Platforma pro zpracování dat síťové forenzní analýzy

Návrh a implementace prototypu

Technická zpráva FIT VUT v Brně

Ondřej Ryšavý a Marek Rychlý

Technická zpráva č. FIT-TR-2017-07
Fakulta informačních technologií, Vysoké učení technické v Brně

Last modified: 4. ledna 2018

Platforma pro zpracování dat síťové forenzní analýzy

Návrh a implementace prototypu

Ondřej Ryšavý a Marek Rychlý

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

Abstrakt Tato zpráva se věnuje návrhu a implementaci prototypu platformy pro zpracování dat síťové forenzní analýzy. Uvažovaná platforma využívá dostupné softwarové technologie pro distribuované zpracování vstupních dat s cílem poskytnou škálovatelnost. Ve zprávě je uveden rozbor problému, přehled požadavků na platformu a její návrh. Dále je implementovaný prototyp vyhodnocen jak po funkční stránce, tak s ohledem na dosažitelnou výkonnost zpracování.

1 Úvod

Bohatost zdrojů dat a exponenciální růst jejich objemu představuje nové výzvy pro mnoho tradičních oblastí ICT. Digitální forenzní analýza bezpečnostních incidentů není výjimkou. Bezpečnostní analytici a vyšetřovatelé čelí problému nedostatečné podpory nástrojů pro zpracování velkého objemu dat. Kořeny tohoto problému spočívají ve skutečnosti, že tento výrazný růst objemu dat znemožňuje použití stávajících metod zpracování. Tradiční postup digitální forenzní analýzy se sestává z dobře definovaných kroků identifikace dat, získávání a extrakce dat, uchování informací a jejich analýzy a sestavení reportu. Tento pracovní postup byl navržen a upřesněn v devadesátých létech, kdy formát dat a dostupné výpočetní technologie a software byly poměrně jednotné. Navíc množství dat, které bylo třeba zpracovat, bylo relativně malé. Ve většině případů bylo možné provést všechny výše uvedené kroky pomocí nástrojů instalovaných na jediné forenzní pracovní stanici.

Kvůli rychlému pokroku v oblasti software a hardware již toto není pravda. Nejen rostoucí velikost dat způsobená poklesem nákladů na ukládání dat a širokopásmového připojení představuje výzvu pro digitální forenzní analýzu. Často je obtížná již fáze získávání dat s pomocí dostupných nástrojů. Například vytvoření obrazu pro dnešní běžné terabajtové disky trvá několik hodin.

Vyšetřovatelé také musí čelit problému rozmanitosti dostupných výpočetních zařízení. Smartphony, tablety a další inteligentní zařízení masivně pronikají na trh. Cloudové služby jsou další technologií, která mění požadavky kladené na digitální forenzní metody. To vše znamená, že klasický přístup založený na použití jediné forenzní pracovní stanice nemůže splnit současné požadavky. V mnoha

případech je množství dat, které mají být zpracovány, přesahuje několik terabajtů. Navíc, některé formy počítačové kriminality zahrnují kombinaci několika sofistikovaných technik a pro jejich vyšetřování je nutné zpracovat a korelovat informace z několika různých a potencionálně velkých datových zdrojů. K řešení tohoto problému je zkoumáno použití technik z oblasti Big Data [6].

V této zprávě je uvažována integrovaná platforma pro analýzu digitálních dat z bezpečnostních incidentů (platforma TARZAN) k řešení výše uvedených problémů. Platforma umožňuje shromažďovat, ukládat a zpracovávat digitální forenzní data za pomocí Big Data technologií a provádět různé analýzy zaměřené na bezpečnost, které se pohybují od odhalení bezpečnostních incidentů IT k inferenčním analýzám dat ze sociálních sítí nebo krypto-měnových transakcí.

Několik přístupů bylo navrženo v oblasti digitální forenzní analýzy. Protože dostupné technologie nejsou vždy vhodné k podpoře dlouhodobých, rozsáhlých analytických úloh [4], začaly se objevovat návrhy na použití Big Data přístupů k digitální forenzní analýze [8], [6]. Níže je uveden přehled prací, které se zabývají problematikou zpracování velkých dat specificky pro oblasti digitální forenzní analýzy.

V [13] byl navržen proces redukce dat založený na selektivní vizualizaci informací s cílem urychlit náročný proces zjišťování důkazů. Tento přístup je založen na vytvoření rychlého náhledu na data, který by měl umožnit lepší orientaci v informacích napříč různými zdroji dat.

Další bavržený přístup cílí na problematiku sběru dat a jejich korelaci (FCCE, [14]). Navržené prostředí uvažuje fáze shromažďování, ukládání a získávání velkých dat pomocí implementace extraktorů z různých datových zdrojů, metod agregace dat, vytvoření škálovatelného úložiště a poskytnutí programátorského API pro přístup k uloženým informacím. Tento přístup byl demonstrován na příkladu identifikace DNS flux, což je technika, kterou používají sítě Botnet pro komunikace s řídícími servery.

V této zprávě uvažujeme jako případovou studii problematiku zpracování zachycených síťových dat. Zatímco hardwarová technologie umožňují zachytit komunikaci na současně používaných rychlostech (1-10Gb/s), existující nástroje pro analýzu zachycené komunikace mají problém zpracovat data již od velikosti jednotek gigabajtů. Nástroje pro všeobecné použití zahrnují síťové analyzátory (Wireshark, TCP výpis), systémy IDS (Snort, Bro), specializované nástroje pro identifikaci zařízení a služeb (Nmap, p0f) a nástroje k identifikaci a analýze bezpečnostních hrozeb.

V následující části je uveden stručný přehled tří volně dostupných nástrojů, které nabízí typické funkce síťové forenzní analýzy. Komplexní přehled síťových forenzních nástrojů lze nalézt v [11].

PyFlag je balík nástrojů pro digitální forenzní analýzu, který může být použitý jak pro klasickou digitální forenzní analýzu úložišt, pro forenzní analýzu paměti a také jako nástroj pro síťovou forenzní analýzu. Tento nástroj vyvinul M. Cohen z Australské federální policie v roce 2005 [5]. PyFlag je navržen na základě koncepce Virtual File System (VFS). Pro každý podporovaný zdroj dat je implementován interpreter, který mapuje datový zdroj do VFS. Při ana-

lýze PCAP souborů je obsah vstupního souboru namapován do VFS. Nástroj analyzuje a rozdělí jednotlivé pakety až na nižší vrstvy protokolů, shromažďuje související TCP pakety do streamů a nakonec spustí analýzu na úrovni aplikačních protokolů. PyFlag tak umožňuje znovu sestavit obsah komunikace, např. webové stránky, e-mailová konverzace atd.

