Distribuované zpracování a analýza dat ze sociálních sítí

Návrh a implementace distribuované architektury

Technická zpráva FIT VUT v Brně

Ing. Radek Burget, Ph.D.

Technická zpráva č. FIT-TR-2018-07
Fakulta informačních technologií, Vysoké učení technické v Brně

Last modified: 10. prosince 2018

Distribuované zpracování a analýza dat ze sociálních sítí

Návrh a implementace distribuované architektury

Ing. Radek Burget, Ph.D.

Vysoké učení technické v Brně, email: burgetr@fit.vutbr.cz

Abstrakt Současné metody rekonstrukce a forenzní analýzy aktivity uživatelů se zaměřují prakticky výhradně na analýzu lokáních počítačů. Stále rostoucí popularita sociálních sítí ale způsobuje, že se stále více aktivit přesouvá právě na sociální sítě. Je proto nezbytné rozšířit analyzovanou časovou osu i o informace o aktivitách na sociálních sítích. V takovém případě je však nutné analyzovat velké množství zdrojů (např. profilů na sociálních sítích) a tedy získat a dále zpracovat velké množství dat. Je tedy nutné navrhnout škálovatelné řešení jak pro získávání, tak pro další analýzu dat ze sociálních sítí. V této technické zprávě se zaměřujeme na analýzu existujících modelů časové osy, existujících souvisejících softwarových řešení pro distribuované zpracování velkého množství dat a zejména na návrh architektury nástroje pro rekonstrukci a analýzu časové osy v distribuovaném prostředí. Rovněž se zabýváme implementací navrženého nástroje na platformě Apache Hadoop.

1 Úvod

Sociální sítě představují v současnosti velmi oblíbený a rozšířený způsob mezilidské komunikace umožňující veřejné nebo cílené sdílení a šíření různých typů obsahu. Tato skutečnost představuje nové výzvy z hlediska vyšetřování a forenzní analýzy. Zatímco dříve se značná část aktivity uživatelů odehrávala na lokálních počítačích, které bylo možno podrobit analýze a získat tak informace o těchto aktivitách, v současnosti se stále více aktivit přesouvá právě na sociální sítě a lokální počítač nebo jiné klientské zařízení je využíváno pouze pro zprostředkování přístupu k těmto sítím.

Existující přístupy k rekonstrukci minulých aktivit uživatele (rekonstrukci časové osy) se stále zaměřují zejména na analýzu lokálního počítače. Umožňují detekovat jak elementární události typu vytvoření a smazání souborů, tak i pokročilejší události jako otevření konkrétního souboru pomocí daného programu apod. Tyto nástroje však neumožňují sledovat další souvislosti, které se odehrávají mimo lokální počítač, tedy na sociálních sítích. Je tedy možné například identifikovat lokální soubory získané z internetu, není již ale žádná evidence o tom, jaká byla historie těchto souborů mimo lokální počítač, kdy byly publikovány na sociální síti, jaký je jejich prvotní zdroj apod.

V rámci projektu Tarzan se proto zabýváme integrací událostí ze sociálních sítí do časové osy. Cílem je shromáždit a analyzovat informace, které by mohly poskytnout odpovědi např. na následující otázky:

- Byl daný lokální soubor stažen ze sociální sítě? Pokud ano, z jakého konkrétního zdroje? (např. profilu)
- Jaká je historie tohoto souboru v rámci sledovaných profilů na sociálních sítích? Kdy byl soubor publikován a kým?
- Existuje souvislost (obsahová, časová, apod.) mezi aktivitou lokálního počítače a některých profilů na sociálních sítích? Lze z toho usuzovat, že např. uživatel počítače je současně vlastníkem daných profilů?

Za účelem zodpovězení těchto otázek proto navrhujeme rozšíření existujících řešení a datových modelů o možnosti analýzy událostí na sociálních sítích. To zahrnuje jednak samotné získání a reprezentaci dat ze sociálních sítí, tak i následnou analýzu umožňující nalezení souvislostí v modelovaných událostech.

Vzhledem k očekávanému velkému množství zdrojů na sociálních sítích, které je takto nutné analyzovat, je nepominutelným požadavkem škálovatelnost celého řešení spočívající v možnosti distribuovat jak získávání dat, tak jejich analýzu na velké množství výpočetních uzlů. V této technické zprávě se proto soustředíme na návrh distribuovaného řešení jak pro získávání, tak i pro analýzu dat. Tento výzkum navazuje na řešení navržené v loňském roce popsané v [4], které rozšiřuje právě o možnost distribuovaného zpracování dat a integraci se souběžně vyvíjenou platformou pro analýzu digitálních dat z bezpečnostních incidentů [21].

V kapitole 2 popisujeme blíže úlohu rekonstrukce časové osy, současný stav problematiky a existující datové modely a nástroje. V kapitole 3 zmiňujeme existující databázová řešení použitelná v distribuovaném prostředí. Kapitola 4 se zaměřuje na distribuované spouštění výpočetních úloh zejména s cílem získávání dat ze sociálních sítí, zatímco v kapitole 5 navrhujeme řešení distribuované analýzy získaných dat. Konečně kapitola 6 je věnována architektuře a uživatelskému rozhraní celého navrženého řešení.

2 Modelování dat souvisejících s využitím sociálních sítí

Zpracovávaná data související s využíváním sociálních sítí zahrnují dvě základní složky:

- Data dostupná přímo prostřednictvím sociálních sítí zahrnující jednotlivé profily a jejich obsah.
- Data dostupná na lokálních zařízeních související s využíváním sociálních sítí. V případě stolních počítačů se jedná zejména o data o využití webových prohlížečů pro přístup k sociálním sítím.

V obou případech je možno na tato data pohlížet jako na záznam elementárních *událostí*, které je zapotřebí dále analyzovat – odhalit jejich vzájemnou

korelaci a nalézt obecnější vzory [11]. Vzhledem k tomu, že zdroje i povaha těchto událostí jsou značně různorodé, je nejprve nutné navrhnout vhodný datový model, který umožní data o událostech z různých zdrojů jednotným způsobem reprezentovat, uložit a dále analyzovat. To umožňuje rekonstruovat *časovou osu* – posloupnost událostí podložených reálnými daty, která jsou k dispozici v uvažovaných datových zdrojích.

2.1 Rekonstrukce a modelování časové osy

Současné přístupy k rekonstrukci časové osy se zabývají téměř výhradně analýzou obrazů lokálních souborových systémů s důrazem na majoritní operační systém Microsoft Windows [7,8,11,28]. Nicméně jak ukážeme v dalších kapitolách, používané datové modely jsou aplikovatelné i v oblasti sociálních sítí.

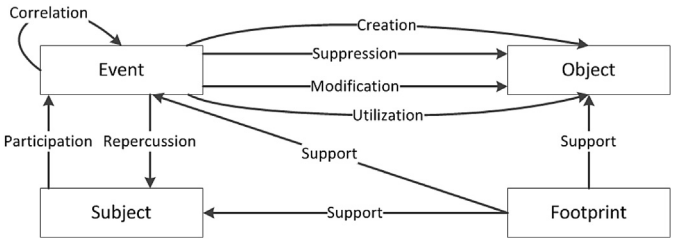
Obraz souborového systému obsahuje až miliony záznamů o nízkoúrovňových událostech souvisejících s využíváním počítače – vytvoření a změny jednotlivých souborů, změna údaje ve Windows Registry, záznamy o programech spuštěných v poslední době, záznamy o navštívených stránkách a dalších operacích v datech webových prohlížečů a mnohé další [11]. Základním cílem rekonstrukce časové osy je nalezení vzorů v těchto elementárních událostech, které lze interpretovat jako události vyšší úrovně, např. zapnutí, počítače, připojení USB přenosného disku, otevření konkrétního dokumentu v konkrétním programu, zveřejnění dokumentu na sociální síti a podobně.

