Mining known attack patterns from security-related events

Nicandro Scarabeo, Benjamin C. M. Fung, Rashid H. Khokhar

Managed Security Services (MSS) have become an essential asset for companies to have in order to protect their infrastructure from hacking attempts such as unauthorized behaviour, denial of service (DoS), malware propagation, and anomalies. A proliferation of attacks has determined the need for installing more network probes and collecting more security-related events in order to assure the best coverage, necessary for generating incident responses. The increase in volume of data to analyse has created a demand for specific tools that automatically correlate events and gather them in pre-defined scenarios of attacks. Motivated by Above Security, a specialized company in the sector, and by National Research Council Canada (NRC), we propose a new data mining system that employs text mining techniques to dynamically relate security-related events in order to reduce analysis time, increase the quality of the reports, and automatically build correlated scenarios.

1 # Mining Known Attack Patterns from Security-Related Events

Nicandro Scarabeo
University of Cassino and
Southern Lazio, Italy
nicandro.scarabeo@unicas.it

Benjamin C. M. Fung
McGill University
Montreal, QC, Canada
ben.fung@mcgill.ca

Rashid H. Khokhar
Concordia University
Montreal, QC, Canada
r_khokh@ciise.concordia.ca

4 ## Abstract

5 Managed Security Services (MSS) have become an essential asset for companies to have in order
6 to protect their infrastructure from hacking attempts such as unauthorized behaviour, denial of
7 service (DoS), malware propagation, and anomalies. A proliferation of attacks has determined
8 the need for installing more network probes and collecting more security-related events in order
9 to assure the best coverage, necessary for generating incident responses. The increase in volume
10 of data to analyse has created a demand for specific tools that automatically correlate events and
11 gather them in pre-defined scenarios of attacks. Motivated by Above Security, a specialized
12 company in the sector, and by National Research Council Canada (NRC), we propose a new data
13 mining system that employs text mining techniques to dynamically relate security-related events
14 in order to reduce analysis time, increase the quality of the reports, and automatically build
15 correlated scenarios.

17 ## 1. Introduction

18 Security Operations Centers (SOC) represent a cornerstone of contemporary security services
19 and are at the core of Managed Security Services (MSS). MSS are generally based on a set of
20 distributed sensors that are deployed on clients' networks. Each sensor contains various security
21 tools such as Intrusion Detection Systems (IDS), asset detection tools, flow analysis tools, etc.
22 The said sensor tools analyze network traffic and send events to a central database (DB)
23 repository for storage or further analysis. Security analysts can display information about
24 security or network events (alerts or logs) and access their details through a Graphical User
25 Interface (GUI) of the SOC. In order to improve overall attack coverage and satisfy the demand
26 for more advanced service features, the volume of data that is collected in the sensor sensibly
27 increases, and it becomes difficult for security analysts to maintain monitoring and analyzing
28 increasingly large quantities of data without incurring *Service Level Agreement* (SLA) violations.

29 Many companies have tried to solve this issue by introducing pre-defined correlation rules in
30 order to identify known network issues and to address client requests. This methodology
31 represents a valid solution to maintain a high service quality, but it does not easily support the
32 integration of new data sources. This method requires considerable maintenance and tends to
33 exclude all collected events that are not part of any correlation rules. In fact, security analysts
34 tend to dedicate most of their time analyzing pre-defined scenarios and writing reports without
35 being able to leverage the unmapped data in order to identify traces of threats.

36 An alternative to tackle the problem is the introduction of data mining techniques. In fact the
37 process which studies how to mine implicit and unknown information from very large volume of
38 data, by using knowledge discover techniques, is known in literature as Data Mining. In the last
39 decade few works have already applied data mining to cyber security in order to detect abnormal
40 patterns in intrusion detection processes (B. Thuraisingham, 2008) (V. Chandola, 2006). These
41 papers were mostly deriving patterns by analyzing information derived by OSI model layer 3,
42 OSI model layer 4, timestamps or, in general, structured protocol-related data stored in
43 databases.
44
45 The objective of this paper is to develop an application that enables security analysts to
46 efficiently analyze large amounts of security information by automatically mapping the
47 message contained in the security-related events to *Common Attack Pattern Enumeration and*
48 *Classification* (*CAPEC*) (Mitre, 2015), which is a publicly available comprehensive dictionary
49 and classification taxonomy of known attack patterns that can be used by analysts, developers,
50 testers, and educators to advance community understanding and enhance defenses. Instead of
51 identifying sequences of events or relating them only through common protocol-related
52 information (i.e., common IP, port, etc.), the proposed system aims to gather events by relating
53 them to known attack patterns, link their phases by analyzing the terms used to describe specific
54 security issues, and applying text mining, which is the process of deriving knowledge (patterns
55 and trends) from unstructured data.. The application is developed in Python.
56
57 An attack pattern gathers the steps, the challenges, and the techniques that need to be executed in
58 order to exploit a vulnerable system. CAPEC (Mitre, 2015) represents an attack pattern as an
59 entity represented by a global description of the attack; a detailed list of macro-steps, subdivided
60 in actions that the attacker could follow; possible consequences of the attack; and methodologies
61 for mitigating it.
62
63 In this paper, the semantic relatedness between sensor event messages and attack pattern fields
64 description has been calculated in order to identify in real time which captured event is useful to
65 fully describe the current attack. Thus, the goal of the present work is to identify the top-K
66 similar security events which can represent a specific attack step, defined in CAPEC. The set of
67 events is presented to the analysts in the form of a recommendation. The analysts can then
68 review it and automatically build a report, which is going to be stored for future analysis and
69 recommendations.
70
71 ***Contribution.*** The major challenge is that there is no direct mapping between CAPEC attack
72 patterns and security event logs. It is non-trivial task to dynamically link a collection of unrelated
73 security events to the attack pattern(s) in CAPEC because both entities are described in
74 unstructured text. In this paper, we present a data mining system that employs text mining
75 techniques to dynamically relate the information between the security events and CAPEC attack
76 patterns. To the best of our knowledge, this is first work to automatically and semantically build
77 attack scenarios by referring to a publicly available collection of attack patterns, resulting in the
78 following capabilities:
79   • With an automatic system gathering collected events, analysts do not need to dig into a
80   high volume of data in order to look for specific information. This significantly reduces
81   the analysis time, referred as the time the analyst spends to look for stored events which

82 could provide strong evidence of the attack under investigation. Thanks to the system
83 here presented, the analyst can receive a list of automatically recommended events that
84 are relevant to the attack under investigation.

85 • The system considerably increases the number of correlated and analyzed events
86 included in an attack report prepared by security analysts for the clients. Furthermore, the
87 quality and accuracy of the reports have been greatly improved. The mapped CAPEC
88 attack patterns not only associate to security events as concrete evidence, but also make
89 recommendations for how to address the security in the "Solutions and Mitigations"
90 section of the associated CAPEC attack patterns.

91 • Extensive experimental results over different types of real-life security events illustrate
92 the effectiveness of our proposed data mining system. We also demonstrate that our
93 method can successfully build correlated scenarios.

94 The rest of the paper is organized as follows. Section 2 reviews background knowledge about the
95 context in question. Section 3 reports related work. Section 4 formally defines the research
96 problem. Section 5 describes the proposed solution. Section 6 discusses the possible limitation of
97 the proposed method and future improvements. Comprehensive experimental results are
98 presented in Section 7. Finally, Section 8 concludes the paper.
99
100
101 ## 2. Background

102
103 This section first introduces the monitoring analysis process which is executed in Security
104 Operation Centers (SOCs). The section continues by articulating the motivation that brought the
105 authors to develop such work. At the end, an example clarifies the usage of this application.

106 *SOC Workflow.* Figure 1 depicts the workflow in a SOC.

