NStreamAware: Real-Time Visual Analytics for Data Streams to Enhance Situational Awareness

Fabian Fischer
University of Konstanz, Germany
Fabian.Fischer@uni-konstanz.de

Daniel A. Keim
University of Konstanz, Germany
Daniel.Keim@uni-konstanz.de

ABSTRACT

The analysis of data streams is important in many security-related domains to gain situational awareness. To provide monitoring and visual analysis of such data streams, we propose a system, called *NStreamAware*, that uses modern distributed processing technologies to analyze streams using *stream slices*, which are presented to analysts in a web-based visual analytics application, called *NVisAware*. Furthermore, we visually guide the user in the feature selection process to summarize the slices to focus on the most interesting parts of the stream based on introduced expert knowledge of the analyst. We show through case studies, how the system can be used to gain situational awareness and eventually enhance network security. Furthermore, we apply the system to a social media data stream to compete in an international challenge to evaluate the applicability of our approach to other domains.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and protection*; C.3.8 [**Computer Graphics**]: Application; H.5.2 [**Information Interfaces and Presentation**]: User Interfaces

General Terms

Real-Time Processing, Data Streams, Situational Awareness, Network Security, Visual Analytics

1. INTRODUCTION

In many security-related scenarios the analysis and situational assessment of data streams is crucial to detect suspicious behavior, to monitor and understand ongoing activities, or to reduce streams to focus on the most relevant parts. For example, in the field of system and network administration, network routers and servers produce a continuous stream of NetFlow records or system log messages, and hundreds of system metrics and performance data. In some times, analysts do a close real-time monitoring, while in other situations analysts have no choice, but to focus only on the most important parts of a data stream. The same is true in the field of law enforcement in the analysis of criminal activities of ongoing threats to maintain situational awareness (SA). In this scenario, analysts need to handle streams of possibly important social media messages and call center messages. Both scenarios are technically related and show the high importance of research in the field of data stream analysis with the analyst in the loop that is a key to enhance situational awareness. The challenge in this field is also to merge and aggregate heterogeneous high velocity data streams. While we do have a wide variety of highly-scalable databases and there has been much research in intrusion and anomaly detection, fully automated systems are not working sufficiently. To convey and support understanding, generate insights, and evaluate hypothesis, analysts needs to have a central role in such a system, to not loose context, and to be able to judge data provenance. The ultimate goal allows the analysts to actually get an idea what is going on in a data stream to gain situational awareness. Such analysts are often "being asked to make decisions on ill-defined problems. These problems may contain uncertain or incomplete data, and are often complex to piece together. Consequently, decision makers rely heavily on intuition, knowledge and experience" [14], which highlights the need to guide analysts to the *right* parts of a data stream, because it is impossible to analyze everything in the same level of detail.

In this paper, we introduce *NStreamAware*, which is a visual analytics system designed to address this challenge using latest analysis technologies available from the big data analysis community [20] and real-time visual analytics research [12].

The main contributions of our work are the following: Firstly, a system architecture, called *NStreamAware*, based on Apache Spark Streaming [2] to summarize incoming data streams in *sliding slices*. Secondly, a web-based visual analytics application, called *NVisAware*, using a novel combination of various visualization techniques within multiple sliding slices to visually summarize the data stream based on selected features steered by a visual analytics interface.

The remainder of this paper is structured as follows: Section 2 elaborates on important design considerations. Section 3 gives an overview of related work. Section 4 describes the different aspects of our approach, while the evaluation is discussed in Section 5. Section 6 discusses limitations and future work and concludes with Section 7.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.
VizSec '14, November 10 2014, Paris, France
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-2826-5/14/11 ...$15.00.
http://dx.doi.org/10.1145/2671491.2671495

2. DESIGN CONSIDERATIONS

Based on the given problem, experience, and expert feedback with earlier work in the field, we identified following design considerations and principles as crucial for our approach.

DC1 **Incorporate novel scalable analytics methods:** Scalable, distributed, and proven large-scale analysis frameworks must be building blocks of a system able to address big data problems. We need to take advantage of such novel technologies from the big data community and use them in visual analytics application. We need to bring those worlds together and keep the analyst in the loop to address complex problems.

DC2 **Enabling real-time monitoring:** While it is not possible to present *all* raw messages for high speed streams, it is still relevant for many scenarios, where analysts want to closely monitor messages from a particular system, or based on a specific filter criteria in real-time. Many available visual analytics systems, however, still do require a static batch loading first. We see the need to be able to directly push data to our system in a streaming fashion, and be able to smoothly switch between monitoring and exploration.

DC3 **Deterministic screen updates, independent from underlying data streams:** The problem in systems supporting DC2 is the high cognitive load for the analysts when analyzing real-time streams. Because of the unpredictable characteristics of data streams with respect to volume, velocity, variety, and veracity, we additionally need visualizations able to decouple the flow-rate of a data stream from screen updates and keep the latter constant and predictable to not overwhelm the user. There is a trade off between DC2 and DC3 to achieve both at the same time.

