Advance Persistent Threat Detection Using Long Short Term Memory (LSTM) Neural Networks

P. V. Sai Charan(✉), T. Gireesh Kumar, and P. Mohan Anand

TIFAC-CORE in Cyber Security, Amrita school of Engineering,
Amrita Vishwa Vidyapeetham, Amrita University, Coimbatore, India
pvsaicharan2011@gmail.com, gireeshkumart@gmail.com,
mohananand1997@gmail.com

Abstract. Advance Persistent Threat (APT) is a malware attack on sensitive corporate, banking networks and stays there for a long time undetected. In real time corporate networks, identifying the presence of intruder is a big challenging task to security experts. Recent APT attacks like Carbanak and The Big Bang ringing alarms globally. New methods for data exfiltration and evolving malware techniques are two main reasons for rapid and robust APT evolution. In this paper, we propose a method for APT detection System for real time corporate and banking organizations by using Long Short Term Memory (LSTM) Neural networks in order to analyze huge amount of SIEM (Security Information and Event Management) system event logs.

Keywords: LSTM · APT · Hadoop · Splunk · Hive

1 Introduction

APT is a combination of several sophisticated attacks which are composed by a professional attacker on a specific sensitive organizations. Usually, Security specialists consider this APT as undetected shady RAT (Remote Access Trojan) operations. According to McAffe survey on APT, 83% of these APT attacks are due to out going sessions through TCP (Transmission Control Protocol) port numbers 80 and 443 [1,2]. Some of organizations counter this thing by web proxies which can inspect HTTP traffic i.e port number 80 but 443 is mostly untouched by most of organizations. Added to this, using reverse shell instead of bind shell for injecting commands into the victims machines is very difficult to identify at both network and host level in real time. There are broadly 6 phases in which APT will infect and spread in a particular targeted network [3].

1.1 Phase1 - Reconnaissance

In this particular phase he tries to gather information about targeted organization by using some social engineering techniques. The key objective of attacker

A. K. Somani et al. (Eds.): ICETCE 2019, CCIS 985, pp. 45–54, 2019.
https://doi.org/10.1007/978-981-13-8300-7_5

in this phase is to infiltrate into organizations network by exploiting some vulnerability.

1.2 Phase 2 – Gaining Access

In this particular phase, payload injection occurs into network by directly exploiting remote access backdoor or making the user to click on malicious link in the form of a phising/spam mail etc. According to IBM researchers, attackers are using deceptive and highly targeted attack tools embedded with Artificial Intelligence which reveals its identity to a specific targeted victim [4]. That means the malicious payload of malware will be hidden in a normal day to day applications to avoid detection by most antivirus and malware scanners. Usually, attackers using video conferencing software, file sharing software until it reaches specific victims, who are identified via indicators such as geo location, voice or facial recognition and other system-level features.

1.3 Phase 3 - Lateral Movement

In this phase, attacker tries to move from initial target to specific targeted part of network by exploiting compromised privileges and as well as configuration weakness in that particular network.

1.4 Phase 4 – Gathering Information

In this phase, attacker captures and observes different workflow patterns in that organization and tries to capture them in the form of logging keystrokes, capturing the screen recordings of user workflows of specific intended targets.

1.5 Phase 5 – Data Exfiltration

In this particular phase, attacker ties to push gathered information to external Control and Command Server. In phase also attackers gained a lead than security experts in order to hide exporting sensitive contents to C2C servers bypassing robust firewalls and Intrusion Detection Systems at Host and Network levels as well. For example, attackers bypassed Google, Adobe firewalls in operation Aurora by sending traffic over TCP with a custom encrypted protocol instead of suing SSL [5].

1.6 Phase 6 - Cleaning

In this particular phase, the attacker tries to clean his operations to cover his identity in order to escape from any further legal actions from targeted organization end.

