Understanding the Context of Network Traffic Alerts

Bram C.M. Cappers* Jarke J. van Wijk†

Department of Mathematics and Computer Science, Eindhoven University of Technology, The Netherlands

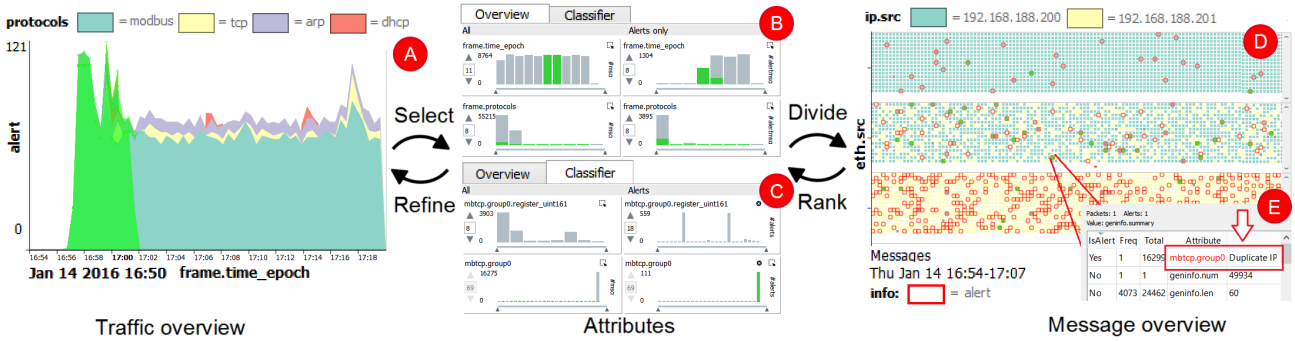

Figure 1: The discovery of Man-in-the-Middle behavior in network traffic meta-data using selection-based attribute ranking.

Abstract

For the protection of critical infrastructures against complex virus attacks, automated network traffic analysis and deep packet inspection are unavoidable. However, even with the use of network intrusion detection systems, the number of alerts is still too large to analyze manually. In addition, the discovery of domain-specific multi stage viruses (e.g., Advanced Persistent Threats) are typically not captured by a single alert. The result is that security experts are overloaded with low-level technical alerts where they must look for the presence of an APT. In this paper we propose an alert-oriented visual analytics approach for the exploration of network traffic content in multiple contexts. In our approach CoNTA (Contextual analysis of Network Traffic Alerts), experts are supported to discover threats in large alert collections through interactive exploration using selections and attributes of interest. Tight integration between machine learning and visualization enables experts to quickly drill down into the alert collection and report false alerts back to the intrusion detection system. Finally, we show the effectiveness of the approach by applying it on real world and artificial data sets.

Keywords: Anomaly detection, network traffic analysis, multivariate analysis, streaming data, interaction, parse data analysis.

Index Terms: C.2.0 [Computer Communication Networks]: General—Security and Protection; H.5.2 [Information Interfaces and Presentation]: User Interfaces; I.3.8 [Computer Graphics]: Applications

1 Introduction

The aim of network forensics is to discover malicious activity inside logs of network traffic. Especially for critical infrastructures, such as power plants, the presence of malicious activity can lead to the malfunction or even destruction of the underlying system. Forensics can no longer limit their analysis to high-level message properties (e.g., length, destination) due to the existence of Advanced Persistent Threats (APTs) [28]. These complex viruses are designed to hide their malicious activity inside the content of messages thereby making them invisible to current flow-based techniques [16].

*e-mail: b.c.m.cappers@tue.nl

†e-mail:j.j.v.wijk@tue.nl

Since manual inspection of network traffic is impossible due to size and complexity, forensic experts use network Intrusion Detection Systems (IDS) to assist them in finding areas of interest. Although these systems automate the analysis of network traffic, the number of (false) alerts is often too large to analyze one by one. Given that alerts in message content analysis can arise at any combination of hundreds of message attributes, the number of alert types greatly varies. In this paper we propose an exploration method that enables experts to gain insight in traffic content by visually exploring and correlating network traffic alerts defined at message-level.

Similar to Livnat et al. [17], we believe that an alert as a result of a complex attack does not stand on its own. The true severity of an alert can not be determined by solely inspecting its structural properties such as what, when, or where that alert has occurred in the network. Instead, we are interested whether the occurrence of an alert was implicitly related to (a collection of) messages or alerts that were sent in the past. For this we need to be able to inspect message collections for correlations between message attributes (e.g., inter-attribute analysis) and inspect trends in these attributes over periods of time, (e.g., intra-attribute analysis). To enable the simultaneous exploration of message-level phenomena (e.g., field misuse) and traffic-level phenomena (e.g., bursts), our exploration method focuses on a tight interaction scheme between well-established visualization and machine learning techniques. In summary, our main contributions are:

- a visual analytics approach to network forensics, enabling experts to:
  - explore and analyze network traffic on both attribute and temporal level using alerts as a ground truth, and
  - identify and confirm (visual) correlations between network traffic messages and alerts using selection-based relevance metrics and conversation analysis.
- a data-driven coupling between machine learning and visualization for the detection and refinement of network alerts.

The paper is structured as follows. First, related work is discussed in Section 2. Next, the scope and approach for traffic analysis are discussed in Sections 3 and 4 respectively. Section 5 presents an overview of the system and shows how the exploration method is applied. Sections 6 and 7 describe two example explorations on real world and artificial data sets and discuss the limitations of the approach. Conclusions and future work are presented in Section 8.

978-1-5090-1605-1/16/$31.00

2 Related Work

A wide range of visualization techniques have been proposed over the years to explore network traffic. We focus here on the approaches that use alerts as a central element. For a broader overview we refer to the surveys of Shiravi et al. [27] and Attipoe et al. [3].