DC4 **Fusion of heterogeneous data sources:** Many available systems do focus on individual data sources, and provide less flexibility to incorporate and correlate various heterogeneous data sources. However, focusing on particular individual data sources helps to develop highly effective specific visualization systems. On the other hand, it is important to cover a broader field of scenarios and tasks, to provide better situational assessment.

DC5 **User-steered feature selection:** Feature selection is an important field to support analysts using appropriate visualization and interaction techniques. Our goal is to enhance understanding of data streams and provide more compact overviews. In this process, we want to integrate the human in the workflow that requires a tight coupling of visual representations, interaction and analytic methods.

3. RELATED WORK

The contributions of our work are related to various research fields, so we discuss various areas in the following section. Many researchers focus on the algorithmic analysis of data streams, especially in the field of stream clustering [1] and event detection. In recent years, there was a focus on social data streams, because of the wide availability of such data. While most of these systems focus on the detection of events, our work contributes more in the field of visualizing a condensed heterogeneous data stream to focus on more interesting changes, omitting or merging less interesting ranges to eventually focus on important parts in more detail. This idea is related to the work of Xie et al. [19] proposing a fully-automated merging algorithm for *time-series* data streams.

A recent study by Wanner et al. [18] takes a look at the evolution of visual analytics applications for event detection for text streams and concludes that "visualizations were primarily used as presentation, but had no interaction possible to steer the underlying data processing algorithm". This confirms our assumption, that many systems do not cover DC5 appropriately. Our approach differs, that we provide interactions, so that users are able to steer the feature selection process. Therefore, the system does not only rely on the fully-automated selection of interesting parts, but on the user-adjusted feature set. The ultimate goal of visual analytics systems for data streams is to enhance situational awareness to facilitate decision making. Endsley provides a widely used generic definition of SA. It "is the perception of the elements in the environment within a volume of time and space, the comprehension of their meaning, and the projection of their status in the near future" [6]. Further work makes it clear, that situation awareness primarily resides "in the minds of humans", while situation assessment better describes the "process or set of processes" leading to the state of SA [16]. In the complex field of computer network security operations, only a combination of various tools used by experienced domain experts, will eventually be able to guide the user to such cognitive state. Franke and Brynielsson [9] give a systematic literature overview specifically for the field of cyber situational awareness.

Furthermore, there is not only work on SA systems, but also visualization techniques (e.g., [7]) designed to convey the current state of the network to best support situational assessment. ELVIS [10] is a highly interactive system to analyze system log data, but cannot be applied to real-time streams. SnortView [11] focus on the specific analysis of intrusion detection alerts and does satisfy DC2. The focus of Event Visualizer [8], is to provide real-time visualizations for event data streams (e.g., system log data) to provide real-time monitoring and possibilities to smoothly switch to exploration mode covering DC2 and DC4. In contrast to this event-based approach, Best et al. [3] proposes another real-time system to enhance situational awareness using the analysis of network traffic based on LiveRAC [13]. The analyzed and aggregated time-series are displayed in a zoomable tabular interface to provide the analyst an interactive exploration interface for time-series data, while our approach is more general to include also other data types (e.g., frequent words or users, hierarchical overviews) addressing DC4.

Additionally, Shiravi et al. [15] provides an extensive overview of various visualization systems for network security based on five major use case classes: Host/Server Monitoring, Internal/External Monitoring, Port Activity, Attack Patterns, and Routing Behavior. The authors also identified the fact, that most security visualization systems, in their current state, are mostly suitable for offline forensics analysis", while "real-time processing of network events requires extensive resources, both in terms of the computation power required to process an event, as well as the amount

of memory needed to store the aggregated statistics" [15]. Compared to work specifically found in one of these use case classes, our approach tries to combine multiple use cases into a real-time visual analytics system and addresses the scalability issues using Apache Spark.

4. VISUAL ANALYTICS SYSTEM

In the following, we describe the building blocks of *NStreamAware*. The overall architecture can be seen in Figure 1. To process the data stream, we made use of various modern technologies to provide a scalable infrastructure for our modular visual analytics system. Our architecture consists of our *REST Service*, *Spark Service* and a web application with various visualizations, called *NVisAware*. To provide proven and scalable data processing, we make use of Apache Spark[1], RabbitMQ[2], ElasticSearch[3], and MongoDB[4].

The *REST Service* (1) connects to the data streams (2) and preprocesses the data and calculates various additional information for the incoming events. The service does also provide a REST interface to retrieve historical data or manage insights. All events are stored to a distributed *ElasticSearch* cluster and are forwarded to our message broker *RabbitMQ*.

The *Spark Service* (3), which runs on top of the Apache Spark Streaming platform for analytics, generates real-time summaries on sliding windows, and stores them to a *MongoDB* database (4). Spark Streaming is a development framework to help to implement analytical algorithms executed in large distributed cluster environments to provide scalability even in big data scenarios. The *Spark Service* is implemented using Scala and calculates various statistics and features based on sliding windows. Table 1 shows a selection of calculated example features for a network security use case. We call these summaries, which are generated in a regular interval, *sliding slices*. Those slices and also a selection of raw messages are eventually forwarded to our web application *NVisAware* (5), so that they can be visualized in the graphical user interface to the analyst using various interactive real-time displays. All modules are loosely coupled, so that they can be run on separate computers or in cluster environments to achieve best performance for large-scale data streams.

