Detecting Stealthy Cobalt Strike C&C Activities via Multi-Flow based Machine Learning

Fabian Martin Ramos
Department of Computer Science
George Mason University
Fairfax, VA 22030, USA
fmartinr@gmu.edu

Xinyuan Wang
Department of Computer Science
George Mason University
Fairfax, VA 22030, USA
xwangc@gmu.edu

***Abstract*—Cobalt Strike is a penetration testing software that supports stealthy command and control (C&C). As Cobalt Strike C&C always encrypts its payload and can mimic various normal traffic (e.g., HTTP, HTTPS), it is very difficult to distinguish and detect stealthy Cobalt Strike C&C traffic from normal (encrypted) traffic. As a result, many recent cyber attacks (e.g., 2020 SolarWind data breach) used Cobalt Strike C&C to surreptitiously control compromised systems and exfiltrate sensitive data without being detected. To protect our mission critical systems from sophisticated cyber attacks, it is critically important to develop capabilities to effectively detect real world Cobalt Strike C&C activities from encrypted traffic.**

In this paper, we investigate how machine learning models can help detect stealthy Cobalt Strike C&C activities from encrypted traffic. Based on analysis of real world Cobalt Strike C&C traffic, we have developed novel machine learning features across multiple packet flows that capture some inherent characteristics of encrypted Cobalt Strike C&C traffic. We have evaluated several machine learning algorithms with our multi-flow based machine learning features and real world Cobalt Strike C&C traffic captured from real world cyber attacks Our experimental results demonstrate that our neural network based detection is able to detect 90.9% real world Cobalt Strike C&C activities with 0.4% false positive rate.

***Index Terms*—Cobalt Strike C&C, Intrusion Detection System, Machine Learning**

I. Introduction

The digital technologies have become a critical part of the lives of billions around the world, used in activities from the exchange of information to monetary transactions. Cyber attacks have emerged as an increasingly more serious threat to these activities, with an estimated global cost of cyber crime nearing 6 trillion dollars per year [1]. Furthermore, cyber attacks are also used as a military action to disrupt social and economic activities, as well as disable critical infrastructure.

Cobalt Strike is a penetration testing software that has recently become a favorite tool among malicious actors to perform sophisticated cyber attacks on vulnerable targets due to its advanced evasion capabilities. As reported by Cisco's Talon Incident Report [2], Cobalt Strike was involved in 66% of the ransomware attacks in the last quarter of 2020. Cobalt Strike is also used as an advanced persistent threat (APT) such as the case of the SolarWinds hack in 2020 [3], where the attackers delivered a customized Cobalt Strike payload via malicious software updates to 18,000 customers of the Orion software, including many global companies (e.g., Microsoft, Cisco) and various US government agencies.

Cobalt Strike establishes a stealthy command-and-control (C&C) channel between an infected victim and the attacker, which enables the attacker to explore, exploit and control the victim system and exfiltrate sensitive information. Cobalt Strike allows attackers to use Malleable C&C profiles to customize and disguise the C&C traffic as legitimate traffic, which makes the C&C traffic very stealthy and hard to detect. For example, the 2020 SolarWinds data breach attack has remained undetected for over 9 months. Microsoft president Brad Smith considered the 2020 SolarWinds attack as "the largest and most sophisticated attack the world has ever seen" [4]. To the best of our knowledge, there is no published approach that can reliably detect real world stealthy Cobalt Strike C&C activities.

In this paper, we investigate how machine learning models can help detect stealthy Cobalt Strike C&C activities from encrypted traffic. Based on analysis of real world Cobalt Strike C&C traffic, we have developed novel machine learning features across multiple packet flows that capture some inherent characteristics of encrypted Cobalt Strike C&C traffic. We have evaluated several machine learning algorithms with our multi-flow based machine learning features and real world Cobalt Strike C&C traffic captured from real world cyber attacks Our experimental results demonstrate that our neural network based detection is able to detect 90.9% real world Cobalt Strike C&C activities with 0.4% false positive rate.

The rest of the paper is organized as follows: section II overviews the Cobalt Strike framework and characterizes its C&C traffic patterns. Section III presents the threat model, feature extraction and machine learning model development of the multi-flow based detection approach. Section IV describes the empirical validation results with the Cobalt Strike C&C traffic from real world attacks. We review related works in section V, and conclude the paper with potential future works in section VI.