107
108 **Figure 1: Workflow in a security operation center (SOC)**

109 The process consists of four different phases, described as follows:
110

111 • **Collection Layer**. The collection layer captures the security-related events that are
112 triggered by specific software. Typical events include (1) network logs, which have been
113 sent by different hosts, and servers to aggregate them in a single location, and (2) alerts,
114 which have been triggered by specific software that employ various real-time traffic
115 monitoring algorithms. Those software can be signature based (e.g., Intrusion Detection
116 Systems (IDS), Asset Detection, etc.) or anomaly based (e.g., Network Behaviour
117 Analysis (NBA), Flow-based Classifier, etc.).
118 • **Intelligence Layer**. The intelligence layer syntactically normalizes and stores the events
119 for further analysis. The events are then compared to existing described scenarios that
120 have been previously stored in the knowledge base (KB) and processed by statistical
121 models, geared at recognizing anomalies. In case of positive matching, correlated events
122 are sent to the analysis layer.
123 • **Analysis Layer**. Security analysts can observe the filtered data through the analysis
124 layer. The analysts generally have tools for grouping data by specific network-protocol

125   variables (e.g., IP, port, etc.) and they seek specific information in order to demonstrate
126   the existence of an attack.

127   • **Response Layer.** In the response layer, the analysts contact the clients by writing reports
128   or sending emails if they are able to present evidence of an attack, or to ask for further
129   information.

130   • **Knowledge base (KB).** The knowledge base stores the pre-defined attack patterns. The
131   intelligence layer checks the real-time data against the pre-defined attack scenarios in the
132   KB in order to provide meta-events to security analysts.

133

134 *Motivation*. Network security monitoring has become an exceedingly essential service, and
135 companies rely on the expertise of reputable Managed Security Service Providers (MSSPs) for
136 guidance and best-practice recommendations in order to detect and prevent malicious activities.
137 Monitoring is conducted by a team of security analysts who evaluate each problem almost
138 manually, while basing their work on their expertise and previous knowledge about IT security
139 requirements. Whether clients request more events be collected or whether they need to install
140 more services to improve attack coverage, this approach will not scale well because the volume
141 of data to be analyzed increases excessively. Hence, network security companies would like to
142 overcome this problem by building pre-defined correlation rules, more or less complex, in order
143 to identify a finite number of malicious or anomalous situations.

144

145 Although this approach scales well in the analysis layer, it has shown multiple limitations for
146 maintenance and attack coverage supervision. Given the highly dynamic security context, this
147 approach needs continuous improvement of existing correlation patterns by tuning parameters
148 that should follow the evolution of the system. Moreover, new correlation patterns need to be
149 introduced in order to discover new threats. Even when using highly advanced correlation
150 patterns, many collected events result in being discarded by these positive filters and never
151 appear on the analysts' screens for review. This methodology is not necessarily wrong at first
152 glance, since many events cannot be fully analyzed if they are not placed in the right context.
153 However, discarding them cannot be the best solution either. Indeed, without rapidly creating
154 new correlated patterns, even critical events that were collected after being triggered by the latest
155 installed advanced security applicative might be discarded because they are not part of any
156 correlation pattern. The idea of this paper is to automatically map all collected events by using a
157 published structure existing in literature that collects all attack patterns. (Mitre, 2015)

158

159 *Example*. Suppose a security analyst needs to investigate the following IDS alert:

160   • `IDS alert tcp $EXTERNAL_NET any -> 192.168.20.13:443 (msg:"FILE-OTHER`
161   `XML exponential entity expansion attack attempt"; ... sid:29800;`
162   `rev:1;)`

163 Using our presented methodology, the IDS alert results in a link to the following CAPEC attack
164 pattern, which in turn is linked with other events that might have been collected in the sensor [2]:

165 `CAPEC-197: XML Entity Expansion`

166 `Description: An attacker submits an XML document to a target application`
167 `where the XML document uses nested entity expansion to produce an excessively`
168 `large output XML. XML allows the definition of macro-like structures that can`
169 `be used to simplify the creation of complex structures. However, this`

```
170  capability can be abused to create excessive demands on a processor's CPU and
171  memory. A small number of nested expansions can result in an exponential
172  growth in demands on memory.
```

```
173  Outcomes: The attacker causes the target application denial of service.
```

174 This attack pattern, defined by CAPEC, can be related to the following event instances:

```
175      a. End Point Solution Alert - High Usage of CPU on Host:$HOME_NET
176      b. Network Behaviour Analysis Alert - Possible Denial of Service in the
177         Network $NET_NAME
178      c. IDS alert tcp $EXTERNAL_NET any -> $HOME_NET 80 (msg:"SERVER APACHE
179         Apache mod_cache denial of service attempt"; ...; sid:12591; rev:7;)
```

180 Thus, through the use of our proposed methodology, the security analyst will know that the IDS
181 alert, identifiable by the `sid=29800`, can be associated to the aforementioned event instances,
182 which are suggested to be checked by the analyst. This means that the concurrent arrival of those
183 events, mapped to a known attack pattern, can provide strong evidence of the attack under
184 investigation. ∎

185
186 **3. Related work**

187 This section first discusses the state-of-the-art techniques in the area of security-related events
188 correlation, followed by a discussion on some related works that concern applying text mining
189 methods in the context of network security and use of recommender system.

190 *Event Correlation Technique in Network Security*

191 ***Rule-Based Reasoning (RBR)***. A common approach to the problem of event correlation is to
192 represent knowledge and expertise in a rule-based reasoning system (Lewis, 1999). RBR systems
193 are often used in artificial intelligence applications such as the expert system, production system,
194 or blackboard system. The system basically makes the inference between the collected data and
195 the Knowledge Level, by a control layer that verify of list of rules or a rule repository
196 representing domain-specific expert knowledge. The knowledge in a RBR system can be updated
197 without changing the program code of the inference engine, but it has several associated
198 weaknesses. First, it is time consuming to manually enter the knowledge into the knowledge
199 base. Second, the rule repository requires constant maintenance, which actually undermines the
200 original purpose of a correlation engine to reduce the workload of administrators. The method
201 proposed in this paper does not need human intervention to create new rules that represent attack
202 patterns. The system will automatically map events to existing description of single phases of the
203 attack. Moreover, the mapping will be automatically updated when a new event is collected or a
204 new attack pattern is created.

205 ***Case-Based Reasoning (CBR).*** The general idea of this approach is to solve new problems based
206 on the solutions of similar past problems. CBR is a well-known research field in artificial
207 intelligence that involves the investigation of theoretical foundations, system development, and
208 practical application in the building of experience-based problem solving (Bergmann, 2003)
209 (Hullermeier, 2007). CBR allows users to reuse knowledge from past cases in order to reduce
210 acquisition efforts, which in turn increases the user's confidence in the system. However, manual

211 knowledge engineering is required for adapting rules, the process is difficult to automate. Thus,
212 it is still a time-consuming task. In contrast, the method proposed in this paper does not need
213 human intervention to update existing attack patterns because the system will constantly update
214 the mapping when a new event/attack pattern is created or modified. Moreover, by referring to
215 external publicly available sources, the presented methodology can also map events to scenarios
216 that have never been recorded in the monitored network.

217 ***Graph-Based Approach (GBA)***. Boris Gruschke (Gruschke, 1998) presented a working principle
218 for event correlation with a dependency graph, a directed graph that models dependencies
219 between the managed objects. Its nodes (objects) reflect the managed objects of the system. Its
220 edges reflect the functional dependencies between managed objects. Although it can manage a
221 complex system and handle dynamic dependencies, the GBA is only capable of handling one
222 problem at a time, i.e., GBA can identify only one root-cause. Identification of all root-causes
223 may not be possible if multiple problems occur within the time period. Moreover, to be managed
224 requires a high level of expertise from the analysts. Due to these limitations, GBA is hardly
225 useable in practical contexts.

