Security Log Analysis in Critical Industrial Systems Exploiting Game Theoretic Feature Selection and Evidence Combination

Marcello Cinque, *Member, IEEE*, Christian Esposito, *Member, IEEE*, and Antonio Pecchia, *Member, IEEE*

***Abstract*—Critical industrial systems have become profitable targets for cyber-attackers. Practitioners and administrators rely on a variety of data sources to develop security situation awareness at runtime. In spite of the advances in security information and event management products and services for handling heterogeneous data sources, analysis of proprietary logs generated by industrial systems keeps posing many challenges due to the lack of standard practices, formats, and threat models. This article addresses log analysis to detect anomalies, such as failures and misuse, in a critical industrial system. We conduct our study with a real-life system by a top leading industry provider in the air traffic control domain. The system emits massive volumes of highly-unstructured proprietary textual logs at runtime. We propose to extract quantitative metrics from logs and to detect anomalies by means of game theoretic feature selection and evidence combination. Experiments indicate that the proposed approach achieves high precision and recall at small tuning efforts.**

***Index Terms*—Anomaly detection, Dempster-Shafer combination rule, feature selection, game theory, log analysis, logarithmic entropy.**

I. Introduction

CRITICAL industrial systems have become profitable targets for cyber-attackers because they are intertwined with many sensitive assets of our daily life, such as power grids, medical, financial, and transportation systems. Practitioners and administrators rely on a variety of **data sources**, such as *intrusion detection systems*, *network audit*, *integrity monitors,* and *system/application logs* to detect security threats and to develop continuous **situation awareness** from data [1], [2]. Recent trends put forth the role of security information and event management (SIEM) [3] products and services to collect and normalize diverse data sources for security analysis. For example, AlienVault USM,[1] IBM QRadar,[2] and Splunk Enterprise Security[3] are examples of well-known SIEM products.

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 [4]. Current SIEM products provide a variety of built-in *adapters*, for importing and parsing data from many standard protocols and applications (e.g., ftp, ssh, telnet, apache). Nevertheless, we observe that data handling capabilities of SIEM are much more limited when it comes to event log files – or simply *logs* – from legacy systems and applications that follow **proprietary formats**. Logs are sequences of text lines reporting housekeeping-error events, traces and dumps of variables collected during operations, which are a byproduct of the system's execution and can be found in many industrial settings [5].

Using SIEM to address proprietary logs is far from being seamless. In fact, current SIEM deployment is mostly about writing *ad hoc* data collectors and compromise indicators – through the concept of **correlation rules** – which makes it hard for human experts to keep up with large volumes of logs and potentially unknown incidents. Setting up well-crafted indicators [6], [7], entails a substantial **threat knowledge** by domain experts: as such, system and application logs remain quite underutilized in security analysis with respect to conventional data sources due to the lack of standard formats and coding practices.

This article addresses log analysis in a critical industrial system. We present a specific application in this context, related to the **detection of anomalies** from logs, such as, *failures* and *misuse*. We conduct our study with a real-life **Air Traffic Control** (ATC) system by a top leading industry provider in electronic and information solutions for defense, aerospace, and land security. The ATC system generates massive volumes of highly-unstructured, proprietary, text logs from various applications at runtime. In order to cope with the challenges in handling

Manuscript received June 13, 2019; revised August 12, 2019; accepted September 9, 2019. Date of publication October 2, 2019; date of current version February 28, 2020. This research was supported in part by the Programme STAR, funded by UniNA and Compagnia di San Paolo under Project "Towards Cognitive Security Information and Event Management (COSIEM)". The system installation was made available by the industrial provider within the context of "Novel Approaches to Protect Critical Infrastructures from Cyber Attacks (NAPOLI FUTURA)" Start-up Project PAC02L1_00161 supported by the Italian Ministry of Education, University and Research. Paper no. TII-19-2501. *(Corresponding author: Christian Esposito.)*

The authors are with the Dipartimento di Ingegneria Elettrica e delle Tecnologie dell'Informazione (DIETI), Università degli Studi di Napoli Federico II, via Claudio 21 80125, Naples, Italy (e-mail: macinque@unina.it; christian.esposito@unina.it; antonio.pecchia@unina.it).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2019.2944477

[1][Online]. Available: http://www.alienvault.com

[2][Online]. Available: http://www-03.ibm.com/software/products/en/qradar-siem

[3][Online]. Available: http://www.splunk.com

the logs through SIEM, we propose an automated approach for extracting quantitative metrics from logs, without making any specific assumptions on the format of underlying data and requiring no *a priori* catalog of compromise indicators. We address the challenges in computing and selecting relevant features by means of a game theoretic approach; anomaly detection is done by means of evidence combination with a variation of the Dempster–Shafer rule exploiting evolutionary game theory.

The **key contributions** of the article – with respect to the data in hand – are the following. We use a coalitional game theory as a foundation of our feature selection scheme, by modeling the selection process as a voting game and basing the selection on the Banzaf power index. Since massive amounts of data have to be processed, applying the feature selection and classification on the dataset *as a whole* is not viable. By means of a divide-et-impera approach, classification is run at the multiple log sources, and outcomes from these processes are aggregated by means of the combination rule from the theory of evidence. The classical formulation of such a rule is affected by returning counter-intuitive results, and a variation exploiting the replicator dynamics from the evolutionary game theory has been applied. A prototype of the proposed scheme has been implemented in Matlab and proved to achieve a high precision and recall at small tuning efforts.

The rest of this article is organized as follows. Section II presents related work in the area. Section III describes the system in hand, the approach for extracting the metrics from logs and the datasets used for the experiments. Section IV addresses feature selection and evidence combination. Section V presents the experiments and a comparison with pertinent state-of-the-art classifiers. Section VI discusses the threats to validity of our work and Section VII concludes the article.

II. Related Work

We position our work with respect to the use of logs for anomaly detection and the adoption of game theory approaches.

A. Anomaly Detection From Logs

The amount of data provided by logs and their subsequent mining can be leveraged to perform anomaly detection [8]. An anomaly can be typically defined as a **deviation** in the behavior of the system from its normative conditions, at a certain time. Such a deviation can manifest as a content modification within the lines in the logs, so that a significant alteration of one or more lines allows discovering an anomalous behavior of the system being monitored. This phase, however, might not give valuable information; as such, an anomaly can be benign or malign, and – in the second case – there may be multiple possible causes for a given anomaly. The assumption is that each cause has its unique footprint within the logs, so that by searching for a given footprint it is possible to go back to the anomaly and strive for a complete diagnosis to support potential recovery and/or maintenance actions. An effective and efficient anomaly detection approach (respectively, a high detection precision and recall, and minimal detection latency and costs) is important. Various applications are possible, spanning from maintenance and fraud prevention to fault detection and monitoring, and across different industrial contexts, such as finance, healthcare, manufacturing, critical infrastructure governance, and multimedia.