II. Background

A. Cobalt Strike

Cobalt Strike was originally developed as a penetration testing tool by Raphael Smudge in 2012 [5]. Specifically, Cobalt

Strike provides functionalities for the penetration testers to 1) perform target reconnaissance and identify known vulnerabilities in targets; 2) create trojans and malicious website clone for drive-by attacks; 3) compromise the chosen vulnerable targets by deploying and injecting malicious agents – called Beacon into the targets; 4) secretly control the compromised systems via key logging, taking screenshots, executing arbitrary command and payload, downloading additional malware, injecting a Beacon in other processes; and 5) encrypt its C&C traffic and mimic popular network traffic such HTTPS. Due to its powerful and stealthy C&C features, Cobalt Strike has been widely used by many real world cyber attacks [6].

Cobalt Strike framework consists of 1) the team server; 2) the client; and 3) the Beacon. The team server is the C&C server that, on behalf of the client, interacts with victims that have been infected with some Beacon. The client is the system used by the attacker to interact with the Beacon via the team server. The Cobalt Strike Beacon is the malware payload that opens and maintains a backdoor on a victim system, and it has two parts: the payload stage, and the payload stager. The payload stager is a smaller program that is used to download the payload stage and inject it into the memory of the victim, and then pass the execution to it. The payload stage is the actual backdoor that runs in memory of the victim that can establish a stealthy C&C channel to the C&C server.

B. Cobalt Strike C&C Traffic Patterns

Cobalt Strike support C&C via multiple popular protocols (e.g., HTTP, HTTPS, DNS). In this paper, we focus on the detection of Cobalt Strike C&C disguised as HTTPS traffic, which is most frequently used by attackers. Our analysis of Cobalt Strike C&C has shown the following common patterns:

- When a victim is infected, the Beacon will periodically communicate with the team server to retrieve any tasks requested by the attacker. It supports both asynchronous and interactive communications, and the time interval that the Beacon sleeps for between calls is determined by the attacker.
- The Beacon initiates the transaction with the C&C server by sending a GET request with information about the infected system. The server then responds with the tasks for the Beacon to execute.
- The Beacon executes the tasks, and sends the execution results back to the C&C server in a POST request. Then, the Beacon discards the contents of the server response.
- By using GET and POST messages, Cobalt Strike C&C imitates a legitimate HTTPS transaction between a client and a server. Each request-response exchange opens a new TCP connection and, in the case of the HTTPS communications, a new TLS session is established.
- Cobalt Strike encodes and encrypts the C&C data exchanged between the Beacon and the C&C server, which defeats any content-based detection.
- Cobalt Strike Beacon connections or sessions tend to be short in time – between a few milliseconds to a few seconds, depending on the channel bandwidth.

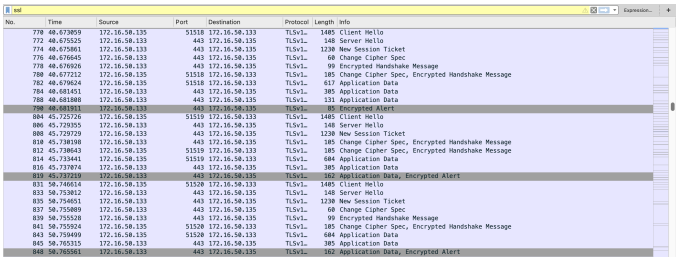


Fig. 1. Captured HTTPS Packets from Cobalt Strike C&C Traffic

Figure 1 shows an interaction between a Cobalt Strike Beacon and a C&C server disguised as HTTPS traffic. Specifically, the interaction involves multiple HTTPS sessions established by standard TLS handshake. Once each HTTPS session is established, the Beacon and the team server start to exchange task requests and task results encoded and encrypted as HTTPS *application data*. If the Beacon receives task requests, it closes the current TCP connection and opens a new TCP connection and TLS session to send the task results back to the server as soon as their execution of the tasks is completed. Each HTTPS session is terminated by *encrypted alert* message.

C. Cobalt Strike Malleable C&C Profiles

Cobalt Strike utilizes Malleable C&C profiles to modify the network indicators of the C&C traffic between the Beacon and the team server, which enables an attacker to disguise the Beacon activity to look like other malware or blend in with legitimate traffic.

The Malleable C&C profile includes 1) the *sleeptime* and *jitter* which modify the Beacon callback interval; 2) the *data jitter* which enables the attacker to append a random length of null data up to the server response payload; 3) the *useragent* which sets the User-Agent string in HTTP requests. The Malleable C&C profile can also configure the HTTP headers, the SSL certificate, the URI of the HTTP requests, and the HTTP verb in the http-post section.

