ANALÝZA FORENZNÝCH ARTEFAKTOV WEBOVÝCH PREHLIADAČOV V OPERAČNÝCH SYSTÉMOCH WINDOWS

Diplomový seminár z informatiky

Autor: Bc. Zuzana Hennlová
Vedúci práce: doc. RNDr. JUDr. Pavol Sokol, PhD. et. PhD.

Úvod

Pri každodennom používaní elektronických zariadení generujú ich používatelia značné množstvo dát. Počas práce s webovým prehliadačom sa na pozadí ukladajú rôzne informácie za účelom zabezpečenia správnej funkčnosti, rýchlosti a prispôsobenia sa používateľovi. Časť z nich môže byť cenným artefaktom pri forenznej analýze, predovšetkým pri hľadaní vektora útoku (ako sa útočník dostal do systému), ktorým môže byť práve aktivita používateľa na webových stránkach. Webové forenzné artefakty sú veľmi užitočné aj pri dokazovaní, kedy slúžia ako dôkazy o navštívení webových stránok či sťahovaní súborov, ktoré súvisia s ilegálnymi aktivitami.

Na forenznú analýzu webových prehliadačov existuje viacero voľne dostupných nástrojov, ktoré dokážu pracovať s časťou dát generovaných konkrétnym prehliadačom. Voľne dostupný je aj známy nástroj Autopsy [1], ktorý dokáže zobraziť značnú časť webových artefaktov. Tieto nástroje však končia zobrazením dát a celé hľadanie podozrivých udalostí je postavené na znalostiach a skúsenostiach forenzného analytika. Ten častokrát potrebuje získať v krátkom čase základné informácie o incidente, čo je pri veľkom množstve dát náročné.

Cieľom tejto diplomovej práce je poskytnúť automatizovanú identifikáciu podozrivých artefaktov, ktorá urýchli proces hľadania podozrivých

[1] https://www.autopsy.com/

udalostí. Podľa portálu Statista [1] bol v Spojených Štátoch za rok 2023 priemerný čas na uzavretie forenzného vyšetrovania 33 dní. Veríme, že urýchlenie forenznej analýzy webových prehliadačov prispeje k redukovaniu celkového času potrebného na vykonanie forenznej analýzy zariadenia. S využitím metód strojového učenia budeme identifikovať anomálie nad časovou osou, ktorá bude obsahovať zjednotené a zotriedené udalosti generované webovými prehliadačmi. To umožní veľmi rýchlu identifikáciu podozrivých aktivít, ktoré je potrebné preveriť ako prvé, ako aj informáciu o časovom rozmedzí, v ktorom nastali atypické udalosti.

V tomto článku sa venujeme oboznámeniu čitateľa s vybranými artefaktmi webových prehliadačov Google Chrome, Mozilla Firefox a Edge pre operačné systémy Windows. Uvádzame ich základný popis, umiestnenie v súborovom systéme ako aj existujúcue nástroje na ich zobrazovanie. Samozrejme chceme v rámci práce preskúmať a popísať aj artefakty iných populárnych prehliadačov, no ich preskúmanie a predovšetkým odskúšanie hľadania anomálií značne závisí od toho, či sa dané prehliadače použili v skúmaných datasetoch.

V druhej kapitole uvádzame prehľad podobných prác a v tretej kapitole sa venujeme popísaniu prístupu k riešeniu stanoveného problému extrakcie, spracovania a zobrazovania dát, vrátane ich obohatenia o identifikované anomálie.

1 Artefakty webových prehliadačov

Počas používania webového prehliadača sa štandardne na pozadí ukladá množstvo dát, či už za účelom urýchlenia načítavania, zjednodušenia prehliadania alebo vykonávania nevyhnutnej funkcionality. Tieto dáta sú v mnohých prípadoch štrukturované a ukladajú sa teda do databáz. Všetky nami skúmané prehliadače využívaju SQLite databázu, no každý z nich vytvára iné databázy s odlišnými tabuľkami a atribútmi. Prehliadače založené na Chromium (napr. Google Chrome, Edge) majú veľmi podobné štruktúry, no aj v nich sa vyskytujú určité odlišnosti.

V nasledujúcich podkapitolách uvádzame niektoré základné artefakty a popisujeme miesto a formu uloženia ako aj existujúce nástroje na zobrazovanie daných artefaktov. Všeobecne je možné na zobrazovanie použiť akúkoľvek aplikáciu určenú na prácu s SQLite jazykom.

Pre účely overovania či dopĺňania informácií o artefaktoch využívame zariadenie s operačným systémom Windows verzie 11 a s prehliadačmi Google Chrome verzie 123.0.6312.123, Mozilla Firefox (Ďalej iba Firefox) verzie 124.0.2 a Edge verzie 125.0.2535.92.

1.1 História prehliadania

Históriou rozumieme navštevované stránky a vyhľadávacie dopyty. K tomu patria informácie o časovej pečiatka udalosti, URL odkaz, počet navštívení stránky a pod. Tabuľky v databáze histórie pre prehliadače Google Chrome a Firefox a vzťahy medzi nimi sú znázornené a bližšie popísané v kapitole 3.2.

Umiestnenie artefaktu

- Google Chrome

  `%USERPROFILE%\AppData\Local\Google\Chrome\User Data\<Profile>\History`

  - Najmä tabuľky s názvami urls, visited_links, visits, ale aj tabuľka stiahnutých súborov.

- Mozilla Firefox - tabuľka moz_places, moz_places_metadata, moz_historyvisits, moz_inputhistory v databáze umiestnentej:

  `%USERPROFILE%\AppData\Roaming\Mozilla\Firefox\Profiles\<random text>.default\places.sqlite`

- Edge

  `%USERPROFILE%\AppData\Local\Microsoft\Edge\User Data\Default\History`

  - Podobné tabuľky ako má prehliadač Chrome.

Nástroje na spracovanie

- **Browsing History View** [2] - podpora pre Internet Explorer, Firefox, Google Chrome a Safari, zobrazuje navštívené URL, názov, čas návštevy, počet návštev stránky, použitý prehliadač a používateľský profil.
- **MZHistoryView** [3] - iba pre prehliadač Firefox
- **ChromeHistoryView** [4] - navštívené stránky cez Google Chrome

1.2 Stiahnuté súbory

Informácie o stiahnutých súboroch slúžia používateľom na rýchly prístup k práve stiahnutým súborom priamo z prehliadača. Nie všetky stiahnuté súbory sú zaznamenané v databáze, najmä v prípadoch, ak bol použitý privátny režim, alebo ak používateľ sám odstránil súbor z histórie. Stiahnuté súbory sú veľmi často príčinou vzniku bezpečnostného incidentu a patria tak k významným artefaktov.

