Defence Research and Development Canada
Recherche et développement pour la défense Canada

CAN UNCLASSIFIED

DRDC | RDDC
technologysciencetechnologie

Malicious activity detection

An analysis of current tools and methodologies for network defence in operational networks

Maxwell Dondo
Madeena Sultana
Grant Vandenberghe
DRDC – Ottawa Research Centre

Defence Research and Development Canada

Reference Document

DRDC-RDDC-2021-D078

May 2021

CAN UNCLASSIFIED

Canada

IMPORTANT INFORMATIVE STATEMENTS

Abstract

Operational networks continue to face increasing and evolving malicious activities that threaten to disrupt ongoing missions. To ensure mission continuity and minimise disruptions due to cyber attacks, network defenders must detect and prevent such malicious activities. As computer networks have become increasingly complex, detecting malicious activities is not trivial. It requires the monitoring of the network as well as the employment of appropriate tools and methodologies to detect possible attacks. In this work, we explore such tools and methodologies. We perform an extensive literature search of the current commercial off-the-shelf (COTS) products that could be used for malicious activity detection. Our work identifies the strategic network placement of the tools to allow for the most visibility of network activities, and thereby minimising the possibility of missing harmful traffic. The COTS products are complemented by the current state-of-the-art research activities in malicious activities to ensure the awareness of the most recent technologies as well as to gain more insights of the directions the research community is moving forward to. The summary discussion of our findings leads to possible future work that we would recommend undertaking to improve the detection of malicious activities in the network.

Significance for defence and security

This work provides computer network defence (CND) researchers and network defenders with an understanding of the tools and methodologies that could be used to detect malicious activities in enterprise networks. The work identifies possible research work activities that Defence Research and Development Canada (DRDC) could perform to complement existing tools' capabilities to identify disruptive activities on the network, in their continuing effort to advise the Department of National Defence (DND)/Canadian Armed Forces (CAF) in defending their networks. Specifically, the work could address a capability requirement by Cyber Defence-Decision Analysis Response (CD-DAR) in understanding the tools and methodologies necessary for monitoring the network to identify possible malicious activities.

Résumé

Les réseaux opérationnels continuent d’être la cible d’activités malveillantes. Ces activités sont en croissance et en évolution et elles constituent une menace pour les missions en cours. Pour assurer la continuité des missions et minimiser les perturbations reliées aux cyberattaques, les responsables de la sécurité des réseaux doivent être en mesure de détecter et de prévenir de telles activités. Vu la complexification des réseaux informatiques, la détection n’est pas chose simple. Elle exige une certaine surveillance des réseaux et l’utilisation d’outils et de méthodologies appropriés.

Dans le cadre de nos travaux, nous avons exploré ce genre d’outils et de méthodologies. Nous avons effectué une recherche documentaire approfondie sur les produits commerciaux actuels qui pourraient être utilisés pour détecter les activités malveillantes. Nous avons déterminé le positionnement stratégique des outils dans les réseaux pour rendre les activités qui s’y déroulent le plus visibles possible et minimiser les possibilités que le trafic nuisible ne soit pas détecté. De plus, nous avons analysé des travaux de recherche de pointe sur les activités malveillantes pour mieux comprendre en quoi consistent les technologies les plus récentes et les orientations actuelles en recherche dans le domaine. Dans notre analyse des résultats, nous proposons des pistes de recherche possible pour les travaux futurs sur le sujet.

Importance pour la défense et la sécurité

Le présent document vise à aider les chercheurs dans le domaine de la défense des réseaux informatiques (CND) et les responsables de la sécurité des réseaux à mieux comprendre les outils et les méthodologies qui peuvent être utilisés pour détecter les activités malveillantes dans les réseaux des organisations. Il fait notamment état de travaux de recherche possibles que Recherche et développement pour la défense Canada pourrait entreprendre pour favoriser l’amélioration de la capacité des outils existants à détecter les activités malveillantes dans les réseaux, et ce, afin de continuer de soutenir les Forces armées canadiennes et le ministère de la Défense et de les aider à protéger leurs réseaux. De façon plus particulière, le document pourrait permettre de répondre au besoin du groupe Cyberdéfense – Décision, analyse et réponse (CD-DAR) d’avoir une meilleure compréhension des outils et des méthodologies nécessaires à la surveillance des réseaux pour mieux détecter les activités malveillantes.

Table of contents

List of figures

List of tables

1 Introduction

The costs associated with the loss of data or disruptions to business processes, such as military missions, are too high for organisations to ignore [1]. Such disruptions can be circumvented or prevented with the right tools and methodologies. There is, therefore, the need for organisations to understand the tools and methodologies that could be used to reduce losses or any other cyber related disruptions. In this work, we identify the latest tools and methodologies that could be used to detect malicious activities on the network.

It is important to understand the placement of the sensors that provide data to the detection tools to effectively detect malicious activities. As described in Section 2, the distributed computer network has a number of segments. Each segment has different appliances, data speeds, and network visibility, requiring the right type of sensor to be put in place. For example, the high-speed subnet requires sensors whose detection rates match the line data speeds, otherwise there would be many missed activities, which could result in missed attack detection opportunities. The section also describes the type of data that can be collected from the network and how it could be stored for later use.

As described in Sections 3, the first step in malicious activity detection is to monitor the network. This is achieved by deploying commercial off-the-shelf (COTS) tools or applying other state-of-the-art methodologies to collect data at the locations identified in Section 2. Monitoring data can be collected from host logs, perimeter defence appliances (e.g., firewalls) logs, server logs, or network packets collected at the host, subnet or the high-speed network sides. The second step is the application of COTS tools and methodologies to analyse the monitoring data to detect possible malicious activities. Such tools and methodologies can be anomaly- or signature-based, and in some cases a combination of the two. Behavioural analytics, which is a subset of anomaly detection, have increasingly become popular approaches as well.

Also described in Sections 3 are the malicious activity detection techniques that can be applied to data from different segments of the network. For that, we partition the network into three segments, the high-speed backbone network, the subnet networks, and the hosts (and servers). Different COTS tools and methodologies are employed to provide detection at each of these network segments. The section concludes by presenting tools and methodologies that can be similarly used for extrusion detection as well as data leakage detection (DLD).

In Section 4, we present a collection of more specialised methods of malicious activity detection. The section covers malware detection though sandboxing, deception, and evasive techniques. Although there are specialised tools and methodologies that could be used for detecting malicious activities this way, the Intrusion Detection Systems (IDSs) described earlier, could, with a lot of effort and skill, be configured to detect such malicious activities. However, the existence of COTS tools and state-of-the-art techniques makes the job easier.

Section 5 summarises our findings and presents possible future work activities for identified technology gaps.

2 Background

The detection of malicious activity in networks is generally split into two main phases. The first phase is to monitor the network and acquire data. The second phase is the analysis of the acquired data to detect and identify any possible malicious activities. This section summarises the collection of such data, its types, and how it can be stored. Detailed discussion on network monitoring and data collection can be found in the literature [2–4].

2.1 Sources of data and the means for its collection

To understand the sources of data for malicious traffic detection, consider the network of a typical large distributed enterprise network. A simplified network diagram of such a complex system is shown in Figure 1.

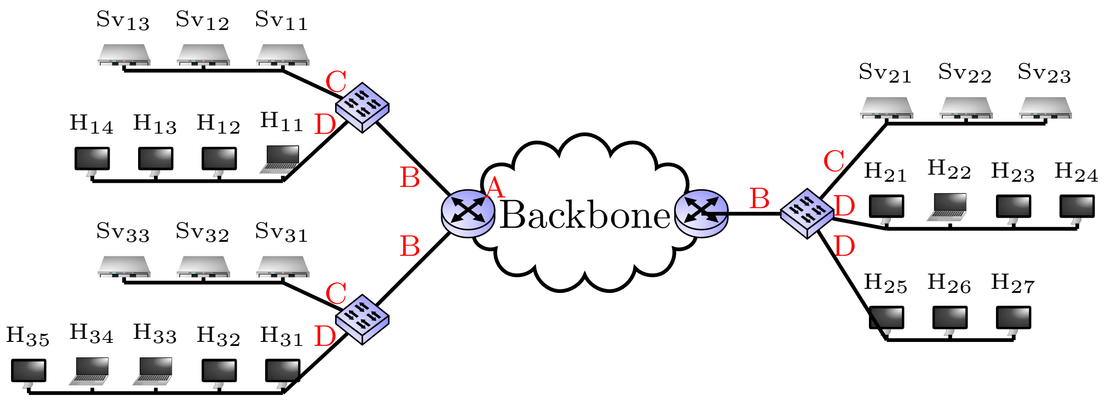

Figure 1: *A simplified example of monitoring needs and sensor placements in an organisational network split into three sites and linked by a high speed backbone. H_{ij} represents hosts, and Sv_{ij} represent local servers. The red letters A to D represent candidate sensor placements discussed in the report.*

In the figure, we show an organisation that is distributed over multiple locations, three in this case. This is typical of military departments such as the Canadian Department of National Defence (DND) or the United States (US) Department of Defence (DoD), which have many bases across their respective countries (or even all over the world). Network infrastructure that facilitates inter-site communications is referred to the *backbone*. Such a backbone, which can sometimes be an Internet Service Provider (ISP), also includes the gateway between the organization and the external world. This connection is typically a high speed connection that sees very high volumes of traffic.

As shown in Figure 1, each site has a number of servers that are located in a demilitarised zone (DMZ). Through a method called network address translation (NAT), the internet protocol (IP) addresses of the sites' hosts are mapped into another IP address (or addresses). Such NATed IP addresses are not visible from outside the network, and many such hosts can use a single external IP address. Therefore, sensors should be strategically placed to allow the ability to see traffic patterns that are useful for the network defender to detect malicious activities.[1] Table 1 outlines what could be achieved from the different sensor placements shown in Figure 1.

Table 1: *Sensor placement in corporate network shown in Figure 1.*

Position	Rationale	Network Traffic	
		Direction	IP address Visibility
A	Best location to sense everything coming into the network.	Incoming	Traffic appears to be destined to one IP address if NATed.
	All traffic leaving the network passes through this sensor.	Outgoing	Traffic appears to originate from one IP address if NATed.
B	Similar to A, this high-speed location sees all traffic coming into this part of the network.	Incoming	Traffic destined to NATed hosts appears to be going to one IP address.
	All traffic leaving this branch of the network passes through this high speed point.	Outgoing	Traffic coming from NATed sources appears to originate from one IP address.
C	This location senses traffic destined to the servers in this branch of the network.	Incoming	True IP addresses visible, even if the network is NATed.
	All traffic originating from the servers in this network branch is visible at this location.	Outgoing	The true destination IP addresses are visible at this location.
D	Traffic destined to hosts is visible at this location.	Incoming	The true IP addresses are visible.
	Traffic originating from the hosts is visible.	Outgoing	The sensor can see the true destination IP addresses.

The sensor placements can be implemented strategically to complement each other. When information is correlated, then a full picture of what is happening can emerge. For example, while the identity of a compromised host's connection to an adversary's network may not be obvious at sensor Location A, it can be identified easily at D.

In practice, there are a number of ways the sensors can be placed [2, 3, 5]. Existing network infrastructure can offer options for monitoring traffic. For example, a network can employ existing capabilities in switches to monitor traffic. The switch ports can be configured to send copies of its traffic through the switched port analyzer (SPAN), which is also referred to as *port mirroring* [3, 6]. The SPAN can see all the port traffic, which can be used to monitor the network for possible malicious activities.

Defenders can also deploy physical appliances, which are called network test access points (TAPs),[2] to monitor network traffic. Preferably located at C and D, the TAPs can provide copies of network traffic for the desired monitoring at those locations [7]. Gigamon argues that TAPs are preferred over SPANs due to their reliability, efficiency and effectiveness [8].

Detection of malicious activities can also be accomplished by reading logs from perimeter defence devices such as routers, firewalls, and servers. In reality, all these activities are also seen by packet sniffers, but the logs give an application level view of those activities,

[1] Refer to literature [2–4] for detailed explanation.
[2] Sometimes referred to as terminal access point [7].

thus providing an easy to read data source. In addition, network activity data may also be collected at each of the hosts on the network.

2.2 Types of data to collect

In Table 2, we list a summary of the types of data that can be collected at different network locations. We cross-reference locations from Figure 1 where such data could be collected.

Table 2: *Types of data to collect.*

Data Type	Location	Format	Comment
Traffic packets	A–D	Binary, text, numeric, or packet capture (pcap)	Compression may be required due to large sizes of pcap files.
Flow data	A–D	Text, numeric	Tool-dependent.
Metadata	A–D	Text, numeric	Tool-dependent.
Log data	Routers, servers, switches, hosts	Text, numeric	Vendor specific formats exist. For example Internet Information Services (IIS) server logs are different from that of Apache's.

The table shows that most of the locations in Figure 1 collect network traffic data in packet capture (pcap), metadata, and text formats. Log data comes mostly in a readable structured text data format.

2.3 Storage

As we will discuss in the next sections, there are many tools that analyse network data in near real-time to detect possible malicious activities. However, most tools process historical network traffic data. Even the tools that perform real-time processing need the ability to correlate with historical activities. Such capabilities can be possible through the efficient storage of the data collected.

Many of the tools we will discuss in the next sections have their own proprietary data storage approaches. However, many COTS tools use open source storage methods. For example, Snort [9] can be configured to use MySQL database for storage. Threat analysis tools can then query such data stores and use the information stored to identify possible malicious activities. In a large organisation with a centralised security team, the data storage could look like the illustration in Figure 2.

The Figure illustrates different sources of data going into an organisational data lake or data mesh [10], which is represented by the cloud. The distributed repository allows for the connection of distributed data across different locations ensuring high data availability and greater autonomy, which allows flexible experimentation and analytics [11, 12]. Individual

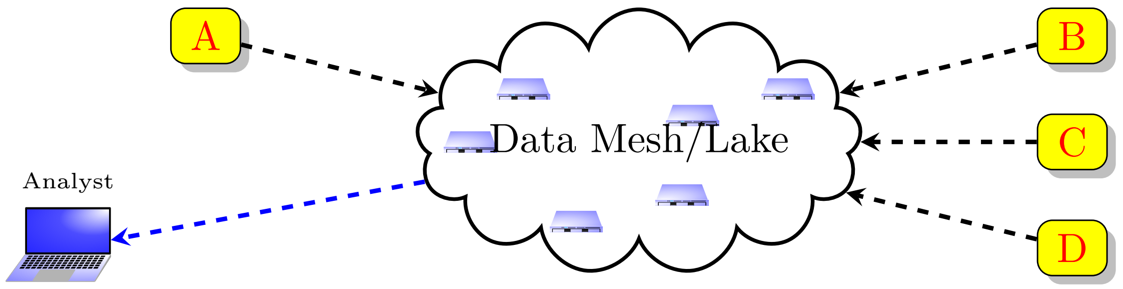

Figure 2: *A centralised data storage for malicious activity detection. Data collected from Locations A to D in Figure 1 is stored in a data lake or mesh and is available for analysis to detect possible malicious activities.*

analytics tools can query that data storage and correlate identification of possible malicious activities.

In the next sections we discuss the monitoring approaches that populate such data storages. We then present methodologies that exploit the data to detect possible malicious activities.

3 Monitoring tools and methodologies for malicious activity detection

Computer networks need monitoring in order to detect and identify malicious activities. Network monitoring can be performed in a number of ways. The most common, and effective approach is to use packet sniffers [3]. Packet sniffers, which are also called packet-, protocol-, or network-analysers, can be used to examine communication data streams in the network. Other monitoring approaches read logs from perimeter defence appliances and servers, such as web and mail servers.

Analysis tools examine data from the monitoring tools to detect and identify possible malicious activities at different parts of the network. These tools range from statistical analysis of the packets data to signature-based identification of malicious activities in traffic flows or logs data. In this section, we first present general COTS monitoring tools and methodologies that we arbitrarily selected from literature. Second, we present another set of tools and low technology readiness level (TRL) [13] methodologies that detect and identify malicious activities by analysing data obtained from the monitoring tools and methodologies.

3.1 Monitoring

It is important to collect meaningful data that would eventually inform analytics tools and methodologies in detecting and identifying malicious activities. As the old adage goes, garbage-in-garbage-out, it is essential that the defenders of the network only collect data that can lead to malicious activity detection. Therefore, the placement and type of sensors as well as the type of data they collect is crucial. In this section, we present both COTS and recent research methodologies for network monitoring as dictated or inferred from literature [14–19]. We break down the COTS monitoring tools into three categories based on the placement of the monitoring sensors in the network.

