2016 IEEE 2nd International Conference on Big Data Security on Cloud, IEEE International Conference on High Performance and Smart Computing, IEEE International Conference on Intelligent Data and Security

Automate Cybersecurity Data Triage by Leveraging Human Analysts' Cognitive Process

Chen Zhong, John Yen, Peng Liu
Pennsylvania State University
{czz111, jyen, pliu}@ist.psu.edu

Robert F. Erbacher
Army Research Laboratory
Robert.Erbacher@gmail.com

***Abstract*—Security Operation Centers rely on data triage to identify the true "signals" from a large volume of noisy alerts and "connect the dots" to answer certain higher-level questions about the attack activities. This work aims to automatically generate data triage automatons directly from cybersecurity analysts' operation traces. Existing methods for generating data triage automatons, including Security Information and Event Management systems (SIEMs), require event correlation rules to be generated by dedicated manual effort from expert analysts. To save analysts' workloads, we propose to "mine" data triage rules out of cybersecurity analysts' operation traces and to use these rules to construct data triage automatons. Our approach may make the cost (of data triage automaton generation) orders of magnitudes smaller. We have designed and implemented the new system and evaluated it through a human-in-the-loop case study. The case study shows that our system can use the analysts' operation traces as input and automatically generate a corresponding state machine for data triage. The operation traces were collected in our previous lab experiment. 29 professional cybersecurity analysts were recruited to analyze a set of IDS alerts and firewall logs. False positive and false negative rates were calculated to evaluate the performance of the data triage state machine by comparing with the ground truth.**

I. Introduction

Due to the cannot-be-underestimated risk caused by cyber-attacks, nowadays the stake of protecting a mission-critical or business-critical enterprise network is already so high that many prominent companies, governments, and military departments must deploy a human-in-the-loop cyber defense system. 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 operating for quite a few years. SOCs employ multiple automated measures to detect cyberattacks, including traffic monitors, firewalls, vulnerability scanners, Intrusion Detection/Prevention System (IDS/IPS), alert correlation, and Security Information and Event Management (SIEM). However, the most sophisticated cyberattack strategies and malware are in fact beyond what these automated attack-analysis measures can handle. Human intelligence, provided by cybersecurity analysts, is playing a critical role in SOCs in "comprehending" the sophisticated attack strategies through advanced correlated diagnosis.

This work was supported by ARO W911NF-15-1-0576, ARO W911NF-13-1-0421 (MURI), and NIETP CAE Cybersecurity Grant.

Human analysts are relied upon heavily to identify the true "signals" from a large volume of noisy IDS alerts and "connect the dots" to answer some higher-level questions about the attack activities, such as "whether the network is under an attack?", "what did the attackers do?", and "what might be their next steps?". They 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 (e.g., IDS alerts, firewall logs, OS audit trails, vulnerability reports, and packet dumps), weeding out the false positives, grouping the related indicator data entries according to the corresponding attack campaigns. Data triage is the most fundamental stage in cyber operations. It 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. In practice, analysts can leverage SIEM systems to get part of the data triage completed.

Data triage is labor-intensive and mostly performed manually by analysts. It's the state of the practice but restricts the efficiency of the analysts generating high-quality incident reports within a limited time. Compared to a computer, human brains have orders of magnitudes smaller data processing throughput. In addition, human beings face unique challenges such as fatigue, anxiety and depression, which a computer would never face. The data, coming from a variety of data sources, are being continuously generated, and thus the volume is overwhelming. 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 are worthy of further analysis. The time pressure is typically huge. The analysts have to resort to day shift to night shift transitions to achieve 24*7 coverage. Due to these challenges, it's usually very difficult for analysts to generate incident reports in a SOC.

This paper aims to take the first step towards automating the human-centric data triage through a new angle. The question to ask is whether analysts' intelligence can be elicited from their operations in performing data triage tasks, and whether the elicited intelligence can be leveraged to build automated system to ease the burden on analysts.

