Livre blanc
Supervision de sécurité

Eric Totel Laurent d'Orazio Erwan Le Merrer

3 octobre 2023

Avant-propos

Ce livre blanc se propose, d'une part de mettre en perspective historique et scientifique les mécanismes de supervision de la sécurité, d'autre part proposer, sur la base de cette perspective, des pistes de recherche pouvant être suivies à l'avenir.

Ce livre fait suite à une série de conférences organisées dans le cadre du semestre « supervision de sécurité » (SUPSEC), financé par la Direction Générale de l'Armement (DGA) et opéré par Inria pour le compte de la communauté académique.

Chacune de ces conférences a adressé un thème particulier. Alors que la première s'est focalisée sur l'état de l'art actuel industriel de la supervision, les suivantes ont abordé des thèmes de recherche qui pourraient être menés dans un futur proche et qui, dans certains cas, répondent à des problèmes ouverts de l'industrie. Ainsi 3 grands thèmes se sont succédés : les mécanismes de big data appliqués à la supervision, l'intelligence artificielle au service de ce domaine, et le cas spécifique et important de la détection des APT (*Advanced Persistent Threat*).

Suivant cette organisation et cette phylosophie, ce livre blanc propose un bref état de l'art dans la partie « Etat de l'art industriel » I alors que chacun des thèmes identifiés se trouvent déclinés dans la partie "Perspectives" II.

Remerciements

Les auteurs tiennent à remercier la DGA pour le financement de ce semestre thématique, le centre Inria de l'Université de Rennes pour avoir piloté ce projet, les orateurs des conférences, ainsi que toutes celles et ceux qui se sont investis dans leur organisation.

Introduction

La supervision de sécurité est un domaine initié dans les années 80. Même si les fondations du domaine restent inchangées depuis 40 ans, les techniques pour les mettre en œuvre ont bien sûr largement évolué. Ce livre est donc rédigé en deux temps. D'une part une première partie sur l'état de l'art (partie I) qui se focalise sur l'évolution du domaine et son état actuel du point de vue industriel (qui en soit est un compte-rendu de la première conférence). D'autre part une deuxième partie sur les pistes prometteuses en terme de recherche. Ces pistes sont identifiées en fonction des conférences qui ont eu lieu sur les domaines du big data, de l'intelligence artificielle et de la détection d'APT.

Table des matières

Première partie

État de l'art industriel

Chapitre 1

Evolution de la supervision de sécurité

1.1 Introduction

L'histoire de la supervision de sécurité nous ramène aux années 80, avec le papier fondateur de James P. Anderson ("Computer Security Threat Monitoring and Surveillance") [7]. Il définit les menaces auxquelles sont confrontés les systèmes informatiques et comment les programmes de surveillance peuvent (et doivent) se reposer sur les fichiers d'audit.

Les premiers prototypes de systèmes de détection d'intrusion (IDS, pour *Intrusion Detection Systems*) arrivent dans les années 90, que ce soit au niveau système ou au niveau réseau (Snort, par exemple).

Toutefois, les informations générées par ces IDS sont souvent peu fiables et engendrent de nombreux faux positifs. Apparait alors la notion de SIEM (pour *Security Information and Event Management*), dont le rôle est de permettre de gérer cet énorme afflux d'événements et d'alertes, de diminuer le nombre de faux positifs et de corréler différents événements pour détecter des attaques multi-étapes. Un exemple important de tel outil est QRadar.

Cependant, la supervision ne se limite pas à la détection, et ainsi apparaissent ces dernières années les SOAR (Security Orchestration, Analytics and Reporting) dont la fonction est d'accélérer la réponse à incidents.

1.2 Etat de l'art

Paradoxalement, alors qu'à la fois Anderson et Denning [31] proposaient des approches que par la suite on qualifiera de « comportementales », les travaux du début des années 90 ont privilégié des approches à base de règles expertes ou générées par l'observation du fonctionnement du SI (approche par reconnaissance de motifs ou « par signature »).

Les premiers travaux ont en particulier abordé la problématique des données permettant d'observer l'activité d'un système d'information ou d'un réseau, pour en déduire qu'il se produit (ou pas) des phénomènes malveillants. De nombreuses sources de données ont été utilisées, pour converger sur 3 grandes

familles : l'observation du trafic réseau (essentiellement TCP/IP), des intercepteurs d'événements insérés dans les noyaux des systèmes d'exploitation (essentiellement pour la détection de code malveillant), et des intercepteurs applicatifs embarqués tant dans des clients (par exemple navigateurs) que dans des services.

La fin de la décennie 90 a vu la stabilisation de l'approche de détection par reconnaissance de motifs (par signature), tout particulièrement appliquée à l'analyse de flux réseau. Cette approche a ensuite été utilisée pour identifier des indicateurs de compromission (IoC pour *Indicator of Compromise*), trace(s) quelconque(s) laissée(s) par une attaque sur le système attaqué. La recherche d'IoC est aujourd'hui communément incluse dans les produits de protection (antivirus, sécurité du navigateur, sécurité du poste de travail). L'activité de recherche liée porte plutôt sur la détection de code malveillant. Dans le domaine commercial, il faut par ailleurs noter l'apparition des produits « EDR » (*Endpoint Detection and Response*). Ces produits sont une évolution des anti-virus, des IDS et des firewall, apportant la détection et la réponse à des intrusions.

A partir du milieu des années 90, s'est finalement également développée l'approche comportementale (appelée aussi détection d'anomalies), dans le but de compléter la couverture de la détection, en particulier pour apporter des capacités de détection des nouvelles formes d'attaques (zero day).

Cette approche a donné lieu à de très nombreux travaux de recherche, un grand élan ayant été insufflé par des jeux de données de la DARPA, mis en forme par KDD et qui, malgré toutes leurs imperfections, ont été largement utilisés par la communauté [69] . Un bon descriptif de toutes ces approches comportementales est présenté dans la référence [23] . La détection d'anomalies reste aujourd'hui un domaine très actif, car les taux de faux positifs demeurent élevés et les données à traiter complexes. Aussi, de nombreux travaux tentent aujourd'hui d'appliquer des techniques de *Machine Learning* (ML) pour des environnements et sur des jeux de données très variés.

Les premiers produits commerciaux de détection d'intrusion sont apparus sur le marché commercial autour de 1996 (ISS RealSecure, par exemple). Les utilisateurs ont rapidement constaté la difficulté de traiter des alertes en grand nombre et de faible qualité. Cela a stimulé l'émergence du domaine de la corrélation d'alertes [30] , dont les résultats ont donné naissance aux outils de type « *Security Information and Event Manager* » (SIEM). De très nombreuses approches (là aussi à base de règle ou privilégiant plutôt la détection d'anomalies, cette fois en examinant les alertes) ont été proposées. Actuellement, comme pour la détection, les techniques de Machine Learning sont privilégiées par les chercheurs.

Le volume d'alertes, y compris corrélées, a nécessité le développement d'approches pour automatiser la réponse aux attaques. Aujourd'hui, le terme « *Security Orchestration, Automation and Response* » (SOAR) est utilisé pour décrire le besoin industriel de plates-formes technologiques permettant de gérer globalement le risque. En termes de recherche, cependant, peu de choses ont été proposées au-delà de l'analyse de données de type « *cyber threat intelligence* » (CTI). L'émergence du terme SOAR par rapport au terme SIEM peut être considéré comme une réponse marketing cherchant à masquer sous un changement de dénomination des insuffisances technologiques majeures pour faire face à la pression des attaquants, en gérant le risque et pilotant la cyber sécurité dans les environnements complexes d'aujourd'hui. Ces insuffisances demandent que des travaux de recherche continuent à être conduits.

Des travaux dans le domaine de la détection d'intrusions existe en France depuis le début des années 90. Plusieurs groupes (Rennes, Evry, Toulouse, Sophia) ont émergé et on fait de nombreuses propositions. D'autres travaux existent aussi en dehors de ces pôles. A titre d'exemple, des mécanismes de détection originaux ont été proposés, comme par exemple la détection de flux d'information illégaux au regard d'une politique de sécurité [102] ; des approches de corrélation d'alertes tenant compte du contexte de production de ces alertes ont été proposées [16] ; la France a aussi produit l'une des rares séries de travaux sur la remédiation [58], allant jusqu'au développement de prototypes au sein des projets européens MASSIF et SOCCRATES.

Le positionnement français a suivi ces dernières années celui de la communauté internationale du domaine : abandon (malheureux à notre sens) de la production « intelligente » d'alertes, report (en conséquence) de la plupart des efforts sur le post-traitement des alertes (corrélation), trop peu d'effort sur la réaction/atténuation. Les travaux actuels font apparaître un intérêt marqué pour la détection d'anomalies (détection comportementale), tant en phase de détection qu'en phase de corrélation, en particulier par utilisation de mécanismes relevant du ML. On note cependant que les travaux réalisés jusqu'à présent, en France comme ailleurs, ne sont pas suffisamment convaincants. Coté recherche, les publications existent mais les résultats restent difficiles à valider et à interpréter. Aucun des résultats obtenus n'est vraiment reproduit en environnement réel, si bien qu'aucune proposition n'a su s'imposer et créer des débouchés industriels significatifs.

Chapitre 2

Problématique

La supervision de sécurité est confrontée à plusieurs problématiques.

La première concerne la quantité de données à traiter. Dans un SOC (*Security Operation Center*), plusieurs téraoctets de données d'audit peuvent être générés chaque jour. Ainsi, face à cet afflux d'information, il est difficile pour les analystes de discriminer efficacement ce qui est normal de ce qui est anormal. L'accès rapide à la base de données d'audit et son interrogation reste donc au cœur du problème, mais ce n'est pas l'unique problème.

En effet, il est par ailleurs nécessaire de faire le lien entre la CTI (*Cyber Threat Intelligence*) et les données d'audit afin d'être capable de qualifier ces dernières (ce qui est réalisé dans les SOAR).

Deuxième partie

Perspectives

Chapitre 3

Supervision et big data

3.1 Introduction

Ces dernières années, la recherche dans les bases de données et les systèmes distribués a abouti à des systèmes permettant le traitement de grands volumes de données sur des infrastructures à grande échelle telles que le Cloud/Fog/Edge Computing (MapReduce, HBase, Spark, etc.). Parallèlement, la gestion des données sur de nouveaux matériels (GPU, FPGA par exemple) a été développée.

L'objectif de ce chapitre est de discuter des orientations de recherche en matière de sécurité avec les Big Data. Plus précisément, l'atelier vise à répondre aux deux questions clés suivantes : comment les technologies Big Data peuvent-elles être appliquées à la surveillance de la sécurité ? et quels sont les défis de recherche dans ce domaine ?

En particulier, nous considérons les orientations suivantes :

— Systèmes : nous discutons de certaines solutions représentatives au sein des écosystèmes Big Data et de surveillance de la sécurité. En particulier, nous présentons des solutions open-source, à savoir Punch et Warp10.
— Analyse de graphes et calcul distribué : parmi les dimensions théoriques couvertes par la surveillance de sécurité à grande échelle, nous discutons des solutions existantes en matière d'analyse de graphes et de traitement distribué.
— Confidentialité : tout en fournissant des outils pour détecter les anomalies et les menaces, il est obligatoire de garantir un autre aspect important de la cybersécurité, à savoir la confidentialité.
— Matériel : l'analyse des données peut être accélérée grâce à de nouveaux matériels. Nous considérons en particulier trois opportunités possibles : NVM, GPU et FPGA.

