UNIVERZITA PAVLA JOZEFA ŠAFÁRIKA V KOŠICIACH

PRÍRODOVEDECKÁ FAKULTA

AUTOMATIZOVANÉ SPRACOVANIE FORENZNÝCH ARTEFAKTOV OPERAČNÉHO SYSTÉMU WINDOWS

2023 **Henrieta PALOČKOVÁ**

UNIVERZITA PAVLA JOZEFA ŠAFÁRIKA V KOŠICIACH

PRÍRODOVEDECKÁ FAKULTA

AUTOMATIZOVANÉ SPRACOVANIE FORENZNÝCH ARTEFAKTOV OPERAČNÉHO SYSTÉMU WINDOWS

BAKALÁRSKA PRÁCA

Študijný program:	Informatika
Pracovisko (katedra/ústav):	Ústav informatiky
Vedúci bakalárskej práce:	doc. RNDr. JUDr. Pavol Sokol, PhD.

Košice 2023 **Henrieta PALOČKOVÁ**

Univerzita P. J. Šafárika v Košiciach
Prírodovedecká fakulta

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta:	Henrieta Paločková
Študijný program:	informatika (jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor:	Informatika
Typ záverečnej práce:	Bakalárska práca
Jazyk záverečnej práce:	slovenský
Sekundárny jazyk:	anglický

Názov: Automatizované spracovanie forenzných artefaktov operačného systému Windows

Názov EN: Automated processing of forensic artifacts from the Windows operating system.

Cieľ: (1) Analyzovať možnosti spracovania forenzných artefaktov operačného systému Windows prostredníctvom dátovej analýzy.
(2) Porovnať existujúce prístupy k automatizovanému spracovaniu forenzných artefaktov operačného systému Windows.
(3) Navrhnúť, implementovať a vyhodnotiť nástroj na automatizované spracovanie forenzných artefaktov z operačného systému Windows.

Literatúra: (1) Javed, A. R., Ahmed, W., Alazab, M., Jalil, Z., Kifayat, K., & Gadekallu, T. R. (2022). A Comprehensive Survey on Computer Forensics: State-of-the-art, Tools, Techniques, Challenges, and Future Directions. IEEE Access.
(2) Hassan, N. A. (2019). Digital Forensics Basics: A Practical Guide Using Windows OS. Apress.
(3) Kävrestad, J. (2020). Fundamentals of Digital Forensics. Springer International Publishing.

Vedúci:	doc. RNDr. JUDr. Pavol Sokol, PhD.
Oponent:	RNDr. Peter Gurský, PhD.
Ústav :	ÚINF - Ústav informatiky
Riaditeľ ústavu:	doc. RNDr. Ondrej Krídlo, PhD.

Dátum schválenia: 15.05.2023

Poďakovanie

Týmto sa chcem poďakovať vedúcemu mojej práce doc. RNDr. JUDr. Pavlovi Sokolovi, PhD. za odborné vedenie, cenné rady, ochotavý prístup a veľkú pomoc počas tvorby práce.

Abstrakt v štátnom jazyku

S narastajúcim trendom kybernetických hrozieb sa riešenie bezpečnostných incidentov stáva nepopierateľnou súčasťou každej organizácie. Tento proces zahŕňa niekoľko krokov počínajúc od zberu a zaisťovania digitálnych stôp, cez ich spracovanie, až po ich celkovú analýzu. Digitálne stopy sa vo svetle digitálnej forenznej analýzy zvyknú označovať aj ako forenzné artefakty. Sú to objekty, ktoré majú nejakú forenznú hodnotu a teda obsahujú dáta alebo dôkazy o tom, že sa niečo stalo a tak tvoria dôležitú súčasť forenznej analýzy. Forenznú analýzu definujeme ako detailný proces vyšetrovania, detekcie a dokumentácie dôvodu, priebehu a následkov bezpečnostného incidentu. Takáto analýza je často zdĺhavá a náročná z hľadiska neprehľadnosti dát. V tejto práci sa venujeme výberu forenzných artefaktov, ktoré sú využiteľné vo forenznom vyšetrovaní a ich spracovaním pomocou nástrojov na parsovanie dát. Nad týmito dátami následne pomocou programovacieho jazyka Python vykonávame základnú analýzu za účelom získania štatistických informácii o zariadení, z ktorého boli dáta vyextrahované a o udalostiach, ktoré sa na ňom udiali. Výstupom tejto práce je nástroj, ktorý po navrhnutí, implementácii a vyhodnotení slúži na automatizované spracovanie forenzných artefaktov z operačného systému Windows a prispieva tak k zníženiu času analytickej činnosti.

Kľúčové slová: digitálna forenzná analýza, digitálna stopa, forenzný artefakt

Abstrakt v cudzom jazyku

With the growing trend of cyber threats, dealing with security incidents is becoming an undeniable part of every organization. This process involves several steps starting from the collection and securing of digital traces, to their processing, to their overall analysis. Digital traces are also commonly referred to as forensic artifacts in the light of digital forensics. They are objects that have some forensic value and thus contain data or evidence that something happened and thus form an important part of forensic analysis. Forensic analysis is defined as the detailed process of investigating, detecting, and documenting the cause, course, and consequences of a security incident. Such analysis is often lengthy and challenging in terms of data opacity. In this work, we discuss the selection of forensic artifacts that are useful in forensic investigations and their processing using data parsing tools. We then perform basic analysis over this data using the Python programming language in order to obtain statistical information about the device from which the data was extracted and the events that occurred on it. The output of this work is a tool that, once designed, implemented and evaluated, is used for the automated processing of forensic artifacts from the Windows operating system, thus contributing to the reduction of the time of the analytical activity.

Keywords: cybersecurity incident, threat intelligence, indicators of compromise

Obsah

Zoznam ilustrácií

Úvod

S rozmachom technológii sa používanie informačných systémov na poskytovanie služieb a uchovávanie informácií v dnešnej dobe stáva rozšíreným vo verejnom aj súkromnom sektore. Jednotlivci vo svojom každodennom živote intenzívne využívajú výpočtové zariadenia. Je zriedkavé vidieť človeka, ktorý nie je závislý od nejakej formy výpočtového zariadenia pri organizovaní svojich digitálnych údajov alebo pri komunikácii s ostatnými. Bezpečnostné hrozby sa časom nepochybne stávajú vážnejšími [1]. Každý, kto používa výpočtové zariadenia, zanecháva za sebou stopy, či už v podobe časových pečiatok alebo záznamov činnosti.

V apríli 2023 dosahoval trhový podiel operačného systému Windows pre stolné počítače 62,65 % [44]. To znamená, že väčšina osobných počítačov na celom svete používa niektorý z operačných systémov Windows. Je zrejmé, že svet bežiaci na počítačoch s týmto operačným systémom pre nás určite znamená to, že väčšina našej digitálnej forenznej práce zahŕňa vyšetrovanie tohto typu operačného systému [1]. Je tak nepopierateľne nevyhnutné pochopiť a analyzovať forenzné artefakty, ktoré sa nachádzajú v operačnom systéme Windows. Cieľom je, aby sa zistilo, ako došlo k útoku, aké škody boli napáchané a v akom rozsahu. Takéto dôkladné vyšetrovanie a hľadanie anomálii v dátach môže byť pre analytika zdĺhavé. Z tohto dôvodu je potrebné nájsť spôsob, ako by sme mohli znížiť čas analytickej činnosti.

Cieľom tejto práce je navrhnúť nástroj na automatizované spracovanie forenzných artefaktov, ktorým by umožnil zrýchliť proces forenznej analýzy prostredníctvom spracovania artefaktov a vygenerovania základných i pokročilejších štatistických údajov a umožnil nám tak získať unikátny pohľad na dáta.

V prvom cieli sa zameriavame na analyzovanie možnosti spracovania forenzných artefaktov operačného systému Windows prostredníctvom dátovej analýzy a to pomocou detailného preskúmania forenzných artefaktov. Druhý cieľ zahŕňa preskúmanie a porovnanie existujúcich prístupov k automatizovanému spracovaniu forenzných artefaktov operačného systému Windows. Posledný cieľ sa zameriava na samotný návrh a implementáciu nástroja, ktorý dokáže automatizovane spracovávať forenzné artefakty a generovať rôzne štatistiky.

Práca je rozdelená do štyroch kapitol. V prvej kapitole sa venujeme teoretickým základom a podrobnej analýze jednotlivých forenzných artefaktov. Táto časť je dôležitá pre porozumenie základných pojmov a problémov súvisiacich s automatizáciou digitálnej forenznej analýzy. Druhá kapitola študuje už existujúcich nástrojov na spracovanie forenzných artefaktov a ich porovnaním medzi sebou a implementáciou nášho riešenia. Tretia kapitola obsahuje popis dátovej sady, ktorú budeme využívať na účel odskúšania a vyhodnotenia nášho nástroja. Štvrtá kapitola rozoberá už konkrétny návrh riešenia, postup spracovania artefaktov a finálnu implementáciu riešenia, ktoré sme navrhli.

1 Úvod do digitálnej forenznej analýzy

Digitálna forenzná veda je odvetvie forenzných vied, ktoré využíva vedecké poznatky na zhromažďovanie, analýzu, dokumentáciu a prezentáciu digitálnych dôkazov súvisiacich s počítačovou trestnou činnosťou na účely ich použitia na súde. Konečným cieľom je zistiť, čo bolo vykonané, kedy to bolo vykonané a kto to urobil. Pojem "digitálna forenzná veda" sa všeobecne používa ako synonymum pre počítačovú forenznú vedu (známu aj ako kyberforenzná veda), ale rozšíril sa na vyšetrovanie všetkých zariadení, ktoré sú schopné uchovávať digitálne údaje [1].

Najjednoduchšie vysvetlenie by mohlo byť, že ide o súbor techník a nástrojov na skúmanie digitálnych úložísk a digitálnych prostredí s cieľom zistiť, čo sa stalo. "Čo sa stalo" v tomto kontexte by mohlo byť, či bol alebo nebol spáchaný trestný čin, či niekto diaľkovo ovládal alebo neovládal určité zariadenie, kedy bola urobená fotografia alebo či bol počítač predmetom narušenia. To znamená, že to môže byť v podstate čokoľvek. Pri pohľade na cieľ niektorých skutočných forenzných vyšetrovaní je však zrejmé, že konštatovanie "Čo sa stalo" nepokrýva celú oblasť počítačových forenznej analýzy, pretože forenzní experti skúmajú aj to, čo sa práve deje [24].

1.1 Digitálna stopa

Digitálne stopy boli predtým definované ako akékoľvek údaje, ktoré môžu preukázať, že bol spáchaný trestný čin, alebo môžu poskytnúť spojenie medzi trestným činom a jeho obeťou alebo trestným činom a jeho páchateľom [25]. Táto definícia sa však príliš zameriava na dôkaz a zanedbáva údaje, ktoré len podporujú vyšetrovanie.

Digitálne stopy je lepšie definovať ako informácie a údaje, ktoré majú hodnotu pre vyšetrovanie a ktoré sú uložené, prijaté alebo odoslané elektronickým zariadením [41]. Podporujú alebo vyvracajú teóriu o tom, ako došlo k trestnému činu, alebo sa týkajú rozhodujúcich prvkov trestného činu, ako je úmysel alebo alibi. Údaje uvedené v tejto definícii sú v podstate kombináciou čísel, ktoré predstavujú informácie rôzneho druhu vrátane textu, obrázkov, zvuku a videa [25].

1.2 Forenzné artefakty v operačnom systéme Windows

Pojem „artefakt“ v súčasnosti nemá formálnu definíciu v oblasti kybernetickej/digitálnej forenznej analýzy. Tento termín bol vo všeobecnosti prijatý v rámci kybernetickej forenznej oblasti pre objekty, ktoré pomáhajú pri napredovaní vyšetrovania [2]. Medzi najznámejšie a najčastejšie analyzované artefakty patria takzvané záznamy udalosti (event logy), ktoré napríklad ukladajú informácie o vytvorení nového procesu alebo o prihlásení sa do zariadenia. Medzi ďalšie patria prefetch súbory, ktoré ukladajú konkrétne údaje o spustených aplikáciách s cieľom pomôcť im pri spustení rýchlejšie. Zaujímavé sú aj LNK (Shell link) súbory, rozsiahle registre alebo MFT (Master File Table) súbory.

V nasledujúcich kapitolách sa budeme venovať vybraným forenzným artefaktom operačného systému Windows. Hlavným cieľom tejto časti práce je popis týchto artefaktov a analýza ich významu pre digitálnu forenznú analýzu.

Zámerne sme v rámci analýzy opomenuli register operačného systému Windows ako celok. Ide o operačným systémom definovanú databázu, v ktorej aplikácie a komponenty operačného systému ukladajú a získavajú konfiguračné údaje. Údaje uložené v registri sa líšia v závislosti od verzie operačného systému Windows. Aplikácie používajú rozhranie API registra na získanie, úpravu alebo vymazanie údajov tohto registra [45].

Tento register v sebe obsahuje niekoľko dôležitých forenzných artefaktov, ktoré mnohokrát nie je možné spracovať automatizovaným spôsobom. Dôvodom je ich nízky počet. Z tohto dôvodu sme sa v práci zamerali len na vybrané forenzné artefakty, ktoré sú uložené v tomto registri a to sú záznamy udalostí, MFT, Journal, Prefetch, Jump listy, Shell linky, Amcache, SRUM a Shellbagy.

1.3 Záznamy udalostí

Operačný systém Windows zaznamenáva dôležité udalosti (hardvérové aj softvérové), ktoré sa stali v systéme, aplikáciám alebo iným službám v zázname udalostí (event logu). Zaznamenávanie udalostí, ako je nedostatok pamäte, nadmerný prístup k pevnému disku, neúspešné prihlásenie a iné, môže pomôcť správcom systému alebo

používateľom zistiť presný zdroj konkrétnej udalosti a pomôcť im predpovedať budúce udalosti (napr. výmena pevného disku pred jeho úplným zlyhaním).

Z forenzného hľadiska záznamy udalostí (event logy) operačného systému Windows pomáhajú vyšetrovateľom zistiť, čo používateľ v určitom čase urobil na zariadení, resp. čo sa vykonávalo na danom zariadení Hlavné prvky každého logu v zázname logov sú nasledovné [1]:

- **Používateľ:** Používateľské meno účtu prihláseného do zariadenia, keď nastala udalosť.
- **ID udalosti:** Číslo vygenerované systémom Windows, ktoré identifikuje typ udalosti.
- **Zdroj:** Objekt, ktorý spôsobil udalosť.
- **Počítač:** Názov počítača, na ktorom došlo k udalosti.
- **Dátum a čas:** Dátum a čas, kedy k udalosti došlo.
- **Popis:** Popis toho, čo sa stalo pri spustení udalosti

Windows záznamy udalostí (event logy) sú uložené v binárnom formáte XML, ktorý je pre textový editor nečitateľný. Operačný systém Windows ponúka jednoduché grafické rozhranie nazývané Event Viewer, ktoré je však schopné čítať protokoly a konvertovať ich na čistý text XML. Predvolené umiestnenie event logov operačného systému Windows je zvyčajne C:\Windows\System32\winevt\Logs [3].

1.3.1 Typy udalostí

Existuje päť typov udalostí podľa ich závažnosti, ktoré možno zaznamenať a ktoré sú popísané v tabuľke č. 1.

Typ udalosti	Popis

Error	Udalosť, ktorá môže naznačovať závažný problém, napríklad stratu údajov alebo stratu funkčnosti. Ak sa napríklad služba počas spúšťania nenačíta, zaznamená sa udalosť Error (Chyba).
Warning	Udalosť, ktorá nemusí byť nevyhnutne významná, ale môže naznačovať možný problém v budúcnosti. Zaznamená sa napríklad vtedy, ak je na disku málo miesta.
Information	Udalosť, ktorá opisuje úspešnú prevádzku aplikácie, ovládača alebo služby. Táto udalosť sa môže zaznamenať napríklad vtedy, keď sa úspešne načíta sieťový ovládač.
Success Audit	Udalosť, ktorá zaznamenáva úspešný auditovaný pokus o prístup k zabezpečeniu. Napríklad úspešný pokus používateľa o prihlásenie do systému.
Failure Audit	Udalosť, ktorá zaznamenáva neúspešný auditovaný pokus o bezpečnostný prístup. Ak sa napríklad používateľ pokúsi získať prístup k sieťovej jednotke a neuspeje z dôvodu neposkytnutia správnych poverení alebo prihlasovacích údajov .