2.1 Alert Visualization

Alert visualizations are designed to gain insight into large alert collections, generated by detection systems such as Snort [4] and Bro [20]. Some examples of well-known visualizations are:

- IDS rainstorm [1] visualizes the severity of Snort intrusion detection alerts by creating a pixel visualization of the IP address space where the alerts reside.
- Snortview [13] visualizes Snort alerts over time according to their type, source, and destination. Glyphs and coloring are used to effectively represent false positives.
- Avisa [26] uses a radial display to visualize the relationship between alert types and hosts. Alerts are visualized as B-Splines from alert type to the corresponding host clustered using edge bundling techniques.

Such methods construct an overview of alert collections by visually encoding their severity, source, and type in a single image. Since these methods only focus on alerts as their data source, their knowledge about the normal traffic is limited, making root-cause analysis on this data very difficult. In addition, the loose coupling between IDS and visualization does not enable experts to report false alerts back to the IDS. We believe that a human in the loop approach [24] is vital for quickly gaining insight and reducing the (false) alerts.

Methods that do incorporate normal traffic and interactive machine learning in their exploration process are PixelCarpet [15] and SNAPS [5]. PixelCarpet uses a pixel visualization where log entries are represented as a stack of pixels. The brightness of a pixel is used to denote the frequency of log record values. Tight coupling between machine learning and visualization is achieved by enabling the user to remove records from the data set and adapt the model accordingly. Although the technique assists experts in identifying areas of interest through interactive machine learning, their method does not scale for hundreds of attributes. The SNAPS system uses a pixel visualization to display the full structure of a network message as a horizontal line of pixels. Alerts inside messages are highlighted on a per-attribute basis and can be refined using machine learning. Unfortunately, since the approach is focused on monitoring traffic it can only inspect small fractions of traffic at the same time. This makes it hard to detect attacks over larger periods in time.

2.2 Exploration

In order to understand the severity and cause of an alert, investigations are needed. Zhang et al. [32] already showed that in flow-based network investigations interaction and multiple views play an important role. In our CoNTA approach, we show that this paradigm can be extended with machine learning and relevance metrics to enable traffic *content* analysis for hundreds of attributes.

Two systems closest to our technique with respect to exploration are VisAlert [17] and Ocelot [2]. VisAlert discovers correlation between network IDS alerts by visually mapping alerts according to three attributes, namely *what*, *when*, and *where*. They use a radial layout and semantic zooming to find overlap between alerts at various levels of detail over time. Ocelot improves decision support for cyber analysts in computer networks by hierarchically grouping host machines according to various attributes. Filtering is used to isolate affected machines from healthy parts of the network.

VizAlert and Ocelot analyze alerts at the level of a host, rather than at the level of a message. Since host-based alerts only convey information about the network-level constraints that have been violated (e.g., policy violations, access attempts, etc.), finding the messages and values that were responsible for these alerts is difficult. In addition, since both methods do not consider the normal events in their decision support analysis, context is lost making it even more difficult to find the root cause of an alert. Finally, both methods do not enable experts to inspect the sequential occurrence of one or more alerts. Kot et al. [14] already indicate that APTs can be the result of a sequence of malicious actions. CoNTA enables experts to visually inspect traffic sequences by inspecting message attributes at the level of network conversations. Interaction enables experts to store and inspect search results in different contexts.

In summary, current methodologies focus on the visualization of large alert collections rather than trying to investigate them through iterative refinement and correlation discovery. The methods that try to discover correlations between alerts and traffic events, either cannot report their findings back to the IDS or limit their analysis to only a few flow-based attributes. Furthermore, their inability to inspect sequential patterns and conversations in normal traffic does not give experts a baseline for determining the severity of an alert.

3 Problem Statement

The exploration and analysis of network traffic is still a challenge. Even for relatively small networks consisting of tens of nodes, the number of messages per day can easily run in the order of thousands. In addition, every message stores multivariate data depending on its type and purpose. In order to protect environments from APTs, network traffic content has to be analyzed. APTs tend to work in three stages, namely *infiltration*, *expansion* and *exploitation*. Infiltration is typically achieved through social engineering [29]. During expansion the threat will try to locate the target machine. The exploitation phase is used to sabotage the system by relying on system vulnerabilities. Since APTs exploit domain-specific properties of the network, infiltration is nearly impossible to prevent. We can however detect signs of the other phases by analyzing unencrypted network traffic between hosts for anomalies.

We focus on potential APTs against assets in industrial control systems and office networks, since the modification of assets (e.g., hardware and files) in these networks can have severe consequences. We can identify two types of threats against these assets, namely *system-related* versus *process-related* threats [10]. System-related threats create malicious traffic at network-level and are typically caused by traditional attacks such as buffer overflows and data tampering, possibly assisted by port scans or complex Man In The Middle behavior (MITM) [18]. Process-related threats generate traffic that is legitimate at network-level, but malicious at the level of assets. Examples are raising the temperature of a heater to a 1000 degrees or unauthorized file access. System-related threats are often used as a first step in the realization of a process-related threat. Process-related attacks are typically the result of an APT.

For the detection of process-related threats, we analyzed application-level protocols SMB2 and ModBus. The Modbus protocol is designed to transfer low-level hardware commands in industrial control systems, whereas SMB2 is used for file management in office networks. System-related attacks can be detected by analyzing flow-based protocols ETH, TCP, and IP.

3.1 Data acquisition

CoNTA applies semantic network traffic analysis by enriching raw network packets with protocol semantics using WireShark [6]. The result is a multivariate table where rows correspond to messages and columns to attributes. Protocol attributes can represent numerical ranges (e.g., port numbers), strings (e.g., ip addresses), or boolean values (e.g., flag data). Depending on the type of message, specific attributes are present. Since the number of possible protocol attributes outnumbers the number of possible attributes in a message (order of 100s vs. 10s), the resulting table is quite sparse (Figure 2).