4.1 REST Service Module

The *REST Service* (1), which is implemented as multi-threaded standalone Java application, provides a REST interface accessible by all other modules, especially the web application. This REST service is used to handle job queuing and to answer data requests. To attach new data streams, the respective jobs can be sent to the service via a defined REST API. The job is added as new thread and the API can be used to control or retrieve status information about these running jobs. Incoming messages from the data stream are then preprocessed, fields are extracted, and eventually treated as individual events, enriched with various additional attributes. The procedure is based on the assigned scenario configuration. For social media messages, sentiment values are calculated, while for IP-related data geo lookups can be made. In practice, many server do not provide very accurate timestamps, therefore, a new field with the current timestamp is added as well, to have a more accurate timings in cases where the workstation does not make use of the network time protocol or uses deviating time settings.

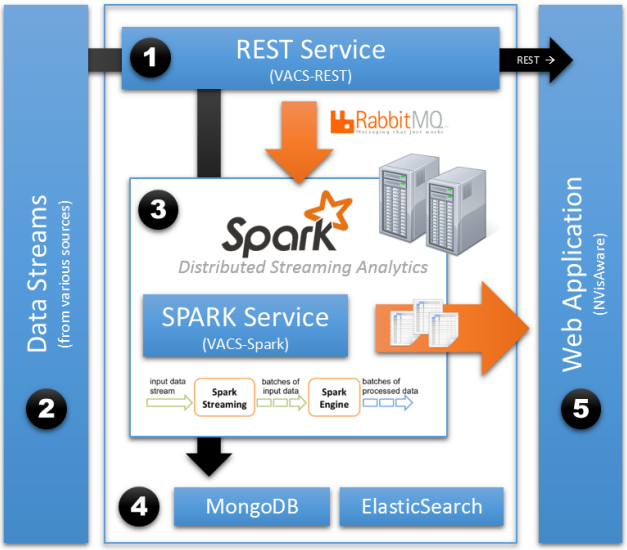

Figure 1: System Architecture: *NStreamAware* uses various modern systems, including Apache Spark, RabbitMQ, MongoDB, and ElasticSearch, to provide the needed scalability for an interactive visual analytics application.

4.2 Module for Spark Streaming

Apache Spark provides distributed memory abstraction, that is fault-tolerant and efficient. This helps to program distributed data processing applications without worrying about fault-tolerance. Apache Spark introduces a programming model, called Resilient Distributed Datasets (RDDs), which provide an interface to coarse-grained transformations (e.g., map, group-by, filter, join). The RDDs can be addresses within Scala similar to normal collections, however, they are indeed spread over the underlying cluster machines. If a *transformation* is called on a RDD, the execution is actually done on various worker machines. When an *action* is called (e.g., count), the result is retrieved from all workers to return final results. We use the streaming extension of Apache Spark and use the same programming model to analyze data streams in real-time. We define a sliding window and connect to a RabbitMQ queue to receive messages forwarded by the *REST Service*. Currently, we defined various feature types to be calculated on the incoming messages: *count*, *set*, *new-set*, *key-value list*, and *key-array list*. All features as seen in Table 1 for example belong to one of these message types. After calculating the various features, they are directly stored to a MongoDB collection. When all features are ready, *NVisAware* is notified via RabbitMQ to retrieve the sliding slice content via the REST API using the appropriate database queries. *Count* provides a simple counter of number of messages. A *set* stores a list of unique values occurred within a sliding window, while a *new-set*

[1]https://spark.apache.org/
[2]http://www.rabbitmq.com/
[3]http://www.elasticsearch.org/
[4]http://www.mongodb.com/

feature will only include values, which have never been seen in the whole stream before. A *key-value list* can be used to count the number of occurrences for all words to gather a list of frequent words. The *key-array list* can be used to store for each key an array of values. This can be used, for example, to track for each IP address, all used port numbers in the sliding window.

Feature	Type	Stream
$\#events$	count	Syslog
$timestamps$	set	Syslog
$\#programs$	count	Syslog
$\#hosts$	count	Syslog
$\#frequentWords$	count	Syslog
$programs$	key-value list	Syslog
$hosts$	key-value list	Syslog
$frequentWords$	key-value list	Syslog
$newHosts$	new-set	Syslog
$newPrograms$	new-set	Syslog
$srcAddr$	key-value list	NetFlow
$dstAddr$	key-value list	NetFlow
$srcPorts$	key-value list	NetFlow
$dstPorts$	key-value list	NetFlow
$topTalker$	key-array list	NetFlow
$\#srcAddr$	count	NetFlow
$\#dstAddr$	count	NetFlow
$\#srcPorts$	count	NetFlow
$\#dstPorts$	count	NetFlow
$ossecAlerts$	key-value list	OSSEC

Table 1: Selection of aggregation features for each sliding slice generated by our implemented analysis and aggregation module.