[2] https://www.nirsoft.net/utils/browsing_history_view.html
[3] https://www.nirsoft.net/utils/mozilla_history_view.html
[4] https://www.nirsoft.net/utils/chrome_history_view.html

Umiestnenie artefaktu

- Google Chrome - tabuľka downloads, downloads_url_chains

  `%USERPROFILE%\AppData\Local\Google\Chrome\User Data\<Profile>\History`

- Mozilla Firefox - tabuľka moz_annos a moz_places na umiestnení:

  `%USERPROFILE%\AppData\Roaming\Mozilla\Firefox\Profiles\<random text>.default\places.sqlite`

- Edge - tabuľky downloads, downloads_slices a downloads_url_chains.

  `%USERPROFILE%\AppData\Local\Microsoft\Edge\User Data\Default\History`

Nástroje na spracovanie

- **FirefoxDownloadsView** [5] - súbory stiahnuté cez tento prehliadač spolu s informáciou o URL, názve súboru s celou cestou umiestnenia, veľkosťou súboru a i.

1.3 Cookies

Cookies sú malé textové súbory, ktoré primárne slúžia na ukladanie preferencií (napr. jazyk, veľkosť textu) či rozpracovaných akcií (napr. uloženie položky v nákupnom košíku) používateľa pre jednotlivé webstránky. Okrem toho sú často používané na uchovávanie informácií o záujmoch používateľa pre poskytnutie lepších odporúčaní. Z forenzného hľadiska je pridanou hodnotou informácia o navštívených stránkach (najmä ak sú súčasťou škodlivej aktivity anti-forenzné techniky - vymazanie histórie prehliadania) [2].

[5] https://www.nirsoft.net/utils/firefox_downloads_view.html

Umiestnenie artefaktu

- Mozilla Firefox - jediná tabuľka s názvom moz_cookies

  `%USERPROFILE%\AppData\Roaming\Mozilla\Firefox\`
  `Profiles\<randomtext>.default\cookies.sqlite`

- Google Chrome

  `%USERPROFILE%\AppData\Local\Google\Chrome\`
  `User Data\<Profile>\Network\Cookies`

- Edge

  `%USERPROFILE%\AppData\Local\Microsoft\Edge\`
  `User Data\Default\Network\Cookies`

1.4 História formulárov

V kontexte prehliadačov sa jedná o akékoľvek textové polia, do ktorých sa dopĺňajú rôzne údaje. Najčastejšie sú to online formuláre ale aj prihlasovacie polia či vyhľadávanie. Názvy vypĺňacích polí spravidla nie sú v databáze jednotné - berie sa názov textového poľa danej stránky, ktorý nie je jednotný, preto sa môžeme stretnúť s rôznymi názvami, napr. hodnota mena sa môže nachádzať v okienku s názvom "Meno", "meno", "krstné meno", "name"... K uloženiu údajov dochádza po odoslaní daného formulára.

Z forenzného hľadiska sú zaujímavé údaje, ktoré vznikli alebo boli použité v čase incidentu. To môže napomôcť k formovaniu obrazu toho, čo robil používateľ na webe v čase, kedy sa vyskytla škodlivá aktivita [3]. Nástroje na extrakciu tohto typu dát sa nám nepodarilo nájsť.

Umiestnenie artefaktu

- **Mozilla Firefox** - predovšetkým tabuľka moz_formhistory, ktorá obsahuje informácie o názve okienka (vypĺňacieho poľa) a jeho hodnote. Ďalšie tabuľky sú moz_deleted_formhistory, moz_sources a

moz_history_to_sources. Posledné 2 mapujú záznamy z histórie na zdroj, z ktorého záznam pochádza (obvykle je to názov vyhľadávača). Záznamy obsahujú aj texty zadávané do vyhľadávacieho okna (searchbar-history). Podľa vlastných pozorovaní sa v prípade práce v súkromnom okne tieto dáta neaktualizujú.

`%USERPROFILE%\AppData\Roaming\Mozilla\Firefox\Profiles\<randomtext>.default\formhistory.sqlite`

- **Google Chrome** - tabuľka autofill obsahuje názov okienka a príslušnú hodnotu s časom vytvorenia a posledného použitia. Databáza obsahuje 33 ďalších tabuliek, ktoré sú poväčšine venované uloženiu konkrétnych typov údajov ako sú napr. kontaktné údaje, platobné karty, adresy.

  `%USERPROFILE%\AppData\Local\Google\Chrome\User Data\<Profile>\Web Data`

- **Edge** - podobná štruktúra ako Google Chrome.

  `%USERPROFILE%\AppData\Local\Microsoft\Edge\User Data\Default\Web Data`

1.5 Vybrané štruktúry databáz

V tejto podkapitole uvádzame popisy vybraných štruktúr databáz prehliadača Firefox a Google Chrome. Pri jednotlivých tabuľkách neuvádzame všetky atribúty vzhľadom na ich množsto. Vybrané sú iba relevantnejšie z nich.

1.5.1 Prehliadač Firefox

Všetky atribúty typu *integer*, ktoré sa týkajú dátumu a času sú v databáze ukladané v unixepoch formáte. Na prevod do klasického formátu je potrebné hotnotu najprv predeliť číslom 1000000 a potom použiť prevod z unixepoch, napr. pri prevode priamo počas operácie JOIN v databáze použijeme "datetime(tabulka.atribut_casu/1000000, 'unixepoch') AS prevedeny_cas".

Tabuľky databázy places.sqlite

Tabuľka **moz_places** obsahuje základné údaje o navštívených stránkach a slúži ako hlavná tabuľka histórie prehliadania.

- **id** (integer): Primárny kľúč tabuľky.
- **url** (string): Odkaz na navštívenú stránku.
- **visit_count** (integer): Celkový počet návštev danej stránky.
- **last_visit_date** (integer): Dátum a čas poslednej návštevy.
- **origin_id** (integer): Odkaz na tabuľku moz_origins ktorá obsahuje informácie o pôvode URL adresy.

Tabuľka **moz_annos** obsahuje anotácie (dodatočné informácie) spojené s navštívenými stránkami.

- **id** (integer): Primárny kľúč tabuľky.
- **place_id** (integer): Cudzí kľúč, ktorý prepája tabuľku moz_annos s tabuľkou moz_places.
- **anno_attribute_id** (integer): Mapovanie na typ atribútu v tabuľke moz_anno_attributes.
- **expiration** (integer): Dátum a čas, kedy anotácia prestáva byť platná.
- **dateAdded** (integer): Dátum pridania anotácie.
- **lastModified** (integer): Dátum poslednej úpravy anotácie.

