FACULTY
OF APPLIED SCIENCES
UNIVERSITY
OF WEST BOHEMIA

University of West Bohemia
Department of Computer Science and Engineering
Univerzitní 8
30614 Plzeň
Czech Republic

Network Communication Protection Based on Anomaly Detection's Incident Handling

State of the Art & Concepts of Doctoral Thesis

Michael Heigl, M.Sc.

Technical Report No. DCSE/TR-2019-03
July 2019
Distribution: confidential

Technical Report No. DCSE/TR-2019-03
July 2019

Network Communication Protection Based on Anomaly Detection's Incident Handling

State of the Art & Concepts of Doctoral Thesis

Michael Heigl, M.Sc.

Abstract

The ubiquitous internetworking of embedded devices in all areas of life is thrived by various trends e.g. the Internet of Things and the advent of disruptive technologies such as Artificial Intelligence or Quantum Computing. However, adversaries also benefit from these developments. Not only the too fast development of the rapidly advancing and pervading technologies lead to a broad spectrum of security problems but also to an increase of capabilities for adversaries in terms of tools, methods and technologies which place great demands on innovative security solutions. The identification of network-based attacks or unknown behavior through Intrusion Detection Systems has established itself as a conducive and mandatory mechanism apart from the protection by cryptographic schemes. However, these systems show various limitations when it comes to reliably identifying and reacting to cyber attacks. This report provides an overview of the state of the art of incident handling in the domain of network communication security including incident detection, analysis and response. Open research questions and challenges are pointed out and fundamental concepts of the future Ph.D. thesis towards an improved incident handling leveraging anomaly-based detection algorithms are sketched out.

This work has been supported by the research project MLPaSSAD under grant No. 13FH645IB6 of the German Federal Ministry of Education and Research.
Copies of this report are available on
`http://www.kiv.zcu.cz/en/research/publications/`
or by surface mail on request sent to the following address:

University of West Bohemia
Department of Computer Science and Engineering
Univerzitní 8
30614 Plzeň
Czech Republic

Acknowledgements

My deepest appreciation goes to my supervisor doc. Ing. Dalibor Fiala, Ph.D. at the University of West Bohemia for his great guidance, his valuable feedback at all times, his expertise in the field of data mining and machine learning and his help to overcome administrative obstacles especially with the Czech study bureaucracy. Furthermore, a special and heartfelt thanks goes to my colleagues of the institute ProtectIT at the Deggendorf Institute of Technology, especially Prof. Dr. Martin Schramm, for enabling me pursuing a Ph.D. degree in cooperation with the University of West Bohemia. His comments and suggestions were always of inestimable value and made enormous contribution to my work in the field of cyber security in particular by his expertise in the field of embedded cryptography. Most parts of this report were elaborated in the context of the research projects Decentralized Anomaly Detection (DecADe) and New Multi-Layer Platforms for Security and Safety-Relevant Automated Driving Functions (MLPaSSAD). Both projects have been supported by the German Federal Ministry of Education and Research (BMBF) under grant code 16KIS0539 (DecADe) and 13FH645IB6 (MLPaSSAD). Along with the projects I want to express my gratitude to the vice president of applied research and technology transfer, Prof. Dr.-Ing. Andreas Grzemba, of the Deggendorf Institute of Technology for granting me one of the coveted doctoral researcher positions. I also owe a very important debt to my family and would also like to express my deepest gratitude for their moral support and warm encouragement during my academic career as well as for their continuous and unparalleled love.

Michael Heigl, Straubing, Germany, July 2019

Contents

List of Tables

List of Equations

List of Figures

1 Introduction

This report is an attempt at capturing the state of the art in the domain of network communication protection by leveraging techniques of incident detection, analysis and response. In this section the motivation of the research work is provided driven by upcoming next generation threats and the great demand towards a comprehensive and automated security solution even when cryptography has been overcome. Research challenges towards intrusion detection are stated, requiring the application of new methods from other disciplines. The section further contains the overall research objective in anticipation of Section 5 and is finalized by an outline of the report.

1.1 Motivation

The constantly growing number of computer components, the spatial extent of networks, their interaction with their environment (e.g. automotive: "Always and Everywhere On") boosted by trends such as Internet of Things/Everything (IoT/E), Connected Cars, Smart Cities, Industrie 4.0, 5G or Software-Defined Everything and the use of new technologies, e.g. Artificial Intelligence (AI), not only lead to the improvement of processes or customer comfort, but also increase system complexity and create new hazard potentials and risks with regard to the information and operational security of these systems. This also offers adversaries a broad spectrum of attack vectors and capabilities such as the usage of cloud or distributed computing, quantum computation for breaking contemporary public-key cryptography [1] or smart AI-powered malware that enables highly sophisticated and stealth attacks. The rise of next generation threats demands "adopting new methods of automated prevention methods" stated by John Samuel, senior vice president and global chief information officer at CGS [2].

Constant monitoring of components, early detection and handling of attacks as well as comprehensive continuous assessments of the security level of the overall system are therefore unavoidable for securing future-oriented IT-systems. Especially, security mechanisms are needed that can dynamically and flexibly detect and respond to network attacks. Methods, for instance, that use already known attack patterns for the detection of incidents are no longer sufficient to secure network infrastructures. Further problems are stated with respect to two possible fields of applications. Problems in the automotive sector primarily comprise the integration of IT security solutions to detect anomalies in the vehicle and the interaction to forward this information to the backend in order to gain added value across fleets, e.g. by analyzing alarm data from several vehicles. In the industrial sector, many companies already use standard IT security solutions such as firewalls, Intrusion Detection/Prevention/Response Systems (IDSs / IPSs / IRSs) or anti-virus software. However, these produce an unmanageable

number of events that cannot be handled by the employees of a Secure Operations Center (SOC) with available capacities/resources. According to Michael Roytman, chief data scientist at Kenna Security, "automation is the name of the game in security" and "machine learning will help filter out the noise" [2]. Automation not only benefits the analysis of alerts but also to select and execute appropriate responses without human intervention. Especially variants of IDSs using anomaly-based machine learning methods have crystallized themselves as a fundamental part in a holistic security solution by identifying new attacks even when cryptography has been broken.

1.2 Research Challenges and Objective

The ever-increasing and more advanced attack capabilities and strategies pose an enormous challenge to classical IDSs and demand more sophisticated and comprehensive solutions in the near future. Referring to [3], current IDSs only cover 25% of their threat taxonomy. The actual percentage might even be lower considering the multitude of degrees of freedom mentioned in this report. A selection of major challenges and open issues in the field of network communication protection, in particular the detection, analysis and response to cyber attacks (using among others anomaly-based machine learning algorithms) are listed in the following (cf. [4, 5, 6, 7]):

- insufficient performance of applied detection systems, especially with regard to anomaly-based IDSs, by
  - missing ability in handling a massive amount of throughput e.g. due to high computational complexity
  - lack of internetworking in a decentralized/distributed fashion; limited scalability, interoperability
  - producing miserable predictive values with respect to the binary classification (e.g. too many false positives and false negatives)
  - no dynamic and adaptive learning working on minimum knowledge for novel attack classification in real-time
  - inefficient pre-processing and poor feature selection
  - limited protection against attacks e.g. IoT-based DDoS
- novel (zero-day exploits) and high sophisticated (distributed, stealth, more targeted, long-term, multi-step) attacks, e.g. Advanced Persistent Threats (APT) can not reliably be detected
- increasing amount of alerts (alert flooding) produced by various detection methods with multiple formats can't reliably be analyzed or today's applied analysis methods are very limited to e.g. simple correlation methods
- safe (with respect to functional safety) implementation of static and dynamic response measures such that always a defined and secure system state remains in order to avoid hazards to life and limb (e.g. attack

when driving a vehicle) or high financial losses (industrial plant is paralyzed) even when using cyber defence mechanisms

- lack of incorporation of anomaly-based IDS outputs into new attack identification, classical systems assume nearly 100% confidence of alerts to map an attack
- proactive (predictive) and immediate (real-time) response to cyber attacks instead of reactive and "a posteriori" response
- lack of automation since often still a high degree of human interaction is necessary, e.g. system administrators analyze (i.e. verify, correlate) alerts, update signatures, etc.
- contextual information often not taken into account e.g. type of systems, applications, users, networks, etc.
- lack of finding the actual root cause of the incident
- inherent resource limitations in terms of memory size, processing speed as well as energy consumption in the embedded domain restrict the usage of complex machine learning based methods

The novel principle "cyber resilience" goes far beyond pure cyber security and takes a comprehensive approach to protect IT infrastructures from cyber attacks by securing and restarting operations after attacks have occurred. IDSs are not sufficient anymore to fulfill the needs and the stated challenges demand for a more holistic solution. The measures and concepts of cyber security, computer forensics, information security, disaster recovery and business continuity management are components of this approach. Cyber resilience bridges the gap for a desired (semi-)automated incident handling with the topics around detection, prevention, prediction and response apart from the protection via cryptography. The term incident, hereinafter, includes intrusions or failures and is an event that negatively affects the protection goals of a system. An event is an observed or detected change to the normal behavior of a system which typically leads to the generation of an alert. It notifies about a particular event or a series of them sent to responsible parties. For a detailed definition of an information security event and an information security incident refer to ISO/IEC 27000 [8]. A further goal is the reaction to incidents based on available knowledge within networks and related incidents (expert knowledge, risk analysis, etc.). This also comprises the output of incident detection systems based on a common format. Especially with regard to anomaly-based detection methods, the output of such do not provide much context to single anomalies, which makes the identification of attacks difficult. Information from multiple detection mechanisms in conjunction with historical data and expert knowledge creates a framework from the detection, to the prediction towards appropriate proactive/reactive reactions. In the context of anomaly-based detection mechanisms such reactions could include the proper adjustment of features or parameters in

order to reduce false positives or to prevent even highly sophisticated multi-staged attacks. Thus, the overall research objective discussed in Section 5 is defined as: **How to increase the network communication security of computer networks with a focus on future-oriented automotive infrastructure by establishing an anomaly-based incident detection to mitigate/counteract novel malicious activity propagation?**

1.3 Report Outline

The rest of this report is structured in accordance with the most important stages of a cyber defense life cycle [9, 10, 11, 12], e.g. shown in Figure 1, namely incident detection, incident analysis and incident response. Section 2 provides an overview of incident detection mechanisms including classical IDSs with their various characteristics, methods and specificities in order to detect security-relevant incidents. A focus in this section lies on anomaly-based machine learning methods which are mandatory in order to detect novel malicious behavior. Two examples for network-based anomaly detection methods, Isolation Forest and Loda, are presented which satisfy requirements stated. Furthermore, the combination of learning methods is discussed which also intersects with the proceeding Section 3 which mainly deals with the analysis of detected incidents. Apart from a taxonomy of alert analysis fields, the section deals with alert pre-processing including the discussion of appropriate alert exchange formats, processing by e.g. presenting alert correlation techniques and post-processing covering among others the prediction of malicious activity. Section 4 discusses incident response systems including their taxonomy, appropriate exchange formats and possible response measures. For those, again two examples are provided focusing on the adaptability of IDSs and the reconfiguration of the network infrastructure applying Software-Defined Networking (SDN). The fundamental concepts of the future Ph.D. thesis are sketched out in Section 5 pointing out some requirements. A short conclusion and a glance at the future work with respect to the preliminary ideas to ongoing developments that imply the aims of the doctoral thesis are finalizing the report in Section 6.

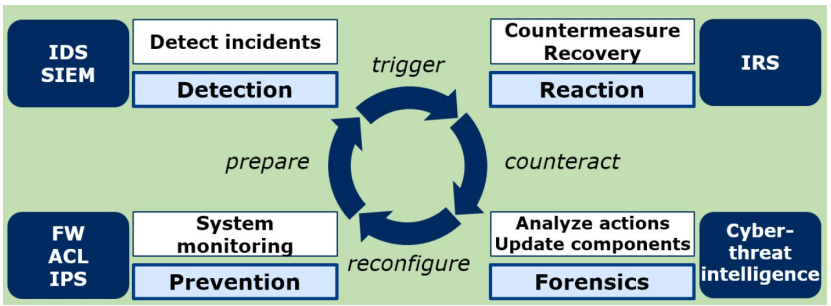

Figure 1: Cyber defense life cycle [4]

2 Incident Detection

Cryptographic mechanisms alone cannot provide a holistic security solution for network communication protection in the future. For instance, if an adversary compromises a sensor node, it is easy to inject malicious data. A possibility to detect attacks is to apply an IDS which is a component that monitors the events occurring in a computer system and/or network such that malicious actions attempting to compromise security primitives can be detected. At the beginning of the 1990s, Todd Herlein was invited to an IEEE conference in Oakland where he first introduced an IDS that was based on network traffic [13]. Since then, some progress has been made with regard to IDSs. Especially in the embedded sector, there has been an enormous increase in research activities in recent years. This is due to (1) the increasing networking of systems, (2) the development of more powerful and complex embedded systems and (3) an increasing demand due to an almost uncountable number of new and high sophisticated attacks. The permeation of connecting everything in various fields, inspired by the so-called "IoT-ification", can not only be seen in the industrial automation and avionics sectors but also in the automotive domain which are common application fields for embedded IDSs.

2.1 Taxonomy of IDSs

An overview and summary of a taxonomy of IDSs is depicted in Figure 2 and can be found in [3]. In the following subsections, details on the taxonomy are provided including architectures, detection methods and other typical characteristics for IDSs.

2.1.1 Architectures

The research and solutions for attack detection mechanisms are wide-ranging and manifold. However, from a higher perspective there are two main architectures of IDSs: *Host-based IDS (HIDS)* and *Network-based IDS (NIDS)*. HIDSs are applied on a single host to monitor all events for malicious actions for instance event logs, system logs, file access, running processes. In [14] an example of a distributed denial-of-service attack detection in cloud computing utilizing a HIDS is presented. To detect network-based attacks in a network consisting of multiple computers, a HIDS must be installed on each of them. If a computer has been compromised or disabled by an attacker, the attack detection system is no longer trustworthy. Monitoring mainly takes place via four types of parameters. Those are the file system, the log files, the operating system kernel and the network connections. For systems that monitor changes to the file system, an attempt is made to detect when an attacker wants to gain control over the file system. For this purpose, changes to

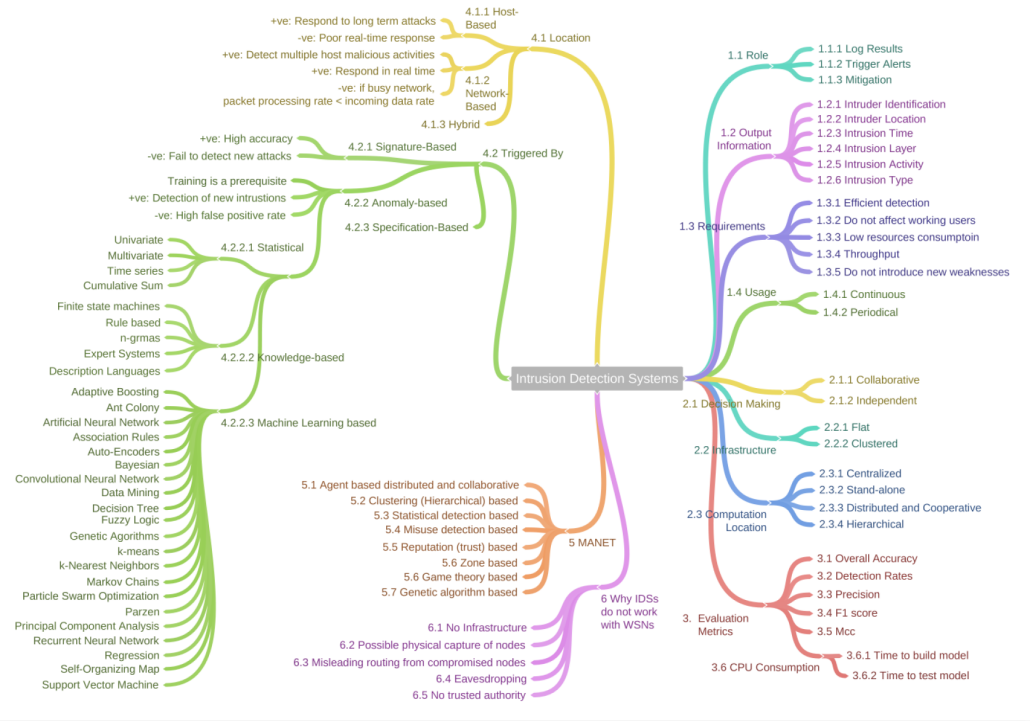

Figure 2: Taxonomy and survey of IDSs [3]

file sizes or file permissions are checked and evaluated cyclically. This method becomes problematic if the file system often undergoes changes, for example through the installation of software. Another possibility is to monitor log files of installed programs. If these reports contain information about possible attacks, the responsible administrators or users of the affected device can be warned. In addition, attacks can be detected directly at the operating system kernel level. The kernel has the ability to monitor all activities of a computer, so it can draw conclusions about malicious activities based on system calls and their parameters. As a countermeasure, the operating system can terminate the affected processes and thus render them harmless. In contrast to a NIDS, the content of network packets is not taken into account by HIDSs. Instead, connections to unauthorized ports are monitored and reported. In addition, port scans and an excessive number of connection attempts can be detected.

NIDSs, in contrast, reside on the network level to monitor and analyze the network traffic or application protocol activity. Those can either be deployed as dedicated sensors/agents either leveraged as specialized hardware or applied as software on a networking element. They record network packets and evaluate them according to e.g. previously described rules or patterns. The former is called *Blacklisting*. Here, rules are defined that describe the properties of network traffic during an attack. In contrast, the other ap-

proach, *Whitelisting*, defines a set of rules containing information about the usual network traffic. If a rule applies, the network traffic in question is not evaluated as an attack. In contrast to host-based attack detection systems, attacks can still be detected if several computers in a network have failed or have been taken over. Since today's networks are built with switches that send incoming network packets only to dedicated ports, the sensors of an NIDS must be connected to the mirroring port. In addition to connecting the target device, all network packets are sent to this port. Another difficulty is the maximum bandwidth of the sensor. The data throughput of modern networks can exceed the processing capability of a sensor and it must discard network packets. This means that a complete monitoring of the network traffic is no longer possible. If these two disadvantages do not occur, an entire computer network can be monitored with a single sensor. The main attention over the past years focused on the application of NIDSs, due to the advent of Anomaly-based NIDSs (A-NIDSs) [15] which can be placed either centralized, decentralized or distributed within networks on either network switching entities (router, gateway, etc.) or dedicated hosts.

2.1.2 Detection Methods

Detection methods for IDSs can be categorized into anomaly-, signature-, hybrid-, and specification-based approaches [16, 17, 18] as shown in Figure 3. However, one may distinguish between two major ones: *misuse-based* and *anomaly-based*.

The misuse-based method, also called signature- or knowledge-based, refers to the detection of attacks whose patterns are already known, such as byte sequences in network traffic. Thus, it 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. Although signature-based IDSs easily detect known attacks, it is impossible to detect unknown attacks whose patterns are not available. Therefore, a main drawback is the lack of signatures that describe all possible variations and non-intrusive activities in network environments [19]. The anomaly-based detection method creates a model of trusted activity from collected data samples and then compares new behavior with that model. Although it allows to detect novel, unknown attacks, it could lead to false negative and false positive alarms, in which trusted but previously unknown activities could be classified as malicious.

Both approaches have their merits and demerits for instance anomaly-based IDSs have a great potential in detecting novel attacks but they tend to be computationally intensive and are prone to false alarm generation. In contrast, misuse-based IDSs are fast in detecting known attacks with a very high accuracy and low false alarm rate but are limited in detecting new attacks. Thus, in recent years, hybrid approaches, e.g. in [20], have crystallized as the trend towards sophisticated IDS solutions. Hybrid IDSs usually

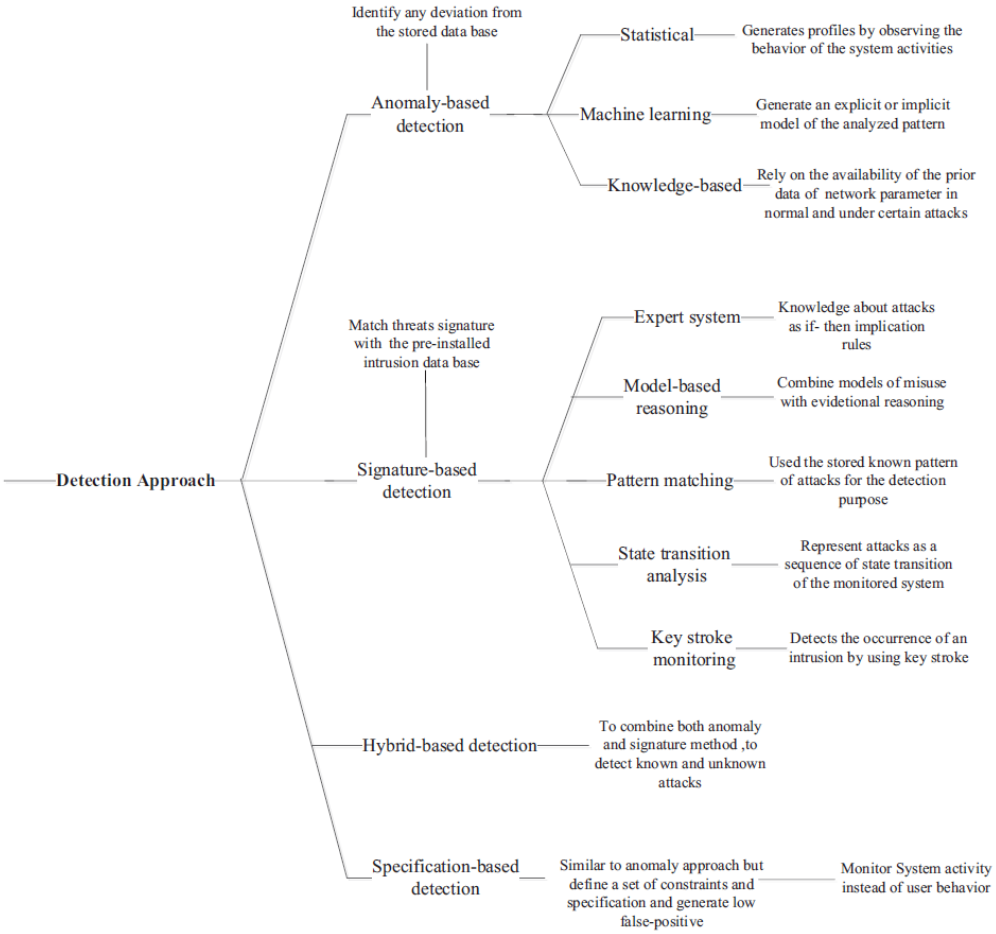

Figure 3: Taxonomy of detection approaches for IDSs [16]

combine the properties of anomaly- and misuse-based IDSs. Thus the advantages of both systems can be used by combining the methods sequentially or in parallel. Another possibility is to combine several anomaly-based methods in order to achieve better detection rates and reduce the number of false alarms. Various techniques have been proposed for detecting incidents based on misuse- and anomaly-based methods, e.g. discussed in [21]. A comprehensive overview of anomaly-based techniques with a focus on statistical techniques and systems, classification-based techniques and systems, clustering and outlier-based techniques and systems, soft computing-based techniques and systems, knowledge-based techniques and systems, and techniques and systems based on combination learners can be found in [7].

Misuse-based Techniques

Techniques for misuse-based IDSs can, according to [22], be based on

- signatures (monitored events are matched against a database of known attack signatures),

- rules (set of "if-then" implication rules to characterize attacks),
- transitions (IDS is composed of a finite state machine where state transitions are used to monitor the system behavior) or
- data mining methods (learning algorithm is trained over a set of labelled "normal" or "intrusive" data).

Anomaly-based Techniques
Techniques for anomaly-based IDSs can, according to [7, 22], be based on

- statistical methods (measure certain system variables over time and derive statistical values e.g. average, standard deviation),
- rules (normal behavior is summarized by a set of rules and anomalous behavior as a deviation from them),
- knowledge-based approaches (rule and expert systems, ontology and logic based),
- soft computing (e.g. genetic algorithms, fuzzy sets),
- distance-based approaches (attempt to overcome limitations of statistical outlier detection approaches in higher dimensional spaces where it becomes increasingly difficult and inaccurate to estimate the multidimensional distributions of the data points and detecting outliers by computing distances among points),
- model/classification-based approaches (anomalies are detected as deviations from a model that represents the normal behavior by using data mining / machine learning techniques e.g. Neural Networks) or
- profiling methods (profiles of normal behavior are built for different types of network traffic, users, programs, etc., and deviations from them are considered as intrusions utilizing data mining techniques or heuristic-based approaches).

2.1.3 Modes and Placement

Operating modes of IDSs can either be *online* (system learns and/or detects anomalies online [close] to real-time) or *offline* (system learns and/or detects anomalies offline) by working *passively* (system is configured to only monitor and analyze network traffic and alert an operator to vulnerabilities and attacks) or *reactively* (system works as in the passive mode and additionally takes pre-defined proactive actions to respond to the threat). If an IDS is operated in online mode, it can detect anomalies during operation and can be partially updated at runtime to create new models. This is a mandatory requirement for systems that are used in real scenarios. In some areas, even real-time recording of incidents is desirable. Systems that work offline are usually applied to existing data sets. By evaluating an IDS solution using existing data sets, one has better comparability with other

(competing) ones. This is particularly popular for scientific work. According to [3, 23, 18], IDSs can be categorized as follows. This especially applies to NIDSs placed in network environments.

- **Centralized**: The centralized computation location works on data collected from the whole network. In the centralized IDS placement, the IDS is placed in a centralized component, for example, in the border router or a dedicated host. A disadvantage of this architecture is that it is difficult (especially with larger networks) to collect all important data at a central instance. This makes a systemic approach almost impossible to implement.
- **Distributed**: Unlike the centralized, the stand-alone computation location works on local data, disregarding decisions from other nodes. In this placement strategy, IDSs are placed in every physical object of the network.
- **Decentralized**: Similar to distributed but the placement follows a certain strategy, e.g. the network topology/hierarchy. An advantage here is that all important data can be captured. Disadvantage can be that not all participants in the network have enough resources available in order to have a complete distributed approach of the methods and to leave certain parts out of the overall system.
- **Hybrid**: Hybrid IDS placement combines concepts of centralized and distributed or decentralized placement to take advantage of their strong points and avoid their drawbacks. A combination of both, centralized and stand-alone, can be achieved through cooperative computation, such that each node can detect an intrusion on its own but also contributes to the overall decision. Here, more powerful components are used at central locations, e.g. to create models, which usually requires more resources. The generated models are then distributed in the network to perform detection. These tests can also be performed by weaker systems, as they require far fewer resources.

Approaches which analyze the data traffic of an entire network with a central component are not target-oriented, since the amount of data to be considered requires a high computing power and thus binds resources. In addition, it is necessary to forward the data to be analyzed to the analysis component, which leads to an additional network load. Furthermore, networks are divided into subnets and an analysis of this kind initially provides little information about the network area in which an anomaly occurs. An alternative, novel concept is the decentralized use of anomaly detection systems. Here network sensors are used which are placed in the subnets of a network. Instead of analyzing the entire network traffic, the data traffic within each subnet is considered separately and only the anomalous alarms are passed on to a central component that correlates them. In addition,

having more and more distributively connected devices, collaborative IDSs therefore promise to even detect highly advanced distributed attacks. Especially decentralized NIDSs are popular and a selection of research work dedicated to them is presented in the following. Already in 2001, an architecture is proposed in [24] that collects decentralized network traffic and sends it to a central server which classifies it. The drawback of this approach is the increased network overhead. Statistical methods are first applied to the monitored data sets in order to classify the outputs with a Neural Network.

Jahnke [25] defines requirements for a decentralized attack detection architecture. These include the ability to work continuously without human interaction, to detect attacks on the IDS itself, or to adapt the IDS to the system or network behavior over time. Jahnke also proposes the use of the Intrusion Detection Message Exchange Format (IDMEF) [26], a data format based on the eXtensilble Markup Language (XML), as a structure for communication between the components of his architecture. Six components are proposed: The sensor is defined as a process that collects or generates measurement data. For each sensor, an adapter is required that monitors the work of the sensor and processes the aggregated data and transforms it into the required form of the IDS. The message distributor is required to receive or send command messages. This is done via the communication channels, of which there are basically two, on the one hand the already addressed command messages, on the other hand one for special events, such as a detected attack. Event processing evaluates the events and determines the behavior of the last component, the reaction unit. However, there is no separation of command and event messages. Even a proposed response measure is not part of this work. A response action requires knowledge of the underlying network and the available response units. Lupu et al. [27] developed a decentralized architecture for attack detection and implemented a communication framework based on IDMEF. The existing libraries for IDMEF, *libprelude-dev* and *libidmef*, has been discussed but an improved software library adapted to the decentralized architecture has been developed. In order to define the reaction to received alarms, an own syntax is used which is based on the developed functions. A programmer is offered the possibility to register his own callbacks which are called at every event, i.e. an incoming message. Thus, a programmer can influence the role of a client himself. The use of IDMEF as an alarm format is suggested because of the suitable structure and the meaningful definition of fields. The presented architecture, however, mostly refers to the evaluation of these alarms and less defines the structure of the sensors. Nevertheless, it represents a suitable basis for the evaluation of generated alarms in a decentralized structure. Hu et al. in [28] presented a method for detecting anomalies in network traffic. They are specifically designed to address the problem of the frequently changing structures of today's computer networks. In addition to the investigation of several different algorithms for the detection of anomalies, a new

architecture of NIDSs is presented. The data of each packet passes through several stages of anomaly detection. First, a local model is used that is only locally present in the respective node. Then further global models, which are present in all nodes of the network, are applied. The approach demands many computing resources. However, those are divided between sensors and a server.

2.2 Anomaly Detection with Machine Learning

An essential focus in current research is anomaly detection using *machine learning* (and data mining methods), which makes it possible to detect both previously known and unknown attacks [29, 30]. An advantage of anomaly detection is the ability to detect impending failures, as they cause a behavior that deviates from the standard of a system. With machine learning methods, various problems can be solved and is also preferably used in the field of anomaly detection. There are a number of algorithms that serve these application areas. In principle, however, a data set is required for each method, which is used in the training phase for parameterization.

2.2.1 Aspects of Machine Learning

A *data set* is a set of m data points with n dimensions in which a dimension is also called a *feature*. In addition to the actual data, a *label*, i.e. the expected result value for the data point can also be available. These labels are required for some algorithms in the training phase. The data set used for the training phase is called the training data set. A data set normally contains raw data which must be translated into an understandable format for a downstream applied algorithm. The corresponding procedure is called *pre-processing*.

