DANTE: Odd ones out - Deep learning using system logs to detect Insider Threat

Qicheng Ma
IDEA
Rensselaer Polytechnic Institute
Troy, New York

Nidhi Rastogi
IDEA
Rensselaer Polytechnic Institute
Troy, New York

***Abstract*—Insider threat is one of the most pernicious threat vector to organizations across the world due to the elevated level of trust and access that an insider is afforded. This type of threat can stem from both malicious and negligent users. In this paper, we propose a novel approach that uses system logs to detect insider behavior using RNNs models. Ground truth is modeled using DANTE and used as baseline for anomalous behavior. For this, system logs are modeled as a natural language sequence and patterns are extracted from these sequences. We create workflows of sequences of actions that follow a natural language logic and control flow. These flows are assigned various categories of behaviors - malignant or benign. Any deviation from these sequences indicates the presence of a threat. We further classify this threats into one of the five categories provided in the CERT Insider Threat dataset. Through experimental evaluation, we show that the proposed model, which is still work-in-progress, can achieve approximately 93% prediction accuracy.**

I. Introduction

Insider threat continues to top the security threat vector list year after year. The federal agency has increased its insider threat related spending and are on target to spend almost $1 billion in fiscal 2020 [1]. When malicious activities are performed by insiders - employees, contractors, temporary hires, interns, ex-employees, no amount of firewalls, multi-factor authentication can protect against their actions. An insider already has access to internal organizational information and have the motive and means to sabotage the system for their own personal or professional gains. And the impact by an insider threat can be far more detrimental than any other threat vector. They can cause compromise of classified information, billions of dollars of lost revenues over a period of time, harms reputation and many other forms of intangible damages.

Covid-19 has further exacerbated the situation. Due to the remote nature of employee work environment, virtual private networks (VPNs) have replaced LANs, and so have the security posture around the employee devices. It has therefore, become easier to allow social engineering techniques to imposter an employee and gain access to internal company resources without raising an alarm.

This research presents a novel anomaly detection based approach that uses system logs to gather malicious insider behavior. Since log data is available in every networked organization, they can be used in tandem to detect other intrusion activities as well. In this research, we propose a natural language based model [2] that uses offline logs as sequences to create sequences and rule-based patterns. The process of log generation follows rules, logic and control flows like a structured form of natural language. Therefore, we propose the use of Long Short-Term Memory (LSTM) [3], which can learn long-range state over a sequence of information. Our model captures sequences of activities across various log data to create workflow models for each separate task. These workflows create a sequence that can be used to identify anomalous patterns and then be classified into one of the five categories of insider threats. Experiments on our LSTM based model show a 99% accuracy in detecting insider behavior across all malicious categories. A point to note here is that since this is a work in progress, at this time, this research can detect only known types of anomalies from the log entries.

II. Related Work

Anomaly detection is frequently used to frame an insider threat detection problem due to the nature of the dataset - it is usually skewed and captures more benign user behavior. Any outlier from a threshold indicates unacceptable behavior, which can be further investigated to determine the category of threat. Neural Networks have also been used to classify and predict presence of an Insider Threat [4].

Recurrent Neural Networks (RNNs) [5] allow persistence of information due to the presence of loops in their networks. This property connects previous occurrence of information to the present one. An example of successful usage of RNNs are language models that predict next word in a sentence, noted by keyboards in smartphones. Long Short Term Memory networks (LSTMs) [3] are a common type of artificial RNNs that can learn long-term information because of feedback connections in their networks. They differ from standard RNNs as they have four neural network layers instead of one in the repeating module.

RNNs have been used to model different types of intrusions, such as real-time insider threat [6], off-line intrusion detection [7], etc. This is mainly due to the promising results from using LSTM models on this category of problems in the cybersecurity domain. Other than neural networks, in [8], researchers use supervised learning that incorporates feedback for insider threat detection.

In [9], the authors use LSTM approach for detecting insider threat alibi the dataset is structured. An interesting aspect of

this research is that it assigns scores to each user's online session (sub-optimal approach), but requires input of an expert to identify suspicious user sessions during training the dataset. Tabish et al. [10] use Hidden Markov Models (HMM) to learn benign user behavior. Any deviation from this implies an insider threat activity. While this requires fewer log entries to train the model, this approach requires expert in the middle to differentiate normal activity from malicious. In addition to this, the variations in normal activity due to external factors have to be accounted for every time the model is trained.

