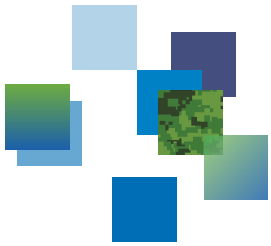
Defence Research and Development Canada
Recherche et développement pour la défense Canada

DRDC | RDDC



A technique to identify indicators for predicting web-based threat activity

Maxwell Dondo
DRDC – Ottawa Research Centre

Defence Research and Development Canada

Scientific Report
DRDC-RDDC-2016-R151
September 2016

Canada

A technique to identify indicators for predicting web-based threat activity

Maxwell Dondo
DRDC – Ottawa Research Centre

Defence Research and Development Canada
Scientific Report
DRDC-RDDC-2016-R151
September 2016

Abstract

Cyber criminals attack and compromise Internet-connected networks on an almost daily basis. One way to help defend these networks is to detect threat indicators, a minimal set of network traffic attributes or features that reflect threat activity on the network. Such indicators can provide network defenders with useful information on what is happening on the network and help them predict what could happen next. Currently, tools and techniques to determine these indicators are scarce. We, therefore, propose an approach to determine these indicators. Our approach, which is based on high-level recommendations from recent papers, uses machine learning to determine which network traffic attributes are indicative of network threat activity. Due to a high prevalence of web-based attacks, we chose to demonstrate our approach on selected web-based attacks. In our approach, we collected traffic attributes from historical network data, labelled the data, and used machine learning to determine a minimal set of traffic attributes, or indicators, that can best describe the selected web threat activity on the network. We show that a set of indicators can be used to identify specific threat activities on the network with high detection rates. We conclude by suggesting a way to extend our approach of determining indicators for web-based attacks to determining indicators for general network attacks, which is a subject of possible future work.

Significance for defence and security

This report is a deliverable for the Cyber Decision Making and Response (CDMR) project, whose objective is to study cyber threat metrics. The work focuses on determining how data and traffic patterns collected from network sensors can inform on cyber threat analysis and modeling. The expected audience for this work is the cyber S&T community and it acts to provide advice in the analysis and modeling of cyber defence projects. The work could provide input to the modeling of threat awareness in major projects such as the Automated Computer Network Defence (ARMOUR) technology demonstration (TD) project, a subject for possible future work.

Résumé

Les cybercriminels attaquent et compromettent les réseaux branchés à internet quasi quotidiennement. Une des façons d'aider à les protéger consiste à déceler les indicateurs de menaces, un nombre minimal de caractéristiques du trafic sur le réseau qui laissent entrevoir la présence de menaces. Ces indicateurs peuvent fournir aux personnes chargées de protéger les réseaux des rensegnements utiles concernant les activités sur le réseau et les aider à prévoir les prochains événements. À l'heure actuelle, les outils et les techniques pour déterminer ces indicateurs sont rares. Nous proposons par conséquent une approche pour ce faire. Cette approche, fondée sur des recommandations de haut niveau dans des publications récentes, emploie l'apprentissage automatique pour déterminer les caractéristiques du trafic sur le réseau qui sont révélatrices de la présence de menaces. En raison du grand nombre d'attaques perpétrées sur le Web, nous avons choisi de démontrer notre approche dans certains cas. Selon celle-ci, nous avons rassemblé les caractéristiques du trafic à partir des données historiques du réseau, nous les avons étiquetées et utilisé l'apprentissage automatique pour déterminer un nombre minimal de cactéristiques ou d'indicateurs du trafic pouvant le mieux décrire la présence de ces cybermenaces sur le réseau. Nous avons démontré qu'un ensemble d'indicateurs pouvait servir à déceler la présence de menaces particulières sur le réseau, et ceci avec un taux de détection très élevé. En conclusion, nous suggérons un moyen d'élargir notre approche pour déterminer les indicateurs d'attaques perpétrées sur le Web afin de définir ceux des attaques générales de réseaux qui pourront faire l'objet de travaux à venir.

Importance pour la défense et la sécurité

Ce rapport constitue un produit livrable du projet prise de décision et intervention en cybernétique (PDIC), dont l'objectif est d'étudier les paramètres des cybermenaces. Les travaux visent à déterminer comment les données recueillies par des capteurs sur les réseaux et la configuraùtion de leur circulation peuvent fournir de l'information pour l'analyse et la modélisation des cybermenaces. Ces travaux sont destinés à la communauté de cyberscience et technologie, qui s'emploie à fournir des conseils en matière d'analyse et de modélisation des projets de cyberdéfense. Ces travaux pourraient contribuer à la modélisation de la sensibilisation aux menaces dans les grands projets, comme le projet de démonstration de technologies (PDT) défense automatisée des réseaux informatiques (ARMOUR), qui pourront faire l'objet de travaux à venir.

Acknowledgements

I would like to thank Christopher McKenzie and Grant Vandenberghe for their contributions in running tests for experiments reported in this report.

This page intentionally left blank.

Table of contents

List of figures

List of tables

This page intentionally left blank.

1 Introduction

In this work, we analyse a collection of hypertext transfer protocol (HTTP) network traffic for indications of threat activity directed at vulnerable internet-connected networks. We analyse specific characteristics or features of the traffic patterns, which we call attributes, to determine which ones most effectively indicate the presence of threat activity on the network. Certain combinations of these attributes can indicate specific threat activity in the network. We call the minimal set of such attributes "threat indicators"[1]. By identifying these threat indicators, network defenders may be able to predict what is currently happening or what could happen next on their networks and assist in decision support.

Network traffic consists of many network activity patterns. These patterns mostly reflect regular network communications such as standard HTTP requests that are associated with web transactions. There may also be patterns that are associated with adversarial activities such as malicious attempts to access unauthorised data through SQL injection attacks. To protect the network and its resources, network defenders look for the malicious patterns in network activity and take appropriate courses of action.

Traditionally, network defenders use their expert knowledge to select particular network traffic attributes to monitor in order to detect the malicious patterns. This approach has focused on monitoring specific signatures, through intrusion detection systems (IDSs), to detect isolated security events. However, security researchers and practitioners have shown that such an approach could miss the bigger picture and fail to detect more complex malicious activities [1, 3–5]. These experts suggest monitoring threat indicators in order to improve network defenders' understanding of network threats and their trends, predict the potential consequences of what could happen next, and assist in decision support. The indicators can be obtained by fusing and correlating historical attack data and using the synthesized information as indicators for network threat activity, as suggested by the model of Espenschied and Gunn [1]. However, except for a very high level list of attributes, their model does not provide the specific indicators that can be used to describe network threat activities.

To determine threat indicators, some researchers have proposed the use of commercial and custom tools [1, 3, 4]. However, commercial and research activities that can determine these indicators are scarce. Commercial approaches use tools such as IDSs and intrusion prevention systems (IPSs) to detect isolated security events. Unfortunately, these events are not always correlated or analysed to have a full picture of the current network state and predict what could happen next. A correlated set

[1] Other literature, such as [2], use the term "feature" instead of attribute, and "feature indicators" for minimal set of features representing identified clusters (similar to identified attacks in our case).

of attacks may show that an adversary is pursuing a certain goal. For example, an IDS can detect insecure direct object reference (IDOR) activities on the network. Although this event may appear as a simple attempt by the attacker to access restricted resources, it may be an indication that the adversary is performing lateral movements as a precursor to data exfiltration. This exfiltration may be difficult to detect on its own since it can appear as normal traffic patterns. But it could be inferred from correlated IDOR activities. In this case, IDS attributes used to identify the IDOR security event can be used as indicators of lateral movements on the network, and remedial action could be taken to prevent possible data exfiltration. However, there may not always be IDS signatures of all forms of attacks (e.g. data exfiltration). In addition, the use of standard IDSs is inefficient and resource intensive in that large numbers of IDS signatures may be required to detect all variations of a single type of attack[2]. For example, SANS lists 70 signatures for Internet Information Services (IIS) directory traversal attack alone [6]. Monitoring a minimal, but representative, set of threat indicators can efficiently improve network defence by helping to predict what could happen next [1].

Based on the work of Espenschied and Gunn [1], we propose a technique to determine a set of indicators from the attributes of threat activity as reflected in historical network attack data. We examined a number of possible sources of the historical attack data. Some examples of these sources are traffic sensors, logs from perimeter defences such as firewalls and routers, and server activity logs. Monitoring all the attributes from these data sources as indicators of given threat activities can be overwhelming, resource intensive and inefficient, and some of the attributes may not be relevant to the type of threat. We therefore developed an algorithm to select a minimal set of indicators from these attributes. Monitoring this subset of attributes can be used to predict what could happen next.

For this work, we narrowed our focus to the analysis of web-based threat activity. We chose this type of network traffic activity since there has been significant web-based threat activity of late. For example, the 2014 Verizon Data Breach Investigations Report (DBIR) [7] found that the highest number of computer security incidents in 2013 were web-based, which made up 35% of the security incidents reported. To demonstrate the validity of our concept, we further narrowed down our focus to the detection of indicators for two types of attacks, representing examples of two different stages in a web-based attack. This narrower focus simplifies our approach to identifying a set of indicators that could be used for correlation with other data to reveal a more complex threat. Further work, which we leave as a future exercise, could determine indicators for other correlation data (malicious or not) necessary to detect a given threat such as the data exfiltration described earlier.

