IEEE
SYSTEMS JOURNAL
A PUBLICATION OF THE IEEE SYSTEMS COUNCIL

IEEE Xplore® Digital Library
http://ieeexplore.ieee.org/Xplore

Learn from Experts' Experience: Towards Automated Cyber Security Data Triage

Chen Zhong *Member, IEEE,*, John Yen *Fellow, IEEE,*, Peng Liu *Member, IEEE,* and Robert F. Erbacher Member, IEEE

***Abstract*—Security Operations Centers (SOCs) employ various cyber defend measures to monitor network events. Apart from these measures, SOCs also have to resort to human analysts to make sense of the collected data for incident detection and response. However, with the oncoming network data collected and accumulated at a rapid speed, analysts are usually overwhelmed by tedious and repeated data triage tasks so that they can hardly concentrate on in-depth analysis to create timely and quality incident reports. This paper aims to reduce the analysts' workloads by developing data triage automatons. We have developed a computer-aided tracing method for capturing analysts' operations while they are performing a task. This paper proposes a graph-based trace mining approach for constructing useful patterns for data triage from the operation traces. Finite state machines can be constructed based on the rules to automate data triage. A human-in-the-loop case study is conducted to evaluate our approach, in which 30 professional analysts were recruited and asked to complete a cyber analysis task with their operations being traced. State machines were constructed based on the traces and then the effectiveness of developing state machines and the performance of state machines are evaluated. The result shows that it is feasible to conduct automated data triage by leveraging analysts' traces. The state machines are able to complete processing a large amount of data within minutes. Comparing the performance of automated data triage with the ground truth, we found a satisfactory false positive rate can be achieved.**

***Index Terms*—Cyber Security Analysis, Data Triage, Security Operations Center, Knowledge Elicitation, Automated System.**

I. Introduction

THE big breaches of Target, JPMorgan Chase, Sony Pictures, and Primera Blue Cross constantly reminds us the cannot-be-underestimated risk caused by cyber attacks. Many prominent companies, government organizations and military departments have invested a lot of money to construct their cyber defense system against cyber attacks. Typically, they usually set up a Security Operations Center (SOC) to do 24*7 monitoring, intrusion detection, and diagnosis (on what is actually happening). In a military setting, CNDSP (Computer Network Defense Service Provider) centers have already been established and operated for quite a few years.

SOCs usually employ multiple automated security measures, such as traffic monitors, firewalls, vulnerability scanners, and Intrusion Detection/Prevention System (IDS/IPS). Besides, SOCs rely heavily on cyber security analysts to investigate the data from security measures to identify the true "signals" from them and "connect the dots" to answer some higher-level questions about the attack activities, for example, whether the network is under an attack; what did the attackers do; and what might be their next steps.

Although the stake of protecting an organization's mission-critical or business-critical network is high, organizations still run short of capabilities of detecting and reacting to the intrusions within their networks. It is because there is a huge gap between the overwhelming data from various security measures (e.g. IDS alerts) and the lack of analytics capabilities. Data analytics conducted by human analysts is crucial because the automated measures are in many cases unable to "comprehend" sophisticated cyber attack strategies even through advanced correlated diagnosis. Specifically, analysts need to conduct a series of analysis, including data triage, escalation analysis, correlation analysis, threat analysis, incident response and forensic analysis [1]. Data triage encompasses examining the details of a variety of data sources (such as IDS alerts and firewall logs), weeding out the false positives, grouping the related indicators so that different attack campaigns (i.e., attack plots) can be separated from each other. Data triage provides a basis for closer inspection in the following analysis to finally generate confidence-bounded attack incident reports. These incident reports will serve as the primary basis for further decision-making regarding how to change current security configuration and act against the attacks.

Data triage is a time-consuming stage in cyber analytics. Compared to a computer, human brains have orders of magnitudes smaller data processing throughput. In addition, human beings may face some challenges unique to humans such as fatigue, anxiety and depression. However, neither the network nor the attack campaign is waiting for the human brains. As the data, coming from a variety of data sources, are being continuously generated and their volume is overwhelming. Therefore, data triage, mostly performed manually by analysts, involves complicated cognitive processes [1], [2]. Since the alert portion of the data contains a large number of false positives, the analysts have to leverage their domain expertise and experience to make fast judgments regarding which parts of the data sources are worthy of further exploration and thus the time pressure is typically huge for them. The analysts have to resort from day shift to night shift transitions to achieve 24*7 coverage. According to a research report by Ponemon Institute [3], only 29% of all alerts received in a SOC have been inspected and among them an average of 40% are false positives. FireEye reported that the average annual operational spending of an organization due to false positives is $17.816 million given the default resource capacity informed through common deployments [4].

The above challenge has formed a big hurdle for analysts generating timely and high-quality incident reports in a SOC.

Digital Object Identifier 10.1109/JSYST.2018.2828832

Therefore, it has been an urgent issue for many organizations as how to ease the analysts' burden on detecting true evidence of cyber attacks and its solution is desirable to generate data triage automatons. Extensive research has been conducted on alert correlation [5], an important milestone in automated data triage. Alert correlation techniques use heuristic rules (a simple form of automaton) to correlate alerts, but usually focus on one specific data source (i.e., IDS alerts) while security analysts have to do cross-data-source analysis in most cases. Alert correlation analysis has later on been integrated with other analysis [6], but its method is still along the heuristic rule. Motivated by the benefits of cross-data-source analysis, SIEM systems (e.g., ArcSight [7]) have been focused on security event correlations across multiple data sources. Rules are usually employed to correlate the event log entries [8]. Although SIEM systems take a big leap forward in generating more powerful data triage automatons, SIEM systems is extremely expensive not only for its license cost but also for the large amount of time and expertise required in constantly system management and customization [9]. Analysts need to develop and test the data triage automatons (e.g. customized filters and complicated correlation rules) that fit the organization's setting [8], [9]. Considering that a large number of complicated data triage automatons need to be generated, the amount of manual effort is enormous. The following is an example of a complicated filtering rule used in a SIEM system [10]:

"`Filters -> EventType (In) [Failure], AND, Context (In) [Context], AND, Destination Port (In) [DestPort], AND, Protocol (In) [Protocol], AND, {Source IP (Not In) [ExcludeSrcIP1], OR, Source IP (Not In) [ExcludeSrcIP2], OR, Source IP (Not In) [ExcludeSrcIP3] },...`"

Therefore, there is a need to reduce analysts' workloads of developing the data triage automatons so that they can focus on their malware and intrusion detection tasks and generate incident reports with higher quality. This paper takes the first step to address the problem. We start with this question: can analysts' experience and expertise be elicited from their operations performed in data triage tasks? If so, can we leverage the elicited expertise to build automated rule-generation system to ease the burden on analysts in data triage?

Our approach is developed based on three insights. Firstly, it is possibly to do non-intrusive tracing of human analysts' concrete data triage operations. Secondly, though challenging, the analysts' traces of data triage operations can be mined in a largely automated way to obtain the good "ingredients" for complicated rules. The above example of SIEM rules can be decomposed into several filtering components [10]. The "`EventType (In) Failure`" is a basic component containing a data field and a constraint in its value, which is a so-called ingredient. The basic filtering components can be connected by logic connector "`AND/OR`", and thus form a complicated filtering or correlation rule. Therefore, once the good ingredients are obtained from experts' traces of former data triage operations, analysts can start form them to construct more complicated rules instead of from scratch.

We have demonstrated that state machines can be constructed from analysts' analytical traces in [11]. Based on our previous work, we further improved the state machine construction algorithm and developed a new method for combining the outputs of multiple state machines. In this paper, we start with defining data triage and analysts' operations, and then describe the refined cyber security data triage system. The main contributions are as follows:

- To the best of our knowledge, we proposed the first method that generates data triage automatons directly from human analysts' operations in an automatic manner. Relating our approach to the rule-based SIEM systems, it reduces analysts' efforts on generating SIEM rules, and therefore can make the cost (of data triage automaton generation) orders of magnitudes smaller.
- The proposed approach has been designed and implemented. The effectiveness of the automation construction has been evaluated through a human-in-the-loop case study: We have recruited professional security analysts to work on a data triage task in our previous experiment; the analysts' operation traces were collected and further be used to construct data triage state machines (DT-SMs).
- The performance of the automated data triage system has been evaluated with a much larger network dataset from the perspective of false positive and false negative by comparing with the ground truth. A new method is proposed to combine the outputs of multiple state machines to make a final decision for the triage. Furthermore, we investigated how the analysts' task performance can affect the automated data triage performance.