4.3 NVisAware Web Application

The graphical user interface is provided by our web application *NVisAware*, which provides various displays. The application is written in HTML5 and JavaScript using various visualization libraries. The display consists of multiple configuration and parameter views and six main tabs: *Real-Time Data Stream*, *Real-Time Sliding Slices*, *Visual Feature Selection*, *Summarized Sliding Slices*, *Event Timeline & Insights*, and *Search & Exploration*. The first display can be seen in Figure 3 and is used to take a look at the raw messages in the data stream.

4.4 Real-Time Sliding Slices

To visually represent the generated sliding slices, we provide a novel visualization with various embedded charts like word clouds, node-link diagrams, treemaps, and counters within each slice. The slices are juxtapositioned next to each other to provide a timeline based on consecutive slices as seen in Figure 4. The prominent background color uses a colormap from dark green over white to pink based on a diverging ColorBrewer set. The color indicates a similarity score to the previous slice to alarm the analyst. In the upper left corner a star icon can be used, to store the slice for further investigations. The slice will also be added to the *Event Timeline & Insights* view, where all starred objects are presented in a traditional interactive timeline to explore the events flagged and labeled by the analysts.

4.5 Visual Feature Selection

In many situations, the analyst is not interested in following the data stream in real-time. However, in some cases a summary of the current data stream should be provided. Fully-automated summarizations are hard to achieve for complex heterogeneous data streams. Therefore, we provide a visual feature selection interface, to steer the merging algorithm based on the user's criteria.

All *count* features in Table 1 can directly be used in the feature timelines in Figure 2. More features can be derived from *key-value lists*. For example the occurrences over time of a specific word found in the stream. Each feature timeline contains many values, one value for each sliding slice observed so far. This data is processed on the server side and each feature timeline is cut into segments: Each timeline is clustered using the DBSCAN algorithm. Afterwards, consecutive slices belonging to the same cluster are merged to a segment. The start and end points of these possibly important segments are visible as vertical colored lines and through the background shading within the timelines. The analyst can visually interpret these segments, modify them, or add new segments for interesting parts, which were not detected by the algorithm. The analyst can remove or reorder the features using drag and drop.

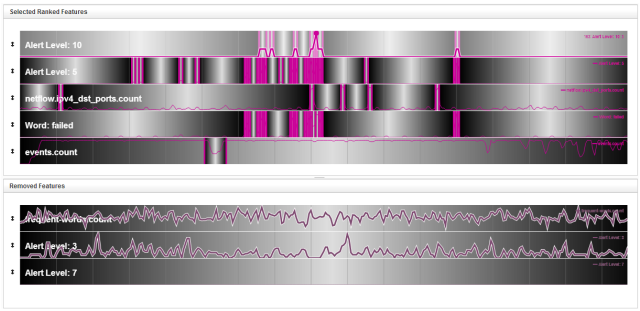

Figure 2: Visual Feature Selection: The analyst is in the loop to steer the merging algorithm to provide meaningful summaries of sliding slices.

The final feature order and selection is sent to the REST service, where all segments are merged together with the given constraints, while ignoring low-ranked conflicting features and keeping non-conflicting and more specific segments.

Eventually, the original sliding slices can be compressed according to the resulting heuristic merge and importance model. Less important segments are merged together providing a multi-focal scaling of the data stream steered by the analyst according to the tasks at hand.

5. EVALUATION

In general, it is quite challenging to evaluate complex visual analytics applications. Individual design decisions can be formally evaluated in user studies and many decisions are indeed based on perception studies. However, proper evaluation of complex expert applications is more than to evaluate all individual design decisions. Describing convincing use cases or presenting case studies with experts are often the only reasonable ways. However, also these results are often subjective and hard to compare to alternative approaches. Another reason is, that "insight, the major aim of visual analytics, is ill-defined and hard to measure" [17]. This is even more true, if we are talking about a mental state of situational awareness as goal of the system. Generally, there is also a lack of proper ground truth, and the sensitive nature of the involved data streams makes it hard to share the data. With that respect international challenges that provide complex but anonymous data streams are very helpful for a proper evaluation based on gained insights.

Having this in mind we decided to go for two directions of evaluations. Firstly, we describe a case study, how our system can be used in an operational computer network of a working group to help the system administrator to stay informed about the most important activities. Secondly, to evaluate the real-time capabilities of our system and the insights management, we actively participated in VAST Challenge 2014 with an early version of our prototype.

5.1 Application for Network Security

To show the capabilities of our system, we implemented our system in a computer network of a working group with about 85 active local devices including workstations, mobile devices, and servers, producing about 1.4 million NetFlow records per day with peaks up to 10 000 records per minute. 13 servers are connected to a central syslog server, producing 30 000 to 80 000 messages per day with individual peaks of up to 5 000 messages per minute. These servers are also monitored using *OSSEC* [4], which is a widely used "host-based intrusion detection system that performs log analysis, file integrity checking, policy monitoring, rootkit detection, real-time alerting and active response". The generated alerts are also pushed to the central syslog server. With this infrastructure in place, we were able to forward the data streams to our *REST Service* to make them available for *NStreamAware*. In the following, we made use of the system log stream (SL), NetFlow stream (NF), and OSSEC alert stream (OS). It would be easy, to further include additional data from the underlying network, for example, system metrics, Snort alerts, or web server access logs.