3.1.1 Backbone (high-speed subnet) monitoring

The monitoring of the backbone requires capabilities that can sniff packets at a fast pace without dropping packets.

3.1.1.1 COTS tools

1. *SentryWire* [20] is a full packet capture appliance that claims lossless capture at speeds ranging from 1Mbps to 1 Tbps. It allows full visibility packet capture that can span over the 146-day industry average to detect advanced persistent threats (APTs) [20].
2. *RSA NetWitness Network* [21] is a full packet capture tool that can be applied on premises information technology infrastructure (ITI), in the cloud, or with virtual infrastructures. It is capable of providing full-packet capture, metadata, and flow-based information from its monitoring.

3. *NetShark* [22] provides a line-rate, multi-gigabit per second monitoring capability. A single appliance can support 1 GB or 10 GB copper or fiber interfaces and it can support up to eight 10 GB interfaces. The traffic captured can be stored for analysis or can be analysed in real-time by fast analytics tools. The vendor, Riverbed, also collects flow data from "standard routers and switches, etc." through its SteelCentral Flow Gateway, allowing more visibility into the network activities. In NATed networks, flow monitoring could be most effective at the subnet level as described in Section 2.

4. *Aukua MGA2510 Analyzer* [23] is a traffic monitoring appliance that can provide full packet capture at line rates. It is capable of ethernet data rates from 10 Mbps to 10 Gbps and supports the new Institute of Electrical and Electronics Engineers (IEEE) 802.3 bz 2.5 Gbps and 5 Gbps ethernet rates.

5. *PacketScan* [24] is a packet capturing tool that is said to monitor networks and be able to capture packets in networks up to 10 Gps. The packets are stored in a database that can be accessed by analysis tools.

6. *SolarWinds Network Performance Monitor Network Performance Monitor (NMP)* [25] provides multi-layered network monitoring through its packet sniffing capabilities. NPM provides flow information by utilizing NetFlow, JFlow, sFlow, NetStream, and IP Flow Information Export (IPFIX) data formats built into most routers and switches.

3.1.1.2 Research methodologies

Some arbitrarily selected recent research activities in high-speed network monitoring are as follows:

1. Kekely et al. [26, 27] proposed a wire-speed packet capturing tool. Their approach claims speeds up to 200 Gps as a combination of two 100 Gps ethernet links. The packets captured are stored for analysis by traffic analysis tools.

2. Cardigliano et al. [28] proposed a wire-speed network monitoring approach using virtual networks. They argue that their approach is cheap to implement and is capable for multi-gigabit monitoring.

3. Dashtbozorgi and Azgomi [29] proposed DashCap, which is a high-speed packet capturing tool that uses multi-core features. The monitored traffic can be stored in a database for analysis in identifying malicious activities.

4. Schneider et al. [30] proposed an approach to monitor high speed interfaces. For example, they are able to monitor 10 GB interfaces distributing the interface traffic through sets of low speed (e.g., 1 GB) interfaces.

5. Deri [31] proposed nCap as a wire-speed packet capturing tool that can be used to monitor the network's high speed backbone. The tool can be considered to have high TRL because of its adoption of commercial network adaptors.

3.1.2 Subnet network monitoring

In this section we present the tools and methodologies for monitoring networks at the subnet level. Tools used to monitor the high-speed side of the network that we listed above, can also be used at this level.

3.1.2.1 COTS tools

In addition to the COTS backbone monitoring tools discussed above, the following COTS tools are commonly used for monitoring the network.

1. *P0f* [32] is a widely used fast lightweight network security monitoring tool. From the monitoring data, it can also be used to identify operating systems (OSs) of hosts connected to the network, perform queries, etc.

2. *Argus*, which stands for Audit Record Generation and Utilization System [33–35], is used for monitoring all aspects of an enterprise network. It is very fast, and can efficiently sift through large datasets and provide comprehensive reports when used for traffic analysis.

3. *Nagios* [36] is a well-regarded network monitoring tool. It monitors networks, connected hosts, and systems in real-time providing mission critical infrastructure monitoring of all network components. Nagios states that such capabilities cover "applications, services, OSs, network protocols, systems metrics, and network infrastructure" [36].

4. *Flowmon Probe* [37] is a hardware or virtual appliance that generates or exports flow data from observed network traffic. It can be connected to the network infrastructure (SPAN or TAP connections possible) for network traffic visibility using its monitoring ports.

5. *Paessler's PRTG Network Monitor* [38] provides network monitoring that is supported by its extensive packet capturing capability, although it only captures the packet headers. PRTG has more than 250 sensors that, among other things, enable it to monitor flow traffic such as NetFlow, IPFIX, JFlow, and sFlow.

6. *Splunk* [39] claims that its *Infrastructure Monitoring* tool is very fast for monitoring the security of the network. It can be used to perform off-line searches and conduct network data analysis in real-time. It can be used to capture, index, and collate network data. It also allows generation of many forms of reports.

There are too many monitoring tools in this category to list. Other notable tools commonly used in this space are Snort [9], tcpdump [40, 41], Wireshark [40, 42], Zeek [43], ManageEngine's Netflow Analzer [44], and SolarWinds (Netflow Traffic AnalyzerNetflow Traffic Analyzer (NTA)) [25]. However, Zeek and SolarWinds' NTA are most effective in monitoring at the backbone.

3.1.2.2 Research methodologies

Some arbitrarily selected research activities showing the current state-of-the-art in network monitoring are as follows:

1. Wellem et al. [45] proposed an efficient sketch based network monitoring framework. It provides flexibilities of flow aggregations and update without reprogramming the hardware data plane. With a throughput as high as 96 GBps, their simulation results on traffic traces demonstrated an ability for selective attack detection.

2. Nguyen et al. [46] proposed a proactive network monitoring solution using deep learning. This study used a series of deep learning techniques such as convolutional neural network (CNN), gated recurrent unit (GRU), long-short term memory (LSTM), etc. for effective monitoring and flow prediction. This has been validated on both proprietary and publicly available datasets.

3. Koning et al. [47] proposed CoreFLow, which is a BRO-based monitoring framework. The approach reconstructs routes of malicious traffic flow. The CoreFlow prototype has been tested on a network utilizing three Bro systems and more than 50 routers.

4. Casino et al. [48] proposed a monitoring methodology named High Entropy Distinguisher High Entropy Distinguisher (HEDGE) that can differentiate between encrypted and compressed packets based on the randomness of the datastream. However, the accuracy of detection of this method depends on the packet size.

5. Lv et al. [49] proposed a large scale network traffic monitoring system. Their approach consolidates Netflow for collecting real-time data, Logstash for transferring data, ElasticSearch for storing, and Kibana for displaying real-time analysis. The proposed system can achieve millisecond responses to 100 million of Netflows, which has the potential of meeting the need of large-scale network traffic.

3.1.3 Host monitoring

This section presents some of the tools and methodologies used for monitoring the network hosts.

3.1.3.1 COTS tools

The tools commonly used for host monitoring are as follows:

1. *OSSEC* [50] is an open-source multi-platform, host-based monitoring tool. It monitors all possible sources of malicious activity, including logs, rootkits, registries, processes, etc. Useful practitioner support, modifications, tricks and techniques, etc. are widely shared by the user community.

2. *System Monitor (Sysmon)* [51] is a Microsoft (MS) Windows service and device driver that is installed on Windows hosts. It monitors activities on the hosts and saves the information to a Windows event log. The log can be analysed to identify possible malicious activities. Possible adversarial tactics, techniques, and procedures (TTPs) can be harvested from the activity logs as well.

3. *TCPView* [52] can be used to monitor network hosts by listing the details of all Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) endpoints on a system. This includes the local and remote addresses and state of TCP connections.

4. *CurrPorts* [53] is host monitoring software that lists all currently opened TCP/IP and UDP ports on the host.

A number of monitoring tools used at the network level can also be used for host monitoring. Examples of such tools are: Snort [9], tcpdump [40, 41], Wireshark [40, 42], Zeek [43].

3.1.3.2 Research methodologies

Some of state-of-the-art research activities in host monitoring are as follows:

1. Berger et al. [54] proposed an approach for cloud-based in-host monitoring to complement passive network monitoring. The complementary monitoring techniques are effective in detecting attacks by providing contextual links to analysts.

2. Haas et al. [55] proposed an approach that collects both network and monitored host data. The approach then combines the Zeek IDS with the Zeek-osquery host monitor to detect malicious activity.

3. Jirsiket et al. [56] developed Stream4Flow as an IP flow host monitoring framework based on Apache Spark Streaming. Their approach addresses the problem of insufficient IP host monitoring faced by classical continuous monitoring systems. The resulting data can be used for possible malicious traffic detection.

4. Yuan et al. [57] proposed HostWatcher, a software defined networking (SDN)-based monitoring system that can monitor every host in cloud data center. Experiments show that HostWatcher's monitoring can allow for the effective mitigation of distributed denial of service (DDoS) attacks on hosts.

5. Kortebi et al. [58] proposed home networks monitoring and anomaly detection solution based on FlowMon components. Experiments on a home network showed the efficiency in monitoring file transfer, video streaming etc., allowing for the ability to detect port scans.

6. Verma et al. [59] provided a metric space framework for general host logs and log sequences based on semantic similarity. They also provided an embedding of the metrics that can be useful for data science analytics, machine learning, and time-series analysis methods.

3.2 Intrusion detection

In this section we identify tools and methodologies used for intrusion detection, which is a very broad topic in cyber security and encompasses many methodologies. In fact, many cyber malicious activity detection methodologies can be argued to be a subset of intrusion detection. In general, intrusion detection involves the collection of network activity through traffic monitoring or packet logging (discussed above), followed by an analysis phase that identifies which of the activities could be malicious. This section presents both mature COTS and recent research methodologies whose TRL levels are still very low.

3.2.1 Backbone network IDS

The IDSs for the backbone network obtains its network data from the location marked A in Figure 1. Since this is a high speed location, it sees a significant amount of network traffic data. IDS tools that are best suited to detect possible malicious activities at this location are those that are able to process large amounts of data in near real-time. For historical and forensic analysis, any tool that can read network packets could perform this task.

3.2.1.1 COTS tools

The COTS tools commonly associated with processing network backbone traffic are as follows:

1. *Snort* [9] is an open source IDS tool that can be used as an intrusion prevention system (IPS). It analyses real-time or historical traffic patterns and compares them against previously written signatures in the form of configuration files. Its rules can be configured for both anomaly- and signature-based detection of malicious activities. Sourcefire's[3] Next-Generation IPS Next-Generation IPS (NGIPS) [60], which is powered by Snort, has a high-speed network intrusion detection system (NIDS) capability of up to 40 Gbps.

2. *Suricata* [61] is a free open-source signature-based real time NIDS and inline IPS that also performs network security monitoring (NSM). It can perform analysis of pcap files to support forensic investigations. Suricata supports standard input and output formats like YAML[4] and JSON.[5] It can also be integrated with other tools such as security information and event management (SIEM) systems, Splunk, Kibana, etc.

3. Trend Micro's *Tipping Point* [62] has NIDS capabilities that allow it to detect malicious activity in real-time. It's machine-learning (ML) capabilities enable it to identify many types to threats, including zero-day attacks.

4. *Packet Data Analysis Packet Data Analysis (PDA)* [24], provided by GL Communications, can be combined with many of their tools to provide online analysis of traffic on

[3] Now part of Cisco.
[4] File format called "YAML Ain't Markup Language."
[5] File format called "JavaScript Object Notation."

the high speed backbone. It can handle many protocols and provide a comprehensive view of analyses results.

Other NGIPSs, such as Cisco's [63] or NSFocus [64], mostly have IDS capabilities that are signature-based. In addition, they have other capabilities, such as intrusion prevention, and can handle high speed traffic pipes.

3.2.1.2 Research methodologies

Some state-of-the-art IDS methodologies we identified for this category are as follows:

1. Wakui et al. [65] proposed a deep learning-based anomaly detection method named General-purpose Anomaly detection Mechanism using Path Aggregate without Labeled data General-purpose Anomaly detection Mechanism using Path Aggregate without Labeled data (GAMPAL) for Internet backbone traffic. They used and unsupervised Long Short-Term Memory Recurrent Neural Network (LSTM-RNN) to predict anomalies based on historical traffic. Evaluation of real-time traffic showed that GAMPAL could detect traffic surges due to suspicious events and DDoS attacks.

2. Durate Jr. et al. [66] proposed a flow-based method for anomalous traffic detection on large-scale backbone networks. The proposed method uses entropy and principal component analysis (PCA) operators calculated from four flow-based traffic features. Performance evaluation on synthetic traffic showed promising results of detecting three types of injected anomalous traffic.

3. Li et al. [67] proposed a multi-Hidden Conditional Random Fields Hidden Conditional Random Fields (HCRF)-based anomaly detection method for backbone networks, the resilient traffic anomaly recognition resilient traffic recognition (R-TAR). The proposed method, which uses temporal and spatial features from both packet and flow-based granularities, obtained promising results on publicly available datasets.

4. Viegas et al. [68] proposed a flow based IDS named BigFLow for high-speed networks. It can extract up to 158 statistical features, which have been used by a stream learning stage engine that classifies normal and attack traffic. BigFlow, which has been evaluated on MAWIFlow dataset [68], can support 10 GBps network bandwidth in a 40-core cluster commodity hardware.

5. Erlacher and Dressler [69] proposed an improved version of the IPFIX-based signature-based IDS (FIXIDS), which claims to be more traffic efficient than Snort and does not degrade event detection rates. FIXIDS, which supports custom and Snort signatures, has been experimentally shown to triple the network throughput capability of Snort from 0.5 GBps to 1.5 GBps as well as identifying hypertext transfer protocol (HTTP) based attacks at 9.5 GBps without packet drops.

3.2.2 Subnet network intrusion detection system (NIDS)

The tools and methodologies described in this section detect and identify malicious activities by analysing traffic monitored in branches marked B, C, and D in Figure 1.

3.2.2.1 COTS tools

Some arbitrarily selected NIDS COTS tools in this category are as follows:

1. *Zeek* [43], formerly known as Bro, is a NIDS that is placed on a sensor to monitor and interpret the network traffic below 10 Gps. It provides output in the form of transaction logs, file content, and other customized output to facilitate manual review or as input to SIEM systems.

2. *Vectra* [70] has an artificial intelligence (AI)-based IDS capability that is touted to detect advanced threats, including zero-day threats. It raises alerts on detection of malicious activities, while maintaining the security of users' data through encryption.

3. Quadrant's Sagan [71] is a free tool that applies both anomaly- and signature-based detection techniques. It can be configured to take automated remediations, making it double as an IPS as well. It has a powerful capability of IP geolocation that alerts on suspected malicious activities originating from the same geographic region. Its rules can be configured to work similar to those of Sourcefire, Snort, or Suricata IDS/IPS engines, allowing Sagan to correlate events across NIDSs.

4. *Extrahop* Reveal(x) [72] is a high throughput data (100 Gps) IDS that helps to discover and classify endpoints and transactions in "real time." It is capable of continuous packet capture, which can be analysed by its ML methodologies to detect attacks. It is in the same category as DarkTrace [73], but claims higher capabilities that include, but is not limited to, ML-based analytics. It allows for auto-discovery, real-time traffic correlations, and can integrate with common SIEMs.

High speed network level IDSs such as Snort [9] or Suricata [61] can also be configured for use at the subnet level. *Information flow IDSs*, which are NIDS that utilise information from network traffic flows to identify and alert on potential malicious activities, also fall under this category. COTS tools examples in this category are: Flowmon [37], Varonics [74], Cisco's NetFlow [75], etc.

It is worth noting here that NIDSs are increasingly adopting user and entity behavior analytics (UEBA) and network behavior anomaly detection (NBAD) technologies to improve the security in enterprise networks. UEBA is a methodology premised on identifying patterns in typical user and entity behaviors and then detecting anomalous activities that do not match expected patterns. NBAD/UEBA methodologies ingest daily traffic data to build a baseline. Statistics as well as ML models can be used to detect and alert on anomalous activities that don't match the baseline. Such approaches can be useful for zero-day attack detection. Among the many available NBAD/UEBA COTS products, we arbitrarily selected the following network IDS tools:

1. *ManageEngine EventLog Analyzer* [44] is a signature- and anomaly-based NIDS (and Host Intrusion Detection System (HIDS)). Its near real-time analysis can provide statistical data, which can be useful for monitoring, baselining, and NBAD capabilities.

