Incident Reaction Based on Intrusion Detections' Alert Analysis

Michael Heigl*†, Laurin Doerr*†, Amar Almaini†, Dalibor Fiala* and Martin Schramm†
*University of West Bohemia, Pilsen, Czech Republic
Email: {heigl, laurind, dalfia}@kiv.zcu.cz
†Deggendorf Institute of Technology, Deggendorf, Germany
Email: {michael.heigl, laurin.doerr, amar.almaini, martin.schramm}@th-deg.de

***Abstract*—The protection of internetworked systems by cryptographic techniques have crystallized as a fundamental aspect in establishing secure systems. Complementary, detection mechanisms for instance based on Intrusion Detection Systems has established itself as a fundamental part in holistic security eco-systems in the previous years. However, the interpretation of and reaction on detected incidents is still a challenging task. In this paper an incident handling environment with relevant components and exemplary functionality is proposed that involves the processes from the detection of incidents over their analysis to the execution of appropriate reactions. An evaluation of a selection of implemented interacting components using technology such as OpenFlow or Snort generally proofs the concept.**

I. Introduction

New and progressive technologies and their fast development in recent years, led to a broad spectrum of security holes considering the increasing possibilities that are also available to attackers [1]. Incidents, such as cyber attacks, become not only more numerous and diverse but also more damaging and disruptive [2]. Apart from the prevention, the detection by Intrusion Detection Systems (IDS), either signature-based or anomaly-based, have made their way into the security area and allow the detection of malicious actions when cryptographic mechanisms are outwitted or broken [3]. According to [1], a concept including cryptographic procedures as well as IDS components is mandatory for a comprehensive security solution. Systems that combine the goods from both worlds are presented in [4]. Adding alarm functionality to cryptograhpic mechanisms, when for instance sequence/packet counters drift apart, the decryption process fails due to tampering or message authentication code checks are violated, can be used to identify malicious behavior in networks. The detection process has only value if there is an appropriate response. New terms, such as Huge Data, state the trend of an ever increasing amount of data, leading to challenges in reliably identifying attacks due to many raised alarms. The alert analysis and planning as well as executing appropriate reactions can no longer be clamped by humans. Also under the circumstances that manual operation cannot guarantee the response time and accuracy, an efficient automatic decision-making support method is desperately needed [5]. New approaches for instance applying machine learning techniques help to handle the detection of massive input data. A semi-automated incident management is crucial to complement prevention and detection methods to a holistic security eco-system. The focus of this paper lies in the identification of required components and possible applications as well as interactions between them to cope with various detected incidents.

The rest of this paper is structured as follows: Section II gives details on incident handling and related work. The proposed concept with relevant components for a comprehensive incident handling environment and exemplary applications are described in section III. Section IV deals with the evaluation of selected parts of the concept. A short conclusion and a glance at the future work of the ongoing research work is finalizing the paper in section V.

II. Incident Handling

Incident handling is a process involving various disciplines in order to remedy violations to certain security policies. Many past incident handling and response planning methods, such as [6], targeted a higher level of abstraction in which mainly plans for human instructions in an incident response team are derived to cope with occurring incidents. An automated approach for responding to incidents has not been focused. However, managing the sheer number of raised alerts overwhelms many instructors such that the ideal considered reaction in realtime to zero-day exploits becomes an unreachable target. Self-learning systems that leverage, e.g. artificial intelligence, seems a promising approach for an automated incident response management. The work in [7] presents a system based on the usage of artificial intelligence in forensics, which collects and stores the data from a crime scene, and examines any correlation with previously solved crimes. The approach is directed to the forensic use case. This impedes the application of the adaption to a realtime response planning within an environment of various techniques of network attack detection. However, the involvement of previous successfully detected network attacks and expert knowledge from security specialists in conjunction with machine learning techniques seems quite promising to an incident handling architecture.

The survey in [2] proposes a Security Operations Center (SOC) including their mission and main functions that serves as an incident management system necessary to detect information security incidents, mitigating possible vulnerabilities exploited and restoring

ISBN 978-80-261-0722-4,

compromised Internet of Things infrastructures. The system might be able to automatically research, gather, ultimately identify incidents, evaluate, categorize and analyze them. However, the work only implemented a subset of parts of the comprehensive survey.

