A Machine Learning Approach for RDP-based Lateral Movement Detection

Tim Bai, Haibo Bian, Abbas Abou Daya, Mohammad A. Salahuddin, Noura Limam and Raouf Boutaba
David R. Cheriton School of Computer Science, University of Waterloo, Ontario, Canada
{tim.bai, haibo.bian, aaboudaya, mohammad.salahuddin, n2limam, rboutaba}@uwaterloo.ca

***Abstract*—Detecting cyber threats has been an on-going research endeavor. In this era, advanced persistent threats (APTs) can incur significant cost for organizations and businesses. The ultimate goal of cyber security is to thwart attackers from achieving their malicious intent, whether it is credential stealing, infrastructure takeover, or program sabotage. Every cyber attack goes through several stages before its termination. Lateral movement (LM) is one of those stages which is of particular importance. Remote Desktop Protocol (RDP) is a method used in LM to successfully authenticate to an unauthorized host that leaves footprints on both host and network logs. In this paper, we propose to detect evidence of LM with an anomaly detection approach that leverages Windows RDP event logs. We evaluate various supervised machine learning (ML) techniques for classifying RDP sessions with high precision and recall. We also compare the performance of our proposed approach to a state-of-the-art approach and demonstrate that our ML model outperforms in classifying RDP sessions in Windows event logs.**

I. Introduction

Advanced persistent threat (APT) is one of the most prominent cyber attacks with the potential to cause significant damage to various organizations and businesses. It is a stealthy attack in which attackers gain unauthorized access to a network for a long period of time. Stuxnet [1], an infamous attack on critical infrastructure, devastated Iran's nuclear program. According to Kaspersky Lab [2], a backdoor program called Carbanak, caused a billion dollar in cumulative losses for a financial institution. Furthermore, more than 80 million social security numbers were siphoned from Anthem, a big health insurance company, which was only detected after nine months [3]. In this attack, Mivast malware [3] masqueraded as a VPN software, serving as a backdoor for the attacker.

Most secured systems maintain a strong boundary between the internet and the intranet, thus attackers choose targets that have access to hosts behind the network security functions (*e.g.*, firewalls, intrusion prevention systems, *etc.*). It is difficult for attackers to launch attacks against protected assets that reside in the intranet. Thus, an attacker usually leverages social engineering techniques (*e.g.*, phishing, pretexting, baiting, *etc.*) to trick insiders into executing malicious code or surrendering credentials. This allows the attacker to gain access to the victim's computer and gradually explore for valuable information by exploiting vulnerabilities of other intranet entities. This is commonly known as Lateral Movement (LM).

Machine Learning (ML) techniques have been widely used for anomaly-based APT detection [4], since ML is an ideal tool to automatically establish the normal behavior of a system [5]. Remote Desktop Protocol (RDP) is designed by Microsoft to provide remote display and input capabilities, while Remote Desktop Service (RDS) is a native service on Microsoft Windows platform that implements RDP. This service is frequently used by legitimate administrators. However, it is also a major tool used by attackers during LM [6], since discriminating between legitimate and malicious use is challenging. We surveyed nine distinct APT incidents and five of them used RDP during the attack. In this work, we focus on detecting anomalous RDP sessions based on host-based evidence. The primary contributions of this work are:

- We highlight the limitations of two publicly available Windows event log datasets from Los Alamos National Laboratory (LANL) [7], [8]. To overcome limitations, we combine the datasets while preserving their realistic property.
- We propose an anomaly-based approach for detecting malicious RDP sessions. We explore different feature sets and evaluate various supervised ML techniques for classifying RDP sessions in Windows event logs.
- We compare the performance of our proposed approach to a state-of-the-art method [9], and demonstrate that our ML model outperforms in the classification of RDP sessions.

The rest of the paper is organized as follows. Section II presents the related works on detecting APT. In Section III, we analyze the dataset employed in this work. We delineate the evaluation results in Section IV. Section V provides a brief summary of our work and instigates future research directions.

II. Related Works

Ussath *et al.* [6] analyze twenty two different APT reports and summarize different techniques employed in corresponding APT campaigns. They also implement a user behavior simulation system [10] to generate user activity logs for Windows platforms. They leverage feed-forward neural networks and recurrent neural networks to identify malicious log events. However, the dataset generated by their simulation system is based on hypothetical assumptions. For example, some of their ML features, such as longitude and latitude of the user, are impractical for most real-world scenarios.

