WASPP: Workflow Automation for Security Policy Procedures

Ren Quinn, Nico Holguin, Ben Poster, Corey Roach, Jacobus (Kobus) Van der Merwe
University of Utah
renquinn@cs.utah.edu, nico.holguin@utah.edu, ben.poster@utah.edu, corey.roach@utah.edu, kobus@cs.utah.edu

I. Introduction

Every day, university networks are bombarded with attempts to steal the sensitive data of the various disparate domains and organizations they serve. For this reason, universities form teams of information security specialists called a Security Operations Center (SOC) to manage the complex operations involved in monitoring and mitigating such attacks. When a suspicious event is identified, members of the SOC are tasked to understand the nature of the event in order to respond to any damage the attack might have caused. This process is defined by administrative policies which are often very high-level and rarely systematically defined. This impedes the implementation of generalized and automated event response solutions, leading to specific ad hoc solutions based primarily on human intuition and experience as well as immediate administrative priorities. These solutions are often fragile, highly specific, and more difficult to reuse in other scenarios.

We argue that a significant barrier to fully-automating information security practices stems from the lack of systematic solutions to the subproblems of information security. This lack of systematization exists in multiple layers of information security; from user-level processes as described above, down to the lower levels of data flow management and enrichment. Historically, data monitored for security-related events has come from different data sources with different purposes.

We address the lack of automation in a SOC by focusing on the implementation of automated *policy response procedures*. In this work, we refer to policy as predefined regulations which govern acceptable use of the network, while policy response procedures are plans or strategies which define the way in which certain events are interpreted (including how to respond to them) based on the context of the predefined regulations.

We present WASPP, a framework that translates high-level policy response procedures to low-level data flow management and automates their execution. It enables holistic information security monitoring and response driven by modular, user-defined, policy response procedure implementations. This approach allows specific, fine-grained implementations to be used interchangeably for multiple different response procedures in a "plug-and-play" paradigm, allowing the framework to not only streamline procedure implementation but also simplify automating general information security functions.

The contributions of this work are (1) a system architecture driven by four design principles to enable generalized automation of policy response procedures, (2) the implementation of a framework which illustrates the architecture's feasibility, and (3) three case studies which show WASPP's effectiveness.

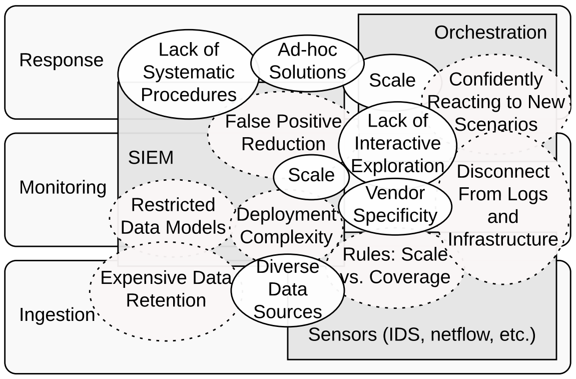


Fig. 1: SOC Architecture with common tools and challenges.

II. Background and Motivation

A. SOC Architecture and Challenges

A Security Operations Center (SOC) is an organization under which security experts work to protect the information systems of an enterprise. Security-relevant data comes into the SOC through various sensors, which is then passed through various monitoring tools used for aggregating, normalizing, enriching, rule-checking, visualizing, and alerting. The goal of this monitoring process is to produce security events which require a responding action, such as adding a firewall rule or deactivating a user's account. This is done by filtering events for those that require special attention, determine the best course of action, and finally execute the action.

Collaborating with the SOC team on our university campus, we formalized the common SOC best practices and challenges into a single diagram (see Figure 1). Three main layers outline the process described above: Ingestion, where raw data is collected, transported, sampled and normalized; Monitoring, which enriches and interprets the data; and Response, where decisions to react to significant events are planned and executed. Within the individual layers, tools have been developed to simplify the process for humans to better understand what is happening on the network. We focus on the most common: sensors, SIEMs, and orchestration. While these tools help SOC teams in protecting users' data, we argue that many other challenges remain which current solutions do not completely overcome. We positioned these challenges in Figure 1 to represent the layers and tools with which they relate.

The fundamental problem we address in this paper is the **lack of systematic procedures** in a SOC. Next, we describe the specific challenges that primarily impede the production of systematic event response solutions in a SOC.