II. Related Work

Researchers have recognized that human analysts play a significant role in SOCs to achieve cyber situational awareness (cyber SA). SA was defined as a dynamic process involving perception, comprehension and projection [12]. Draw on the cognitive theory, the data triage analysis, as the first step of cyber SA, is essentially a complex analytical reasoning process. Cyber security analysts rely heavily on their domain and experiential knowledge to make quick decisions on suspicious network events. To represent such knowledge, logic rules have been used to capture cyber security experts' knowledge [13]. A structure-based representation (named AOH-tree) was proposed to better model the context of data triage analysis [14]. A graph representation is used in this paper to model the temporal and logical relationships between analysts' data triage operations.

Considering the challenges faced by analysts, various technologies have been developed to facilitate data analysis. An empirical study was conducted to evaluated the utility of visualization tools for enhancing analysts' cyber situation awareness based on the experience and insight of the real-world analysts [15]. With the same focus on the cognitive process of cyber security analysts, several cognitive task analysis studies have been conducted to investigate how the analysts transform the raw network data into cyber situational awareness in a SOC [1], [16], [2]. According to these empirical studies, the analysts' experience and domain knowledge plays an important role in accomplishing a task. Zhong et al. proposed an method for collecting analysts' experience cases [17], and developed a retrieval method for making suggestions to analysts based on the collected experience cases of expert analysts [14]. This paper is closely related to the previous work because we directly adopt the previous method for capturing the experience cases (i.e., operation traces), instead of reinventing the wheel. However, this work takes a big step forward by constructing semantic models based on the operation traces and eliciting useful information to automate the triage analysis. This paper extends a previous work [11] by improving the design of the DT-SM system and conducting more systematic evaluation of the performance of the DT-SMs.

Prior works on alert correlation and SIEM are complimentary to but distinct from our work. Regarding the works on alert correlation, a lot of methods have been proposed involved a wide range of technologies. Sadoddin and Ghorbani [5], in their extensive survey, classified the state-of-the-art alert correlation techniques, each of which has pros and cons. They also pointed out that knowledge acquisition is critical for most alert correlation methods. For example, the prerequisites and consequences of alerts need to be specified for rule-based correlation (e.g., [18], [19], [20]). Attack scenarios models (e.g., temporal logic formalism [21]) should be defined for scenario-based correlation. The focus of our work is the automatic knowledge elicitation, with the goal of eliciting the attack path pattern from analysts' operations in their previous analysis processes. The data triage state machine has three merits: (1) It can ease analysts' burden by applying the data constraints they used previously to triage new data so that analysts can exert more effort on detailed investigation or on coping with new challenges, and thus in turn improving the overall quality of incident reports in an SOC. (2) It is by nature enterprise specific and network specific. A DT-SM constructed from an analyst's operation trace could be directly used by another analysts. (3) The

TABLE I
NOMENCLATURE TABLE

Symbol	Description
$\mathbb{C}$	characteristic constraint
$O = (t, f_{\mathbb{C}})$	data triage operation captured at time t and with characteristic constraint $\mathbb{C}$
$isEql(O_1, O_2)$	O_1 is equal to O_2
$isSub(O_1, O_2)$	O_1 is subsumed by O_2
$isCom(O_1, O_2)$	O_1 is complementary with O_2
$>_t$	happen-after relationship
$\mathcal{G}$	characteristic constraint graph
$>_{\mathbb{C}t}$	con-happen-before relationship
$attack_{(\mathbb{C}_1,\ldots,\mathbb{C}_m)}$	attack path pattern

DT-SM can adapt well to the highly dynamic cyber environment because it takes in the inputs of analysts' operation traces. These traces can be collected continuously as long as the capturing toolkit is launched while analysts are working.

III. The Cyber Security Data Triage System

The cyber security data triage operation and trace are defined in this section. Table I lists the notations to be used in the remaining parts of the paper.

A. Data Triage Model

As the first step to achieve cyber situation awareness [1], data triage usually involves examining the details of alerts and various reports, filtering the data sources of interest for further in-depth analysis, and correlating the relevant data. Figure 1 describes a process of data triage process. The analyst perform a series of data triage operations, each of which manipulates the data sources to gain understanding of the potential attack paths. An attack path includes a series of steps, each of which is performed on the basis of previous steps with the goal of reaching to a final target.

Data sources refer to the network monitoring data generated by multiple security measures, which contain some evidence of the attack paths. Although the data sources may vary across networks equipped with different measures, they can be classified into two main categories: stable data (e.g., network topology and policy of use) and timing data (e.g., IDS alerts, firewall logs and vulnerability reports). We only consider the timing data in the context of data triage. The timing data can be modeled as a sequence of network connection events collected in temporal order (defined in detail below).

In Figure 1, the task data sources have been generalized to a sequence of network connection events, and the events in the sequence belong to different attack paths, and thus can be viewed irrelevant. A data triage analyst's responsibility is to make fast decisions about which network connection events could be related to an attack path and worth further investigation. By leveraging domain knowledge and experience, an analyst need to decide what characteristics of the network connection events can be viewed as suspicious events. Once the characteristics have been identified, the analyst can narrow down the search space by using the characteristics as constraints. The scenario in Figure 1 illustrates several key concepts in a data triage process: network connection events, analyst's data triage operations, and the characteristic constraints of events. The detailed definitions of these concepts are provided as follows.

We consider the attack paths at the abstraction level of network connection event. A **network connection event** (written in short as event in this paper) is defined by a tuple with 8 attributes,

$$e = < time, conn, ip_{src}, port_{src}, ip_{dst}, port_{dst}, protocol, msg >,$$

where *time* is the time when the connection activity is performed, *conn* is the type of connection activity (a connection can be "Built", "Teardown", and "Deny"), ip_{src}, $port_{src}$ and ip_{dst}, $port_{dst}$ are the ip address and port of the source and destination respectively, *protocol* is the connection protocol type, and *msg* is the message associated with the connection. A successful TCP communication session usually is created by a "Built" event and ended with a "Teardown" event. A "Deny" event indicates a failed connection attempt. An attack path is represented as a finite sequence of events $(e_1, \ldots, e_n)$.

B. Analysts' Data Triage Operation

Analysts conduct *data triage operations* to filter suspicious network connection events in the data sources. According to the study on analysts' daily operations in data triage [1], these operations can be performed in three ways: to filter based on a condition ($FILTER$), to search by using a keyword ($SEARCH$) and to directly select a portion of connection events with common characteristics ($SELECT$). The three types are explained as follows.

- FILTER(D_{in}, D_{out}, C): Filter a set of connection events (D_{in}) based on a condition (C) on the attribute values of the connection events and result in a subset (D_{out}).
- SEARCH(D_{in}, D_{out}, C): Search a keyword (C) in a set of connection events (D_{in}). It results in a subset (D_{out}).
- SELECT(D_{in}, D_{out}, C),$D_j \subseteq D_i$: Select a subset of connection events (D_{out}) in a set of connection events (D_{in}), the subset of connection events (D_{out}) has a characteristic (C).

Each data triage operation filters out a subset of events from the original set of events specified in the data sources. The characteristics of the subset events are determined by the constraint defined in the data triage operation. An atomic constraint is a filtering component that specifies the value range of an attribute of the event,

$$T_i = r_v(attr, val),$$

where r_v is the relationship between values, $r_v = \{=, <>, >, <, <=, >=\}$. Recall the previous example of a SIEM rule in Section I. The "`EventType (In) Failure`" is an atomic constraint represented by =($attr$, 'Failure').

A constraint on the subset characteristics can be multidimensional because a event has multiple attributes. A constraint can be represented by a predicate in disjunctive normal form, named "**Characteristic Constraint** (CC)",

$$\mathbb{C} = \bigvee (T_1 \wedge \ldots \wedge T_n),$$

where $T_i (1 \le i \le n)$ is an atomic predict.

Let $\mathbb{C}$ be a characteristic constraint specified in a data triage operation, and let D be a set of events. A data triage operation is represented by a 2-tuple,

$$O_1 = (t, f_{\mathbb{C}}),$$

where t is the timestamp of the analyst performing this operation, $f_{\mathbb{C}} : P(D) \to P(D)$, such that $f_{\mathbb{C}}(D) = \{e_k | e_k \in D, C \text{ holds on } e_k\}$, and $P(D)$ is the power set of D.

C. Data Triage Operation Trace

A data triage operation trace consists of a series of data triage operations performed by an analyst and the underlying temporal and logic relationships between the operations.

