The 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications
21-23 September, 2017, Bucharest, Romania

Combining Network Access Control (NAC) and SIEM functionality based on Open Source

Prof. Dr. K.-O. Detken [1], M. Jahnke [1], Prof. Dr. C. Kleiner [2], M. Rohde[2]
[1] DECOIT GmbH, Fahrenheitstraße 9, D-28359 Bremen, detken/jahnke@decoit.de, http://www.decoit.de
[2] University of Applied Sciences and Arts of Hanover, Ricklinger Stadtweg 120, D-30459 Hanover,
carsten.kleiner/ marius.rohde @hs-hannover.de, http://www.hs-hannover.de

Security Information and Event Management (SIEM) solutions are expensive, difficult to handle, and still a further security system without interface to existing Network Access Control (NAC) or other security components of an enterprise. Thus, the research project CLEARER [2] aims at developing a security system, which extends existing NAC systems with SIEM functionality, additional analyzing methods and dynamical compliance support. This goal will only be reached by using Open Source Software (OSS). The expected result, a common security platform for attack prevention will be affordable for small and medium-sized enterprises (SME) in contrast to other vendor-based SIEM systems.

Keywords: NAC, SIEM, Open Source, IF-MAP, Bro, Esper, Banana, Cassandra, Solr, CBOR, ironVAS, AMQP.

I. INTRODUCTION

Network Access Control (NAC) systems are used for computer and endpoint security with anti-virus, host intrusion prevention, and vulnerability assessment mechanisms in enterprise environments. Such systems control user and system authentication. Additionally, security enforcement is a core feature, but an analysis of IT security attacks or risk management is out of scope of these systems. Therefore, *Security Information and Event Management (SIEM)* systems have been developed to find out how secure an enterprise network really is.

In detail Network Access Control (NAC) systems work as a computer-based networking solution with a set of protocols to define and implement a central policy for an enterprise. This policy describes how to secure access to network nodes by devices when they initially attempt to access the network. Additionally, NAC mechanisms integrate an automatic remediation process, which includes fixing non-compliant nodes before allowing access into the network. Furthermore, NAC allows the operation of network components like routers, switches, and firewalls with server systems and client-computer equipment to ensure the IT platform is operating securely. A basic standard of NAC is IEEE 802.1X. This specification works port-based, provides authentication mechanisms to devices, and defines the Extensible Authentication Protocol (EAP) to support service identification and point-to-point encryption, if needed.

Therefore, the main goal of a NAC system is authorization, authentication, and accounting of network connections. In addition, role-based control of users, devices, or applications are possible. The policies include types of computers and roles allowed to access the network and enforce them on switches, routers, and other network components. While conventional IP networks enforce access policies in terms of IP addresses, NAC systems attempt to do so, based on authenticated user identities for end-station like laptops or desktop computers. Consequently, the main benefit of NAC systems is to prevent potentially malicious or infected devices from entering the network in order to keep other components of the network clean.

Nevertheless, a NAC system is not comparable with a SIEM system directly. Because SIEM technology provides in detail near real-time analysis of security alerts, which have been generated by network hardware and applications. SIEM systems can be used as software, appliances or managed services, and are also used to log security data and generate reports for compliance purposes. The objective of SIEM is to help companies respond faster to attacks and organize mountains of log data. A key focus is to monitor and help to manage user and service privileges, directory services and other system configuration changes, as well as to provide log auditing as well as review and incident response. [1]

That means, if enterprises only use a NAC system, important functionality to analyze security mechanisms is still missing. Denying devices access to the network does not prevent all potential security threats. Actually, many threats with high potential of damage consist of complex attack processes where individual steps are executed on devices that are honestly connected to the network. Therefore, it is necessary to find out what is going on within the network in real-time. Additionally, it should be essential to define rules for the allowed network access and services for any user. These rules based on IT compliance should be changeable dynamically if the attack situation varies, but are fixed in most systems.

