DNS není dostupný nebo má problémy. Systém DNS také podporuje ukládání do mezipaměti, což umožňuje resolverům DNS ukládat výsledky dotazů DNS po určitou dobu, čímž se snižuje počet dotazů, které je třeba odeslat serverům DNS, a tím se zlepšuje výkon celého systému.

1.1.3 NTP

NTP (Network Time Protocol)[54][51] je široce používaný protokol pro synchronizaci hodin počítačových systémů v síti. Byl vyvinut v 80. letech 20. století a od té doby byl implementován na široké škále platforem, od osobních počítačů až po vestavěné systémy. NTP je určen k synchronizaci hodin počítačových systémů se světovým koordinovaným časem (UTC), který je založen na mezinárodním atomovém čase (TAI). UTS se šíří různými prostředky do referenčních hodin, které přesně udávají aktuální čas.

NTP používá UDP k odesílání a přijímaní časových značek po síti[54]. Jedná se o jeden z nejstarších aktuálně používaných internetových protokolů. Tento protokol je důležitý pro synchronizaci hodin na přibližně 25 milionech serverů, pracovních stanic a osobních počítačů ve veřejných internetových a soukromých sítích. NTP funguje na modelu klient-server, kde klienti posílají požadavky časovému serveru, který odpovídá aktuálním časem. Klient pak upraví své vlastní hodiny na základě rozdílu mezi svým časem a časem přijatým ze serveru. Tento proces se může pravidelně opakovat, aby byla zachována synchronizace.

NTP je důležitý pro správu sítě, jelikož zabezpečení, plánování a ladění síti zahrnuje určení, kdy k událostem dochází. Přesný čas na síťových zařízeních je vyžadován pro ověřování digitálních certifikátů, vytváření záznamů s přesnými časovými značkami a omezení provozu na základě času. Přesné časové značky jsou klíčem k analýze příčin, určení doby vzniku problémů a nalezení korelací. Pokud se síťová zařízení rozcházejí v synchronizaci

o několik milisekund (v extrémech i o několik sekund), může být velmi obtížné určit sled událostí. Dále je NTP důležitý pro sledování narušení bezpečnosti, využití sítě nebo problémů, které se týkají velkého počtu komponent. Je důležité, aby časy modifikace souborů byly konzistentní bez ohledu na to, na jakém počítači se souborové systémy nacházejí. Na NTP spoléhají také fakturační služby další aplikace, které musí znát přesný čas. Například některé legislativní a finanční služby vyžadují ze zákona velmi přesné měření času[54].

NTP je hiearchický systém s různými úrovněmi vrstev. úroveň vrstvy udává vzdálenost od referenčních hodin, přičemž nižší úrovně jsou blíže referenčním hodinám, kde referenční hodiny jsou obvykle cesiový oscilátor nebo přijímač GPS, který poskytuje přesný zdroj času. K odesílání a přijímání časových značek pomocí protokolu UDP používá NTP systémy typu peer-to-peer a klient-server[54].

1.1.4 SIP

SIP[46] (Session Initiation Protocol) je signalizační protokol používaný k navazování, udržování a ukončování komunikačních relací v reálném čase, například hlasových a video hovorů, v sítích IP. SIP je textový protokol, který využívá architekturu klient-server, kdy uživatelský agent SIP (UA, user agent) posílá požadavek serveru SIP, který mu odpovídá. Zprávy s požadavky a odpověďmi jsou formátovány pomocí syntaxe podobné protokolu HTTP. SIP je protokol typu peer-to-peer, což znamená, že umožňuje přímou komunikaci mezi dvěma uživatelskými agenty bez potřeby centralizovaného serveru. Díky tomu je SIP škálovatelný a spolehlivý protokol, který lze použít v široké škále síťových topologií.

Protokol SIP se používá pro celou řadu aplikací, včetně přenosu hlasu přes IP (VoIP), zasílání rychlých zpráv a multimediálních konferencí. SIP podporuje řadu funkcí, včetně přidržení hovoru, přepojení a přesměrování hovoru. Protokol SIP také podporuje multimediální relace a umožňuje přenos zvuku, videa a dalších typů dat. SIP je spolehlivá a uživatelsky přívětivá metoda pro navazování a udržování komunikačních relací v reálném čase[59].

1.1.5 SMTP

SMTP (Simple Mail Transfer Protocol) je komunikační protokol používaný pro odesílání zpráv elektronické pošty mezi servery a klienty. Je součástí aplikační vrstvy sady protokolů TCP/IP a je zodpovědný za přenos e-mailových zpráv přes internet. SMTP má dlouhou historii, která začíná na počátku 80. let 20. století. Byl navržen s cílem zjednodušit a zlepšit výměnu textových zpráv mezi různými servery SMTP. V průběhu let se protokol SMTP vyvíjel a přizpůsoboval tak, aby zvládal složitější struktury e-mailů, přílohy a obsahy typu MIME (Multipurpose Internet Mail Extension).

Servery SMTP jsou aplikace, které poskytují služby jiným aplikacím v síti, tzv. klientům. Jsou zodpovědné za odesílání e-mailů pomocí protokolu SMTP. Existují dva typy serverů SMTP: *běžné a vyhrazené*. Běžné servery se používají pro odesílání standardních osobních e-mailů, zatímco vyhrazené servery se používají k zpracování hromadných a transakčních e-mailů.

Protokol SMTP určuje postup pro výměnu dat mezi klientem a serverem. Když uživatel odešle e-mail, e-mailový klient otevře spojení SMTP s e-mailovým serverem pomocí přenosového protokolu TCP (Transmission Control Protocol). Klient poté odešle několik příkazů s obsahem e-mailu, včetně hlavičky a těla e-mailu. Na serveru je spuštěn program MTA (Mail Transfer Agent), který kontroluje název domény příjemce a dotazuje se systému DNS, aby zjistil IP adresu příjemce. Jakmile je přenos dat dokončen, klient to oznámí serveru a ten spojení uzavře.

Příkazy SMTP[39] jsou textové instrukce, které klientovi nebo serveru říkají, jak má s daty pracovat a co s nimi má dělat. Mezi nejběžnější příkazy SMTP patří HELO (zahájení relace), EHLO (zahájení relace s podporou rozšířených funkcí), MAIL FROM (určení e-mailové adresy odesílatele), RCPT TO (zadání e-mailové adresy příjemce), DATA (označení začátku dat zprávy) a QUIT (ukončení relace).

Pro ověření a identifikaci odesílatele a správné šifrování a dešifrování e-mailů se používá SMTP authentication[2], čímž se chrání soukromí dat během přenosu. Dále to pomáhá přecházet spamovým a phishingovým útokům tím, že ztěžuje zachycení a přečtení e-mailů špatnými subjekty. Mezi běžné metody ověřování patří PLAIN, kde se uživatelské jméno a heslo odesílá v prostém textu, LOGIN, které vyžaduje odeslání údajů v kódování Base64, nebo CRAM-MD5 s využitím mechanismu výzvy a odpovědi. V základu není SMTP zabezpečený protokol, protože data jsou přenášena v prostém textu. Pro zvýšení zabezpečení se proto SMTP používá ve spojení s TLS nebo SSL.

1.1.6 RTSP

RTSP[35] (Real-Time Streaming Protocol) je síťový protokol aplikační úrovně navržený k multiplexování a paketování multimediálních přenosových toků, jako jsou interaktivní média, video a zvuk, prostřednictvím vhodného přenosového protokolu. Protokol RTSP se používá v zábavním a komunikačních systémech k řízení serverů pro streamování médií a umožňuje navazování a řízení relací médií mezi koncovými body. Klienti vydávají příkazy, jako je přehrávání, nahrávání nebo pozastavení, pro řízení přehrávání médií v reálném čase ze serveru ke klientovi (video) nebo naopak (nahrávání hlasu).

Protokol RTSP je v některých ohledech podobný protokolu HTTP[25], ale liší se tím, že má stav, používá identifikátor pro sledování souběžných relací a používá protokol TCP pro udržování spojení mezi koncovými body. Výchozí port pro RTSP je 554, který podporuje TCP i UDP, ale UDP se pro řídící požadavky zpravidla nepoužívá. Protokol RTSP sám o sobě nezpracovává přenos dat, ale běžně se používá s protokolem RTP (Real-time Transport Protocol) a protokolem RTCP (Real-Time Control Protocol) pro doručování mediálních dat.

1.1.7 QUIC

QUIC[23] (Quick UDP INternet COnnections) je síťový protokol transportní vrstvy navržený společností Google s cílem snížit latenci ve srovnání s protokolem TCP. Jedná se o multiplexovaný transport postavený na UDP. QUIC a jeho vlastnosti používá například HTTP/3, včetně absence blokování Head-Of-Line mezi jednotlivými toky. QUIC je navržen nad datagramy UDP, což má usnadnit nasazení a zabránit problémům pocházejícím ze síťových zařízení, která zahazují pakety z neznámých protokolů. To také umožňuje, aby implementace QUIC fungovaly v uživatelském prostoru, takže například prohlížeče mohou implementovat QUIC, aniž by vyžadovaly úpravy jádra.

QUIC[23] poskytuje také řadu vylepšení s cílem zrychlit provoz HTTP a zvýšit jeho bezpečnost. Tato vylepšení zahrnují vestavěné zabezpečení (a výkon), přičemž počáteční handshake QUIC kombinuje typický třícestný handshake, který se používá u TCP, s handshake TLS 1.3, který zajišťuje ověřování koncových bodů a také vyjednávání kryptografických parametrů. Tím je zajištěno, že spojení je vždy ověřeno a zašifrováno, a počáteční navazování spojení je díky tomu rychlejší. QUIC obsahuje mechanismy pro řízení velikosti datagramů nesoucích pakety QUIC a také pravidla pro řízení přenosu, opakovaného přenosu a potvrzování dat.

1.2 TLS

Protokol TLS[49] (Transport Layer Security) je kryptografický protokol používaný pro bezpečnou komunikaci přes internet. Je široce používán pro bezpečné prohlížení webových stránek, odesílání e-mailů nebo souborů a další formy komunikace. Protokol TLS poskytuje bezepčnou vrstvu nad transportními protokoly TCP/IP a používá symetrické šifrování i šifrování s využitím veřejného klíče pro bezpečné odesílání soukromých dat.

TLS využívá kombinaci symetrické a asymetrické kryptografie k přenosu dat[49]. Při symetrické kryptografii se data šifrují a dešifrují pomocí tajného klíče, který je znám odesílateli i příjemci. Symetrická kryptografie je efektivní z hlediska výpočtů, ale společný tajný klíč je nutné bezpečně vyměnit, což může být bezpečnostní riziko. Asymetrická kryptografie používá dvojice klíčů - veřejný a soukromý klíč. Veřejný klíč je matematicky příbuzný soukromému klíči, ale při dostatečné délce klíče je výpočetně nepraktické odvodit soukromý klíč z veřejného klíče. Díky tomu může odesílatel použít veřejný klíč příjemce k zašifrování dat, která mu chce poslat a příjemce poté data dešifruje pomocí svého soukromého klíče.

Protokol TLS je postaven nad protokolem TCP/IP a klient musí nejprve dokončit 3-cestný handshake[1] TCP se serverem. Klient musí serveru oznámit, že si přeje připojení TLS namísto standardního nezabezpečeného připojení, proto odešle zprávu popisující, kterou verzi protokolu TLS a jaké šifrovací techniky chce použít. Pokud server požadované technologie nepodporuje, spojení přeruší. Jinak odpoví potvrzením a digitálním certifikátem, který obsahuje jeho veřejný klíč. Pokud klient nevěří, že certifikát je pravý, přeruší spojení, aby neodesílal soukromá data podvodníkovi.