2. *Stealthwatch:* Cisco's Stealthwatch [76] combines its multi-layer ML and behavioural modeling capabilities and global threat intelligence awareness to detect possible malicious activities. It's NBAD capabilities can also be configured to work by ingesting sensor data from third-party tools.

3. *Darktrace* [73] has a number of detection tools to detect novel attacks at an early stage. It uses artificial intelligence (AI) methodologies to learn the current enterprise to understand "self" so as to detect any small changes that can be brought about by attacks.

4. *Splunk* [39] has a UEBA add-on that aggregates a variety of data sources to profile user and entity behaviours. It uses ML techniques to detect anomalies and raise alerts.

5. *Varonics* [74] uses statistical, ML, and deep learning techniques for corporate user behaviour modeling. It alerts on detected anomalous behaviour.

6. *Securonix* [77] is a UEBA tool that is modeled on a Hadoop big security data lake. It profiles user behaviour from data and alerts on possible malicious activities.

7. *ATA* [78], Microsoft's advanced threat analytics, ingests information from multiple data sources such as logs from SIEMs, Windows Event Forwarding Windows Event Forwarding (WEF), or Windows Event Collector Windows Event Collector (WEC) to learn the user and entity behaviour of the organization. Using ML and other data analytics methods, it then builds behavioural profiles from which it can detect and alert on malicious activities.

While UEBA tools are mostly applied to the enterprise network, they can also be applied to individual hosts (with the HIDS in the next section) to monitor and protect specific critical resources.

3.2.2.2 Research methodologies

Some of the recent state-of-the-art activities in this category are as follows:

1. Van et al. [79] proposed a deep learning anomaly-based NIDS. However, according to the authors, the classification is only limited to four groups of attacks. Therefore, it is unsuitable for application with operational networks where the scope of threats is broad.

2. Sohi et al. [80] proposed a deep learning approach that generates synthetic signatures to improve detection rates of unseen and zero day attacks. The authors used a recurrent neural network (RNN) to generate synthetic signatures of unseen mutants of attacks and feed them to Bro for intrusion detection. Results demonstrated 17% improvement of Bro's performance on publicly available data.

3. Hwang et al. [81] proposed D-PACK, a deep learning-based anomalous network traffic detection method, which auto-profiles traffic patterns and filters abnormal traffic. D-PACK, which consists of a supervised CNN and an unsupervised Autoencoder, demonstrated very high detection rates of anomalous traffic on publicly available datasets.

4. Chkirbene et al. [82] proposed to combine supervised random forests and classification and regression trees (CART) in a sequential technique for anomaly detection. The random forest classifier was used to select important features. Although the overall anomaly detection rate was promising, it failed to detect some anomalies such as backdoor, denial of service (DoS), shellcode, worms etc. due to lack of training data.

5. Jiang et al. [83] proposed a NIDS combining a CNN and a Bi-directional long short-term memory Bi-directional long short-term memory (BiLSTM) to detect a wide range of attacks with high accuracy. The method used up-sampling techniques to handle small and unbalanced training samples. However, the detection rate was poor for some attacks such as backdoors, worms, etc., due to a very small size of training samples.

3.2.3 Host Intrusion Detection Systems (HIDSs)

HIDSs analyse the data collected from the monitoring of individual hosts. Below are some COTS tools in this category.

3.2.3.1 COTS tools

The COTS products listed here are based on what is described in open literature [84–87]. The use of some tools at the host level may not provide the most efficient way of their applications. For example, Snort may be used as a HIDS, but it could serve a better purpose if set up to monitor the subnet where the host resides. However, as the literature points out, Snort would still be useable as a HIDS. The HIDS tools in this category are as follows:

1. *OSSEC* [50] is an open-source multi-platform, HIDS tool. It uses its signature-based methodologies to analyse the traffic it collects. Its data sources include logs, rootkits, registries, processes, etc. Useful practitioner support, modifications, tricks, and techniques are widely shared by the user community.

2. *NTA*, which is SolarWinds' Security Event Manager security event manager (SEM) [25], is a signature-based HIDS that can be configured to perform automated remediation, making it an IPS as well. It can perform real-time event analysis and notify the administrator through its extensive reporting capability. Its universal serial busses universal serial busses (USB) monitoring capability can be useful for insider threat detection. NTA uses its NetFlow collector and analyser to show traffic sources and destinations, the top talkers, the highest bandwidth consumers, etc. It encrypts log data to protect it from unauthorised access. Its centralised log repository also allows for easy access and rapid search capabilities.

3. *Samhain* [88] is a multi-platform, free, and open-source signature-based HIDS tool that can be configured to run in a stealth monitoring mode, preventing attackers from noticing that it is running. Its capabilities include file integrity checking, logs monitoring, rootkit detection, etc. This HIDS, which can also be used as a host monitoring tool, can provide output reports in many formats.

4. *ManageEngine EventLog Analyzer* [44] is a signature-based security tool that has both HIDS and NIDS capabilities. It can collect a variety of logs for central storage and analysis. It provides real-time analysis that provides statistical data, which can be useful for monitoring capabilities as well.

Other tools that are used as NIDS, such as Splunk [39], Snort [9], Sagan [71], etc., are also well-suited as HIDS.

3.2.3.2 Research methodologies

The arbitrarily selected state-of-the-art research works in this area are as follows:

1. Zhang et al. [89] proposed a HIDS that is able to produce early detection of intrusions in the Linux environment. They developed an anomaly detection HIDS using ML models based on the partial analysis of syscalls that are invoked by their execution.

2. Deshpande et al. [90] proposed a cloud-based HIDS that is based on system calls (syscalls) [91]. The approach analyses syscall traces and alerts the user when malicious activities are identified.

3. Besharati et al. [92] proposed a HIDS anomaly detection system for the cloud environment. Their approach uses multiple ML techniques to produce an accurate low TRL HIDS.

4. The Ontology-based HIDS proposed by Can et al. [93] is based on comparing ontologies to a known repository of malware signatures. The signature-based HIDS scans for malware on the host and alerts matches to previously known signatures.

5. Maske et al. [94] proposed an anomaly-based HIDS. Although their training dataset is limited, their approach analyses and applies a decision engine, the extreme learning machine, to syscalls to detect activities that are anomalous.

3.3 Extrusion detection

Extrusion detection tools focus on analysing system activity and outbound traffic. The aim is to detect malicious users, malware, or any other traffic that could be harmful to other assets in the network. This allows for the detection of attack attempts that may originate from already compromised systems, thereby preventing them from reaching their intended targets or containing the attack. Data for such detection is often collected from monitoring at leaf nodes in the network.

3.3.1 COTS products

Some monitoring and IDS tools such as Snort, Wireshark, Palo Alto, etc., can be configured for extrusion detection as well. But, there are tools that are more specialised to extrusion detection. The followings are some commonly used COTS tools for extrusion detection.

1. *Varonics* [74] allows network owners the ability to track, visualize, analyse, and protect unstructured data. That includes insider threat detection and data exfiltration.

2. *Argus* [33, 34] is used for monitoring all aspects of an enterprise network. QoSient explicitly states that Argus can detect exfiltration attempts from the network.

3. *Deep Discovery Inspector* [95], provided by Trend Micro, is a sandboxing tool that comes as a physical or virtual appliance. It detects advanced malware "that typically bypasses traditional security defenses and exfiltrates sensitive data."

NBAD/UEBA tools, which profile and monitor user and entity behaviors, are excellent tools to detect extrusion attempts. Based on learned behaviour, they can detect and alert on anomalous and potentially malicious activities that could be data exfiltration attempts. Some of the arbitrarily selected COTS tools in this category are: Varonics [74], Palo Alto [96], Fortinet [97], Securonix [77], Splunk [39], etc.

3.3.2 Research methodologies

Some of the current arbitrarily selected state-of-the-art activities in this area are as follows:

1. Nadler et al. [98] proposed an approach to detect tunneling and data exfiltration over the domain name service (DNS). Their approach categorises internet domains that are used for data exfiltration and blocks them to prevent data exfiltration.

2. Bubnov [99] proposed a Partially Observable Markov Decision Process (POMDP) approach for detecting data exfiltration by DNS tunneling. At each egress point, the approach enforces a DNS query blocking strategy based on a predefined parametric POMDP formulation.

3. Ahmed et al. [100, 101] applied an ML-based technique (iForest) to detect DNS exfiltration and tunneling in real time. They trained and successfully tested their approach on DNS traffic data collected from a large university and a government institute.

4. Le et al. [102] proposed a genetic programming genetic programming (GP) based methodology for malicious insider threat detection. By allowing a previously trained GP set to adapt, their approach was able to detect potential malicious activity with detection rates comparable to ML techniques.

5. Fadolalkarim and Bertino [103] proposed detecting possible malicious insider data exfiltration by creating and learning user activity profiles. Their approach then compares learned profiles to daily usage to detect possible malicious activities.

3.4 Data leakage detection

DLD for data leakage protection (DLP) supports access control to an enterprise's data. This prevents end-users from stealing or destroying an organisation's valued data. In this case, an intruder may manage to reach the organisation's data, but exfiltration or destruction attempts would be detected. There is a wide variety of tools that cover DLP. Such tools can be stand-alone IPSs that may work hand-in-hand with the IDSs, SIEM systems, endpoint protection appliances, and anti-malware systems.

DLP can be challenging because of the multiple steps involved in its implementation. It requires tools that:

1. Identify where sensitive data is stored
2. Group data according to its sensitivity or security policy definitions
3. Track and log all access to sensitive data
4. Track and log device configuration changes
5. Track log changes to detect unusual activities
6. Track, prioritise, and implement up to date software patching
7. Monitor and secure external connections

Some of these activities can be performed by some of the tools and methodologies discussed above. But, there are tools and methodologies that are specific to DLP as discussed next.

3.4.1 COTS products

1. CoSoSys' *Endpoint Protector* [104] specialises in on-site and cloud-based DLP systems. It monitors and protects sensitive data with its "advanced multi-OS data loss prevention." Endpoint Protector also monitors and protects universal serial busses (USBs), attached devices such as cameras, and peripheral ports to stop data theft and data loss. Its implementation requires a central Endpoint Protector Server appliance that communicates with client software installed on each endpoint in the network.

2. *Solarwinds* [25] is a SEM that manages Snort data. It is provided with a broad set of rules, and can also be used as an IPS and a DLP tool. The monitoring capability tracks data access and generates alerts when the data is copied or transferred. It monitors users through a combination of information from the Active Directory (AD), Windows File Share, SharePoint, and Microsoft Exchange. That way, users' activities can be monitored and unusual activities detected, although it is not clear how it can tell that an activity is unusual.

3. *Symantec's* DLP [105] tool monitors, discovers, and protects sensitive data in the organisation. It can be applied in cloud applications, endpoints, networks and data centers. It can also monitor data held on servers, desktops, mobile devices, or cloud storage. Its technique is to initially identify the locations of sensitive data. It then logs all access to sensitive data. Users that have raised alerts before are tracked.

4. *Teramind* DLP [106] monitors user actions and validates them against pre-defined DLP rules and takes courses of action if a rule and/or condition is violated. It uses OCR and natural language processing to scan all documents. Then it prioritizes those that contain sensitive information such as personally identifiable information, personal financial data, and personal health information.

5. *Prevention*, Check Point Data Loss Prevention [107], monitors data to protect against unintentional loss. It achieves that by scanning and securing data passing through the gateway.

6. *SecureTrust* DLP Discover [108] provides content visibility to both monitor and prevent data loss across the network, endpoints, and cloud. By monitoring the network, it can detect insider risk to help identify potential data loss. It scans all communication channels for privacy or policy violations. Such channels include file transfer, email, chat, file sharing, blogs, and social media.

7. *A-DLP*, Clearswift's Adaptive Data Loss Prevention Adaptative Data Loss Prevention (A-DLP) [109] tool, performs deep packet inspection to monitor data to dynamically and automatically detect and redact sensitive data. It is made up of seven packages that monitor endpoint activity, web applications and file transfers, data leaks through email, data access through the web, document access, external and internal email servers, etc.

8. *McAfee* [110] monitors network data. It then classifies the data by sensitivity and prioritises the protection of data.

9. *Digital Guardian* [111] monitors and logs all actions to sensitive data. Then it uses that information to flag suspicious activity.

10. *Fortideceptor* [97] is a deception-based approach to detect data leakage, both for insider and external threats. The tool uses an automated orchestration of decoys and tokens to create a network that fakes critical information technology (IT) assets in the organisation. The tool makes the fake assets indistinguishable from real IT assets.

3.4.2 Research methodologies

There is significant research interest in DLP as evidenced by the number of current research activities in that area [112, 113]. Most of the methodologies have very low TRLs, so they cannot be applied without going through the laborious task of transforming them into applications that analysts can use in an operational environment.

1. Gupta and Kush [114, 115] proposed an approach for DLD through email. Their algorithm, which matches patterns stored in a database, detects possible malicious attempts to steal data through email and raises alerts.

2. Costante et al. [116] developed a white-box (transparent profiles) data leakage detection approach that can detect anomalies in database transactions. Their approach claims very low false positives, which is very good for an anomaly detection methodology.

3. Peneti et al. [117] proposed a timestamp-based approach to detect data leakage problems. Their approach is based on the assumption that data access is controlled by time periods. Their algorithm detects attempted unauthorised access during restricted time periods.

4. Fu et al. [118] proposed a DLD framework for Hadoop. Their approach collects data from a Hadoop cluster, which they analyse in their forensic server to automatically detect and trigger alerts on any suspicious data leakage behaviour. The framework has been evaluated on simulated scenarios.

5. Lu et al. [119] developed a collaborative graph approach for data leakage detection in big data networks named CoDLD. Their approach uses a three-step process to detect and alert on distributed data leaks. The proposed method obtained high accuracy while tested on subsets of hundreds to thousands of documents.

6. Awad et al. [120] designed and implemented Peeper, which is a policy-based data leakage detection system whose operation is based on OS call provenance. Experimental results demonstrated potential of exfiltration detection in real-time. However, the security of the system itself hasn't been explored in this study.

4 Other detection methodologies

In this section we explore other methods that can be used to detect some arbitrarily selected malicious activities. Some of the methods described here can be specialised cases of the tools and methodologies described in preceding sections. Nonetheless, they provide an understanding of how certain malicious activities can be detected and equip defenders with the knowledge they need to prevent similar attacks. Due to the ever-evolving nature of cyber threats, some categories in this section do not have COTS tools, but have emerging methodologies that have been published in open literature.

4.1 Sandboxing for malware detection

Malware sandbox tools detect malicious activities by confining the actions of applications, such as opening an MS Excel document, to an isolated and controlled environment–the sandbox. Analytic methodologies in the sandbox then analyse the object and its behaviour within this controlled and safe environment. The analysis allows defenders to understand the object's characteristics and behaviours, thus enabling the discovery of possible malicious objectives. Malicious code triggers an alarm and courses of action (COAs) are taken.

Malware detection also includes the detection of backdoors, which can be ransomware, spyware, or trojans. Backdoors come in the form of malware deliberately installed (sometimes with the unknown complicity of the user) by manufacturers or cyber criminals to allow system access and elevate privileges [121].

4.1.1 COTS Products

The following arbitrarily selected COTS Sandboxing tools are used to detect malware:

1. Sonicwall's Capture Advanced Threat Protection (ATP) [122] detects and stops unknown threats at the gateway. It is a cloud-based multi-engine sandbox that analyses suspicious code to detect and block known and unknown (zero-day) malware.

2. McAfee [123] can perform static code analysis as well as dynamic analysis through its sandboxing capabilities to detect malware. It also utilises machine learning to increase zero-day threat and ransomware detection.

3. FireEye [124] has the Malware Analysis (AX series) tool that provides a sandbox solution to test, replay, characterise, and document advanced malicious activities.

4. Trend Micro's Deep Discovery Inspector [95] is a sandboxing tool that comes as a physical or virtual appliance that can detect and prevent breaches. It detects advanced malware "that typically bypasses traditional security defenses and exfiltrates sensitive data" [95].

5. The Sandblast [125] threat emulation tool uses the sandbox to detect malware. Its detection capability is able to identify and alert on previously unknown malware.