Ce chapitre est organisé comme suit. La section 2 décrit les motivations. La section 3 se concentre sur les systèmes. La section 4 aborde ensuite les graphes et le traitement distribué. La section 5 est dédiée à la confidentialité. La section 6 présente les nouveaux matériels. Enfin, la section 7 fournit les conclusions.

```
host    logname time    method  url     response        bytes   referer useragent
199.72.81.55 - - [01/Jul/1995:00:00:01 -0400] "GET /history/apollo/ HTTP/1.0" 200 6245
unicomp6.unicomp.net - - [01/Jul/1995:00:00:06 -0400] "GET /shuttle/countdown/ HTTP/1.0" 200 3985
199.120.110.21 - - [01/Jul/1995:00:00:09 -0400] "GET /shuttle/missions/sts-73/mission-sts-73.html HTTP/1.0" 20
burger.letters.com - - [01/Jul/1995:00:00:11 -0400] "GET /shuttle/countdown/liftoff.html HTTP/1.0" 304 0
199.120.110.21 - - [01/Jul/1995:00:00:11 -0400] "GET /shuttle/missions/sts-73/sts-73-patch-small.gif HTTP/1.0"
burger.letters.com - - [01/Jul/1995:00:00:12 -0400] "GET /images/NASA-logosmall.gif HTTP/1.0" 304 0
burger.letters.com - - [01/Jul/1995:00:00:12 -0400] "GET /shuttle/countdown/video/livevideo.gif HTTP/1.0" 200
205.212.115.106 - - [01/Jul/1995:00:00:12 -0400] "GET /shuttle/countdown/countdown.html HTTP/1.0" 200 3985
d104.aa.net - - [01/Jul/1995:00:00:13 -0400] "GET /shuttle/countdown/ HTTP/1.0" 200 3985
129.94.144.152 - - [01/Jul/1995:00:00:13 -0400] "GET / HTTP/1.0" 200 7074
unicomp6.unicomp.net - - [01/Jul/1995:00:00:14 -0400] "GET /shuttle/countdown/count.gif HTTP/1.0" 200 40310
unicomp6.unicomp.net - - [01/Jul/1995:00:00:14 -0400] "GET /images/NASA-logosmall.gif HTTP/1.0" 200 786
```

FIGURE 3.1 – Exemple de données de journaux d'un serveur HTTP server

3.2 Motivation

Le développement de l'informatique dans notre vie quotidienne a accru les risques d'attaques potentielles. Par exemple, alors que les villes intelligentes tendent à fournir une gestion efficace des ressources énergétiques, elles peuvent entraîner une panne de courant.

Dans un tel contexte, les réseaux et les télécommunications revêtent un intérêt particulier. D'une part, la 5G améliore la sécurité et la confidentialité, mais d'autre part, le nombre de cas d'utilisation et la complexité du système augmentent également : nouvelles technologies radio ; séparation du plan de contrôle et du plan utilisateur ; découpage du réseau en tranches (eMBB, URLLC et MIoT) ; architecture basée sur les services ; réseaux définis par logiciel, voire virtualisation des fonctions réseau. Plus de détails sur l'analyse de sécurité du système 5G sont disponibles dans [48].

Cette section présente les principaux concepts de ce rapport, à savoir la surveillance de la sécurité et son lien avec la gestion des données dans le cloud computing. Elle décrit les technologies Big Data dans le cloud computing qui peuvent être utilisées.

3.2.1 Surveillance de la sécurité

La surveillance de la sécurité consiste à collecter et à analyser des indicateurs pour détecter les menaces potentielles en matière de sécurité. Par exemple, les administrateurs peuvent accéder à des tableaux de bord pour analyser les journaux. La figure 3.1 présente des exemples de données collectées à partir de certains journaux d'un serveur HTTP.

Dans un tel contexte, l'outil de surveillance repose sur différents types de requêtes. Elles peuvent consister en des requêtes d'intervalle, par exemple pour obtenir le trafic sur une période donnée (par exemple entre 0h00 et 1h00 le premier juillet). Elles peuvent également inclure des requêtes d'agrégation, par exemple pour repérer la machine ayant le plus de trafic. De toute évidence, la surveillance de la sécurité est de plus en plus complexe, avec le nombre croissant de systèmes à surveiller et la complexité de leur environnement.

3.2.2 Cloud Computing et Big Data

Les experts en surveillance de la sécurité sont directement confrontés aux problématiques du Big Data. En effet, ils doivent surveiller efficacement divers

systèmes, et leur nombre augmente, en particulier en raison de la quantité croissante d'objets connectés (IoT). De plus, ils sont intéressés par la collecte et le stockage des données sur de longues périodes, afin de détecter des intrusions sophistiquées avec des événements répartis sur de grandes plages de temps.

Au cours de la dernière décennie, de nombreux systèmes de gestion de données ont été proposés pour répondre aux comportements spécifiques du cloud computing [9] et du Big Data, en particulier la scalabilité/élasticité, afin d'exploiter les ressources des grands centres de données. Certaines solutions consistent en des systèmes avec un langage déclaratif sur des systèmes de fichiers distribués comme Pig [78], SCOPE [22], Hive [101] ou Jaql [15]. D'autres consistent en un magasin de données de type colonnes comme Big Table [24] ou Cassandra [62]. Au-dessus de ces systèmes, différents moteurs d'exécution peuvent être déployés, tels que MapReduce [29], Tez [89], Spark [10] ou Flink [21]. Ces outils peuvent être considérés comme des éléments essentiels pour gérer un grand nombre de journaux dans la surveillance de la sécurité.

3.3 Systèmes

La communauté Big Data est dynamique et l'écosystème est donc riche. À titre d'illustration, on peut voir la représentation donnée par Matt Truck [1].

Dans cette section, nous citerons quelques systèmes représentatifs de classes de solutions actuellement utilisées dans la surveillance de la sécurité afin de donner un aperçu des outils utilisés dans ce contexte.

3.3.1 Logiciels de surveillance de la sécurité

Les solutions de surveillance de la sécurité dédiées permettent de collecter, indexer et corréler des données pour créer des collections. Ces collections sont ensuite utilisées pour générer des rapports et des alertes, en plus de visualisations et de tableaux de bord. Splunk [2], ElastisSearch [3] et SEKOIA sont des logiciels représentatifs.

Par exemple, SEKOIA [4] permet aux équipes d'experts en cybersécurité de se concentrer sur l'anticipation et l'automatisation.

3.3.2 Plateformes de traitement et d'analyse des données

La PunchPlatform est une plateforme Big Data axée sur le déploiement industriel, le traitement critique de bout en bout des données et les applications d'analyse des données. Elle offre un concept de pipeline de données mis en œuvre sur Apache Kafka [5]. Cela permet aux utilisateurs de concevoir des pipelines de données industriels où ils peuvent brancher leurs traitements où ils en ont besoin, depuis l'étiquetage simple des données jusqu'aux algorithmes complexes d'apprentissage automatique distribués.

L'une de ses principales différences par rapport à une configuration Elastic-Search-Logstash-Kibana (ELK) est de permettre à l'utilisateur de déployer des

1. https ://mattturck.com/data2020/
2. https ://www.splunk.com
3. https ://www.elastic.co
4. https ://www.sekoia.io
5. https ://kafka.apache.org

traitements arbitraires dans les moteurs Storm [103] et Spark, et pas seulement des filtres logstash. La PunchPlatform est équipée de parseurs de journaux prêts à l'emploi. Ceux-ci sont déployés automatiquement dans le flux de données. De plus, un langage de script est disponible pour écrire des parseurs personnels sur la PunchPlatform.

La plateforme permet la recherche (en utilisant Kibana ou les API Elastic-Search natives) ainsi que l'apprentissage automatique. Elle offre des fonctions supplémentaires telles que la multi-tenance, l'archivage à long terme utilisant le stockage d'objets CEPH et le déploiement multi-site.

3.3.3 Base de données de séries temporelles

Une série temporelle est une série de points de données indexés (ou répertoriés ou représentés graphiquement) dans l'ordre du temps. Le plus souvent, une série temporelle est une séquence prise à des points dans le temps successifs et également espacés. Ainsi, il s'agit d'une séquence de données à temps discret. Des exemples de séries temporelles sont les hauteurs des marées océaniques, les comptages de taches solaires et la valeur de clôture quotidienne du Dow Jones Industrial Average. Une base de données de séries temporelles (TSDB) est un système logiciel optimisé pour stocker et servir des séries temporelles à l'aide de paires associées de temps et de valeurs. Des exemples de TSDB incluent InfluxDB [6] et Prometheus [7]. Dans cet article, nous nous concentrerons sur Warp 10.

La plateforme centrale de Warp 10 [8] est conçue pour gérer et simplifier le traitement des séries chronologiques. Elle comprend une base de données de Geo Time Series™ (GTS) et un moteur analytique. Chacun peut être utilisé séparément, mais bien sûr, ils fonctionnent très bien ensemble. Chaque mesure possède un temps spécifique, une valeur et des métadonnées spatiales facultatives telles que des coordonnées géographiques et/ou une altitude. Ces mesures que nous appelons des Geo Time Series™ (GTS).

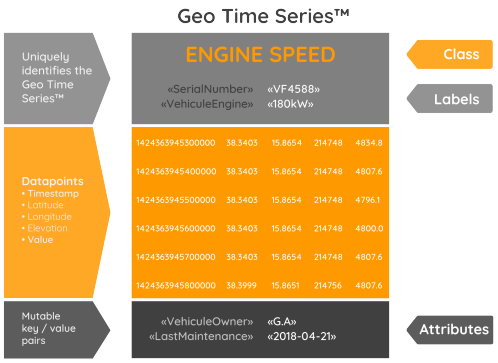


FIGURE 3.2 – Représentation des Geo Time Series™ de Warp 10™

La figure 3.2 illustre une GTS. L'en-tête présente la classe (vitesse du mo-

6. https ://www.influxdata.com/
7. https ://prometheus.io
8. https ://www.warp10.io

teur) et certaines étiquettes (numéro de série et moteurs de véhicules). Les points de données sont composés d'un horodatage, d'une latitude, d'une longitude, d'une altitude et d'une valeur. La partie inférieure fournit des paires clé/valeur supplémentaires (propriétaire du véhicule et dernière maintenance).

WarpScript™ est le langage analytique dédié de Warp 10, spécialement conçu pour l'analyse complexe de données de séries chronologiques à toutes les échelles. Un exemple de Warp 10 est illustré par la figure 3.3.3. WarpScript™ offre plus de 1000 fonctions, allant des statistiques simples aux algorithmes complexes tels que la détection de motifs et d'anomalies.

WarpScript™ est un langage de programmation basé sur le flux de données. Cela diffère d'un langage de requête qui peut récupérer des données et effectuer des calculs simples. WarpScript™ possède la puissance d'un langage de programmation Turing complet, avec des structures conditionnelles, des boucles et des transferts de contrôle asynchrones. Le moteur d'exécution WarpScript™ est la partie de la plateforme Warp 10™ qui exécute le code WarpScript™. Il suffit de soumettre le code via une requête HTTP, et le moteur l'exécutera près des données et renverra le résultat sous la forme d'un objet JSON qui peut être intégré dans une application. Le langage WarpScript™ est également intégré dans de nombreux langages et environnements de traitement des données existants tels que R, Python, Zeppelin, Jupyter, Spark, Pig, Storm ou NiFi, avec la capacité d'utiliser le même code WarpScript™ dans tous ces environnements. Cela augmente considérablement l'efficacité.

```
[
$read_token
'gov.noaa.storm.wind'
{}
'2015-02-01T00:00:00.000z'
'2015-01-01T00:00:00.000z'
] FETCH 'fetch_wind' STORE
[ $fetch_wind bucketizer.mean NOW 1 d 0 ]
BUCKETIZE 'bucketizedWind' STORE
```

Dans WarpScript™, comme dans tous les langages de programmation concaténatifs, chaque expression est une fonction et la juxtaposition des expressions représente la composition des fonctions, à la manière des pipelines Unix. WarpScript™ gère un pipeline de données avec lequel toutes les fonctions interagissent, en récupérant leurs arguments en vidant le pipeline et en le remplissant avec leurs résultats. Le pipeline de données est passé de fonction en fonction lors de l'exécution du code WarpScript™. Le moteur d'analyse de Warp 10™, le moteur d'exécution WarpScript™ de Warp 10™ accessible via l'endpoint /api/v0/exec, renvoie une liste JSON contenant les éléments du pipeline de données après l'exécution du code. Le premier élément de cette liste contient l'élément le plus récent du pipeline.

3.4 Analyse de graphes et calcul distribué

3.4.1 Analyse de graphes

Un graphe est une structure de données fondamentale composée de nœuds et d'arêtes. Les graphes sont omniprésents dans différents domaines tels que le graphe du Web [39], les réseaux sociaux [40], la vision par ordinateur [54] et les expressions géniques [59]. Les tâches typiques de l'analyse de graphes sont le regroupement de graphes, la classification de graphes et la prédiction de liens. Les embeddings de nœuds et les réseaux neuronaux graphiques (GNN) sont largement utilisés pour de telles tâches afin d'obtenir des résultats d'analyse de haute qualité.