III. Threat Model

Our model creates intricate log sequences for individual users. The assumption here is that the log data creation, and storage is secure and protected from any adversarial attack. We also assume that despite the access levels provided to employees, where some of them own administrator level access (system admin, their supervisor, IT managers) to servers and code that generated log files, as individuals they cannot modify the log files or the source code. The attack that we consider in this research is the insider threat who has access to various systems at different privilege levels.

IV. Data

Availability of a good dataset is crucial for monitoring tasks, and a successful analysis. Likewise, Insider threat detection also benefits from the collection of a dataset that can enable predictions using machine learning models. In line with this, we choose to use the CERT dataset [11], which is a decent size collection of synthetic insider threat test data sets. It contains both benign data from normal users as well as data from malicious actors [1]. Also, for this research, we use this dataset since it contains a rich collection of log files from emails, http connections, emails, device connections, user logon as well as psychometric tests (based on test results of OCEAN[2]).
The dataset had multiple releases (r1-r6), and generally speaking, the later releases included a superset of the earlier releases. The insider threat detection methodology was proposed and evaluated on one of the six versions of this dataset. The dataset also includes the answer key file which comprises details of the malicious activity included in each dataset, including descriptions of the scenarios enacted and the identifiers of the synthetic users involved.
The r5.2 was chosen as the primary dataset for our research since, unlike the rest of the data sets, this one contained all attack scenarios described in the dataset. These are described below:

1) Personnel worked after hours, used removable drive and uploaded data to `wikileaks.org`. Left the organization soon after.
2) Personnel visited job websites, and solicited employment from competitor(s). Abnormal increase in usage of removable drive for data transfer. Left the organization soon after.
3) Disgruntled system administrator, who downloaded a keylogger, used a removable drive for data transfer to their supervisor's computer. Collected keylogs were used to log in as their supervisor. Alarming mass emails were sent out that caused panic in the organization. Left workplace right after this incident.
4) Over a period of three months, personnel frequently logged into other user's computer. Searched and forwarded files on personal email address.
5) Personnel laid off. Uploaded documents to `Dropbox`.

A. About the Dataset

Since we primarily use revision 5.2 of the dataset, the following description applies only to this version. We specifically employ the following data sources as input to our detection models - web history (`http.csv`), email log (`email.csv`), device access (`device.csv`), logon (`logon.csv`), file access (`file.csv`), and user data in the LDAP folder. While details of each of the log files can be gained from pulling the data from the link provided earlier, below we describe in detail some of the features that we captured before pre-processing:

1) `LDAP` - employee name, user ID, email, role, Projects, business unit, functional unit, department, team, supervisor.
2) `http.csv` - event ID (20 character long hash), time stamp, user ID, PC ID, web-page URL, activity ("WWW Visit", "WWW Download", "WWW Upload".), content.
3) `email.csv` - event ID, time stamp, user ID, PC ID, to, cc, bcc, from, activity (Send, Receive or View), size, attachments, content.s
4) `device.csv` - event ID, time stamp, user ID, PC ID, file-tree, activity (connect or disconnect)
5) `logon.csv` - event ID, time stamp, user ID, PC ID, activity (logon or logoff activity).
6) `file.csv` - event ID, time stamp, user ID, PC ID, filename, activity (open, copy, write, delete), to removable media, from removable media, content (hexadecimal encoded file header and a list of content keywords).

B. Data pre-processing

The available data sources described in previous section, while in tabulated format, require processing to make them suitable for ingesting in our prediction model. For example, user specific information across different sources was collated based on their relationship with other users (e.g. supervisor or not), logon or logoff into their own personal computers and other computers, their designation (administrator with access to multiple computers), hours of activity, sequence of activities across various logs, etc. For a given user - computer IDs that were accessed, domain category of the website accessed (described next), email domain category - internal or external to the organization and so on.

In our model, we also identify acceptable working hours based on rules for both weekdays, weekends, and holidays. Websites were categorised based on their primary activity - job hunting websites, hacktivist websites, and file sharing

[1] https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=508099
[2] https://en.wikipedia.org/wiki/Big_Five_personality_traits

websites. This allows normalization and quantification of data points which would otherwise be a challenge to utilize in modeling. Each user's activity is tracked by user activity, and not based on daily, weekly or monthly activity. This allows detection at a much granular level, and also real-time detection of anomalous behavior when we prepare our model for real-time detection at a later stage.

V. DANTE

In this section, we describe the proposed model DANTE.