Extensive research has been conducted to assist cybersecu-

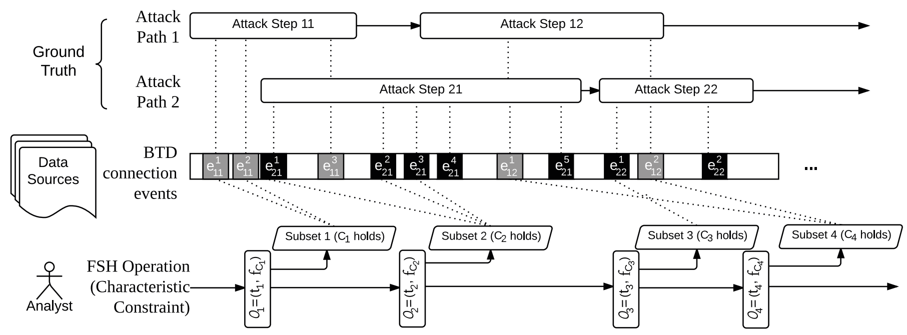

Fig. 1. A scenario of data triage process demonstrates an analyst filtering and correlating the network connection events based on characteristic constrains.

rity analysts. Alert correlation [2], is an important milestone in automated data triage. Alert correlation techniques use heuristic rules (a simple form of automaton) to correlate alerts. However, it is limited by analyzing only one data source (i.e., IDS alerts) while analysts must do cross-data-source analysis in most cases. Alert correlation analysis has later on been integrated with other analysis [3], but the method is still along the heuristic rule angle. Motivated by the benefits of cross-data-source analysis, SIEM systems (e.g., ArcSight [4]) have been focusing on security event correlations across multiple data sources. Although SIEM systems take a big leap forward in generating more powerful data triage automatons, SIEM systems require expert analysts to spend dedicated manual effort on developing the data triage automatons [21], in order to generate a large number of complicated rules (complicated data triage automatons). Therefore, there is a need to make SIEM systems more economically sustainable.

This new human-centric angle sets our work apart from the existing works on generating data triage automatons. Our approach is enabled by three insights. Firstly, it is actually possibly to do non-intrusive tracing of human analysts' concrete data triage operations. Secondly, though challenging, the operation traces could be analyzed in a largely automated way. Thirdly, though challenging, the result of analyzing the traces could be automatically transformed into a state machine, which is essentially a data triage automaton. Accordingly, our approach works as follows. First, an analyst's operation trace can be collected by an existing cognitive tracing toolkit while he/she is performing a data triage task. Second, we use a graph to represent the analyst's data triage operations and their temporal and logical relationships. Third, we mine the constructed graph for patterns. Each pattern defines a class of network connection events belonging to a multistage attack. Fourth, a data triage state machine (DT-SM) can be constructed based on the patterns for automating data triage.

The main contributions are as follows. (a) To the best of our knowledge, this is the first method that generates data triage automatons directly from human analysts' operation traces in an automatic manner. (b) Compared to the existing data triage automaton generation methods, our method may make the cost (of data triage automaton generation) orders of magnitudes smaller. (c) The automatic data triage system is evaluated through a human-in-the-loop case study. With IRB approved, we've recruited 29 professional cybersecurity analysts and captured their traces of operations to construct a DT-SM. (d) We've evaluated the effectiveness of the DT-SM construction and the data triage results based on false positive and false negative rates.

II. Cybersecurity Data Triage

A. A Data Triage Scenario

Data triage is the most fundamental step in cyber defense operations [1]. Data triage analysis 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. Fig. 1 shows a scenario of data triage process, in which analysts perform data triage operations on a set of given data sources to discover attack paths.

In this example (Fig. 1), two attackers attack a network via different attack paths. An *attack path* is a series of steps and each step is performed on the basis of previous steps with the goal of reaching to a final target. The *data sources* collected by the sensors reporting the details of network connection can be generalized to the *network connection events* (will be defined in detail below). A network connection event can be either a malicious event belonging to an attack path or a normal event. An analyst performs *data triage operations* to filter the network connection events to make fast decisions about which connection events could be related to an attack path and worth further investigation. Each analyst's data triage operation filters the data based on a condition on the network connection events. These are critical components in a data triage system and are explained as follows.

B. Network Connection Event

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

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

where $time$ is the time when the connection activity is performed, $conn$ is the type of connection activity (a connection can be "Built", "Teardown", or "Deny"), ip_{src}, $port_{src}$ and ip_{dst}, $port_{dst}$ are the IP address and port of the source and destination respectively, $prot$ 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" connection event and ended with a "Teardown" connection event. "Deny" refers to a failed connection attempt. An attack path is represented as a finite sequence of connection events $(e_1, \ldots, e_n)$.