L'embedding de nœud est une technique qui permet d'incorporer chaque nœud d'un graphe dans un espace multidimensionnel qui capture la proximité des nœuds dans le graphe. La proximité d'un nœud est définie en fonction de ses caractéristiques de nœud et de sa topologie de voisins. Par exemple, deux personnes dans un réseau social devraient avoir des embeddings similaires si leurs caractéristiques (comme les hobbies) sont similaires et qu'ils sont de bons amis. Cette proximité est connue pour être efficace pour les tâches d'analyse de graphes mentionnées ci-dessus. En particulier, il existe différents algorithmes qui exploitent différents types de proximités, les proximités du 1er ordre/2e ordre (aspect microscopique) et les proximités d'ordre supérieur (aspect macroscopique). Par exemple, le clustering spectral [73] utilise la proximité du 1er ordre (distance à un saut) pour calculer les clusters de graphes. SCAN [109] est une autre technique de clustering basée sur la densité qui utilise la proximité du 2e ordre (distance à deux sauts). Les réseaux de convolution graphiques récents (GCN) utilisent généralement les proximités du 1er et du 2e ordre, appelées GCN à deux couches, pour la classification de nœuds semi-supervisée. Les GCN ont eu une grande influence sur de nombreux articles ultérieurs, cependant, ils présentent deux inconvénients lorsqu'ils augmentent le nombre de couches de convolution afin d'utiliser efficacement un petit nombre d'étiquettes : le surajustement et le lissage excessif. Les GCN à plusieurs couches ont tendance à surajuster car leur nombre de paramètres est élevé. De plus, ils ont tendance à être trop lissés : les opérations de convolution font souvent en sorte que l'embedding de tous les nœuds soit similaire, ce qui rend difficile la détermination de la frontière de classe des tâches de classification.

Pour résoudre ces problèmes, a été proposé le réseau ANEPN (Adaptive Node Embedding Propagation Network) [76] qui utilise des proximités d'ordre supérieur en augmentant de manière adaptative le nombre de sauts de propagation. La nouveauté de l'ANEPN est qu'il maintient le nombre de couches à deux (ce qui évite le surajustement) et apprend des modèles en minimisant la perte combinée de proximité et de perte d'anti-proximité (ce qui sépare avec succès le nombre d'opérations de convolution du maintien des proximités afin d'éviter le lissage excessif).

L'évaluation comparative de différents GCN a également été abordée [67] afin de clarifier leurs avantages/inconvénients en utilisant un générateur de graphes synthétiques [68]. En générant divers graphes synthétiques, nous révélons que 1) les GNN, y compris les méthodes de pointe, souffrent d'un problème de déséquilibre de classe qui détériore généralement les performances de classification multi-classes, et 2) les GNN qui généralisent aux graphes ayant peu

d'arêtes dans chaque classe (que nous appelons graphes hétérophiles) fournissent des gains marginaux de performance de classification dans un environnement d'hétérophilie par rapport à un classifieur agnostique du graphe, tel que MLP (perceptron multicouche).

3.4.2 Analyse de graphes distribuée

L'informatique en nuage (cloud computing) est une approche prometteuse pour effectuer des analyses de graphes sur des données à grande échelle. Cependant, il existe deux problèmes fondamentaux liés à l'analyse de graphes en utilisant des serveurs distribués dans le cloud.

Le premier problème est celui de la partition des graphes. Les moteurs de graphes distribués exécutent des processus analytiques après avoir partitionné les données du graphe d'entrée et les avoir assignées aux ordinateurs distribués. La qualité de la partition du graphe influence largement le coût de communication et l'équilibrage de charge entre les ordinateurs pendant le processus d'analyse. Une technique efficace de partitionnement de graphes a été proposée [79] qui permet d'obtenir à la fois un faible coût de communication et un bon équilibrage de charge entre les ordinateurs. Cette technique produit des clusters équilibrés en étendant un regroupement de graphes efficace basé sur la modularité [92]. Cette technique a été mise en œuvre sur le moteur de graphes distribués PowerGraph. Les résultats montrent que cette technique de partitionnement réduit le coût de communication et améliore ainsi le temps de réponse des modèles d'analyse de graphes.

Le deuxième problème concerne les algorithmes d'analyse de graphes ou les techniques d'apprentissage automatique qui sont itératifs par nature. Cependant, les principaux frameworks distribués tels que MapReduce ou Spark ne sont pas optimisés pour le traitement itératif. OptIQ [80], une approche d'optimisation de requêtes pour les requêtes itératives dans un environnement distribué a été proposée. OptIQ élimine les calculs redondants entre différentes itérations en étendant les techniques traditionnelles de matérialisation de vues et d'évaluation de vues incrémentales. L'efficacité d'OptIQ a été vérifiée à l'aide des requêtes de PageRank et de regroupement k-means sur des ensembles de données réelles. Les résultats montrent qu'OptIQ atteint une grande efficacité, jusqu'à cinq fois plus rapide que ce qui serait possible sans éliminer les calculs redondants entre les itérations.

3.5 Partage de données conscient de la confidentialité et de la sécurité

Les avancées technologiques telles que les appareils IoT, les systèmes cyberphysiques, les appareils mobiles intelligents, les systèmes cloud, l'analyse des données, les réseaux sociaux et les capacités de communication accrues permettent de capturer, de traiter rapidement et d'analyser d'énormes quantités de données à partir desquelles extraire des informations cruciales pour de nombreuses tâches critiques telles que la sécurité des soins de santé et la cybersécurité. Dans le domaine de la cybersécurité, ces tâches comprennent l'authentification des utilisateurs, le contrôle d'accès, la détection d'anomalies, la surveillance des utilisateurs et la protection contre les menaces internes. En

collectant et en exploitant des données sur les déplacements des utilisateurs, les contacts et les épidémies de maladies, il est possible de prédire la propagation des maladies dans des zones géographiques. Et ce ne sont là que quelques exemples.

3.5.1 Confidentialité

L'utilisation de données pour ces tâches soulève cependant de graves préoccupations en matière de confidentialité. Les données collectées, même si elles sont anonymisées en supprimant les identifiants tels que les noms ou les numéros de sécurité sociale, peuvent permettre de réidentifier les individus auxquels des données spécifiques sont liées lorsqu'elles sont associées à d'autres données. De plus, les organisations, telles que les agences gouvernementales, ont souvent besoin de collaborer sur des tâches de sécurité, ce qui implique l'échange de jeux de données entre différentes organisations, rendant ainsi ces ensembles de données accessibles à de nombreuses parties différentes. Les violations de la confidentialité peuvent se produire à différents niveaux (par exemple, réseaux, hôtes, applications) et composants de nos systèmes interconnectés. Un exemple d'attaque contre la confidentialité dans le contexte des réseaux cellulaires est l'attaque par canal auxiliaire ToRPEDO qui exploite le protocole de paging pour suivre les utilisateurs [49]. D'autre part, si l'on considère les applications mobiles, ces applications présentent des vulnérabilités, telles que celles liées à l'authentification [65, 66], ce qui entraîne un manque de sécurité, ce qui compromet ensuite la confidentialité. Il est important de mentionner que la sécurité et la confidentialité sont deux exigences différentes, mais la sécurité est une condition préalable à la confidentialité. L'utilisation de techniques d'apprentissage automatique menace davantage la confidentialité en raison d'attaques telles que les attaques d'inversion par lesquelles une partie peut déduire le contenu sensible des échantillons de données utilisés pour l'apprentissage. Enfin, l'adoption croissante des appareils portables et la diffusion continue des données à partir de ces appareils permettent à une partie de collecter des données géotemporelles détaillées sur les individus.

Il semblerait donc que la protection de la vie privée soit aujourd'hui impossible. Cependant, de nombreuses techniques de confidentialité ont été proposées au fil des ans, notamment des techniques de liaison de données préservant la confidentialité, des protections contre les attaques d'inversion d'apprentissage automatique, des anonymiseurs de réseau, des techniques de calcul multipartite sécurisé (SMC), le chiffrement homomorphique, la gestion de l'identité numérique préservant la confidentialité, y compris les systèmes de pseudonymes, les ponctuations de contrôle d'accès (AC) pour les données en continu [72], et le "mode" anonyme pour les applications mobiles [91]. La question principale est donc la suivante : *qu'est-ce qui est nécessaire pour assurer la confidentialité ?* Ce qui est nécessaire, c'est de combiner ces approches pour une "protection de la confidentialité en profondeur" en développant des environnements de préservation de la confidentialité holistiques. En effet, les utilisateurs considèrent la confidentialité comme importante, mais ils estiment souvent que la confidentialité est complexe à gérer. Cependant, une question clé est "confidentialité personnelle versus sécurité collective" ? Plus précisément : (i) Comment pouvons-nous permettre aux personnes de faire leurs choix concernant cette question ? (ii) Comment pouvons-nous concilier ces deux objectifs apparemment contradic-

toires ? Nous pensons que répondre à ces questions est un défi qui nécessite des approches facilitant la compréhension par les utilisateurs des risques/bénéfices liés à la divulgation de certaines de leurs données personnelles, ainsi que des mécanismes leur permettant de prendre conscience de la manière dont leurs données sont utilisées et de participer aux processus associés. La transparence des données [14] et l'utilisation basée sur des politiques de données sont deux éléments clés pertinents pour ces problématiques.

3.5.2 Partage de données conscient des politiques de contrôle d'accès

Avec l'avènement des technologies de l'information récentes et l'augmentation de la quantité de données, de plus en plus d'entreprises et d'organisations collaborent en partageant et en échangeant des données à des fins d'apprentissage et de recherche. Pour assurer efficacement la sécurité et la confidentialité, les propriétaires de données attachent un ensemble de règles définies comme une politique de contrôle d'accès [38]. Cependant, lorsque les données sont partagées entre plusieurs sources, il peut y avoir des chevauchements de données. Ces redondances peuvent constituer une menace lorsque les enregistrements de la même entité ne sont pas considérés au même niveau de confidentialité [36]. Dans cette situation, il est nécessaire de mettre en place un filtrage approprié des réponses à une requête. Par conséquent, pour assurer la sécurité et la confidentialité des données, chaque source, construite de manière indépendante des autres, définit sa propre politique de contrôle d'accès. Cette dernière fournit des informations considérées comme sensibles et qui ne doivent donc pas être divulguées.

Des travaux de recherche se sont intéressés à la conception et mise en œuvre d'un cadre permettant un partage sécurisé de données entre deux sources [57]. Le partage de données repose sur l'établissement de correspondances entre les entités des deux sources. Ils s'intéressent à l'utilisation de règles de correspondance d'entités [96] entre les instances afin d'augmenter le résultat des requêtes tout en assurant l'application des politiques de sécurité. De plus, ces résultats cherchent à combler le fossé en matière de sécurité qui apparaît lorsque deux enregistrements, provenant de sources différentes et représentant la même entité du monde réel, ne sont pas considérés avec le même degré de sensibilité.

A d'abord été étudié le problème de la publication de données en présence de règles de contrôle d'accès. Le contexte a été considéré comme celui où une source de données est décrite par un ensemble de vues de publication et de règles de restriction d'accès. Une vue est une table représentant un résultat de requête destiné à être publié. L'objectif est de détecter les vues qui divulguent des informations sensibles et, au lieu de les neutraliser, une révision des vues est proposée [3]. Cette approche utilise les conditions nécessaires et suffisantes pour qu'une vue soit conforme à une demande de politique. Un travail préliminaire consiste en une méthode indépendante des données pour réviser les vues qui ne préservent pas la confidentialité. L'objectif de ce processus de révision est de trouver un équilibre entre la restriction d'accès aux données et la disponibilité des données.

Ensuite, une méthodologie axée sur la correspondance des entités et les politiques pour fournir un cadre sécurisé de partage de données est proposée [2], [4]. Est présenté un algorithme permettant de traduire une requête soumise à

un schéma en une requête augmentée pour l'autre schéma afin de capturer les tuples concernés, sur la base de règles de correspondance d'entités. Ensuite, est proposée une méthodologie pour répondre aux requêtes tout en maximisant le partage et en préservant les politiques de contrôle d'accès locales en évitant toute fuite d'inférence qui pourrait résulter de la correspondance des entités.

3.6 Matériel

3.6.1 Nouvelles technologies de mémoire et de stockage

De nouvelles mémoires émergent et l'hétérogénéité des systèmes de stockage augmente. Dans un contexte centré sur les données, les applications ont besoin de plus de puissance de traitement pour gérer de grandes quantités de données. Les systèmes à plusieurs cœurs ont été adoptés pour augmenter la puissance de traitement. Cependant, cela met encore plus de pression sur les sous-systèmes de mémoire pour qu'ils fonctionnent de manière efficace. Heureusement, les révolutions de la fabrication de puces et de la conception matérielle permettent l'émergence de nouveaux dispositifs de mémoire/stockage non volatils à ultra-faible latence et à ultra-haute capacité (par exemple, Intel Optane [53] et OCSSD [8], déjà disponibles sur le marché ou presque). Ces nouveaux dispositifs peuvent flouter, voire éliminer la frontière entre la couche de traitement et la couche d'E/S dans l'architecture de Von Neumann. Néanmoins, la conception de la pile logicielle héritée (par exemple, la mise en mémoire tampon redondante des dispositifs vers le système et les applications) constitue un fardeau important et entrave l'amélioration des performances du système d'applications à usage intensif de données en exploitant pleinement les nouveaux dispositifs de mémoire/stockage émergents.