Handshake[53][1] u protokolu TLS je proces navázání zabezpečeného spojení mezi klientem a serverem. Handshake začíná, když klient odešle serveru zprávu *ClientHello*, která obsahuje nejvyšší verzi TLS, kterou klient podporuje, seznam podporovaných sad šifer a náhodné číslo. Server odpoví zprávou *ServerHello*, která obsahuje zvolenou verzi TLS, zvolenou sadu šifer a další náhodné číslo. Server také zašle svůj digitalní certifikát, který obsahuje jeho veřejný klíč. Klient pak digitální certifikát ověří, aby se ujistil, že byl vydán důvěryhodnou certifikační autoritou a že odpovídá názvu domény serveru. Klient poté vygeneruje tajný klíč (pre-master secret) a zašifruje jej veřejným klíčem serveru a odešle jej serveru, který ho dešifruje svým soukromým klíčem. Klient i server poté použijí tento tajný klíč k vygenerování hlavního tajného klíče, který se používá k šifrování a dešifrování veškeré následující komunikace.

Součástí hanshake protokolu TLS je také kód MAC (Message Authentication Code)[36], který zajišťuje, že zprávy vyměňované během handshake nebudou zfalšovány. MAC je generován pomocí hašovací funkce a hlavního klíče. Handshake je ukončen, když klient odešle zprávu *Finished*, která obsahuje MAC všech zpráv vyměněných během handshake. Server pak odešle zprávu *Finished*, která také obsahuje všechny MAC všech zpráv.

TLS se vyvinul z protokolu SSL (Secure Socket Layers), který původně vyvinule aspolečnost Netscape Communications Corporation v roce 1994. Protokol TLS byl poprvé specifikován v roce 1999 jako protokol nezávislý na aplikacích, a přestože nebyl přímo schopen spolupracovat s protokolem SSL 3.0, v případě potřeby nabízel nouzový režim. Protokol SSL 3.0 je však nyní považován za nezabezpečený a v červnu 2015 byl vyřazen z používání s doporučením, aby se používal protokol TLS 1.2. V současné době se připravuje verze TLS 1.3, která upustí od podpory méně bezpečných algoritmů[49].

Protokol TLS nezabezpečuje data v koncových systémech. Zajišťuje pouze bezpečné doručování dat přes internet, čímž zabraňuje možnému odposlechu a změně obsahu. Protokol TLS je obvykle implementován nad protokolem TCP za účelem šifrování protokolů

aplikační vrstvy jako jsou HTTP, FTP, SMTP a IMAP, ale může být implementován také v protokolech UDP, DCCP a SCTP (například pro použití v aplikacích VPN a SIP)[49].

1.3 Monitorování síťového provozu

Síťový provoz[4] je tok dat v síti, který zahrnuje všechny informace vyměňované mezi zařízeními připojenými k síti. Zahrnuje různé typy dat, jako jsou e-maily, soubory, webové stránky a jiné multimediální obsah, které jsou přenášeny mezi zařízeními, jako jsou počítače, servery, směrovače a přepínače. Naprostá většina síťového provozu je dnes šifrovaná (zhruba 80% všech přenášených dat[24]). Šifrovaný síťový provoz je postup zabezpečení dat, který převádí běžné čitelné informace na nesrozumitelnou šifru a umožňuje bezpečné přenášení dat bez toho, aby byly vystaveny zlým subjektům. Standardní technologie SSL (Secure Sockets Layer) je výchozí formou ochrany síťových dat pro internetovou komunikaci.[37]

Analyzování síťového provozu[48] je klíčové pro identifikaci komunikace malwaru a útočníků, kteří skrývají své aktivity v bezpečném šifrovaném provozu. Analýzu šifrovaného provozu lze rozdělit na tři úrovně/kategorie: jednoduchá analýza provozu, rozšířená analýza certifikátů a pokročilá kryptoanalýza. Jednoduchá analýza provozu zahrnuje sledování síťových transakcí pomocí IP adres, portů, protokolů a času přenosů. Rozšířená analýza certifikátů se zabývá podrobnostmi použitého šifrování, například sadami šifer a jejich rozšířeními. Pokročilá kryptoanalýza zkoumá charakteristiky a příznaky síťového provozu, například vzory v posloupnostech délek a časů paketů v jednotlivých tocích i mimo ně.

Síťový provoz lze analyzovat na dvou úrovních, na úrovni paketů nebo na úrovni síťových toků. Při analýze síťového provozu na úrovni paketů se zkoumají jednotlivé pakety, které procházejí sítí. Hloubková kontrola paketů (DPI) je však pro šifrovaný síťový provoz nepoužitelná, jelikož bez šifrovacího klíče nelze přečíst zašifrovaný obsah paketů. Místo toho lze použít nástroje, které hledají a detekují anomálie v síti, jako jsou například neobvyklá spojení mezi počítači, neobvyklé použití portů TCP/UDP nebo prostý textový provoz na portu 443, který je vyhrazen pro protokol TLS (Transport Layer Security).

Jelikož není možné na šifrovaný síťový provoz aplikovat hloubkovou analýzu paketů, tak se pro shromažďování a analýzu dat používají protokoly Netflow a IP Flow Information Export (IPFIX), které analyzují síťový provoz pomocí toků. Tyto protokoly lze použít k analýze pomocí metadat spojených s každým tokem, jako jsou zdrojové a cílové IP adresy, porty, protokol nebo časové značky.

Další metodou analýzy šifrovaného síťového provozu jsou proxy servery SSL/TLS. Tyto servery fungují jako prostředníci mezi klientem a serverem a při průchodu dešifrují a znovu šifrují provoz. Tím může proxy server kontrolovat provoz na přítomnost malwaru, zakázaných stránek a dalších hrozeb. Proxy servery však zvyšují složitost sítě a mohou zpomalovat provoz. Další jejich nevýhodou je, že mohou přinést zranitelnosti a ohrozit bezpečnost šifrovaného provozu.

1.4 Netflow

Netflow[10] je síťový protokol vyvinutý společností Cisco pro monitorování a sběr dat o síťovém provozu na Cisco zařízeních. Tyto data obsahují informace o množství síťového provozu, odkud tento provoz přichází a jaká je jeho cílová IP adresa. Existuje několik verzí Netflow, mezi nejpopulárnější patří verze 5, kde se používá předem daná kolekce polí[26].

NetFlow[43][11][26] je klíčový síťový protokol pro sběr a monitorování síťového provozu vyvinutý společností Cisco. Umožňuje exportovat statistiky o provozu na síti jako záznamy NetFlow, které lze poté zachytit a dále zpracovávat a analyzovat pomocí kolektoru a analyzátoru. Netflow poskytuje podrobné informace o zdroji odkud síťový provoz pochází, jaká je jeho cílová IP adresa, jaký protokol byl použit a informace o využití šířky pásma. Tím umožňuje správcům sítí monitorovat síťové konverzace a rychle identifikovat problémy s výkonem a potenciální bezpečnostní hrozby. Tento protokol plní klíčovou roli při analýze síťového provozu, jelikož nabízí komplexní pohled na síťové toky a možnost sledování i velkých objemů dat na síti a díky tomu je široce používaný už od svého uvedení v roce 1995.

1.4.1 Struktura Netflow

Exportér

Netflow exportér je zařízení na síti, které sbírá data o tocích a odesílá je na kolektor pro další zpracování. Exportér identifikuje jeden tok jako soubor paketů, které mají společné vlastnosti jako zdrojové a cílové IP adresy, zdrojové a cílové porty, protokol a typ služby. Toky jsou exportovány pokud jsou neaktivní po určitý předem daný interval, což znamená, že nepřišel žádný další paket, který by patřil do daného toku po určitý interval, nebo pokud specifické TCP příznaky značí konec toku[43].

Proces práce exportéru se skládá z několika hlavních kroků.

- Sběr dat o tocích

  Exportér sbírá data o tocích ze síťových zařízení jako jsou směrovače a nebo firewally.

- Export dat

  Odesílání všech ukončených a neaktivních toků, které ještě nebyly odeslány na vybraný kolektor pomocí UDP (User Datagram Protocol).

- Expirace toků

  Exportér pravidelně kontroluje všechny aktivní toky, zda některý z nich není expirovaný. Každý tok může expirovat v jednom ze dvou případů[43]:

  - Neaktivní přerušení
    K exportování toku dojde, pokud nebyly během určité doby zaznamenány pakety, které patří do daného toku, poté je tedy předpokládáno, že tok skončil a je ukončen. Typická doba po které dojde k exportování neaktivního toku je 15 sekund.
  - Aktivní přerušení
    Pokud některý tok trvá déle než určitý čas je exportován. Nejčastěji je v základu nastaveno 30 minut, ale to je obvykle příliš dlouhá doba, a proto se doporučuje používat mnohem kratší interval (např. 5 minut nebo i jenom jedna minuta) a díky tomu je možné lépe sledovat, co se průběžně děje na síti. Toto pravidlo je do exportéru zavedeno například kvůli nekonečným tokům.

- Konfigurace

  Pro správné fungování je nutné správné nastavení exportéru. Musí být upřesněny jisté detaily jako cílová IP adresa a port kolektoru, ID zdroje, verze Netflow a další parametry k ujištění, že dojde ke správné výměně dat.

Kolektor

Netflow kolektor je síťová aplikace, která přijímá, zpracovává a ukládá Netflow záznamy exportované ze směrovačů, rozdělovačů a dalších zařízeních na síti neboli z exportérů. Kolektor přijímá pakety s exportovanými toky, ze kterých následně extrahuje data o jednotlivých tocích. Dále tyto data připravuje k dalšímu zpracování a ukládá je. Hlavní funkce Netflow kolektoru jsou:

- Příjem UDP datagramů

  Přijímá paketů se záznamy o tocích z jednoho nebo více zařízení, které je použito jako exportér

- Převod binárních dat

  Překládá binární data na text a čísla pro jednoduché čtení dat pro uživatele.

- Zmenšování velikosti dat

  Zmenšuje velikost dat pomocí zadaných filtrů a seskupování pro optimalizaci analýzy a pro lepší využití dostupného úložiště.

- Ukládání výsledných dat

  Ukládá přepracovaná data o tocích v prostých souborech nebo v SQL databázích pro další analýzu.

- Odesílání záznamů

  Přeposílá zachycená data o tocích aplikacím nebo jiným Netflow analyzátorům na oddělených výpočetních strojích pro hlubší analýzu.

Analyzátor

Netflow analyzátor je nástroj, který je navržený pro monitorování a hloubkovou analýzu síťových toků. Slouží k odhalování problémů na části sítě, ze které máme naše toky. Díky tomu mohou správci sítí provádět vylepšení sítí, plánovat kapacitu na jistých úsecích nebo detekovat různé hrozby a zvyšovat bezpečnost sítě[33].

1.4.2 Netflow v5

Netflow v5[13] (také nazývaný „tradiční Netflow“) je nejpopulárnější verze. Formát paketu je zafixovaný a díky tomu je jednodušší tyto pakety sbírat a dekódovat. V této verzi není zavedený koncept o příchozích a odchozích paketech. Proto zde není žádné pole, které by určovalo směr. Každý paket Netflow má přesně danou strukturu hlavičky i dat, které přenáší.

1.4.3 Netflow v9

Netflow v9 představila flexibilní Netflow, kde nejsou pevně daná pole v každém Netflow paketu. Místo toho jsou definovaná pro jednotlivé toky mezipaměti, což umožňuje přizpůsobitelnější export dat (můžeme si nastavit, jaké informace o zachyceném síťovém provozu chceme z exportéru dostávat).