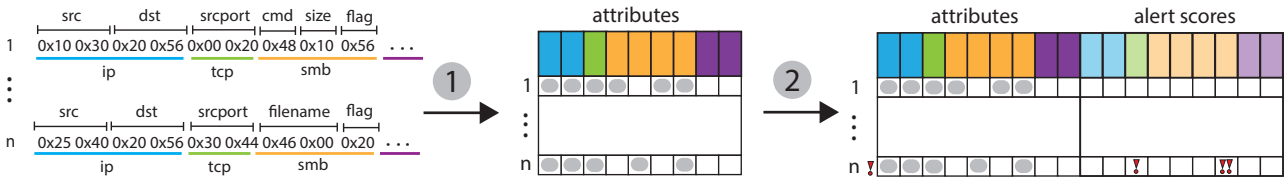

Figure 2: 1) Serialization of PCAP traffic with WireShark. 2) Machine learning produces scores per attribute whether these are suspicious or not.

For the classification of anomalous behavior in the traffic, we aim for an anomaly detection approach, since APT activity is typically not captured by existing signature based approaches [8]. The resulting alerts are used as a basis for the analysis of network traffic. Without loss of generality, we can model an alert as a weighted vector of message values, where the weights describe the extent to which the IDS considers that value malicious. More formally, let D be our multivariate table with N messages and M attributes.

$$D = \{m_{ij}, i = 1,\ldots,N; j = 1,\ldots,M\} \quad (1)$$

where m_{ij} represents the message value of message i at attribute A_j. Furthermore, let S represent a table of alert data such that:

$$S = \{s_{ij} \in [0,1], i = 1,\ldots,N; j = 1,\ldots,M\} \quad (2)$$

where s_{ij} represents an IDS *alert score* for value m_{ij}. A message m_i is considered *malicious* if and only if:

$$\exists_j [j = 1,\ldots,M | s_{ij} > \tau] \quad (3)$$

where τ is a decision threshold that for the sake of simplicity is set to 0.95. This model enables us to define classifier refinement techniques that are independent of the underlying machine learning (see Section 4.6). Classifiers that do not directly support probability based classification results can obtain these through posterior probability estimation [12].

Network messages are classified using a probabilistic based IDS for industrial control systems as defined by Yüksel et al. [31]. This classification technique maintains histograms on a per-attribute basis and uses dynamic thresholds to determine the severity of an alert. For the detection of contextual anomalies involving combinations of values, they derive new attributes using domain knowledge and Pearson's Chi-Square test [21] for statistical independence.

4 CoNTA

The size and complexity of network traffic data makes digital network forensics a challenging task. Especially when trying to find the root cause of high-level anomalies, the lack of traffic content can severely limit the investigation. To enable experts in discovering anomalies in this data, we aim for a scalable and interactive visual analytics approach that is coherent with the workflow of traditional digital forensics and cyber defence models [7, 33].

We tackle scalability by summarizing network traffic in a configurable table, enabling experts to inspect trends and outliers from various perspectives by splitting the data over multiple rows and columns. Detailed exploration is achieved by storing message selections as contexts and inspecting them with respect to these contexts. To handle the large amount of attributes in the data, attributes are represented as scented widgets [30] that can be ranked and filtered according to characteristics in selected message collections. Large alert collections are tackled by reporting false classification results back to machine learning and enabling experts to analyze alerts from the viewpoint of both messages and attributes.

4.1 Exploration process

Figure 3 shows a schematic overview of the exploration process. Similar to Pollitt's model [22], our approach considers three phases,

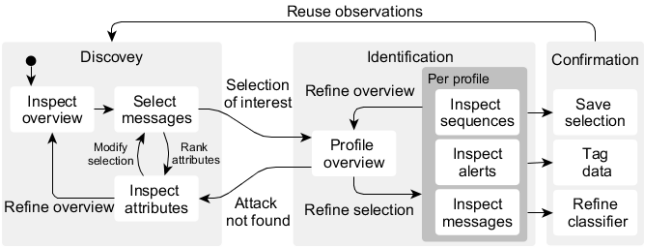

Figure 3: The CoNTA workflow model for network traffic exploration. Experts use alerts and a traffic overview to find areas of interest over time in up to three attributes. Selected areas are refined by (de)selecting messages according to suggested attributes of interest. Identification of the underlying problem is achieved by splitting the traffic according to profiles and testing the presence of message values and sequential patterns in multiple contexts. New contexts and attributes are obtained by saving selections of interests. Experts can use this to keep track of their exploration process, compare selections in different contexts, or report false alerts back to the IDS.

namely *discovery*, *identification*, and *confirmation*. In the *discovery* phase, experts search the data for areas of interest using alerts as a starting point. In order to determine when a particular subset of alerts is of interest, experts need to find any form of similarity (e.g., originating from the same source, sharing the same attribute etc.) between alerts. For this experts need to be able to:

- compare traffic between multiple entities to discover oultiers;
- locate trends and sequences in attributes over time; and
- inspect messages individually to determine their severity.

In CoNTA, experts start their analysis with an *overview* of the traffic over time on any desired attribute of choice. *Suggestions* for possible interesting attributes are provided in the attribute view using selection-based ranking techniques (Section 4.5).

The *identification* phase tries to locate potential causes of the selected data by splitting the traffic in one or more groups (also referred to as profiles [19]) and inspect them in various *contexts*. This involves creating *hypotheses* and verifying them by testing the data for *structural* properties, such as when and who produced the data and what data were accessed. The inspection of *sequential* properties enables experts to find malicious message orderings in conversations. This includes the detection of conflicts of interest (e.g., approving your own file requests), and violations in operational integrity (e.g., closing the gas valve before lighting a fire). *Confirmation* is the phase where conclusions are drawn from the hypotheses. This either results in:

- storing selections into contexts for *reuse* in investigations;
- *tagging* traffic with new data to accelerate analysis; and/or
- *reporting* false alerts back to the machine learning by retraining the data on subsets of the traffic.

CoNTA uses four linked views to assist experts throughout the three exploration phases. In the next sections we discuss the functionality and design decisions for each view separately. For a demonstration of the system in practice, we refer to the supplementary video [1].

[1] https://www.youtube.com/watch?v=yOXDZYKCKZ0

Figure 4: Graphical user interface of the implemented prototype and components: a) Time table for the visual inspection of correlations between 3 attributes over time. Settings with respect to the type of cell visualization (i.e., heatmaps, line charts or pixel maps), axis scaling, and ordering are set using controls in e). b) The context view enables experts to revisit their exploration process and visually compare selections. c) The attribute view shows trends and patterns in selections on a per-attribute basis. Settings with respect to attribute ordering, binning, and classifier settings can be adjusted using the controls surrounding the view. d) Temporal patterns in network conversations can be discovered in the conversation view enabling the expert to inspect the possible presence of malicious messages sequences.