C. Analysts' Data Triage Operation

Analysts conduct 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, to search using a keyword and to directly select a portion of connection events with common characteristics. All these operations are named "Data Triage Operation". 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).

1) Characteristic Constraint: Each data triage operation filters out a subset of connection events from the original set of connection events specified in the provided data sources. The characteristics of the subset connection events are determined by the constraint defined in the data triage operation. An atomic constraint specifies the value range of an attribute of the connection event,

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

where r_v is the relationship between values, $r_v = \{=, <>, >, <, <=, >=\}$.

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

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

where $T_i (1 \leq i \leq n)$ is an atomic predict. Let $\mathbb{C}$ be a characteristic constraint specified in a data triage operation, and let D be a set of connection events reported in the data sources. 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, $f_{\mathbb{C}} : P(D) \rightarrow P(D)$, 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.

III. The Automatic Data Triage Approach

To capture and leverage human intelligence in data triage, our approach includes four steps. (1) We capture the traces of analysts' operations in performing data triage tasks. Each data triage operation filters the data based on a characteristic constraint that specifies a subset of connection events. (2) A CC-Graph is constructed to represent each analyst's data triage operations and their logical and temporal relationships. (3) We mine the CC-Graphs to construct the attack path patterns, each of which specifies a class of connection events as the traces of a multistage attack. (4) We use these attack path patterns to construct a state machine for automated data triage.

A. Step 1: Identify Data Triage Operations

A toolkit, named ARSCA, has been developed for non-intrusively tracing an analyst's operations in cyberattack analysis [8]. An analyst can conduct FILTER, SEARCH, and SELECT operations when he/she is performing a data triage task in the environment of ARSCA. ARSCA has been tested in an experiment with real analysts [7]. Without reinventing the wheels, we direct adopt ARSCA in this work to capture the traces of analysts' data triage operations. Once the analyst finishes his/her data analysis task, ARSCA outputs a log file, in which the items in the log are in the structural format: `<Item Timestamp=[TIMESTAMP]> [ACTION_TYPE] ([CONTENT]) </item>`. It contains the detailed description of an operation and a time stamp that indicates when the operation was performed. TABLE I shows the items in a ARSCA log that corresponds to the data triage operations.

B. Step 2: Construct Characteristic Constraint Graph

Analysts perform each data triage operation based on a certain characteristic constraint (CC). The logic between the data triage operations illustrates the analyst's underlying data triage strategy.

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 connection events D, we define three types of logical relationships between them based on their characteristic constraints: "is-equal-to", "is-subsumed-by" and "is-complementary-with".

$$\begin{aligned} isEql(O_1, O_2) &\iff \mathbb{C}_1 \leftrightarrow \mathbb{C}_2, \\ isSub(O_1, O_2) &\iff \mathbb{C}_1 \rightarrow \mathbb{C}_2, \\ isCom(O_1, O_2) &\iff \mathbb{C}_1 \rightarrow \neg\mathbb{C}_2 \; and \; C_2 \rightarrow \neg C_1, \end{aligned}$$

where $isEql(\cdot,\cdot)$ and $isCom(\cdot,\cdot)$ are bidirectional relationships but $isSub(\cdot,\cdot)$ is a unidirectional relationship.

2) Characteristic Constraint Graph (CC-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)", $G(trace) =< V, R_l >, V = \{O_1, \ldots, O_n\}$, $R_l \subseteq V * V, l \in \{isEql \succ_t, isSub \succ_t, isCom \succ_t\}$. The nodes of the graph are the data triage operations. A directed edge between two nodes represents a conjunction of a constraint-related relationship and a "happen-after" relationship (i.e. $isEql \succ_t$, $isSub \succ_t$, $isCom \succ_t$), and the label of the edge indicates the types of relationships.

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 happened before it. If an 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 data triage operations may imply the changes of the analyst's focus of attention.