Ces nouvelles mémoires émergentes perturbent la communauté scientifique en poussant à revoir la conception des algorithmes et des applications. La mémoire flash est déjà considérée comme une technologie mature, utilisée comme support de stockage hautes performances dans les systèmes embarqués, les infrastructures de cloud computing et de calcul haute performance (HPC), voire dans les ordinateurs portables courants. Plusieurs algorithmes et applications traditionnels ont été révisés afin de tirer pleinement parti des propriétés de la mémoire flash [18], tels que les systèmes embarqués [77], les bases de données [61], les systèmes de stockage en cloud [17] ou le calcul haute performance (HPC) [75].

D'autres mémoires non volatiles (NVM) sont étudiées pour compléter la hiérarchie mémoire traditionnelle [19]. Parmi celles-ci, on peut citer la MRAM (Magnetoresistive RAM [60]), qui pourrait tirer parti de la faible puissance de fuite pour améliorer l'efficacité énergétique de plusieurs applications dans les systèmes embarqués. La FeRAM (Ferroelectric RAM) est déjà utilisée dans plusieurs plates-formes basées sur des DSP ou dans des équipements automobiles pour l'enregistrement de données, et constitue un bon choix pour l'électronique portable [41]. La mémoire à changement de phase (PCM)[1], basée sur la résistivité d'un alliage de chalcogénures pour représenter les bits, est déjà largement déployée dans les SSD Intel Optane en raison de ses performances intéressantes et de ses propriétés énergétiques. La mémoire résistive (ReRAM), compatible avec les processus de fabrication semi-conducteurs classiques, fait partie de la classe des memristors[110, 99] et est également à l'étude depuis plus d'une

décennie.

En plus de ces nouvelles technologies qui optimisent la latence, le débit de transfert et le temps de réponse pour accéder aux données, de nouvelles pistes de recherche consistent à éviter de tels transferts en déployant la puissance de calcul près du stockage ou en effectuant des calculs près de la mémoire. Les dispositifs de stockage computationnel permettent d'exécuter des logiciels au sein du dispositif de stockage, déchargeant ainsi le processeur, la mémoire et les bus de cette charge [11]. Le calcul en mémoire permet d'avoir une puissance de calcul au sein du tableau de mémoire [71], évitant les transferts de données entre la mémoire et le processeur.

L'émergence de ces nouvelles technologies de mémoire et de stockage rend la supervision et la sécurité des données plus complexes en raison de l'hétérogénéité des technologies de stockage et de mémoire, de l'hétérogénéité de la pile logicielle déployée sur celles-ci et de la disparité des applications utilisées.

3.6.2 Nouveau traitement

Cadre de mise en cache sémantique vers l'accélération FPGA

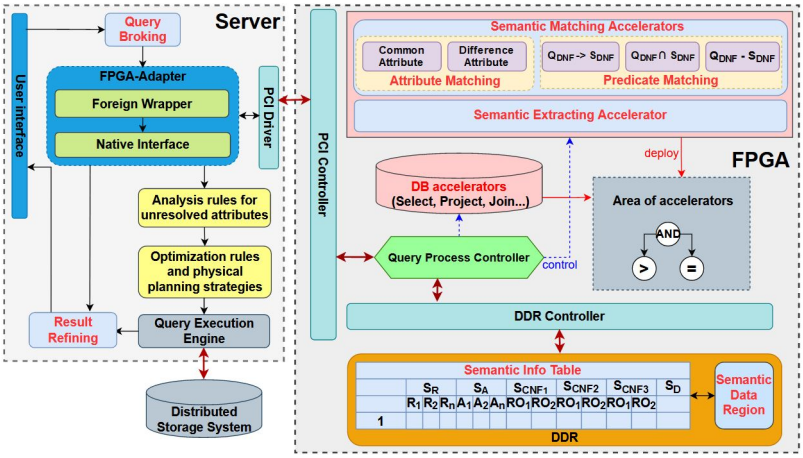


FIGURE 3.3 – MASCARA FPGA

Avec l'émergence de nouveaux systèmes de gestion de données (DMS) dans le contexte du *big data et du cloud computing*, la mise en cache des données est devenue importante car elle permet de réduire l'exécution inutile des requêtes. Étant donné la latence relativement élevée dans la communication entre la couche de calcul et la couche de stockage du DMS, nous considérons que la mise en cache des données au niveau de la couche de calcul est désormais plus importante. Ainsi, nous avons besoin d'un cadre de mise en cache en tant que système de gestion de cache (CMS) dans la couche intermédiaire du DMS, ce qui facilite le processus de construction de services de cache dans diverses applications.

Cependant, bien que des frameworks de services de cache aient été étudiés, la plupart d'entre eux sont présentés par rapport aux mécanismes de mise en cache traditionnels (mise en cache de pages ou de blocs), tels que les services de cache

adaptables (ACS) [33], Amazon Elastic Cache [6], et la mise en cache en mémoire Redis [85]. Comme approche alternative, la mise en cache sémantique (SC) permet d'exploiter les ressources dans le cache et les connaissances contenues dans les requêtes [28]. Ainsi, SC peut être considérée comme un candidat à exploiter dans la couche intermédiaire du DMS. Néanmoins, la complexité de la réécriture des requêtes en SC (problème NP-complet [87, 44, 43]) peut entraîner une surcharge élevée en raison de ses calculs excessifs. Il est à noter qu'en plus de l'algorithme de réécriture des requêtes, la capacité de l'infrastructure joue un rôle important pour résoudre le problème présenté en SC. De plus, aucune des études précédentes, telles que [98, 32, 86, 55, 56, 42, 63, 88, 34, 35, 104], n'a présenté SC en tant que système de gestion de cache (CMS) dans la couche intermédiaire du DMS.

FPGA

Pour surmonter le problème de la réécriture des requêtes en SC, il existe deux approches en termes de mise à l'échelle (mise à l'échelle verticale) avec des ressources matérielles.

En particulier, la première approche consiste à améliorer les unités centrales de traitement (CPU) avec la technologie multi-cœurs ou à en utiliser davantage. Étant donné que les CPU atteignent une limite de "mur de puissance" (loi de Moore), en tant qu'approche alternative, il a été proposé de les remplacer ou de les accélérer avec du matériel spécialisé de calcul intensif, en particulier les réseaux de portes programmables sur terrain (FPGA). Les FPGA ont été considérés comme de bons candidats en raison de leur parallélisme élevé de multi-tâches, de leur reconfigurabilité, de leur faible consommation d'énergie et de leur capacité à être connectés au CPU en tant qu'accélérateur de périphérique d'E/S [37]. De plus, les FPGA ont été acceptés progressivement dans de nombreuses études, y compris les accélérations analytiques de base de données, le traitement massivement parallèle (MPP) ou même des produits commerciaux ([90, 94, 106, 81, 100, 25, 5]). Cependant, les réseaux de portes programmables sur terrain (FPGA) n'ont pas encore été envisagés pour gérer les calculs excessifs de la réécriture des requêtes en SC.

Par conséquent, notre objectif est de combiner trois aspects : le cadre du système de gestion de cache (CMS), SC et l'accélération de base de données basée sur FPGA afin d'accélérer le traitement des requêtes de plage dans le domaine des données massivement distribuées. Selon cet objectif, nous réalisons les contributions suivantes :

1) ModulAr Semantic CAching fRAmework (MASCARA) dans la couche intermédiaire du DMS [52]. MASCARA divise et regroupe les fonctionnalités, les calculs et les procédures de SC en modules et en étapes. Plus précisément, ce travail peut être réalisé en définissant des modèles, des structures de données et des interfaces. Ainsi, la principale contribution de cette architecture réside dans sa flexibilité, sa capacité d'évolutivité et son adaptabilité à différents environnements, infrastructures et exigences.

2) Une heuristique de fusion avec une nouvelle fonction de valeur de remplacement en termes de gestion de cache de MASCARA [51]. En particulier, l'heuristique peut décider quand fusionner des régions de données en se basant sur la récence de l'utilisation (localité temporelle) et le pourcentage de contribution à la réponse (localité spatiale) qui sont présentés à travers une nouvelle fonction de

remplacement. Elle parvient à un bon équilibre entre différents aspects, tels que le temps de réponse, le taux de réussite et l'utilisation de l'espace de cache par rapport aux approches conventionnelles. 3) Traitement multi-vues pour gérer les requêtes de sélection-projection-jointure dans MASCARA. En particulier, cette procédure décompose une requête de jointure (inner join) originale en sous-requêtes (sélection-projection) qui appartiennent à différentes relations ou vues jointes. 4) Un modèle coopératif, appelé MASCARA-FPGA (comme le montre la Figure 3.3), où le traitement des requêtes est accéléré en ce qui concerne la réécriture des requêtes et une partie de l'exécution des requêtes [50].

Comme on peut le voir, la réécriture des requêtes de MASCARA, appelée Query Trimming, est maintenant déchargée et accélérée sur FPGA dans l'exécution en pipeline de bas en haut. En détail, les deux sous-étapes, Semantic Matching et Semantic Extracting, sont converties en accélérateurs correspondants. Par exemple, la catégorie Semantic Matching peut comprendre des accélérateurs (également appelés noyaux) pour deux fonctions principales : Attribute Matching et Predicate Matching. Il est important de noter qu'un noyau peut être divisé en moteurs accélérés plus petits pour augmenter le niveau de parallélisme des tâches sur FPGA. En particulier, le noyau Predicate Matching peut être divisé en trois fonctions distinctes en tant que moteurs : Intersection ($Q_{DNF} \wedge S_{DNF}$), Implication ($Q_{DNF} \rightarrow S_{DNF}$) et Difference ($Q_{DNF} - S_{DNF}$). Les autres étapes de MASCARA (c'est-à-dire Query Broking, Result Refining) n'entraînent pas de surcharge de calcul, elles n'ont donc pas besoin d'être accélérées sur FPGA. En plus de la réécriture des requêtes, MASCARA-FPGA exécute également en parallèle une liste de requêtes de sondage générées grâce aux opérateurs de base de données (DB) sur FPGA, tels que le filtre, le projecteur et le tri-merge-join. En complément des accélérateurs, nous organisons et gérons également SC dans la mémoire externe (c'est-à-dire la mémoire vive dynamique DRAM) du FPGA, car elle offre un espace suffisant pour les applications de big data (par exemple, plus de 64 Go) et fournit une connexion haut débit aux noyaux. De plus, MASCARA-FPGA comprend un adaptateur FPGA qui peut encapsuler ou envelopper les fonctionnalités natives dans des interfaces de haut niveau. Plus précisément, l'adaptateur FPGA comble le fossé entre l'application de haut niveau de MASCARA et les accélérateurs de bas niveau sur FPGA en utilisant l'interface Java Native Interface (JNI). Enfin, un petit composant appelé Query Process Controller (QPC) est responsable de la gestion des signaux, des statuts et des résultats renvoyés dans le flux de travail de Query Trimming et de l'exécution de la requête de sondage sur FPGA.

3.6.3 Détection d'anomalies en temps réel sur flux de données à l'aide de GPU

Dans de nombreuses applications de surveillance, telles que les réseaux de capteurs pour la collecte et l'analyse de données biologiques, géologiques et environnementales, ainsi que la surveillance de la consommation d'énergie, les données se présentent sous forme de flux de données. Un flux de données est une séquence de points de données avec des horodatages qui présente de nombreuses caractéristiques spéciales, notamment une taille infinie, une transience, une incertitude, une distribution dynamique et une multidimensionalité. Pour les applications impliquant plusieurs flux de données liés, des caractéristiques supplémentaires existent, telles qu'une arrivée de données asynchrone, des re-

lations dynamiques entre les flux et une hétérogénéité de schéma. Bien que les flux de données diffèrent des données non-streams (régulières) à bien des égards, ils ne sont pas exempts de valeurs aberrantes. Une valeur aberrante est un point de données qui diffère significativement des autres points de données du même ensemble de données. Pratiquement, les valeurs aberrantes sont inévitables dans tout processus d'acquisition de données, car elles peuvent être introduites pour de nombreuses raisons, telles que des activités malveillantes ou des erreurs d'instrumentation. Ainsi, la détection des valeurs aberrantes est une partie importante du processus d'analyse des données. Au lieu de considérer la détection des valeurs aberrantes comme une boîte noire, il est nécessaire de fournir des explications sur les valeurs aberrantes découvertes pour que la détection des valeurs aberrantes soit bénéfique pour l'utilisateur.