1.5 IPFIX

Internet Protocol Flow Information Export (IPFIX) je protokol založený na Netflow verzi 9. Byl vyvynutý skupinou Internet Engineering Task Force (IETF) v roce 2013. Byl navržen tak, aby byl univerzálnější než NetFlow, který funguje pouze se zařízeními Cisco. IPFIX dokáže shromažďovat datové pakety z široké škály produktů a zařízení různých výrobců, díky čemu je populární u organizací, které potřebují efektivní řešení pro analýzu provozu ze zařízení jiných výrobců než Cisco. Uživatelé IPFIX protokolu mohou přizpůsobit své požadavky tak, aby systém plnil určité úlohy, například organizoval informace nebo prováděl základní analýzu dat. IPFIX je také schopen integrovat do procesu exportu více informací, což umožňuje přizpůsobit sběr a analýzu dat. Oproti Netflow je jeho nejvýznamnější rozdíl flexibilita. Uživatelé si mohou přizpůsobit šablony zpráv IPFIX. Tyto šablony popisují data, které chtějí uživatelé shromažďovat. Netflow je také schopen shromažďovat informace nad rámec svého standardu, ale jde o mnohem složitější a časově náročnější proces. U IPFIX mohou uživatelé využívat také pole s proměnnou délkou, což umožňuje shromažďovat data, jako jsou adresy URL a zprávy.

1.5.1 Ipfixprobe

Ipfixprobe[14] je open-source flow exporter, který vytváří biflow ze vstupních paketů a exportuje je pomocí protokolu IPFIX. Jedná se o projekt vytvořený sdružením vysokých škol CESNET a Akademií věd ČR, které provozuje a rozvíjí výzkumnou a vzdělávací síť. Tento nástroj je navržen jako účinné a efektivní řešení pro měření a monitorování sítě, zejména pro vysokorychlostní spoje a vestavěná zařízení s omezenými zdroji. Toho dosahuje využitím subsystému Netfilter conntrack na úrovni jádra pro zachycování a agregaci síťových toků, místo aby se spoléhal na knihovny pro zachycování paketů v uživatelském prostoru, jako je libpcap, které mohou přinášet režii a ztráty paketů.

Nástroj také podporuje různé zásuvné moduly, které obohacují exportovaná data o tocích, například zásuvný modul TLS fingerprinting, který k tokům přidává informace o JA3. Poskytuje také praktický skript, který zjednodušuje proces vytváření nových zásuvných modulů.

Kapitola 2

Malware

Malware neboli škodlivý software je takový software, který má za úkol poškodit server, počítač nebo nějaký jiný cíl. Snaží se poškodit systém na napadeném stroji i celou síť, na kterou je stroj napojený. Kybernetické útoky se nejčastěji zaměřují na cloudové systémy internetu věcí. K těmto útokům jsou použity různé typy malware. Jedny z nějčastějších typů jsou například spyware, ransomware, trojské koně, boti, scareware, rootkity, červi a viry. Útočníci se snaží najít a zneužít slabiny nebo chyby systémů. [44] Pokud se jim to podaří, může dojít ke krádeži dat, osobních údajů, k úniku jiných citlivých informací nebo k poškození nebo i zhroucení celého systému.

2.1 Typy malware

1. Virus

   Viry jsou programy navrženy tak, aby se replikovaly a šířily. Po infikování způsobují různé škodlivé následky jako je poškození dat, zhroucení celého systému nebo krádež informací. Viry fungují tak, že se připojí k hostitelským souborům nebo programům, díky kterým se rozšiřují. Tyto souboru mohou být různých typů jako například spustitelné programy, dokumenty nebo mediální soubory. Po aktivaci mohou viry spouštět škodlivý kód, replikovat se a šířit do dalších systémů například pomocí příloh e-mailů, infikovaných webových stránek nebo vyměnitelných úložných zařízení.

   ### Druhy virů

   - Souborové infikárory
     Tyto viry se připojí ke spustitelným souborům a po jejich spuštění se šíří.
   - Viry zaváděcího sektoru
     Infikují zaváděcí sektor úložných zařízení a ohrožují schopnost infikovaného systému se spustit.
   - Makroviry
     Běžně se vyskytují v dokumentech s vloženými makry a využívají zranitelností v kancelářském softwaru.
   - Polymorfní viry
     Tyto viry mění svůj kód, aby se vyhnuly detekci antivirovými programy, což je dělá velmi náročné na boj proti nim.

- Vícedílné viry
  Tyto viry kombinují vlastnosti souborových infikátorů a virů zaváděcího sektoru. Infikují více oblastí systému, čímž způsobují rozsáhlé škody.

Dopad virů

Následky infekce virem mohou být velmi závažné, od ztráty dat, nestability systému až po finanční ztráty. Viry mohou ukrást citlivé informace, vyřadit systémy z provozu, což může vést k výpadkům a nákladné obnově nebo mohou sloužit jako nosiče dalšího škodlivého softwaru.

2. Červ

   Červi jsou škodlivé programy, které se autonomně šíří v počítačových sítích a infikují systému bez toho, aby bylo potřeba lidského zásahu. Na rozdíl od virů, které k replikaci potřebují hostitelský soubor, se červi mohou šířit samostatně.

   Hlavní vlastnosti červů:

   - Samostatná replikace
     Červi se dokáží sami kopírovat a šířit se do dalších počítačů prostřednictvím síťových připojení, příloh e-mailů nebo jiných online komunikačních kanálů.
   - Autonomní šíření
     Při infikování systémů se červi nespoléhají na další soubory, což jim umožňuje rychlé šíření a působení rozsáhlých škod.
   - Spotřeba zdrojů
     Po infikování stroje mohou spotřebovávat systémové zdroje, jako například paměť nebo využívat síťové připojení, což vede ke zpomalení systému, pádům a nestabilní síti.
   - Přenášení souborů
     Někteří červi mohou přenášet další programy a soubory, které po infikování kradou data, mažou soubory, šifrují data nebo vytvářejí tzv. zadní vrátka pro další neoprávněný přístup škodlivých subjektů.

   Druhy červů:

   - E-mailoví červi
     Šíří se pomocí příloh e-mailů a infikují systémy, když uživatelé otevřou škodlivé soubory.
   - IM-červi, IRC-červi
     Využívají komunikační sítě (Instant Messenger a Internet Relay Chat) k přenosu svých kopií a infikování dalších hostitelských počítačů.
   - Net-Worms
     Šíří se prostřednictvím síťových sdílení a zneužívají zranitelností připojených systémů bez nutnosti hostitelských souborů.

3. Trojský kůň

Trojský kůň je forma malwaru, která se maskuje jako legitimní software, aby pronikl do počítačových systémů. Nejčastěji se jedná o soubor, který lze zadarmo stáhnout nebo může být schovaný například v příloze e-mailu. Po stažení a spuštění se provede škodlivý kód a podle toho, k čemu útočník Trojského koně vytvořil může například vytvořit *zadní vrátka*, sbírat informace o uživateli nebo ukrást citlivá data. Známky napadení počítače Trojským koněm může být neobvyklá aktivita, jako je změna některých nastavení apod. Velmi často spoléhají na taktiky sociálního inženýrství, aby nalákaly uživatele ke stažení a spuštění. [29]

Nejčastější typy Trojských koní:

- Trojský kůň na zadní vrátka
  Vytváří tajný vstupní bod do infikovaného systému, který poté mohou útočníci zneužít k neoprávněnému přístupu nebo k získání kontroly nad systémem.
- Stahovací trojský kůň
  Do již infikovaných systémů instaluje další malware, čímž rozšiřuje rozsah útoku.
- Trojský kůň na výkupné
  Po infikování systému kontaktuje vlastníka a požaduje výkupné za zrušení škod, které byly způsobeny na systému, jako například snížení výkonu nebo zablokování přístupu k datům.
- Troský kůň pro vzdálený přístup
  Umožňuje útočníkům vzdáleně ovládat systém. Díky tomu mohou ukrást data nebo sledovat aktivitu uživatele.
- Trojský kůň Banker
  Zaměřuje se na bankovní účty. Snaží se ukrást citlivé bankovní a platební informace.

4. Ransomware

   Ransomware je forma malwaru, která zašifrovává data, soubory, celá zařízení nebo systémy obětí a dělá je nedostupnými, dokud není útočníkovi zaplaceno výkupné. Zahrnuje taktiku kybernetického vydírání, kdy útočník vyhrožuje smazáním zašifrovaných souborů nebo systému. Ransomware se nejčastěji šíří prostřednictvím phishingových e-mailů, zranitelností RDP (Remote Desktop Protocol) a jiných softwarových chyb. Jedním z nejnovějších trendů je zaměřování útoků na velké organizace a na vzestupu je i Ransomware-as-a-Service (RaaS). Nejlepší ochranou proti tomuto druhu malwaru je pravidelné a bezpečné zálohování. Pokud dojde k infikaci je důležitá rychlá izolace infikovaných systémů, prošetření narušení a obnova pomocí záloh. [45]

5. Spyware

   Jedná se o malware, který je navržen tak, aby pronikl do počítačového zařízení a tam shromažďoval údaje o uživateli, které následně předává třetí straně bez souhlasu uživatele. Sleduje aktivity uživatele, krade citlivé informace, jako jsou hesla nebo bankovní údaje, nebo může zasahovat do ovládání počítače instalací dalšího softwaru nebo přesměrováním webových prohlížečů. Šíří se například pomocí přibalených softwarových balíčků, napadených webových stránek nebo škodlivých e-mailových příloh. Může být použit také na sledování uživatelské činnosti na firemních, sdílených nebo veřejných počítačích, kdy je nainstalován záměrně. [31]

   Spyware lze rozdělit několika hlavních kategorií[47]:

- Adware

  Zobrazuje v systému nežádoucí reklamy.
- Systémový monitor

  Velmi dotěrný typ, který může sledovat téměř vše, co uživatel dělá a dokonce i provádět změny v systému.
- Keylogger

  Zaznamenává všechny stisky kláves na infikovaném zařízení a tyto informace ukládá do zašifrovaného souboru.
- Sledovací spyware

  Sleduje online aktivity uživatele jako je vyhledávání nebo stahování a tyto informace sdílí s třetími stranami.
- Trojský kůň

  Maskuje se jako legitimní software, aby přiměl uživatele k jeho instalaci a poté umožňuje útočníkům přístup k datům napadeného počítače.

6. Rootkit

   Rootkity jsou programy určené k zajištění neoprávněného přístupu a kontroly nad počítačovým systémem. Svoji přítomnost se snaží co nejlépe skrýt před uživateli a bezpečnostními mechanismy, a díky tomu mohou být obtížné na odhalení a odstranění. Jejich cílem může být například poskytnout vzdálený přístup, ukrást citlivá data, deaktivovat bezpečnostní protokoly nebo vytvořit zadní vrátka pro další malware.[38]

   Typy rootkitů:

   - Rootkit zavaděče

     Infikuje zavaděč systému, což umožňuje načtení škodlivého softwaru ještě před operačním systémem, čímž se ztěžuje jeho detekce.
   - Firmwarový rootkit

     Nabootuje do počítače a spustí se z něj systém. Ukrývá se ve firmwaru[1] a jelikož bezpečnostní nástroje obvykle firmware na přítomnost malwaru neskenují je otížné ho odhalit.
   - Rootkit jádra

     Útočí na základ operačního systému a poskytuje útočníkům hlubokou kontrolu na systémem.
   - Rootkit paměti

     Vyskytuje se v RAM paměti počítače, kde provádí škodlivé činnosti a u toho zpomaluje systém.
   - Aplikační rootkit

     Modifikuje běžné soubory kódem rootkitu, čímž poskytuje neoprávněný přístup útočníkům, kdykoli jsou infikované soubory spuštěny.

