Application of distributed computing and machine learning technologies to cybersecurity

Hamza Attak[1], Marc Combalia[2], Georgios Gardikis[3], Bernat Gastón[2], Ludovic Jacquin[1], Dimitris Katsianis[4], Antonis Litke[5], Nikolaos Papadakis[5], Dimitris Papadopoulos[5], Antonio Pastor[6], Marc Roig[2], Olga Segou[7]
(All authors contributed equally and are presented in alphabetical order.)

[1]Hewlett Packard Enterprise, [2] i2CAT, [3] Space Hellas S.A., [4]inCITES Consulting SARL, [5] Infili Technologies P.C., [6] Telefonica I+D, [7] Orion Innovations P.C.

Abstract. SHIELD is a distributed cyber-security system that leverages Network Function Virtualisation for dynamically deploying virtual Network Security Functions. The security functions send network traffic's monitoring data to a big-data store. The Data Analysis and Remediation Engine executes security analytics modules on top of monitoring data modules in order to detect threats. The security analytics heavily leverage Machine Learning algorithms for detecting anomalies and classifying threats. This paper presents the different Machine Learning algorithms and details the obtained results and the direction taken by the project with regards to its implementation, including business capabilities for the cybersecurity solution.

Keywords: Machine Learning, Cybersecurity, Data Analytics, NFV, SDN, Business Model

1 Introduction to SHIELD: Machine-Learning-based Security

SHIELD is an information-driven cybersecurity platform, based on Software Defined Network (SDN), Network Function Virtualisation (NFV), big Data Analytics (DA) and infrastructure's attestation. SHIELD's goal is the design and development of a novel cybersecurity framework, focusing on offering Security-as-a-Service in an evolved telco environment. All stakeholders (ISPs, companies, end users, cybersecurity agencies and security vendors) are considered in the project SHIELD's architecture, which consists of the following key components: (i) the virtual Network Security Functions (vNSFs) implement the traffic processing functionalities; (ii) the vNSF Orchestrator manages the vNSFs lifecycle; (iii) the Network Infrastructure supports the execution and management of the vNSFs; (iv) the vNSF Store makes the vNSFs available to the vNSFO; (v) the Trust Monitor verifies the infrastructure and its services' integrity; (vi) the Data Analytics and Remediation Engine (DARE) acts as an information-driven intrusion detection and prevention platform; and (vii) the Security Dashboard provides a graphical front-end of the platform that allows the operators to apply mitigation actions. The overall – simplified – SHIELD's workflow for detecting a threat is: (1) vNSFs spread over the network, collect network traffic's information and send it to the DARE's data store; (2) the DARE's security analytics' engines continuously analyse the data available in the data store; and (3) upon detection of a potential threat, the DARE notifies the operator through the dashboard.

This paper focuses on the DARE, which couple Artificial Intelligence (AI) with security. It features cognitive and analytical components that are able to predict vulnerabilities and attacks by leveraging Big Data analytics and Machine Learning techniques. The DARE is one of the central pillars of the SHIELD platform, featuring cognitive and analytical components capable of predicting specific vulnerabilities and attacks by leveraging Big Data analytics and machine learning techniques.

It consists of three main components:

(i) The Data Acquisition module, which is responsible for the ingestion of the selected datasets (in flow, DNS, proxy formats) and their preparation for further processing.

(ii) The Data Analytics Engine, which leverages two different analytics modules, producing packet and flow analytics by using scalable machine-learning techniques. It involves the Cognitive DA module based on open-source, state-of-the-art distributed computing tools, that allow for batch and stream processing of large amounts of data and the Security DA module, a closed-source system based on a combination of analytics techniques that can efficiently process a vast amount of network data online to discover and classify cybersecurity threats. All proposed algorithms will be implemented in the open-source Cognitive DA module.

(iii) The Remediation Engine uses the analysis from the data analytics modules and is fed with alerts and contextual information in order to determine a mitigation plan for the existing threats.

The DARE infrastructure deploys a collection of state-of-the-art technologies (e.g. Hadoop, Spark, Kafka) in the form of an integrated ecosystem. It consists of a cluster of nodes running Linux with virtualisation support. Each node performs several operations and is orchestrated by cluster and resource management technologies specifically designed for big data applications.

2 Machine Learning in Cybersecurity

ML techniques can currently counter Cyber-attacks [1] in two different ways: first, with anomaly detection, where unsupervised algorithms are trained to learn the trusted behaviour in order to detect any unusual ones; second, with threat classification, where supervised learning is used to classify known attacks. The former aims to detect 0-day attacks while the latter is effective for known attacks. Existing cybersecurity solutions combine both techniques: first, they apply anomaly detection, looking for malicious flows, and then use threat classification to identify the type of attack. Current unsupervised anomaly detection algorithms can be sorted in three categories [2]:

- **Reconstruction**: is based on data compression and posterior reconstruction, behaviours reconstructed with low error rate are considered “normal” while high error rate ones are considered anomalous.