Anomaly detection is typically based on the extraction of **quantitative metrics** from logs, which is a crucial task. This has been shown to be valuable across different application domains for *dependability* and *security* purposes.

1) Anomaly Detection for Dependability Purposes: The work [9] presents an anomaly detection technique for sporadic Cloud operations. The technique correlates event logs and Cloud metrics to detect anomalies. Event logs are checked against a set of regular expressions to identify the type of reported events. An anomaly detection approach that leverages natural language processing is presented in [10]. The approach analyzes event logs by a software system during different events. The analysis is performed through the Google *word2vec* algorithm,[4] which maps words to a high dimensional metric space. An approach for mining console logs to detect run-time problems in large-scale systems is presented by [11]. The approach extracts structured information from console logs and constructs feature vectors from the extracted information. To this aim, both console logs and source code are analyzed to determine message types and extract variable values contained in the log. The approach then creates a vector representation of console logs by counting the number of message types in each log. The study [12] analyzes the effectiveness of different log parsers in the context of a real-world log mining task. The task requires log parsing to generate an event count matrix (through the TF-IDF heuristic), which is fed into an anomaly detection model. The results of the study highlight that parsing errors cause up to an order of magnitude performance degradation.

2) Anomaly Detection for Security Purposes: The technique in [13] aims to detect early-stage infections targeting enterprise networks. The technique leverages network logs, e.g., DNS and web proxy logs, to build a graph representing the communication between internal hosts and external domains. A graph theoretic approach, called *belief propagation*, is used to identify domains that are indicative of early-stage malware infections. Starting from a seed of known malicious domains or hosts, the approach iteratively computes scores for other domains contacted by known compromised hosts. In [14] an entropy-based security analytic approach is presented, which aims to automatically measure the occurrence of interesting activity within textual run-time log streams. The method exploit a term weighting scheme, which makes no assumptions on the structure of underlying lines in the logs. A hierarchical approach to mine high volumes of threat alarms from raw massive logs generated by an intrusion prevention system is presented in [15]. The approach uses a variant of the Choquet Integral mining technique to group alarms with similar characteristics into one cluster, then it ranks the clusters based on their characteristics and report the alarm urgency to administrators. The approach in [16], named

[4][Online]. Available: https://code.google.com/p/word2vec/

DeepLog, uses a deep neural network to model a system log as a natural language sequence. DeepLog automatically learns log patterns from normal execution, and detect anomalies when log patterns deviate from the model trained with logs from normal executions.

Log analysis for anomaly detection is intertwined with many peculiar issues, such as characteristics and scale of the input data, the need for real-time processing or batching, the presence of labeled data for training, and potential constraints on available resources. Overall, these are exacerbated in critical industrial systems – such as the one addressed by this article – due to the ever-changing nature of the data, patterns relating to anomalies, and the lack of a precise model for what is *interesting* to detect, which is typically required by SIEM. Our article aims to address these challenges by using game theoretic approaches for feature selection and evidence combination.

B. Game Theoretic Approaches

Many contributions apply game theoretic approaches to face attack detection and monitoring in computer systems. The work [17] proposes a two-stage game model to address decisions on executing attack/monitoring actions and to determine detection thresholds. Authors in [18] formulate a two-player Stackelberg security game to compute optimal thresholds for detection purposes. The article [19] addresses the problem of detecting data exfiltration in computer networks while considering a sequential game of imperfect information. Dynamic Bayesian networks combined with quantal response equilibrium (QRE) have been used in [20] to extract inference and fuse different types of insider information for behavior analysis; the approach aims to avoid traditional intrusion detection systems shortcomings when facing insider attacks. In [21] authors propose Q-Learning to react automatically to the adversarial behavior of a suspicious user to secure the system. The work compares variations of Q-Learning with a traditional stochastic game. The Dempster–Shafer theory has been used for fusing the outcome of multiple alerts/alarms from intrusion detection systems (IDSs) for detecting Distributed Denial of Service (DDoS) attacks, such as in [22], [23]. However, the reliance on a robust weighting framework has been always felt as a weakness, and the classic formulation of the aggregation rule has been applied, without considering its limits.

Feature selection is a key aspect in order to boost the precision in predicting and classifying anomalies, but the popular approaches are based on evolutionary optimization, such as in [24]. None of the available approaches leverages game theory for feature selection from logs in security analysis, which has been proven, in the case of multiobjective optimization, to have a low complexity, to offer a good scalability and to provide an empirical very near-optimal solution [25].

III. System and Datasets

We conduct our study with a real-life critical industrial system. In the following, we describe the system, our approach to extract quantitative metrics from run-time logs, and the datasets collected under normative operations and anomalous conditions used to conduct the experiments.

TABLE I
Log Sources by System Node

node	name of the log file
D02	D02PAN, D02msg
DB1	DB1PAN, DB1msg
FP1	FP1PAN, FP1NTN, FP1LNR, FP1AFS, FP1msg
MN1	MN1PAN, MN1MNA, MN1msg
MS1	MS1PAN, MS1msg
SFN	SFNPAN, SFNmsg

A. Reference System

The system consists of a set of distributed nodes that host **ATC applications**. The applications handle different inputs, such as radar traces, weather data, and *flight plans*, i.e., expected routes, trajectories, and vector information of flights. Communication across the nodes is ensured by redundant LANs; a dedicated LAN is used for monitoring purposes. Examples of typical ATC applications are: *trajectory monitoring*, which implements flight progress monitoring and trajectory recalculations; *database server*, which stores flight data, and *controller working position*, i.e., an human machine interface (HMI) dealing with interactive visualizations of flight plans, trajectories, conflicts, weather-related information and other support data to ATC operators.

A full-fledged installation of the ATC system was granted by the industry provider within the context of the project "Novel Approaches to Protect Critical Infrastructures from Cyber Attacks" (NAPOLI FUTURA). The system is deployed in a controlled environment; the **workload**, i.e., the library of inputs submitted to the system, reflects the nominal usage profile of the applications by real ATC operators. In this respect, the system is exercised with 1) operational radar traces from real aircraft operations, which were recorded in the field by the industry provider and are replayed in our controlled environment, and 2) test suites developed by the industry provider, which invoke typical ATC operations, such as creating and updating flight plans, monitoring trajectories, generating reports, acquiring and transferring flight controls.