[2] Signature-based IDSs look for specific patterns. Even the slightest variation in an attack pattern can result in their failure to detect specific attacks.

We chose attacks that represent the "lateral movement" and "foothold" attack stages. The IDOR as a form of the web-based "lateral movement" and remote file injection (RFI) activity as an example of "foothold" attack stage. The IDOR attacks can allow an adversary to have access to unauthorised content and may also allow unauthorised lateral movements on the web server. RFI attacks allow the remote execution of malicious content on affected internal hosts. These attacks exploit applications that dynamically reference unauthorised remote scripts indicated through user input. Note that the two attacks happen to be mostly detectable by IDSs as isolated security events. We use the events as examples of attack patterns in the data in order to demonstrate our approach. Our approach can be implemented with any type of web-based threat, whether the threat is detectable by IDSs as isolated security events or not.

In our work, we took a number of steps to determine a minimal set of threat indicators for each of the two types of web-based attacks. We used web logs as the source of historical data for web-based attacks. We then extracted attributes from that data as suggested in the Genome project [1]. After considering other techniques, such as principal component analysis, to reduce the attribute set to a small but representative set of threat indicators, we chose to use the theory of information gain as implemented through a decision tree machine learner. The technique is simple to implement and has been shown to be effective in handling web-based data similar to what we use in our work [8].

We used four different datasets to conduct our experiments. The datasets came from lab simulations, cyber defence exercises, vulnerable web application activities and regular network traffic. Applying our model to these datasets, we were able to determine two sets of indicators, one set for each of the two types of attacks. Our experimental results show that we were able to determine the indicator sets which can identify the specific network threat activities with detection rates as high as 99%[3] for some datasets. Based on the success of our experiments, we make suggestions of generalising our approach to determining indicators for all forms of network attacks, which is one of the subjects of our future work recommendations.

The rest of the document is organised as follows: In Section 2, we present background activities related to our work. We follow this with a description of our model and our experimental approach in Section 3. We present our experimental results in Section 4. In Section 5, we provide recommendations and suggestions for future work. We then present conclusions to our work in Section 6.

[3] Such high detection rates are consistent with those found by other researchers using similar web-based data [8, 9].

2 Background: How existing approaches and technologies handle events

In this section we first provide background information about threat indicators for predicting network threat activity. We then present background research work and existing technologies that are related to our work. It is important to note that while the focus of our work is on identifying a set of indicators (a minimal set of attributes) that can be used to point to web-based threat activity on the network, work on attack detection is well understood and applied to many commercial off-the-shelf (COTS) products and is therefore not the focus of this work.

2.1 Using indicators to predict threat activity

The indicators that we propose in this work can reflect activities that can be linked to parts of the threat sequence as illustrated in an example in Figure 1. In the figure, we show the ten-step threat sequence[4] (linked by solid arrows) and two sets of indicators

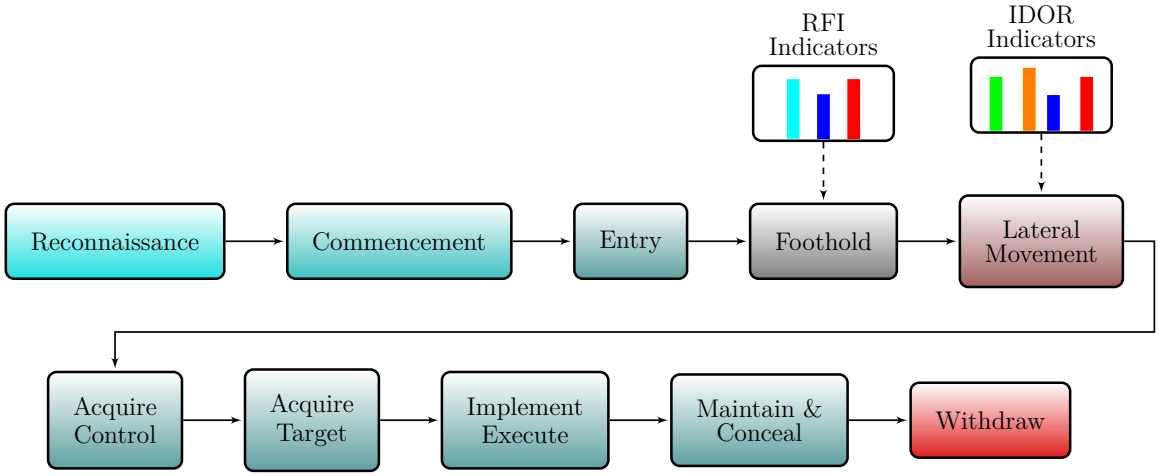


Figure 1: *The threat sequence (Adapted without permission from [1]).*

for RFI and IDOR types of attacks[5]. The indicators could be used to predict what could happen next or what the likely next target of attacks is [1, 5]. For example, the IDOR indicators[6] in Figure 1 show "lateral movement" activity. We know that such

[4] This is the ten-step extension of the "kill chain" by Espenschied and Gunn [1].
[5] We chose to place the indicators where they are shown in the "kill chain" based on the model in [1].
[6] There are other attacks or activities that could represent lateral movements, IDOR is one of them. Our indicators only detect the lateral movements as a result of IDOR attack. We therefore refer the indicators as "IDOR indicators" instead of "lateral movement indicators".

an activity is an important precursor to data exfiltration and the establishment of command and control (C2). We can therefore take the necessary courses of action to either prevent or limit the impact of such activities on the network.

Researchers suggest the use of commercial and other custom correlation tools to synthesize attack attributes and determine threat indicators [1, 3, 4]. Following this suggestion, we examined COTS products that could do this. We also looked at what the research community has offered that could be useful for this type of work. Our findings are presented in the following sections (2.2 and 2.3).

2.2 Commercial activities

There are many commercial products that are used to detect unique attacks on a computer network. Products such as Snort [10], Symantec [11], Click Security [12], and RSA web threat detection [13], use signatures to detect discrete events. Of the available commercial products, RSA has the most in common with our approach. It uses historical attack data to detect web-based attacks, much like our approach. But RSA "...uses targeted rules to detect..." unique types of web attacks [13], which approach suffers the same limitations as previously mentioned for IDSs in general, which is that if the attack changes slightly, a new rule is required. And similar to other COTS products, but unlike our approach, RSA does not provide indicators of the web threat activity on the network. Unfortunately, RSA and other similar COTS products do not provide a full situational awareness of threat activity on the network in the way that threat indicators would [1, 3, 4].

The only product we found that provides a solution that is close to our work is Trend Micro's Deep Discovery [14]. Trend Micro notes that threat activities used in targeted attacks may not be detected by standard perimeter defences or other security measures. Instead, they propose attack detection using a six-stage advanced persistent threat (APT) attack sequence [14]. However, the product does not seem to provide indicators of threat activity on the network at each stage of their APT model in order to enable defenders to predict what could happen next. As pointed out by Espenschied and Gunn [1], these indicators are an important piece in providing comprehensive network security, a piece missing in this product.

2.3 Research activities

There is a significant amount of research work in detecting network attacks. Researchers have used a variety of approaches to detect web-based attacks on the network. The trend of these research activities, such as those published in [9,15–17], is to use historical attack data to model web-based attack detection. However, those approaches give a "binary" answer to the detection of isolated web-based attacks, which is a weakness, as

described earlier [1,3,4,14]. In contrast, our approach, which we demonstrate through the use of two unique attacks, shows how indicators could be used to reveal more about the presence of web-based threat activity on the network, activity that could lead to other forms of attacks such as data exfiltration.

One approach that is close to ours is the recent work by Nguyen *et al.* [18]. In that work, the authors used different machine-learning algorithms to determine an optimal set of features to detect web-based attacks. Similar to our approach, the authors labelled historical data and trained their algorithms to detect attacks using a reduced feature set. However, their final result only focussed on providing the correct detection of unique web-based attacks and analysing the effects of feature reduction. The work does not provide a final feature set or show what their feature reduction means to monitoring threat activity on the network. In contrast, our approach determines a reduced feature set, and then uses it as indicators for that particular threat activity, instead of a unique detection. The indicators can then be used to relate to ongoing network threat activities, thus enabling network defenders to predict what could happen next and therefore provide appropriate courses of action.

In the next section, we present how we modeled our approach to achieve this objective.

3 System model and experimental approach

This section describes how we determined web-based threat indicators using historical attack data and how we implemented our system model. The system model for our approach is illustrated in the block diagram in Figure 2.