Rapid growth in the design of sophisticated malware tools not only causing a great matter of threat for global IT industry but also defense and banking

industry as well. Recently, the word APT has become the common tool for cyber warfare between countries. Carbank attack in 2015 has literally infected thousands of victims all over the world. This Carbanak APT totally effected almost equal to $1 Billion though various operations like generating bogus accounts and using fake services to collect the money, transferring money to cybercriminals using the SWIFT (Society for Worldwide Interbank Financial Telecommunication) network etc. Kaspersky dig into very same issue and published a detailed report which gives detailed analysis about attack pattern [6]. But recently, The Big Bang APT in 2018 came up with much more robust and sophisticated way which mainly focused on targeting the Palestinian Authority [7]. This APT malware contains a number of modules that perform certain functionalities such as taking screenshots, obtaining a list of files, retrieving system information, restarting the system and self-deletion. The malware will fetch additional modules from the Command and Control server if it finds something of interest. Because of merge of cybercrime and APT, emerging malware techniques, hybrid methods of data exfiltration like pass by hash, fragmentation of bigger APT groups, embedding Artificial Intelligence in APT framework to reach specific intended target, all these kind of scenarios making APT attacks great matter of concern globally. In order to deal with target specified next generation intelligent APT attack we propose a novel method for APT detection mechanism by using Long Short Term Memory (LSTM) Neural network.

This paper is organized as follows. Section 1 details about APT overview and Phases of APT, followed by the challenges that corporate network face with APT. Section 2 explains Related works and problems involved in current methodologies. Problem statement and Proposed system for APT detection using LSTM Neural networks explained under Sect. 3. Implementation and results are detailed in Sect. 4. Conclusion and future work is discussed under Sect. 5.

2 Related Works

Research work on various APT detection methods have been evolving rapidly from last 5 years. Many hybrid techniques have been developed by integrating both network level detection and behavior based abnormality identification techniques. Fabio Pierazzi has proposed a novel method for APT detection analysis of high volumes of network traffic from different network probes [8]. By assigning scores for different traffic flows, they identified the APT behavior at data exfiltration phase. Similarly, Guodongzhao proposed a method in which a dedicated system in network will detect the APT malware based on malicious DNS [9]. It uses a combination of anomaly based detection and signature based detection to find the malicious APT C2C domains. But in the real environments time it's very difficult to rely on DNS based APT detection methods because latest APT malwares are not using malicious flux service or DGA (Domain Generation Algorithm) domains. On the other hand, there is plenty of research work going on to detect APT kind of persistent hidden malware by observing abnormal user behavior by auditing large amount of semi structure server log files in real time corporate networks [10,11].

Recently, Artur Rot has proposed a multi-layered approach to detect APT in which the seven-layer model based on OSI creates an environment where each layer can't defend an APT attack on its own, but their combination gives us fruitful results [12]. Although many layers exist in this model, Sandboxing layer plays a key role in detection of APT. In one way sandboxing seems efficient in APT detection, its have its own disadvantages as well. Sandboxing technique is still vulnerable to the zero-day vulnerabilities and the high probability for the inclusion of new evasion measures. Along with this new generation APT are Anti-Sandbox resistant in nature. So, traditional sandboxing techniques may not work in future in the case of APT [13]. Added to this, there are many big players in market like QRadar IBM, Q1 Labs Qradar, and NetIQ Security Manager SIEM tools which can able to deal with different varieties of log files in real time APT detection [14,15]. On the other side of coin Roman Jasek has proposed a novel method for detection of APT by using Honeypot [16,17]. In this method, Honeypot agents installed to trap the malicious users and also these technique will be very efficient to observe the attack patterns of malicious users especially in long run which is exactly suitable for APT attack scenarios.

3 Proposed Work

Although many traditional and hybrid methods are available in detection of APT malware, still number of APT attacks increasing rapidly at global level. Attackers made APT sandboxing resistant so that traditional sand boxing techniques may not work with robust new generation APT detection. Next generation AI based APT techniques are very hard to identify by any of existing APT detection methods. In this paper, we propose a novel method for APT detection in real time by using Long Short Term Neural Networks which is a varient of Recurrent Neural network (RNN) with added storage capabilities.

In this paper, we have collected huge system event log files from APT infected machines by installing splunk forwarder at host level [18]. In the splunk forwarder we specify the IP address for splunk server machine so that all those logs generated at the client side will be pushed to server machine. Usually, in real time these log files are unstructured and very huge in its size. In the initial stage we have loaded this log file to HDFS for further processing. We have used open source big data tool named HIVE to extract event codes, corresponding message along with the time stamp [19,20].