226 *Text Mining in Security-Related Application*

227 Text mining has been applied to network security, but its research problems differ from the ones
228 presented in this paper. By using text categorisation it is possible to distinguish and learn the
229 characteristics of normal and malicious behaviour in regards to the words used in specific
230 contexts. In (García Adevaa & Pikatza Atxab, 2007), for example, the methodology is applied to
231 log entries generated by web application servers in order to detect abnormal behaviour or fraud
232 (Weston & Hand, 2008). In (De Vel, Anderson, & Corney, December 2001),an investigation into
233 e-mail content was performed for the purpose of forensic investigation. With such methodology,
234 it is not only possible to recognize the topic discussed in the e-mail, but also abnormal
235 conversation that might be related to targeted keywords. Similar goals have been studied in order
236 to recognize maliciousness in social networks involving a large-scale data stream (Keim, Krstaji,
237 & Bak, 2010). In (Thompson, 2005), through text mining it was possible to link entities across
238 documents in order to recognize when different name strings are potential references to the same
239 entity. In (Gegick, Rotella, & Xie, 2010), a system was implemented with an approach that
240 applied text mining on natural-language descriptions of bug reports (BRs) in order to train a
241 statistical model on already manually-labeled BRs to identify security bug reports (SBRs) that
242 were manually mislabelled as not-security bug reports (NSBRs). In contrast, the research
243 problem studied in this paper is to dynamically link a collection of unrelated security events to
244 abstraction identifiable in the attack pattern(s) by using text mining techniques that can deal with
245 unstructured text.

246 *Recommender System in Security-Related Application*

247 Recommender systems have been an important research field in connection with information
248 retrieval and information filtering systems for two decades. These systems are commonly applied
249 in the industry of electronic commerce to facilitate customers in making decision on online
250 purchases. Recommender systems are also widely applied to non-commercial applications for
251 predicting and recommending items or actions based on user preferences and rating. There are
252 few examples in literature which use recommender systems to help IT-professionals to deal with
253 security and performance issues. For example, (Sampaio, 2014) developed a recommender

254 system to solve performance issues occurring to network applications. The process proceeds with
255 the identification of performance-related problems in networks and the characterization of user-
256 profiles. Based on specific attributes which are used to characterize the sample cases, the work
257 introduces the calculus of distance between a new case and stored ones. Another example is
258 given by (Lyons, 2014) which elaborated a hybrid recommender system to generate list of cyber
259 defense actions to mitigate cyber-security attacks. (Lyons, 2014) has a similar goal of this paper
260 but it is specifically designed for defender counteractions and it takes into the consideration of
261 only one attack-scenario. The recommended actions are based on historical data and are
262 elaborated by a collaborative filtering approach.

263

264 ## 4. Problem description

265 In this section we first formally define the notions of security event and attack pattern followed
266 by a description of the research problem. A *security event* contains a set of attributes and a text
267 description. For example, an attribute can be an IP source, an IP destination, or a port source, etc.
268 (Yurttas). A text description can be "*MS-SQL raise error possible buffer overflow*".

269 **Definition 1. (security event):** Let $E(A_1, ..., A_m, D)$ be a collection of security events, where A_i is
270 an security event attribute and D is set of distinct words in all text descriptions in E. A security
271 event $e \in E$ contains $\{a_1, ..., a_m, d\}$, where $a_i \in A_i$ and $d \subseteq D$. While d represents the description
272 of the event, the other a_i refers to protocol-related attributes.

273 An attack pattern in CAPEC, the publicly available comprehensive dictionary and classification
274 taxonomy of known attack patterns, contains a set of attack steps, where each step is a text
275 description that describes the details of the attack step, consequence, attack prerequisites, etc.

276 **Definition 2. (attack pattern):** Let P be the set of attack patterns. Each attack pattern $p \in P$
277 contains a set of attack steps, denoted by $p = \{s_1, ..., s_u\}$, where each attack step s_j contains a set
278 of words.

279 To provide a concise description, in the rest of the paper, we assume that the text description of
280 the security events in E and the text description of the attack steps in P share the same set of
281 vocabularies D. Alternatively, the method may define a map to translate between the two sets of
282 vocabularies. The following distance function measures the distance between an attack event and
283 and an attack step.

284 **Definition 3. (distance function):** Let $dist(e,s)$ be a distance function that measures the
285 distance between an attack event e and an attack step s.

286 The attack event e and the attack step s are represented as two sets of words. The distance
287 function, which will be further instantiated in Section 5.4, calculates the distance between two
288 sets of words.

289 The research problem is to identify the topmost similar security events for each attack step in the
290 attack patterns defined in CAPEC. The similarity is defined as the inverse of the distance
291 function, $sim(e,s) = 1/dist(e,s)$. In short, if two sets of words have more words in common, the

292 smaller the distance. If the sets do not share words, the distance is the maximum. Definition 4
293 provides a formal description of the research problem.

294 **Definition 4. (attack patterns and security events matching problem):** Given a collection of
295 security events E and a set of attack patterns P, the research problem is to identify the top-K
296 similar security events $\{\hat{e}_1,...,\hat{e}_K\} \subseteq E$ for each attack step s of each attack pattern $p \in P$, such
297 that for $\forall \hat{e}_x \in \{\hat{e}_1,...,\hat{e}_K\}$ $\forall e_y \notin \{\hat{e}_1,...,\hat{e}_K\}$, $dist(\hat{e}_x,p.s) \le dist(e_y,p.s)$, where K is a user-
298 defined positive integer threshold.

299 An attack step is an abstract description in CAPEC. The top-K events $\{\hat{e}_1,...,\hat{e}_K\}$ provide a more
300 concrete description of the security information in the context of a security event. By observing
301 the top-K security events of each attack step in a given attack pattern, a security analyst can
302 identify and describe the evolution of an attack through the evidence of collected events.

303

304 ### 5. Proposed solution

305 Figure 2 depicts an overview of the proposed solution, followed by a detailed description of each
306 component.

307 As shown in Figure 2, through an off-line process the set that includes all the description $d \in E$
308 goes through pre-processing steps that sanitize sensitive data before it is stemmed. The stemming
309 process is achieved by using optionally the Porter stemming algorithm (Porter, 1980), the
310 Lancaster stemming algorithm (Paice, 1990), or the Regex-driven algorithm (Qiang Yu, 2015),
311 as described in detail later on. From the stemmed data, specific lists of stop words are filtered out
312 so it can go through the tokenization process.

313 Once tokenized, from the description field a vector v is extracted that maps the content of d in a
314 dictionary D. To enforce the mapping and give more relevance to specific security-related
315 expressions, the maximal frequent sequences are also extracted, in accordance with (Ahonen-
316 Myka, 1999), and considered as a single term of the dictionary. As representation of the vector v,
317 the framework allows the user to choose among a binary, TF, or TFIDF representation.

318

319

320 **Figure 2: Pipeline diagram illustrating the overall process of the proposed data mining system**

321 Each of these can be set up in the configurable framework. As seen from Figure 2, each s_j
322 characterizing $p = \{s_1, ..., s_u\} \in P$ goes through the same process, which transforms s_j in a
323 specified vector w. Once the vectors are built it is necessary to set up the distance function (e,s) ,
324 which can be either the Asymmetric Distance (Similarity of asymmetric binary attributes) or the
325 Cosine Similarity function (G. Salton, 1986).

326 Both functions interpret valuable alternatives to represent the $dist(v,w)$ that is going to be used
327 in the 'Document Mapping Step' to find the top-K similar security events that satisfy the

328 problem statement demarcated in Definition 4. The algorithm that has been chosen to achieve
329 this result is the k-nearest-neighbour classification (Cover & Hart, 1967).

5.1 *Preprocessing*

331 The following preprocessing procedures are applied on the document $d \subseteq D$.

5.1.1 General Sanitizing

333 In each document, the terms, containing symbols or number, are removed. This step, for
334 example, removes terms that represent byte signatures, such as '*50 4B 03 04'*, or binary flag,
335 such as 0x0014.