- **Boundary**: focuses on finding boundaries around the normal behaviour, every behaviour outside the defined range is considered anomalous.
- **Density Estimation**: estimates the probability density function of the training data, it discriminates anomalous behaviours using a threshold value.

The most used supervised classification algorithms in cybersecurity are Support Vector Machines (SVM), Decision trees and Naïve Bayes classifiers. More specifically, [3] has shown that SVMs can be successful in the task of traffic classification, and [4] shows that tree-based methods exhibit very high accuracy measures, while also reducing the need of feature pre-processing. Algorithms based on neural networks are also being considered, but their use seems to be mostly restricted to data of high dimensionality.

3 Machine Learning in SHIELD

3.1 Streaming Ingestion

In SHIELD, vNSFs act – amongst other functions - as probes. vNSFs collect data and feed it into the DARE for further analysis. This data cover NetFlow, Web proxy services and DNS logs for example. In order to collect data, the DARE's software agents are integrated in each vNSFs. They acquire raw data and transform it to a format suitable for over-the-network transmission. The data is sent to the DARE's streaming service, which splits the incoming streams into smaller specific topics and smaller partitions, while also creating a data pipeline for each topic. It is then stored - in a structured manner - in a Hive database, operating on top of the Hadoop Distributed File System (HDFS). The ML algorithms run as Spark jobs over the Hive data.

3.2 Anomaly Detection

Latent Dirichlet Allocation (LDA)

LDA is a generative probabilistic topic modelling algorithm, originally used for collections of discrete data such as text corpora [5]. It defines a three-level hierarchical Bayesian model, in which each item of a collection is modelled as a finite mixture over an underlying set of topics. SHIELD applies LDA to network traffic, by converting the log entries to words through aggregation and discretization. This way, documents correspond to IP addresses, words to log entries (matching an IP address) and topics to normal network traffic profiles. LDA's goal is to deduce a probabilistic model for each IP's behaviour and to assign a probability of occurrence to each log entry, meaning that the less probable an event is, the more suspicious it is considered.

The DARE Cognitive DA leverages Apache Spot [6] as a built-in module for the LDA implementation. This implementation performs complex feature extraction techniques (word creation from network logs) to infer the disparities in network traffic. Due to its unique requirements regarding data pre-processing, and its peculiarities in the training procedure, LDA cannot be directly compared to the rest of the anomaly detection algorithms proposed in this research.

One Class Support Vector Machines

One class SVM [7] is an unsupervised ML algorithm using novelty detection. It is based on SVM, but simplifies the problem by classifying one data point to a class. The nature of SVMs allows to create non-linear decision boundaries using non-linear functions in hyperplanes through Kernel functions. Two important aspects are implied by this algorithm: first, being unsupervised, it does not require any type of labelling for the training phase; second, the novelty detection implies that the traffic must be clean in the training phase. SHIELD identified these two as critical to work on zero-days attacks, where no training dataset or labelling process is possible.

The DARE Cognitive DA integrates One Class SVM by upgrading the Spot ML framework. The implementation extracts and normalizes relevant data from Netflow. Then it focuses on the relevant features of the network traffic, such as protocol family, ports, IPs, flow duration, bytes and packets per flow or number of similar flows.

Isolation Forest (IForest)

IForest [8] provides anomaly detection through the observation's concept of "isolate", after applying random forest of decision trees. Anomaly observations are easy to isolate as they translate into shorter paths in the trees. IForest is suitable for several numbers of datasets and shows acceptable memory usage [9] [10]. This makes for a very promising anomaly detection technique in cybersecurity based on big network flows. The training process is achievable with normal and anomalous traffic in the same dataset, which makes it directly usable in practice.

The DARE Cognitive DA integrates iForest by upgrading the Spot ML framework.

Local Outlier Factor (LOF)

LOF [11] is an anomaly detection algorithm based on outlier detection, where data points are seized using local density deviation with respect to their k-nearest neighbours. The outliers are the points with substantially lower density. LOF has shown good results with cybersecurity attacks [12], but it requires high computational resources.

The DARE Cognitive DA leverages Apache Spot for the LOF implementation.

Deep Learning (DL)

Due to their ability to extract complex features from raw data, DL techniques make sense in the field of anomaly detection. Autoencoders are the most used architecture for anomaly detection. It relies on input signal rebuilding after putting it through a compressive path. Autoencoders are mainly composed by an encoder and decoder. The encoder's task is to compress the input data into a low dimension vector, while the decoder uses this vector as input and tries to reconstruct the data with minimal loss. After training, the Autoencoder has adjusted its parameters to optimally reconstruct data similar to the one it was trained with. Anomalies will present a high reconstruction error rate after being forwarded through this architecture. Then, the reconstruction error can be used to label a certain data point as anomalous.

Recurrent neural networks (RNN) have also been successfully used in the task of unsupervised anomaly detection. In these architectures, the prediction error can be used as an anomaly measurement. RNNs permit to find temporal structures present in the data which could be indicative of network threats. In [13], authors use Long Short-Term Memory RNNs, to predict the next flow in the network, and then use the prediction error to label this flow as anomalous if needed.

