2018 IEEE International Symposium on Software Reliability Engineering Workshops

Challenges and Directions in Security Information and Event Management (SIEM)

Marcello Cinque*†, Domenico Cotroneo*†, Antonio Pecchia*†
*Dipartimento di Ingegneria Elettrica e delle Tecnologie dell'Informazione,
Universitá degli Studi di Napoli Federico II,
Via Claudio 21, 80125, Naples, Italy
†Critiware S.r.l.
Via Carlo Poerio 89/A, 80121 Naples, Italy
{macinque, cotroneo, antonio.pecchia}@unina.it

***Abstract*—Security Information and Event Management (SIEM) is the state-of-the-practice in handling heterogeneous data sources for security analysis. This paper presents challenges and directions in SIEM in the context of a real-life mission critical system by a top leading company in the Air Traffic Control domain. The system emits massive volumes of highly-unstructured text logs. We present the challenges in addressing such logs, ongoing work on the integration of an open source SIEM, and directions in modeling system behavioral baselines for inferring compromise indicators. Our explorative analysis paves the way for data discovery approaches aiming to complement the current SIEM practice.**

***Index Terms*—SIEM, security, log analysis, information retrieval, Latent Dirichlet Allocation**

I. Introduction

Situational awareness consists in *compiling*, *processing* and *fusing* data, along with the ability to comprehend their technical implications to make informed decisions [1]. Cyber security capitalizes on a variety of data sources –such as, *intrusion detection systems*, *network audit*, *integrity monitors*, *system* and *application logs*– and hierarchical transformations [2] to develop situational awareness from data.

Security Information and Event Management (SIEM) is the state-of-the-practice to address the complexity underlying the collection and normalization of diverse data sources for security analysis. SIEM is the *core component* of any typical Security Operations Center (SOC), i.e., the centralized response team dealing with security incidents within an organization [3]; however, we observe that integrating a SIEM in a given system is far from being *seamless*.

In fact, today's SIEM deployment is mostly about writing ad-hoc data collectors and *compromise indicators* –through the concept of **correlation rules**– which makes it hard for humans to keep up with large volumes of data and potentially *unknown* attacks. Setting up well-crafted indicators entails a substantial **threat knowledge** by domain experts: as such, in spite of the valuable technical advances, SIEM strongly depends on many cognitive human processes. It is well accepted that human analysts remain a critical component in successful computer defense [2].

This paper presents our industrial experience with SIEM in a real-life **Air Traffic Control** (ATC) system by LEONARDO[1], a top-leading industrial company in electronic and information technologies for defense, aerospace, and land security. The ATC system emits massive volumes of highly-unstructured text logs, which we aim to address trough SIEM. We discuss the key challenges in SIEM with respect to the logs in hand; OSSEC[2] –i.e., an open source data monitoring solution that allows configuring incidents that administrators wish to be alerted on– is also used to explore SIEM integration with the target ATC system.

Based on this experience, we put forth the construction of **behavioral baselines** to profile regular system's behavior and legitimate operations: to this objective, we present a novel approach based on the use of Latent Dirichlet Allocation [4]. Compromise detection is pursued by recognizing the actions that lie outside the regular baselines rather than relying on pre-established SIEM correlation rules. While dealing with a practical industry problem, our explorative analysis paves the way for *data discovery* approaches as a promising direction to complement the current SIEM practice.

II. Reference System

A full-fledged installation of the ATC system was made available by LEONARDO within the context of "Novel Approaches to Protect Critical Infrastructures from Cyber Attacks" (NAPOLI FUTURA), i.e., an academia-industry project led by Critiware S.r.l.[3], a spin-off company of the University of Naples. During the project we implemented a number of research initiatives and prototypes, such as *integrated monitoring*, virtual machines *live migration* and security *data analytics* with Apache Storm, and gained a strong field experience in a real industry setup.

The system –which is depicted in Fig. 2– consists of several distributed nodes, e.g., *D02*, *MN1* and *SFN*, that implement critical ATC functions, such as flight progress monitoring (*MONT*), conflict detection (*MTCD*) and a controller working position (*CWP*).