Dalším populárním nástrojem je Network Miner. Tento open source nástroj, který integruje paket sniffer a analyzátory protokolů vyšší vrstvy, umožňuje pasivní sledování a analýzu síťového provozu. Network Miner nabízí několik užitečných funkcí, jako je možnost identifikace operačního systému, klasifikace provozu a opětovné sestavení přenesených souborů pro HTTP, FTP, TFTP a SMB protokoly.

Xplico je modulární nástroj zaměřený na rekonstrukci datového obsahu přenášeného v síti. Software se sestává ze vstupního modulu, který zpracovává zdrojová data, dekódovací modul vybavený protokolovými parsery pro dekódování provozu a export obsahu a výstupního modulu, který organizuje dekódovaná data a zobrazuje je uživatel. Na rozdíl od PyFlag a Network Mineru, Xplico není typická aplikace pro stolní počítače, ale je nasazena jako server s webovým rozhraním. Autoři tvrdí, že nástroj umí analyzovat velké soubory PCAP o mnoha gigabajtech. Vzhledem k tomu, že Xplico je klasická architektura klient-server, výkonnost nástroje je omezena možnostmi serveru.

Novým přístupem je použití technologií pro distribuované zpracování dat, poprvé uvedený v [9]. Zde popisovaný systém, používá knihovnu libpcap pro zpracování dat o velikosti jednotek terabajtů za pomoci Apache Spark. Omezení systému spočívá v nemožnosti jeho rozšíření o jiné datové zdroje a analýzu dat ze sítě v širších souvislostech.

Další přístupy zabývající se zpracování velkých dat pro účely digitální forenzní analýzy či bezpečnosti sítě byly prezentována v [17], [3], [12] a [7]. Projekty Apache Metron [1] a Apache Spot [2] jsou z těchto snah v nejpokročilejším stavu. Oba projekty se snaží vytvořit obecný rámec pro bezpečnostní analýzu, identifikaci IT hrozeb, sekundární zpracování dat, extrakci dat z různých protokolů, e-mailů, detekci narušení, atd. Oba tyto projekty jsou primárně a úzce zaměřeny na síťová data a neumožňují najít souvztažnosti s jinými forenzními daty a poskytnout tak komplexní platformu pro digitální forenzní analýzu velkých dat.

2 Požadavky na platformu

Integrovaná platforma pro zpracování dat digitální forenzní analýzy musí splňovat požadavky, které odpovídají předpokládáným scénářům použití. Požadavky jsou rozděleny do několika kategorií podle jejich významu. Priority požadavků odpovídají plánovanému výstupu, kterým je prototyp nástroje nikoliv komerční produkt. V textu je platforma pro zpracování digitálních dat označena také jako systém.

2.1 Předpokládané použití systému

Systém bude použit pro offline analýzu digitálních dat. Systém bude zpracovávat a analyzovat různé typy digitálních dat z různých datových zdrojů. Nicméně jako primární zdroje dat budou sloužit data představující zachycenou síťovou komunikaci a další informace, které lze získat ze síťových prvků či serverů.

- Systém bude zajišťovat načtení a zpracování dat z různých zdrojů. V některých případech bude nezbytné data předzpracovat před jejich vložením do systému.
- Data zpracována systémem budou v systému uložena, což umožní jajich další analýzu. Způsob uložení dat bude záviset na povaze dat. V systému bude možné kombinovat různé způsoby uložení, například SQL, NoSQL databáze či distribuovaný souborový systém.
- Pro data uložená v systému bude možné spouštět analytické úlohy. Analytické úlohy jsou specifické pro aplikační doménu a budou používat různé vypočetní postupy - dotazování nad databází, či strojové učení a data mining.
- Systém bude používán administrátorem respektive bezpečnostním analytikem pro získání informací o incidentech zachycených v uložených datech nebo pro identifikaci nových hrozeb pomocí nalezení vzorů či specifických informací. Pro základní a typické úlohy bude systém poskytovat grafické uživatelské rozhraní přístupné pomocí webového prohlížeče
- Systém bude exportovat vybraná data pro jejich další zpracování jinými nástroji. Například bude možné exportovat vybrané síťové toky ve formě pcap souborů.

2.2 Scénáře použití - Síťová forenzní analýza

Pro první fázi řešení byl identifikován scénář, který měl za cíl pomoci vytvořit návrh platformy. V tomto scénaři se jedná o vytvoření systému pro forenzní síťovou analýzu. Tento systém bude umožňovat zpracování síťové komunikace podobně jako nástroje typu Network Miner, Xplico či Netfox Detective. Rozdíl oproti těmto nástrojům je v tom, že systém bude využívat distribuované zpracování, což by mělo usnadnit jeho nasazení na vetší množství dat a zároveň získat potřebné výsledky v kratším čase.

2.3 Požadavky

Z uvedeného plánované použití systému a na základě popsaného scénáře byly navrženy požadavky na systém. Požadavky jsou uvedeny níže v jednotlivých kategorií. Uživatelské a funkční požadavky vychází z předpokládaného využití systému a výše uvedených scénářů. Dále jsou doplněny dalšími požadavky na výkonnost, bezpečnost, spolehlivost, spravovatelnost a rozšiřitelnost.

Uživatelské

REQ-USR1: Systém bude umět zpracovávat zachycený provozu ve formátu pcap kompatibilním s nástrojem tcpdump.

REQ-USR2: Systém bude prezentovat výsledky zpracování pomocí tabulek, grafů a jiných vizualizačních prostředů ve webovém rozhraní

REQ-USR3: Systém bude umět zpracovávat vstupní data z různých zdrojů, minimálně z HDFS, lokálního souborového systému, či ze specifikovaného URL pomocí protokolu HTTP.

REQ-USR4: Systém bude dostupný pomocí webového rozhraní a příkazového řádku.

Funkční požadavky

REQ-FUN1: System bude umět počítat Netflow informace pro vstupní PCAPy a interně je reprezentovat vhodným způsobem pro další analýzu.

REQ-FUN2: Systém bude umět extrahovat informace z vybraných aplikačních protokolů a tyto ukládat pro další zpracování.