Tabuľka **moz_anno_attributes** obsahuje typy anotácií, ktoré sa môžu viazať na jednotlivé stránky. Sú iba s 3 typy anotácií. Pri stiahnutí súboru vzniknú 2 nové záznamy, jeden typu "downloads/metaData", pri ktorom sa zapíšu metadáta sťahovania (koniec sťahovania, vľkosť súboru) a jeden typu "downloads/destinationFileURI". Ten ukladá cestu v súborovom systéme, kde sa pri sťahovaní uložil súbor. Tretí typ neobsahoval žiadne zaujímavé hodnoty a nesúvisí so sťahovaním.

- **id** (integer): Primárny kľúč tabuľky.
- **name** (string): Jeden z troch typov: downloads/destinationFileURI, downloads/metaData, URIProperties/characterSet.

Tabuľka **moz_historyvisits** drží informácie o návštevách stránok, dopĺňa informácie z hlavnej tabuľky moz_places predovšetkým o spôsob, akým sa navštívila stránka.

- **id** (integer): ID každej návštevy – každé otvorenie aj rovnakej stránky má iné ID.
- **from_visit** (integer): ID pôvodcu.
- **place_id** (integer): ID mapované na tabuľku moz_places.
- **visit_date** (integer): Dátum a čas navštívenia.
- **visit_type** (integer): Typ návštevy. Rozlišuje sa až 9 rôznych typov, ako sa mohol používateľ prekliknúť na danú stránku, napr. kliknutím na odkaz, ktorý otvoril novú kartu, navštívením záložky, opätovným načítaním stránky a iné. [6]

Tabuľka **moz_inputhistory** ukladá vyhľadávania URL adries a tým možňuje nahradiť vyhľadávané výrazy kompletnou URL adresou (napr. pri písaní výrazu "face" sa ponúkne možnosť navštívenia webovej stránky "www.facebook.com").

- **place_id** (integer): ID na mapovanie na konkrétnu stránku v tabuľke moz_places.
- **input** (string): Reálny vstup používateľa (URL pred automatickým doplnením).
- **use_count** (integer): Počet použití daného vstupu.

Záložky sa ukladajú v tabuľke **moz_bookmarks**. Napovedajú o aktivatách, ktoré sú pre používateľa bežné.

[6] https://gist.github.com/olejorgenb/9418bef65c65cd1f489557cfc08dde96

- **id** (integer): ID záložky.
- **title** (string): Názov záložky.
- **fk** (integer): Odkazuje na place_id.
- **dateAdded** (integer): Dátum pridania záložky.
- **lastModified** (integer): Dátum poslednej úpravy záložky.
- **guid** (string): Globálne ID záložky. Odkazuje na rovnomenný atribút guid v tabuľke moz_bookmarks_deleted. V nej sa nachádza iba dátum odstránenia záložky.

Tabuľky databázy favicons.sqlite

Hlavnou tabuľkou v tejto databáze je **moz_icons**. Obsahuje viacero atriibútov, ktoré slúžia na správne zobrazenie (rozmer, umiestnenie a pod.). Potenciálne zaujímavé sú iba nasledujúce atribúty.

- **id** (integer): Primárný kľúč - ID každého záznamu.
- **icon_url** (string): URL odkaz pre zobrazenie konkrétnej ikony.
- **expire_ms** (integer): Čas exspirácie platnosti záznamu.

V tabuľke **moz_pages_w_icons** sa uchováva mapovanie stránok na ikony.

- **id** (integer): ID ikony.
- **page_url** (string): Príslušný URL odkaz.
- **page_url_hash** (integer): Tento digitálny odtlačok (hash) odkazu sa zhoduje s hashom uloženým v databáze places.sqlite. Na základe toho pridávame informácie o ikonách k dátam histórie prehliadania.

Tabuľky databázy formhistory.sqlite

Hlavnú tabuľku prestavuje **moz_formhistory**, v ktorej sú uložené konkrétne hodnoty dopĺňané do formulárov.

- **fieldname** (string): Názov dopĺňaného poľa. Ukladá sa na základe pomenovnania poľa v kóde na konkrétnej webovej stránke (napr. meno, surname, adresa, bydlisko, e-mail address a pod.).
- **value** (string): Konkrétna hodnota, ktorá bola v minulosti vyplnená používateľom a bude sa ponúkať aj v novom formulári.
- **timesUsed** (integer): Počet použití doplnenia tejto hodnoty.
- **firstUsed** (integer): Dátum a čas prvého použitia.
- **lastUsed** (integer): Dátum a čas posledného použitia.
- **guid** (string): Globálny identifikátor v rámci databázy.

Informácie o vymazaných záznamoch dopĺňania sa ukladajú do tabuľky **moz_deleted_formhistory**.

- **timeDeleted** (integer): Čas vymazania záznamu.
- **guid** (string): Identifikátor záznamu - koreluje s atribútom guid v hlavnej tabuľke.

Tabuľky databázy cookies.sqlite

Databáza obsahuje jedinú tabuľku s názvom **moz_cookies**.

- **name** (string): Pomenovanie súboru cookie.
- **value** (string): Príslušná hodnota.
- **host** (string): Doménové meno.
- **expiry** (integer): Koniec platnosti.
- **lastAccessed** (integer): Dátum a čas posledného použitia cookie.
- **creationTime** (integer): Dátum a čas vytvorenia.

1.5.2 Prehliadač Google Chrome

V tomto prehliadači sa konvercia času na čitateľný formát robí najprv predelením hodnoty číslom 1000000 a odčítaním hodnoty 11644473600. Tým získame unixepoch formát, ktorý je možné previesť pomocou funkcie datetime. Príklad: datetime("atribut_casu"\1000000-11644473600, 'unixepoch').

Tabuľky databázy History

Záznamy o sťahovaných súboroch sú v tabuľke **downloads**.

- **current_path** (string): Cesta v súborovom systéme, kde je súbor práve uložený.
- **target_path** (string): Cesta v súborovom systéme, kde bol súbor ukladaný počas sťahovania.
- **total_bytes** (integer): Celková očakávaná veľkosť súboru v bajtoch.
- **state** (integer): Status sťahovania s hodnotami od 0 do 4 [4].
- **danger_type** (integer): Vyhodnotenie bezpečnosti sťahovaného súboru. Nadobúda hodnoty 0 až 9. 0 označuje súbor bez nájdenej hrozby. Ostatné hodnoty predstavujú napr. nebezpečnú URL, nezvyčajný obsah, či potenciálne nevyžiadaný súbor. [4]
- **hash** (string): Digitálny odtlačok stiahnutého súboru.
- **last_access_time** (integer): Čas posledného prístupu.
- **tab_url** (string): URL odkaz stránky, z ktorej sa iniciovalo stiahnutie.
- **last_modified** (string): Čas poslednej úpravy.