Tab. 1 Popis typov udalostí. Prevzaté z [26]

1.3.2 ID udalosti typu Logon

V rámci práce sme sa bližšie zamerali na už spomínané ID udalosti. To nám pomáha identifikovať, aká udalosť nastala v zariadení. Logon udalosti zaznamenávajú každé prihlásenie a odhlásenie na počítači a o aký typ prihlásenia ide. Vieme z nich vyčítať každý legitímny aj nelegitímny pokus o prihlásenie, či už bol úspešný alebo nie. V tabuľke č. 2 sú popísané všetkých Logon udalosti a ich ID.

Logon udalosti	**Popis**
4624	Používateľ sa úspešne prihlásil do počítača. Informácie o type prihlásenia nájdete v tabuľke Typy prihlásenia nižšie.

Logon udalosti	Popis
4625	Zlyhanie prihlásenia. Došlo k pokusu o prihlásenie s neznámym používateľským menom alebo známym používateľským menom so zlým heslom.
4634	Proces odhlásenia používateľa bol dokončený.
4647	Používateľ spustil proces odhlásenia.
4648	Používateľ sa úspešne prihlásil do počítača pomocou explicitných poverení, pričom už bol prihlásený ako iný používateľ.
4779	Používateľ odpojil reláciu terminálového servera bez odhlásenia.

Tab. 2 . ID udalosti typu Logon [4]

V tabuľke č. 2 sú popísané typy prihlásení logon.

Typ prihlásenia	Názov	Popis
2	Interactive	Používateľ sa prihlásil do počítača.
3	Network	Užívateľ alebo počítač sa prihlásil do tohto počítača zo siete.
4	Batch	Batch typ prihlásenia používajú batch servery, kde môžu byť procesy vykonávané v mene užívateľa bez jeho priameho zásahu.
5	Service	Služba bola spustená manažérom riadenia služieb.
7	Unlock	Táto pracovná stanica bola odomknutá.
8	NetworkCleartext	Používateľ sa k tomuto počítaču prihlásil zo siete. Heslo používateľa bolo odovzdané do overovacieho balíka v nehašovanej podobe. Zabudovaná autentifikácia zabalí všetky hešovacie poverenia pred ich odoslaním cez sieť. Poverenia neprechádzajú sieťou v čistom texte.

Typ prihlásenia	Názov	Popis
9	NewCredentials	Volajúci naklonoval svoj aktuálny token a špecifikoval nové poverenia pre odchádzajúce pripojenia. Nová prihlasovacia relácia má rovnakú lokálnu identitu, ale používa iné poverenia pre iné sieťové pripojenia.
10	RemoteInteractive	Používateľ sa k tomuto počítaču prihlásil vzdialene pomocou terminálových služieb alebo vzdialenej pracovnej plochy.
11	CachedInteractive	Používateľ sa prihlásil na tento počítač pomocou sieťových poverení, ktoré boli uložené lokálne v počítači. Radič domény nebol kontaktovaný na overenie poverení.

Tab. 3 . Typy prihlásenia ID udalostí [4]

1.4 Forenzné artefakty spojené so súborovým systémom

New Technology File System (NTFS) je súborový systém, ktorý operačný systém Windows používa na efektívne ukladanie, organizovanie a vyhľadávanie súborov na pevnom disku.

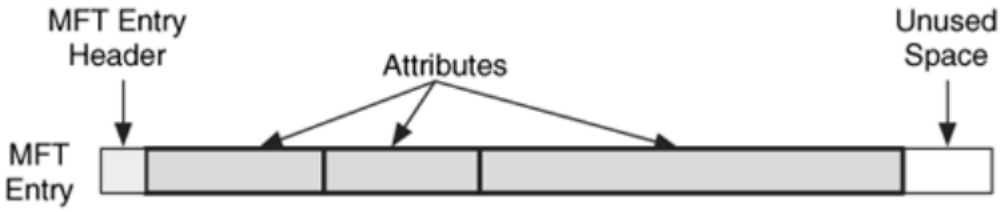
V súborovom systéme NTFS je každý objekt súborom a ako taký zdedil všetky vlastnosti súboru. Patria sem aj metadáta súborového systému, ktoré definujú štruktúru samotného súborového systému. NTFS organizuje štruktúry na zväzku disku do štyroch logických blokov

1.4.1 MFT

Najdôležitejšou funkciou systému NTFS je hlavná tabuľka súborov (Master File Table - MFT), ktorá je implementovaná ako pole záznamov. Každý súbor a každý adresár má aspoň jeden záznam v MFT vrátane samotnej MFT [17]. V rámci tohto súboru sa udržiavajú nasledujúce informácie o súboroch:

- názov súboru,
- veľkosť,
- umiestnenie súboru na disku,
- informáciu o časových pečiatkach,
- oprávnenia,
- obsah súboru (ak je súbor menší ako ~ 900 bajtov).

Každý záznam v MFT má predvolenú pevnú veľkosť 1024 B. Prvých 42 bajtov v zázname súboru je vyhradených pre hlavičku záznamu MFT a zvyšné bajty sa používajú na uloženie tzv. atribútov. Atribút je malá dátová štruktúra so špecifickým účelom, napríklad $STANDARD_INFORMATION, $FILE_NAME alebo $DATA. Dva atribúty, $STANDARD_INFORMATION a $FILE_NAME, sú prítomné v každom zázname súboru nezávisle od typu súboru [17]. Na Obr. 1 je znázornená štruktúra záznamu MFT.



Obr. 1 Štruktúra záznamu MFT. Prevzaté z [17]

$STANDARD_INFORMATION je typu 0x10. Ide o povinný, prvý atribút v každom súbore. Zvykne byť pravidelne aktualizovaný.

$FILE_NAME je typu 0x30 a obsahuje uchováva názov súboru a číslo inódu nadradeného adresára, ktorý obsahuje tento súbor. Môže mať viacero atribútov názvu súboru, aby podporoval krátky názov súboru založený na operačnom systéme MS-DOS. Odkaz na nadradený adresár má dve časti [21]:

- 6 bajtov pre číslo inódu nadradeného adresára a
- bajty pre interné sekvenčné číslo na kontrolu integrity systému NTFS.

MFT ukladá štyri časové značky pre každý objekt súboru v $STANDARD_INFORMATION a $FILE_NAME atribútoch (uvádzajú sa vo formáte FILETIME[18]):

- **Create (C) -** čas vytvorenia záznamu o súbore.
- **Modify (M)** - čas poslednej úpravy atribútu $DATA (obsah súborového objektu).
- **Access (A)** - čas posledného prístupu k obsahu súboru.
- **Modification time MFT** (X) - čas poslednej modifikácie metadát súboru.

Súborový systém NTFS vyhradzuje priestor pre MFT, aby sa MFT pri svojom raste udržal čo najviac súvislý. Priestor vyhradený súborovým systémom NTFS pre MFT v každom zväzku sa nazýva zóna MFT. Z tohto priestoru sa prideľuje aj priestor pre súbory a adresáre, ale až po pridelení všetkého priestoru zväzku mimo zóny MFT [19]. Po vyčerpaní voľného miesta v pôvodnom súbore $MFT sa zóna MFT rozdelí na polovicu. Ak je celá Zóna MFT vyčerpaná a súbor MFT musí ešte rásť, potom sa súbor $MFT fragmentuje. Fragmentovaný súbor $MFT už nemožno defragmentovať [21].

1.4.2 Journal

Journal USN (Update Sequence Number Journal), ako už názov napovedá, je databáza na ukladanie všetkých zmenených informácií zväzku systému NTFS. Každý zväzok NTFS má svoju vlastnú databázu USN Journal. Obsahuje niekoľko užitočných informácií, medzi ktoré zaraďujeme [22]:

- odkaz na nadradený MFT,
- informácie o zmenách,
- časová pečiatka - časová pečiatka zmeny súboru,
- názov súboru - názov súboru, ktorý sa zmenil,
- atribút - používa sa hlavne na určenie medzi súborom a adresárom,
- dôvod - Akcia, ku ktorej došlo, príklady sú:
    - CLOSE,
    - DATA_EXTEND,
    - DATA_OVERWRITE,
    - DATA_TRUNCATION,
    - FILE_CREATE,

- FILE_DELETE,
- RENAME_NEW_NAME,
- SECURITY_CHANGE.

Journal USN je štruktúrovaný v dvoch alternatívnych dátových tokoch (ADS) a v súbore slack. Prúdy denníka sú uložené v súboroch $UsnJrnl:$J (ADS) a $UsnJrnl:$Max. Súbor $J je redší súbor. Všetky údaje, ktoré sa ukladajú v tomto journále, sú uložené sekvenčne vo formáte celočíselného čísla bez znamienka v súbore $J. Keď veľkosť súboru USN Journal prekročí definovanú hodnotu, žurnál sa otočí a začne prepisovať staré údaje (FIFO prístup) [23].

Tento súbor využívajú najmä zálohovacie aplikácie na určenie súborov, ktoré sa zmenili od poslednej operácie zálohovania. Pri každej zmene zväzku sa do súboru pridá záznam. Každý záznam je identifikovaný 64-bitovým poradovým číslom aktualizácie alebo USN [20].

1.5 Forenzné artefakty spojené so spúšťaním súborov

Analýza vykonávania programov odhaľuje, čo bolo v systéme spustené a často je zmysluplnou snahou odhaliť činnosti používateľov vykonávané v operačnom systéme. Analýzou artefaktov vykonávania programov môžu forenzní vyšetrovatelia zistiť, aké programy boli nainštalované v systéme a históriu ich vykonávania, dokonca aj dôkazy o odstránených programoch. Artefakty vykonávania programov sa vytvárajú a ukladajú na rôzne miesta na pevnom disku, ktoré operačný systém ďalej využíva na poskytovanie lepších používateľských používateľského zážitku [12].

1.5.1 Prefetch

Prefetching je funkcia používaná v rámci operačného systému Windows na zrýchlenie načítania aplikácií. Keď používateľ spustí aplikáciu prvýkrát, operačný systém Windows vytvorí Prefetch súbor, a následne zaznamená, ktoré súbory boli načítané v rámci tohto spustenia aplikácie. Napomáha to tomu, že pri ďalšom spustení aplikácie, systém Windows načíta tieto súbory rýchlejšie. Z digitálneho forenzného hľadiska nám funkcia Prefetch môže povedať, ktoré programy boli spustené v cieľovom systéme, aj

keď bol predmetný program po jeho spustení odinštalovaný, pretože sa to zaznamená v priečinku Windows Prefetch [1].

Konfigurácia nástroja Prefetch je uložená v nasledujúcom kľúči databázy Registry systému Windows: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control Session Manager\Memory Management\PrefetchParameters. Windows ukladá tieto súbory do adresára C:\Windows\Prefetch. Všetky tieto súbory sú pomenované podľa bežných kritérií [1].

Predvolená konvencia pomenovania súboru prefetch je <Názov_spustiteľného>-<Heš_z_adresára>.pf. Ako príklad si vezmime program zoom.exe by sa mohol zobraziť ako ZOOM.EXE-87652BD0.pf, kde 87652BD0 je heš cesty, z ktorej bol spustený program zoom.exe. To znamená, že pre tú istú aplikáciu môže existovať viacero súborov prefetch, ktoré však zodpovedajú samostatným umiestneniam adresárov [42].

Operačný systém Windows pravidelne skúma obsah súborov .pf a zapisuje súbory a adresáre používané procesom do súboru layout.ini, ktorý obsahuje pôvodné názvy ciest k súborom umiestnených v Prefetch [2].

Okrem už spomínaného názvu a cesty obsahujú súbory prefetch aj podrobnosti o počte spustení aplikácie, podrobnosti o zväzku, ako aj informácie o časovej značke, ktoré uvádzajú, kedy bola aplikácia spustená prvýkrát a naposledy. V prípade systému Windows 8+ teraz súbory prefetch obsahujú až osem časových značiek o tom, kedy bola aplikácia naposledy spustená, čo vyšetrovateľom poskytuje niekoľko ďalších časových značiek, ktoré pomáhajú vytvárať časovú os udalostí v systéme [43].

Vo všeobecnosti je prvý čas spustenia spojený s časom vytvorenia súboru .pf. Toto však nemusí platiť vždy, keďže samotný priečinok Prefetch je obmedzený iba na 1024 súborov pre operačný systém Windows 8/10. Ak aplikácia nie je už nejaký čas spustená, môže sa odstrániť z priečinka Prefetch. Ak je aplikácia spustená po jej odstránení z priečinka, novému Prefetch súboru (.pf) bude priradený nový čas vytvorenia, pretože ide o úplne nový súbor. Inými slovami, pre presnosť interpretácie .pf súboru, prvé spustenie je prvé spustenie aplikácie, pokým súbor nebol vymazaný. Nie je to prvýkrát, čo bola aplikácia spustená, aj keď to tak môže byť [2].

1.5.2 Jump Listy

Jump listy (zoznamy) boli pridané do operačného systému Windows 7 v júli 2009 ako nová funkciou a neskôr boli zachované aj v nasledujúcich verziách operačného systému Windows vrátane systému Windows 10. Softvérové programy a operačné systémy vytvárajú Jump listy, aby používatelia mohli "preskočiť" priamo na nedávno otvorené súbory a priečinky. Funkcia zobrazuje len malý výber položiek na paneli Štart alebo paneli úloh. Väčšina zozbieraných údajov nie je viditeľná. Používatelia operačných systémov Windows 7 a Windows 8 môžu pomocou nástroja "regedit" zmeniť hodnotu registra, ktorá riadi počet zobrazených položiek zoznamu skokov. Počet položiek Jump zoznamu, ktoré zobrazuje operačný systém Windows 10 a 11, však nie je možné zmeniť, pretože je pevne zakódovaný [8].

Z hľadiska digitálnej forenznej analýzy je možné z Jump zoznamov získať nasledujúce informácie:

- názov a cestu k cieľovému súboru alebo disku,
- veľkosť cieľového súboru,
- sadu časových pečiatok, ktoré popisujú cieľový súbor,
- názov a sériové číslo disku, v ktorom je uložený cieľový súbor,
- prvú dostupnú MAC adresu zariadenia, na ktorom bol prístup k cieľovému súboru.

Časová pečiatka pripojená ku každému zoznamu Jump List je jednou z najcennejších informácií, ktoré možno zo zoznamov Jump List vyčítať. Tieto informácie sú potrebné pre forenzných vyšetrovateľov na zostavenie presnej časovej osi udalostí. Niektoré Jump Listy zobrazujú nedávno zatvorené webové stránky a súbory, zatiaľ čo iné zobrazujú najčastejšie prístupy, preto to neplatí vždy pre každú aplikáciu. Ďalšie dôležité informácie je možné nájsť v adresári súboru Jump Listed, ktorý sa nazýva Jump List hex. Ďalšia vec, ktorú treba mať na pamäti, je, že operačný systém Windows štandardne obmedzuje počet Jump Listov, ktorých zvyčajne môže byť len 10 [9].

Názvy súborov sa začínajú s AppId, čo je reťazec 16 hexadecimálnych číslic, za ktorým nasledujú tajné prípony súborov automaticDestinations a customDestinations. Prvá skupina obsahuje údaje o tom, ako sa dátové súbory používajú. V závislosti od aplikácie sú položky zoradené buď podľa Najnovšie používané (MRU), alebo podľa Najčastejšie používané (MFU). Typ súboru sa nachádza v druhej vymenovanej skupine

customDestinations. Jedinečný program zodpovedný za tento konkrétny cieľový súbor udržiava údaje v ňom obsiahnuté a úlohy určené týmto druhom súboru [10].

.