For SHIELD, we have created an Autoencoder composed of three layers. The first one containing 16 neurons corresponding to the following variables of the netflow traffic:

- Protocol - has been encoded using on-shot technique producing a vector of 5 variables.
- Flags - have been encoded using one-shot technique producing a vector of 6 variables.
- Duration of the flow
- Origin and destination ports
- Number of packets and number of bytes per flow

The inner layer is composed by 12 neurons and the outer layer contains 16 neurons again. Although the tested neural network is very simple (only one inner layer) the objective at this stage is to determine the variables that are more relevant to attack detection using neural networks. The use of more complex construction produce results more difficult to analyse. A future objective is to test more complex neural networks and fine-tune them to obtain better results

3.3 Threat Classification

Random Forest

Random forest is an ensemble supervised method used for classification. It constructs a multitude of decision trees at training time and outputs the mode of the classes of the individual trees as the final class [14]. Then it applies the general technique of bootstrap aggregation (or bagging) to tree learners, leading to a better performance model by decreasing the variance, without increasing the bias [15]. Random forest is considered one of the best-performing ML algorithms [16], mainly because of its ability to remove decision trees' habit of overfitting the training set [17] and of its unmatched classification accuracy compared to current algorithms. In the case of network traffic classification, the datasets are usually unbalanced since the majority class (normal traffic) is usually orders of magnitude higher than the minority classes (attack flows). Therefore, classifiers are overwhelmed by the dominating class and tend to ignore the flows related to malicious activity. Random forest is of no exception, thus techniques like cost-sensitive learning and oversampling of the minority class are leveraged to tackle this issue. For our implementation, the Spark MLlib framework [18] was used to design a Random Forest model of 50 trees, which was trained and evaluated using the datasets described later. A parameterization grid was set to select the optimal values for the maximum tree depth (length of the longest path from a root to a leaf) and feature subset size (number of features to consider for splits at each node).

Multi-Layer Perceptron (MLP)

MLP is a class of feedforward artificial neural networks, consisting of at least three layers of nodes (input, hidden, and output layers). In MLP, each neuron unit calculates the linear combination of its real-valued inputs and passes it through a threshold activation function. Learning occurs iteratively, by changing connection weights after each piece of data is processed, based on the amount of outputted errors compared to the expected result (backpropagation). The use of non-linear activation functions in the neural nodes can be implemented to reproduce a nonlinear function mapping, allowing to solve non-linearly separable problems, such as network anomaly classification [19] [20]. Using a carefully chosen set of features of the Netflow protocol as input signal, we were able to train and compare several MLP architectures to classify multiple normal and anomalous states, using the Deep Learning Studio platform [21] which leverages the open-source Keras neural network library [22]. The proposed architecture involves an input layer, a batch-normalisation layer, two hidden dense layers consisting of 36 and 12 nodes respectively and the output layer. The rectified linear unit (ReLU) is chosen as the activation function of the hidden dense layers, while Softmax is used for the output layer. The model was trained for 10 epochs using the Adagrad optimizer and categorical cross-entropy as loss function. The selected MLP model was integrated in the Cognitive DA module of the DARE, and it was further compared to the Random Forest classifier in terms of speed, robustness, and accuracy in capturing the essence of this system.

4 Experimental Evaluation and Results

4.1 Datasets

To successfully build efficient anomaly detection and threat classification models, a big amount of data is required to train and test their detection accuracy. At the same time, most of the existing network traffic datasets that are publicly available are either outdated, unlabelled or unreliable. Some of these suffer from lack of traffic diversity and volume, some do not cover the variety of known attacks, while others are missing or hiding features that are present in the most common network protocols. For the purpose of training and evaluating the algorithms of this paper, the CICIDS2017 [23] dataset was primarily used. This dataset contains benign traffic along with the most up-to-date common attacks, approaching real-world data as much as possible. The traffic is available in packet (PCAP) format, while the dataset also includes the results of the network traffic analysis using CICFlowMeter with labeled flows, based on the time stamp, source and destination IPs, source and destination ports, protocols and attack, in text format (CSV). The latter format which includes labelled flows was extremely useful in our case, since it allowed for the training and the evaluation of our proposed algorithms. Moreover, the abundance of network traffic features (more than 80) facilitated the extraction of only the relevant ones found in the netflow (NFCAPD) protocol.