A. Log Parsing

We parse the log entries into structured representation for each user, and across all log entries associated with that user in a given time frame. Prior work shows that from each log entry, a message type can be extracted as the log key [12], [13]. When logs are generated from a source code, print statements are used to write statements that match an event, along with values that describe that event. For example, for a print code - `printf("Logged into %f.", PC_ID")`, the value of PC_ID is ***e*** and the statement *"Logged into *."* is the key, ***k***. See Table I for more examples.

Log Message	key	values
t_1 **Logoff from PC-003**	k_1	[t_1, PC-003]
t_2 **Delete file-GdY68**	k_2	[t_2, file-GdY68]
t_3 **Access file-HJd84**	k_3	[t_3, file-HJd84]

TABLE I
EXEMPLAR ENTRIES FROM LOG FILES

B. Model Description

The architecture is shown in Figure 1 with three main components : log key-value generation from log data, log parsing, workflow construction, and anomaly detection using DANTE.

C. Workflows

In this section, we describe the creation of log key sequences that form workflows of each user per day. Log printing source code has a finite number of print statements, and therefore, can print a finite number of log keys. The set $K = k_1,k_2,...,k_n$ is that finite set of unique log keys. In Figure 2, we show a few examples of how the key-value pairs are generated and used to create a workflow.

Once log entries are parsed into log keys, the log key sequence reflects an execution path that leads to that particular execution order of the log print statements. We can model anomaly detection in a log key sequence (workflow) as a multi- class classification problem, where each distinct log key defines a class. We train our model as a multi-class classifier over recent context. The input is a history of recent log keys, and the output is a probability distribution over the n log keys from K, representing the probability that the next log key in the sequence is a key $k_i \epsilon$K.

D. Model Training

- Training data for Insider Threat detection model are log entries from normal log. Each log entry is parsed to a log key and a parameter value vector. The log key sequence parsed from a training log file is used by LSTM model to train a log key anomaly detection model, and to construct system execution workflow models for diagnosis purposes. For each distinct key k, our model also trains and maintains a model for detecting anomalies as reflected by these metric values, trained by the parameter value vector sequence of k.

E. Model Testing

The problem of assigning probabilities to sequences of words drawn from a fixed vocabulary is widely studied by the natural language processing (NLP) community [14]. In our case, each log key can be viewed as a word taken from the vocabulary K. The typical language modeling approach for assigning probabilities to arbitrarily long sequences is the N-gram model. The intuition is that a particular word in a sequence is only influenced by its recent predecessors rather than the entire history.

F. LSTM

An LSTM-based model can encode complex patterns, such as log sequences, and maintain long-range state over a sequence. Therefore, we use LSTM models to detect anomalous behavior in work flows created using log-key sequences. An input consists of a window w of h log keys, and an output is the log key value that comes right after w. We use the categorical cross-entropy loss for training. After training is done, we can predict the output for an input $(w = m_{t-h},...,m_{t-1})$ using a layer of h LSTM blocks. Each log key in w feeds into a corresponding LSTM block in this layer.

Anomaly detection - For anomaly detection from log keys, the input is a sequence of log keys of length h from recent history, and the output is a probability distribution of all possible log key values. The training set from benign users forms the baseline. The difference between a prediction and an observed parameter value vector is measured by the mean square error (MSE). At deployment, if the error between a prediction and an ob- served value vector is within a high-level of confidence interval of the above Gaussian distribution, the parameter value vector of the incoming log entry is considered normal, and is considered abnormal otherwise.

VI. EXPERIMENTS AND RESULTS

We deployed our model on google colab[3] to run experiments. We separated the log entries into three sets - one for training, one for testing and one for validation. During testing, DANTE measures the Gaussian distribution of MSE between the proposed key and the actual key in a given workflow. If it is within an acceptable confidence interval of the MSE, the next key is accepted, else considered anomalous.

[3]https://colab.research.google.com/

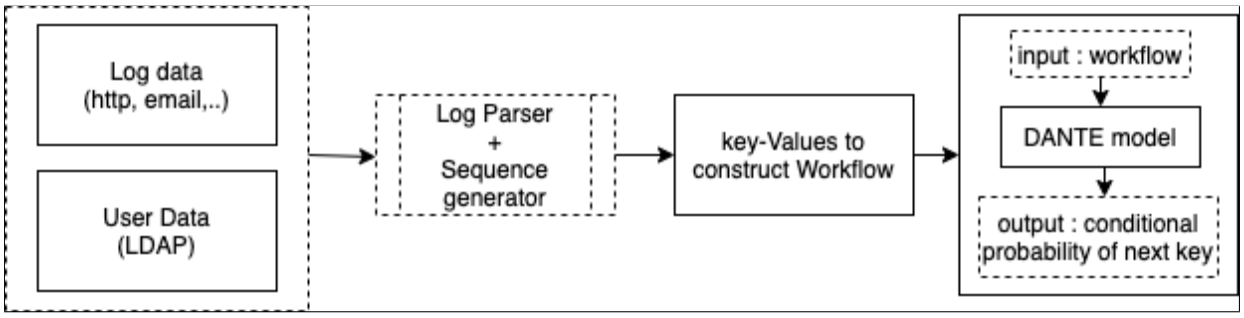

Fig. 1. Block diagram showing various DANTE components.