In order to combine those important features, the research project *CLEARER* [2] has been started. Many enterprises have a NAC or similar security system in the field, but are still missing the functionality of real-time analysis of network traffic, user authentication, or network attacks. Additionally, the existing NAC system has to work together with other SIEM systems, if an enterprise wants to extend its security strategy.

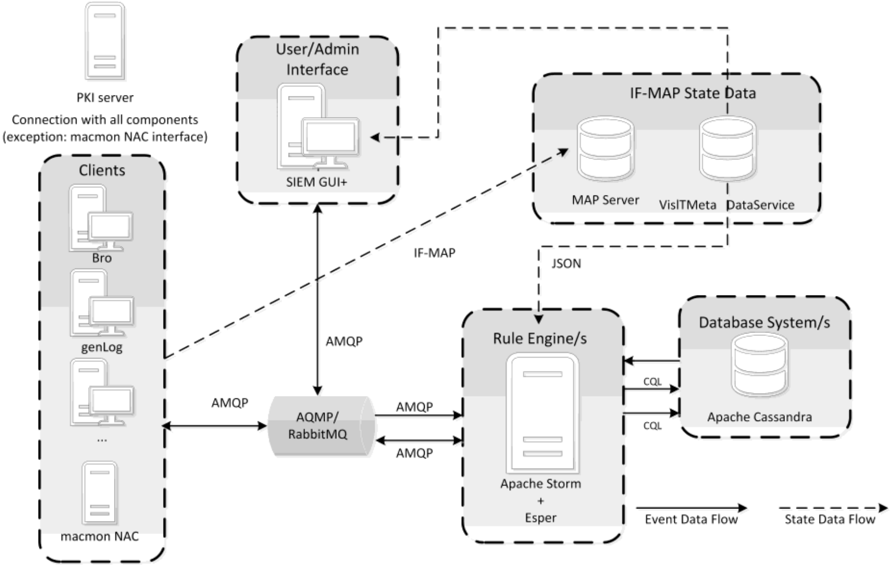

Figure 1. Architecture overview of CLEARER

Therefore, CLEARER worked out a SIEM extension for existing NAC systems to bring all the advantages together. Furthermore, the project works on a dynamical IT compliance implementation to react on new attacks and rule changes faster than before. All components are based on open source software, which makes this solution even affordable for small and medium-sized enterprises (SME).

II. Related Work

A. *Other SIEM-Solutions*

The Gartner 2016 SIEM Magic Quadrant study [16] presents an overview about available commercial SIEM-approaches. The study shows that a few big manufacturers dominate the market and most of them focus on major enterprises. Therefore, the amount of purchasable candidates for small companies is little. Two of the most promising solutions for small companies are the *Open Source Security Information Management (OSSIM)* system of the manufacturer AlienVault [17] and the Open-Source-based SIEM of *rt-solutions.de GmbH* [18]. OSSIM does not provide a multi-server installation. Therefore, companies have to be aware of the necessity of more and more expensive hardware or even new, horizontal scaling, products, to keep step if the company grows. Additionally, OSSIM does not support log-management. Thus, important information sources cannot be integrated into the SIEM infrastructure. Finally OSSIM is not anymore native open source software, but a proprietary system by AlienVault. Therefore open interfaces are not available and own software extensions cannot be implemented.

The Open-Source-based SIEM system of the rt-solutions.de GmbH is relatively similar to the CLEARER approach. It provides a comprehensive SIEM system with analyzing functions and a log-management linkage. However, in contrast to the open-source-based SIEM system of rt-solutions, CLEARER focusses on compliance. CLEARER connects NAC with SIEM and supports the enforcement of compliance policies in the company's IT. In addition, to the best of our knowledge, no open source SIEM system provides a connection to the protocol *Interface for a Metadata Access Point (IF-MAP)* [19].

B. *Scientific Work*

The area of complex event processing is already known for some time and scientific papers in this area have led to the development of processing technologies. One example of such a complex event processing software is Esper which will be used in the architecture of the CLEARER system as described in section III.