The analyst opened the web application *NVisAware* in a modern web browser and added the data streams as jobs to the server-side *REST Service*. Seconds later, the first messages appeared in the *Real-Time Data Streams* tab as seen in Figure 3. This view is a split-screen showing the real-time events of SL and OS as textual messages, similar to a traditional *tail -f* command on UNIX systems. The bottom window presents a zoomable geographical map to plot and cluster extracted geographic locations. NF records are not plotted to the geographic map, because a geographic map of the total IP traffic will most likely not provide actionable insights. However, mapping specific IP addresses of successful logins can be worth monitoring to identify suspicious behavior or to reveal misuse of login credentials. Furthermore,

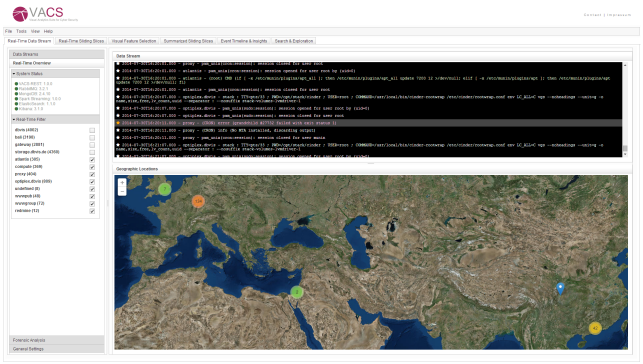

Figure 3: Real-Time Data Stream: Display to monitor the incoming live streams as raw messages and plot extracted geographic locations to a map.

real-time filtering and search can be applied to reduce the number of live events shown in the display.

The *Spark Service* was operated in local mode on a normal workstation *Dell OptiPlex 980, Core i7-860, 8GB RAM 4x 2.80GHz* with 10 separate working threads. To provide further scalability the service could also be deployed to a cluster of hardware machines running Apache Spark or to a cloud-based deployment. To provide a new sliding slice every 30 seconds, we initialized the system with a batch and slide interval of 30s and a window length of 60s. These settings depend on the general characteristics of the data streams.

To reduce the cognitive load, the analyst decided to switch to the real-time sliding slices visualization as seen in Figure 4 showing an example of five consecutive slices. The interactive display can be explored by the analyst while new slices are continuously added to the right in regular intervals to support situational awareness. The first slice contains critical OSSEC alerts (L5, L10, L3) visualized in a small treemap widget (1). Alerts with a severity of 10 should warn the analyst of ongoing security issues, which should be explored using drill-down functions. Those alerts are related to authentication issues as seen in the word cloud (2). Another treemap widget in the first slice (3) gives an overview of involved programs. The third slice suddenly reveals a high port usage (4), which can be recognized at the port counter. The treemap of source hosts (5) reveals the source host. The analyst can use the IP-Port node-link diagram based on NF (6) to visually explore those suspicious connections.

Later on, the analyst decided to not look on all sliding slices, but to compress the view based on specific features. Figure 5 shows that the analyst is interested in slices with highly critical OSSEC alerts of level 10, segments based on the number of syslog messages received, and based on the number of destination ports utilized in the computer network. Based on this selection the slices are merged accordingly. (1) relates to the segments relating to a port scan. After that, there were no important slices according to the feature selection, so a long time span is merged to a single summary slice (2). The analyst was also interested in the message drop in (3). Then various OSSEC alerts occurred in multiple sliding slices (4). This area seams to be highly suspicious, leading to many individual summary slices to provide more details. Eventually, there are further suspicious events based on NF data in (5) and another peak with OSSEC alerts in (6) related to invalid SSH logins.