6. Cuckoo Sandbox [126] is an open source automated malware analysis system that provides an automated analysis of suspicious files. It analyses the behaviours of suspected files in an isolated sandbox environment, and detects possible malicious files. Because of its simplicity and effectiveness, it has found extensive use with researchers.

7. FortiSand [127] consists of a detection and analysis engine to detect and capture malware characteristics and techniques. It also claims to be able to identify zero-day threats, and to have AI detection capabilities. Another sandbox malware detection tool from Fortinet is Forcepoint [128], which claims zero false positives in detecting even the newest and advanced malware.

4.1.2 Research methodologies

Complementing the above COTS tools, the following emerging research activities propose approaches to detect cyber malicious activities:

1. Amer et al. [129] proposed a sandbox-based malware detection approach in a windows environment. Their approach clusters functions that have similar contextual features, and use Markov chains to predict malicious content through application programming interface (API) call sequences.

2. Wang et al. [130] proposed a malware detection technique based on a sandbox. Their approach uses multidimensional feature extraction to train a classifier to identify possible malicious activities.

3. Hansen et al. [131] presented a random forest based classifier to detect malware in a sandbox. Their approach detects both known and unknown malware, while providing malware classification as an additional capability.

4. Wysopal and Eng [132] propose an approach to detect application backdoors, which are code embedded inside legitimate applications.

5. Ali et al. [133] developed MALGRA, an ML-based adaptive technique to detect malicious code in a sandbox. Based on their dynamic analysis technique, they are able to identify indicators of compromise, which they use as classification features in their approach.

4.2 Deception techniques to allow detection of malicious activities

Computer security deception is defined as being those actions taken to deliberately mislead attackers and to thereby cause them to take (or not take) specific actions that can aid cyber defenders [134]. It is a signature-less and behavior agnostic threat detection methodology, whose success predominantly depends on obscurity through lures and decoys to entice, engage, misdirect and ultimately assist to detect attackers. However, according to an old security adage that says security through obscurity is no security at all [135, 136], it can be argued that deception adds no security layers to networks. Fortunately, by observing the adversary's activities and TTPs, defenders can strengthen their defence knowing what attackers are interested in. In this section we present some of these deception techniques, and explain how they assist in detecting malicious activities.

4.2.1 Fakes and decoys

As the name implies, fakes and decoys are network nodes that look real, although they are not.

4.2.1.1 COTS tools

Some known tools and methodologies on this topic are:

1. ShadowPlex [137] deploys decoys, which look like fully functional computer hosts, throughout the network in order to confound attackers and detect their TTPs as they manoeuvre around the network. The decoys divert attacker attention from real assets while alerting of their presence.
2. TheatDefend [138] provides endpoint deception through the deployment of decoys throughout the network. It also allows defenders to harvest attacker TTPs with which they can further defend their networks.
3. TrapX [139] DeceptionGrid, is a deception network solution that emulates traps, which are decoys. The tool analyses traffic and triggers alerts whenever intrusions are detected.
4. Countercraft [140] uses a combination of decoys and fakes to confound the attacker. Based on the activity detected, it raises alarms and collects TTPs. It can be deployed across a variety of platforms, such as servers or desktop computers.
5. Illusive Networks [141] provides deception through decoys and traps. It allows defenders to collect adversarial TTPs, which can be used to strengthen network defences.

4.2.1.2 Research methodologies

1. Chakraborty et al. [142] proposed an approach that duplicates every repository document with its fake. That way, an attacker would spend significant time to identify real documents from fakes, increasing their chances of being caught and allowing defenders to collect attackers' TTPs.
2. In an effort to increase the attacker's time and effort, Karuna et al. [143] developed a text manipulation approach that modifies the comprehensibility of a text document. Their approach uses a genetic algorithm that manipulates real documents' comprehensibility to hard-to-comprehend, but believable fake documents. This has the potential to confound attackers allowing defenders to collect TTPs.

3. Whitham [144] analysed the weaknesses of existing fake generation techniques, and argues that they are a major hindrance to the wider adoption of the technique in preventing document theft. He then goes on to recommend a set of requirements for generating fake documents that could be effective in confounding attackers, thereby increasing the chances to detect possible malicious activities.

4. Skillicorn et al. [145] attempted to reduce the chances of cyber documents exfiltration by creating many fake versions of all critical documents. For the attacker to be successful, they would need to exfiltrate all the document versions, an option that increases their chances of getting caught.

5. Sharma [146] attempted to tackle insider data theft by creating decoy documents. The approach profiles behaviour through challenge questions before presenting the potential thief with generated fake documents.

4.2.2 Honeypots

Honeypots are computer systems, mostly virtual machines (VMs) or services, which are configured to deceive attackers and make them believe that they are interacting with a real system. The technique can be more effective than trying to spot intrusions using classical IDS techniques in that it is not supposed to get any traffic. So, any activity that reaches it is likely to be a probe or an attack attempt.

There are many honeypots that can be specifically developed to detect a particular type of attack or general attacks on particular services. So, we arbitrarily selected some examples for this report. The honeypot tools we identified are as follows:

4.2.2.1 COTS tools

1. Illusive Networks [141] has COTS tools that use honeypots to deceive attackers and collect their TTPs. The honeypots are used as decoys and traps as described above.

2. Kippo [147] is a Linux-based Secure Shell (SSH) honeypot that can also offer fake file systems and content to attackers. It can log brute force attacks as well as the complete shell interaction with the attacker, allowing defenders to collect attacker TTPs.

3. ElasticHoney [148] is an example of a database honeypot whose objective is to detect and prevent Elasticsearch attacks, which happen frequently in the wild. This is a lightweight tool that can catch malicious activities attempting to exploit remote code execution vulnerabilities.

4. Honeymail [149] is an Simple Mail Transfer Protocol (SMTP) honeypot that is touted to stop SMTP-based attacks. By utilising configurable email response messages, Honeymail can detect and prevent attacks against SMTP servers. It also has the capability to protect against DDoS attacks due to “massive connections.”

5. Snare [150] is a web application honeypot that captures malicious activity from the internet. It logs the activity for analysis to provide defenders with possible attacker TTPs.

4.2.2.2 Research methodologies

1. EMPHAsis [151] is a conceptual model for malware detection using honeypots. It uses high interaction honeypots to collect and analyse data for possible malicious activity.
2. Chong and Koh [152] demonstrated the use of honeypots in cyber deception to detect malicious activities. They implemented active honeypots that also provided deceptive responses while tracking attacker activities. They were able to collect TTPs of attackers.
3. Djanali et al. [153] proposed an approach to use a honeypot to increase a system's ability to detect SQL injection attacks. Similar to decoys and fakes, their approach places a honeypot to hide an actual web server from attackers. Attack queries are sent to the honeypot, allowing defenders to identify the attackers and their TTPs.
4. Mia et al. [154] presented a game-theoretic deception model to make decoys, such as honeypots, less distinguishable from real objects. Their approach makes it difficult for attackers to distinguish between real and fake objects, thus making attackers spend more time and resources in their attacks, thereby increasing the chances of being detected as they move around.
5. Changwook-Park et al. [155] proposed a deception tree model to implement a cyber-deception operation. Their approach can distinguish fake targets as either human or machine from both attack and defence perspectives. In the process, the defender is able to determine the attacker's TTPs to better defend their networks.

4.2.3 Other deception methodologies

The techniques in this section are a mixture of low TRL concepts and mature tools and methodologies.

4.2.3.1 Concealment

This is one interesting technique that does not seem to have taken off. But the idea is very ingenious. For example Yuill et al. [156] develop a model for deceptive hiding, a technique that could confound the attacker and make them expend significant resources looking for objects where they should be, but they are not, thus increasing their chances of detection. Yuill et al. say that their deceptive hiding defeats one or more of the attacker's kill-chain processes, catching them in the act or at least collect TTPs.

As an example in concealment, consider the files that are important to an attacker. Blocking access to these files could either prevent an attack or its propagation. It is up to the network defenders to identify and prioritise the importance of these files and hide them accordingly. For example, network defenders can store a fake password file at `/etc/passwd` or `/etc/shadow`, and then store the real one in say `/.secure/passwd` where no one expects it to be. That way, when attackers come looking for a password file, they would get useless information. As another example of files that attackers are usually interested in to erase their trails, network defenders could also hide `syslog` file and move it from the `/var/log` folder to somewhere under a different name where attackers do not expect it to be.

This technique, though valid, has many challenges. The OS needs to be made aware of this change, and there is no easy way to do it. Again, social engineering and spear phishing attacks can still reveal information to the attacker. Caution should also be exercised when hiding things. First, defenders have to be able to locate the things when they need them, and it should be easy for maintenance. In addition, it is necessary to have a "fall-back" plan. If attackers are desperate to delete the `syslog` file and can't find it in the `/var/log` folder where it is supposed to be, they might have to format the hard drive [156], which could result in a more serious damage.

4.2.3.2 Response crafting

In this approach, responses to attack messages can be crafted so as to throw attackers off their trail. For example, spear phishing email could be responded to while providing fake information. Attackers' attempts to launch attacks based on that fake information would fail, and their attempts can be detected allowing for attribution or the collection of TTPs.

The work by Al-Shaer [157] has a component that crafts responses to attackers. The approach intercepts TCP sessions and then crafts responses back to the potential attackers to mislead them. In earlier work, Rowe et al. [158] develop a prototype of an "intelligent software decoy." The decoy, on detecting suspicious activities (based on policy rules) on the network, maintains communications with the potential attackers while providing them with deceptive information that their attacks are successful. Since the decoys are software wrappers around applications and require monitoring, this technique would be difficult to implement on large networks. Both methodologies have very low TRLs. A higher TRL approach was proposed by Rowe and Goh using SnortInline to manipulate reconnaissance packets of honeypot attackers [159].

4.2.3.3 Aliasing

This is an old deceptive technique, that researchers have used in the aliasing of multiple addresses to confuse attackers. In the process, defenders are able to track attackers and harvest TTPs. The Deception Toolkit (DTK) [160, 161] is an old, mature tool that aliases an existing host into an imaginary host that has multiple currently known vulnerabilities. As the attacker attempts to attack the host, they will expend significant amounts of time attacking a non-existing host, thereby increasing their chances of being caught. Unfortunately, we couldn't find follow-up work on this technique, and we presume the work to be abandoned or came to an end.

4.2.3.4 Protocol diversity and manipulation

When attackers launch attacks, they expect to receive responses from the intended targets. Those responses can help the attackers to establish the success or failure of their attacks. For example, to map a network, Nmap relies on the responses it gets from the target to establish the target IP address, type (e.g., Unix or Windows), OS version, and what services are running on the system. Network defenders can thwart these efforts by denying the attackers this information, or better still, give them wrong information. In the process, attackers' motives and TTPs can be identified, improving chances of attribution and the strengthening of perimeter defences.

One approach of achieving this misinformation is through manipulating protocols so that the system responds with misleading information as described earlier in the use of SnortInline by Rowe and Goh [159]. Rowe et al. [162] generalise that technique by introducing protocol heterogeneity through a "new" set of protocols which are based on a finite state machine. The "new" protocols meet the behaviour of the old protocols, but do not include unnecessary or vulnerable behaviour that can open doors for attackers. Bilinski et al. [163] propose a game theoretic approach that manipulates the information available to the attacker. That allows the defenders to identify and learn the potential behaviours of the attackers.

4.3 Techniques to detect evasive malicious activities

Attackers can use many evasive techniques to avoid detection. They could achieve that through obfuscation, The onion router (Tor), virtual private networks (VPNs), etc. The detection techniques and methodologies discussed so far (e.g., intrusion and extrusion detection, data leakage detection), combined with operator skill and art, could be used to detect such evasive techniques. However, there are specialised tools and methodologies that have been specifically developed to detect attacker evasion. We present some of these tools and methodologies in this section.[6]

4.3.1 Obfuscation

To evade detection, attackers often obfuscate their malicious data. Obfuscation is a systematic mangling of data to confuse whoever tries to read that data without knowing how to reverse the obfuscation process. The obfuscation or packing [164–166] of data is achieved through compression, minification [164], encryption, and other methodologies of varying complexities. For example Javascript is often obfuscated for malicious purposes, allowing attackers to bypass security tools.

The majority of the malicious traffic detection discussed earlier in this paper are mostly applicable to plain traffic with a plain payload. If the traffic or its contents are obfuscated,

[6] We are aware that there are many other evasive malicious activities out there. However, we arbitrarily elected to present these in our work.

the techniques are likely to fail, allowing attackers to evade detection. There are a number of techniques to detect obfuscated malicious traffic, we discuss some of them in this section.

4.3.1.1 COTS tools

The following are the some of the common obfuscation detection tools in use:

1. The RDG[7] Analyzer by Malware-Analyzer [167] is a mature obfuscation detection tool that can detect compilers, encryptors, or packers used to obfuscate data. It can even unpack some of the obfuscated data, providing analysts with knowledge on how to handle encrypted malware in their networks.
2. Opentext's Windows Executable Packer Detection tool [168] can analyze windows executable files to detect any packing or encryption of malicious data.
3. The signature based packer detection signature based packer detection (SPADE) methodology by Naval et al. [169] is another packer detection methodology that is based on known signatures.

4.3.1.2 Research methodologies

There is significant low TRL activity in this category. Some of the approaches are as follows:

1. Nachenberg patented a technique to detect obfuscated extrusion attempts of an organisation's data [170]. His low TRL technique detects outgoing obfuscated data at the gateway and/or the client workstation.
2. In their work Xu et al. [171] outlined some of the common obfuscation methodologies used in malware. They then proposed their approach to detect obfuscated malware on the Windows platform.
3. Bat-Erdene et al. [172] propose an approach to detect packed and repacked malware. Zhang et al. [173] introduced entropy classifiers to detect encrypted bot traffic.
4. Biondi at al. [174] proposed a machine learning based approach to detect packing. The authors based their machine learning algorithm on a ground truth that they created themselves and features that they extracted to maximise detection. A similar machine learning approach was also used by Tellenbach et al. [175]. These approaches imply very close to signature-based learning, but would still be very effective if the ground truth is updated often.
5. Li et al. [176] developed the Entropy Signal Reflects the Malicious Document Entropy Signal Reflects the Malicious Document (ESRMD), which is an entropy-based approach to detect email-based phishing cyber attacks. It can detect obfuscated malware based on the entropy signature of the file.

[7] The vendor does not state what RDG stands for.

6. Anderson and McGrew [177] developed a supervised machine learning approach to detect encrypted malware. They base their approach on training collected and correlated transport layer security (TLS), DNS, and HTTP traffic metadata.

7. Lin et al. [178] proposed an approach to detect encrypted malware. Their approach uses a Naive Bayes classifier (PP-NBC) to recognize potential malware instances from API call fragments.

8. Vidal et al. [179] proposed a statistical approach to detect malware through mimicry [180], while Yi et al. [181] employed a branch sequence model to detect and prevent mimicry attacks.

4.3.2 Steganography

Another evasive transmission method is steganography [182,183]. This is a process of hiding data in images, information that could be used for adversarial malicious activities. Its detection is not trivial, but there are tools and ongoing research on detecting steganography. StegDetect [184, 185] is an open-source automated tool that can detect steganographic information in images. It can detect information embedded by using several different steganographic methods in JPEG[8] images. Wetstonetech's StegoHunt Suite [186] can be used to detect the presence of both steganographic data and the software used for the data hiding. StegoHunt is also able to detect the carrier files. Researchers have also developed approaches to detect steganography in JPEG images. For example, Andriotis et al. [187] propose an approach that can detect both the steganographic information and the algorithm used to embed the information into the image.

4.3.3 Covert channels

Cyber covert channels involve the transmission of malicious information between subjects [188–191]. Such malicious activities include data exfiltration, botnet command and control (C2) channel, etc. Their detection is so complex that this area is of significant research interest. Although there are many types of covert channels, researchers have focused on two main categories [188], covert storage channels (CSC) and the covert timing channels (CTC). High TRL covert channel detection tools are virtually non-existent. However, skilled and determined analysts can use some of the COTS tools described earlier, such as Snort, tcpdump, Wireshark, etc., to detect some covert channels [192].

A list of some arbitrarily selected methodologies from the extensive literature in this area is as follows:

1. Chen et al. [193] proposed an approach that uses Long Short Term Memory (LTSM) networks to detect DNS covert channels. Their approach uses the DNS packets' fully qualified domain names fully qualified domain name (FQDN) as the LTSM input for training, achieving very high detection accuracy rates.