As an example, Fig. 2 is a partial CC-Graph, which is constructed based on an analyst's trace. The complete graph includes 32 nodes

TABLE I
AN EXAMPLE OF ARSCA LOG

Oper-ation	Examples in ARSCA Log	Description

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)

SELECT
+
LINK
	`<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>`
	`<Item Timestamp=07/31`
	`13:20:43>`
	`LINK (Same DstPort)`
	`</Item>`
		Filter the a set Select a set of connection events (i.e. the underl-ined firewall log entries) with a common charact-eristics (i.e. DstPort=6667)

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.

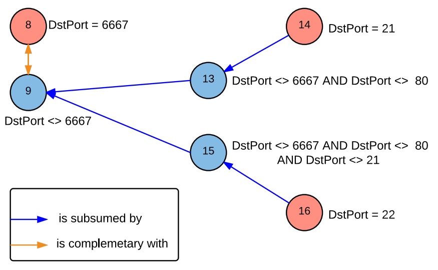

Fig. 2. A partial characteristic constraint graph constructed from an analyst's trace. The nodes are the data triage operations. The node text is the index of their temporal order. The node color refers to whether the data triage operation results in an observation of malicious connection events (red refers to yes, while blue refers to no). The blue edges are the "isSub" relationships, while orange edges are the "isCom" relationships.

and 261 edges. Node 8 is a data triage operation using the constraint "DstPort=6667", which resulted in a subset of data which were identified by the analyst as suspicious events (represented in red color). After that, the analyst then screened out the connection 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 connection events with other characteristics after investigating the connection 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 means the analyst gradually screened out the connection events to narrow down the scope. The data triage operation of Node 14 is the end of this narrowing process and it lets the analyst generate another hypothesis. Similarly, the sequence of Node 9, Node 15 and Node 16 indicates the same strategy used by the analyst.

C. Step 3: Mine the Characteristic Constraint Graphs

A CC-Graph contains rich information about the strategy used by an analyst in data triage, which can answer the following questions. (1) Given a set of data triage operations, which one is critical for the analyst to perform effective triage? (2) Given the critical data triage operations, what sequence these operations should be performed to detect the events of an attack path? Given a CC-Graph $G =< V, R_l >$, two steps are taken to elicit such key characteristic constraints from the CC-Graphs.

1) Extract the Critical Endpoints in "isSub" Subgraphs: Given data triage operations O_1 and O_2, $isSub \succ_t (O_2, O_1)$ indicates that O_2 is performed after O_1, and O_2 further narrows the O_1's connection event set by using a more strict characteristic constraint. As the case described in Fig. 2, analysts may conduct a series of data triage operations with the "is-subsumed-by" (isSub) relationships to gradually narrow down the connection events. The endpoints of such processes can be viewed as the critical characteristic constraints that represent a noteworthy set of connection events. To investigate the endpoints, we consider the subgraph that only consists of the edges of the relationship $isSub \succ_t$ and $isEql \succ_t$ (isEql is a special case of isSub) and name it "isSub" subgraph.

2) Mutually Exclusive Characteristic Constraints: It's desirable to have the endpoints of an "isSub" subgraph be mutually exclusive. To ensure the nodes to be mutually exclusive, we examine the logical 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 considering the edges of "isCom" relationships in $G_{induced}$, we apply the Bron-Kerbosch algorithm [9] to find all the maximum cliques in $G_{induced}$. A maximum clique is a subset of the nodes, 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 has an overlap with a $v_j \notin V_{clique}$, iff there exists a connection event e satisfies both $\mathbb{C}_i$ and $\mathbb{C}_j$. To remove the overlap, we adjust the characteristic constraint used in v_i (say $\mathbb{C}_i$) to $\mathbb{C}_i \wedge \neg\mathbb{C}_j$. If $\mathbb{C}_i \wedge \neg\mathbb{C}_j$ is not false, we add the adjusted v_i to $V_{clique}(V_{clique} \in \mathcal{G}_C)$.