7. Botnet

   Botnety jsou sítě napadených počítačů nebo zařízení infikovaných botovým malwarem, které mohou být ovládány hackerem. Tato propojená zařízení, známá jako boti, jsou

[1] https://cs.wikipedia.org/wiki/Firmware

používána k provádění různých kybernetických aktivit, jako je spouštění automatických skriptů, provádění útoků, jako je distribuované odepření služby (DDoS), a šíření malwaru. Dále mohou být botnety použity například k rozesílání hromadných spamových e-mailových kampaní, generování falešného internetového provozu, phishingových schémat nebo k útokům hrubou silou s cílem získat přístup k webovým stránkám nebo systémům. Nové botnety vznikají infikováním počítačů nebo jiných zařízení malwarem, čímž se z nich stanou boti, které pak mohou být vzdáleně a hromadné ovládány.[32][42]

2.2 Ochrana proti útokům

2.2.1 Prevence

Prevence proti malwarovým útokům je prvním krokem ke kybernetické bezpečnosti a ochraně citlivých dat. Cílem těchto preventivních opatření je minimalizovat riziko samotného napadení malwarem zavedením strategií a postupů, které odrazují, odhalují a blokují škodlivé aktivity. Preventivní strategie by měly být doplněny o pasivní a aktivní bezpečnostní opatření, jako je monitorování, plánování reakce na incidenty a nástroje pro zabezpečení sítě pro vytvoření komplexní obrana proti kybernetickým hrozbám.

Základní preventivní opatření proti malwaru[41][8]:

- Silná hesla

  Zavedení složitých a jedinečných hesel pro všechny účty a systémy. Hesla je nutné pravidelně aktualizovat a vyhnout se jejich sdílení.

- Dvoufaktorové ověřování (2FA)

  Přidání další úrovně zabezpečení pomocí 2FA, kdy je kromě hesla vyžadována druhá forma ověření. 2FA je nejčastěji používáno pro citlivé účty a služby, jako je například bankovnictví.

- Firewall

  Nasazení firewallů, které monitorují a kontrolují příchozí a odchozí síťový provoz. Je nutná konfigurace, aby tyto firewally blokovaly podezřelé nebo známé škodlivé IP adresy a domény.

- Pravidelné aktualizace softwaru

  Aktualizace softwaru a operačního systému pomocí nejnovějších bezpečnostních záplat. Nejjednodušší proces jak toho docílit je povolení automatické aktualizace.

- Školení o povědomí bezpečnosti

  Vzdělání uživatelů v oblasti rozpoznávání pokusů o phishing, podezřelých odkazů a dalších typů malwaru.

- Filtrování e-mailů

  Implementování řešení pro filtrování e-mailů, aby došlo k zablokování spamu a pokusů o phishing. Mezi osvědčené praktiky také patří školení pro uživatele, aby rozpoznávali a nahlašovali podezřelé e-maily.

- Filtrování webu

  Implementace nástrojů pro filtrování webu, aby byl zablokovaný přístup na známé škodlivé webové stránky a domény. Opět je důležité školit uživatele, aby nenavštěvovali podezřelé a nedůvěryhodné webové stránky.

- Ochrana koncových bodů

  Instalace, pravidelná údržba a aktualizace antimalwarového softwaru na všech zařízeních. Další důležitou činností je nastavení tohoto softwaru tak, aby automaticky a pravidelně vyhledával malware.

- Segmentace sítě

  Rozdělení sítě na menší segmenty, aby se v případě napadení malwarem omezilo jeho šíření. Mezi těmito vytvořenými segmenty je nutné implementovat přísné řízení přístupu.

- Zálohování a obnova dat

  Vytváření pravidelných záloh důležitých dat a systémů. Tyto zálohy je důležité testovat na obnovitelnost a zajistit jejich bezpečné uložení.

2.2.2 Pasivní zabezpečení

Pasivní zabezpečení se zaměřuje na monitorování a analýzu síťových aktivit bez aktivní interakce se sítí. Tento přístup má za cíl odhalit bezpečnostní hrozby a předcházet jim. Hlavním cílem je odhalení anomálií, neoprávněných přístupů a jiné škodlivé aktivity v síti bez narušení běžného provozu. Tímto je možné odhalovat bezpečnostní hrozby a předcházet jim. Pasivní zabezpečení se spoléhá na nástroje jako jsou například systémy detekce narušení (IDS[2]), řešení správy bezpečnostních informací a událostí (SIEM[3]) a analýza síťového provozu. které umožňují identifikovat podezřelé vzorce a chování. Díky průběžnému monitorovaní je možné odhalit případné hrozby velmi brzy a zastavit je ještě předtím, než dojde ke škodě nebo alespoň případné škody omezit na minimum[17].

Pasivní bezpečnostní opatření:

- Monitorování síťového provozu

  Pasivní zabezpečení zahrnuje sledování provozu na síti za účelem odhalení neobvyklé činnosti, pokusů o neoprávněný přístup nebo potenciálních narušení bezpečnosti. K zachycení a analýze datových paketů a k následnému hledání známek škodlivé činnosti se používají nástroje jako paketové odposlechy a síťové analyzátory.

- Analýza záznamů

  Analýza systémových záznamů, záznamů událostí a záznamů o činnosti uživatelů může pomoci identifikovat bezpečnostní incidenty nebo neobvyklé chování.

- Prověřování zranitelností

  Provádění skenování zranitelností za účelem identifikace slabých míst v systémech, aplikacích a dalších síťových zařízeních. Nástroje pro skenování zranitelností pomáhají pří posuzování stavu zabezpečení a určování priorit při nápravě.

[2] https://www.geeksforgeeks.org/intrusion-detection-system-ids/
[3] https://www.microsoft.com/cs-cz/security/business/security-101/what-is-siem

- Monitorování zpravodajství o hrozbách

  Mezi část pasivního zabezpečení patří využívání informačních kanálů o hrozbách, abychom byli informováni o vznikajících hrozbách, zranitelnostech a trendech útoků. Díky tomu je možné se proaktivně bránit proti novým hrozbám.

- Šifrování a ochrana dat

  Zavedením šifrovacích protokolů a jiných opatření na ochranu dat se výrazně snižuje riziko neoprávněného přístupu a úniku citlivých dat. Šifrování zabezpečuje data v klidu i při přenosu.

- Plánování reakce na incidenty

  Vypracování a procvičování plánů reakce na incidenty je klíčové pro rychlou a účinnou reakci na zjištěné bezpečnostní problémy a na různé útoky.

2.2.3 Aktivní zabezpečení

Aktivní bezpečnostní systém zahrnují monitorování v reálném čase a okamžitou reakci na bezpečnostní situace. V případě útoku dochází k proaktivním krokům k ochraně systémů a dat. Tyto kroky jsou navrženy tak, aby předcházely, detekovaly a reagovaly na útoky.

Honeypot

Používání honeypotů je jedním z mechanismů počítačového zabezpečení. Slouží hlavně k odhalení, ale i k odvrácení nebo potlačení pokusů o neoprávněný přístup k počítači nebo síťovému systému. Jsou navrženy tak, aby se útočníkům jevily jako legitimní systém, počítač nebo server, ale ve skutečnosti jsou izolovány a monitorovány za účelem odhalení a následné analýzy případných škodlivých aktivit. [18]

Honeypoty lze rozdělit do tří kategorií podle komplexnosti: čisté honeypoty, honeypoty s vysokou mírou interakce a honeypoty s nízkou mírou interakce. **Čisté honeypoty** jsou plnohodnotné produkční systémy, které slouží k monitorování aktivit útočníků pomocí odposlechu, který je instalovaný na spojení honeypotu se sítí. **Vysoce interaktivní honeypoty** napodobují činnost produkčních systémů, které hostují různé služby, a tím odlákává pozornost útočníků od skutečných systémů, a tím je okrádá o čas. **Nízkointerakční honeypoty** simulují pouze služby, které útočníci často požadují, spotřebovávají relativně málo prostředků a mohou odhalit útoky bez nutnosti je sledovat. Další možností jak lze rozdělit honeypoty je na fyzické a virtuální. **Fyzické honeypoty** jsou skutečné stroje s vlastní IP adresou a **virtuální honeypoty** jsou nainstalovány a simulovány v síti z různých operačních systémů.

Dále lze honeypoty rozdělit na dvě kategorie dle toho, jaké mají cíle: výzkumné a produkční honeypoty.[20] **Výzkumné honeypoty** shromažďují informace o útocích. Používají se zejména ke studiu škodlivých činností v otevřených sítích a shromažďují informace o útocích nebo například zranitelnostech, na které se útočníci zaměřují. Tyto informace mohou být použity jako podklad pro preventivní obranu, stanovení priorit oprav a budoucí investice. **Produkční honeypoty** se oproti tomu zaměřují na identifikaci aktivních kompromitací v interní síti a na oklamání útočníka. Stále je prioritou sběr informací, jelikož díky honeypotům máme další možnosti monitorování sítě. Produkční honeypoty jsou umístěny mezi ostatními produkčními servery a jsou na nich spuštěny stejné služby, které běží na skutečných serverech. Výzkumné honeypoty bývají složitější a ukládají více typů dat než produkční honeypoty.

Honeypoty přinášejí organizacím mnoho výhod, včetně zpomalení útočníků, testování procesů reakce na incidenty a shromažďování informací o nástrojích, postupech a taktice útočníků. Vyžadují však pečlivé nasazení a údržbu, aby byla zajištěna jejich účinnost a aby se předešlo potenciálním rizikům, jako je přilákání zbytečné pozornosti útočníků nebo falešné pozitivní výsledky.

2.3 Použití strojového učení v oblasti malwaru

Strojové učení se stále častěji používá při detekci a analýze malwaru. Algoritmy strojového učení mohou pomoci identifikovat škodlivé vzorce v síťovém provozu a odhalit potenciální hrozby analýzou dat.[3] Pro detekci malwaru se používají různé algoritmy a modely strojového učení, jako například Decision Tree, Random Forest, k-Nearest Neighbor, Decision Table, Extreme Gradient Booting, Multilayer Preceptron nebo neuronové sítě.

Detekce malwaru pomocí strojového učení lze využít různými způsoby, například k detekci malwaru před otevřením podezřelých souborů, k identifikaci typu malwaru a ke sledování chování škodlivých nebo podezřelých programů. Antivirový software na bázi strojového učení využívá ke sledování chování programů detekci anomálií a nevyžaduje k tomu aktualizace signatur virů.

Dále lze strojové učení použít k detekci škodlivých aktivit ve velkých souborech dat, které jsou například v oblastech zabezpečení sítí nebo monitorování elektronické pošty. Software pro vyhodnocování a monitorování zranitelností založený na strojovém učení může zvýšit rychlost odhalování kybernetických útoků a v průběhu času rozvíjet přesnost detekce.

Programy se strojovým učením také mohou pomoci při detekci botů a rozpoznávání mezi dobrými boty pomocí analyzování souborů dat se vzory chování. Algoritmy strojového učení lze také použít k rozpoznání útoků spáchaných jinými podobnými algoritmy nebo pomocí umělé inteligence, jelikož i hackeři tyto postupy používají ke spáchání zločinů.

Před nasazením do provozu je nutné natrénovat modely strojového učení. Nejčastější typy trénování jsou[28][30]:

- Učení pod dohledem

  Tato technika zahrnuje trénování modelů pomocí označených datových sad (např. u datových sad obsahujících zachycený síťový provoz je označeno, zda se jedná o malware), které předpovídají výstupy na základě tréninkových dat. Zahrnuje klasifikační a regresní metody, kdy se stroj učí na základě dvojic vstup-výstup a poté tyto znalosti využívá k předpovídání výsledků pro nové datové body.

- Učení bez dohledu

  Při učení bez dohledu se modely trénují na datech, která nebyla klasifikována nebo označena. Model tedy jedná samostatně a objevuje vzory a struktury v datech bez explicitního vedení. Mezi běžné přístupy učení bez dohledy patří shlukování, redukce dimenzionality a algoritmy asociačních pravidel.