Data související s využitím počítače lze tedy modelovat na více úrovních:

- Surová data získaná ze zdrojového souborového systému pomocí různých nástrojů (nejběžnější z nich zmiňujeme v kapitole 2.2).
- Elementární události přímo zachycené v datech (*ingestion facts* [28], *low-level events* [23])
- Odvozené události získané další analýzou elementárních událostí (*inferences* [28], *higher-level events* [23]).

Základní datový model pro reprezentaci časové osy je podrobně rozpracován např. v [7]. Jak je patrné z obrázku 1, modelovaná data obecně popisují *události* (*Event*) iniciované nějakým *subjektem* (např. uživatelem počítače) a týkající se příslušných *objektů* (např. souborů, dokumentů) v konkrétním čase. Informace o těchto entitách jsou podloženy reálnou *stopou* (*Footprint*) v souborovém systému. Přitom množství informací, které jsou dostupné o subjektech a objektech se může výrazně lišit podle typu události. Úkolem dalšího odvozování je proto mimo jiné doplnění chybějících informací například pomocí časové korelace jednotlivých událostí [8]. K elementárním subjektům a objektům mohou být rovněž k dispozici různé druhy doplňujících informací, jako např. jméno a typ daného dokumentu, přihlašovací jméno u uživatele apod., které je třeba do modelu rovněž zahrnout.

Jelikož obecně platí, že jak jednotlivé subjekty, tak objekty mohou figurovat ve větším počtu událostí, je třeba modelovat i shodu a souvislosti mezi jednotlivými subjekty a objekty, které se jsou zapojeny v událostech. Z toho důvodu



Obrázek 1. Základní datový model pro popis událostí [7]

vede modelování souvisejících dat typicky na grafový popis, který zachycuje jednotlivé subjekty a objekty, jejich role v rámci modelovaných událostí a další doplňující vztahy a informace.

Typický grafový přístup představuje např. Schelkoph a kol. [23]. Popsaný přístup je založen na grafové databázi *Neo4j* [9] založené na grafovém modelu *labeled property graph*. Tento model umožňuje ukládání jednoduchých dat (tzv. vlastností) přímo v uzlech grafu, což zvyšuje kompaktnost výsledného modelu. Databáze Neo4j je určena zejména k rychlému a efektivnímu procházení grafů, což umožňuje reprezentovat a analyzovat velké množství databází.

Výrazně rozšířenější v oblasti forenzní analýzy je využití ontologického popisu a sémantických technologií RDF [15] a OWL [5]. Typickými představiteli tohoto přístupu jsou Forensic of Rich Events (FORE) [22], Digital Forensics Analysis Expression (DFAX) [6] nebo Ontology for the Representation of Digital Incidents and Investigations (ORD2I) [8]. Obdobný popis používá i nástroj TimelineAnalyzer vyvíjený v rámci projektu Tarzan [4] zaměřený na analýzu událostí ze sociálních sítí. Grafový model RDF vytvářející graf z elementárních trojic subjekt – predikát – objekt je obecně méně kompaktní, něž výše zmíněný labeled property graph, na druhou stranu však využití ontologií jako sdílených konceptuálních modelů umožňuje snazší integraci takto popsaných dat s dalšími datovými sadami a následné dotazování přes tzv. propojená data (*linked data*). Současně se jedná o standardizované technologie, které jsou podporovány řadou existujících softwarových řešení.

Oba zmíněné grafové modely umožňují popsat informace získané analýzou lokálního souborového systému. Model typu *labeled property graph* je obecně kompaktnější a přímo navržený pro zpracování rozsáhlých dat. Na druhou stranu ontologické modely nabízí propracovaný formální základ a umožňují snazší integraci jak existujících konceptuálních popisů dat tak i vlastních datových sad, přičemž i v této oblasti existují nástroje umožňující pracovat s daty velkého rozsahu [29]. V návrhu řešení popsaného v této technické zprávě jsme proto použili ontologický model, který podrobněji popisujeme v kapitole 2.3.

Aby bylo možné sestavit grafový model popisující konkrétní události a související informace, je třeba nejprve získat surová data o těchto událostech z příslušných zdrojů – souborových systémů nebo sociálních sítí. Samotný proces analýzy

souborového systému a extrakce informací o jednotlivých nalezených událostech je realizován samostatnými softwarovými nástroji.

2.2 Nástroje pro extrakci dat o událostech

Pro extrakci dat o událostech z lokálního souborového systému případně ze sociálních sítí existuje řada softwarových nástrojů. Úkolem takového nástroje je procházení relevantních částí souborového systému nebo komunikace s aplikačním rozhraním sociální sítě a poskytnutí dat o nalezených událostech ve strukturované podobě. Výše zmíněné analytické nástroje následně z těchto dat budují příslušný grafový model, který lze dále analyzovat.

Pomineme-li uzavřené komerční nástroje, které nelze jednoduše integrovat do většího celku, většina současných přístupů využívá následující běžně dostupné analytické nástroje: [8,11,28]

- **log2timeline/plaso**[1] [10] – obecný nástroj pro extrakci časových údajů z různých souborů ve zkoumaném souborovém systému a jejich agregaci. Tento nástroj je schopen projít souborový systém extrahovat události z více než 100 podporovaných souborových a databázových formátů. Kromě systému Windows podporuje i další běžné systémy: Linux, Mac OS X a Android. Výsledkem procházení jsou údaje obvykle o velkém množství nalezených událostí, které jsou uloženy ve speciálním datovém souboru. Následně jsou k dispozici nástroje umožňující další procházení a filtrování získaných dat a jejich export do několika formátů umožňujících další zpracování. Z takto získaného výstupu lze potom konstruovat ontologický model [8].
- **TEAR** (Temporal Event Abstraction and Reconstruction) extraktor – extrakční nástroj vytvořený v rámci stejnojmenného projektu [17,23].
- Mnohé další obdobné nástroje byly vytvořeny v rámci výzkumných projektů, jako např. PyDFT [11], CyberForensic TimeLab [18].

Pokud jde o získání událostí ze samotných sociálních sítí, je výběr použitelných nástrojů značně omezený. Tento stav je dán zejména skutečností, že jednotlivé sociální sítě mají značně odlišná aplikační rozhraní, které umožňují získávat rozdílné informace s různou mírou detailu a přístup k těmto aplikačním rozhraním je často různými způsoby omezen [4]. Proto se existující nástroje zaměřují na jednu sociální síť (např. Facebook [13,16] nebo Twitter [12]), případně existují pouze jako akademická studie [19].

V rámci projektu jsme proto navrhli a implementovali vlastní nástroj TimelineAnalyzer [4], jehož rozšiřitelná architektura umožňuje extrahovat informace o událostech z různých sociálních sítí. Získaná data jsou pak reprezentována opět s využitím ontologického modelu.