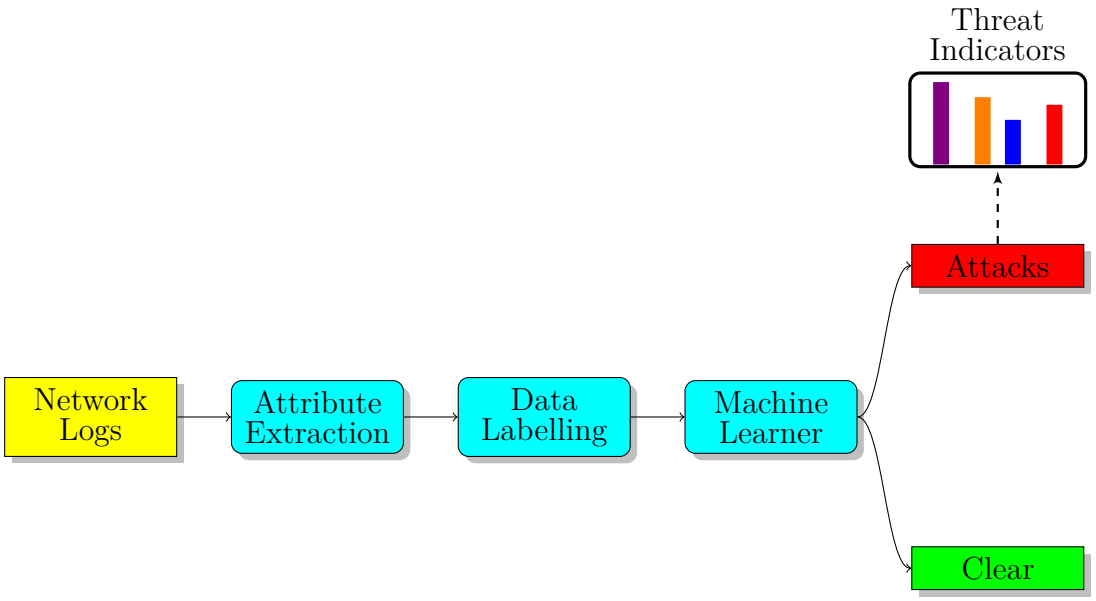


Figure 2: *System model.*

As shown in Figure 2, data, in the form of network logs, is presented to an attribute extraction module. This module extracts attributes from the logs data and passes it on to the data labelling module where data is labelled according to its attributes and attack classes. In the machine learning module, a learner is trained on the labelled data and learns to classify it as either attack or normal traffic data. From the identified attack classes the machine learner provides the set of threat indicators related to each attack type. The modules in the system model are described in detail in the sections that follow.

3.1 Network logs

As mentioned earlier, our work focuses on web-based threat activity, and demonstrated using two attack examples, IDOR and RFI. IDOR attacks occur when web applications expose information to an attacker, which can allow attackers to exfiltrate important data. Examples of this type of attack are open redirects and directory traversal. Attackers often use open redirects in phishing attacks. They take advantage of poorly implemented web application parameters to redirect users to a malicious site where

they can become infected by downloading malicious content. In directory traversal types of attacks, the attacker directly accesses resources that they are not supposed to access.

RFI attacks are a form of malicious phishing and spear phishing activities. In an RFI attack, an attacker injects a malicious uniform resource identifier (URI) into what looks like a safe link. A vulnerable web server then fetches the malicious content on behalf of the client and sends it to the client, who can then become infected. Once the client is infected, the attacker can establish C2 and perform other malicious activities.

All web activity is logged into access and error logs by the web server using the Apache common log format. The resulting logs make it possible for network defenders to study and understand actual and attempted threat activity on the network, which is the subject of this work. The "Network Logs" block in Figure 2 represents this logs data. Analysing such log activities as proposed in our work, could provide an understanding of threat activities, determining what the attacker was attempting to do, and predicting what the attacker could do next.

An example of an access log entry is shown in Listing 1. Table 1 summarises the

Listing 1: *A typical log entry used for this work.*

```
1
2 192.168.3.1 - - [17/Jun/2014:16:35:01 -0400] "GET /?include=http
    ://192.3.44.12/hacked.txt HTTP/1.1" 404 291 "http://www.
    example.com/index.html" "Mozilla/5.0 (X11; Linux x86_64; rv
    :22.0) Gecko/20100101 Firefox/22.0 Iceweasel/22.0"
3
```

Table 1: *The meanings of the log entries in Listing 1 [19].*

Log field	Log entry
Source internet protocol (IP) Address	192.168.3.1
Timestamp	[17/Jun/2014:16:35:01 -0400]
HTTP method	GET
HTTP version	HTTP/1.1
Requested resource	/?include=http://192.3.44.12/hacked.txt
Satatus code	404
Size of object returned	291
Referer	http://www.example.com/index.html
Client browser information	Mozilla/5.0 (X11; Linux x86_64; rv:22.0) Gecko/20100101 Firefox/22.0 Iceweasel/22.0

meaning of this log entry. The remote host is identified by the source IP address. A timestamp represents the time the request was received. Every request has a method, which is "GET" in this case, and an HTTP version (1.1 in this case). The requested resource represents the web content the client is looking for. A status code, which represents the success or failure of the request, is represented by 404. Based on this request, the client receives an object of size 291. A "referer" is the site that referred the client to the site, and the client browser is the application that the client claims to be using to make the request[7].

The error log takes the following form:

```
[Wed Oct 10 11:32:54 2001] [error] [client 192.168.4.1] client
denied by server configuration: /input/docs/organisation/orgchart
```

The entry is made up of the timestamp, the error severity (`error`), the IP address of the client that generated the error, and the error message. Such an error log goes hand in hand with the access log. This particular error log shows an attempt to access a resource `/input/docs/organisation/orgchart` that is not available. This information is valuable in that it gives an indication what information the adversary is looking for.

3.2 Attribute extraction and data labelling

Based on web logs, we determined a list of attributes that we could use to identify the attacks that we are focusing on in this work. The list in Table 2 represents a complete set of the attributes available (based on our log sources) that can be used as indicators of the web-based threat activity that we are focusing on in this work.

As shown later in our experiments, some of these attributes turned out to be either irrelevant as indicators or weak indicators of the web threat activity we were focussing on. Such attributes could be used to inform on other activities by the threat actor. But, this is beyond the scope of the current work and is a possible topic for future work.

The next step in our approach is to label the log data using the identified attributes. We used two types of data. One set of data was generated in the lab, and the other was obtained from real network activity. We labelled the first set of data when it was generated in the lab. We labelled all attack patterns as attack ("Attack") and regular traffic as ("Clear") as we generated them. We used the other two datasets for testing our approach.

[7] This browser information should not be trusted since a malicious client can use the field to mislead.

Table 2: *Summary of web-based attributes drawn from activity log entries.*

Attribute	Description
Source IP address	The IP address of the source and whether it appears in advisory or intelligence listings
Request method	The HTTP request method, such as GET, POST
HTTP version	Version of HTTP used
Request length	Length of HTTP request
Length of URI	Length of URI
Requested resource type	Type of resource being requested, such as asp, cgi
Traversal codes in request	Any directory traversal codes in request; this is either plain or encoded as hexadecimal characters
Direct commands	Attempts to access sensitive files or execute shell commands
Vulnerable directory access	Attempts to access a known, or previously known, vulnerable directory such as WebDav
Incorrect directory access	Attempts to access unavailable or unauthorised resource; number of such failures
Response code	Response code returned to the resource requester
Response size	Size of the response sent to user
Injected URLs	Injected URLs and the malicious forms it can take
Same request frequency	Number of requests by the same source
Same URI request frequency	Number of requests for the same URI by the same source
Inter-request interval	Time interval between requests
Interval for blocked requests	Time interval between blocked requests

3.3 Machine learner: Supervised machine learning

To determine our set of indicators, we chose to use machine learning. Machine learning techniques classify data in both supervised and unsupervised cases. In supervised machine learning, training examples include labels that show what the training objective is. In unsupervised learning, learners classify data without class labels. In our work, we use supervised learning because we use labelled training data.

For our learner, we chose the decision tree, which has been found to produce highly accurate classifications with web-based data [8, 18]. The decision tree is not only highly accurate, but, of the possible models we considered, it is also the easiest learner from which to extract the final set of indicators after training.

Decision tree classifiers assign classes to data patterns depending on the data's attributes. The tree is made up of decision nodes and leaf nodes. At each decision node, the classifier splits the data according to the value of an attribute that is selected to give the best data partition into its classes. The results of the split are placed on the leaf nodes. Each leaf node, in turn, becomes a decision node and the splitting test is applied to that node as well. The process is repeated until the patterns are grouped into their classes or until a predetermined stopping criteria is reached.

The splitting criterion is based on one of many available decision tree algorithms. In this work, we focus on the C4.5 algorithm, a popular decision tree training algorithm [20]. This algorithm has been successfully used to classify data similar to ours [8]. After considering other training algorithms, we determined that this algorithm produces the best classification results for our type of data. Detailed discussion of decision trees is beyond the scope of our work, interested readers can refer to other literature such as [20, 21].

The C4.5 algorithm uses information gain to decide how to classify data. For example, consider the measure of effectiveness for using an attribute A to classify data. The algorithm determines the information gain, or reduction in entropy, caused by splitting the data using attribute A [20]. The information gain I is given by:

$$I(S, A) = E(S) - \sum_{i=1}^{k} \frac{|S^i|}{|S|} E(S^i) \tag{1}$$

where $E(S) = \sum_{i=1}^{k} -p_i \log_2(p_i)$ is the entropy of a data collection, S, that can take k classes, p_i is the fraction of S belonging to class i; S^i is the subset of S that has attribute A belonging to class i of k, and $|\cdot|$ is the cardinality operator.

The gain ratio of splitting the data into k classes over an attribute A is given by

$$\text{Split}(A) = -\sum_{i=1}^{k} \frac{|S^i|}{|S|} \times \log_2 \left(\frac{|S^i|}{|S|} \right) \tag{2}$$

Each attribute is tested at the decision node. The attribute that results in the highest gain ratio is selected and used to split the data. Using the remaining attributes, the algorithm repeats the process until complete classification, and the learner is therefore, fully trained. The fully trained decision tree can be recalled with patterns that were not necessarily used during training.