4.2 Time Table

When analyzing network traffic, the number of messages is typically larger than the number of available pixels on the screen. To provide an overview of the traffic we use a table, where attributes can be inspected over time by grouping the network traffic over at most two attributes of choice. The main motivation for introducing a table of small multiples over a large single is to assist analysts in profiling, where they can inspect traffic with respect to certain attribute values. This enables experts for instance to spot trends or compare traffic over time on a per user or daily basis. For the analysis of the traffic as one large single, the axes of the table can be set to `None`. Figure 5 shows a schematic overview of how the time table can be configured. Since repeatedly printing the axis labels for every table cell separately is redundant and can clutter the visualization, a small legend is used instead to inform the expert about the active axes and scaling (Figure 4a). For the detection of similarities between one or more table cells, a third attribute of choice can be visualized using color.

The table axes enable experts to analyze categorical and numerical attributes by binning the traffic in non-overlapping intervals. The axes inside each table cell can be used to inspect continuous attributes, such as time or message length. Since time plays a key role in network analysis, the cell's X-axis is always set to time.

Experts can modify the bin sizes of the table and cell axes depending on their task and available space. Small bin sizes are suitable for outlier detection at pixel-level, whereas larger bin sizes can be used for the detection of temporal patterns over larger periods in time. For the detection of patterns between bins, experts can sort axis values by their frequency or rarity. If the number of bins in an attribute becomes too large, experts can enable scroll bars by defining an upperbound on the number of visible bins. Predominating bins can be shown or hidden using the control options (Figure 6a).

4.2.1 Table cells

Following from the problem statement, analysts need to be able to compare traffic between profiles, inspect attributes over time, and

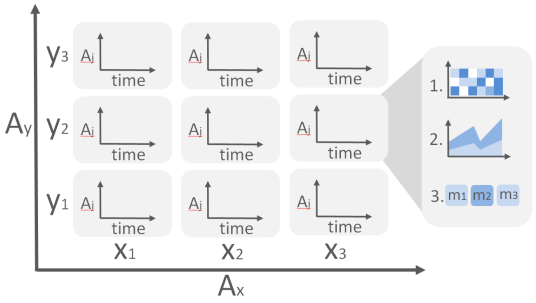

Figure 5: The time table visualizes continuous attribute A_i over time by splitting the traffic over multiple cells using A_x and A_y.

inspect messages individually. Since every task requires a different perspective, CoNTA supports three visualizations that can be used in the table cells, namely heatmaps, line charts, or pixel maps. The heatmap and line chart are designed to obtain an overview of attributes over the entire traffic. Line charts can be used for inspecting trends, whereas heatmaps enable experts to compare attribute behavior over larger collections of profiles. For the inspection of individual messages, experts can switch to a pixel map, where every message is represented as a rectangle (Figure 1d). By default, messages in the pixel map are colored according to their protocol. Malicious messages are indicated by a red box. Experts can enlarge the pixels through zooming to reveal a summary description of each message. Alternatively, experts can hover over a message to inspect their values by means of a popup (Figure 1e).

When selecting a numeric attribute A for the Y-axis of the table cells, the height of every line chart point p is determined by taking the average of all message values with attribute A in p. Attributes `#msg` and `#alerts` are an exception to this rule, since they represent the number of (malicious) messages in each point in time. The color of a heatmap cell is determined according to the chosen colormap. Line charts can also be split based on this colormap to gain insight in the value distribution of a particular attribute over time. This results in a stacked line chart as depicted in Figure 1a.

4.3 Context View

Experts can save selected messages of interest by assigning a name to them and defining them as a new context. The context view maintains a history of all contexts the expert is interested in. When creating a new context, experts can create a new attribute separating the selected messages from the non-selected. This attribute is added to the data and can be used for further analysis. This enables experts to tag the data with more domain-specific information during exploration.

For every context the number of messages and malicious messages is displayed. To stay aware of the size of the context, gray and red histograms are used to show the fraction of messages and alerts that are contained in the context with respect to the entire data set. The hierarchy in the view shows the ordering in which the selections were created. Context c is a child of parent context d if and only if c was created when the expert was exploring d. This relationship implies that the messages contained in a child context are always a subset of the messages in the parent.

4.3.1 Multi-context

One danger of drilling down in the data, is that overview can be easily lost. Eventually, the fraction of interesting data can become so small that any (visual) significant difference can be misleading due to bad scaling [11]. In CoNTA, experts are enabled to show one context c with respect to an ancestor. The result is that the ancestor context is added transparent in the background of c (Figure 4a, c, and d). This enables experts to see any trends and outliers that were currently not incorporated in the current context without losing overview.

We use glyphs in front of the contexts to show when multi-context is enabled. The filled inner circle represents the context of interest (in foreground), whereas the context with the filled outer circle is only visible in the background. Figure 4b shows how these glyphs are applied. Experts can enable multi-context by selecting an ancestor context while pressing `Alt`. To preserve the parent child relation in the context view, experts cannot refine their selection using the messages in the background context.

4.4 Conversation view

For the inspection of sequential patterns in conversations, we enable experts to inspect message attributes using a node-link diagram. Let A represent the attribute of interest. The graph is constructed by creating a node for every value in A and there is an edge (v_1, v_2) if and only if a message with $v_1 \in A$ is followed by a message with $v_2 \in A$ in current context c. The thickness of the edges represents the frequency in which values follow each other. Since the resulting graph greatly depends on the chosen attribute and context, we use the general-purpose Dot [9] algorithm for the layout.

Note that network traffic consists of multiple conversations running in parallel. Since the order in which conversations are interleaved in the traffic does not have any meaning, by default only sequential patterns within the conversations are being considered. For TCP traffic, a conversation (also known as a *session*) is defined as the traffic between two IP addresses between two port numbers. In case of numerical attributes, values are binned to reduce the number of nodes in the graph. Experts can prevent nodes with a high degree from occluding the visualization by hiding them using a slider.