user ID	date	key	values
AAB1302	2010-10-03	[62, 62, 62, 62, 62, 62, 63, 63, 63]	['[(9, 0.23760533), (34, 0.22418228), (59, 0.5...
AAB1762	2010-05-26	[63, 63, 63, 62, 62, 62, 63, 63, 63, 62, 62, 6...	['[(53, 0.4396325), (65, 0.18133092), (66, 0.3...
AAB1762	2011-05-23	[62, 62, 62, 63, 63, 63, 47, 47, 47, 62, 62, 6...	['[(37, 0.9789852)]', '[(10, 0.17000163), (35,...
AAC0904	2011-05-25	[62, 62, 62, 62, 62, 62, 62, 62, 62, 63, 63, 6...	['[(53, 0.15477093), (81, 0.8273755)]', '[(20,...
AAC1033	2010-09-30	[62, 62, 62, 63, 63, 63, 63, 63, 63, 63, 63, 6...	['[(33, 0.64795625), (70, 0.31273544), (87, 0....
AAC1033	2011-05-10	[63, 63, 63, 63, 63, 63, 63, 63, 63, 63, 63, 6...	['[(7, 0.96128404), (17, 0.014326925)]', '[(7,...

Fig. 2. A snippet of key value generation. For each user, in a given day, each key is associated with a value and the probability of it's occurrence.

The github code has been made available as open source[4] and can be downloaded and used in tandem with the CERT dataset r5.2.

VII. Results

In the Table II, we summarize the number of users identified as insider threats and the number of logs classified as threat scenarios described in Section IV. Note, four out of five cases were classified.

	No. of Users	No. of Logs
Total	2000	79856704
Benign	1901	79846376
S-1	29	486
S-2	30	6477
S-3	10	184
S-4	30	3181

TABLE II
SUMMARY OF PREDICTED INSIDERS, LOG ENTRIES FOR EACH SCENARIO.

The model shows an accuracy of 93.29% with an average loss of 0.230025 for 500 epochs.

Figure 3 shows the detection accuracy of the model. Also, in Figure 5, we show the average loss as training dataset increases. The accuracy is approximately 93% and the average loss is less than .23.

[4]https://github.com/aiforsec/InsiderThreat

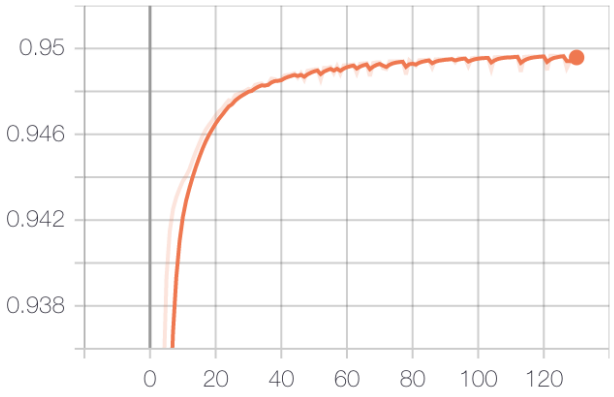

Fig. 3. Evaluation of DANTE - Epoch vs Accuracy (approx. 93%)

VIII. Discussion and Future Work

This is a work in progress and is part of a larger road map. In this paper, we present the first part of DANTE model where logs are used to create a sequence of events for each user for a given day. Since log creation follows control logic and flow, it is assumed that this flow can be dealt in the same way as natural language models are. However, there are several challenges that need to be addressed before we can publish a full-paper. Anomalous behavior assumes that the baseline has been fully modeled, and hence anything outside the threshold will be considered as potential threat. Like most