3.4 Experimental methodology

In this section, we present how we conducted our experiment. We describe the datasets that we used and how we implemented machine learning in our work.

3.4.1 Data sources

We experimented with four datasets: a lab-generated Lab dataset, a 2011 cyber defense exercise (CDX) dataset, a lab-generated Damn Vulnerable Web Application (DVWA) dataset and a dataset from real network activity (the Network dataset). Summary descriptions and sizes of these datasets are shown in Table 3.

Table 3: *Datasets used in our experiments. The size represents the total number of events in the dataset.*

Dataset	Description	Size	Attacks	
			IDOR	RFI
Lab	Data generated in the lab using existing IP randomisation techniques	7 112	4 138	356
DVWA	Data generated in the lab by attacking a vulnerable web server, the DVWA [22] server	3 073	624	0
CDX[a]	Data generated during the 2011 cyber defence exercises [23]	5 837	1 239	0
Network[a]	Real network data. Data includes blue teaming attack activities	37 286	3 560	944

[a] Attacks manually labelled to aid the discussion of learner performances.

We generated the Lab dataset using virtual machines (VMs), while considering two classes of attacks: IDOR and RFI. We collected activity log files that were generated when we targeted a web server with normal and attack web transactions from a client host. In a separate but similar data-collection experiment, we generated the DVWA dataset by repeatedly carrying out normal and attack web transactions on a vulnerable DVWA web server. We used the DVWA and Open Web Application Security Project (OWASP) samples of attack transactions to attack the web server.

The normal but arbitrary web transactions added necessary noise to the datasets (Lab and DVWA). The CDX dataset was collected during the 2011 CDX exercises [23]. The Network dataset was obtained from real network transactions on an operational network. It included blue-teaming activities, which we found valuable in testing our model.

For testing purposes, we used a script to aid us in manually counting the number of attacks in the CDX and Network datasets. We carried out individual experiments for each class of attacks, starting with the insecure direct object reference (IDOR).

3.4.2 Training and testing

In this section we discuss how we trained and tested the machine learners. The discussion also includes how we dealt with a common source of machine learning errors, over-fitting. We also present the machine learner attributes and how we measured training and testing performances.

3.4.2.1 Training learners with n-fold cross validation

In our work, we used a standard machine learning practice called *stratified n-fold cross validation* to train, evaluate, and validate the accuracy of our machine learners [18, 24–26]. In stratified n-fold cross validation, the dataset presented to the machine learner is divided equally into n sets. Data patterns are randomly assigned to each set. The assignments ensure that each set contains approximately an equal number of class patterns; in our case, we labelled patterns as either "Attack" or "Clear". For example, if 10% of the data patterns in a dataset are attack patterns, then about 10% of the patterns in each of the n sets would be attack patterns.

The first $n-1$ sets of data patterns are used for training. The remaining set is used to validate and evaluate the classifier's performance. The process is repeated n times ensuring that each set is used as a test set. The learner's performance is the average of the n performance values obtained during the full training cycle [27]. In our experiments, we used $n = 10$, which is standard practice in machine learning for datasets of our sizes [24, 26].

3.4.2.2 Handling *over-fitting* in machine learning

In machine learning, a learner can fail to generalise the data but perfectly fit the training data. This results in classification errors when the learner is presented with previously unseen data. This situation occurs when either the number of training examples is too small to produce a representative sample of the expected output or when there is too much noise in the data. This is a common problem in machine learning that experts handle in different ways.

The C4.5 decision tree algorithm we used avoids over-fitting through a number of techniques. It allows for stopping tree growth earlier to ensure that the tree does not necessarily perform a perfect classification and, therefore, over-fit the data [21]. The algorithm also allows for post-pruning over-fit trees. Mitchel [21] argues that post-pruning, which removes insignificant classification attributes, has been found to be very successful in reducing over-fitting. Mitchel also points out that the use of training and validation datasets reduces over-fitting. The most effective way of achieving this training and validation steps is the n-fold cross validation [18, 24–26], which we used in our work.

3.4.3 Machine learning attribute variables

In this section, we define the attribute variables for the machine learner. Table 4 shows a matrix that maps attributes from Table 2 with the machine learner variables derived from these attributes. We define machine learner variables as being of type integer, text, or boolean. In Table 4, integer variables are used for response codes, response sizes, request lengths, and request frequencies. Each pattern has a text label of either "Attack" or "Clean" as mentioned in Section 3.2.

Table 4: *The attribute matrix to map attributes to machine learner variables.*

Attribute	ML variable	Attribute	ML variable
Source IP address	IP subnet (first 2 octets)	Direct commands	Shell commands
Source IP address	Intelligence report	Vulnerable directory access	High reference attempt
Source IP address	Advisory reports	Vulnerable directory access	Failed access attempt
Request method	HTTP method	Vulnerable incorrect directory access	Failed access attempt
HTTP version	HTTP version	Response code	Response code
Request length	Request length	Response size	Response size
Length of URI	High path length	Injected URLs	IP injection
Requested resource type	Shell commands	Same request frequency	Number of requests from source
Requested resource type	Sensitive files	Same URI request frequency	Number of requests for same URL
Requested resource type	php and CGI vulnerability	Requested resource type	Include code injection
Traversal codes in request	Plain directory jump	Traversal codes in request	Unicode directory jump

Similar to other researchers in this field, we simplify our solution approach by rep-

resenting the presence or absence of certain strings in the log entries as boolean variables [28]. For example, "Shell Commands" would be represented by a boolean value indicating the presence of a shell command (such as `C:/.../deltree`) in the log entry. During attribute extraction, we used a script to check for the presence of these strings against a list of other shell commands[8].

A similar technique was used for all boolean variables. The rest of the boolean variables represent the presence of the source IP address in intelligence or advisory reports, HTTP method used (True for "GET" and False for all other[9]), HTTP version (True for version 1.1 and False for version 1.0), presence of malicious files such as "Shell Commands" or injection commands, and presence directory jump commands. We simulated the intelligence reports and advisory reports by using valid IP address lists from the Internet Storm Center's advisories [29]. We searched for the presence of our log IP addresses on these lists and assigned the variable a boolean value of either True or False.

The machine learner variables are discrete characteristics of the attributes in Table 2. They are, therefore, fine-grained attributes of the traffic patterns characterised by the attributes in Table 2. For example, the "Source IP address" attribute of Table 2 is now replaced by the more detailed machine learner attributes, "IP subnet", "Intelligence report" and "Advisory report". In the rest of the document, we refer to the machine learner variables in Table 4 as the traffic pattern attributes. In our work, we present data patterns, representing the full list of 22 input machine learner attributes, to the machine learner for training and testing.

3.4.4 Calculating performance measures

To judge the performance of our machine learning approach, we use detection measures that are commonly used in machine learning. Machine learning approaches measure performance by the number of correct decisions the learner makes. The performance is determined by comparing the predicted outcomes with the actual, or true, outcomes through what is commonly referred to as the *confusion matrix*. The confusion matrix, which is illustrated in Table 5, shows the definitions of the predicted and actual, or true, outcomes.

In the table, TP, TN, FP, and FN respectively represent the numbers of true positives, true negatives, false positives, and false negatives. Using these detection measures, the performance is calculated as the fraction of the correctly predicted classifications. We used the following detection measures.

[8] We assumed that this list is exhaustive. However, in a real application, an additional shell script could test the validity of each shell command.

[9] The web transactions in our datasets mostly used the "GET" method in their attack data. An extension of our approach to cover more general data should include other request methods.

Table 5: *The confusion matrix.*

	True Clear	True Attack
Predicted Clear	TN	FN
Predicted Attack	FP	TP

- Precision: Represents a measure of correct predictions. It is a fraction of actual attacks (TP) out of the total number of examples the machine learner identified as attacks (TP + FP). It is defined as:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{3}$$

- Recall (or true positive rate): Represents the fraction of real attack examples that the mechine learner identified over all the attack examples given. This performance measure is defined as follows:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{4}$$

- The false positive and false negative rates are respectively defined as follows:

$$\text{False Positive Rate} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{5}$$

$$\text{False Negative Rate} = \frac{\text{FN}}{\text{TP} + \text{FN}} \tag{6}$$

Ideally, we want machine learners that deliver high precision and high recall.

4 Experiments and discussion of results

This section presents the results and analyses from two experiments. The first experiment focuses on finding indicators for IDOR type of attacks. The second experiment focuses on determining indicators for RFI type of attacks. We follow the presentation of the two experiments with examples of how the threat indicators can be used in operational networks.

4.1 Experiment 1: Determining indicators for IDOR attacks

In this experiment, we used the 10-fold cross validation method to train a learner to determine indicators for IDOR attacks using one of the two labelled datasets (DVWA and Lab). First, we train a learner on one dataset and name the learner after the dataset it was trained on. For example, the DVWA learner was trained using the DVWA dataset. Then we test that learner on all the other datasets (Lab, DVWA, CDX, and Network). We repeat the process on the second dataset to create and test the second learner. Then we analyse the results from testing both learners on our datasets.

Our training results are summarised in Table 6. With a training precision rate of 99% for both learners, the minimal number of attributes that each learner needed in order to classify the data are as listed in the table. For example, the Lab learner achieved its classification with 6 attributes. This minimal set of attributes is an example of the set of indicators that we were looking for.