The area of detection and classification of distributed events has also been in the focus of many research papers. But most papers focus on different domains and the detection and classification mechanisms are domain-specific. Article [21] is concerned with pattern recognition in event streams for security monitoring. The main idea is a parallelization of queries in order to deal with the high frequency. Our approach is different as we rely on the automatic parallelization features of the underlying database system Apache Cassandra as will be explained later; this had not been available for the work in [21].

In contribution [22] the authors describe an approach to efficiently execute extremely large sets of detection rules for security incidents. This approach may complement our work, but the focus in the CLEARER project is presumably on huge sets of available data that has to be analyzed and less in the number of rules. As the rules are defined to take care of certain compliance requirements the number of rules is probably not as large, but the data sets are. Similarly, the focus of article [23] is also on an efficient execution of the detection rules and less on how to learn and define these rules beforehand.

Some research papers are also concerned with learning, prioritization and assessment of detection rules in network security ([24], [26]), but they focus on IDS systems delivering base data which had been state of the art at that time. While the general ideas are still valid, the use of SIEM and NAC systems pose completely new challenges in terms of both available information as well as action options.

Contribution [25] describes a framework for event prioritization that is similar to concepts required in our project, but they are also not based on the SIEM system data available for us.

Finally, none of the described approaches integrates event detection and classification with taking concrete actions based on the integration of a NAC system as is the focus of our work.

III. Architecture of CLEARER

The *architecture of CLEARER* (see figure 1) is roughly adapted from the lambda architecture [20] for big data computing frameworks. As within the lambda architecture, there is a real-time computation path for events and a batch like computation path for IF-MAP state information. In addition, historical events can be reemitted in the event-stream-framework Storm to be reprocessed in a batch like manner. The events generated by the clients are transmitted using an AMQP message queue to the storm cluster. The IF-MAP state information is directly sent to the MAP Server for further processing. Subsequently, a detailed description of the architecture is given, including the following main components:

- SIEM GUI+
- Ticket system
- NAC system
- Banana auditing tool
- Rule engine with Apache Storm and Esper
- Apache Cassandra database system
- IF-MAP state data, including MAP server and VisITMeta data service
- Bro
- Private Key Infrastructure (PKI)

The central web-based graphical user interface (GUI) for user or administrator is named *SIEM GUI+*. It is based on the work of another research project (SIMU [1]) and will be the main interface for administration and analysis. Especially, user rights management should be integrated for different access rights. Additionally, the LDAP based user management should be an integral part of this GUI. Via a dashboard, relevant events and alarms should be identifiable and the compliance log of the new system should be visible. A system status view will show an overview about active sensors and other system components like real-time information of *RabbitMQ* and *Apache Storm*.

A *ticket system* will be integrated to manage all tasks for the user and give hints of new alarms or events. User and ticket management is an integral part of the SIEM GUI+ and a complete new development.

The NAC system is a third party system and manages the defined policy. That can be the NAC system of *macmon secure* or another solution like *PacketFence* as open source software. In both cases, an interface adaptation is necessary, like REST or CLI.

As *auditing tool*, *Banana* has been chosen. Log information is visible in an efficient way and unlike the status display of the SIEM GUI+ it contains a historical view of log information. The log view of Banana includes free configuration of diagrams and reports. By this strict separation, a simpler isolation of data pseudonymization on the status view is possible. For logging, for all relevant base data and events *Apache Cassandra database* will be used. *Apache Solr* [13] ensures efficient search functionality. Solr is a powerful search engine like ElasticSearch, but can be used without additional Solr cluster for data storage. Banana has its own GUI and offers a user-friendly interface for initialization of diagrams and reports. Reports can involve arbitrary levels of detail. Consequently an end-result based drill down to the source data is possible.

Two systems are the central communication point of this architecture. Regarding status information, an *MAP server* is available which includes current system status of all involved components. The MAP graph history may be relevant for an analysis. Therefore, the data service of *VisITMeta* is available to get access to states of the past.