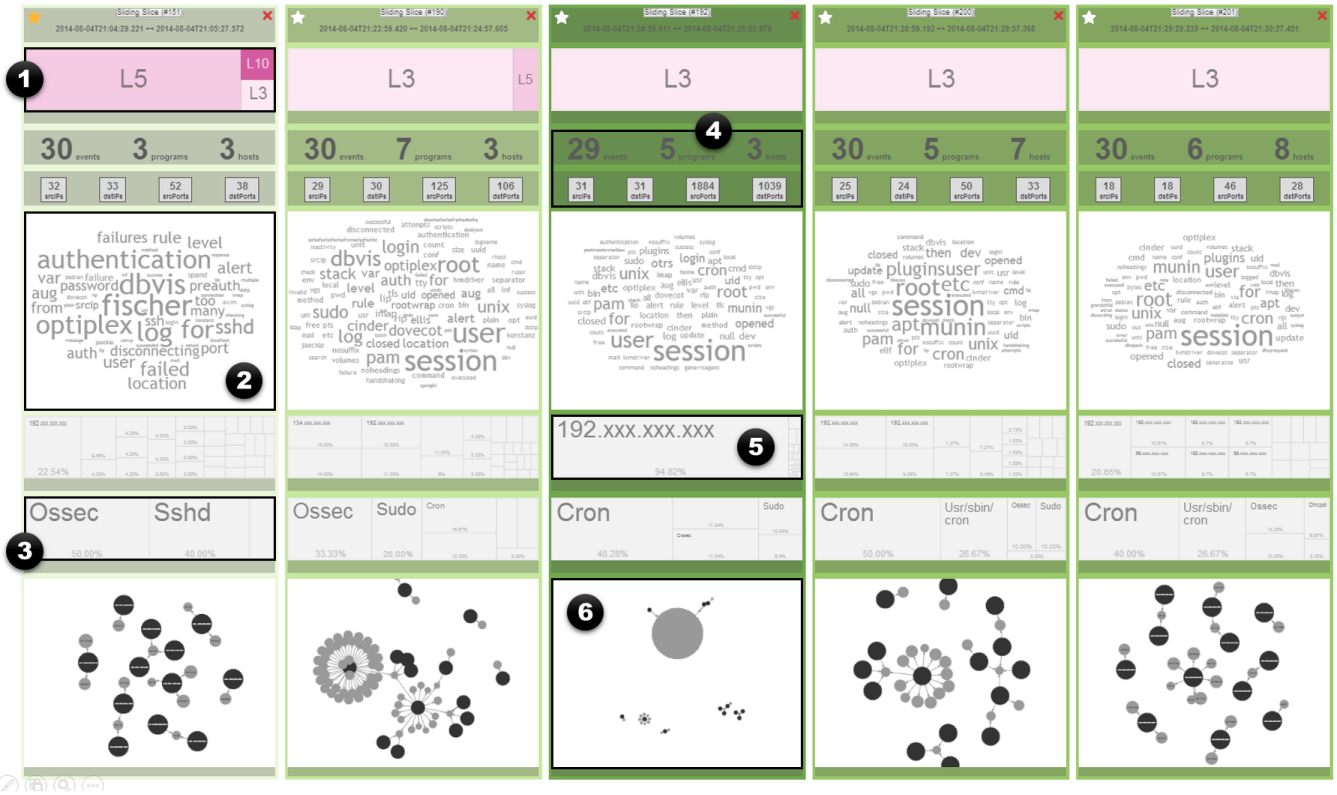

Figure 4: Real-Time Sliding Slices: Visualization to monitor data streams using sliding slices. The interactive display can be explored by the analyst while new sliding slices are continuously added.

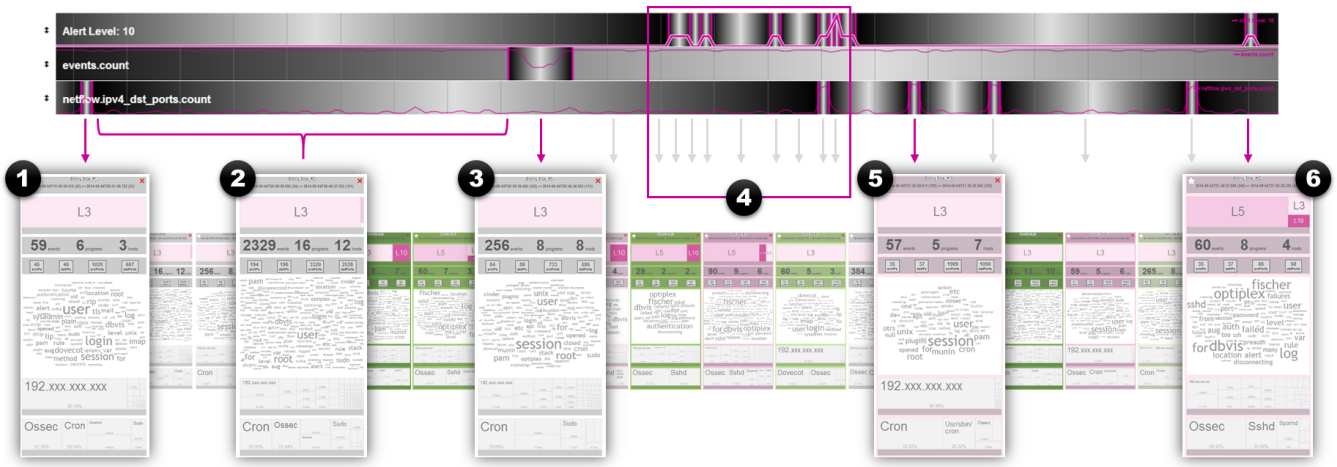

Figure 5: Visual feature selection is used to provide meaningful summary timelines based on merged slices.

5.2 Application for Social Media Streams

We successfully participated in this year's VAST Challenge 2014[5] to evaluate the applicability of our method to solve the given streaming challenge, in particular Mini Challenge 3 (MC3). In this particular challenge, *NStreamAware* won the award for an outstanding comprehensive submission.