978-3-903176-24-9 © 2019 IFIP

Diverse Data Sources. Analyzing network security events requires tediously combining heterogeneous data sources in order to understand the context of an event. Even with modern tools, a significant amount of manual effort is still required.

Vendor Specificity. Despite existing solutions, a recurring challenge is that most common tools are highly proprietary (and typically standalone). This hinders integration with other tools used in a SOC, locking a team into a particular vendor's ecosystem. However, this limits what a SOC can accomplish as different vendors may provide different features.

Ad-hoc Solutions. Many event response strategies require specific custom implementations. These include writing specific rules for a firewall or other management tool, or writing one-off scripts. This practice creates a complex and fragile collection of highly-specific solutions which, with each new deployment, further increases implementation complexity.

Lack of Interactive Exploration. When an event is acknowledged, its severity must be determined. However, this often requires manual exploration of relevant datasets, with analysts implementing the solution, and making adjustments, as they explore. When a solution is found, the current state of the implementation is used as is resulting in ad-hoc solutions.

Scale. Most individual sensors scale well in their own scope. But when aggregating multiple data sources, the overall scale can be excessive. This necessitates some form of sampling or summarization in order to reduce the data consumption to a manageable size, limiting the analysis that can be performed.

B. *Related Work*

Security Information and Event Managers (SIEMs) are state-of-the-art systems designed to serve as the primary data warehouse for network security information [6]. Custom rule monitoring and event correlation attempt to provide a systematic encapsulation of a SOC's policies and procedures, all through a dashboard interface. However, most of the commonly used SIEMs are proprietary and require organizations to adapt to restricted data models that are not always extensible [31]. This results in deployments with a small number of datasets, leaving users to manually combine events with other datasets that do not fit with their SIEM's paradigm.

Work has been done to specifically address the weaknesses of SIEMs. White papers and blogs touch on the topic frequently [9], [10], [12], [15], [22], [24], [26], [28], [29], but do not fundamentally solve the challenges we address in this paper. Zomlot et al. discussed the common use of SIEMs and their challenges [6]. Sapegin et al. integrate advanced anomaly detection with a custom-built SIEM [23]. Terzi et al. propose a Big Data Analytics system that performs anomaly detection using machine learning as an alternative to SIEMs [30]. Other works have focused on simply extending SIEMs in order to address a subset of specific SIEM weaknesses [11], [21], [27]. Our work has a larger scope than these works by addressing the fundamental needs of a SOC.

VAST [31] is designed to replace SIEMs using large-scale IDS log processing. VAST uses pipelines to execute performant, low-level queries on network data, distributing query execution functions such as indexing and archiving using the actor model. In contrast, WASPP uses pipelines to orchestrate and compose multiple queries at a higher level to form automated response policy procedures. WASPP is built on top of a scalable data management and query processor, and VAST could serve as this underlying structure for WASPP.

III. Design Principles and Abstraction

To solve the challenges of information security in a general manner, we need to fundamentally rethink the workflows that drive policy response procedure execution. If we can systematize the individual functions of a SOC, we can provide holistic automation. In this section, we present a set of principles by which automated SOC response systems should be designed in order to overcome the challenges discussed above. In order to complement these design principles, we discuss a new abstraction for systematically encoding response procedures.

A. *Design Principles*

Combining an understanding of the challenges with knowledge of what is currently being done to address them, we suggest four necessary principles that should guide any fundamental design decisions of a holistic information security automation system. The specific challenges addressed by each principle are printed in small caps to help identify them.

Unified Systematic Interfaces. Ideally, there should be a single interface to all types of data. This improves integrating DIVERSE DATA SOURCES as well as VENDOR SPECIFIC tools. It also simplifies the DEPLOYMENT COMPLEXITY of integrating new tools that analyze the data.

Modularity with Composability. Any piece of a policy procedure should available for systematic reuse in any scenario. Such pieces include matching, aggregating, normalizing, and enriching data, as well as executing response actions. This reduces the tendency for AD-HOC SCRIPTING of response functions in favor of more SYSTEMATIC PROCEDURES.

Flexibility with Extensibility. SOC teams need enough primitive operations to systematically define automation strategies for most of the tasks associated with policy response procedures. When this toolset fails to cover certain scenarios, analysts should be able to define custom components of the procedure implementation in the same systematic manner as the primitive operations. If a system is flexible and extensible, implementing functions such as analytics-based enrichment or custom response actions becomes simpler and more conducive to SYSTEMATIC PROCEDURES and automation.