Selecting a node v in the conversation view will highlight all messages in the traffic with value v. Selecting an edge (v_1, v_2) selects all message pairs in the conversations where v_1 is indirectly followed by v_2. A visualization of the network topology can be obtained by creating a graph of all IP or MAC addresses in the network. In contrast to other attributes, both source and destination addresses should be taken into account when constructing this graph. Experts can use this graph to filter the traffic on entire conversations and hosts.

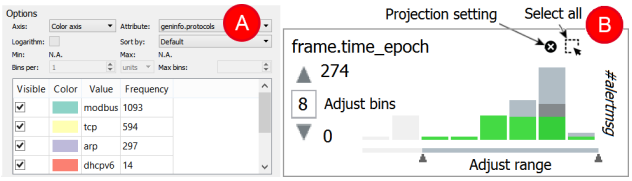

Figure 6: a) Options to adjust axis ordering and scaling. b) Overview attribute widget. Projection settings are visible in the classifier tab.

4.5 Attribute view

The attribute view shows an overview of all attributes in the traffic using scented widgets. Every attribute is represented as a histogram showing the value distribution of that attribute. The histogram is interactive and can be used to select and deselect messages with specific attribute values. The span slider below every histogram is used to enforce selections within specific value ranges. Time table axes can be set to a particular attribute through a context-menu. The number of bins in a histogram depends on the attribute's distribution. For categorical attributes, there is a bin for every value in that attribute. If the number of categorical values exceeds 20, a miscellaneous bin is introduced to represent the remaining values instead. By default, categorical bins are sorted by their frequency.

For numerical attributes, the number of bins of the histogram are computed using Scott's rule [25]. Experts can modify this number to gain more insight in the outliers of the attribute or to determine the granularity in which experts can interact with the histogram.

Histograms are split into two columns. The left column represents the value distribution of the attributes according to all the traffic in the current context. The right column shows the value distribution of the same attribute only considering malicious traffic. The separation enables experts to compare characteristics of malicious messages with the rest of the traffic. In addition, the selection of bins in the right histogram enables experts to search for malicious messages with certain values. More about this in Section 5.

4.5.1 Attribute ranking

Since traffic data can consist of hundreds of attributes, inspecting every attribute manually is impractical. Instead, experts can find interesting attributes by sorting them according to various metrics:

- *Alphabetical* sorts the attributes by name.
- *Most Alerts* sorts attributes by counting the number of times a particular attribute was considered malicious in the current selection.
- *Relevance* sorts attributes by computing the information gain [23] for each attribute with respect to the current selected messages. Attributes score high if they can separate selected messages from non-selected messages at best.

Experts can reapply the metrics on specific subsets of attributes by filtering them by name using a textual interface. This is in particular useful when experts are only interested in relationships between attributes within for instance the same protocol.

4.6 Classifier intergration

False positive rates of network IDSs are still high. To tackle this issue, we enable experts to optimize the underlying classifier through refinement. Since machine learning techniques in IDSs widely vary, we aim for a data-driven approach rather than a classifier-dependent one. In our approach, we support four operations:

- filtering;
- projection;
- binning; and
- self-training [34].

Experts can refine classifier scores by training the IDS on specific subsets of the traffic. With *filtering*, only messages with values that fit in the specified ranges are included during classification. *Projection* determines which set of attributes should be taken into account during classification. Projection can be used to exclude attributes that are sensitive to false positives (e.g., identifiers).

Yüksel et al. already showed that false positive rates in numeric attributes can be improved by reducing the granularity of the attribute's value distribution through *binning* [31]. Especially when dealing with values whose alert scores are close to the decision boundary, decreasing the granularity of the distribution can prevent these alerts from happening. *Self-training* enables experts to report valid messages back to the classifier by labeling them as safe. Instead of ignoring these messages in future classifications, the messages are moved to the training set of the IDS. This enables the IDS to prevent similar alerts in other parts of the data.

In CoNTA, experts can instantly apply filtering, selection, and binning using the attribute view in Section 4.5. When switching to the classifier settings, histograms are shown for every attribute in the data set displaying the number of times attribute values were considered malicious. The range slider of a histogram represents the filtering settings, whereas the number of bins shown represents the granularity setting of the classifier for that attribute. Experts can exclude alerts from the projection using the controls in Figure 6b. Self-training is achieved through selection and a context-menu.

5 Interaction

The views in CoNTA work at various levels of abstraction. The line chart, heatmap, and histograms inspect patterns at traffic level, whereas the pixel map and conversation view work at message and conversation level respectively. For the investigation of alerts in CoNTA, linking and interaction play a key role. To ensure that brushing and linking is consistent and understandable over all views, we decided to use messages as a central concept. Message-oriented interaction across views enables experts to reason about their selections as sets of messages. Additional messages can be selected in different views by selecting visual elements while holding the `Ctrl` key (e.g., set union). Deselecting elements will remove the messages from the current selection (e.g., set difference).

To preserve consistency when creating a selection, every visual element (i.e., heatmap cell, histogram bar, linechart series, and graph node/edge) is filled with a green color proportional to the fraction of the selected messages in that item (Figure 7). To ensure that the intensity of the item's background color is preserved, transparency is added to the selection color. Similarly, hovering the mouse over an item will show the fraction of hovered items (that were not already selected) as a translucent dark gray color on top of the selection. This enables experts to see the impact of the new selection before applying it. To prevent elements and selections from getting visually too small to properly interact with them, the height of every element and selection is set to a minimum of two pixels.

6 Use cases

We tested the effectiveness of our method on one artificial and one real world data set. The first data set represents the simulation of a fully functional artificial water plant consisting of 5 hosts, 80,000 messages and 170 attributes. The data set was designed by an external security company that is specialized in the detection of malicious activity in industrial control systems. To show the practical existence and impact of APTs, they injected an APT to damage the facility. The second data set is obtained by recording 3 days of internal SMB2 network traffic from a university for which there was no ground truth known beforehand. The data set corresponds to approximately 800,000 messages, 400 attributes, and was sent by approximately 20 hosts. For a better experience of the interaction and use cases in practice, we refer to the supplementary video.