B. Log Sources and Computation of the Scores

The system generates massive volumes of proprietary text logs. On average, the system generates more than 15 000 log lines per minute, which are used by the applications to record various events occurred during operations. In this context, logs serve a variety of purposes, such as *control-flow tracing*, *dump* of variables or data structures, and traditional *event reporting*. In this article, we consider a total of **16 log files** across the nodes of the system, which are listed in Table I by originating node. Logs include standard *syslogs*, i.e., `*msg` file names in Table I and legacy application logs. Logs do not mandate a structured format. Fig. 1 shows examples of lines from two log files called `D02PAN` and `MN1MNA`. It can be noted that lines are strongly heterogeneous in both formats and semantics.

```
1 [03/02/18 15:54:51.410] MSG: switch: RX msg=0x40bf8208 userref=INT_USERREF status=0 bufsize=256 bufid=5491
2           PAN: DELETEBUFFER msg[0x40bf8208] bufid[5491] bufsize[256] file[src/switch.cxx] line[1214]
3 updfdpfields — DBmask_rjd 0 MSGmask_rjd 0
4 [03/02/18 16:17:46.540] ITF: FlightCounter —CheckFliCounter— Pending: 0, Active: 10, Live: 10, Terminate: 2
5 19 09:26:13.936 [RCDT—006] Acd: Message NOTORDVOLO [0x63d4c000] received from ACD.
6 19 09:26:13.992 [AC03—026] Warning: No DBConnection found in TM for flight key '1630500'
7 19 09:26:14.019 [AC03—026] Acd: HandleNotifyOrder NOTIFYORDER received — FIF RGG2658 Order OK.
8 19 09:26:14.019 [AC03—026] Env: lockWriter [RGG2658]: 1
```

Fig. 1. Example of lines from two log files: `D02PAN` (lines 1 to 4) and `MN1MNA` (lines 5 to 8).

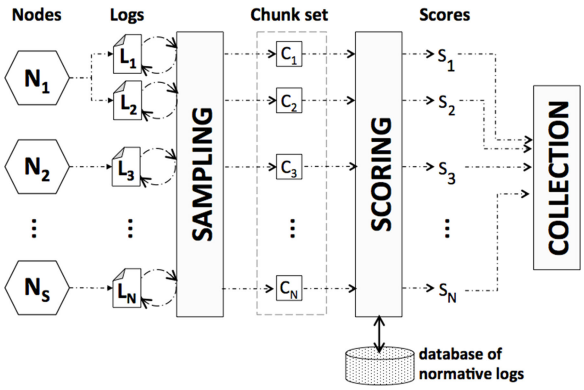

Fig. 2. Proposed log analysis approach.

To cope with structure and format heterogeneity, we propose to transform logs into **quantitative scores**. Scores can be handled more conveniently for feature selection and anomaly detection with respect to *raw* text lines. Transforming logs into metrics is done with no application knowledge, no assumptions on the format/semantics of underlying logs, and no *a priori* catalogues of suspicious events' symptoms. Let us consider a system composed by the nodes N_j ($1 \leq j \leq S$), such as shown in Fig. 2, which generate a given number of **log files** denoted by L_i ($1 \leq i \leq N$). Transforming logs into scores is based on three tasks, i.e., *sampling*, *scoring,* and *collection*.

Sampling is a periodic task that stores the most recent lines in the logs at regular intervals. Given a log L_i, every P time units the *sampling task* acquires the lines of L_i generated during the past P, creating a *chunk*. In our analysis the period P is set to 10 sec: it is a tradeoff between the latency of the detection and the need for ensuring a reasonable number of lines per chunk. Sampling is done for each L_i; therefore, the task generates a set of chunks – **chunk set** hereinafter – at every P. The *chunk set* encompasses chunks C_i corresponding to the logs L_i; overall, a *chunk set* consists of the lines in the logs generated by the system during the past P.

The **scoring** task computes a quantitative score for each chunk in the *chunk set*. There exist many methods for computing the scores, such as [9]–[11], [13], [14], [26], [27]. Here we use a **term-weighting approach**, such as [14] and [27], which was demonstrated to be effective for handling text logs and that was easy to be ported to our ATC domain. For each chunk C_i, the scoring task 1) extracts all the terms – a *term* is a sequence of characters separated by one(more) whitespace(s) – from the chunk, and 2) counts the occurrences of each term within the chunk. Let x_t denote the number of occurrences of the term t in the chunk (with $1 \leq t \leq T$, where T is the total number of terms in C_i). The score of the chunk C_i (also known as *logarithmic entropy*) is $S_i = \sqrt{\sum_{t=1}^{T} (e_t \cdot \log_2(1 + x_t))^2}$, with $S_i \geq 0$. In particular, e_t (with $0 \leq e_t \leq 1$) is the *entropy* of the term t, which is the amount of information carried out by t across the chunks that belong to a given log file. The **entropy** e_t is computed by checking x_t against an historical database of normative logs collected during regular system operations (represented in Fig. 2). Such a database is populated offline. Due to space limitation, we do not present all the steps that underlie the computation of the entropy. Noteworthy, e_t tends to be high for the terms of C_i that occur *no* or *few* times in the database of normative logs.

The output of the *scoring task* is a **vector of scores** S_i ($1 \leq i \leq N$), i.e., one score per chunk in the *chunk set*. The vector of scores is archived by the **collection** task; a new vector of scores is generated at every P and it is archived for subsequent analysis.

C. Datasets

Datasets used to conduct the experiments encompass vectors of scores computed from the logs of the ATC system during normative operations and anomalous conditions.

Normative operations are obtained with the workload generated by a set of test suites, which are used by the industry provider to emulate the nominal usage profile by real ATC operators. As explained above, the test suites issue typical ATC operations. Moreover, the system is exercised with representative radar traces recorded by the provider.

Beside normative operations, we also collect the logs during further five settings, each reproducing a potential **anomaly**, such as a brute-force authentication scan, misuse of ATC functions or tampering with operating system processes and resources. We reproduce the anomalies during the execution of the normative test suites, so that anomalous events are overlapped with regular operations as it would occur during system production. The choice of the anomalies is inspired by the well-consolidated attack phases in [28]:

1) **brute-force scan**: an attacker attempts to gain unauthorized access to the system. This is emulated through brute-force login attempts at the *D02* node – the front end of the ATC system – with the aim of guessing the credentials of a legitimate user;
2) **log deletion**: an attacker attempts to cover his/her traces by deleting some of the application logs;

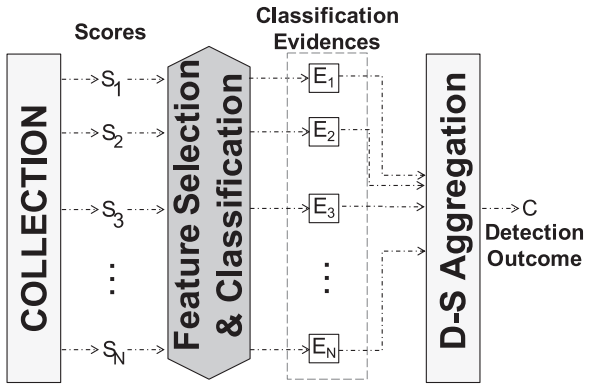

Fig. 3. Proposed classification approach.

3) **hang**: one of the ATC applications becomes unresponsive and it does not deliver any service within a reasonable timeframe;
4) **denial of service**: an attacker misuses the system for personal gain. He/she requests the creation of a new flight every second with the aim of slowing down the system response;
5) **reroute**: an attacker tampers with the flight data in the system: a flight is rerouted every five seconds and rerouting is done toward the same destination.