V tabuľke **visits** sú zaznamenané jednotlivé navštívenia stránok s dodatočnými informáciami.

- **id** (integer): ID záznamu.
- **url** (integer): ID korešpondujúce si ID príslušnej URL v tabuľke urls.

- **visit_time** (integer): Čas pristúpenia na stránku.
- **from_visit** (integer): ID stránky, z ktorej sa prekliklo na túto stránku.
- **transition** (integer): Spôsob navštívenia tejto stránky. Reperezentované sú numerickou hodnotou, ktorú je najprv potrebné transformovať, čím získame jednu z 10 možností. [5]
- **visit_duration** (integer): Trvanie návštevy stránky v milisekundách.

Tabuľka **urls** obsahuje všetky jedinečné navštívené stránky spolu s počtom navštívení a dátumom posledného otvorenia stránky.

- **id** (integer): Primárny kľúč databázy, ktorý jednoznačne identifikuje URL.
- **url** (string): URL odkaz.
- **visit_count** (integer): Počet otvorení danej stránky.
- **last_visit_time** (integer): Dátum a čas poslednej interakcie s danou stránkou.

Tabuľka **keyword_search_terms** obsahuje výrazy písané priamo do adresného riadka.

- **url_id** (integer): Identifikátor URL odkazu v tabuľke *urls*.
- **term** (string): Vyhľadávaný výraz.

Tabuľky databázy Cookies

Databáza Cookies obsahuje 2 tabuľky, popíšeme iba tabuľku **cookies**, nakoľko druhá obsahuje nepodstatné metadáta.

- **creation_utc** (integer):
- **host_key** (string): Doména, ktorá pracuje s daným cookie. Nemusí to byť nutne navštívená stránka.

- **name** (string): Názov daného cookie súboru.
- **encrypted_value** (string): V súčasnosti je štandard, že všetky cookie súbory sú ukladané v zašifrovanej forme.
- **expires_utc** (integer): Dátum a čas, kedy končí platnosť záznamu.
- **is_httponly** (integer): Niektoré cookie sú použiteľné iba s HTTP protokolom.
- **is_persistent** (integer): Ak je hodnota 1, cookie ostane uložené aj po ukončení spojenia až do jeho nastavenej exspirácie.

2 Podobné práce

V tejto kapitole uvádzame prehľad prác (tabuľka 1), ktoré sa zaoberajú forenznou analýzou webových prehliadačov, resp. sa venujú inej téme s využitím metód, ktoré by mohli byť použité pre náš zámer. Mnoho článkov sa zaoberá iba popisom základných atribútov, prípadne ich obnove. Posledné 2 práce síce využívajú metódy strojového učenia nad forenznými dátami, no nie nad zjednotenými dátami z prehliadačov. Článok [6] taktiež používa metódy strojového učenia, no nad úplne inými dátami. Vo všetkých troch je však vidieť pomerne dobré výsledky, preto sme optimistickí aj pri našej práci.

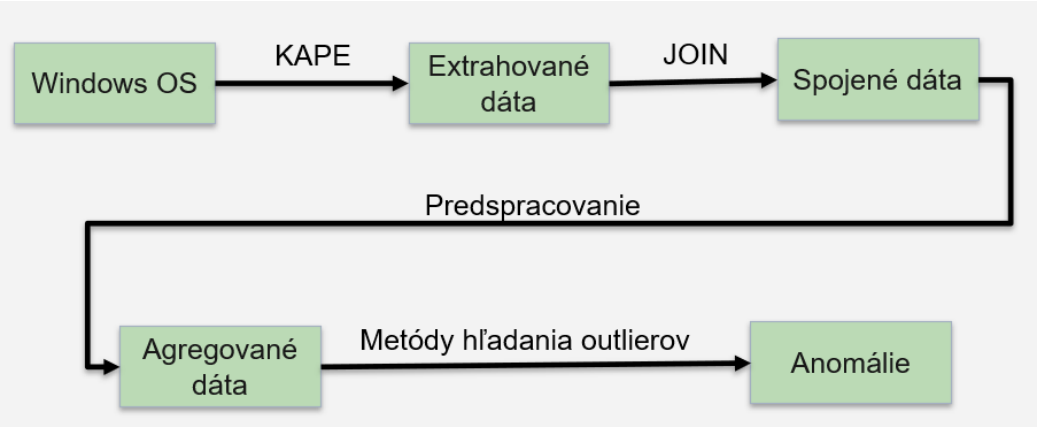
Názov	**Rok**	**Popis**
Web Browser Forensics in Google Chrome, Mozilla Firefox, and the Tor Browser Bundle [7]	2020	Práca sa venuje identifikácii artefaktov a ich obnove. Skúmané sú viaceré módy prehliadania.
AIBFT: Artificial Intelligence Browser Forensic Toolkit [6]	2021	Článok popisuje možnosť použitia strojového učenia na detekciu škodlivých webov. Najlepšiu presnosť (99.8%) dosahujú s použitím algoritmu náhodných lesov. Nosným artefaktom je webová vyrovnávacia pamäť (cache).
WEB BROWSER FORENSICS: Evidence collection And Analysis for Most Popular Web Browsers usage in Windows 10 [8]	2018	Práca venovaná základným artefaktom (história, cookies, dopyty, stiahnuté súbory) prehliadačov a ich popisom.
Firefox Concrete Architecture [9]	-	Táto práca sa snaží čo najpodrobnejšie popísať štruktúru prehliadača Mozilla Firefox. Jedná sa o staršiu verziu, no mnohé princípy a popisy ostávajú nezmenené.

Názov	**Rok**	**Popis**
WEB BROWSER FORENSICS: GOOGLE CHROME [10]	2017	Príspevok venovaný základným artefaktom prehliadača Google Chrome. Okrem databáz využíva Prefetch súbory a výpis (dump) pamäte RAM.
Anomaly detection in a forensic timeline with deep autoencoders [11]	2021	Práca využíva strojové učenie na detekciu anomálií nad časovou osou záznamov udalostí s 96% presnosťou.
Detection of relevant digital evidence in the forensic timelines [12]	2022	Práca je venovaná identifikácii outlierov nad binárnymi dátami, ktoré vznikli ako transformácia dát zo súborového systému. Cieľom je automatizovať hľadanie podozrivých záznamov pri forenznej analýze.