Table 6: *Training results for individual datasets.*

Learner	Number of attributes
Lab	6
DVWA	3

Although the results were produced with high precision rates in both learners, each learner came to a different conclusion as to which set of indicators best described the detected type of attack. In Table 7, we list the indicators (a subset of the attributes listed in Table 4) identified by each learner. The check marks in Table 7 represent the indicators identified by each individual learner. For example, the DVWA learner achieved classification with the attributes "High Path Length", "Request Length", and "Sensitive Files", as indicated by the check marks. The Lab learner, on the other hand needed more attributes for its learning.

The indicators listed in Table 7 vary by learner, which means that the indicators also vary by the dataset used to train the learner. However, we expected our approach

Table 7: *Training results showing the attributes identified by each learner.*

Attributes	Learner	
	DVWA	Lab
High Path Length	✓	✓
Request Length	✓	
Requests for Same URL		✓
Plain Directory Jump		✓
Sensitive Files	✓	✓
Shell Commands		✓
Response Size		✓

to determine one consistent set of indicators that could detect the attack type in any dataset. We therefore carried out additional tests on the learners to determine a minimal set of indicators common to all learners. In the additional tests, we presented each learner with data that was not used during the learner's own training, which was data from the other datasets as well as each learner's data that was not used in training. The results from the learners' performances are displayed in the graph in Figure 3.

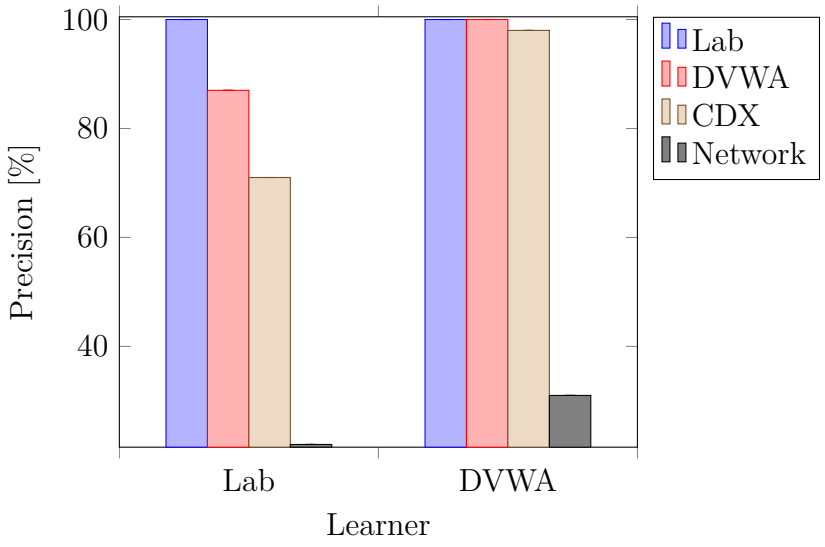
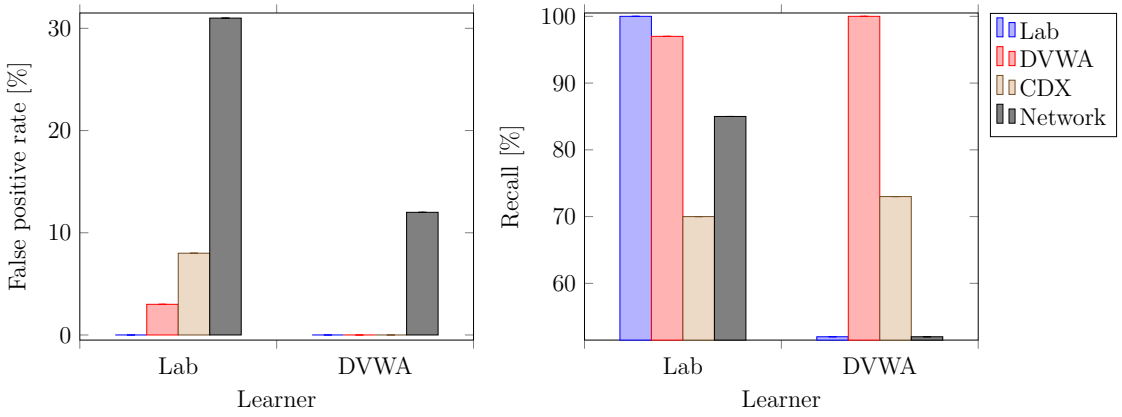


Figure 3: *Detection rates (precision) for each learner when tested with each dataset.*

In the graph, the horizontal axis represents the learners. The vertical axis represents the precision rates for each learner when tested with each dataset. The legend represents the individual datasets that were presented to each learner. For example,

when the Lab learner (trained from the lab-generated Lab dataset) was presented with the DVWA dataset, the second entry on the legend, the learner classified this dataset with a precision of about 87%. The rest of the graph represents the performances of each learner when tested with each one of the four datasets.

The performances of the learners (Lab, DVWA) shown in Figure 3 are not consistent over all datasets. As expected, each of the learners recorded high performances with data they were trained on. The DVWA learner also recorded high precision rates for two other datasets, but very low performances with the Network dataset. In fact, the learners recorded very low performance rates with the Network dataset. The implications of these results can be made more clear by looking at the false positive and recall rates for the learners.



(a) False positives.

(b) Recall (true positives).

Figure 4: *Machine learner performances.*

Figure 4 shows graphs for the recall and false positive rates for each learner. As with Figure 3, the horizontal axes represent the learners, while the legend represents the individual datasets that were tested for each learner. From Figure 4(a), the highest false positive rates resulted from testing the Network dataset on either learner. The rest of the graphs show false positive rates below 10%, which is generally a good rate. However, the recall rates (true positives) show significant inconsistencies across learners and datasets. Only the DVWA dataset had a recall rate above 80% for both learners. The CDX dataset also had a recall rate above 70% with both learners. The lowest recall rate was recorded for the Lab dataset on the DVWA learner.

The high precision and recall rates as well as the low false positive rates for each learner when tested on its own dataset mean one of two possibilities. The first possibility is that the well-known machine learning problem of over-fitting resulted in the high detection rates above. However, we took steps to avoid this problem through early

stopping, allowing for pruning and the use of 10-fold cross validation. These steps, which are well known in machine learning as measures to avoid over-fitting [24, 30], suggest that our learners did not experience this problem.

The second possibility points to the datasets themselves. Because we used 10-fold cross validation, the high performance rates for each learner show that the learning process was effective. Each dataset was, with cross validation, able to validate its own learning with its own dataset. However, when presented with completely new data patterns, both learners (Lab and DVWA) produced many errors in classification, as reflected in the false positive rates and some low recall rates. One plausible reason for this behaviour is that these learners were not presented with enough classification examples to be able to generalise a broad range of patterns. This might explain why the learners were not able to classify some data patterns that closely matched what they had previously seen during their own training. The performance of the learners against the Network dataset supports this hypothesis. This dataset came from blue teaming activities on the network. Our learners were not trained on enough examples to detect all these blue teaming activities, which reinforces the conclusion that training with a broad dataset produces consistent results.

The detection failure experienced by the learners is a known problem with supervised machine learning, as observed by other researchers [8]. Supervised machine learners perform poorly when presented with patterns that do not match training examples. This is the problem that the learners experienced. It is therefore necessary to train the machine learners with a dataset that contains a broad set of examples. Such a training approach reduces the chances of a machine learner handling a case for which it has not been trained. As a result of such training, the machine learner rarely misclassifies.

Given that the lab learner consistently produced better results than the DVWA learner, we adopt its results over the Lab learner's. We also adopt its six attributes as the indicators for IDOR threat activities.

4.2 Experiment 2: Determining indicators for RFI attacks

We conducted our second experiment in the same manner as the previous one, by using a 10-fold cross validation training on a labelled dataset. In this case the labelled DVWA dataset did not have any RFI attack patterns. Therefore, we only used the Lab learner for training and testing. We also tested the Lab learner against all the other datasets.

With a precision rate of over 99%, the learner used three attributes to classify RFI activities in the data. The identified attributes are: "php and CGI vulnerability", "Include code injection ", and "IP injection". Similar to the previous experiment, we carried out additional tests on the learner using data from other datasets. With the

Lab and Network data, the recall and precision rates were almost 100%. There were no false positives recorded by the learner on any of the datasets.

Although this experiment did not record any false positives, there were false negatives as shown in Table 8. The only test that had a non-zero false negative was with the learner's own dataset, the Lab dataset. It had a false negative rate of 1.78%.

Table 8: *False negative rates.*

Learner	Test dataset			
	Lab	Network	CDX	DVWA
Lab	1.78%	0%	0%	0%

False negative results are generally expected in machine learning. Low false negative rates of the order of 1.78%, are in the expected range. The CDX and DVWA datasets did not have RFI attack patterns, so this result is not surprising. However, the zero false positive and negative rates on the Network data could be an indication of over-fitting. Since we took steps to avoid over-fitting, the results suggests that the three attributes identified by this learner are effective indicators for RFI threat activity. We therefore adopt the three attributes from the Lab learner as our indicators for RFI threat activities.

4.3 Examples of monitoring threat activity using selected indicators

In this section, we present examples of how the indicators that we identified in this work could be used to monitor threat activity in an operational network. We first consider indicators for IDOR attacks, and then look at indicators for RFI attacks.