All other internal communication will be carried out by the *RabbitMQ message broker* [15] and the *AMQP* protocol. AMQP stands for advanced message queuing protocol and is an open standard for message-oriented middleware. The defining features are message orientation, queuing, routing, reliability, and security. Especially event data and correlation results are transferred by AMQP. Depending on the use case, different AMQP exchanges and message queues can be used to simply send a message to many interested recipients. They only have to subscribe to the AMQP exchanges on the RabbitMQ.

The event correlation is the central module of the CLEARER architecture. This module works with raw data from different sensors and recognizes relevant patterns with help of a policy. By these patterns, pre-defined and abstracted information will be generated. This information will be used by the rule engine to find out rule breaches with regard to the existing enterprise compliance.

The framework *Apache Storm* [8] is used, because event data has to be filtered before they will go through the correlation. It is a free and open source distributed real-time computation system. Storm makes it easy to reliably process unbounded streams of data for real-time processing. Storm allows controlling data streams within a cluster of computing machines. Therefore, Storm is highly scalable and uses concepts of Spouts (data source) and Bolt (data handling) to define a topology. Within this topology, lots of Spouts and Bolts can be linked with each other for data stream processing. The only limitations are that Spouts can just deliver data without receiving it and Bolts cannot use cyclical connections.

Bro [14] is an open source network-monitoring framework. It is also a security monitor that inspects all traffic on a link in depth for malicious traffic. Basically, *Bro* is an intrusion detection system (IDS). In addition, it has some more features, like collecting network measurement, conducting forensic investigations or traffic baselining.

The term *personal key infrastructure (PKI)* does not mean a classic but a self-developed PKI infrastructure. No usual PKI can be used in this project, because the certificates are located in Java key-stores exclusively. Furthermore, the entire PKI functionality is not required. The CLEARER components are delivered with initial certificates, which enable them to communicate with the developed PKI and receive the production keys directly.

The CLEARER topology will be provided by data via AMQP Spouts from RabbitMQ directly. Exchanges will be subscribed to and the broker delivers event data. Spouts generate Storm tuples from the received data and send it to the topology for processing. The tuple is the main data structure in Storm. A tuple is a named list of values, where each value can be of any type. Storm Bolts will realize the data correlation. The received event tuples from Bolts will be converted in well-known data format for the correlation with the policy.

Beside the bolt correlation, further Bolts will be required for the output of correlation results to the RabbitMQ broker and for pre-filtering of events. This is necessary in order to filter events, which do not fit into the maximum time distance. As the bolts need to access historical event information contained by the event database, this late event information needs to be handled separately.

Regarding the event correlation, the tools *Simple Event Correlator (SEC)* [9], *Esper* [10], and *Drools Fusion* [11] were evaluated. The tool SEC is written in Perl and used mainly Regular Expression (RegEx) based rules. If more complex rules need to be created, e.g. for multistate analysis, it is necessary to use Perl code for rule definitions. That is not practicable and difficult to handle. Esper and Drools Fusion are very similar regarding functionality. They differ on the rule definitions and how these rules can be integrated into other systems. Drools Fusion uses a Java-similar language called Domain-Specific Language (DSL) for rule definitions. In contrast, Esper works with a SQL-similar query language. Drools Fusion is an integral part of the Drools Suite. This is a business rules management system and Drools Fusion cannot be used without it. However, Esper is a standalone correlation engine, which can be used independently from other components within CLEARER and is thus used in the project. Esper is also used as assessment engine because of its enormous correlation possibilities. For scalability, Esper agents can be deployed on different machines in order to process larger amounts of data.

As a main database requirement, horizontal write scalability is necessary due to the many different sensors generating data. That is the reason why *Apache Cassandra* [12] has been chosen. Cassandra offers horizontal scalability, fast write rates, and basic security features. Several parameters like consistence, availability, partition tolerance, write and read performance are fine-tunable on a per request basis. Additionally, the database management system (DBMS) must be able to support an audit mechanism for compliance reasons. Therefore, all relevant events as well as correlated events have to be stored in the database, which will be a huge amount of data. Besides data storage, the Complex Event Processing (CEP) engine communicates per Cassandra Query Language (CQL) with the DBMS to analyze and process events with help of historical information. For compliance reasons, all events of the whole architecture have to be stored immutable. Due to the immutable storage of events, an auditor can be certain that the system is not manipulated by careless employees, or attackers who want to erase their tracks. Thus, database manipulation has to be logged for every operation. These logs are an integral part of the database, but only accessible by administrators. An auditor could directly identify event manipulations. Consequently, the database is a crucial element of the implementation of compliance requirements.