[8] An image file format that stands for Joint Photographic Experts Group (JPEG).

2. Han et al. [194] proposed the use of k-NearestNeighbor k-NearestNeighbor (kNN) algorithm to detect CTC. Their approach is based on the time interval and payload statistics to train a ML model to detect CTC-based malicious activities.

3. Ayub et al. [195] proposed a supervised ML CSC detection approach that detects malicious activities independent of the protocol used. Their approach was also used to generate data that can be used for CSC detection of IP, TCP, and DNS traffic.

4. Gunadi and Zander [196] extended Bro (now Zeek) by developing a plugin that detects covert channels. Their approach uses mathematical models such as Kolmogorov-Smirnov Test, Entropy, etc. to develop metrics using time, modulation, etc. to detect the covert channels.

5. Saeli et al. [197] proposed a ML approach based on their covert channel anomaly detectors. Their model performs user behaviour analysis to create user profiles, thus allowing for the detection of deviations from the norm. They claim that their unsupervised ML approach can detect zero-day attacks.

4.3.4 The onion routing (Tor)

Tor is open-source software that enables anonymous computer network communication in addition to providing encryption. Tor traffic is designed to look like normal hypertext transfer protocol secure (HTTPS) traffic and is routed through an overlay network of volunteers that is made up of several thousands of relays that hide the user's identity and activities from anyone performing monitoring network traffic, making it difficult, if not impossible, to detect using the classical detection tools described above. We present some tools and methodologies to help operators understand the malicious Tor-based activities they may face.

4.3.4.1 COTS tools

Mature Tor detection methods vary from code snippets to COTS tools as follows:

1. Bjerke and Roy [198] offer code snippets to detect Tor traffic. Defenders can adapt such code to their Tor detection needs.

2. LogPoint [199] claims to detect Tor users. It uses logs from firewalls, proxy servers, and endpoint to identify the source of the Tor endpoint connection.[9]

3. CapLoader [200] is a Windows-based tool that can assist in identifying Tor sessions through a statistical analysis of protocols it identifies. It's capability does not depend on port numbers as most IDS tools do. Because Tor does not only encrypt traffic, it is designed to look like HTTPS traffic, making it harder to detect. But, analysts can use CapLoader to distinguish between different types of Secure Socket Layer (SSL) traffic, which can ultimately reveal the Tor traffic.

[9] Advanced adversaries can still use VPNs and Tor bridges to circumvent detection and blocking.

4.3.4.2 Research methodologies

There is interest in identifying Tor traffic, the majority of which use ML approaches. We arbitrarily selected the following methodologies for our work:

1. Mayank and Singh [201] proposed an approach to detect Tor traffic through an analysis that characterises the TLS link, which is supposed to make the connection secure. From that analysis, they were able to detect and block the Tor traffic originating from a Tor browser.

2. Saputra et al. [202] proposed a deep packet inspection approach to detect Tor traffic. Their approach then uses the identified pattern as a proxy server signature to block the traffic.

3. Lashkari et al. [203] proposed a deep learning approach to detect Tor traffic. They used time-based features as the training and detection characteristics to detect the malicious traffic. Sarkar et al. [204] and Cuzzocrea et al. [205] also proposed ML to identify Tor traffic, with very good success rates with test data.

4.4 Attacks on security tools

As cyber adversarial sophistication increases and continuously evolves, attackers have targeted the network defence tools to prevent them from detecting the attackers [206, 207]. For example, the recent attack on SolarWinds' Orion network management products [207], shows that well-funded nation state actors can compromise security tools to achieve their goals. Such attacks, which involve sophisticated methods, are very hard to detect and would require a combination of some of the malicious detection tools described above and determined skilled analysts. Some computer network defence (CND) communities often share hints and signatures to detect identified sophisticated attacks. For example, following extensive forensic analyses involving network traffic, log and configuration files, etc., FireEye set up a GitHub repository [208] to share workarounds to detect the SolarWinds attacks [207]. Although such workarounds are not tools *per se*, they are the most mature short-term solutions to detect similar malicious activities.

In simpler instances, attackers are known to evade known tools. There are tools such as nMap or Metasploit Framework that can evade firewall rules. Attack tools such as Metasploit Framework or Saint can also be configured to attack vulnerable IDS software; particularly if the software's unpatched vulnerability is publicly known.[10] These tools have signatures that defenders can use with signature-based tools to detect possible adversarial activities originating from them.

As discussed earlier, many detection tools are embracing AI-based technology [72, 73, 76, 209]. However, it has been shown that AI systems could be fooled by being trained with

[10] Advanced attackers can also exploit zero-day vulnerabilities.

poisoned or misleading data [210–213]. In this section, we present a few arbitrarily selected methodologies that could detect such adversarial activities.

4.4.1 COTS Tools

We are currently not aware of specific COTS tools that can be used to detect specific malicious activities against cyber security tools. However, the COTS tools that we have discussed so far, combined with the efforts of skilled and determined analysts, can be selectively combined to identify malicious activities aimed at cyber security detection tools. We, therefore, are not re-listing these tools here.

4.4.2 Research methodologies

Although we found no specific COTS tools to detect malicious activities aimed and defence tools, there is significant research work in this area as evidenced by the following arbitrarily selected publications:

1. Qiu et al. [214] developed a methodology to mitigate attacks on artificial neural networks (ANNs). Their approach, which mitigates against advanced gradient-based attacks, claims to be effective against state-of-the-art techniques.

2. Paudice et al. [215] proposed an anomaly detection based defense mechanism against data poisoning attack. Their approach is a generalized defence mechanism that is not specific to network traffic data.

3. Shan et al. [216] proposed a honeypot approach to detect attacks against defensive ANNs. Their approach injects trapdoors as honeypot vulnerabilities and classification categories, which attracts attackers seeking adversarial examples to exploit. Attackers' activities are then detected as their optimisation algorithms converge to the trapdoors.

4. Pawlicki et al. [217] proposed an adversarial ML approach to detect attacks on ML-based IDSs. Their approach was successfully applied to detecting four different adversarial evasion attacks.

5. Usama et al. [218] proposed a generative adversarial networks generative adversarial networks (GANs) approach to mitigate against possible malicious activities aimed at ML-based IDSs. Their approach is said to inoculate the ML-based IDS to make it more robust to possible malicious activities.

6. Aiken and Hayward [219] investigated perturbation based adversarial attack mitigation against ML-based IDSs within SDN. Based on their test results, they made recommendations to improve the robustness of ML-based IDSs.

5 Discussion and next steps

In this section we summarise our findings. We also propose possible future work activities on the gaps that we identified.

5.1 Monitoring

We identified tools and methodologies to monitor the network to detect potential malicious activities. The majority of the tools come with their own analytics engines as well. The most significant gap that we found was monitoring the high-speed side of the network. A significant number of old tools would not be able to monitor the network at high speeds without losing packets. However, recent advances in monitoring hardware, such as NetShark [22] or PacketScan [24] have shown that high speed monitoring is possible. Recent research activities aim to push the high speed monitoring speeds to higher limits. Most COTS literature makes claims that do not often stand up to validation scrutiny. Therefore, a possible future work activity in this regard would be to validate the line speed captures that are claimed by the high speed products, and confirm that there are no packets lost during the monitoring.

We also talked about the placement of sensors to detect malicious activity. However, the literature does not say or define what the most optimal sensor placements could be. Haphazard placement can lead to blind spots and unnecessary collection of redundant data. We think that a possible research activity could determine the most optimal placement of sensors to maximise detection while minimising the cost of acquiring and maintaining the sensors.

The enterprise-wide data collected from the monitoring tools need to be stored in repositories that are easily accessible (e.g. through indexing) to analytics tools. In large organisations, such data storage can be made complicated by its high volumes. We envision such storage to require well-defined strategies to allow for efficient storage and data retrieval. This is an area that is continuously evolving, and possible future research work could help identify and establish organisational tools and methodologies for data storage and its management.

5.2 Intrusion detection

The two major categories of intrusion detection are signature- and anomaly-based. The COTS products as well as recent literature claim significant efficiency in detecting malicious activities based on these approaches. However, the signature-based methods are not very efficient in detecting zero-day attacks. Some remedies to this problem have been proposed in recent works [80] by leveraging deep learning and AI techniques to generate signatures of attack variants. Although such methods seem promising, the performance needs to be evaluated exhaustively with a wide variety of zero-day attacks, an activity that we would recommend as a possible future work activity.

On the other hand, anomaly-based intrusion detection methodologies can identify and alert on possible attacks by measuring deviations from the normal. However, anomaly-based intrusion detection can generate a significant number of false alarms, which may add overhead to the analysts. We also observed that many recent anomaly-based intrusion detection methods reported poor performance on detecting some attacks such as backdoor, worm, shellcode, etc. due to the scarcity of publicly available data to train them properly. Anomaly detection, which includes among many other challenges, detection of coordinated attacks, flow measurements, and encrypted data, is therefore an area that still needs significant research input. We recommend such activities for possible future work.

In addition, UEBA and NBAD tools and methodologies rely on learning network or entity behaviours and then alert on anything outside the norm. However, advanced attackers have been known to infiltrate victim networks and stay there for very long periods of time, while collecting valuable information and sending it out just like any regular user would. Under such conditions, there is a danger that UEBA and NBAD methodologies would consider the behaviour of the intruders that are already inside the network as *normal activity*, and won't trigger alerts. Therefore this methodology of detecting malicious activity should be complemented with other defence mechanisms. We defer the study on how that could be achieved to possible future work activities.

5.3 Other specialty methodologies

We have explored a number of arbitrarily selected specialty methods of detecting malicious activity in the network. As mentioned in the report, the classic detection tools described in Section 3 can be used, with more effort and analyst skill, to detect malicious activities targeted by these methods. In most of the categories, there is significant recent research interest, which is mainly targeting a limited set of detection types using anomaly detection techniques.

As mentioned in Section 3, deception techniques such as decoys and honeypots do not add any security layers to the network. Their defence is premised on obscurity. However, that may not stop a determined attacker from hitting a real target. If that happens, there is a possibility of significant losses. Network owners should be aware of the level of risk they are exposed to, and what their risk tolerance is. This type of analysis has not been attempted, so we recommend possible future work to explore methodologies to give network owners guidance on the levels of risk they are exposed to when using such methodologies.

Detection of malicious activity in obfuscated content is one challenging area that is attracting some research attention. AI-based methods have been researched and continue to be improved. However, the anomaly detection methodologies they produce have a narrow focus, making it difficult to adopt them in practical solutions that would require multiple such methodologies into applications. Since this is still an active research area, we recommend possible future work to look into these approaches with the aim of improving them or integrating them into mature COTS tools.

Attribution of network activity is still a challenging problem, with very few concrete solutions. In a world where it is important to know who your adversaries are, it is important to find an effective solution to the use of Tor. We therefore recommend possible future research work to look into this topic.

5.4 Detecting attacks on security tools

Cyber security tools that we have looked at in this report aim to defend the network against possible malicious activities. It is therefore obvious that if an attacker wants to launch malicious activities undetected, they could consider attacking or disabling CND tools.

There are tools such as nMap that can evade firewall rules. Attack tools such as Metasploit Framework can also be configured to evade perimeter defences. They can take it further as attack tools against vulnerable IDS software. However, we are not aware of any specific COTS tools that are specifically for attacking other security tools such as Snort. There are signature-based tools, such as Snort, that can be configured to detect malicious evasion activities. Metasploit Framework traffic is mostly encrypted, so obfuscation-based detection techniques could detect attempted evasion techniques. Detection of attacks on vulnerable IDS software may be easier if the vulnerability is known, so detection signatures could be written in IDS tools such as Snort. Such an approach would not help if the vulnerability is a zero-day.

One way to detect possible attacks on security tools is to provide a tool with dynamic signatures of possible attack TTPs of adversaries that are known, through intelligence gathering, to be targeting security tools. Since such a tool or methodology does not exist, we recommend future research activities could consider such a problem.

Most organisations have well-defined vulnerability, remediation and recovery procedures. The procedures do not take into account possible vulnerability holes that can be created when security tools are attacked as what happened to Orion [207] or CCleaner [206]. This is a challenging situation especially when the attack is a zero-day and there are no workarounds to protect the network and no clear methods to detect such activity. We recommend possible future work to explore best practices to handle such cases.

5.5 Detecting attacks on AI tools

We are aware that many cyber tools are embracing AI technology, which have complemented classical malicious activity detection methodologies. However, it has been shown that AI systems could be fooled in many ways [212, 214, 220, 221]. To prevent this from happening, defenders could detect and block malicious attempts to feed misleading data to AI-based cyber tools.

We envisage that the malicious attempts could involve compromising the training data and inserting misleading information. There are currently no COTS tools to detect such malicious activities that we are aware of. But, there are state-of-the-art projects that propose defense mechanisms by placing honeypots to deceive attackers [216], filtering the poisoned examples before training [215], training the IDS with adversarial examples for robustness [218], etc. However, since these are emerging directions of research, the scarcity of publicly available practical training datasets as well as proper evaluation strategies are obvious. Therefore, we recommend such activities as potential future research work.

5.6 Options selection

In this report, we have presented multiple options of mature COTS tools and recent state-of-the-art methodologies for malicious activity detection. On the surface, it may be difficult for a user to determine or select the best tools and methodologies to use in their environment. We did not address such work in this report. But, we recommend users to carry out, with the help of operations research (OR) specialists,[11] a detailed options analysis on candidate tools and methodologies. That way, the tools and methodologies that provide the best capabilities for the environmental setting as well as the user's budget can be selected for use. We recommend possible future work to investigate how to best implement such an analysis and help the user to determine the best tools and methodologies for their environment.

[11] Defence Research and Development Canada (DRDC) – CORA Research Centre could be engaged to assist in such work.

6 Concluding remarks

In this Reference Document we have performed a literature exploration of tools and methodologies that could be used in detecting malicious activities in the network. The literature covers the diverse network options where the different types of monitoring tools could be set up to enable network defenders to detect such activities in a distributed enterprise network. Data produced by the monitoring devices can be stored and analysed by a diverse set of tools and methodologies to detect possible malicious activity. Recent advances in detection technologies are shown to complement mature commercial off-the-shelf (COTS) tools and address technology gaps that researchers have identified. From the literature, we also identified some gaps that we feel should be further researched on to strengthen malicious activity detection in enterprise networks, and proposed that future research work consider addressing them. One example of such an activity is to understand how adversarial machine-learning (ML) techniques could improve malicious traffic detection in enterprise networks. We plan to undertake that research activity next.

References

[1] IBM (2019), IBM security's cost of a data breach report 2019 (online), https://www.ibm.com/security/data-breach (Access Date: 22 june 2020).

[2] Northcutt, S. and Novak, J. (2002), Network intrusion detection, Sams Publishing.

[3] Bejtlich, R. (2004), The Tao of network security monitoring: beyond intrusion detection, Pearson Education.

[4] Northcutt, S., Winters, S., Frederick, K., Zeltser, L., and Ritchey, R. W. (2002), Inside network perimeter security: The definitive guide to firewalls, VPNs, routers, and intrusion detection systems, Pearson Education.

[5] Noel, S. and Jajodia, S. (2008), Optimal ids sensor placement and alert prioritization using attack graphs, *Journal of Network and Systems Management*, 16(3), pp. 259–275.

[6] Cisco (2020), Catalyst Switched Port Analyzer (SPAN) Configuration Example (online), https://www.cisco.com/c/en/us/support/docs/switches/catalyst-6500-series-switches/10570-41.html#anc6 (Access Date: 24 November 2020).

[7] Gigamon (2020), Understanding Network TAP and Bypass TAP Devices (online), https://www.gigamon.com/products/access-traffic/network-taps.html (Access Date: 24 November 2020).

[8] Gigamon (2020), To TAP or SPAN? (online), https://www.gigamon.com/content/dam/resource-library/english/white-paper/wp-tap-vs-span.pdf (Access Date: 24 November 2020).

[9] Roesch, M. et al. (1999), Snort: Lightweight intrusion detection for networks., In *Lisa*, pp. 229–238.

[10] Johnson, L. (2020), What is a Data Mesh? (online), Trustgrid, https://trustgrid.io/what-is-a-data-mesh/ (Access Date: 24 November 2020).