Logs collected during normative and anomalous conditions are split into chunks in order to obtain vectors of scores by means of the analysis steps described in Section III-B. Noteworthy, we *augment* each vector with a **binary label**, i.e., `0` (*NORM*) or `1` (*ANOM*), which denotes whether the vector comes from a normative or an anomalous collection setting. The label is used to assess the effectiveness of anomaly detection as explained in Section V-A. We *randomly* select 75% normative/anomalous vectors from the entire dataset in order to obtain the **training set** that will be used to train the detectors; the remaining 25% vectors represent the **test set**, which is used to assess the accuracy of the detection and for comparison purposes. Training/test sets are disjoint and reflect normative and anomalous vectors uniformly.

IV. Proposed Approach

Based on the streams of scores computed from the logs over time, we aim to detect anomalies that cause a deviation from the normative system behavior. To this objective, it is important to extract the relevant features from the scores, which is an open challenge. As a concrete example of detection approach from [14], if the score of a chunk is below a given threshold, while another score is above another threshold, then an anomaly is assumed to have occurred. Such an approach is typically determined and maintained by a domain expert after having scrutinized the data. However, in order to improve the robustness of the anomaly detection an automatic approach for feature selection is demanding. Fig. 3 shows the approach proposed in the article, which consists of two key parts.

The stream of scores from each log file is seen as a signal varying over the time, which can be summarized by considering information-theoretic and statistic features computed from the signal, such as the mean, mode, median, max, min, range, variance, standard deviation, the third and fourth standardized sample moment, skewness, harmonic mean or interquatile range, Shannon entropy, or Log energy. The first problem to be considered, namely **feature selection**, is to select one or more of these features so as to have a better characterization of the signal for anomaly detection. Our solution to such a problem is described in the Section IV-A.

Feature selection might be applied by jointly considering all the scores from all the logs of the system; however, this might be overwhelming because in a generic critical system multiple log sources are available. So, it is more reasonable to have feature selection performed multiple times – one per each log file – so that we have multiple event estimations for the target classes. At this time, the second problem arises: how to **aggregate** multiple estimations in order to achieve a *global* detection. Our solution to aggregation is presented in Section IV-B.

A. Game Theorethic Feature Selection

We can indicate with s_i the stream/signal of scores coming from the *i*th log, while f_j indicates the *j*th feature we can compute over a given signal as $v_{ij} = f_j(s_i)$, i.e., the value of the *j*th feature applied to the $i-th$ signal. If we indicate with c_k the *k*th target class representing a certain anomaly, we aim at finding a subset S of all the available features, whose total number is indicated with ν, that are relevant to a target class and interdependent among each others

$$\arg\max_S I(S; c_k), \quad \text{s.t. } |S| \leq \nu \tag{1}$$

where the function I measures the relevance of a group of features to a given target class [29]. Specifically, the mutual information between two variables X and Y is the measure of the shared information among them, or the information the first variable holds about the other one. We can use such a measure to obtain the relevance of a give feature f_i, or a set of features, to a target class c_k when applied to a signal s_i as the entropy of the target class minus the entropy of the target class known the values of the signal when the selected feature is applied

$$I(f_j(s_i); c_k) = H(c_k) - H(c_k | f_j(s_i)) \tag{2}$$

so that the overall mutual information of a set of features S is given by the mean of the mutual information of its elements

$$I(S; c_k) \approx \frac{1}{|S|} \sum_{f_j \in S} I(f_j(s_i); c_k). \tag{3}$$

Such a feature selection can be computed in isolation per each signal or globally by considering all the available signals and features kinds. In both cases, solving the optimization in (1) is known to be an NP-hard problem, because the set of possible combinations of features grows exponentially. So, computing it globally for all the possible features and data sources can be overwhelming.

Within the current literature there are multiple means for feature selection [30]; however, most of them are likely to ignore some features having strong discriminatory power as a group but weak as individuals. In fact, it is possible to have multiple features that together can have a high impact on the event

detection or classification, but each of them has a low individual impact. This is because the typical information theory-based measurements to determine feature interdependence and relevance fail to get a glimpse of how multiple features collaborate with each other and what is the contribution of each feature to the overall performance of a classifier or detector. To this aim, a series of new methods flourished by drawing from the theory of **coalitional games** [31], so as to optimize the task of feature selection. In such a class of games, players do not act individually, without interacting or cooperating with the others, but they group together and the joint actions are studies so as to bring a gain to the group and its members. The aim of the game is to study how groups are made and how the group gain is distributed among the group participants.

A coalitional game [32] is presented by the following couple (N, ν), where N is the player set while $v(S)$ for every $S \subseteq N$ is a function that assign a real number to a given coalition of players. Such a mapping is named as characteristic function, if the returning real number falls within the interval [0,1] the game is named as simple (1 is assigned to the grand coalition N containing all the players), and assign a payoff to a given coalition, where the empty coalition gets a null value. Each player for being a member of a coalition gets an individual gain out of the payoff obtained by the coalition, and such assignment is fair, in the sense that the individual gain is proportional to the contribution of the player to the coalition success. Multiple possible coalition can be formed, and the player must decide which coalition to join based on the individual gain it can achieve, without the possibility of leaving a coalition to join another one since being more beneficial. In order to quantify such a bargaining power of a player within a coalition the **Shapley cost-sharing rule** [32] must be computed as a possible solution to the coalitional game. The marginal importance of a player within a coalition is given by the difference of coalition payoff with and without the given player, as follows:

$$\Delta_i(S) = \nu(S \cup \{i\}) - \nu(S), \quad \text{with } i \notin S \tag{4}$$

so that the Shapley value is the average of its marginal importance over all the possible coalitions it can join, as given by the following payoff function:

$$\Phi_i(\nu) = \frac{1}{n!} \sum_{\pi \in \Pi} \Delta_i(S_i(\pi)) \tag{5}$$

where n is the number of players, Π is the set of all possible permutations of the players over the set N while $S_i(\pi)$ is the group of players before the ith player in the permutation π. The limitation of the Shapley value is that its computation is NP-hard and cannot be determined in polynomial time, so a number of heuristics come up in order to simply its calculus, such as utilizing the **Banzhaf power index** [33]. Such a solution is applied to the so-called voting games, where there is a winning coalition (i.e., the one having the highest gain or $\nu(S) = 1$) and a losing one (i.e., the one having the highest gain or $\nu(S) = 0$). The Banzhaf power index measure by simply counting, for each player, the number of winning coalitions it can participate but which are not winning if it does not participate. Within the context of a feature selection process, all the features can be assumed as the players of a simple coalitional game, which, similarly to a voting game, there can be two possible coalitions, the winning one containing the selected features, and the losing one containing the discarded features. A coalition wins over the other one if its mutual information is greater. A given player is named as a swinger if its removal from a winning coalition has the effect of making it a losing coalition, or $\forall f_j \in S : \nu(S) = 1 \vee \nu(S\{f_j\} = 0$. The Banzhaf power index of the jth player f_j is the normalized number of times such a player plays the role of swinger