To ensure fast write requests all events will be written to only one node of the Cassandra cluster. The replication of events is performed by the database system itself. To ensure a high eventual read consistency, the read request is performed on n/2+1 (quorum) Cassandra nodes. In consequence of the immutability of all events, high consistency should be achieved.

All events will be forwarded directly from the correlation component Apache Strom to the Cassandra database. The connection takes place by the integral plugin for Apache Storm and allows a lossless connection to Cassandra. However, all IF-MAP status data are stored into the data-service of VisITMeta. A connection from CEP engine Esper and the correlation of Apache Storm to the data-service of VisITMeta will be developed.

IV. Data Exchange Format

As data exchange format for the event data between the different *IF-MAP clients* and the rule engine a standard format is necessary. It has to distinguish between the format for the content and the format for the serialization. The content format describes which data structure will be transferred (e.g. event data includes two fields as “time stamp of the event” and “information about the recognized attack”). The other format of serialization includes how this content has to be coded for the transmission. Text-based formats like XML or JSON can be used, but for huge amounts of data a binary format is necessary which is more compact.

As content format, the project CLEARER uses the *intrusion detection message exchange format (IDMEF)* [4]. This format is used with slight changes, because in the original encoding it used a text-based XML format. Additionally, it is not clear yet which data format of the employed sensors will be used. Therefore, the content format should be very close to the specification of IDMEF in order to reduce the effort regarding adaption of the available XML representation. Furthermore, the compatibility is important for communication with other systems and interfaces.

Finally, the format of IDMEF offers a good basis to define a more specific content format, because it is possible to describe many necessary data items efficiently like ID of a sensor, timestamp of an event, or timestamp of the recognition of source and target. An overview of the schema of IDMEF is shown in figure 2.

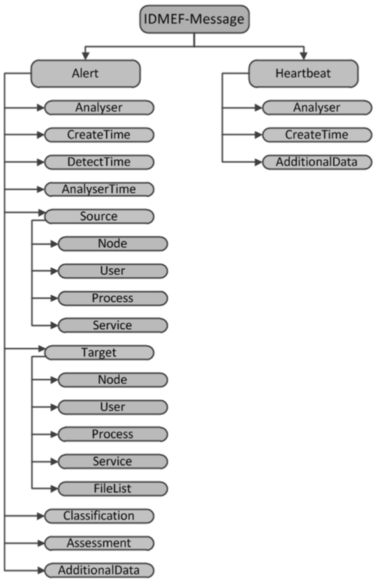

Figure 2. Overview about IDMEF schema [5]

For the serialization format, two protocols have been analyzed: *CBOR (RFC-7049)* [6] and *Protocol Buffers (protobuf)* [7]. CBOR stands for Concise Binary Object Representation and is a data format, which has been designed for an extremely small code size, fairly small message size, and extensibility without the need for version negotiation. CBOR is based on the wildly successful JSON data model: numbers, strings, arrays, maps (called objects in JSON), and a few values such as false, true, and null. One of the major practical benefits of JSON is that successful data interchange is possible without transmitting the schema information explicitly in the messages. The schema is previously known by all components and thus transmission speed can be increased significantly by only sending the raw structured information.

Some applications that would like to use JSON need to transport binary data, such as encryption keys, graphic data, or sensor values. In JSON, these data has to be encoded (usually in base64 format), adding complexity and bulk. Some applications also benefit from CBOR itself being encoded in binary. This saves bulk and allows faster processing. One of the major motivators for the development of CBOR was the Internet of Things, which will include very simple, inexpensive nodes where this counts.