Scalability. Any modern management platform should SCALE to withstand the increasingly large load of network traffic and security alerts. When the system scales, sampling and summarization are less important because more resources are available to handle larger volumes of incoming data. This is especially enticing to prevent the use of RESTRICTED DATA MODELS which are naturally imposed by normalization.

B. Systematic Response Pipeline Abstraction

In order to materialize our design principles, we propose a new abstraction by which we can automate policy response procedures, *systematic pipelines*. Inspired by traditional workflow diagrams (often used statically in SOCs to describe procedural processes), a systematic pipeline mimics the way security analysts reason about responding to events.

Pipelines are composed of a sequence of **nodes**, each performing a fine-grained operation which contributes to the overall event investigation and response process. Three different node constructs make up a pipeline: source nodes, intermediate nodes, and terminating nodes. Source nodes are the entry points to pipelines, forwarding records from external tools to other nodes. Intermediate nodes ingest records from other nodes (whether source or intermediate), performing incremental processing of the data passing through them (i.e., data enrichment, correlation, filtering or analytics), and emit modified, filtered, or new records that other nodes can further ingest and interpret. Terminating nodes only ingest data, usually triggering a response action to the alert as needed.

The **edges** between nodes in a pipeline represent data flow; directing data through dynamic data streams. Events and records passing through a pipeline are incrementally filtered and enriched until the right level of detail is obtained in order to respond according to policy. The modularity and flexibility of a pipeline is evident by the lightweight compositional structure of edges directing data between different nodes.

IV. WASPP Architecture

We designed the WASPP architecture to enable simple yet general specification of security policy response procedures with the goal of increased automation, through the construction of systematic pipelines. Figure 2 illustrates the WASPP architecture in the context of the SOC architecture. The workflow begins with users delivering systematic response procedure pipelines through the Pipeline Specification component at the top of the architecture. Here the user can access metadata about the pipeline nodes being managed by the system, and then reason about implementing response procedures as a pipelines. The user-defined pipelines are interpreted, translated to low-level data flow queries, and executed, all in Pipeline Orchestration, a process based on different Node Behavior implementations. Each Node Behavior builds on the functionality provided by the data management platform, particularly the Declarative Query Engine. The following subsections explain in more detail how each of the components of the WASPP architecture respectively follows our design principles.

A. Unified Systematic Interfaces

Data Stream Management. At the bottom of WASPP, Data Streams represent the flow of records from heterogeneous data sources in the SOC, directed through a single platform with a unified, systematic interface. This establishes data streams as the primary low-level data abstraction, a foundation on which to build high-level abstractions that simplify user interaction.

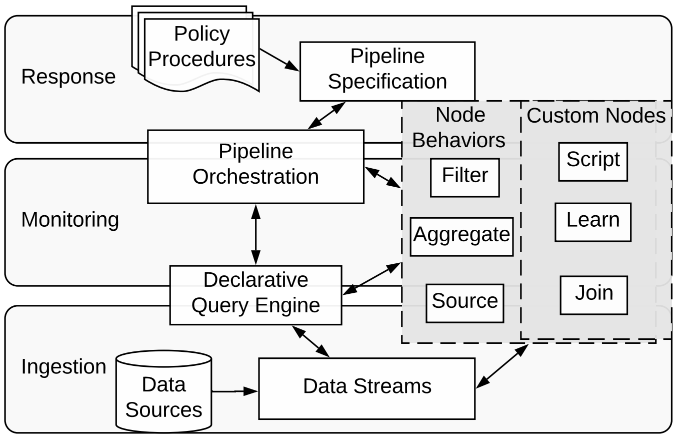


Fig. 2: WASPP system architecture.