$$\beta_j(\nu) = \frac{\eta_j(\nu)}{2^{n-1}} \tag{6}$$

where the function $\eta_i(\nu)$ count the times the player is a swinger for all the possible winning coalitions, while the denominator of the formula is the possible number of coalitions that the players can form. Based on the previous equations of mutual information, we can determine the conditional mutual information of a feature with respect to a target class known another feature as follows:

$$I(f_j(s_i); c_k | f_p(s_i)) = H(f_j(s_i)|c_k) - H(f_j(s_i)|c_k, f_p(s_i)) \tag{7}$$

where $i \neq p$. Two features are said to be interdependent if $I(f_j(s_i); c_k|f_p(s_i)) > I(f_j(s_i); c_k)$, so that if we consider a set of features S and a the feature f_j, we can count how many features in S are interdependent with f_j as follows:

$$\gamma_j^S = \mathbf{1}(I(f_p(s_i); c_k|f_j(s_i)) > I(f_p(s_i); c_k)) \ \forall f_p \in S. \tag{8}$$

The feature f_j is a swinger for S if it increases the relevance of the set of features if added and the number of interdependent belonging features is greater than half, or

$$\zeta_j^S = \begin{cases} 1, & I(f_j(s_i); c_k|f_p(s_i)) > I(f_j(s_i); c_k), \gamma_j^S \geq \frac{|S|}{2} \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

We conclude that the Banzhaf power index can be computed as follows:

$$\beta_j(\nu) = \frac{\sum_{S \subset N \setminus j} \zeta_j^S}{2^{n-1}}. \tag{10}$$

The Banzhaf power index is the guidance to the feature selection to resolve optimization in (1). At each iteration, the mutual information of the unselected features with the target class is computed according to (3) weighted by its Banzhaf power index. The obtained value is called victory degree, and the feature with the highest value is included in the selected feature set, and the overall approach is repeated until reaching the desired number of features, i.e., k, as illustrated in Algorithm 1.

B. Game Theorethic Estimation Aggregation

A naive solution to estimation aggregation is a majority voting scheme, where if the majority of the detectors have the same outcome, it will be assumed as the global estimation. However, it is not rare that a single detector cannot reach an outcome (i.e., it cannot assign one of the target classes as its outcome), so this naive solution is complex to apply. A more suitable approach can be the **Dempster–Shafer (D–S) combination rule** [34]. Specifically, the set of target classes is equivalent to the so-called

Algorithm 1: Feature Selection Based on Coalitional Game.

Data: Set of Features F, Set of signals S, target class c_k
Result: set of selected features with size k
initialization: R = $\emptyset$;
while $k > 0$ **do**
  for *each element* f_j *in* F **do**
    Compute the mutual information for f_j: $I(f_j(s_i)|c_k)$;
    Compute the Banzhaf power index for f_j: $\beta_j(\nu)$;
    Compute Victory degree $V(f_j) = I(f_j(s_i)|c_k) \cdot \beta_j(\nu)$;
  end
  Select f_{max} with highest Victory degree;
  $F = F \setminus f_{max}, R = R \cup \{f_{max}\}$;
  $k = k - 1$;
end

Algorithm 2: Evidence Combination Based on the Replicator Equation of an Evolutionary Game.

Data: Set of evidences supporting target classes in C
Result: The most supported target class $c - k$
initialization: R = , $k = 0$, compute mass functions in M;
while $k > 0$ **do**
  Update mass functions in M according to Equation 14;
  $k = k + 1$;
  if $\exists! e \in M : e > 0$ **then**
    break;
  end
end
Select c_k as the only element with $m(c_k) > 0$;

discernment frame, and if an individual estimator is not able to provide a response, it is possible to have a null element $\emptyset$, or a subset of the target classes. The super-set composed by 2^N combinations of the target classes consists in the corresponding power set of the discernment frame, namely Ω, where there are singleton elements, one per each target class, or their combinations. Based on the outcome of each individual estimator, we can define a mass function, whose domain assumes values in [0,1], measuring in percentage the preference of the estimators toward a given target class. It can be defined according to the Lagrangian definition of probability as the number of estimators returning the given class, as a singleton or in a proper combination with other classes, normalized over the total number of estimators in the system. Thus, the obtained overall mass value is used to quantify the believe of a given target class

$$bel(c_k) = \sum_{\emptyset \neq B \subset A} m(B) \;\; \forall A \subseteq \Omega. \tag{11}$$

The probability to globally return a target class is bounded between belief from (11), which is the total belief that the target class is true and plausibility, which is the minimum amount of belief related to the target class

$$bel(c_k) \geq P(H_i) \geq pl(H_i) = bel(\Omega) - bel(c_k). \tag{12}$$

It is possible to derive a crisp number from the previous interval by means of the so-called pignistic probability transformation, which is built based on the expected utility theory to represent beliefs when taking the optimal decision that maximizes the expected utility within the context of a decision-making process. Such a transformation determines the pignistic probability as follows:

$$BetP(A) = \sum_{W \subseteq \Omega} = \frac{m(W)}{|W|}, \;\; A \in W \tag{13}$$

where $|W|$ is the number of entities contained in the W set. Therefore, the outcome of our aggregation is given by the target class that has obtained the greatest value of the pignistic probability. The main critics to D–S combination rule is to return counter-intuitive results when combining unreliable evidences [35] and/or conflicting evidences from independent sources [36]. In order to improve the detection of a potential problem in the aggregation process, special formulations of the mass functions and other concepts of the D–S theory emerged over the last decade. In [37], a different approach has been formulated in order to find the most supported and acceptable target class, based on the indications collected by the individual estimators, leading to the **Evolutionary Combination Rule** (ECR). Specifically, the elements of the frame of discernment are seen as strategies within the context of an evolutionary process, and the elements with the highest fitness survive and are used to further simplify the analysis. The elements of the game are expressed in terms of sets whose elements are taken from the frame of discernment, so the assessment of a strategy against another one can be based on the similarity among sets in terms of Jaccard similarity coefficients, namely $J_\Omega(A, B)$, where it is 1 is the two strategies are identical, 0 if different, and $\frac{c}{n}$ where c is the number of elements in common between the two strategies and n is the number of elements in the union of the two strategies. Starting from the mass functions computed from the collected evidences, it is crucial to analyze their evolution over the time, where players in a population assumes certain strategies and over the time this decision changes so that all the population may have the same strategy.