[1]http://www.leonardocompany.com/en
[2]https://ossec.github.io/docs/manual/non-technical-overview.html
[3]http://www.critiware.com/

978-1-5386-9443-5/18/$31.00 ©2018 IEEE
DOI 10.1109/ISSREW.2018.00-24

```
[03/02/18 15:54:51.410] MSG: switch: RX msg=0x40bf8208 userref=INT_USERREF status=0 bufsize=256 bufid=5491
          PAN: DELETEBUFFER msg[0x40bf8208] bufid[5491] bufsize[256] file[src/switch.cxx] line[1214]
updfdpfields – DBmask_rjd 0 MSGmask_rjd 0
[03/02/18 16:17:46.540] ITF: FlightCounter –CheckFliCounter– Pending: 0, Active: 10, Live: 10, Terminate: 2
```

Fig. 1: Few lines from the *D02-PAN* log source.

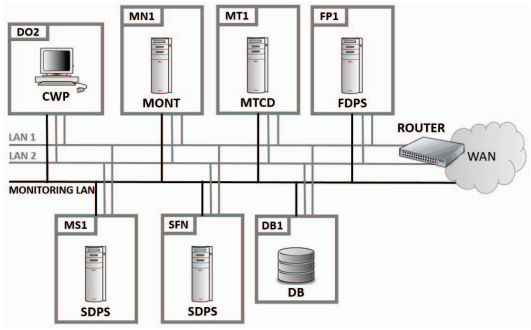

Fig. 2: Architecture Air Traffic Control (ATC) system.

The system is deployed in a controlled testing environment, and exercised with representative test suites by the industry provider [5]; the test suites issue typical ATC operations, such as *flights creation*, *re-route* or *removal*.

The system emits massive volumes of proprietary text logs. In this context, logs serve a variety of purposes, such as *control-flow tracing*, *dump* of variables or data structures, and traditional log-based *event reporting*. There exist around 20 log sources across the 7 nodes of the system, including *standard* syslogs and *legacy* application logs. Application logs do not mandate a structured format, such as the lines shown in Fig. 1. On average, the system generates more than 15,000 log lines *per minute*; lines are not uniformly distributed across the log sources. For example, Fig. 3 shows the number of new lines available in two differrent logs every 10*sec* over 30*min* of operations. The *D02-PAN* log occasionally generates around 3,000 lines within 10*sec*; *MN1-PAN* is less verbose. We aim to address such logs trough SIEM.

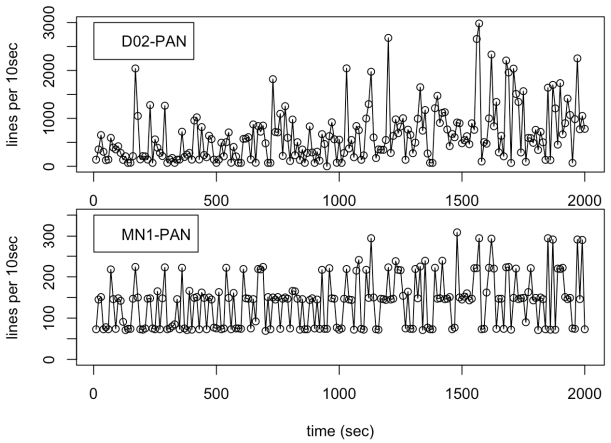

Fig. 3: Number of lines generated by two log data sources every 10*sec* over 30*min* of operation.

III. Challenges in SIEM

We discuss the key challenges in SIEM with respect to the logs in hand. Our discussion focuses on format and data analysis issues, which closely relate to the ATC system.

Data Formats. Up-to-date SIEM frameworks typically rely on internal representation formats (e.g., *MDI Fabric* and *Common Information Model* used by LogRhythm[4] and Splunk ES[5], respectively) to import and consolidate many data sources that can be encountered in production systems. Above-mentioned frameworks provide built-in adapters, which well cope with importing *structured* data into the internal representation; however, built-in capabilities are much more limited when it comes to unstructured text, such as the logs addressed by this paper. For example, in case of a *syslog* source, only a small number of standard fields (e.g., timestamp, hostname, severity) is automatically recognized by the mentioned products, while the *free-text* message is imported in its raw format.

Correlation rules. Although SIEM products provide a consolidated framework for *setting* and *monitoring* data correlation rules for compromise detection, the definition of the rules is up to the analysts. This approach closely resembles traditional business intelligence, where humans must know "in advance" what they should be looking for. However, security attacks reflect into patterns –typically spanning different raw data sources– that are unknown to analysts, and are difficult to envision, beforehand (i.e., at the time correlation rules are set up into the SIEM). Differently from this perspective, we put forth the concept of **data discovery** to complement SIEM, meaning that *data* should reveal analysts what is really interesting from a security standpoint.