Pre-processing is a significant part of machine learning and essential in order to enable an efficient data analysis and to improve the performance of the algorithm [31, 32]. Data must first be collected and corresponding information must be extracted from the raw data (*feature extraction*). Furthermore, a selection of the relevant data for the analysis must be carried out. This is followed by the actual pre-processing of the data. Possible goals of the data pre-processing are, for example, to convert the data into an optimal form for the analysis in order to increase the performance of the anomaly detection or to reduce the amount of data to be analyzed and thus to preserve resources. Data pre-processing and analysis must take place during the operation of the underlying network with the intention of detecting anomalies promptly. In general data pre-processing describes the necessary steps before an analysis of data can occur. Data pre-processing methods (with their respective functions) can be divided into the categories *data cleaning* (handling of anomalous, missing, erroneous, inconsistent data), *data integra-*

tion (merging of multiple data sources, handling of redundant data), *data transformation* (scaling, normalization and categorization of data) and *data reduction* (dimensionality and quantity reduction, discretization, compression of data) [33, 34]. Methods for data scaling or normalization are among others [35] the min-max or unit scaling (L1- or L2-norm). Two popular methods exist for dimensionality reduction [36]: Principal Component Analysis (PCA) and Random Projections. PCA originates from statistics and is used to locate patterns in high-dimensional data. After such a pattern has been found in a set of data, its dimension can be reduced (compression). Decisive is that with the reduction, the information represented by the data is largely preserved. Due to the Johnson-Lindenstrauss lemma described in [37] several possibilities were developed to map higher order matrices into lower dimensions. This lemma states that at data points in a high dimension only a small distortion occurs by mapping into a lower dimension with a certain probability. Random projection is taking advantage of this lemma. Clustering is another method of pre-processing data. Here, data is assigned to clusters according to its nature, such that the data is categorized. This classification can be used as a pre-processing measure to implement various measures: (1) The categorization of data may assist the analytic process in the classification of data. (2) The type of further processing can be selected on the basis of the cluster membership of data. (3) Depending on the cluster, data sampling is possible. (4) Each cluster can be further processed by a separate algorithm which allows a detailed analysis of the data. Sampling in statistics refers to the selection of a subset of instances from within a statistical population in order to estimate characteristics of the whole population. In pre-processing, sampling selects a subset, a representative, of a set to allow an analysis of only the subset while loosing as little information as possible. By this measure, the amount of data for analysis is reduced, and therefore fewer resources are required. For a discussion on various sampling methods refer to Subsection 4.3.1.

Before features can be extracted from the data, they have to be identified. This process is called *feature selection*. "Feature Engineering" [38] and "Feature Learning" [39] plays an important role in building machine learning based IDSs since the chosen feature set (the collection of selected features) highly affects the performance of the IDS. For network-based features, one may distinguish basic features (derived from packet headers [meta data] without inspecting the payload, e.g. ports, MAC or IP addresses), content-based features (derived from payload assessment having domain knowledge e.g. protocol specification), time-based features (temporal features obtained from e.g. message transmission frequency, sliding window approaches) and connection-based features (obtained from a historical window incorporating the last n packets) [7, 40]. For instance, Wang et al. propose an automated feature learning approach in [41]. They abstract network traffic such that for spatial features traffic is transformed into "traffic images" to exploit the

advantages of image processing e.g. image classification based on geometric features to classify the traffic images, which also indirectly achieves the goal of identifying malicious traffic. For temporal features, the time series analysis method is applied to detect malicious behavior on extracted temporal features. The so-called Hierarchical Spatial-Temporal Features-based IDS is divided into two major steps: first the low-level spatial features of network traffic are learned using deep convolutional neural networks and then high-level temporal features are learned using long term short memory networks. For dimensionality reduction instead of the well-known PCA, the t-SNE algorithm is used. Different feature representations can be used to address different fields of anomaly detection. Some of them are considered naive when they contain basic information about the software or network (e.g. IP source and destination address of a data packet), while others are considered rich when they represent deeper details (e.g. temporal relations of payload content) [38]. According to [3], features can be obtained by the following processes: feature construction creates new features by mining existing ones by finding missing relations within features. While extraction works on raw data and/or features and apply mapping functions to extract new ones. Selection works on getting a significant subset of features. This helps reduce the feature space and reduce the computational power [3]. "Unsupervised Feature Selection" tries to find a relevant subset of features that preserves the inherent structure as much as possible [42]. This means that it tries to reduce the number of features without complicating the detection of anomalies. Since feature selection is an NP-hard problem [43], there are many approximation solution methods. Luo et al. [44] follows a way to select features based on "Adaptive Reconstruction Graphs". This led to the realization that omitting features can improve the result. If some features can be ignored, the calculation time is also reduced. Wieland et al. [43] follows an approach based on a Support Vector Machine (SVM). This is used to model the relationship between the distribution of a particularly invasive mosquito species. The work was able to identify new features that improve the underlying detector. However, the method used is extremely computationally demanding and therefore not suitable for the usage on low-powered devices. Aljawarneh et al. [45] developed a hybrid model using the following classifiers: J48, Meta Pagging, RandomTree, REPTree, AdaBoostM1, DecisionStump and NaiveBayes. An information gain detector based on mutual information was used to assign an information score to all possible features. The hybrid model was then applied to the best eight features. The calculation of the information gain detector is unfortunately dependent on information that is not always available when using unsupervised learning.

Many algorithms used in machine learning work in two (or three) phases, the *training phase*, (the verification phase) and the *evaluation phase*. In the training phase, a state or *model* is build by the selected algorithm. This model is then used by the algorithm in the evaluation phase to obtain the

result. An additional verification phase might help to verify and optimize a built model before the actual operating (evaluation) phase. However, there are also algorithms that work in one step. In most cases, this involves obtaining information from the existing training data set (data mining). Basically, machine learning based algorithms can be classified into three learning methods. The biggest difference can be seen in the already known basis of information. The learning method can already provide information about which problem the algorithm can solve, e.g. labels for the classification problem. In *supervised learning* (1), e.g. [46], the labels for the training data set are required. These are included in the models during the training phase. This means that the algorithm adapts the values of the model to the known labels. The goal is to approximate an unknown function $f(x)$ with the resulting value Y. Here x are the data points and Y are the known labels. If the function $f(x)$ is approximated, the resulting value can be calculated for each additional data point. The *unsupervised learning* (2), e.g. [47], is classified by algorithms that do not require labels in the training phase. Mostly, these algorithms are used to analyze the data set more precisely and to model it. Clustering, i.e. the division of similar data points into groups, is a well-known representative of this learning method. *Semi-supervised learning* (3), e.g. [48], is a trade-off between supervised learning and unsupervised learning. Here a label is only available for a subset of the data points. This procedure is used if there are many data points and a label cannot be assigned to them completely manually. *Hybrid* approaches, e.g. [49], try to exploit the benefits of the aforementioned for instance the better detection rate when having labelled training data and mitigate their demerits e.g. having a false alarm rate when assuming that normal data points are far more frequent than anomalies.

Furthermore, in machine learning different types of problems, that should be solved with methods of machine learning or data mining, are distinguished. In *classification*, data should be divided into already known classes or groups. The *regression* problem is very similar to the classification problem, but the result is not a class but a numerical value. With *clustering*, a set of data points shall be divided into classes. Each class should only contain points that are similar. Here it can be further differentiated whether the number of contained classes is already known before or an unknown number of classes are contained in the data set. The task *outlier detection* belongs to the class of algorithms which can detect anomalies with the properties that these are few compared to the normal data and differ substantially from it. Such algorithms detect outliers during the evaluation phase without previously known information about the data e.g. clusters.

2.2.2 Example of Two Outlier Detection Algorithms

Anomaly detection is a subcategory of classification since there are only two classes: the anomaly and the normal data class. Often it is also desired to get not only the class but also a value indicating how likely a data point is an anomaly, thus assigning scoring values to data points. Bhuyan in [7] categorizes anomalies into performance-related (caused by network or system failures or performance degradation e.g. broadcast storm, babbling idiots, transient congestion, vulnerabilities) and security-related (disagreement of normal and expected network traffic) ones. According to [50], three kind of security-related anomalies exist: (1) point anomalies, if a single event can be considered anomalous given a notation of normality, (2) contextual anomalies when an event can be considered anomalous in respect to a given context deduced from an event's behavioral attributes (the same attributes might not be considered anomalous in another context), and (3) collective anomalies, if a series of events is considered anomalous (even if single events are not considered anomalous their collective relation might be [51]). Context-aware IDSs are able to include particular context e.g. network topology, protocols, system configuration when trying to identify malicious activity. A categorization of four classes into network-related context, target configuration, vulnerability assessment and attack side effects is shortly discussed in [5]. In the following two representative examples (Isolation Forest [52] and Loda [53]) for anomaly detection algorithms based on unsupervised outlier detection algorithms are presented which have been chosen according to the following criteria for the application as a NIDS:

- operation without knowledge of data labels
- possibility of online (real-time) detection of anomalies
- detection of previously unknown, distributed and advanced attacks (e.g. APT)
- detection of point and context anomalies
- modelling and operation in environments which might contain anomalous data
- applicability on lightweight devices with little available resources
- scalability and flexibility in use
- coping with high dimensional data sets

The **Isolation Forest** algorithm is predestined for anomaly detection because it does not use distance or density methods which makes it much less computational intensive compared to methods such as KNN [54] and also because it is well suited for real-time usage unlike most algorithms [53]. In addition, it belongs to unsupervised learning and does not require a labelled training data set, such as Hoeffding Trees [55], as it recognizes patterns and does not sort out packets based on their label. Furthermore, it is independent

of the scaling of the data set dimensions since its threshold for determining anomalies is based on the tree depth [56]. This algorithm attempts to separate outliers from other data points by isolating them by taking advantage of the fact that data points that differ from other data points require fewer steps to be isolated from them as shown in Figure 4 (a). In addition, the algorithm uses the observation that when a data set is represented in a binary search tree, anomalies are inserted in a tree at a shallower depth than normal values as depicted in Figure 4 (b).

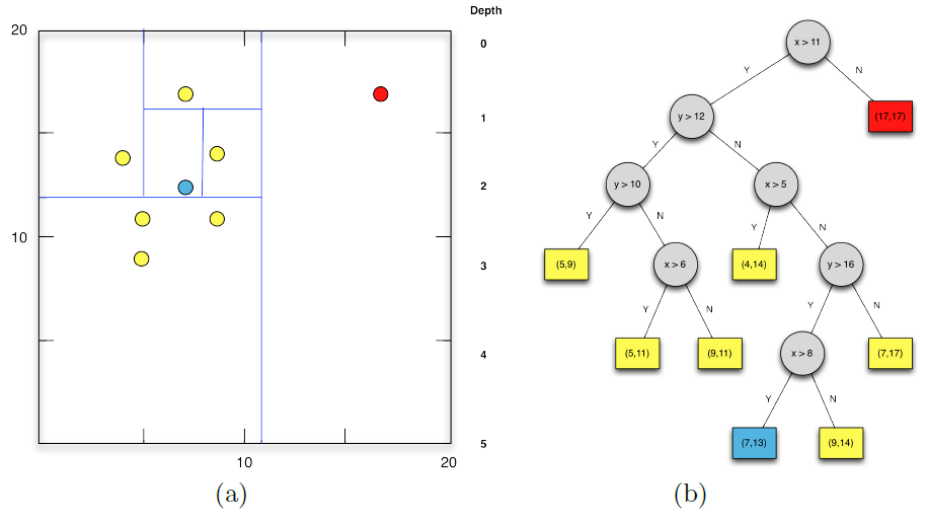

Figure 4: (a) Isolating an outlier, (b) Representation of a tree model [56]

The Isolation Forest algorithm forms several *Isolation Trees* in the training phase. These Isolation Trees are then the model for classification in the evaluation phase. Each tree is a real binary tree, whose nodes are provided with different information. In the training phase, the training data set X is available with n data points. The training data set is divided into t subsets X' and X. It applies $X' \subset X$. Each subset X' contains ψ data points. An Isolation Tree is formed from each X'. This is done by recursively dividing X' by randomly choosing a feature q and a value p. p is a random value between the minimum and the maximum of the feature q of all data points at a node of the tree. New child nodes are formed by processing all data points for which $p_q < p$ applies, where p_q is the value of the feature q of a data point, in the left child node. All data points to which $p_q \geq p$ applies are processed further in the right child node. The recursion ends when fewer than two data points have arrived in a node or all data points are equal. The values q and p become attributes of the current node. Each node of a tree has either no child nodes or two. This is repeated for each X'. After the training phase, t Isolation Trees exist.

To classify a data point x, all Isolation Trees are traversed from the data point during the evaluation phase. The data point travels through the

nodes to the previously trained values for q and p. If the data point $p_q < p$ applies, if p_q is the value of the feature q of the data point, the data point moves to the left child node. If $p_q \geq p$ applies, the data point moves to the right child node. This happens until an end node is reached. The result of this migration is the tree depth. From all reached depths $h(x)$ the average $E(h(x))$ is computed. The calculation of the resulting *score* is shown in Equation 1. First, $c(\psi)$ must be calculated. This equation is borrowed from the number of unsuccessful searches in a binary search tree. It represents the average depth reached by a binary tree when it contains ψ data points. n is the number of data points used to build a model. H represents the "$\psi-1$"-th subsequent element of the harmonic sequence. This can be calculated approximately by Equation 3. The variable γ represents the Euler-Mascheroni constant ($\approx 0,57721$). The main advantage of the Isolation Forest algorithm is its low time complexity. This is $O(t\psi^2)$ in the training phase and $O(nt\psi)$ in the evaluation phase. It should be noted in particular that ψ can and should be kept small to avoid the effect of *swamping*.

$$score(x, \psi) = 2^{-\frac{E(h(x)))}{c(\psi)}} \tag{1}$$

$$c(\psi) = \begin{cases} 2H(\psi - 1) - \frac{2(\psi-1)}{n} & \text{for } \psi > 2, \\ 1 & \text{for } \psi = 2, \\ 0 & \text{for } \psi < 2 \end{cases} \tag{2}$$

$$H(n) \approx ln(n) - \gamma \tag{3}$$

Loda [53] is presented as another algorithm besides Isolation Forest with similar properties. It belongs to the outlier detection algorithms with unsupervised learning as well and consists of a collection of k one-dimensional histograms, each histogram approximates the probability density of the input data projected onto a single projection vector. Projection vectors diversify individual histograms which is a necessary condition to improve the performance of individual classifiers. To train the algorithm, projection vectors w_i are first generated and histograms initialized. Each projection vector is generated during the initialization of the associated histogram by first randomly selecting $d^{-\frac{1}{2}}$, different from zero, features and then randomly generating non-zero values according to $\mathcal{N}(0, 1)$. The histograms of each projection vector are updated with $z_i = x_j^T w_i$, where x^T is the transposed sample vector. The features used must be of approximately the same order of magnitude. Loda's output $f(x)$ on a sample x is the average of the logarithm of probabilities estimated on a single projection vector (Equation 4).

$$f(x) = -\frac{1}{k}\sum_{i=1}^{k} \log \hat{p}_i(x^T w_i) \tag{4}$$

Loda is especially useful in domains where a large amount of samples have to be processed because its design achieves a very good weighting between accuracy and complexity. The algorithm exists in different variants for batch and online learning. With the batch variant, data instances are collected and collectively used in the training routine. In this routine the projection vectors and the histograms are generated. In the online version, a histogram is continuously updated which makes it possible to use even on devices with very low resources and thus eliminates the splitting of the modelling phase and evaluation phase (as necessary with Isolation Forest). However, in the early running time of the online variant, the algorithm will probably produce more false positives than the batch version, since the histograms need a certain amount of time to be fully updated. Loda can handle missing variables and can sort features according to their contribution to the anomaly score. Also the anomaly detection does not fail completely if single sensors are missing. In its original form, the algorithm returns a score value. The larger the value, the more likely it is an indication of an anomaly. However, this score value can be reduced to a probability by Equation 5). Here $f(x)$ is the score value of Loda from Equation 4.

$$\hat{p}(x) = 1 - e^{-f(x)} \tag{5}$$

The complexity theory provides a measure for the representation of the differences between Isolation Forest and Loda. Table 1 shows the effort of resources (time and memory complexity) for the execution of both anomaly detection algorithms according to [53]. In the table, n denotes the number of samples for the training phase, d the number of features (dimensions), k the number of trees (Isolation Forest) or the number of histograms (Loda), l the number of samples for the construction of a single tree (Isolation Forest) or the length of an observation window for the continuous histograms (Loda) and b the number of histogram classes (Histogram bins) for Loda. In Table 1 a distinction is made between Loda with two alternating histograms (1) and the implementation with a continuously updated histogram (2). The construct of a binary tree contained in the Isolation Forest can be created in two ways. If all elements for the creation are not known in advance, then each element of the l elements must be added one after another. In the worst case, the complexity of the insertion is $\mathcal{O}(l)$ and so $\mathcal{O}(l^2)$ results. In the more likely case (also called average case), all l elements are known in advance and could be sorted by $\mathcal{O}(l \log l)$ and inserted afterwards. For this, one takes the middle element, insert it as root node and proceed recursively for the remaining elements. At the end one gets a so-called "balanced" tree, where l elements were inserted with $\log l$. Thus, a time complexity of $\mathcal{O}(l \log l)$ can also be achieved for the creation. The memory complexity when learning the model is $\mathcal{O}(n)$, where n is the number of elements in the data set to be learned. Once the model has been trained, the memory complexity is reduced to the number of memory complexities per binary tree $\mathcal{O}(k \log l)$

and is significantly less than $\mathcal{O}(n)$. In contrast to the learning phase, the time complexity of the classification is reduced by the number of subsamples l and results for a binary tree in $\mathcal{O}(l)$ in the worst case and $\mathcal{O}(\log l)$ in the average case. Thus, the number of k trees for the time complexity is $\mathcal{O}(kl)$ in the worst case and $\mathcal{O}(k \log l)$ in the average case. For the Loda variant (2), the time complexity for the training phase is $\mathcal{O}(nkd^{-1/2})$. The complexity results mainly from the nested loops with the limits n and k. The use of "Very Sparse Random Projection" [57] yields a speedup of $\sqrt{d}$ from which the factor $d^{-1/2}$ results.

	Time complexity		**Space complexity**
	Training	**Classification**	
Isolation Forest	$\mathcal{O}(kl \log l)$	$\mathcal{O}(k \log l)$	$\mathcal{O}(kl)$
Loda (1)	$\mathcal{O}(nkd^{-1/2})$	$\mathcal{O}(k(d^{-1/2} + b))$	$\mathcal{O}(k(d^{-1/2} + b))$
Loda (2)	$\mathcal{O}(nkd^{-1/2})$	$\mathcal{O}(kd^{-1/2})$	$\mathcal{O}(k(d^{-1/2} + b + l))$

Table 1: Time/space complexity of Isolation Forest and Loda (cf. [53])

2.2.3 Combining Classifiers

Apart from an incident analysis (refer to Section 3 - aggregation, alert fusion), which combines the alarms of different detection measures (*algorithm external combination*), there are also mechanisms that can be used for consensus finding within an algorithm (*algorithm internal combination*). The combination of learners is categorized by Bhuyan in [7] to ensemble-based techniques (algorithm internal) utilizing bagging, boosting and stack generalization, fusion-based techniques (algorithm external) combining several disparate data sources at the data level, feature level or decision level. Sadighian in [5] categorizes fusion approaches into *Winner-take-all* approaches (final decision over the outputs from various IDSs is made based on the decision of the IDS that has the highest measurement value, e.g. majority vote, weighted majority vote, behavior knowledge space, naive-Bayes combination, and Dempster-Shafer combination) and *Weight-based* approaches (assigning weights to each IDS as its importance indicator on the final decision which is then made based on the weighted sum of the measurement values of all the IDSs, e.g. using Neural Networks or weighted average).

The concept of ensemble-based approaches combining several "weak" classifiers in order to gain a "strong" one is becoming more and more popular. These methods weigh the individual outputs and combine them (ensemble) to obtain a better results. Loda, as well as Isolation Forest, are based on the principle of producing a strong classifier by combining multiple weak ones (trees/histograms). However, there exist further work exploiting this concept. Amudha et al. [58] investigates bagging and boosting as two possible

methods for ensemble learning methods. Bagging performs random sampling, whereas boosting performs sampling based on a continuously updated distribution. Hu et al. introduced the AdaBoost algorithm in [59], which like Loda, generates a strong classifier from weak classifiers (decision stumps). Kitsune uses an ensemble of simple neural networks to distinguish between anomalies and normal behavior. The research of Mirsky et al. [60] shows that Kitsune works comparably to offline anomaly detectors and uses few resources. All this research work show that an ensemble of weak classifiers provides better results than individual classifiers and works at the same level as strong classifiers, while even preserving resources.

Kittler et al. [61] present many basic considerations for the combination of classifiers. These include many rules, such as the product, sum, minimum, maximum and median rules, as well as majority voting. A surprising finding is that the comparatively restrictive sum rule even produced better results than other rules. Voting, in general, is used to generate a collective decision. The four main components of a voting algorithm are input data, output data, input votes and output votes. Exact and inexact in this context indicates whether input objects are regarded as inflexible values or flexible neighborhoods, i.e. whether discrete or non-discrete values exist. There are different types of voting algorithms, e.g. consensus and compromise voting. Compromise are mainly voting variants based on the median or mean. Preset and adaptive indicate whether weightings are set or can change over runtime. Other variants are called threshold and plurality. Threshold voting means that the output weight exceeds a value, where plurality identifies an output that has maximum support from the inputs. [62]

With consensus voting, an anomaly is only recognized if all classifiers recognize it. In majority voting, an anomaly is detected when the majority detects an anomaly. Consensus voting prefers false negatives compared to false positives [63]. Gao et al. [64] describe the use of consensus voting for multiple atomic detectors to improve detection rates. Lin et al. [65] describe a creditability-based weighted voting system that assesses the creditability of each anomaly detection algorithm. This is done by comparing the results of the algorithms with known results of the network trace, in particular the information of the confusion matrix parameters. Unfortunately this is not compatible with unsupervised learning. Thus, a way must be found to obtain information whether an anomaly has been correctly detected or not. This is difficult to be achieved with unsupervised learning as there is no such information available. Based on this comparison, the weightings of the individual algorithms are then determined. Giacinto et al. [66] investigate different approaches of disparate classification to obtain a single result. They judge the "Dynamic Classifier Selection" algorithm as the best one. It selects for each pattern the classifier that finds the correct classification, if such a classifier exists. Aburomman et al. [67] introduce several ways to combine different classifiers and state that voting-based systems are the common

method. Errors introduced by one classifier can be corrected by another if all classifiers have a similar performance. If the reliability of each classifier can be estimated in advance, it is possible to increase the accuracy by weighted voting. Weighted voting can be used more generally than simple majority voting and is therefore useful in a broader context. If all weightings are set to 1, simple unweighted voting results. It is important to note that the classifiers must be sufficiently different, otherwise there will be no significant improvements.

Many recent publications deal with the use of machine learning algorithms for anomaly detection. In [68, 69, 70] different algorithms and methods are tested. In some cases, multi-stage methods are presented. Several algorithms are concatenated to achieve better results. Disadvantage is that the computational complexity of this method is higher by the application of several algorithms than with single-staged ones. Therefore, this approach is less suitable in environments characterized by less available resources without any modification or combination with other methods like sampling. The doctoral thesis of Taylor [20] presents a hybrid automotive anomaly-based IDS with a two-staged detector. Special attention is paid to frequency-based and sequence-based detection, which are specified for their application in order to identify CAN-frames which deviate from their normal transmission frequency or from their order in transmission (sequence). A so called anomaly score is calculated for consensus finding of the different systems and as a decisive feature for alarm generation. The authors of [71, 72] proposed a lightweight IDS for wireless sensor networks based on the combination of the anomaly- and misuse-based technique to offer a high detection rate. The approach is integrated in a cluster-based topology, to reduce communication costs, which leads to improving the lifetime of the network. The incoming data is first provided to the faster signature-based component and if indicated abnormal provided to the anomaly-based SVM. A decision making model combines then the outputs of both techniques, determines whether an intrusion occurred and classifies the type of the attack. The incident is then reported to an administrator for supervision. Guo et al. present a two-staged hybrid approach in [73] that deploys an anomaly detection component in the first stage and its output in a second stage either forwarded to a second anomaly detection component (in the abnormal case of stage 1) or forwarded to a misuse detection component (in the normal case of stage 1). The misuse-based component is able to classify between an attack or not and the anomaly-based component between normal and abnormal connections. Since misuse-based techniques are typically less complex than anomaly-based ones a better approach would be to apply the misuse-based component in stage 1 similarly to the work in [74]. Thus, static checks are used which correspond to misuse-based (specification-based) detection by applying simple rules based on known communication matrices used in the automotive sector (CAN message catalogue). Those filter out inappropriate

communication e.g. exceeding payload values in a first place before features are extracted for a common basis to apply anomaly-based machine learning algorithms. A simple anomaly analyzer evaluates the outputs of e.g. recurrent neural networks, One-Class SVM (OCSVM) and Loda in order to filter out false positives before logging detected anomalies.

Maglaras et al. propose IT-OCSVM in [75], a distributed intrusion detection system in a SCADA network characterized by a three layer hierarchical abstraction into field network, operation network and IT network. It uses a central OCSVM and a cluster of automatically produced ones, one for each source that induces significant traffic in the system, an embedded ensemble mechanism, an aggregation method and a k-means clustering procedure that categorizes aggregated alerts using IDMEF messages. The detection functionality of the IT-OCSVM is composed of pre-processing (feature extraction from raw data containing all forms: continuous, discrete and symbolic and mapping to numeric-values), the selection of the most appropriate features (divided into content and time-based features), the creation of cluster of OCSVM models (trained on discrete sources), testing of the traffic dataset (containing malicious attacks), the ensemble of classifiers (combining the output of the different OCSVM modules using mean majority voting), social analysis (technique using Spearman rank correlation coefficient to add weight to alerts produces from different sources, e.g. difference between mainly used protocols during normal and abnormal operation of a node), the fusion of information/alarms (multiple anomaly outcomes are gathered and classified in terms of importance by k-mean clustering; groups alerts per source node and gives final scores to aggregated alerts based on the initial values and the number of similar initial alerts) and communication of the mechanism (IDMEF file exchange for alerts in terms of e.g. importance, position, time). The ensemble based mechanism for the outcome of the central and the split OCSVMs is computed with $q_e(i,j) = \sum_{n=1}^{N} w_i d_t(i,j)$ where $d_t(i,j)$ is the outcome of each classifier n for sample data i originating from node j with assigned weight w_i.

2.3 IDS Evaluation Metrics

According to [7], metrics for IDS evaluation can be divided into data quality (quality, reliability, validity, completeness of e.g. data source, selection of samples, sample size, time of data), correctness and efficiency as shown in the taxonomy of Figure 5.

Evaluation metrics to compare performance (efficiency) and effectiveness (correctness) can be generally classified into cost-based metrics, information-theoretical metrics [76], binary classification and resulting from binary classification, Receiver Operating Characteristic (ROC) [5, 77]. Efficiency deals with the resources needed by the system executing the IDS including e.g. CPU cycles or memory demands. Further the timeliness is a metric that

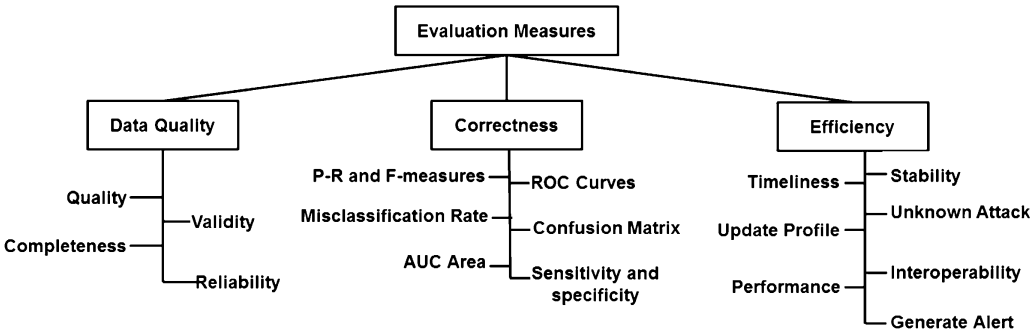

Figure 5: Taxonomy of evaluation measures [7]

defines how quickly a response is performed after an incident has been detected. Correctness represents the ability of the system to distinguish between malicious and non-malicious behavior (classification performance) by measures such as ROC curve, Area Under the Curve (AUC), precision, recall, F-measure, confusion matrix, misclassification rate, sensitivity, and specificity. Cost-based metrics assign a cost measure to weight false positive and false negative rate to consider a trade-off between the cost of a damage by a successful attack and the costs for impacts of false alarms. Especially for machine learning based IDSs, a high detection rate is essential. However, when measuring the accuracy of IDSs, particularly for the problem of statistical classification, different characteristic values are used. A so-called confusion matrix is utilized to compare the performance of such algorithms. The focus of the performance lies on the predictive power of a model and not on the speed the model performs classification into normal or abnormal classes (binary classification). The confusion matrix is represented by Table 2, in which each row represents the instances of a predicted class, while each column represents an actual class.

	Actual Non-Anomaly	**Actual Anomaly**
Predicted Non-Anomaly	True Negative (TN)	False Negative (FN)
Predicted Anomaly	False Positive (FP)	True Positive (TP)

Table 2: Confusion matrix for IDS evaluation

Where:

TN: normal event/behavior classified as a normal event/behavior
FN: intrusion/anomaly classified as a normal event/behavior
FP: normal event/behavior classified as an intrusion/anomaly
TP: intrusion/anomaly classified as an intrusion/anomaly

Many other characteristic values (sensitivity, specificity, positive/negative predictive value, etc. [5, 7, 78]) can be derived from the parameters of Table 2. Two examples, the False Positive Rate (FPR) and the True Positive Rate (TPR) computed by $FPR = \frac{FP}{FP+TN}$ and $TPR = \frac{TP}{TP+FN}$ are used to derive the ROC metric. The ROC-curve is a visual representation of the diagnostic ability of a binary classifier. An example of a ROC-curve is shown in Figure 6 in which the blue curve represents a random classifier whose output is completely random. The curve of a well-performing IDS is above the blue curve. This means that the top left corner of the plot is the "ideal" point with a FPR of zero, and a TPR of one. This is not very realistic but it does mean that a larger AUC is usually better. The "steepness" of ROC-curves is also important, since it is ideal to maximize the TPR while minimizing the FPR. With the help of such metrics results of different anomaly detection algorithms can be reliably compared or the anomaly detection algorithm under test can be optimized.