The requirement of the challenge was to analyze the data streams made available by the organizers over a *WebSocket* connection in real-time. The data stream contained a stream with micro blog and call center messages covering a time

[5]The VAST Challenge [5] is an international competition with complex tasks that should be solved using novel visual analytics tools.

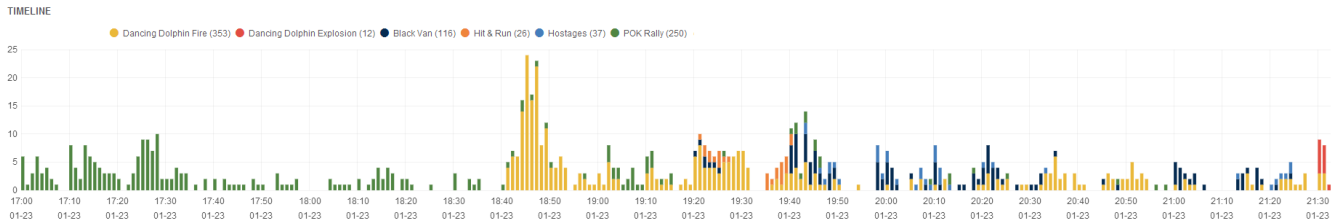

Figure 6: The colored histogram available within *NVisAware* highlights major events based on extracted keywords and insights of interesting events, which were identified by the analyst in real-time.

from 17:00 to 21:30. A first analysis had to be sent to the organizers within three hours after first connecting to the final data stream from 20:00 to 21:30, which could only be streamed once, to force the participants to do real-time processing and provide immediate situational assessment under time pressure.

The fictional but realistic scenario was the so-called *Kronos Incident* in which several employees of a company named *GAStech*, located at the island of *Kronos* went missing. Because of an ongoing conflict between an organization known as the *Protectors of Kronos (POK)*, they are suspected in the disappearance. Within that challenge, the main focus of MC3 was to analyze a real-time data stream based on micro blog records that have been identified by automated filters as being potentially relevant to the ongoing incident and text transcripts of emergency dispatches by the local police and fire departments. During the real-time analysis, we could identify multiple interesting events using *NVisAware* and could make sense of the overall story, as summarized in Figure 6. The first event, which is present from 17:00 until around 19:00, is a rally, organized by POK with different speakers and a concert. Various leaders of POK and many supporting persons are attending this gathering.

At one point an incoming sliding slice caught our attention by the red colored background indicating much changes to the previous slice as seen in Figure 7. And indeed various people are actively talking about a major fire. Using the geographic map, the location can be easily identified, which was the so-called *Dancing Dolphin Apartment*. After firefighters were able to get the fire under control, we realized subtle messages around 21:00 that the fire flared up again, resulting in an explosion around 21:30.

The most important event with respect to the overall situation was an incident related to a black van involved in an hit and run. A biker got hit by the car. In the following, the van was spotted again at another location and could be stopped at *Gelato Galore* by the police leading to a standoff. It turns out that there are two hostages and two kidnappers in the van. In the process one of the suspects shot a police officer. After a while the SWAT team arrived and suddenly the suspects surrendered and could be taken into custody. It turned out that the two hostages were indeed employees of *GAStech*. With this information and insights from the data streams, we were also able to participate in the grand challenge, in which two other mini challenges with additional datasets had to be solved. After combining all the information, we gained a high level of situational awareness and were able to make informed decisions and isolate locations, where the other hostages are likely to be held.

Figure 7: Three consecutive stream slices out of many slices of the data stream to show the evolving topics in the word cloud. The red background color reflects a low similarity value to raise attention.

6. LIMITATIONS AND FUTURE WORK

Various performance measurements and evaluations [20] showed that Spark Streaming is scalable and fault-tolerant. The different database technologies can also scale out horizontally to handle large data volumes. However, the system still needs to be applied to a larger computer network, which is part of the future work. The main limitation with respect to performance and scalability issues could be found in the web application, developed in HTML5 and JavaScript. While the backend is able to use capped collection, and provides data rotation and retention strategies, the real-time graphical user interface does not. When displaying hundreds of sliding slices at the same time the performance decreased, because of browser and memory restrictions of the workstation. However, the responsiveness could be improved by including paging mechanisms in all views and by integrating automatic heuristics to remove old elements. More work needs to be done to keep the display interactive when analysts use the web application for hours without manually removing or reloading some displays.

Automatically defining good sizes for the sliding windows is also planed for the future. The merging model based on the feature selection process, could be applied to the real-time stream in the future, to actually merge sliding slices in real-time, which is not fully implemented yet. Tracking individual events over time was not the focus of this work, however, more work seems to be promising to extend the approach in that respect as well. To comply with privacy

standards and to prevent misuse of such a system, more research should be conducted to integrate privacy-aware technologies, establish multiple access levels, or implementing a two-man rule approach within the visual analytics system.