III. A Multi-Flow based Detection Approach

A. Threat Model

Our objective is to develop an effective machine learning based detection of stealthy Cobalt Strike C&C activities, and we focus on detecting Cobalt Strike Beacon traffic disguised as HTTPS traffic. We assume the following regarding the Cobalt Strike C&C traffic:

- The system has already been infected with a Cobalt Strike HTTPS Beacon and the initial infection has not been detected by an IDS/IPS.
- The attackers use a single C&C server to communicate with the Beacon but they can use any Malleable C&C profile configuration.

B. Cobalt Strike HTTPS Traffic Analysis

From an observer's perspective, the HTTPS traffic between the Beacon and the Cobalt Strike server is very similar to the legitimate HTTPS traffic between a HTTPS client and a HTTPS server. Since Cobalt Strike C&C always encrypts its payload and can mimic normal HTTPS traffic, it is infeasible to use content or simple header information to effectively distinguish Cobalt Strike C&C traffic from normal HTTPS traffic.

The key to distinguish stealthy Cobalt Strike C&C traffic from normal HTTPS traffic is to identify those inherent characteristics of encrypted Cobalt Strike traffic and build novel machine learning features based those inherent traffic characteristics.

By analyzing the interactions between HTTPS client and HTTPS server, we observe the following characteristics of normal HTTPS traffic: 1) HTTPS servers tend to be the first to send TLS application data to clients; and 2) HTTPS servers tend to send more TLS application data to clients than clients send to servers. In contrast, the Cobalt Strike HTTPS beacon traffic exhibits some inherently unique pattern with various malleable C&C profiles.

First, unlike legitimate traffic which is mostly aperiodic, Cobalt Strike C&C traffic exhibits strong periodic pattern as the Beacon periodically calls back to the team server to retrieve tasks. Second, Cobalt Strike C&C used multiple consecutive TLS sessions between one Beacon and the server. Third, the Beacon TLS sessions normally have very short duration, while legitimate TLS sessions that tend to last longer. Lastly, Cobalt Strike allows attackers to modify the server IP address and ports that are used in the transactions between the Beacon and the team server. Similarly, the attacker can modify the values in the headers of the HTTPS packets exchanged, and the size of the data payload sent by the server.

Due to mainly being used for data exfiltration, the Beacon sends more data to the C&C server than it receives in most scenarios, even when a high *data jitter* is configured in the Malleable C&C profile. Furthermore, the Beacon always initiates the transactions with the team server by sending the first *application data* packet after the TLS session has been established via the TLS handshake. Such patterns are very rare in normal HTTPS traffic.

C. Feature Extraction

Based upon the above analysis, we are convinced that it is better to consider a group of HTTPS transactions between a client and a server (or the Beacon and the team server) instead of the individual transactions when extracting machine learning features to differentiate the HTTPS Beacon traffic from legitimate HTTPS traffic.

In order to capture the inherent interaction patterns of the Cobalt Strike C&C traffic, we consider those TLS sessions between the same pair of IP addresses that are not less than 30 seconds apart as one TLS group, and we define the following group-based features to capture the unique characteristics of Cobalt Strike C&C traffic:

- **dur:** the total duration of all the HTTPS sessions within one HTTPS group.
- **nses:** the number of HTTPS sessions with the HTTPS group.
- **cltmoredata:** the fraction of the HTTPS sessions within the group that the client sends more application data than server.
- **clt1stdata:** the fraction of the HTTPS sessions within the group that the client sends the application data to server first.
- **dur_mean:** the mean of the duration (in seconds) of all HTTPS sessions within the group.
- **dur_std:** the standard deviation of the duration (in seconds) of all HTTPS sessions within the group.
- **int_mean:** the mean of the interval between successive HTTPS sessions within the group.
- **int_std:** the standard deviation of the interval between successive HTTPS sessions within the group.

Feature *nses* enables us to capture inherent patterns among multiple TLS sessions of the Cobalt Strike Beacon traffic. The *clt1stappdata* feature captures the percentage of the TLS sessions in which the client sends the first *application data* packet after the session is established. Similarly, *cltmoredata* documents the percentage of the TLS sessions in which the victim sends more data to the team server than it receives. The mean and standard deviation (*dur_mean*, *dur_std*) of the duration enable us to identify those HTTPS groups with short session duration. Finally, features *int_mean*, *int_std* enable us to differentiate largely periodic Beacon traffic from largely aperiodic normal HTTPS traffic at the granularity of TLS sessions.