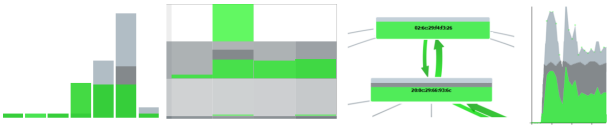

Figure 7: Brushing and linking is applied accross all views in CoNTA.

6.1 Water plant

6.1.1 Discovery

We initially start exploring the data set by inspecting the number of alerts in the network over time using a line chart. We select the burst period between 16:55 and 17:10 PM in the line chart and save the messages in a new context called "alert burst" (Figure 1a). According to the topology of the network, most traffic was created by three nodes: the water tank (`...:80`); the SCADA system monitoring the plant (`...:88`); and a router in between (`...:69`) (Figure 4d). Alerts that were caused by infrequent protocols in `frame.protocols` are removed using projection (Figure 1b). Alerts that involve rarely active hosts are removed by selecting the infrequent bins in the right `ip.src` histogram of the attribute viewer and reporting them back to the IDS. After selecting the new context, we sort the attribute for most common alerts. The attribute `mbtcp.reg_uint16` scores high indicating that many different messages with strange register values were seen by the IDS. Our eye was caught by the attribute `mbtcp.group0` whose right histogram shows that there are 50 messages with alerts that have the value `duplicate IP` (Figure 1e).

6.1.2 Identification

We select all alerts with the strange value by switching to the classifier interface using the right histogram in the attribute view (Figure 1c). Sorting the selection by relevance shows that most selected messages were received by two MAC-addresses. We group the traffic per MAC-address by setting the time table's Y-axis to `eth.src`. Switching to the pixel map shows that most alerts are present at the water tank (Figure 1d). Coloring the messages by IP source reveals that the router uses the same IP addresses as the other nodes, suggesting the presence of man-in-the-middle activity. We select the conversations between the three nodes using the edges in the conversation view, filter the traffic by `Modbus`, and create a new context for them for further investigation.

6.1.3 Confirmation

Now that we know that this router is suspicious, the next step is to find out what the router is aiming for. We select all malicious messages that were sent by the router using the right histogram `eth.src` in the attribute viewer. Filtering this view by only Modbus fields and sorting the widgets by relevance reveals that most alerts were caused when reading particular registers (Figure 4c). Since each register in the plant stores its own data, we group the traffic per register by setting the time table X-axis to `mbtcp.ref_num` and table cell Y-axis to `mbtcp.reg_uint16`. Figure 4e now shows how register values sent by the water tank are actually perceived by the SCADA system and vice versa. Register 1 shows the status of the tank's valve, where the height of the register value describes the extent to which the valve is open. Register 5 represents the overflow flag the tank raises when the water level exceeds a certain threshold. Note that the close valve commands that are sent to the router are not forwarded to the tank. Further note that the tank's overflow flag is suppressed by the router. The result is that the tank overflows while the SCADA system is unaware of this situation. Finally, we select the `All` context in the background to see in the conversation window that this router apparently also has conversations with other hosts in the network (Figure 4d).

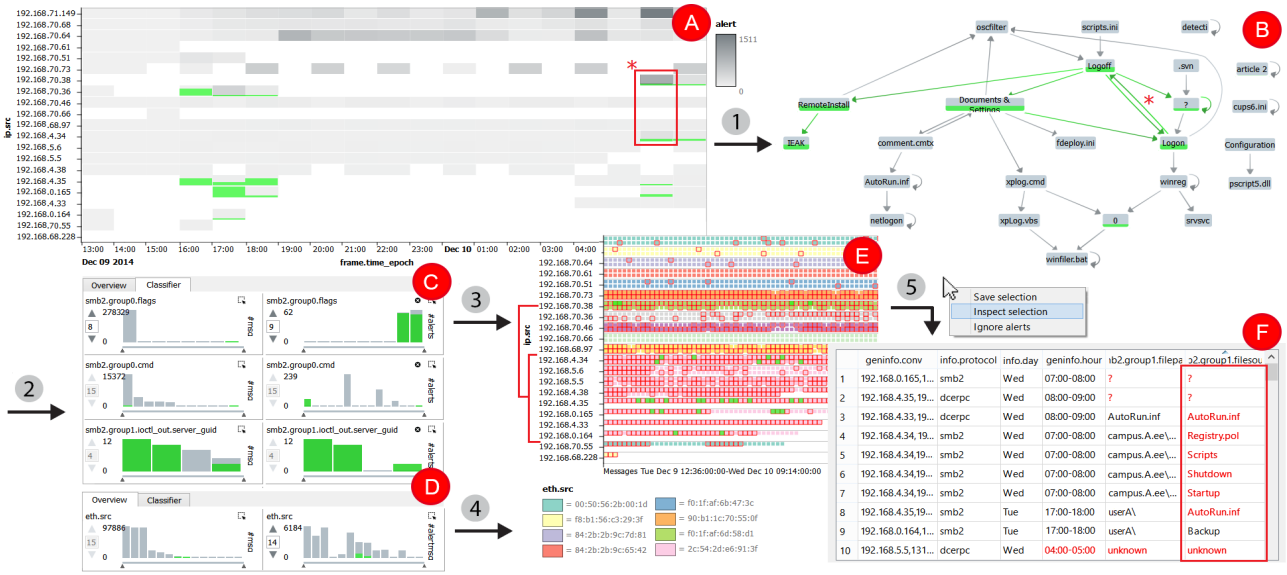

Figure 8: a) Heatmap showing the number of alerts per IP address per hour. b) Conversation view on `smb2.filenames` to detect the presence of strange access orderings. c) Attribute sorting on Most Alerts shows that selected alerts use the same `smb2.group0.flags`. d) Sorting attributes by relevance after selecting messages with the same flags. e) Pixel map per IP colored by `ip.src`. f) Tabular view of alert selection.