To be able to grow with its applications and to incorporate future developments, a format specification needs to be extensible. CBOR defines tags as a mechanism to identify data that warrants additional information beyond the basic data model. Both future RFCs and third parties can define tags, so innovation is “permission-less” but can still be coordinated.

Protocol Buffers (protobuf) are Google's language-neutral, platform-neutral, extensible mechanism for serializing structured data, e.g. it is similar to XML but smaller, faster, and simpler. The structure of data has to be defined once, and then specifically generated source code is used to easily write and read your structured data to and from a variety of data streams and using a variety of languages.

Despite the apparent advantage of code generation availability of protobuf, CBOR has been chosen for the serialization format. This is due to the easier integration of CBOR into the other (Java based) components of the system. In addition, the compression level of binary data is better in CBOR.

V. INTEGRATION OF NAC SYSTEM

CLEARER wants to extend existing NAC systems with SIEM functionality. Therefore, an *interface* to the existing NAC systems has to be used for communication. Regarding the German NAC system by macmon secure the *macutil tool* can be used to establish processes from outside into the NAC system. This tool can be accessed by command line interface (CLI) or https. Because of a necessary additional agent on the NAC system for CLI, we chose https as protocol. Additionally, macutil allows establishing some functionality of the NAC system (e.g. lock/unlock of MAC addresses), which is also important for the SIEM architecture of CLEARER. Therefore, both systems have benefits regarding their interaction.

Figure 3 shows the concept of integration of the macutil interface to the system of CLEARER. For the communication with arbitrary NAC systems, the functionality is encapsulated within a NAC Actuator. This actuator handles the incoming requests and transforms them into a format, which can be used by the connected interfaces. Regarding the macutil interface, it is an ordinary https request with integrated authentication.

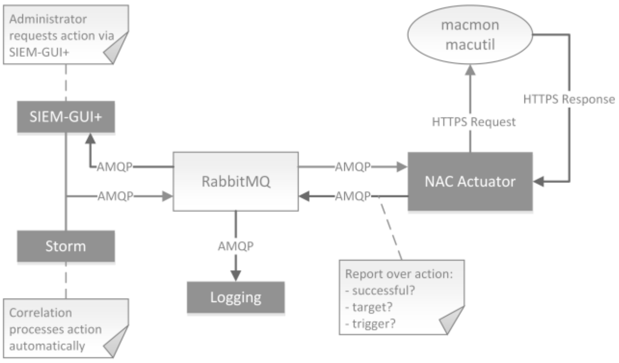

Figure 3. Concept of the integration of macmon macutil

Requests for lock or unlock of a mobile device can be generated by two actuators. On one hand the SIEM GUI+ is able to do this, if an administrator has to handle special events. On the other hand, enforcement can be established automatically if this is wanted. Then the request comes via Bolt from the Storm topology directly. In both cases, the request will be handled via RabbitMQ and the AMQP protocol via NAC actuator. This actuator executes this action via the NAC interface and processes the response. In every case, an entry has been written into the log system, including executed action, action result, target, and the triggered components. If another NAC system shall be used, only the implementation of the NAC actuator has to change. All other functionality remains the same.

Because of some limitations of the macutil interface and the requirement of independency of the NAC system, CLEARER analyzes *REST interfaces*. With this approach it will be possible to integrate arbitrary systems, without communication boundaries. But this is future work.

VI. Conclusions

This paper describes a SIEM-based architecture with direct connectivity to an existing NAC system. One main goal of the project CLEARER is to easily fulfill IT compliance standards regarding security for small and medium-sized enterprises (SME) easily. Another goal is to proof these compliance standards to different authorities.

Additionally, the main focus for all used components was on open source software to limit license costs and to use open interfaces. Regarding the amount of data and its access patterns, which the new architecture has to handle, new database approaches are used. In contrast to traditional database applications, high write scalability is needed with low requirements on read operations. In addition, auditing features for all changes on database content is required for compliance reasons.

Acknowledgment

The authors give thanks to the BMWi [3] for the financial support as well as to all other partners involved into the research project CLEARER for their great collaboration. The project consists of the industrial partners DECOIT GmbH, macmon secure gmbh, IT-Security@Work and the research partner University of Applied Sciences and Arts of Hanover.