[11] Moses, B. (2020), What is a Data Mesh — and How Not to Mesh it Up? A beginner's guide to implementing the latest industry trend: a data mesh (online), Towards data science, https://towardsdatascience.com/what-is-a-data-mesh-and-how-not-to-mesh-it-up-210710bb41e0 (Access Date: 3 December 2020).

[12] Amazon Web Services (2020), What is a data lake? (online), Amazon, https://aws.amazon.com/big-data/datalakes-and-analytics/what-is-a-data-lake/ (Access Date: 24 November 2020).

[13] Government of Canada (2020), Technology readiness levels (online), Government of Canada, https://www.ic.gc.ca/eic/site/080.nsf/eng/00002.html (Access Date: 7 October 2020).

[14] Watson, J. (2020), 11 Best Packet Sniffers in 2020 (online), Comparitech, https://www.comparitech.com/net-admin/packet-sniffer-network-analyzers/ (Access Date: 15 December 2020).

[15] DNSstuff (2019), Best 10 Packet Sniffer and Capture Tools in 2020 (online), DNSstuff, https://www.dnsstuff.com/packet-sniffers (Access Date: 15 December 2020).

[16] Wilson, M. (2020), 10 Best Network Monitoring Tools & Software of 2020 (online), PCWDLD.com, https://www.pcwdld.com/best-network-monitoring-tools-and-software (Access Date: 15 December 2020).

[17] Keary, T. (2020), 10 Best Network Monitoring Tools & Software of 2020 (online), Comparitech, https://www.comparitech.com/net-admin/network-monitoring-tools/ (Access Date: 15 December 2020).

[18] Software Testing Help (2020), Top 10 Best Network Monitoring Tools (2020 Rankings) (online), SoftwareTestingHelp, https://www.softwaretestinghelp.com/network-monitoring-tools/ (Access Date: 15 December 2020).

[19] Lofgren, L. (2020), Best Network Monitoring Software and Tools (online), Quicksprout, https://www.quicksprout.com/best-network-monitoring-software/ (Access Date: 15 December 2020).

[20] SentryWire (2020), A Complete Packet Capture Appliance & Network Security Platform (online), https://www.sentrywire.com/packet-capture (Access Date: 24 November 2020).

[21] RSA (2020), RSA NetWitness Network (online), https://www.rsa.com/en-us/products/threat-detection-response/network-security-network-monitoring (Access Date: 24 November 2020).

[22] Riverbed (2020), Five Fun Facts about SteelCentral NetShark (online), Riverbed, https://www.riverbed.com/ (Access Date: 3 December 2020).

[23] Aukua Systems Inc. (2020), Aukua MGA2510 Analyzer (online), https://www.aukua.com/index.html (Access Date: 24 November 2020).

[24] GL Communications (2020), Wire-speed Packet Filter and Captureon 10 Gbps Ethernet Networks (online),

https://www.gl.com/packetscan-all-ip-packet-analyzer.html (Access Date: 23 November 2020).

[25] SolarWinds Worldwide (2020), Solarwinds (online), https://www.solarwinds.com/ (Access Date: 11 June 2020).

[26] Kekely, L., Špinler, M., Friedl, Sikora, J., and Kořenek, J. (2018), Accelerated Wire-Speed Packet Capture at 200 Gbps, In *2018 28th International Conference on Field Programmable Logic and Applications (FPL)*, pp. 455–4551.

[27] Kekely, L. and Puš, V. (2018), Wire-Speed Packet Capture and Replay at 200 Gbps, *pdfs.semanticscholar.org*.

[28] Cardigliano, A., Deri, L., Gasparakis, J., and Fusco, F. (2011), vPF_RING: Towards wire-speed network monitoring using virtual machines, In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, pp. 533–548.

[29] Dashtbozorgi, M. and Abdollahi Azgomi, M. (2009), A high-performance software solution for packet capture and transmission, In *2009 2nd IEEE International Conference on Computer Science and Information Technology*, pp. 407–411.

[30] Schneider, F., Wallerich, J., and Feldmann, A. (2007), Packet capture in 10-gigabit ethernet environments using contemporary commodity hardware, In *International Conference on Passive and Active Network Measurement*, pp. 207–217, Springer.

[31] Deri, L. (2005), nCap: wire-speed packet capture and transmission, In *Workshop on End-to-End Monitoring Techniques and Services*.

[32] Zalewski, M. (2014), P0f (online), https://lcamtuf.coredump.cx/p0f3/# (Access Date: 19 November 2020).

[33] QoSient (2019), Audit network activity (online), https://openargus.org/oldsite/index.shtml (Access Date: 11 June 2020).

[34] QoSient (2020), QoSient (online), https://www.qosient.com/ (Access Date: 11 June 2020).

[35] Argus (2020), ARGUS Sensor (online), Argus, https://openargus.org/ (Access Date: 19 November 2020).

[36] Nagios Enterprises (2020), Nagios: The Industry Standard In IT Infrastructure Monitoring (online), https://www.nagios.org/ (Access Date: 11 June 2020).

[37] Flowmon (2020), Flowmon (online), https://www.flowmon.com/en/solutions/security-operations/network-behavior-analysis-anomaly-detection (Access Date: 6 July 2020).

[38] Paessler AG (2020), Packet Capture Tool PRTG:All-in-one monitoring (online), Paessler, https://www.paessler.com/packet_capture (Access Date: 15 December 2020).

[39] Splunk (2020), Splunk Enterprise Security (online), https://www.splunk.com/en_us/software/enterprise-security.html (Access Date: 19 November 2020).

[40] Goyal, P. and Goyal, A. (2017), Comparative study of two most popular packet sniffing tools-Tcpdump and Wireshark, In *2017 9th International Conference on Computational Intelligence and Communication Networks (CICN)*, pp. 77–81, IEEE.

[41] The Tcpdump Group (2020), TCPDUMP & LiBPCAP (online), https://www.tcpdump.org/ (Access Date: 11 June 2020).

[42] Wireshark (2020), Wireshark (online), https://www.wireshark.org/ (Access Date: 11 June 2020).

[43] Zeek (2020), Zeek (online), https://zeek.org/ (Access Date: 11 June 2020).

[44] ManageEngine (2020), Flow-based Network Traffic Monitoring for in-depth traffic analysis (online), https://www.manageengine.com/products/netflow/ (Access Date: 6 July 2020).

[45] Wellem, T., Lai, Y., Huang, C., and Chung, W. (2019), A Flexible Sketch-Based Network Traffic Monitoring Infrastructure, *IEEE Access*, 7, pp. 92476–92498.

[46] Nguyen, G., Dlugolinsky, S., Tran, V., and López García, . (2020), Deep Learning for Proactive Network Monitoring and Security Protection, *IEEE Access*, 8, pp. 19696–19716.

[47] Koning, R., Buraglio, N., de Laat, C., and Grosso, P. (2018), CoreFlow: Enriching Bro security events using network traffic monitoring data, *Future Generation Computer Systems*, 79, pp. 235–242.

[48] Casino, F., Choo, K. R., and Patsakis, C. (2019), HEDGE: Efficient Traffic Classification of Encrypted and Compressed Packets, *IEEE Transactions on Information Forensics and Security*, 14(11), pp. 2916–2926.

[49] Lv, B., Yu, X., Xu, G., Yin, Q., and Shi, Z. (2018), Network Traffic Monitoring System Based on Big Data Technology, *Proceedings of the 2018 International Conference on Big Data and Computing*, pp. 27–32.

[50] OSSEC (2020), OSSEC (online), https://www.ossec.net/ (Access Date: 11 june 2020).

[51] Russinovich, M. and Garnier, T. (2020), Sysmon v12.02 (online), Microsoft, https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon (Access Date: 24 November 2020).

[52] Microsoft (2011), TCPView (online), https://docs.microsoft.com/en-us/sysinternals/downloads/tcpview (Access Date: 11 June 2020).

[53] Nirsoft (2020), CurrPorts (online), http://www.nirsoft.net/ (Access Date: 11 June 2020).

[54] Berger, S., Chen, Y., Hu, X., Pendarakis, D., Rao, J. R., Sailer, R., Schales, D. L., and Stoecklin, M. P. (2016), Closing the loop: Network and in-host monitoring tandem for comprehensive cloud security visibility, *IBM Journal of Research and Development*, 60(4), pp. 10:1–10:12.

[55] Haas, S., Sommer, R., and Fischer, M. (2020), Zeek-Osquery: Host-Network Correlation for Advanced Monitoring and Intrusion Detection, In Hölbl, M., Rannenberg, K., and Welzer, T., (Eds.), *ICT Systems Security and Privacy Protection*, pp. 248–262, Cham: Springer International Publishing.

[56] Jirsik, T. (2018), Stream4Flow: Real-time IP flow host monitoring using Apache Spark, In *NOMS 2018–2018 IEEE/IFIP Network Operations and Management Symposium*, pp. 1–2.

[57] Yuan, B., Zou, D., Jin, H., Yu, S., and Yang, L. T. (2020), HostWatcher: Protecting hosts in cloud data centers through software-defined networking, *Future Generation Computer Systems*, 105, pp. 964–972.

[58] Kortebi, A., Aouini, Z., Juren, M., and Pazdera, J. (2016), Home Networks Traffic Monitoring Case Study: Anomaly Detection, In *2016 Global Information Infrastructure and Networking Symposium (GIIS)*, pp. 1–6.

[59] Verma, M. E. and Bridges, R. A. (2018), Defining a Metric Space of Host Logs and Operational Use Cases, In *2018 IEEE International Conference on Big Data (Big Data)*, pp. 5068–5077.

[60] Sourcefire (2021), Next-Generation IPS (online), Sourcefire, https://www.ndm.net/sourcefirestore/ngips/next-generation-ips (Access Date: 2 February 2021).

[61] Suricata (2020), Suricata (online), https://suricata-ids.org/ (Access Date: 11 June 2020).

[62] Trend Micro (2021), TippingPoint Threat Protection System (online), Trend Micro, https://www.trendmicro.com/en_ca/business/products/network/intrusion-prevention/tipping-point-threat-protection-system.html (Access Date: 2 February 2021).

[63] Cisco (2021), Secure IPS (NGIPS) (online), Cisco, https://www.cisco.com/c/en_ca/products/security/ngips/index.html (Access Date: 2 February 2021).

[64] NS Focus (2021), Next Generation Intrusion Prevention System (online), NS Focus, https://nsfocusglobal.com/next-generation-intrusion-prevention-ngips/ (Access Date: 2 February 2021).

[65] Wakui, T., Kondo, T., and Teraoka, F. (2019), GAMPAL: Anomaly Detection for Internet Backbone Traffic by Flow Prediction with LSTM-RNN, In *International Conference on Machine Learning for Networking*, pp. 196–211, Springer.

[66] Duarte Jr, E. P., Hara, C., Torres Jr, P., and Gomes, C. (2020), BackStreamDB: A stream processing engine for backbone traffic monitoring with anomaly detection, *Security and Privacy*, 3(3), e106.

[67] Li, X., Zhang, X., and Wang, D. (2018), R-TAR: A Resilient Traffic Anomaly Recognition Mechanism for Backbone Network, In *2018 10th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, Vol. 01, pp. 62–66.

[68] Viegas, E., Santin, A., Bessani, A., and Neves, N. (2019), BigFlow: Real-time and reliable anomaly-based intrusion detection for high-speed networks, *Future Generation Computer Systems*, 93, pp. 473–485.

[69] Erlacher, F. and Dressler, F. (2020), On High-Speed Flow-based Intrusion Detection using Snort-compatible Signatures, *IEEE Transactions on Dependable and Secure Computing*, p. 1–1.

[70] Vectra (2020), Vectra AI (online), Vectra, https://www.vectra.ai/ (Access Date: 2 February 2020).

[71] Quadrant (2020), Sagan (online), https://quadrantsec.com/sagan_solution/ (Access Date: 24 November 2020).

[72] Extrahop (2020), ExtraHop Reveal(x) (online), https://www.extrahop.com/products/security/comparison/darktrace/ (Access Date: 6 July 2020).

[73] Darktrace (2020), Self-learning cyber AI (online), https://www.darktrace.com/en/ (Access Date: 6 July 2020).

[74] Varonics (2020), Varonics (online), https://www.varonis.com/ (Access Date: 11 June 2020).

[75] Cisco (2012), Introduction to Cisco IOS NetFlow: A Technical Overview (online), Cisco, https://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/ios-netflow/prod_white_paper0900aecd80406232.html (Access Date: 25 November 2020).

[76] Cisco (2020), Cisco Stealthwatch Enterprise (online), https://www.cisco.com/c/en_ca/products/security/stealthwatch/index.html (Access Date: 7 December 2020).

[77] Securonix (2020), Securonix Security Operations & Analytics Platform (online), https://www.securonix.com/ (Access Date: 24 November 2020).

[78] Microsoft (2015), What is Advanced Threat Analytics? (online), https://docs.microsoft.com/en-us/advanced-threat-analytics/what-is-ata (Access Date: 6 July 2020).

[79] Nguyen Thanh Van, Tran Ngoc Thinh, and Le Thanh Sach (2017), An anomaly-based network intrusion detection system using Deep learning, In *2017 International Conference on System Science and Engineering (ICSSE)*, pp. 210–214.

[80] Sohi, S. M., Ganji, F., and Seifert, J.-P. (2018), Recurrent neural networks for enhancement of signature-based network intrusion detection systems, *arXiv preprint arXiv:1807.03212*.

[81] Hwang, R., Peng, M., Huang, C., Lin, P., and Nguyen, V. (2020), An Unsupervised Deep Learning Model for Early Network Traffic Anomaly Detection, *IEEE Access*, 8, pp. 30387–30399.

[82] Chkirbene, Z., Eltanbouly, S., Bashendy, M., AlNaimi, N., and Erbad, A. (2020), Hybrid Machine Learning for Network Anomaly Intrusion Detection, *2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT)*, pp. 163–170.

[83] Jiang, K., Wang, W., Wang, A., and Wu, H. (2020), Network Intrusion Detection Combined Hybrid Sampling With Deep Hierarchical Network, *IEEE Access*, 8, pp. 32464–32476.

[84] Klein, E. (2020), Top 5 open-source HIDS systems (online), logz.io, https://logz.io/blog/open-source-hids/ (Access Date: 7 December 2020).

[85] Samson Jr, R. (2020), Top 10 Intrusion Detection And Prevention Systems (online), Clearnetwork, https://www.clearnetwork.com/top-intrusion-detection-and-prevention-systems/ (Access Date: 7 December 2020).

[86] Cooper, S. (2019), Host-Based Intrusion Detection Systems Explained – 6 Best HIDS Tools Reviewed (online), Comparitech, https://www.comparitech.com/net-admin/hids-tools-software/ (Access Date: 7 December 2020).

[87] Wilson, M. (2020), Best Host-Based Intrusion Detection Systems (HIDS) Tools & Software (online), PCWDLD.com, https://www.pcwdld.com/host-based-intrusion-detection-systems-hids-tools-and-software (Access Date: 7 December 2020).

[88] Samhain Labs (2020), Samhain: The Samhain file integrity / host-based intrusion detection system (online), https://la-samhna.de/samhain/ (Access Date: 20 November 2010).

[89] Zhang, X., Niyaz, Q., Jahan, F., and Sun, W. (2020), Early Detection of Host-based Intrusions in Linux Environment, In *2020 IEEE International Conference on Electro Information Technology (EIT)*, pp. 475–479.

[90] Deshpande, P., Sharma, S. C., Peddoju, S. K., and Junaid, S. (2018), HIDS: A host based intrusion detection system for cloud computing environment, *International Journal of System Assurance Engineering and Management*, 9(3), pp. 567–576.

[91] Pendleton, M. and Xu, S. (2017), A dataset generator for next generation system call host intrusion detection systems, In *MILCOM 2017–2017 IEEE Military Communications Conference (MILCOM)*, pp. 231–236.

[92] Besharati, E., Naderan, M., and Namjoo, E. (2019), LR-HIDS: logistic regression host-based intrusion detection system for cloud environments, *Journal of Ambient Intelligence and Humanized Computing*, 10(9), pp. 3669–3692.

[93] Can, O., Unalir, M. O., Sezer, E., Bursa, O., and Erdogdu, B. (2017), An ontology based approach for host intrusion detection systems, In *Research Conference on Metadata and Semantics Research*, pp. 80–86, Springer.