The CICIDS2017 benchmark dataset contains the abstract behaviour of 25 users based on the HTTP, HTTPS, FTP, SSH, and email protocols. The data capturing period starts at 9 a.m., Monday, July 3, 2017 and ends at 5 p.m. on Friday July 7, 2017, for a total of 5 days. Monday is the only day that contains exclusively normal activity so that only benign traffic is created for this day. For the rest of the days, apart from benign traffic, several different attacks are performed, including Brute Force FTP, Brute Force SSH, DoS, Heartbleed, Web Attack, Infiltration, Botnet and DDoS. As follows:

- **Monday**: Benign traffic only
- **Tuesday:** Bruteforce attack using a variety of password cracking tools (Hydra, Medusa, Ncrack, Metasploit etc.) and multi-threaded FTP-Patator and SSH-Patator.
- **Wednesday**: DoS attacks using a 4 different tools (Hulk, GoldenEye, Slowloris, Slowhttptest) and Heartbleed attack using Heartleech to scan for vulnerabilities.
- **Thursday morning:** Web attack using the Damn Vulnerable Web App (DVWA), which is a vulnerable PHP/MySQL web application.
- **Thursday afternoon:** Infiltration attack using Metasploit to infect a victim using a malicious file from the cloud or USB, then nmap and portscan on the victim network.
- **Friday morning:** Botnet attack using ARES, a Python-based botnet, which can provide remote shell, file upload/download, capturing screenshots and key logging.
- **Friday Afternoon**: DDoS attack using the Low Orbit Ion Canon (LOIC) to send UDP, TCP, or HTTP requests to the victim.
- **Friday Afternoon-2:** Portscan attack over the all Windows machines, by the main nmap switches.

Additionally to the aforementioned dataset, a PCAP from malware-traffic-analysis.net [24] was used, it contains a sample of the WannaCry ransomware spreading via the EternalBlue exploit. The WannaCry crypto worm is a modern network threat that resulted in a worldwide cyber-attack in May 2017. It targeted computers running the Microsoft Windows operating system by encrypting data and demanding ransom payments in the Bitcoin cryptocurrency. It propagated through EternalBlue, an exploit in older Windows systems made public a few months prior to the attack. WannaCry uses the MS17-010 exploit to spread to other machines through NetBIOS. The malware contains exploits in its body that are used during the exploitation phase. After installed, it generates random IP addresses, not limited to the local network. Thus, it can spread not only to other machines in same network, but also across the Internet if sites allow NetBIOS packets from outside networks. This could explain its widespread infection to unpatched systems. For our experiments, the Wannacry traffic was mixed with the normal traffic (Monday) from the CICIDS2017 dataset, as a means to further evaluate the efficiency of our proposed anomaly detection algorithms against state-of-the-art attacks.

4.2 Analysis

Data pre-processing

The use of the aforementioned datasets involved a number of pre-processing steps to facilitate their exploitation in the training and evaluation of the proposed cybersecurity algorithms. More specifically, both anomaly detection and threat classification implementations require that the input data is in CSV format and that it follows an appropriate schema. Since the aim of our cybersecurity solution is to detect anomalies by leveraging the Netflow traffic protocol (NFCAPD), only features that are present in this protocol were used in the analysis procedure.

As both the CICIDS2017 and the Wannacry datasets contain much richer information compared to the Netflow protocol, a python script was used to automate their conversion to a more suitable format of reduced dimensionality. The features that were segregated by this script are displayed in table 1.

Abbreviation	**Data field**
tr[1]	Flow received timestamp
td	Flow duration
sa[1]	Source IP address
da[1]	Destination IP address
sp	Source Port
dp	Destination Port
pr	Protocol
flg[2]	Flags
ipkt	Input Packets
ibyt	Input Bytes
Label	Threat type of the individual flow

Table 1. Traffic features used by the cybersecurity algorithms

In the case of Oneclass SVM, IForest, LoF, we decided to transform these features following the same method in order to compare them. First, flags and protocols categorical variables were translated to binary arrays of features (one hot encoding) so it can increase the algorithm performance, without introducing artificial distances. Second, some of the features offer a wider frequency distribution over logarithm scale, so we use their logarithmic values. This is the case of td, ipkt and ibyt. Finally, we added a counter of similar flows (time,sa,da,sp,dp,pr,flg).

Anomaly detection

Previous sections performed the analysis of the state of the art on anomaly detection and the explanation of the different algorithms that have been used in SHIELD. In this subsection, the details of the experiments performed with these algorithms and the obtained results are displayed.

Results are presented for two algorithms (OCSVM and Autoencoders) out of the five that have been explained. Note that LDA, the original algorithm used in Spot, is not a machine learning algorithm but a pure probabilistic algorithm. LDA does not produce a model from a training set which can be validated with an independent testing set, but

[1] Only used for flow identification, not used in the training procedure
[2] Used exclusively by the anomaly detection algorithms

uses a single dataset and classifies the flows by probability. Hence, it cannot be compared with the other ML based algorithms of the paper. In the case of Isolation Forest and Local Outlier Factor, the performance results are far away from the OCSVM and autoencoders. They are therefore not relevant for comparison.

All the algorithms are trained with a subset containing the 80% of the traffic corresponding to Monday in the CICIDS2017. The produced models are tested against 3 different test (subsection 3.1), namely Wednesday and Friday Afternoon-2 of the CICIDS2017 dataset and the WannaCry ransomware spreading via the EternalBlue exploit (mixed with 20% of the Monday traffic). All of them contain benign and malign traffic in different proportions, specifically:

- Wednesday contains 3% of malign traffic
- Friday Afternoon-2 contains 55% of malign traffic
- Wannacry test set is mixed with 20% of Monday traffic, creating an overall traffic with 1% of malign data.