In [8] an intrusion prevention system for cloud environments based on Snort and OpenFlow is presented in which Snort is acting as an IPS. A Snort Agent filters network traffic on SDN-switches and forwards alerts to a server. On the server, the alarms are correlated and new filter rules are generated. The rules will be distributed to the SDN-Switches via an OpenFlow controller. The rule generation is roughly described and need more effort in research. There is no correlation of related incidents, which occur on different points of time, because no historical data is used. The system further is only working autonomously so that the plausibility of each new rule must be questioned.

There are a lot more incident management systems, but they mainly operate on their own without the interaction of other components. This limits their usability and reliability in more complex systems for incident and response management. For example, a combination of the three presented concepts would provide a strong improvement in attack detection. [7] could generate a correlation model for [8] which is able to perform reactions in realtime. A SOC could be used to bring detected incidents in a human readable form and expert knowledge can then bring updates into the correlation model. Another aspect of the systems is that they are not designed to work partially autonomous. Therefore a novel concept for incident management is proposed addressing not only those topics but also allowing the application of various detection mechanisms in order to exploit each method's strengths.

III. Proposed Concept

In the context of this paper the proposed concept for a partially autonomous incident management comprises several disciplines shown in figure 1. The *Detection* module provides information about detected incidents from various input data source to the *Analysis* module. Multiple mechanisms can be involved but each does not necessarily provide information about an actual attack or system failure. This component is continuously updated with new incident patterns identified by the *Post-Processing* module. The *Analysis* component is the main intelligence of the incident handling environment. Different advanced methods such as data mining, clustering, and correlation can achieve the detection of already known or novel attacks as well as system failures based on the provided information from the *Detection* module. The *Analysis* component contains historical data (database accessibility) with previous detected attacks and their respective response measures. Similar to the previous component, input from the *Post-Processing* component continuously updates the database. If the *Analysis* module identifies already known patterns, the *Response* component is instructed to automatically execute a reaction. An expert is informed about performed responses from this module. The *Post-Processing* module receives the analyzed data (e.g. patterns) from the *Analysis* module and refines it for the experts in form of visualization or logging. It receives the instructions or confirmations about proposed reactions from the *Expert Knowledge*, updates the database and if required performs adjustments of detection mechanisms. Human experts or security analysts operate within the *Expert Knowledge* module for instance in a SOC to cross-evaluate post-processing measures or monitor automated reactions. Orchestrating each module is a challenging task. Thus, a flexible *Communication Framework* is needed enabling the frictionless exchange of information.

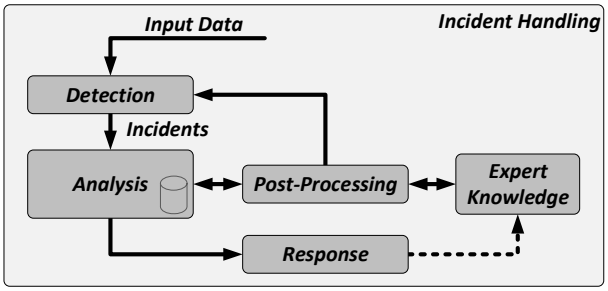

Fig. 1. Incident handling environment

A. Detection

The detection of incidents can be performed by either using an IDS or exploiting the nature of cryptographic mechanisms as described in the following. The focus of this paper lies in the detection of incidents based on network traffic but the concept also allows input from different sources such as the firmware in use or the output of vulnerability assessments.

Intrusion Detection: IDSs can be used to detect malicious activities or policy violations by monitoring the network traffics or system activities and can be categorized in three common architectures: Host-based IDS (HIDS), Application-based IDS (AIDS) and Network-based IDS (NIDS). Input data for incident handling from HIDS could be detected anomalies for instance in parsed log-files. In the past years, the main attention focused on the application of NIDS [9] which can be placed either centralized, decentralized or distributed within networks on either network switching entities (router, gateway, etc.) or dedicated hosts. There are two mainstream methods for IDS to detect intrusions: misuse-based and anomaly-based. The former, also called signature- or knowledge-based, is founded on a set of rules or patterns describing network attacks which are either pre-configured by the system or manually by an administrator. The other method collects data containing examples of normal behavior and builds a model of familiarity, therefore, any action that deviates from the model is considered suspicious and is classified as an anomaly. The main advantage compared to misuse-based systems is that they are able to detect zero-day intrusions [10]. However, anomaly-based IDS suffer from false-negative and false-positive alarms. To overcome the drawbacks and exploit the advantages of both methods, hybrid systems are proposed such the one presented in [11].