- Učení s částečným dohledem

  Učení s částečným dohledem je mezistupeň, který pracuje s kombinací označených a neoznačených dat. Využívá označená data, aby pomohl algoritmům identifikovat vzory v neoznačených datech. Tato technika je užitečná zejména v případech, kdy je nedostatek označených dat, ale mnoho neoznačených dat.

- Učení posilováním

  Další technikou pro trénování modelů strojového učení je posilovací učení. Při tom se model učí metodou pokusů a omylů interakcí s prostředím a získává zpětnou vazbu v podobě odměn nebo trestů. Podobá se učení pod dohledem, ale místo poskytování explicitních tréninkových dat se tento model učí prostřednictvím zkoumání a využívání akcí s cílem maximalizovat odměny.

- Hloubkové učení

  Hloubkové učení zahrnuje trénování modelů na velkých reprezentativních souborech dat s cílem naučit se komplexní vzory a vlastnosti. Techniky hloubkového učení, jako jsou neuronové sítě, se používají k extrakci abstrakcí na vysoké úrovni z dat, což modelu umožňuje provádět přesnější předpovědi a klasifikace. Hloubkové učení je obzvláště účinné při zpracování složitých a vícerozměrných dat, což z něj činí mocný nástroj při detekci malwaru.

2.3.1 Typy modelů strojového učení

Rozhodovací stromy

Rozhodovací stromy[50] jsou základním algoritmem strojového učení pod dohledem, který se často používá pro klasifikační i regresní úlohy. Jsou strukturovány jako vývojové diagramy, začínají v kořenovém uzlu konkrétní otázkou týkající se dat, větví se na potenciální odpovědi a dále vedou k dalším rozhodovacím (vnitřním) uzlům, které dále zpřesňují výsledky, až se dostanou do koncových neboli listových uzlů, kde se učiní konečné rozhodnutí nebo předpověď. Proces tvorby rozhodovacího stromu zahrnuje rekurzivní rozdělení dat na základě hodnot atributů. Tento algoritmus pomáhá vizualizovat složitá rozhodnutí v hierarchické struktuře, což usnadňuje interpretaci a analýzu výsledků na základě vstupních znaků.

Rozhodovací stromy jsou snadno pochopitelné a interpretovatelné, takže jsou přístupné i neodborníkům a jsou vhodné pro klasifikační i regresní úlohy. Mezi další výhody patří efektivní práce s chybějícími nebo odlehlými hodnotami a numerickými i kategoriálními daty. Dále nabízejí jednoduché vizuální znázornění rozhodovacích procesů a pomáhají při rozhodovací analýze ve strojovém učení a poskytují přehled o důležitosti jednotlivých charakteristik při rozhodování.

Součásti a terminologie[21]:

- **Kořenový uzel:** Výchozí bod rozhodovacího stromu, který představuje původní volbu nebo funkci.
- **Vnitřní/rozhodovací uzly:** Uzly stromu, v nichž se na základě hodnot atributů provádí volby vedoucí k dalším uzlům.
- **Listové/rozhodovací uzly:** Koncové body větví, kde jsou rozhodnutí nebo předpovědi dokončeny.
- **Větve (hrany):** Vazby mezi uzly znázorňující cesty rozhodování.
- **Dělení (Splitting):** Rozdělení uzlu na dílčí uzly na základě rozhodovacího kritéria.
- **Nadřazený uzel:** Uzel, který se dělí na podřízené uzly.

- **Dceřinný uzel**: Uzel vytvořený v důsledku rozdělení.
- **Rozhodovací kritérium:** Pravidlo nebo podmínka použitá k rozdělení dat ve vnitřním uzlu.
- **Ořezávání:** Odstranění větví nebo uzlů za účelem zlepšení generalizace a zabránění nadměrnému přizpůsobení[4].

Typy rozhodovacích stromů[50]:

- **Klasifikační stromy:** Určují, zda událost nastala, nebo ne, což často vede k binárnímu výsledku. Rozdělují data na základě proměnných a klasifikují výsledky.
- **Regresní stromy:** Předpovídají spojité hodnoty na základě minulých dat, předpovídají například ceny nebo množství. Zaměřují se spíše na předpovídání číselných hodnot než na klasifikaci dat.

Náhodný les (Random forest

Náhodné lesy[57] jsou metodou skupinového učení, která kombinuje více rozhodovacích stromů s cílem zlepšit přesnost a stabilitu modelů strojového učení. Jedná se o výkonný a všestranný algoritmus, který lze použít pro klasifikační i regresní úlohy.

Klíčovou myšlenkou náhodných lesů je vytvoření souboru rozhodovacích stromů, z nichž každý je trénován na jiné podmnožině trénovacích dat a náhodné podmnožině funkcí. To přináší rozmanitost a snižuje riziko nadměrného přizpůsobení, ke kterému může dojít u jednotlivých rozhodovacích stromů.

Proces fungování algoritmu náhodného lesa[22]:

- **Sáčkování:** Algoritmus začíná vytvořením několika zaváděcích vzorků trénovacích dat. Každý zaváděcí vzorek je náhodnou podmnožinou původních dat.
- **Trénování rozhodovacího stromu:** Pro každý zaváděcí vzorek je natrénován rozhodovací strom. Stromy jsou tvořeny bez ořezávání, což je činí velmi složitými.
- **Vzorkování vlastností:** Při rozdělování uzlu v rozhodovacím stromu bere algoritmus v úvahu pouze náhodnou podmnožinu dostupných vlastností/rysů. Tím se dále zvyšuje rozmanitost mezi stromy.
- **Agregace:** Pro vytvoření předpovědi pro nový údaj provede každý rozhodovací strom v lese předpověď. Konečný výstup je určen buď tím, co bylo předpovězeno většinově (pro klasifikaci), nebo zprůměrováním předpovědí (pro regresi).

Náhodnost v krocích *sáčkování* a *vzorkování vlastností* pomáhá v dekoraci jednotlivých stromů, což vede k vytvoření souboru, který je přesnější a robustnější než kterékoli jednotlivé rozhodovací stromy. Dále jsou náhodné lesy, díky své skupinové povaze, méně náchylné k nadměrnému přizpůsobení trénovacích dat. Mezi další výhody patří například efektivní zpracování různorodých dat nebo paralelizace trénování jednotlivých stromů.

[4] Zahrnutí více parametrů než lze odůvodnit daty, což může snížit přesnost.

KNN

Algoritmus KNN (K-Nearest Neighbors) je metoda strojového učení pod dohledem, která je použitelná pro klasifikační i regresní úlohy. Jedná se o jednoduchý, ale výkonný algoritmus, který nachází široké uplatnění v oblasti rozpoznávání vzorů, těžby dat a detekce narušení.

Základní koncepty algoritmu KNN:

- **Základní princip:** KNN je neparametrický algoritmus, což znamená, že nečiní žádné základní předpoklady o rozdělení dat. Místo toho klasifikuje datové body na základě jejich podobnosti s jinými datovými body v trénovací množině.
- **Princip fungování:** Při dané sadě trénovacích dat a novém datovém bodu přiřadí KNN nový bod do skupiny na základě analýzy trénovací sady. Učiní tak, že na základě metriky vzdálenosti zohlední K nejbližších sousedů nového bodu.
- **Intuice:** Podle toho, do jaké skupiny patří nejbližší sousedé, klasifikuje KNN nový bod. Například bod, který je blízko shluku bodů klasifikovaných jako *malware* bude pravděpodobně klasifikován jako *malware*.

KNN je univerzální algoritmus, který dokáže zpracovat numerická i kategorická data, čímž je vhodný pro různé typy datových souborů v klasifikačních i regresních úlohách. KNN je také neparametrická metoda, což znamená, že provádí předpovědi na základě podobnosti dat, aniž by předpokládala rozdělení dat. Ve srovnání s jinými algoritmy je tato metoda méně citlivá na odlehlé hodnoty, takže je vhodná pro datové sady se zašuměnými nebo nepravidelnými datovými body. K identifikaci nejbližších sousedů datového bodu a k předpovědím na základě jejich hodnot používá KNN metriky vzdálenosti, například euklidovskou vzdálenost.

Při praktické implementaci KNN je klíčový výběr správné hodnoty K, která určuje počet sousedů, kteří jsou při předpovědích bráni v úvahu. Podobně jako u jiných algoritmů strojového učení je pro optimální výkon nezbytné předzpracovat data pomocí normalizace a škálování. K vyhodnocení výkonnosti modelů KNN na testovacích souborech dat se používají metriky, jako jsou matice záměny a skóre přesnosti.

Naivní Bayes

Naivní Bayes je pravděpodobnostní algoritmus strojového učení založený na Bayesově teorému, který se díky své jednoduchosti, účinnosti a efektivitě používá v různých klasifikačních úlohách. Algoritmus je známý svou schopností zpracovávat data s vysokou dimenzí, čímž je obzvláště užitečný při klasifikaci textů, filtrování spamu a dalších. Dále je využitelný například v lékařství při predikci nemoci na základě příznaků a údajů o pacientovi nebo při předpovídání počasí. Vychází z Bayesovy věty, která počítá pravděpodobnost hypotézy vzhledem k důkazům. „Naivní“ část této metody odkazuje na předpoklad, že prvky jsou podmíněně nezávislé vzhledem k označení třídy. Tento předpoklad zjednodušuje a napomáhá efektivitě a rychlosti algoritmu.

Tato metoda je jednoduchá na implementaci a je výpočetně efektivní, takže je vhodná pro velké soubory dat, ale funguje dobře i s omezeným počtem trénovacích dat. Algoritmus je rychlý a rychle provádí předpovědi, zejména v souborech dat s vysokou dimenzí. Metoda Naivní Bayes má ale i několik nevýhod. Předpoklad nezávislosti rysů nemusí platit ve všech reálných souborech dat, což může ovlivnit přesnost modelu. Dále může být ovlivněn irelevantními atributy a může nepozorovaným událostem přiřadit nulovou pravděpodobnost, což vede ke špatné generalizaci a potenciálním problémům s nepozorovanými daty.

2.3.2 Datové sady

Datové sady [34][56] v souvislosti se strojovém učení jsou soubory dat, které se vkládají do algoritmů pro trénování modelů. Tyto soubory dat obsahují označené příklady, které pomáhají algoritmům pochopit vzory a vztahy v datech. Slouží k tomu, aby se model naučil předpovídat na základě vstupních údajů. Proces trénování modelu strojového učení zahrnuje jeho vystavení datovým sadám, čímž se mu umožňuje učit se ze vzorů přítomných v datech a podle toho upravovat svoje vnitřní parametry. Kvalitu a přesnost výsledných modelů strojového učení nejvíce ovlivňuje kvalita, množství a relevance trénovacích dat. Proto je nutné zajistit, aby datové sady byly čisté, strukturované a aby pokrývaly oblast problému, který má výsledný algoritmus řešit.

Datové sady jsou naprosto klíčové pro trénování modelů strojového učení, protože slouží jako základ, podle kterého se modely učí předpovídat a rozhodovat. Pro vývoj přesných a spolehlivých modelů strojového učení je nezbytné mít kvalitní datové sady, které poskytují potřebné informace a vzory, které umožňují algoritmům učit se a zobecňovat data, která jsou jim předložena. I nejsofistikovanější algoritmy mohou být neúčinné pokud byly natrénovány bez kvalitních trénovacích dat.