Behavioral baselines. Modeling log sources for data discovery is a crucial step. In the context of our ATC system, it requires novel methods to handle unstructured text by capitalizing on *natural language processing* and *information retrieval* to reveal interesting activity and correlations within runtime data. In fact, our logs are fraught with messages intended for humans rather than machines, such as (adapted to English language):

```
WARNING I received zero on 'address'. I am
    converting to default 'DUMMY ADDRESS' value.
This is the first call to inputmsgproc – number of
    pending flights: 10.
Found not linked flight. Candidate to be linked is
    FLIGHT_ID.
```

We claim that *behavioral baselines* –which synthesize the normative system behavior– play a key role in security. In many production setting, detection is pursued by recognizing

[4]http://logrhythm.com/products/siem
[5]http://www.splunk.com

the actions that lie outside the notion of regular baseline (rather than correlation rules, such as with SIEM) because there is no *ground truth* to train anomaly detection approaches.

Lack of ground truth. The lack of *ground-truth*[6] (i.e., the knowledge of the incidents occurred in a system at the time data were collected) for *training* and *testing* incident detection approaches is well-recognized by academia and industry [6]. In many cases, security resembles an **unsupervised problem**, where it is made the implicit assumption that regular instances are far more frequent than anomalies [7]. There are several challenges to obtain incidents' labels in a production setting. First, it might be really expensive, if not even prohibitive because labeling is done by human experts. Although data are investigated upon incidents, they are usually not assigned a label in production. Then, the ground truth will likely not cover all possible types of incidents. Even more important, incidents are known to have a strong dynamic nature: new incidents might be encountered for which there is no labeled data at all. Finally, incidents-related data are far fewer when compared to the regular data, leading to very unbalanced distributions when trying to learn a supervised model. For example, authors in [8] observe that several incident classes, such as *spam* and *infected hosts*, had very few occurrences over a long time frame (5 years) in the systems addressed by their study.

IV. Explorative Integration of ATC-SIEM

We explore the use of OSSEC in the reference ATC system. OSSEC mixes together security aspects ranging from HIDS (host-based intrusion detection) to log monitoring and SIEM. Noteworthy, OSSEC is a featured component by OSSIM, i.e., a well-consolidated open source SIEM by AlienVault[7].

OSSEC provides built-in alerting for a variety of well-established data sources and protocols, such as *Apache*, *ftpd* and *syslog*. Overall, this capability is embedded through a number of `xml` files. Files encode precise detection rules that are checked against runtime data, and are available under the "*rules*" OSSEC folder. A partial listing of the files available under "*rules*" is shown in Fig. 4.

We aim to implement domain-dependent ATC detection rules in OSSEC. This objective requires an adaptation to allow OSSEC accepting our proprietary application logs. We developed a *connector* program, shown in Fig. 5. Given a log file, the **connector** i) polls for new lines at a certain sampling rate, ii) wraps each new legacy log line into a standard syslog message –accepted by OSSEC–, iii) forwards it to the OSSEC server at the default port 514.

[6]Also known as *oracle* or *label*.

[7]https://www.alienvault.com/products/ossim

```
apache_rules.xml     hordeimp_rules.xml  telnetd_rules.xml
apparmor_rules.xml   ids_rules.xml       openbsd_rules.xml
arpwatch_rules.xml   imapd_rules.xml     unbound_rules.xml
[...]
```

Fig. 4: Examples of files under OSSEC "*rules*" folder.

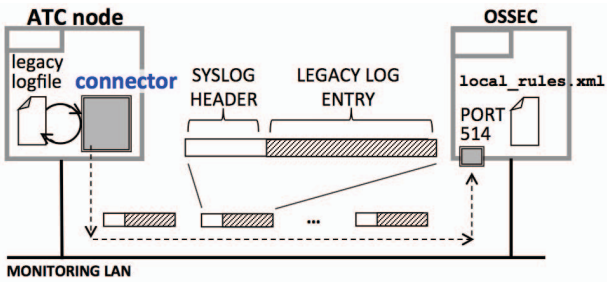

Fig. 5: OSSEC-ATC system explorative integration.

SIEM rule:

```
<rule id="100002" level="8">
  <match>calcnextsecgeog</match>
  <description>An ATC keyword for test purposes</description>
</rule>
```

Corresponding runtime alert:

```
** Alert 1522067341.4526: mail  - local,syslog,
2018 Mar 26 14:29:01 DB1->10.5.11.80
Rule: 100002 (level 8) -> 'An ATC keyword for test purposes'
Mar 26 12:29:23 DB1 PAN [03/02/18 15:46:30.595] FLI:
calcnextsecgeog - Call
```

Fig. 6: An example of domain-dependent SIEM rule/alert.