Tabuľka 1: Prehľad existujúcej literatúry

3 Návrh riešenia

V rámci diplomovej práce chceme vybudovať nástroj, ktorý bude schopný pracovať nad viacerými webovými prehliadačmi. Vizuálny proces prevodu surových dát na relventné informácie je znázornený na obrázku 1. Nástroj zozbiera relevantné dáta spojí ich a pripraví ich na ďalšiu fázu agregáciou, prevodom dát na binárne a numerické hodnoty a odstránením atribútov, ktoré neobsahujú relevantné hodnoty. Následne pomocou metód hľadania outlierov - dát, ktorá sá odlišujú od ostatných, získame anomálie v dátach. Tie predstavujú hlavnú pridanú hodnotu voči bežným, voľne dostupným nástrojom, nakoľko dáta nie len zobrazujeme ale ich aj analyzujeme a identifikujeme záznamy, ktoré sú podozrivé a forenzný analytik by sa na ne mal pri vyšetrovaní zamerať.



Obr. 1: Vizuálny návrh riešenia

Na odskúšanie využijeme dáta z viacerých voľne dostupných datasetov, čo umožní lepšie optimalizovať metódy vyhľadávania anomálií. Nižšie uvádzame tabuľku (2) niektorých takýchto datasetov.

Okrem takýchto dát budeme používať aj umelo vygenerované dáta, nakoľko je nedostatok zdrojov, ktoré by obsahovali dáta aj z iných prehliadačov. Generovanie aktivity prebieha s využitím knižnice Playwright [7], vďaka ktorej je možné bez interakcie používateľa klikať na odkazy (ukážka na

[7] https://playwright.dev/python/docs/intro

Obraz disku	**OS**	**Odkaz na stiahnutie**
BloomCON2022 ForensicsChallenge.E01	Windows 10 Pro	Google Disk
sda Image.E01	Windows 10 Pro	Part 1, Part 2, Part 3, Part 4, Part 5
Horcrux.E01	Windows 10 Pro & Kali Linux	Dropbox
LoneWolf.E01	Windows 10 Education	Digital Corpora
bart.E01	Windows 11 Pro	Nist
MaxPowersC Drive.E01	Windows 10 Enterprise	N/A
MUS-CTF-19-DESKTOP-001.E01	Windows 10 Enterprise	Digital Corpora
Laptop1Final.E01	Windows 11 Home Insider Preview	Digital Corpora
PC-MUS-001.E01	Windows 11 Home	Magnetforensics
HD1.E01	Windows 10 Pro	Digital Corpora
base-rd01-cdrive.E01	Windows 10 Enterprise	N/A

Tabuľka 2: Popis vybraných zdrojov dát

obrázku 2), hľadať výrazy na základe vopred vytvoreného zoznamu slovných spojení, posúvať sa na stránke a pod. Celý kód umožňuje po zadaní počtu navštívených stránok a počtu akcií vykonaných na každej stránke náhodne otvárať webové stránky z pripraveného zoznamu a v náhodných intervaloch vykonávať náhodne zvolené aktivity. Odskúšané fungovanie je pre prehliadač Chrome a Firefox.

```
    def click_random_link(page):
        try:
            links = page.query_selector_all("a[href]")
            if links:
                random_link = random.choice(links)
                if random_link.is_visible():
                    random_link.click()
                    page.wait_for_load_state('networkidle')
                    print("Clicked random link successfully."
                        )
            else:
                print("No links found on this page.")
        except Exception as e:
            print(f"Error clicking random link: {e}")
```

Obr. 2: Ukážka výberu náhodného odkazu zo stránky

3.1 Extrakcia dát

Na extrakciu dát využívame na pozadí nástroj Kroll Artifact Parser and Extractor (ďalej iba KAPE[8]). Vďaka nemu sme schopní extrahovať dáta zo zvoleného pripojeného disku a previesť ich do čitateľnej podoby (najčastejšie CSV súbor). KAPE je open-source nástroj určený na extrakciu forenzných artefaktov z digitálnych dát. V prípade extrakcie z obrazu disku je potrebné mať disk aktívne pripojený na čítanie. Je to však flexibilný nástroj, ktorý umožňuje definovanie vlastných objektov (artefaktov), s ktorými bude

[8] www.kroll.com

nástroj pracovať. Poskytuje grafické rozhranie ako aj možnosť používania v príkazovom riadku, ktorá môže byť súčasťou výsledného skriptu diplomovej práce. Na správne fungovanie extrakcie a spracovania dát je potrebné nástroju udeliť administrátorské oprávnenia.

V časti "Targets" určenej na extrakciu, volíme nasledujúce možnosti: SANAS_triage, Recycle Bin Data and Info, SQLite databases, Windows Index Search, Browser Cache, EdgeChromium Extensions. V časti "Modules" určenej na spracovanie extrahovaných dát spúšťame moduly EZParser a RegRipper. Tým z datasetu získame všetky potrebné informácie pre ďalšiu analýzu. Nakoľko KAPE umožňuje vytváranie vlastných modulov a targetov, spomínané možnosti sme zakomponovali do vlastného modulu a targetu, teda nie je potrebné manuálne označovať všetky vypísané možnosti, stačí označiť len jednu.

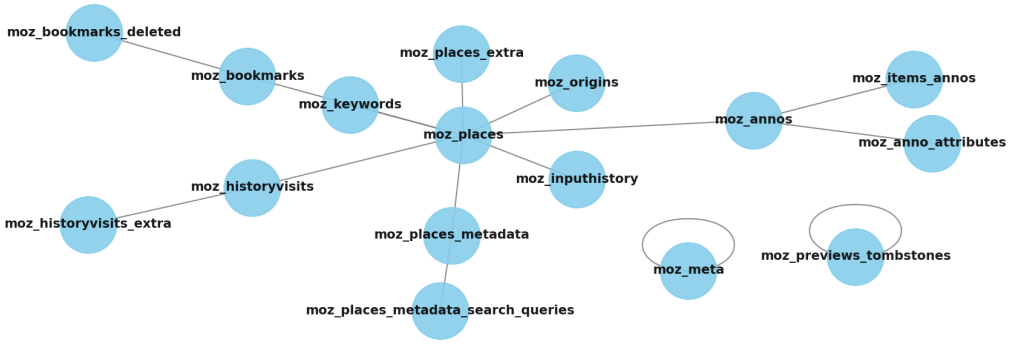
3.2 Spracovanie dát

Ďalším krokom je spracovanie dát. Tento krok predstavuje nosnú časť práce a je rozsiahly, nakoľko je nutné zadefinovať spojenia všetkých tabuliek v rámci každej databázy do jedného celku. Na to je potrebné identifikovať vzťahy medzi tabuľkami každej databázy a následne aj vzťahy medzi databázami samotnými, čo vyžaduje porozumenie jednotlivých atribútov v databázach.