D. Step 4: State Machine Based on Attack Path Pattern

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

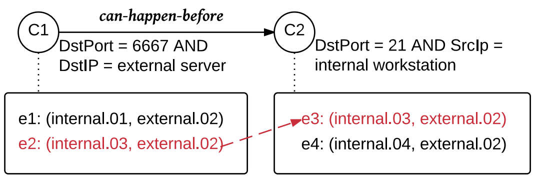
can-happen-before
C1
C2
DstPort = 6667 AND
DstIP = external server
DstPort = 21 AND SrcIp =
internal workstation
e1: (internal.01, external.02)
e2: (internal.03, external.02)
e3: (internal.03, external.02)
e4: (internal.04, external.02)

Fig. 3. The connection events (in the boxes) that satisfy the characteristic constraints (represented by cycles) in an attack path pattern can be further correlated based on connections between IPs. (The IP addresses are coded as "internal.XX" or "external.XX".

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 by $\succ_{\mathbb{C}t}(\cdot,\cdot)$. 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 that a set of connection events that satisfy $\mathbb{C}_1$ occur before the other set of events involving the same hosts that satisfy $\mathbb{C}_2$ 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 connection events in the task performed by the analyst. Assume that the connection events come in sequence over time $E=(e_1,\dots,e_n)$, we say that the sequence of connection events satisfy a sequence of characteristic constraints $(\mathbb{C}_1,\dots,\mathbb{C}_m)$ defined in an attack path pattern $\mathcal{G}_{\mathbb{C}}$ iff $\forall \mathbb{C}_i(1\le i\le m), \exists e_p(1\le p\le n)$ 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 $(\mathbb{C}_1,\dots,\mathbb{C}_m)$ and E, the attack path instances detected based on $(\mathbb{C}_1,\dots,\mathbb{C}_m)$ is a sequence of connection event sets, $attack_{(\mathbb{C}_1,\dots,\mathbb{C}_m)}(\mathcal{E})=\{(\mathcal{E}_1,\dots,\mathcal{E}_m)\}$, where $\mathcal{E}_{ik}(1\le i\le m)=\{e_{ik}|e_{ik}$ *is a connection event from* $source_{ik1}$ to $destination_{ik2}$ *that satisfies* $C_i\}$.

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

E. Step 5: Post-precessing

To enable analysts review and modify the result of data triage, the output of the automated data triage system should be readable for analysts. Therefore, a post-processing is necessary to aggregate the DT-SM's output. We leverage a heuristic to correlate the output instances based on the linkage of source IP and destination IP of the connection events: the attacker or the infected machines are likely to be involved in multiple steps in an attack path. Therefore, given a characteristic constraints $\mathbb{C}_1$, the IP addresses that appear in the connection events that satisfy $\mathbb{C}_1$ is very likely to appear in the connection events that satisfy the other characteristic constraint $\mathbb{C}_2$ that $\succ_{\mathbb{C}t}(\mathbb{C}_2,\mathbb{C}_1)$. In the case shown in Fig. 3, we correlate e_2 and e_3 as a refined instance, which indicating an attack path involving the pair of hosts ("internal.03" and "external.02").

"Occurrence rate" is calculated for each pair of hosts that are involved in the connection events (as source and destination) of the instances. The value range of the occurrence rate is [0, 1]. It can be calculated as follows. Given an instantiated characteristic constraint sequence $(\mathbb{C}_1,\dots,\mathbb{C}_m)$, and a pair of hosts (ip_1,ip_2), if there are α sets of connection events, each of which satisfy a characteristic constraint in $(\mathbb{C}_1,\dots,\mathbb{C}_m)$, and the pair (ip_1,ip_2) appears in all the α connection event sets, we say the occurrence rate is α/m.

Based on the heuristic and the definition of occurrence rate, we correlate the attack path instances in DT-SM's output by the IP linkage. We set up a value of occurrence rate as the threshold. Any pair of hosts whose occurrence rate is lower than the threshold will not be linked in the instances. If the threshold is too large, it will screen out too many instances and result in a small number of output instances. However, if it is too small, the post-processing will results in too many instances which may annoy analysts. Therefore, it's critical to decide an appropriate value of the threshold based on the prior knowledge.

IV. Evaluation

We implemented the intelligent data triage system. To evaluate it, we focus on these aspects: (1) the effectiveness of the DT-SM construction, and (2) the performance of the DT-SM on data triage with a selected threshold of the occurrence rate.