Our model identified six indicators for identifying IDOR type of attacks. These indicators are: "High Path Length", "Requests for Same URL", "Plain Directory Jump", "Sensitive Files", "Shell Commands", and "Response Size". As an example of how these indicators can be used in an operational network setting, we present a graphical representation of the relative numbers of indicator signals in our datasets, as shown in Figure 5. We used different datasets to represent different data snapshots for different time periods in an operational network.

In the figure, the horizontal axis represents the datasets used. The legend represents the indicators, while the vertical axis represents the number of indicator signals as a percentage of the total number of IDOR attacks in the dataset[10].

[10] Note that the total percentage values can exceed 100% in a given dataset because multiple indicators can be present in a single attack.

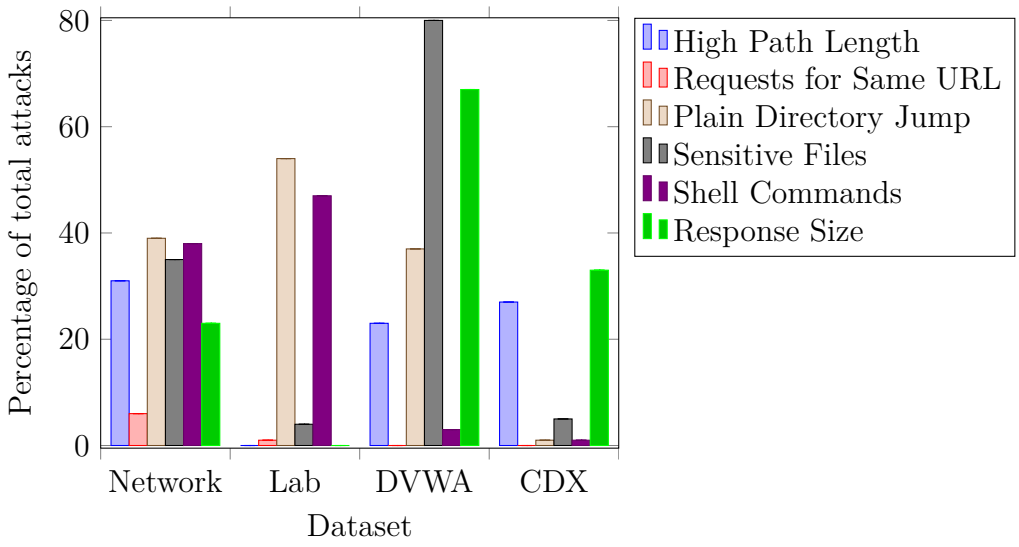


Figure 5: *Distribution of IDOR threat indicators in the datasets.*

Figure 5 shows that the indicator-mix in the datasets varied widely. For example, from the Lab dataset, the highest indicator present was the "Plain Directory Jump". This was followed by the "Shell Commands" indicator. The DVWA dataset showed a high prevalence of sensitive file activity, while indicator activity on the Network dataset is almost uniform, covering all of the six indicators. These indicators could provide valuable information on adversary activities on the network and might even reveal the attackers' *modus operandi*. For example, the presence of the "High Path Length" indicator in the CDX dataset can imply the activities of a sophisticated attackers that might either be using long path names to confuse perimeter defence systems or include sophisticated path redirects in their web requests. Correlating such knowledge with other information such as the source IP addresses can provide useful information about that attacker and how they operate. As it turns out, the activities in the CDX dataset are from cyber exercises where very sophisticated attackers took part.

Figure 6, shows the variations of the three indicators for RFI attacks in two datasets. It is formatted in the same way as Figure 5 and shows the distribution of the "php and CGI vulnerability" (phpCgiVuln), "Include code injection" (includeInject), and "IP injection" (IPInject) indicators in the three datasets used.

As explained earlier, there were three RFI indicators for the Lab learner. The visual representation of the relative numbers of each activity indicator, as shown in Figure 6, can provide network defenders with useful information about adversary activity on the network. For example, the high prevalence of the IP injection indicators in the Network dataset may give the defender a clue as to what the source of malicious activity is, and help to predict what could happen next.

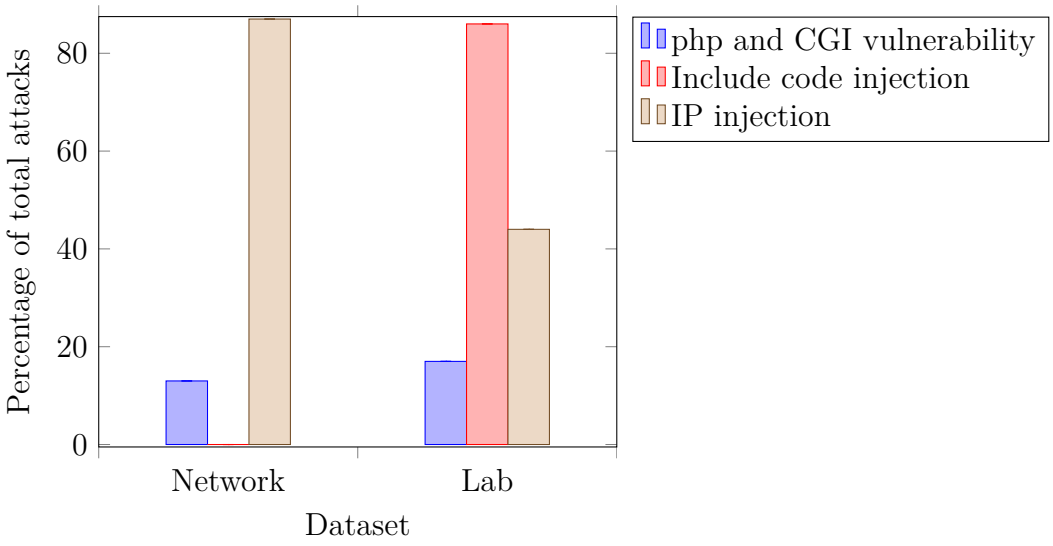


Figure 6: *Distribution of RFI threat indicators in the datasets.*

To demonstrate a possible live network monitoring scenario, we show, in Figure 7, a detailed visualisation of the threat indicator activity on the Lab dataset. We use this Lab dataset to represent a time snapshot of traffic from an operational network. The figure shows, on the vertical axis, the number of indicator signals detected in the data. The horizontal axis represents the three indicators identified for the RFI attacks. The graph shows that most of the malicious activity for this dataset originated from code injection, which can be useful information for the network defender in determining what to look for or predicting what could happen next.

4.4 Summary on experiments

Overall, our experimental results show that, from a set of network activity attributes, we could determine a minimal set of indicators that reflects certain web-based threat activity on the network. Our machine learning approach enabled us to identify minimal indicator sets on most datasets. Our argument in this work is that if such indicators are monitored on the network, adversarial activity related to an attack can be detected. Such information could be useful in predicting what could happen next [1, 3, 4]. Suitable courses of action could then be taken to improve network security.

As an example of the usefulness of such monitoring, consider an analyst monitoring the network. Assume the analyst sees a prevalence of the following activity on the network: “Plain Directory Jump”, “Unicode Directory Jump”, “Sensitive Files”, “Shell Commands”, and “High Reference Attempt”. The analyst can then infer web-based lateral movements, a known precursor to data exfiltration activities, and take defensive

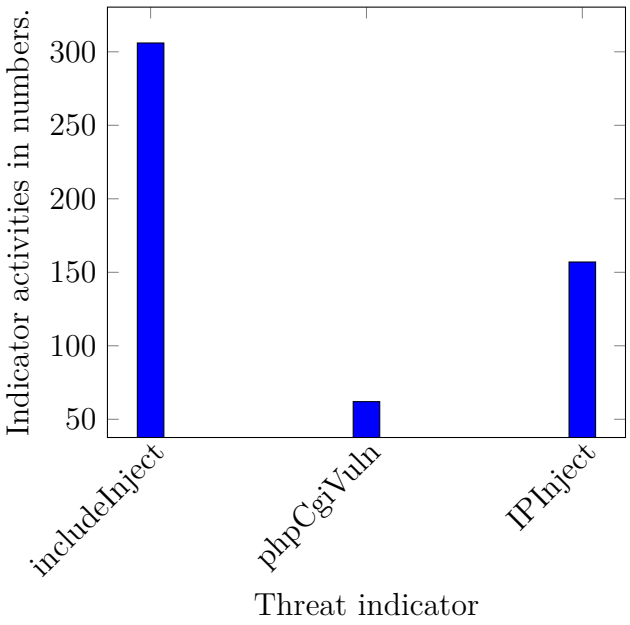


Figure 7: *RFI threat indicators for the lab dataset.*

courses of action (COA) such as shutting down external file transfer services in order to prevent any possible data exfiltration that could follow. From the relative numbers of these indicators, and analyst can also infer, with the help of other data, the tools techniques and practices (TTPs) of the adversary. The IDOR indicator set from the CDX data is a prime example. As mentioned earlier, the prevalence of "Unicode Directory Jump" indicators in that data can reflect on the activities of an adversary of high attacker complexity as characterised by Mateski *et al.* [5]. The indicator information can be correlated with other data sources to provide a more comprehensive understanding of the threat and implement appropriate COAs.