From the literature of evolutionary game theory [38], the **replicator dynamics** is a powerful concept and mathematically describes the idea that those individuals performing better (as they have the most suitable strategy) have more offspring and thus their frequency in the population grows so that after a while all the population has a single strategy and none of the other strategies will be able to invade the population). This leads to the concept of Evolutionary Stable Strategy, which will be the target class to be returned as result of the combination of all the evidences obtained from the individual estimators. According to the classic formulation of the replicator equation, the relative frequency of a strategy A evolves over the time as

$$\frac{dm(A)}{dt} = m(A)(f_A - \phi), \;\; A \subseteq \Omega, A \neq \emptyset \tag{14}$$

used by Algorithm 2, so that A is an element of the power set made of elements from the frame of discernment, f_A is the fitness

TABLE II
CONFUSION MATRIX

		detector	
		NORMATIVE	ANOMALY
label	NORMATIVE	True Negative (TN)	False Positive (FP)
	ANOMALY	False Negative (FN)	True Positive (TP)

of such an element

$$f_A = \sum_{B \subseteq \Omega, B \neq \emptyset} m(B) J_\Omega(A, B) \tag{15}$$

and ϕ is the average fitness of the current population

$$\phi = \sum_A \sum_B m(A) J_\Omega(A, B) m(B), \quad A, B \subseteq \Omega, A, B \neq \emptyset. \tag{16}$$

The analysis in [37] proved the existence of a unique solution for this evolutionary game, and the increments to the mass functions by the replicator dynamics is able to resolve conflicts and lead to the most supported strategy.

V. EXPERIMENTAL RESULTS

We assess the accuracy of the proposed anomaly detection approach and compare it with other classification methods. Evaluation metrics and results are presented in the following.

A. Evaluation Metrics

We use the label as *ground truth* (also known as *oracle*), to assess the outcome of a detector applied to a set of labeled vectors. Assessment is done according to the confusion matrix in Table II. For example, a *real* anomaly (i.e., a vector obtained during one of the anomalous settings in Section III-C) but deemed "normative" by the detector represents a false negative (FN); a *real* anomalous vector flagged as an "anomaly" by the detector is a true positive (TP). As such, FN and false positives (FP) mark the cases where the detector misclassified normative/anomalous vectors. The four sets of TN, FP, FN, and TP in Table II are the basis for calculating **precision** (P), **recall** (R), and **F-measure** (F) as follows:

$$P = \frac{TP}{TP + FP} \quad R = \frac{TP}{TP + FN} \quad F = 2 \cdot \frac{P \cdot R}{P + R}$$

accuracy is the percentage ratio of correctly classified vectors divided by the cardinality of the input set of vectors to be classified, i.e., $\frac{TN+TP}{TN+TP+FN+FP} \cdot 100$.

B. Assessment of the Proposed Approach

The proposed approach has been implemented as a set of functions and scripts in Matlab, to acquire training and test set from the ATC case study described in Section III-C, and to discriminate potential anomalies from the normative behavior. The proposed classification approach capitalizes on filtering so that the random variable describing the application of a feature to a flow of scores is binary with value 1 if a score is lower than the scalar value of the feature applied to the flow; 0, otherwise. Moreover, we have simplified the Banzhaf power index computing it with respect to the selected features at each iteration of the algorithm.

TABLE III
METRICS OF THE CLASSIFIERS ON THE TEST SET

	precision	**recall**	**F-measure**	**accuracy**
Decision Tree	0.97	0.97	0.97	97.3%
Multilayer Perc.	0.94	0.94	0.94	94.3%
CART	0.93	0.93	0.93	93.5%
Bayesian Network	0.88	0.83	0.84	82.9%
Proposed approach	0.94	0.96	0.95	98.8%

The proposed anomaly detection approach achieves a precision of 0.94; its recall is equal to 0.96, such as reported by the bottom row of Table III. The proposed approach is relatively simple to use as the only parameter to be set if the number of features to be selected (i.e., k in Algorithm 1), which is set equal to three in the conducted experiments.

C. Comparison With Existing Classification Methods

Detection of anomalies within a dataset of labeled numeric vectors can be typically done by means of well-consolidated classifiers by the data mining community. As such, we compare our approach with other classification methods. Since our datasets are labeled, we opt for supervised methods.

The leftmost column of Table III (with the exception of the bottom row that refers to the proposed approach) shows the classifiers assessed in this article, which have been selected across the most commonly adopted by the literature. It is worth noting that we selected a mixture of methods from the practitioners' perspective. In fact, tree-*based* methods, such as decision trees, and classification and regression trees (CART), produce an output that is comprehensible and useful to practitioners; multilayer perceptrons and Bayesian networks – which produce probabilities and/or weights – have lower explicative power for practical purposes.

Each classifier has been trained/tested with the datasets described in Section III-C. Table III shows the metrics of precision/recall/F-measure/accuracy obtained on the test set; all the classifiers are tested with the same set of vectors. The four classifiers are sorted by descending accuracy (rightmost column), while the bottom row is our approach. We observe that the decision tree is the top-performing with an accuracy of 97.3%; interestingly, it can be noted that the performance of our approach is in-line with the decision tree.

We closely look into the model produced by the decision tree to gain insights into pros and cons of this classifier in the context of our proposal. Fig. 4 shows some of the decision paths of the model: again, the model is learned from the vectors in the training set and it is used to classify the vectors of the test set. Fig. 4 uses the names of the log files as in Table I, which shows the logs by originating system node.

For example, many vectors obtained during *normative* operations are characterized by scores that lead to the path highlighted in bold in Fig. 4, which ends at *line 16*; on the contrary, a vector that matches `D02msg` ≤ 2.00 *AND* `SFNPAN` > 6.75 (i.e., *line 18*) is deemed anomalous. The interesting by-product of the decision tree is thus the availability of explicative rules,

```
1 D02msg <= 2.00
2 |   SFNPAN <= 6.75
3 |   |   MN1MNA <= 0
4 |   |   |   FP1AFS <= 1.40
5 |   |   |   |   DB1PAN <= 0: 1-"ANOM" (68.0)
6 |   |   |   |   DB1PAN > 0
7 |   |   |   |   |   SFNPAN <= 0.19: 0-"NORM" (8.0/1.0)
8 |   |   |   |   |   SFNPAN > 0.19
9 |   |   |   |   |   |   DB1PAN <= 0.29: 1-"ANOM" (13.0)
10 |   |   |   |   |   |   DB1PAN > 0.29: 0-"NORM" (6.0/1.0)
11 |   |   |   FP1AFS > 1.40: 1-"ANOM" (122.0)
12 |   |   MN1MNA > 0
13 |   |   |   D02PAN <= 0: 1-"ANOM" (68.0)
14 |   |   |   D02PAN > 0
15 |   |   |   |   D02PAN <= 10.7
16 |   |   |   |   |   FP1NTT <= 0.27: 0-"NORM" (133.0/4.0)
17 ... omitted ...
18 |   SFNPAN > 6.75: 1-"ANOM" (156.0)
19 D02msg > 2.00: 1-"ANOM" (135.0)
```

Fig. 4. Decision tree model and relationships among the scores.

which highlight the relationships across the logs of the system within normative/anomalous operations. These rules can be leveraged by human experts to craft security compromise indicators, such as the *correlation rules* required by up-to-date SIEM tools. As a drawback, rules are inferred from *raw* scores, with no specific feature selection strategies. In consequence, the **detection outcome** is strongly sensitive to small variations of the scores.