The obtained results are shown with the accuracy (true positives plus true negatives divided by all the samples), precision (true positives divided between true positives plus false positives) and recall (true positives divided by true positives plus false negatives) in table 2.

Dataset	**One-class SVM**			**Autoencoder**		
	Accuracy (%)	**Precision (%)**	**Recall (%)**	**Accuracy (%)**	**Precision (%)**	**Recall (%)**
Wednesday (4 DoS, Heartbleed)	0.66	0.48	0.57	0.66	0.70	0.12
Friday afternoon (LOIC DDoS)	0.50	0.99	0.49	0.98	0.97	0.99
Wannacry	0.83	0.99	0.83	0.97	0.25	0.99

Table 2. Anomaly detection results

Fig. 1. Comparison of anomaly detection algorithms in terms of accuracy

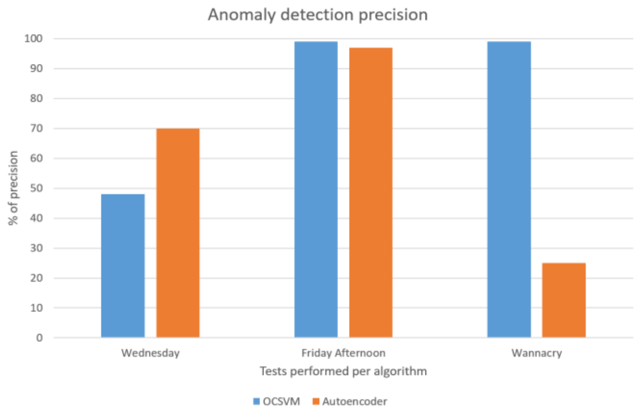

Fig. 2. Comparison of anomaly detection algorithms in terms of precision

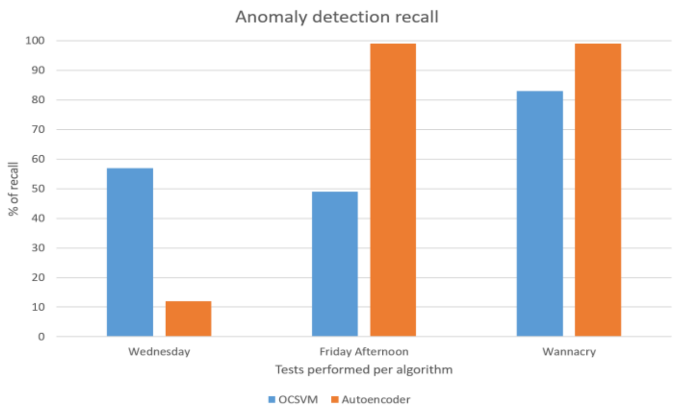

Fig. 3. Comparison of anomaly detection algorithms in terms of recall

The results shows that Deep Learning Autoencoder outperforms OCSVM in terms of accuracy, meaning that it correctly classifies more traffic between malicious and benign traffic. However, it has worse results in terms of precision in the case of Wannacry, which means that it produces many false positives. In the case of recall, the results depend mostly on the attack type. While in Wednesday traffic Autoencoder produces many false negatives (malicious traffic which is labelled as benign), on the other attacks, its recall metric outperforms OCSVM.

Some of the low results of Autoencoders may be due to the higher sensibility of neural networks over traditional approaches like OCSVM. This higher sensibility means that autoencoders have high potential but need to be much more finely tuned in order to obtain stable results. Moreover, the simplicity of the used neural networks also contributes to less smooth results. An Autoencoder with more inner layers could improve the situations. Another possible improvement would be to build a hybrid detector which combines the stability of OCSVM with the improved sensibility of Autoencoders.

Threat classification

The two threat classification models previously described, namely the Random Forest (RF) and the Multi-Layer Perceptron (MLP), were trained and evaluated using the CICIDS2017 dataset. More specifically, for each one of the individual datasets containing network threats (all except for Monday), 80% of the logs was randomly selected for the training of the models, while the remaining 20% of each dataset was set aside to be used as an evaluation set. To ensure unbiased results, a 10-fold cross-validation approach was followed on the training set, assessing the predictive performance of the models outside the training sample. The training set was therefore partitioned into 10 equal subsets (each one of them called a fold), and was trained using 9 of them, while the last one was kept for validation. This procedure resulted in 10 candidates, RF and MLP models, which accuracies are calculated by predicting the class of the logs on the left-out validation fold. The best candidate model for each approach is then selected to be further evaluated on the previously unseen evaluation set (20% of the overall dataset). Table 3 compares the predictive performance of the two derived models for each individual dataset. Since some of the classification tasks contained only 2 possible classes, namely normal traffic and one threat (binary classification), while others contain more than one threats (multiclass classification), accuracy, precision and recall were used as universal metrics to allow for comparison between the different models and problems. Other model-specific metrics (e.g Loss, AUC) are also calculated and are available upon request.