Although this work is a small and simplified example, it can be broadened to include indicators from other attacks or attack stages. Such information, as well as other network data, can be correlated to give the analyst an indication of how certain adversaries conduct their attacks. Our model can enhance or be complementary to existing IDS systems by incorporating a trained machine learner such as the IDOR learner into an IDS. Such an enhanced IDS could be used to provide network defenders with broad information about threat activities on the network.

5 Recommendations and future direction

The previous section presented the results of two experiments to demonstrate how to determine indicators for certain web-based threat activity. In this section, we present some general recommendations for the design and implementation of this model, based on the results of our experiments. We conclude the section with a presentation of possible future work activities.

5.1 General recommendations on the system model

In our work, we were able to determine two sets of indicators for two web-based threat activities using a supervised machine learning approach. The resultant indicators can can be used to identify threat activity on the network. We make some recommendations to our system model shown in Figure 2 in order to generalise it for indicators of other web-based attacks.

Our first and most important recommendation to generalise this approach would be to use machine learning models that can be trained to detect other known forms of web-based threat activity. One approach to achieve this would be to train individual machine learners for each known type of web attack. This approach is illustrated in Figure 8.

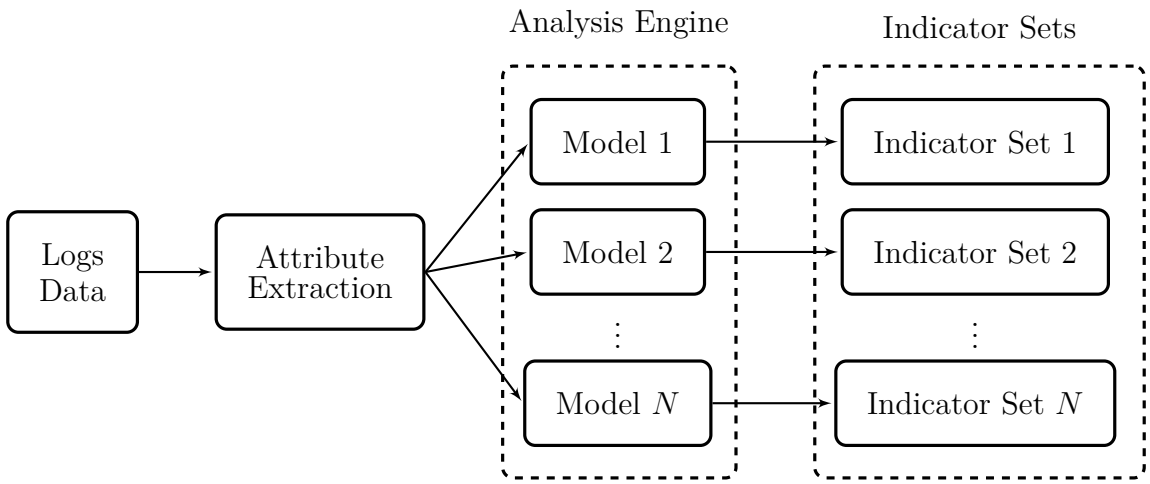


Figure 8: *A system model generalisation for determining N web-based threat indicator sets.*

Similar to the work by Corona and Giacinto [9], the generalised model in Figure 8 consists of arrays of small models, each handling a specific type of web-based threat activity. Examples of these activities include IDOR, RFI and data exfiltration attacks. In Figure 8, if we have N known types of web attacks, each of the N models would

use the set of extracted attributes from logs data. The model would then use machine learning to determine a minimal set of indicators for detecting this threat activity in network data. However, there is a huge number of web-based attacks, and some of them are evolving daily. So, having individual learners for individual types of attacks can be cumbersome and may be difficult to implement in practice, which could be a topic for future research.

In our work, we used a supervised machine learning technique on labelled data. We used lab-labelling and, like other researchers performing similar work, our expert knowledge about web-based attacks to label the datasets. Even if we assume total accuracy in our labelling, there is always the possibility of new or zero-day types of attacks that may not be reflected in training datasets at any given time. As shown in our work, this may result in a poor set of indicators. Our second recommendation, therefore, is that data should be labelled often or whenever a new form of attack is known. That ensures that the machine learner is trained on the most current attack types.

5.2 Future work

The objective of our work was to determine a set of indicators that reflect web-based threat activity on the network. Our model demonstrated that objective using two web-based attacks and achieved high detection rates. Ultimately, we aim to exploit this technology to the benefit of current and future Department of National Defence (DND) projects. To advance this technology to that stage, we suggest the following research and development activities:

- **Using unsupervised machine learning:** Instead of labelling attack data for supervised machine learning, future work could investigate the use of unsupervised approaches. In such an approach, the machine learner decides the classes of data by considering all the attributes. It will be up to an expert analyst to determine which class of data belongs to which type of attack. Although this kind of approach requires extra human intervention and is prone to high false positive rates, it eliminates the need for subjective labelling and could be able to detect indicators for zero-day attacks.
- **Incorporating intelligence and advisory data:** Our model was based on an idealised dataset that did not have significant input for intelligence and advisory data. Although such attributes were rejected by our model as being unimportant, some experts in the field believe such information should be considered to further analyse identified threats [31]. We therefore suggest that future work could look at incorporating these attributes when analysing identified threat activities. Alternatively, future work could find ways of incorporating these attributes into a trained machine learner.

- **Informing on major projects such as ARMOUR TD project [32]:** The indicators that we determined in this work can be an important input to major defence projects such as ARMOUR. Currently ARMOUR does not have an understanding of threat. It does not collect information relevant to threat activity understanding. We intend to extend our work to pull threat indicators into ARMOUR in order to provide the project with awareness of threat activities on the network.

- **Sharing information:** Currently, there is significant interest within our coalition partners to share cyber security threat information. An example of such interest is NATO's Cyber Defence Data Collaboration and Exchange Infrastructure (CDXI) project [33]. The indicators we determined in this work, which are useful in describing web-based threat activities, could be shared through such projects. Future work could investigate ways of bundling the indicator information into data objects that can be shared with such partners.

- **Training continuously:** The nature of attacks keeps evolving. New attacks come and some evolve over the years. A supervised learner trained on an old set of attacks can easily become obsolete if it is not updated regularly. Future research work could help determine how often this training needs to be performed and the amount of incremental data needed to keep the learner current. Such retraining could be implemented as part of regular patch scheduling.

- **Gathering exploitation requirements:** The threat indicators that we have determined in this work are useful when they are used in conjunction with continuous monitoring. When used that way, they can alert on threat activities detected in network traffic. For this to happen, the DND needs, a detailed design of this approach and a description of how it could be seamlessly integrated into existing sensor or monitoring networks to provide near real-time analysis of network traffic.

- **Generalising the threat indicator model:** We have suggested a generalised model for web-based threat activities. Although web threats are the biggest source of threat activity on the network, other potentially more risky activities also need to be modeled. Future work could follow up on this work by looking at other ways of generalising this problem to cover all network activity.

6 Conclusion

In this work, we used a machine learning technique based on information gain to determine a minimal set of indicators for predicting web-based threat activity on the network. We used web-based attack data from datasets obtained from lab simulations, cyber defence exercises, and regular operational network traffic. Applying our approach to these datasets for two types of web-based attacks, we were able to determine two significantly reduced sets of indicators, one set for each type of attack. In both cases, we achieved detection rates as high as 99% when using those indicators. The high detection rates are encouraging. They strongly suggests that if network defence and monitoring tools use our approach, they can be able to detect threat activity with high precision. However, our work also showed that the training data needs to be broad enough to cover all current types of attacks; otherwise, the set of indicators determined would lead to significant misclassifications. This suggests the need for learner retraining with the most up-to-date datasets.

To advance and exploit this work, we recommend the generalisation of the problem to include all forms of threats. Such a generalised model could allow the ability to share threat information through collaborative projects, such as NATO's Cyber Defence Data Collaboration and Exchange Infrastructure (CDXI) project. The model could also inform on network threat activity in major projects, such as the Automated Computer Network Defence (ARMOUR). To advance and fully exploit our findings, future work could explore the numerous suggestions we identified in this report.

References

[1] Espenschied, J. and Gunn, A. (2012), Threat Genomics, *Metricon 7.0. Bellevue, WA*.

[2] Kim, B., Rudin, C., and Shah, J. A. (2014), The Bayesian Case Model: A Generative Approach for Case-Based Reasoning and Prototype Classification, In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K., (Eds.), *NIPS 2014: Advances in Neural Information Processing Systems*, pp. 1952–1960, Curran Associates, Inc.

[3] Hutchins, E. M., Cloppert, M. J., and Amin, R. M. (2011), Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains, *Leading Issues in Information Warfare & Security Research*, 1, 80.

[4] Croom, C. (2011), The cyber kill-chain: a foundation for a new cyber-security strategy, *High Frontier*, 6, 52–6.

[5] Mateski, M., Trevino, C. M., Veitch, C. K., Michalski, J., Harris, J. M., Maruoka, S., and Frye, J. (2012), Cyber threat metrics, (Technical Report SAND2012-2427) Sandia National Laboratories.

[6] SANS, Intrusion Detection FAQ: What are unicode vulnerabilities on Internet Information Server (IIS)? http://www.sans.org/security-resources/idfaq/iis_unicode.php. (Access Date : 16 July 2015).

[7] Verizon (2014), 2014 Data Breach Investigations Report, *Verizon*.