1.5.3 Shell link (LNK)

Súbor **Shell Link** alebo Shortcut systému Windows (prípona LNK) je typ súboru, ktorý spoločnosť Microsoft široko používa v rôznych verziách operačného systému Windows na pomoc pri spúšťaní aplikácií, prístupe k súborom prostredníctvom zachovania odkazu na cieľový súbor a v scenároch prepojenia a vloženia objektov (OLE). Súbor skratky začína veľkosťou hlavičky reprezentovanou 4 bajtmi podpisu 0x0000004C (desatinná hodnota 76) [12].

Takéto súbory sa bežne nachádzajú na pracovnej ploche používateľa. Môžeme ich však vidieť roztrúsené aj na iných miestach. Súbory LNK môže vytvoriť používateľ alebo ich môže automaticky generovať operačný systém Windows, keď používateľ otvorí miestny alebo vzdialený súbor. Súbory LNK obsahujú okrem zariadenia, v ktorom sa súbor aktuálne nachádza, aj množstvo užitočných informácií o zariadení, v ktorom bol prvýkrát vytvorený. Súbory LNK sú z forenzného hľadiska cenné, pretože odhaľujú nasledujúce informácie [1]:

- atribúty času MAC (čas vytvorenia, modifikácie a prístupu) pre samotný súbor LNK a pre prepojený súbor.
- Predchádzajúce činnosti používateľa na zariadení. Napríklad ak podozrivý presunie súbor na jednotku USB alebo ho natrvalo odstráni zo svojho/počítača, súvisiaci súbor LNK bude stále existovať, čo poskytne cenné informácie o tom, čo sa vykonalo na cieľovom systéme predtým.
- Veľkosť prepojeného súboru.
- Pôvodnú cestu k prepojenému súboru.
- Sériové číslo a názov zväzku, na ktorom sa nachádza prepojený súbor. Adresu MAC sieťového adaptéra a pôvodná sieťová cesta pôvodného zariadenia.

1.5.4 Amcache

Infraštruktúra kompatibility aplikácií bola zavedená v operačných systémoch Windows počnúc systémom Windows XP. Táto infraštruktúra je opísaná v dokumentoch spoločnosti Microsoft [13]. Zjednodušene povedané, umožňuje spustiť aplikáciu, aj keď už nie je plne kompatibilná so systémom, na ktorom beží, alebo ak sa zmenila verzia závislosti. Táto infraštruktúra, nazývaná aj Shim Infrastructure, poskytuje digitálnemu riešiteľovi dva artefakty: vyrovnávaciu pamäť kompatibility aplikácií (nazývanú aj ShimCache) a od systému Windows 7 aj AmCache. AmCache sa v súčasnosti vyskytuje v dvoch rôznych formátoch súborov: súbor BCF s názvom RecentFileCache.bcf, a v registri pod názvom AmCache.hve [14].

Súbor Amcache.hve je cenným zdrojom pre analýzu nedávno spustených aplikácií a histórie ich vykonávania v kontexte analýzy činnosti používateľa. Súbor Amcache.hve sa pôvodne objavil v operačnom systéme Windows 8 ako náhrada za súbor RecentFileCache.bcf operačného systému Windows 7.

Súbor Amcache.hve uchováva v súbore registra informácie o funkcii Windows Application Experience and Compatibility. Forenzné vyšetrovanie súboru Amcache.hve môže odhaliť významné atribúty, ako je názov a verzia programu, cesta k vykonávaciemu súboru, veľkosť súboru, časová pečiatka inštalácie, časová pečiatka prvého spustenia, heš spustiteľného súboru atď. Tieto informácie pretrvávajú v súbore Amcache.hve aj po odstránení aplikácií a adresára Prefetch v počítači používateľa. Z tohto dôvodu je Amcache.hve užitočný aj pri detekcii antiforenzných nástrojov, čím odhaľuje zámery používateľa [15].

Veľmi významný je atribút "Čas posledného zápisu", ktorý je uložený na každom kľúči spustiteľného súboru. Táto časová pečiatka sa vytvorí pri prvom spustení spustiteľných súborov. Vytvára sa len pri prvom spustení a potom sa nedá zmeniť. Pomocou tejto funkcie možno skontrolovať čas prvého spustenia súboru [16].

1.5.5 SRUM

Nová technológia známa ako System Resource Usage Monitor (SRUM) bola prvýkrát uvedená v systéme Windows 8. Systém SRUM sleduje informácie o procesoch a sieti v priebehu času v databáze . Podrobnosti o procesoch, používateľoch, cykloch CPU

a sieťových údajoch doručených alebo prijatých konkrétnym procesom patria medzi údaje, ktoré SRUM zhromažďuje. Väčšina databázy SRUM sa nezobrazuje a len malá časť týchto údajov je prístupná koncovým používateľom. Pri uvedení systému Windows 8.1 v októbri 2013 sa táto technológia začala používať a rozvíjať. SRUM poskytuje forenzným vyšetrovateľom historický pohľad na predchádzajúce používanie počítačového systému. Táto databáza môže byť pre forenzných vyšetrovateľov veľmi užitočná pri sledovaní činnosti používateľov a spájaní činností procesov, používateľov a siete. [29].

Umiestnenie databázy SRUM je C:WindowsSystem32sruSRUDB.dat. V súbore sa nachádza databáza ESE (Extensible Storage Engine). Ide o rovnaký typ databázy, aký používajú aplikácie Microsoft Exchange, Active Directory, Microsoft Search a ďalšie aplikácie. Aktualizácie databázy sa vykonávajú každú hodinu a pri vypnutí systému. Informácie, ktoré ešte neboli vložené do databázy, sa dočasne uchovávajú v registri [39]. Artefakty nástroja System Resource Utilization Monitor sú rozdelené do kategórií [40]:

- využitie aplikačných zdrojov,
- sieťové pripojenia,
- využitie energie,
- využitie siete,
- energy Usage (Long Term) - využitie energie SRUM (dlhodobé),
- push Notification Data (údaje o oznámení Push).

Spomedzi rôznych kategórií artefaktov je SRUM Application Resource Usage jednou z najužitočnejších kategórií. Je to najmä z toho dôvodu, že sleduje každý exe súbor, ktorý sa spustil v systéme, bez ohľadu na to, či ešte existuje na disku alebo nie. Súčasne ukladá úplnú cestu, z ktorej sa aplikácia spustila. To nám môže pomôcť odfiltrovať neočakávané cesty aplikácií [39].

1.6 Shellbagy

Dôležitou súčasťou viacerých forenzných vyšetrovaní je register operačného systému Windows. Ako sme už vyššie uviedli, ide o databázu, ktorá obsahuje konfiguračné údaje o hardvéri, softvéri a operačnom systéme počítača. Jednou zo zložiek registra, ktorá môže byť pri vyšetrovaní kľúčová, je shellbag. Kľúče registra známe ako shellbagy sa vytvárajú vždy, keď sa v počítači pristupuje k priečinku a používajú sa na uloženie možností zobrazenia priečinkov v Prieskumníkovi súborov. Pomocou shellbagov možno vytvoriť históriu činností používateľa v systéme, ktorá sa dá použiť aj na obnovenie stratených údajov súborového systému [35]. Tieto Shellbagy obsahujú informácie s forenznou hodnotou ako napríklad [36]:

- Identifikácia súborov, ku ktorým konkrétny používateľ pristupoval pomocou Prieskumníka systému Windows, či už lokálne, vzdialene alebo z odpojiteľného disku, ako je napríklad disk USB, externý pevný disk alebo akýkoľvek iný disk.
- Prítomnosť záložných kópií odstránených alebo prepísaných súborov z predchádzajúcich priečinkov.
- Množstvo používateľov, ktorí mohli mať prístup k určitým adresárom v rámci celého systému.
- Spôsob prístupu k určitému priečinku, ktorým môže byť buď zástupca, alebo koreňový adresár prieskumníka systému Windows.

Kľúče registra BagMRU a Bags tvoria dve hlavné zložky informácií o artefaktoch Shellbags. Kľúče, ktoré reprezentuje BagMRU, sú na rozdiel od bežného BagMRU kľúče pracovnej plochy, pretože podriadené kľúče nie sú priradené k žiadnym konkrétnym priečinkom. Tým, že tieto hodnoty vytvárajú podobnú stromovú štruktúru, ukladajú názvy a cesty k priečinkom, zatiaľ čo kľúče Bags obsahujú preferencie pre pozíciu, režim zobrazenia a veľkosť okna [37]. Kľúč BagMRU a jeho podkľúče uchovávajú informácie o priečinkoch, ktoré používateľ naposledy prezeral. Tieto stránky sú špeciálnym typom kľúča MRU (Most Recently Used) [38].

2 Porovnanie existujúcich prístupov

V predchádzajúcej kapitole sme sa venovali teoretickejšej časti, ktorá je založená podrobnom rozobraní jednotlivých forenzných artefaktov. Táto časť je dôležitá pre porozumenie základných pojmov a problémov súvisiacich s automatizáciou digitálnej forenznej analýzy. V tejto kapitole sa zameriavame na existujúce prístupy, resp. riešenia, ktoré napomáhajú automatizovanému spracovaniu forenzných artefaktov. Súčasťou tejto kapitoly je porovnanie týchto riešení z rôznych hľadísk a najmä porovnanie s našim riešením.

2.1 MSTIC Jupyter and Python Security Tool

Prvým príkladom je nástroj od spoločnosti Microsoft s názvom Microsoft Threat Intelligence (**MSTICPy)** [31]. Ide o súbor nástrojov v programovacom jazyku Python určených na pomoc bezpečnostným analytikom pri ich výskume a analýze bezpečnostných hrozieb. Cieľom MSTICPy je:

- Poskytnúť bloky základných funkcií, ktoré uľahčia proces vývoja a používania zápisníkov pre bezpečnostnú analýzu.
- Znížiť množstvo kódu potrebného v zápisníkoch (jupyter notebookoch), aby sa zvýšila ich použiteľnosť.
- Sprístupniť funkcionalitu všetkým, aby ju mohli využívať a podieľať sa na jej vývoji.

Tento nástroj zahŕňa štyri funkčné oblasti, medzi ktoré zaraďujeme:

- vyhľadávanie a importovanie údajov,
- obohatenie údajov,
- analýzu údajov a
- vizualizáciu.

Vyhľadávanie a importovanie údajov. Ide o proces získavania bezpečnostných údajov do jupyter notebooku. Patria sem poskytovatelia údajov, predpripravené dopyty na prístup k rôznym úložiskám bezpečnostných údajov (napríklad Azure Sentinel, Microsoft Defender, Splunk a Microsoft Graph) a moduly na ukladanie/vyberanie súborov z úložiska Azure blob a nahrávanie údajov do Azure Sentinel/Splunk.

Obohatenie údajov sa týka prvkov, ktoré ponúkajú viac kontextu k udalostiam pozorovaným v údajoch, ako sú napríklad informácie o hrozbách a vyhľadávanie geografickej polohy. Okrem toho má k dispozícii rozhranie Azure API, ktoré možno použiť na získanie informácií o zdrojoch, ako sú virtuálne počítače a predplatné v službe Azure.

Analýza údajov. Jednotlivé časti (balíky) sa v tejto oblasti zameriavajú na komplexnejšie spracovanie údajov vrátane dekódovania z base64 kódovania, zhlukovania, analýzy časových radov, detekcie anomálií a extrakcie vzorov kompromitácie (Indicator of Compromise - IoC). Pivotné funkcie, ktoré poskytujú prístup k mnohým funkciám MSTICPy prostredníctvom entít (napríklad všetky funkcie súvisiace s IP adresou sú prístupné ako metódy triedy entít IpAddress), sú ďalším prvkom, ktorý sem zahŕňame, ale v skutočnosti patria do všetkých troch prvých kategórií.

Vizualizácia pozostáva z nástrojov, ako sú časové osi udalostí, stromy procesov, mapovanie, morfogramy a vizualizácia časových radov, ktoré pomáhajú používateľom vidieť údaje alebo výsledky analýz. Do tejto časti patria aj početné widgety poznámkového bloku, ktoré uľahčujú alebo urýchľujú činnosti, ako je napríklad zadávanie rozsahov dátumov dopytov a výber položiek zo zoznamu. Nachádzajú sa tu aj prehliadače spravodajských informácií o hrozbách a výstrahách, ako aj množstvo ďalších prehliadačov pre komplexné údaje a na navigáciu po interných funkciách.

2.2 Nástroje projektu DS4N6 (Data Science Forensics)

Prvým príkladom nástrojov od skupiny DS4N6 je nástroj Chrysalis [32]. Ten umožňuje forenznému analytikovi vykonávať forenznú analýzu a dátovú analýzu a tým, že zjednodušuje prijímanie a analýzu výstupov z forenzných nástrojov (plaso, kape, kansa, volatility atď.) v prostredí Jupyter/pandas Data Science.

Digitálni forenzní analytici môžu CHRYSALIS ľahko používať. Nie sú potrebné žiadne odborné znalosti jazyka Python a má používateľsky prívetivé rozhranie. Skúsení digitálni forenzní analytici ho môžu používať na využitie prístupov dátovej analýzy a strojového učenia na zvýšenie kvality komplexnosti analýzy.

Chrysalis obsahuje 9 základných funkcií nápomocných pre vykonanie úplného vyšetrovania. V tabuľke č. 3 sú popísané základné funkcie systému Chrisalis.

Funkcia	**Použitie**	**Typ**	**Popis**
whatis()	whatis(obj)	CLI	Identifies the forensic data type of an object (DataFrame -df- or DataFrame Collection -dfs-)
xread()	xread(options)	GUI	Reads tool output data (e.g. plaso output) and stores it in a df/dfs
xmenu()	xmenu(obj)	GUI	Used to easily select a dataframe from dfs, or a column from a df, displaying the selected data and allowing manual (Excel-like) analysis on it
xanalysis()	xanalysis(obj, options)	GUI	Displays a menu with the advanced analysis functions available for the data type (i.e. forensic artifact) given
xdisplay()	xdisplay()	GUI	Used to select the display settings for the dataframes that will be displayed (max. rows, max. columns, etc.)
simple()	df.simple(options)	CLI	Simplifies forensic output (df) showing only the most interesting columns for analysis.
xgrep()	xgrep(obj, options)	CLI	UNIX-like grep for the DataFrame world. Allows the user to search for a regular expression in a DF column or full DF
plaso_get_evtxdfs()	plaso_get_evtxdfs (obj,options)	CLI	Creates dictionary of events from evtx files using Plaso Dataframe dictionary and the hostname.
evtid_dfs_build()	evtid_dfs_build(obj)	CLI	Creates dictionary of event IDs from Security/System events DataFrame. This helps to identify events based on individual event IDs.

Tab. 3 . základné funkcie systému Chrisalis [32]

Iným príkladom je nástroj **DAISY** [33], ktorý je ďalším krokom projektu DS4N6. Cieľom tohto projektu je uľahčiť prijatie dátovej vedy a umelej inteligencie širokou komunitou DFIR.

Je to virtuálny stroj, ktorý zjednodušuje prijímanie a analýzu výstupov forenzných nástrojov dátovej analýzy a strojového učenia. Je navrhnutý tak, aby bol pre bežného forenzného pracovníka čo najintuitívnejší a najjednoduchší na používanie. Obsahuje nápovedu, postupy dokumentáciu, poznámky a všetko, čo je po ruke, aby bol prechod zo sveta DFIR (Digital Forensics and Incident Response) do sveta dátovej analýzy a čo najpohodlnejší. Cieľom bolo, aby bolo používanie čo najintuitívnejšie a najjednoduchšie pre priemerného forenzného pracovníka.

DAISY obsahuje malú sadu bežných forenzných nástrojov na spracovanie digitálnych stôp - ako napr. nástroj na spracovanie časovej osi (plaso) alebo nástroj na analýzu operačnej pamäte (volatility).