2.3 Ontologické modelování forenzních dat

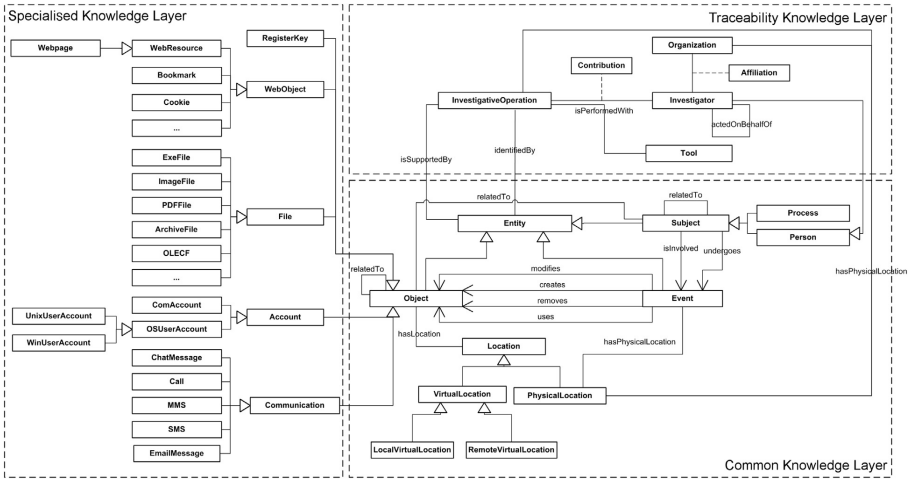
Ontologie představují způsob, jak vytvořit formální konceptuální model dané domény, který je explicitně popsán a je možno jej sdílet mezi aplikacemi [27].

[1] https://github.com/log2timeline/plaso

Důležitou vlastností ontologií je jejich opakovaná použitelnost a možnost kombinace většího počtu ontologií pro popis sémantiky dat.

Ontologie používané v oblasti popisu časové osy pro účely forenzní analýzy jsme v krátkosti zmínili v kapitole 2.1. Zejména ontologie ORD2I [8] představuje podrobně rozpracovanou konceptualizaci problému v několika vrstvách, jak je patrné z obrázku 2:

1. **Vrstva základních znalostí** (Common Knowledge Layer) definuje pojmy odpovídající základnímu modelu na obr. 1 popsanému výše.
2. **Vrstva speciálních znalostí** (Specialized Knowledge Layer) doplňuje definici konkrétnějších pojmů, které mohou figurovat jako objekt v událostech – soubory různých typů, webové objekty, uživatelské účty a další.
3. **Vrstva sledovatelnosti** (Traceability Knowledge Layer) definuje pojmy související s průběhem vlastního vyšetřování.

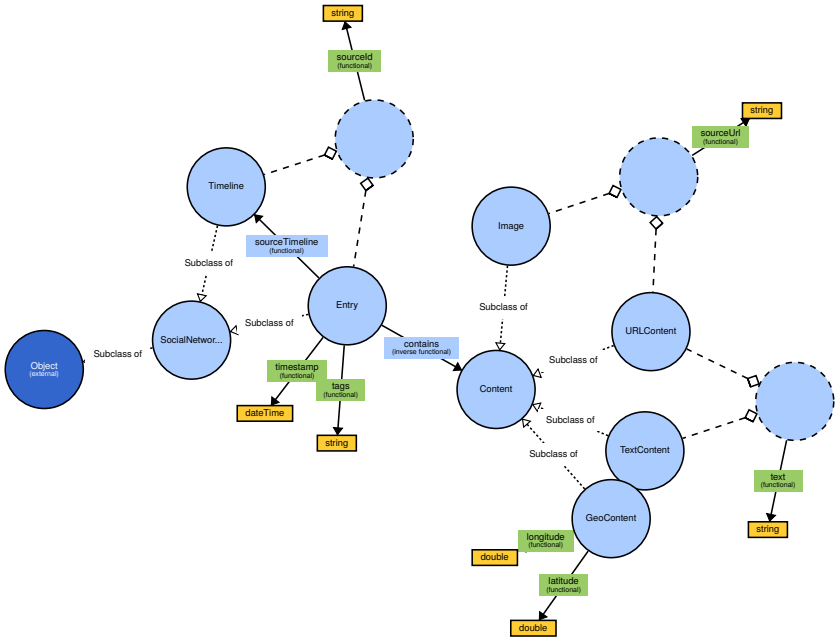


Obrázek 2. Ontologie ORD2I [8]

Celá ontologie je formálně definována pomocí jazyka OWL a umožňuje konstruovat tvrzení ve formě RDF trojic, které ve výsledku vytvářejí graf popisující časovou osu. Na rozdíl od přístupu využívajícího databázi Neo4j je v RDF každé tvrzení vázáno na ontologický model domény a má tedy formálně definovanou sémantiku.

Obdobně náš nástroj TimelineAnalyzer reprezentuje data o příspěvcích na sociálních sítích pomocí vlastní ontologie popsané v [4]. Protože domény modelované pomocí ontologií ORD2I a TimelineAnalyzer se do značné míry překrývají, upravili jsme následně ontologii TimelineAnalyzer tak, aby bylo možno současně využít již definované obecnější koncepty z ontologie ORD2I. Konkrétně ontologie

TimelineAnalyzer rozšiřuje ontologii ORD2I o nový typ objektů *SocialNetworkObject* a jeho konkrétnější varianty *Timeline* a *Entry*, jak je patrné z obrázku 3. To umožňuje efektivně reprezentovat vztahy mezi lokálními událostmi popsanými pomocí ORD2I a událostmi získanými ze sociálních sítí.



Obrázek 3. Ontologie TimelineAnalyzer

3 Databázová řešení pro zpracování rozsáhlých dat

V rámci projektu Tarzan byla navržena společná platforma pro zpracování rozsáhlých dat založená na technologii Apache Hadoop a souvisejících technologiích [21]. Vzhledem ke své rozšiřitelnosti a široké škále existujících řešení založených na technologii Hadoop představuje tato platforma rovněž vhodný pro zpracování rozsáhlých dat souvisejících s využitím sociálních sítí. V případě vyvíjeného nástroje TimelineAnalyzer je třeba řešit ukládání grafových dat ve formátu RDF popsaných v předchozí kapitole a následné dotazování nad těmito daty za účelem jejich další analýzy. Obě tyto úlohy jsou implementovány v distribuovaném prostředí vzhledem k uvažovanému rozsahu zpracovávaných dat.

K ukládání RDF dat na platformě Hadoop se využívá několik vzájemně provázaných řešení realizujících jednotlivé logické vrstvy ukládání dat. Jedná se o následující technologie:

1. **Apache HDFS** (Hadoop Distributed File System)[2] – distribuovaný souborový systém realizující ukládání datových souborů a jejich sdílení v rámci výpočetních uzlů.
2. **Apache HBase** [3] – distribuovaná sloupcová databáze určená pro distribuované zpracování rozsáhlých dat. Využívá HDFS pro ukládání dat jednotlivých tabulek.
3. **Halyard** [4] – úložiště RDF dat vybudované nad databází HBase realizující transformaci RDF dat na tabulky sloupcové databáze a distribuované vyhodnocování dotazů nad grafovými daty.

Nástroj TimelineAnalyzer následně využívá tuto kompozici technologií pro ukládání a dotazování dat ze sociálních sítí prostřednictvím aplikačního rozhraní Halyard, jak je popsáno v dalších kapitolách.