D. Machine Learning Model Development

We follow a series of steps in the development of our machine learning models such that each model may perform its best in detecting stealthy Cobalt Strike C&C traffic via given machine learning features. These steps can be generally divided into *data exploration*, *model building*, *hyperparameter tuning* and *model selection*.

The *data exploration* phase consists of the study of the input data and data collection, data preprocessing and feature engineering. The *model building* phase requires the separation of the collected data into two datasets, the selection of machine learning algorithms and building the machine learning models using the selected algorithms. The *hyperparameter tuning* phase involves the computation of the model performance on the test data and the modification of the model hyperparameters (parameters whose value can be modified to control the learning process) until a desired performance level has been reached or all possibilities have been exhausted. Finally, the *model selection* phase will compare the performance of all the different machine learning models generated and choose the model that performs best for the problem at hand.

We use the following popular metrics to evaluate the performance of different machine learning models: 1) accuracy; 2) weighted precision; 3) true positive rate (TPR); 4) false

positive rate (FPR); and 5) F1 score. The most important metrics will be the TPR, which is used to assess each model's ability to detect the Beacon C&C traffic, and the FPR, that reports the percentage of false alarms that each model may produce.

We also use the graphical confusion matrix to show the performance results of the best machine learning model. A confusion matrix is a table commonly used to describe the performance of a machine learning model when performing classification tasks over a test dataset whose labels (or classes) are known. It does so by establishing a relationship between the real label of a record and the predicted label of the same record, which graphically illustrates the number of records that have been correctly and incorrectly classified.

Given that most Internet traffic are not Cobalt Strike C&C traffic, we strive to obtain low detection FPR and high detection TPR at the same time to avoid the base rate fallacy [7] in real world scenario. We have selected the following machine learning algorithms:

- **Random forest:** A random forest algorithm combines many decision trees algorithms to reduce the risk of overfitting.
- **Neural network:** It consists of multiple hidden layer nodes, with each layer fully connected to the next layer, employing back propagation to modify the weight of each link between nodes during training.
- **Linear support vector machine:** It constructs a hyperplane using the OWLQN optimizer to optimize the Hinge loss, which can be used for binary classification.
- **Naïve Bayes:** Family of probabilistic classifiers based on the application of the Bayes theorem with strong independence assumptions between the features.
- **K-means:** A clustering algorithm for unsupervised machine learning that groups data records into a predefined number of clusters.

IV. Experiments and Results

A. Dataset Acquisition

As mentioned in the previous section, the development of a machine learning model requires a training dataset to train the machine learning algorithm and build the model. The evaluation of the resulting model needs a separate testing dataset. Both datasets should contain both legitimate traffic and Cobalt Strike C&C traffic.

Since there are currently no publicly available datasets containing Beacon traffic, it will be necessary to generate the datasets from network traffic packet captures (PCAPs) and use a proprietary software to extract the traffic characteristics into features that can be used by the machine learning model to classify the traffic.

The traffic used to generate the training dataset has been extracted from various sources. We have obtained the legitimate HTTPS traffic from the public datasets CICIDS17 [8] of the Canadian Institute of Cybersecurity and CTU-Normal-20 [9] of the Stratosphere Research Laboratory. The HTTPS Beacon traffic, on the other hand, has been generated and collected in our lab environment using 29 different Malleable C&C profiles and commands to emulate advanced penetration threats (APT), crimeware or normal traffic, or generated using randomizers such as [10]. The use of different Malleable C&C profiles reduces the possibility of overfitting the machine learning model to a specific Malleable C&C profile (i.e., the machine learning model is too closely aligned to the training dataset) thus enabling the machine learning models to perform better in the detection of previously unseen real world Cobalt Strike traffic.