2.3 Live-Forensicator

Black Widow Toolbox obsahuje nástroj s názvom **Live-Forensicator** [34], ktorý pomáha forenzným vyšetrovateľom a pracovníkom, ktorí reagujú na bezpečnostné incidenty, vykonávať krátke forenzné vyšetrovania. Dosahuje to zhromažďovaním rôznych systémových údajov na neskoršie vyhodnotenie pri hľadaní abnormálneho správania sa alebo neočakávaného vstupu údajov. Taktiež sleduje zvláštne súbory alebo akcie a upozorňuje na nich vyšetrovateľa.

V prípade, ak je prijaté upozornenie spojené s útokom ransomvéru, jedna z užitočných vecí, ktoré aplikácia Live-Forensicator dokáže je prehľadávať všetky priečinky v operačnom systéme a hľadať súbory s podobnými príponami ako známy ransomvér. Na tento účel použite parameter -RANSOMWARE.

Pri skúmaní prostriedku, ktorý komunikuje so známou škodlivou IP adresou, môže Live-Forensictor zaznamenávať sieťovú prevádzku pomocou funkcie netsh trace. To umožňuje analyzovať súbor pcapng do nástroja Wireshark a vyhľadávať command-and-control servery (C&C servery).

Pre účely zachovania integrity artefaktov, aby nedošlo k ich kompromitácii, Live-Forensicator môže pomocou argumentu -ENCRYPTED zašifrovať forenzný artefakt

špeciálnym kľúčom, ktorý bol vygenerovaný náhodne. Stačí, ak kľúč uchováte v bezpečí a môžete ho dešifrovať kedykoľvek a kdekoľvek, dokonca aj pomocou inej kópie programu Live-Forensicator. Túto úlohu spracúva súbor FileCryptography.psm1. Na dešifrovanie takto zašifrovaných artefaktov slúži argument -DECRYPT.

Nástroj Live-Forensictor taktiež vyhľadáva podozrivé aktivity v záznamoch logov na základe dlhého zoznamu škodlivých spustiteľných súborov a príkazov prostredia powershell, na ktoré sa dopytuje v týchto záznamoch.

Je veľmi dôležité mať na pamäti, že tento nástroj neposkytne úplný záver analýzy. Namiesto toho musí vyšetrovateľ analyzovať výsledky a určiť, či má vyvodiť záver alebo vykonať dôkladnú štúdiu.

2.4 Porovnanie riešení

V tejto časti sa venujeme porovnaniu nástrojov, resp. riešení, ktoré umožňujú automatizované spracovávanie a analýzu forenzných artefaktov v operačnom systéme Windows.. Na porovnanie sme si vybrali tri nástroje, ktoré sme si bližšie popísali v predchádzajúcich kapitolách. Týmito nástrojmi sú MSTICPy, Chrysalis a Live-Forensicator. Všetky tieto nástroje sú majú otvorený kód, ktorý je zverejnený na platforme GitHub.

	MSTICPy	**Chrysalis**	**Live-Forensicator**	**Naše riešenie**
Otvorený kód	✓	✓	✓	✓
Programovací jazyk	Python	Python	PowerShell	Python, Powershell
Formát vstupu	Nespracované dáta, csv, yaml,	Nespracované dáta, ham	Nespracované dáta	Nespracované dáta, csv
Parsovanie	✓	✓ (využitie externých nástrojov)	✓	✓

Vyhľadávanie signatúr a pravidiel	✓	×	✓	×
Formát výstupu	Jupyter notebook výstupy	Jupyter notebook výstupy	html	Jupyer notebook výstupy, html, csv

Tab. 4 . Porovnanie existujúcich prístupov

V tabuľke č. 4 sa nachádza už spomínané porovnanie existujúcich prístupov a nami implementovaného riešenia. Faktory, ktoré v tomto porovnaní zvažujeme, sú programovací jazyk využitý v danom nástroji, formáty vstupov a výstupov. Dôležité je aj či jednotlivé nástroje sú schopné parsovať vstupné dáta a či dokážu aj priamo vyhľadávať signatúry alebo pravidlá naznačujúce kompromitáciu alebo nechcenú aktivitu. Prvé dva nástroje sú napísané v jazyku Python zatiaľ čo tretí Live-Forensicator je vo forme PowerShell-ového skriptu. Náš nastroj využíva oba jazyky. Powershell slúži na spracovanie dát a Python na načítanie spracovaných csv súborov a prácu s nimi.

Všetky štyri nástroje, vrátane nášho, prijímajú na vstupe nespracované dáta. MSTICPy zbiera nespracované dáta priamo z úložísk bezpečnostných údajov Azure Sentinel, Microsoft Defender, Splunk a Microsoft Graph. Na vstupe je však možné dať aj už spracované dáta vo formáte csv alebo yaml. Zatiaľ čo Chrysalis pracuje s nespracovanými dátami využíva aj Harmonizovaný model artefaktov (HAM). Tento formát zodpovedá údajom, ktoré boli harmonizované podľa modelu HAM, pokiaľ ide o názvy stĺpcov a typy údajov.

Keďže všetky nástroje pracujú s nespracovanými dátami, sú schopné tieto dáta parsovať samé bez použitia externých nástrojov až na výnimku nástroja chrisalis. Ten síce vie pracovať aj s nespracovanými dátami ale využíva aj spracované dáta z rôznych nástrojov ako sú napríklad Kape[1], Volatility[2] a ďalšie. Nástroj MSTICPy je schopný okrem parsovania vykonať aj priamu analýzu časových radov využitím zhlukovania

[1] Dostupné na: https://www.kroll.com/en/insights/publications/cyber/kroll-artifact-parser-extractor-kape

[2] Dostupné na: https://www.volatilityfoundation.org/releases

a umožňuje detekciu anomálií. Taktiež zvláda extrakciu vzorov kompromitácie (Indicator of Compromise - IoC) pomocou už zabudovaných vzorov, ktoré vieme prepísať alebo doplniť vlastným. Live-Forensicator je tiež schopný vyhľadávať v dátach škodlivú aktivitu pomocou zoznamu škodlivých spustiteľných súborov, ako sme už popísali vyššie. Posledným faktorom pre porovnanie je formát výstupu týchto nástrojov. Prvé dve nástroje, rovnako ako náš, využívajú základné výstupy poskytované Jupyter Notebookom, ako sú pdf, html a Markdown. Výstupy z Live-Forensicator-a sú výhradne vo formáte html. Náš nástroj poskytuje aj csv výstupy pri spracovávaní dát..

3 Nástroj na spracovanie forenzných artefaktov z operačného systému Windows

V predchádzajúcich kapitolách sme sa venovali existujúcim prístupom k automatizovanému spracovaniu forenzných artefaktov. Jedným z hlavných cieľom tejto bakalárskej práce je navrhnúť a implementovať nástroj na automatizované spracovanie forenzných artefaktov z operačného systému Windows. Nástroj by mal byť schopný pracovať so širším okruhom artefaktov. V tejto kapitole sa bližšie zameriame na prípadovú štúdiu a dátovú sadu. Súčasne si opíšeme manuálne spracovávanie forenzných artefaktov pomocou powershell skriptu a nástroje, ktoré tu využívame. Na záver si popíšeme implementáciu nášho riešenia na zhotovenie nástroja na automatizované spracovanie forenzných artefaktov v operačnom systéme Windows.

3.1 Dátová sada

Pre účel tejto práce sme vybrali dátovú sadu vytvorenú z prípadu „Odcudzenej Szechuanskej omáčky“. Ide o dátovú sadu, ktorá je prístupná prostredníctvom portálu DFIR Madness a je určená na výučbu forenzných analytikov v oblasti digitálnej forenznej analýzy, reakcie na bezpečnostné incidenty a vyhľadávania bezpečnostných hrozieb [27].

Predmetom ukážkového prípadu je analýza neoprávneného prieniku do počítačovej siete organizácie CITADEL a zverejnenie exkluzívneho receptu na už spomínanú "sečuánsku omáčku". Zámerom tohto prieniku bolo poškodiť spoločnosť a zabrániť mu získať akúkoľvek formu konkurenčnej výhody s touto omáčkou na trhu. Je dôležité poznamenať, že počítač výrobcu omáčky, bol jediným miestom, kde sa recept uchovával.

V prípade sa skúmajú rôzne forenzné stopy, najmä digitálne stopy zo servera - doménového kontroléra (s označením „DC“), klientskeho zariadenia (s označením „DESKTOP“) vrátane zachytenej sieťovej prevádzky a operačnej pamäte. Dostupné sú tieto digitálne stopy:

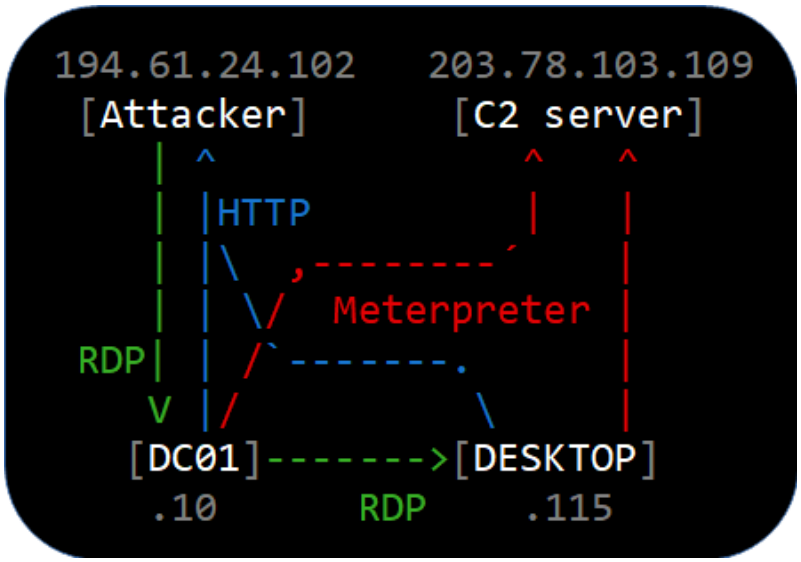
- case001-pcap.zip,
- DC01-autorunsc.zip,

- DC01-E01.zip,
- DC01-memory.zip,
- DC01-pagefile.zip,
- DC01-ProtectedFiles.zip,
- DESKTOP-E01.zip,
- DESKTOP-SDN1RPT-autrunsc.zip,
- DESKTOP-SDN1RPT-memory.zip,
- Desktop-SDN1RPT-pagefile.zip,
- DESKTOP-SDN1RPT-Protected Files.zip.

Keďže digitálnych stôp na analýzu je veľa, v tomto forenznom školení sa nachádzajú aj navádzajúce otázky, ktoré majú riešiteľov udržať na správnom kurze. Zistenie operačných systémov skúmaných zariadení je prvou podpornou informáciou pri analýze. Vieme, že na klientskom počítači bol identifikovaný operačný systém Windows 10 a na serveri operačný systém Windows Server 2016.

Úlohou je zistiť, či boli do systému nainštalované škodlivé aplikácie, vrátane miesta a času inštalácie softvéru. V tomto prípade sa tiež zisťuje, či boli v systéme vytvorené, upravené alebo vymazané nejaké informácie a či nedošlo k úniku údajov.

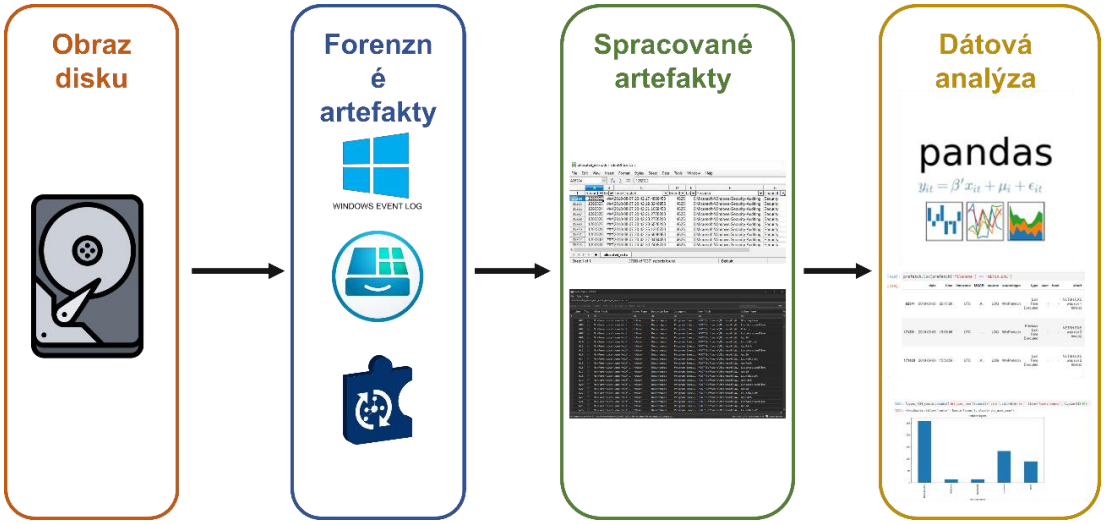
Jednou z kľúčových úloh bolo zistiť, či došlo, resp. nedošlo k úniku údajov zo zariadenia. V prípade potvrdenia úniku bolo úlohou identifikovať spôsob, ako k nemu došlo (viď. Obr. č. 2). Na spustenie útoku na službu vzdialenej plochy (RDP) bol použitý útok hrubou silou z adresy IP 194.61.24.102. O chvíľu neskôr sa útočník úspešne prihlásil pomocou účtu Citadel/Administrator. Keď útočník prišiel k doménovému radiču (DC) pre doménu, spustil druhé pripojenie RDP z doménového radiča na klientske zariadenie s použitím rovnakých prihlasovacích údajov. Prostredníctvom aplikácie Internet Explorer bol na server stiahnutý škodlivý softvér známy ako coreupdater.exe. Útočník potom začal využívať Meterpreter a reláciu grafického rozhrania RDP na vyhľadávanie, exfiltráciu a manipuláciu s údajmi.



Obr. 2 Schéma postupu útoku. Prevzaté z [46].

3.2 Návrh riešenia

Návrh nášho riešenia vychádza zo štandardného procesu digitálnej forenznej analýzy obrazu (imagu) disku a je doplnený o časť venovanú samotnej dátovej analýze. Schéma návrhu riešenia je zobrazená na Obr. č. 3.



Obr. 3 Návrh spracovania forenzných artefaktov v operačnom systéme Windows

Nástroj na vstupe pracuje s obrazom disku v pôvodnom (raw) formáte. Nástroj následne pomocou rôznych nástrojov (napr. nástroj KAPE) extrahuje súbory, v ktorých

sú obsiahnuté forenzné artefakty. Nástroj KAPE [3] sa používa na vyhľadávanie a kopírovanie súborov zo zdrojového umiestnenia. V prípade súborov, ktoré sú uzamknuté operačným systémom, druhý beh obchádza uzamknutie. Na konci procesu KAPE vytvorí kópiu a zachová metadáta o všetkých dostupných súboroch zo zdrojového umiestnenia do daného adresára [18]. Po zaistení metadát sa dostávame do fázy ich spracovania do rozumnej podoby tabuľkovo štruktúrovaných súborov CSV. Na to využijeme niekoľko forenzných nástrojov od Erica Zimmermana[4], ktorých úlohou je najmä vytiahnutie informácií z forenzných artefaktov. Na to máme vytvorený skript v programovacom jazyku PowerShell, pomocou ktorého vieme vyhotoviť toto spracovanie automatizovane. Takto spracované dáta môžeme začať analyzovať. Na analýzu použijeme rozhranie Jupyter Notebook[5] a programovací jazyk Python vo verzii 3, ktoré sú považované za vhodné nástroje k dátovej analýze. Nakoniec sa vytvorí sumárny report obsahujúci štatistické údaje o riešenom prípade, resp. indikátory poukazujúce na spôsob kompromitácie systému.

Pri analýze si vyberáme atribúty, ktoré pre nás majú najvyššiu forenznú hodnotu. Následne dáta filtrujeme, agregujeme a spájaním rôznych dátových rámcov nachádzame súvis medzi jednotlivými artefaktami a ich atribútmi. Výstupom tohto nástroja štatistické informácie o vstupných dátach.