Cryptographic Mechanisms: Another possibility of incident detection is to exploit violations of the nature of cryptographic mechanisms, such as MACsec, IPsec or TLS, for alert generation that define a security infrastructure to provide data confidentiality, data integrity and data origin authentication. MACsec, as an example, relies on GCM-AES to ensure the confidentiality and integrity of all the network traffic mitigating attacks on layer-2 protocols [12]. In a MACsec-protected network further unauthorized LAN connections can be identified and excluded from the network communication over IEEE 802.1X port based authentication. Thus, mismatching packet numbers, failing decryption or the detection of unauthorized LAN connections could lead to an alarm generation. For Message Authentication Codes, alarm generation could be applied for mismatching hash digests. A combination of cryptographic prevention using continuous authentication for the detection is proposed in [13].

The concept in this work is designed for modularity to utilize various detection mechanisms in a hybrid approach exploiting each strengths to avail the merits and alleviate their demerits. However, the utilization strongly depends on the environment, application field and task such that it might not be feasible to use multiple mechanisms. For evaluation, the focus for a first proof of concept relies on only applying one signature-based IDS providing detected incidents to the *Analysis* component.

B. Analysis

Many work, for instance [5], make assumptions for response planning that each alert raised by a detection engine is treated as one attack. The aggregation and fusion of alarms should be taken into account by the detection system but they assume 100% confidence of the alerts. Sanity checks of the detection engine ensuring a better certainty is often out of scope. However, a detection system might raise false alerts.

Especially in environments in which multiple detection mechanisms operate, an intelligent alert analysis is vital to handle the massive amount of alarms from various sources and to intelligently react to incidents. In [14] for instance a comparison of supervised, semi-supervised and unsupervised learning methods for anomaly-based NIDS have been examined each having it's particular strengths but their detection capability differ significantly. According to [15], IDSs are not enough to detect complex attacks over a network. For detecting attacks with high accuracy, the input of each mechanism might be important but operating for instance in safety-critical environments, cross-evaluation or plausibility checks of various inputs is essential before performing a reaction. A standardized format, such as the Intrusion Detection Message Exchange Format (IDMEF) [16], is needed to bridge the outputs of various detection mechanisms and to perform an alert analysis on which basis appropriate reactions can be initiated. Fitting the alert output of cryptographic schemes or anomaly-based algorithms into IDMEF provides interoperability with already IDMEF-capable IDSs such as Snort/Barnyard2, OSSEC, Surricata or Prelude.

Generally three different analysis methods exist: similarity-based, sequential-based and case-based. For similarity-based methods, the correlation is based on feature similarity, such as IP addresses and ports associated with their occurrence. These methods are relatively easy to implement. The correlation can be improved as more specific features are detected. But no complex correlations can be identified. Sequential-based correlation uses relationships between multiple causal events. Therefore, pre- and postconditions have to be considered. Preconditions can be a set of requirements needed to trigger an alarm. Postconditions are the consequences of the execution of an event. Case-based correlation is defined for resolving new incidents based on similar incidents. The quality depends on the algorithms responsible for recognizing specific patterns of behavior. Often a combination of the analysis methods is used.

The framework described in [17] uses offline and online correlation. The former is used to generate a correlation model from historical alarms. It is regularly updated with new data from online correlation which generates meta-alerts based on real-time alerts. The meta-alerts are compared, then prioritized and clustered. Next, pattern mining extracts features for each cluster and generates an attack pattern graph. Finally, a Graph Frequency Pattern Mining Algorithm extracts common patterns from each cluster. The framework is optimized for a graphic information exchange to administrators, which makes it difficult to use in a partially autonomous system.

Different aspects of the analysis have to be combined for semi-automatic systems. Thus, historical data can be used to quickly evaluate known attack patterns and instruct the *Response* module. The correlation model must be constantly customized to recognize new patterns and should use a combination of the three analysis methods. For this, an interaction with the *Post-Processing* unit is important. By using IDMEF, different detection systems can be used simultaneously as information source.