6.2 University

Next, we show how sequences of messages can assist the expert in investigating when and how files are accessed in the network. We first create a new context only containing SMB2 messages with file names by selecting all bins in the left histogram of the attribute `smb2.filesource`. Since the number of hosts in the network is significantly larger compared to the previous data set, we start exploration by creating a heatmap showing the number of messages per IP address (Figure 8a). IP addresses in the subnets `*.*.4.*`, `*.*.5.*`, and `*.*.71.*` generate traffic over day and night. Switching to the pixel map and coloring the pixels according to their SMB2 command, shows that these addresses are servers only sending response messages. After sorting the Y-axis by frequency and coloring the heatmap by number of alerts per hour shows that on December 10 06:00-07:00 AM IP addresses `192.168.4.34` and `192.168.70.38` generated more alerts with respect to the other time slots. Selecting both heatmap cells shows in the conversation view that both hosts were communicating with each other.

We inspect the file access behavior of the hosts by creating a node link diagram of all the files that were accessed in the network. Frequent files that are accessed by everyone (e.g., the file `spoolss` whenever a document is printed) are hidden using the visibility slider. Besides the strange `?` in the graph, we see that a `Login` file is indirectly followed by a `Logoff`. Selecting the edge from `Logon` to `Logoff` shows that this behavior is generated by `192.168.4.34` and `192.168.70.38` (Figure 8a, red box). Selecting the suspicious heatmap cells shows in the graph that these hosts accessed a wide variety of files within an hour such as `Autorun.inf`, `scripts.ini`, and `RemoteInstall` (Figure 8b). Sorting the attribute view on *Most alerts* shows that most of the selected messages have the `smb2.group0.flag` set to `8` instead of `0` (Figure 8c). Deselecting the other flag values and sorting the attributes by relevance shows that these alerts where only generated by two MAC addresses (Figure 8d). We switch back to the pixel map and color the messages by MAC address. This shows that one user runs multiple virtual machines on the same host (Figure 8e). The interesting part however is that the servers we detected with sub nets `*.*.4.*` and `*.*.5.*` are all originating from the same machine. Inspecting the alerts that were generated by these addresses using a table view, we can see that the machine was accessing rather interesting file names in the network (Figure 8f).

7 Discussion and Limitations

The use cases in Section 6 illustrate that there is a strong interplay between high-level traffic overviews, low-level message views, and attributes. The tight linking between the different views plays a key role in understanding how high-level phenomena such as bursts relate to the presence of low-level alerts in messages. By tagging message collections through (de)selection, network traffic can be incrementally enriched with intuitive domain-specific descriptions.

The definition of an outlier greatly depends on the domain knowledge and the context in which the data is observed. The access of a file X does for instance not have to be malicious in general, but can be dangerous when performed by a certain user. The exploration method should therefore be flexible and expressive enough to create and inspect new selections without much effort. The time table facilitates this by enabling experts to inspect traffic with visualizations they are familiar with. Combined with multi-context functionality, outliers can be inspected in various contexts with a single mouse click. Being able to select and deselect messages based on their attributes, values, and temporal occurrence in conversations, while directly gaining feedback on both message and traffic level provides experts with a powerful exploration mechanism.

Like any methodology, there are limitations. First, the number of small multiples in the time table does not scale well when considering attributes with many different values. Although the expert is enabled to hide values and use scroll bars to limit the number of displayed values, this only solves the problem partly. Furthermore, the node-link diagram in the conversation view does not scale when visualizing large networks. Note however that the analysis of alerts hardly involves the analysis of all the traffic in the network at once. Since the analysis of alerts quickly narrows the area of interest, we decided to choose visualization methods based on their understandability and commonality, rather than their scalability.

Second, the interaction with attributes is limited to the number of visible scented widgets. Showing too many attributes will break the interaction whereas too few attributes will increase the risk of missing potential correlations. Although sorting, filtering, and scrolling helps to find interesting attributes, creating queries involving many attributes can become a burden and a textual interface is preferred.

Third, the proposed classifier refinement approach implicitly assumes that the underlying classification model is suitable for semi-supervised learning. Although the interaction enables experts to

train the classifier additionally on specific parts of the traffic, there is no clear boundary between fitting and overfitting the underlying model. The extent to which an expert can detect a false positive can greatly influence the classifier's performance in a good or bad way.

8 Conclusions and future work

We presented a novel approach for domain experts to explore large message collections using automatic generated alerts and interaction as a solid basis. The ability to interactively switch from traffic-level overviews to message-level details enables experts to investigate the relationship between high-level traffic phenomena and low-level message fields, while staying aware of other concepts such as conversations and sequential patterns. The combination of attribute-based scented widgets and selection-based relevance metrics enables experts to search through large attribute collections and refine classification results in multiple dimensions. Since the methodology exhibits the structure of time-dependent multivariate data, it is general and flexible enough to be applied in other domains. We have shown the effectiveness of the approach on real world and artificial data sets clearly illustrating the complexity network analysts have to deal with.

For future work it is interesting to see how we can enrich our method by training new classifiers on specific sub parts of the traffic. This would enable experts to interactively test the IDS performance for different types of profiles. Developing an intuitive interaction mechanism however is nontrivial. Furthermore, extensive evaluation is required to study the approach's real-time capabilities and effectiveness in different network environments and domains.

Acknowledgements

This work is funded by SpySpot, a project in the Cyber Security program of Netherlands Organisation for Scientific Research (NWO). The authors wish to thank Elisa Costante from Security Matters, Erik Boertjes from TNO, Kay Reijnen and Jeroen Teeuwen et al. from Motto Communications, colleague Paul van der Corput, and in particular SpySpot members Sandro Etalle, Ömer Yüksel, Davide Fauri, and Jerry den Hartog for their valuable comments throughout this work. The artificial dataset is an output of the European Commission project FP7-SEC-607093- PREEMPTIVE.

References

[1] K. Abdullah, C. Lee, G. Conti, J. Copeland, and J. Stasko. IDS Rainstorm: Visualizing IDS Alarms. In *Proceedings of the IEEE Workshops on Visualization for Computer Security*, VIZSEC '05, pages 1–, Washington, DC, USA, 2005. IEEE Computer Society.