Príklad identifikovaných vzťahov - databáza places.sqlite

Väčšina tabuliek tejto databázy sa dá spojiť operáciou LEFT JOIN k tabuľke "moz_places", ktorá je základnou tabuľkou databázy. Vzťahy medzi jednotlivými tabuľkami znázorňuje obrázok 3 , z ktorého je vidieť, že všetky tabuľky okrem dvoch sú priamo alebo nepriamo napojené na "moz_places".

Dve izolované tabuľky "moz_meta" a "moz_previews_tombstones" neobsahujú žiadne relevantné dáta, preto sa im ďalej nevenujeme. Taktiež vynecháme tabuľky "moz_items_annos", "moz_historyvisits_extra", "moz_places_extra" a "moz_places_metadata_search_queries", ktoré väčšinou neobsahujú žiadne dáta.



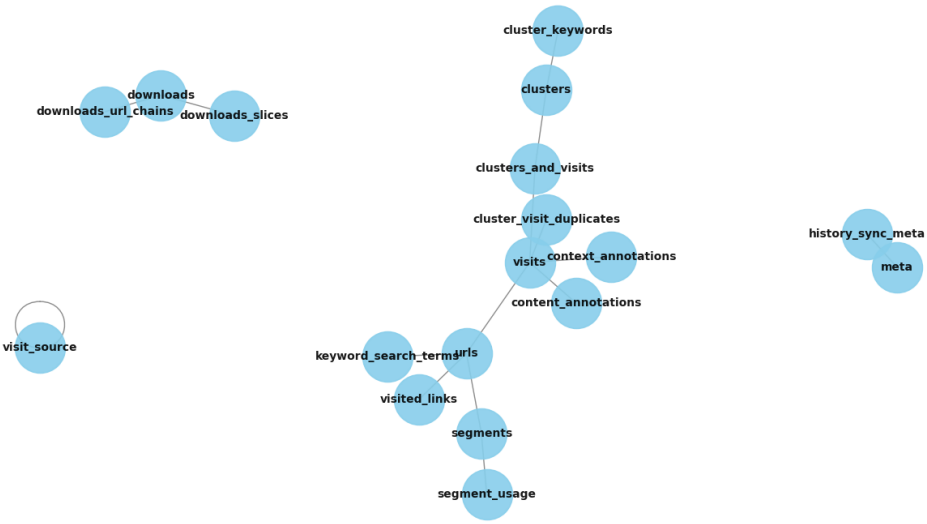
Obr. 3: Graf vzťahov medzi tabuľkami v Mozilla Firefox

Príklad identifikovaných vzťahov - databáza History

Databáza History prehliadača Google Chrome je o niečo komplikovanejšia. Vzťahy medzi tabuľkami tvoria nesúvislý graf, viď. obrázok 4. Kým väčšina tabuliek databázy je prepojená, tabuľky "meta" a "history_sync_metadata" sa ukazujú ako tabuľky bez priamych vzťahov k ostatným, na druhej strane ich dáta sú irelevantné. Taktiež tabuľka "visit_sources" nemá jednoznačné prepojenie, no nesie iba 2 atribúty - id a source, preto ju môžeme z úvah vylúčiť. Problémom je však dôležitá tabuľka "downloads", ktorá nie je nijak prepojená s hlavnou skupinou tabuliek. Má samostatný atribút "site_url", ktorý však nie je možné efektívne namapovať na atribút inej tabuľky. Predokladáme, že obdobné problémy sa ešte pri spájaní všetkých dát vyskytnú. Riešením je uložiť výsledky samostatných dopytov do dátových rámcov a spojiť dáta až pomocou knižnice Pandas, nakoľko je takéto spájanie prostredníctvom SQLite dopytov takmer neriešiteľné.

Postup spájania

Na samotné spojenie tabuliek budeme najprv čítať dáta z databázy pomocou knižnice sqlite3 v prostredí Python. To umožní uložiť dáta do rámca knižnice Pandas, s ktorým môžeme ďalej pracovať lepšie ako dopytovaním sa do databázy. Zároveň budeme redukovať duplicitné atribúty ako aj tie, ktoré sú z hľadiska forenzného vyšetrovania nerelevantné alebo neobsahujú



Obr. 4: Graf vzťahov medzi tabuľkami v Google Chrome

dostatok dát. Tieto informácie zisťujeme vypísaním rôznych štatistík (počet unikátnych hodnôt, maximum, minimum, najčastejšia hodnota a pod.) nad dátami a s využitím funkcie "columns.duplicated()" z knižnice Pandas. Týmto spôsobom získame relevantné dáta zo všetkých vybraných databáz. Tie potrebujeme ďalej prepojiť medzi sebou. V tejto fáze už väčšinou neexistuje atribút, na základe ktorého by sa dali dáta jednoducho spájať. Napríklad tabuľka moz_places a moz_formhistory nemajú žiadne spoločné atribúty. V histórii formulárov však máme atribút firstUsed a lastUsed, ktoré hovoria o tom, kedy boli hodnoty použité. Tieto časové pečiatky korelujeme s časovými pečiatkami "visit_date_15min". Pre pridanie dát z databázy cookies.sqlite zas potrebujeme identifikovať zhodu domény cookie záznamu a URL odkazu navštívenej stránky, pri čom jedna stránka môže mať uložených viacero cookies.

3.3 Agregácia a numerická reprezentácia dát

Najprv dáta agregujeme na základe časového okna. Napr. pri prehliadači Firefox agregujeme všetky dáta podľa atribútu "visit_date". Agregačná funkcia vytvorí zoznam všetkých záznamov, ktoré sú v 15-minútovom časovom

intervale, viď. obrázok 5.

```
# Round 'visit_date' to the nearest 15-minute interval
data['visit_date_15min'] = data['visit_date'].dt.floor('15T')

# Apply `list` to all columns except 'visit_date_15min'
aggregated_data = data.groupby('visit_date_15min').agg(
    {col: list for col in data.columns if col != '
        visit_date_15min'}
).reset_index()
```

Obr. 5: Ukážka agregačnej funkcie

Pre aplikovanie algoritmov strojového učenia na vyhľadávanie anomálií v dátach musíme ďalej previesť textové atribúty aj zoznamy numerických dát na ich numerickú reprezentáciu. Každý atribút je jedinečný a vyžaduje iný spôsob prevodu. V tabuľke 3 uvádzame príklady niektorých vybraných prevodov. Zoznam atribútov sa prevádza pomocou funkcií ako maximum, medián, počet prvkov, súčet a i. na jednu číselnú hodnotu, ktorá popisuje daný atribút pre jedno 15-minútové okno. Takto upravené dáta už môžeme dávať ako vstupy pre metódy hľadania anomálií. Je potrebné poznamenať, že prevod na číselné hodnoty je niekedy výhodné robiť už pri čítaní dát z databázy, resp. vo fáze ešte pred agregáciou, preto nové atribúty vznikajú počas celého procesu úpravy dát. Zároveň musíme zachovať niektoré pôvodné atribúty, ktoré nám neskôr umožnia spätne získať pôvodné nemodifikované dáta.