CICIDS2017	Random Forest			MultiLayer Perceptron		
Day	**Acc. (%)**	**Prec. (%)**	**Rec. (%)**	**Acc. (%)**	**Prec. (%)**	**Rec. (%)**
Tuesday (Bruteforce, FTP+SSH Patator)	1.00	1.00	1.00	0.97	0.75	0.27
Wednesday (4 DoS, Heartbleed)	0.98	0.98	0.98	0.92	0.92	0.86
Thursday morning (DVWA Web attack)	0.99	0.99	1.00	1.00	1.00	0.63
Thursday afternoon (Infiltration)	1.00	1.00	0.99	1.00	1.00	0.70
Friday morning (ARES bot)	1.00	1.00	1.00	1.00	0.93	0.63
Friday afternoon (LOIC DDoS)	0.98	0.98	0.98	0.93	0.89	1.00
Friday afternoon-2 (Portscan)	1.00	1.00	1.00	0.98	0.97	1.00

Table 3. Threat classification results

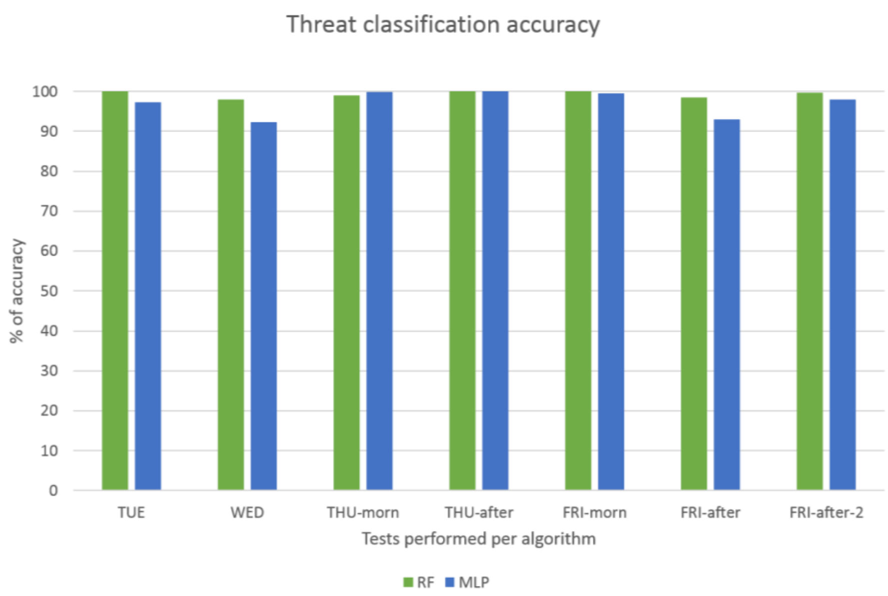

Fig. 4. Comparison of threat classification algorithms in terms of accuracy

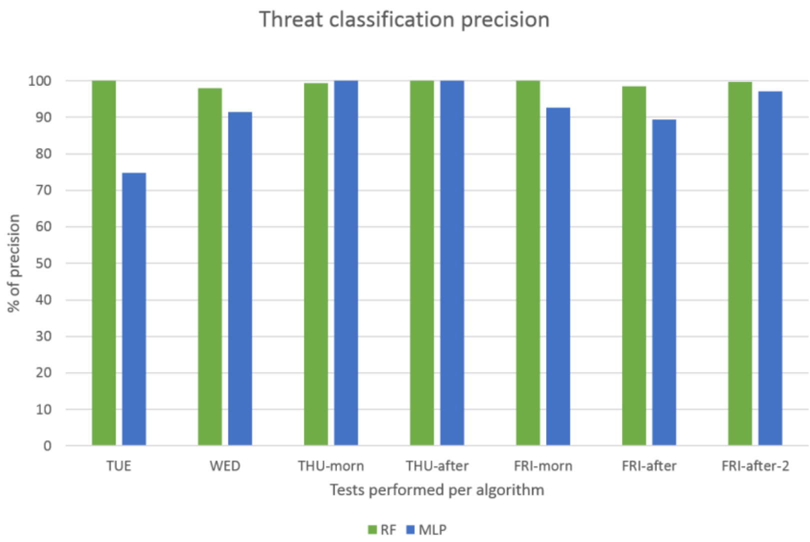

Fig. 5. Comparison of threat classification algorithms in terms of precision

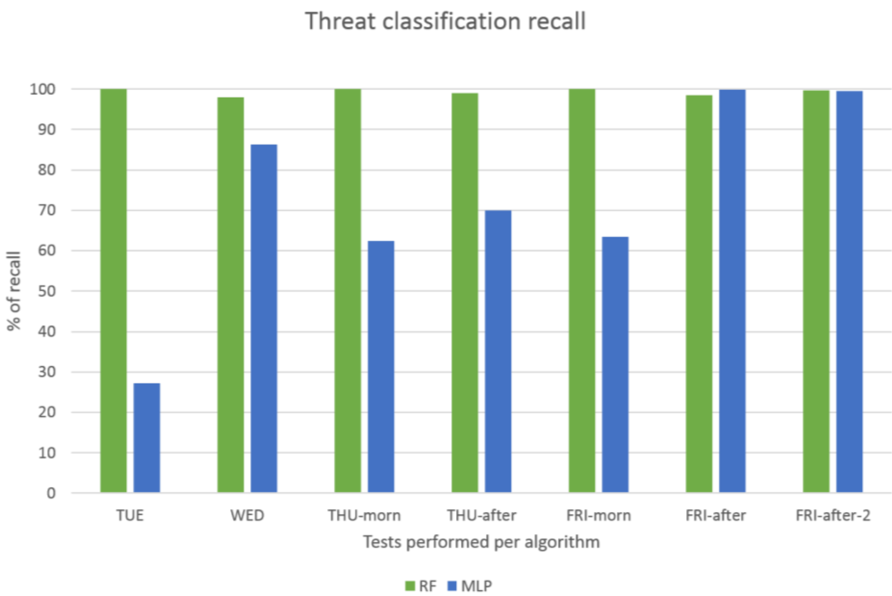

Fig. 6. Comparison of threat classification algorithms in terms of recall

Based on the obtained results, we can conclude that the predictive power of the RF model surpasses that of the MLP in almost every threat classification case. The latter, while it achieves satisfactory results in almost every benchmarking scenario, it is lagging behind the RF in cases of heavily imbalanced datasets. This can be partly attributed to the fact that our MLP model was designed and tuned to fit all classification tasks, hence no fine-tuning or resampling techniques with regards to each specific dataset were performed (e.g. the same number of layers, nodes, activation functions, and features were selected for every case). Generally, it can be difficult to achieve a good, universal MLP parameterization via optimization, since these models suffer from diminishing returns with added complexity, and are prone to overfitting when used in combination with imbalanced datasets. On the other hand, Random Forest seems to lead to robust and efficient detection of a large number of attacks, regardless of their type. Moreover, the training and execution time of the Random Forest approach was significantly less compared to that of the MultiLayer Perceptron. It should be noted that these observations and results are in accord with earlier research on the same dataset by [25], nevertheless the predictive ability of our proposed models significantly outmatches the experimental results of other suggested algorithms on the same dataset.

5 Business perspective

This last part focuses on insights on the business potential of the presented algorithm models as part of the SHIELD solution and elaborates on the business model which is derived from similar cybersecurity solutions. We propose the Telecom Security Service (TSS) business model that introduces a network-based solution which combines the availability of a Telco network operator, including NFV technology, with the SHIELD Framework to deploy a MSS (Managed Security Service). The TSS service is a complete solution that offers value to enterprises through SD-WAN type technology combined with a distributed NFV framework (uCPE [26] and vNSFO-Orchestrator), Visualization data (Big Data), machine learning intelligence and response based on network data and events (DARE). Models proposed and evaluated in this paper are integrated into the DARE to offer clear cybersecurity alerts and offer response capabilities through NFV framework. The main added values include:

- Maximum flexibility and agility in the enterprise demands, with a modular service offered by the network softwarisation and expandable catalogue in a vNSF Store.
- Centralisation of the intelligence and control of the security enterprise resources, in a unified dashboard.
- Improvement of network security driven by a pervasive and coherent policy based model.
- Cost reduction and operational synergies based on NFV optimized for uCPE solutions: one hardware, multiple vNSFs.
- Short time to market, no complex network deployments, exploitation of cloud and virtualization technology to quick contract and deploy.
- Possibility to outsource the day-by-day Security operation activities in Telco SoC: incident handling, security policies deployment, preventive actions or continuous monitoring. All this action are performed without losing control or visibility.

SHIELD technology can be applied in a wide range of potential different business models. Integrating the SHIELD framework in a Telco infrastructure for security services will allow to enhance the telco security service portfolio in a cost-effective manner, achieving better and faster response to new demands at the same time.

6 Conclusion and future work

This paper summarizes the SHIELD architecture and describes the different Machine Learning algorithms used to secure the users from malicious actors. The functionality of SHIELD's analytics component - the DARE - is showcased, by presenting several anomaly detection and threat classification models that are developed to address real-world cybersecurity scenarios like exploiting the Netflow traffic protocol. These models are evaluated using modern benchmarking datasets and achieve valid results that reflect the capabilities of a reliable cybersecurity platform. More specifically, in terms of anomaly detection, the One-Class-SVM, the Isolation Forest, the Local Outlier Factor and the deep-learning autoencoder methods were compared, in a range of different cybersecurity problems. Deep Learning autoencoders seem to have a very high potential but as often, deep learning algorithms need to be finely tuned in order to obtain stable results and avoid performance issues. With regards to threat classification, the tree-based Random Forest and the deep-learning based MultiLayer Perceptron both obtained remarkable results that exceed the performance of other reported algorithms on the same dataset, with the former being more robust in multiclass classification problems. For the exploitation of the aforementioned technologies, the Telecom Security Service business model is proposed, combining network function virtualisation and centralized control of the Security-as-a-Service platform for enhanced security and operability with reduced cost.