1) Relationship between Characteristic Constraints: Let $O_1 = (t_1, f_{\mathbb{C}_1}(D))$ and $O_2 = (t_2, f_{\mathbb{C}_2}(D))$ be two different data triage operations performed on a set of network events D, we define three types of logical relationships between them based on their associated characteristic constraints: "is-equal-to", "is-subsumed-by" and "is-complementary-with".

$$isEql(O_1, O_2) \iff \mathbb{C}_1 \leftrightarrow \mathbb{C}_2,$$

$$isSub(O_1, O_2) \iff \mathbb{C}_1 \to \mathbb{C}_2,$$

$$isCom(O_1, O_2) \iff \mathbb{C}_1 \to \neg\mathbb{C}_2 \; and \; \mathbb{C}_2 \to \neg\mathbb{C}_1,$$

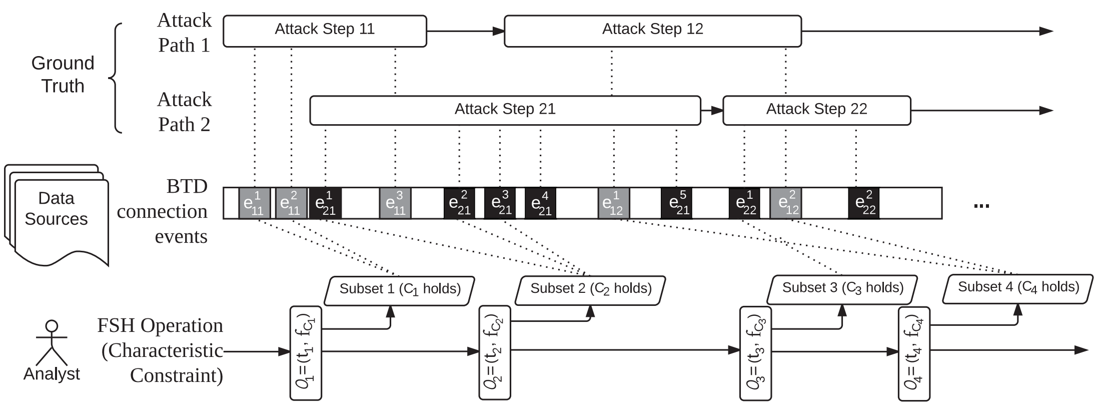

Fig. 1. A scenario of data triage process demonstrates an analyst filtering the network connection events based on characteristic constrains.

where $\rightarrow$ refers to implication, $isEql(\cdot,\cdot)$ and $isCom(\cdot,\cdot)$ are bidirectional relationships but $isSub(\cdot,\cdot)$ is a unidirectional relationship. A special case of "is-complementary-with" is "*is-absolute-complementary-with*": $isCom(O_1, O_2)$ and O_1 and O_2 are absolute complement given a common set of network events, that is, $\mathbb{C}_1 = \neg\mathbb{C}_2$. For example, given a set of network events with a common characteristic "SrcPort <> 80", the data triage operations using the characteristic constraints "SrcPort = 6667" and "SrcPort <> 80 AND SrcPort <> 6667" are absolute complement.

2) Characteristic Constraint Graph: Combining the logical relationships with the temporal relationships, an analyst's triage process can be modeled as a directed graph, named "Characteristic Constraint Graph (CC-Graph)", $G(trace) =< V, \{R_l\} >, V = \{O_1, \ldots, O_n\}$, $R_l \subseteq V * V, l \in \{isEql >_t, isSub >_t, isCom >_t\}$. The vertexes of the graph are the data triage operations, and a directed edge between two vertexes represents a conjunction of a constraint-related logical relationship and a "happen-after" relationship (i.e. $isEql >_t$, $isSub >_t$, $isCom >_t$).

The reason why edges of CC-Graph are defined with both the temporal and logical relationship is because we are especially interested in how an operation is related to the operations that were conducted before it. A data triage operation results in a subset of network connections with a certain characteristics. If the analyst found the subset noteworthy, he/she may generate several hypotheses about the possible attack path based on the observation. To further investigate these hypotheses, the analyst may perform further data triage operation to narrow down the search space. Therefore, the sequence of these operations and their relationships can imply the analyst's data triage strategy.

IV. The Automated Data Triage Approach

We propose an approach to building an automated data triage system based on the captured traces of analysts' cognitive processes of data triage, including four main steps. (1) We identify the data triage operations from the collected traces along with the characteristics constrains used in these data triage operations. (2) We represent the data triage operations in the collected traces and their temporal and logical relationships in a Characteristic Constraint Graph (CC-Graph). (3) To mine the rule ingredients, we analyze the CC-Graphs to construct the key data characteristic constraints as suspicious event detectors. These constraints are further correlated with the original network data sources to identify the "can-happen-before" relationships among the suspicious events. The constrains and their relationships represent various attack path patterns. Each attack path pattern, which is formally represented, has a semantic meaning that defines a class of network connections indicating multi-step attacks. Analysts can review, modify and extend them. (4) The formally represent attack path patterns can be directly used to initialize a finite state machine for conducting automated data triage, just as adding rules to a SIEM system.

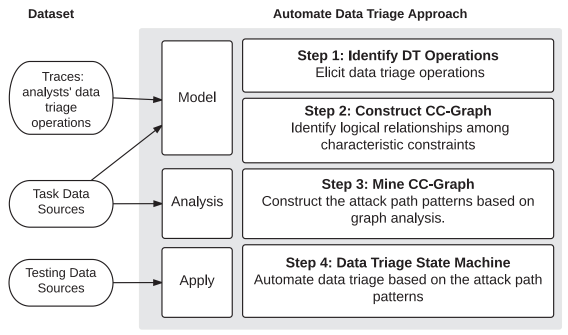
Dataset
Automate Data Triage Approach
Traces: analysts' data triage operations
Task Data Sources
Testing Data Sources
Model
Analysis
Apply
Step 1: Identify DT Operations
Elicit data triage operations
Step 2: Construct CC-Graph
Identify logical relationships among characteristic constraints
Step 3: Mine CC-Graph
Construct the attack path patterns based on graph analysis.
Step 4: Data Triage State Machine
Automate data triage based on the attack path patterns

Fig. 2. The overview of the automated data triage approach.

A. Step 1: Identify Data Triage Operations

To capture analysts' data triage operations, a toolkit, named ARSCA, has been developed to track an analyst's analytical reasoning processes in a minimum-intrusive manner [22]. ARSCA integrates the network monitoring data collected from multiple sources and provides the analysts with the data triage functions including SEARCH, FILTER and SELECT. Figure 3 displays the function areas in the ARSCA's interface. Region 1 and 2 provide functions of filtering by condition and searching by keyword, thus supporting the operations of FILTER and SEARCH. ARSCA lets an analyst select the entries in a provided data source as the suspicious events (Region 3), thus supporting the SELECT operation [22].

ARSCA automatically records an analyst's data triage operations while the analyst is performing a data triage task [17]. Once the analyst finishes a task, a piece of log recording the analyst's data triage operations will be created in the following structural format:
```
<Item Timestamp=[TIMESTAMP]>
[ACTION_TYPE]
([CONTENT])
</item>
```
Each item contains the information of the occurring time, type and characteristic constraint of a data triage operation. "ACTION_TYPE" refers to the types of data triage operations, including FILTER, SELECT (LINK) and SEARCH. LINK is an additional type of SEARCH, which corresponds to the LINK function provided by ARSCA which enables an analyst to specify the common characteristics of a subset of the selected network events. Several examples of the recorded data triage operations, corresponding to FILTER, SELECT (LINK), and SEARCH, are demonstrated in Table II. It indicates that the "CONTENT" field of each record captures the information about the characteristic constrains specified in the data triage operation, which are the underlined parts in Table II

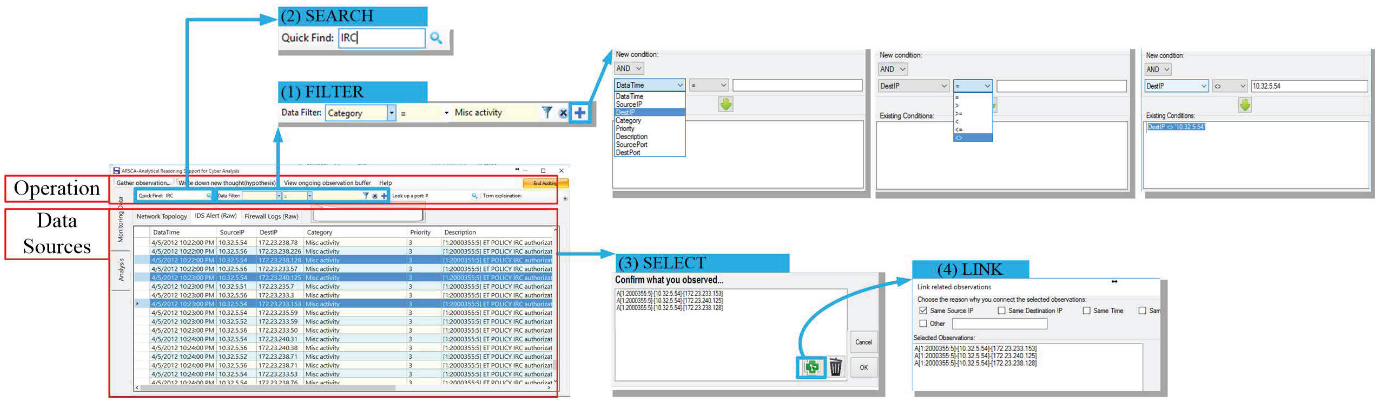

Fig. 3. Main components of the ARSCA user interface (in Data View and Analysis View) [17]