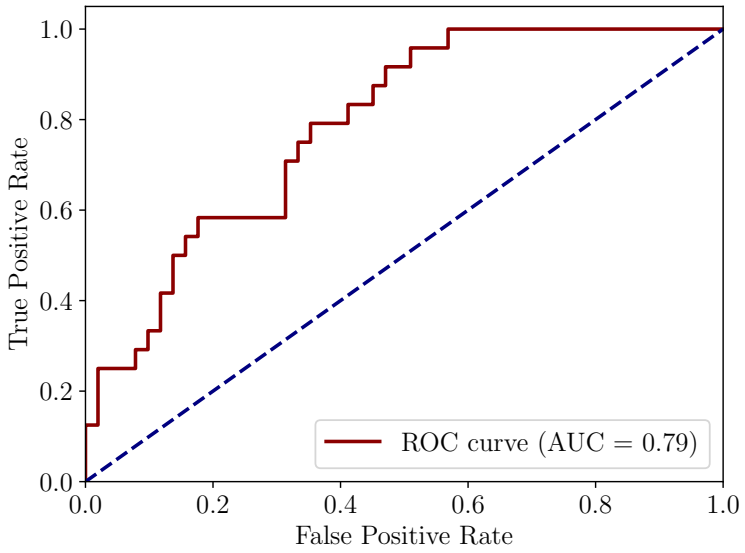

Figure 6: Example of an ROC-curve

Further performance metrics and capabilities that typically characterize IDSs are listed in the following.

- **Resource requirements (efficiency)**: Resources needed to be allocated by the system including memory usage, CPU load/cycles and disk space.
- **Overhead:** Computation and communication overhead - especially considering collaborative IDSs, a reasonable overhead of communication effort and computation must be achieved.

- **Throughput**: This metric defines the level of traffic up to which the IDS performs without dropping any data instance e.g. a packet.
- **Timeliness**: Average/maximal time between an intrusion's occurrence and its reporting.
- **Resilience**: States how resistant an IDS is to an attacker's attempt to disrupt the correct operation of the IDS or malfunctions of the component.
- **Ability to correlate event**: States how well an IDS correlates attack events from e.g. routers, firewalls, application logs. This already refers to incident analysis functionality.
- **Detection of "zero-day" intrusions**
- **Capacity verification for NIDSs**: Ability of inspection into deeper levels of e.g. network packets.
- **Stress Handling**: The point of breakdown is defined as the level of network or host traffic that results in a shutdown or malfunction of IDSs.
- **Depth/Coverage of detection capability**: It is defined as the number of attack signature patterns and/or behavior models known to it. (Which attacks can be detected?)
- **Reliability of attack detection**: It is defined as the ratio of false positives to total alarms raised - accuracy.
- **Error reporting and recovery**: The ability of an IDS to correctly report errors and recover from them.
- **Self-configuration:** Ability to automatically adjust itself without manual intervention.
- **Interaction capability with other systems**: The ability of an IDS to interact with other systems such as firewalls or anti-virus systems.
- **Attack analysis/identification**: It is the ability to report the extent of damage and compromise due to intrusions and to identify an attack based on common names or exploits (assumes 100% confidence).

3 Incident Analysis

Many works, e.g. [79], make the assumption for response planning that each raised alarm (output of an IDS) is treated as one attack (100% confidence of the alerts). However, this might be true when applying misuse-based IDSs which commonly have a high true positive and low false negative rate. In order to detect new attacks with high accuracy, the input of various detection mechanism (including anomaly-based ones) 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 in order to reduce false positives to a minimum. A comparison of supervised, semi-supervised and unsupervised learning methods for A-NIDSs has been examined in [80], each having its particular strengths but their detection capability differ significantly. Not only this circumstance but also (1) the handling of a massive amount of alerts from various applied detection sources is a requirement towards incident analysis and (2) the safe selection and execution of a following incident response measure. Hence, according to [81], IDSs are not enough to detect complex attacks over a network. Even they are able to detect some basic attacks, e.g. fabrication and suspension attack, they fail to detect more sophisticated ones such as the masquerade attack [82]. An intelligent incident (alert) analysis is therefore necessary in order to

- gain knowledge of multiple detection sources by using a unified format,
- identify the root cause of an incident,
- recognize pattern between the alerts and historic events,
- reduce the number of alerts, cluster and correlate them in order to prepare the essential information for an administrator,
- predict the propagation of malicious action (in this context referred to malware and cyber attacks).

Alert analysis techniques and methods help to manage and diagnose e.g. to deal with (a huge amount of) alerts gathered from (various) incident detection components by filtering out alerts, grouping and correlating them or prioritize important ones. Alerts typically incorporate information (alert feature) regarding the creation/detection time of an attack or suspicious event, its description and severity, etc. which is defined by the alert format used. According to [7], alert management contains three major components: *alert correlation*, *alert merging* (aggregation) and *alert clustering*. For the sake of generalization, alert analysis can be broken down into three main fields [6] as shown in Figure 7: *pre-processing* (e.g. alert normalization, redundancy elimination, false positive reduction), *processing* (e.g. alert correlation techniques, new attack scenario detection) and *post-processing* (e.g. alert prioritization metrics and intention recognition, prediction). For further literature

to each component in Figure 7 refer to [6]. It is noted that the boundaries of the categorization into pre-processing, processing and post-processing might become blurred since for instance a system incorporating alert correlation might feature prediction capability as a processing and only visualization as post-processing functionality.

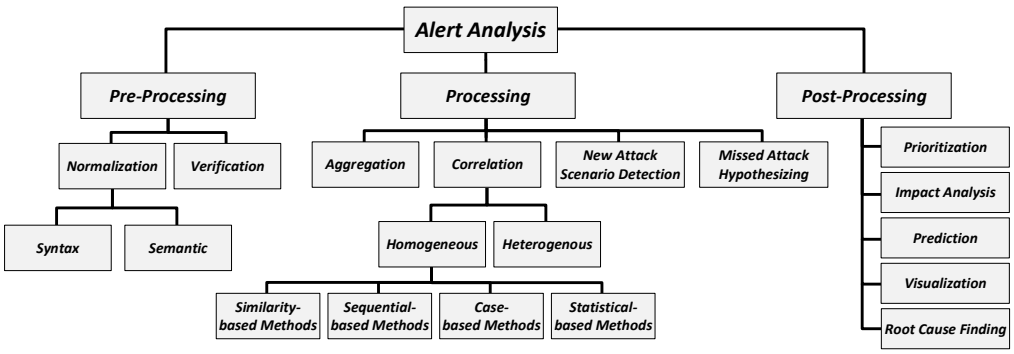

Figure 7: Taxonomy of alert analysis fields (cf. [6])

3.1 Pre-Processing

Pre-processing is the process performed before attack scenario construction. It is composed of *normalization* and *verification* which are fundamental steps before e.g. a correlation can be accomplished.

3.1.1 Normalization

Security Information and Event Management (SIEM) systems are designed to help network administrators, typically working in a SOC, to manage security tools e.g. IDSs operating in the network infrastructure. Typically, the work of SIEM systems is to aggregate, standardize and correlate alarms. Today, SIEM systems mainly use internal proprietary formats to describe alerts. Most of those are inspired of log management tools such as Splunk and based on syslog with a simple but limited key-value paradigm. However, the heterogeneity and diversity of existing security tools pose a significant challenge to SIEM and SOC due to the multitude and diversity of alert sources demanding the need for a common format. Normalization is used for the translation of a raw alert into a standardized alert format, e.g. IDMEF as proposed by [7]. With the growth of semantic technology and the inability of e.g. IDMEF only presenting a syntax for formatting, new studies try to introduce new data models for handling alerts semantically in order to provide a robust solution [6]. A structured overview of various existing

exchange formats ("describes a structure for the processing, storage, or display of data" [83]) and protocols ("a set of rules defining how to interconnect network devices and establish a channel to transmit network datagrams, representing exchange formats, across a computer network" [84, 85]) targeted to the IDS domain is provided in [83, 86]. Application domains of other exchange formats in IT security is depicted in Figure 8. For a comprehensive overview of standardization attempts for security automation refer to [4].

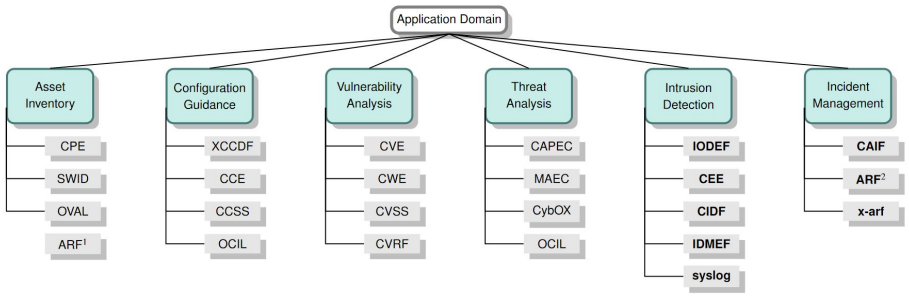

Figure 8: Application domains of exchange formats [83]

Koch et al. in [86] state several technical requirements for data formats and exchange procedures for sharing information of interest to IDSs, response systems and to management systems including vendor independence, near real-time capability and scalability e.g. for decentralized approaches. Message exchange protocols discussed are (1) proprietary protocols, (2) Simple Network Management Protocol (SNMP), (3) Common Intrusion Detection Framework (CIDF), (4) Intrusion Detection Message Exchange Format (IDMEF) including IDMEF Communication Protocol (IDP) and the newer and recommended Intrusion Detection eXchange Protocol (IDXP), (5) Incident Object Description and Exchange Format (IODEF), (6) Format for Incident Report Exchange (FINE) and (7) Intruder Detection and Isolation Protocol (IDIP). However, the authors of [83] state that Koch et al. do not differentiate between a high-level description of functional requirements, an exchange format or an exchange protocol. Thus, Steinberger et al. in [83] reviewed 10 exchange formats and 7 exchange protocols that can be used to share security event related information in context of intrusion detection and incident handling with respect to their use-case scenario. They further provided inter alia an assessment of the exchange formats for the interoperability and a qualitative evaluation and comparison of the formats and protocols in context of high-speed networks. Apart from the aforementioned formats and protocols, the authors introduce further formats (8) Common Announcement Interchange Format (CAIF), (9) Common Event Expression (CEE), (10) Messaging Abuse Reporting Format (ARF), (11) x-arf, (12) Syslog Message Format - IETF RFC 3164 and further protocols (13) Real-time Inter-network Defense (RID), (14) Extensible Messaging and Presence Protocol (XMPP), (15) CEE Log Transport Protocol (CLT), (16) Simple

Mail Transfer Protocol (SMTP) and (17) Syslog protocol. The results of the evaluation carried out by the authors regarding the discussed exchange formats and exchange protocols are depicted in Figures 9 and 10.

Criterion	CIDF	IODEF	CAIF	IDMEF	ARF	CEE	X-ARF		Syslog	
							v0.1	v0.2	RFC 3164	RFC 5425
Interoperability	−	−	−	−	+	+	+	+	+	+
Extensibility	+	+	+	+	+	+	+	+	+	+
Scalability	−	−	−	−	−	−	−	−	−	−
Aggregability	−	−	+	0	−	−	−	+	−	−
Protocol independency	−	0	+	0	+	0	+	+	+	+
Human readability	−	−	−	−	+	+	+	+	+	+
Machine readability	+	+	+	+	+	+	+	+	−	+
Integrity & Authenticity	−	−	−	−	−	−	−	+	−	−
Confidentiality	−	−	−	−	−	−	−	+	−	−
Practical application	−	0	0	0	0	−	0	0	+	+

Legend: high (+), medium (0) and low (−)

Figure 9: Evaluation summary of exchange formats [83]

Criterion	CIDF	RID	XEP-0268	IDXP	SMTP	CLT	Syslog	
							RFC 3164	RFC 5425
Confidentiality	+	+	+	+	−	+	−	+
Authenticity	+	+	+	+	−	+	−	+
Integrity	+	+	+	+	−	+	−	+
Reliable message transport	+	+	+	+	+	+	−	+
Interoperability	−	+	−	+	+	−	+	+
Scalability	+	−	+	+	+	+	+	+
Practical application	−	0	−	0	+	−	+	+

Legend: high (+), medium (0) and low (−)

Figure 10: Evaluation summary of exchange protocols [83]

Even if Steinberger et al. provided a comprehensive survey, they did not mention IDIP which seems a promising candidate for automated incident response execution and a few other recent formats e.g. the JavaScript Object Notation (JSON) serialized Intrusion Detection Extensible Alert (IDEA). According to [86], IDMEF and IDXP can have a likewise effect on research and deployment of intrusion detection technology what HTML and HTTP did for the Internet growth. However, Steinberger et al. conclude that it is still a challenge to find a standardized exchange format and protocol that is thoroughly validated and tested in full scale of industry trials [83].

A significant drawback of IDMEF is the usage of XML which makes it easier to develop and deploy, but it comes with a performance cost. Due to the structure of XML, the data encoded is typically very large (for instance in comparison to JSON), mainly because of XML's closing tags. Therefore parsing XML messages is still a relatively slow task today [86]. The Warden [87] and MISP [88] projects also bring further interesting approaches. Warden uses a client-server architecture and offers two different types of

clients: the receiving-client and the sending-client. With the sending-client events can be sent to the Warden server, with the receiving-client events received at the Warden server are also sent to the client side. The Warden team designed the IDEA format for the transmission of events. It is based on already existing data formats for the transmission of security relevant information (mainly IDMEF) and aims to eliminate weaknesses that these data formats bring to their system. For example, messages in the AbuseHelper format consist of any number of keys and associated values which makes this format easily extensible but can lead to inconsistencies in automated message processing. The IDEA format should lie between the complexity and depth of IDMEF and the loose structure of the AbuseHelper format [89].

3.1.2 Verification

Verification or alert validation is necessary since many problems can occur, such as misconfiguration, low accuracy of applied detection methods, and lack of attention to contextual information during the alerts analysis [90]. Therefore verification tries to recognize if any changes have been taken place in the system monitored since when any new device has been installed in the system it may produce irrelevant alerts. It helps to filter out alerts with low-interest, irrelevant alerts or some known false positives. Validation can also refer to post-processing. In [91] an approach is presented that defines three major dimensions to recognize attacks and identify the target by validation of alert correlation systems: prioritization (assigning weight to each alert based on the probability that it may indicate an attack and dispensation of the target), multi-step correlation (alert correlator can reconstruct a multi-step attack scenario by correlating different individual attack steps which is important to infer attack intention and their effective response) and multi-sensor correlation (combines multiple alerts received from different sensors to create an overall picture of the system) [7].

3.2 Processing

Processing mainly copes with the attack scenario construction and contains *aggregation*, *correlation*, *new attack scenario detection* and *missed attack hypothesizing*. The outcome of processing serves as a basis, for instance, to recognize the intention of an adversary which is performed in post-processing.

3.2.1 Aggregation

Aggregation or alert merging, respectively alert fusion, refers to the reduction of alerts by combining multiple (and possibly heterogeneous sources) of them to yield a more precise and descriptive result of IDSs. Alert fusion is a special case (sometimes a sub-process) of alert correlation that collects and

analyzes alerts independently generated from the same potentially malicious event by different IDSs, in order to make an appropriate final decision about the event [92]. The main idea behind aggregation is to provide clustering and grouping similar alerts based on their features in order to eradicate duplicates having the same root cause. Especially by the application of distributed or decentralized IDSs, the exploitation of alert fusion enhances the overall detection efficiency, improves detection accuracy, fault-tolerance, stability, and reliability of IDSs and helps to make appropriate decisions [5]. The process is closely related to Subsection 2.2.3 (algorithm external combination). Finally, with the aggregation of duplicate and redundant alerts, the uninteresting ones are eliminated and a big view of the security situation is provided by fusing the same events [6]. Weng et al. for instance propose an alarm reduction for distributed IDSs using edge computing in [93] exploiting the strengths of cloud computation while offloading only a limited amount of information by processing data at the edge for shorter response time and energy saving. Their proposed framework consists of three layers which are structured hierarchically from the infrastructure to the cloud side: IDS layer (performs traffic inspection and false alarm reduction by exploiting the strengths of distributed IDSs), Edge layer (aggregate data from IDS layer and select most appropriate machine learning algorithm from a predefined pool), Cloud layer (providing sufficient computation resources for deploying intelligent alarm filters).

3.2.2 Correlation

The homogeneous alert correlation refers to a case that each of the monitoring devices like IDSs examine the same type of data, whereas in the other one, various deployed sensors examine different types of events and raw data sources [5]. Salah et al. [94] and Hubballi et al. [95] provide a comprehensive overview in the field of alert correlation which is defined as a measure of the relation between multiple alarms such that new meanings can be assigned to them. Thus, not only the verification of the alerts' validity can be verified but also complex attack scenarios can be identified. The alert correlation process comprises different approaches available in the literature [95, 96, 97, 98, 99, 100] and has been classified in [7, 101, 102]. However, it must be noted that due to different types of attacks with different sophistication level, there might be limitations in the handling of the multitude of alerts with equal importance. Hence, it might not be sufficient to rely on a single component but rather on different ones to concentrate on various aspects of the general correlation problem. Several factors that can be used to assess correlation algorithms are stated in [102] composed of algorithm capability (e.g. alert verification, attack sequence detection), algorithm accuracy, algorithm computation power, required knowledge base and algorithm extendibility and flexibility.

Alert correlation techniques try to reconstruct the attack scenarios from alerts which may exhibit an attack that involves multiple stages in compromising a network [7]. A taxonomy of alert correlation techniques is provided by [94] (Figure 12) including types of application and architecture for the correlation process. In [94, 103] alert correlation architectures are categorized into *centralized* (data collection performed locally and reported as alerts to a central server executing correlation analysis), *distributed* (alerts or high-level meta-alerts are exchanged, aggregated, and correlated in a completely cooperative and distributed fashion between equally weighted agents; communication is performed using a peer-to-peer protocol) and *hierarchical* (referred to as decentralized by the author of this work; separated correlation into hierarchical layers of local analysis, regional analysis and global analysis) and shown in Figure 11.

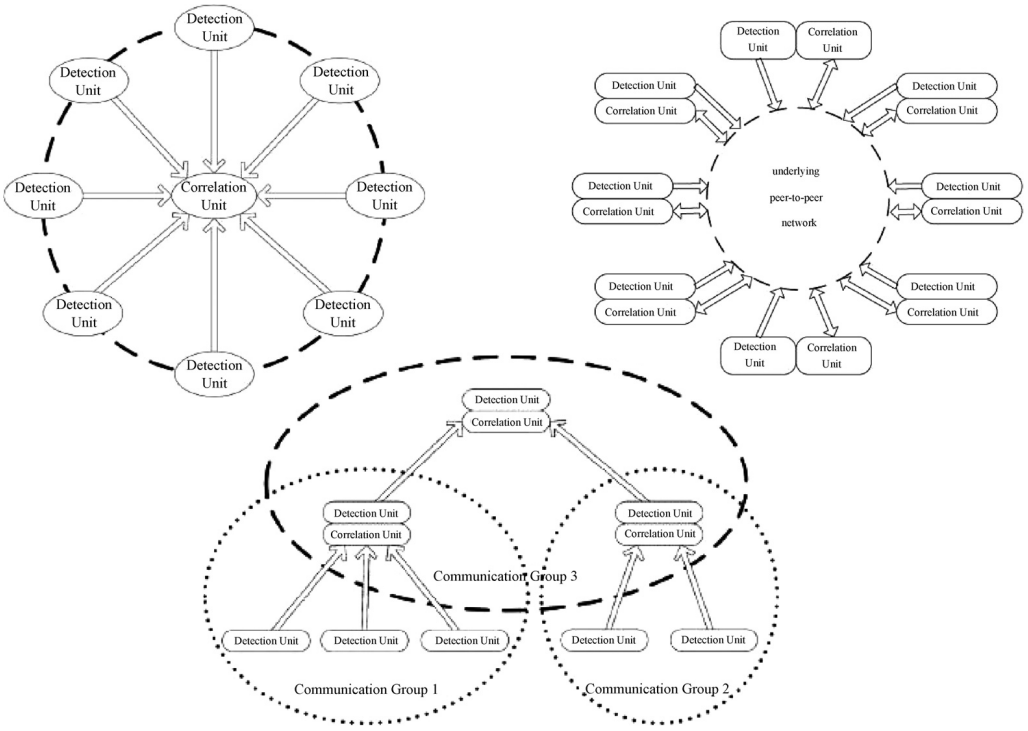

Figure 11: Different alert correlation architectures - centralized (left), distributed (right) and hierarchical (bottom) [103]

The number of data sources - single or multiple - with respect to Figure 12 state that the alert correlation method is sourced either by a single data source, e.g. a database or a single security measure, or by a collaborative set which allows a more precise and coherent view about the observed system. The authors further subdivide the correlation methods into *similarity-based*, *sequential-based* and *case-based methods*, whereas the authors of [7, 102] introduce apart from similarity-based and case-based

(referred to as *knowledge-based*) *statistical-based* methods and *hybrid* approaches (both not shown in Figure 12). However, it must be noted that the categorization is not completely precise and methods from each class may show similar behavior or rely on comparable mechanisms.

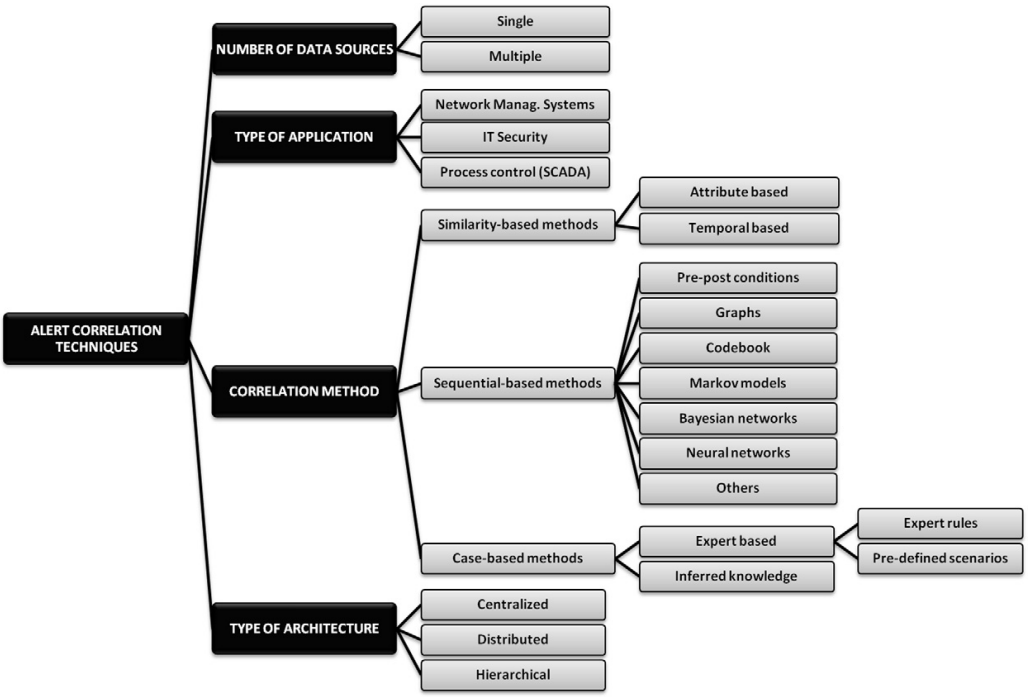

Figure 12: Taxonomy of alert correlation techniques [94]

Similarity-based methods correlate alerts based on similarities of selected features such as source/destination IP address, time or protocol information and are designed to reduce the total number of alarms through aggregation or clustering [94]. Therefore, they can be further subdivided into attribute-based (similarities between attributes/features) and temporal-based (temporal time relations) techniques. Mirheidari et al. categorize similarity-based algorithms into ones that are based on simple rules, hierarchical rules or machine learning [102]. Usually, similarity functions are defined for the individual alarm attributes and applied to two alarms. Together with appropriate attribute weightings, a similarity value is determined that reflects how well two alarms match. These values are useful because it can be assumed that alarms with high similarity can be part of the same attack or suspicious event. The resulting similarity between two alerts can be calculated according to Equation 6 in which X^i is the candidate meta alert i for matching; Y is the new alert; j is the index over the alert features; E_j is the expectation of similarity for feature j; and X_j as well as Y_j are the values for feature j in alerts X and Y [7]. Further, attribute-based similarity measures can be computed using metrics such as Euclidean, Mahalanobis,

Minkowski and/or Manhatten distance functions [94]. Temporal-based methods, in order to find temporal relationships between alerts, typically rely on time-windows such that only alerts observed in a short time are to be correlated. A benefit is to reduce the number of alerts generated by the same event in a certain period of time [94].

$$Sim(X^i, Y) = \frac{\sum_j E_j Sim(X^i_j, Y_j)}{\sum_j E_j} \tag{6}$$

Valdes and Skinner presented a probabilistic approach in [104] that falls into the area of similarity-based methods which extends the idea of multi-sensor data fusion for alert correlation. This method shall find its use in the handling of alarms generated by heterogeneous sensors. The correlation algorithm expects features from reported alarms in a self-defined alert template. For comparable features, suitable similarity functions are defined, whereby the features of incoming alarms will be compared with a list of already existing meta-alerts and result in values between 0 (mismatch) and 1 (exact match). The similarity value is composed of a weighted average of the features, but if one does not exceed the minimum similarity threshold, the complete alarm is not considered similar. The alarm is correlated with the most similar meta-alert, otherwise the alarm forms a new meta-alert thread. The alert fusion considers feature overlap (new and existing alerts may share some common features), feature similarity (value of similarity scores of same type of feature), minimum similarity and the expectation of similarity. Since sensors can classify attacks differently, a matrix of similarities between attack classes is used to compare them. The correlator checks whether the sensor identification and the incident class match exactly. Then it checks if all overlapping features at least match the minimal similarity and calculates the similarity values. If this is the case, the overall similarity is calculated.

Zhuang et al. [105] rely on the work of [104] and extend it with a rule-based knowledge base. A correlation system architecture consisting of alert collection, alert verification, data fusion and correlation is proposed and explained. The correlation process takes into account the features IP addresses, port numbers and time stamps. Alarms from different sources are collected by the alert collection module and the features are passed on to the alert verification. In order to support heterogeneous sensors, different plugins are used which process the respective alarms, e.g. Snort alarm $\rightarrow$ Snort plugin). The alarm information is additionally appended with information about the plugin that processed it. The transmitted alarms are checked for false positives by the alert verification based on information regarding the network topology as well as the hosts and a confidence value is determined. Legitimate alarms are then grouped by the data fusion component using similarity functions. Using the knowledge base, the last step in the correlation process is to classify the alarms into an attack scenario, which is also referred to as

a schema. A schema consists of a number of rules which use the information from the previous steps such as the confidence value to describe the state of the monitored system. The description of a schema by the rules resembles a tree structure and attacks are detected if the rules of a schema are met from root to leaf. Similarity-based methods prove to be suitable for alert clustering and reducing the number of alarms as well as discovering simple attacks with a small number of features. In addition, they are easy to implement and work well for a known set of alerts with a known feature set, but find their weakness in recognizing causal and other statistical relationships between alarms, limited to known alerts only and incapable of identifying complicated attacks [7, 94, 103].

Statistical-based methods, according to [5, 7, 102], rely on statistical causality analysis to correlate alerts that are related to some specific attacks in order to reconstruct attack scenarios. Similar attacks have similar statistical attributes, and so, they can be categorized easily corresponding to different attack stages. Statistical computation can be categorized into (1) detection of repeated and repetition patterns; (2) estimation of causal relationships between alerts, predicting next alert occurrence, and detecting attacks; and (3) combining reliability by mixing completely similar alerts. Since these methods are based on statistics, pre-defined knowledge about attacks scenarios is not required. However, they lack in discovering dependencies, structural cause relationships between alerts and it is difficult to estimate correlation parameters. Exemplary work using statistical-based methods is provided in [100, 106].

Sequential-based methods attempt to determine causal relationships between alarms using defined *preconditions* and *consequences*. This is done by describing events or states, which are necessary for an attack step, as preconditions and describing the respective effects or states, which result from successful execution of this attack step, as consequences. Thus, they are not limited to known attacks but the correlation may result in many false correlations due to the misconfiguration of the relationships mainly represented as logical operators such as AND/OR or the inadequate quality of sensors. According to [94], they can further be subdivided into e.g. pre/post conditions (using the concept of hyper-/meta-alerts as a tuple of prerequisites and consequences), graphs (directed acyclic graphs in which the set of nodes represent alarms and the edges represent the temporal relation), codebook (matrix representation of alerts (rows) and problem symptoms as columns), Markov models (stochastic model composed of discrete states and a matrix of state transition probabilities trained by sequence of events), Bayesian networks (probabilistic directed acyclic graphical model of alerts representing their probabilistic inference), Neural Networks (a collection of connected units or nodes called neurons working interconnected to perform alert correlation).

Ning et al. [107] present a possibility to reconstruct attack scenarios based on prerequisites and consequences by correlating alarms. For the representation of prerequisites and consequences, the use of predicates is suggested which can be linked by logical combinations if necessary. The prerequisites and consequences of an event are represented by so called *hyper alerts* which encode the knowledge about an attack. A hyper alert consists of three components: fact, prerequisite and consequence. Fact states what information is reported with the alert. This consists of a set of attributes, each with its own range of possible values. Prerequisite is a logical combination of predicates whose variables occur in fact and specify which criteria must be met for an attack to be successful. Consequence describes the effects if the attack is actually successful. A correlation occurs when the consequences of one hyper alert fulfill the prerequisites of another one. This is also referred to intuitively as a "prepares-for" relationship, since two hyper alerts h_1 and h_2 are correlated if h_1 is prepared for the following hyper alert h_2. By describing a hyper alert type, hyper alert instances are created when events occur that are described in the prerequisites of the hyper alert types. From the prepare-for relationship and thus the relationships between hyper alerts, the authors create an Alert Correlation Graph (ACG) based on hyper alerts to represent attack scenarios step by step. The ACGs consist of a number of nodes (hyper alerts) and edges, which represent the connection between nodes and thus the relation. The result is a directed acyclic graph that corresponds to the detected scenario. Zhu and Ghorbani also use the concept of hyper alerts to determine attack scenarios in [108]. Their correlation engine is based on neural networks using Multilayer Perceptrons (MLP) and SVM. MLP and SVM learn the desired behavior with one training set, using a total of 6 features. The networks are used to decide whether two alarms should be correlated and, if so, provide a correlation value between 0 and 1. Determined correlation values between two alarms are stored in an alert correlation matrix and later updated by the correlation engine. In addition, alarms with the best matching alarms are grouped into hyper-alerts using thresholds. Based on this, graphs are created to show the attacker's approach.