TABLE I
Records in Training and Testing Datasets

Record Type	Training Dataset	Testing Dataset
Legitimate HTTPS	17,271	644
Lab HTTPS Cobalt Strike	29	0
Real HTTPS Cobalt Strike	0	33
Total Records	17,300	677

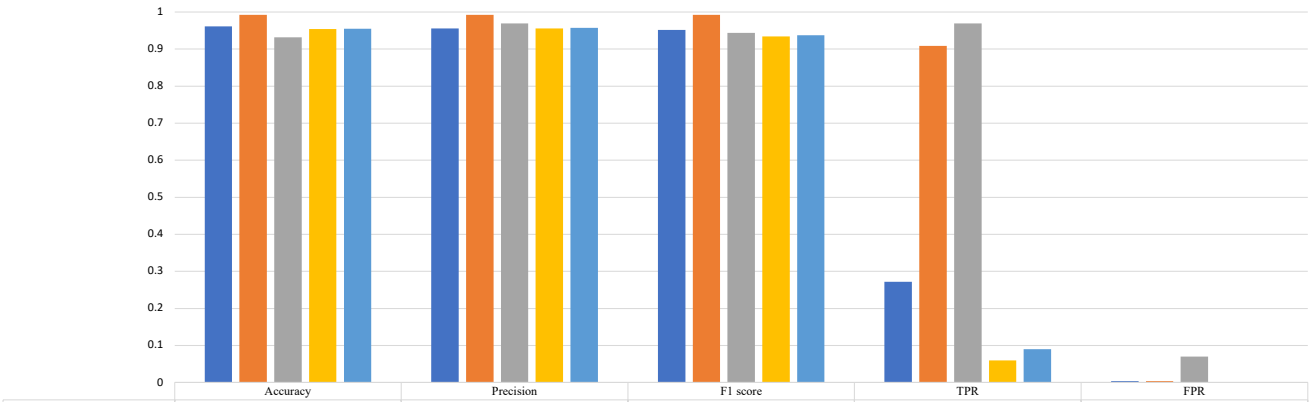
The testing dataset used to evaluate the Machine learning model has been obtained from the cybersecurity blog malware-traffic-analysis.net [11], which posts traffic from real world cyber attacks reported by companies such as Palo Alto Networks' Unit 42 and other cybersecurity blogs. The dataset includes real HTTPS Beacon traffic, legitimate HTTPS traffic and HTTPS traffic generated from other malware. All the traffic in these traces has been manually labeled by cybersecurity teams, which allows us to measure the performance of our machine learning based detection.

Each dataset record represents a group of HTTPS sessions between a pair of IP addresses in a PCAP. As shown in Table I, the training dataset contains 17,300 records out of which 29 records correspond to HTTPS Beacon traffic, and 17,271 records correspond to legitimate traffic. The testing dataset has 677 records with 33 HTTPS Beacon traffic records captured from real world cyberattacks and 644 records of legitimate or other malware traffic.

B. Feature Engineering

Every record of the dataset has 13 features extracted from the acquired traffic using a proprietary software tool. The first 5 features include: src_ip, src_p, dst_ip, dst_p and proto to uniquely identify each packet flow. To avoid making the machine learning model to be biased against certain IP addresses, and obtaining similar results to those using a signature-based approach, we choose not using these flow identifiers in training our machine learning models.

We use the remaining 8 features in the classification of the traffic. Specifically, the number of sessions (*nses*) that form each HTTPS group will be selected as an input feature due to the assumption that an attacker would require at least 10 connections (or HTTPS sessions in this case) between the Beacon and the C&C server to exfiltrate a significant amount of information about a system. As proof of this assumption, all of the real Beacon traffic records that will be used to test



	Accuracy	Precision	F1 score	TPR	FPR
Random forest	0.961	0.956	0.952	0.272	0.004
Neural network	0.992	0.992	0.992	0.909	0.004
Naïve Bayes	0.932	0.969	0.944	0.969	0.07
Linear Support Vector Machine	0.954	0.956	0.934	0.06	0
K-Means	0.955	0.957	0.937	0.09	0

Fig. 2. Detection Performance Comparison of Different Machine Learning Models

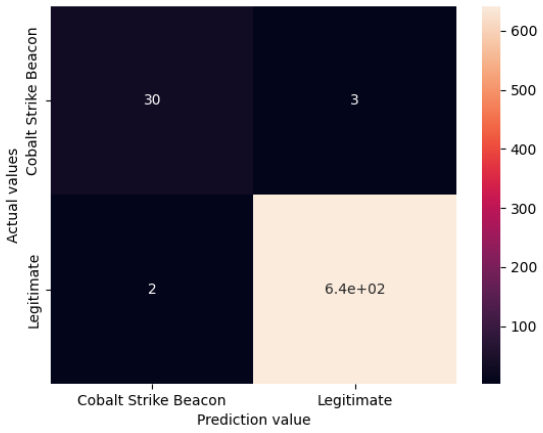


Fig. 3. Confusion matrix of the Neural Network Model

the model, have more than 10 sessions per group, while many of the legitimate traffic groups have less than 5 sessions.