7. CONCLUSIONS

Based on various design considerations addressing current limitations of related work, we developed a visual analytics system, called *NStreamAware* to analyze data streams. The web application *NVisAware* has been built as integral part of the whole system, because it enabled the analysts to interact with the whole visual analytics system and steer the clustering process to condense data streams to meaningful segments. The system incorporates novel scalable analytics methods (DC1) through Apache Spark Streaming. It enables real-time monitoring with interactive filtering (DC2), which is implemented using RabbitMQ as message broker to provide real-time communication between the different modules. Furthermore, we developed a visualization technique to visually represent the generated *sliding slices*, to present data streams using deterministic screen updates using aggregations on sliding windows (DC3). This approach also makes it possible to combine various heterogeneous data sources (DC4) and include them with various embedded visualizations like word clouds, node-link diagrams, treemaps, and counters. To summarize data streams and to provide a multi-scale timeline to compress the stream based on user-steered interesting functions, we integrated an interactive visual feature selection display (DC5). Eventually, we evaluated our system using data streams of an operational computer network and used our system to successfully participate in the VAST Challenge 2014 to show the applicability for other domains.

8. REFERENCES

[1] C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu. A Framework for Clustering Evolving Data Streams. In *Proceedings of the 29th International Conference on Very Large Data Bases - Volume 29*, VLDB '03, pages 81–92. VLDB Endowment, 2003.

[2] Apache. Spark Streaming. https://spark.apache.org/streaming/.

[3] D. M. Best, S. Bohn, D. Love, A. Wynne, and W. A. Pike. Real-time Visualization of Network Behaviors for Situational Awareness. In *Proceedings of the Seventh International Symposium on Visualization for Cyber Security*, VizSec '10, pages 79–90, New York, NY, USA, 2010. ACM.

[4] D. B. Cid. Open Source Host-based Intrusion Detection System. http://www.ossec.net/.

[5] K. Cook, G. Grinstein, and W. Mark. VAST Challenge 2014 - The Kronos Incident. http://vacommunity.org/VAST+Challenge+2014.

[6] M. R. Endsley. Toward a Theory of Situation Awareness in Dynamic Systems. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 37(1):32–64, 1995.

[7] R. F. Erbacher. Visualization Design for Immediate High-level Situational Assessment. In *Proceedings of the Ninth International Symposium on Visualization for Cyber Security*, VizSec '12, pages 17–24, New York, NY, USA, 2012. ACM.

[8] F. Fischer, F. Mansmann, and D. A. Keim. Real-Time Visual Analytics for Event Data Streams. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, SAC '12, pages 801–806, New York, NY, USA, 2012. ACM.

[9] U. Franke and J. Brynielsson. Cyber Situational Awareness - A Systematic Review of the Literature. *Computers & Security*, 46(0):18 – 31, 2014.

[10] C. Humphries, N. Prigent, C. Bidan, and F. Majorczyk. ELVIS: Extensible Log VISualization. In *Proceedings of the Tenth Workshop on Visualization for Cyber Security*, VizSec '13, pages 9–16, New York, NY, USA, 2013. ACM.

[11] 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.

[12] F. Mansmann, F. Fischer, and D. Keim. Dynamic Visual Analytics - Facing the Real-Time Challenge. In *Expanding the Frontiers of Visual Analytics and Visualization*, pages 69–80. Springer London, 2012.

[13] P. McLachlan, T. Munzner, E. Koutsofios, and S. North. LiveRAC: Interactive Visual Exploration of System Management Time-series Data. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 1483–1492, New York, NY, USA, 2008. ACM.

[14] J. C. Roberts, D. A. Keim, T. Hanratty, R. R. Rowlingson, R. Walker, M. Hall, Z. Jacobson, V. Lavigne, C. Rooney, and M. Varga. From Ill-Defined Problems to Informed Decisions. *EuroVis Workshop on Visual Analytics (2014)*, 2014.

[15] 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, Aug 2012.

[16] G. Tadda and J. Salerno. Overview of Cyber Situation Awareness. In *Cyber Situational Awareness*, volume 46 of *Advances in Information Security*, pages 15–35. Springer US, 2010.

[17] J. van Wijk. Evaluation: A Challenge for Visual Analytics. *Computer*, 46(7):56–60, July 2013.

[18] F. Wanner, A. Stoffel, D. Jäckle, B. C. Kwon, A. Weiler, and D. A. Keim. State-of-the-Art Report of Visual Analysis for Event Detection in Text Data Streams. In *EuroVis - STARs*, pages 125–139, Swansea, UK, 2014. Eurographics Association.

[19] Z. Xie, M. O. Ward, and E. A. Rundensteiner. Visual Exploration of Stream Pattern Changes Using a Data-Driven Framework. In *Advances in Visual Computing*, volume 6454 of *Lecture Notes in Computer Science*, pages 522–532. Springer Berlin Heidelberg, 2010.

[20] M. Zaharia, T. Das, H. Li, S. Shenker, and I. Stoica. Discretized Streams: An Efficient and Fault-tolerant Model for Stream Processing on Large Clusters. In *Proceedings of the 4th USENIX Conference on Hot Topics in Cloud Computing*, HotCloud'12, pages 10–10, Berkeley, CA, USA, 2012. USENIX Association.