Kapitola 3

Implementace

3.1 Průzkum datových sad

3.1.1 CTU-SME-11

Datová sada CTU-SME-11[60] [6] je označená datová sada vytvořená studentem ČVÚT v Praze Štěpánem Bendlem v roce 2023, která obsahuje zachycený reálný síťový provoz s nezávadnými i škodlivými toky, navržená tak, aby napodobovala malou až středně velkou síť. Zachycuje sedm dní síťového provozu (od 20. do 26. února 2023) z jedenácti zařízení s různými operačními systémy a hardwarovými konfiguracemi připojených ve vnitřní síti. Tento soubor dat zachycuje lidmi generovaný neškodný provoz, ale i různý malware. Útoky jsou vedeny na vnitřní a vnější síť nebo se jedná o pokusy o exfiltraci dat. CTU-SME-11 je profesionálně označkovaná datová sada a díky tomu ji lze použít k efektivnímu trénování a vyhodnocování výkonnosti modelů strojového učení. Skládá se z necelých 100 000 000 exprety označených síťových toků (přibližně 160 GB souborů pcap).

3.1.2 VHS-22

VHS-22[52] je velmi různorodá datová sada o síťovém provozu určená k především k detekci hrozeb. Obsahuje toky extrahované z pěti různých zdrojů síťového provozu pomocí softwarové síťové sondy. Tato sada byla vytvořena extrakcí dat na úrovni toků a obsahuje různé typy útoků, jako jsou botnety, webové útoky a pokusy o útoky hrubou silou, kde každý z útoků pochází z různých zdrojů a v různá časová období.

Tato datová sada je označena štítky tříd označujícími normální toky a toky útoků spolu se specifickými štítky útoků a štítky zdrojových datových souborů. Každý tok má 45 parametrů uložených ve formátu CSV, které lze využít pro trénování modelů strojového učení.

3.1.3 UWF-ZeekData22

UWF-ZeekData22[16][5] je komplexní datová sada o síťovém provozu založená na frameworku MITRE ATT&CK[1], který byl představen v roce 2023. Tato datová sada je veřejně dostupná a skládá se z datových souborů Zeek označených pomocí frameworku MITRE ATT&CK. Obsahuje soubory ve formátech Pcap, parquet a CSV, kde soubory CSV jsou podmnožinou souborů parquet. Byl vytvořen ze zachyceného provozu z 10. února 2022 během určitých hodin. Z každé hodiny je vytvořen samostatný soubor CSV, kde poměr

[1] https://attack.mitre.org/

benigních a škodlivých dat je zhruba 80/20. Každý tok má 45 parametrů uložených ve formátu CSV, které mohou být využity k trénování modelů strojového učení.

3.2 Příprava dat

Pro svou rozsáhlost, aktuálnost a možnost extrahování dat z pcap souborů byla pro další analýzu a trénování modelů vybrána datová sada CTU-SME-11. Další její velkou výhodou jsou označení toků, která byla určena experty.

V datové sadě CTU-SME-11 jsou data rozdělena po jednotlivých dnech, kdy byl provoz sledován. Na každý den je vytvořen *pcap* soubor, který obsahuje všechny zachycené pakety. Další důležitý soubor pro naši analýzu je *conn.log.labeled* vytvořený pomocí nástroje Zeek[2], který obsahuje všechny zachycené toky během daného dne. Každý z těchto toků je označen pomocí štítku, který označuje, zda se jedná o benigní nebo škodlivý tok.

Pomocí nástroje Ipfixprobe byly zpracovány soubory *pcap*, ze kterých byly vygenerovány toky (IPFIX), s dalšími informacemi o protokolech vyšších vrstev (např. aplikačních) a statistické informace o paketech v jednotlivých tocích. Ipfixprobe byl použit pomocí tohoto příkazu:

```
ipfixprobe -i "pcap;file=$pcap_file" -p pstats -p http -p rtsp
-p tls -p dns -p sip -p ntp -p smtp -p phists -p quic -p icmp
-o "unirec;i=f:$trap_file
p=(pstats,http,rtsp,tls,dns,sip,ntp,smtp,phists,quic,icmp)"
```

Tímto příkazem se z paketů v souboru $pcap_file vygeneruje soubor $trap_file, ve kterém jsou uloženy vygenerované toky IPFIX. Každý tok obsahuje obvyklé informace (IP adresy zdroje a cíle, protokol transportní vrstvy, port zdroje a cíle, čas), ale díky přidaným argumentům jsou ke každému toku přidány informace o určených protokolech aplikační (HTTP, RTSP, DNS, SIP, NTP, SMTP), transportní (TLS, QUIC) a síťové vrstvy (ICMP). Díky argumentu *pstats* jsou ke každému toku přidány statistické informace o prvních 30 paketech (velikosti, časové značky, směry, TCP příznaky). Argument *phists* zařizuje přidání informací o časech mezi pakety a velikostech paketů v obou směrech (od zdroje do cíle i zpět).

Poté byl použit nástroj *logger*, kterému byl předložen vygenerovaný soubor *trap*, který byl následně tímto nástrojem převeden na soubor *csv*, pomocí tohoto příkazu:

```
logger -i "f:$trap_file" -t > "$csv_file"
```

Tento proces byl proveden pro všechny dny a stroje v datové sadě.

Následovalo převedení zeek záznamů, kdy ze souboru *conn.log.labeled* byly vyextrahovány informace o tocích (*'ts', 'id.orig_h', 'id.resp_h', 'id.orig_p', 'id.resp_p', 'proto', 'label', 'detailed_label'*). Díky těmto informacím je možné přiřadit k tokům v *csv* souborech označení zda se jedná o benigní nebo malware toky.

Popis jednotlivých sloupců:

- ts: časová značka, kdy začal daný tok
- id.orig_h: zdrojová IP adresa

[2] https://zeek.org/

- id.resp_h: cílová IP adresa
- id.orig_p: číslo zdrojového portu
- id.resp_p: číslo cílového portu
- proto: protokol transportní vrstvy (tcp/udp)
- label: označení, zda se jedná o benigní nebo malware tok
- detailed_label: detailnější popis toku

Toky vytvořené pomocí nástroje *ipfixprobe* i toky vyextrahované ze zeek záznamů byly načteny pomocí Python modulu pandas[3] a následně spojeny do společného *csv* souboru. Ke spojení byly využity vlastnosti toků, což znamená, že pro každý tok byl vytvořen hash z IP adres a portů zdroje a cíle a protokolu. Tyto hashe byly pospojovány pomocí pandas funkce *merge*. Před spojením toků bylo nutné ještě zkontrolovat, zda od sebe dané toky nejsou příliš vzdálené v čase, jelikož toky mohou skončit aktivním nebo neaktivním přerušením a hned poté může začít nový tok, který má stejné IP adresy, porty i protokol. Všechny toky, které byly označeny pomocí sloupce *label* jsou následně uloženy do *csv* souboru, který je připraven na další zpracování.

3.3 Průzkum dat

Všechny *csv* soubory byly načteny pomocí python modulu pandas a spojeny do jednoho souboru. Dále byl využit modul matplotlib pro zobrazení grafů s informacemi o obsahu datové sady. Kvůli velikosti sady byly operace s ní prováděny s využitím výpočetních a úložných kapacit MetaCentra[4].

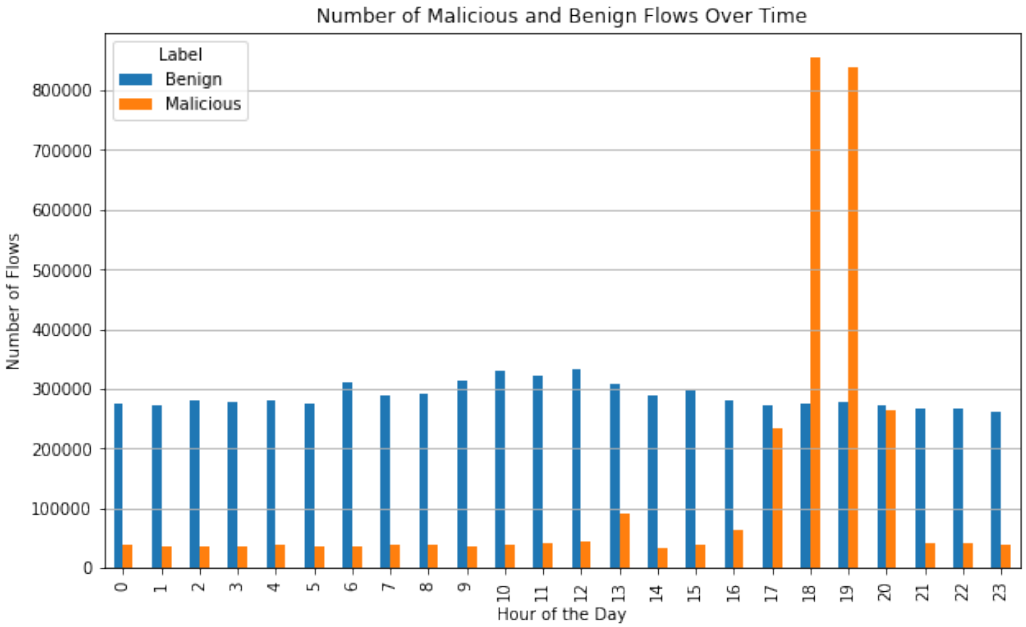
Na obrázku 3.1 můžeme pozorovat počet toků v datové sadě za všechny dny během určitých časů. Můžeme pozorovat, že každou hodinu bylo zachyceno zhruba 300 000 benigních toků a několik desítek tisíc škodlivých (malware) toků. Výjimkou jsou škodlivé toky ve večerních hodinách (17-20), kdy se jejich počet velmi navyšuje a přesahuje i 800 000.

3.3.1 Druhy protokolů v datové sadě

V tabulce 3.1 jsou zobrazeny údaje o zastoupení sledovaných protokolů v datové sadě. V případě transportních protokolů můžeme pozorovat, že asi 63,5 % všech toků využívá protokol TCP. Z těchto toků s protokolem TCP je zhruba 81 % škodlivých. Zbylých 36,5 % toků využívá protokol UDP, ale pouze 18,9 % z nich je škodlivých.

Při pozorování aplikačních protokolů je významný protokol DNS, který je použit v téměř 10,8 % všech toků a 81 % z nich je je škodlivých. Další zajímavý protokol je HTTP, který se používá ve skoro 30 % datové sady, ale jen v jednom procentu případů se jedná o malware. SIP je protokol, který se v datové sadě téměř nenachází (pouze 0,1 %), ale téměř všechny jeho výskyty jsou v malware tocích. Ostatní sledované protokoly jsou v datové sadě zastoupeny jen ve velmi malém množství a jsou škodlivé jen z malé části.

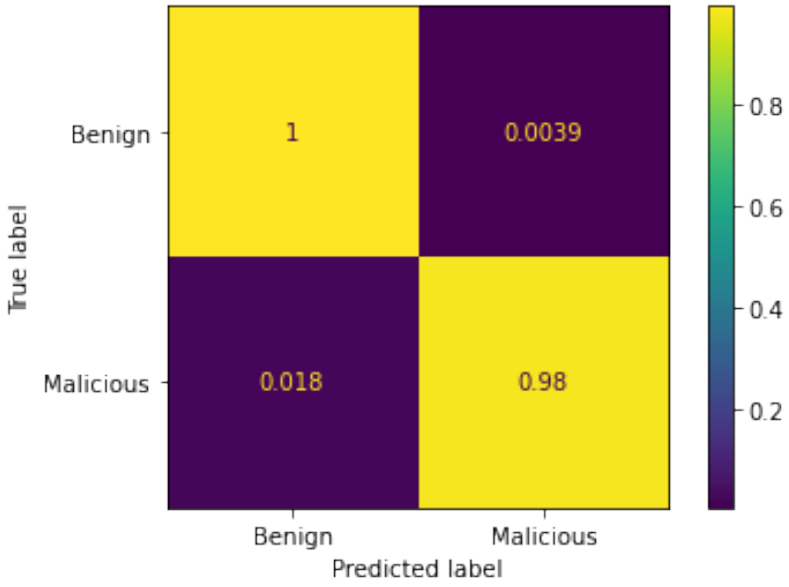
[3] https://pandas.pydata.org/
[4] https://metavo.metacentrum.cz/



Obrázek 3.1: Počet a druh toků za hodinu

Protokol	Zastoupení v sadě (%)	Malware (%)
TCP	63,5	81
UDP	36,5	18,9
DNS	10,8	81,5
NTP	0,8	0,2
HTTP	29,8	1,3
SIP	0,1	99,9
SMTP	0	0
RTSP	0,1	0,0004
QUIC	0,8	0,0001
TLS	1,2	14,7

Tabulka 3.1: Zastoupení protokolů



Obrázek 3.2: Náhodný les - matice zmatení

3.3.2 Statistické informace o tocích

Díky nástroji ipfixprobe jsou o tocích získány statistické informace, jako například velikosti, časy a směry paketů a TCP příznaky. Tyto data jsou ale ve formátu, který nelze použít pro trénování modelu (pole čísel nebo časů). Proto byla použita knihovna *nemea-fet*[5], která je navržená tak, aby převedla statistické informace v polích na číselné informace (integer/float). Tyto převedená data lze poté použít k efektivnějšímu trénování modelu.