The *cltmoredata* and *clt1stappdata* will be key features in the detection of Beacon traffic, as they have been specifically extracted due to being related to certain characteristics of the Beacon traffic. The rest of the features (*dur*, *dur_mean*, *dur_std*, *int_mean* and *mean_std*) will be used as input to the Machine learning models depending on how they affect the performance of the different models.

Since all the eight possible input features take continuous values, it will not be necessary to perform feature encoding for the group-based dataset. Like the connection-based dataset, the input features will be grouped into a vector-type feature called features using VectorAssembler, which will be used as the input of the model to facilitate its building process.

C. Group Based Cobalt Strike Detection

Using the Apache Spark machine learning library MLlib, we have developed the machine learning model and have conducted hyperparameter tuning to achieve best possible performance for each of the algorithms. While other machine learning libraries (e.g., TensorFlow) may have more sophisticated machine learning algorithms, they lack the Spark's ability to process large amounts of data in real-time.

Of the machine learning algorithms, the linear support vector machine and the k-means models fail to perform any significant detection of the Beacon traffic using any combination of input features or hyperparameters, as they achieve less than a 10% true positive rate.

As shown in Figure 2, the random forest model achieves a 27.2% detection rate with a false positive rate of 0.4% using the features *nses*, *cltmoredata*, *clt1stappdata*, *int_mean* and *int_std* as inputs. Due to its low detection rate, the random forest model would fail to detect most of the Cobalt Strike Beacon infections.

On the other hand, the naïve bayes model is built with the hyperparameter modelType = ḿultinomialánd achieves a 96.9% detection rate with a false positive rate of 7% using *nses*, *cltmoredata*, *clt1stappdata* and *dur_std* as input features. While it has high detection rate, the relatively high false alarm rate of the naïve bayes model prevents it from being a viable detection model in real world environments.

The neural network model performs the best, as it achieves a high true positive rate close to 91% while maintaining a 0.4% false positive rate, which is low enough to be deployed in a real environment. It uses all 8 features as inputs for the model, and the L-BFGS solver as hyperparameter. We have tried different seed values, which set the initial weights of the neural network algorithm, to obtain the best possible results.

It is important to note that different seeds could result different results obtained for the neural network model with the same hyperparameters and input values. For example, one

of the seeds resulted in 13 mixed traffic records being misclassified as Beacon traffic, or a 2.1% false alarm rate. However, upon closer inspection, at least eight out of the thirteen records that were misclassified belonged to two malware samples, ICEID and Bazarloader, which employ C&C communications to send information to infected hosts. Thus, it can be assumed that if malicious C&C traffic generated from other malware adopts similar patterns to the Cobalt Strike Beacon traffic, the intrusion detection system may be able to detect it.

The resulting group-based network intrusion detection system from this experiments will use the developed neural network model to make predictions on the HTTPS traffic captured after the information from the HTTPS session groups is extracted using the propietary program. Figure 3 shows the confusion matrix of the neural network model when tested against real traffic from Cobalt Strike attacks that include HTTP and HTTPS Beacon traffic, normal traffic, and malicious traffic from other malware sources. Since the focus of this paper is to detect Cobalt Strike C&C traffic, the malicious traffic from other sources has been labeled "legitimate".

V. Related Works

Machine learning techniques have been widely used in building cyberattack detection systems [12]–[17]. Most existing machine learning based cyberattack detection approaches (e.g., convolutional neural network based detection [17]) are designed to detect known exploits rather than stealthy C&C traffic, and they have achieved good results in the classification of known attacks from public cybersecurity datasets such as CICIDS17 and UNSW-NB15.

The increasing use of Cobalt Strike recent massive data breach attacks (e.g., 2020 Solarwind attack) has sprouted an interest from security researchers to develop the capability to determine if a host has been infected with a Cobalt Strike Beacon. However, many existing machine learning based detection approaches have been shown to be vulnerable to evasion [18]. Based on an extensive analysis on the Cobalt Strike C&C traffic encoding [19] and encryption [20] of the payload as well as the Malleable C&C profiles, Navarrete et al. [21] explained how Cobalt Strike's versatility makes it difficult to detect.