3.1 Ukládání dat na platformě Hadoop

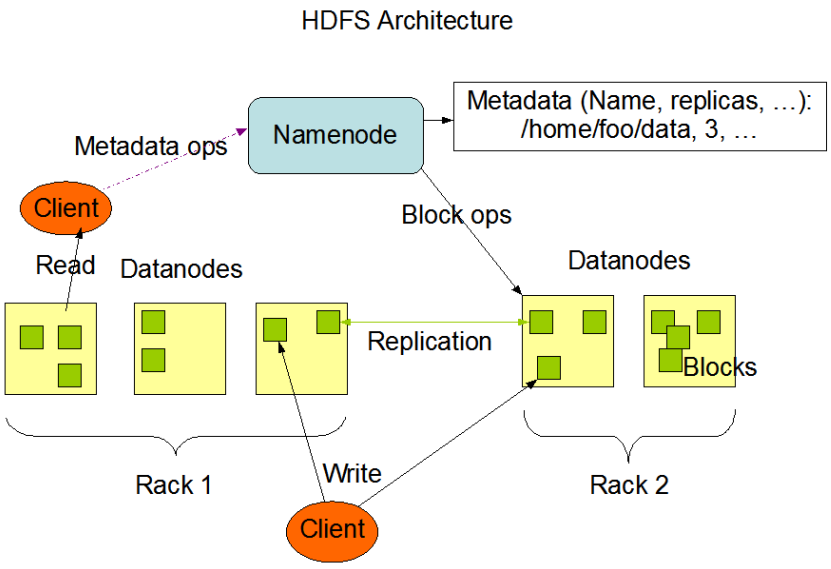
Standardní technologie platformy Hadoop zahrnují distribuovaný souborový systém HDFS pro ukládání souborů a několik sloupcových databází pracujících nad tímto souborovým systémem.

Hadoop Distributed File System – HDFS Souborový systém HDFS je založen na architektuře *master–slave* [30]. Jeho architektura je znázorněna na obrázku 4. HDFS cluster se skládá z jednoho uzlu zvaného *Name Node* (jmenný uzel), který udržuje informace o jménech uložených souborů a další metadata nutná pro distribuci dat jednotlivých souborů a několika uzlů *Data Node* (datové uzly), které realizují fyzické ukládání souborů, nebo jejich částí. Klientská aplikace prostřednictvím jmenného uzlu získává informace o tom, na kterých datových uzlech jsou fyzicky uloženy jednotlivé části požadovaného souboru a následně získává konkrétní obsah souboru přímou komunikací s příslušnými datovými uzly. Datové uzly rovněž komunikují mezi sebou, což umožňuje realizovat replikaci uložených dat za účelem zvýšení spolehlivosti a zrychlení přístupu k datům.

Jednotlivé soubory jsou pro ukládání rozděleny na bloky o velikosti nejčastěji 128 MB, které jsou distribuovány na jednotlivé datové uzly a replikovány. Ve standardním nastavení je každý blok replikován právě třikrát. [1].

HDFS je určen pro uložení obecně dále nestrukturované informace, která je následně paralelně zpracovávána výpočetními uzly. V případě, že chceme uložit strukturovanou informaci ve formě tabulek, lze využít některé ze sloupcových databází pracujících nad HDFS.

[2] https://wiki.apache.org/hadoop/HDFS/
[3] https://hbase.apache.org/
[4] https://merck.github.io/Halyard/



Obrázek 4. Architektura HDFS [1]

Sloupcové databáze nad HDFS Mezi nejčastěji využívané sloupcové databáze, které využívají HDFS jako úložiště dat, patří Cassandra[5] a již zmíněná HBase. Obě řešení využívají podobný přístup k reprezentaci dat pomocí širokých sloupců (*wide column*), liší se však řadou architektonických detailů [3]. Vzhledem k tomu, že další vrstvy výše popsaného řešení využívají databázi HBase, zaměříme se podrobněji na toto řešení.

HBase reprezentuje uložená data pomocí tabulek (*table*), které se skládají z buněk (*cell*). Buňky jsou organizovány do řádků (*rows*), přičemž každý řádek je identifikován jednoznačným klíčem (*row key*) a do sloupců, kde sloupec je identifikován kvalifikátorem (*qualifier*). Sloupce lze sdružovat do rodin (*column family*). Jednotlivé řádky tabulky jsou ukládány vždy tak, že jsou lexikograficky uspořádány podle hodnoty klíče a rozděleny do souvislých *regionů*, které jsou následně ukládány do souborů na HDFS. To umožňuje efektivní nalezení konkrétního řádku a při vhodně zvoleném způsobu generování klíčů i efektivní procházení souvislých datových regionů.

Architektura systému HBase využívá centrální uzel *HMaster*, který spravuje metadata tabulek a zajišťuje distribuci dat a větší počet uzlů regionů (*region nodes*), které zajišťují fyzické uložení konkrétního regionu – souvislé oblasti dat. Uzly regionů realizují vlastní ukládání dat na HDFS, přičemž se snaží o fyzické uložení dat na nejbližší datový uzel HDFS. Vhodnou metodou generování *row*

[5] https://cassandra.apache.org/

key je tedy možno zajistit rovnoměrnou distribuci dat na regiony a tím i efektivní distribuované zpracování uložených dat.

3.2 Distribuované grafové databáze

Pro ukládání grafových dat ve formátu RDF je možno využít některého z existujících RDF úložišť. Mezi nejvíce používaná úložiště patří např. RDF4J[6] (dříve Sesame), Blazegraph[7] nebo Virtuoso[8]. Experimentálně lze rovněž využít grafovou databázi Neo4j pro ukládání RDF dat, přestože její datový model je poněkud odlišný [2]. Všechna zmíněná úložiště disponují aplikačním rozhraním, které umožňuje aplikacím číst a vkládat data do úložiště a podporují vyhodnocování dotazů v některém dotazovacím jazyce, nejčastěji SPARQL [20].

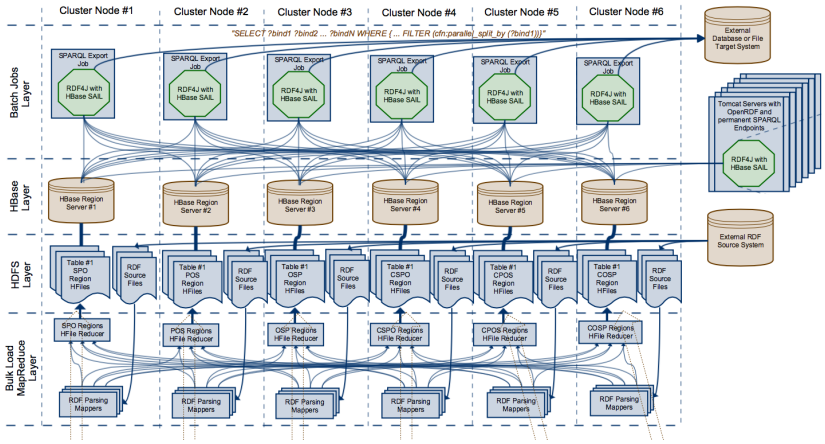
Žádné z těchto úložišť však není přímo integrováno s platformou Hadoop z hlediska ukládání zpracovávaných dat. Některá z těchto úložišť (Neo4j, Blazegraph) podporují vytváření vlastních clusterů pro škálování na více uzlů a v případě Neo4j jsou k dispozici softwarové konektory umožňující jeho využití jako zdroje dat pro Hadoop, nicméně vlastní úložiště tvoří vždy samostatný celek mimo vlastní Hadoop cluster.