Nejprve byla použita funkce *convert_times*, která převádí veškeré časy v načtené datové sadě na *datetime*, včetně doby trvání jednotlivých toků. Následně byla statistická data převedena pomocí funkce *extract_features*. Tato funkce extrahuje z polí se statistickými daty informace, jako například průměrnou, maximální nebo minimální dobu mezi pakety v jednom nebo druhém směru, počty různých TCP příznaků nebo průměrnou dobu trvání paketů v toku. Tyto extrahované informace jsou následně uloženy pro každý tok a mohou být dále použity.

3.4 Trénování a testování modelů

Pro trénování modelu strojového učení byla použita Python knihovna scikit-learn[6]. Nejprve byla načtená datová sada náhodně rozdělena na trénovací a testovací část. K tomu byla využita funkce *train_test_split*. K trénování modelu strojového učení bylo použito 70 % datové sady. Jako klasifikátory byly vybrány vyextrahované statistické informace, které poskytují 50 různých metrik.

Po rozdělení dat, bylo zahájeno trénování vybraného modelu strojového učení:

- **Náhodný les**

[5] https://docs.danieluhricek.cz/fet/
[6] https://scikit-learn.org/stable/

První vybraný model strojového učení byl náhodný les. Tento model byl v porovnání s ostatními rychlostně efektivní, ale také velmi přesný v předpovědích při testování. Celková přesnost tohoto modelu byla **99,5 %**, což je nejvyšší přesnost ze všech testovaných modelů. Na matici zmatení 3.2 lze pozorovat, že model správně předpověděl, že se jedná o benigní tok, v téměř 100 % případů. Správné odhalení škodlivých malware toků mělo o něco menší, ale stále velmi vysokou přesnost (98 %).

- **Rozhodovací strom**

  Dalším testovaným modelem strojového učení byl rozhodovací strom. Ten byl v rozhodování velmi rychlý a i velmi přesný. Jeho celková přesnost byla 99,4 %.

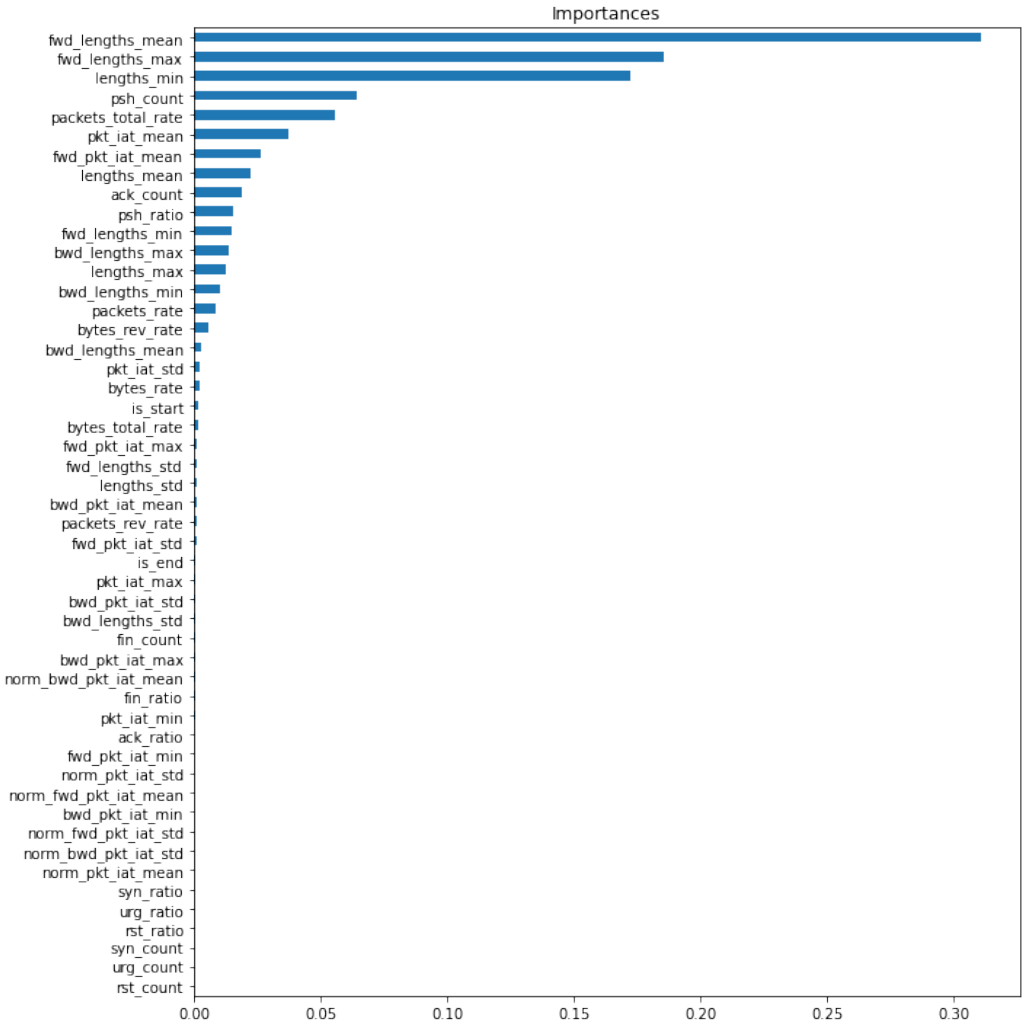
- **Naivní Bayes**

  Jako poslední byl použit model Naivní Bayes, který byl také velmi rychlý, ale oproti ostatním modelům vyl velmi nepřesný v předpovědích. Jeho přesnost byla pouze 17 %. Hlavním problémem tohoto modelu byla správná předpověď benigních toků, kdy model klasifikoval téměř všechny toky jako škodlivé a pouze 0,1 %

3.5 Hodnocení klasifikátorů

Po vyhodnocení modelů strojového učení bylo provedeno hodnocení použitých klasifikátorů. Na obrázku 3.3 jsou zobrazeny použité klasifikátory, které jsou seřazeny podle toho, jak jsou důležité při rozhodování. Lze tedy odvodit, že největší rozhodovací sílu mají délky paketů (minimální, maximální a průměrná délka). Dalšími důležitými klasifikátory byly: četnost příznaků TCP (hlavně ACK a PSH) nebo časy mezi jednotlivými pakety v tocích.

Oproti tomu vlastnosti toků, jako například TCP příznaky URG, RST nebo FIN, jsou stejné pro benigní i škodlivé toky, takže mají nízkou důležitost při rozhodování, o jaký typ toku se jedná.



Obrázek 3.3: Důležitost klasifikátorů

Závěr

Tato bakalářská práce se zabývá problematikou malwaru v sítích a jeho detekcí s využitím statistických informací o tocích. V úvodní části byly popsány aplikační protokoly a následně možnosti sledování síťového provozu na úrovni toků za pomocí protokolu NetFlow a IPFIX a byla popsána činnost protokolu NetFlow. Z důvodu možnosti výběru exportovaných dat byla veškerá další práce prováděna s protokolem IPFIX, pomocí nástroje ipfixprobe.

Následně byly představeny typy malware, které se šíří po síti, jako jsou viry, červi, trojské koně, ransomware a další. Byly také probrány různé praktiky a techniky, které se používají k prevenci a odhalování malwaru, včetně použití strojového učení, které je v dnešní době v oblasti internetové bezpečnosti velmi rozšířené. U strojového učení byly rozebrány různé typy trénování modelů a poté i jednotlivé příklady modelů, kdy byly probrány hlavně modely, které mají využití při detekci malwaru.

Před trénováním samotného modelu strojového učení bylo nutné vybrat kvalitní datovou sadu. Byly prozkoumány volně dostupné sady (CTU-SME-11, VHS-22, UWF-ZeekData22) a následně byla vybrána datová sada CTU-SME-11 vytvořená v Praze v roce 2023, která obsahuje necelých 100 000 000 označených síťových toků. Tato datová sada obsahuje také *pcap* data, která byla převedena pomocí nástroje ipfixprobe do toků IPFIX, kterým poté byly přiřazeny označení, zda se jedná o malware nebo benigní tok.

Připravená datová sada byla poté použita k natrénování několika různých modelů, kdy nejlepší model byl Náhodný les s přesností okolo 99,5 %. K rozhodování zda se jedná o benigní nebo škodlivý tok byly využity různé klasifikátory, přičemž mezi nejvýznamnější patřily informace o četnostech určitých TCP příznacích nebo délky jednotlivých paketů v tocích.

Tyto experimenty s datovou sadou potvrdily, že tato vybraná sada je expertně označená a použitelná pro trénování modelů strojového učení, které poté mohou být použity například ke klasifikování dalších datových sad nebo statické analýze zachycené síťové komunikace na úrovni toků.

Literatura

[1] Academy, K. *Transport Layer Security (TLS)* [online]. 2024 [cit. 2024-4-20]. Dostupné z: https://www.khanacademy.org/computing/computers-and-internet/xcae6f4a7ff015e7d:online-data-security/xcae6f4a7ff015e7d:secure-internet-protocols/a/transport-layer-security-protocol-tls.

[2] AfterLogic. *SMTP authentication in detail* [online]. 2006 [cit. 2024-4-22]. Dostupné z: https://afterlogic.com/mailbee-net/docs/smtp_authentication.html.

[3] Akhtar, M. S. a Feng, T. Malware Analysis and Detection Using Machine Learning Algorithms. *Symmetry*. 1. vyd. 2022, sv. 14, č. 11. DOI: 10.3390/sym14112304. ISSN 2073-8994. Dostupné z: https://www.mdpi.com/2073-8994/14/11/2304.

[4] Ashtari, H. *What Is Network Traffic Analysis? Definition, Importance, Implementation, and Best Practices* [online]. 2022 [cit. 2024-4-4]. Dostupné z: https://www.spiceworks.com/tech/networking/articles/network-traffic-analysis/amp/.

[5] Bagui, S. S., Mink, D., Bagui, S. C., Ghosh, T., Plenkers, R. et al. Introducing UWF-ZeekData22: A Comprehensive Network Traffic Dataset Based on the MITRE ATT&CK Framework. *Data*. 1. vyd. 2023, sv. 8, č. 1. DOI: 10.3390/data8010018. ISSN 2306-5729. Dostupné z: https://www.mdpi.com/2306-5729/8/1/18.

[6] Bendl, S., Valeros, V. a Garcia, S. *CTU-SME-11: a labeled dataset with real benign and malicious network traffic mimicking a small medium-size enterprise environment*. Zenodo, květen 2023. DOI: 10.5281/zenodo.7958259. Dostupné z: https://doi.org/10.5281/zenodo.7958259.