C. Response

Response or reaction implies the set of actions the system executes after incidents have been detected. Such a reaction could include reconfiguring the network through Software-Defined Networking (SDN) techniques, generating new configurations for firewalls, creating new misuse-based IDS rules or adapting the parameters for incident detection mechanisms.

An example for the latter is to adjust the sampling rate for IDSs. Such systems are applied either in environments characterized by having a large amount of network traffic or to save system resources (memory, CPU consumption). An adaptive sampling mechanism is presented in [18]. As a reaction to identified incidents by detection mechanisms, the sampling rate could be increased to analyze a greater amount of

traffic in-depth if the amount of data for analysis is unsatisfactory to make a precise decision.

A further automatic response later evaluated in the paper is to automatically generate rules for an IDS. Previous work in this field faced issues regarding automation, robustness and elaboration in real network environments. Addressing those challenges, as an example, the work in [19] proposes an automatic signature generation method called SigBox for fine-grained traffic generation using modified sequence pattern algorithm targeting content, packet, and flow signatures for Snort.

Exploiting SDN technology to reconfigure the networking environment is a further reaction possibility. Controlling network flows dynamically enables to separate malicious (or suspicious) network flows from benign ones dynamically. For example, supposed that a NIDS detects some suspected flows, the flows can be rerouted for in-depth investigation, e.g., in a honeypot [20]. Further firewall functionality can be implemented using SDN. When a switch receives a new packet and there is no rule matching this packet in the flow table, it reports it to the SDN-controller which forwards, the packet to the firewall application. The firewall checks whether the incoming packet violates security policies or not and enforces a new flow rule accordingly. This rule is delivered to the switch by the controller and all future packets from the flow of the first packet would be handled directly in the switch without the need to interact with the controller again [21]. Another possibility applying SDN to respond to a specific incident is through network separation. In traditional networks, the common way to separate a network is employing Virtual LAN (VLAN) technique, which adds specific IDs in a packet header (12-bits VLAN ID field). However, VLAN technology incurs scalability limits in large-scale networks, since it can only assign 4,096 different virtual networks. SDN-based separation solutions provide the capability of different level abstractions with desired security properties, which not only separates the network segments efficiently at scale, but also veils the physical view of networks to users [22].

Above mentioned response measures are examples that can be executed autonomously within the concept if instructed by the *Analysis* component if a historic similar incident already occurred in the past. If an automatic reaction is not possible, further examination is necessary in the *Post-Processing* module which also allows expert interaction.

D. Post-Processing

Post-Processing features can have a broad spectrum ranging from visualizing analyzed incidents to working them up, logging each interaction, as well as performing cross-evaluation of already executed reactions. Visual analysis or presentation can help in intrusion detection and to prepare a flood of complex data for the end user accordingly. Efficient incident information visualization is an important element, which is not only vital for unknown attack detection. Appropriate presentation methods for different groups of implicated persons are necessary to display incidents helping to recognize and respond to attacks quickly. It also allows an interaction, such as the confirmation of reaction execution through expert knowledge by a human especially necessary in highly critical environments. Apart from this, a cross-evaluation could help to mitigate false-negatives and reinforce IDS functionality through the involvement of various data sources available from the system or an administrator. An administrator or developer needs a deeper and detailed insight, for instance to clarify incidents from a forensic point of view. In such case, the Honeypot example mentioned above in the context of SDN can be an appropriate method to deliver in-depth investigated statements about the incidents. A novel real-time visualization, also considering a cross-evaluation by humans, is proposed in [23] to effectively display many security events collected by multiple IDS.

The lore gained from the *Post-Processing* component including *Expert Knowledge* continuously improves the quality of the incident handling environment by adjusting the *Detection* or updating the *Analysis* module which has then an immediate influence on the *Response* component.

E. Communication Framework

A crucial part in order to orchestrate the incident handling components and enabling an exchange of information is the communication. This is especially the case if the components are not implemented on the same physical system. Thus, a framework that satisfies the following requirements is needed.

- Guaranteeing the authenticity of each component
- Logging of each interaction for traceability and forensics
- Guaranteeing the authenticity, confidentiality, integrity and privacy of exchanged information
- Enforcing policy-based access restrictions

A proposed system for the concept that can be utilized as a communication backbone satisfying the requirements above, is the Distributed Smart Space Orchestration System (DS2OS) [24]. DS2OS divides the framework in two layers: Middleware and Services. The middleware provides a uniform, scenario-independent interface for participating components. It is utilized by a distributed system whose components may run on distributed nodes that have a sufficient amount of free resources and allows storing and brokering state context between multiple computation devices (e.g computers, embedded controllers).