[8] Salem, B. and Karim, T. (2008), Classification features for detecting server-side and client-side Web attacks, In Jajodia, S., Samarati, P., and Cimato, S., (Eds.), *Proceedings of The IFIP Tc 11 23rd International Information Security Conference*, Vol. 278 of *IFIP:The International Federation for Information Processing*, pp. 729–733, Springer US.

[9] Corona, I. and Giacinto, G. (2010), Detection of server-side web attacks., *Journal of Machine Learning Research-Proceedings Track*, 11, 160–166.

[10] Snort (2014), Snort. https://www.snort.org/. (Access Date : 15 October 2014).

[11] Symantec (2014), Symantec. http://www.symantec.com. (Access Date : 15 October 2014).

[12] Click Security (2014), Click Security. https://www.clicksecurity.com/. (Access Date : 15 October 2014).

[13] RSA (2013), RSA threat detection. http://www.emc.com/domains/rsa/index.htm. (Access Date : 15 October 2014).

[14] Trend Micro (2013), Countering the advanced persistent threat challenge with Deep Discovery. http://www.trendmicro.ca/cloud-content/us/pdfs/business/white-papers/wp_deepdiscovery.pdf. Access Date (17 November 2014).

[15] Tan, Z., Jamdagni, A., He, X., Nanda, P., Liu, R., Jia, W., and Yeh, W.-c. (2010), A two-tier system for web attack detection using linear discriminant method, In Soriano, M., Qing, S., and López, J., (Eds.), *Information and Communications Security*, Vol. 6476 of *Lecture Notes in Computer Science*, pp. 459–471, Springer Berlin Heidelberg.

[16] Ulmer, C., Gokhale, M., Gallagher, B., Top, P., and Eliassi-Rad, T. (2011), Massively parallel acceleration of a document-similarity classifier to detect web attacks, *Journal of Parallel and Distributed Computing*, 71(2), 225 – 235.

[17] Chwalinski, P., Belavkin, R., and Cheng, X. (2013), Detection of HTTP-GET attack with clustering and information theoretic measurements, In Garcia-Alfaro, J., Cuppens, F., Cuppens-Boulahia, N., Miri, A., and Tawbi, N., (Eds.), *Foundations and Practice of Security*, Vol. 7743 of *Lecture Notes in Computer Science*, pp. 45–61, Springer Berlin Heidelberg.

[18] Nguyen, H., Torrano-Gimenez, C., Alvarez, G., Petrović, S., and Franke, K. (2011), Application of the generic feature selection measure in detection of web attacks, In Herrero, A. and Corchado, E., (Eds.), *Computational Intelligence in Security for Information Systems*, Vol. 6694 of *Lecture Notes in Computer Science*, pp. 25–32, Springer Berlin Heidelberg.

[19] The Apache Software Foundation (2015), Log files. http://httpd.apache.org/docs/2.4/logs.html. (Access Date : 16 July 2015).

[20] Criminisi, A., Shotton, J., and Konukoglu, E. (2012), Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning, *Found. Trends. Comput. Graph. Vis.*, 7, 81–227.

[21] Mitchell, T. (1997), Machine learning, McGraw Hill.

[22] RandomStorm (2014), Damn Vulnerable Web Application. http://www.dvwa.co.uk/. (Access Date : 15 October 2014).

[23] CDX (2011), CDX Data Sets. http://www.westpoint.edu/crc/SitePages/DataSets.aspx. (Access Date : 15 October 2014).

[24] Kohavi, R. et al. (1995), A study of cross-validation and bootstrap for accuracy estimation and model selection, In *IJCAI*, Vol. 14, pp. 1137–1145.

[25] Ruggieri, S. (2002), Efficient C4.5, *IEEE Transactions on Knowledge and Data Engineering*, 14(2), 438–444.

[26] Arlot, S., Celisse, A., et al. (2010), A survey of cross-validation procedures for model selection, *Statistics surveys*, 4, 40–79.

[27] Anguita, D., Ghio, A., Ridella, S., and Sterpi, D. (2009), K-fold cross validation for error rate estimate in support vector machines., In *DMIN*, pp. 291–297.

[28] Macskassy, S. A., Hirsh, H., Banerjee, A., and Dayanik, A. A. (2003), Converting numerical classification into text classification, *Artificial Intelligence*, 143, 51–77.

[29] DShield (2014), Internet Storm Center. https://www.dshield.org/. (Access Date : 6 November 2014).

[30] Cawley, G. C. and Talbot, N. L. (2010), On over-fitting in model selection and subsequent selection bias in performance evaluation, *The Journal of Machine Learning Research*, 11, 2079–2107.

[31] Hartley, M. (2014), Strengthening the cyber kill chain with cyber threat intelligence. http://www.isightpartners.com. (Access Date : 15 October 2014).

[32] Sawilla, R. and Wiemer, D. (2011), Automated computer network defence technology demonstration project (ARMOUR TDP): Concept of operations, architecture, and integration framework, In *HST 2011: IEEE International Conference on Technologies for Homeland Security*, pp. 167–172.

[33] Dandurand, L., Bouillon, E., and Lagadec, P. (2011), High-level requirements for a cyber defence data collaboration and exchange infrastructure, *NATO*, Vol. 3175.

This page intentionally left blank.

Acronyms and abbreviations

APT	advanced persistent threat
ARMOUR	Automated Computer Network Defence
C2	command and control
CDMR	Cyber Decision Making and Response
CDX	cyber defense exercise
CDXI	Cyber Defence Data Collaboration and Exchange Infrastructure
COTS	commercial off-the-shelf
COA	courses of action
DBIR	Data Breach Investigations Report
DND	Department of National Defence
DVWA	Damn Vulnerable Web Application
HTTP	hypertext transfer protocol
IDS	intrusion detection system
IDOR	insecure direct object reference
IIS	Internet Information Services
IPS	intrusion prevention system
IP	internet protocol
OWASP	Open Web Application Security Project
PDIC	prise de décision et intervention en cybernétique
PDT	projet de démonstration de technologies
RFI	remote file injection
TD	technology demonstration
TTP	tools techniques and practice
URI	uniform resource identifier
VM	virtual machine

This page intentionally left blank.

DOCUMENT CONTROL DATA

(Security markings for the title, abstract and indexing annotation must be entered when the document is Classified or Protected.)

1. ORIGINATOR (The name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Centre sponsoring a contractor's report, or tasking agency, are 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.)

   UNCLASSIFIED

2b. CONTROLLED GOODS

   (NON-CONTROLLED GOODS)
   DMC A
   REVIEW: GCEC DECEMBER 2012

3. TITLE (The complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S, C or U) in parentheses after the title.)

   A technique to identify indicators for predicting web-based threat activity

4. AUTHORS (Last name, followed by initials – ranks, titles, etc. not to be used.)

   Dondo, M.

5. DATE OF PUBLICATION (Month and year of publication of document.)	6a. NO. OF PAGES (Total containing information. Include Annexes, Appendices, etc.)	6b. NO. OF REFS (Total cited in document.)
September 2016	46	33

7. DESCRIPTIVE NOTES (The category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.)

   Scientific Report

8. SPONSORING ACTIVITY (The name of the department project office or laboratory sponsoring the research and development – include address.)

   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.)	9b. CONTRACT NO. (If appropriate, the applicable number under which the document was written.)
05ac	

10a. ORIGINATOR'S DOCUMENT NUMBER (The official document number by which the document is identified by the originating activity. This number must be unique to this document.)	10b. OTHER DOCUMENT NO(s). (Any other numbers which may be assigned this document either by the originator or by the sponsor.)
DRDC-RDDC-2016-R151	

11. DOCUMENT AVAILABILITY (Any limitations on further dissemination of the document, other than those imposed by security classification.)

   Unlimited

12. DOCUMENT ANNOUNCEMENT (Any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in (11)) is possible, a wider announcement audience may be selected.)

   Unlimited

13. ABSTRACT (A brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual.)

Cyber criminals attack and compromise Internet-connected networks on an almost daily basis. One way to help defend these networks is to detect threat indicators, a minimal set of network traffic attributes or features that reflect threat activity on the network. Such indicators can provide network defenders with useful information on what is happening on the network and help them predict what could happen next. Currently, tools and techniques to determine these indicators are scarce. We, therefore, propose an approach to determine these indicators. Our approach, which is based on high-level recommendations from recent papers, uses machine learning to determine which network traffic attributes are indicative of network threat activity. Due to a high prevalence of web-based attacks, we chose to demonstrate our approach on selected web-based attacks. In our approach, we collected traffic attributes from historical network data, labelled the data, and used machine learning to determine a minimal set of traffic attributes, or indicators, that can best describe the selected web threat activity on the network. We show that a set of indicators can be used to identify specific threat activities on the network with high detection rates. We conclude by suggesting a way to extend our approach of determining indicators for web-based attacks to determining indicators for general network attacks, which is a subject of possible future work.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (Technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus. e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

cyber threat; web threat; web security; machine learning

DRDC | RDDC

SCIENCE, TECHNOLOGY AND KNOWLEDGE FOR CANADA'S DEFENCE AND SECURITY | SCIENCE, TECHNOLOGIE ET SAVOIR POUR LA DÉFENSE ET LA SÉCURITÉ DU CANADA

www.drdc-rddc.gc.ca