Il existe trois principaux types d'explications des valeurs aberrantes [83] : la causalité des valeurs aberrantes, les attributs aberrants et le classement des valeurs aberrantes. La causalité des valeurs aberrantes explique ce qui provoque qu'un objet/événement de données soit une valeur aberrante. Cela peut être déduit en examinant les interactions causales entre les valeurs aberrantes, car une valeur aberrante peut également provoquer l'apparition d'une autre valeur aberrante, ainsi que les interactions causales entre les éléments internes et les valeurs aberrantes, car un élément interne peut également provoquer l'apparition d'une autre valeur aberrante. Les attributs aberrants font référence aux attributs responsables de l'anormalité des valeurs aberrantes. L'explication dans ce cas peut être donnée sous forme d'un ensemble, dont chaque membre se compose d'un attribut et d'un score indiquant la contribution de l'attribut à l'anormalité de la valeur aberrante détectée. Le classement des valeurs aberrantes révèle le niveau d'importance de chaque valeur aberrante dans un ensemble de valeurs aberrantes détectées. La plupart des algorithmes d'explication des valeurs aberrantes existants fournissent un seul type d'explication de manière isolée, telles que les interactions causales entre les valeurs aberrantes dans [64], [107], les attributs aberrants dans [27], [45], et le classement des valeurs aberrantes dans [105], [93]. Aucun des algorithmes ne fournit les trois types d'explications des valeurs aberrantes [83]. De plus, le nombre d'algorithmes existants qui expliquent les valeurs aberrantes dans les flux de données est très limité et ils se concentrent principalement sur les attributs aberrants uniquement [105], [111], [97], [82].

L'explication des valeurs aberrantes pour les flux de données est plus difficile que pour les données régulières en raison des caractéristiques des flux de données et des défis posés par le Big Data, qui nécessitent des algorithmes parallèles en ligne et évolutifs pour expliquer les valeurs aberrantes en temps réel, aussi proches que possible de l'arrivée des données, afin de pouvoir prendre des mesures en temps opportun pour les applications. Cependant, les travaux existants sur les explications des valeurs aberrantes ne traitent que certaines caractéristiques des flux de données de manière isolée. De plus, bien que les matériels parallèles courants tels que les GPU offrant des performances informatiques élevées soient devenus populaires et abordables, aucun travail existant sur les explications des valeurs aberrantes dans les flux de données n'est conçu pour les GPU. Les défis de recherche des GPU pour le Big Data incluent le débit faible de l'interface hôte vers le GPU, l'espace mémoire réduit des GPU, la bande passante mémoire globale faible par rapport au nombre de threads et l'équilibrage de charge. Ce dont nous avons besoin pour faire avancer l'état de l'art dans ce

domaine, c'est un algorithme d'explication des valeurs aberrantes intégratif qui peut prendre en compte toutes les caractéristiques des flux de données, tirer parti des matériels parallèles courants évolutifs et, en même temps, fournir les trois types d'explications pour les valeurs aberrantes découvertes. L'algorithme doit relever les défis de recherche à la fois des flux de données et des GPU pour le Big Data.

3.7 Conclusion

La cybersécurité peut être considérée comme un cas concret de Big Data. Par conséquent, diverses directions peuvent être envisagées du point de vue de la scalabilité dans la gestion des données. Dans cet article, nous avons d'abord présenté certains systèmes liés à la surveillance de la sécurité, à savoir SEKOIA, Punch et Warp10. Nous avons ensuite discuté des orientations de recherche du point de vue de l'analyse de graphes (distribuée). Le problème de la confidentialité et du contrôle d'accès a ensuite été abordé. La dernière dimension prise en compte dans cet article est l'accélération matérielle avec la possibilité d'intégrer des GPU, des FPGA ou des NVM.

Bien évidemment, d'autres orientations méritent d'être étudiées. Pour en citer quelques-unes, on peut mentionner : la représentation des connaissances, l'apprentissage automatique, l'intelligence artificielle, la détection des anomalies et des menaces, l'analyse des données spatiales-temporelles.

Chapitre 4

Supervision et Intelligence artificielle

La multiplication des appareils électroniques utilisés au quotidien (e.g. avec l'IoT), mais aussi les projets d'envergure comme les villes connectées ou la rationalisation du travail via les outils numériques entraînent un élargissement considérable de la surface d'attaque potentielle pour les entreprises et les institutions [47].

Face à cette vulnérabilisation potentielle croissante de tous les domaines de l'activité économique et de nos vies numériques, mais aussi du fait de l'automatisation des attaques via –par exemple– les dernières avancées du machine learning (les bots et leur contrôle furtif), la réponse à la massification des attaques ne peut s'appuyer uniquement sur la multiplication nécessairement limitée des interventions humaines. Un support numérique et automatisé à la sécurisation est donc une nécessité absolue (problème I).

Un tel besoin d'automatisation de la prévention et de la réponse aux attaques peut aujourd'hui s'appuyer sur les avancées majeures du machine learning (plus généralement appelé Intelligence Artificielle ou IA en abrégé). En effet, les progrès réalisés en reconnaissance d'images, en traitement de données temporelles et en prédiction de séries temporelles ont déjà conduit à des propositions d'application à la supervision de sécurité.

Le principal avantage d'une approche basée sur le machine learning est tout d'abord une moindre prépondérance de la nécessaire labellisation manuelle des attributs à traiter automatiquement. En effet, des avancées en machine learning permet d'assouplir cet aspect par l'extraction partiellement automatique de ces attributs comme pré-requis à l'approche.

Cette automatisation de l'identification et du raffinement des attributs se heurte néanmoins à la qualité des données qui sont fournies à ces méthodes d'apprentissage. Cependant, la massification des attaques et la précision parfois limitée des capteurs posent de nouveaux défis. Les approches d'apprentissage automatique sont historiquement basées sur des techniques supervisées, où l'apprentissage est effectué sur des ensembles de données les plus propres possibles, c'est-à-dire contenant les labels les plus précis possibles pour chaque événement d'intérêt (problème II). Cependant, force est de constater que la masse de données collectées ne permet pas de qualifier au mieux l'événement

en cours ; des méthodes pour surmonter ces problèmes doivent être introduites, afin que l'apprentissage automatique puisse également déployer ses performances dans le domaine de la supervision de sécurité.

Cette même modalité de massification des données et par conséquent le plus grand nombre d'attributs potentiellement en jeu dans la détection précise d'une attaque impose une nécessaire transparence. Il est constitutif de la nature des réseaux de neurones, par exemple, qu'ils ne soient pas explicables nativement, car étant constitués par essence de tableaux de flottants, et ceci contrairement aux algorithmes qui sont explicables du fait de leur structuration autour de changements décisionnels clairs. (problème V).

D'autres approches décident de dépasser ce cadre et considèrent simplement l'impossibilité de retracer certains facteurs comme l'identité d'un participant à un dépôt de code public (problème III), ou encore plus radicalement en considérant un système surveillé comme une boîte noire dont le fonctionnement doit être observé pour mieux la comprendre (problème IV). Cette dernière approche remonte à la conception et au contrôle des systèmes cybernétiques, mais connaît un regain d'intérêt du fait de la possibilité d'utiliser l'apprentissage automatique pour des performances jusqu'alors inatteignables.

4.1 Propositions faites à SUPSEC

Un système de recommandation pour aider les analystes Face au déluge de données, Kraken [20] a été présenté comme un outil pour aider les analystes à naviguer dans des données complexes (journaux des centres d'opérations de sécurité). Cela peut être fait pratiquement en améliorant les outils visuels utilisés dans les enquêtes des analystes. Le recommandeur proposé par Kraken prend en entrée le contexte de sécurité d'une entreprise, et produit un plan de prévention pour cette entreprise. Il utilise les décisions passées d'un analyste de triage, pour décider si une alerte mérite une analyse. Quant aux recommandateurs dans la plupart des situations, il y a un problème de démarrage à froid (coldstart). Kraken est construit comme un système de recommandation basé sur la connaissance, qui prend la requête de l'utilisateur et la connaissance du domaine comme entrées, ce qui pourrait aider à prévenir ce problème de démarrage à froid. En pratique, lorsqu'un drapeau sur un problème possible est levé, une liste de types de données intéressants est rassemblée et envoyée au recommandeur, pour obtenir la liste des actions les plus intéressantes à prendre à afficher à l'analyste. Il est clair que les systèmes de recommandation modernes fonctionnent désormais à l'aide de modèles d'apprentissage profonds (classificateurs en particulier) [46].

Automatiser la gestion des vulnérabilités (problème I) Face à la complexité croissante des systèmes connectés, et dans le contexte de l'informatique autonome où l'on souhaite déléguer les fonctionnalités de gestion aux réseaux eux-mêmes, ont été évoqués les moyens d'automatisation [47]. Une première étape nécessaire consiste pour les agents du réseau à évaluer leur propre exposition à des problèmes de sécurité, via le langage OVAL ; ces descriptions sont transformées en règles de politique interprétables par le système de configuration de CFengine. Une formulation SAT est mise en place afin de sélectionner les corrections, et les résultats sont présentés sur l'ensemble de données de vulnérabilité

Cisco IOS. La résolution SAT dans ce cas est légère car prend environ deux secondes par exécution. Pour les vulnérabilités réparties sur plusieurs appareils, l'extension DOVAL d'OVAL est exploitée et l'automatisation est basée sur des stratégies collaboratives.

En ce qui concerne la dimension temporelle de l'apparition des vulnérabilités, et sur les mobiles par exemple, une sélection de tests peut être effectuée sur la base d'une analyse d'utilité, avec des fonctions d'utilité de test statistique, des fonctions de temps delta ou des fonctions de sélection de test.

Un autre aspect de l'automatisation, plus lié au côté opérationnel, est la configuration automatisée des ressources virtualisées. Les aspects saillants sont la nécessité de repenser le cycle de vie de la gestion de la sécurité, dans le cadre d'une stratégie de sécurité définie par logiciel basée sur la reconstruction de ressources virtuelles à la volée, ou en reconstruisant des images machine unikernel lorsqu'un correctif l'exige.

Les méthodes d'apprentissage d'ensemble (ensemble learning) sont recommandées pour la supervision de sécurité afin de les appliquer aux systèmes IoT à grande échelle et de détecter les attaques avancées telles que les APTs par exemple.

Un défi identifié pour l'automatisation de la supervision est de pouvoir combler le fossé entre l'IA courante et les techniques de vérification formelle.

Modélisation des systèmes automatisés (problèmes I et IV) Les avantages pour l'automatisation de la vérification, de la prédiction ou de la détection d'anomalies ont été exposés. Pour ce faire, il a été proposé d'utiliser des données d'exécution pour modéliser le système sans le connaître, c'est-à-dire en l'abordant comme une boîte noire [26]. À cette fin, des ensembles d'événements du système sont capturés et pris en compte. Les autres formalismes pour la modélisation globale sont : les automates à états finis, les réseaux de Petri ; ils peuvent être étendus avec une notion temporelle. Un autre corpus de travaux intéressant concerne les algorithmes génétiques qui peuvent apprendre des données du système (par exemple avec GenProgTA). Les travaux discutés portent sur les automates temps réel déterministes, et un algorithme (nommé TAG) pour créer séquentiellement l'automate est proposé, ainsi qu'une implémentation avec un package Python. Les expériences montrent un bon compromis entre rappel et précision de la tâche étudiée.

Automatisation avec un pipeline piloté par l'IA pour la détection et la classification d'intrusions (problème I) La fréquence et le volume élevés de données à surveiller ne peuvent être traités que de manière automatisée. De nombreuses méthodes d'IA nécessitent de grandes quantités de données labellisées pour leur formation, qui ne sont pas disponibles sur le terrain, ou avec une telle quantité de bruit que ces méthodes sont entravées.

Cela nécessite la génération automatique de features, un apprentissage faiblement-supervisé de la détection ou de la classification des incidents de sécurité, un réglage automatique de l'architecture du modèle pour la détection et la classification des intrusions, et l'évaluation automatique de la précision et de la stabilité.

Sur l'exemple de la détection de botnets, des approches basées sur des graphes peuvent être adoptées. Suivant un tel modèle, les nœuds sont des

adresses IP sources, des adresses IP destinations et des nœuds basés "paquets". Les réseaux de neurones à base de graphes (graph neural networks) [113] peuvent être usités sur des voisinages limités dans ce graphe pour effectuer une classification précise des événements. Le besoin de labels est ainsi supprimé (problème de résolution II), par la génération d'une représentation caractéristique de chaque paquet et des adresses IP observées sur les nœuds. L'automatisation s'appuie sur ce contexte dans la détermination du nombre de voisins nécessaires pour coder les caractéristiques de chaque paquet vu.