N. Kanzig et al. [22] proposed a method to identify C&C channels with a random forest classifier using features extracted from CICFlowMeter, an open-source tool for extracting flows from packet traces. Their work evaluates the model using Cobalt Strike traffic from the Locked Shields [23] competition datasets from 2017 and 2018, but does not consider the *data jitter* that can be introduced into the server responses, which will affect the packet flow features that they use to perform detection. In contrast, our multi-flow based detection can handle different data jitter and sleep jitter through the use of features such as *cltmoreappdata* and *int_std*.

Van der Eijk et al. [24] presented an approach to detect Cobalt Strike Beacons using NetFlow data along with a detection algorithm that classified the data depending on the features and threshold values. They set the thresholds of the detection algorithm with the values generated from a single Malleable C&C profile, and evaluated the algorithm with lab generated HTTP and HTTPS Beacon traffic and four Malleable C&C profiles.

Martin Ramos et al. [25] recently proposed using network flow information in building machine learning based detection of Cobalt Strike activities, and their machine learning model has achieved 47% detection true positive rate and 1.4% false positive rate. Instead of using the features of the overall network flow, our machine learning model uses features of the multiple flows sessions that are inherent to stealthy Cobalt Strike activities. Such unique features enabled our machine learning model to detect real world Cobalt Strike activities from encrypted traffic with 90.9% true positive rate and 0.4% false positive.

In summary, most previous machine learning based Cobalt Strike C&C detection approaches were evaluated with lab generated Cobalt Strike C&C traffic and a very few malleable C&C profiles. It is unknown if those machine learning based approaches were able to detect real world Cobalt Strike C&C traffic. In contrast, our multi-flow based approach was trained with 29 different Malleable C&C profiles and empirically evaluated with Cobalt Strike C&C traffic from real world cyberattacks. Empirical results show that our multi-flow based machine learning model performs significantly better than previous single flow based machine learning model [25].

VI. Conclusions

In this paper, we present a machine learning based approach to identify the stealthy Cobalt Strike C&C Beacon traffic. The key contribution of this work is the development of novel machine learning features across multiple packet flows that captured some inherent characteristics of encrypted Cobalt Strike C&C traffic.

Built upon our unique multi-flow machine learning features extracted from the set of all TLS sessions between two hosts, our machine learning models have been trained with lab generated Cobalt Strike HTTPS traffic using 29 different Malleable C&C profiles. We have then validated our machine learning models with real world Cobalt Strike Beacon infections captured from real world cyberattacks. Out neural network model is able to detect 90.9% previously unknown Cobalt Strike C&C traffic while maintaining a low 0.4% false positive rate. Our experimental results demonstrate that it is feasible to detect stealthy real world Cobalt Strike C&C activities from encrypted traffic once appropriate machine features have been identified.

As a future work, we plan to investigate if other machine learning models (e.g., CNN, RNN) could achieve better detection results and if our multi-flow based machine learning is effective in detecting other cyberattack traffic.

References

[1] C. Stouffer, "115 Cybersecurity Statistics + Trends to Know in 2023," September 2021, https://us.norton.com/blog/emerging-threats/cybersecurity-statistics.