Projekt Halyard oproti tomu představuje možnost jak ukládat RDF data přímo v Hadoop clusteru s využitím databáze HBase a souborového systému HDFS [24]. Je založen na existujícím systému RDF4J, který rozšiřuje právě o možnost ukládání dat v distribuovaném prostředí Hadoop. Dále obsahuje nástroje pro dávkové vkládání velkých objemů dat do úložiště (*bulk load*) a implementuje paralelní vyhodnocování dotazů v jazyce SPARQL na více výpočetních uzlech. Architektura celého řešení je patrná z obrázku 5.

Aby bylo dosaženo maximální efektivity při čtení uložených dat, jsou ukládané RDF trojice uloženy v tabulce HBase redundantně v šesti regionech. Z hodnoty subjektu (S), predikátu (P) a objektu (O) trojice, případně kontextu (C), se vypočte jednoznačná mapovací hodnota (hash), která se následně použije jako klíč řádku v tabulce HBase. Každá hodnota se uloží šestkrát s různým pořadím jednotlivých částí: SPO, POS, OSP, atp. To umožňuje efektivní nalezení trojic odpovídajících různým dotazům nad databází. Například chceme-li získat všechny trojice s nějakou hodnotou predikátu P, vypočtením příslušné hodnoty nalezneme v regionu POS přímo první řádek, který tomuto dotazu odpovídá a všechny další odpovídající trojice jsou uloženy v řádcích, které bezprostředně následují. Obdobně lze postupovat při vyhledávání pomocí jiné hodnoty. Kompletní hodnota celé trojice je využita jako kvalifikátor sloupce databáze, přičemž hodnota uložená v tomto sloupci není podstatná. Toto řešení elegantně řeší situace, kdy pro dvě různé trojice dostaneme stejnou hodnotu hash.

Kvůli výše popsaným vlastnostem a zejména pro snadnou integraci s existující platformou využívající Apache Hadoop jsme se v projektu TimelineAnalyzer

[6] http://rdf4j.org/
[7] https://www.blazegraph.com/
[8] https://virtuoso.openlinksw.com/



Obrázek 5. Architektura úložiště Halyard [24]

rozhodli využít Halyard jako primární úložiště dat souvisejících s využitím sociálních sítí. To umožňuje realizovat dávkové plnění datové úložiště z různých zdrojů a následně spouštět analytické dotazy nad databází, jak popisujeme v dalších kapitolách.

4 Distribuované získávání dat ze sociálních sítí

Jednotlivé sociální sítě, jako např. Twitter nebo Facebook, disponují obvykle webovým aplikačním rozhraním, které umožňuje aplikacím třetích stran do jisté míry získávat informace publikované v rámci uživatelských profilů. Přístup k těmto informacím může být různým způsobem omezen v závislosti na nastavení soukromí jednotlivých uživatelů sociálních sítí. Podrobný přehled aplikačních rozhraní jednotlivých sociálních sítí a souvisejících omezení jsme zpracovali v [4].

Při rekonstrukci a analýze časových řad je třeba počítat se zpracováním dat z velkého množství (řádově tisíců) profilů na sociálních sítích. Každý z těchto profilů může obsahovat velké množství příspěvků v závislosti na uvažovaném časovém rozmezí a publikační aktivitě vlastníka profilu, přičemž jednotlivé příspěvky mohou obsahovat nejen textové informace, ale i obrazový nebo multimediální obsah. Pouhé stažení dat ze všech těchto profilů proto představuje časově poměrně náročnou úlohu a je tedy vhodné uvažovat nad jeho paralelizací.

Celý proces lze rozdělit na dvě části, které je třeba paralelizovat:

– **Stahování dat** prostřednictvím aplikačního rozhraní sociálních sítí. Zde je třeba implementovat softwarové moduly komunikující s aplikačním rozhraním jednotlivých uvažovaných sociální sítích a které mohou být současně distribuovány na jednotlivé uzly výpočetního clusteru.

– **Ukládání dat** do RDF úložiště Hallyard. Zde může být úzkým hrdlem dotazovací rozhraní serveru Halyard. Je třeba zajistit, aby získaná data neproudila do úložiště jedním úzkým místem na jednom výpočetním uzlu, ale aby byla skutečně paralelně ukládána.

Výsledný produkt je k dispozici jako samostatná aplikace Socializer[9] [29]. Tato aplikace řeší obě úlohy zmíněné výše. Pro vlastní připojení k webovému aplikačnímu rozhraní sociálních sítí jsme využili existující klientské moduly implementované v rámci řešení projektu Tarzan v minulém roce [4]. V současnosti jsou k dispozici moduly pro sociální sítě Facebook a Twitter. Tyto moduly byly doplněny o společné aplikační rozhraní a výsledkem jejich činnosti jsou data ve formátu RDF odpovídající ontologiím popsaným v kapitole 2.3.

Druhou část představuje databázový klient, jehož úkolem je ukládání získaných dat do RDF úložiště. Abychom eliminovali potenciální úzké hrdlo spočívající v dotazovacím rozhraní Halyardu, databázový klient ukládá data přímo do příslušné tabulky databáze HBase ve tvaru, který Halyard následně dokáže zpracovat. K tomu jsou využity nástroje *bulk load* [26], které jsou součástí Halyardu a původně jsou určeny pro rychlé vkládání rozsáhlých datových sad ze souborů do databáze. V rámci aplikace Socializer jsme doplnili možnost propojení s klientskými moduly sociálních sítí.

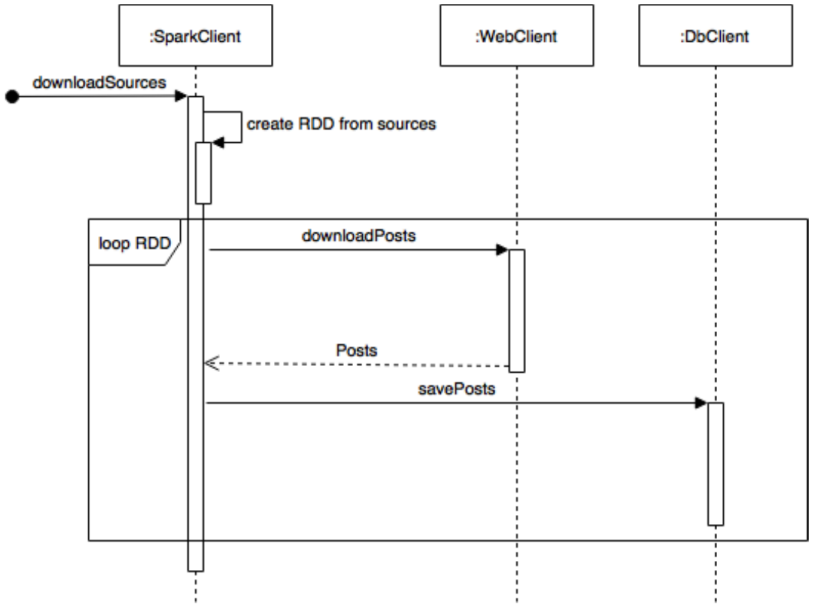
Propojení klientských komponent sociálních sítí a databázového klienta zajišťuje aplikace, která využívá platformu Spark[10]. Ta umožňuje distribuovat prováděnou úlohu na libovolný počet výpočetních uzlů. Spark je kompatibilní s Apache Hadoop a je součástí existující obecné platformy Tarzan [21]. Aplikace implementuje distribuovaný proces realizující stahování a následně ukládání dat, jak je naznačeno na obr. 6.