[94] Maske, S. A. and Parvat, T. J. (2016), Advanced anomaly intrusion detection technique for host based system using system call patterns, In *2016 International Conference on Inventive Computation Technologies (ICICT)*, Vol. 2, pp. 1–4.

[95] Trend Micro (2020), Deep Discovery Inspector: Detect targeted attacks and targeted ransomwareanywhere in your network (online), https://www.trendmicro.com/en_ca/business/products/network/advanced-threat-protection/inspector.html (Access Date: 22 June 2020).

[96] Palo Alto Networks (2020), Threat Prevention (online), https://www.paloaltonetworks.com/ (Access Date: 6 July 2020).

[97] Fortinet (2020), FortiDeceptor (online), https://www.fortinet.com/products/fortideceptor (Access Date: 23 June 2020).

[98] Nadler, A., Aminov, A., and Shabtai, A. (2019), Detection of malicious and low throughput data exfiltration over the DNS protocol, *Computers & Security*, 80, pp. 36–53.

[99] Bubnov, Y. (2019), DNS Data Exfiltration Detection Using Online Planning for POMDP, *European Journal of Engineering Research and Science*, 4(9), pp. 22–25.

[100] Ahmed, J., Habibi Gharakheili, H., Raza, Q., Russell, C., and Sivaraman, V. (2020), Monitoring Enterprise DNS Queries for Detecting Data Exfiltration From Internal Hosts, *IEEE Transactions on Network and Service Management*, 17(1), pp. 265–279.

[101] Ahmed, J., Gharakheili, H. H., Raza, Q., Russell, C., and Sivaraman, V. (2019), Real-Time Detection of DNS Exfiltration and Tunneling from Enterprise Networks, *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, pp. 649–653.

[102] Le, D. C., Zincir-Heywood, A. N., and Heywood, M. I. (2019), Dynamic Insider Threat Detection Based on Adaptable Genetic Programming, *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 2579–2586.

[103] Fadolalkarim, D. and Bertino, E. (2019), A-pandde: Advanced provenance-based anomaly detection of data exfiltration, *Computers & Security*, 84, pp. 276–287.

[104] CoSoSys Ltd. (2020), Endpoint Protector (online), https://www.endpointprotector.com/ (Access Date: 19 June 2020).

[105] Broadcom (2020), Symantec Data Loss Prevention (online), https://www.broadcom.com/products/cyber-security/information-protection/data-loss-prevention (Access Date: 19 june 2020).

[106] Teramind (2020), Teramind DLP (online), https://www.teramind.co/product/dlp-data-loss-prevention (Access Date: 19 June 2020).

[107] Check Point software technologies Ltd. (2020), Check Point Data Loss Prevention (online), https://www.checkpoint.com/products/data-loss-prevention/ (Access Date: 19 June 2020).

[108] Trustwave Holdings Inc. (2020), SecureTrust DLP Discover (online), https://www.securetrust.com/download/dlp-discover-data-sheet/ (Access Date: 19 June 2020).

[109] Clearswift (2020), Adaptive Data Loss Prevention (A-DLP) (online), https://www.clearswift.com/solutions/adaptive-data-loss-prevention (Access Date: 19 June 2020).

[110] McAfee (2020), Total Protection for Data Loss Prevention (DLP) (online), https://www.mcafee.com/enterprise/en-ca/products/total-protection-for-data-loss-prevention.html (Access Date: 22 June 2020).

[111] Digital Guardian (2020), Digital Guardian DLP (online), https://digitalguardian.com/ (Access Date: 22 June 2020).

[112] Alneyadi, S., Sithirasenan, E., and Muthukkumarasamy, V. (2016), A survey on data leakage prevention systems, *Journal of Network and Computer Applications*, 62, pp. 137–152.

[113] Cheng, L., Liu, F., and Yao, D. (2017), Enterprise data breach: causes, challenges, prevention, and future directions, *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 7(5), e1211.

[114] Gupta, K. and Kush, A. (2017), A Review on Data Leakage Detection for Secure, *International Journal of Engineering and Advanced Technology (IJEAT)*, 7(1), pp. 153–159.

[115] Kaur, K., Gupta, I., and Singh, A. K. (2017), Data leakage prevention: e-mail protection via gateway, In *Journal of Physics: Conference Series*, p. 012013, IOP Publishing.

[116] Costante, E., [den Hartog], J., Petković, M., Etalle, S., and Pechenizkiy, M. (2017), A white-box anomaly-based framework for database leakage detection, *Journal of Information Security and Applications*, 32, pp. 27–46.

[117] Peneti, S. and Rani, B. P. (2016), Data leakage prevention system with time stamp, In *2016 International Conference on Information Communication and Embedded Systems (ICICES)*, pp. 1–4, IEEE.

[118] Fu, X., Gao, Y., Luo, B., Du, X., and Guizani, M. (2017), Security Threats to Hadoop: Data Leakage Attacks and Investigation, *IEEE Network*, 31(2), pp. 67–71.

[119] Lu, Y., Huang, X., Li, D., and Zhang, Y. (2018), Collaborative graph-based mechanism for distributed big data leakage prevention, In *2018 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–7, IEEE.

[120] Awad, A., Kadry, S., Maddodi, G., Gill, S., and Lee, B. (2016), Data Leakage Detection Using System Call Provenance, In *2016 International Conference on Intelligent Networking and Collaborative Systems (INCoS)*, pp. 486–491.

[121] Malwarebytes (2020), Backdoor Computing Attacks (online), https://www.malwarebytes.com/backdoor/ (Access Date: 23 October 2020).

[122] Sonicwall (2020), Sonicwall Capture advanced threat protection (ATP) (online), https://www.sonicwall.com/products/firewalls/security-services/capture-advanced-threat-protection/ (Access Date: 22 June 2020).

[123] McAfee (2020), McAfee Advanced Threat Defense (online), https://www.mcafee.com/enterprise/en-ca/products/advanced-threat-defense.html (Access Date: 22 June 2020).

[124] FireEye (2020), Malware Analysis (online), https://www.fireeye.com/solutions/malware-analysis.html (Access Date: 22 July 2020).

[125] Check Point (2020), SandBlast Threat Emulation Sandboxing (online), https://www.checkpoint.com/products/threat-emulation-sandboxing/ (Access Date: 22 June 2020).

[126] Cuckoo (2019), Cuckoo Sandbox (online), https://cuckoosandbox.org/ (Access Date: 25 November 2020).

[127] Fortinet (2020), FortiSandbox: Zero-day Threat Protection (online), https://www.fortinet.com/products/sandbox/fortisandbox (Access Date: 22 June 2020).

[128] Fortinet (2020), Advanced Malware Detection (online), https://www.forcepoint.com/product/add-on/advanced-malware-detection (Access Date: 22 June 2020).

[129] Amer, E. and Zelinka, I. (2020), A dynamic Windows malware detection and prediction method based on contextual understanding of API call sequence, *Computers & Security*, 92, p. 101760.

[130] Wang, C., Ding, J., Guo, T., and Cui, B. (2017), A malware detection method based on sandbox, binary instrumentation and multidimensional feature extraction, In *International Conference on Broadband and Wireless Computing, Communication and Applications*, pp. 427–438, Springer.

[131] Hansen, S. S., Larsen, T. M. T., Stevanovic, M., and Pedersen, J. M. (2016), An approach for detection and family classification of malware based on behavioral analysis, In *2016 International Conference on Computing, Networking and Communications (ICNC)*, pp. 1–5.

[132] Wysopal, C. and Eng, C. (2020), Static Detection of Application Backdoors (online), Veracode, https://www.veracode.com/ (Access Date: 23 October 2020).

[133] Ali, M., Shiaeles, S., Bendiab, G., and Ghita, B. (2020), MALGRA: Machine learning and N-GRAM malware feature extraction and detection system, *Electronics*, 9(11), p. 1777.

[134] Joint Chiefs of Staff (2006), Joint Publication 3-13.4: Military Deception, (Technical Report JP 3-13.4) US Department of Defense.

[135] Scarfone, K., Jansen, W., and Tracy, M. (2008), Guide to general server security, *NIST Special Publication*, 800(s 123).

[136] Stewart, J. M., Chapple, M., and Gibson, D. (2012), CISSP: Certified Information Systems Security Professional Study Guide, John Wiley & Sons.

[137] Acalvio (2020), ShadowPlex autonomous deceptor (online), http://www.acalvio.com/ (Access Date: 23 June 2020).

[138] Attivo (2020), ThreatDefend Detection and Response Platform (online), https://attivonetworks.com/ (Access Date: 22 June 2020).

[139] TrapX (2020), TrapX Security DeceptionGrid (online), https://trapx.com/partners/mcafee-deception-grid/ (Access Date: 22 June 2020).

[140] Counter Craft (2020), CounterCraft Cyber Deception Platform (online), https://www.countercraft.eu/ (Access Date: 22 June 2020).

[141] Illusive Networks (2020), Stop Attack Movement from Anywhere to Anywhere (online), https://www.illusivenetworks.com/ (Access Date: 24 June 2020).

[142] Chakraborty, T., Jajodia, S., Katz, J., Picariello, A., Sperli, G., and Subrahmanian, V. S. (2019), FORGE: A Fake Online Repository Generation Engine for Cyber Deception, *IEEE Transactions on Dependable and Secure Computing*, p. 1–1.

[143] Karuna, P., Purohit, H., Jajodia, S., Ganesan, R., and Uzuner, O. (2020), Fake Document Generation for Cyber Deception by Manipulating Text Comprehensibility, *IEEE Systems Journal*, pp. 1–11.

[144] Whitham, B. (2014), Design requirements for generating deceptive content to protect document repositories, *Australian information warfare and security conference*.

[145] Skillicorn, D., Li, X., and Chen, K. (2018), Reversing the asymmetry in data exfiltration, *arXiv preprint arXiv:1809.04648*.

[146] Sharma, S. (2017), Data theft prevention using user behavior profiling and decoy documents, In *2017 International Conference On Smart Technologies For Smart Nation (SmartTechCon)*, pp. 957–961.

[147] Kippo (2016), Kippo - SSH Honeypot (online), https://github.com/desaster/kippo (Access Date: 24 November 2020).

[148] Wright, J. (2015), A Simple Elasticsearch Honeypot (online), elastichoney, https://github.com/jordan-wright/elastichoney (Access Date: 24 November 2020).

[149] Honeymail (2016), SMTP honeypot (online), https://github.com/sec51/honeymail (Access Date: 24 November 2020).

[150] Snare (2019), Snare and Tanner (online), The honeypot Project, https://www.honeynet.org/projects/active/snare-and-tanner/ (Access Date: 24 November 2020).

[151] Furfaro, A. and Lupia, F. (2020), Gathering Malware Data through High-Interaction Honeypots, In *28th Italian Symposium on Advanced Database Systems*.

[152] Chong, W. H. and Koh, C. K. (2018), Learning Cyberattack Patterns With Active Honeypots, (Technical Report) NAVAL POSTGRADUATE SCHOOL MONTEREY CA MONTEREY United States.

[153] Djanali, S., Arunanto, F., Pratomo, B. A., Studiawan, H., and Nugraha, S. G. (2014), SQL injection detection and prevention system with raspberry Pi honeypot cluster for trapping attacker, In *2014 International Symposium on Technology Management and Emerging Technologies*, pp. 163–166.

[154] Miah, M. S., Gutierrez, M., Veliz, O., Thakoor, O., and Kiekintveld, C. (2020), Concealing Cyber-Decoys using Two-Sided Feature Deception Games, In *HICSS*, pp. 1–10.

[155] Changwook-Park and Kim, Y. (2019), Deception Tree Model for Cyber Operation, In *2019 International Conference on Platform Technology and Service (PlatCon)*, pp. 1–4.

[156] Yuill, J., Denning, D., and Feer, F. (2006), Using deception to hide things from hackers: Processes, principles, and techniques, *Journal of Information Warfare*, 5(3), pp. 26–40.

[157] Al-Shaer, E., Duan, Q., and Jafarian, J. H. (2012), Random host mutation for moving target defense, In *International Conference on Security and Privacy in Communication Systems*, pp. 310–327, Springer.

[158] Rowe, N. C., Michael, J. B., Auguston, M., and Riehle, R. (2002), Software decoys for software counterintelligence, *IA Newsletter*, 5(1), pp. 10–12.

[159] Rowe, N. C. and Goh, H. C. (2007), Thwarting cyber-attack reconnaissance with inconsistency and deception, In *2007 IEEE SMC Information Assurance and Security Workshop*, pp. 151–158, IEEE.

[160] DTK (1999), Deception Toolkit (online), http://www.all.net/dtk/ (Access Date: 15 October 2020).

[161] Cohen, F. (2000), A mathematical structure of simple defensive network deceptions, *Computers & Security*, 19(6), pp. 520–520.

[162] Rowe, J., Levitt, K. N., Demir, T., and Erbacher, R. (2012), Artificial diversity as maneuvers in a control theoretic moving target defense, In *National Symposium on Moving Target Research.*

[163] Bilinski, M., Ferguson-Walter, K., Fugate, S., Gabrys, R., Mauger, J., and Souza, B. (2019), You only lie twice: A multi-round cyber deception game of questionable veracity, In *International Conference on Decision and Game Theory for Security*, pp. 65–84, Springer.

[164] Schulz, M. (2015), Detecting Minified JavaScript Code (online), https://mariusschulz.com/blog/detecting-minified-javascript-code (Access Date: 16 October 2020).

[165] McAfee (2020), Malware Packers Use Tricks to Avoid Analysis, Detection (online), https://www.mcafee.com/blogs/enterprise/malware-packers-use-tricks-avoid-analysis-detection/ (Access Date: 16 October 2020).

[166] Aebersold, S., Kryszczuk, K., Paganoni, S., Tellenbach, B., and Trowbridge, T. (2016), Detecting obfuscated javascripts using machine learning, In *ICIMP 2016 the Eleventh International Conference on Internet Monitoring and Protection, Valencia, Spain, 22-26 May 2016*, Vol. 1, pp. 11–17, Curran Associates.

[167] RDGsoft (2017), RDG Packer Detector (online), RDGsoft.net, http://www.rdgsoft.net/ (Access Date: 16 October 2020).

[168] OpenText Corp. (2020), Windows Executable Packer Detection (online), https://www.guidancesoftware.com/app/Windows-Executable-Packer-Detection (Access Date: 16 October 2020).

[169] Naval, S., Laxmi, V., Gaur, M. S., and Vinod, P. (2012), Spade: Signature based packer detection, In *Proceedings of the First International Conference on Security of Internet of Things*, pp. 96–101.

[170] Nachenberg, C. (2012), Extrusion detection of obfuscated content. US Patent 8,181,036.

[171] Xu, J., Sung, A. H., Mukkamala, S., Liu, Q., et al. (2007), Obfuscated malicious executable scanner, *Journal of Research and Practice in Information Technology*, 39(3), p. 181.

[172] Bat-Erdene, M., Kim, T., Park, H., and Lee, H. (2017), Packer detection for multi-layer executables using entropy analysis, *Entropy*, 19(3), p. 125.

[173] Zhang, H., Papadopoulos, C., and Massey, D. (2013), Detecting encrypted botnet traffic, In *2013 Proceedings IEEE INFOCOM*, pp. 3453–1358.

[174] Biondi, F., Enescu, M. A., Given-Wilson, T., Legay, A., Noureddine, L., and Verma, V. (2019), Effective, efficient, and robust packing detection and classification, *Computers & Security*, 85, pp. 436–451.

[175] Tellenbach, B., Paganoni, S., and Rennhard, M. (2016), Detecting obfuscated JavaScripts from known and unknown obfuscators using machine learning, *International Journal on Advances in Security*, 9(3/4), pp. 196–206.

[176] Li, X., Shan, Z., Liu, F., Chen, Y., and Hou, Y. (2019), A consistently-executing graph-based approach for malware packer identification, *IEEE Access*, 7, pp. 51620–51629.

[177] Anderson, B. and McGrew, D. (2016), Identifying encrypted malware traffic with contextual flow data, In *Proceedings of the 2016 ACM workshop on artificial intelligence and security*, pp. 35–46.

[178] Lin, Z., Xiao, F., Sun, Y., Ma, Y., Xing, C.-C., and Huang, J. (2018), A Secure Encryption-Based Malware Detection System., *TIIS*, 12(4), pp. 1799–1818.