Our wrapping consists in augmenting the original line emitted by the system with a number of standard syslog fields, such as *timestamp*, *hostname* and *application*. As a result, we leveraged OSSEC's built-in syslog monitoring capabilities. For example, we could hook new SIEM correlation rules on the top of the existing syntax and alerting system. To this objective, we extended the `local_rules.xml` file.

Fig. 6 shows a proof-of-concept SIEM rule –encoded with XML syntax– that we implemented to raise an alert whenever the ATC system generates a line that matches the "*calcnextsecgeog*" keyword; Fig. 6 also shows the corresponding runtime alert, where it can be noted the syslog-wrapped original line (bottom two lines of Fig. 6). Beside our augmentation, OSSEC can also reveal compromises that are tracked by conventional log sources emitted by the ATC system, such as syslog.

V. Modeling Behavioral Baselines

Work presented in Section IV resembles the typical *use case* of SIEM, where analysts hard-code compromise indicators through XML files. While we provided ATC experts with a practical approach to monitor application logs through OSSEC, it remains hard to hypothesize, *beforehand*, how security attacks will actually reflect across runtime logs and, in turn, coding effective SIEM rules. Noteworthy, differently from many conventional ICT domains, there is not yet a mature-enough **threat model** for ATC systems, which further exacerbates the use of SIEM in this domain.

As such, we put forth approaches to handle text data in order to model *regular* behavioral baselines: compromise detection is pursued by recognizing the actions in the logs that escape the regular baselines.

A. Model Construction

In the context of our ATC system we explore the use of **Latent Dirichlet Allocation** (LDA), proposed in [4]. LDA is a *topic modeling* technique. Given a set of documents, **topic models** are built upon the intuition that each document is a mixture of a number of topics and that each word in a document is attributable to one of the topics.

Given a log source in our system, a *document* consists of a chunk of log collected over a certain amount of time (10*sec* in our explorative study); a *set of documents* is a continuous sequence of chucks; *topics* represent the actions run by the system that caused the generation of certain lines and words in the log chunk.

The key input parameter for obtaining an LDA model is the **number of topics** K. We assess K by means of a sensitivity analysis. Let us describe the procedure that we used to obtain a suitable LDA model and K for a given log source L:

- We run the ATC system and the normative workload, beforehand. During the progression of the run, we split –on-line– L into 10*sec* chunks.
- The set of chunks available at the end of the run represents the input for the topic modeling step. Chunks are parsed in order to remove variables, and a **chunk-term matrix** is constructed (each element i,j of the matrix is the number of occurrences of the term j in the chunk i).
- We split the chunk-term matrix by rows into 10 folds F (with $1 \leq F \leq 10$). For each fold f we **train** an LDA model with 9 folds ($f \neq F$) and **test** it with the held-out fold (f=F); this is iterated over a set of candidate Ks.

An LDA model is tested by computing the **perplexity** parameter, which quantifies how-well the model predicts an arbitrary set of documents (typically not used for training): the lower the perplexity, the better the prediction. Models and perplexity are obtained here with the *R* **topicmodels** package [9].

Fig. 7 depicts the sensitivity of the perplexity with respect to the number of topics K for the log source *D02-PAN*. For each K (x-axis), the boxplot summarizes the perplexity obtained over the 10 folds; the solid line is the average perplexity. According to the literature [9], K=12 –which follows the *knee-point* of the perplexity– denotes an optimal number of topics.

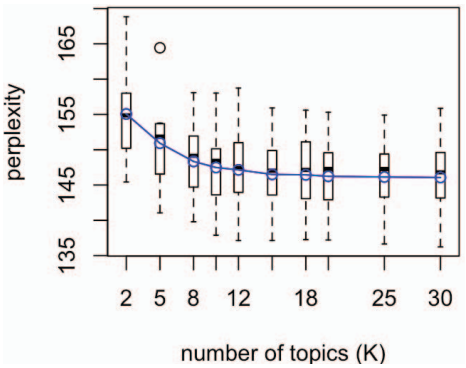

Fig. 7: Sensitivity analysis of K (*D02-PAN*).