Vstupem celé úlohy získávání dat je seznam zdrojů – textový soubor obsahující jména profilů na sociálních sítích Twitter a Facebook. Tento vstupní soubor je uložen v souborovém systému HDFS. Pomocí nástrojů platformy Spark je vstupní soubor načten a reprezentován pomocí speciální datové struktury *Resilent Distributed Dataset* (RDD) [14], která reprezentuje kolekci profilů ke stažení. Spark poté vytvoří pro každý z těchto profilů samostatnou úlohu a tyto úlohy distribuuje na jednotlivé výpočetní uzly Spark clusteru. Každá z takto distribuovaných úloh provede kontaktování příslušné sociální sítě, stažení požadovaného profilu a uložení získaných dat do úložiště HBase. Výsledkem je paralelní stažení a uložení dat z požadovaných profilů kde stupeň paralelizace je dán počtem dostupných výpočetních uzlů.

5 Další analýza získaných dat

Distribuované stahování profilů ze sociálních sítí umožňuje naplnit datové úložiště surovými daty, která reprezentují *elementární události*, jak jsou popsány

[9] https://github.com/nesfit/socializer
[10] https://spark.apache.org



Obrázek 6. Výpočet Spark klienta v aplikaci Socializer [29]

v kapitole 2.1. Jednotlivé zveřejněné a stažené příspěvky tedy odpovídají základním modelovaným událostem. Ty jsou modelovány pomocí vrstvy *základních znalostí* použité ontologie, jak byla popsána v kapitole 2.3.

Další analýzu těchto dat lze rozdělit na dva kroky:

1. Automatické odvození událostí vyšší úrovně (*higher-level events*) na základě předem známých vzorů.
2. Následná analýza a vyhodnocení uživatelem (expertem).

Využití ontologického modelu a RDF úložiště pro reprezentaci dat umožňuje s výhodou využít dotazovací jazyk SPARQL pro nalezení známých vzorů v datech. Tento jazyk umožňuje efektivní nalezení uzlů a podgrafů v celkovém grafu na základě libovolných vzorů a následně i vkládání nových RDF trojic do grafu na základě nalezených informací.

V souvislosti s rekonstrukcí časové osy mohou být typické např. následující úlohy:

- Které příspěvky v rámci všech profilů sdílí stejná URL?
- Byl konkrétní obrázek nahrán nebo stažen pomocí daného webového prohlížeče?

Při použití ontologie TimelineAnalyzer (prefix `ta`), lze např. vyhledání sdílených URL v příspěvcích zapsat pomocí jazyka SPARQL takto:

```
CONSTRUCT { ?c1 ta:sameURL ?e2 }
  WHERE {
    ?c1 rdf:type ta:URLContent .
    ?c2 rdf:type ta:URLContent .
    ?c1 ta:sourceUrl ?url .
    ?c2 ta:sourceUrl ?url .
    ?e1 ta:contains ?c1 .
    ?e2 ta:contains ?c2 .
    FILTER ( ?e1 != ?e2 )
  }
```

Dotaz vyhledá příspěvky `e1` a `e2`, které obsahují obsah `c1` respektive `c2`, přičemž v obou případech jde o obsah typu URL, který se odkazuje na stejné cílové URL (`?url`). Pro každou takto nalezenou dvojici se vytvoří nový predikát `sameURL`, který reprezentuje fakt, že příspěvek `c1` se odkazuje na stejné URL, jako obsah `e2`.

Obdobně nalezení všech obrázků zveřejněných v profilech a jejich zdrojových URL lze zapsat takto:

```
SELECT ?img ?url
  WHERE {
    ?img ta:sourceUrl ?url .
    ?img rdf:type ta:Image
  }
```

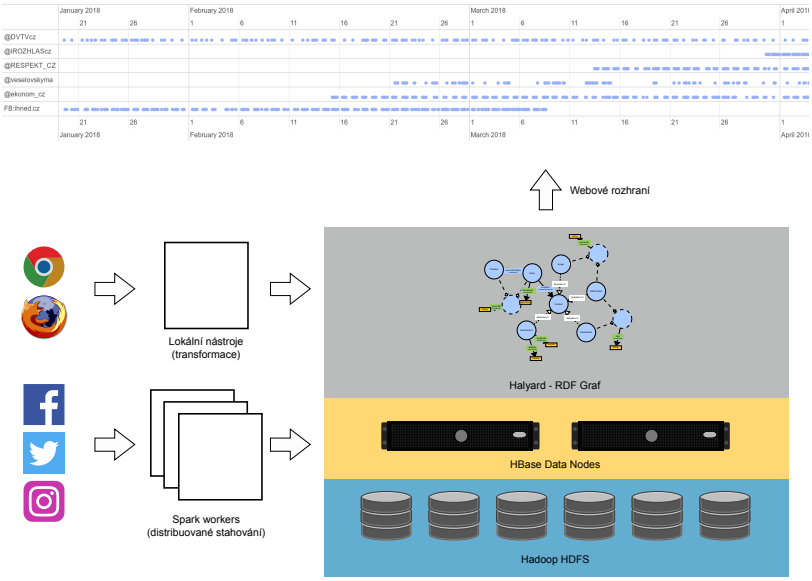
Nalezená URL je následně možno srovnat se všemi existujícími URL obdobně jako v předchozím příkladu a případně vložit nové RDF trojice do grafu. Tímto způsobem obohatíme výsledný graf o explicitní hrany mezi souvisejícími událostmi podle uvedených vzorů.

Uvedené dotazy v části `WHERE` specifikují omezení na tvar hledaných podgrafů v celkovém grafu. Při předpokládané značné velikosti celého grafu může nalezení všech takový podgrafů časově velmi náročné. Zvolené úložiště Halyard však umožňuje vyhodnocení jednotlivých omezení opět paralelizovat pomocí Hadoop clusteru [25] a výsledky našich experimentů ukazují, že v porovnání s centralizovaným vyhodnocením na jednom uzlu dosahuje výrazně vyššího výkonu [29].

Další krok představuje analýza získaných a odvozených událostí člověkem – expertem. K tomuto účelu bylo vytvořeno nové grafické uživatelské rozhraní systému TimelineAnalyzer, které popisujeme v další kapitole.

6 Výsledná architektura řešení

Prvky distribuovaného získávání a analýzy informací ze sociálních sítí popsané v předchozích kapitolách byly zařazeny do existujícího nástroje TimelineAnalyzer. Oproti předchozí verzi popsané v [4] proto byly provedeny změny v architektuře s cílem nasadit nástroj v distribuovaném prostředí Hadoop. Výsledná architektura je patrná z obrázku 7.