Dedicated functions of the components described above within the incident handling environment can be implemented as dedicated services. The explicit separation from the middleware as a used communication platform makes it possible to create services dynamically for the different use cases. In addition, services can be dynamically started, stopped or migrated between the distributed middleware components, depending on the available resources, for example. Possible services for the incident handling eco-system are

data aggregation, feature extraction, intrusion detection, alert analysis, logging, or reaction execution.

IV. Evaluation

The proposed automated incident handling environment ranging from detection, analysis, reaction to post-processing measures is evaluated within a virtualized Proxmox environment as depicted in figure 2. Parts of the incident handling environment of figure 1 are implemented in the *Incident Management* component using open source tools as well as the in Python language written Incident Processing Application (IPA). Communication takes place in form of local services via DS2OS. In a simple scenario, a proof of concept is established.

Host 1 is communicating with *Host 2* over the network route including the Open vSwitches *OVS 0* and *OVS 1*. The route from *OVS 0* to *Host 2* over *OVS 2* is not configured by default. An *Attacker* tries to send malicious messages to *Host 1* and *Host 2*. Intrusion detection is performed on the *Incident Management* entity using the open source NIDS Snort based on the mirrored traffic from *OVS 0*. Further the entity contains the Python rules based correlation engine Prelude-Correlator from the Prelude-SIEM used for a simple alert analysis. Thus, it receives the IDMEF-based alerts from the IDS and raises an alert in the case of a successful correlation based on a provided ruleset.

As an automated response, based on a correlation alert stored in a database representing a reaction based on a historical incident, the *Incident Management* entity creates a new Snort rule and reloads it (IV-A). A second evaluation includes the interaction between post-processing and admin confirmation (IV-B) to reconfigure the Open vSwitches *OVS 1* and *OVS 2* based on the OpenFlow controller OpenMUL.

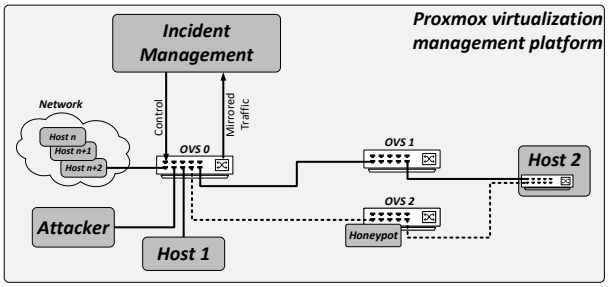

Fig. 2. Evaluation environment for incident detection and reaction

A. IDS Reconfiguration

Snort analyzes the mirrored traffic from the network using Winpcap, analyzes for matches to a defined rule set and generates alerts in the unified2 binary output format. Allowing Snort to write these output files will not cause Snort missing network packets. The open source interpreter Barnyard2 is then used to parse the binary data into IDMEF in a separate task. Using Prelude, the generated IDMEF alerts are provided to a central management unit called the Prelude Manager. In the scenario two rules are existing. One, exemplary shown below, that creates an alarm of any IP address, except *Host 1*'s, is sending any message to *Host 2* and one vice versa.

alert any !$IP_HOST_1 any -> $IP_HOST_2 any (msg:"Unknown host communicating to Host 2"; sid:10000000; rev:1;)

IPA contains functionality from the Prelude-Correlator which is used to trigger a simple correlation alert if both of the above rules match for the IP address of the *Attacker* entity. Further the application creates and activates a new Snort rule, as a response measure, that explicitly triggers for any communication sent by the attacker. Based on the results of an already executed reaction in the past, stored in a SQL database showing the same behavior, a new rule mask is provided for response. Thus, first a new Snort ID must be determined, the new rule (shown below) and the new sid-entry for the sid-msg.map file created, loaded to the Snort system and added to the appropriate files.

alert any $IP_ATTACKER any -> any any (msg:"Unknown host (attacker) communicating"; sid:10000002; rev:1;)

In order to activate the new rule without missing any packets, Snort needs be reloaded using a SIGHUP signal. Snort utilized in inline mode connecting the attacker to the network would even allow to block all further activities if Snorts rule header is configured to drop the packets when the rule matches.