3.3 Použité technológie

Pre prácu s forenznými artefaktmi je vhodné prostredie Jupyter Notebook. Je to interaktívny webový nástroj, ktorý umožňuje miešať text (Markdown) a kód (viacero programovacích jazykov) vo formáte bunka po bunke. Jupyter Notebook sa nedávno vyvinul do flexibilnejšieho prostredia s názvom JupyterLab [5].

Okrem spustenia staršieho Jupyter Notebooku, JupyerLab obsahuje možnosť otvárať editory kódu, konzoly, shell terminály operačného systému, prehliadač súborov

[3] Dostupné na: https://www.kroll.com/en/insights/publications/cyber/kroll-artifact-parser-extractor-kape

[4] Dostupné na: https://ericzimmerman.github.io/#!index.md

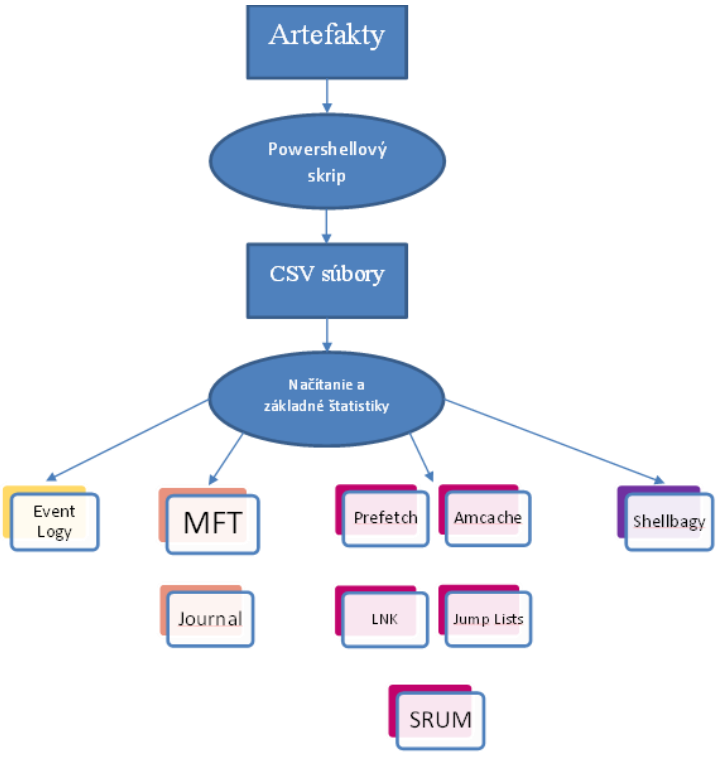
[5] Dostupné na: https://jupyter.org/

a viacero panelov pre konfiguráciu a spravovanie. Z tohto dôvodu je ideálnym nástrojom pre dátovú analýzu a spracovanie dát [5].

Programovací jazyk, ktorý som si vybrala na prácu s Jupyter Notebookom je Python verzie 3. Python verzie 3 je voľne dostupný, interpretovaný, vysokoúrovňový jazyk a poskytuje skvelý prístup pre objektovo orientované programovanie. Je to jeden z najlepších jazykov používaných dátovými vedcami pre rôzne projekty/aplikácie dátovej vedy. Python poskytuje skvelé funkcie na prácu s matematikou, štatistikou a vedeckými funkciami. Poskytuje skvelé knižnice, ktoré sa zaoberajú vedou o údajoch a jej aplikáciou. Najviac používané knižnice v dátovej vede sú Numpy, Pandas, Matplotlib a ďalšie [6].

3.4 Implementácia riešenia

Pre implementáciu nástroja sme sa rozhodli použiť programovací jazyk Powershell, ktorým vieme jednoducho automatizovať spracovanie vybraných forenzných artefakty do CSV súborov. Druhý použitý jazyk je Python vo verzii 3, keďže je to jeden z najlepších jazykov používaný na dátovú analýzu. Pracujeme vo webovom prostredí Jupyter Notebook, ktorý sme podrobnejšie už opísali v predchádzajúcich kapitolách. Na Obr. č. 4 je znázornený graf nášho nástroje.



Obr. 4 Diagram navrhovaného nástroja

3.4.1 Spracovanie forenzných artefaktov

Cieľom tejto časti je spracovať vybrané forenzné artefakty do CSV súborov, ktoré sa následne vložia vo forme dátových rámcov.

Keďže väčšina artefaktov je uložená v binárnom formáte, potrebujeme ich dostať do podoby, s ktorou v našom vybranom prostredí a programovacom jazyku vieme pracovať. Využili sme na to parsovacie nástroje od Erica Zimmemana. Tieto nástroje sa spravidla sa používajú na to, aby sa existujúce, často neštruktúrované a nečitateľné údaje, stali zrozumiteľnejšími.

3.4.2 Nástroje Erica Zimmermana

Nástroje od Erica Zimmerman [6] predstavujú nástroje na čítanie forenzných artefaktov využitím príkazového riadku. Majú štandardizovaný výstup vo forme CSV, XML a JSON súborov. V našom prípade používame CSV formát. Práca s nimi je v celku jednoduchá. Štandardne tieto nástroje využívajú argument -f na spracovanie len jedného súboru a argument -d na spracovanie celého priečinku. Je potrebné poznamenať, že argument -d nie je dostupný pre všetky nástroje. Po použití jedného z týchto argumentov nasleduje cesta k artefaktu alebo priečinku, kde sú artefakty, s ktorým chceme pracovať, uložené. Po upresnení zdrojových súborov nasleduje prepínač –csv, ktorým upresňujeme to, že nami žiadaný výstup je vo formáte CSV. Za týmto parametrom zadávame cestu k priečinku, kde chceme uložiť výstup. Ako posledný zadávame prepínač –csvf nasledovaný názovom výstupného súboru a prípona .csv. Tento posledný prepínač nie je povinný. Pri jeho vynechaní nástroj súboru priradí automaticky vygenerovaný názov. V niektorých prípadoch nie je dokonca vôbec dostupný. Ukážka použitia nástroja na logoch udalostí:

```
EvtxECmd.exe -f <cesta k súboru> --csv <cesta k výstupnému priečinku> --csvf
<názov výstupného súboru>

EvtxECmd.exe -d <cesta k priečinku> --csv <cesta k výstupnému priečinku> --
csvf <názov výstupného súboru>
```

[6] Dostupné na: https://github.com/EricZimmerman/

Špecifický je nástroj **MFTECmd**[7], ktorý spracováva súbory $MFT a $J záznamy. Tu máme možnosť spojiť tieto dva súbory a to tak, že prázdny atribút „ParentPath“ vo výstupe pre súbor $J bude doplnený o úplnú cesta k nadradenému adresáru. Môžem to spraviť tak, že za argument -f určíme cestu k súboru $J a následne použijeme argument -m na zadefinovanie cesty k súboru $MFT.

Na využívanie týchto nástrojov sme si pripravili skript v jazyku Powershell, ktorý si po spustení vyžiada cestu k adresáru s týmito nástrojmi. Teraz si vieme vybrať, či chceme použiť všetky alebo iba niektoré z nástrojov. Ak sme si vybrali, že chceme použiť iba niektoré, vymenuje iba tie zo zadaného priečinka, ktoré vie sám aj spúšťať a požiada nás, aby sme napísali ktoré chceme použiť a oddelili ich čiarkou. Potom postupne pre každý nástroj vyžiada cestu k súborom, ktoré chceme spracovať a miesto, kam chceme uložiť výstup. Názvy súborov sú preddefinované v skripte pre ich jednoduchšie načítavanie do Jupyter notebookov.

3.5 Načítanie súborov do Jupyter Notebooka

Pre každý forenzný artefakt sme si pripravili osobitný Jupyter Notebook, v ktorom budeme pracovať s našimi súbormi vo formáte CSV. Na začiatok sa tieto súbory previedli do dátových rámcov použitím knižnice Pandas[8]. Po prevedení si pre každý forenzný artefakt zobrazíme základnú štatistiku v podobe počtu záznamov za celé obdobie a za jednotlivé dni osobitne. Vypíšeme počet názvy jednotlivých atribútov označujúcich stĺpce datafame-ov. Príklad prevedenia CSV výstupov zo záznamov udalostí:

```
evt = pd.read_csv("EventLogs_DC_Output.csv")

evt_Desktop = pd.read_csv("EventLogs_Desktop_Output.csv")
```

3.6 Práca so záznamami udalostí (event log)

Forenznému artefaktu EventLog sme sa bližšie venovali v kapitole 1.3. Prvým krokom v rámci analýzy je zobrazenie základný štatistických informácií týkajúcich sa

[7] Dostupné na: https://github.com/EricZimmerman/MFTECmd

[8]Dostupné na: https://pandas.pydata.org/

týchto záznamov. Zaujímajú nás počet rôznych ID udalostí, ktoré sa udiali v systéme a počet ich výskytov za celé obdobie. Tu môžeme sledovať nezvyčajný nárast počtu udalostí konkrétneho typu. Taktiež nás zaujímajú typy užívateľov. Následne si zobrazíme rozdelenie počtu udalostí v čase. To nám pomáha pri identifikácii prípadných skokov alebo medzier na časovej osi jednotlivých typov udalostí.

	index	EventIdAll	CountAll	index	EventIdZaObdobie	CountZaObdobie	index	EventIdZa24h	CountZa24h
0	453	36874	29205	187.0	4624.0	1518.0	313.0	36888.0	29205.0
1	454	36888	29205	194.0	4672.0	1388.0	312.0	36874.0	29205.0
2	291	4624	2540	189.0	4634.0	1359.0	101.0	261.0	1938.0
3	298	4672	2351	257.0	7036.0	546.0	207.0	4624.0	1364.0
4	293	4634	2245	231.0	5320.0	489.0	96.0	217.0	1355.0
5	125	261	1938	245.0	5858.0	405.0	209.0	4634.0	1199.0
6	116	217	1356	167.0	4017.0	158.0	213.0	4672.0	1189.0
7	382	7036	1176	215.0	5017.0	158.0	274.0	7036.0	555.0
8	344	5320	831	307.0	62171.0	126.0	246.0	5320.0	405.0
9	359	5858	660	306.0	62170.0	126.0	60.0	102.0	398.0

Obr. 5 Top 10 udalostí za časové obdobia

3.6.1 Top 10 udalostí za časové obdobia

V tejto časti si vyfiltrujeme top 10 udalostí za jednotlivé časové obdobia. Predvolené je celé obdobie a posledných 24 hodín, za ktoré máme dostupné dáta. Ďalej si pomocou parametrov fromDate a toDate vieme nastaviť nejaké konkrétne obdobie, ktoré nás v predchádzajúcom výstupe zaujalo. Na výstupe (Obr. č. 5) dostávame tabuľku obsahujúcu všetky tri filtrovania vedľa seba.

Príklad agregácie dát podľa množstva jednotlivých typov udalostí:

```
evt[(evt['TimeCreated'] > fromDate) & (evt['TimeCreated'] <
toDate)].groupby(["EventId"]).agg({'EventId': ['size']}).reset_index()
```

Typicky najfrekventovanejšími udalosťami sú udalosti s ID 4624, 4672 a 4634. Tie nám hovoria o tom, či sa používateľ prihlásil a odhlásil z operačného systému úspešne. V prípade, ktorý analyzujeme, je možné vidieť, že najväčší počet záznamov udalostí je s ID 36888 a 36874. Dokonca sme zistili, že tieto udalosti nastali len

v posledný deň záznamov. Ide o netradičnú aktivitu, ktorá môže naznačovať pokus o pripojenie sa útočníka na server.

3.6.2 Prvý a posledný čas udalosti

Zaujímavou informáciou môže byť prvý a posledný čas, kedy sa nejaká udalosť udiala. Čas trvania medzi prvou a poslednou udalosťou môže naznačovať dĺžku činnosti alebo bezpečnostného incidentu. Tieto informácie môžu byť užitočné na pochopenie rozsahu a vplyvu bezpečnostného incidentu a na posúdenie rozsahu akejkoľvek škodlivej činnosti. Môžu tiež pomôcť identifikovať vzory alebo nezrovnalosti dátach. Neobvyklé medzery, opakované akcie v krátkom časovom úseku alebo udalosti, ktoré sa vyskytujú mimo bežných prevádzkových hodín, môžu vzbudiť podozrenie a podnietiť ďalšie vyšetrovanie. Vytvorili sme si zoznam všetkých udalostí s ich popisom a už spomínaný prvým a posledným časom diania (viď Obr. č. 6).

	EventId	MapDescription	MinTimeCreated	MaxTimeCreated
0	0	None	2020-09-19 03:56:05.4013073	2020-09-19 04:36:05.1406058
1	1	The system time was changed	2020-09-17 15:51:29.8610578	2020-09-19 04:46:05.3441145
2	2	An account was logged on	2020-09-17 15:51:31.6576930	2020-09-19 04:47:05.3312711
3	3	None	2020-09-17 15:51:31.3609968	2020-09-19 03:57:40.7451445
4	4	An account was logged off	2020-09-17 15:51:30.9231956	2020-09-19 04:36:42.9209714
...	...	...	...	...
459	50041	None	2020-09-17 17:55:52.1791005	2020-09-18 22:28:18.3884534
460	51046	None	2020-09-17 15:52:01.4630897	2020-09-19 01:22:41.1926968
461	51047	None	2020-09-17 15:52:22.0729473	2020-09-18 23:10:48.6747401
462	62170	None	2020-09-17 16:46:19.8850510	2020-09-19 04:36:04.1092987
463	62171	None	2020-09-17 16:46:19.9000030	2020-09-19 04:36:04.1092987

Obr. 6 Top 10 udalostí za časové obdobia

Príklad agregácie dát podľa prvého a posledného času diania udalosti:

```
evt.groupby('EventId').agg({ 'MapDescription': 'first','TimeCreated': ['min',
'max']})
```

3.6.3 Udalosti typu logon

V tejto časti sme sa zamerali na udalosti typu logon. Cieľom je zistiť, koľko krát došlo k rôznym typom prihlásení v systéme. Pred začiatkom práce je však nutné skontrolovať názvy stĺpcov, v ktorých sa nachádzajú jednotlivé parametre logov. Typy prihlásenia poskytujú informácie o metóde overovania použitej počas procesu prihlásenia. Tieto informácie môžu zahŕňať interaktívne prihlásenia pomocou hesla, prihlásenia pomocou čipovej karty, anonymné prihlásenia atď. Zisťovanie typov prihlásenia, ktoré sa odchyľujú od normy alebo naznačujú neobvyklé metódy overovania, môže pomôcť identifikovať potenciálne narušenia bezpečnosti alebo kompromitované účty. Na výstupe (Obr. č. 7) dostávame tabuľku obsahujúcu počty jednotlivých typov prihlásení za celé obdobie, za nami zvolené obdobie, ktoré sme si zvolili na začiatku a za posledných 24 hodín v nami dostupných dátach.

		CountAll	CountObdobie	Count24h
EventId	PayloadData2			
4624	LogonType 0	11	5.0	5.0
	LogonType 10	4	0.0	4.0
	LogonType 2	43	18.0	26.0
	LogonType 3	2298	1401.0	1238.0
	LogonType 5	176	86.0	89.0
	LogonType 7	8	8.0	2.0
4625	LogonType 3	95	0.0	95.0
	LogonType 7	1	1.0	0.0
4634	LogonType 10	3	0.0	3.0
	LogonType 2	10	2.0	10.0
	LogonType 3	2224	1349.0	1184.0
	LogonType 7	8	8.0	2.0

Obr. 7 Ukážka počtu prihlásení jednotlivých typov

Na Obr. č. 5 môžeme vidieť pomerne vysoký počet neúspešných prihlásení za posledný deň, čo znova môže naznačovať útočníkove pokusy o prihlásenie sa na server. Príklad filtrovania dát podľa počtu typov prihlásenia:

```
dfAll = evt[evt['PayloadData2'].str.contains('^LogonType')==True]

dfoutAll                                                                =
dfAll.groupby(['EventId','PayloadData2']).size().to_frame('CountAll')
```

3.6.4 Udalosti podľa MITRE ATT&CK

MITRE ATT&CK je celosvetovo dostupná databáza poznatkov o technikách útočníka a taktikách založených na pozorovaniach z reálneho sveta. Znalostná databáza ATT&CK sa používa ako základ pre vývoj špecifických modelov hrozieb a metodík v súkromnom sektore, vo vláde a v komunite produktov a služieb informačnej a kybernetickej bezpečnosti [48].