Obrázek 7. Architektura distribuované verze systému TimelineAnalyzer pro sběr a analýzu dat ze sociálních sítí

Jádrem systému je distribuované úložiště implementované v několika vrstvách pomocí technologií HDFS, HBase a Halyard, jak bylo popsáno v kapitole 3. Toto úložiště je plněno daty z různých zdrojů pomocí samostatných modulů. Moduly realizující získávání dat lze rozdělit na dvě skupiny:

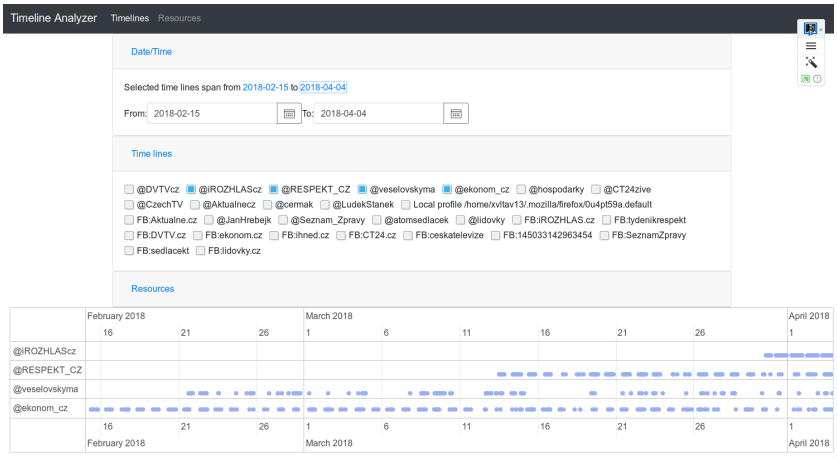
- Moduly získávající data analýzou lokálního souborového systému – jedná se zejména o analýzu lokálních profilů webových prohlížečů, jak bylo popsáno v [4].
- Moduly pro stahování dat ze sociálních sítí prostřednictvím jejich aplikačního rozhraní – tyto moduly byly nahrazeny novou distribuovanou verzí implementovanou v rámci samostatného softwarového projektu Socializer, jak je popsáno v kapitole 4. Získaná data jsou ukládána přímo do úložiště HBase, což zaručuje maximální propustnost.

Návrh systému TimelineAnalyzer počítá s možným přidáváním dalších modulů pro vstup dat v budoucnosti podle potřeby.

Jako poslední část bylo implementováno nové webové rozhraní, které umožňuje uživateli spouštění automatických analytických kroků popsaných v kapitole 5 a následné procházení nalezených událostí a jejich vyhodnocení. Toto analytické rozhraní komunikuje s úložištěm Halyard pomocí dotazů v jazyce SPARQL. Halyard provádí vyhodnocení těchto dotazů nad uloženými daty. Zejména je možno:

– Procházet graficky zobrazenou časovou osu událostí a filtrovat podle zdroje, času a dalších kritérií, jak je znázorněno na obrázku 8.
– Sledovat posloupnost událostí týkající se konkrétního souboru nebo dokumentu. Na obrázku 9 je vidět posloupnost událostí, která vedla k uložení lokálního souboru (fotografie) původně zveřejněného na sociální síti. Lze vysledovat čas zveřejnění a zobrazení a stažení souboru pomocí lokálního prohlížeče.

Zobrazení časové osy v části *Timelines* umožňuje výběr zdrojů pro zobrazení (profily na sociálních sítích nebo lokální profily), filtrování podle zadaného časového rozmezí a interaktivní procházení časové osy s možností zobrazení detailů jednotlivých událostí.

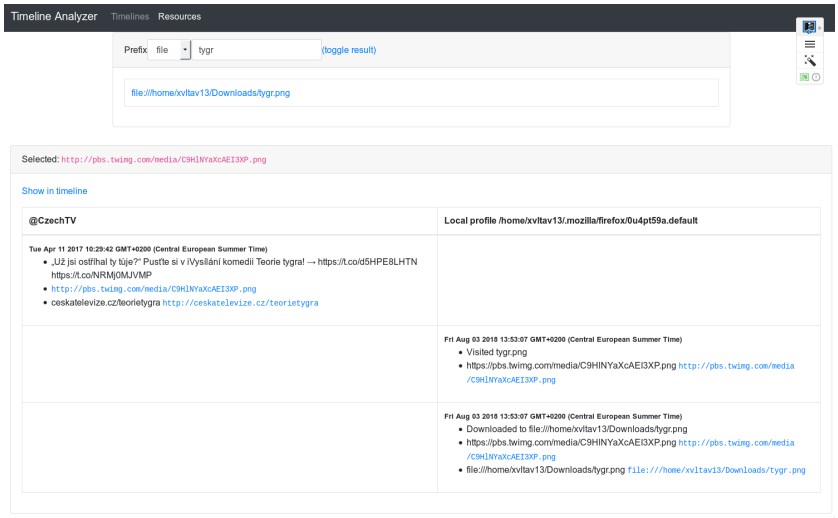


Obrázek 8. Procházení časové osy pomocí webového rozhraní nástroje TimelineAnalyzer

V části *Resources* je možno vyhledávat jednotlivé dokumenty (např. fotografie), které mají vztah k některým událostem na časové ose. Je možno pracovat jak s lokálními soubory, tak i s URL vzdáleně uložených souborů. Je tedy možno např. ověřit, zda lokální soubor pochází ze sociální sítě (tzn. zda byl stažen lokálním uživatelem, kdy a z jakého profilu na sociální síti) a případně lze dále dohledat, na kterém nebo kterých ze sledovaných profilů byl tento soubor zveřejněn a kdy se tak stalo.

Celý systém je implementován jako klientská aplikace na platformě Angular a TypeScript a je součástí aktuální verze balíku TimelineAnalyzer[11].

[11] https://github.com/nesfit/timeline-analyzer/tree/master/timeline-client-angular



Obrázek 9. Výpis událostí souvisejících s daným souborem

7 Závěr

Rekonstrukce časové osy umožňuje modelování a uložení a další analýzu informace o časových událostech souvisejících s používáním jak lokálních počítačů, tak i profilů na sociálních sítích. Současné přístupy k tomuto problému se však věnují prakticky výhradně zpracování dat z lokálních počítačů a to centralizovaným způsobem. Vzhledem k rozsahu dat, který je nutno zpracovat, je pak tato analýza obvykle velmi časově náročná.

Jako součást výzkumu v rámci projektu Tarzan jsme proto navrhli rozšíření přístupů k rekonstrukci časové osy o analýzu událostí na sociálních sítích. Popsaný přístup využívá existující ontologické datové modely, rozšiřuje je však o další prvky související se sociálními sítěmi a navrhuje využití jazyka SPARQL pro další analytické kroky.

Jako další krok jsme navrhli architekturu distribuovaného řešení, které umožňuje využít velkého počtu výpočetních uzlů jak pro paralelní získávání dat ze sociálních sítí, tak i pro následnou analýzu. Řešení je postaveno na platformách Apache Hadoop a Spark a na distribuovaném úložišti RDF dat Halyard. Implementovaný prototyp celého řešení je k dispozici jako software s otevřeným kódem.