The authors of [109] present an approach to reconstruct attack scenarios from alarms coming from heterogeneous sensors. The process is divided into two steps: semantic-based alert clustering and causality-based attack analysis. For the semantic analysis of alarms, an ontology is introduced using classes and explicitly defined relationships that contain relevant information regarding the intrusion detection environment. The emerging relationships between intrusion alerts can be used to determine how relevant semantic alarms are. Semantically related alarms are converted into ACGs. The ACG is a non-directed weighted graph with nodes representing alarms and edges representing the relationships between alarms based on the previously determined semantic relevance with numerical values ([0,1]). Groups of nodes are determined from the created ACG, whereby all nodes of a group are

connected to each other by edges. In graph theory such groups are called cliques, here cliques are regarded as candidate attack scenarios. Based on the time stamp information from the alarms, the sequence of the individual steps of the attack is determined. If the dependencies between preconditions and consequences are clearly defined, sequential-based methods are well suited for detecting known but also unknown attack scenarios allowing to detect zero-day and multi-step attacks. However, the weak point here is that individual steps of an attack can be overlooked by the applied IDSs which would not lead to the fulfillment of a consequence (refer also to hypothesizing). Furthermore, in a network with heterogeneous sensors describing an attack differently but with the same meaning, the alarms must be described for each sensor. A combined approach with a similarity-based method for clustering and normalizing alarms would be well suited here.

Case-/Knowledge-based methods, also referred to as alert correlation based on known scenarios, usually rely on well-described attack definitions in a knowledge base. The knowledge can be based on either prerequisites and consequences, attack scenarios or case-based reasoning which is defined as the process of solving new problems based on the solutions of similar past problems [102, 110]. Those are typically described by rules such as in [111] or a correlation language such as LAMBDA [112], STATL [113] or CAML [114]. Methods from correlation languages, from data mining or machine learning search the knowledge base for the best fitting case and update it if the case is successfully solved. According to [94], examples for case matching algorithms are nearest neighbor, inductive, and knowledge-based indexing and case-based methods can be further subdivided into expert-based (knowledge base is build by human using expert rules or predefined scenarios) and inferred knowledge (symbolic classification rules are automatically constructed from some training cases - alerts or meta-alerts whose classification is known - by machine learning). According to [7], the main drawbacks of these methods are the manual definition of prerequisites, the limitation to deal with new pattern, the difficulty in updating the correlation knowledge, the inability to discover structure and statistical relationships and their impracticability for the use in large scale or real time environments due high computational expense.

To exploit the benefits of the different techniques, hybrid approaches are often proposed. Ahmadinejad et al. [115], for instance, present a model consisting of two modules for alarm correlation. Received alerts are passed to the first module to check if it can be placed in an already known attack scenario. The analysis is done using attack graphs, here called “queue graph”, by checking for incoming alarms whether a prepares-for relationship can be identified with already existing alarms. Using a depth search in the queue graph, alarms or steps of a known attack can be found that have been overlooked by the IDS. A threshold value is used to decide whether this belongs to a known attack scenario with missing detected steps or whether a po-

tentially unknown scenario exists. Alarms that cannot be classified in the queue graph by the first module are forwarded to the second module of the model for similarity-based analysis. Selected features are taken into account and a similarity vector with values [0,1] is formed on the basis of similarity functions. Based on the similarity vector and the existing hyper alert, a "CorrelationThreshold" is used to decide with which hyper alert the new alarm should be correlated. The authors of [116] propose a hybrid approach that is based on hierarchical clustering composed of an offline correlator (aggregates historical data, extracts attack strategy graphs and uses hierarchical clustering to group similar attack strategy graphs – the attack characteristics of each cluster is then identified) and an online correlator (generates hyper-alerts which contain useful attributes for security analysts, hyper-alerts are composed of different low-level alerts and are updated in real-time as the upcoming low-level alerts are triggered, hyper-alerts are associated to the clusters generated by the offline correlator in order to understand the characteristic of an attack). Since the approach correlates historical alerts into clusters using data mining techniques and associates upcoming alerts to these clusters in real time, an efficient security alert analysis technique could be achieved and useful information from historical data can be discovered to assist the analysis of new alerts that reduces the time between the detection and response to an incident.

3.2.3 New Attack Scenario Detection

New attack strategy detection copes with the discovery of novel attack scenarios from the sequence of events and tries to overcome limitations of correlation methods that are typically unable to extract unknown malicious behavior of intruders [117]. This includes finding new multi-step attack scenarios from the analysis of alerts to which no "template" is available. Simple changes to those templates (or attack patterns) might result in a failure of attack detection. Especially, with the application of anomaly-based IDSs which output either simple classification values - anomaly or not - or scoring values representing only indicators of malicious activity it is more difficult to analyze alerts than with misuse-based ones. Those are able to interpret events/indicators due to their existing knowledge base and provide more detailed information. Soleimani and Ghorbani present an approach in [118] that aggregates alerts and generates episodes which represent a sequence or a partially ordered collection of events. After a learning phase that includes learning real multi-step attacks, the framework is able to either filter critical episodes predicting future steps of attacks or to filter uncritical episodes which might correspond to new attack strategies. A three-phase alert correlation framework called 3PAC is proposed by Ramaki and Rasoolzadegan in [119] which processes real-time alerts, correlates them utilizing causal knowledge discovery, constructs attack scenarios via the Bayesian network

concept and is able to predict next attack steps. The authors state that Bayesian inference model in conjunction with statistical data mining techniques has several merits for alert analysis e.g. processing speed, incorporating expert knowledge, computation of a correlation output probability instead of a binary result. New attack strategies are extracted from the attack tree construction phase based on classified benign episodes. Those serve as a basis for the presented causal knowledge analysis algorithm to identify critical new episodes. ZePro [120] by Sun et al. is an approach targeted towards zero-day attack path identification. The approach assumes that a chain of attack actions, typically composed of both zero-day and non-zero-day exploits, forming an attack path is necessary by adversaries to achieve their malicious goal. ZePro builds a comprehensive network-wide graph based upon system calls from which it effectively and automatically identifies zero-day attacks.

3.2.4 Missed Attack Hypothesizing

Missed attack hypothesizing is dealing with the problem of false negatives that might occur with IDSs which does not lead to the generation of an alert. Alert analysis, especially considering multi-step attack detection relying on each alert, must be able to cope with the missing ones. However, this functionality poses still a major problem in the field of alert analysis. According to [6], hypothesizing can be categorized into approaches with or without any predefined knowledge about attacks. The former require knowledge e.g. in form of attack templates to compare the received attack types with existing attack patterns. Exemplary work is provided in [121, 122]. The more interesting and innovative hypothesizing approach uses data-mining techniques that can generate some artificial clusters that represent attack classes which are then validated whether the quality of the cluster is higher than a certain threshold representing a missed attack. According to [123], three main steps are necessary for the hypothesis process: cluster generation, cluster validation, and cluster tuning. Fatma and Limam present a two-staged approach in [124] that firstly deals with false positive alerts aggregated from multiple IDSs and secondly tries to identify potential false negatives representing missed attacks. Therefore, the first stage clusters IDS alerts into a set of meta-alerts based on several attributes and identifies false positives using binary optimization. The second stage discards meta-alerts that have been created by the majority of IDSs. Remaining alerts are grouped and yield the set of potential false negatives from which via a binary classification algorithm the set of false negatives is identified.

3.3 Post-Processing

Post-processing is performed after alert correlation, in particular attack scenario construction, and is composed of *intention recognition*, *(propagation)*

prediction, *alert prioritization*, *impact analysis* and *visualization* (presentation, representation and visualization of detected security events or corresponding attack scenarios in a SIEM and/or to a SOC).

3.3.1 Intention Recognition

Intention recognition is mainly a forensics topic and deals with the derivation of an adversaries' intention (typically unpredictable) caused by a malicious activity by analyzing the alert analysis results. It deals with the interpretation and judgment of the purpose, vision and intention of attackers according to their behavior and network environment by analyzing the alert information. However, in a broader spectrum, it tries to give a reasonable explanation of the real purpose of malicious activity and predicts the subsequent attack steps which is, according to [125], the premise and foundation of threat analysis and the important part of network security situation awareness. Malicious activity in this work is either referred to as (self-propagating) malware as it is mainly discussed in the literature and represented by Worm/Virus software, by a human-controlled or an AI-driven attack. However, with the advent of artificial intelligence for cyber attacks the boundaries blur which might makes it difficult to strictly distinguish between them. Malicious activity may contain parts of the attack phases (sometimes referred to as kill chain) itemized and summarized below. The main goal of malicious activity is, however, to infect/compromise victim systems even if the motivation/intention behind may differ.

- **Planning and Discovery:** Social Engineering, Permission/Authorization, Information Gathering, Scoping and Exploration, Service Identification, Scanning and Fingerprinting
- **Exploitation and Assessment:** Vulnerability Identification, Vulnerability Assessment, Enumeration, Gaining Access, Exploitation (External vs. Internal)
- **Post-Exploitation and Reporting:** Discovery and Forensics, Finding Analysis, Data Collection, Maintaining Access, Covering/Cleaning Tracks, Placing Rootkits/Backdoors, (Network) Spreading, Privilege Escalation, Reporting

A kill chain describes the ability to disrupt the sequence of events an attacker must perform in order to achieve success during an attack. Thus, it breaks down attacks into levels which represent the structure and procedure of an attack. For each level, the model indicates which activities attackers undertake, so that one can set up the defense including reaction mechanisms accordingly. Although the cyber kill chain is already several years old, it can not only be applied to classic malware attacks. APTs can also be mapped to it and broken down into steps. While the attacker has to go through the complete process to reach the target according to the model, the defense

can try at any level to interrupt the kill chain and thus stop the attack. However, it is necessary to build up the defense in several levels, because at each level the attacker can already do damage. The earlier the attack can be detected and stopped, the less damage can be expected. Several existing kill chain models are discussed and a new kill chain model is proposed in [126] for remote security log analysis with SIEM software.

Malware are meant to be programs that self-propagate across a network exploiting security flaws having the ability to propagate from host/network to host/network. The program first explores vulnerabilities in the network by utilizing various discovery techniques and infects entities by exploiting them. The infected entity serves then to spread itself automatically or through human triggering. Typical properties are the fast and various different automated scanning techniques, the functional range of either targeted to one or more (limited) number of vulnerabilities to exploit and spread as well as the aim of infecting as much vulnerable victims as possible (dedicated for large-scale networks). However, if vulnerabilities are once fixed, the certain type of self-propagating malware is no threat anymore. In contrast to an automated malware, the human-controlled attack performing the above listed attack phases has different properties. Scanning, especially in the sense of APTs, could take more time when performed stealthy. They have a large number of tools at their disposal with a broad spectrum of attack vectors to gain access. If a vulnerability has been fixed, it might exist various others such that the host can be reinfected again. Further, the latent time between the initial possible low-privileged access to gain system privileges which might be necessary to further reach other networks (e.g. pivoting on dual-homed machines or gateways) can be longer. Hacker might adapt the exploits for gaining access or privilege escalation which often results in a crash of a system depending whether the attacker has not tried the exploit in advance. This could translate into a slightly higher death rate of machines compared to the malware case. The goal differs from malware since maybe only certain targets are tried to be compromised, e.g. high value administrating machines, or all machines should be compromised depending on the initiator of the malicious activity (script kiddie, expert hacker, hacker organization) with their motivation in mind. However, from the information gained from analysis it might be possible to draw conclusions on the initiator. With a dedicated motivation in mind the human-controlled attack is more targeted for small-scale networks e.g. inside a companies industrial site. Hui and Kun proposed a dynamic real-time network attack intention recognition method based on attack route graphs [125]. By correlating real-time network attacks and vulnerabilities in their framework, shown in Figure 13, the method determines spread routes and stages of an attack based on graph theory and probability theory. They are then able to dynamically reason the possible intrusion intention and its probability according to the attack behavior characteristics and the network environment.

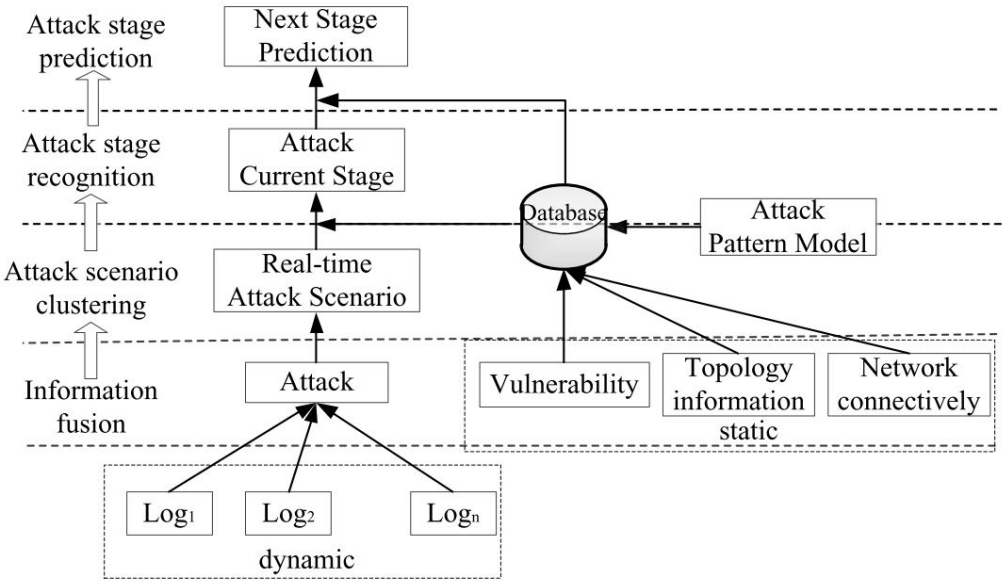

Figure 13: Framework of real-time network attack intention recognition [125]

3.3.2 Prediction

Intention recognition is closely related to prediction (or attack forecasting, attack projection) which tries to predict the next step of malicious activity during an ongoing (multi-step) attack with reference to the killchain model and how likely a next step will occur. Abdlhamed et al. categorize the prediction of cyber-attacks into three major methodologies alerts correlation, system call sequences as well as statistical methods and discuss prediction methods including hidden Markov models, Bayesian networks and genetic algorithms [127]. They state that the type of prediction is an important factor, for instance, solutions dedicated to predict attacks usually use hidden Markov models, while devoted for forecasting the intentions or the abnormal events mostly exploit the Bayesian networks. Anumol in [128] proposes an IPS which performs event analysis and predicts future probable multi-step attacks using a SVM based on network log files collected in the OSSIM SIEM solution. Features consisting of e.g. number of packets, number of bytes or packet rate are subject to formulas for information gain in order to get the best features with maximum gain ratio. However, the information provided by the article is quite sparse and its title therefore misleading in terms of predicting multi-step attacks based on raised alerts. As a proactive approach targeting to prevent attackers from reaching their malicious goals, Ramaki and Atani provide a survey of early warning systems which act beyond the scope of IDSs or IPSs in [129]. In order to proactively counteract new emerging high sophisticated threats, the aim of early warning systems complementary to intrusion detection/response is to detect poten-

tial malicious behavior in a system, evaluating its scope and implementing appropriate response mechanism as early as possible (Figure 14). According to [130], the term early has two different meanings: (1) starting on-time to prevent or minimize the damage or damages and (2) the ability to process incomplete information. A discussion of characteristics (e.g. functionality, detection scope, challenges, data collection), architectures (centralized, hierarchical, distributed) and a comprehensive overview of existing early warning systems as well as those under research and development is provided by [129]. Further work of Ramaki et al. [131, 119] intensively deals with real-time alert correlation and prediction using e.g. Bayesian networks.

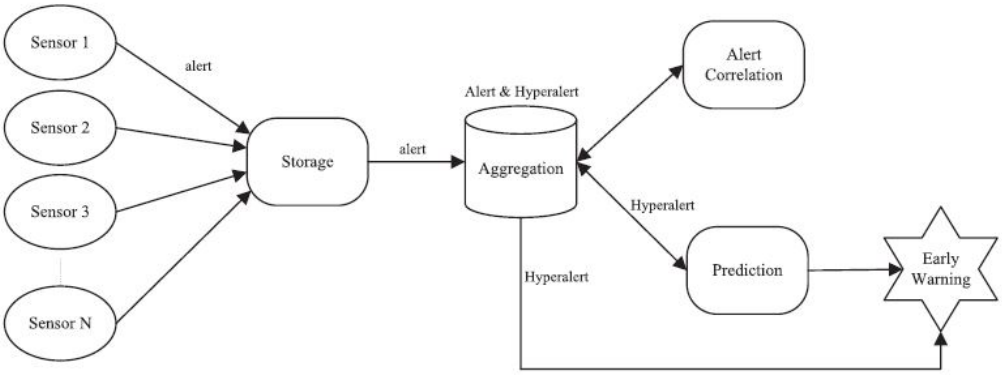

Figure 14: Operation of a generic early warning system [129]

In [101] Ghasemigol et al. present an attack forecasting approach in form of an extended attack graph which is able to predict future network attacks based on information such as intrusion alerts, active responses, and service dependencies. Therefore, they combine the information acquired from an uncertainty-aware attack graph, a hyper-alerts graph, a multi-level response graph and a dependency graph into their proposed forecasting attack graph which increases its accuracy for predicting future attacks based on the additional information. The problem of a real-time multi-step attack prediction has been targeted by Holgado et al. in [132]. In this work, IDSs send their alerts using IDMEF to the system which turns them into hidden Markov model capable observations utilizing a clusterization process that incorporates a tag (inferred by matching the significant words in the alert description with the words occurrence frequency in the CVE reports) and a severity (corresponding to the alerts severity parameter). The hidden Markov model states represent the chain of different attack steps. By computing the mean number of alerts for each state, a final attack probability can be computed. The training of the hidden Markov model is performed by applying a supervised and an unsupervised (Baum-Welch) algorithm. An evaluation is performed using the LLDDOS1.0 attack scenario of the DARPA dataset and the alerts provided by a Snort IDS. The authors of [133] are modelling

vehicle states, e.g. door open/closed, vehicle moving/stationary, using a hidden Markov model extracted from the CAN network traffic. Considering the movement of a vehicle as a sequence of states that depend on the previous ones, the model can be derived and anomaly detection is performed within a sliding window of n previous observations that slide over the various states. If the posterior probability is 0 or less than a defined threshold, an alert is generated indicating an anomaly. In each case an observation would be a vector of different sensor values that generates a set of probabilities corresponding to each observation. The proposed approach seems quite promising for detecting anomalies in time series data. However, it could be seen as a utility for further appliances to work beside low-level detection mechanisms on a higher level of abstraction for cross-evaluation. Thus, for instance if the car is driving with an implausible speed and the applied IDS sensors detect a huge number of alerts, the security state of the vehicle is critical and immediate interaction by the driver needs to be taken. The prediction system of Abdlhamed et al. in [127] incorporates multiple sources of information and different methodologies. There are two modes of operation: if there are little security incidents, prediction is done by using statistical methods; if the information is sufficient, the system builds attack scenarios and constructs profiles for suspected users. A risk assessment is dynamically calculated when there is abnormal behavior affecting the system. The prediction is produced when the profiled user actions and the dynamic risk assessment indicates specific stages on the security master plan. However, the system predicts attacks over multiple days as shown in the evaluation which does not satisfy real-time requirements.

Apart from the selection of aforementioned prediction approaches, malware propagation (prediction) models can be exploited to forecast the further network spreading of malicious activity. Not only attacks controlled by humans but also unassisted malware gains more sophistication. The classical separation of malware in the categories with their different flavours, e.g. fuzzers, backdoors, denial-of-services, exploits, shellcode execution, worms, viruses, trojans, is not possible anymore since new malware show various characteristics or functionality of other. Therefore classical Worms can show behavior/characteristics/functionality (e.g. stealthiness, polymorphism, context awareness) of other malware and turn into self-spreading programs. Their propagation characterized by random effects can be modeled using a stochastic process. Those effects can be malware-related (e.g. scanning strategy), network-related (e.g. bandwidth, topology), system-related (e.g. vulnerable hosts), policy-related (e.g. intrusion prevention) or human-related (e.g. removal, patching, isolating, restoring) and leads to an overall complexity. Especially when considering the human-related portion which mainly affects the response/reaction to malware, their actions could be performed (semi-)automated. By understanding the behavior (propagation) of malicious activity which inside an IT-network could help incident management

to be a step ahead. The model of malicious propagation based on information gained from detection and analysis components could help to individuate and describe symptoms of malicious activity such that useful data could be provided to trigger emergency responses or the implementation of automatic reactions. Mathematical models in general can be categorized into three different characteristics [134]: (1) deterministic or stochastic, (2) continuous or discrete, and (3) global or individual. Those models depend on whether the variables (and parameters) are random or not, if the variables take an infinite or finite number of values and aims either to simulate the behavior of a complex system providing the global evolution or, in contrast, only focuses on the dynamics of individual nodes. Malware propagation models are based on those initially developed for the spreading of infectious diseases. The epidemiological models are compartmental, that is, the population (through which the infectious disease is propagated) is divided into different types of behavior bearing in mind the characteristics of the disease: susceptible, exposed (with or without symptoms), infectious, recovered, quarantined, vaccinated, isolated, and so on [134]. Possible states and interactions of the compartment models, on which e.g. Susceptible-Infected (SI), Susceptible-Infected-Recovered (SIR), or Susceptible-Exposed-Infectious-Recovered (SEIR) are based, are shown in Figure 15.

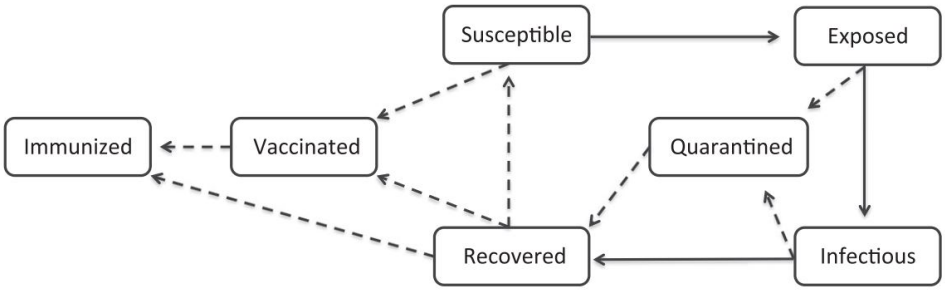

Figure 15: Different types of behavior in compartmental models [134]

Epidemic models in general can be classified into *deterministic* ones e.g. [135], *stochastic* ones e.g. [136] which typically refer to global models representing the dynamic of the overall population without taking into account the local interactions between individuals and, according to [137], *spatial-temporal* ones considering individual-based models e.g. cellular automata [138] or agent-based [139]. Deterministic models are usually based on Ordinary Differential Equations (ODE) or difference equations and perform better in large-scale networks. Stochastic models can be subdivided into three types differing in their underlying assumptions regarding time and state variables: Markov chains (discrete time with discrete state variable), Markov chains (continuous time with discrete state variable) and Stochastic Differential Equations (SDE) with continuous time and discrete state

variables. Stochastic models show their strength considering small-scale networks. However, both global models offer a good insight in the dynamics of networks providing characteristics e.g. stability or equilibrium [140] but do not consider individualized infections because the global parameters are fixed. Thus, changes of individual computers by e.g. updated anti-virus software are not taken into account. Furthermore, global models assume a homogeneous distribution of malware in a network meaning that each system is connected to another. This is true considering large-scale networks e.g. malware spreading in the Internet but in microscopic environments e.g. local networks the results are less reliable since the dynamics strongly depend on individual interactions. Individual-based systems overcome those drawbacks by including other information e.g. update policy, applied operating systems and the topology by e.g. utilizing directed or undirected graphs, but demand a high computational cost when considering large-scale networks or require a comprehensive gathering of information from the system components. A comprehensive introduction in the field of graph theory, network topologies, community structures and diffusion models with regard to malicious attack propagation is given by [141].

A selection of research activities regarding the use of SDE for malware propagation is presented in the following. The authors of [142] provide a survey and comparison of worm propagation models by categorizing them into scan-based (propagation without dependence on topology possible by various scanning techniques) and topology-based (spreading through topological neighbors)) worms based on their characteristics. The authors provide information about propagation models on these categories but, however, they do not consider stochastic models e.g. in form of SDE. In the first part of the dissertation [143], stochastic epidemic models and inference for the propagation of computer viruses are studied. A comprehensive literature review on deterministic and stochastic models is provided with a focus on Markovian- and SDE-based SEIR models. With respect to SDE, the author developed a new model for multi-group stochastic SEIR. It is stated that although ODEs can be safely used to approximate a stochastic process when the population size is large, no probabilistic event is considered. Moreover, the ODEs only describe the average tendency of virus propagation. Thus, deterministic models cannot represent rare events such as saturation and extinction of malicious activity. The two main findings of the work in [143] considering the multi-group based SDE approach is that a SEIR framework including a latent period is superior to other models and the impact of the network structure can be explored via multi-group variants with different parameterization within subnets/subgroups clustered of individuals which differ substantially in communication activity profiles and in their purposes. For future work, the authors propose to consider the integration of human countermeasures in the model since appropriately including the effects of such countermeasures can substantially further improve such a models's predic-

tive ability and the impact of the malicious activity propagation. Another point is to consider the infection rate which is normally constant as a function of time. This can especially be the case when an adversary is invading deeper inside a network trying to reach a high value target such as a domain admin machine. The author also states the investigation of the effects of the network topology on computer malware propagation must be included in further work for instance by tying together the ideas and tools of random graph dynamics to describe the stochastic behavior of the topological structure of large computer network.

In [136], the authors build on a stochastic worm propagation model based on SDEs modeling random effects during worm spreading. Derived from the paper, the essential SDE for the modeling of infected hosts is given by Equation 7 in Itô notation in which $I(t)$ is the function representing the infected hosts. N are the unique hosts in a network scanned by the worm where N_S is the number that could potentially become infected. $< \beta >$ is the mean of the total infection rate incorporating randomness including the worm's decisions e.g. scanning strategy, scan rate or changes in the network environment e.g. bandwidth, congestion. Using the Euler-Maruyama method, Figure 16 shows 10 plotted paths in red including the computed mean (500 simulated paths) for the infection function $I(t)$ of Equation 7 where $N = 254$, $N_s = 100$, $< \beta >= 1.4$ and $I_0 = 5$.

$$dI(t) = \frac{< \beta >}{N}(N_S - I(t)) \cdot I(t) \cdot dt + \frac{1}{N}(N_S - I(t)) \cdot I(t) \cdot dB(t) \qquad (7)$$

The authors state that the size of the network, small-scale or large-scale, e.g. Internet, needs to be considered since a small network size reduces the time for early detection but increases false alarms because large-scale networks will diffuse the network heterogeneity's and better describe the phenomenon. In order to counteract this, the authors provide a theoretical estimation of a critical network size which is sufficient to be monitored. For network monitoring and intrusion detection this information for critical size in subnetworks can be useful. Thus, worm projection basically involves collecting data and then estimating the infection rate and expected damage caused by the worm. Early projection results, paired with a well-established early warning policy, may lead to robust response strategies against fast-spreading, unknown worms. Since, in an unknown network the size could either be far smaller, far greater or close to the critical size, a hierarchical distributed early warning system is proposed by the authors. Each network domain k monitors n subnetworks with variable sizes and internal characteristics (e.g. bandwidth, topology). An early detection component which is able to detect the presence of a fast-spreading worm and is able to define the worm propagation model parameters. Each domain is locally monitored by one Local Monitoring Center (LMC). A local agent runs in each LMC

and is programmed to act as a communication interface between the LMC and the root of the hierarchy, namely a Global Monitoring Center (GMC). The operation of the local agent is practically the basic requirement in order to participate in the warning system. Finally, the GMC receives infection information from the LMCs and sends back warning information for an emergency response. The LMCs could, as a response measure, adapt their network or host-level firewall policies, automatic quarantine policies or disconnect/isolate particular hosts or services. The authors state that an early warning system is meant to complement current systems and they suggest the deployment of an anomaly-based IDS that will collect preliminary data from default locations, analyzes it and makes a decision on whether the examined traffic contains potential scanning worm behavior.

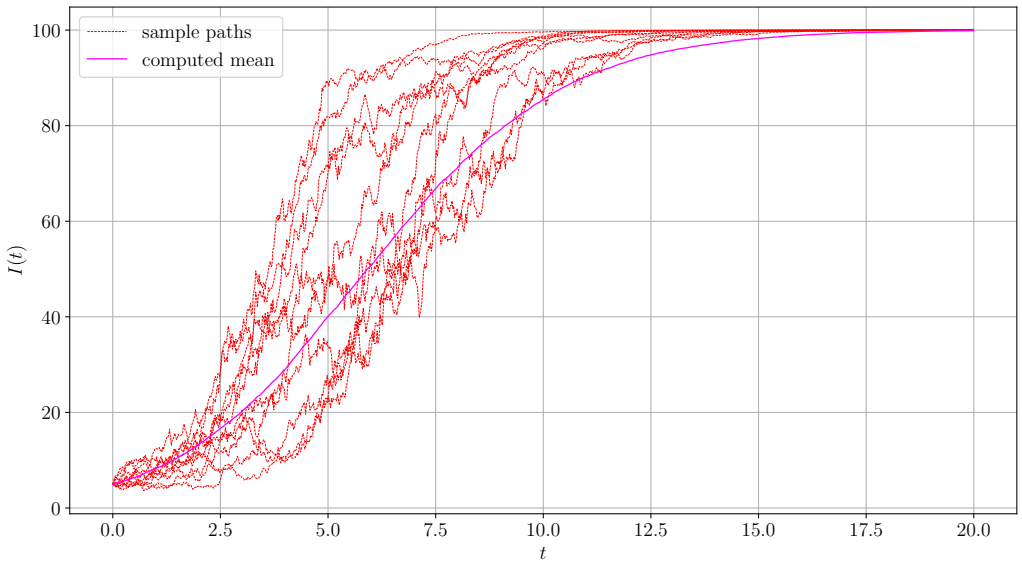

Figure 16: Simulation of an SDE for malware propagation (cf. [136])

3.3.3 Impact Analysis

The characteristics of the malicious activities' impact may also be different (impact analysis) apart from the stated above. For instance, a malware exploiting a certain discovered vulnerability that effects all models of a series (either a certain vehicle [automotive] or Programmable Logic Controller (PLC) [industrial] type) has a more devastating impact on the whole vehicle fleet [automotive] or various companies applying the PLC type [industrial] than a single hacked/compromised vehicle [automotive] or compromised industrial plant [industrial] by an adversary. Along with this, the topology of single compromisable entities and reachability with others, e.g. Electronic Control Units (ECU) inside a vehicle and their connection between them vs. the vehicles of a whole fleet and their communication among them and their infrastructure, plays a major role and has to be considered. Apart from

the close relation of impact analysis with topics such as incident response and forensics analysis, the most inferred is threat and risk assessment. Its main goal is to identify the consequences of malicious activity which helps administrators to find out the implicit and explicit relations between the attacks and the assets of an organization using information sources such as asset database, topology database, and vulnerability database for analyzing the impacts [144]. Risk management describes the process of the consideration of potential risks in a certain domain. It is further composed of risk assessment and risk mitigation. The former is the process of determining, analyzing, and interpreting the risk analysis results, and risk mitigation is the process of selecting and implementing security controls to reduce risks to an acceptable level [101]. Especially in the automotive domain, the Hazard and Risk Analysis (HARA) targeted for safety as well as the Threat and Risk Analysis (TARA) targeted for security are widespread. Network risk assessment approaches can be categorized into network risk assessment based on (1) attack graphs, (2) dependency graphs and (3) on non-graph approaches e.g. hidden Markov models or fuzzy logic [145]. An overview and discussion of available literature on these categories can be found in [101].