Z jednotlivými taktikami sa môžu spájať rôzne druhy udalostí, ktorých prítomnosť v systéme síce nemusí nevyhnutne znamenať že daná taktika bola použitá útočníkom na tomto zariadení ale môžu analytika nasmerovať správnou cestou pri analýze. V rámci práce sme spracovali niekoľko taktík, ktorých identifikácia je užitočná pri forenznej analýze. Na Obr. č. 8 môžeme vidieť výber niekoľkých taktík s uvedeným konkrétnych typov udalostí.

```
TA0001 Initial access
4625 Login denied due to account policy restrictions
33205 Login failure from a single source with a disabled account
TA0002 Execution
5145 Remote schedule task creation via named pipes
4698 Schedule task created with suspicious arguments
4688 Scheduled task creation
TA0008 Lateral Movement
4825 Denied RDP login with valid credentials
5140 Admin share accessed via SMB (basic)
5145 Impacket WMIexec execution via SMB admin share
TA0003 Persistence
4720 Local user account created on a single host
4726 Hidden account creation (with fast deletion)
```

Obr. 8 Zoznam vybraných taktík podľa MITRE ATT&CK

Pomocou tohto zoznamu sme následne zistili počet výskytov Id udalostí spojených s týmito taktikami a čas prvého a posledného výskytu (viď. Obr. č. 9). Sú definované štyri zoznamy: tactics_ids, tactics_names, tactics_events_ids a tactics_events_names. Tieto zoznamy uchovávajú informácie týkajúce sa taktík, ich názvov, ID udalostí a názvov udalostí. Potom pomocou cyklu a funkcie "zip()" iterujeme cez tieto štyri zoznamy súčasne. To umožňuje spracovať zodpovedajúce prvky každého zoznamu spoločne. Tu extrahujeme ID taktiky a názov taktiky. V ďalšom cykle vypíšeme vypíše ID udalosti a názov udalosti. Počet udalostí s konkrétnym ID udalosti sa vypočíta filtrovaním DataFrame evt na základe ID udalosti a použitím atribútu ".shape[0]" na získanie počtu

riadkov. Minimálna a maximálna hodnota stĺpca "TimeCreated" vo filtrovanom DataFrame sa určí tak, aby predstavovala prvý výskyt a posledný výskyt udalosti. Získané informácie vrátane ID taktiky, názvu taktiky, ID udalosti, názvu udalosti, počtu, minimálneho času vytvorenia a maximálneho času vytvorenia sa priradia k príslušným stĺpcom v DataFrame mitre_evt. Ukážka vzorového kódu:

```
tactics_ids = ["TA0001","TA0002","TA0008","TA0003"]

tactics_names = ["Initial access","Execution","LateralMovement", ...]

tactics_events_ids = [[4625,33205],[5145,4698,4688],[4825,5140,5145], ...]

tactics_events_names = [["Login denied due to account policy restrictions, ..]]

for tactic_id, tactic_name, event_ids, event_names in zip(tactics_ids,tactics_names,tactics_events_ids,tactics_events_names):
        print(tactic_id,tactic_name)
        for id, name in zip(event_ids, event_names):
            print(id,name)
            count = evt[evt["EventId"].isin([id])].shape[0]
            MinTimeCreated =evt[evt["EventId"].isin([id])].TimeCreated.min()
            MaxTimeCreated =evt[evt["EventId"].isin([id])].TimeCreated.max()
            mitre_evt.loc[i,"TacticID"] = tactic_id
            mitre_evt.loc[i,"TacticName"] = tactic_name
            mitre_evt.loc[i,"Description"] = name
            mitre_evt.loc[i,"EventID"] = id
            mitre_evt.loc[i,"Count"] = count
            mitre_evt.loc[i,"MinTimeCreated"] = MinTimeCreated
            mitre_evt.loc[i,"MaxTimeCreated"] = MaxTimeCreated
            i +=1
```

	TacticID	TacticName	Description	EventID	Count	MinTimeCreated	MaxTimeCreated
0	TA0001	Initial access	Login denied due to account policy restrictions	4625.0	107.0	2020-09-17 15:52:01.0000000	2020-09-19 03:21:46.5653915
1	TA0001	Initial access	Login failure from a single source with a disa...	33205.0	0.0	NaN	NaN
2	TA0002	Execution	Remote schedule task creation via named pipes	5145.0	0.0	NaN	NaN
3	TA0002	Execution	Schedule task created with suspicious arguments	4698.0	0.0	NaN	NaN
4	TA0002	Execution	Scheduled task creation	4688.0	0.0	NaN	NaN
5	TA0008	Lateral Movement	Denied RDP login with valid credentials	4825.0	0.0	NaN	NaN
6	TA0008	Lateral Movement	Admin share accessed via SMB (basic)	5140.0	0.0	NaN	NaN
7	TA0008	Lateral Movement	Impacket WMIexec execution via SMB admin share	5145.0	0.0	NaN	NaN
8	TA0003	Persistence	Local user account created on a single host	4720.0	16.0	2020-09-17 17:57:13.1766557	2020-09-18 04:39:13.2910106
9	TA0003	Persistence	Hidden account creation (with fast deletion)	4726.0	9.0	2020-09-18 01:05:16.2656295	2020-09-18 04:34:14.0249863

Obr. 9 Ukážka tabuľky výskytov udalostí podla MITRE ATT&CK

3.7 Práca s Prefetch záznamami

Tieto záznamy môžeme získať iba z koncových zariadenia, keďže štandardne sa na serveroch prefetching nevykonáva. Na začiatok si znova vygenerujeme nejakú základnú štatistiku a to tak, že sčítame všetky unikátne hodnoty v jednotlivých stĺpcoch dátového rámca. To nám ukáže koľko rôznych programov sa v dátovom rámci nachádza a koľko z nich bolo spúšťaných v odlišné časy. Príklad kódu, ktorý nám zabezpečí túto činnosť: `for index in df.columns: print(index,": ",prefetch[index].nunique())`

3.7.1 Posledné spustenia súborov

Keďže niektoré súbory môžu byť spustené z rôznych miest, môžeme mať viacero prefetch záznamov pre ten istý súbor. Ak teda chceme vidieť posledné spustenie konkrétneho súboru, musíme prejsť všetky záznamy prislúchajúce jemu. Vytvorili sme si tabuľku (viď Obr. č. 10), kde sú zobrazené posledné spustenia všetkých súborov a zoradili sme si ich vzostupne a zostupne. Tu môžeme vidieť, ktoré súbory neboli spúšťané nejakú dobu a ktoré boli spustené ako posledné. Na vytvorenie tabuľky sme zoskupili súbory podla ich názvu. Potom sme na stĺpec "LastRun" v rámci každej skupiny použili agregačná funkcia. Agregačná funkcia, ktorá sa tu používa, je "max" a tá vypočíta maximálnu hodnotu stĺpca "LastRun" pre každú skupinu. Stĺpec "LastRun" obsahuje časové pečiatky ktoré možno navzájom porovnávať. Ukážka vzorového kódu:

```
Last_Run = prefetch.groupby('ExecutableName').agg({'LastRun':
        ['max']}).reset_index()
```

	index	ExecutableName	LastRunDescending	index	ExecutableNameAscending	LastRunAscending
0	2	AUDIODG.EXE	2020-09-19 05:18:45	28	GENVALOBJ.EXE	2020-09-18 04:58:07
1	76	SVCHOST.EXE	2020-09-19 05:18:23	16	DASHOST.EXE	2020-09-18 04:58:09
2	55	RUNTIMEBROKER.EXE	2020-09-19 05:16:47	69	SLUI.EXE	2020-09-18 04:58:16
3	79	TASKHOSTW.EXE	2020-09-19 05:13:16	33	LPREMOVE.EXE	2020-09-18 05:47:54
4	74	SPPSVC.EXE	2020-09-19 05:12:53	50	OOBENETWORKCONNECTIONFLOW.EXE	2020-09-18 05:47:55
5	4	BACKGROUNDTRANSFERHOST.EXE	2020-09-19 05:10:45	86	USEROOBEBROKER.EXE	2020-09-18 05:49:09
6	61	SECURITYHEALTHHOST.EXE	2020-09-19 05:10:23	24	FODHELPER.EXE	2020-09-18 05:49:10
7	27	FTK IMAGER.EXE	2020-09-19 05:09:56	96	WINDOWS.WARP.JITSERVICE.EXE	2020-09-18 05:51:12
8	11	CONSENT.EXE	2020-09-19 05:09:54	26	FSQUIRT.EXE	2020-09-18 05:51:22
9	18	DLLHOST.EXE	2020-09-19 05:09:52	73	SPPEXTCOMOBJ.EXE	2020-09-18 05:51:26

Obr. 10 Ukážka posledných spustení súborov

3.7.2 Súbory s jedným spustením

Pri skúmaní systému z hľadiska možného narušenia bezpečnosti alebo infikovania škodlivým softvérom môžu súbory, ktoré boli spustené iba raz, naznačovať prítomnosť podozrivého alebo škodlivého softvéru. Škodlivý softvér často používa techniky, ako sú útoky bez súborov alebo samočinné mazanie, aby sa vyhol odhaleniu. Identifikácia súborov, ktoré boli spustené iba raz, môže vyšetrovateľom pomôcť zamerať sa pri analýze na potenciálne škodlivé súbory, ktoré mohli byť zapojené do útoku. Preto sme si vytvorili tabuľku (viď Obr. č. 11) pomocou knižnice Bokeh [9], v ktorej vieme interaktívne zoraďovať tieto súbory na základe ich názvu, hešu, veľkosti a času spustenia.

#	ExecutableName	Hash	Size	*LastRun*
24	FTK IMAGER.EX	E67CC266	113012	2020-09-19 05:
11	DLLHOST.EXE	5DC108BA	40516	2020-09-19 05:
30	MSINFO32.EXE	C3C668DA	41432	2020-09-19 05:
48	SNIPPINGTOOL.	B23F9DB3	47730	2020-09-19 05:
45	SC.EXE	6C4D4413	5526	2020-09-19 03:
7	COREUPDATER.	157C54BB	24316	2020-09-19 03:
5	CHXSMARTSCR	D6E6DEB7	117330	2020-09-19 03:
44	RUNTIMEBROKE	D14F4AE9	16338	2020-09-19 03:
46	SECHEALTHUI.E	290FDD39	112568	2020-09-19 03:
34	ONEDRIVE.EXE	5361D4E	182986	2020-09-19 03:

Obr. 11 tabuľky so súbormi spustenými 1 krát

[9] Dostupné na: https://docs.bokeh.org/en/3.0.2/index.html

Na vytvorenie interaktívnej tabuľky pomocou knižnice Bokeh si potrebujeme zadefinovať zdroj údajov, ktorý je v tomto prípade dataframe s názvom "part_prefetch" a definuje stĺpce, ktoré sa majú v tabuľke zobraziť (vrátane polí ako "ExecutableName", "Hash", "Size" a "LastRun"). Na koniec použijeme funkciu "show()" na zobrazenie dátovej tabuľky. Ukážka vzorového kódu:

```
part_prefetch = prefetch[prefetch["RunCount"] == 1]

part_prefetch = part_prefetch[["ExecutableName","Hash","Size","LastRun"]]

source = ColumnDataSource(part_prefetch)

columns = [#TableColumn(field="SourceCreated", title="SourceCreated",

       formatter=DateFormatter())

       TableColumn(field="ExecutableName", title="ExecutableName"),

       TableColumn(field="Hash", title="Hash"),

       TableColumn(field="Size", title="Size"),

       TableColumn(field="LastRun", title="LastRun")]

data_table = DataTable(source=source, columns=columns, width=400, height=280)

show(data_table)
```

3.8 Práca s MFT a Journal záznamami

Forenzným artefaktom MFT a Journal sme sa bližšie venovali v kapitolách 1.4.1 a 1.4.2. Analýza týchto forenzných artefaktov začína zo základnými štatistickými informáciami. Pre tento účel sme si znovu vybrali doménový kontrolér a artefakt Journal, ktorý zaznamenáva všetky zmeny v súboroch od ich posledného zálohovania. Dátový rámce, v ktorom sme si z neho uložili údaje, sa nazýva JaMFT. Ten je obohatený o cestu k rodičovskému súboru z tabuľky MFT. Jedna z informácií, ktorú tu môžeme získať, je počet zmien v súboroch od posledného zálohovania za jednotlivá dni. Umožňuje nám to vidieť nadmernú manipuláciu so súbormi na časovej osi.

3.8.1 Identifikácia prípon súborov

Journal ponúka aj atribút s názvom „Extention“, ktorý obsahuje prípony súborov, na ktorých došlo k nejakej zmene, či už ide o modifikáciu, vymazanie alebo pridanie súboru. Frekvencia častého výskytu konkrétnej prípony súboru medzi spustenými súbormi môže potenciálne naznačovať prítomnosť ransomvéru. Z toho dôvodu sme vypočítali sme koľko krát došlo k zmene v súboroch s jednotlivými príponami a zobrazili vzostupne top 10 prípon s najvyšším počtom výskytov za celé obdobie, za obdobie vybrané užívateľom a posledných 24 hodín. Klasicky medzi najpočetnejšie prípony za celé obdobie súborov patrí .log, ktorý nasleduje .dll, .tmp a .dat (viď Obr. č. 12).

	index	Extension	CountAll	index	ExtensionZaObdobie	CountZaObdobie	index	ExtensionZa24h	CountZa24h
0	392	.log	18001	40.0	.dat	1413.0	37.0	.dat	1277.0
1	352	.dll	9117	29.0	.XML	1088.0	26.0	.XML	864.0
2	431	.tmp	6440	62.0	.log	735.0	62.0	.log	753.0
3	346	.dat	5381	51.0	.evtx	674.0	47.0	.evtx	651.0
4	323	.aux	2552	27.0	.TMP	634.0	24.0	.TMP	520.0
5	426	.settingcontent-ms	2241	50.0	.etl	324.0	74.0	.tmp	270.0
6	444	.xml	1985	35.0	.chk	241.0	32.0	.chk	262.0
7	401	.mui	1800	23.0	.LOG1	234.0	46.0	.etl	243.0
8	320	.XML	1791	77.0	.tmp	227.0	20.0	.LOG1	242.0
9	363	.exe	1789	65.0	.newcfg	189.0	64.0	.newcfg	216.0

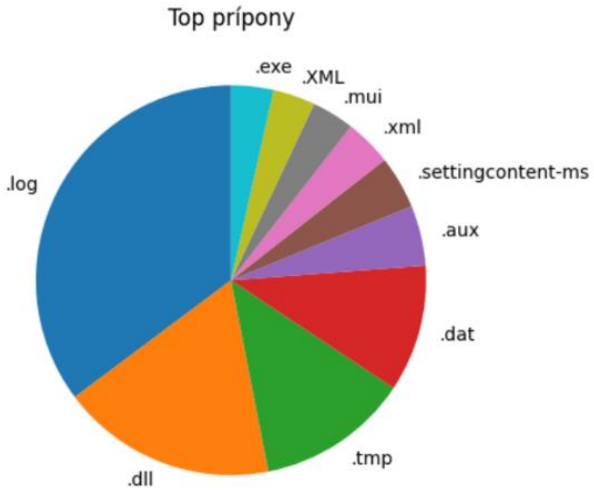
Obr. 12 Ukážka najviac sa vyskytujúcich súborových prípon za určité obdobia

Na vytvorenie tabuľky prípon sme zoskupili záznamy podľa ich prípon uložených v stĺpci "Extension" . Potom sme na tento stĺpec v rámci každej skupiny použili agregačnú funkciu "size" ktorá spočíta ich výskyt. Z tohto výsledku sme následne vytvorili koláčový graf obsahujúci top 10 prípon, ktorý môžeme vidieť na Obr. č. 13. Na vytvorenie tohto grafu sme použili knižnicu matplotlib[10]. Táto knižnica je jednoduchá na použitie a ponúka mnoho možností si ju prispôsobiť. Na vytvorenie grafu sme upravili agregovaný dataframe a zobrazili len top 10 prípon pomocou funkcie "head()”. Ukážka vzorového kódu:

```
extentions_all = df_merge.groupby([„Extension“]).agg({‚Extension‘:
```

[10] Dostupné na: https://matplotlib.org/3.5.3/api/_as_gen/matplotlib.pyplot.html

```
      [,size']}).reset_index()
extentions = extentions_all.head(10)
plt.pie(extentions["CountAll"], labels = extentions.Extension, startangle = 90)
plt.title("Top prípony")
plt.show()
```



Obr. 13 Ukážka grafu najviac sa vyskytujúcich súborových prípon za celé obdobie

3.8.2 Manipulácia s časovými pečiatkami

V rámci analýzy pokračujeme s MFT záznamami. Aby sme mali dáta úplné spojili sme dataframe-y MFT a JaMFT do jedného pomocou ľavého join-u. Parametre, podľa ktorých ich spájame, sú „EntryNumber“ a „SequenceNumber“. Ich kombináciou vieme presne určiť, o ktorý súbor ide.