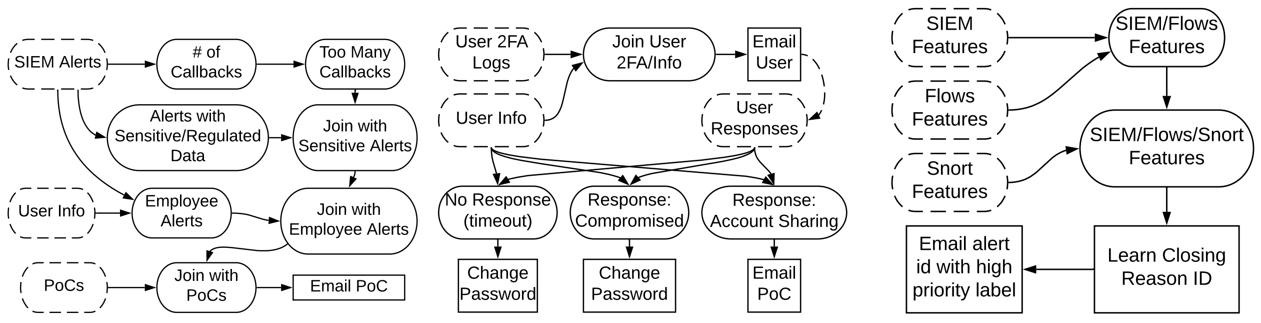
Declarative Query Engine. With the foundation of a unified, systematic platform managing multiple heterogeneous datasources, it is easier to build simple, yet powerful, user-level abstractions. The Declarative Query Engine provides flexibility to balance between systematic interfacing and user-level reasoning. It simplifies the filtering, aggregating, etc., of the low-level data streams while maintaining the unification provided by the Data Stream Management platform, without restricting to any particular data model. Work has shown that declarative interfaces simplify network management by reducing the need to reason about how data operations are carried out [1], [7], [8], [13], [14], [17]–[20], [25], [32]. This becomes a core building block on which we systematize various functions in a SOC.

Pipeline Specification. With a standardized, dynamic interface for running queries over diverse data streams, it is easier to dynamically orchestrate these queries as part of pipelines. This component allows users to design pipelines according to our design principles. An advantage to defining response procedures as pipelines is their systematic representation of the way SOC processes are conceptualized, as step-by-step sequences of finely-grained operations. Pipelines are also inherently modular, each node in the pipeline represents an atomic step in the data flow that can be used by other pipelines.

Pipeline Orchestration. This component essentially performs the necessary plumbing between the user-level and system-level interface abstractions, in order to provide automated execution of user-defined policy procedures. This includes translating the high-level node definitions to low-level streams, as well as driving dynamic stream management.

B. Modularity with Composability

Pipeline Specification and Orchestration. Data pipelines, as a model for policy procedures, inherently provide modularity and composability. Nodes of a pipeline can be diverse, and re-used in different pipelines. Users define pipelines with various nodes, according to their respective functionality.



(a) SIEM Response (b) 2FA Monitoring (c) Learning SIEM Response Codes

Fig. 3: WASPP pipelines for three case studies.

C. *Flexibility with Extensibility*

Pipeline Orchestration and Node Behaviors. Composing node behaviors in the pipeline abstraction enables the implementation of diverse policy procedures. While different node types follow different design principles, composing them together provides flexibility and extensibility. The WASPP framework defines three specific core node types for basic pipeline functionality: **Source Nodes** set up data streams, usually from an external tool. **Filter Nodes** apply rules to incoming records, forwarding those that match given criteria. They also reduce records, focusing on a specific portion of each record. **Aggregate Nodes** summarize input streams.

Policy procedures may require more functionality than basic data transformations. For example, analytics-based enrichment, real-time stream joining, and scripted actions are important functions within a SOC that generally lack systematic functionality. When properly systematized, these functions can be modularly defined and dynamically composed in a flexible and extensible manner. In our implementation of the WASPP framework, we implemented three custom nodes to demonstrate such extensibility. **Learn Nodes** enable analytics-based enrichment by reading streams, performing complex computations, and producing learned observations to a new stream, all defined by a user via Policy Specification. **Join Nodes** combine streams in real time. **Script Nodes** encapsulate a dedicated process which executes user-defined, scripted actions upon receiving a trigger via a data stream. For example, this could be to update a firewall rule or send an email.

D. *Scalability*

Data Stream Management. Providing scalability in a SOC means handling high volumes of raw data in a cost-effective way. In order to establish a unified interface at the system level, the Declarative Query Engine should sit above a scalable system capable of high throughput and low latency data flow. Data Streams should be handled by a highly scalable message passing system or stream processing framework [2]–[5], [16].

V. Evaluation

We evaluate WASPP through three case studies which demonstrate the design principles discussed in Section III. Each case study is implemented as a WASPP pipeline inspired by policy response procedures used in our university SOC, and tested with data collected from our university campus network. Diagrams representing the pipelines are presented in figure 3.