En ce qui concerne la possibilité d'un système efficace de détection d'intrusion basé sur le flux de trafic réseau, la principale question de recherche est liée à la possibilité de concevoir une évaluation automatique de la précision et de la stabilité. AdvCat est proposé pour attaquer le problème en utilisant des logs, ce qui s'avère être un problème d'optimisation combinatoire, et qui se traite via de la maximisation sous-modulaire. Les expériences réalisées sur deux ensembles de données de référence (HDFS et IPS) concluent que l'état de l'art de la détection d'anomalies basée sur l'apprentissage profond est extrêmement vulnérable à de légères perturbations sur les logs des événements de sécurité ; il s'agit d'un appel clair à d'autres recherches sur le sujet.

Un autre travail basé sur la modélisation sous forme de graphes a été proposé par Pierre-François Gimenez, visant la détection d'intrusion et la détection d'anomalies. Les sondes réseau capturent des données qui forment à leur tour un graphe riche avec des données hétérogènes (par exemple, adresses IP, entrées DNS, ports de destinations, protocoles). La technique mise à profit est l'utilisation d'auto-encodeurs neuronaux : l'erreur de reconstruction de cet auto-encodeur est utilisée comme score d'anomalie. Les performances sont testées sur le jeu de données DAPT2020 avec des attaques APT. Le rappel est bon mais les faux positifs sont élevés (22%). Une amélioration peut provenir du fait de ne pas traiter les arêtes du graphe de manière indépendante : des mécanismes d'attention pourraient aider à exploiter plus finement les voisinages de graphes. Les explications peuvent également aider à comprendre les faux positifs, mais peu de travaux existent dans l'état de l'art en matière d'apprentissage non supervisé.

Surveillance proactive du réseau L'intérêt [74] de tirer parti de l'apprentissage profond pour l'appliquer à la modélisation de séries temporelles d'événements de sécurité a été rappelé.

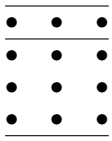
Les réseaux de neurones récurrents ont une notion du temps dans leur conception, mais ils ne sont pas la seule alternative pour ces séries temporelles, car les perceptrons multicouches classiques, ou des LSTM ou GRU plus récents sont également à considérer. La qualité de la prédiction varie selon les protocoles cibles de la détection (par exemple http, ssh, sip). Un déploiement a vu le jour dans une pile logicielle Kafka et Kibana, pour la prédiction de l'attaque sur certains ports, en temps réel, au cours de la guerre russo-ukrainienne ; avec pour résultat des alertes précises déclenchées.

Aucune solution universelle pour tout protocole n'a été trouvée, et les travaux futurs pourront s'intéresser à la construction d'une méthodologie modèle-hybride pour améliorer ces performances de prédiction.

Considérer l'inférence causale assistée par l'humain Il a été préconisé [112] l'intégration des connaissances humaines dans les tâches de détection, au lieu de s'appuyer entièrement sur l'apprentissage automatique. En effet, avec trois piliers, association/corrélation, intervention et contrefactuels (et si un expert agissait différemment ?), les experts peuvent considérablement réduire la masse des données à traiter par la tâche d'apprentissage (problème II), découvrir des relations causales dans ces données, mais également fournir une explicabilité grâce à la visualisation. Des améliorations de performances brutes par rapport à l'utilisation de modèles sans inférence causale humaine ont également été montrées, principalement dues à la diminution de la quantité de données à traiter.

Détection d'attaques dans les dépôts de code public (problème III) SIVA (SIlicom Versatile Artificial intelligence) a été proposé par SILICOM pour détecter les comportements inhabituels dans les commits de code, afin d'éviter les attaques dans la supply chain [95]. En particulier, le but est de fournir un outil pour authentifier les commits d'un développeur dans ces supply chain continues. Les techniques d'IA en jeu concernent l'apprentissage des signatures des développeurs (habitudes syntaxiques, lexicales et comportementales GIT) et la détection de comportements ou de codes inhabituels sur GIT, les injections par les développeurs (potentiellement symptomatique d'attaques). L'apprentissage s'effectue à partir du graphe acyclique composé d'agents causaux participant au problème. Une bibliothèque d'attention utilisée conjointement (Attention lib) aide à SIVA pour faire émerger des attentions simples ou complexes, surveiller des caractéristiques dans l'environnement observable, et retourner le résultat sous forme de présence de caractéristiques dans cet environnement. Des explications pour les événements détectés peuvent être fournies par un tel outil.

Techniques classiques de classification d'images pour la détection d'anomalies L'automatisation de la détection (problème I) pourrait également s'appuyer sur des techniques établies dans le domaine de la reconnaissance/classification d'images. En effet, si des images peuvent être créées à partir de données de trafic réseau brutes (par exemple, des requêtes DNS), alors une classification avec un réseau de neurones profond convolutif classique peut être utilisée [84]. Le labelling des images est effectué à partir d'une classification "bénigne" contre "malveillante", en utilisant le gain de fonctionnalité XGBoost ou la corrélation de Pearson. S'en suit un entraînement supervisé classique. Les performances sur le jeu de données CIRA-CIC-DoHBrw-2020 sont très intéressantes.



Chapitre 5

Détection d'APT

Les APT (pour Advanced Persistent Threat) sont des attaques qui sont perpétuées sur une longue durée (jusqu'à plusieurs mois). Du fait de cet étalement dans le temps des actions de l'attaquant, il est difficile de détecter de telles attaques. Aujourd'hui, les travaux sur ce type d'attaques en sont encore à leur balbutiement. Le semestre SUPSEC a mis en lumière des travaux récents dans ce domaine.

5.1 Dépendances causales

Les premiers travaux présentés (Thèse de Charles Xosanavongsa [108]) décrivent la façon dont peuvent être calculés les liens de causalité entre événements hétérogènes. La méthode formelle définie dans ces travaux montre qu'en se basant sur les définitions de causalité de Lamport et d'Ausbourg, on peut calculer la causalité entre les événements de logs générés dans un système distribué. Cette relation de causalité permet de construire des graphes de causalité entre événements de nature différente (événements systèmes, événements réseau, événements applicatifs, alertes d'IDS). Le principal intérêt de cette approche est que l'on peut calculer pour un événement anormal donné (un indice de compromission) le graphe des événements qui lui sont directement causalement liés. Ainsi il est possible d'expliquer une attaque en extrayant le sous-graphe associé à un indice de compromission.

La Figure 5.1 montre le résultat du calcul de causalité entre des événements de logs. Quelle que soit la distance dans le temps, l'explication d'une attaque est incluse dans le sous graphe de dépendance de l'indice de compromission. Cette méthode est donc tout à fait adaptée pour repérer des patterns d'attaques incluses dans les APT.

Actuellement, cette approche a été complètement définie formellement, mais n'a été implémentée que de manière approximative (dans le sens où le calcul effectué est une approximation du comportement réel du système). Des travaux sont en cours pour réaliser des calculs exacts de causalité, permettant ainsi théoriquement de construire des logs qui renfermeraient les dépendances causales entre événements hétérogènes. Une telle approche accélèrerait considérablement la capacité à expliquer un événement de compromission en capturant tous les événements qui ont mené à cet IOC.

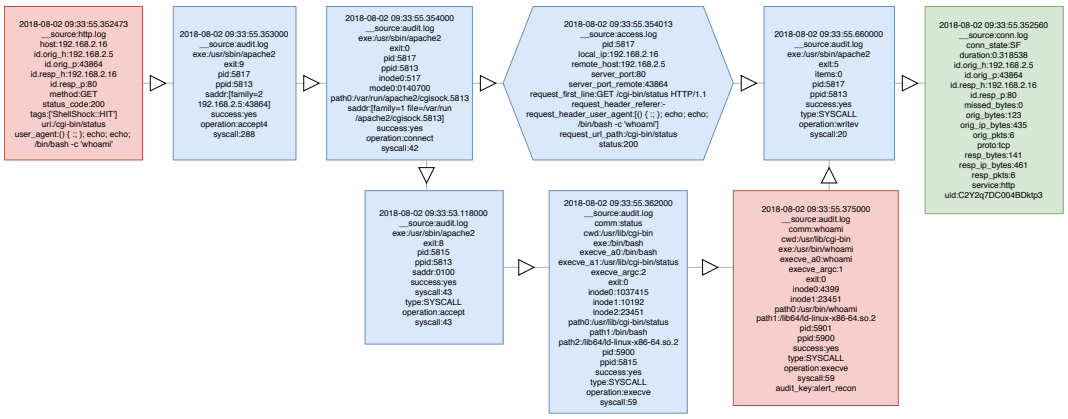


FIGURE 5.1 – Dépendances causales entre événements de logs

5.2 Utilisation de la Provenance

Une variante du calcul de causalité s'appelle la *Provenance* [12]. Cette méthode repose sur les calculs de causalité entre conteneurs d'information dans le système, et repose sur la création d'un graphe de provenance. Contrairement à la méthode décrite Section 5.1, le graphe n'inclut pas la notion d'ordre, mais uniquement les relations de causalité entre objets du système. Le résultat obtenu est décrit dans la Figure 5.2. Cette figure représente le graphe de provenance entre objets du système pour l'exécution d'un binaire malicieux.

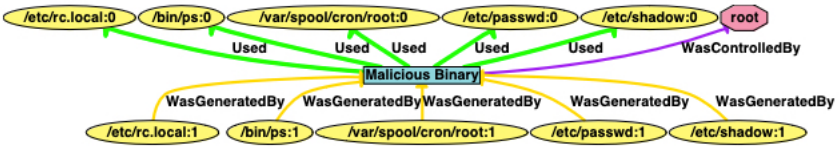


FIGURE 5.2 – Graphe de provenance

De nombreux travaux utilisant la provenance ont vu le jour. La Figure 5.3 établit un état de l'art des approches ayant été développées autour de la provenance.

Les travaux présentés à SUPSEC sur ce type d'approche sont les travaux sur Holmes [70]. Ils se placent dans la catégorie "sous-graphes", "détection par règles" de la Figure 5.3.

L'approche de Holmes consiste à analyser le graphe de provenance pour extraire les sous-graphes de l'attaquant, et à faire correspondre ce sous-graphe à des techniques d'attaques. Les sous-graphes étant identifiés, ils servent de bases de règles pour détecter le scénario d'attaque dans un graphe de provenance observé à l'exécution. L'architecture de Holmes est décrite dans la figure 5.4.

La deuxième approche qui a été présentée durant le workshop SUPSEC s'appelle ANUBIS. Comme pour Holmes, ANUBIS se base sur les graphes de provenance, mais cette fois pour générer des traces d'événements système qui représentent les comportements de l'attaquant. Ensuite, un réseau de neurone

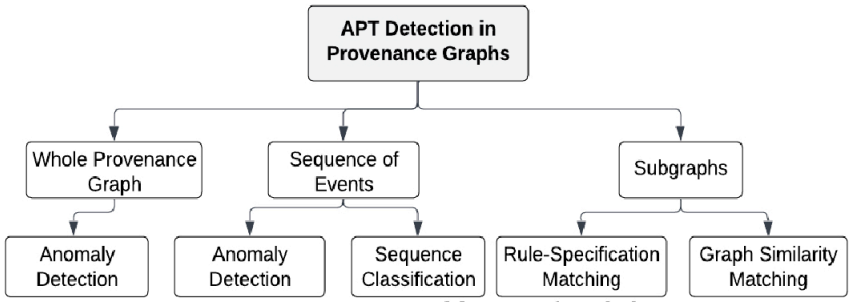


FIGURE 5.3 – Travaux de détection autour de la provenance

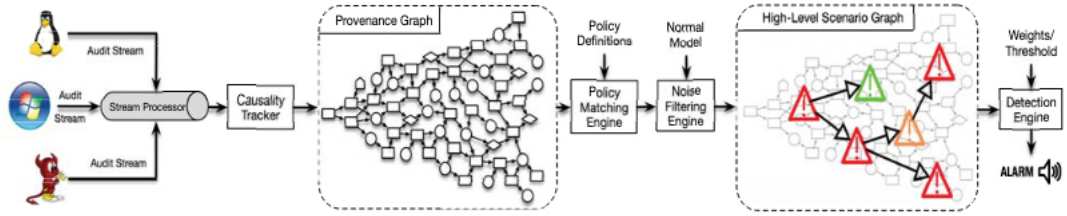


FIGURE 5.4 – Architecture de Holmes

LSTM est entraîné pour pouvoir, à l'exécution, détecter des séquences anormales dans le système.

Les travaux sur les graphes de provenance peuvent tout à fait être appliqués dans le cas des graphes de causalité, ouvrant des perspectives de travail intéressantes.