Kaiafas *et al.* [9] successfully employ an ensemble of classifiers for detecting malicious events in the LANL dataset [7]. However, the authors are oblivious to the biased nature of the dataset. Based on our analysis (*cf.*, Section III), all red team events in the dataset originate from four unique hosts. This implies that the ML classifiers will be biased to the source host feature (employed in [9]) in training and inference. We highlight this limitation in Section IV.

Siadati *et al.* [11] implement a system that extracts logon patterns for anomaly detection. They propose a novel pattern mining algorithm that is scalable for large dataset. Their system consist of two components, an exact matching classifier and a pattern matching classifier. While the exact matching classifier is prone to logon history poisoning, the pattern matching classifier complements it by matching a logon to all possible combination of attributes that describe it. While the authors propose host-based anomaly detection that leverages pattern matching, the focus of our work is to harness ML techniques for anomaly detection.

III. Dataset

The dataset plays a crucial role in the success of ML. However, Windows event log datasets that represent real user behavior are fairly limited. Most publicly available datasets, such as [12], [13], facilitate network-based intrusion detection. In contrast, host event logs contain sensitive information limiting their distribution by organizations [10]. To overcome this limitation, researchers (*e.g.*, [10]) often simulate user and attacker behavior to generate synthetic datasets. However, datasets generated using this approach are purely based on hypothetical assumptions, and may not depict real-world user behavior. Therefore, to preserve the realism of user behavior, we leverage and combine two real datasets from LANL, namely *comprehensive* [7] and *unified* [8] datasets.

A. Comprehensive Events Dataset

The comprehensive dataset [7] spans 58 days, and consist of activities generated from 12,425 users and 17,684 computers. The dataset is divided into five different logs, namely authentication, process, flow, DNS and red team logs. The red team log contains a subset of events from the authentication log, which are generated from red team activities (*e.g.*, compromise events). Hence the red team log provides the ground truth for ML. In this work, we leverage the authentication and red team logs for detecting malicious RDP sessions. However, based on the dataset description and our observations, there are limitations in the authentication log:

- The number of red team events is very small, accounting for less than 0.0001% of the total events and only appear in certain time intervals.
- There are no logoff events, making it impossible to deduce certain crucial features, such as the logon session duration.
- The timestamp is obfuscated in UNIX time epoch. As a result, it is difficult to categorize events into days, which could be a discriminating feature to identify abnormal usage.
- A large number of RDP logon events have the same source and destination host, which is beyond reason.

B. Unified Events Dataset

The unified dataset [8] is collected within LANL over a 90 day interval. Unlike the previous dataset, this dataset provides comprehensive and detailed Windows event logs including the missing logoff events. Although the timestamps in this dataset are also obfuscated, events are already divided into days. However, the primary limitation of this dataset is the lack of red team activities, *i.e.*, this dataset only contains benign user activities. Furthermore, the source host is missing in some 4624 LogonType 10 events and all 4625 LogonType 10 events. The 4624 event records all successful logons and event 4625 records logon failures with reason, while type 10 in both events indicate that RDP is used for remote login. Both of these events are crucial for tracking (malicious) RDP sessions [14].

C. Combining Datasets

Both datasets have limitations according to their authors [15] and our observations. Hence, we decided to inject red team events from the comprehensive dataset [7] into the unified dataset [8]. Since these two datasets were collected within the same organization, we do not lose the properties and patterns of attack events. However, these two datasets are obfuscated with different hash functions and cannot be simply merged. Also, recall that the red team events originate from only four unique hosts. Indeed we could have mapped these four source hosts into a larger group of hosts in our new dataset to avoid any bias in the ML classifier. However, we did not choose this approach to preserve the authenticity of the attacks.

Instead, we proceed as follows. Let R be the collection of red team logon events from the comprehensive dataset and B the collection of benign RDP logon events extracted from the unified dataset. For each event $e_i \in R$, we map the source host Src_i to a randomly selected unique source host Src_j from an event $e_j \in B$. We further map the user name and destination host tuple $\{Usr_i, Dst_i\}$ of e_i, to a randomly selected unique tuple $\{Usr_k, Dst_k\}$ from an event $e_k \in B$. After mapping, we insert, in chronological order, the modified red team events e'_i into the set B, labeled as *malicious*. There are no changes needed for timestamp, since the the unified dataset already spans the red team event's time interval.