REQ-FUN3: Systém bude umět počítat základní statistiky pro Netflow a agregované Netflow záznamy: Timeline, Hosts wit top transfer/flows, LAN vs WAN traffic, Server response time, Top Internet destination, Structure of the traffic grouped by application protocols, Top used applications.

REQ-FUN4: Systém bude umět analyzovat informace z hlaviček protokolu HTTP: URL, Host, Content-Type, Cookies.

REQ-FUN5: Systém bude umět analyzovat informace z DNS protokolu, minmálně: DNS ID, Query records, Answer records, RR records.

REQ-FUN6: Systém bude umět hledat klíčová slova v jednotlivých paketech.

REQ-FUN7: Systém bude umět analyzovat informace v SSL/TLS komunikaci: Certificate, TLS cipher suites, TLS handshake type, TLS extensions, TLS version, TLS key length, TLS session ID, TLS random,

Výkonnostní požadavky

REQ-PRF1: Systém bude umět zpracovat více dat než běžně dostupné nástoje. Doba zpracování bude menší než při zpracování dostupnými nástroji. Doba závisí na dostupných HW prostředcích: Zpracování vstupních PCAPu o velikosti 10 GB dat 10 s na sestavě s 64 vCPU a 64 GB RAM.

REQ-PRF2: Systém bude škálovatelný přidáním dalších výpočetních uzlů.

Spolehlivost

REQ-REL1: Systém bude umožňovat obnovení činnosti v případě nedostupnosti bez nutnosti pokročilých zásahů administrátora. Ideálně by mělo stačit pouze restartovat služby systému.

REQ-REL2: Data v systému budou uložena s takovou mírou redundance, aby nedošlo k jejich ztrátě při selhání jednoho uzlů.

REQ-REL3: Nepředpokládá se, že systém bude obsahovat originální data a není tedy nutné provádět zálohy.

REQ-REL4: V případě selhání SW/HW v průběhu výpočtu bude možné výpočet po obnovení systému restartovat.

Bezpečnostní požadavky

REQ-SEC1: Uživatelský přístup k systému bude zabezpečeným webovým rozhraním: (i) Webové rozhraní bude vyžadovat autentizaci, (ii) Komunikace prohlížeč-server bude šifrovaná.

REQ-SEC2: Administrátorské konzole jednotlivých služeb systému budou dostupné pouze lokálně.

REQ-SEC3: Administrátorský přístup pomocí SSH bude dostupný z veřejné sítě. Administrátor musí používat přihlašování pomocí klíče, nikoliv hesla.

REQ-SEC4: Data v datovém úložišti budou volitelně šifrována.

REQ-SEC5: Komunikace mezi jednotlivými službami v systému nebudou dostupné vně systému a budou chráněny firewallem.

Systémové požadavky

REQ-SYS1: Systém bude tvořen výpočetním klusterem. Bude možné přidávat další uzly.

REQ-SYS2: Komunikace uvnitř klusteru by měla být oddělena vhodným způsobem od ostatní komunikace.

REQ-SYS3: Systém bude obsahovat minimálně jeden hlavní uzel, který bude sloužit pro správu systému.

REQ-SYS4: Výpočetní kluster může být nasazen jak privátně tak v CLOUDu.

Hardwarové požadavky

REQ-HWR1: Systém bude používat běžně dostupný hardware. Může jím být jak serverové tak běžné PC.

REQ-HWR2: Jednotlivé uzly systému by měly splňovat minimální požadavky: Vícejádrové CPU (minimálně 4), minimálně 2GB $\times$ počet jader CPU, Alespoň 1 Gb etherent rozhraní, Minimálně 2 $\times$ 1TB disk.

Softwarové požadavky

REQ-SWR1: Systém bude používat vhodnou distribuci OS (CentOS, RedHat, Ubuntu).

REQ-SWR2: Systém bude relizován pomocí technologií Apache Hadoop: HDFS, Spark, Kafka, Hive.

REQ-SWR3: Aplikace pro systém budou implementovány v jazycích: Java/Scala, Python, C♯.

3 Návrh

Systém je navržen jako distribuované prostředí, které využívá především technologie z rodiny Apache Hadoop. Systém je navržen pro provoz v prostředí privátního datacentra od několika uzlů do několika desítek až stovek uzlů.

3.1 Architektura

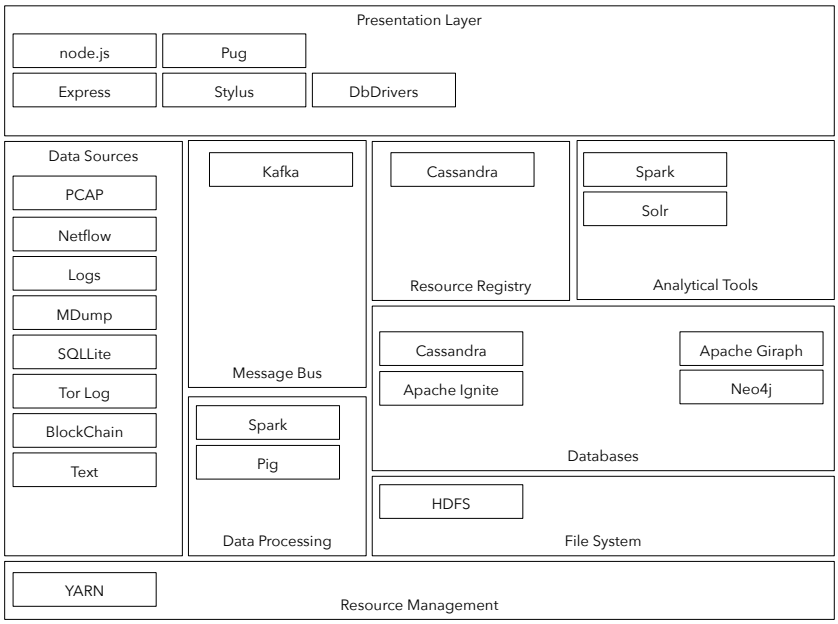
Architektura systému (viz obrázek 1) zahrnuje:

- Message Bus – zajišťuje komunikaci uvnitř systému mezi jednotlivými částmi. Realizováno pomocí Apache Kafka[1].
- Databases – datové úložiště bude realizováno formou distribuované NoSQL databáze pro uložení primární dat různého formátu. Relizováno pomocí Cassandra[2], Apache Ignite [3].
- File Storage - pro uložení nových nezpracovaných dat nebo objektů typu soubor bude použit distribuovaný souborový systém. Relizováno pomocí Apache HDFS [4].
- Object Registry - register zdrojů slouží pro uložení informace o všech objektech v systému. Každý datový objekt, který má být systémem zpřístupněn musí mít identifikaci. Tato identifikace společně s metainformacemi je uložena v Resource Registry. Data budou uložena v databázi a poskytována ve formátu odvozeném z AFF4 [5]
- Resource Management - modul, který sparvuje výpočetní zdroje a prostředky. Poskytne Apache YARN [6].
- Data Processing - komponenty pro distribuované zpracování dat ze vstupních zdrojů. Relizováno v prostředí Apache Spark [7].
- UI Server – je aplikační server, který bude poskytovat uživatelské rozhraní pro koncové uživatele. Řešení je podstaveno na light-weight web serveru implementovaného s pomocí technologie Node.js a implementačním jazyce JavaScript/TypeScript.
- Data Source Adapter – slouží pro vkládání dat z různých datových zdrojů.

3.2 Zpracování dat (ETL)

Úvodní operací pro nová data je jejich zpracování v modulu Data Processing. Toto zpracování zahrnuje operace známé jako Extract, Transform and Load (ETL), které pro vstupní syrová data generují zpracovaná data jenž mohou být uložena a dále zpracována v systému. Tyto operace zahrnují:

[1] https://kafka.apache.org/
[2] https://cassandra.apache.org/
[3] https://ignite.apache.org/
[4] https://wiki.apache.org/hadoop/HDFS/
[5] http://forensicswiki.org/wiki/AFF4
[6] https://hortonworks.com/apache/yarn/
[7] https://spark.apache.org/



Obrázek 1: Architektura platformy

- Filtrování, transformace a čištění dat je důležité pro odstranění chybných redundatních záznamů, sjednocení informací z různých zdrojů a úpravu strukturu dat.
- Konverze dat do efektivního formátu se provádí pro optimalizaci dalších operací s daty. Některé formáty jsou vhodné pro přenos mezi systémy nebo jsou snadno čitelné (JSON, CSV, plain text), ale nejsou vhodné pro efektivní zpracování tak jako například binární formáty (Parquet, Avro, ProtocolBuffers).
- Rozdělení dat a optimalizace datových struktur pro uložení v databázi se provádí zejména z důvodu zrychlení často používaných operací.

Zpracování dat začíná jejich uložením do souborového systému platformy či zasláním na Message Bus. Poté je spuštěna aplikace pro Data Processing. Jako primární nástroj pro Data Processing je uvažován Apache Spark. Výsledek zpracování je poté uložen v databázi či poslán k dalšímu zpracování.

Uložení do HDFS Data uložená do HDFS jsou zpracována pomocí programů v Data Processing. Jednou z možností je použití Apache Spark pro načtení dat z HDFS, jejich zpracování pomocí implementované Spark aplikace a uložení výsledku do databáze. Jinou možností je výsledek poslat prostřednictvím Message Bus pro další zpracování.

Zaslání do Message Bus V případě menšího množství dat, či zpracování online datového streamu je možné použít prostředků Message Bus jako vstupního bodu platformy pro zpracování dat. Zde je nutné, aby byl v systému aktivní odběratel pro uvedený typ datového zdroje, který data dále zpracuje či uloží do databáze.

3.3 Vstup / Výstup

Platforma TARZAN je určena pro distribuované zpracování vstupních dat a jejich následnou analýzu. Vstupní data mohou poskytovat různé zdroje. Platforma poskytuje možnosti pro uložení vstupních dat v jejich původní podobě a následnou extrakci, transformaci a uložení v podobě, která je vhodná pro další analýzu. Výsledný prototyp platformy bude podporovat datové zdroje uvedené v následující podkapitole. Nicméně platforma je navržena tak, aby bylo možné nové datové zdroje přidávat. K tomu je nutné definovat vstupní adaptér, což je program, který umí načíst a dekódovat formát dat a program pro Data Processing, jenž provedene nezbytné transformace nutné pro uložení zpracovaných dat v datovém úložišti platformy.

Vstupními daty může být libovolný datový zdroj, pro který je vytvořen adaptér, který umožnuje data číst z HDFS úložiště či prostřednictvím Message Bus. V prostředí systému TARZAN se uvažuje minimálně o následujícím typu dat:

- Zachycená síťová komunikace ve formátech PCAP nebo PCAP NG
- Netflow záznamy ve formátu NFDUMP
- Logovací soubory ve formátu Syslog
- Windows Event Log
- Obrazy disků ve formátu RAW

Další datové zdroje a podporované formáty budou přidávány dle potřeb. Výstupní data jsou určena pro export dat ze systému a jejich zpracování dalšími specializovanými nástroji. Zejména se jedná o export vybrané komunikace ve formátu PCAP.

3.4 Analýza Dat

Zpracovaná data jsou uložena do databáze (Cassandra, Hive) a jsou k dispozici pro další analýzu. Cílem analýzy dat je zpracovat uložená data a představit je způsobem, který bude srozumitelný uživatelům. Analýza dat může mít jeden či více kroků:

- dotazování nad databází a zobrazení výsledků prostřednictvím UI uživatelovi, nebo
- spuštění analytického algoritmu (SparkSQL, SparkMLlib) a uložení výsledků opět do databáze, ze které pak mohou být dále analyzovány či vizualizovány.

Analytická fáze bude využívat různých typů metod:

– Data Processing - zpracování dat pomocí operací, ať již formou SQL popřípadě NoSQL dotazování či vytvoření složitější výpočetní PIPELINE.
– Data Mining - je metoda pro identifikaci vzorů v datech, za použití různých technik strojového učení, statistiky a dotazování v databázích.
– Machine Learning - je kolekce algoritmů, sloužicí především pro problémy klasifikace a predikce.

Vzhledem k tomu, že uvažovaná platforma integruje různé typy dat, je možné provádět různé typy analýzy. Složitější analýza je reprezentována jako DATAFLOW, který se skládá z těchto operací:

– Filtrace - data jsou filtrována na základě hodnot z vybraných atributů.
– Transformace - vstupní data jsou modifikována na výstupní pomocí definované operace. Takto mohou vznikat nové hodnoty, které jsou vypočteny ze vstupních dat.
– Redukce - vstupní data jsou redukována. Oproti filtraci toto znamená například odstranění některých nepotřebných atributů či požití komprese k odstranění redundance v datech.
– Paralelizace - vstupní datová kolekce je rozdělena do více bloků pro podporu paralelního zpracování.
– Korelace - operace, která identifikuje vzájemný vztah v datových sadách.