3.3.4 Prioritization

Prioritization focuses on categorizing and ranking alerts based on their importance since generated alerts by security components do, according to [146], not have equal importance. Salah et al. state that for analyzing the significance of suspicious events, a prioritization component needs to be added to the correlation system [94] such that it can be distinguished between moderate and devastating threats/attacks. Porras et al. propose an alert correlation and ranking technique called M-Correlator [147] which ranks alerts based on the likelihood of the attack to succeed, the importance of the targeted asset, and the amount of interest in the type of attack [148]. Alsubhi et al. state in their work that these techniques are promising in the evaluation of alerts generated by signature-based IDSs, but cannot evaluate alerts raised by anomaly-based IDSs, since they heavily rely on the vulnerability knowledge base. Thus, in their work [148] they extend Porras et al.'s approach by offering a technique which works with both signature-based and anomaly-based IDSs and makes use of additional criteria, such as the sensor sensitivity, relationship between alerts, service stability, and social activity between source and target for a more accurate evaluation of the alerts. A few alert prioritization score metrics from [148] are categorized in [7] to applicability metric (applicability of a raised alert to the current environment based on information from various knowledge bases, e.g. vulnerability knowledge base), victim metric (specifying the properties of critical machines, services, applications, accounts, and directories in the current network environment), sensor status metric (based on Bayesian detection rate formula estimating

true positive probability that an alert is raised when an attack is detected), attack security metric (measuring the risk level of a vulnerability based on known attack metrics such as MITRE or CVE), service vulnerability metric (representing the strengths and weaknesses of a host based on the targeted services) and social activity metric (exploiting features of a social network to find hidden participants in a communication session). McElwee et al. propose a deep learning based approach for prioritizing and responding to alerts in [149]. Their FASTT (Federated Analysis Security Triage Tool) concept uses a TensorFlow Deep Neural Network classifier to automatically categorize IDS alerts and determine which type of security analyst should review the alerts. In addition, FASTT uses an Elasticsearch indexed data store to speed the retrieval of IDS alerts and a Kibana user interface to allow flexible display of visualizations and dashboards that can be tailored to meet the security analysts' workflow. ACSAnIA (A Comprehensive System for Analysing Intrusion Alerts) is proposed by Shittu et al. in [150] which is a post-correlation framework that consists of seven components. *Offline Correlation* uses a set of historic alerts to build a correlation model which gets updated periodically by the *Online Correlation* module. This module, for every incoming alert, analyzes it against a set of past alerts of a certain time window. *Meta-alert Comparison* measures differences between meta-alerts that are produced by the correlation process and *Meta-alert Prioritisation* maps a prioritizing value to each meta-alert based on the degree to which it is an outlier computed by the local outlier factor algorithm. *Meta-alert Clustering* receives the set of meta-alerts and groups them into clusters. The *Attack Pattern Discovery* receives the clusters of meta-alerts and attempts to extract a set of representative features for each cluster using frequent pattern mining. The *Reporting System* receives the outputs and prepares them for human interaction. The results of alert prioritization are useful for countermeasure selection, where the system is able choose a suitable response automatically based on the assigned values for detected attacks (Section 4).

3.3.5 Root Cause Finding

Alerts typically are indicators of problems representing the symptoms of incidents but do not provide explicit information regarding the actual root cause of them. However, identifying the "culprit" of malicious activity is not only important for forensic purposes but also for finding the causation as quickly as possible to establish countermeasures as nearly located as possible. Simple mechanisms such as IP traceback, a method for identifying the origin of a packet on the Internet based on its source address, seem insufficient for root cause finding since for the propagation of malicious activity, the source of packets is almost never the source of it but just one of the many propagation hops. Furthermore, IP-spoofing prevents its application. Even the detection of stepping-stones, by techniques stated and discussed by Kumar

and Gupta [151], only focuses on the IP level of packets which make the root cause finding difficult. Furthermore, they are vulnerable to time delays, chaff perturbation, and have a high false positive rate. Root cause finding methods are tractable estimators performing on multiple topologies to find propagation sources in higher level, e.g. application level to find logical structures, of networks. Approaches of identifying malicious attack sources can be divided into three main categories: the complete observation based source detection (requires a complete observation of the attacked network after a certain time of the malicious propagation), the snapshot based source detection (requires a partial observation of the attacked network at a certain time), and the detector/sensor based source detection (requires the observation of a small set of nodes but all the time in the attacked network) [141]. Figure 17 illustrates the three categories and Figure 18 a taxonomy of source identification methods based on this categorization. For a detailed description of the following refer to [141].

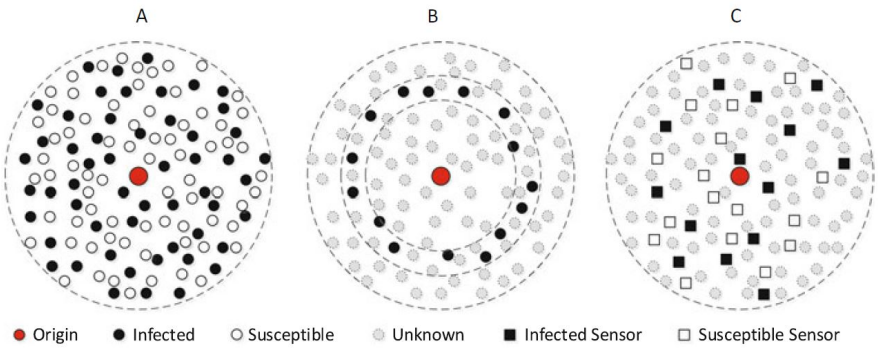

Figure 17: Illustration of three categories of observation in networks. (a) Complete observation; (b) Snapshot; (c) Sensor observation [141]

Examples for the former are the Rumor Center of the work in [152] or Dynamic Age of [153]. In their works [152, 154, 155], Shah and Zaman provided models for rumor spreading in a network based on the SI model and then constructed an estimator for the rumor source based on a novel topological quantity, called rumor centrality. They established a maximum likelihood estimator for a class of graphs: regular tree graph, general tree graph, and general graph. The second category deals with the problem that in real-world networks a complete observation might not be possible, thus, a given snapshot at a certain time serves as a basis for source identification. A representative of this class is the work of Zhu and Ying [156] which base their solution on the SIR model and provide a reverse infection algorithm based on a given network snapshot. A node with the minimum infection eccentricity, called Jordan center, is the estimated source node by the algorithm. Approaches using sensor observations are, among others, based on Bayesian [157], Gaussian [158] or Moon-Walk [159] (Figure 18).

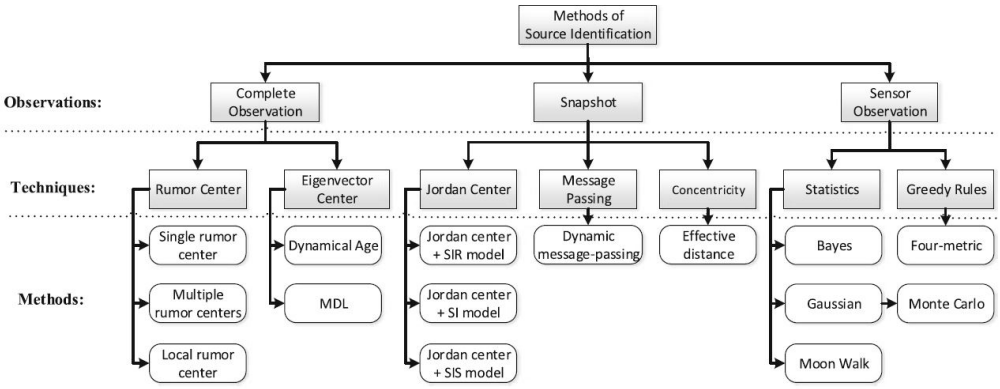

Figure 18: Taxonomy of current source identification methods [141]

The aforementioned approaches for root cause identification mentioned in [141] are, however, not relying on the output of IDSs in form of alerts. Julisch [160] initially proposed a different approach for source identification using a set of unlabeled NIDS alerts and generating clusters of similar types. The work discusses alarm clustering as a method that supports the discovery of root causes. Another clustering based approach is Y-AOI [161] that bases on the Y-means anomaly detection method [162] and uses an attribute-oriented induction algorithm. Firstly, Y-means divides the alerts into several clusters based on their occurrence time and secondly an adopted oriented induction algorithm inducts the clusters into short, highly comprehensible and more informative summary tables which help administrators to more easily find the root cause. Al-Mamory and Zhang use root cause analysis in [163] to discover the sources that make IDSs trigger a large number of alerts by supposing that most of the root causes are no attacks. Similar alarms are clustered by their semi-automated clustering system, also basing on the attribute-oriented induction algorithm, helping the security analyst in specifying the root causes behind these false alarms and in writing accurate filtering rules. Kechadi et al. present an approach called Behavioral Proximity Discovery which is a framework for root cause analysis that consists of three complementary clustering algorithms based on alarm behaviors. The first algorithm (SM) identifies periodic alarm behaviors. The two other algorithms (FECk and CUFRES) correlate events leading to the identification of faults by the network operator [164]. An automated root cause identification approach is proposed by Cotroneo et al. in [165]. Their framework (Figure 19) consists of a filter and a decision tree to address large number of alerts and to support the automated identification of root causes by adopting term weighting and conceptional clustering approaches to fill the gap between the unstructured textual alerts and the formalization of the decision tree.

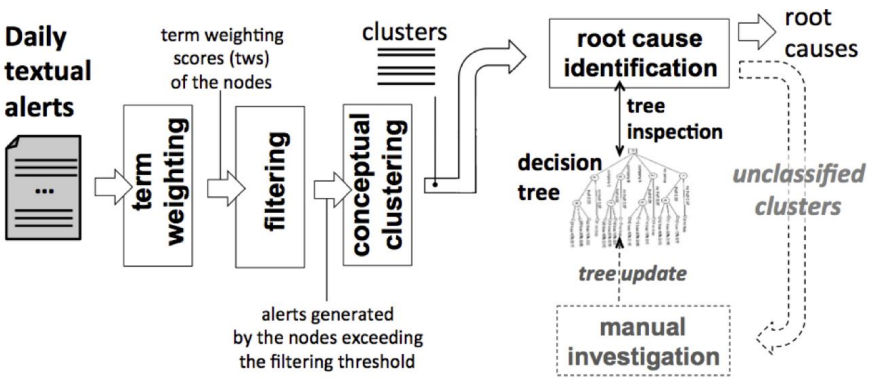

Figure 19: Root cause identification framework [165]

Another promising possibility for identifying the root cause comes from the field of exploiting the physical properties of network participants. Physical device fingerprinting builds fingerprints to uniquely identify a machine by e.g. measuring the differences in machine internal clock signals. In [166] a time-based device fingerprinting technique is proposed that is generic and can work with different functions, making the method adaptable to different environments. Cho and Shin proposed an anomaly detection approach called Clock-based IDS (CIDS) that is based on the unique clock skews of an ECU in [167]. Since the CAN frame in an in-vehicle communication does not deliver any information about its source ECU, the downside of many IDS approaches is that anomalies cannot be traced back to its source. By associating a clock skew to each ECU in the CAN network, every message can be backtracked to its source ECU. CIDS first creates a norm baseline based on clock skews which is done by measuring the intervals of periodic messages constructed on the recursive least squares technique. After the construction of the norm baseline, each ECU is associated with its own clock behavior or fingerprint. The detection of anomalies is conducted from using the cumulative sum method which is used to detect abnormal shifts in identification errors. With Viden (Voltage-based attacker identification), Cho and Shin provide another possibility in [168] to identify the attacker ECU by measuring and utilizing voltages on the in-vehicle network. Viden exploits the voltage measurements to construct and update the transmitter ECUs' voltage profiles as their fingerprints and uses these profiles to pinpoint the attacker ECU with a low false identification rate. A major benefit of Viden compared to CIDS is the ability to function even in the presence of event-triggered messages. CIDS rely on the timing information from periodic messages but since Viden determines the transmitter ECU based on voltages, it is able to pinpoint attacker ECU regardless of the communication technique.

4 Incident Response

Incident response or reaction implies the set of actions a system executes after security-relevant incidents have been identified. An IPS, as a proactive solution, work as an IDS but in the case of an detection it drops malicious traffic automatically before it causes any harm to the network rather than raising an alarm afterward. Similar to the term countermeasure defined in RFC 2828 [169] as "an action, device, procedure, or technique that reduces a threat, a vulnerability, or an attack by eliminating or preventing it, by minimizing the harm it can cause, or by discovering and reporting it so that corrective action can be taken", a response refers to a specific action that is triggered by an alert or an alert analysis after an intrusion detection phase (reactive). Evaluating the severity of attacks, identifying the cause of incidents, and selecting an appropriate response under considerations of e.g. the correct time or the available resources are typical tasks of an IRS. Intrusion risk assessment is closely related to the field of IRSs. This process helps to determine the probability that a detected incident is a valid, action demanding attack towards an important compromised target that requires a certain form of a response. Properties and characteristics that influences the response model e.g. incident response time are provided in [170].

4.1 Taxonomy of Intrusion Response Systems

A taxonomy for IRSs is provided by Shameli-Sendi et al. in [11] and shown in Figure 20. The characteristics and criteria of their taxonomy resulted from a comprehensive literature review and are listed in the following. Some additional requirements that should be fulfilled for the development of an IRS stated by [170, 171] are added in *italics* below.

- ***Activity:*** The activity of a triggered response can be categorized into passive (do not attempt to minimize damage already caused by the attack or prevent further attacks - main goal is notification) and active (aim to minimize the damage done by the attacker and/or attempt to locate or harm the attacker).
- **Level of Automation**: An important feature of an IRS is whether it can be fully automated or requires administrator intervention after each incident. Therefore, the level of automation can be categorized into notification systems (alert information is used by an administrator to select applicable response measures), manual response systems (pre-configured set of responses based on the type of attack that an administrator has to trigger) and automated response systems (automated execution of responses without human intervention). *Cooperation and autonomy for response systems are two features used respectively in network and host intrusion detection system. Thus, it is necessary to have both features in a single system to be more accurate.* According

to [16], automated IRSs can further be divided: adaptive (feedback loop to evaluate previous response), expert (response decisions are based on one or more metrics) and association (simple decision table approach in which a specific response is associated with a specific attack).

- **Response Cost**: Knowing the power of responses to attune the response cost with attack cost plays a critical role in IRSs. The evaluation of the positive effects and negative impacts of responses are very important to identify response cost. *Thus, the selected response should not be more costly than the attack.*
- **Response Time**: This criterion refers to whether the response can be applied with some delay after an intrusion is detected (delayed, reactive) or before the attack affects the target by applying an intrusion prediction mechanism or IPS (proactive). *Thus, proactive response mechanisms should be included within the intrusion response system to enable early response to intrusions.*
- **Adjustment Ability (*Adaptiveness*)**: Usually, an IRS framework is run with a number of pre-estimated responses. It is very important to readjust the strength of the responses depending on the attacks. *Adaptiveness is a powerful feature required to ensure normal functionality while still providing effective defense against intrusive behavior, and to automatically deploy different responses on the basis of the state of the current system.* Adjustment models can be categorized into non-adaptive and adaptive. The former keeps the order of responses during the life of the IRS, whereas the adaptive has the ability to adjust the order based on their history. Non-adaptive adjustment models can be converted into adaptive ones. Response goodness refers to the history of success and failures of each response mitigated over time.
- **Response Selection (*Effectiveness*)**: The task of an IRS is to choose the best possible response. Existing techniques vary in the way response selection is achieved. *The response should be tailored to the type of attack and incident context. In fact, the response could be different for the same attack affecting two kinds of resources.*
- **Applying Location**: There are different locations in the network to mitigate attacks. Using information from an "attack path" (consisting of (1) adversary's start point, (2) firewalls/routers, (3) intermediary devices and (4) target device) appropriate locations with the lowest penalty cost for implementing response measures can be identified.
- **Deactivation Ability**: Another distinguishing feature that separates IRSs is response deactivation (response lifetime), which can take into account users' needs in terms of Quality of Service (QoS). Most countermeasures are temporary actions which have an intrinsic cost or induce side effects on the monitored system, or both. Thus, there must be an ability to deactivate response measures.

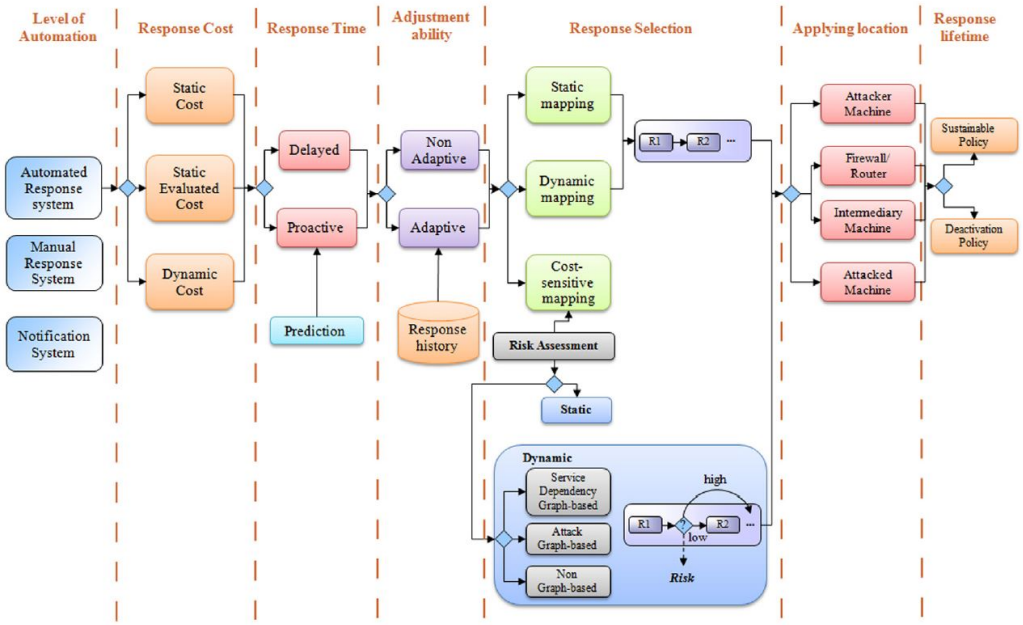

Figure 20: Taxonomy of intrusion response systems [11]

A phylogenetic tree showing the history and advancement of intrusion response approaches is provided by [16]. The authors further propose characteristics (adaptive nature, cost-sensitive, semantic coherence, manage false alarm, and response metrics policy) for designing proper response systems since poorly designed ones may lead to a generation of a large number of false alarms and degrade the performance of the system. They state that existing IRSs are unable to provide a real-time optimum response because of the absence of semantic coherence and dynamic response metrics features. A survey of research activities with respect to IRSs highlighting their novelty and core characteristics of relevant ones is provided in [4]. Challenges and future direction of IRSs are discussed in [4, 16] and a selection illustrated in Figure 21. Further, open issues and some proposed solutions in the field of IRSs in general and with respect to cloud-environments for smart mobile devices are provided in [9, 10].

4.1.1 Response Cost

Response cost describes the impact of implementing responses on the system's protection goals. This may not only include security-relevant ones, such as data confidentiality, integrity and authenticity, but also safety-critical ones e.g. availability or performance. For instance, switching off a service to mitigate an attack results in the loss of its function. An evaluation of the response cost makes only sense when considering possible attack scenarios. A dynamic response model (static cost, static evaluated cost, dynamic evaluated cost) offers the best response based on the current situation of the

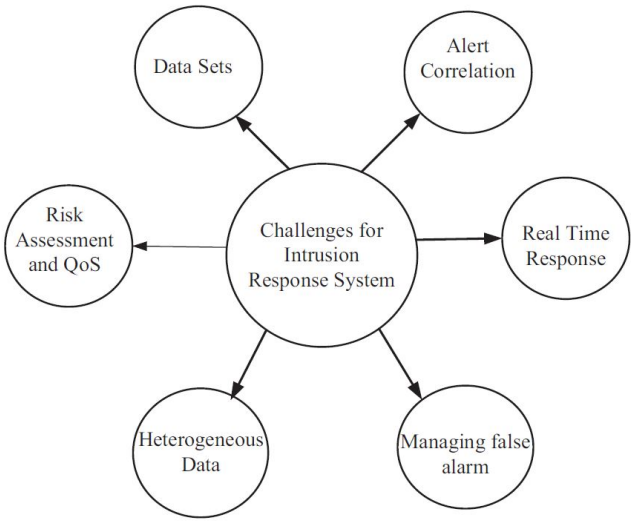

Figure 21: Challenges for IRSs [16]

network, and so the positive effects and negative impacts of the responses must be evaluated online at the time of the attack [11]. For the static cost model an expert assigns a static value to a response ($R^s_{cost} = const.$). If an evaluation function is applied, typically incorporating the impacts on the protection goals, associated with each response, a statically evaluated cost is obtained ($R^{se}_{cost} = f(x)$). In the dynamic evaluated cost (R^{de}_{cost}) the systems' situation is evaluated in an online fashion such that certain responses are only implemented if other entities e.g. other services, processes, resources are not threatened.

Herold presents possibilities to assess the costs of a response as well as the potential damage of a security incident in [170]. The author categorizes them into basic approaches, e.g. [172, 173, 174, 175], that select a response if the cost of it is lower or equal to the damage cost, approaches including system importance, e.g. [176, 177, 178], approaches using probabilities based on the success likelihood of an ongoing attack, e.g. [178, 179], and approaches including IDS capabilities e.g. [180, 181, 182]. Fessi et al. for example present an approach for automated reaction measures with a multi-attribute decision model and a cost-benefit analysis of the selected reaction in [183]. They present an intrusion response architecture composed of a collection module, a detection module and a response module, with the actual focus of the paper on the response module. The response module contains information regarding the known intrusions, possible reaction measures in case of an intrusion and a response mechanism for deciding on a suitable reaction measure. The problem of finding a suitable response to an intrusion is defined as a muti-attribute decision making problem. It is based on the three parameters: *financial cost* (loss for the company in financial terms), *enterprise reputation loss* (reputation of the company which is related to

its survival and existence), *processing resources* (regarding personnel and information assets). These attributes are normalized using a weighted linear combination method such that the values can be used later in the evolutionary algorithm's fitness function. Using the evolutionary algorithm, the best possible response measure for intrusion is then determined by a cost-benefit analysis. A matrix with n columns, which represent the resources in the system, and m rows, representing the reaction measures, is used. This means that one row indicates the effects of the reaction measure m on the overall system and is regarded as a chromosome for the algorithm. The evolutionary algorithm follows the usual operations (random initial population, selection, crossover, mutation, natural selection) and, after scheduling, provides the optimal reaction measure in terms of cost-benefit for the overall system. Shameli-Sendi and Dagenais proposed a practical framework in [173] for online response cost evaluation to encounter the problem that typically a good response decreases the service quality. Thus, a balance between response cost and the cost for an attack in a cost-sensitive fashion can be achieved. A service dependency graph is utilized to consider the negative effects that consist of impacts on e.g. the network or hosts. Their framework includes information of affected resources determined by an expert's opinion.

4.1.2 Response Selection

Security administrators often face multi-criteria decision making problems when it comes to select an optimal response in a timely and cost-effective fashion. According to [11], there are three response selection models namely *static mapping* (an alert is mapped to a predefined response), *dynamic mapping* (response mapping to an attack differs depending on e.g. system state, attack metrics such as frequency or severity, or network policy) and *cost-sensitive mapping*. Static mapping itself can be exploitable since an attacker may predict response measures. Dynamic mapping lacks on intelligence by improving itself with every new responded attack without any dedicated upgrade. Cost-sensitive mapping attempts to attune intrusion damage cost which represents the amount of damage to an attack target when the IDS is either unavailable or ineffective. It is closely related to risk assessment which itself can be categorized into *static* and *dynamic*. Dynamic (real-time) risk assessment approaches can subdivided into (1) *attack graph-based* (referring to Chapter 3 by constructing attack steps based on correlation methods), (2) *service dependency graph-based* (impact on confidentiality, integrity, and availability on a service interacting with or depending on others), and (3) *non graph-based* (risk assessment carried out independent of the attack by performing risk analysis on alert statistics and other information provided by alerts) [11, 101]. An overview of attack modeling techniques including their strengths and weaknesses is presented in [4] structured into attack graph, attack tree, service dependency graph and Markov decision process models. A

method for risk assessment based on the dynamic Bayesian network (static Bayesian network extended in time) is proposed by Wang et al. in [184].

Already in 2000, Schackenberg and Djahandari proposed an architecture based on the Intruder Detection and Isolation Protocol (IDIP) in [185] which represents an early work in the area of automated intrusion response. IDIP's objective is to track intrusions by sharing information between neighboring devices to attempt responding or tracing back the attack along it's path. Local IDIP agents are equipped with detection, audit or response functionality and reports are not only distributed among each neighboring agent for local decision making but also sent to a discovery coordinator which is able to correlate reports and to gain a better overview. For response functions, IDIP defines trace messages (including a description of the anomaly, a value indicating how certain the detector is of this attack, a severity value based on the potential services lost from this attack, and a requested response) and directives (messages sending "do" and "undo" messages in order to e.g. block or unblock network traffic). With the Cooperative Intrusion Traceback and Response Architecture (CITRA) presented in [186], the authors extended their work to trace attacks to their source and block them as close as possible to it. CITRA allows to use immediate responses and utilizes a simple cost model to select a limited number of responses which is based on thresholds and can not be adapted. However, CITRA's main focus is on the traceback of an attacker to the source of the security incident.

Zonouz et al. propose a game-theoretic intrusion response engine in [187], called the Response and Recovery Engine, which allows to analyze security incidents and taking their optimal countermeasures using attack-response trees modelling a two-player Stackelberg stochastic game. The engine considers inaccuracies associated with IDS alerts and chooses an optimal response action by solving a partially observable competitive Markov decision process derived from the attack-response trees. An extraordinary approach has been presented by Sharma et al. in [188, 189] proposing an intrusion response mechanism inspired by plant-based biology. The PIRIDS (Plant-based Inspiration of Response in IDS) is a three-layered bioinspired detection and a response method composed of the layers PRR (Pattern Recognition Receptors), Guard Agent & SAR (Systematic Acquired Resistance) and HR (Hypersensitive Response). With their approach the authors were able to cope with known attacks, zero-day ones and the infection spread of malicious activities could be stopped.

Kholidy et al. present ACIRS (Autonomous Cloud Intrusion Response System) in [190], an effective attack and vulnerability detection and response framework to accurately identify the attacks targeted to cloud environments. Its architecture integrates both, behavior and knowledge based techniques, and considers different service models and user requirements, collects and correlates security events and user behaviors from all environments in the cloud system. It provides a security measure to assess the system risk to

select an appropriate response to mitigate the risk consequences. Alert integration, correlation and risk assessment is based on IDMEF to include multiple detection mechanisms such as Snort (network-based) and OSSEC (host-based) for instance. Thus, as normalization process, all alerts are brought into IDMEF to simplify their analysis and correlation in the next layer. A prioritization process ensures that each detector (analyzer/sensor) has its own prioritization system. Alert correlation and summarization correlates a large number of normalized alerts from different detectors to highlight the few critical ones. This technique looks for evidences of an alert to discover if it signals a true attack and it correlates logically related alerts. Logically related alerts denote the same attack signature, have the same source and destination addresses and are close in time. These alerts may also denote a step of a multi-stage or compound attack that consists of a set of steps performed by one attacker. Beside reducing false positives alerts and summarizing the huge number of alerts to the cloud administrator, this results in a correlation process that deals efficiently with multi-stage attacks and facilitates the risk assessment and mitigation processes. The authors propose to use OSSIM, an open source correlation engine that uses a tree of logical conditions (rules) or AND/OR tree in the correlation process. In each correlation level of the correlation tree (or respective to a "tree" of hyper-alerts) a risk value is updated according to an equation in the paper. The respective correlation and risk assessment flowchart with N correlation levels, each with different number of rules, is shown in the publication.

In [176], Ossenbühl et al. present REASSESS (Response Effectiveness Assessment), a response selection model that evaluates negative and positive impacts associated for countermeasures to mitigate network-based attacks. Requirements are automatic deployment, scalability, adaptability, system independency, calculation efficiency, usability and protection preventing unauthorized access by deploying security mechanisms. The negative effects lead from possible service degradation, penalty costs can occur from service level agreement violations. Furthermore, alert priorities are taken into account. REASSESS is based on several stages related to the NIST incident response cycle for the response selection process as shown in Figure 22 and follows a sequential execution. The assumptions, namely aggregation and confidence which assume that each alert raised by a detection engine is treated as one attack and a strong confidence with a hundred percent certainty of alerts, however, strongly mitigate the usage of anomaly-based IDSs which might raise a large number of false alerts. Furthermore, REASSESS in its current state is working sequential, not able to cope with multiple IDS nodes, does not use common standards for the exchange of incident related information and is not capable of coping with more advanced attack scenarios due to the lack of an alert correlation mechanism.