We extract a total of 222,692 events with ID 4624, 4625 and 4634, and authentication type 10. We discard all 4625 (failed logon) events and 4624 events with missing source host. After removing the invalid data entries and extracting relevant features (*cf.*, Section IV), we end up with 56,837 events. The significant reduction in datapoints come from combining logon events (ID 4624) with their corresponding logoff events (ID 4634) into RDP session events with well-defined session length. Benign logon events from the unified dataset with no corresponding logoff events are omitted as well. It is important to note that the injected red team authentication events only contain logon events (ID 4624) but no logoff events (ID 4634). Hence, this hampers the computation of malicious RDP session's duration. To this end, we generate a session duration for each red team event from a normal distribution $\mathcal{N}(\mu, \sigma^2)$, where μ and σ are the mean and standard deviation, respectively, computed from all benign RDP session's duration. Though a random distribution may be more reasonable, as attacks can last for any duration, we assume that attackers have similar behavior (session durations) to benign users.

IV. Experimentation

A. Environment Setup

1) Hardware: The data analysis, visualization and pre-processing are performed on a cluster of four nodes, each featuring an Intel(R) Xeon(R) E3-1230 v3 3.30GHz CPU and 16GB RAM. Nodes are interconnected with 10Gbps Ethernet. Model training and validation are performed on an Amazon AWS EC2 t3.medium instance.

2) Software: A Logstash instance is deployed to ingest the dataset into an ElasticSearch [16] cluster, and Kibana is used for data visualization. For data pre-processing, a variety of Python packages, including Numpy [17], Scipy [18] and Pandas [19] are employed. The ML models are developed in Python with Scikit-learn [20] and Keras [21] libraries.

B. Evaluation

1) Metrics: We define malicious RDP sessions as positive subjects and use a variety of performance metrics to evaluate the different ML techniques. The accuracy indicates the percentage of sessions that are correctly classified. Whereas, precision is the percentage of sessions that have been identified as malicious are indeed malicious. A higher precision implies a higher confidence in the true nature of the sessions flagged as malicious (*i.e.*, lower false positives). On the other hand, recall is the percentage of malicious sessions that have been correctly identified. A higher recall implies a higher confidence that malicious sessions were not missed (*i.e.*, lower false negatives).

We also present the F_1 score, a harmonic mean of precision and recall. This metric provides the aggregate performance of a classifier. Though accuracy also depicts the overall performance, F_1 score is more reliable when the dataset is imbalanced (in our case, more benign than malicious datapoints). To illustrate the performance of the classifiers at different classification thresholds, we leverage the Precision-Recall (PR) curve. We also use the Average Precision (AP) score, which is the weighted average of precision at each decision threshold, and estimates the area under the PR curve.

2) ML Techniques: We employ various ML techniques to evaluate our approach for detecting malicious RDP sessions. We leverage Logistic Regression (LR), a classic regression model that is known to capture the relationship between variables. Similarly, we employ Gaussian-NB (GNB), a probabilistic classifier based on Bayes' theorem, without specifying any prior distribution. We also evaluate the Decision Tree (DT) classifier with a maximum depth of three and criterion entropy. Furthermore, we evaluate Random Forest (RF) and LogitBoost (LB), which are ensemble methods built on top of DT. RF tends to solve the over-fitting problem in DT, while LB combines a set of weak learners to construct a strong learner.

3) Results: To validate our ML models, we first employ k-fold cross validation ($k = 10$). Recall that all the attacks in the employed dataset originate from four unique source hosts. Therefore, a classifier that uses the source host feature may tend to predict all events with these source hosts as malicious, leading to a bias in classification. Therefore, we remove from our feature set those features that cause this bias, namely user, source host and destination host. Table I depicts the result after removing these features. The DT algorithms have both high precision and recall, with LB using DT regressor outperforming all other classifiers. This is primarily because LB classifiers are designed to boost the performance of existing classifiers [22]. Even though the probabilistic GNB classifier under performs the DT family of classifiers, it outperforms LR.