B. Network Reconfiguration

In a second scenario, IPA is used to reconfigure the network based on SDN technology. Thus again the first Snort rule from above is used to detect malicious hosts communicating to *Host 2*. If the *Attacker* is sending malicious packets to *Host 2* via the route consisting of *OVS 0* and *OVS 1* an alarm is raised. Since no historical event is known in the database, the post-processing is presenting an admin the event in the graphical front-end Prewikka of the Prelude-SIEM and waiting for a possible reaction by the admin. Then, the *Incident Management* entity is reconfiguring *OVS 1* and *OVS 2* such that network traffic is further only forwarded by *OVS 2* for in-depth investigation within a honeypot. Thus, IPA is determining the dpid of the two Open vSwitches created by the OpenFlow controller in order to create new OpenFlow rules. It further temporarily saves the existing rules and determines whether a new OpenFlow rule must be created for the attacker's IP address. If the new rules are added, the controller distributes them triggered by a command line interface command of IPA. While in the reconfiguration process, malicious traffic might still flow for a certain duration. In a measurement, the duration from the detection by Snort until the final reconfiguration of the Open vSwitches could be determined to an average of 733.76 ms for 74 measurements. Further work aims to reduce this time.

V. Conclusion and Future Work

The necessity for protecting computer networks by the detection of incidents and partially autonomous reaction demands a comprehensive incident handling

environment. The proposed concept in this paper identifies several required components ranging from various detection mechanisms to reaction possibilities and provides exemplary functionality. The evaluation provides a first proof of concept that includes an automatic reaction to detected incidents based on historical information and in a second scenario the involvement of expert confirmation including a reaction over a post-processing measure.

The detection of incidents in the evaluation was based on the single IDS Snort. Exploiting the advantages of the proposed concept to include various detection mechanisms for instance by anomaly-based machine learning methods have already been implemented and evaluated during the writing of this paper. Further research needs to be done to extend them with IDMEF and to integrate them into the incident handling environment. In addition, the concept is not limited to a central incident detection and reaction component. The next step of evaluation is also to include distributed components enabled by the strengths of the DS2OS framework to address distributed anomaly detection approaches.

Instead of relying on simple analysis techniques such as with Prelude-Correlator more sophisticated approaches as proposed in [25] are aimed to be integrated in the near future. Multiple techniques applied alongside in a hybrid fashion could improve the incident analysis capabilities. Targeting the requirement for traceability and forensics of actions within the incident handling a promising approach could be based on the usage of Distributed Ledger Technology for instance implemented with Blockchains. Further research aims to exploit such technology to either log each interaction tamper-resistant implemented in the DS2OS communication framework or to exploit their consensus algorithm for conformity of various detection mechanism output.

ACKNOWLEDGMENT

This research work has been supported by the research projects no. 16KIS0539 and 03FH016IA5 of the German Federal Ministry of Education and Research and by the Ministry of Education, Youth and Sports of the Czech Republic under grant No. LO1506.

REFERENCES

[1] Vittor T. R., Sukumara T., Sudarsan S. D., and Starck J., *Cyber security - security strategy for distribution management system and security architecture considerations*, 70th Annual Conference for Protective Relay Engineers (CPRE), 2017

[2] Miloslavskaya N., *Security operations centers for information security incident management*, IEEE 4th International Conference on Future Internet of Things and Cloud, 2016

[3] Arrington B., Barnett L., Rufus R., and Esterline A., *Behavioral modeling intrusion detection system (BMIDS) using internet of things (IoT) behavior-based anomaly detection via immunity-inspired algorithms*, 25th International Conference on Computer Communication and Networks (ICCCN), 2016

[4] Studnia I., Alata E., Nicomette V., Kaâniche M., and Laarouchi Y., *A language-based intrusion detection approach for automotive embedded networks*, 21st IEEE Pacific Rim International Symposium on Dependable Computing, 2014

[5] Ossenbuhl S., Steinberger J., and Baier H., *Towards Automated Incident Handling: How to Select an Appropriate Response against a Network-Based Attack?*, 9th International Conference on IT Security Incident Management and IT Forensics, 2015

[6] Takano M., *ICS Cybersecurity Incident Response and the Troubleshooting Process*, SICE Annual Conference 2014, 2014