[7] Bitdefender. *Types of Security Protocols (And How They Protect You from Risks)* [online]. 2024 [cit. 2024-5-1]. Dostupné z: https://www.bitdefender.com/cyberpedia/types-of-security-protocols/.

[8] Brooks, R. *How to prevent malware attacks* [online]. 2023 [cit. 2024-3-24]. Dostupné z: https://nordlayer.com/blog/how-to-prevent-malware-attacks.

[9] Bukauskas, D. *What is HTTP, and how does it work?* [online]. 2023 [cit. 2024-4-21]. Dostupné z: https://nordvpn.com/blog/what-is-http/.

[10] Cameron, A. *Netflow* [online]. 2016 [cit. 2024-4-4]. Dostupné z: https://www.liveaction.com/glossary/netflow/.

[11] Cameron, A. *NetFlow* [online]. 2016 [cit. 2024-4-4]. Dostupné z: https://www.liveaction.com/glossary/netflow/.

[12] CATO. *6 Network Security Protocols You Should Know* [online]. 2024 [cit. 2024-5-1]. Dostupné z: https://www.catonetworks.com/network-security/network-security-protocols/.

[13] CCIE. *9.7.1 - Netflow v5 & v9* [online]. 2024 [cit. 2024-4-5]. Dostupné z: http://www.bscottrandall.com/9.7.1.

[14] CESNET. *Ipfixprobe - IPFIX flow exporter* [online]. 2024 [cit. 2024-4-20]. Dostupné z: https://github.com/CESNET/ipfixprobe.

[15] CloudFlare. *What is DNS? | How DNS works* [online]. 2024 [cit. 2024-16-4]. Dostupné z: https://www.cloudflare.com/learning/dns/what-is-dns/.

[16] Dr. Sikha Bagui, D. S. B. *UWF-ZeekData22 Dataset* [online]. 2022 [cit. 2024-4-2]. Dostupné z: https://datasets.uwf.edu/.

[17] ExterMetwprls. *Active Vs Passive Monitoring* [online]. 2024 [cit. 2024-3-24]. Dostupné z: https://www.extnoc.com/learn/networking/active-vs-passive-monitoring.

[18] Fortinet. *What Are Honeypots (Computing)?* [online]. 2024 [cit. 2024-3-25]. Dostupné z: https://www.fortinet.com/resources/cyberglossary/what-is-honeypot.

[19] Fortinet. *What Is DNS (Domain Name System)?* [online]. 2024 [cit. 2024-4-16]. Dostupné z: https://www.fortinet.com/resources/cyberglossary/what-is-dns.

[20] Gartner. *Honeypots* [online]. 2024 [cit. 2024-3-25]. Dostupné z: https://www.rapid7.com/fundamentals/honeypots/.

[21] GeeksforGeeks. *Decision Tree in Machine Learning* [online]. 2024 [cit. 2024-4-18]. Dostupné z: https://www.geeksforgeeks.org/decision-tree-introduction-example/.

[22] GeeksforGeeks. *Random Forest Algorithm in Machine Learning* [online]. 2024 [cit. 2024-4-18]. Dostupné z: https://www.geeksforgeeks.org/random-forest-algorithm-in-machine-learning/.

[23] Ghedini, A. *The Road to QUIC* [online]. 2018 [cit. 2024-4-24]. Dostupné z: https://blog.cloudflare.com/the-road-to-quic.

[24] Hancock, A. *The Last Mile of Encrypting the Web: 2023 Year in Review* [online]. 2023 [cit. 2024-4-4]. Dostupné z: https://www.eff.org/deeplinks/2023/12/year-review-last-mile-encrypting-web.

[25] Henken, P. *RTSP – all you need to know about real-time streaming protocol* [online]. 2021 [cit. 2024-4-23]. Dostupné z: https://corp.kaltura.com/blog/rtsp-streaming/#.

[26] IBM. *What is NetFlow?* [online]. 2016 [cit. 2024-4-4]. Dostupné z: https://www.ibm.com/topics/netflow.

[27] IONOS. *What is HTTP?* [online]. 2020 [cit. 2024-4-21]. Dostupné z: https://www.ionos.com/digitalguide/hosting/technical-matters/what-is-http/.

[28] Jadav, P. *Malware Detection Using Machine Learning Techniques* [online]. 2022 [cit. 2024-3-28]. Dostupné z: https://www.einfochips.com/blog/malware-detection-using-machine-learning-techniques/.

[29] Johansen, A. G. *What is a Trojan?* [online]. 2020 [cit. 2024-3-21]. Dostupné z: https://us.norton.com/blog/malware/what-is-a-trojan.

[30] Kaspersky. *Machine Learning for Malware Detection* [online]. 2021 [cit. 2024-3-28]. Dostupné z: https://media.kaspersky.com/en/enterprise-security/Kaspersky-Lab-Whitepaper-Machine-Learning.pdf.

[31] Kaspersky. *Spyware: What It IS and How to Protect Yourself* [online]. 2024 [cit. 2024-3-22]. Dostupné z: https://usa.kaspersky.com/resource-center/threats/spyware.

[32] Kaspersky. *What is a Botnet?* [online]. 2024 [cit. 2024-3-22]. Dostupné z: https://usa.kaspersky.com/resource-center/threats/botnet-attacks.

[33] Kentik. *NetFlow Collector* [online]. 2023 [cit. 2024-4-5]. Dostupné z: https://www.kentik.com/kentipedia/netflow-collector/.

[34] Khan, R. *Importance of Datasets in Machine Learning and AI Research* [online]. 2022 [cit. 2024-4-2]. Dostupné z: https://www.datatobiz.com/blog/datasets-in-machine-learning/.

[35] Klyushkov, M. *What is RTSP and why is it necessary? RTSP protocol explained* [online]. 2022 [cit. 2024-4-23]. Dostupné z: https://flussonic.com/blog/news/about-rtsp/.

[36] Lake, J. *What is TLS and how does it work?* [online]. 2023 [cit. 2024-4-20]. Dostupné z: https://www.comparitech.com/blog/information-security/tls-encryption/.

[37] LiveAction. *Encrypted Network Traffic* [online]. 2024 [cit. 2024-4-4]. Dostupné z: https://www.liveaction.com/glossary/encrypted-network-traffic/.

[38] Malwarebytes. *Rootkit* [online]. 2024 [cit. 2024-3-22]. Dostupné z: https://www.malwarebytes.com/rootkit.

[39] Markova, V. *SMTP (Simple Mail Transfer Protocol) explained*. 2023 [cit. 2024-4-22]. Dostupné z: https://www.cloudns.net/blog/smtp-simple-mail-transfer-protocol-explained/.

[40] Mockapetris, P. *Domain names - Concepts and Facilities* [online]. 1987 [cit. 2024-4-16]. Dostupné z: https://datatracker.ietf.org/doc/html/rfc1034.

[41] NordLayer. *How to prevent malware attacks* [online]. 2024 [cit. 2024-3-24]. Dostupné z: https://nordlayer.com/blog/how-to-prevent-malware-attacks.

[42] Paloalto. *What is a Botnet?* [online]. 2024 [cit. 2024-3-22]. Dostupné z: https://www.paloaltonetworks.com/cyberpedia/what-is-botnet.

[43] Petryschuk, S. *NetFlow Basics: An Introduction to Monitoring Network Traffic* [online]. 2022 [cit. 2024-4-4]. Dostupné z: https://www.auvik.com/franklyit/blog/netflow-basics/.

[44] Priyadarshan, P., Sarangi, P., Rath, A. a Panda, G. Machine Learning Based Improved Malware Detection Schemes. In: Prizadarshan, P., ed. *2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*. 2021, s. 925–931. DOI: 10.1109/Confluence51648.2021.9377123. ISBN 978-1-6654-1451-7.

[45] Rangel, M. *Ransomware prevention* [online]. 2023 [cit. 2024-3-22]. Dostupné z: https://www.veeam.com/blog/ransomware-prevention-best-practices.html.

[46] Rosenberg, Schulzrinne, Camarillo, Johnston, Peterson et al. *SIP: Session Initiation Protocol* [online]. 2002 [cit. 2024-4-21]. Dostupné z: https://datatracker.ietf.org/doc/html/rfc3261/.

[47] Seguin, P. *What Is Spyware* [online]. 2023 [cit. 2024-3-22]. Dostupné z: https://www.avast.com/c-spyware.

[48] Seltzer, L. *3 ways to monitor encrypted network traffic for malicious activity* [online]. 2019 [cit. 2024-4-4]. Dostupné z: https://www.csoonline.com/article/567027/3-ways-to-monitor-encrypted-network-traffic-for-malicious-activity.html.

[49] Society, I. *TLS Basics* [online]. 2024 [cit. 2024-4-20]. Dostupné z: https://www.internetsociety.org/deploy360/tls/basics/.

[50] Staff, C. *Decision Trees in Machine Learning: Two Types (+ Examples)* [online]. 2023 [cit. 2024-4-18]. Dostupné z: https://www.coursera.org/articles/decision-tree-machine-learning.

[51] Systems, C. *Network Time Protocol* [online]. 2018 [cit. 2024-4-15]. Dostupné z: https://www.cisco.com/c/dam/en/us/td/docs/ios-xml/ios/bsm/configuration/xe-3se/3650/bsm-xe-3se-3650-book.html.

[52] Szumelda, P., Orzechowski, N., Rawski, M. a Janicki, A. VHS-22 – A Very Heterogeneous Set of Network Traffic Data for Threat Detection. In: David Megías, R. D. P., ed. *Proceedings of the 2022 European Interdisciplinary Cybersecurity Conference*. New York, NY, USA: Association for Computing Machinery, 2022, s. 72–78. EICC '22. DOI: 10.1145/3528580.3532843. ISBN 9781450396035. Dostupné z: https://doi.org/10.1145/3528580.3532843.

[53] T. Dierks, C. A. *The TLS Protocol* [online]. 1999 [cit. 2024-4-20]. Dostupné z: https://datatracker.ietf.org/doc/html/rfc2246.

[54] Telesis, A. *Network Time Protocol (NTP)* [online]. Allied Telesis, 2024 [cit. 2024-4-15]. Dostupné z: https://www.alliedtelesis.com/sites/default/files/documents/configuration-guides/ntp_feature_overview_guide.pdf.

[55] tutorialspoint. *HTTP* [online]. 2024 [cit. 2024-4-20]. Dostupné z: https://www.tutorialspoint.com/http/.

[56] Weedmark, D. *Machine Learning Model Training: What It Is and Why It's Important* [online]. 2021 [cit. 2024-4-2]. Dostupné z: https://domino.ai/blog/what-is-machine-learning-model-training#body__0d2488ae9917.

[57] Whitfield, B. *Random Forest: A Complete Guide for Machine Learning* [online]. 2024 [cit. 2024-4-18]. Dostupné z: https://builtin.com/data-science/random-forest-algorithm.

[58] Yadav, H. *Application Layer Protocols in Computer Networks* [online]. 2023 [cit. 2024-5-1]. Dostupné z: https://www.prepbytes.com/blog/computer-network/application-layer-protocols-in-computer-networks/.

[59] Zerby, J. *What Is Session Initiation Protocol (SIP) & How Does It Work?* [online]. 2023 [cit. 2024-4-21]. Dostupné z: https://www.nextiva.com/blog/sip-protocol.html.

[60] Štěpán, B. *A Network Dataset of Normal, Malware, Attack and Background Traffic on a Real Network*. Prague, CZ, 2023. Diplomová práce. Czech Technical University in Prague, Faculty of Electrical Engineering, Department of Computer Science. Dostupné z: https://dspace.cvut.cz/bitstream/handle/10467/109254/F3-DP-2023-Bendl-Stepan-Stepan_Bendl_Master_Thesis.pdf?isAllowed=y&sequence=-1.