Jednou z antiforenzných techník je takzvaný „timestomping“. Ide o techniku, pri ktorej útočník mení časovú pečiatku súboru [29]. To vieme detegovať v MFT záznamoch pomocou časových pečiatok atribútov $STANDARD_INFORMATION (SI) a $FILE_NAME (FN). Ak zistíme, že časová pečiatka atribútu SI je menšia (skorší dátum) ako pečiatka atribútu FI, mohlo dôjsť k manipulácii s nimi. Ďalší nám dostupný parameter je „uSecZeros“. Ak je jeho hodnota True, znamená to, že niektorá z časových pečiatok má vynulované subsekundy. Tieto dva indikátory ale nie sú absolútnym dôkazom manipulovania s pečiatkami. V našom prípade sme si vyfiltrovali také záznamy, ktoré spĺňajú tieto podmienky. Pre tento výstup sme si vygenerovali interaktívnu tabuľku

v ktorej je možné filtrovať dáta podľa dátumu (viď Obr. č. 14). Ukážka vzorového kódu spájania dataframe-ov a získania záznamov v ktorých mohlo dôjsť k manipulácii s časovými pečiatkami:

```
df_merge = pd.merge(left=MFT, right=JaMFT,how='left',
      left_on=['EntryNumber','SequenceNumber'],
            right_on=['ParentEntryNumber','SequenceNumber'])
df_merge[(df_merge["SI<FN"].isin([True])& df_merge['uSecZeros'].isin([True]))]
```

21 Aug 2013 23:36:31 .. 18 Sep 2020 23:33:54

#	EntryNumber	SequenceNu	FileName	Created0x10	FileSize
0	897	271	vmrawdsk.inf	2020-03-30 2	2946
1	1400	15	concrt140.dll	2019-03-09 0	333056
2	5848	1	amd64_micr	2014-03-21 1	0
3	5935	1	amd64_micr	2014-03-21 1	0
4	11604	1	amd64_micr	2013-08-22 1	0
5	11612	1	amd64_micr	2014-03-21 1	0
6	15937	1	wow64_micr	2013-08-22 1	0
7	16770	1	x86_microso	2013-08-22 1	0
8	16992	1	x86_microso	2014-03-21 1	0
9	16992	1	x86_microso	2014-03-21 1	0

Obr. 14 Ukážka záznamov, v ktorých mohlo dôjsť k manipulácii s časovými pečiatkami

V rámci tejto časti sme si ešte naviac vybrali tie záznamy, ktoré majú čas zmeny časovej pečiatky v denníku (Journál). Výstupom boli 4 súbory (viď Obr. č. 15).

#	Name	EntryNumber	SequenceNumber	SI	UpdateTimestamp	UpdateReasons
0	dnscmd.exe.mui	5848	1	true	2020-09-17 17:51:03.	FileDelete|Close
1	dsrolesrv.dll.mui	5935	1	true	2020-09-17 17:50:58.	FileDelete|Close
2	GPRSoP.dll.mui	16992	1	true	2020-09-17 17:51:19.	FileDelete|Close
3	gpmgmt.dll.mui	16992	1	true	2020-09-17 17:51:20.	FileDelete|Close

Obr. 15 Ukážka upravenej tabuľky z Obr. č. 1

3.9 Práca so SRUM

Forenznému artefaktu System Resource Usage Monitor (SRUM) sme sa bližšie venovali v kapitole 1.5.5. Rovnako ako u predchádzajúcich forenzných artefaktoch, začíname so základnou štatistikou. Okrem počtov rôznych záznamov vieme zobraziť aj počet rôznych súborov zaznamenaných v tomto forenznom artefakte. Podstatný je aj údaj o rôznych typoch užívateľov, ktorý sa v systéme nachádzajú.

3.9.1 Spúšťanie súborov rôznymi užívateľmi

V databáze SRUM existujú rôzne typy používateľov na základe ich identifikátorov zabezpečenia (SID) alebo typov používateľských účtov. Údaje SRUM môžu poskytnúť prehľad o činnostiach týchto používateľov. Rôzny používatelia môžu spúšťať rôzne spustiteľné súbory na základe svojich rolí a oprávnení. Identifikácia rôznych typov používateľov a s nimi spojených spustiteľných súborov môže byť cenná pre pochopenie správania sa používateľov a potenciálnych bezpečnostných incidentov.

Údaj, ktorý sa snažíme získať, je počet rôznych typov užívateľov, ktorí spúšťali jednotlivé súbory a počet súborov spustených každým typom užívateľa (viď. Obr. č. 16 a Obr. č. 17).

SidType	CountFiles
Administrator	62
LocalService	7
LocalSystem	84
NetworkService	12
UnknownOrUserSid	117

Obr. 16 Počet typov užívateľov, ktorí spustili programy

#	ExecutableName	Count
1	AM_Delta.exe	1
3	AM_Engine.exe	1
4	AppHostRegistrationVerifier.exe	1
5	ApplicationFrameHost.exe	2
124	audiodg.exe	1
125	browser_broker.exe	2
6	ByteCodeGenerator.exe	1
7	CloudExperienceHostBroker.exe	1
126	cmd.exe	3
8	CompatTelRunner.exe	1

Obr. 17 Počet súborov spustených rôznymi používateľmi

Zaujímajú nás hlavne dva typy používateľských účtov, a to Administrator a LocalSystem. Administrator má úplnú kontrolu a dohľad nad systémom SRUM. Tento účet je zodpovedný za konfiguráciu a správu systému, definovanie kontroly prístupu a monitorovanie využívania zdrojov v sieti. Používateľský účet LocalSystem je preddefinovaný miestny účet, ktorý používa správca riadenia služieb. Má rozsiahle oprávnenia na miestnom počítači a vystupuje ako počítač v sieti.

Z tabuľky všetkých súborov sme vybrali len také, ktoré boli spustené aspoň jedným z nich a doplnili údaj, ktorým konkrétne alebo dokonca oboma (viď Obr. č. 18).

#	ExecutableName	Count	UserType
50	RuntimeBroker.exe	2	Administrator
51	SIHClient.exe	1	LocalSystem
52	SearchFilterHost.exe	1	LocalSystem
53	SearchIndexer.exe	1	LocalSystem
54	SearchProtocolHost.e:	3	Both
55	SecurityHealthHost.ex	2	Administrator
56	SecurityHealthService	1	LocalSystem
57	SecurityHealthSystray.	2	Administrator
58	SettingSyncHost.exe	2	Administrator
59	Setup.exe	1	LocalSystem

Obr. 18 Súbory spúšťané Administrátorom, LocalSystémom alebo oboma

Na doplnenie údaju, ktorý označuje, či daný súbor spustil Administrator, LocalSystem alebo obaja, sme využili funkciu lambda, ktorá slúži na priradenie označení "Both", "Administrator", "LocalSystem" alebo prázdny reťazec každému riadku dataframe-u administrator_files na základe podmienok a hodnôt prítomných v srum DataFrame. Tu môžeme vidieť príslušný kód:

```
administrator_files.apply(lambda row: 'Both' if row.name in
      set(srum.loc[srum['SidType'] == 'Administrator', 'ExeInfo'].values) &
      set(srum.loc[srum['SidType'] == 'LocalSystem', 'ExeInfo'].values)
      else 'Administrator' if row.name in srum.loc[srum['SidType'] ==
            'Administrator', 'ExeInfo'].values
      else 'LocalSystem' if row.name in srum.loc[srum['SidType'] ==
            'LocalSystem', 'ExeInfo'].values else '', axis=1)
```

3.9.2 Informácie o spustených programoch

Ďalším krokom je prehľad o spustených programoch, najmä o prvom a poslednom spustení (viď Obr. č. 19). Tieto časy spustenia súborov môžu poskytnúť prehľad o spôsoboch ich používania. Tieto informácie môžu byť užitočné pri pochopení frekvencie a opakovanosti používania súborov. Ak súbor vykazuje nedávnu časovú pečiatku posledného spustenia, ale predtým nebol spustený, môže to znamenať potenciálne podozrivú alebo neoprávnenú činnosť. Na získanie prvého, posledného a priemerného času spustenia pre jednotlivé súbory sme ich zoskupili podľa ich názvu a následne použili agregačné funkcie "max", "min" a "mean" ktoré porovnávajú časové pečiatky ako numerické hodnoty. Ukážka vzorového kódu:

```
srum.groupby('ExeInfo')['Timestamp2'].agg(['min', 'max','mean']).reset_index()
```

	ExeInfo	min	max	mean
0	AM_Base.exe	2020-09-18 22:56:00	2020-09-18 22:56:00	2020-09-18 22:56:00.000000000
1	AM_Delta.exe	2020-09-18 22:56:00	2020-09-18 22:56:00	2020-09-18 22:56:00.000000000
2	AM_Delta_Patch_1.323.1437.0.exe	2020-09-19 01:54:00	2020-09-19 01:54:00	2020-09-19 01:54:00.000000000
3	AM_Engine.exe	2020-09-18 22:56:00	2020-09-18 22:56:00	2020-09-18 22:56:00.000000000
4	AppHostRegistrationVerifier.exe	2020-09-18 22:56:00	2020-09-18 23:10:00	2020-09-18 23:03:00.000000000

Obr. 19 Prvé, posledné a priemerné spúšťanie programov

3.10 Amcache

Forenznému artefaktu Amcache sme sa bližšie venovali v kapitole 1.5.4, kde sme opísali od kedy evidujeme tento forenzný artefakt v operačnom systéme Windows a aký má vo forenznej analýze význam. Teraz sa zameriame na prácu s ním.

3.10.1 Obohacovanie záznamov

V tejto časti sme sa zamerali na obohacovanie záznamov o informácie z nezávislej služby Hybrid Analysis[11], ktorú poskytuje spoločnosť Falcon Sandbox. Služba Hybrid analysis analyzuje jednotlivé vzorky a vyhodnocuje ich škodlivosť. Samotné obohacovanie sme uskutočnili overením hešov súborov pomocou knižnice VxAPI[12], ktorá dokáže komunikovať s touto službou. Túto knižnicu je potrebné pre tento účel konfigurovať tak, že do súboru config.py vložíme vlastný API kľúč, ktorý je dostupný po zaregistrovaní sa do služby. Musíme sa tiež uistiť, že názov servera je nastavený na domovskú stránku služby Hybrid Analysis. Na spustenie skriptu "vxapi..pi" používame metódu "os.system()". Na tomto mieste uvádzame ukážku extrakcie hešov:

```
for i in amcache["SHA1"]:
os.system("python3 VxAPI/vxapi.py search_hash " + i + " >> hash.txt")
os.system("echo "" >> hash.txt")
```

Tento skript poskytuje množstvo užitočných parametrov z ktorých sme my použili parameter "search_hash". Ten v službe Hybrid Analysis vyhľadáva všetky dodatočné informácie užitočné pri forenznej analýze. Nájdené informácie ukladáme do textového súboru s ktorým budeme neskôr pracovať.

Z textového súboru ďalej extrahujeme informácie nápomocné pri forenznom vyšetrovaní, ktoré následne dopĺňame do našich záznamov. Zaujímavými údajmi sú [47]:

- **av_detect** - rozsah viacnásobného skenovania antivírusovými systémami (0-100),

[11] Dostupné na: https://www.hybrid-analysis.com/

[12] Dostupné na: ....

- **threat_score** - heuristicky určená hodnota, ktorá vyjadruje stupeň potenciálne škodlivého správania súboru (na základe statickej, dynamickej alebo hybridnej runtime analýzy),
- **threat_level** - hodnoty úrovne ohrozenia, ktoré určia, či je súbor škodlivý,
- **verdict** - verdikt, podľa ktorého sa majú filtrovať výsledky. Môže byť "1- whitelisted" (biela listina), "2- no verdict" (bez verdiktu), "3- no specific threat" (bez konkrétnej hrozby), "4- suspicious" (podozrivé) alebo "5- malcious" (škodlivé),
- **vx_family** - názov rodiny alebo klasifikácia počítačového vírusu alebo vzorky škodlivého softvéru.

Ďalším krokom extrakcie sú Id a popis techník predstavujúcich "spôsob", akým útočník dosiahne cieľ vykonaním určitej akcie. Vďaka tomuto kroku forenzný analytik vie, aké znaky si má všímať v skúmaných forenzných artefaktoch.

Pre jednoduchú extrakciu týchto dát sme si textový súbor rozdelili do skupín. Každá skupina obsahuje dáta získané pre jeden heš. Tieto skupiny postupne v cykle načítavame vo formáte json, s ktorým vieme lepšie pracovať. Následne iterujeme nad každým atribútom jednej skupiny a vyberáme už spomínané dáta, ktoré následne alokujeme na ich prislúchajúce miesto do nového dataframe-u. Okrem toho sme použili druhý cyklus, ktorý iteruje cez kľúč "mitre_attcks" v jednotlivých skupinách. V rámci tohto cyklu sa z každého slovníka "mitre_attck" vyberú hodnoty "attck_id" a "technique" a pridajú sa do príslušných zoznamov attck_ids_all a techniques_all. Ukážka vzorového kódu:

```
for group in data_groups:

results = json.loads(group)

for result in results:

            sha1 = result['sha1']

            av_detect = result['av_detect']

            threat_score = result['threat_score']

            threat_level = result['threat_level']

            verdict = result['verdict']
```

```
                vx_family = result['vx_family']
                df_hash.loc[amcache["SHA1"]==sha1, 'av_detect'] = av_detect
                df_hash.loc[amcache["SHA1"]==sha1,      'threat_score']      =
threat_score
                df_hash.loc[amcache["SHA1"]==sha1,      'threat_level']      =
threat_level
                df_hash.loc[amcache["SHA1"]==sha1, 'verdict'] = verdict
                df_hash.loc[amcache["SHA1"]==sha1, 'vx_family'] = vx_family
        for mitre_attck in result['mitre_attcks']:
                attck_ids_all.append(mitre_attck['attck_id'])
                techniques_all.append(mitre_attck['technique'])
print('ATT&CK IDs:', attck_ids_all)
print('Techniques:', techniques_all)
```

3.10.2 Súbory, ktoré nie sú komponentom operačného systému

Amcache obsahuje údaj o tom, či daný súbor je komponentom operačného systému. Znamená to, že tento súbor je zvyčajne nainštalovaný ako súčasť operačného systému Windows a považuje sa za nevyhnutný pre jeho fungovanie. Tento údaj môže byť užitočný pri rozlišovaní medzi súbormi operačného systému a aplikáciami nainštalovanými alebo upravenými používateľom alebo potencionálnym útočníkom (viď obrázok č. 20).