[7] Hasan R., Raghav A., Mahmood S., and Hasan M. A., *Artificial Intelligence Based Model for Incident Response*, 2011 International Conference on Information Management, Innovation Management and Industrial Engineering, 2011

[8] Xing T., Huang D., Xu L., Chung C. J., and Khatkar P., *SnortFlow: A OpenFlow-based intrusion prevention system in cloud environment*, Second GENI Research and Educational Experiment Workshop (GREE), 2013

[9] Fallstrand D., and Lindstroem V., *Applicability analysis of intrusion detection and prevention in automotive systems*, Master's Thesis in Computer Systems and Networks on the Chalmers University of Technology Goteborg, [Online], Available: http://publications.lib.chalmers.se/records/fulltext/219075/219075.pdf, 2015

[10] Ashfaq R. A. R., Wang X.Z., Huang J.Z., Abbas H., and He Y. L., *Fuzziness based semi-supervised learning approach for intrusion detection system*, Journal Information Sciences, 2017

[11] Taylor A., *Anomaly-based detection of malicious activity in in-vehicle networks*, PhD Thesis at the University of Ottawa, [Online], Available: https://ruor.uottawa.ca/bitstream/10393/36120/3/Taylor_Adrian_2017_thesis.pdf, 2017

[12] IEEE Computer Society, *IEEE Standard for Local and Metropolitan Area Networks: Media Access Control (MAC) Security*, IEEE Std 802.1AE-2006, 2006

[13] Boudguiga A., Witold K., Boulanger A., and Chiron P., *A Simple Intrusion Detection Method for Controller Area Network*, IEEE ICC 2016 Communication and Information Systems Security Symposium, 2016

[14] Arunraj N. S., Hable R., Fernandes M., Leidl K., and Heigl M., *Comparison of Supervised, Semi-supervised and Unsupervised Learning Methods in Network Intrusion Detection System (NIDS) Application*, Anwendungen und Konzepte der Wirtschaftsinformatik, 2017

[15] Suarez-Tangil G., Palomar E., de Fuentes J. M., Blasco J., and Ribagorda A., *Automatic Rule Generation Based on Genetic Programming for Event Correlation*, Computational Intelligence in Security for Information, 2009

[16] Debar H., et al., *The Intrusion Detection Message Exchange Format (IDMEF)*, RFC 4765, 2007

[17] Shittu R., Healing A., Ghanea-Hercock R., Bloomfield R., and Rajarajan M., *Intrusion alert prioritisation and attack detection using post-correlation analysis*, Computers & Security, 2015

[18] Taejin H., Sunghwan K., Namwon A., Jargalsaikhan N., Chiwook J., JongWon K., and Hyuk L., *Suspicious traffic sampling for intrusion detection in software-defined networks*, Journal Computer Networks, 2016

[19] Shim K. S., Yoon S. H., Lee S. K., and Kim M. S., *SigBox: Automatic Signature Generation Method for Fine-grained Traffic Identification*, Jounral of Information Science and Engineering 33, 2017

[20] Shin S., Xu L., Hong S., and Gu G., *Enhancing Network Security through Software Defined Networking (SDN),* 25th International Conference on Computer Communication and Networks (ICCCN), 2016

[21] Giotis K., Androulidakis G., and Maglaris V., *Leveraging SDN for efficient anomaly detection and mitigation on legacy networks*, Third European Workshop on Software-Defined Networks, 2014.

[22] Lehocine M. B., and Batouche M., *Flexibility of managing VLAN filtering and segmentation in SDN networks*, International Symposium on Networks, Computers and Comm., 2017.

[23] Song B., Choi S. S., Choi J., and Song J., *Visualization of Intrusion Detection Alarms Collected from Multiple Networks*, Lecture Notes in Computer Science (Including Subseries Lecture Notes in AI and in Bioinformatics), 2017

[24] Pahl M. O., Carle G., and Klinker G., *Distributed Smart Space Orchestration*, Network Operations and Management Symposium 2016 - Dissertation Digest, 2016

[25] Kawakani C. T., Junior S. B., Miani R. S., Cukier M., and Zarpelo B. B., *Intrusion alert correlation to support security management*, XII Brazilian Symposium on Information Systems in the Cloud Computing Era, 2016