A. Experiment Dataset

1) ARSCA Logs Collected from a Lab Experiment: With the IRB approved, we recruited 30 full-time professional cybersecurity analysts in an experiment. The analysts were asked to accomplish a cybersecurity data triage task using ARSCA. ARSCA audited the their operations while they were performing the task [7]. We decided to keep 29 trace logs after manually reviewing them. In total, we collected 29 ARSCA logs with 1104 items. The average log length is 31.17.

2) Task Data Sources and Scenario: The cyberattack analysis task is designed based on the cyber situation awareness task in VAST Challenge 2012 with a setting of a bank's network with approximately 5000 machines. The VAST challenge provides participants with IDS alerts and firewall log. 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 [5]. 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 connection event, 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 our limited task time. Therefore, only small portions of data were selected and provided to the participants to enable them to complete the task in the experimental time (60 minutes). The selected task data sources include 239 IDS alerts and 115,524 firewall logs, reporting the network connection events that happened in a 10-minute time window (4/5 22:15-22:25). The task data sources contain the evidence of three attack steps: (1) IRC communication between the internal workstations with a set of external Command and Control (C&C) servers, (2) denied data exfiltration attempts using FTP connections, and (3) successful data exfiltration using SSH connections.

B. Effectiveness of DT-SM Construction

The effectiveness of DT-SM construction is determined by three intermediate results: (1) the identifed data triage operations, (2) the CC-Graph constructed, and (3) the attack path pattern constructed from the CC-Graphs.

We first evaluated the accuracy of the automatic 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 identifer, and the other is the evaluator: given an ARSCA log, the main coder first read throught the trace and specified the data triage operations in the log; the evaluator then read the trace for the second time and proofreaded the data triage operations identifed by the main identifier; everytime when a disagreement occured, 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 system identified 358 data triage operations in total. Comparing them with the ground truth, there are 332 data triage operations correctly identified by the system. There are 62 operations are not identified by the system. Therefore, the false positive rate is 0.073 and the false negative rate is 0.157.

Based on the automatically identified data triage operations, 29 CC-Graphs are constructed. The average number of nodes is 34.3. The average number of edges is 83.6, including 12.6% isEql edges, 23.6% isSub edges, and 63.8% isCom edges. The dominant percentage of isCom edges shows that most data triage operations have a mutually exclusive relationship.

To evaluate the effectiveness of attack path pattern construction, we constructed an attack path pattern for each CC-Graph. We had to exclude 5 pieces of ARSCA log because less than 2 data triage operations were identified in these traces. 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 the characteristic constraints in them 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 gathered in the post-task questionnaires 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 observations (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. The most common characteristic constraints included in the attack path patterns are `` ``DSTPORT = 6667 AND PRIORITY = Info AND PROTOCOL = TCP AND SERVICE = 6667_tcp'', ``SRCIP = 172.23.235.* AND DSTIP = 10.32.5.* AND DSTPORT = 21 AND OPERATION = Deny AND PRIORITY = Warning AND SERVICE = ftp'', ``SRCIP = 172.23.234.* AND DSTIP = 10.32.5.* AND DSTPORT = 22''. ``

C. Performance of DT-SM

1) Testing Data: We want to evaluate the performance of the DT-SM on the data sources that are much larger than the data sources that were used for constructing them (the data sources analyzed by the analysts when generating ARSCA logs). 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 get the testing data sources from the original VAST data. The testing data set contains 1,810 IDS alerts and 599, 489 firewall log, without overlap with the task data set.

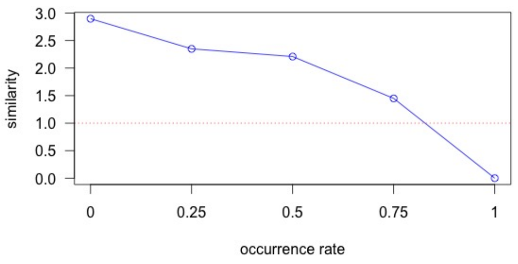

Fig. 4. The similarity value changes with different IP occurrence rate.

Although the VAST Challenge provides an attack description, it's still not apparent regarding whether each alert/log entry in the data sources is related to attack or not. To determine the ground truth before evaluating the DT-SM's data triage result, we manually processed the testing data sources.