5.1.2 Stemming

337 In each document, words that have morphological forms are normalized to their canonical form.
338 Three algorithms were analyzed and can be set up in the code:

339 • The Porter stemming algorithm (Porter, 1980), which removes the commoner morphological
340   and inflexional endings from words in English for this purpose. (Porter, 1980)
341 • The Lancaster stemming algorithm (Paice, 1990), an aggressive stemming algorithm that
342   reduces English words to their roots. For example, the word 'maximal' becomes 'maxim',
343   and the word 'cement' becomes 'cem'.
344 • A Regex-driven stemming algorithm (Qiang Yu, 2015) that uses regular expressions to
345   identify morphological affixes. Any substrings that match the regular expressions will be
346   removed.

5.1.3 Stopwords Removal

348 Common English words, such as "a" and "the", are removed. We compiled an additional static
349 list of stopwords that are common in the security contexts, e.g., "security","attacker", to better
350 index the documents.

5.1.4 Duplicates Stripper

352 Identical documents are removed. This can happen because different revisions of the same event
353 might exist in the system.

5.1.5 Tokenization

355 Given a sequence of terms in a document, the tokenization process separates the terms into
356 tokens by using white spaces and punctuation marks as separators. This operation transforms the
357 sequence of terms d in vector of terms $\vec{d}$.

5.2 *Mining Maximal Frequent Sequences*

359 In the documents, it is possible to find interesting sequences, e.g., 'denial servic', 'stack overflow
360 attempt', login attempt', 'download attempt', which represent well the threats described in the
361 event. In order to give more weight to these terms in the indexing phase, a maximal frequent
362 sequence mining algorithm has been implemented. The algorithm chosen for this purpose has

363   been coded following the Apriori-like method described in (Ahonen-Myka, 1999), and the set of
364   sequences $MFS = \{q_1, .., q_{|MFS|}\}$ has been determined. In the implementation, we employed an
365   Apriori-like method. However, the main contribution of this paper is to propose a methodology
366   to tackle the problem described in the paper. The one may replace specific algorithms, e.g., the
367   maximal frequent sequence mining method, by other more efficient algorithms based on their
368   preferences and needs. The result will be the same.

369   **5.3 *Document Representation***

370   After the preprocessing, the set of distinct and ordered terms in a given document set is denoted
371   by $Dict = \{w_1, .., w_{N_d}\}$. Next, all documents are indexed by their terms. Three representations
372   have been implemented and they are described in the next sections.

373   *5.3.1 Binary Representation*

374   For each vector $\vec{d}$, we build a binary vector of $|Dict|$ elements denominated $\vec{b} = [v_1, .., v_{|Dict|}]$
375   where

376
$$v_j = \begin{cases} 1 \text{ if } w_j \in d \\ 0 \text{ otherwise} \end{cases}$$

377

378   *5.3.2 TF Representation*

379   For each vector $\vec{d}$, we build a numerical vector of $|Dict|$ elements denominated
380    $\vec{t} = [v_1, .., v_{|Dict|}]$ (1) such as

381
$$v_j = count(w_j, d)$$

382   where count(x,y) is a function that counts the occurrences of the term x in the set y. A dual
383   representation related to TF is called **Inverse Document Frequency** and is calculated as
384   follows:

385
$$v_j = count(w_j, d)/|Dict|$$

386   Generally this representation is preferred when it is important to diminish the weight of terms
387   that occur very frequently in the document set and increases the weight of terms that occur rarely
388   (Luhn, 1960).

389

390   *5.3.3 TF-IDF Representation*

391   For each vector $\vec{d}$, we build a numerical vector of $|Dict|$ elements denominated
392    $\vec{g} = [v_1, .., v_{|Dict|}]$ (2) such as

393

$$v_j = \frac{count(w_j, d)}{|Dict|} \cdot \log\left(\frac{|D|}{count(w_j, D)}\right)$$

394

395 where $count(w_j, D)$ counts the number of elements of D that contain the term w_j. This
396 representation, proposed in (Spärck Jones, 1972), gives a higher weight to a high term frequency
397 (in the given document) and a lower weight to document frequency of the term in the whole
398 collection of documents; the weights hence tend to filter out common terms (Term frequency–
399 Inverse document frequency).

400 The representation chosen for the first experimental results is the binary representation, but all of
401 them have been coded and can be set up by the user.

402 After implementing such models, it was noticed that it was necessary to enforce the
403 representations of the documents by also taking into consideration the terms that often come in
404 sequences. Thus, in order to give more weight to those cases, the representation, aforementioned,
405 has been improved by counting the maximal frequent sequences as additional single terms of the
406 dictionary. By doing so, the vector $\vec{b} = [v_1, .., v_{|Dict|}]$ will have extra $|MFS|$ number of elements,
407 resulting in:

$$\vec{b}^* = [v_1, .., v_{|Dict|}, v_{|Dict|+1}, ..., v_{|Dict|+|MFS|}]$$

408   (3)

409 and consequently, if binary representation is used,

$$v_j = \begin{cases} 1 \ if \begin{cases} j \le |Dict| \ and \ w_j \in d \\ j > |Dict| \ and \ q_{j-|Dict|} \in d \end{cases} \\ 0 \ otherwise \end{cases}$$

410

411 where $MFS = \{q_1, .., q_{|MFS|}\}$ represents the ordered set of maximal frequent sequences.

412 *5.3.3 TF-IDF Representation*

413 For each vector $\vec{d}$, we build a numerical vector of |Dict| elements denominated

414 $\vec{g} = [v_1, .., v_{|Dict|}]$ (2) and $\vec{b}^* = [v_1, .., v_{|Dict|}, v_{|Dict|+1}, ..., v_{|Dict|+|MFS|}]$ (3).

415

416 **5.4 Documents Similarity Function**

417 Having determined the way to represent terms and documents, we utilized the asymmetric
418 distance and cosine similarity measure to measure the similarity between the vectors.

419 *5.4.1 Asymmetric Distance*

420 The simultaneous absence of many terms in the vectors suggests the most relevant results to be
421 the discovery of having something in common, rather than sharing the lack of specific terms: the
422 agreement of two 1's (a present-present match or a positive match) is more significant than the
423 agreement of two 0's (an absent-absent match or a negative match). The asymmetric distance has
424 been calculated in according to (Similarity of asymmetric binary attributes) and its value spans
425 between 0 and 1.

426 *5.4.2 Cosine Similarity Measure*

427 The present distance has been preferred to other known distances because the cosine similarity
428 measure abstracts the length of the documents and focuses on the correlation between their
429 vectors. Since all axis coordinates are positive, the cosine similarity value between any two term
430 vectors is always between zero and 1:

$$0 \leq \vartheta(\vec{g}_i, \vec{g}_j) \leq \pi/2 \rightarrow \cos \vartheta \in [0,1]$$

432 where ϑ is the angle between $\vec{g}_i$ and $\vec{g}_j$. (Salton & McGil, 1986)

433 Since the binary representation was chosen for the experimental results, as a consequence the
434 asymmetric distance was selected. However, both have been coded and can be set up by the user,
435 and, in case of a different representation, the cosine similarity distance is automatically selected.

436

437

438 **5.5 Semantic Relatedness**

439 After choosing the representation model and the documents similarity function, we employed a
440 machine learning algorithm, K-nearest-neighbour (Cover & Hart, 1967), to determine the closest
441 events of a specific attack step. The algorithm is usually used for determining the class of a
442 specific unclassified element by evaluating the classes of the nearest elements. In our case, this is
443 not necessary, and the method is just used to pick the top-K related events $\{\hat{e}_1, ..., \hat{e}_K\}$ of a specific
444 attack step s_i. However, in such a system it can be verified that some attack steps do not share
445 enough terms with any of the events being captured in the sensor. This can happen if there is no
446 tool in the sensor that can trigger an event that describes an action defined in a specific attack
447 field. The result of that would be in classifying as neighbours some events that are the closest
448 compared to others, but in fact very far ($dist(e,s) \approx 1$). For this reason, a maximal tolerated
449 distance is fixed, and all the neighbours above this threshold are discarded.