3.4 Identifikácia anomálií

Na pripravené dáta aplikujeme algoritmy identifikácie anomálií. Nakoľko naše dáta nie sú ohodnotené, využívať budeme metódy bez učiteľa. Do úvahy prichádza viacero metód, medzi ktoré radíme LOF, ECOD, iForest, COPOD, PCA, ROD a ďalšie.

LOF (Local Outlier Factor) je metóda, ktorá na identifikáciu bodov, ktoré sa javia ako anomálne, využíva vyhľadávanie najbližších susedov. LOF

Názov atribútu	**Funkcia**	**Nový atribút**
url	maximálna entropia	url_entropy_max
url	medián entropie	url_entropy_median
typed	medián	isTypedMedian
history_visit_id	dĺžka zoznamu	totalPageVisits
history_visit_id	dĺžka množiny	distinctPageVisits
downloads/destination FileURI	súčet nenulových prvkov	downloadedFiles
visit_type	1, ak zoznam obsahuje typ 1	TRANSITION_LINK _visit
visit_type	1, ak zoznam obsahuje typ 3	BOOKMARK_visit
formhistory.value	1, ak zoznam obsahuje výraz zo zoznamu podozrivých výrazov	isBlacklisted
formhistory.value	1, ak zoznam obsahuje výraz, v ktorom sa vyskytuje "mail"	isMailValue

Tabuľka 3: Numerická reprezentácia vybraných atribútov

sa odlišuje tým, že identifikuje body, ktoré sú odľahlé vzhľadom na lokálny zhluk bodov, nie na celkové dáta. [13]

ECOD (Empirical Cumulative Distribution Functions for Outlier Detection) využíva empirické kumulatívne distribučné funkcie na detekciu anomálií. Tento prístup vychádza z myšlienky, že anomálie budú mať výrazne odlišné distribučné vlastnosti v porovnaní s normálnymi dátami. [14]

COPOD (Copula-Based Outlier Detection) využíva matematický koncept kopúl na modelovanie multivariačnej distribúcie dát a identifikuje body, ktoré majú nízku pravdepodobnosť v tejto distribúcii ako anomálie. [15]

PCA (Principal Component Analysis) je metóda založená na rozklade matíc. Extrahuje dôležité informácie a reprezentuje ich ako hlavné komponenty. [16] Anomálie sa identifikujú ako body, ktoré sú významne vzdialené od hlavných komponentov v transformovanom priestore.

Ako jednu z možností pre hľadanie anomálií zvažujeme aj použitie **TODS (Automated Time-series Outlier Detection System)**. TODS je automatizovaný systém detekcie odchýlených hodnôt (outlierov) s využitím strojového učenia. Umožňuje spracovanie údajov, časových radov, analýzy (extrakcie) príznakov a i. TODS poskytuje širokú škálu algoritmov vrátane všetkých algoritmov bodovej detekcie podporovaných systémom PyOD, najmodernejších algoritmov vzorovej (kolektívnej) detekcie, ako sú DeepLog, Telemanon, a tiež rôznych algoritmov na vykonávanie systémovej detekcie.[17]

V tejto časti sa do značnej miery inšpirujeme aj existujúcou prácou s názvom "Detection of relevant digital evidence in the forensic timelines" [12], ktorá sa rovnako venuje identifikácii anomálií forenzných artefaktov, pričom je zameraná na dáta zo súborového systému ako takého. My sa pokúsime o niečo podobné s dátami z webových prehliadačov.

Pre identifikáciu najvhodnejších metód a ich parametrov najprv spúšťame všetky metódy s využitím rôznych parametrov. Najčastejšie upravovaným parametrom je tzv. contamination value, čiže hodnota udávajúca koľko percent dát je anomálnych. Tu skúšame nasledujúce hodnoty: 0.001, 0.002, 0.005, 0.01, 0.02, 0.05. Výnimkou je metóda LOF (ukážka implementácie viď. 6), ktorá na vstupe potrebuje počet najbližších susedov, ktorí

majú byť považovaní za anomáliu a vzdialenostnú metriku. V tomto prípade spúšťame LOF s počtom susedov 5, 10, 15, 20, 30 a 50 a 10 rôznych metrík. Globálnu premennú i používame na zaznamenávanie výsledkov všetkých metód a ich parametrov pre ďalšie porovnanie. Metódy iForest, LODA a iNNE majú ďalšie škálovateľné parametre, pri ktorých taktiež skúšame viacero hodnôt. Metóda ROD sa oproti ostantým ukázala ako veľmi pomalá aj pri testovaní jediného parametra, preto s touto metódou ďalej nepracujeme.

```
global i
for metric_value in ["braycurtis","canberra","chebyshev","
    cityblock","correlation", "cosine","euclidean","minkowski"
    ,"sqeuclidean","jaccard"]:
     for neighbors_value in [5,10,15,20,30,50]:
         for contamination_value in
             [0.001,0.002,0.005,0.01,0.02,0.05]:
              model1 = LocalOutlierFactor(n_neighbors =
                  neighbors_value, metric = metric_value,
                  contamination = contamination_value, n_jobs =
                  -1)
              y_pred = model1.fit_predict(df)
              outlier_index = np.where(y_pred == -1)

              outl_results.loc[i,"Method"] =  "LOF"
              outl_results.loc[i,"Parameters"] = str(
                  metric_value) +"_"+ str(neighbors_value) +"_"+
                   str(contamination_value)
              outl_results.loc[i,"Outl"] = outlier_index[0]
              outl_results.loc[i,"Outl_count"] = len(
                  outlier_index[0])
              i += 1
```

Obr. 6: Implementácia metódy LOF

Validáciu nájdených anomálií budeme vykonávať ohodnotením jednotlivých záznamov na základe dostupných popisov skúmaných datasetov, ako aj manuálnym preskúmaním najčastejšie identifikovaných anomálií. Pre výsledné anomálie budeme spätne dohľadávať pôvodné dáta, aby mal fo-

renzný analytik k dispozícii všetky informácie, nie len informácie pripravné pre metódy strojového učenia.