TABLE II
THE ARSCA LOG ITEMS RELATED TO DATA TRIAGE OPERATIONS

Operation	Examples in ARSCA Log	Description

item_FILTER	`<Item Timestamp= 07/31 13:01:41>`
	`FILTER(`
	`SELECT * FROM IDS Alerts WHERE`
	`SrcPort = 6667`
	`</Item>`
		Filter the a set of connection events (i.e.IDS Alerts) based on a condition (i.e. SrcPort=6667)

item_SELECT	`<Item Timestamp= 07/31 13:20:29>`
	`SELECT (`
	`FIREWALL-[4/5/2012 10:15 PM]-`
	`[Built]-[TCP](172.23.233.57/3484,`
	`10.32.5.58/6667),`
	`FIREWALL-[4/5/2012 10:15 PM]-`
	`[Teardown]-[TCP](172.23.233.52`
	`/5694,10.32.5.59/6667),`
	`FIREWALL-[4/5/2012 10:15 PM]-`
	`[Built]-[TCP](172.23.233.57/3484,`
	`10.32.5.58/6667),`
	`FIREWALL-[4/5/2012 10:15 PM]-`
	`[Teardown]-[TCP](172.23.233.58`
	`/3231,10.32.5.51/6667),`
	`</Item>`
		Filter the a set Select a set of connection events (i.e. the underlined firewall log entries) with a common characteristics (i.e. DstPort=6667)

item_LINK	`<Item Timestamp= 07/31 13:20:43>`
	`LINK (Same DstPort)`
	`</Item>`
		Identify the common characteristics (i.e.DstPort) of a selected set of connection events

item_SEARCH	`<Item Timestamp= 08/09 11:08:01>`
	`SEARCH(`
	`Firewall Log, 172.23.233.52)`
	`</Item>`
		Search a keyword in a set of connection events specified in the Firewall log.

TABLE III
IDENTIFYING DATA TRIAGE OPERATIONS FROM ARSCA LOG

Item Sequence in ARSCA Log	Characteristic Constraint in a data triage operation
item_FILTER(cond[a])$\longrightarrow$ item_OTHER[b]	$\mathbb{C}$ = cond
item_FILTER(cond) $\rightarrow$ item_SELECT($\{e_i\}$) $\rightarrow$ item_OTHER	$\mathbb{C}$ = cond
item_FILTER(cond) $\rightarrow$ item_SELECT($\{e_i\}$) $\rightarrow$ item_OTHER	$\mathbb{C}$ = cond
item_FILTER(cond) $\rightarrow$ item_SELECT($\{e_i\}$) $\rightarrow$ item_LINK(attr[c])	$\mathbb{C}$ = cond $\cap$ (attr = attr$_{e_i}$)
item_SEARCH(kwd[d]) $\rightarrow$ item_OTHER	$\mathbb{C}$ = (attr$_{infer}$ = kwd)
item_SEARCH(kwd) $\rightarrow$ item_SELECT($\{e_i\}$) $\rightarrow$ item_OTHER	$\mathbb{C}$ = (attr$_{e_i}$ = kwd)
item_SELECT($\{e_i\}$) $\rightarrow$ item_OTHER	$\mathbb{C} = \wedge_i \vee_j$ (attr$_j$ = val$_{e_i,attr_j}$)
item_SELECT($\{e_i\}$) $\rightarrow$ item_LINK(attr) $\rightarrow$ item_OTHER	$\mathbb{C} = \wedge_i$ (attr = $\wedge$ val$_{e_i,attr}$)

[a]filtering condition
[b]other log items
[c]the attribute of the events of interest
[d]search keyword

Focused on the data triage operations, a trace parser is developed to identify the data triage operations from the captured traces. The difficulty lies in the mapping from the ARSCA trace items to the data triage operations which is not a one-to-one matching, considering the fact that an analyst may conduct several successive actions as one data triage operation. More specifically, a item_FILTER or a item_SEARCH item could be followed by a item_SELECT item, which means, after narrowing down the search space by performing filtering or searching, the analyst may further select some network events in the filtered subset as suspicious events. However, sometimes analysts may directly select a subset of network events without performing any filtering or searching in advance. Both cases indicate one data triage operation. Furthermore, a item_LINK item may appear immediately after a item_SELECT item when the analyst wants to specify the common characteristics of the selected network events. Given all these situations, we mainly refer to the eight types of action sequences in Table III to parse an ARSCA trace into a set of data triage operations.

B. Step 2: Construct Characteristic Constraint Graph

Given the data triage operations identified from the traces, we construct characteristic constraint graphs by determining the relationships among these data triage operations. The algorithm of characteristic constraint graph construction is described in Algorithm 1. An example of characteristic constraint graph is shown in Figure 4. The nodes are the data triage operations, along with the text which are indexes of them in the temporal order. The nodes in square are those triggered at least one hypotheses during an analyst's analysis process, while circles not triggering any hypothesis. There are four different types of edges which corresponds to different logical relationships between data triage operations (i.e., "*is-equal-to*", "*is-subsumed-by*", "*is-complementary-with*" and absolute "*is-complementary-with*") combined with their temporal relationships (i.e., "*happen-after*". Figure 4 is a partial characteristic constraint graph, with 6 nodes and their relationships. Node 8 is a data triage operation using the constraint "DstPort=6667". It is represented in a square because it resulted in some interesting findings and triggered a hypothesis. After performing this data triage operation, the analyst screened out the

```
Data: $S_{op}$, a sequence of data triage operations in temporal order
Result: $\mathcal{G}$, a characteristic constraint graph
$V = \{op | op \in S_{op}\}$; // vertex set
$E = \emptyset$; // edge set
for $i$ in 2:len($S_{op}$) do
    $\mathbb{C}_i$ = characteristic constraint of $S_{op}[i]$;
    for $j$ in 1:i do
        $\mathbb{C}_j$ = characteristic constraint of $S_{op}[j]$;
        $rela$ = relation( $\mathbb{C}_j$, $\mathbb{C}_i$);
        if label is not null then
            $E = E + < S_{op}[j], S_{op}[i], rela >$;
        end
    end
end
relation($\mathbb{C}_1$, $\mathbb{C}_2$)
    if $\mathbb{C}_1 \rightarrow \mathbb{C}_2$ and $\mathbb{C}_2 \rightarrow \mathbb{C}_1$ then
        return "isEql" ;
    end
    else if $\mathbb{C}_1 \rightarrow \mathbb{C}_2$ then
        return "isSub" ;
    end
    else if $\mathbb{C}_1 \rightarrow \neg\mathbb{C}_2$ and $\mathbb{C}_2 \rightarrow \neg\mathbb{C}_1$ then
        return "isCom" ;
    end
    return "Other";
end
```

Algorithm 1: Construct a Characteristic Constraint Graph

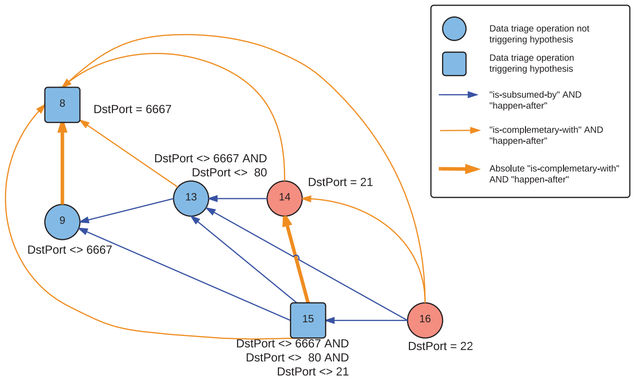
Data triage operation not triggering hypothesis
Data triage operation triggering hypothesis
"is-subsumed-by" AND "happen-after"
"is-complemetary-with" AND "happen-after"
Absolute "is-complemetary-with" AND "happen-after"
8
DstPort = 6667
DstPort <> 6667 AND DstPort <> 80
13
14
DstPort = 21
9
DstPort <> 6667
15
16
DstPort <> 6667 AND DstPort <> 80 AND DstPort <> 21
DstPort = 22

Fig. 4. A partial characteristic constraint graph constructed from an analyst's trace. The nodes are the data triage operations, with the index of their temporal order. A squared node refers to a data triage operation based on which the analyst generated a following hypothesis about the possible attack path. However, circle nodes are the data triage operations that did not result in any hypothesis.

network events with destination port 6667 by conducting another data triage operation (i.e., Node 9). It may indicate that the analyst switched his attention to the unexplored network events with other characteristics after investigating the network events via destination port 6667. The following data triage operations of Node 13 and Node 14 have an "is-subsumed-by" relationship with the data triage operation of Node 9. It implies the analyst gradually screened out the network events to narrow down the scope. The data triage operation of Node 14 is the end of this narrowing process and it let the analyst generate another hypothesis. Similarly, the sequence of Node 9, Node 15 and Node 16 indicates the same strategy employed by the analyst.