5.3 Approche dirigée par la sémantique pour tracer des campagnes d'attaques

Les attaques APT peuvent être décrites comme une succession de TTP telles qu'elles sont décrites par Mitre Att&ck. Malheureusement cette description est très informelle (langage textuel). Les travaux présentés ici [13] envisagent de formaliser la description des TTPs pour permettre de détecter directement dans un jeu de données les avancées de l'attaquant. Ces travaux ont été appliqués à un jeu de données PWNJUTSU pour retracer des campagnes d'attaques.

Conclusion et discussion

Les travaux présentés dans ce document illustrent les perspectives de recherche qui peuvent être appliquées dans le cadre de la supervision de sécurité. Il faut noter que les domaines de recherche ne sont pas disjoints. En effet, l'IA est présente dans la détection des APT (cf. Section 5.2). D'autre part, les recherches menées dans le domaine des APT reposent sur l'utilisation de jeux données de grande envergure, nécessitant des mécanismes sous-jacent de big data.

Bibliographie

[1] A survey of phase change memory systems. *Journal of Computer Science and Technology*, 30(1) :121–144, 2015.

[2] Juba Agoun and Mohand-Saïd Hacid. Data sharing in presence of access control policies. In *CoopIS*, 2019.

[3] Juba Agoun and Mohand-Saïd Hacid. Data publishing : Availability of data under security policies. In *International Symposium on Methodologies for Intelligent Systems*, pages 277–286. Springer, 2020.

[4] Juba Agoun and Mohand-Saïd Hacid. Access control based on entity matching for secure data sharing. *Service Oriented Computing and Applications*, 16(1), 2022.

[5] Amazon. Amazon ec2.

[6] Amazon. Amazon elasticache.

[7] James P. Anderson. Computer security threat monitoring and surveillance. In *21st NISCC conference*, 1980.

[8] Ashutosh Sharma Arka Sharma, Amit Kumar. Open channel ssd. *FreeBSD Journal*, 2021.

[9] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D Joseph, Randy H Katz, Andy Konwinski, Gunho Lee, David A Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia. A view of cloud computing. *Communications of the ACM*, 53(4), 2010.

[10] Michael Armbrust, Reynold S Xin, Cheng Lian, Yin Huai, Davies Liu, Joseph K Bradley, Xiangrui Meng, Tomer Kaftan, Michael J Franklin, Ali Ghodsi, and Matei Zaharia. Spark SQL : Relational Data Processing in Spark. In *Proceedings of the SIGMOD International Conference on Management of Data*, Melbourne, Victoria, Australia, 2015.

[11] Antonio Barbalace and Jaeyoung Do. Computational storage : Where are we today ? January 2021. Conference on Innovative Data Systems Research 2020, CIDR 2020 ; Conference date : 11-01-2021 Through 15-01-2021.

[12] Adam Bates, Dave (Jing) Tian, Kevin R. B. Butler, and Thomas Moyer. Trustworthy Whole-System Provenance for the Linux Kernel. pages 319–334, 2015.

[13] Aimad Berady, Mathieu Jaume, Valérie Viet Triem Tong, and Gilles Guette. PWNJUTSU : A Dataset and a Semantics-Driven Approach to Retrace Attack Campaigns. *IEEE Transactions on Network and Service*

Management, 19(4) :5252–5264, December 2022. Conference Name : IEEE Transactions on Network and Service Management.

[14] Elisa Bertino, Shawn Merrill, Alina Nesen, and Christine Utz. Redefining data transparency : A multidimensional approach. *Computer*, 52(1), 2019.

[15] Kevin S Beyer, Vuk Ercegovac, Rainer Gemulla, Andrey Balmin, Mohamed Y Eltabakh, Carl-Christian Kanne, Fatma Özcan, and Eugene J Shekita. Jaql : A Scripting Language for Large Scale Semistructured Data Analysis. *PVLDB*, 4(12), 2011.

[16] B.Morin, L.Mé, H.Debar, and M.Ducassé. M2d2 : A formal data model for ids alert correlation. In *RAID 2002*.

[17] Djillali Boukhelef, Jalil Boukhobza, Kamel Boukhalfa, Hamza Ouarnoughi, and Laurent Lemarchand. Optimizing the cost of dbaas object placement in hybrid storage systems. *Future Generation Computer Systems*, 93 :176–187, 2019.

[18] Jalil Boukhobza and Pierre Olivier. *Flash Memory Integration : Performance and Energy Issues, 1st Edition*. ISTE Press - Elsevier, 2017.

[19] Jalil Boukhobza, Stéphane Rubini, Renhai Chen, and Zili Shao. Emerging nvm : A survey on architectural integration and research challenges. *ACM Trans. Des. Autom. Electron. Syst.*, 23(2), nov 2017.

[20] Romain Brisse, Simon Boche, Frédéric Majorczyk, and Jean-François Lalande. Kraken : A knowledge-based recommender system for analysts, to kick exploration up a notch. In *14th International Conference on Security for Information Technology and Communications*, 2021.

[21] Paris Carbone, Asterios Katsifodimos, Stephan Ewen, Volker Markl, Seif Haridi, and Kostas Tzoumas. Apache Flink™ : Stream and Batch Processing in a Single Engine. *IEEE Data Engineering Bulletin*, 38(4), 2015.

[22] Ronnie Chaiken, Bob Jenkins, Per-Åke Larson, Bill Ramsey, Darren Shakib, Simon Weaver, and Jingren Zhou. SCOPE : easy and efficient parallel processing of massive data sets. *PVLDB*, 1(2), 2008.

[23] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection : A survey. *ACM computing surveys (CSUR)*, pages 1–58, 2009.

[24] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C Hsieh, Deborah A Wallach, Michael Burrows, Tushar Chandra, Andrew Fikes, and Robert E Gruber. Bigtable : A Distributed Storage System for Structured Data. *ACM Transactions on Computer Systems (TOCS)*, 26(2), 2008.

[25] Derek Chiou. The microsoft catapult project. In *2017 IEEE International Symposium on Workload Characterization (IISWC)*, pages 124–124, 2017.

[26] Lénaïg Cornanguer, Christine Largouët, Laurence Rozé, and Alexandre Termier. Tag : Learning timed automata from logs. In *36th AAAI Conference on Artificial Intelligence (AAAI'22)*, 2022.

[27] Xuan-Hong Dang, Ira Assent, Raymond T. Ng, Arthur Zimek, and Erich Schubert. Discriminative features for identifying and interpreting outliers. In *ICDE*, 2014.

[28] Shaul Dar, Michael J. Franklin, Björn ór Jónsson, Divesh Srivastava, and Michael Tan. Semantic data caching and replacement. In *Proceedings of the 22th International Conference on Very Large Data Bases*, VLDB

'96, page 330–341, San Francisco, CA, USA, 1996. Morgan Kaufmann Publishers Inc.

[29] Jeffrey Dean and Sanjay Ghemawat. MapReduce : simplified data processing on large clusters. *Communications of the ACM*, 51(1), 2008.

[30] H. Debar and A. Wespi. Aggregation and correlation of intrusion-detection alerts. In *Workshop on Recent Advances in Intrusion Detection*, pages 85–103, 2001.

[31] Dorothy E. Denning. An intrusion detection model. *IEEE Transactions on Software Engineering*, 1987.

[32] Prasad M. Deshpande, Karthikeyan Ramasamy, Amit Shukla, and Jeffrey F. Naughton. Caching multidimensional queries using chunks. *SIGMOD Rec.*, 27(2) :259–270, jun 1998.

[33] Laurent d'Orazio, Fabrice Jouanot, Cyril Labbé, and Claudia Roncancio. Building adaptable cache services. In *MGC@Middleware*, 2005.

[34] Laurent d'Orazio, Claudia Roncancio, Cyril Labbé, and Fabrice Jouanot. Semantic caching in large scale querying systems. *Revista Colombiana de Computación*, 9, 06 2008.

[35] Laurent d'Orazio and Mamadou Kaba Traoré. Semantic caching for pervasive grids. In *Proceedings of the 2009 International Database Engineering and Applications Symposium*, IDEAS '09, page 227–233, 2009.

[36] Hazem Elmeleegy, Mourad Ouzzani, Ahmed K. Elmagarmid, and Ahmad M. Abusalah. Preserving privacy and fairness in peer-to-peer data integration. In Ahmed K. Elmagarmid and Divyakant Agrawal, editors, *SIGMOD*, 2010.

[37] Jian Fang, Yvo T. B. Mulder, Jan Hidders, Jinho Lee, and H. Peter Hofstee. In-memory database acceleration on fpgas : a survey. *The VLDB Journal*, 29 :33–59, 2019.

[38] Elena Ferrari. *Access Control in Data Management Systems*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2010.

[39] Gary William Flake, Steve Lawrence, C Lee Giles, and Frans M Coetzee. Self-organization and identification of web communities. *Computer*, 35(3) :66–71, 2002.

[40] Santo Fortunato. Community detection in graphs. *Physics reports*, 486(3-5) :75–174, 2010.

[41] Yoshihisa Fujisaki. Review of emerging new solid-state non-volatile memories. *Japanese Journal of Applied Physics*, 52(4R) :040001, apr 2013.

[42] Parke Godfrey and Jarek Gryz. Answering queries by semantic caches. In *Database and Expert Systems Applications*, pages 485–498, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg.

[43] Sha Guo, Wei Sun, and M.A. Weiss. On satisfiability, equivalence, and implication problems involving conjunctive queries in database systems. *IEEE Transactions on Knowledge and Data Engineering*, 8(4) :604–616, 1996.

[44] Sha Guo, Wei Sun, and Mark A. Weiss. Solving satisfiability and implication problems in database systems. *ACM Trans. Database Syst.*, 21(2) :270–293, jun 1996.

[45] Nikhil Gupta, Dhivya Eswaran, Neil Shah, Leman Akoglu, and Christos Faloutsos. Beyond outlier detection : Lookout for pictorial explanation. In *ECML PKDD*, 2018.

[46] Udit Gupta, Samuel Hsia, Vikram Saraph, Xiaodong Wang, Brandon Reagen, Gu-Yeon Wei, Hsien-Hsin S. Lee, David Brooks, and Carole-Jean Wu. Deeprecsys : A system for optimizing end-to-end at-scale neural recommendation inference. In *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*, pages 982–995, 2020.

[47] Adrien Hemmer, Mohamed Abderrahim, Remi Badonnel, and Isabelle Chrisment. An ensemble learning-based architecture for security detection in iot infrastructures. In *2021 17th International Conference on Network and Service Management (CNSM)*, pages 180–186. IEEE, 2021.

[48] Gerrit Holtrup, William Lacube, Dimitri Percia David, Alain Mermoud, Gérôme Bovet, and Vincent Lenders. 5g system security analysis. *CoRR*, abs/2108.08700, 2021.

[49] Syed Rafiul Hussain, Mitziu Echeverria, Omar Chowdhury, Ninghui Li, and Elisa Bertino. Privacy attacks to the 4g and 5g cellular paging protocols using side channel information. In *NDSS*, 2019.

[50] Van Long Nguyen Huu, Laurent d'Orazio, Emmanuel Casseau, and Julien Lallet. MASCARA-FPGA cooperation model : Query trimming through accelerators. In *SSDBM*, 2021.

[51] Van Long Nguyen Huu, Laurent d'Orazio, Emmanuel Casseau, and Julien Lallet. Cache management in MASCARA-FPGA : from coalescing heuristic to replacement policy. In *DaMoN@SIGMOD*, 2022.

[52] Van Long Nguyen Huu, Julien Lallet, Emmanuel Casseau, and Laurent d'Orazio. MASCARA (modular semantic caching framework) towards FPGA acceleration for iot security monitoring. *Open Journal of Internet of Things*, 6(1), 2020.

[53] Joseph Izraelevitz, Jian Yang, Lu Zhang, Juno Kim, Xiao Liu, Amir Saman Memaripour, Yun Joon Soh, Zixuan Wang, Yi Xu, Subramanya R. Dulloor, Jishen Zhao, and Steven Swanson. Basic performance measurements of the intel optane DC persistent memory module. *CoRR*, abs/1903.05714, 2019.

[54] Ashesh Jain, Amir R Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-RNN : Deep learning on spatio-temporal graphs. In *Proceedings of CVPR*, pages 5308–5317, 2016.

[55] Björn ór Jónsson, María Arinbjarnar, Bjarnsteinn órsson, Michael J. Franklin, and Divesh Srivastava. Performance and overhead of semantic cache management. 6(3) :302–331, aug 2006.

[56] A.M. Keller and J. Basu. A predicate-based caching scheme for client-server database architectures. In *Proceedings of 3rd International Conference on Parallel and Distributed Information Systems*, pages 229–238, 1994.

[57] Anastasios Kementsietsidis and Marcelo Arenas. Data sharing through query translation in autonomous sources. In *VLDB*, 2004.