2) Selecting the Occurrence Rate for Post-processing: We need to choose an occurrence rate as the threshold in post-processing. We select the data sources of another 10-minute time window from the original VAST data set (4/5 18:05-18:15) for determining the occurrence rate We first identify the instances in the ground truth based on the given attack scenario, and then test different values of the occurrence rate and choose the one that can result in the instances similar to the instances in ground truth. The similarity is calculated by:

$$similarity = \frac{1}{|(ip1, ip2)|} * \sum_{(ip1,ip2) in GT} \frac{\#\ of\ (ip1, ip2) in\ SM\ instances}{\#\ of\ (ip1, ip2) in\ GT\ instances}$$

Given the output of DT-SM running on the selected data sources, we tested four values of the IP occurrence rate in post-processing: 0, 0.25, 0.5, 0.75 and 1. Fig. 4 shows the similarity values for each occurrence rate. The similarity is closest to 1 when we select 0.75 as the occurrence rate.

3) False Positives and False Negatives: The performance of the DT-SM is measured by false positive and false negative rates. We use e^{SM} to denote the network connection events in the instances outputted by the DT-SM and e^{T} to denote the connection events included in the ground truth. We calculate the false positive and false negative rate using the following formula:

$$false\ positive = 1 - \eta \sum_{instance(SM)} \frac{\sum_{(ip1,ip2)} |\{e^{SM} | e^{SM} = e^{T}, e^{SM} \in instance\}|}{\sum_{(ip1,ip2)} |\{e^{SM} | e^{SM} \in instance\}|}$$

$$\eta = \frac{1}{|attack\ path\ instance|}$$

$$false\ negative = 1 - \sum_{instance(GT)} \frac{\sum_{(ip1,ip2)} |\{e^{SM} | e^{SM} = e^{T}, e^{T} \in instance\}|}{2}$$

where $instance(SM)$ refers to the attack path instances in the DT-SM output, $instance(GT)$ is the attack path instances in the ground truth. $\{ip1, ip2\}$ refers to an unordered pair of hosts involved in an connection event, either of which could be the source.

We ran the DT-SM constructed from the operation traces of the 29 participates over the testing data (1,810 IDS alerts and 599,489 firewall log). It takes DT-SM less than 1 second to finish the processing the data which usually takes analysts hours to process. The false positive rate is 0.025 and the false negative rate is 0.272.

The false positive is satisfactory, but the false negative still needs to be improved. However, the low recall rate isn't caused by the limitation of the approach but the limitation of the experiment setting. The traces of analysts' data triage operations in analyzing 10-minute-time-window events cannot fully embody all the expertise needed for analyzing 100-minute-time-window events. Therefore, the attack path pattern constructed based on the data triage operation traces can't perfectly solve a broader problem (i.e., triage the events in the 100-minute time window). In the real-world practice, analysts keep analyzing the continuously incoming data sources. As long as ARSCA is deployed, it can automatically generate new logs, and therefore the ARSCA logs can be continuously imported into our data triage system and update the attack path pattern accordingly. It can result in more comprehensive attack path patterns because a sufficient set of endpoints in the "isSub" subgraph can be identified, and thus may increase the recall rate the DT-SM.

V. RELATED WORK

Researchers have recognized the significant role of human analysts in cyber analysis. Several exciting field studies have focused on understanding analysts' analytical processes [1], [7], [11]. Our current work is closely related to the ARSCA tool of Zhong et al. [8] because we directly borrow the operation traces collected by this tool (i.e., ARSCA) as the input, instead of reinventing the wheel. 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.

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 using a wide range of technologies. Sadoddin and Ghorbani [2] provided a good survey that classifies the state-of-the-art alert correlation techniques, each of which has pros and cons. It 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., [14]–[16]; attack scenarios models (e.g., temporal logic formalism [17]) 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 DT-SM has three merits: (1) It eases analysts' burden by applying the data constraints they used previously to triage new data so that analysts can spend more effort on more detailed investigation or handling new challenging problems, thus improving the overall quality of incident reports in an SOC. (2) It's by natural enterprice specific and network specific. One DT-SM constructed from one analyst's operation trace could be found directly useful by another analyst. (3) The DT-SM can suit the highly dynamic cyber environment because it takes the input of analysts' operation traces. These traces can be collected continuously as long as the capturing toolkit can be launched while analysts are performing cybersecurity analysis tasks.