C. Step 3: Mine the Characteristic Constraint Graphs

Several steps are taken to mine the CC-Graphs for identifying the characteristic constraints from the analysts' traces for effective data triage. First of all, it is critical to distinguish the critical data triage operations that lead analysts to key findings from the exploratory data triage operations recorded in the traces. Considering that identified critical data triage operations may overlap, a following step is to adjust the identified data triage operations to make sure they contain mutually exclusive characteristic constraints. After the adjustment, data triage operations are the "candidates" for constructing state machines. To make sure each state machine is built up for detecting evidence of one attack chain, the third step is taken to group the nodes according to which attack chain they may belong to based on heuristics.

1) Extract the Critical Endpoints in "isSub" Subgraphs: Given data triage operations O_1 and O_2, $isSub >_t (O_2, O_1)$ indicates that O_2 is performed after O_1, and O_2 further narrows the O_1's network event set by using a more strict characteristic constraint. As the case described in Figure 4, analysts may conduct a series of data triage operations with the "is-subsumed-by" (isSub) relationships to gradually narrow down the network events. The endpoints of such processes can be viewed as the critical characteristic constraints that represent a noteworthy set of network events. To investigate the endpoints, we consider the subgraph that only consists of the edges of the relationship $isSub >_t$ and $isEql >_t$ (isEql is a special case of isSub) and name it "isSub" subgraph.

2) Mutually Exclusive Characteristic Constraints: It is desirable to have the endpoints of an "isSub" subgraph be mutually exclusive. To ensure the nodes to be mutually exclusive, we examine the logic relationships between two nodes, and add additional characteristic constraints to one of them as follows. Let V_{ends} be the set of endpoints in an "isSub" subgraph, several steps are taken to make sure every two endpoints in this subgraph have an "isCom" relationship. (1) If two endpoints have an "isEql" relationship, we drop one from V_{ends}. (2) Given V_{ends}, we consider the subgraph induced by V_{ends} from the original CC-Graph, $G_{induced} =< V_{ends}, \{R_{logic}\} >$, where $R_{logic} = \{< v_i, v_j > | isEql(v_i, v_j)\ or\ isSub(v_i, v_j)\ orisCom(v_i, v_j)\}$. Only focusing on the edges of "isCom" relationships in $G_{induced}$, we apply the Bron-Kerbosch algorithm to find all the maximum cliques in $G_{induced}$. A maximum clique is a subset of the vertexes, represented by $\mathcal{G}_C \in V_{induced}$, such that $\forall v_i, v_j \in \mathcal{G}_C, isCom(v_i, v_j)$, and $\nexists v_k \in V_{induced}$, $v_k \notin \mathcal{G}_C$, and $\forall v_i \in \mathcal{G}_C$, $isCom(v_k, v_i)$. The largest maximum clique is denoted as V_{clique}. (3) We add each $v \in V_{ends}, v \notin V_{clique}$ into V_{clique} by removing the overlap between v and $v_j \in V_{clique}$. For $v_i \notin V_{clique}$, it overlaps with a $v_j \notin V_{clique}$, iff there exists a network event e satisfies both $\mathbb{C}_i$ and $\mathbb{C}_j$. To remove that overlap, we adjust the characteristic constraint used in v_i (say $\mathbb{C}_i$) to $\mathbb{C}_i \wedge \neq \mathbb{C}_j$. If $\mathbb{C}_i \wedge \neq \mathbb{C}_j$ is not false, we add the adjusted v_i to $V_{clique}(V_{clique} \in \mathcal{G}_C)$.

3) Group the "Candidates" based on Heuristics: Given a set of adjusted data triage operations as the "candidates" for constructing state machines, we group them according to which attack chain they belong based on heuristics. One heuristic is to group the data triage operations whose characteristic constraints are set on a same IP (e.g., a suspected attack or a potential target), considering the fact that an attack chain usually has one common attacker and target. The second heuristic is that two data triage operations can be grouped if they were arranged in a same path of AOH-Trees [14] by an analyst when he/she was performing a task. According to the design of AOH-Trees, a descendant action node is viewed as a follow-up action for investigating a hypothesis generated based on previous action. These heuristics are generated based on the domain knowledge learned from our previous experience. The characteristic constraints of these "candidates" are as readable as SIEM rules. Domain experts could also create or modify these heuristics for grouping the candidates if we apply this approach to real world.

D. Step 4: State Machine for Real-Time Pattern Matching

1) Attack Path Pattern: Given the critical characteristic constrains mined from the CC-Graphs (i.e., the nodes in V_{clique}), attack path patterns can be constructed based on the critical nodes and their temporal orders. An attack path pattern is a sequence of characteristic constraints in temporal order, which specifies a class of attack path instances. The temporal relationship is a "can-happen-before" relationship between two characteristic constraints, denoted as $\succ_{\mathbb{C}_t}(\cdot,\cdot)$. Let $\mathbb{C}_1$ and $\mathbb{C}_2$ be two pre-specified characteristic constraints, $\succ_{\mathbb{C}_t}(\mathbb{C}_1, \mathbb{C}_2)$ refers to the analysts' knowledge about the attack: it happened and could happen again if a set of network events satisfied $\mathbb{C}_1$ occurred before the other set of events involving the same hosts that satisfies $\mathbb{C}_2$ in an attack path. For example, one typical step in a botnet attack is the IRC communication between internal workstation with the external C&C servers. The IRC communication can be followed by a data exfiltration using FTP protocol. Therefore, this attack can be discovered by the attack pattern path: "`(SrcIP = external servers AND SrcPort = 6667) OR (DstIP = external servers AND DstPort = 6667)`" and "`SrcIP = internal workstation AND DstPort= 21`".

The "can-happen-before" relationships of the nodes in an attack pattern is identified based on the temporal order of the corresponding network events in the task performed by the analyst. Assume that the network events come in sequence over time $E = (e_1, \ldots, e_n)$. We say this sequence of network events satisfies a sequence of characteristic constraints $(\mathbb{C}_1, \ldots, \mathbb{C}_m)$ defined in an attack path pattern $\mathcal{G}_{\mathbb{C}}$, iff we have, $\forall \mathbb{C}_i (1 \le i \le m), \exists e_p (1 \le p \le n)$ satisfies $\mathbb{C}_i \rightarrow \forall \mathbb{C}_j (1 \le j < i)$, $\exists e_q (1 \le q \le p)$ satisfies $\mathbb{C}_j$. Therefore, given $(\mathbb{C}_1, \ldots, \mathbb{C}_m)$ and E, the attack path instances detected based on $(\mathbb{C}_1, \ldots, \mathbb{C}_m)$ is a sequence of network event sets, $attack_{(\mathbb{C}_1,\ldots,\mathbb{C}_m)}(\mathcal{E}) = \{(\mathcal{E}_1, \ldots, \mathcal{E}_m)\}$, where $\mathcal{E}_{ik}(1 \le i \le m) = \{e_{ik} | e_{ik}$ *is a network event from* $source_{ik1}$ to $destination_{ik2}$ *that satisfies* $C_i\}$. The algorithm for identifying the "can-happen-before" relationships is demonstrated in Algorithm 2.

```
Data: $D_{task}$: a sequence of network event sets;
      $C$: a set of characteristic constrains
Result: $\mathcal{G}_C =< C, \{\prec_{\mathbb{C}_t}\} >$, an attack path pattern
1  $map < \mathbb{C}, \{e\} >$ // map each characteristic constraint
       to the network events that satisfies it, the
       network events are in temporal order
2  for $e_i$ in $D_{task}$ do
3      if exists $\mathbb{C}$ holds on $e_i$ then
4          $map$.put($\mathbb{C}$, $e_i$);
5      end
6  end
7  for $\mathbb{C}_1$, $\mathbb{C}_2$ in $C$ do
8      Find the first $e_1$ in $map(\mathbb{C}_1)$, and the first $e_1$ in $map(\mathbb{C}_1)$ that
         $e_1.\{src, dst\} = e_2.\{src, dst\}$;
9      if $e_1.time < e_2.time$ then
10         add $\prec_{\mathbb{C}_t}(\mathbb{C}_1, \mathbb{C}_2)$;
11     end
12 end
```