	FileKeyLastWriteTimestamp	Name	Size	SHA1	av_detect
1	2020-09-19 03:40:45	coreupdater.exe	7168	fd153c66386ca93ec9993d66a84d6f0d129a3a5c	83.0

Obr. 20 Súbor, ktorý nie je komponentom operačného systému, chýba mu názov produktu a bol označený za škodlivý

3.10.3 Chýbajúci názov produktu

Parameter "ProductName" v Amcache sa vzťahuje na názov produktu spojeného so záznamom aplikácie. Pokročilí útočníci môžu použiť obfuskačné alebo antiforenzné techniky na skrytie alebo úpravu informácií uložených v databáze Amcache. To môže

zahŕňať zmenu alebo odstránenie parametra "ProductName", aby sa sťažila forenzná analýza a identifikácia nainštalovaných aplikácií (viď Obr. č. 22). Na získania záznamov, kde chýba údaj o názve produktu a súbor nie je komponentom operačného systému sme filtrovali dáta podľa stĺpcov "IsOsComponent", kde nadobúda hodnotu "false" a "ProductName", ktorý je prázdny. Ukážka vzorového kódu:

```
amcache[(amcache["IsOsComponent"] == False) &
        (amcache["ProductName"].isna())][["FileKeyLastWriteTimestamp",
        "Name","Size","SHA1","av_detect"]]
```

3.11 Prepojenie artefaktov s MFT

Artefakty ako LNK, Jumplist a Shellbag obsahujú parameter vstupného čísla a sekvenčného čísla, podľa ktorých je možné ich presne priradiť k ich prislúchajúcemu záznamu v MFT. Keďže súbory a adresáre zaznamenané v tých forenzných artefaktoch sa už nemusia nachádzať v MFT, je vhodné disponovať informáciou o tom, že sa predtým nachádzali v systéme. Po prepojení s MFT vieme vďaka prázdnym parametrom určiť, ktoré z nich boli zmazané a ktoré sa v systéme ešte nachádzajú. Ukážka kódu pre spájanie dvoch dataframe-ov artefaktov:

```
lnk['TargetMFTEntryNumber'] = lnk['TargetMFTEntryNumber'].apply(lambda x:
int(x, 16))
lnk['TargetMFTSequenceNumber'] = lnk['TargetMFTSequenceNumber'].apply(lambda
x: int(x, 16))
```

Ak boli odstránené záznamy z MFT, vo forenznom artefakte s názvom jumplist môžeme zistiť, ktorý zo súborov alebo adresárov bol potencionálne zmazaný. To môžeme zistiť aj tým, že si vyfiltrujeme iba tie záznamy, ktorých čas vytvorenia a čas modifikácie je rozdielny a ich veľkosť je nulová (viď Obr. č. 21).

	Path	CreationTime	LastModified	TargetCreated	TargetModified	FileSize
6	ms-settings:network	NaN	9/18/2020 21:41	NaN	NaN	0
16	C:\Users\mortysmith\Desktop	9/18/2020 21:42	9/19/2020 3:47	9/18/2020 22:46	9/18/2020 22:47	0
34	knownfolder:{33E28130-4E1E-4676-835A-98395C3BC...	9/18/2020 21:42	9/18/2020 23:01	9/18/2020 22:46	9/18/2020 23:01	0
35	knownfolder:{754AC886-DF64-4CBA-86B5-F7FBF4FBC...	9/18/2020 21:42	9/18/2020 22:47	9/18/2020 22:46	9/18/2020 22:47	0
39	E:\DESKTOP-SDN1RPT	9/19/2020 1:24	9/19/2020 5:13	9/19/2020 5:09	9/19/2020 5:13	0

Obr. 21 Potencionálne zmazané súbory alebo adresáre

Záver

Každým dňom ľudia využívajú výpočtové zariadenia čoraz viac, a tým narastá pravdepodobnosť kybernetického útoku a následne aj incidentu. V prípade, keď dôjde k takému incidentu, je potrebné rýchlo reagovať, aby sme čo najskôr zistili jeho príčinu a získali digitálne stopy o jeho existencii. S tým vie pomôcť rýchle spracovanie a analýza forenzných artefaktov. Z tohto vyplýva nutnosť vyvíjať stále efektívnejšie nástroje na spracovanie forenzných artefaktov a ich analýzu.

Prvým cieľom tejto práce bola analýza možností spracovania forenzných artefaktov operačného systému Windows prostredníctvom dátovej analýzy. Tomuto cieľu sme sa venovali v prvej kapitole. V rámci nej sme si teoreticky priblížili jednotlivé forenzné artefakty a základné definície s nimi súvisiace. Hlavnou podstatou bolo pochopenie princípu ako fungujú forenzné artefakty a aký majú význam vo svetle forenznej analýzy. Zámerne sme v rámci analýzy opomenuli register operačného systému Windows ako celok. Tento register v sebe obsahuje niekoľko forenzných artefaktov, ktoré mnohokrát nie je možné spracovať automatizovaným spôsobom. Dôvodom je ich množstvo. V práci sme sa zamerali na Jump listy, ktoré sú uložené v tomto registri.

Druhý cieľom tejto práce bolo preskúmanie a porovnanie existujúcich prístupov k automatizovanému spracovaniu forenzných artefaktov operačného systému Windows. Tomuto cieľu sa venujeme v druhej kapitole. V rámci nej študujeme existujúce nástroje a prístupy k automatizovanému spracovaniu forenzných artefaktov. Táto kapitola má štyri časti, z ktorých prvé tri korešpondujú s jednotlivými nástrojmi, ktoré sme si vybrali na porovnávanie. Posledná časť je samotné porovnávanie nástrojov nielen medzi sebou ale aj s implementáciou nášho nástroja. Tieto nástroje dokážu pracovať so surovými nespracovanými dátami získať základné informácie súvisiace s nimi. Niektoré dokážu aj zložitejšie vyhľadávanie na základe preddefinovaných pravidiel.

V tretej kapitole sa venujeme porozumeniu dátovej sady z prípadu ukradnutej Sečuánskej omáčky, vybraného pre účel tejto práce. Rozoberáme, pre aké účely je poskytovaný a aké dáta z neho vieme získať. Približujeme aj dôvod, prečo je táto dátová sada vhodná na odskúšanie metód analýzy dát.

Posledným cieľom, ktorému sa venujeme v tretej kapitole, je návrh a implementácia nástroja na automatizované spracovanie forenzných artefaktov operačného systému Windows. Návrh riešenia je rozdelený do troch častí, ktorými sú

triáž obrazu disku, spracovanie metadát a následná analýza a práca so skracovanými dátami. Na spracovanie artefaktov sme si vybrali sériu parsovacích nástrojov od Erica Zimermana. Na následnú analýzu a získavanie štatistických informácií sme využili webové prostredie Jupyter Notebook a programovací jazyk Python vo verzii 3. Rozhodli sme sa nástroj poňať ako sadu jednotlivých skriptov. Takýto prístup je možné vidieť aj z niektorých podobných prístupov (napr. MSTIC a DS4N6 Chrysalis a Daisy).

Nástroj, ktorý sme navrhli, umožňuje urýchliť prácu forenzného analytika a prispieť tak k rýchlemu riešeniu bezpečnostných incidentov. V budúcnosti je tento nástroj možné rozšíriť o analýzu ďalších forenzných artefaktov alebo hlbšiu analýzu už aktuálne spracúvaných. Niektoré z uvedených forenzných artefaktov je možné spracovať ako časový rad (napr. SRUM artefakt) alebo ako graf (napr. záznamy logov). Súčasne je zaujímavé na základe získaných údajov automatizovane ohraničiť časový rámec bezpečnostného incidentu.

Zoznam použitej literatúry

1. Hassan, N. A. (2019). Digital forensics basics: A practical guide using Windows OS. Apress.

2. Harichandran, V. S., Walnycky, D., Baggili, I., & Breitinger, F. (2016). Cufa: A more formal definition for digital forensic artifacts.

3. Do, Q., Martini, B., Looi, J., Wang, Y., & Choo, K. K. (2014). Windows event forensic process. In Advances in Digital Forensics X: 10th IFIP WG 11.9 International Conference, Vienna, Austria, January 8-10, 2014, Springer Berlin Heidelberg.

4. Audit logon events. [online] Dostupné na: https://learn.microsoft.com/en-us/windows/security/threat-protection/auditing/basic-audit-logon-events [Pristúpené 22. Januára 2023].

5. Beginner's Guide to Jupyter Notebook & JupyterLab. [online] Dostupné na: http://www.ds4n6.io/blog/20070702.html [Pristúpené 23. Januára 2023].

6. Python for Data Science, [online] Dostupné na: https://www.geeksforgeeks.org/python-for-data-science/ [Pristúpené 23. Januára 2023].

7. Shashidhar, N., & Novak, D. (2015). Digital forensic analysis on prefetch files. International Journal of Information Security Science.

8. Antonovich, C. (2014). Jump List Forensics. Patrick Leahy Center for Digital Investigation (LCDI), Champlain College Miller Center, Burlington, USA.

9. Ghafarian, A. (2015). Investigating Forensics Values of Windows Jump Lists Data.

10. Naiqi, L., Zhongshan, W., & Yujie, H. (2008, August). Computer forensics research and implementation based on NTFS file system. In 2008 ISECS International Colloquium on Computing, Communication, Control, and Management. IEEE.

11. Singh, B., & Singh, U. (2018). Program execution analysis in Windows: A study of data sources, their format and comparison of forensic capability. Computers & Security.

12. Understanding Shims, [online] Dostupné na: https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-7/dd837644(v=ws.10). [Pristúpené 9. Apríla 2023].

13. Lagny, B. (2019). Analysis of the amcache. ANSSI-DFIRSummit.

14. Singh, B., & Singh, U. (2016). Leveraging the windows amcache. hve file in forensic investigations. Journal of Digital Forensics, Security and Law.

15. Kim, M., & Lee, S. (2015, October). Forensic analysis using amcache.hve. In Digital Forensics and Cyber Crime: 7th International Conference, ICDF2C 2015. Springer Seoul, South Korea.

16. Huebner, E., Bem, D., & Wee, C. K. (2006). Data hiding in the NTFS file system. digital investigation.

17. Knyazeva, N., Khorkov, D., & Vostretsova, E. (2020). Building Knowledge Bases for Timestamp Changes Detection Mechanisms in MFT Windows OS. 2020 Ural Symposium on Biomedical Engineering, Radioelectronics and Information Technology (USBEREIT).

18. Master File Table (Local File Systems) - Win32 apps. [online] Dostupné z: https://learn.microsoft.com/en-us/windows/win32/fileio/master-file-table [Pristúpené 15. Apríla 2023].

19. Zhang, Z., Shi, J., & Hu, L. (2015). Towards enumeration of NTFS using USN journals under UEFI. In Trustworthy Computing and Services: International Conference, ISCTCS 2014, Beijing, China, November 28-29, 2014, Revised Selected papers. Springer Berlin Heidelberg.

20. NTFS File System Overview [online] Dostupné z: http://www.c-jump.com/bcc/t256t/Week04NtfsReview/Week04NtfsReview.html#W01_0150_master_file_table [Pristúpené 15. Apríla 2023].

21. Usn Journal Forensics Extraction for Efficient Investigation , [online] Dostupné na: https://www.otorio.com/resources/usnjrnl-extraction-for-efficient-investigation/ [Pristúpené 15. Apríla 2023].

22. USN Journal, [online] Dostupné na: https://forensafe.com/blogs/usnjournal.html [Pristúpené 15. Apríla 2023].

23. Kävrestad, J. (2020). Fundamentals of Digital Forensics. Springer International Publishing.

24. Casey, E. (2011). Digital evidence and computer crime: Forensic science, computers, and the internet. Academic press.

25. Event Types - Win32 apps. [online] Dostupné z: https://learn.microsoft.com/en-us/windows/win32/eventlog/event-types [Pristúpené 16. Apríla 2023].

26. DFIR Madness. 2022. DFIR Madness. [online] Dostupné z: https://dfirmadness.com/about/ [Pristúpené 16. Apríla 2023].

27. Introducing KAPE – Kroll Artifact Parser and Extractor. [online] Dostupné z: https://www.kroll.com/en/insights/publications/cyber/kroll-artifact-parser-extractor-kape [Pristúpené 16. Apríla 2023].

28. Minnaard, W., de Laat, C. T. A. M., & van Loosen MSc, M. (2014). Timestomping ntfs. IMSc final research project report, University of Amsterdam, Faculty of Natural Sciences, Mathematics and Computer Science.

29. Khatri, Y. (2015). Forensic implications of system resource usage monitor (SRUM) data in windows 8. *Digital Investigation*.

30. Đuranec, A., Topolčić, D., Hausknecht, K., & Delija, D. (2019, May). Investigating file use and knowledge with Windows 10 artifacts. In *2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. IEEE.

31. MSTIC Jupyter and Python Security Tools. [online] Dostupné z: https://msticpy.readthedocs.io/en/latest/index.html [Pristúpené 29.4.2023]

32. CHRYSALIS. [online] Dostupné z: https://www.ds4n6.io/tools/chrysalis.html [Pristúpené 29.4.2023]

33. DAISY: Say Hi to the New DS/AI-for-DFIR Virtual Machine! [online] Dostupné z: http://www.ds4n6.io/blog/21051704.html [Pristúpené 29.4.2023]

34. Live-Forensicator [online] Dostupné z: https://github.com/Johnng007/Live-Forensicator [Pristúpené 30.4.2023]

35. Mize, R. (2018). *Behavior of Shellbags in windows 10* (Doctoral dissertation, Utica College).

36. Pulega, D. (2013). Shellbags Forensics: Addressing a misconception(Interpretation, step-bystep testing, new findings and more). [online] Dostupné z:

http://www.4n6k.com/2013/12/shellbags-forensics-addressing.html [Pristúpené 15.5.2023]

37. Mbatha, M. P. (2016). *Windows Registry Forensic Artifacts; Shellbags for Computer Security* (Doctoral dissertation, University Of Nairobi).

38. Zhu, Y., Gladyshev, P., & James, J. (2009). Using shellbag information to reconstruct user activities. *digital investigation*, *6*, S69-S77.

39. SRUM: Forensic Analysis of Windows System Resource Utilization Monitor. [online] Dostupné z: https://www.magnetforensics.com/blog/srum-forensic-analysis-of-windows-system-resource-utilization-monitor/ [Pristúpené 4.5.2023]

40. Investigating Windows System Resource Usage Monitor (SRUM). [online] Dostupné z: https://forensafe.com/blogs/srudb.html [Pristúpené 5.5.2023]

41. A Simplified Guide To Digital Evidence. [online] Dostupné z: https://www.forensicsciencesimplified.org/digital/ [Pristúpené 9.5.2023]

42. Johannes Ullrich (2022) Forensic Value of Prefetch. [online] Dostupné z: https://isc.sans.edu/diary/Forensic+Value+of+Prefetch/29168/ [Pristúpené 9.5.2023]

43. Forensic Analysis of Prefetch files in Windows. [online] Dostupné z: https://www.magnetforensics.com/blog/forensic-analysis-of-prefetch-files-in-windows/

44. Desktop Operating System Market Share Worldwide - April 2023 [online] Dostupné z: https://gs.statcounter.com/os-market-share/desktop/worldwide [Pristúpené 18.5.2023]

45. Registry [online] Dostupné z: https://learn.microsoft.com/en-us/windows/win32/sysinfo/registry [Pristúpené 18.5.2023]

46. Walkthrough of DFIR Madness PCAP [online] Dostupné z: https://www.netresec.com/?page=Blog&month=2021-07&post=Walkthrough-of-DFIR-Madness-PCAP [Pristúpené 18.5.2023]

47. Falcon Sandbox Public API [online] Dostupné z: https://www.hybrid-analysis.com/docs/api/v2 [Pristúpené 18.5.2023]

48. MITRE ATT&CK [online] Dostupné z: https://attack.mitre.org/ [Pristúpené 18.5.2023]

Prílohy

Príloha A: Bakalárska práca v elektronickej podobe, prílohy v elektronickej podobe.

Príloha B: Zdrojový kód nástroja na automatizované spracovanie forenzných artefaktov