An optimal metric-aware response selection strategy using mixed integer linear programming to obtain an optimal subset of responses is presented

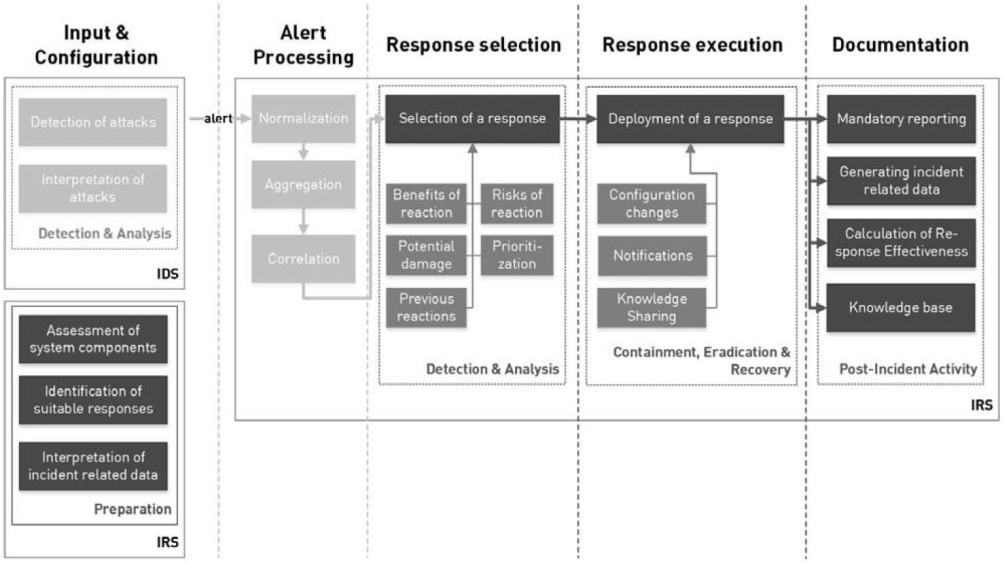

Figure 22: Response selection process of REASSESS [176]

by Herold et al. in [191]. Thus, they were able to provide response strategies much faster and with higher quality than using simplistic heuristics allowing e.g. a larger number of responses or entities. GhasemiGol et al. proposed a network attack forecasting strategy in [101] by using an attack graph and a dependency graph approach enriched by analyzed IDS alerts (hyper-alerts graph) to identify possible risky nodes and IRS activated responses (multi-level response graph). The utilized attack graph handles the uncertainty of an attack probability by measuring the probability of vulnerability exploitation. Instead of using IDMEF for alert analysis, the authors use their E-correlator, which is a similarity correlation system that functions on raw alerts and outputs a directed hyper-alert graph. They define a response graph with eight levels (refer to Figure 25). By the combination of various graph-based approaches, the authors were able to handle uncertainties of current attack graphs and predict future network attacks by the inclusion of additional information. In their consecutive article [192], GhasemiGol et al. extended their work to a foresight model for intrusion response management that includes a response cost estimation based on whether the impact of a response is negative or positive on each level of their multi-level model considering the confidentiality, integrity, and availability parameters. In a recent work, Shameli-Sendi et al. propose a framework for selecting and deploying optimal countermeasures to intrusions dynamically in [193]. Therefore, an optimal countermeasure is identified by evaluating the current and potential damage cost, the accuracy of the countermeasure risk reduction, the impact on QoS and the balance between countermeasure and attack damage cost. An advantage of this framework is that countermeasures are not predictable for an adversary since they are not statically defined.

4.2 Intrusion Response Representation

With respect to Herold [170], responses may be reconfigurations of hosts or network elements. Targeting the implementation of reactions on network elements, Herold presents Simple Network Management Protocol (SNMP) and Network Configuration Protocol (NETCONF) defined in RFC 6241 [194]. NETCONF makes it possible to read out, install and manipulate the configuration of devices and is based on a simple RPC layer (Remote Procedure Call). It uses XML-based encryption to transfer both configuration data and protocol messages. Among the advantages are the human readable representation of the data, the reusability of the message structures as well as the easy extensibility. The data model belonging to NETCONF is called YANG (Yet Another Next Generation). This model explicitly and unambiguously defines the structure, syntax and semantics of the configuration and operational data of network devices. It thus offers a uniform interface to their manipulable and operational data. NETCONF has not been defined to inherit information of intrusion responses. However, it could be applied when it comes to network device reconfiguration triggered by an IRS.

The IDIP application and message layer is defined in [195] providing details on the objectives, specification and operations of IDIP and has been used by [185, 186]. IDIP applications use trace messages to describe network-based intrusions which are passed to neighboring devices to trace the path of the intrusion and provide the information necessary for each device along this path to determine an appropriate response. Other IDIP application messages are used to support this tracing and response mechanism. As stated in the specification, IDIP is designed to minimize the size and number of messages required to support intrusion response. Application layer messages are primarily sent only after an intrusion has been detected. Once the response has been initiated, the protocol attempts to only send messages to components that potentially could have witnessed parts of the attack. In addition, IDIP components send reports of the responses to a centralized management component called the discovery coordinator. [195]

Since IDMEF does lack in messages for specifying responses, Klein et al. introduce the Intrusion Response Message Exchange Format (IRMEF) in [196]. It extends IDMEF as shown in Figure 23 by a response class that use elements that are already defined in the IDMEF message class. Those are the CreateTime, set to the time the response message is created, the DetectTime to schedule the time of response execution, the Source issuing the response, the Target(s) to which the response should be applied, an Assessment field containing the action that should be performed e.g. kill process and an AdditionalData field containing optional elements e.g. parameters for the response e.g. process ID for the process to be killed. The proposed communication protocol between those agents and a console is SNMP in version 3. This scheme does not allow to schedule more actions in an appropriate

manner, for example, neither sequential or otherwise timed actions or the interconnection of multiple actions are possible nor the specification of the control flow in more detail [170].

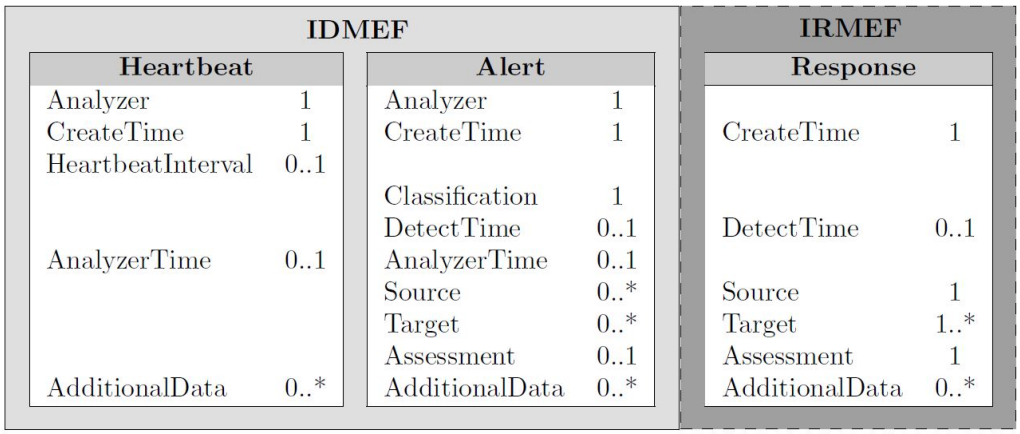

Figure 23: Intrusion response message format - IRMEF [196]

An intrusion response message format similar to the IDMEF is proposed in [192] and represented by XML based on the multi-level model information from [101]. Figure 24 shows the corresponding data model consisting of the Intrusion Response Message (IR-Message) and its subclasses providing further response information. The attributes of the IR-Message are a response identification number (Response-ID), a Response-status indicating the response condition being active or inactive and a Response-cost containing the impact of the response on the network entities. The Response-Target contains information about the target with respect to the authors' multi-level response model. Response-Type indicates the kind of responses in terms of impact level and Response-Location the place of applying responses such as Firewall. The Response-Action shows the kind of actions that can be applied by responses e.g. shutdown, reset, block, notify.

4.3 Possible Response Measures

Table 3 illustrates different classes of response actions with examples that might be applicable, depending on the type of consequence and the involved assets. Those measures are used to neutralize an individual attack or execute preventive measures to secure the system against future attacks of the same type. GhasemiGol et al. define different levels in [192], as shown in Figure 25, that can also be applied to map response measures to different levels. Those are composed of *notification-level* (generation of a report or alarm), *attacker-level* (targeting the attacker machine by e.g. blocking the attacker IP), *vulnerability-level* (patching or updating software to remove vulnerabilities e.g. CVE countermeasures), *file-level* (block file or change file

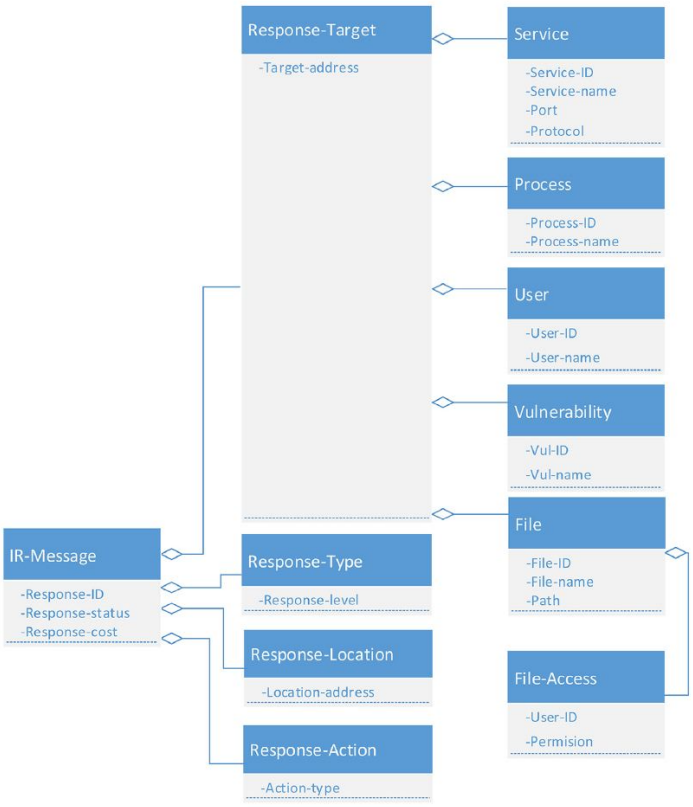

Figure 24: Intrusion response message format - IR-Message [192]

permissions), *user-level* (block user or reduce user privilege), *service-level* (block affected processes, services or ports), *host-level* (affects victim machine by e.g. shutting it down), and *unclassified-level*.

According to [198], responses with respect to virtual cloud systems can be categorized into filtering (e.g. updating upstream filters to block traffic), rate limiting (attempts to relieve the pressures on bottlenecks), adapting the use of virtual machines (e.g. increase or decrease its number) and identifying the attack source (using trace-back techniques). A more generic categorization into passive and active responses with lists of common responses per category based on the work of [10, 171, 199] and inspired by the multi-level categorization of [192] is illustrated in Figure 26. The dotted arrow indicates that some response measures (unclassified-level) categorized under passive might also be active ones depending on their interpretation and level of automation. In the following sections, two examples for response mechanisms are presented: adapting IDS components by changing its sampling rate as well as self-regulation and the reconfiguration of a network infrastructure to enhance security applying SDN.

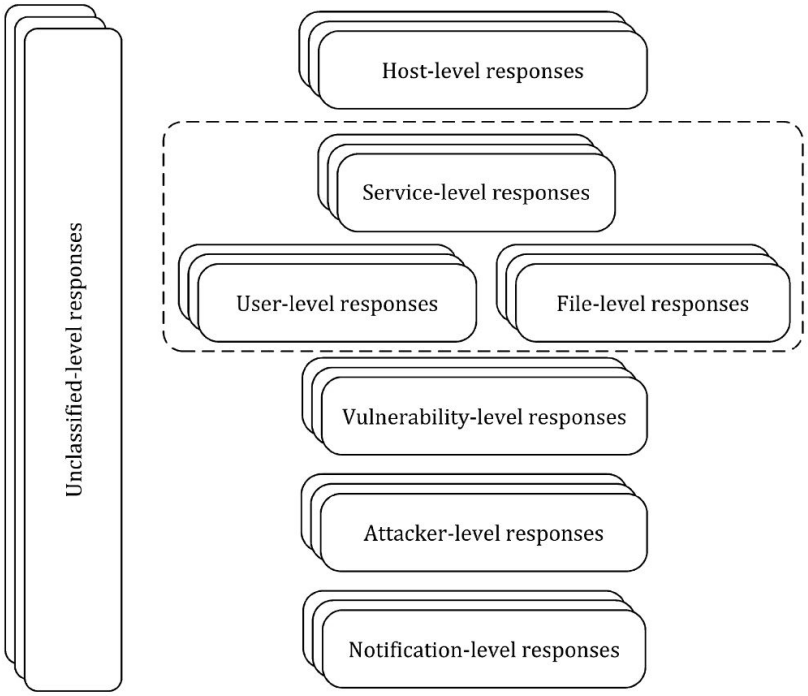

Figure 25: Different intrusion response levels [192]

4.3.1 Adaptive IDS

Note: This subsection has been published by the author in [200] and states two possibilities, sampling and self-regulation, that can be used to adapt IDSs as a response measure. In this report the work can be categorized into the field of adaptive IRSs. This is because the presented Uncoupled MAC algorithm, a cryptographic scheme that applies message authentication codes decapsulated from the original packet, is able to dynamically adjust its sampling parameters depending on the detection of both a message's data integrity as well as its authenticity violation. The term adaptive in the context of this work describes techniques of sampling and self-regulation. To cope with the increasing amount of traffic within networks while reducing large memory and CPU processing requirements, sampling turned out to be a promising scalable data aggregation technology for IDSs since the processing capacity of such systems are typically much smaller than the amount of data to be inspected. Because sampled traffic is an incomplete approximation of the actual one, multiple mechanisms for sampling data exist.

Sampling In [201] a difference between packet- and flow-based sampling, crucial for the working of NIDSs, together with deterministic and non-deterministic methods is made. Packet-based sampling is simple to implement with low CPU power and memory requirements but is inaccurate for infer-

Action class	Description	Examples
Rollback	bring the component back to a saved secure state	restart virtual machine or software
Rollforward	find a new state from which the component can operate	restore or update process
Isolation	perform physical/logical exclusion of the "faulty" components	block attacker's IP address
Reconfiguration	reconfiguring a system, component or reassign tasks to others, activate spare components	reconfigure network routing
Reinitialization	check/record new configuration and update system tables	restart a TCP connection

Table 3: Response and recovery action classification (cf. [197])

ence of flow statistics like size distribution of the original flows. In contrast, flow-based sampling overcomes the limitations of packet-based sampling but suffers from prohibitive memory and CPU power requirements and is still too complex to implement [202]. The flow-size based sampling technique in [201] assumes that network attacks usually use small flows as traffic source. With the proposed selective sampling strategy such flows are sampled with a constant probability. Other related work evaluated that packet sampling has a negative impact on the efficiency of anomaly-based IDSs increasing the false positives but performs best when using a random flow sampling strategy. However, it is possible to maintain a high level of security while selectively inspecting packets with a minimal amount of processing overhead. An analytic and statistical model for the process of network intrusions has been introduced in [203] supporting the experimental results of [201] demonstrating that it is sufficient to inspect only a small number of sampled packets. In [204] a packet- and time-driven traffic sampling strategy for an IDS in a SDN is proposed that fully utilizes the inspection capability of malicious traffic, while maintaining the total aggregate volume of the sampled traffic below the inspection processing capacity of the IDS. The packet-driven approach inspects a packet every $1/x$ packets for a sampling rate x and the time-driven inspects all the packets within a time window of sampling rate x each sampling interval. The time-driven mechanism has the advantage of detecting stateful attacks because it captures all the packets for a certain time duration. However, if packets are mainly sent event-triggered, the time-driven approach is not feasible since there could be phases of sampling in which no packets are inspected. This could easily happen in networks with high fluctuations of the bandwidth. If an intruder is able to compromise the IDS or might know the sampling rate, he could exploit this knowledge by performing malicious activities outside the sampling interval. By increasing a sampled injection of malicious packets, he could also extract the sam-

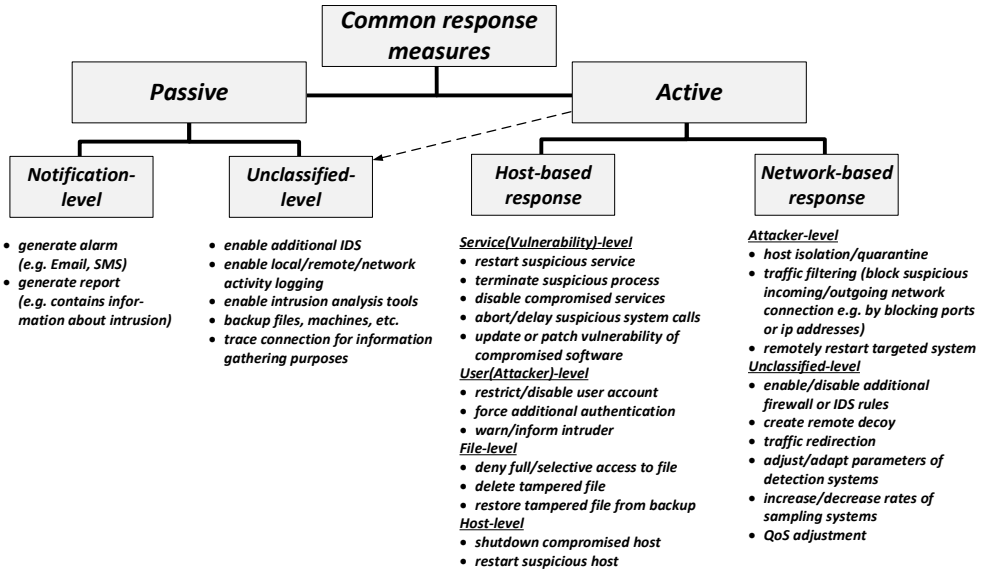

Figure 26: List of common response measures (cf. [171, 10, 192, 199])

pling rate information by observing the reaction of the IDS in a trial and error fashion. Thus, a combination of a packet- and time-driven mechanism could mitigate such problems by applying a random chosen sampling interval within fixed boundaries.

Self-Regulation Self-regulating sampling mechanisms have been presented in [205], for instance a method managing the processor usage in a network device through adaptive sampling in network security applications. The authors state that a wide range of common network anomalies only require a single sample in order to provide 100% accuracy of detection but there are also other network anomalies which cannot be detected with a single sample. An example is an anomaly which misuses a protocol for purposes which were not meant for it. This requires more advanced techniques than a simple signature check. Cryptographic mechanisms could be used to overcome such limitations. In [206] an adaptive packet-level sampling method on different traffic fluctuations and burst scales has been introduced. The method can dynamically adjust each packet sampling probability depending on the magnitude of traffic fluctuation. This approach achieves higher accuracy in contrast to random sampling methods. Another adaptive sampling method for anomaly detection algorithms has been presented in [207]. The adaptive sampling described is a promising general sampling technique that preserves well the traffic feature distributions and at the same time is able to improve the detection capabilities of the system. A hybrid sampling algorithm combining both flow statistics and feedback to intelligently choose the packets to sample is presented in [208] in order to achieve self-regulation. The sampling

rate is determined by the current workload in the cloud, and thus minimizing the effects to normal workload. By the cloud-based IDS framework defined, an off-the-shelf IDS can be utilized in a cloud environment by reducing and balancing the data collection (packet capturing, filtering, sampling rate) and computation workload dynamically according to the resource utilizations in the cloud. Another example of adaptive sampling systems is the work presented in [209] that aims to effectively reduce the volume of traffic that Peer-to-Peer (P2P) botnet detectors need to process while not degrading their detection accuracy. The system first identifies a small number of potential P2P in high-speed networks for botnet detection. In a 2-step approach first a suspicious host identification is performed by roughly sampling the traffic in order to detect potential P2P bots quickly. Second an in-depth analysis with more fine-grained detectors achieve an accurate detection on the identified hosts.

Applying sampling techniques in conjunction with a self-regulating IDS helps to reduce the measurement overhead for an IDS in terms of CPU, memory or bandwidth enabling the application of a partial IDS in future connected embedded systems. Similar to the concept of partial networking, the IDS components regulate their activeness such that in times of higher detection of malicious actions within the network more packets will be sampled leading to a higher resource consumption. On the other hand, in times of less or no detection, the IDS components lower their sampling or might even partially turn off completely. Furthermore, by using adaptive techniques that regulate, for instance, IDS relevant parameters or the sampling rate, the security level of a system can be adjusted by preserving a controllable overhead on resources.

4.3.2 Network Reconfiguration Leveraging SDN

Note: The following paragraph has been published by the author in [12] presenting a generic incident handling framework. 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 [210]. 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 [211]. 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 [212].

A "pluggable" software platform aimed to provide centralized administration and experimentation for anomaly detection techniques in software defined data centers is provided by [213]. The proposed SDN-PANDA consists of three controller-centric application modules responsible for (1) data collection and pre-processing of switch aggregated flow statistics, (2) anomaly detection based on a flexible interface and (3) performing response actions defined on standard policies independent of the anomaly detection method. The response policies must be accurately defined such that they only mitigate the identified attack and not cause a loss of service by e.g. dropping packets. To address this issue they propose to apply redundant services in different spots of the infrastructure and reroute the traffic in the case of incidents which allows e.g. the investigation of the compromised host while still guaranteeing the availability of the service. In [214] the authors leverage SDN and Network Function Virtualization (NFV) technologies for incident response in industrial control systems. They provide potential response use cases including rerouting attacker traffic to a honeypot, changing forwarding rules to drop communications or transfer services from compromised devices to redundant ones using virtualized resources. In a proof of concept they are extending MiniCPS providing SDN functionality and deploy an IDS in the SDN controller which is working threshold-based and compares actual and estimated sensor values. Incident response functionality is quite limited following preconfigured policies such as discarding packets from the compromised sensor and replacing them with the estimated values of the IDS. Afterwards, the SDN controller modifies the flow table of the SDN-capable switch to leverage the response. The authors state that although their application is simple, the model can be extended to include different detection mechanisms and various incident-response policies for different types of attacks. A similar work [215], utilizing SDN and NFV, proactively detects (low-level deep packet inspection) and mitigates botnets in 5G mobile networks by dynamic reconfiguration (isolating botnet communication).

5 Aims of the Ph.D. Thesis

Referring to the challenges stated in Subsection 1.2 and the overall research objective "**How to increase the network communication security of computer networks with a focus on future-oriented automotive infrastructure by establishing an anomaly-based incident detection to mitigate/counteract novel malicious activity propagation?**", this section deals with the aims of the future Ph.D. thesis. Most of the state of the art research are targeted towards the office IT or cloud environments having a large amount of resources available. However, to the best of the authors knowledge, no system with the focus of this work is aimed to automotive environments which poses special requirements towards e.g. resource-saving and safety. Novel attack detection based on the output of anomaly-based methods is still an unsatisfactory solved problem and will be targeted in the thesis. Furthermore, most of the aforementioned frameworks designed for incident detection, analysis and response cover various fields but each solution has its merits and demerits such that an adaption of one of them is not feasible. However, building a complete new framework is not efficient such that in the Ph.D. thesis existing works are evaluated against the requirements and integral parts are combined and extended. The resulting generic **An**omaly-based Incident **De**tection and **R**esponse **S**ystem, called **ANDERS** (*subject to change*), consists of components that are, depending on their application level with respect to an abstracted system hierarchy (as shown in Figure 27 - cf. mist-fog-cloud hierarchy [216, 217]), leveraging different capability in terms of incident detection, analysis and response. In the Ph.D. thesis, the conceptional design and interactions of the components will be provided according to requirements defined. Exemplary properties of ANDERS are listed below (*subject to change*).

- **Property 1: Modularity/Flexibility/Interoperability** - generic framework application with different capability; interfaces allowing to exchange modules/components e.g. lightweight IDMEF/IRMEF; various techniques can be applied to provide security incidents
- **Property 2: Scalability** - local or decentralized appliance
- **Property 3: Determinism** - timeliness; fast response for real time systems; online detection in time-efficient manner within a temporal window adequate for the underlying system; on-the-fly analysis and proactive/reactive response during malicious activity propagation
- **Property 4: Effectiveness/Correctness** - detection of attacks based on detected anomalies; decrease false positives / negatives and increase detection accuracy
- **Property 5: Efficiency** - resource conservation by e.g. partial working and sampling aimed for embedded use; saving computation and communication cost; preventing flooding of alert information

- **Property 6: Preciseness** - recognition of anomalies/attacks near the origin when considering the hierarchical location of modules (root cause)
- **Property 7: Predictability** - predicting the spread of malicious activity and forecast the next steps of an adversary; next steps of a multi-staged attack; stochastic propagation prediction to forecast the boundary level with respect to the abstracted hierarchy; for response techniques considering small- and large-scale environments
- **Property 8: Robustness** - stability; noise in the data should not affect the output e.g. in the presence of attacks they should be detected as well; coping with high dimensional data; reliable operation facing high throughput
- **Property 9: Adaptability** - each module can be self-regulating or regulated by other modules e.g. the adjustment of sampling rates in order to lower the resource consumption linked to the detection of incidents; consideration of QoS; "hyperparameter" adjustment e.g. the algorithm-specific parameter or feature set; response activation and deactivation by considering the rate of attack or network risk tolerance
- **Property 10: Advancement** - detection of high sophisticated attack scenarios e.g. APTs; understanding attack behavior/actions (kill chain); detecting zero day exploits, new attack strategies and dealing with hypothesizing
- **Property 11: Responsiveness** - (semi-)automatic reaction to incidents or attacks; combining proactive IPS and reactive IRS capability; attack context-aware response selection
- **Property 12: Safeness** - safety-conform response cost computation and response execution selection; consideration and combination of automotive HARA and TARA information
- **Property 13: Interdisciplinarity** - possibility to include various other disciplines, e.g. risk assessment, vulnerability information, social analysis

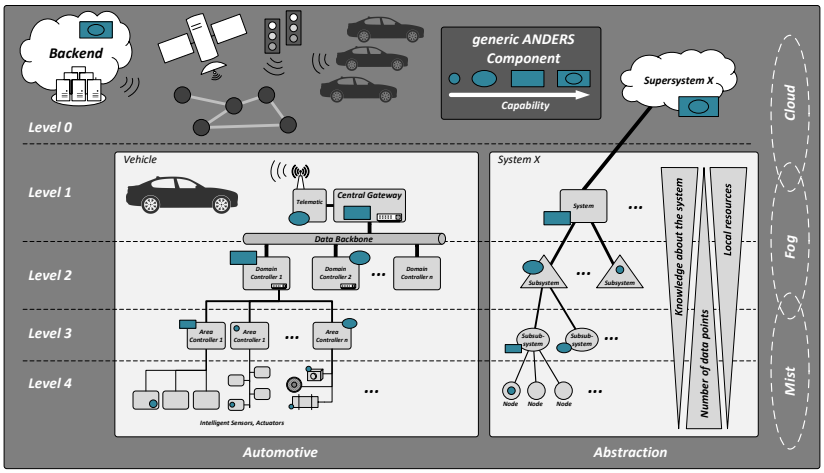

Figure 27: Abstracted next generation automotive network infrastructure with exemplary placed ANDERS components

An exemplary architecture for a generic ANDERS component is shown in Figure 28. A detailed description will be given in the Ph.D. thesis. However, a focus with respect to the overall research objective lies in an adaptive A-NIDS and the derivation of (multi-staged)(novel) attack identification utilizing methods among others from alert correlation as well as predicting the propagation of malicious activity in order to select a suitable response. Applying methods for root cause identification and response cost, e.g. risk assessment, helps to select appropriate countermeasures. Since ANDERS components might have different capability stages, the communication with e.g. human experts (security administrator), the infrastructure or other ANDERS components is a core feature.

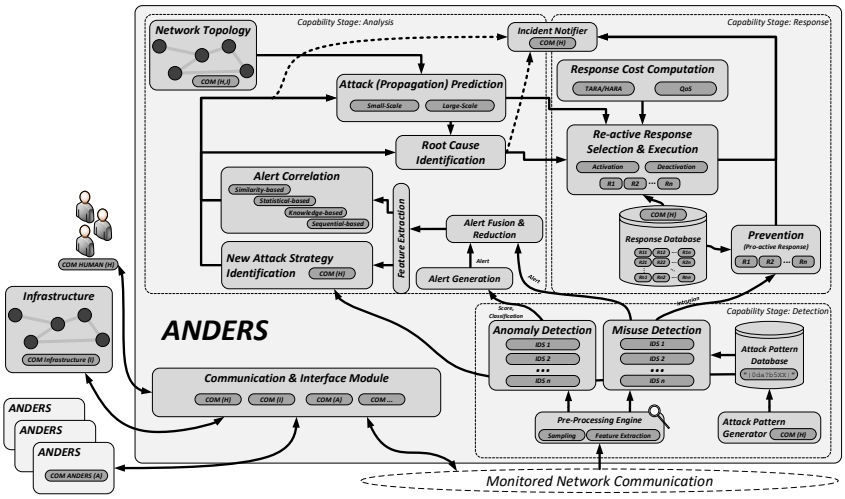

Figure 28: Architecture of a generic ANDERS component

The resulting prototype system will be embedded in a SDN-infrastructure, similar to [214], since it allows an immediate and scalable implementation of the next generation incident handling framework in the network infrastructure by integrating an ANDERS component directly in the network forwarding elements as a kind of NFV-function for detection, analysis and response. SDN is a key future technology and can not only be used for future in-vehicle network communication, e.g. [218, 219, 220], or for vehicular ad-hoc networks [221, 222, 223, 224, 225, 226] but also in combination with other trends or benefiting from them such as IoT, 5G, Industry 4.0 [217, 227, 228]. A possible use case for the application of ANDERS components in a hierarchical SDN-based vehicular ad-hoc network communication is shown in Figure 29. There, a novel malicious hacker activity propagating over various network hops will be detected by a low-level ANDERS component utilizing anomaly-based methods which provide alert information to the next ANDERS components in the hierarchical topology. Those are then performing attack strategy identification, predict the propagation and identify the root

cause in order that, at latest, the ANDERS component close to the victim is able to respond to the attack. It is desired that the ANDERS framework is mitigating malicious activity as close as possible to the root cause.

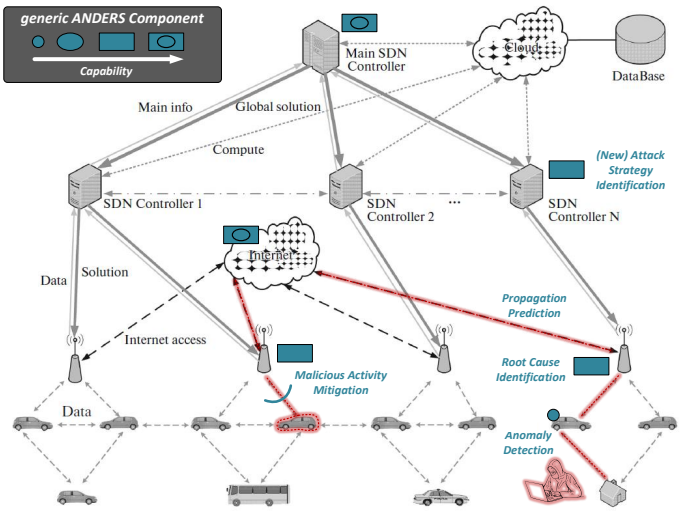

Figure 29: Possible ANDERS use case of malicious activity propagation in a SDN-based vehicular ad-hoc network (cf. [226])

To achieve the aims of the Ph.D., the overall objective needs to be broken down into research questions that will be discussed in the thesis. Candidate (exemplary) research questions are listed as follows (*subject to change*).