Algorithm 2: Identifying "can-happen-before" relationships

2) State Machine: A finite state machine is constructed to automate data triage, given an attack path pattern, named "DT-SM". A state transition is defined as $\delta : S * D \rightarrow S$. Given the current state $S_i \in S$ and a new network event e_i, we have $\delta(S_i, e_i) = S_(i+1)$ iff $\exists C_j$, that $\nexists(\mathbb{C}_0, \ldots, \mathbb{C}_n) \in S_i$ and $\mathbb{C}_j \in (\mathbb{C}_0, \ldots, \mathbb{C}_n)$, e_i satisfies C_j and $\exists(\mathbb{C}_0, \ldots, \mathbb{C}_n) \in S_i$, that $\succ_{\mathbb{C}_t}(\mathbb{C}_i, \mathbb{C}_j)$. Therefore, $S_(i+1) = S_i - (\mathbb{C}_0, \ldots, \mathbb{C}_i) + (\mathbb{C}_0, \ldots, \mathbb{C}_j)$. Therefore, given the input of a sequence of network events in temporal order, a DT-SM takes one network event at one time and examine whether this event triggers state transition. Once all the input network events have been processed, the DT-SM output the instances of the attack path pattern. Each instance is a sequence of network event sets that satisfy a characteristic constraint sequence in the attack path pattern.

V. Evaluation

We implemented the automated data triage system and tested it in our experiment. Our evaluation focuses on (1) the effectiveness of constructing the data triage automatons (DT-SM) based on the traces, (2) the usefulness of the data triage rules mined from the traces and the performance of the constructed DT-SMs, and (3) the impact factors of the DT-SMs' performance.

A. Experiment Dataset

1) ARSCA Logs Collected from a Lab Experiment: We recruited 30 full-time professional cyber analysts in a previous experiment and asked them to accomplish a cyber attack data triage task. The ARSCA toolkit was used to audit the analysts' operations while they were performing the task [17]. We have to screen out one ARSCA log because the participant didn't accomplish the task. At last, we collected 29 ARSCA logs in total with 1104 trace items. The average length of the logs is 31.17.

2) Task Data Sources: The cyber attack analysis task is designed using the data provided by VAST Challenge 2012, which is a cyber situational awareness task with a setting of a bank's network with approximately 5000 machines. The VAST challenge provides participants with IDS alerts and firewall log from the network. VAST Challenge 2012 asked the participants to identify the noteworthy attack incidents happened in the 40 hours covered by the IDS alerts and firewall logs [23]. The entire data sources were composed of 23,595,817 firewall logs and 35,709 IDS alerts. The task scenario underlying the data sources is a multistage attack path that begins with normal network connection, including regional headquarters computers communicating with internal headboards financial sever and normal web browsing. The attack starts when an internal workstation was infected with a botnet due to an inserted USB. The botnet replicated itself to other hosts on the network, and meanwhile the bonnet communicated with several C&C servers. The botnet kept infecting additional computers. It attempted to exfiltrate data using FTP connections but it failed. After that, the botnet successfully exfiltrated data using SSH connections. In the following hours, the majority of the botnet communication and data exfiltration kept happening.

The original data sources provided in the VAST challenge are too large for human analysts to analyze during the limited task time. Therefore, only small portions of data were selected from the original dataset and provided to the participants to enable them to complete the task in the experimental session (60 minutes). Two time windows were selected and each of them were used to make a cyber security analysis task for the subjects in that experiment, which are the time window I and time window II shown in Table IV. In the experiment where ARSCA logs were collected, 10 analysts accomplished the task of time window I, and 19 analysts accomplished the task of time window II.

B. Evaluation of DT-SM Construction

1) Data Triage Operation Identification: We first evaluated the accuracy of the automated data triage operation identification by comparing with the data triage operation identified by human. We had two persons manually parse the ARSCA logs, and the identified data triage operation serves as the ground truth. Both persons are familiar with the task and the ARSCA toolkit. One person is the main coder, and the other is the evaluator: given an ARSCA log, the main coder first read through the trace and specified the data triage operations in the log; the evaluator then read the trace for the second time and proof-readed the data triage operations identified by the main identifier; every time when a disagreement occurred, both of them went back to the ARSCA log and tried to reach an agreement by discussion. Given the 29 ARSCA logs, 1181 log items were manually analyzed and 394 data triage operations were identified.

The automatic system identified 348 data triage operations in total. Comparing them with the ground truth, there were 322 data triage operations correctly identified by the system. There were 62 data triage operations in the ground truth but not identified by the system. Therefore, the false positive rate is 0.075 and the false negative rate is 0.161.

In total, 29 CC-Graphs were constructed from the 29 traces of the professional analysts' cognitive processes collected in our previous experiment. These CC-Graphs indicated that the analysts' analytical reasoning processes of data triage are non-linear.

Table V summarizes the statistics of the CC-Graph. The large standard deviation of node number and edge number of the CC-Graphs also indicate that there exist large individual differences among analysts. The edges of "isSub" and "isCom" (i.e., corresponding to

TABLE IV
ATTACK GROUND TRUTH IN 3 SELECTED TIME WINDOWS

Time Window	Attack Scenario (Steps)	Instances	Data Sources
Scenario from the beginning (started from 4/5 20:25): An internal workstation got infected with a botnet due to a USB insertion. The botnet spread fast.			
I: 4/5 22:15-22:25 (10 min)	(1) The botnet communicated with the external C&C servers using IRC. (2) Failed attempted to exfiltrate data using FTP. (3) Successful attempted to exfiltrate data using SSH.	8 external servers, 105 internal workstation. 247 ATTACK PATH PATTERN instances.	# IDS Alerts: 239 # Firewall Log: 115,524
Scenario in the middle (about 20 hours, 4/5 22:26-4/6 17:56): The IT department noticed the problem and rebooted some infected workstations several times. The data exfiltration drops, but the majority of the botnet traffic still exists.			
II: 4/6 18:05-18:15 (10 min)	(1) The botnet spread (inquiring the network hardware) (2) New IRC communications. (3) New attempt to exfiltrate data using FTP.	11 external servers, 233 internal workstations. 233 ATTACK PATH PATTERN instances.	# IDS Alerts: 228 # Firewall Log: 48,012
III: 4/6 18:16-19:56 (100 min)	(1) Additional workstations were infected. (2) The botnet communication continued. (3) FTP exfiltration continued.	13 external servers, 517 internal workstations. 3055 ATTACK PATH PATTERN instances.	# IDS Alerts: 1810 # Firewall Log: 599,489

TABLE V
SUMMARY STATISTICS FOR THE CC-GRAPHS

	Nodes	Avg. degree	Max SCC size	Edges	"isEql" edges	"isSub" edges	"isCom" edges
Mean	12.00	7.59	11.62%	38.55	2.069	10.41	26.07
StDev	6.79	5.32	6.83%	47.75	3.105	14.98	34.74

TABLE VI
TOP 10 CHARACTERISTIC CONSTRAINTS IN THE ATTACK PATH PATTERN BY ANALYZING THE 29 ANALYSTS' DATA TRIAGE OPERATIONS

1	DSTPORT = 6667 AND SRCIP = 172.23.*.* AND DSTIP = 10.32.5.* AND PROTOCOL = TCP AND SERVICE = 6667_tcp
2	SRCPORT = 6667 AND SRCIP = 10.32.5.* AND DSTIP = 172.23.233.* AND PRIORITY = 3 AND DESCRIPTION = [1:2000355:5] ET POLICY IRC authorization message
3	SRCIP = 172.23.*.* AND DSTIP = 172.23.0.10 AND DSTPORT = 445 AND PRIORITY = 3
4	SRCIP = 172.23.235.* AND DSTIP = 10.32.5.* AND DSTPORT = 21 AND OPERATION = Deny AND PRIORITY = Warning AND PROTOCOL = TCP AND SERVICE = ftp
5	SRCIP = 172.23.*.* AND DSTIP = 10.32.5.* AND DSTPORT = 6667 AND OPERATION = Deny
6	SRCIP = 172.23.*.* AND DSTPORT = 53
7	SRCIP = 172.23.234.* AND DSTIP = 10.32.5.* AND DSTPORT = 22
8	SRCIP = 10.32.5.* AND DSTIP = 172.23.1.168 AND DSTPORT = 6667
9	DSTPORT = 80 AND DSTIP = 10.32.0.100 AND SRCIP > 172.23.0.0
10	SRCIP = 172.23.0.108 AND DSTPORT = 6667 AND DSTIP = 10.32.5.*

the "is-subsumed-by and happen-after" and "is-complementary-with and happen-after" respectively) account for a large portion of the CC-Graph edges. Besides, it also shows that, in average, a maximum strongly connected component (SCC) accounts for 11.62% of a CC-Graph. SCC represents a set of mutually connected data triage operations, which may imply how an analyst detected suspicious network events and update his/her mental model along with switching his/her attention from one hypothesis to another. A large SCC indicates that a large number of data triage operations are directly or indirectly connected in an analyst's analytical reasoning process.