1) SIEM Monitoring: A common SOC practice is deciding when a SIEM alert merits response, which is often a matter of false positive reduction. For this case study, we automate this process by filtering alerts according to a series of rules. The first rule checks if the host/user mentioned in the alert has been the focus of a previous alert. If so, we check if the user was an employee of the university, or if the event concerns sensitive or regulated data. If any record matches these rules, the alert is likely a high priority and is worth responding to. At this point, the pipeline notifies the Point of Contact (PoC) for the particular host or user mentioned in the alert.

2) User 2FA Logs: In order to combat attempts to compromise user accounts, especially those of employees or others with access to sensitive or valuable information, adoption of 2-Factor Authentication (2FA) is growing. Therefore, it is important to identify any potential breech of the 2FA. However, due to the unpredictable nature of human behavior, it is difficult for an automated process to understand scenarios such as when a user login event is the result of a user sharing their credentials with a family member who is in another country, or if the account was compromised. Therefore, a human usually monitors the alerts, and manually asks the user about the situation when needed. For this case study, we automate the steps a human would perform in this process.

3) SIEM Alert Interpretation Learning: Some SIEMs can annotate alerts with a user-defined response code which reflects how a security analyst interpreted and responded to each alert. This serves as a history of how the SOC handles certain situations which is useful for auditing. It also provides a reference for in the case that similar issues occur in the future, or to train new analysts by showing them how to interpret patterns in alerts. We used these response codes as ground truth labels for machine learning to automate the decision-making process. For feature selection, WASPP's pipeline approach fits naturally as we can filter particular fields and seamlessly combine different data sources.

References

[1] P. Alvaro, T. Condie, N. Conway, K. Elmeleegy, J. M. Hellerstein, and R. Sears, "Boom analytics: Exploring data-centric, declarative programming for the cloud," in *Proceedings of the 5th European Conference on Computer Systems*, ser. EuroSys '10. New York, NY, USA: ACM, 2010, pp. 223–236. [Online]. Available: http://doi.acm.org/10.1145/1755913.1755937

[2] Apache Software Foundation, "Flink." [Online]. Available: https://flink.apache.org

[3] ——, "Samza." [Online]. Available: https://samza.apache.org

[4] ——, "Spark streaming." [Online]. Available: https://spark.apache.org/streaming/

[5] ——, "Storm." [Online]. Available: https://storm.apache.org

[6] S. Bhatt, P. K. Manadhata, and L. Zomlot, "The operational role of security information and event management systems," *IEEE security & Privacy*, no. 5, pp. 35–41, 2014.

[7] X. Chen, Y. Mao, Z. M. Mao, and J. Van der Merwe, "Declarative configuration management for complex and dynamic networks," in *Proceedings of the 6th International COnference*. ACM, 2010, p. 6.

[8] X. Chen, Y. Mao, Z. M. Mao, and J. E. van der Merwe, "Knowops: Towards an embedded knowledge base for network management and operations." in *Hot-ICE*, 2011.

[9] A. Chuvakin, "Lets define "siem"!" https://blogs.gartner.com/anton-chuvakin/2017/08/14/lets-define-siem/, 2017.

[10] K. CRAWLEY, "How siem correlation rules work," https://www.alienvault.com/blogs/security-essentials/how-siem-correlation-rules-work, 2018.

[11] M. Di Mauro and C. Di Sarno, "Improving siem capabilities through an enhanced probe for encrypted skype traffic detection," *Journal of Information Security and Applications*, vol. 38, pp. 85–95, 2018.

[12] H. Eshagh, "The elasticsearch siem architecture of a nonprofit: Security at the nature conservancy," https://www.elastic.co/blog/elasticsearch-siem-architecture-of-a-nonprofit-security-at-the-nature-conservancy, 2017.

[13] N. Foster, M. J. Freedman, R. Harrison, J. Rexford, M. L. Meola, and D. Walker, "Frenetic: A high-level language for openflow networks," in *Proceedings of the Workshop on Programmable Routers for Extensible Services of Tomorrow*, ser. PRESTO '10. New York, NY, USA: ACM, 2010, pp. 6:1–6:6. [Online]. Available: http://doi.acm.org/10.1145/1921151.1921160

[14] T. L. Hinrichs, N. S. Gude, M. Casado, J. C. Mitchell, and S. Shenker, "Practical declarative network management," in *Proceedings of the 1st ACM workshop on Research on enterprise networking*. ACM, 2009, pp. 1–10.