- **Research Question 1:** How must the (non-)functional requirements of the proposed next generation incident handling framework be formulated?
- **Research Question 2:** How do the integral parts of the system design, satisfying the requirements, look like?
- **Research Question 3:** How can adaptive A-NIDSs be improved over runtime by the incorporation of feedback?
- **Research Question 4:** How can the performance of a (new) attack identification method based on A-NIDSs reliably be evaluated considering the characteristic that real-life alerts are cursed with false positives and false negatives and do not map to a certain attack (100% confidence)?
- **Research Question 5:** How does exchanged propagation information look like (cf. IDIP) and can be used to improve micro- and macroscopig propagation prediction?
- **Research Question 6:** How can intra- and inter-network root cause analysis be leveraged to improve the overall system functionality?
- **Research Question 7:** How does a safety-tailored response cost and selection look like and be leveraged for proactive/reactive response execution?
- **Research Question 8:** How does network topology information benefit the individual components with respect to their different capability stages in a centralized and decentralized orchestration?

6 Conclusion & Future Work

Future IT systems will be exposed to increasing risks due to new emerging technologies and trends with blurring borders (keyword "IoT-ification") associated with novel and highly advanced attack possibilities, e.g. AI-driven malware. Cryptography alone can not postulate protection for network communication in the future wherefore advanced and intelligent mechanisms need to be developed for cyber defense. In this report comprehensive state-of-the-art knowledge on (automated) incident handling including detection, analysis and response has been provided as well as open issues pointed out for proactive/reactive security in network communication protection. The stated fields contain various and diverse disciplines each and discussed work aimed towards unified incident handling frameworks do not cover all aspects sufficiently. Thus, the overall objective has and exemplary research questions of the future Ph.D. thesis have been proposed including requirements and a conceptional illustration of the application for ANDERS.

As part of future work, the overall objective need to be sharpened and the research questions formulated. Since this research field is highly dynamic, the state of the art must continuously be tracked besides. The framework's requirements, derived from the properties, must by sharpened and evaluated against the comprehensive state-of-the-art work (cf. [94, 102, 229, 230]) such that the elementary components and benefits of the novel concept crystallize out. Future work needs to broken down in the following research tasks to achieve the goal (which might result in individual publications).

- **Future Work 1:** Improving adaptive A-NIDS components over runtime using a weighting-based approach being still efficient to achieve e.g. the best feature, parameter or algorithm set for the underlying network incorporating feedback from an analysis component (Exemplary architecture with A-NIDS components is shown in Figure 30).
- **Future Work 2:** On-the-fly malicious activity propagation prediction in small- and large-scale environments using online parameter estimation to update e.g. agent/graph-based parameters (small-scale view) or (stochastic) differential equations (large-scale view).
- **Future Work 3:** Implementation of a (new) attack strategy identification module tailored to A-NIDS by comparing patterns to known ones under consideration of hypothesizing.
- **Future Work 4:** Realizing a testbed for the evaluation of (new) attack strategy identification modules based on the output and impact of various A-NIDS utilizing common data sets (Exemplary architecture is shown in Figure 31).
- **Future Work 5:** Response cost computation under consideration of (automotive) HARA/TARA and response selection based on identified root cause (intra- and inter-network).

- **Future Work 6:** Establishing an evaluation environment, e.g. simulated using Mininet, for SDN-based in-vehicle networking [219, 220] and extended with a vehicular ad-hoc networking infrastructure [226, 231] that can be used to integrate the framework's components, similarly to MiniCPS [214], in order to comprehensively test the detection, analysis and reaction functionality.

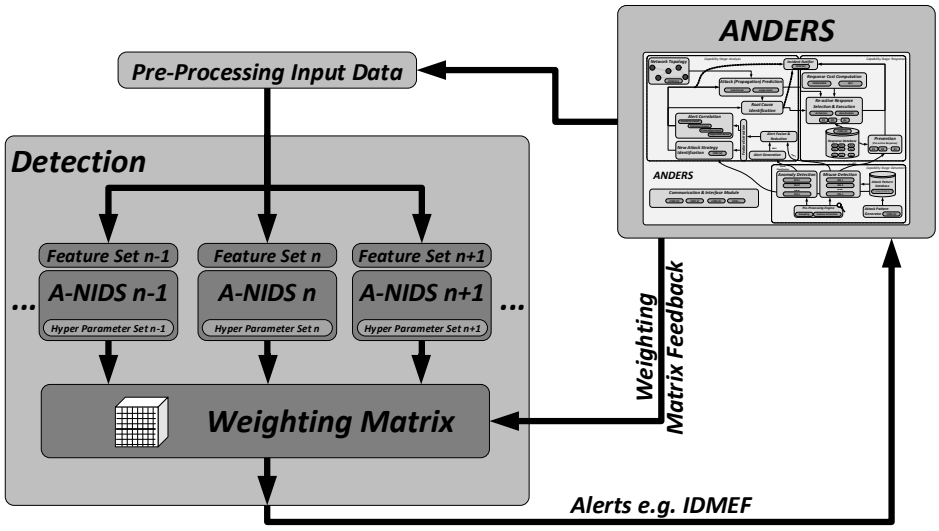

Figure 30: Improving anomaly-based detection results over runtime using a feedback approach

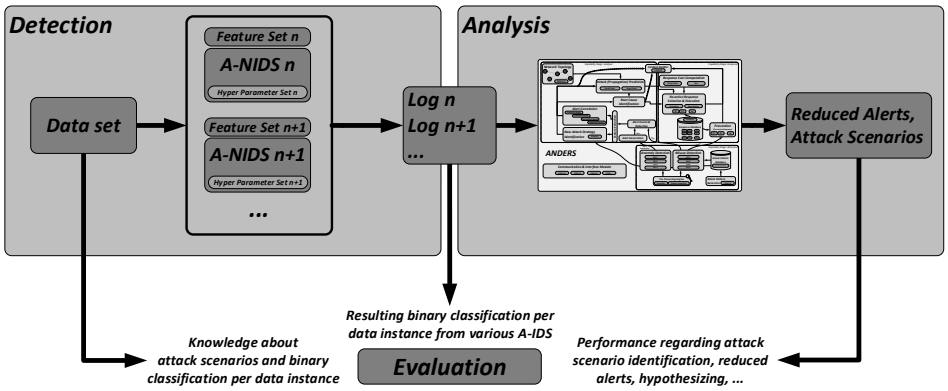

Figure 31: Conception of the proposed testbed for (new) attack identification based on anomaly detection algorithms

Author's Publications

Talks

Karl Leidl, Martin Aman, Michael Heigl, Andreas Grzemba, *Intrusion Detection Sensoren für industrielle Netzwerke*, CYBICS - Cyber Security for Industrial Control Systems, Würzburg, Germany, June 2016

Laurin Doerr, Michael Heigl, Andreas Grzemba, Christian Boiger, *IT-Security-Architektur für Next-Generation Kommunikationssysteme im Automobil*, VDI/VW-Gemeinschaftstagung: Fahrerassistenzsysteme und automatisiertes Fahren, Wolfsburg, Germany, November 2016

Michael Heigl, Karl Leidl, *An Approach to an Embedded Anomaly-Based IDS on the Example of SOME/IP*, 3rd Vector Testing Symposium, Stuttgart, Germany, Mai 2017

Michael Heigl, *Distributed Embedded Incident Detection and Response Mechanisms*, ProtectIT Security Convention 2017 (ProSecCon'17), Technische Hochschule Deggendorf, Deggendorf, Germany, July 2017

Michael Heigl, *Decentralized Anomaly Detection*, Tag der Forschung 2018, Technische Hochschule Deggendorf, Deggendorf, Germany, March 2018

Conference Papers

Michael Heigl, Martin Aman, Andreas Fuchs, Andreas Grzemba, *Securing Industrial Legacy System Communication Through Interconnected Embedded Plug-In Devices*, Applied Research Conference, pp. 501-508, ISBN 978-3-86460-494-2, Pro BUSINESS GmbH, Augsburg, Germany, June 2016

Michael Heigl, Martin Schramm, Laurin Doerr, Andreas Grzemba, *Embedded Plug-In Devices to Secure Industrial Network Communications*, 2016 International Conference on Applied Electronics (AE), pp. 85-88, `https://doi.org/10.1109/AE.2016.7577247`, Pilsen, September 2016

Martin Schramm, Reiner Dojen, Michael Heigl, *Experimental Assessment of FIRO- and GARO-based Noise Sources for Digital TRNG Designs on FPGAs*, 2017 International Conference on Applied Electronics (AE), pp. 221-226, `https://doi.org/10.23919/AE.2017.8053618`, Pilsen, September 2017

Laurin Doerr, Michael Heigl, Dalibor Fiala, Martin Schramm, *Assessment Simulation Model for Uncoupled Message Authentication*, 2017 International Conference on Applied Electronics (AE), pp. 45-48, `https://doi.org/10.23919/AE.2017.8053580`, Pilsen, September 2017

Michael Heigl, Laurin Doerr, Amar Almaini, Dalibor Fiala, Martin Schramm, *Incident Reaction Based on Intrusion Detections' Alert Analysis*, 2018 International Conference on Applied Electronics (AE), pp. 45-50, https://doi.org/10.23919/AE.2018.8501419, Pilsen, September 2018

Michael Heigl, Martin Schramm, Dalibor Fiala, *A Lightweight Quantum-Safe Security Concept for Wireless Sensor Network Communication*, 2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), pp. 906-911, https://doi.org/10.1109/PERCOMW.2019.8730749, Kyoto, Japan, March 2019

Michael Heigl, Laurin Doerr, Martin Schramm, Dalibor Fiala, *On the Energy Consumption of Quantum-Resistant Cryptographic Implementations Suitable for Wireless Sensor Networks*, Proceedings of the International Conference on Security and Cryptography - SECRYPT, Prague, Czech Republic, July 2019 [accepted]

Laurin Doerr, Michael Heigl, Dalibor Fiala, Martin Schramm, *Comparison of Energy-Efficient Key Management Protocols for Wireless Sensor Networks*, 2019 International Electronics Communication Conference (IECC 2019), ISBN 978-1-4503-7177-3, Okinawa, Japan, July 2019 [accepted]

Journal Articles

Nari S. Arunraj, Robert Hable, Michael Fernandes, Karl Leidl, Michael Heigl, *Comparison of Supervised, Semi-supervised and Unsupervised Learning Methods in Network Intrusion Detection System (NIDS) Application*, Anwendungen und Konzepte der Wirtschaftsinformatik (AKWI), [S.l.], n. 6, pp. 10-19, ISSN 2296-4592, November 2017. https://ojs-hslu.ch/ojs302/index.php/AKWI/article/view/89.

Martin Schramm, Reiner Dojen, and Michael Heigl, *A Vendor-Neutral Unified Core for Cryptographic Operations in GF(p) and GF(2^m) Based on Montgomery Arithmetic*, Security and Communication Networks, vol. 2018, Article ID 4983404, June 2018. https://doi.org/10.1155/2018/4983404.

Michael Heigl, Laurin Doerr, Nicolas Tiefnig, Dalibor Fiala, Martin Schramm, *A resource-preserving self-regulating Uncoupled MAC algorithm to be applied in incident detection*, Computers & Security, Volume 85, pp. 270-287, ISSN 0167-4048, Mai 2019, https://doi.org/10.1016/j.cose.2019.05.010.

References

[1] T. R. Vittor, T. Sukumara, S. D. Sudarsan, and J. Starck, "Cyber security - security strategy for distribution management system and security architecture considerations," *2017 70th Annual Conference for Protective Relay Engineers (CPRE)*, pp. 1–6, 2017.

[2] G. Press, "60 cybersecurity predictions for 2019," *Forbes, [online] https://www.forbes.com/sites/gilpress/2018/12/03/60-cybersecurity-predictions-for-2019*, [22.07.2019].

[3] H. Hindy, D. Brosset, E. Bayne, A. Seeam, C. Tachtatzis, R. C. Atkinson, and X. J. A. Bellekens, "A taxonomy and survey of intrusion detection system design techniques, network threats and datasets," *arXiv preprint arXiv:1806.03517*, vol. abs/1806.03517, 2018.

[4] P. Nespoli, D. Papamartzivanos, F. G. Mármol, and G. Kambourakis, "Optimal countermeasures selection against cyber attacks: A comprehensive survey on reaction frameworks," *IEEE Communications Surveys Tutorials*, vol. 20, no. 2, pp. 1361–1396, 2018.

[5] A. Sadighian, "Intrusion detection from heterogenous sensors," *Ph.D. thesis, Polytechnique Montreal, https://publications.polymtl.ca/1702/1/2015_AlirezaSadighian.pdf*, 2015.

[6] A. A. Ramaki, A. Rasoolzadegan, and A. G. Bafghi, "A systematic mapping study on intrusion alert analysis in intrusion detection systems," *ACM Computing Surveys*, vol. 51, no. 3, pp. 1–41, 2018.

[7] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Network traffic anomaly detection techniques and systems," *Network Traffic Anomaly Detection and Prevention*, pp. 115–169, 2017.

[8] ISO/IEC 27000:2018, "Information technology – security techniques – information security management systems – overview and vocabulary," *International Organization for Standardization, [online] https://www.iso.org/standard/73906.html*, [15.06.2019].

[9] Z. Inayat, A. Gani, N. B. Anuar, S. Anwar, and M. K. Khan, "Cloud-based intrusion detection and response system: Open research issues, and solutions," *Arabian Journal for Science and Engineering*, vol. 42, no. 2, pp. 399–423, 2017.

[10] S. Anwar, J. Mohamad Zain, M. F. Zolkipli, Z. Inayat, S. Khan, B. Anthony, and V. Chang, "From intrusion detection to an intrusion response system: Fundamentals, requirements, and future directions," *Algorithms*, vol. 10, no. 2, 2017.

[11] A. Shameli-Sendi, M. Cheriet, and A. Hamou-Lhadj, "Taxonomy of intrusion risk assessment and response system," *Computers & Security*, vol. 45, pp. 1–16, 2014.

[12] M. Heigl, L. Doerr, A. Almaini, D. Fiala, and M. Schramm, "Incident reaction based on intrusion detections' alert analysis," *2018 International Conference on Applied Electronics (AE)*, pp. 1–6, 2018.

[13] L. T. Heberlein, G. V. Dias, K. N. Levitt, B. Mukherjee, J. Wood, and D. Wolber, "A network security monitor," *1990 IEEE Computer Society Symposium on Research in Security and Privacy*, pp. 296–304, 1990.

[14] A. N. Jaber, M. F. Zolkipli, H. A. Shakir, and M. R. Jassim, "Host based intrusion detection and prevention model against DDoS attack in cloud computing," *Advances on P2P, Parallel, Grid, Cloud and Internet Computing*, pp. 241–252, 2018.

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

[16] Z. Inayat, A. Gani, N. B. Anuar, M. K. Khan, and S. Anwar, "Intrusion response systems: Foundations, design, and challenges," *Journal of Network and Computer Applications*, vol. 62, pp. 53–74, 2016.

[17] M.-Y. Su, "Using clustering to improve the KNN-based classifiers for online anomaly network traffic identification," *Journal of Network and Computer Applications*, vol. 34, no. 2, pp. 722–730, 2011.

[18] B. B. Zarpelão, R. S. Miani, C. T. Kawakani, and S. C. de Alvarenga, "A survey of intrusion detection in internet of things," *Journal of Network and Computer Applications*, vol. 84, no. January, pp. 25–37, 2017.

[19] R. A. R. Ashfaq, X.-Z. Wang, J. Z. Huang, H. Abbas, and Y.-L. He, "Fuzziness based semi-supervised learning approach for intrusion detection system," *Information Sciences*, vol. 378, pp. 484–497, 2017.

[20] A. Taylor, "Anomaly-based detection of malicious activity in in-vehicle networks," *University of Ottawa, https://ruor.uottawa.ca/bitstream/10393/36120/3/Taylor_Adrian_2017_thesis.pdf*, 2017.

[21] I. Butun, S. D. Morgera, and R. Sankar, "A survey of intrusion detection systems in wireless sensor networks," *IEEE Communications Surveys Tutorials*, vol. 16, no. 1, pp. 266–282, 2014.

[22] A. M. Ahmed, "Online network intrusion detection system using temporal logic and stream data processing," *Ph.D. thesis, University of Liverpool, https://livrepository.liverpool.ac.uk/12153/1/AbdulbasitAhmed_June2013_12153.pdf*, 2013.

[23] E. Vasilomanolakis, S. Karuppayah, M. Mühlhäuser, and M. Fischer, "Taxonomy and survey of collaborative intrusion detection," *ACM Comput. Surv.*, vol. 47, no. 4, pp. 55:1–55:33, 2015.

[24] Z. Zhang, J. Li, C. Manikopoulos, J. Jorgenson, and J. Ucles, "HIDE: a hierarchical network intrusion detection system using statistical preprocessing and neural network classification," *In Proc. IEEE Workshop on Information Assurance and Security*, pp. 85–90, 2001.

[25] M. Jahnke, "An open and secure infrastructure for distributed intrusion detection sensors," *In Proceedings of the NATO Regional Conference on Communication and Information Systems (RCMCIS'02), Zegrze, Poland*, 2002.

[26] H. Debar, D. Curry, and B. Feinstein, "The intrusion detection message exchange format (IDMEF)," *RFC 4765, RFC Editor, https://www.ietf.org/rfc/rfc4765.txt*, 2007.

[27] R. Lupu, R. Badea, and I. Cosmin Mihai, "Agent-based IDMEF alerting infrastructure for distributed intrusion detection and prevention systems: Design and validation," *In 2016 International Conference on Communications (COMM)*, pp. 281–284, 2016.

[28] W. Hu, J. Gao, Y. Wang, O. Wu, and S. Maybank, "Online adaboost-based parameterized methods for dynamic distributed network intrusion detection," *IEEE Transactions on Cybernetics*, pp. 66–82, 2014.

[29] J. D. Parmar and J. T. Patel, "Anomaly detection in data mining: A review," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 7, no. 4, pp. 32–40, 2017.

[30] S. Omar, A. Ngadi, and H. H. Jebur, "Machine learning techniques for anomaly detection: An overview," *International Journal of Computer Applications*, vol. 79, no. 2, pp. 33–41, 2013.

[31] K. Kuźniar and M. Zając, "Some methods of pre-processing input data for neural networks," *Computer Assisted Methods in Engineering and Science*, vol. 22, pp. 141–151, 2015.

[32] S. Ramírez-Gallego, B. Krawczyk, S. García, M. Woźniak, and F. Herrera, "A survey on data preprocessing for data stream mining: Current status and future directions," *Neurocomputing*, vol. 239, pp. 39–57, 2017.

[33] J. Singh Malik, P. Goyal, and M. K. Sharma, "A comprehensive approach towards data preprocessing techniques, & association rules," *Proceedings of the 4th National Conference INDIACom*, pp. 12–21, 2010.

[34] D. Tomar and S. Agarwal, "A survey on pre-processing and post-processing techniques in data mining," *International Journal of Database Theory and Application*, vol. 7, no. 4, pp. 99–128, 2014.

[35] B. K. Singh, K. Verma, and A. S. Thoke, "Investigations on impact of feature normalization techniques on classifier's performance in breast tumor classification," *International Journal of Computer Applications*, vol. 116, no. 19, pp. 11–15, 2015.

[36] H. Xie, J. Li, and H. Xue, "A survey of dimensionality reduction techniques based on random projection," *arXiv preprint arXiv:1706.04371*, vol. abs/1706.04371, 2017.

[37] N. Lim, R. J. Durrant, and N. Zealand, "Linear dimensionality reduction in linear time: Johnson-Lindenstrauss-type guarantees for random subspace," *arXiv preprint arXiv:1705.06408v1*, 2017.

[38] S. M. Ghaffarian and H. R. Shahriari, "Software vulnerability analysis and discovery using machine-learning and data-mining techniques: A survey," *ACM Computing Surveys*, vol. 50, no. 4, pp. 56:1–56:36, 2017.

[39] M. E. Aminanto, R. Choia, H. C. Tanuwidjaja, P. D. Yoo, and K. Kim, "Deep abstraction and weighted feature selection for Wi-Fi impersonation detection," *IEEE Transactions on Information Forensics and Security 13*, pp. 621–636, 2018.

[40] F. Iglesias and T. Zseby, "Analysis of network traffic features for anomaly detection," *Machine Learning*, vol. 101, no. 1, pp. 59–84, 2015.

[41] W. Wang, Y. Sheng, J. Wang, X. Zeng, X. Ye, Y. Huang, and M. Zhu, "HAST-IDS: Learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection," *IEEE Access*, vol. 6, pp. 1792–1806, 2018.

[42] X. He, D. Cai, and P. Niyogi, "Laplacian score for feature selection," *Advances in neural information processing systems*, pp. 507–514, 2006.

[43] R. Wieland, A. Kerkow, L. Früh, H. Kampen, and D. Walther, "Automated feature selection for a machine learning approach toward modeling a mosquito distribution," *Ecological Modelling*, vol. 352, pp. 108–112, 2017.

[44] M. Luo, F. Nie, X. Chang, Y. Yang, A. G. Hauptmann, and Q. Zheng, "Adaptive unsupervised feature selection with structure regularization," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 4, pp. 944–956, 2018.

[45] S. Aljawarneh, M. Aldwairi, and M. B. Yassein, "Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model," *Journal of Computational Science*, vol. 25, pp. 152–160, 2018.

[46] W. Khreich, E. Granger, A. Miri, and R. Sabourin, "Adaptive ROC-based ensembles of HMMs applied to anomaly detection," *Pattern Recognition*, vol. 45, no. 1, pp. 208–230, 2012.

[47] P. Casas, J. Mazel, and P. Owezarski, "Unsupervised network intrusion detection systems: Detecting the unknown without knowledge," *Computer Communications*, vol. 35, no. 7, pp. 772–783, 2012.

[48] J. Zhang, C. Chen, Y. Xiang, and W. Zhou, "Semi-supervised and compound classification of network traffic," *32nd International Conference on Distributed Computing Systems Workshops*, vol. 7, no. 4, pp. 252–261, 2012.

[49] J. Zhang and M. Zulkernine, "A hybrid network intrusion detection technique using random forests," *First International Conference on Availability, Reliability and Security (ARES'06)*, pp. 262–269, 2006.

[50] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 15:1–15:58, 2009.

[51] I. Friedberg, F. Skopik, G. Settanni, and R. Fiedler, "Combating advanced persistent threats: From network event correlation to incident detection," *Computers & Security*, vol. 48, pp. 35–57, 2015.

[52] F. T. Liu, K. M. Ting, and Z. Zhou, "Isolation forest," *2008 Eighth IEEE International Conference on Data Mining*, pp. 413–422, 2008.

[53] T. Pevný, "Loda: Lightweight on-line detector of anomalies," *Machine Learning*, vol. 102, no. 2, pp. 275–304, 2016.

[54] Z. Ding and M. Fei, "An anomaly detection approach based on isolation forest algorithm for streaming data using sliding window," *IFAC Proceedings Volumes*, vol. 46, pp. 12–17, 2013.

[55] B. Pfahringer, G. Holmes, and R. Kirkby, "New Options for Hoeffding Trees," *AI 2007: Advances in Artificial Intelligence*, pp. 90–99, 2007.

[56] L. Sun, S. Versteeg, S. Boztas, and A. Rao, "Detecting anomalous user behavior using an extended isolation forest algorithm: An enterprise case study," *arXiv preprint arXiv:1609.06676*, vol. abs/1609.06676, 2016.

[57] P. Li, T. J. Hastie, and K. W. Church, "Very sparse random projections," *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 287–296, 2006.

[58] P. Amudha, S. Karthik, and S. Sivakumari, "Classification techniques for intrusion detection - an overview," *International Journal of Computer Applications*, vol. 76, no. 16, pp. 33–40, 2013.

[59] W. Hu, W. Hu, and S. Maybank, "Adaboost-based algorithm for network intrusion detection," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 38, no. 2, pp. 577–583, 2008.

[60] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: an ensemble of autoencoders for online network intrusion detection," *arXiv preprint arXiv:1802.09089*, 2018.

[61] J. Kittler, M. Hatef, R. P. Duin, and J. Matas, "On combining classifiers," *IEEE transactions on pattern analysis and machine intelligence*, vol. 20, no. 3, pp. 226–239, 1998.

[62] B. Parhami, "Voting algorithms," *IEEE Transactions on reliability*, vol. 43, no. 4, pp. 617–629, 1994.

[63] V. Hodge and J. Austin, "A survey of outlier detection methodologies," *Artificial intelligence review*, vol. 22, no. 2, pp. 85–126, 2004.

[64] J. Gao, W. Fan, D. Turaga, O. Verscheure, X. Meng, L. Su, and J. Han, "Consensus extraction from heterogeneous detectors to improve performance over network traffic anomaly detection," *INFOCOM, 2011 Proceedings IEEE*, pp. 181–185, 2011.

[65] Y.-D. Lin, Y.-C. Lai, C.-Y. Ho, and W.-H. Tai, "Creditability-based weighted voting for reducing false positives and negatives in intrusion detection," *Computers & Security*, vol. 39, pp. 460–474, 2013.

[66] G. Giacinto, F. Roli, and L. Didaci, "Fusion of multiple classifiers for intrusion detection in computer networks," *Pattern recognition letters*, vol. 24, no. 12, pp. 1795–1803, 2003.

[67] A. A. Aburomman and M. B. I. Reaz, "A survey of intrusion detection systems based on ensemble and hybrid classifiers," *Computers & Security*, vol. 65, pp. 135–152, 2017.

[68] F. Cheng and X. Qiu, "Network anomaly detection based on frequent sub-graph mining approach and association analysis," *2016 IEEE International Conference on Network Infrastructure and Digital Content (IC-NIDC)*, pp. 12–16, 2016.

[69] Y. Liu, H. Xu, H. Yi, Z. Lin, J. Kang, W. Xia, Q. Shi, Y. Liao, and Y. Ying, "Network anomaly detection based on dynamic hierarchical clustering of cross domain data," *2017 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, pp. 200–204, 2017.

[70] X. Zhao, G. Wang, and Z. Li, "Unsupervised network anomaly detection based on abnormality weights and subspace clustering," *In 2016 Sixth International Conference on Information Science and Technology (ICIST)*, pp. 482–486, 2016.

[71] Y. Maleh, A. Ezzati, Y. Qasmaoui, and M. Mbida, "A global hybrid intrusion detection system for wireless sensor networks," *Procedia Computer Science*, vol. 52, pp. 1047–1052, 2015.

[72] M. Yassine and A. Ezzati, "Lightweight intrusion detection scheme for wireless sensor networks," *IAENG International Journal of Computer Science*, vol. 42, pp. 347–354, 2015.

[73] C. Guo, Y. Ping, N. Liu, and S.-S. Luo, "A two-level hybrid approach for intrusion detection," *Neurocomputing*, vol. 214, pp. 391–400, 2016.

[74] M. Weber, S. Klug, E. Sax, and B. Zimmer, "Embedded hybrid anomaly detection for automotive can communication," *9th European Congress on Embedded Real Time Software and Systems (ERTS 2018)*, 2018.

[75] L. A. Maglaras, J. Jiang, and T. J. Cruz, "Combining ensemble methods and social network metrics for improving accuracy of OCSVM on intrusion detection in SCADA systems," *Journal of Information Security and Applications*, vol. 30, pp. 15–26, 2016.

[76] G. Gu, P. Fogla, D. Dagon, W. Lee, and B. Skorić, "Measuring intrusion detection capability: An information-theoretic approach," *Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security*, pp. 90–101, 2006.

[77] T. Holz, "Security measurements and metrics for networks," *Dependability Metrics: Advanced Lectures, Springer Berlin Heidelberg*, pp. 157–165, 2008.

[78] D. Ashok Kumar and S. R. Venugopalan, "A novel algorithm for network anomaly detection using adaptive machine learning," *Progress in Advanced Computing and Intelligent Engineering*, pp. 59–69, 2018.

[79] S. Ossenbuehl, J. Steinberger, and H. Baier, "Towards automated incident handling: How to select an appropriate response against a network-based attack?" *2015 Ninth International Conference on IT Security Incident Management IT Forensics*, pp. 51–67, 2015.

[80] N. S. Arunraj, R. Hable, M. Fernandes, K. Leidl, and M. Heigl, "Comparison of Supervised, Semi-supervised and Unsupervised Learning Methods in Network Intrusion Detection System (NIDS) Application," *Anwendungen und Konzepte der Wirtschaftsinformatik (AKWI)*, vol. 6, pp. 10–19, 2017.

[81] G. Suarez-Tangil, E. Palomar, J. M. de Fuentes, J. Blasco, and A. Ribagorda, "Automatic rule generation based on genetic programming for event correlation," *Computational Intelligence in Security for Information Systems*, pp. 127–134, 2009.

[82] K.-T. Cho and K. G. Shin, "Fingerprinting electronic control units for vehicle intrusion detection," *25th USENIX Security Symposium (USENIX Security 16)*, pp. 911–927, 2016.

[83] J. Steinberger, A. Sperotto, M. Golling, and H. Baier, "How to exchange security events? overview and evaluation of formats and protocols," *Proceedings of the 2015 IFIP/IEEE International Symposium on Integrated Network Management, IM 2015*, pp. 261–269, 2015.

[84] M. Kerrisk, "The linux programming interface," *Muenchen: No Starch-Press*, 2010.

[85] J. Postel, "Internet protocol," *RFC 791, RFC Editor, https://www.ietf.org/rfc/rfc791.txt*, 1981.

[86] R. Koch, M. Golling, and G. Dreo, "Evaluation of state of the art IDS message exchange protocols," *International Journal of Computer and Systems Engineering*, vol. 7, pp. 1017–1026, 2013.

[87] CESNET, "Warden - a system for efficient sharing information about detected events (threats)," *[online] https://warden.cesnet.cz/en/index*, [22.07.2019].

[88] MISP-Project, "MISP - Malware Information Sharing Platform," *[online] https://www.misp-project.org/*, [22.07.2019].

[89] CESNET, "IDEA - Intrusion Detection Extensible Alert," *[online] https://idea.cesnet.cz/en/index*, [22.07.2019].