2) Attack Path Pattern Construction: To evaluate the effectiveness of attack path pattern construction, we constructed an attack path pattern for each CC-Graph. We had to exclude 5 ARSCA logs because less than 2 data triage operations were identified in these logs. In total, we have 32 attack path patterns constructed, containing 81 characteristic constraints. The average number of the nodes in the attack path patterns is 2.781.

We evaluated these attack path patterns by checking whether its characteristic constraints are critical and mutually exclusive. To decide whether the characteristic constraints in an attack path schema are critical, we mapped them to the analysts' answer to the task question that asked about the most important observations. We found 79 out of the 89 characteristic constraints in the 32 attack path patterns mentioned by the analysts that lead to important observation (88.76%). As for the 10 characteristic constraints not mentioned in analysts' answer, we found 6 of them also lead to hypotheses in analysts' ARSCA log. All the characteristic constrains in the attack path pattern are mutually exclusive. Table VI lists the top 10 characteristic constraints included in the attack path pattern.

C. Performance of DT-SM

We evaluated the data triage performance of DT-SM on the data sources that are much larger than the task data sources used for constructing DT-SM (used in the ARSCA logging experiment). Considering the task data sources selected from a 10-minute-time window, we chose a 100-minute time window (4/6 18:16-19:56) to collect the testing data sources from the original VAST data sources, which corresponds to the data in time window III in Table IV.

The second problem is to determine the ground truth before evaluating the DT-SM's data triage results. Although the VAST Challenge provides an attack description, it is still not apparent regarding whether each alert/log entry in the data sources is related to an attack or not. Therefore, we manually processed the data sources in time window III and tagged each data entry regarding to which attack step it is related.

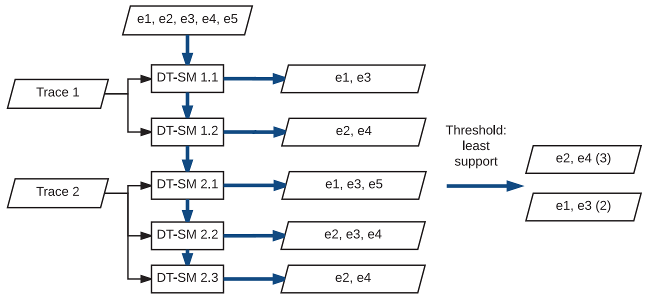

Fig. 5. Identifying suspicious network event sequences based on multiple DT-SMs

A set of DT-SMs were constructed from the traces, each of them can detect a set of event sequences. We combined the results

TABLE VII
PERFORMANCE OF THE DT-SMS BUILT ON THE TRACES FROM ANALYSTS WITH DIFFERENT TASK PERFORMANCE. THE THRESHOLD OF LEAST SUPPORT IS 2.

Group	Performance Score (1-5)	Performance
High-Performance	4-5	False Positive: 0.010 False Negative: 0.225
Low-Performance	2-3	False Positive: 0.015 False Negative: 0.350

of multiple DT-SMs by selecting the event sequences with high occurrence frequency. Figure 5 demonstrates an example of how the DT-SMs results are combined. A threshold can be set up to determine what is the least frequency of occurrence for the event sequences of interest (i.e., "least support"). Given the output of each DT-SM, we calculated the occurrence of frequency of each event sequence and selected those with occurrence of frequency above the threshold.

The performance of the DT-SM was measured by false positive and false negative rates. We use e^S to denote the network events in the output of the DT-SM S, and e^T to denote the network events included in the ground truth T. We calculated the false positive and false negative rate using the following formula:

$$false\ positive = 1 - \frac{|\cup\{e^T|e^T \in S\}|}{|\cup\{e^S|e^S \in S\}|} \quad (1)$$

$$false\ negative = 1 - \frac{|\cup\{e^T|e^T \in S\}|}{|\cup\{e^T|e^T \in T\}|} \quad (2)$$

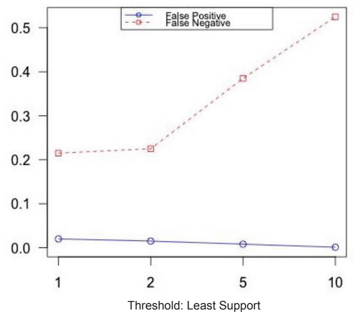

Fig. 6. The performance of DT-SMs with different thresholds of least support.

We constructed a DT-SM constructed based on the data triage operations from the 29 participates and ran it on the testing data. The performance of the DT-SMs (i.e., false positive and false negative) built on their traces is shown in Figure 6. We set the threshold of least support as 1, 2, 5 and 10 respectively. The result shows that, in general, the false positive rates are satisfactory. However, the false negative rates are high. As the threshold increases, the false positive rate decreases while false negative rate increasing. Comparing the results at different thresholds, we found that the best result occurred when the threshold of least support was set as 2. One of the possible reasons of the high false negative rate can be that the participants failed to detect some suspicious network events in the task so that their traces don't involve the effective data triage operations. Then, we will evaluate the effects of analysts' task performance on the DT-SMs' performance.

D. Effect of Analysts' Task Performance on the DT-SM's Performance

We know that the performance of a DT-SM is mainly determined by the quality of the attack path patterns mined from the analysts' traces, and analysts' traces are closely related to analysts' task performance. Therefore, we compared the performance of the DT-SMs built on the traces from the analysts with different task performance.

In the experiment in which the traces were captured, analysts were asked about what the attack scenario underlies the provided data sources after accomplishing the task. We evaluated the analysts' task performance by comparing their answers with the ground truth and gave each of them a score in the range of [1, 5]. Among the 29 participants, the first 10 analysts, who participated in the first task working with the data sources of time window I, displayed a diverse range of performance, while the other 19 analysts with the data sources of time window II presented identical results with similar scores. Therefore, we only took the traces of the first 10 analysts into examination. We used each analyst's trace to construct a DT-SM and then ran the DT-SM on the testing data sources of time window III. We divided the participants into two groups according to their task performance: Group 1 of 5 participants with high task score (i.e., either 4 or 5), and Group 2 of 5 participants with low task score (i.e., either 2 or 3). Table VII shows the comparison results of the false positive and false negative rates of the DT-SMs built on the traces in these two groups. It shows that the false positive rates in both groups are very small. However, the first group of DT-SMs, which corresponds to higher task performance, have much smaller false negative. It can be explained by the fact that analysts with good performance in the task were able to detect most of the attack-related network events; their data triage operations may contain more useful characteristic constraints compared with the analysts who performed poorly in the task.

1) Worst Case Analysis: : We noticed a DT-SM built on the trace of an analyst (whose ID is 08239) had the highest false negative rate. We looked into the attack path patterns constructed based on his trace, and found a sequence of 3 characteristic constraints: (1) `DSTPORT=6667`, (2) `DSTIP = 10.32.0.* AND DSTPORT = 80 AND PRIORITY = Info AND PROTOCOL = TCP`, and (3) `SRCIP = 10.32.5.* AND SRCPORT = 6667 AND DSTIP = 10.32.0.* AND PROTOCOL = TCP`. According to the ground truth, the second characteristic constraint is not related to any attack step of the ground truth. It suggests the characteristic constraint is misleading and results in quite a few instances with irrelevant characteristic constrains so that the false negative is much higher. Such case can be avoided by combining the ARSCA traces of multiple analysts together.