In the future, we would like to include more types of modern attacks in different OSI layers, as well as combine them with the existing ones for the evaluation of all our proposed models on a comprehensive dataset. We are also planning to examine feature selection and resampling techniques to further improve our detection results.

References

1. S. Omar, A. Ngadi, and H.H. Jebur, "Machine learning techniques for anomaly detection: An overview", International Journal of Computer applications, vol. 79, No 2, 2013
2. M Umer, M. Sher, and Y. Bi, "Applying One-Class Classification Techniques to IP Flow Records for Intrusion Detection", Baltic Journal of Modern Computing, num 1, pp 70-86, http://dx.doi.org/10.22364/bjmc.2017.5.1.05
3. H.Kim, K.Claffy, M.Fomenkov, D.Barman, M.Faloutsos, K.Lee: "Internet traffic classification demystified: myths, caveats, and the best practices." In Proc. of ACM CoNEXT, 2008 Madrid, Spain
4. Y.Lim, H.Kim, J.Jeong, C.Kim, T.Kwon, Y.Choi: "Internet traffic classification demystified: on the sources of the discriminative power." 2010, In CoNEXT, pg. 9
5. D.M.Blei, A.Y.Ng, M.I.Jordan: "Latent Dirichlet Allocation," Journal of Machine Learning Research, 2003, vol. 3, no. 4-5, pp. 993-1022

6. "Apache Spot," The Apache Foundation. Available at: http://spot.incubator.apache.org/
7. Schölkopf, B., Williamson, R. C., Smola, A. J., Shawe-Taylor, J., & Platt, J. C. (2000). Support vector method for novelty detection. In Advances in neural information processing systems (pp. 582-588)
8. Liu, F. T., Ting, K. M., & Zhou, Z. H. (2008, December). Isolation forest. In Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on (pp. 413-422)
9. Carrasquilla, U. (2010). Benchmarking algorithms for detecting anomalies in large datasets. MeasureIT, Nov, 1-16
10. Domingues, R., Filippone, M., Michiardi, P., & Zouaoui, J. (2018). A comparative evaluation of outlier detection algorithms: Experiments and analyses. Pattern Recognition, 74, 406-421
11. Breunig, Kriegel, Ng, and Sander (2000) LOF: identifying density-based local outliers. Proc. ACM SIGMOD
12. Lazarevic, A., Ertoz, L., Kumar, V., Ozgur, A., & Srivastava, J. (2003, May). A comparative study of anomaly detection schemes in network intrusion detection. In Proceedings of the 2003 SIAM International Conference on Data Mining (pp. 25-36). Society for Industrial and Applied Mathematics
13. N.n. Thi, V.L Cao, and N. Le-khac, "One class collective anomaly detection based on LSTM-RNNs", Transactions on Large-Scale Data- and Knowledge-Centered Systems XXXVI, Springer Berlin Heidelberg, 2017, ISBN: 978-3-662-56265-9
14. T.K.Ho: "Random Decision Forests", Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC, 1995, pp. 278–282
15. L.Breiman: "Bagging predictors", Machine Learning, 1996, 24 (2): 123–140. CiteSeerX 10.1.1.32.9399 Freely accessible. doi:10.1007/BF00058655
16. M.Fernández-Delgado, E.Cernadas, S.Barro, D.Amorim: "Do we need hundreds of classifiers to solve real world classification problems?", The Journal of Machine Learning Research, 2014, 15(1):3133–3181
17. G.Biau: "Analysis of a random forests model", J. Mach. Learn, 2012, Res. 13, 1, 1063-1095
18. "Apache Spark MLlib" , The Apache Foundation, Available at: https://spark.apache.org/mllib/
19. C.Siaterlis, V.Maglaris: "Detecting incoming and outgoing DDoS attacks at the edge using a single set of network characteristics", 10th IEEE Symposium on Computers and Communications (ISCC'05), 2005, pp. 469-475. doi: 10.1109/ISCC.2005.50
20. S.Andropov, A.Guirik, M.Budko: "Network anomaly detection using artificial neural networks", 20th Conference of Open Innovations Association (FRUCT), St. Petersburg, 2017, pp. 26-31. doi: 10.23919/FRUCT.2017.8071288
21. "Deep Learning Studio", deepcognition.ai, Available at: https://deepcognition.ai/products/desktop/
22. "Keras: The Python Deep Learning library", Available at: https://keras.io/
23. "Intrusion Detection Evaluation Dataset (CICIDS2017)", Canadian Institute for Cybersecurity, available at: http://www.unb.ca/cic/datasets/ids-2017.html
24. "PCAP of WannaCry spreading through EternalBlue", Malware Traffic Analysis Net, Available at: https://www.malware-traffic-analysis.net/2017/05/18/index2.html
25. I. Sharafaldin, A. Lashkari, A Ghorbani: "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization", 4th International Conference on Information Systems Security and Privacy (ICISSP), Portugal, January 2018
26. "Understanding the use of Universal CPE", Available at:, https://www.sdxcentral.com/articles/contributed/understanding-use-universal-cpe/2017/07/