[2] D. Liebenberg and C. Huey., "Quarterly Report: Incident Response Trends in Summer 2020," September 2020, http://blog.talosintelligence.com/2020/09/CTIR-quarterly-trends-Q4-2020.html.
[3] FireEye, "Highly Evasive Attacker Leverages SolarWinds Supply Chain to Compromise Multiple Global Victims With SUNBURST Backdoor," December 2020, https://www.fireeye.com/blog/threat-research/2020/12/evasive-attacker-leverages-solarwinds-supply-chain-compromises-with-sunburst-backdoor.html.
[4] L. Tung, "Microsoft: Solarwinds attack took more than 1,000 engineers to create."
[5] "Software for Adversary Simulations and Red Team Operations," https://www.cobaltstrike.com.
[6] A. Rahman, "Defining Cobalt Strike Components So You Can BEA-CONfident in Your Analysis," October 2021, https://www.mandiant.com/resources/blog/defining-cobalt-strike-components.
[7] S. Axelsson, "The Base-Rate Fallacy and Its Implications for the Difficulty of Intrusion Detection," in *Proceedings of the 6th ACM Conference on Computer and Communications Security (CCS 1999)*. ACM, November 1999, pp. 1–7.
[8] C. I. for Cybersecurity, "Intrusion Detection Evaluation Dataset (CIC-IDS2017)," https://www.unb.ca/cic/datasets/ids-2017.html.
[9] S. R. Laboratory, "Malware Capture Facility Project," https://www.stratosphereips.org/datasets-normal.
[10] "Malleable-C2-Randomizer," https://github.com/bluscreenofjeff/Malleable-C2-Randomizer.
[11] "A source for packet capture (pcap) files and malware samples," https://www.malware-traffic-analysis.net/index.html.
[12] C. Sinclair, L. Pierce, and S. Matzner, "An Application of Machine Learning to Network Intrusion Detection," in *Proceedings of the 5th Annual Computer Security Applications Conference (ACSAC 1999)*, December 1999.
[13] P. Sangkatsanee, N. Wattanapongsakorn, and C. Charnsripinyo, "Practical Real-Time Intrusion Detection Using Machine Learning Approaches," *Computer Communications*, vol. 34, no. 18, pp. 2227–2235, December 2011.
[14] M. Kulariya, P. Saraf, R. Ranjan, and G. P. Gupta, "Performance Analysis of Network Intrusion Detection Schemes Using Apache Spark," in *Proceedings of the 2016 International Conference on Communication and Signal Processing (ICCSP 2016)*. IEEE, April 2016, pp. 1973–1977.
[15] P. Toupas, D. Chamou, K. M. Giannoutakis, A. Drosou, and D. Tzovaras, "An Intrusion Detection System for Multi-class Classification Based on Deep Neural Networks," in *Proceedings of the 18th IEEE International Conference On Machine Learning And Applications (ICMLA 2019)*. IEEE, December 2019.
[16] P. G. Quinan, I. Traore, U. R. Gonhdi, and I. Woungang, "Unsupervised Anomaly Detection Using a New Knowledge Graph Model for Network Activity and Events," in *Proceedings of the 4th International Conference on Machine Learning for Networking (MLN 2021)*. Springer LNCS 13175, December 2021, pp. 117–130.
[17] E. Tufan, C. Tezcan, and C. Acartürk, "Anomaly-Based Intrusion Detection by Machine Learning: A Case Study on Probing Attacks to an Institutional Network," *IEEE Access*, vol. 9, pp. 50 078–50 092, 2021.
[18] J. Gardiner and S. Nagaraja, "On the Security of Machine Learning in Malware C&C Detection: A Survey," *ACM Computing Surveys*, vol. 49, no. 3, pp. 1–39, September 2017.
[19] C. Navarrete, D. Sangvikar, A. Guan, Y. Fu, Y. Jia, and S. Shibiraj, "Cobalt Strike Analysis and Tutorial: CS Metadata Encoding and Decoding," May 2022, https://unit42.paloaltonetworks.com/cobalt-strike-metadata-encoding-decoding/.
[20] C. Navarrete, D. Sangvikar, A. Guan, Y. Fu, Y. Jia, and S. Shibiraj, "Cobalt Strike Analysis and Tutorial: CS Metadata Encryption and Decryption," July 2022, https://unit42.paloaltonetworks.com/cobalt-strike-metadata-encryption-decryption/.
[21] C. Navarrete, D. Sangvikar, A. Guan, Y. Fu, Y. Jia, and S. Shibiraj, "Cobalt Strike Analysis and Tutorial: How Malleable C2 Profiles Make Cobalt Strike Difficult to Detect," March 2022, https://unit42.paloaltonetworks.com/cobalt-strike-malleable-c2-profile/.
[22] N. Känzig, R. Meier, L. Gambazzi, V. Lenders, and L. Vanbever, "Machine Learning-based Detection of C&C Channels with a Focus on the Locked Shields Cyber Defense Exercise," in *Proceedings of the 11th International Conference on Cyber Conflict (CyCon 2019)*. IEEE, May 2019, pp. 1–19.
[23] CCDCOE, "Locked Shields," https://ccdcoe.org/exercises/locked-shields/.
[24] V. van der Eijk and C. Schuijt, "Detecting Cobalt Strike Beacons in NetFlow Data," https://rp.os3.nl/2019-2020/p29/report.pdf.
[25] F. M. Ramos and X. Wang, "A Machine Learning Based Approach to Detect Stealthy Cobalt Strike C&C Activities from Encrypted Network Traffic," in *Proceedings of the 5th International Conference on Machine Learning for Networking (MLN'2022)*, November 2022.