[179] Vidal, J. M. and Monge, M. A. S. (2019), Adversarial Communication Networks Modeling for Intrusion Detection Strengthened against Mimicry, In *Proceedings of the 14th International Conference on Availability, Reliability and Security*, pp. 1–6.

[180] Malaver, S. (2018), Mimicry: The Evolution of Deceptive Defenses and Attacks (online), Minerva, https://blog.minerva-labs.com/mimicry-the-evolution-of-deceptive-defenses-and-attacks (Access Date: 19 October 2020).

[181] Yi, H., Kim, G., Lee, J., Ahn, S., Lee, Y., Yoon, S., and Paek, Y. (2018), Mimicry resilient program behavior modeling with lstm based branch models, *arXiv preprint arXiv:1803.09171*.

[182] Richer, P. (2003), Steganalysis: Detecting hidden information with computer forensic analysis, *SANS/GIAC Practical Assignment for GSEC Certification, SANS Institute*, Vol. 6.

[183] Kessler, G. C. (2015), An overview of steganography for the computer forensics examiner, *Forensic science communications*, 6(3), pp. 1–27.

[184] Provos, N. (2020), StegDetect: detect steganographic content in images (online), Apponic, https://stegdetect.apponic.com/ (Access Date: 23 October 2020).

[185] Khalind, O. S., Hernandez-Castro, J. C., and Aziz, B. (2013), A study on the false positive rate of Stegdetect, *Digital Investigation*, 9(3–4), pp. 235–245.

[186] Wetstonetech (2017), StegoHunt MP: Steganalysis and Steganography Detection Tool (online), https://www.wetstonetech.com/products/stegohunt-steganography-detection/ (Access Date: 23 October 2020).

[187] Andriotis, P., Oikonomou, G., and Tryfonas, T. (2013), JPEG steganography detection with Benford's Law, *Digital Investigation*, 9(3–4), pp. 246–257.

[188] Cabuk, S., Brodley, C. E., and Shields, C. (2009), IP covert channel detection, *ACM Transactions on Information and System Security (TISSEC)*, 12(4), pp. 1–29.

[189] Department of National Defence (2015), The Defence Terminology Bank (online), http://terminology.mil.ca/ (Access Date: 10 November 2020).

[190] Latham, D. C. (1986), Department of defense trusted computer system evaluation criteria, (Technical Report DoD 5200.28-STD) US Department of Defense.

[191] Termium Plus (2021), Covert channel (online), Department of National Defence, https://www.btb.termiumplus.gc.ca (Access Date: 3 February 2021).

[192] Infosec (2016), Lab: Identifying the use of Covert Channels (online), Infosec, https://resources.infosecinstitute.com/topic/lab-identifying-the-use-of-covert-channels/ (Access Date: 2 february 2021).

[193] Chen, S., Lang, B., Liu, H., Li, D., and Gao, C. (2021), DNS Covert Channel Detection Method Using the LSTM Model, *Computers & Security*, p. 102095.

[194] Han, J., Huang, C., Shi, F., and Liu, J. (2020), Covert timing channel detection method based on time interval and payload length analysis, *Computers & Security*, 97, p. 101952.

[195] Ayub, M. A., Smith, S., and Siraj, A. (2019), A Protocol Independent Approach in Network Covert Channel Detection, In *2019 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)*, pp. 165–170.

[196] Gunadi, H. and Zander, S. (2017), Bro covert channel detection (BroCCaDe) framework: scope and background, (Technical Report) Technical Report.

[197] Saeli, S., Bisio, F., Lombardo, P., and Massa, D. (2020), DNS Covert Channel Detection via Behavioral Analysis: a Machine Learning Approach, *arXiv preprint arXiv:2010.01582*.

[198] Bjerke, J. and Roy, N. (2020), Basic TOR Traffic Detection (online), Splunk Security Essentials, https://docs.splunksecurityessentials.com/content-detail/sser_tor_traffic/ (Access Date: 19 November 2020).

[199] LogPoint (2020), Detecting Tor use with LogPoint (online), https://www.logpoint.com/en/blog/detecting-tor-use-with-logpoint/ (Access Date: 19 October 2020).

[200] Netresec (2020), CapLoader (online), https://www.netresec.com/index.ashx?page=CapLoader (Access Date: 21 October 2020).

[201] Mayank, P. and Singh, A. K. (2017), Tor traffic identification, In *2017 7th International Conference on Communication Systems and Network Technologies (CSNT)*, pp. 85–91.

[202] Saputra, F. A., Nadhori, I. U., and Barry, B. F. (2016), Detecting and blocking onion router traffic using deep packet inspection, In *2016 International Electronics Symposium (IES)*, pp. 283–288, IEEE.

[203] Lashkari, A. H., Draper-Gil, G., Mamun, M. S. I., and Ghorbani, A. A. (2017), Characterization of tor traffic using time based features., In *ICISSp*, pp. 253–262.

[204] Sarkar, D., Vinod, P., and Yerima, S. Y. (2020), Detection of Tor traffic using deep learning, In *17th ACS/IEEE International Conference on Computer Systems and Applications (AICCSA 2020)*, ACS/IEEE.

[205] Cuzzocrea, A., Martinelli, F., Mercaldo, F., and Vercelli, G. (2017), Tor traffic analysis and detection via machine learning techniques, In *2017 IEEE International Conference on Big Data (Big Data)*, pp. 4474–4480.

[206] Cimpanu, C. (2019), Avast: No plans to discontinue CCleaner following second hack in two years (online), ZDNet, https://www.zdnet.com/article/avast-no-plans-to-discontinue-ccleaner-following-second-hack-in-two-years/ (Access Date: 14 Novemebr 2020).

[207] US Department of Homeland Security (2020), Mitigate SolarWinds Orion Code Compromise (online), cyber.dhs.gov, https://cyber.dhs.gov/ed/21-01/ (Access Date: 14 December 2020).

[208] FireEye (2020), FireEye Mandiant SunBurst Countermeasures (online), FireEye, https://github.com/fireeye/sunburst_countermeasures (Access Date: 23 December 2020).

[209] Symantec (2018), Intrusion Prevention System forActive Directory Networks (online), https://www.javelin-networks.com/ (Access Date: 7 july 2020).

[210] Ayub, M. A., Johnson, W. A., Talbert, D. A., and Siraj, A. (2020), Model Evasion Attack on Intrusion Detection Systems using Adversarial Machine Learning, *2020 54th Annual Conference on Information Sciences and Systems (CISS)*, pp. 1–6.

[211] Shi, Y., Zeng, H., and Nguyen, T. T. (2019), Adversarial Machine Learning for Network Security, *2019 IEEE International Symposium on Technologies for Homeland Security (HST)*, pp. 1–7.

[212] Anderson, H. S., Kharkar, A., Filar, B., Evans, D., and Roth, P. (2018), Learning to evade static PE machine learning malware models via reinforcement learning, *arXiv preprint arXiv:1801.08917*.

[213] Madani, P. and Vlajic, N. (2018), Robustness of Deep Autoencoder in Intrusion Detection under Adversarial Contamination, *Proceedings of the 5th Annual Symposium and Bootcamp on Hot Topics in the Science of Security*.

[214] Qiu, H., Zeng, Y., Zheng, Q., Zhang, T., Qiu, M., and Memmi, G. (2020), Mitigating Advanced Adversarial Attacks with More Advanced Gradient Obfuscation Techniques, *arXiv preprint arXiv:2005.13712*.

[215] Paudice, A., Muñoz-González, L., Gyorgy, A., and Lupu, E. C. (2018), Detection of adversarial training examples in poisoning attacks through anomaly detection, *arXiv preprint arXiv:1802.03041*.

[216] Shan, S., Wenger, E., Wang, B., Li, B., Zheng, H., and Zhao, B. Y. (2019), Using Honeypots to Catch Adversarial Attacks on Neural Networks, *arXiv preprint arXiv:1904.08554*.

[217] Pawlicki, M., Choraś, M., and Kozik, R. (2020), Defending network intrusion detection systems against adversarial evasion attacks, *Future Generation Computer Systems*.

[218] Usama, M., Asim, M., Latif, S., Qadir, J., and Ala-Al-Fuqaha (2019), Generative Adversarial Networks For Launching and Thwarting Adversarial Attacks on Network Intrusion Detection Systems, *2019 15th International Wireless Communications Mobile Computing Conference (IWCMC)*, pp. 78–83.

[219] Aiken, J. and Scott-Hayward, S. (2019), Investigating Adversarial Attacks against Network Intrusion Detection Systems in SDNs, *2019 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, pp. 1–7.

[220] Dickson, B. (2020), The security threat of adversarial machine learning is real (online), TechTalks, https://bdtechtalks.com/2020/10/26/adversarial-machine-learning-threat-matrix/ (Access Date: 3 March 2021).

[221] Nguyen, T. T. and Reddi, V. J. (2019), Deep reinforcement learning for cyber security, *arXiv preprint arXiv:1906.05799*.

List of symbols/abbreviations/acronyms/initialisms

AD	Active Directory
A-DLP	Adaptative Data Loss Prevention
AI	artificial intelligence
ANN	artificial neural network
API	application programming interface
APT	advanced persistent threat
ATP	Advanced Threat Protection
BiLSTM	Bi-directional long short-term memory
C2	command and control
CAF	Canadian Armed Forces
CART	classification and regression trees
CD-DAR	Cyber Defence-Decision Analysis Response
CND	computer network defence
CNN	convolutional neural network
COA	course of action
COTS	commercial off-the-shelf
CSC	covert storage channels
CTC	covert timing channels
DDoS	distributed denial of service
DLD	data leakage detection
DLP	data leakage protection
DMZ	demilitarised zone
DND	Department of National Defence
DNS	domain name service
DoD	United States (US) Department of Defence
DoS	denial of service
DRDC	Defence Research and Development Canada
DTK	Deception Toolkit
ESRMD	Entropy Signal Reflects the Malicious Document
FQDN	fully qualified domain name
GAMPAL	General-purpose Anomaly detection Mechanism using Path Aggregate without Labeled data
GAN	generative adversarial network
GP	genetic programming
GRU	gated recurrent unit
HCRF	Hidden Conditional Random Fields
HEDGE	High Entropy Distinguisher
HIDS	Host Intrusion Detection System
HTTP	hypertext transfer protocol

HTTPS	hypertext transfer protocol secure
IDS	Intrusion Detection System
IEEE	Institute of Electrical and Electronics Engineers
IIS	Internet Information Services
IP	internet protocol
IPFIX	IP Flow Information Export
IPS	intrusion prevention system
ISP	Internet Service Provider
IT	information technology
ITI	information technology infrastructure
JPEG	Joint Photographic Experts Group
kNN	k-NearestNeighbor
LSTM	long-short term memory
ML	machine-learning
MS	Microsoft
NAT	network address translation
NBAD	network behavior anomaly detection
NGIPS	Next-Generation IPS
NIDS	network intrusion detection system
NMP	Network Performance Monitor
NSM	network security monitoring
NTA	Netflow Traffic Analyzer
OR	operations research
OS	operating system
PCA	principal component analysis
pcap	packet capture
PDA	Packet Data Analysis
POMDP	Partially Observable Markov Decision Process
RNN	recurrent neural network
R-TAR	resilient traffic recognition
SDN	software defined networking
SEM	security event manager
SIEM	security information and event management
SMTP	Simple Mail Transfer Protocol
SPADE	signature based packer detection
SPAN	switched port analyzer
SSH	Secure Shell
SSL	Secure Socket Layer
TAP	test access point
TCP	Transmission Control Protocol
TLS	transport layer security

Tor	The onion router
TRL	technology readiness level
TTPs	tactics, techniques, and procedures
UDP	User Datagram Protocol
UEBA	user and entity behavior analytics
US	United States
USB	universal serial busses
VM	virtual machine
VPN	virtual private network
WEC	Windows Event Collector
WEF	Windows Event Forwarding

DOCUMENT CONTROL DATA

*Security markings for the title, abstract and keywords must be entered when the document is sensitive.

1. ORIGINATOR (The name and address of the organization preparing the document. A DRDC Centre sponsoring a contractor's report, or a tasking agency, is entered in Section 8.) DRDC – Ottawa Research Centre 3701 Carling Avenue, Ottawa ON K1A 0Z4, Canada	2a. SECURITY MARKING (Overall security marking of the document, including supplemental markings if applicable.) CAN UNCLASSIFIED 2b. CONTROLLED GOODS NON-CONTROLLED GOODS DMC A

3. TITLE (The document title and subtitle as indicated on the title page.)

Malicious activity detection: An analysis of current tools and methodologies for network defence in operational networks

4. AUTHORS (Last name, followed by initials – ranks, titles, etc. not to be used. Use semi-colon as delimiter.)

Dondo, M.; Sultana, M.; Vandenberghe, G.

5. DATE OF PUBLICATION (Month and year of publication of document.) May 2021	6a. NO. OF PAGES (Total pages, including Annexes, excluding DCD, covering and verso pages.) 65	6b. NO. OF REFS (Total cited in document.) 221

7. DOCUMENT CATEGORY (e.g., Scientific Report, Contract Report, Scientific Letter)

Reference Document

8. SPONSORING CENTRE (The name and address of the department project or laboratory sponsoring the research and development.)

DRDC – Ottawa Research Centre
3701 Carling Avenue, Ottawa ON K1A 0Z4, Canada

9a. PROJECT OR GRANT NO. (If appropriate, the applicable research and development project or grant number under which the document was written. Please specify whether project or grant.) 05ac - Cyber Decision Making and Response (CDMR)	9b. CONTRACT NO. (If appropriate, the applicable contract number under which the document was written.)
10a. DRDC PUBLICATION NUMBER DRDC-RDDC-2021-D078	10b. OTHER DOCUMENT NO(s). (Any other numbers which may be assigned to this document either by the originator or by the sponsor.)

11a. FUTURE DISTRIBUTION WITHIN CANADA (Approval for further dissemination of the document. Security classification must also be considered.)

Public release

11b. FUTURE DISTRIBUTION OUTSIDE CANADA (Approval for further dissemination of the document. Security classification must also be considered.)

Public release

12. KEYWORDS, DESCRIPTORS or IDENTIFIERS (Use semi-colon as a delimiter.)

Cyber; Cyber Attribution; Traffic Analysis

13. ABSTRACT/RÉSUMÉ (When available in the document, the French version of the abstract must be included here.)

Operational networks continue to face increasing and evolving malicious activities that threaten to disrupt ongoing missions. To ensure mission continuity and minimise disruptions due to cyber attacks, network defenders must detect and prevent such malicious activities. As computer networks have become increasingly complex, detecting malicious activities is not trivial. It requires the monitoring of the network as well as the employment of appropriate tools and methodologies to detect possible attacks. In this work, we explore such tools and methodologies. We perform an extensive literature search of the current COTS products that could be used for malicious activity detection. Our work identifies the strategic network placement of the tools to allow for the most visibility of network activities, and thereby minimising the possibility of missing harmful traffic. The COTS products are complemented by the current state-of-the-art research activities in malicious activities to ensure the awareness of the most recent technologies as well as to gain more insights of the directions the research community is moving forward to. The summary discussion of our findings leads to possible future work that we would recommend undertaking to improve the detection of malicious activities in the network.

Les réseaux opérationnels continuent d'être la cible d'activités malveillantes. Ces activités sont en croissance et en évolution et elles constituent une menace pour les missions en cours. Pour assurer la continuité des missions et minimiser les perturbations reliées aux cyberattaques, les responsables de la sécurité des réseaux doivent être en mesure de détecter et de prévenir de telles activités. Vu la complexification des réseaux informatiques, la détection n'est pas chose simple. Elle exige une certaine surveillance des réseaux et l'utilisation d'outils et de méthodologies appropriés.

Dans le cadre de nos travaux, nous avons exploré ce genre d'outils et de méthodologies. Nous avons effectué une recherche documentaire approfondie sur les produits commerciaux actuels qui pourraient être utilisés pour détecter les activités malveillantes. Nous avons déterminé le positionnement stratégique des outils dans les réseaux pour rendre les activités qui s'y déroulent le plus visibles possible et minimiser les possibilités que le trafic nuisible ne soit pas détecté. De plus, nous avons analysé des travaux de recherche de pointe sur les activités malveillantes pour mieux comprendre en quoi consistent les technologies les plus récentes et les orientations actuelles en recherche dans le domaine. Dans notre analyse des résultats, nous proposons des pistes de recherche possible pour les travaux futurs sur le sujet.