[58] C. Kiennert, Z. Ismail, H. Debar, and J. Leneutre. A survey on game-theoretic approaches for intrusion detection and response optimization. *ACM Computing Surveys (CSUR)*, pages 1–31, 2018.

[59] Brian Kulis, Sugato Basu, Inderjit Dhillon, and Raymond Mooney. Semi-supervised graph clustering : a kernel approach. *Machine learning*, 74(1) :1–22, 2009.

[60] Emre Kültürsay, Mahmut Kandemir, Anand Sivasubramaniam, and Onur Mutlu. Evaluating stt-ram as an energy-efficient main memory alternative. In *2013 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, pages 256–267, 2013.

[61] Arezki Laga, Jalil Boukhobza, Frank Singhoff, and Michel Koskas. Montres : Merge on-the-run external sorting algorithm for large data volumes on ssd based storage systems. *IEEE Transactions on Computers*, 66(10) :1689–1702, 2017.

[62] Avinash Lakshman and Prashant Malik. Cassandra : a decentralized structured storage system. *Operating Systems Review*, 44(2) :35–40, 2010.

[63] Dongwon Lee and Wesley W. Chu. Semantic caching via query matching for web sources. In *Proceedings of the Eighth International Conference on Information and Knowledge Management*, CIKM '99, page 77–85, New York, NY, USA, 1999. Association for Computing Machinery.

[64] Wei Liu, Yu Zheng, Sanjay Chawla, Jing Yuan, and Xing Xie. Discovering spatio-temporal causal interactions in traffic data streams. In Chid Apté, Joydeep Ghosh, and Padhraic Smyth, editors, *SIGKDD*, 2011.

[65] Siqi Ma, Elisa Bertino, Surya Nepal, Juanru Li, Diethelm Ostry, Robert H. Deng, and Sanjay Jha. Finding flaws from password authentication code in android apps. In Kazue Sako, Steve A. Schneider, and Peter Y. A. Ryan, editors, *ESORICS*, 2019.

[66] Siqi Ma, Juanru Li, Hyoungshick Kim, Elisa Bertino, Surya Nepal, Diethelm Ostry, and Cong Sun. Fine with "1234" ? an analysis of SMS one-time password randomness in android apps. In *ICSE*, 2021.

[67] Seiji Maekawa, Koki Noda, Yuya Sasaki, and Makoto Onizuka. Beyond real-world benchmark datasets : An empirical study of node classification with GNNs. In *NeurIPS*, 2022.

[68] Seiji Maekawa, Yuya Sasaki, George Fletcher, and Makoto Onizuka. GenCAT : Generating attributed graphs with controlled relationships between classes, attributes, and topology. *CoRR*, abs/2109.04639, 2021.

[69] J. McHugh. Testing intrusion detection systems : a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory. *ACM Transactions on Information and System Security (TISSEC)*, pages 262–294, 2000.

[70] Sadegh M. Milajerdi, Rigel Gjomemo, Birhanu Eshete, R. Sekar, and V.N. Venkatakrishnan. HOLMES : Real-Time APT Detection through Correlation of Suspicious Information Flows. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 1137–1152, San Francisco, CA, USA, May 2019. IEEE.

[71] Sparsh Mittal, Gaurav Verma, Brajesh Kaushik, and Farooq A. Khanday. A survey of sram-based in-memory computing techniques and applications. *Journal of Systems Architecture*, 119 :102276, 2021.

[72] Rimma V. Nehme, Hyo-Sang Lim, and Elisa Bertino. FENCE : continuous access control enforcement in dynamic data stream environments. In *ICDE*, 2010.

[73] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. On spectral clustering : Analysis and an algorithm. In *NIPS*, pages 849–856. MIT Press, 2001.

[74] Giang Nguyen, Stefan Dlugolinsky, Viet Tran, and Álvaro López García. Deep learning for proactive network monitoring and security protection. *IEEE Access*, 8 :19696–19716, 2020.

[75] Louis-Marie Nicolas, Luis Thomas, Yassine Hadjadj-Aoul, and Jalil Boukhobza. Slrl : A simple least remaining lifetime file evicition policy for hpc multi-tier storage systems. In *Proceedings of the Workshop on Challenges and Opportunities of Efficient and Performant Storage Systems*, CHEOPS '22, page 33–39, New York, NY, USA, 2022. Association for Computing Machinery.

[76] Yuya Ogawa, Seiji Maekawa, Yuya Sasaki, Yasuhiro Fujiwara, and Makoto Onizuka. Adaptive node embedding propagation for semi-supervised classification. In *ECML/PKDD*, volume 12976 of *Lecture Notes in Computer Science*, pages 417–433. Springer, 2021.

[77] Pierre Olivier, Jalil Boukhobza, Eric Senn, and Hamza Ouarnoughi. A methodology for estimating performance and power consumption of embedded flash file systems. *ACM Trans. Embed. Comput. Syst.*, 15(4), aug 2016.

[78] Christopher Olston, Benjamin Reed, Utkarsh Srivastava, Ravi Kumar, and Andrew Tomkins. Pig latin : a not-so-foreign language for data processing. In *SIGMOD*, Vancouver, BC, Canada, 2008.

[79] Makoto Onizuka, Toshimasa Fujimori, and Hiroaki Shiokawa. Graph partitioning for distributed graph processing. *Data Sci. Eng.*, 2(1) :94–105, 2017.

[80] Makoto Onizuka, Hiroyuki Kato, Soichiro Hidaka, Keisuke Nakano, and Zhenjiang Hu. Optimization for iterative queries on mapreduce. *Proc. VLDB Endow.*, 7(4) :241–252, 2013.

[81] M. Owaida, D. Sidler, K. Kara, and G. Alonso. Centaur : A framework for hybrid cpu-fpga databases. In *2017 IEEE 25th Annual International Symposium on Field-Programmable Custom Computing Machines*, pages 211–218, 2017.

[82] Egawati Panjei, Le Gruenwald, Eleazar Leal, and Christopher Nguyen. Micro-clusters-based outlier explanations for data streams. In *ANDEA@KDD*, 2021.

[83] Egawati Panjei, Le Gruenwald, Eleazar Leal, Christopher Nguyen, and Shejuti Silvia. A survey on outlier explanations. *VLDBJ*, 31(5), 2022.

[84] Amélie Raymond, Baptiste Brument, and Pierre Parrend. Viznn : Visual data augmentation with convolutional neural networks for cybersecurity investigation. In *Upper-Rhine Artificial Intelligence Symposium 2021 (UR-AI 2021), 27 octobre 2021, Kaiserslautern, Germany*, 2021.

[85] Redis. Redis enterprise cache services.

[86] Qun Ren, Margaret H. Dunham, and Vijay Kumar. Semantic caching and query processing. 15(1) :192–210, jan 2003.

[87] Daniel J. Rosenkrantz and Harry B. Hunt III. Processing conjunctive predicates and queries. In *Proceedings of the Sixth International Conference on Very Large Data Bases - Volume 6*, VLDB '80, page 64–72. VLDB Endowment, 1980.

[88] Norvald H. Ryeng, Jon Olav Hauglid, and Kjetil Nørvåg. Site-autonomous distributed semantic caching. In *Proceedings of the 2011 ACM Symposium on Applied Computing*, SAC '11, page 1015–1021, New York, NY, USA, 2011. Association for Computing Machinery.

[89] Bikas Saha, Hitesh Shah, Siddharth Seth, Gopal Vijayaraghavan, Arun C Murthy, and Carlo Curino. Apache Tez : A Unifying Framework for Modeling and Building Data Processing Applications. In *SIGMOD*, Melbourne, Victoria, Australia, 2015.

[90] Behzad Salami, Gorker Alp Malazgirt, Oriol Arcas-Abella, Arda Yurdakul, and Nehir Sonmez. Axledb : A novel programmable query processing platform on fpga. *Microprocessors and Microsystems*, 51 :142–164, 2017.

[91] Bilal Shebaro, Oyindamola Oluwatimi, Daniele Midi, and Elisa Bertino. Identidroid : Android can finally wear its anonymous suit. *Transactions on Data Privacy*, 7(1), 2014.

[92] Hiroaki Shiokawa, Yasuhiro Fujiwara, and Makoto Onizuka. Fast algorithm for modularity-based graph clustering. In *AAAI*. AAAI Press, 2013.

[93] Md Amran Siddiqui, Alan Fern, Thomas G. Dietterich, Ryan Wright, Alec Theriault, and David W. Archer. Feedback-guided anomaly discovery via online optimization. In *SIGKDD*, 2018.

[94] David Sidler, Zsolt Istvan, Muhsen Owaida, Kaan Kara, and Gustavo Alonso. Doppiodb : A hardware accelerated database. In *Proceedings of the 2017 ACM International Conference on Management of Data*, page 1659–1662, 2017.

[95] Silicom. Why Characterizing Authors of PorTIONs of code and commit logs ? https ://silicom.fr, 2022.

[96] Rohit Singh, Venkata Vamsikrishna Meduri, Ahmed K. Elmagarmid, Samuel Madden, Paolo Papotti, Jorge-Arnulfo Quiané-Ruiz, Armando Solar-Lezama, and Nan Tang. Synthesizing entity matching rules by examples. *PVLDB*, 11(2), 2017.

[97] Fei Song, Yanlei Diao, Jesse Read, Arnaud Stiegler, and Albert Bifet. EXAD : A system for explainable anomaly detection on big data traces. In *ICDM Workshops*, 2018.

[98] Michael Stonebraker, Anant Jhingran, Jeffrey Goh, and Spyros Potamianos. On rules, procedure, caching and views in data base systems. In *Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data*, SIGMOD '90, page 281–290, New York, NY, USA, 1990. Association for Computing Machinery.

[99] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams. The missing memristor found. *Nature*, 453(7191) :80–83, 2008.

[100] Bharat Sukhwani, Hong Min, Mathew Thoennes, Parijat Dube, Bernard Brezzo, Sameh Asaad, and Donna Eng Dillenberger. Database analytics : A reconfigurable-computing approach. *IEEE Micro*, 34(1) :19–29, 2014.

[101] Ashish Thusoo, Joydeep Sen Sarma, Namit Jain, Zheng Shao, Prasad Chakka, Ning Zhang 0002, Suresh Anthony, Hao Liu, and Raghotham Murthy. Hive - a petabyte scale data warehouse using Hadoop. In *ICDE*, Long Beach, California, USA, 2010.

[102] V. Viet Triem Tong, A.Clark, and L.Mé. Specifying and enforcing a fine-grained information flow policy : Model and experiments. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, 2010.

[103] Ankit Toshniwal, Siddarth Taneja, Amit Shukla, Karthikeyan Ramasamy, Jignesh M Patel, Sanjeev Kulkarni, Jason Jackson, Krishna Gade, Maosong Fu, Jake Donham, Nikunj Bhagat, Sailesh Mittal, and Dmitriy V Ryaboy. Storm@twitter. In *SIGMOD*, 2014.

[104] Andrei Vancea and Burkhard Stiller. Coopsc : A cooperative database caching architecture. In *WETICE*, pages 223–228, 2010.

[105] Krishnamurthy Viswanathan, Choudur Lakshminarayan, Vanish Talwar, Chengwei Wang, Greg Macdonald, and Wade Satterfield. Ranking anomalies in data centers. In *NOMS*, 2012.

[106] Louis Woods, Zsolt István, and Gustavo Alonso. Ibex : An intelligent storage engine with support for advanced sql offloading. *Proc. VLDB Endow.*, 7 :963–974, 2014.

[107] Lei Xing, Wenjun Wang, Guixiang Xue, Hao Yu, Xiaotong Chi, and Weidi Dai. Discovering traffic outlier causal relationship based on anomalous DAG. In *ICSI*, 2015.

[108] Charles Xosanavongsa. *Heterogeneous Event Causal Dependency Definition for the Detection and Explanation of Multi-Step Attacks.* phdthesis, CentraleSupélec, June 2020.

[109] Xiaowei Xu, Nurcan Yuruk, Zhidan Feng, and Thomas A. J. Schweiger. SCAN : a structural clustering algorithm for networks. In *KDD*, pages 824–833. ACM, 2007.

[110] J. Joshua Yang and R. Stanley Williams. Memristive devices in computing system : Promises and challenges. *J. Emerg. Technol. Comput. Syst.*, 9(2), may 2013.

[111] Haopeng Zhang, Yanlei Diao, and Alexandra Meliou. Exstream : Explaining anomalies in event stream monitoring. In *EDBT*, 2017.

[112] Ying Zhao and Lauren Jones. *Integrating Human Reasoning and Machine Learning to Classify Cyber Attacks*, pages 147–165. Springer International Publishing, Cham, 2021.

[113] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks : A review of methods and applications. *AI open*, 1 :57–81, 2020.