This can be experimentally seen by perturbing the training data by either adding a constant (e.g., 0.1) to each score with a given probability, namely 0.25, or subtracting the same constant value to each score with the same probability. This perturbation reflects into smoother classification boundaries within the training set, and here approaches like decision trees fail to be efficient and become unstable. For example, by looking at Fig. 4, it can be easily noted that a small variation of the score with respect to the threshold 6.75 of the log `SFNPAN` would lead the classification across rather different paths (i.e., either *line 2* or *line 18*). Similar considerations apply to the remaining thresholds along the paths.

Our approach is able to mitigate the impact of noise in the dataset thanks to its game theoretic approach, that applies continuous improvements and refactoring of the classification models, so that data perturbation does not have any effect. By means of the selection of a large set of features, value perturbation shall have a limited impact on the training. For example, the above mentioned perturbation strategy may have impacted the mean but not the median statistical feature.

VI. Threats to Validity

As for any measurement study, there may be concerns regarding the validity and generalizability of the proposal and results. We briefly discuss them, based on the aspects of validity listed in [39].

Construct validity: This article develops around the intuition that quantitative scores extracted from logs can be used to obtain effective features for anomaly detection. This is pursued by instantiating our experiments in the context of a real-life industrial system by a top leading provider. We rely on well-founded methods for extracting quantitative scores from logs and feature selection.

Internal and conclusion validity: We use a mixture of datasets consisting of real logs collected under normative operations and anomalies inspired by current literature. We combine scores from different log sources and test a wide set of information-theoretic and statistic features to summarize the scores. Overall, this mitigates internal validity threats and provides a reasonable level of confidence on the conclusions.

External validity: Our proposal should be applicable to other industrial systems. We require no modifications of the target system. Our proposal is inherently *nonintrusive* because it relies on the sole use of logs; moreover, the approaches for computing the scores and selecting the features do not embed any knowledge of the system in hand. The details provided should reasonably support the replication of our experience by future researchers and practitioners.

VII. Conclusion

This article proposed an approach for game theoretic feature selection and evidence combination to analyze the logs of a critical industrial system. We addressed the problem of detecting anomalies from runtime logs; the approach achieves high precision and recall at small tuning efforts.

A particularly challenging topic was the real-time anomaly detection from streaming data based on logs of production systems, so to achieve **online** detection and recovery from failures/attacks [40]. Our approach assumed a batching approach and was hardly applied for an online classification, so as a future work we planned to devise an unsupervised version of our approach by leveraging online data clustering and evolutionary game theory. This can allow to avoid the need of training data to be processed offline, to deal with ever changing anomaly patterns, and reducing the costs in the tuning of the system.

Future work will be also devoted to evaluate the proposed method on different systems as well as on different security datasets, in order to understand limits and boundaries of the techniques across other security scenarios. Moreover, future efforts will be devoted to complement current SIEM technologies with the proposed techniques, in order to deal with unstructured data sources.

References

[1] A. D'Amico and K. Whitley, "The real work of computer network defense analysts," in *Proc. Workshop Visualization Comput. Secur.*, pp. 19–37. Berlin, Heidelberg, Germany: Springer, 2008.

[2] U. Franke and J. Brynielsson, "Cyber situational awareness–A systematic review of the literature," *Comput. Secur.*, vol. 46, pp. 18–31, 2014.

[3] D. R. Miller, S. Harris, A. Harper, S. VanDyke, and C. Blask, *Security Information and Event Management (SIEM) Implementation*. New York, NY, USA: McGraw-Hill Education, 2010.

[4] S. Bhatt, P. K. Manadhata, and L. Zomlot, "The operational role of security information and event management systems," *IEEE Secur. Privacy*, vol. 12, no. 5, pp. 35–41, Sep./Oct. 2014.

[5] A. Pecchia, M. Cinque, G. Carrozza, and D. Cotroneo, "Industry practices and event logging: Assessment of a critical software development process," in *Proc. IEEE 37th Int. Conf. Softw. Eng.*, 2015, pp. 169–178.

[6] E. E. Eljadi and Z. A. Othman, "Anomaly detection for PTM's network traffic using association rule," in *Proc. IEEE 3rd Conf. Data Mining Optim.*, 2011, pp. 63–69.

[7] M. Cinque, D. Cotroneo, and A. Pecchia, "Challenges and directions in security information and event management (SIEM)," in *Proc. IEEE Int. Symp. Softw. Rel. Eng. Workshops*, 2018, pp. 95–99.