TABLE I
RDP SESSION CLASSIFICATION (*user*, *src* AND *dst* FEATURES REMOVED)

Classifier	Accuracy	Precision	Recall	F_1
LR	98.50%	11.34%	1.87%	0.321
DT	99.90%	99.04%	93.58%	0.962
GNB	99.60%	87.31%	82.11%	0.846
RF	99.94%	99.59%	96.13%	0.978
LB	99.99%	99.87%	99.47%	0.997

The authors in [9] improve the performance of their stand-alone classifiers by consolidating them using ensemble ML. We employ a similar approach with Majority Voting (MV) algorithm and a careful selection of the classifiers. The best performing ensemble has minor improvements in precision, but results in a much lower recall than stand-alone LB. Evidently, MV is unable to boost the performance of the stand-alone classifiers, as shown in Table II. Therefore, we use LB (with 100 estimators) as *Our Model*.

TABLE II
MAJORITY VOTING FOR RDP SESSION CLASSIFICATION USING SELECTIVE CLASSIFIERS (*user*, *src*, AND *dst* FEATURES REMOVED)

Classifiers	Accuracy	Precision	Recall	F_1
GNB, RF, LB	99.95%	99.87%	96.26%	0.980
GNB, RF, DT	99.91%	99.58%	93.32%	0.964
GNB, LB, DT	99.91%	99.73%	93.32%	0.964
RF, LB, DT	99.95%	99.73%	96.13%	0.979

C. Comparative Analysis

We compare our stand-alone LB classifier with Kaiafas *et al.* [9]. We implement their approach and evaluate the corresponding model on our dataset. During feature extraction, we omit the *geometric distribution* feature, since all the failure events are filtered out due to missing source host in the dataset. As shown in Table III, with all available features, the recall of Kaiafas's model is slightly lower than our model (first two rows without *). Though their precision is better, the F_1 score indicates an overall performance drop in comparison to our model. Furthermore, after the removal of user name, source host and destination host features from both models, Kaiafas's model has a significant drop in recall from 98.67% to 90.66% (last two rows with *). On the other hand, the training time (TT) of Kaiafas's model is about 80% higher than our model. This can primarily be attributed to the larger number of features and construction of extra classifiers. Therefore, our model outperforms the state-of-the-art in RDP session

TABLE III
RDP SESSION CLASSIFICATION USING STAND-ALONE LB VS. [9]

Classifier	Accuracy	Precision	Recall	F_1	TT (s)
Our Model	99.99%	99.87%	99.73%	0.998	11.28
Kaiafas *et al.*	99.98%	100.00%	98.67%	0.993	20.48
*Our Model	99.98%	99.87%	99.47%	0.992	10.53
*Kaiafas *et al.*	99.88%	100.00%	90.66%	0.951	18.19

* = Model validation without *user*, *src* and *dst* features

classification, and promises suitability for online host-based anomaly detection.

Finally, to further evaluate the models against zero-day threats, we perform a robustness test. We split the dataset into training (75%) and testing (25%). While the training set contain attacks originating from three different sources, the testing set contains an additional source that does not appear in the training set. In Fig. 1, we present the PR curve, which illustrates the trade off between precision and recall at different thresholds. As evident, our model's PR curve is very close to a perfect classifier and yields an AP score of 0.95. This asserts the robustness of our model to detect threats from new (unseen) attack sources. However, it is unfeasible to plot a PR curve for a MV classifier, such as Kaiafas's model. A MV classifier depends on decisions made by several classifiers and a single chosen threshold across classifiers in not suitable. Therefore, we compare the overall robustness of the two classifiers using the F_1 score. While our model obtains a high F_1 score of 0.914, Kaiafas's model scores a low F_1 score of 0.675.

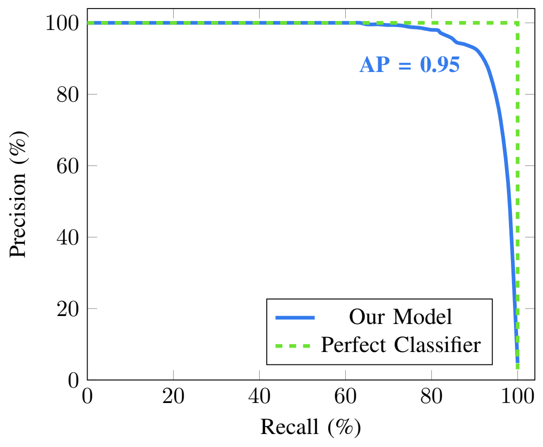

Fig. 1. Precision-Recall curve of our model

V. Conclusion and Future Work

We leverage Windows event logs for anomaly-based detection of malicious RDP sessions. With the identified shortcomings of two public datasets, we synthesize a combined dataset that remains faithful to the attack models. Using the combined dataset, we identify relevant features, and leverage DT, RF, FNN, GNB, and LB to detect malicious log entries. After evaluating the classifiers in both stand-alone and ensemble settings, we chose LB as the best model with respect to accuracy, recall and precision in RDP session classification. LB shows promising results and outperforms a state-of-the-art model [9] in recall and training time for detecting malicious Windows RDP sessions.