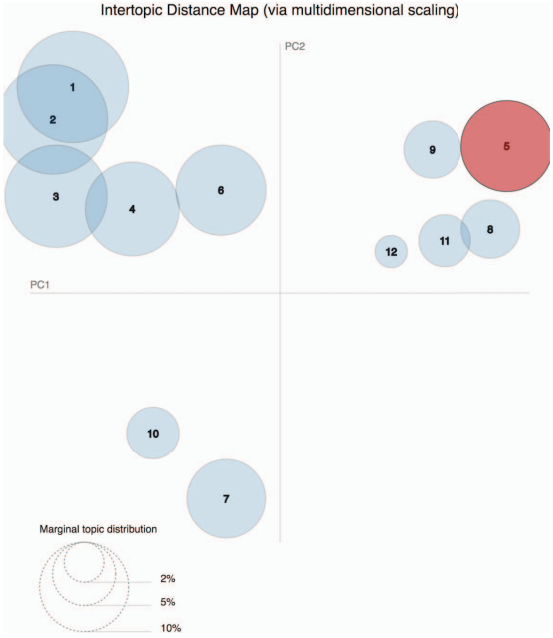

Fig. 8: LDA model obtained with 200 chunks from *D02-PAN*.

Fig. 8 provides a representation of the LDA model with the *LDAvis*[8] package obtained with 200 chunks from *D02-PAN* (12 topics). The size of the clusters represents the marginal distribution of the topics.

The *LDAvis* package provides an interactive tool for exploring and visualizing the topics. Fig. 9 shows the composition of Topic 5 and the contribution of each word to the topic and the overall documents. On average, we observe that some words are topic-specific, i.e., they are likely to appear only in one or few topics (e.g., *timer* in Fig. 9).

B. Compromise detection

Our proposition is that anomalous log chunks –caused by system misuse rather than legitimate actions issued by the ATC system– should deviate from an LDA model that summarizes regular operations. We use the *perplexity* metric to measure the extent of the deviation: a high perplexity indicates that the content of a chunk is potentially caused by actions (topics) that are not in the normative model learned above.

A preliminary experiment with the *D02-PAN* log source is presented here to substantiate this claim. We **synthesize** *anomalous chunks* by perturbing the original content of the logs. Anomalous chunks mimic regular chunks in terms of content, average number of lines and verbosity of each line (i.e., 500 and 12, for *D02-PAN*); however, they also embed a certain amount of anomalous lines consisting of **unknown terms** (i.e., terms that lie outside the behavioral baseline).

We measure the value of perplexity achieved by anomalous chunks against the normative LDA model for increasing levels of anomalous lines. Levels are set to LOW, $MEDIUM$,

[8]The model is obtained with the *R* **text2vec** package by Dmitriy Selivanov and contributors; visualization is done through the *LDAvis* package wrapped by text2vec. Please refer to http://text2vec.org/index.html.

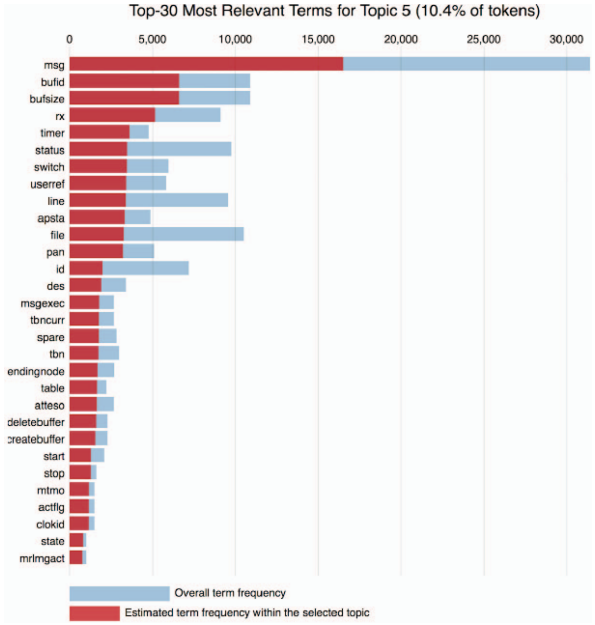

Fig. 9: Interactive visualization for Topic 5; bars indicate the frequency of a word within a topic *wrt* to the overall frequency.

and $HIGH$, which denote chunks with 1%, 5% and 10% anomalous lines out of the total size of the chunk. For each level we generate 30 synthetic chunks. Results are shown in Fig. 10. Boxplots summarize the variability of the perplexity by level; *REGULAR* shows the perplexity of regular chunks (i.e., no anomalous lines), and it is given as a reference.