Ďalším plánovaným prístupom je do dát manuálne vkladať riadky, ktoré by mali byť vyhodnotené ako anomálne. Tieto riadky budeme pridávať do už spracovaných numerických dát a budeme simulovať podozrivú aktivitu používateľa, resp. útočníka. Tak bude možné jednoznačne odsledovať, či vložené anomálie budú použitými metódami naozaj identifikované.

Na základe týchto výsledkov odsledujme metódy a ich parametre, ktoré dosahujú najprijateľnejšie výsledky. Najdôležitejšie je odhaliť čo najviac anomálií. Na druhom mieste bude počet nesprávne určených anomálií, nakoľko nechceme generovať priveľa nesprávne určených výsledkov, ktoré by forenzný analytik musel ďalej ručne preskúmať.

3.5 Zobrazovanie dát

Dôležité je zamyslieť sa aj nad spôsobom zobrazovania dát. Do úvahy prichádzajú rôzne formy zobrazovania, ktorých implementáciu budeme zvažovať podľa výsledkov predchádzajúcich fáz. Prvým spôsobom je klasické zobrazovanie rámcov priamo v prostredí Jupyter Notebook. Nevýhodou je, že pri veľkom množstve dát sa rámce stávajú neprehľadnými. Preto budeme používať túto možnosť iba ako pracovné riešenie.

Ďalšou ideou je použiť Elasticsearch na uloženie dát a Kibanu na ich zobrazovanie. V tomto prípade je nevýhodou nutnosť implementovania tejto platformy. Prekonanie tejto prekážky by však umožnilo veľmi prehľadne dáta zobrazovať, filtrovať či dokonca generovať nad nimi upozornenia na základe nastavených pravidiel. Bolo by možné definovať napr. jednoduché pravidlo pre upozornenie na anomálne záznamy.

Ďalšou možnosťou je využiť pokročilejšie knižnice na zobrazovanie. Knižnica Bokeh umožňuje interaktívnu vizualizáciu dát s využitím HTML a JavaScriptu. Knižnica Dash dokáže zobrazovať (nie len) interaktívne tabuľky s možnosťou filtrovania.

Konkrétne dáta, ktoré budeme zobrazovať sú dáta prislúchajúce nájdeným anomáliám vrátane ich doplnenia o pôvodné informácie, o ktoré

sme v procese hľadania anomálií prišli (v dôsledku agregácie a prevodu na numerické hodnoty). Ďalej je to graf závislosti času a počtu navštívených stránok so zvýraznením anomálnych úsekov.

4 Záver

Tento článok uvádza do problematiky forenznej webovej analýzy, popisuje problémy a súčasné riešenia. Súčasťou sú teoretické poznaky o vybraných artefaktoch prehliadačov, ku ktorým v rámci diplomovej práce pribudnú ďalšie artefakty a ich popisy. Značná časť článku je venovaná postupom a návrhom ako pristúpiť k problematike analýzy webových artefaktov a ako ich vyhodnocovať. Zjednotením popísaných krokov (extrakcia, spracovanie dát, identifikácia anomálií, vizualizácia) by sme mali získať ucelený nástroj/skript, ktorý je možné použiť na dátach z operačného systému Windows. Nástroj dáta zjednotí, vyhodnotí nad nimi anomálie a prehľadne to znázorní. Z výsledku by malo byť na prvý pohľad jasné, kedy sa objavili prvé anomálne záznamy a ktoré ďalšie záznamy sú podozrivé. Vďaka tomu sa dokáže forenzný analytik veľmi rýchlo zorientovať v dátach, pozrieť identifikovaný časový úsek a sformulovať hypotézu ohľadom incidentu.

Literatúra

[1] Statista. Cyber incident response timeline u.s. 2023, 2023.

[2] Amir M. Hormozi. Cookies and privacy. *Information Systems Security*, 13:51–59, 2005.

[3] web.dev. Autofill, 2021.

[4] Ryan Benson. Chrome values lookup tables, 2 2019.

[5] Ryan Benson. Chrome transition values, 2014.

[6] Hyunmin Kim, In Seok Kim, and Kyounggon Kim. Aibft: Artificial intelligence browser forensic toolkit. *Forensic Science International: Digital Investigation*, 36:301091, 3 2021.

[7] Rebecca Nelson, Atul Shukla, and Cory Smith. Web browser forensics in google chrome, mozilla firefox, and the tor browser bundle. *Studies in Big Data*, 61:219–241, 2020.

[8] David Mugisha. Web browser forensics: Evidence collection and analysis for most popular web browsers usage in windows 10. *International Journal of Cyber Criminology*, 54, 2018.

[9] Vlaho Djurkovic, Mike Tomlinson, and Zheng Fang. Firefox concrete architecture.

[10] Digvijaysinh M Rathod and Digvijaysinh Rathod. Web browser forensics: Google chrome. *International Journal of Advanced Research in Computer Science*, 8, 2017.

[11] Hudan Studiawan and Ferdous Sohel. Anomaly detection in a forensic timeline with deep autoencoders. *Journal of Information Security and Applications*, 63:103002, 12 2021.

[12] Eva Markova, Pavol Sokol, and Kristina Kovacova. Detection of relevant digital evidence in the forensic timelines. *2022 14th International*

Conference on Electronics, Computers and Artificial Intelligence, ECAI 2022, 2022.

[13] Hoss Belyadi and Alireza Haghighat. *Chapter 4 - Unsupervised machine learning: clustering algorithms*, pages 125–168. Gulf Professional Publishing, 2021.

[14] Zheng Li, Yue Zhao, Xiyang Hu, Nicola Botta, Cezar Ionescu, and George H. Chen. Ecod: Unsupervised outlier detection using empirical cumulative distribution functions. *IEEE Transactions on Knowledge and Data Engineering*, 35:12181–12193, 12 2023.

[15] Zheng Li, Yue Zhao, Nicola Botta, Cezar Ionescu, and Xiyang Hu. Copod: Copula-based outlier detection. *Proceedings - IEEE International Conference on Data Mining, ICDM*, 2020-November:1118–1123, 11 2020.

[16] Hervé Abdi and Lynne J. Williams. Principal component analysis. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2:433–459, 7 2010.

[17] Kwei-Herng Lai, Daochen Zha, Guanchu Wang, Junjie Xu, Yue Zhao, Devesh Kumar, Yile Chen, Purav Zumkhawaka, Minyang Wan, Diego Martinez, and Xia Hu. Tods: An automated time series outlier detection system. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(18):16060–16062, May 2021.