References

[1] K.-O. Detken, T. Rix, C. Kleiner, B. Hellmann, L. Renners: *SIEM Approach for a Higher Level of IT Security in Enterprise Networks*. Proceedings of the 8th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), Volume 1, p. 322-327, 24.-26. September, University of Warsaw, ISBN 978-1-4673-8359-2, Warschau 2015

[2] CLEARER project website: *http://www.clearer-project.de*

[3] Federal Ministry of Economic Affairs and Energy: *http://www.bmwi.de/Navigation/EN/Home/home.html*

[4] H. Debar, D. Curry, B. Feinstein: *The Intrusion Detection Message Exchange Format (IDMEF).* RFC-4765, IETF 2007

[5] Francais: *Schéma du format IDMEF*. Wikipedia, license Creatice Commons Attribution-Share Alike 4.0 International, 2017

[6] CBOR: *RFC 7049 Concise Binary Object Representation*. http://cbor.io, IETF standard (stable reference), IETF 2013

[7] Protcol Buffers: https://developers.google.com/protocol-buffers/

[8] Apache Storm Framework: http://storm.apache.org

[9] Simple Event Correlator (SEC): https://simple-evcorr.github.io

[10] EsperTech: http://www.espertech.com/esper/

[11] Drools Fusion: http://drools.jboss.org/drools-fusion

[12] Apache Cassandra Database: http://cassandra.apache.org

[13] Apache Solr: http://lucene.apache.org/solr/

[14] Bro: https://www.bro.org

[15] RabbitMQ: https://www.rabbitmq.com

[16] K. M. Kavanagh, O. Rochford, T. Bussa: *Magic Quadrant for Security Information and Event Management.* Gartner, 10. August 2016 https://www.gartner.com/doc/3406817/magic-quadrant-security-information-event

[17] AlienVault-Webseite: https://www.alienvault.com/

[18] D. Mahrenholz, R. Schumann: *Open-Source-SIEM im Eigenbau*. P. Schartner, P. Lipp. „DACH Security 2016“, syssec 2016

[19] TCG Trusted Network Communications: *TNC IF-MAP Binding for SOAP*. TCG 2014

[20] N. Marz, J. Warren, *Big Data: Principles and best practices of scalable realtime data systems.* Manning Publications, ISBN 978-1-6172-9034-3, 2015.

[21] Cagri Balkesen, Nihal Dindar, Matthias Wetter, and Nesime Tatbul: RIP: run-based intra-query parallelism for scalable complex event processing. In Proceedings of the 7th ACM international conference on Distributed event-based systems (DEBS'13). ACM, New York, NY, USA 2013, 3-14.

[22] Lars Baumgärtner, Christian Strack, Bastian Hoßbach, Marc Seidemann, Bernhard Seeger, and Bernd Freisleben: Complex event processing for reactive security monitoring in virtualized computer systems. In Proceedings of the 9th ACM International Conference on Distributed Event-Based Systems (DEBS '15). ACM, New York, NY, USA 2015, 22-33.

[23] Stephen, J.J., Gmach, D. ; Block, R. ; Madan, A. ; AuYoung, A.: Distributed Real-Time Event Analysis. In: Autonomic Computing (ICAC), 2015 IEEE International Conference on, IEEE, 2015, 11-20.

[24] J. Yu, Y. V. Ramana Reddy, S. Selliah, S. Kankanahalli, S. Reddy, and V. Bharadwaj: TRINETR: an intrusion detection alert management systems. in 13th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2004. WET ICE 2004, 2004, pp. 235–240.

[25] A. Kim, M. H. Kang, J. Z. Luo, and A. Velasquez: A Framework for Event Prioritization in Cyber Network Defense, Jul. 2014. http://www.dtic.mil/get-tr-doc/pdf?AD=ADA608707

[26] T. Pietraszek: Alert Classification to Reduce False Positives in Intrusion Detection. PhD Thesis, Albert-Ludwigs-Universität Freiburg, 2006.