VI. DISCUSSION AND LIMITATION

This work verifies that it is feasible to elicit attack path patterns by modeling and mining the traces of analysts' data triage cognitive processes. In spite of that, it has some limitations. The first one lies in that the traces of the analysts' cognitive processes of analyzing 10-minute-time-window events cannot fully embody the analysis expertise needed for analyzing 100-minute-time-window events in full play. Therefore, the DT-SMs built on the attack path pattern constructed from the traces can't process the network events in the 100-minute time window completely. However, it is not a limitation of the automated data triage system but a problem with the limited set of the traces. We decided to recruit 30 professional analysts based on the consideration of the requirements for domain knowledge in our experiment and the real-world constraints on the accessibility the subjects. Although the size of the current trace collection is not large, we had observed that each trace contains a good amount of detail of an analystâĂŹs analytical process and collected traces demonstrate rich diversity of analytical strategies [17]. These observations enable us to believe the current trace collection is large enough to prove the feasibility of the data triage automaton. In terms of the real-world practice, analysts keep on analyzing the continuous incoming data sources, meanwhile the analysts' traces will automatically grow by ARSCA. Therefore, in this case, the input ARSCA traces will be continuously imported into our system and update the attack path pattern accordingly.

Besides, there still lies some room for improving the false negative rate in several important aspects. Firstly, it would be helpful to enlarge the set of ARSCA traces because introducing more data triage operation traces will enable the system to construct more full-fledged attack path patterns. Meanwhile, a sufficient set of endpoints in the "isSub" subgraph will enable us to conduct frequency item mining to select the significant endpoints, which can lower the false negative rate. In addition, we can also emphasize the non-action-related information captured in ARSCA traces, which is the analysts' hypotheses. Working with ARSCA in a data triage task, analysts may note down their hypotheses to interpret their previous observations or to explain their action plan for further inspection. Such information ought to be useful to improve the accuracy of specifying the characteristic constraints in the identified data triage operations.

Thirdly, although the automated system is developed to reduce labor costs and to improve quality in incident reports, the automation meanwhile creates new tasks for analysts, such as working together with ARSCA for cognitive task tracing and comprehending the output of the automated data triage system. Therefore, it is necessary to conduct further evaluation of the impact of the automated data triage system on cyber security analysts' work efficiency. Torkzadeha and Doll suggested a four factor instrument that measures how technology on work impact task productivity, task innovation, customer satisfaction and management control [24]. Similarly, we need to generate an instrument for measuring the impacts of the automated data triage system on the work efficiency of analysts. Based on the measurement instrument, we can further test our hypotheses about the positive influence of the automated system on reducing the analysts' workloads and improving the quality of incident reports.

VII. Conclusion

Built on the process tracing method for capturing the traces of analysts' data triage cognitive processes, this paper proposes a system that constructs data triage automatons by mining the data triage operations recorded in the traces. To develop the automated data triage system, we proposed a graph-based trace mining method for discovering the critical data characteristic constrains that can be used as rules for data filtering and correlation. Given these rules, finite state machines were constructed to conduct automatic data triage. We evaluated the effectiveness of developing the state machines and their data triage performance on a much larger data set. The result shows that it is feasible to conduct automated data triage by leveraging analysts' data triage traces. The state machines are able to complete processing a large amount of data within minutes. Comparing the performance of automated data triage with the ground truth, we found that a satisfactory false positive rate is achieved, though the false negative rate is relatively higher and yet need further improvement. Besides, another important implication gained from the study is that selecting the traces from experts with better task performance can ultimately improve the performance of the automated data triage system.

VIII. Acknowledgement

This work is supported by MURI grant ARO W911NF-09-1-0525. Peng Liu was supported by ARO W911NF-15-1-0576, ARO W911NF-13-1-0421, and NSF CNS-1422594. Chen Zhong was also supported by IUK Summer Faculty Fellowship.

References

[1] A. D'Amico and K. Whitley, "The real work of computer network defense analysts," in *VizSEC 2007*, pp. 19–37, Springer, 2008.

[2] J. Yen, R. F. Erbacher, C. Zhong, and P. Liu, "Cognitive process," in *Cyber Defense and Situational Awareness*, pp. 119–144, Springer, 2014.

[3] P. Institute, "The state of malware detection and prevention," , Cyphort, 2016.

[4] FireEye, "The total cost of handling too many alerts versus managing risk," , 2016.

[5] R. Sadoddin and A. Ghorbani, "Alert correlation survey: framework and techniques," in *Proceedings of the 2006 international conference on privacy, security and trust: bridge the gap between PST technologies and business services*, pp. 37–38, ACM, 2006.

[6] Y. Zhai, P. Ning, and J. Xu, "Integrating ids alert correlation and os-level dependency tracking," in *ISI*, pp. 272–284, Springer, 2006.

[7] ArcSight, "Building a successful security operations center," , 2010. Research 014-052809-09.

[8] D. Nathans, *Designing and Building Security Operations Center*. Syngress, 2014.

[9] D. Miller, S. Harris, A. Harper, S. VanDyke, and C. Blask, *Security information and event management (SIEM) implementation*. McGraw Hill Professional, 2010.

[10] McAfee, "Siem best practices: Correlation rule and engine debugging," , 2014. Report No. PD25633.

[11] C. Zhong, J. Yen, P. Liu, and R. F. Erbacher, "Automate cybersecurity data triage by leveraging human analysts' cognitive process," in *Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS), 2016 IEEE 2nd International Conference on*, pp. 357–363, IEEE, 2016.

[12] M. R. Endsley, "Toward a theory of situation awareness in dynamic systems," *Human factors*, vol. 37, no. 1, pp. 32–64, 1995.

[13] P.-C. Chen, P. Liu, J. Yen, and T. Mullen, "Experience-based cyber situation recognition using relaxable logic patterns," in *Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), 2012 IEEE International Multi-Disciplinary Conference on*, pp. 243–250, IEEE, 2012.

[14] C. Zhong, D. Samuel, J. Yen, P. Liu, R. Erbacher, S. Hutchinson, R. Etoty, H. Cam, and W. Glodek, "Rankaoh: Context-driven similarity-based retrieval of experiences in cyber analysis," in *Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), 2014 IEEE International Inter-Disciplinary Conference on*, pp. 230–236, IEEE, 2014.

[15] C. J. Garneau, R. F. Erbacher, R. E. Etoty, and S. E. Hutchinson, "Results and lessons learned from a user study of display effectiveness with experienced cyber security network analysts," *Learning from Authoritative Security Experiment Results*, pp. 33–34, 2016.

[16] R. F. Erbacher, D. A. Frincke, P. C. Wong, S. Moody, and G. Fink, "A multi-phase network situational awareness cognitive task analysis," *Information Visualization*, vol. 9, no. 3, pp. 204–219, 2010.

[17] C. Zhong, J. Yen, P. Liu, R. Erbacher, R. Etoty, and C. Garneau, "An integrated computer-aided cognitive task analysis method for tracing cyber-attack analysis processes," in *Proceedings of the 2015 Symposium and Bootcamp on the Science of Security*, pp. 8–9, ACM, 2015.

[18] P. Ning, Y. Cui, and D. S. Reeves, "Constructing attack scenarios through correlation of intrusion alerts," in *Proceedings of the 9th ACM Conference on Computer and Communications Security*, pp. 245–254, ACM, 2002.

[19] F. Cuppens and A. Miege, "Alert correlation in a cooperative intrusion detection framework," in *Security and privacy, 2002. proceedings. 2002 ieee symposium on*, pp. 202–215, IEEE, 2002.

[20] S. J. Templeton and K. Levitt, "A requires/provides model for computer attacks," in *Proceedings of the 2000 workshop on New security paradigms*, pp. 31–38, ACM, 2001.

[21] B. Morin and H. Debar, "Correlation of intrusion symptoms: an application of chronicles," in *RAID*, vol. 3, pp. 94–112, Springer, 2003.

[22] C. Zhong, J. Yen, P. Liu, R. Erbacher, R. Etoty, and C. Garneau, "Arsca: a computer tool for tracing the cognitive processes of cyber-attack analysis," in *Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), 2015 IEEE International Inter-Disciplinary Conference on*, pp. 165–171, IEEE, 2015.

[23] K. Cook, G. Grinstein, M. Whiting, M. Cooper, P. Havig, K. Liggett, B. Nebesh, and C. L. Paul, "Vast challenge 2012: Visual analytics for big data," in *Visual Analytics Science and Technology (VAST), 2012 IEEE Conference on*, pp. 251–255, IEEE, 2012.

[24] G. Torkzadeh and W. J. Doll, "The development of a tool for measuring the perceived impact of information technology on work," *Omega*, vol. 27, no. 3, pp. 327–339, 1999.