DATAFLOW tvoří výpočetní PIPELINE, kterou je možné realizovat v uvažovaném distribuovaném prostředí. V projektu se uvažuje o použití platformy Apache Spark, která poskytuje množství operací pro tvorbu DATAFLOW.

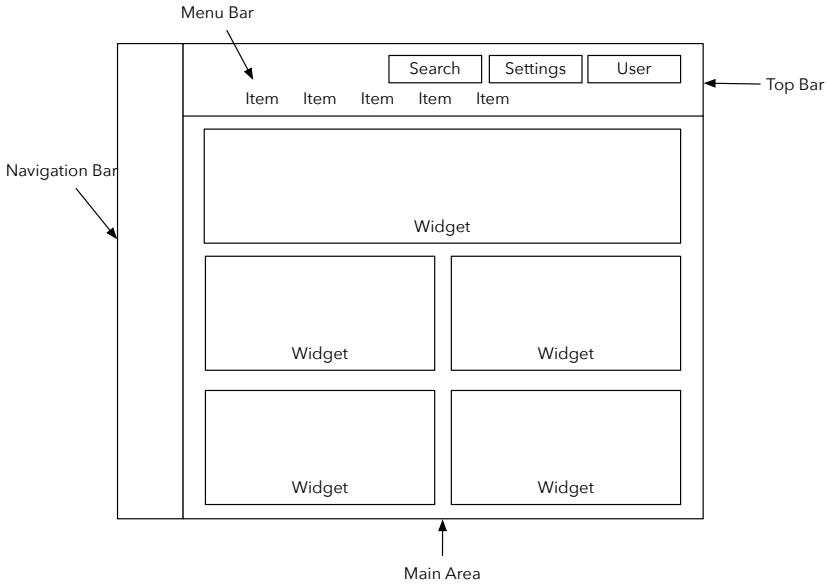
3.5 Uživatelské rozhraní

Systém bude poskytovat jednotné webové uživatelské rozhraní. Obrazovka rozhraní (viz obrázek 2) je rozdělena na horní řádek, ve kterém je umístěno menu pro přístup k dalším stránkám a tlačítka pro přístup k nastavení, vyhledávání v systému, a informacemi o aktuálně přihlášeném uživateli. V hlavní oblasti jsou umístěny bloky, které zobrazují jednotlivé informace. Těmito bloky mohou být tabulky, grafy či jiné vizualizační prvky. Navigation Bar slouží pro přepínání panelů.

Úvodní stránkou je tkz. DASHBOARD, který poskytuje základní přehled o analyzovaných datech a dalších možnostech. Pomocí prvku Navigation Bar je možné se přepínat na další stránky, které poskytují jiné pohledy na data. Každá stránka je zaměřena na analýzu specifických vlastností. DASHBOARD se bude skládat z WIDGETů, které představují souhrnné informace o datech a budou také poskytovat odkazy na detailní pohledy.

4 Implementovaný prototyp

V první fázi byl implementován prototyp pro demonstraci zpracování dat vybraných scénářů a jejich prezentaci. Tento prototyp uvažuje, že vstupními daty



Obrázek 2: Uživatelské rozhraní platformy

jsou PCAP soubory. Pro tento typ vstupních dat byl implementován prototyp jako kolekce knihoven v jazyce Java, který umožňuje data zpracovat v prostředí Apache Spark a dále analyzovat a výsledky vizualizovat v prostředí Apache Zeppelin. Tento prototyp byl otestován v sestaveném výpočetním uzlu na různých datasetech. Datasety obsahovaly vždy zdrojové PCAP soubory a v někerých případech také anotaci dat. Datasety byly upraveny (rozděleny na menší PCAPy) a nahrány do HDFS systému.

V první fázi jsou uložená data zpracována pomocí Spark programu a implementované knihovny pro manipulaci s PCAP soubory. Následně jsou pakety uspořádány do toků. V třeím kroce se pomocí Spark SQL knihovny provádí analýza dat a výsledky jsou zobrazeny pomocí Apache Zeppelin.

4.1 Výpočetní prostředí

Výpočetním prostředím je Apache Spark. Pro implementaci prototypu byl použit výpočetní uzel Supermicro SuperTwin2 6026TT-TF, který obsahuje 8 procesorů Intel (R) Xeon E5520 @ 2.26 GHz ve čtyřech výpočetních uzlech propojených 1Gbit Ethernet sítí. Spark cluster je konfigurován jako jeden MASTER uzel a 4 WORKER uzly. Každý uzel má k dispozici 16 vCPU a 48GB RAM. Prostředí používá HDFS distribuovaný systém o celkové kapacitě 7,47 TB.

4.2 UI

Jako uživatelské rozhraní bylo použito prostředí Apache Zeppelin. Toto prostředí poskytuje nástroj pro interaktivní analýzu dat pomocí různých systémů, včetně Apache Spark. Analýza představuje tkz. NOTEBOOK, který zahrnuje program pro Apache Spark (v jazyce SCALA), dotazy Spark SQL a dokumentace či komentáře k analýze. Výsledek analýzy je možné zobrazit ve formě reportu.

4.3 Netflow Analýza

Analýza netflow záznamů se používá pro monitorování sítě a také pro detekci anomálií. Implementovaný prototyp demonstruje možnosti platformy pro tento typ analýzy. Pro monitorování a získání přehledu o komunikaci ve zdrojových datech jsou vypočteny a zobrazeny informace:

- Timeline - ukazuje množství vzniklých flow, přenesených dat/paketů v čase
- Host with Top Data Transfer - zobrazuje zařízení, které vytvořily nejvíce flow, respektive přenesly nejvíce dat
- HTTP vs HTTPs - zobrazuje množství HTTP a HTTPs komunikace v čase
- LAN vs WAN - ukazuje množství komunikace v rámci lokální sítě a komunikaci do Internetu
- Most Frequently Queried Domains - ukazuje která DNS jména byla nejvíce zastoupena v DNS dozazech.
- Server Response Time - pro každou klient server komunikaci je vypočtena čas potřebný pro první odpověď serveru. Tyto informace jsou pak agregovány do několika skupin a představují důležitou informaci pro monitorování služeb.
- Structure of Email Traffic - ukazuje členění provozu souvisejícího s elektronickou poštou podle použitých protokolů. Anomálie v této komunikaci může představovat například šíření SPAMu.
- Top Hostnames - ukazuje prvních N webových serverů, se kterými bylo komunikováno seřazeno podle množství komunikace.
- Top Internet Endpoints - představuje prvních N adres v internetu, se kterými bylo komunikováno podle množství komunikace.
- Top Mail Senders and SMTP Servers - prvních N počtovních klientů a serverů seřazených podle monžství generovaného provozu.