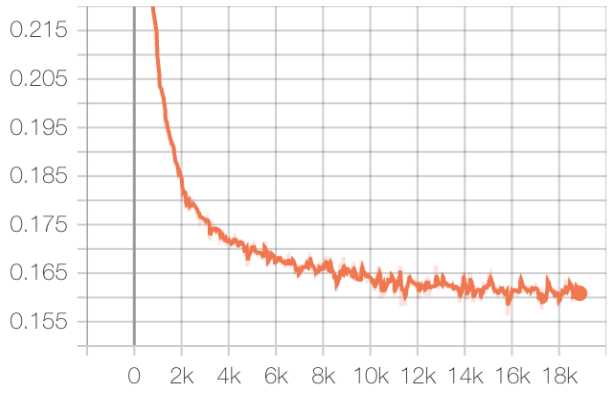

Fig. 4. Evaluation of DANTE - Training dataset vs Average Loss (approx 0.25).

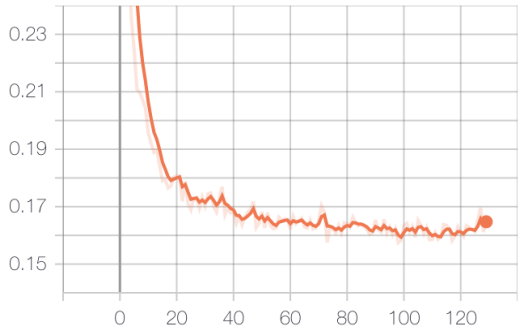

Fig. 5. Evaluation of DANTE - Epoch count vs Average Loss.

anomaly detection models, this approach suffers from high false positives and therefore, false alarms. We will be exploring clustering techniques that identify various behaviours of the users without a-priori knowledge of benign user behavior. The second problem we are trying to solve is the problem of loops within a key-values, where the same value might reoccur. A third problem we're looking into is the granularity of log key-value pair. The current approach uses the smallest level of log sequence, and this might lead to overlooking new patterns by new users who join the organization with new roles and responsibilities, and hence a new work flow. Some of these challenges include the assumption of unchanged source code that is used to generate logs statements. An insider knows no bounds, and with the right privileges, they can poison both the model as well as the data that is used to create the baseline (benign behavior) or models for various malignant behavior.

REFERENCES

[1] L. Criste, "Insider threat market to top $1 billion in fiscal 2020: This is," Aug 2020. [Online]. Available: https://about.bgov.com/news/insider-threat-market-to-top-1-billion-in-fiscal-2020-this-is/

[2] M. Du, F. Li, G. Zheng, and V. Srikumar, "Deeplog: Anomaly detection and diagnosis from system logs through deep learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1285–1298.

[3] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[4] E. Lopez and K. Sartipi, "Detecting the insider threat with long short term memory (lstm) neural networks," *arXiv preprint arXiv:2007.11956*, 2020.

[5] T. Mikolov, S. Kombrink, L. Burget, J. Černockỳ, and S. Khudanpur, "Extensions of recurrent neural network language model," in *2011 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2011, pp. 5528–5531.

[6] J. Lu and R. K. Wong, "Insider threat detection with long short-term memory," in *Proceedings of the Australasian Computer Science Week Multiconference*, 2019, pp. 1–10.

[7] R. C. Staudemeyer, "Applying long short-term memory recurrent neural networks to intrusion detection," *South African Computer Journal*, vol. 56, no. 1, pp. 136–154, 2015.

[8] D. C. Le and A. N. Zincir-Heywood, "Machine learning based insider threat modelling and detection," in *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. IEEE, 2019, pp. 1–6.

[9] A. Tuor, S. Kaplan, B. Hutchinson, N. Nichols, and S. Robinson, "Deep learning for unsupervised insider threat detection in structured cybersecurity data streams," *arXiv preprint arXiv:1710.00811*, 2017.

[10] T. Rashid, I. Agrafiotis, and J. R. Nurse, "A new take on detecting insider threats: exploring the use of hidden markov models," in *Proceedings of the 8th ACM CCS international workshop on managing insider security threats*, 2016, pp. 47–56.

[11] M. Collins, M. Theis, R. Trzeciak, J. Strozer, and J. Clark, "Common sense guide to mitigating insider threats fifth edition," 2016.

[12] M. Du and F. Li, "Spell: Streaming parsing of system event logs," in *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 2016, pp. 859–864.

[13] J.-G. Lou, Q. Fu, S. Yang, Y. Xu, and J. Li, "Mining invariants from console logs for system problem detection." in *USENIX Annual Technical Conference*, 2010, pp. 1–14.

[14] C. Manning and H. Schutze, *Foundations of statistical natural language processing*. MIT press, 1999.