Interestingly, the overlap between *REGULAR* and *LOW* boxplots is small, which means that LDA is sensitive to anomalies in the logs. For example, with a perplexity **threshold** of 165 –dashed line in Fig. 10– to classify regular/anomalous chunks, 3 regular chunks are deemed anomalous (false positives) and 2 anomalous chunks are deemed regular (false negatives). This figures lead to remarkable **recall** and **precision** of 0.93 and 0.90 in the worst case scenario *LOW*. Noteworthy, *MEDIUM* and *HIGH* boxplots do not overlap at all *REGULAR*: in these cases, LDA allows pinpointing all anomalous chunks with *no* false positives/negatives.

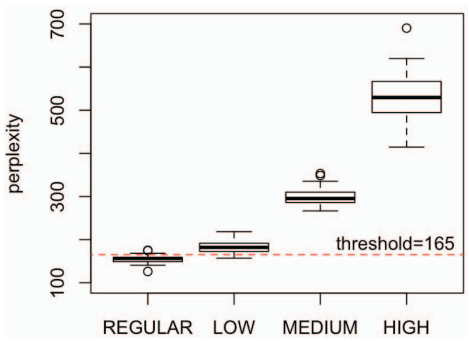

Fig. 10: Perplexity of regular and anomalous (*LOW*, *MEDIUM*, *HIGH* levels) log chunks.

This data modeling experience paves the way for automated approaches to mine *interesting activity* –i.e., lines that should be followed up by security analysis– from unstructured text logs. It must be noted that, differently from many existing approaches, the adoption of LDA for modeling behavioral baselines requires i) no labeled datasets for training purposes, ii) no prior domain/background knowledge in terms of SIEM rules, and iii) it is entirely automated. This is a promising direction to complement the current SIEM practice. To the best of our knowledge we are aware of very few initiatives in topic modeling for the analysis of security logs [10].

VI. Conclusion and Future Work

This paper described our experience with SIEM in a real-life industry system. While dealing with a practical problem, our analysis paves the way for novel approaches to complement the current SIEM practice. In the future, we will explore the use of *plugins* to extend traditional SIEM's capabilities; for example, AlienVault provides a large number of plugins, and it supports creating new plugins *from scratch*. We will extend the experiments in order to validate our preliminary findings, and to address correlation techniques for the detection of *stealthy* and *multi-step* attacks across the logs. Visualization is a further research area that we will investigate in the future. Finally, we plan to develop proof-of-concept automatons aiming to mimic the cognitive analysts' work in investigating security data.

Acknowledgment

This research is partially supported by Programme STAR, financially supported by UniNA and Compagnia di San Paolo under project "Towards Cognitive Security Information and Event Management" (COSIEM) and by the RT-CASE project, funded by the Dept. of Electrical Engineering and Information Technology of the Federico II University of Naples, Italy.

References

[1] U. Franke and J. Brynielsson. Cyber situational awareness - A systematic review of the literature. *Comput. Secur.*, 46:18 – 31, 2014.
[2] A. D'Amico and K. Whitley. The real work of computer network defense analysts. In *VizSEC 2007*, pages 19–37. Springer, 2008.
[3] S. Bhatt, P. K. Manadhata, and L. Zomlot. The operational role of security information and event management systems. *IEEE Security Privacy*, 12(5):35–41, 2014.
[4] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *J. of Machine Learning Research*, 3:993–1022, 2003.
[5] M. Cinque, R. Della Corte, and A. Pecchia. Entropy-based security analytics: Measurements from a critical information system. In *The 47th IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2017)*, pages 379–390. IEEE, 2017.
[6] D. Cotroneo, A. Paudice, and A. Pecchia. Empirical analysis and validation of security alerts filtering techniques. *IEEE Transactions on Dependable and Secure Computing*, 2017.
[7] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3):15:1–15:58, July 2009.
[8] A. Sharma, Z. Kalbarczyk, J. Barlow, and R. Iyer. Analysis of security data from a large computing organization. In *The 41st IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2011)*, pages 506–517. IEEE, 2011.
[9] B. Grün and K. Hornik. topicmodels: An R package for fitting topic models. *Journal of Statistical Software, Articles*, 40(13):1–30, 2011.
[10] J. Huang, Z. Kalbarczyk, and D. M. Nicol. Knowledge discovery from big data for intrusion detection using LDA. In *2014 IEEE International Congress on Big Data*, pages 760–761, June 2014.