4.4 Analýza Paketů

Analýza paketů umožňuje získat detailní informace o provozu. V případě nešifrované komunikace je také možné extrahovat obsah zachycené komunikace. V implementovaném prototypu byly demonstrovány následující metody:

- Extrakce HTTP URL - pomocí HTTP parseru je zpřistupněna HTTP hlavička. Tyto informace je možné dále zpracovávat. Například je možné vyhledávat URL pomocí regulárního výrazu, nebo hledat specifické URL, jenž je používáno Malware.

– Extrakce a reassembling TCP flow - obsah TCP komunikace je složen do podoby datového toku a může být dále analyzován, nebo uložen do výstupního souboru.
– Extrakce souborů pro vybrané protkoly (HTTP, SMTP) - pomocí extrakce TCP flow je možné dále analyzovat obsah pro vybrané protokoly a z něho extrahovat přenášené soubory. Toto je demonstrováno na příkladu SMTP a HTTP protokolů.
– Keywords - v průbehů zpracování vstupních paketů je možné provádět vyhledávání předem definovaného seznamu kličových slov.
– DNS - extrakce informací z DNS provozu poskytuje možnosti další analýzy, například identifikace kompromitovaných DNS serverů a další bezpečnostní rizika spojená s DNS systémem.
– Extrakce informací z SSL/TLS - pskytnutí informace z TLS/SSL provozu (certifikáty, použité šiforvací algoritmy, metainformace o TLS komunikaci) může být užitečná například pro analýzu šiforvané komunikace. Jelikož množství škodlivého toku je dnes šifrované, je toto čast jediný způsob jak tuto závadnou komunikaci detekovat.

5 Vyhodnocení

Kromě otestování funkcionality prototypu bylo provedeno také výkonností vyhodnocení. Byly vybrány tři druhy výpočetních úloh různé náročnosti:

1. Capture info - toto úloha spočívá ve výpočtu parametrů zdrojových PCAP souborů podobně jako nástroj capinfo aplikace Wireshark. Výsledkem výpočtu je množství paketů, počet bajtů za sekundu, celková velikost přenesených dat, průměrná velikost paketu, atd. Výpočet probíhá v jednoduché pipeline, která přečte všechny pakety a aktualizuje průběžné hodnoty. Výsledek je poté agregací průběžných hodnot (obrázek 4a).
2. Flow aggregation - Tato úloha nejprve agreguje načtené pakety do toků a poté nad vytvořenými toky provádí další analýzu. Pro účely testování bylo počítáno vždy N toků s nejvíce přenesenými daty (obrázek 4b).
3. HTTP Request Analysis - V této úloze je použit jednoduchý HTTP request parser pro získání informací z HTTP hlaviček. Testovací úloha představovala výpočet množství požadavků seskupených podle jednotlivých HOST hodnot (obrázek 4c).

Při testování byl měřen čas potřebný k dokončení úlohy. Jako zdrojová data byl použit dataset o velikosti 10GB. Tento dataset byl rozdělen na PCAPy o velikost cca 100MB, které byly uloženy do HDFS systému.

6 Plán dalšího vývoje

V roce 2018 bude prototyp platformy dále rozvíjen několika směry. Budou specifikovány a posléze implementovany další scénáře použití a rozšířeny možnosti

```
val frames = sc.hadoopFile("hdfs://storage.local/cap/*.cap", ...)
val packets = frames.map(x => Packet.parsePacket(x._2.get().asInstanceOf[RawFrame]))
val capinfo = packets.map(Statistics.fromPacket).reduce(Statistics.merge)
```

(a) Capture Info

```
val frames = sc.hadoopFile("hdfs://storage.local/cap/*.cap", ...);
val packets = frames.map(x=> Packet.parsePacket(x._2.get().asInstanceOf[RawFrame]))
val flows = packets.map(x=>(x.getFlowString(),x))
val stats = flows.map(x=>(x._1,Statistics.fromPacket(x._2)))
                 .reduceByKey(Statistics.merge)
stats.takeOrdered(10)(Ordering[Int].reverse.on(x=> x._2.getPackets()))
                            .map(c=>Statistics.format(c._1, c._2)).foreach(println)
```

(b) Flow aggregation

```
def xf(x:PacketPayload) = x.getPacket().extendWith("http",
                                    HttpRequest.tryParseRequest(x.getPayload()))
val packets = frames.map(x=> Packet.parsePacket(x._2.get().asInstanceOf[RawFrame],
                                                toConsumer(xf)))
val http = packets.filter(x=> x.containsKey("http.request"))
val hosts = http.map(x => (x.get("http.host"),1)).reduceByKey(_ + _)
hosts.takeOrdered(20)(Ordering[Integer].reverse.on(x=> x._2)).foreach(println)
```

(c) HTTP Request Analysis

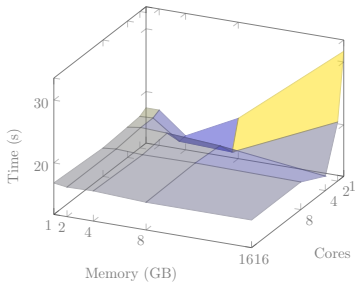
Obrázek 3: Scala kód pro různé operace

zpracovávat nové datové zdroje. Druhým směrem bude realizace analytických metod za použití knihoven Spark MLLib [16] a Spark GraphX [18]. První z nich představuje implementaci Machine Learning algoritmů. Druhá knihovna obsahuje efektivní immplementaci grafových algorithmů.

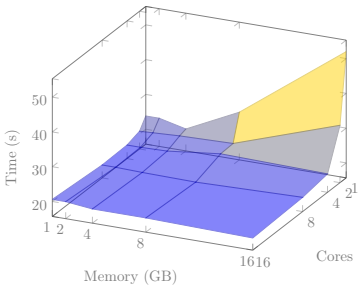
6.1 Scénáře použití

Bude rozšířena kolekce scénářů uvažovaných použití platformy na základě výstupů z ostatních skupin v roce 2017.