450

451 **6. Discussion**

452 As mentioned above, the proposed methodology automatically maps security-related events to
453 pre-defined attack patterns. This means that security analysts have a much better understanding
454 of the events being collected from a specific network because of their classification and the
455 possibility to directly relate a raw event to a sequence of attack steps. The use of CAPEC is

456 useful to help analysts in examining data and generating responses without obligating them to
457 remember too many notions and saving time they would spend on the Internet to look for
458 confirmation and recommendations.

459 Moreover, through this methodology the analyst can clearly see the attack steps that are not
460 mapped to any collected events and the mapped events that were not received in the sensor. The
461 first case can suggest to the analysts which specific technology the company is missing in order
462 to detect that specific attack steps; the second case instead suggests the attack steps that were
463 missed by the tool installed in the sensor.

464 Given the vastness and the diversity of the collected events, we can argue that the mapping
465 process might take too much time. This is actually not relevant since the mapping process can be
466 executed off-line because the instances of the whole set of events are available. In real-time only
467 the occurrences of the mapped events are checked and shown to security analysts in conjunction
468 with the attack pattern associated with them. The system does not require specific maintenance
469 either because the discovery of new attack patterns and the creation of new collectable events
470 have a slow evolution, which could be dealt with by running the algorithm once a day on the new
471 data only.

472 By relating events to the closest attack step, it is possible that a specific event is not mapped to
473 any attack step. In that case we suggest the following:

474 • Evaluate the criticality of the event, and its worthiness to be collected;
475 • Evaluate the usage of synonyms to describe the content of the event; and
476 • Evaluate the possibility of creating a new attack pattern in which the event could be
477 inserted.

478 The last suggestion reveals that the presented methodology does not need to be applied on
479 CAPEC documents only, but it can be extended to any kind of document that aims to describe a
480 Security related topic.

481

482 **7. Experimental Evaluation**

483 The objective of the experiments is to illustrate the precision of the algorithm, the capability of
484 generating new correlation patterns in an automatic fashion, and the diversification of the attacks
485 being reconstructed through alerts. All experiments were conducted on a PC running Windows 7
486 on an Intel(R) Core (TM) i3 CPU with 4 GB RAM.

487 *7.1 Data description*. For the following experiments, the events are represented by Snort Alerts;
488 in particular, we have selected the top 1,000 most frequently analyzed alert instances in a MSS,
489 extracted from (Sourcefire, 2015). To represent attack patterns, we have used the CAPEC List
490 Version 2.6 (http://capec.mitre.org/data/#downloads), including 450 different attack patterns
491 with 2,917 fields (n) that describe the attack steps and related information.

492 Following the framework depicted in Figure 2 we employed the regex-driven algorithm as the
493 stemming algorithm, binary vectors as data representations, and asymmetric distance as the
494 similarity function.

495 **7.2 Precision**. The system takes two user-specified thresholds k and $dist_{MAX}$ as inputs, where k is
496 the number of neighbours to be defined in the K-NN algorithm, and $dist_{MAX}$ is the maximum
497 tolerated distance between events and attack steps.

498 Figure 3 depicts the total number of neighbours being found (a.k.a. close relationships between
499 Snort Alerts and attack steps) by the K-NN algorithm with respect to $dist_{MAX}$, having fixed
500 $k = 10$. We set $dist_{MAX}$ to a range of values between 0.55 and 0.8. We observe that the number of
501 neighbours spans from 3 to 1,465 when the Maximum Tolerated Distance ($dist_{MAX}$) increases
502 from 0.55 to 0.80. However, comparing that assignment to the ground truth, the precision of the
503 algorithm decreases with the increasing of $dist_{MAX}$. In fact, Figure 4 shows that the percentage of
504 error (e) spans from 0% to 27%. We can argue that the best result consists of having a high
505 number of neighbours with very limited number of errors, in order to not mislead the analysts by
506 associating security-related events to wrong attack steps. Having analyzed Figure 4, we can
507 assume that the best results are observed when $dist_{MAX}$ =0.70. The local minimum of the Error
508 function, for $dist_{MAX}$ =0.70, is determined by the fact that, within the range $dist_{MAX}$
509 $\in [0.55 - 0.70]$, the number of wrong relationships is caused by few misleading words and it is
510 much lower than the number of total relationships established by the algorithm. By increasing
511 the $dist_{MAX}$, the number of total neighbours increases sensibly, although limited on the upper part
512 by $n * k$, but the error function assumes values that outdo the undesirable threshold of 25%; so
513 we can assume that $dist_{MAX}$ =0.70 represents the best setting in order to have good precision and
514 a high number of neighbours.

515

516

517 **Figure 3: Number of Neighbours (N) to Maximum Tolerated Distance ($dist_{MAX}$), and difference between**
518 **correct and wrong relationships.**

519

520

521 **Figure 4: Percentage of wrong relationships to Maximum Tolerated Distance ($dist_{MAX}$)**

522

523 **7.3 Diversification of Neighbours**. Having fixed $dist_{MAX}$ =0.70, which is the value that assures
524 the best results according to Figure 4, Figure 5 depicts the total number of events being related to
525 every single attack pattern, represented by a CAPEC ID. The result suggests that the proposed
526 method dynamically defines multiple correlation patterns, each containing a certain number of
527 alerts. As shown in Figure 5, 53 attack patterns have been related to a number of different Snort
528 alerts, which span from 1 to 27 attributions, based on the similarity between attack step fields
529 and the events. Figure 6 depicts the distribution of the attack patterns according to the number of
530 events related to them. We can argue that the relationship between an attack pattern and a single
531 Snort alert (first bar of the graph shows in Figure 6, in correspondence of N=1) is not relevant
532 because security analysts are only interested in relating events being triggered by the sensors.
533 However, those relationships help analysts in retrieving useful information about the totality of
534 the attack, e.g., missing steps or related information about the attack as prerequisite, mitigations,
535 etc.

536

537   **Figure 5: Number of different alerts being related to single attack patterns, represented by the CAPEC ID**

538

539   **Figure 6: Attack pattern occurrences to events being related to the attack**

540

541 *7.4 Computational Time*. The computational time was analyzed by taking in consideration,
542 incrementally, the whole set of distinct messages included in Snort Alerts (Sourcefire, 2015). For
543 the analysis of the computational time, the four different phases of the framework have been
544 clocked by varying the number of distinct Snort alert messages considered.

545 As shown in Figure 7, the time necessary to map the whole set of distinct messages, extractable
546 from Snort alerts, to the whole set of CAPEC fields, is quantifiable in about two hours and
547 twenty minutes. By considering that the system needs to be updated when the CAPEC structure
548 is modified (every three months, (Mitre, 2015)) or when new Snort rules are published (every
549 couple of day), the computational time can be considered very low, because it is below that
550 margin. It is important to note that the whole process is executed off-line: when an analyst is
551 called to interpret a specific alert in-line, the system will map it to pre-defined attack patterns by
552 using previously stored results. Thus, the analyst can avoid manually mapping the alerts and
553 efficiently obtaining the results. Figure 8 shows that the extraction of the top-K alerts, described
554 in section 5.5, it is the one which mostly contributes to the total computational time, and it must
555 be improved in a new re-factoring of the system.