VI. CONCLUSION

Aimed to ease cybersecurity analysts' burden in tedious data triage analysis, we developed an automated system which can generate data triage automatons directly from cybersecurity analysts' operation traces. We have designed and implemented the new system and evaluated it through a human-in-the-loop case study. This work shows that it is feasible to elicit attack path patterns by modeling and mining the traces of analysts' data triage operations.

REFERENCES

[1] DAmico, Anita, and Kirsten Whitley. *The real work of computer network defense analysts.* In VizSEC 2007, pp. 19-37. Springer Berlin Heidelberg, 2008.

[2] Sadoddin, Reza, and Ali 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, p. 37. ACM, 2006.

[3] Zhai, Yan, Peng Ning, and Jun Xu. *Integrating IDS alert correlation and OS-level dependency tracking.* In Intelligence and Security Informatics, pp. 272-284. Springer Berlin Heidelberg, 2006.

[4] E. S. M. ArcSight, *ArcSight ESM white paper,* Online, 2010.

[5] Cook, Kristin, Georges Grinstein, Mark Whiting, Michael Cooper, Paul Havig, Kristen Liggett, Bohdan Nebesh, and Celeste Lyn 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.

[6] Cheung, Steven, Ulf Lindqvist, and Martin W. Fong. *Modeling multistep cyber attacks for scenario recognition.* In DARPA information survivability conference and exposition, 2003. Proceedings, vol. 1, pp. 284-292. IEEE, 2003.

[7] Zhong, Chen, John Yen, Peng Liu, Rob Erbacher, Renee Etoty, and Christopher 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, p. 9. ACM, 2015.

[8] Zhong, Chen, John Yen, Peng Liu, Rob Erbacher, Renee Etoty, and Christopher 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.

[9] D. R. Aloysius, *Bron-Kerbosch Algorithm,* 2012.

[10] Csardi, Gabor, and Tamas Nepusz. *The igraph software package for complex network research.* InterJournal, Complex Systems 1695, no. 5 (2006): 1-9.

[11] Erbacher, Robert F., Deborah A. Frincke, Pak Chung Wong, Sarah Moody, and Glenn Fink. *A multi-phase network situational awareness cognitive task analysis.* Information Visualization 9, no. 3 (2010): 204-219.

[12] Goodall, John R., Wayne G. Lutters, and Anita Komlodi. *Developing expertise for network intrusion detection.* Information Technology People 22, no. 2 (2009): 92-108.

[13] Chen, Po-Chun, Peng Liu, John Yen, and Tracy 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] Ning, Peng, Yun Cui, and Douglas 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.

[15] Cuppens, Frdric, and Alexandre Miege. *Alert correlation in a cooperative intrusion detection framework.* In Security and Privacy, 2002. Proceedings. 2002 IEEE Symposium on, pp. 202-215. IEEE, 2002.

[16] Templeton, Steven J., and Karl Levitt. *A requires/provides model for computer attacks.* In Proceedings of the 2000 workshop on New security paradigms, pp. 31-38. ACM, 2001.

[17] Morin, Benjamin, and Herv Debar. *Correlation of intrusion symptoms: an application of chronicles.* In Recent Advances in Intrusion Detection, pp. 94-112. Springer Berlin Heidelberg, 2003.

[18] Dain, Oliver, and Robert K. Cunningham. *Fusing a heterogeneous alert stream into scenarios.* In Proceedings of the 2001 ACM workshop on Data Mining for Security Applications, vol. 13. 2001.

[19] Smith, Reuben, Nathalie Japkowicz, Maxwell Dondo, and Peter Mason. *Using unsupervised learning for network alert correlation.* In Advances in Artificial Intelligence, pp. 308-319. Springer Berlin Heidelberg, 2008.

[20] Sommer, Robin, and Vern Paxson. *Outside the closed world: On using machine learning for network intrusion detection.* In Security and Privacy (SP), 2010 IEEE Symposium on, pp. 305-316. IEEE, 2010.

[21] Granadillo, Gustavo Daniel Gonzalez. "Optimization of cost-based threat response for Security Information and Event Management (SIEM) systems." PhD diss., Institut National des Tlcommunications, 2013.