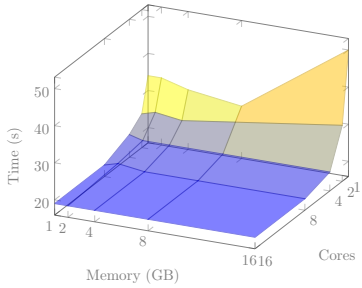
Zpracování TOR dat Tento scénář zahrnuje rozšíření systému o sběr a extrakci dat z TOR databáze uzlů. Tato databáze přístupná službou ExoneraTor uchovávaá historické údaje o uzlech sítě TOR. Zpracováním těchto dat a jejich následnou analýzou budou zpřístupněny tyto informace:



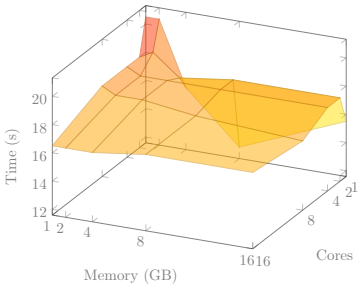
(a) Výpočetní čas pro operaci Capture Info



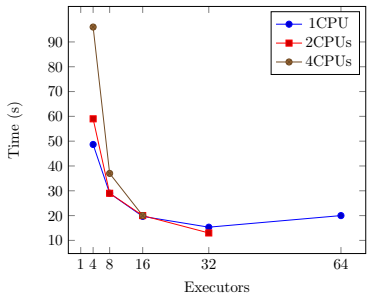
(b) Výpočetní čas pro operaci Flow Aggregation



(c) Výpočetní čas pro Http Request Analysis



(d) Výpočetní čas pro operaci count()



(e) Capture Info: Čas výpočtu pro různou konfiguraci RAM a CPU

Mem (GiB)	vCPU 1	2	4	8	16
1	64	32	16	8	4
2	59	32	16	8	4
4	38	29	16	8	4
8	17	17	14	7	4
16	7	7	7	7	4

(f) Počet Spark Executor instancí pro různé konfigurace MEM a CPU

Obrázek 4: Výpočetní čas pro různé experimenty

– Existence TOR uzlu pro vybranou IP adresu
– Identifikace všech TOR uzlů pro specifikouvanou síť
– Seznam TOR uzlů, které komunikovaly s určitým serverem
– Zjištění zda určitý uživatel používal aktivně TOR síť

6.2 Identifikace uživatelů ze síťových dat

Tento scénář se zaměřuje na analýzu různých síťových aktivit s cílem nalézt elektronické identity uživatelů:

– Nalezení identit v různých datových zdrojích, například logovací soubory, PCAP soubory, authentizační servery, atd.
– Analýza informačních zdrojů a odvození vztahů mezi jednotlivými identitami na základě definovaných pravidel
– Seskupení identit na základě analýzy grafu vztahů mezi jednotlivým identitami.
– Vytvoření databáze identit s možností identifikace souvisejících identit jak v aktuálních datech tak v minulosti.

6.3 Datové zdroje

Platforma bude rozšířena o možnost zpracovávat další datové zdroje:

Netflow ve formátu nfdump Formát NFDUMP je kompaktní formát pro uložení netflow dat. Tento formát umožňuje ukládat také IPFIX data. Pro podporu nfdump jako vstupního formátu pro navrhovaný systém je potřeba vytvořit vhodný adaptér. Tento adaptér musí správně interpretovat flow záznamy ve vstupním souboru. Jakmile budou Netflow záznamy správně načteny do prostředí, je možné na ně použít metody implementované v prototypu.

Pcap soubory v JSON formátu Nástroj TSHARK umožňuje získat informaci o paketech ve formátu JSON. Přestože toto je relativně pomalý způsob zpracování, výhodou je, že takto lze získat plně dekódovaný paket pro velké možství protokolů. Načtení JSON souborů je snadné, neboť každý paket je umístěn na samostatném řádku ve vstupním souboru. Po načtění tohoto vstupu je možné pracovat s pakety stejným způsobem jako v již implementovaném prototypu.

Log soubory ze síťových služeb a zařízení Další typem zdrojových dat jsou soubory s logovacími informacemi. Tyto soubory jsou ve většině případů textové soubory, které mají různou strukturu. Proto je nutné pro každou podporovanou službu mít příslušný dekodér, který dokáže extrahovat zajímavé informace. Kromě textových souborů se uvažuje také podpora souborů EVTX [15], což jsou soubory s logovacími informacemi používané na platformě Windows. Tyto soubory mají formát založený na Binary XML.

6.4 Analýza dat

Budou zkoumány možnosti využití dalších Spark knihoven pro analýzu dat. V současné době, využívá implementace protoypu pouze knihovnu Spark a Spark SQL.

Spark Machine Learning Spark MLlib[8] je kolekce algoritmů strojového učení optimalizovaná pro prostředí Apache Spark. V roce 2018 se uvažuje o použití těchto algoritmů pro úlohy klasifikace dat. Zejména se uvažuje o těchto úlohách:

- Klasifikace síťového provozu podle aplikačních protokolů
- Identifikace vybraných bezpečnostních hrozeb, například malware pomocí jeho charakteristických atributů
- Klasifikace šifrovaného provozu na základě SSL/TLS charakteristik [10]

Spark GraphX Apache Spark Graph X je knihovna implementující grafové algoritmy[9]. GraphX přináší novou třídy Graph, která je reprezentována kolekcí vrcholů a hran. Tyto kolekce jsou implementovány jako RDD a je tedy možné používat veškeré RDD operace. Navíc GraphX implementuje nové grafové algoritmy, které jsou prováděny parallelně.

V projektu budou zkoumány možnosti knihovny GraphX pro analýzu informací, které mají přirozenou grafovou reprezentaci, například BitCoin transakce, grafy identity a podobně.

7 Závěr

Tato zpráva poskytla přehled o stavu platformy pro digitální forenzní analýzu ke konci roku 2017. Platforma se nachází ve stavu experimentálního prototypu jenž demonstruje možnosti zpracování zachycené komunikace ve formě PCAP souborů. Tato výpočetně náročná úloha je řešena distribuovaně pomocí technologie Apache Spark. Realizací prototypu bylo ověřeno, že je možné tuto technologii úspěšně použít pro tento a podobné typy úloh, které jsou typické pro zpracování digitálních forenzních dat. Prototyp byl také použit v sadě experimentů mající za cíl ukázat škálovatelnost navrženého řešení.