[2] D. Arendt, R. Burtner, D. Best, N. Bos, J. Gersh, C. Piatko, and C. Paul. Ocelot: user-centered design of a decision support visualization for network quarantine. In *Visualization for Cyber Security (VizSec), 2015 IEEE Symposium on*, pages 1–8. IEEE, 2015.

[3] A. Attipoe, J. Yan, C. Turner, and D. Richards. Visualization tools for network security. *Electronic Imaging*, 2016(1):1–8, 2016.

[4] J. Beale, A. Baker, J. Esler, and T. Kohlenberg. *Snort: IDS and IPS toolkit*. Syngress, 2007.

[5] B. Cappers and J. van Wijk. SNAPS: Semantic Network traffic Analysis through Projection and Selection. In *Visualization for Cyber Security (VizSec), 2015 IEEE Symposium on*, pages 1–8. IEEE, 2015.

[6] G. Combs et al. Wireshark-network protocol analyzer: http://www.wireshark.org/. *Version 0.99*, 5, 2008.

[7] A. DAmico and K. Whitley. The real work of computer network defense analysts. In *VizSEC 2007*, pages 19–37. Springer, 2008.

[8] S. Dua and X. Du. *Data mining and machine learning in cybersecurity*. CRC press, 2016.

[9] E. Gansner, E. Koutsofios, S. North, and K.-P. Vo. A technique for drawing directed graphs. *IEEE Transactions on Software Engineering*, 19(3):214–230, 1993.

[10] D. Hadziosmanovic, D. Bolzoni, S. Etalle, and P. Hartel. Challenges and opportunities in securing industrial control systems. In *Complexity in Engineering (COMPENG), 2012*, pages 1–6. IEEE, 2012.

[11] D. Huff. *How to lie with statistics*. WW Norton & Company, 2010.

[12] J. Kittler, M. Hatef, R. Duin, and J. Matas. On combining classifiers. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 20(3):226–239, 1998.

[13] H. Koike and K. Ohno. Snortview: Visualization system of snort logs. In *Proceedings of the 2004 ACM Workshop on Visualization and Data Mining for Computer Security*, VizSEC/DMSEC '04, pages 143–147, New York, NY, USA, 2004. ACM.

[14] A. Kott, C. Wang, and R. Erbacher. *Cyber Defense and Situational Awareness*. Springer, 2014.

[15] J. Landstorfer, I. Herrmann, J. Stange, M. Dork, and R. Wettach. Weaving a carpet from log entries: A network security visualization built with co-creation. In *Visual Analytics Science and Technology (VAST), 2014 IEEE Conference on*, pages 73–82. IEEE, 2014.

[16] B. Li, J. Springer, G. Bebis, and M. H. M.H. Gunes. A survey of network flow applications. *Journal of Network and Computer Applications*, 36(2):567–581, 2013.

[17] Y. Livnat, J. Agutter, S. Moon, R. Erbacher, and S. Foresti. A visualization paradigm for network intrusion detection. In *Information Assurance Workshop, 2005. IAW'05. Proceedings from the Sixth Annual IEEE SMC*, pages 92–99. IEEE, 2005.

[18] U. Meyer and S. Wetzel. A man-in-the-middle attack on umts. In *Proceedings of the 3rd ACM workshop on Wireless security*, pages 90–97. ACM, 2004.

[19] N. Nykodym, R. Taylor, and J. Vilela. Criminal profiling and insider cyber crime. *Computer Law & Security Review*, 21(5):408–414, 2005.

[20] V. Paxson et al. Guide, Bro Quick Start. http://www.bro-ids.org.

[21] R. Plackett. Karl pearson and the chi-squared test. *International Statistical Review/Revue Internationale de Statistique*, pages 59–72, 1983.

[22] M. Pollitt. An ad hoc review of digital forensic models. In *Systematic Approaches to Digital Forensic Engineering, 2007. SADFE 2007. Second International Workshop on*, pages 43–54. IEEE, 2007.

[23] J. Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

[24] B. Rogowitz and A. Goodman. Integrating human-and computer-based approaches to feature extraction and analysis. In *IS&T/SPIE Electronic Imaging*, pages 82910W–82910W. International Society for Optics and Photonics, 2012.

[25] D. Scott. Scott's rule. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(4):497–502, 2010.

[26] H. Shiravi, A. Shiravi, and A. Ghorbani. Ids alert visualization and monitoring through heuristic host selection. In *Information and Communications Security*, pages 445–458. Springer, 2010.

[27] H. Shiravi, A. Shiravi, and A. Ghorbani. A survey of visualization systems for network security. *Visualization and Computer Graphics, IEEE Transactions on*, 18(8):1313–1329, 2012.

[28] R. Sloan. Advanced Persistent Threat. *Engineering & Technology Reference*, 1(1), 2014.

[29] S. Stasiukonis. Social engineering, the usb way. *Dark Reading*, 7, 2006.

[30] W. Willett, J. Heer, and M. Agrawala. Scented widgets: Improving navigation cues with embedded visualizations. *Visualization and Computer Graphics, IEEE Transactions on*, 13(6):1129–1136, 2007.

[31] O. Yüksel, J. den Hartog, and S. Etalle. Reading Between the Fields: Practical, Effective Intrusion Detection for Industrial Control Systems. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, SAC '16, pages 2063–2070. ACM, 2016.

[32] T. Zhang, Q. Liao, and L. Shi. Bridging the gap of network management and anomaly detection through interactive visualization. In *2014 IEEE Pacific Visualization Symposium*, pages 253–257. IEEE, 2014.

[33] C. Zhong, D. Kirubakaran, J.Yen, P. Liu, S. Hutchinson, and H. Cam. How to use experience in cyber analysis: An analytical reasoning support system. In *Intelligence and Security Informatics (ISI), 2013 IEEE International Conference on*, pages 263–265. IEEE, 2013.

[34] X. Zhu and A. Goldberg. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, 3(1):1–130, 2009.