[8] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 1–58, 2009.
[9] M. Farshchi, J.-G. Schneider, I. Weber, and J. Grundy, "Metric selection and anomaly detection for cloud operations using log and metric correlation analysis," *J. Syst. Softw.*, vol. 137, pp. 531–549, 2018.
[10] C. Bertero, M. Roy, C. Sauvanaud, and G. Tredan, "Experience report: Log mining using natural language processing and application to anomaly detection," in *Proc. IEEE 28th Int. Symp. Softw. Rel. Eng.*, 2017, pp. 351–360.
[11] W. Xu, L. Huang, A. Fox, D. A. Patterson, and M. I. Jordan, "Mining console logs for large-scale system problem detection," in *Proc. 3rd Conf. Tackling Comput. Syst. Problems Mach. Learn. Techn.*, 2008, p. 4.
[12] P. He, J. Zhu, S. He, J. Li, and M. R. Lyu, "An evaluation study on log parsing and its use in log mining," in *Proc. IEEE Int. Conf. Dependable Syst. Netw.*, 2016, pp. 654–661.
[13] A. Oprea, Z. Li, T. Yen, S. H. Chin, and S. Alrwais, "Detection of early-stage enterprise infection by mining large-scale log data," in *Proc. IEEE 45th Annu. Int. Conf. Dependable Syst. Netw.*, 2015, pp. 45–56.
[14] M. Cinque, R. Della Corte, and A. Pecchia, "Entropy-based security analytics: Measurements from a critical information system," in *Proc. IEEE 47th Annu. Int. Conf. Dependable Syst. Netw.*, 2017, pp. 379–390.
[15] Y. Meng, T. Qin, Y. Liu, and C. He, "High threat alarms mining for effective security management: Modeling, experiment and application," in *Proc. IEEE Symp. Comput. Commun.*, 2018.
[16] M. Du, F. Li, G. Zheng, and V. Srikumar, "DeepLog: Anomaly detection and diagnosis from system logs through deep learning," in *Proc. IEEE SIGSAC Conf. Comput. Commun. Secur*, 2017, pp. 1285–1298.
[17] H. Wu, W. Wang, C. Wen, and Z. Li, "Game theoretical security detection strategy for networked systems," *Inf. Sci.*, vol. 453, pp. 346–363, 2018.
[18] A. Ghafouri, A. Laszka, W. Abbas, Y. Vorobeychik, and X. Koutsoukos, "A game-theoretic approach for selecting optimal time-dependent thresholds for anomaly detection," *Auton. Agents Multi-Agent Syst.*, vol. 33, no. 4, pp. 430–456, 2019.
[19] K. Durkota, V. Lisý, C. Kiekintveld, K. Horák, B. Bošanský, and T. Pevný, "Optimal strategies for detecting data exfiltration by internal and external attackers," in *Proc. Decis. Game Theory Secur.* Cham, Switzerland: Springer, 2017, pp. 171–192.
[20] K. Tang, M. Zhao, and M. Zhou, "Cyber insider threats situation awareness using game theory and information fusion-based user behavior predicting algorithm," *J. Inf. Comput. Sci.*, vol. 8, no. 3, pp. 529–545, 2011.
[21] K. Chung, C. A. Kamhoua, K. A. Kwiat, Z. T. Kalbarczyk, and R. K. Iyer, "Game theory with learning for cyber security monitoring," in *Proc. IEEE 17th Int. Symp. High Assurance Syst. Eng.*, 2016, pp. 1–8.
[22] A. M. Lonea, D. E. Popescu, and H. Tianfield, "Detecting DDos attacks in cloud computing environment," *Int. J. Comput. Commun. Control*, vol. 8, no. 1, pp. 70–78, 2013.
[23] T. Sommestad and H Holm, "Alert verification through alert correlation–An empirical test of SnIPS," *Inf. Secur. J.: A Global Perspective*, vol. 26, no. 1, pp. 39–48, 2017.
[24] J. Duan, Z. Zeng, A. Oprea, and S. Vasudevan, "Automated generation and selection of interpretable features for enterprise security," in *Proc. IEEE Int. Conf. Big Data*, 2018, pp. 1258–1265.
[25] Z. Fei, B. Li, S. Yang, C. Xing, H. Chen, and L. Hanzo, "A survey of multi-objective optimization in wireless sensor networks: Metrics, algorithms, and open problems," *IEEE Commun. Surv. Tut.*, vol. 19, no. 1, pp. 550–586, 1Q 2017.
[26] A. Pecchia, D. Cotroneo, R. Ganesan, and S. Sarkar, "Filtering security alerts for the analysis of a production SaaS cloud," in *Proc. IEEE 7th Int. Conf. Utility Cloud Comput.*, 2014, pp. 233–241.
[27] J. Stearley and A. J. Oliner, "Bad words: Finding faults in Spirit's syslogs," in *Proc. IEEE Int. Symp. Cluster Comput. Grid*, 2008, pp. 765–770.
[28] D. Ruiu, *Cautionary Tales: Stealth Coordinated Attack Howto*, 1999. http://www.ouah.org/stealthhowto.html
[29] M. Bennasar, Y. Hicks, and R. Setchi, "Feature selection using joint mutual information maximisation," *Expert Syst. Appl.*, vol. 42, no. 22, pp. 8520–8532, 2015.
[30] A. Jovic, K. Brkic, and N. Bogunovic, "A review of feature selection methods with applications," in *Proc. IEEE 38th Int. Conv. Inf. Commun. Technol., Electron. Microelectronics*, 2015, pp. 1200–1205.
[31] X. Sun, Y. Liu, J. Li, J. Zhu, X. Liu, and H. Chen, "Using cooperative game theory to optimize the feature selection problem," *Neurocomputing*, vol. 97, pp. 86–93, 2012.
[32] R. Gibbons, *A primer in game theory*. New York, NY, USA: Harvester Wheatsheaf, 1992.
[33] P. Dubey and L.S. Shapley, "Mathematical properties of the Banzhaf power index," *Math. Oper. Res.*, vol. 4, no. 2, pp. 99–131, 1979.
[34] M. Casanovas and J. M. Merigó, "Fuzzy aggregation operators in decision making with Dempster–Shafer belief structure," *Expert Syst. Appl.*, vol. 39, no. 8, pp. 7138–7149, 2012.
[35] D. Han, Y. Deng, and C. Han, "Sequential weighted combination for unreliable evidence based on evidence variance," *Decis. Support Syst.*, vol. 56, pp. 387–393, 2013.
[36] L. A. Zadeh, "Review of a mathematical theory of evidence," *AI Magazine*, vol. 5, no. 3, 1984, pp. 81–83.
[37] X. Deng, D. Han, J. Dezert, Y. Deng, and Y. Shyr, "Evidence combination from an evolutionary game theory perspective," *IEEE Trans. Cybern.*, vol. 46, no. 9, pp. 2070–2082, Sep. 2016.
[38] C. P. Roca, J. A. Cuesta A, and A. Sánchez, "Evolutionary game theory: Temporal and spatial effects beyond replicator dynamics," *Phys. Life Reviews*, vol. 6, no. 4, pp. 208–249, 2009.
[39] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén. *Experimentation in Software Engineering: An Introduction*, New York, NY, USA: Kluwer Academic, 2000.
[40] S. Ahmad, A. Lavin, S. Purdy, and Z. Agha, "Unsupervised real-time anomaly detection for streaming data," *Neurocomputing*, vol. 262, pp. 134–147, 2017.

Marcello Cinque received the graduation (Hons.) degree in computer engineering and the Ph.D. degree in computer engineering from the University of Naples, Napoli, Italy, in 2003 and 2006, respectively.

Currently, he is an Associate Professor with the Department of Electrical Engineering and Information Technology of the Federico II University of Naples, Naples. His research interests include dependability assessment of critical systems and log-based failure analysis.

Dr. Cinque is Chair and/or Technical Program Committee member of technical conferences and workshops on dependable systems, including IEEE International Conference on Dependable Systems and Networks, European Dependable Computing Conference (EDCC), and International Conference on Dependability (DEPEND).

Christian Esposito received the Ph.D. degree in computer engineering and automation from the Federico II University of Naples, Naples, Italy, in 2009.

He is currently an Assistant professor at the Department of Electrical Engineering and Information Technology (DIETI). His research interests include reliable and secure communications, middleware, distributed systems, positioning systems, multi-objective optimization, and game theory.

Dr. Esposito has served as a Reviewer for several international journals and conferences, and has been a Program Committee member or organizer of about 40 international conferences/workshops. He has also served as Guest Editor for several journals, and is a member of three journal Editorial Boards.

Antonio Pecchia received the B.S., M.S., and Ph.D. degrees in computer engineering from the Federico II University of Naples, Naples, Italy, in 2005, 2008, and 2011, respectively.

He is currently an Assistant Professor with the Department of Electrical Engineering and Information Technology of the Federico II University of Naples. He is a cofounder of the Critiware spin-off company. His research interests include data analytics, log analysis, empirical software engineering, dependable and secure distributed systems.

He serves as Technical Program Committee member and Reviewer in conferences and workshops on software engineering and dependability, such as International Symposium on Software Reliability Engineering (ISSRE), European Dependable Computing Conference (EDCC) and Latin-American Symposium on Dependable Computing (LADC). He is the Editor-in-Chief of the *International Journal of Open Source Software and Processes* and an Associate Editor of IEEE ACCESS.