556

557 **Figure 7 Computational Time calculated by varying the number of Snort Alerts Messages to be mapped to**
558 **the whole set of Capec Fields**

559

560

561 **Figure 8 Computational Time calculated by varying the number of Snort Alerts Messages to be mapped to**
562 **the whole set of Capec Fields**

563

564

565   **8. Conclusions**

566 In this paper, we have proposed and developed a data mining frameworks that employs text
567 mining techniques to dynamically relate the information between the security-related events and
568 CAPEC attack patterns, both described in unstructured text. Moreover, we introduced an
569 automatic system to gather events that can significantly reduce the analysis time and increase the
570 number of correlated and analyzed events included in an attack report prepared by security
571 analysts for the clients, improving the quality and accuracy of the reports. For future work, it
572 would be interesting to investigate the impact of including other standards in addition to CAPEC,

573   e.g., *Extensible Configuration Checklist Description Format* (*XCCDF*) (XCCDF - The
574   Extensible Configuration Checklist Description Format). Also, it would be interesting to study
575   the clusters of words that characterize the events that are not currently classified in order to
576   understand the missing attack patterns or attack steps in CAPEC.

577

578

579

580

References

Ahonen-Myka, H. (1999). Finding All Maximal Frequent Sequences in Text. *Proc. of the ICML99 Workshop on Machine Learning in Text Data Analysis* , 11-17.

B. Thuraisingham, L. K. (2008). Data Mining for Security Applications. *IEEE/IFIP International Conference on Embedded and Ubiquitous Computing.*

Bergmann, R. (2003). Developing Industrial Case-Based Resoning Applications: The INRECA Methodology. *Springer Science & Business Media , 1612*.

Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on , 13* (1), 21 - 27.

De Vel, O., Anderson, A., & Corney, M. (December 2001). Mining e-mail content for author identification forensics. *ACM SIGMOD Record , 30* (4), 55 - 64.

G. Salton, M. M. (1986). *Introduction to Modern Information Retrieval.* New York, NY, USA: McGraw-Hill, Inc.

García Adevaa, J. J., & Pikatza Atxab, J. M. (2007). Intrusion detection in web applications using text mining. *Engineering Applications of Artificial Intelligence , 20* (4), 555–566.

Gegick, M., Rotella, P., & Xie, T. (2010). Identifying security bug reports via text mining: An industrial case study. *Mining Software Repositories (MSR), 2010 7th IEEE Working Conference on* , 11 - 20.

Gruschke, B. (1998). Integrated event management: Event correlation using dependency graphs. *Proc. of the 9th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management* .

Hullermeier, E. (2007). Case-Based Approximate Reasoning. *Theory and Decision Library. Springer , 44*.

Keim, D. A., Krstaji, M., & Bak, P. (2010). Methods for Interactive Exploration of Large-Scale News Streams. In *Web Intelligence and Security: Advances in Data and Text Mining Techniques for Detecting and Preventing Terrorist Activities on the Web.*

Lewis, L. (1999). Event correlation in spectrum and other commercial products. *Technical Note* , 1–8.

Luhn, H. P. (1960). Keyword-in-context index for technical literature. *American Documentation , 11(4)*, 288–295.

Lyons, K. B. (2014). *Thesis: A Recommender System in the Cyber Defense Domain.* Ohio: Air Force Institute of Technology.

Mitre. (n.d.). Retrieved from https://cve.mitre.org

610   *Mitre*. (2015). Retrieved from CAPEC from Mitre, a not-for-profit organization that operates research
611   and development centers sponsored by the US federal government: https://capec.mitre.org/

612   Paice, C. D. (1990). Another Stemmer. ACM SIGIR Forum 24.3.

613   Porter, M. (1980). An algorithm for suffix stripping. Program 14.3 .

614   Qiang Yu, H. H. (2015). An Efficient Exact Algorithm for the Motif Stem Search Problem over Large
615   Alphabets. *IEEE/ACM TRANSACTIONS ON COMPUTATIONAL BIOLOGY AND BIOINFORMATICS,* .

616   Salton, G., & McGil, M. (1986). *Introduction to Modern Information Retrieval.* New York: McGraw-Hill.

617   Sampaio, L.-N. .. (2014). A knowledge and collaboration-based CBR process to improve network
618   performance-related support activities. *Expert Systems with Applications - Elsevier* .

619   *Similarity of asymmetric binary attributes*. (n.d.). Retrieved from
620   http://en.wikipedia.org/wiki/Jaccard_index#Similarity_of_asymmetric_binary_attributes

621   Sourcefire. (2015). *Snort Rules*. Retrieved from https://www.snort.org/downloads

622   Spärck Jones, K. (1972). A Statistical Interpretation of Term Specificity and Its Application in Retrieval.
623   *Journal of Documentation 28* , 11–21.

624   *Term frequency–Inverse document frequency*. (n.d.). Retrieved from
625   http://en.wikipedia.org/wiki/Tf%E2%80%93idf

626   Thompson, P. (2005). Text Mining, Names and Security. *Journal of Database Management (JDM) , 16* (1),
627   6.

628   V. Chandola, E. E. (2006). Data Mining for Cyber Security. In *Data Warehousing and Data Mining*
629   *Techniques for Computer Security.* Springer.

630   Weston, & Hand, J. D. (2008). Statistical Technique for Fraud Detection, Prevention and Assessment. In
631   *Mining Massive Data Sets for Security* (pp. 257-270).

632   *XCCDF - The Extensible Configuration Checklist Description Format.* (n.d.). Retrieved from
633   http://scap.nist.gov/specifications/xccdf/

634   Yurttas, S. (n.d.). *The relational data model U/W - chapter 3.* Retrieved from
635   http://courses.cse.tamu.edu/yurttas/csce315/CN/b/db-03.pdf