Reference

1. HDFS Architecture Guide. Apache Hadoop, 2018, [Online; navštíveno 26.11.2018].

URL http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html

2. Anadiotis, G.: Graph databases and RDF: It's a family affair. ZDNet, Květen 2017, [Online; navštíveno 26.11.2018].
URL https://www.zdnet.com/article/graph-databases-and-rdf-its-a-family-affair/
3. Bekker, A.: Cassandra vs. HBase: twins or just strangers with similar looks? ScienceSoft, Červen 2018, [Online; navštíveno 26.11.2018].
URL https://www.scnsoft.com/blog/cassandra-vs-hbase
4. Burget, R.: Sociální sítě: Sběr a analýza dat v souvislosti s bezpečnostními incidenty. Technická Zpráva FIT-TR-2017-11, FIT VUT v Brně, 2017.
5. Carroll, J.; Herman, I.; Patel-Schneider, P. F.: OWL 2 Web Ontology Language RDF-Based Semantics (Second Edition). W3C recommendation, W3C, Prosinec 2012, https://www.w3.org/TR/2012/REC-owl2-rdf-based-semantics-20121211/.
6. Casey, E.; Back, G.; Barnum, S.: Leveraging CybOX^{TM} to standardize representation and exchange of digital forensic information. *Digital Investigation*, ročník 12, 2015: s. S102 – S110, ISSN 1742-2876, doi:https://doi.org/10.1016/j.diin.2015.01.014, dFRWS 2015 Europe.
URL http://www.sciencedirect.com/science/article/pii/S1742287615000158
7. Chabot, Y.; Bertaux, A.; Nicolle, C.; aj.: A complete formalized knowledge representation model for advanced digital forensics timeline analysis. *Digital Investigation*, ročník 11, 2014: s. S95 – S105, ISSN 1742-2876, doi:https://doi.org/10.1016/j.diin.2014.05.009, fourteenth Annual DFRWS Conference.
URL http://www.sciencedirect.com/science/article/pii/S1742287614000528
8. Chabot, Y.; Bertaux, A.; Nicolle, C.; aj.: An ontology-based approach for the reconstruction and analysis of digital incidents timelines. *Digital Investigation*, ročník 15, 2015: s. 83 – 100, ISSN 1742-2876, doi:https://doi.org/10.1016/j.diin.2015.07.005, special Issue: Big Data and Intelligent Data Analysis.
URL http://www.sciencedirect.com/science/article/pii/S1742287615000869
9. Chao, J.: Graph Databases for Beginners: Native vs. Non-Native Graph Technology. Červenec 2016, [Online; navštíveno 22.11.2018].
URL https://neo4j.com/blog/native-vs-non-native-graph-technology/
10. Guðjónsson, K.: Mastering the Super Timeline With log2timeline. SANS, Srpen 2010, [Online; navštíveno 23.11.2018].
URL https://www.sans.org/reading-room/whitepapers/logging/paper/33438
11. Hargreaves, C.; Patterson, J.: An automated timeline reconstruction approach for digital forensic investigations. *Digital Investigation*, ročník 9, 2012: s. S69 – S79, ISSN 1742-2876, doi:https://doi.org/10.1016/j.diin.2012.05.006, the Proceedings of the Twelfth Annual DFRWS Conference.
URL http://www.sciencedirect.com/science/article/pii/S174228761200031X

12. Howden, C.; Liu, L.; Ding, Z.; aj.: Moments in Time: A Forensic View of Twitter. In *2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing*, Aug 2013, s. 899–908.
13. Huber, M.; Schmiedecker, M.; Leithner, M.; aj.: Social Snapshots: Digital Forensics for Online Social Networks. In *Annual Computer Security Applications Conference (ACSAC)*, 12 2011.
14. Karau, H.; Konwinski, A.; Wendell, P.; aj.: *Learning Spark: Lightning-Fast Big Data Analytics*. O'Reilly Media, Inc., první vydání, 2015, ISBN 1449358624, 9781449358624.
15. Lanthaler, M.; Wood, D.; Cyganiak, R.: RDF 1.1 Concepts and Abstract Syntax. W3C recommendation, W3C, Únor 2014, http://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/.
16. Mulazzani, M.; Huber, M.; Weippl, E. R.: Social Network Forensics: Tapping the Data Pool of Social Networks. In *Eighth Annual IFIP WG 11.9 Internaional Conference on Digital Forensics*, 2011.
17. Okolica, J. S.: *Temporal Event Abstraction and Reconstruction*. Dizertační práce, Air Force Institute of Technology Wright-Patterson AFB United States, Prosinec 2017.
18. Olsson, J.; Boldt, M.: Computer forensic timeline visualization tool. *Digital Investigation*, ročník 6, 2009: s. S78 – S87, ISSN 1742-2876, doi:https://doi.org/10.1016/j.diin.2009.06.008, the Proceedings of the Ninth Annual DFRWS Conference.
URL http://www.sciencedirect.com/science/article/pii/S1742287609000425
19. Polakis, I.; Ilia, P.; Tzermias, Z.; aj.: Social Forensics: Searching for Needles in Digital Haystacks. In *4th International Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS)*, Nov 2015.
20. Prud'hommeaux, E.; Seaborne, A.: SPARQL Query Language for RDF. W3C recommendation, W3C, Leden 2008, http://www.w3.org/TR/2008/REC-rdf-sparql-query-20080115/.
21. Ryšavý, O.; Rychlý, M.: Platforma pro zpracování dat síťové forenzní analýzy: Návrh a implementace prototypu. Technická Zpráva FIT-TR-2017-07, FIT VUT v Brně, 2017.
22. Schatz, B. L.; Mohay, G. M.; Clark, A.: Rich Event Representation for Computer Forensics. In *Asia Pacific Industrial Engineering and Management Systems APIEMS 2004*, 2004.
23. Schelkoph, D.; Peterson, G.; Okolica, J.: Digital Forensics Event Graph Reconstruction. In *10th EAI International Conference on Digital Forensics and Cyber Crime*, Sep 2018.
24. Sotona, A.: Inside Halyard: 1. Triples, Keys, Columns and Values – everything upside down. Květen 2017, [Online; navštíveno 16.11.2018].
URL https://www.linkedin.com/pulse/inside-halyard-1-triples-keys-columns-values-upside-adam-sotona/
25. Sotona, A.: Inside Halyard: 3. Sipping a river through drinking straws. Květen 2017, [Online; navštíveno 16.11.2018].

URL https://www.linkedin.com/pulse/inside-halyard-3-sipping-river-through-drinking-straws-adam-sotona/
26. Sotona, A.: Inside Halyard: 4. Bulk operations – shifting a mountain. Květen 2017, [Online; navštíveno 16.11.2018].
URL https://www.linkedin.com/pulse/inside-halyard-4-bulk-operations-shifting-mountain-adam-sotona/
27. Svátek, V.; Vacura, M.: Ontologické inženýrství. In *DATAKON 2007*, Brno: Masarykova univerzita, 2007, s. 60–91.
28. Turnbull, B.; Randhawa, S.: Automated event and social network extraction from digital evidence sources with ontological mapping. *Digital Investigation*, ročník 13, 2015: s. 94 – 106, ISSN 1742-2876, doi:https://doi.org/10.1016/j.diin.2015.04.004.
URL http://www.sciencedirect.com/science/article/pii/S1742287615000444
29. Tutko, J.: *Distribuované zpracování rozsáhlých dat na platformě Java*. Diplomová práce, Vysoké učení technické v Brně, Fakulta informačních technologií, 2018.
URL http://www.fit.vutbr.cz/study/DP/DP.php?id=20647
30. White, T.: *Hadoop: The Definitive Guide*. O'Reilly Media, Inc., první vydání, 2009, ISBN 0596521979, 9780596521974.