We intend to explore further model parameter tuning, test with other uncharted features and incorporate unsupervised learning as a pre-processor to pruning datapoints. In addition, our approach can benefit from online learning. Training ML models from scratch can be computationally intensive, time consuming, and prohibitive. Therefore, it is crucial to retrain ML models as new data becomes available, thus accommodating ML models boundary changes after deployment. We will also extend our approach to other session-based protocols, as suitable datasets become available.

Acknowledgments

This work is supported in part by the Royal Bank of Canada, and the NSERC CRD Grant No. 530335.

References

[1] S. Karnouskos, "Stuxnet worm impact on industrial cyber-physical system security," in *Proceedings of IEEE IECON*, 2011.
[2] Kaspersky Lab, "Carbanak apt: The great bank robbery," 2015, accessed: 2019-03-03. [Online]. Available: https://securelist.com/the-great-bank-robbery-the-carbanak-apt/68732/
[3] J. DiMaggio, "The black vine cyberespionage group," Aug 2015, accessed: 2019-02-28. [Online]. Available: https://securelist.com/the-great-bank-robbery-the-carbanak-apt/68732/
[4] R. Boutaba *et al.*, "A comprehensive survey on machine learning for networking: evolution, applications and research opportunities," *Journal of Internet Services and Applications*, vol. 9, no. 1, 2018.
[5] S. Ayoubi *et al.*, "Machine learning for cognitive network management," *IEEE Communications Magazine*, vol. 56, no. 1, pp. 158–165, 2018.
[6] M. Ussath *et al.*, "Advanced persistent threats: Behind the scenes," in *Proceedings of CISS*, 2016.
[7] A. D. Kent, "Comprehensive, Multi-Source Cyber-Security Events," Los Alamos National Laboratory, 2015.
[8] M. J. M. Turcotte *et al.*, *Unified Host and Network Data Set*. World Scientific, Nov 2018, ch. Chapter 1.
[9] G. Kaiafas *et al.*, "Detecting malicious authentication events trustfully," in *Proceedings of IEEE NOMS*, April 2018.
[10] M. Ussath *et al.*, "Identifying suspicious user behavior with neural networks," in *Proceeding of IEEE CSCloud*, June 2017.
[11] H. Siadati and N. Memon, "Detecting structurally anomalous logins within enterprise networks," in *Proceedings of the ACM CCS*, 2017, pp. 1273–1284.
[12] S. García *et al.*, "An empirical comparison of botnet detection methods," *Comput. Secur.*, vol. 45, 2014.
[13] E. B. B. Samani *et al.*, "Towards effective feature selection in machine learning-based botnet detection approaches," *IEEE CNS*, 2014.
[14] JPCERT Coordination Center, "Detecting lateral movement through tracking event logs," Dec 2017, accessed: 2019-02-26. [Online]. Available: https://www.jpcert.or.jp/english/pub/sr/ir_research.html
[15] A. D. Kent, "Proceedings of Cybersecurity Data Sources for Dynamic Network Research," in *Dynamic Networks in Cybersecurity*, Jun. 2015.
[16] C. Gormley and Z. Tong, *Elasticsearch: The Definitive Guide*, 1st ed. O'Reilly Media, Inc., 2015.
[17] T. E. Oliphant, *Guide to NumPy*, 2nd ed. USA: CreateSpace Independent Publishing Platform, 2015.
[18] E. Jones *et al.*, "SciPy: Open source scientific tools for Python," 2001–, accessed Mar 2019. [Online]. Available: http://www.scipy.org/
[19] W. McKinney, "Data structures for statistical computing in python," in *Proceedings of SciPy*, 2010.
[20] Pedregosa *et al.*, "Scikit-learn: Machine learning in python," *J. Mach. Learn. Res.*, vol. 12, Nov. 2011.
[21] F. Chollet *et al.*, "Keras," https://keras.io, 2015.
[22] J. Friedman *et al.*, "Additive logistic regression: a statistical view of boosting," *Annals of Statistics*, vol. 28, p. 2000, 1998.