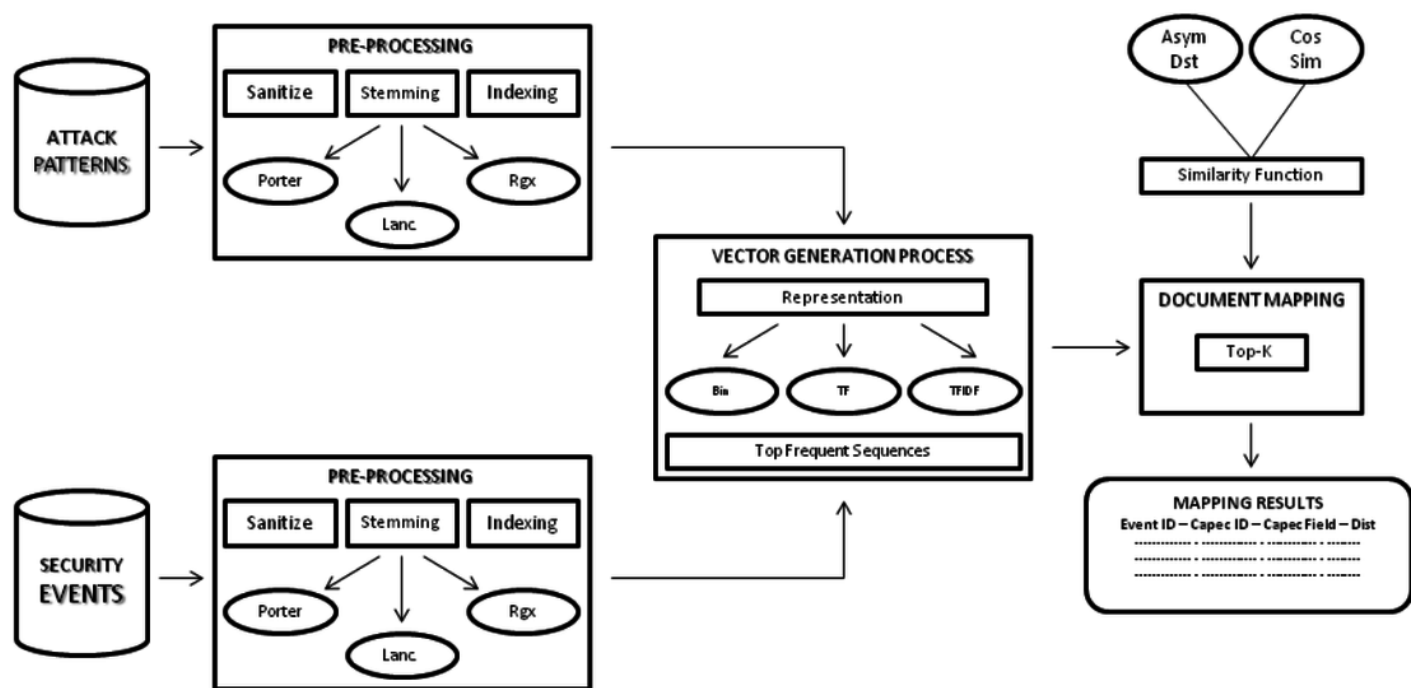
636

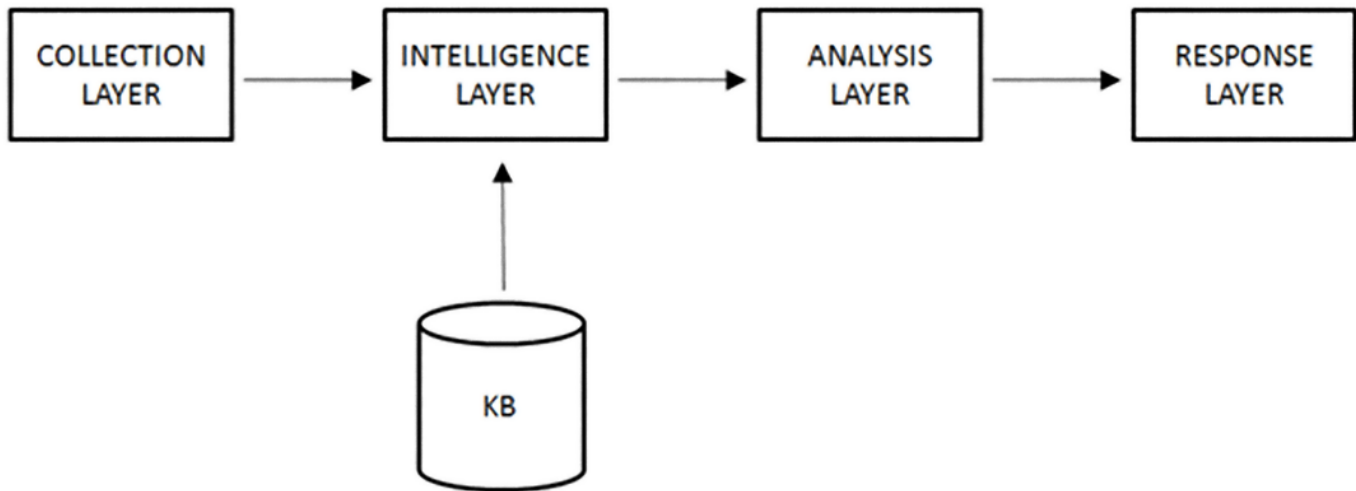
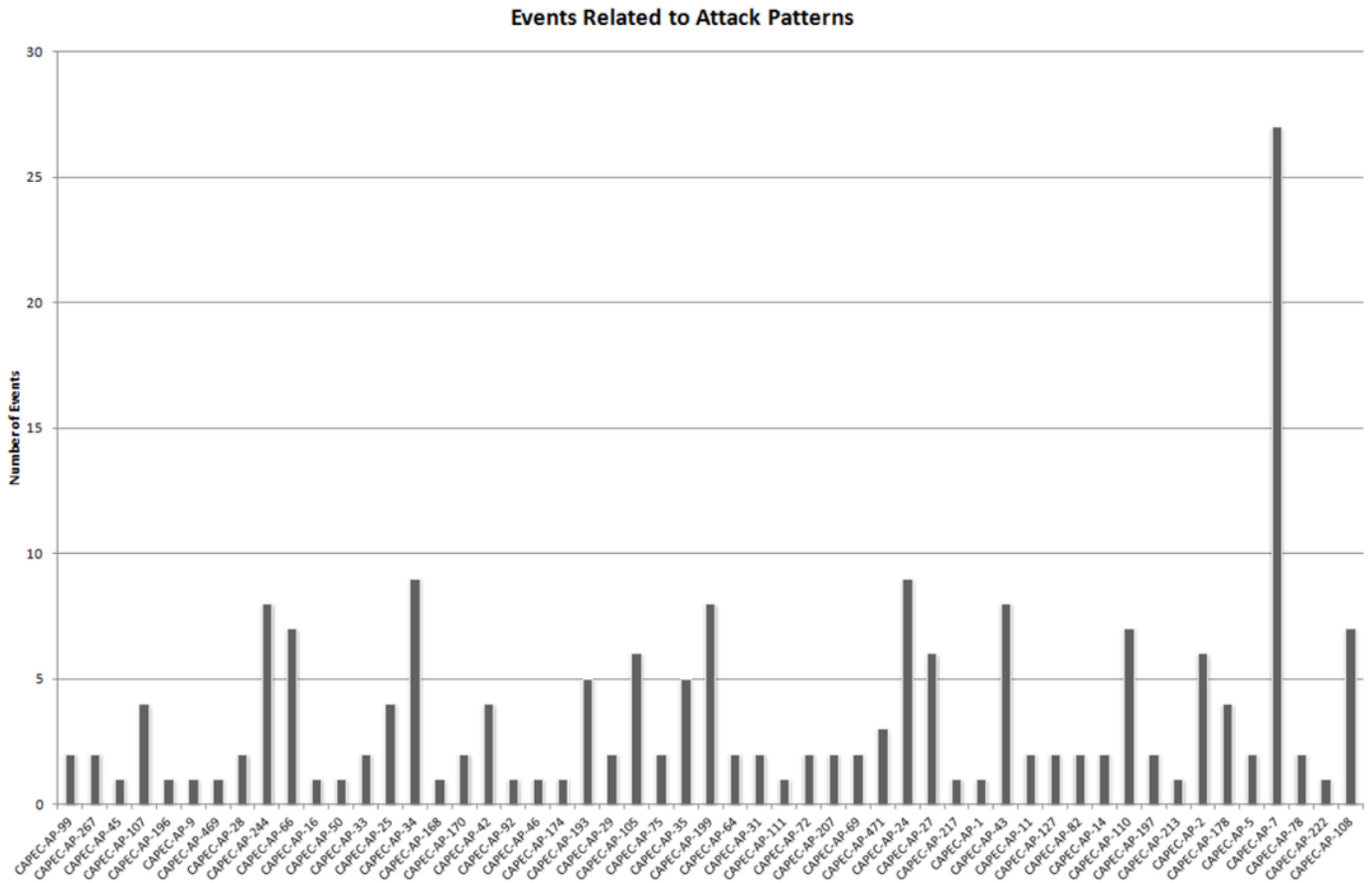
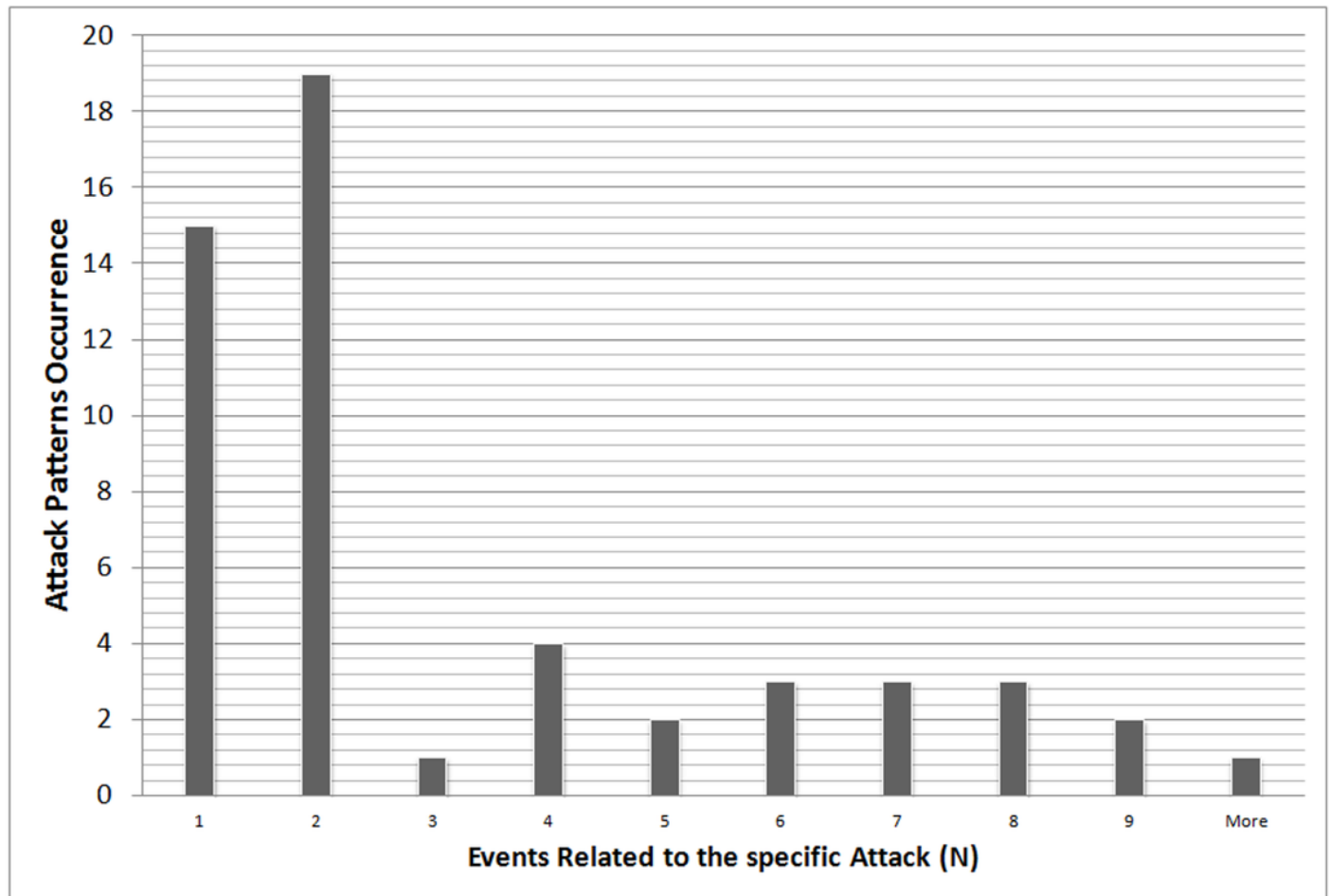
637