[15] K. M. Kavanagh, O. Rochford, and T. Bussa, "Magic quadrant for security information and event management," *Gartner Group Research Note*, 2015.

[16] J. Kreps, N. Narkhede, J. Rao *et al.*, "Kafka: A distributed messaging system for log processing," in *Proceedings of the NetDB*, 2011, pp. 1–7.

[17] C. Liu, B. T. Loo, and Y. Mao, "Declarative automated cloud resource orchestration," in *Proceedings of the 2Nd ACM Symposium on Cloud Computing*, ser. SOCC '11. New York, NY, USA: ACM, 2011, pp. 26:1–26:8. [Online]. Available: http://doi.acm.org/10.1145/2038916.2038942

[18] B. T. Loo, T. Condie, M. Garofalakis, D. E. Gay, J. M. Hellerstein, P. Maniatis, R. Ramakrishnan, T. Roscoe, and I. Stoica, "Declarative networking: Language, execution and optimization," in *Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '06. New York, NY, USA: ACM, 2006, pp. 97–108. [Online]. Available: http://doi.acm.org/10.1145/1142473.1142485

[19] ——, "Declarative networking," *Commun. ACM*, vol. 52, no. 11, pp. 87–95, Nov. 2009. [Online]. Available: http://doi.acm.org/10.1145/1592761.1592785

[20] T. Nelson, A. D. Ferguson, M. J. G. Scheer, and S. Krishnamurthi, "Tierless programming and reasoning for software-defined networks," in *Proceedings of the 11th USENIX Conference on Networked Systems Design and Implementation*, ser. NSDI'14. Berkeley, CA, USA: USENIX Association, 2014, pp. 519–531. [Online]. Available: http://dl.acm.org/citation.cfm?id=2616448.2616496

[21] E. S. Novikova, Y. A. Bekeneva, and A. V. Shorov, "Towards visual analytics tasks for the security information and event management," in *" Quality Management, Transport and Information Security, Information Technologies"(IT&QM&IS), 2017 International Conference*. IEEE, 2017, pp. 90–93.

[22] J. Oltsik, "Goodbye siem, hello soapa," https://www.csoonline.com/article/3145408/data-protection/goodbye-siem-hello-soapa.html, 2016.

[23] A. Sapegin, D. Jaeger, F. Cheng, and C. Meinel, "Towards a system for complex analysis of security events in large-scale networks," *Computers & Security*, vol. 67, pp. 16–34, 2017.

[24] A. Sathye, "Siem what next?" https://www.happiestminds.com/blogs/siem-what-next/, 2017.

[25] W. Shen, A. Doan, J. F. Naughton, and R. Ramakrishnan, "Declarative information extraction using datalog with embedded extraction predicates," in *Proceedings of the 33rd International Conference on Very Large Data Bases*, ser. VLDB '07. VLDB Endowment, 2007, pp. 1033–1044. [Online]. Available: http://dl.acm.org/citation.cfm?id=1325851.1325968

[26] K. Sheridan, "Future of the siem," https://www.darkreading.com/threat-intelligence/future-of-the-siem-/d/d-id/1328457, 2017.

[27] G. Suarez-Tangil, E. Palomar, A. Ribagorda, and Y. Zhang, "Towards an intelligent security event information management system," 2014.

[28] D. Swift, "Successful siem and log management strategies for audit and compliance," *SANS Institute Infosec Reading Room, November*, vol. 4, p. 40, 2010.

[29] A. Teixeira, "Get over siem event normalization," https://medium.com/@ateixei/get-over-siem-event-normalization-595fc36559b4, 2017.

[30] D. S. Terzi, R. Terzi, and S. Sagiroglu, "Big data analytics for network anomaly detection from netflow data," in *Computer Science and Engineering (UBMK), 2017 International Conference on*. IEEE, 2017, pp. 592–597.

[31] M. Vallentin, V. Paxson, and R. Sommer, "Vast: A unified platform for interactive network forensics." in *NSDI*, 2016, pp. 345–362.

[32] W. Zhou, M. Sherr, T. Tao, X. Li, B. T. Loo, and Y. Mao, "Efficient querying and maintenance of network provenance at internet-scale," in *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '10. New York, NY, USA: ACM, 2010, pp. 615–626. [Online]. Available: http://doi.acm.org/10.1145/1807167.1807234