[90] A. AlEroud and G. Karabatis, "Beyond data: Contextual information fusion for cyber security analytics," *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, pp. 73–79, 2016.

[91] J. Haines, D. K. Ryder, L. Tinnel, and S. Taylor, "Validation of sensor alert correlators," *IEEE Security & Privacy*, vol. 99, no. 1, pp. 46–56, 2003.

[92] G. Gu, A. A. Cárdenas, and W. Lee, "Principled reasoning and practical applications of alert fusion in intrusion detection systems," *Proceedings of the 2008 ACM Symposium on Information, Computer and Communications Security*, pp. 136–147, 2008.

[93] W. Meng, Y. Wang, W. Li, Z. Liu, J. Li, and C. W. Probst, "Enhancing intelligent alarm reduction for distributed intrusion detection systems via edge computing," *Information Security and Privacy*, pp. 759–767, 2018.

[94] S. Salah, G. Maciá-Fernández, and J. E. Díaz-Verdejo, "A model-based survey of alert correlation techniques," *Computer Networks*, vol. 57, no. 5, pp. 1289–1317, 2013.

[95] N. Hubballi and V. Suryanarayanan, "False alarm minimization techniques in signature-based intrusion detection systems: A survey," *Computer Communications*, vol. 49, pp. 1–17, 2014.

[96] F. Valeur, G. Vigna, C. Kruegel, and R. A. Kemmerer, "Comprehensive approach to intrusion detection alert correlation," *IEEE Transactions on dependable and secure computing*, vol. 1, no. 3, pp. 146–169, 2004.

[97] A. Siraj, "A unified alert fusion model for intelligent analysis of sensor data in an intrusion detection environment," *Ph.D. thesis, Mississippi State University*, 2006.

[98] H. T. Elshoush and I. M. Osman, "An improved framework for intrusion alert correlation," *Proceedings of the World Congress on Engineering*, vol. 1, pp. 1–6, 2012.

[99] P. Ning, Y. Cui, and D. S. Reeves, "Analyzing intensive intrusion alerts via correlation," *International Workshop on Recent Advances in Intrusion Detection*, pp. 74–94, 2002.

[100] F. Maggi and S. Zanero, "On the use of different statistical tests for alert correlation," *International Workshop on Recent Advances in Intrusion Detection*, pp. 167–177, 2007.

[101] M. GhasemiGol, A. Ghaemi-Bafghi, and H. Takabi, "A comprehensive approach for network attack forecasting," *Computers & Security*, vol. 58, pp. 83–105, 2016.

[102] S. A. Mirheidari, S. Arshad, and R. Jalili, "Alert correlation algorithms: A survey and taxonomy," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8300 LNCS, pp. 183–197, 2013.

[103] H. T. Elshoush and I. M. Osman, "Alert correlation in collaborative intelligent intrusion detection systems - a survey," *Applied Soft Computing*, vol. 11, no. 7, pp. 4349–4365, 2011.

[104] A. Valdes and K. Skinner, "Probabilistic alert correlation," *International Workshop on Recent Advances in Intrusion Detection*, pp. 54–68, 2001.

[105] X. Zhuang, D. Xiao, X. Liu, and Y. Zhang, "Applying data fusion in collaborative alerts correlation," *2008 International Symposium on Computer Science and Computational Technology*, vol. 2, pp. 124–127, 2008.

[106] W. N. Thurman, M. E. Fisher *et al.*, "Chickens, eggs, and causality, or which came first," *American journal of agricultural economics*, vol. 70, no. 2, pp. 237–238, 1988.

[107] P. Ning, Y. Cui, D. S. Reeves, and D. Xu, "Towards automating intrusion alert analysis," *2003 Workshop on Statistical and Machine Learning Techniques in Computer Intrusion Detection*, pp. 1–19, 2003.

[108] B. Zhu and A. A. Ghorbani, "Alert correlation for extracting attack strategies," *IJ Network Security*, vol. 3, no. 3, pp. 244–258, 2006.

[109] S. Saad and I. Traore, "Extracting attack scenarios using intrusion semantics," *International Symposium on Foundations and Practice of Security*, pp. 278–292, 2012.

[110] J. M. Vidal, A. L. S. Orozco, and L. J. G. Villalba, "Alert correlation framework for malware detection by anomaly-based packet payload analysis," *Journal of Network and Computer Applications*, vol. 97, pp. 11–22, 2017.

[111] X. Zhuang, D. Xiao, X. Liu, and Y. Zhang, "Applying data fusion in collaborative alerts correlation," *2008 International Symposium on Computer Science and Computational Technology*, vol. 2, pp. 124–127, 2008.

[112] F. Cuppens and R. Ortalo, "Lambda: A language to model a database for detection of attacks," *International Workshop on Recent Advances in Intrusion Detection*, pp. 197–216, 2000.

[113] S. T. Eckmann, G. Vigna, and R. A. Kemmerer, "STATL: An attack language for state-based intrusion detection," *Journal of computer security*, vol. 10, no. 1-2, pp. 71–103, 2002.

[114] S. Cheung, U. Lindqvist, and M. W. Fong, "Modeling multistep cyber attacks for scenario recognition," *Proceedings DARPA Information Survivability Conference And Exposition*, vol. 1, pp. 284–292, 2003.

[115] S. H. Ahmadinejad, S. Jalili, and M. Abadi, "A hybrid model for correlating alerts of known and unknown attack scenarios and updating attack graphs," *Computer Networks*, vol. 55, no. 9, pp. 2221–2240, 2011.

[116] C. T. Kawakani, S. B. Junior, R. S. Miani, M. Cukier, and B. B. Zarpelão, "Intrusion alert correlation to support security management," *XII Brazilian Symposium on Information Systems-Information Systems in the Cloud Computing Era*, pp. 313–320, 2016.

[117] G. Suarez-Tangil, E. Palomar, A. Ribagorda, and I. Sanz, "Providing siem systems with self-adaptation," *Information Fusion*, vol. 21, pp. 145–158, 2015.

[118] M. Soleimani and A. A. Ghorbani, "Multi-layer episode filtering for the multi-step attack detection," *Computer Communications*, vol. 35, no. 11, pp. 1368–1379, 2012.

[119] A. Ahmadian Ramaki and A. Rasoolzadegan, "Causal knowledge analysis for detecting and modeling multi-step attacks," *Security and Communication Networks*, vol. 9, no. 18, pp. 6042–6065, 2016.

[120] X. Sun, J. Dai, P. Liu, A. Singhal, and J. Yen, "Using bayesian networks for probabilistic identification of zero-day attack paths," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 10, pp. 2506–2521, 2018.

[121] P. Ning and D. Xu, "Hypothesizing and reasoning about attacks missed by intrusion detection systems," *ACM Transactions on Information and System Security*, vol. 7, no. 4, pp. 591–627, 2004.

[122] G. Tedesco and U. Aickelin, "Real-time alert correlation with type graphs," *arXiv preprint arXiv:1004.4089*, vol. abs/1004.4089, 2010.

[123] G. P. Spathoulas and S. K. Katsikas, "Enhancing IDS performance through comprehensive alert post-processing," *Computers & Security*, vol. 37, pp. 176–196, 2013.

[124] H. Fatma and M. Limam, "A two-stage process based on data mining and optimization to identify false positives and false negatives generated by intrusion detection systems," *2015 11th International Conference on Computational Intelligence and Security*, pp. 308–311, 2015.

[125] Q. Hui and W. Kun, "Real-time network attack intention recognition algorithm," *International Journal of Security and its Applications*, vol. 10, no. 4, pp. 51–62, 2016.

[126] B. D. Bryant and H. Saiedian, "A novel kill-chain framework for remote security log analysis with SIEM software," *Computers & Security*, vol. 67, pp. 198–210, 2017.

[127] M. Abdlhamed, K. Kifayat, Q. Shi, and W. Hurst, "A system for intrusion prediction in cloud computing," *Proceedings of the International Conference on Internet of Things and Cloud Computing*, pp. 35:1–35:9, 2016.

[128] E. T. Anumol, "Use of machine learning algorithms with SIEM for attack prediction," *Intelligent Computing, Communication and Devices*, pp. 231–235, 2015.

[129] A. A. Ramaki and R. E. Atani, "A survey of IT early warning systems: architectures, challenges, and solutions," *Security and Communication Networks*, vol. 9, no. 17, pp. 4751–4776, 2016.

[130] M. Apel, J. Biskup, U. Flegel, and M. Meier, "Towards early warning systems – challenges, technologies and architecture," *Critical Information Infrastructures Security*, pp. 151–164, 2010.

[131] A. A. Ramaki, M. Khosravi-Farmad, and A. G. Bafghi, "Real time alert correlation and prediction using bayesian networks," *2015 12th*

International Iranian Society of Cryptology Conference on Information Security and Cryptology (ISCISC), pp. 98–103, 2015.

[132] P. Holgado, V. A. VILLAGRA, and L. Vazquez, "Real-time multistep attack prediction based on hidden Markov models," *IEEE Transactions on Dependable and Secure Computing*, 2018.

[133] S. N. Narayanan, S. Mittal, and A. Joshi, "Using data analytics to detect anomalous states in vehicles," *arXiv preprint arXiv:1512.08048*, vol. abs/1512.08048, 2015.

[134] A. M. del Rey, "Mathematical modeling of the propagation of malware: a review," *Security and Communication Networks*, vol. 8, no. 15, pp. 2561–2579, 2015.

[135] M. Trawicki, "Deterministic SEIRS epidemic model for modeling vital dynamics, vaccinations, and temporary immunity," *Mathematics*, vol. 5, no. 1, 2017.

[136] E. Magkos, M. Avlonitis, P. Kotzanikolaou, and M. Stefanidakis, "Toward early warning against internet worms based on critical-sized networks," *Security and Communication Networks*, vol. 6, no. 1, pp. 78–88, 2013.

[137] S. Peng, S. Yu, and A. Yang, "Smartphone malware and its propagation modeling: A survey," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 2, pp. 925–941, 2013.

[138] S. H. White, A. M. Del Rey, and G. R. Sánchez, "Modeling epidemics using cellular automata," *Applied Mathematics and Computation*, vol. 186, no. 1, pp. 193–202, 2007.

[139] J. Pan and C. C. Fung, "An agent-based model to simulate coordinated response to malware outbreak within an organisation," *International Journal of Information and Computer Security*, vol. 5, no. 2, pp. 115–131, 2012.

[140] Y. El Ansari, A. El Myr, and L. Omari, "Deterministic and stochastic study for an infected computer network model powered by a system of antivirus programs," *Discrete Dynamics in Nature and Society*, vol. 2017, 2017.

[141] J. Jiang, S. Wen, B. Liu, S. Yu, Y. Xiang, and W. Zhou, *Malicious attack propagation and source identification*. Springer, 2019, vol. 73.

[142] Y. Wang, S. Wen, Y. Xiang, and W. Zhou, "Modeling the propagation of worms in networks: A survey," *IEEE Communications Surveys Tutorials*, vol. 16, no. 2, pp. 942–960, 2014.

[143] L. Xu, "Markovian and stochastic differential equation based approaches to computer virus propagation dynamics and some models for survival distributions," *Ph.D. thesis, Department of Mathematics and Computer Science, Faculty of New Jersey Institute of Technology and Rutgers, http://archives.njit.edu/vol01/etd/2010s/2011/njit-etd2011-064/njit-etd2011-064.pdf*, 2011.

[144] R.-r. Xi, X.-c. Yun, Z.-y. Hao, and Y.-z. Zhang, "Quantitative threat situation assessment based on alert verification," *Security and Communication Networks*, vol. 9, no. 13, pp. 2135–2142, 2016.

[145] A. Shameli-Sendi, M. Cheriet, and A. Hamou-Lhadj, "Taxonomy of intrusion risk assessment and response system," *Computers & Security*, vol. 45, pp. 1–16, 2014.

[146] G. Apruzzese, M. Marchetti, M. Colajanni, G. G. Zoccoli, and A. Guido, "Identifying malicious hosts involved in periodic communications," *2017 IEEE 16th International Symposium on Network Computing and Applications (NCA)*, pp. 1–8, 2017.

[147] P. A. Porras, M. W. Fong, and A. Valdes, "A mission-impact-based approach to INFOSEC alarm correlation," *Recent Advances in Intrusion Detection*, pp. 95–114, 2002.

[148] K. Alsubhi, E. Al-Shaer, and R. Boutaba, "Alert prioritization in intrusion detection systems," *NOMS 2008 - 2008 IEEE Network Operations and Management Symposium*, pp. 33–40, 2008.

[149] S. McElwee, J. Heaton, J. Fraley, and J. Cannady, "Deep learning for prioritizing and responding to intrusion detection alerts," *MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM)*, pp. 1–5, 2017.

[150] R. Shittu, A. Healing, R. Ghanea-Hercock, R. Bloomfield, and M. Rajarajan, "Intrusion alert prioritisation and attack detection using post-correlation analysis," *Computers & Security*, vol. 50, pp. 1–15, 2015.

[151] R. Kumar and B. B. Gupta, "Stepping stone detection techniques: Classification and state-of-the-art," *Proceedings of the International Conference on Recent Cognizance in Wireless Communication & Image Processing*, pp. 523–533, 2016.

[152] D. Shah and T. Zaman, "Detecting sources of computer viruses in networks: Theory and experiment," *Proceedings of the ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pp. 203–214, 2010.

[153] M. C. V. Fioriti and J. Palomo, "Predicting the sources of an outbreak with a spectral technique," *Applied Mathematical Sciences*, vol. 8(135), pp. 6775–6782, 2014.

[154] D. Shah and T. Zaman, "Rumors in a network: Who's the culprit?" *IEEE Transactions on Information Theory*, vol. 57, no. 8, pp. 5163–5181, 2011.

[155] D. Shah and T. Zaman, "Rumor centrality: A universal source detector," *Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems*, pp. 199–210, 2012.

[156] K. Zhu and L. Ying, "A robust information source estimator with sparse observations," *Computational Social Networks*, vol. 1, no. 1, p. 3, 2014.

[157] F. Altarelli, A. Braunstein, L. Dall'Asta, A. Lage-Castellanos, and R. Zecchina, "Bayesian inference of epidemics on networks via belief propagation," *Physical review letters*, vol. 112, p. 118701, 2014.

[158] P. C. Pinto, P. Thiran, and M. Vetterli, "Locating the source of diffusion in large-scale networks," *arXiv preprint arXiv:1208.2534*, vol. abs/1208.2534, 2012.

[159] Y. Xie, V. Sekar, D. A. Maltz, M. K. Reiter, and H. Zhang, "Worm origin identification using random moonwalks," *2005 IEEE Symposium on Security and Privacy (SP'05)*, pp. 242–256, 2005.

[160] K. Julisch, "Clustering intrusion detection alarms to support root cause analysis," *ACM transactions on information and system security (TISSEC)*, vol. 6, no. 4, pp. 443–471, 2003.

[161] J. Kim, G. Lee, J.-t. Seo, E.-k. Park, C.-s. Park, and D.-k. Kim, "Y-AOI: Y-means based attribute oriented induction identifying root cause for IDSs," *Fuzzy Systems and Knowledge Discovery*, pp. 205–214, 2005.

[162] Y. Guan, A. A. Ghorbani, and N. Belacel, "Y-means: a clustering method for intrusion detection," *CCECE 2003 - Canadian Conference on Electrical and Computer Engineering*, vol. 2, pp. 1083–1086, 2003.

[163] S. O. Al-Mamory and H. Zhang, "Intrusion detection alarms reduction using root cause analysis and clustering," *Computer Communications*, vol. 32, no. 2, pp. 419–430, 2009.

[164] M. Kechadi, J. H. Bellec, A. Tari *et al.*, "Behavioural proximity discovery: an adaptive approach for root cause analysis," *International Journal of Business Intelligence and Data Mining*, vol. 6, no. 3, pp. 259–282, 2011.

[165] D. Cotroneo, A. Paudice, and A. Pecchia, "Automated root cause identification of security alerts," *Future Generation Computer Systems*, vol. 56, no. C, pp. 375–387, 2016.

[166] I. Sanchez-Rola, I. Santos, and D. Balzarotti, "Clock around the clock: Time-based device fingerprinting," *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1502–1514, 2018.

[167] K.-T. Cho and K. G. Shin, "Fingerprinting electronic control units for vehicle intrusion detection," *25th USENIX Security Symposium*, pp. 911–927, 2016.

[168] K. Cho and K. Shin, "Viden: Attacker identification on in-vehicle networks," *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1109–1123, 2017.

[169] R. Shirey, "Internet security glossary," *RFC 2828, RFC Editor, https://rfc-editor.org/rfc/rfc2828.txt*, 2000.

[170] N. Herold, "Incident handling systems with automated intrusion response," *Ph.D. thesis, Technical University Munich, Germany, https://dblp.org/rec/bib/phd/dnb/Herold17*, 2017.

[171] N. Stakhanova, S. Basu, and J. Wong, "A taxonomy of intrusion response systems," *International Journal of Information and Computer Security (IJICS)*, vol. 1, 2007.

[172] C. Strasburg, N. Stakhanova, S. Basu, and J. S. Wong, "Intrusion response cost assessment methodology," *Proceedings of the 4th International Symposium on Information, Computer, and Communications Security*, pp. 388–391, 2009.

[173] A. Shameli-Sendi and M. Dagenais, "Orcef: Online response cost evaluation framework for intrusion response system," *Journal of Network and Computer Applications*, vol. 55, pp. 89–107, 2015.

[174] G. Gonzalez Granadillo, H. Débar, G. Jacob, C. Gaber, and M. Achemlal, "Individual countermeasure selection based on the return on response investment index," *Computer Network Security*, pp. 156–170, 2012.

[175] A. Fawaz, R. Berthier, and W. H. Sanders, "Cost modeling of response actions for automated response and recovery in ami," *2012 IEEE Third International Conference on Smart Grid Communications (SmartGridComm)*, pp. 348–353, 2012.

[176] S. Ossenbühl, J. Steinberger, and H. Baier, "Towards automated incident handling: How to select an appropriate response against a network-based attack?" *2015 Ninth International Conference on IT Security Incident Management IT Forensics*, pp. 51–67, 2015.

[177] V. Mateos, V. A. Villagrá, F. Romero, and J. Berrocal, "Definition of response metrics for an ontology-based automated intrusion response systems," *Computers & Electrical Engineering*, vol. 38, no. 5, pp. 1102–1114, 2012.

[178] N. Stakhanova, S. Basu, and J. Wong, "A cost-sensitive model for preemptive intrusion response systems," *21st International Conference on Advanced Information Networking and Applications (AINA '07)*, pp. 428–435, 2007.

[179] W. Kanoun, N. Cuppens-Boulahia, F. Cuppens, S. Dubus, and A. Martin, "Intelligent response system to mitigate the success likelihood of ongoing attacks," *2010 Sixth International Conference on Information Assurance and Security*, pp. 99–105, 2010.

[180] J. Baayer and B. Regragui, "New cost-sensitive model for intrusion response systems minimizing false positive," *IJMER - International Journal of Modern Engineering Research 2*, 2012.

[181] Z. Zhang, P.-H. Ho, and L. He, "Measuring IDS-estimated attack impacts for rational incident response: A decision theoretic approach," *Computers & Security*, vol. 28, no. 7, pp. 605–614, 2009.

[182] W. T. Yue and M. Çakanyıldırım, "A cost-based analysis of intrusion detection system configuration under active or passive response," *Decision Support Systems*, vol. 50, no. 1, pp. 21–31, 2010.

[183] B. Fessi, S. Benabdallah, N. Boudriga, and M. Hamdi, "A multi-attribute decision model for intrusion response system," *Information Sciences*, vol. 270, pp. 237–254, 2014.

[184] J. Wang, K. Fan, W. Mo, and D. Xu, "A method for information security risk assessment based on the dynamic bayesian network," *2016 International Conference on Networking and Network Applications (NaNA)*, pp. 279–283, 2016.

[185] D. Schnackenberg, K. Djahandari, and D. Sterne, "Infrastructure for intrusion detection and response," *Proceedings DARPA Information Survivability Conference and Exposition. DISCEX'00*, vol. 2, pp. 3–11, 2000.

[186] D. Schnackengerg, H. Holliday, R. Smith, K. Djahandari, and D. Sterne, "Cooperative intrusion traceback and response architecture (CITRA)," *Proceedings DARPA Information Survivability Conference and Exposition II. DISCEX'01*, vol. 1, pp. 56–68, 2001.

[187] S. A. Zonouz, H. Khurana, W. H. Sanders, and T. M. Yardley, "RRE: A game-theoretic intrusion response and recovery engine," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 2, pp. 395–406, 2014.

[188] R. Sharma, H. Kalita, and B. Issac, "Plant based biologically inspired intrusion response mechanism : An insight into the proposed model PIRIDS," *Journal of Information Assurance and Security*, vol. 11, no. 6, pp. 340–347, 2016.

[189] R. K. Sharma, B. Issac, and H. K. Kalita, "Intrusion detection and response system inspired by the defense mechanism of plants," *IEEE Access*, vol. 7, pp. 52 427–52 439, 2019.

[190] H. A. Kholidy, A. Erradi, S. Abdelwahed, and F. Baiardi, "A risk mitigation approach for autonomous cloud intrusion response system," *Computing*, vol. 98, no. 11, pp. 1111–1135, 2016.

[191] N. Herold, M. Wachs, S.-A. Posselt, and G. Carle, "An optimal metric-aware response selection strategy for intrusion response systems," *Foundations and Practice of Security*, pp. 68–84, 2017.

[192] M. GhasemiGol, H. Takabi, and A. Ghaemi-Bafghi, "A foresight model for intrusion response management," *Computers & Security*, vol. 62, pp. 73–94, 2016.

[193] A. Shameli-Sendi, H. Louafi, W. He, and M. Cheriet, "Dynamic optimal countermeasure selection for intrusion response system," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 5, pp. 755–770, 2018.

[194] R. Enns, M. Björklund, A. Bierman, and J. Schönwälder, "Network configuration protocol (NETCONF)," *RFC 6241, RFC Editor, https://www.ietf.org/rfc/rfc6241.txt*, 2011.

[195] K. Djahandari and D. Schnackenberg, "Intruder detection and isolation protocol (IDIP) application layer protocol definition," *Active*

Networks Intrusion Detection and Response Program Technical Information Report, vol. Prepared Under Contract N66001-00-C-8602 for SPAWARSYSCEN San Diego, 2002.

[196] G. Klein, H. Rogge, F. Schneider, J. Toelle, M. Jahnke, and S. Karsch, "Response initiation in distributed intrusion response systems for tactical MANETs," *2010 European Conference on Computer Network Defense*, pp. 55–62, 2010.

[197] A. Avizienis, J. . Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 1, pp. 11–33, 2004.

[198] A. Carlin, M. Hammoudeh, and O. Aldabbas, "Intrusion detection and countermeasure of virtual cloud systems - state of the art and current challenges," *International Journal of Advanced Computer Science and Applications*, vol. 6, no. 6, 2015.

[199] T. Xing, Z. Xiong, D. Huang, and D. Medhi, "SDNIPS: Enabling software-defined networking based intrusion prevention system in clouds," *10th International Conference on Network and Service Management (CNSM) and Workshop*, pp. 308–311, 2014.

[200] M. Heigl, L. Doerr, N. Tiefnig, D. Fiala, and M. Schramm, "A resource-preserving self-regulating Uncoupled MAC algorithm to be applied in incident detection," *Computers & Security*, vol. 85, pp. 270–287, 2019.

[201] G. Androulidakis and S. Papavassiliou, "Improving network anomaly detection via selective flow-based sampling," *IET Communications*, vol. 2, no. 3, pp. 399–409, 2008.

[202] J. Mai, A. Sridharan, C. N. Chuah, H. Zang, and T. Ye, "Impact of packet sampling on portscan detection," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 12, pp. 2285–2298, 2006.

[203] E. G. Bakhoum, "Intrusion detection model based on selective packet sampling," *EURASIP Journal on Information Security*, 2011.

[204] T. Ha, S. Kim, N. An, J. Narantuya, C. Jeong, J. Kim, and H. Lim, "Suspicious traffic sampling for intrusion detection in software-defined networks," *Computer Networks*, vol. 109, pp. 172–182, 2016.

[205] R. E. Jurga and M. M. Hulboj, "Technical report - packet sampling for network monitoring," *CERN openlab report, https://openlab-mu-internal.web.cern.ch/openlab-mu-internal/03_documents/3_technical_documents/technical_reports/2007/rj-mm_samplingreport.pdf*, 2007.

[206] L.-B. Xu, G.-X. Wu, and J.-F. Li, "Packet-level adaptive sampling on multi-fluctuation scale traffic," *2005 International Conference on Communications, Circuits and Systems*, vol. 1, pp. 604–608, 2005.

[207] K. Bartos, M. Rehak, and V. Krmicek, "Optimizing flow sampling for network anomaly detection," *2011 7th International Wireless Communications and Mobile Computing Conference*, pp. 1304–1309, 2011.

[208] Q. Xia, T. Chen, and W. Xu, "CIDS: Adapting legacy intrusion detection systems to the cloud with hybrid sampling," *2016 IEEE International Conference on Computer and Information Technology (CIT)*, pp. 508–515, 2016.

[209] J. He, Y. Yang, X. Wang, and Z. Tan, "Adaptive traffic sampling for P2P botnet detection," *International Journal of Network Management*, vol. 27, no. 5, p. e1992, 2017.

[210] S. Shin, L. Xu, S. Hong, and G. Gu, "Enhancing network security through software defined networking (SDN)," *2016 25th International Conference on Computer Communication and Networks (ICCCN)*, pp. 1–9, 2016.

[211] K. Giotis, G. Androulidakis, and V. Maglaris, "Leveraging SDN for efficient anomaly detection and mitigation on legacy networks," *Proceedings of the 2014 Third European Workshop on Software Defined Networks*, pp. 85–90, 2014.

[212] M. B. Lehocine and M. Batouche, "Flexibility of managing VLAN filtering and segmentation in SDN networks," *2017 International Symposium on Networks, Computers and Communications (ISNCC)*, pp. 1–6, 2017.

[213] B. R. Granby, B. Askwith, and A. K. Marnerides, "SDN-PANDA: Software-defined network platform for anomaly detection applications," *2015 IEEE 23rd International Conference on Network Protocols (ICNP)*, pp. 463–466, 2015.

[214] A. F. Murillo Piedrahita, V. Gaur, J. Giraldo, A. A. Cardenas, and S. J. Rueda, "Leveraging software-defined networking for incident response in industrial control systems," *IEEE Software*, vol. 35, no. 1, pp. 44–50, 2018.

[215] M. G. Pérez, A. H. Celdrán, F. Ippoliti, P. G. Giardina, G. Bernini, R. M. Alaez, E. Chirivella-Perez, F. J. G. Clemente, G. M. Pérez, E. Kraja, G. Carrozzo, J. M. A. Calero, and Q. Wang, "Dynamic reconfiguration in 5G mobile networks to proactively detect and mitigate botnets," *IEEE Internet Computing*, vol. 21, no. 5, pp. 28–36, 2017.

[216] M. Iorga, L. Feldman, R. Barton, M. J. Martin, N. Goren, and C. Mahmoudi, "Fog computing conceptual model," *NIST Special Publication*, no. 500–325, 2018.

[217] R. Shrestha, R. Bajracharya, and S. Y. Nam, "Challenges of future VANET and cloud-based approaches," *Wireless Communications and Mobile Computing*, vol. 2018, pp. 1–15, 2018.

[218] M. Doering and M. Wagner, "Retrofitting SDN to classical in-vehicle networks: SDN4CAN," *Universität Tübingen, http://hdl.handle.net/10900/78141*, 2017.

[219] K. Halba and C. Mahmoudi, "In-vehicle software defined networking: An enabler for data interoperability," *Proceedings of the 2nd Interna-*

tional Conference on Information System and Data Mining, pp. 93–97, 2018.

[220] M. D. Z. Khan, M. Chowdhury, M. Islam, C. Huang, and M. Rahman, "In-vehicle false information attack detection and mitigation framework using machine learning and software defined networking," *arXiv preprint arXiv:1906.10203*, vol. abs/1906.10203, 2019.

[221] A. Alioua, S.-M. Senouci, and S. Moussaoui, "dSDiVN: A distributed software-defined networking architecture for infrastructure-less vehicular networks," *Innovations for Community Services*, pp. 56–67, 2017.

[222] C. Jiacheng, Z. Haibo, Z. Ning, Y. Peng, G. Lin, and S. Xuemin, "Software defined internet of vehicles: architecture, challenges and solutions," *Journal of Communications and Information Networks*, vol. 1, no. 1, pp. 14–26, 2016.

[223] W. B. Jaballah, M. Conti, and C. Lal, "A survey on software-defined VANETs: Benefits, challenges, and future directions," *arXiv preprint arXiv:1904.04577*, vol. abs/1904.04577, 2019.

[224] A. Mahmood, W. E. Zhang, and Q. Z. Sheng, "Software-defined heterogeneous vehicular networking: The architectural design and open challenges," *Future Internet*, vol. 11, no. 3, 2019.

[225] T. Häckel, P. Meyer, F. Korf, and T. C. Schmidt, "Software-defined networks supporting time-sensitive in-vehicular communication," *arXiv preprint arXiv:1903.08039*, vol. abs/1903.08039, 2019.

[226] M. O. Kalinin, V. M. Krundyshev, and P. V. Semianov, "Architectures for building secure vehicular networks based on SDN technology," *Automatic Control and Computer Sciences*, vol. 51, no. 8, pp. 907–914, 2017.

[227] A. A. Khan, M. Abolhasan, and W. Ni, "5G next generation VANETs using SDN and fog computing framework," *2018 15th IEEE Annual Consumer Communications Networking Conference*, pp. 1–6, 2018.

[228] R. Poler, A. Tsuchiya, A. Ortiz, I. Koshijima, and F. Fraile, "Software-defined networking firewall for industry 4.0 manufacturing systems," *Journal of Industrial Engineering and Management*, vol. 11, no. 2, pp. 318–333, 2018.

[229] R. Yusof, S. R. Selamat, and S. Sahib, "Intrusion alert correlation technique analysis for heterogeneous log," *International Journal of Computer Science and Network Security*, vol. 8, no. 9, pp. 132–138, 2008.

[230] J. M. Chahira, J. K. Kiruki, and P. K. Kemei, "A review of intrusion alerts correlation frameworks," *International Journal of Computer Applications Technology and Research*, vol. 5, no. 4, pp. 226–233, 2016.

[231] S. Indriyanto, M. N. D. Satria, A. R. Sulaeman, R. Hakimi, and E. Mulyana, "Performance analysis of VANET simulation on software defined network," *2017 3rd International Conference on Wireless and Telematics (ICWT)*, pp. 81–85, 2017.