638

639

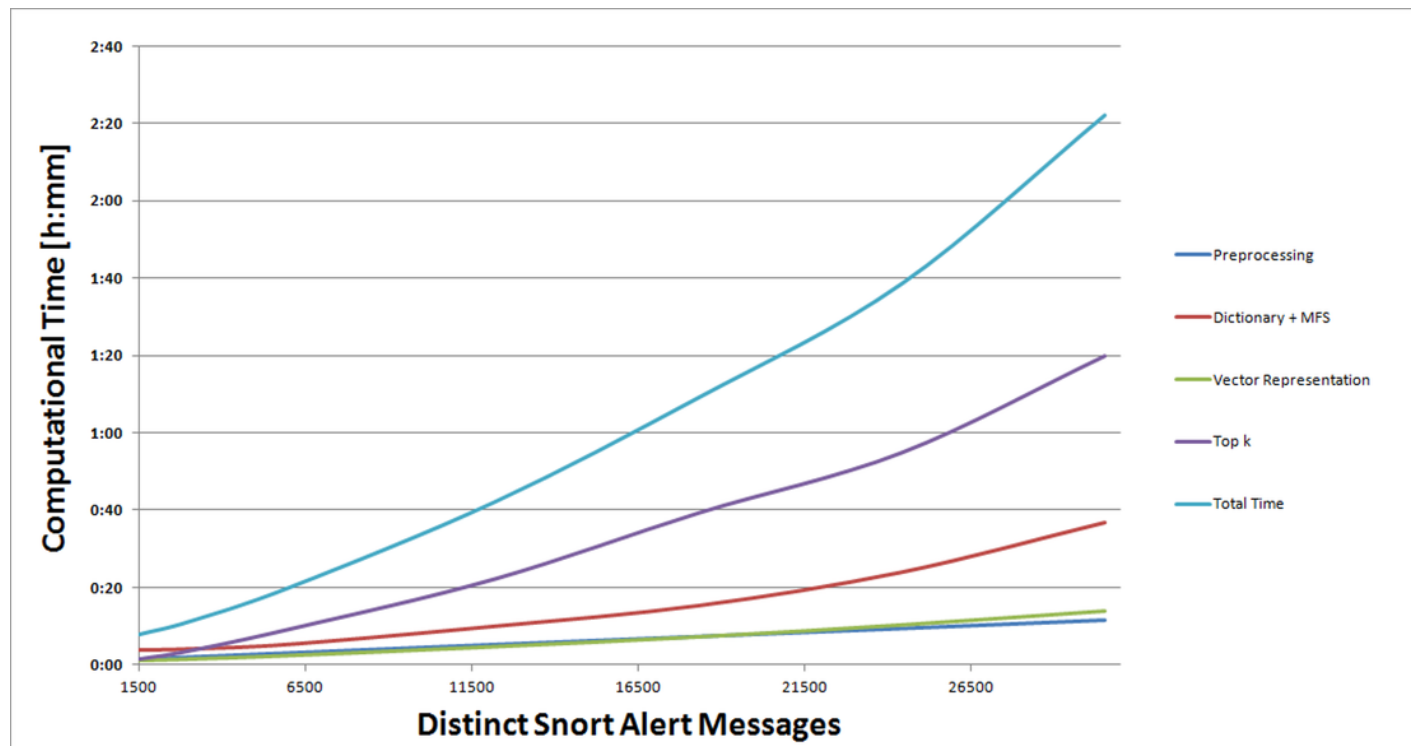
1

Pipeline diagram illustrating the overall process of the proposed data mining system

2

Workflow in a security operation center (SOC)

Number of different alerts being related to single attack patterns, represented by the CAPEC ID

Attack pattern occurrences to events being related to the attack

Computational Time calculated by varying the number of Snort Alerts Messages to be mapped to the whole set of Capec Fields

CAPEC field	2917	2917	2917	2917	2917	2917	2917
SNORT alert messages	1527	3054	6108	12216	18324	24432	30540
Preprocessing	01:33.586	02:06.753	03:11.068	05:19.088	07:22.241	09:26.917	11:35.045
Dictionary + MFS	03:48.584	04:06.285	05:21.457	09:59.710	15:19.017	23:59.598	36:51.010
Vector Representation	00:56.705	01:18.640	02:16.312	04:35.949	07:09.244	10:18.189	13:57.312
Top k	01:28.566	03:39.832	09:22.247	22:11.590	39:13.145	54:47.812	1:19:47.72
Total Time	07:47.441	11:11.510	20:11.084	42:06.337	1:09:03.65	1:38:32.52	2:22:11.09

6

Computational Time calculated by varying the number of Snort Alerts Messages to be mapped to the whole set of Capec Fields

Number of Neighbours (N) to Maximum Tolerated Distance (distMAX), and difference between correct and wrong relationships.

Percentage of wrong relationships to Maximum Tolerated Distance (distMAX)