Další vývoj zahrnuje především rozvoj prototypu do prostředí, které bude možné dále rozšiřovat o nové datové zdroje a integrovat nové techniky zpracování. Pro tyto účely byly identifikovány další datové zdroje, které budou zpřístupněny pomocí ETL modulů. Pro další rozvoj funkcionality byly vybrány nové případy použití, které se zaměřují na bezpečnostní analýzu síťové komunikace včetně možností identifikace potencionálních hrozeb v šifrovaném datovém toku.

[8] https://spark.apache.org/mllib/
[9] https://spark.apache.org/graphx/

Reference

1. Apache Metron: Real-Time Big Data Security. 2016.
   URL https://metron.incubator.apache.org/
2. Apache Spot (incubating): A Community Approach to Fighting Cyber Threats. 2016.
   URL https://spot.incubator.apache.org/
3. Aupetit, M.; Zhauniarovich, Y.; Vasiliadis, G.; aj.: Visualization of actionable knowledge to mitigate DRDoS attacks. In *2016 IEEE Symposium on Visualization for Cyber Security (VizSec)*, Oct 2016, s. 1–8, doi:10.1109/VIZSEC.2016.7739577.
   URL http://dx.doi.org/10.1109/VIZSEC.2016.7739577
4. Cardenas, A. A.; Manadhata, P. K.; Rajan, S. P.: Big Data Analytics for Security. *IEEE Security Privacy*, ročník 11, č. 6, Nov 2013: s. 74–76, ISSN 1540-7993, doi:10.1109/MSP.2013.138.
   URL http://dx.doi.org/10.1109/MSP.2013.138
5. Cohen, M. I.: PyFlag: An advanced network forensic framework. In *Proceedings of the 2008 Digital Forensics Research Workshop*, DFRWS, Srpen 2008.
   URL http://www.pyflag.org
6. Guarino, A.: *Digital Forensics as a Big Data Challenge*. Wiesbaden: Springer Fachmedien Wiesbaden, 2013, ISBN 978-3-658-03371-2, s. 197–203, doi:10.1007/978-3-658-03371-2_17.
   URL http://dx.doi.org/10.1007/978-3-658-03371-2_17
7. He, L.; Tang, B.; Zhu, M.; aj.: *NetflowVis: A Temporal Visualization System for Netflow Logs Analysis*. Cham: Springer International Publishing, 2016, ISBN 978-3-319-46771-9, s. 202–209, doi:10.1007/978-3-319-46771-9_27.
   URL http://dx.doi.org/10.1007/978-3-319-46771-9_27
8. Jagadish, H. V.; Gehrke, J.; Labrinidis, A.; aj.: Big Data and Its Technical Challenges. *Commun. ACM*, ročník 57, č. 7, jul 2014: s. 86–94, ISSN 0001-0782, doi:10.1145/2611567.
   URL http://doi.acm.org/10.1145/2611567
9. Lukashin, A.; Laboshin, L.; Zaborovsky, V.; aj.: *Distributed Packet Trace Processing Method for Information Security Analysis*. Cham: Springer International Publishing, 2014, ISBN 978-3-319-10353-2, s. 535–543, doi:10.1007/978-3-319-10353-2_49.
   URL http://dx.doi.org/10.1007/978-3-319-10353-2_49
10. Petr Velan, Milan Cermák, Pavel Celeda; Drašar, M.: A survey of methods for encrypted traffic classification and analysis. *Internation Journal of Network Management*, , č. October 2005, 2007: s. 17–31, ISSN 10557148, doi:10.1002/nem.
    URL http://onlinelibrary.wiley.com/doi/10.1002/nem.604/abstract
11. Pilli, E. S.; Joshi, R.; Niyogi, R.: Network forensic frameworks: Survey and research challenges. *Digital Investigation*, ročník 7, č. 1-2, 2010: s. 14–27, ISSN 1742-2876, doi:https://doi.org/10.1016/j.diin.2010.02.003.
    URL
    http://www.sciencedirect.com/science/article/pii/S1742287610000113

12. Promrit, N.; Mingkhwan, A.: Traffic Flow Classification and Visualization for Network Forensic Analysis. In *2015 IEEE 29th International Conference on Advanced Information Networking and Applications*, March 2015, ISSN 1550-445X, s. 358–364, doi:10.1109/AINA.2015.207.
URL http://dx.doi.org/10.1109/AINA.2015.207
13. Quick, D.; Choo, K.-K. R.: Big forensic data reduction: digital forensic images and electronic evidence. *Cluster Computing*, ročník 19, č. 2, 2016: s. 723–740, ISSN 1573-7543, doi:10.1007/s10586-016-0553-1.
URL http://dx.doi.org/10.1007/s10586-016-0553-1
14. Schales, D. L.; Hu, X.; Jang, J.; aj.: FCCE: Highly scalable distributed Feature Collection and Correlation Engine for low latency big data analytics. In *2015 IEEE 31st International Conference on Data Engineering*, April 2015, ISSN 1063-6382, s. 1316–1327, doi:10.1109/ICDE.2015.7113379.
URL http://dx.doi.org/10.1109/ICDE.2015.7113379
15. Schuster, A.: Introducing the Microsoft Vista event log file format. *Digital Investigation*, ročník 4, č. SUPPL., 2007: s. 65–72, ISSN 17422876, doi:10.1016/j.diin.2007.06.015.
16. Siegal, D.; Guo, J.; Agrawal, G.: Smart-MLlib: A high-performance machine-learning library. In *Proceedings - IEEE International Conference on Cluster Computing, ICCC*, 2016, ISBN 9781509036530, ISSN 15525244, s. 336–345, doi:10.1109/CLUSTER.2016.49.
17. Wullink, M.; Moura, G. C. M.; Muller, M.; aj.: ENTRADA: A high-performance network traffic data streaming warehouse. In *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, April 2016, s. 913–918, doi:10.1109/NOMS.2016.7502925.
URL http://dx.doi.org/10.1109/NOMS.2016.7502925
18. Xin, R. S.; Gonzalez, J. E.; Franklin, M. J.; aj.: GraphX: A Resilient Distributed Graph System on Spark. *First International Workshop on Graph Data Management Experiences and Systems*, 2013: str. 2, ISSN 0002-9513, doi:10.1145/2484425.2484427, 1402.2394.