3.1 Extracted Event Code Structure from HIVE Tool

"4/10/18 6:22:23:00 PM, 7036, The WinHTTP Web Proxy Auto-Discovery Service service entered the stopped state"

Sample extracted event code structure is given below. After processing at HIVE tool, the output log file consists of time stamp followed by event code generated at that particular time and the system event message corresponding to that particular event code [21] (Table 1).

Table 1. Observed APT pattern from event code sequence

S.no	Event code	Event code meaning
1	4648	This event is generated when a process attempts to logon
2	4624	This event is generated when a logon session is created
3	4672	Special privileges assigned to new logon SecurityPrivilege TakeOwnershipPrivilege LoadDriverPrivilege BackupPrivilege RestorePrivilege DebugPrivilege SystemEnvironmentPrivilege ImpersonatePrivilege
4	7036	This event indicates that the firewall has been moved to stopped state
5	1014	This event Name resolution for some external C&C server
6	7036	This event indicates WinHTTP Web Proxy Auto-Discovery Service service entered the stopped state
7	7036	This event indicates the Application Experience service entered the stopped state

Once log processing completed at HIVE level, a series of dependent event at the Reducer level. We Have identified a event pattern (series of events) which exactly replicate APT behavior in the real time. The series of Event codes which indicates that behavior of APT are actually Interdependent on each other. That means if we need to detect APT and raise a alarm based on a particular event code then we need to remember all the previous six event codes. In order to address this problem we have used LSTM Neural Networks to efficiently train these interdependent event pattern in our case [22].

3.2 Long Short Term Memory (LSTM) Neural Network in APT Detection

LSTM is a special kind of Recurrent Neural Networks (RNN) capable of learning long term dependencies proposed by Hochreiter and Schmidhuber to overcome vanishing gradient problem in standard RNN [23]. LSTM consists of one input layer, one output layer, and one recurrent hidden layer. The hidden layer contains a memory cell with self-connections in order to memorize the temporal state. The two gates named input gate and output gate is used for regulating the information flow through the cell. Constant Error Carousel (CEC) will be considered as the core of memory cell which is a recurrently self-connected linear unit, and the cell state usually represented by the activation of the CEC. Multi-

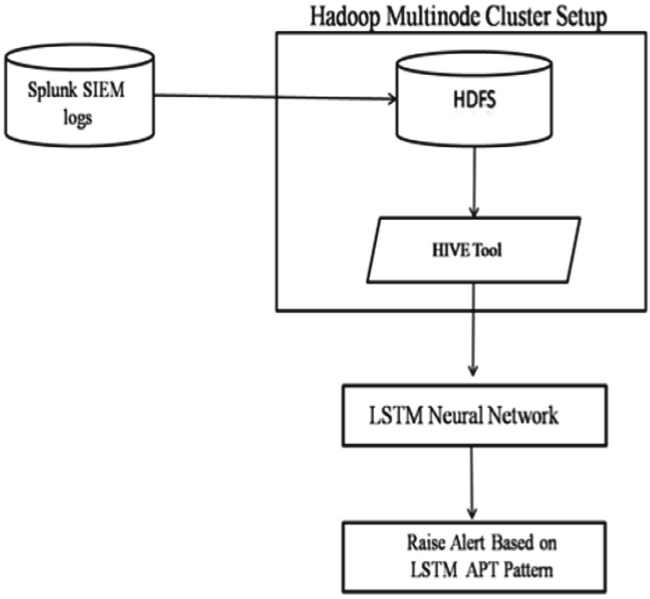

Fig. 1. Proposed system archirtecture

plicative gates can learn to open and close because of this CEC, which indirectly helps LSTM NN to solve the vanishing gradient problem (Fig. 1).

The input is denoted as $x = (x_1, x_2, \ldots, x_T)$, and the output denoted as $y = (y_1, y_2, \ldots, y_T)$ and T will be the time taken for identifying exact APT event code pattern. In our APT detection case, x can be considered as series of event code data that we extracted fom the Splunk machine and y is the estimated time to detect exact APT event code pattern. The objective of LSTM NN is to detect exact APT event code pattern based on previous event codes without specifying information about how many steps should be traced back. In real time implementation, we will get the exact APT detection time by iteratively calculating the following the equations at the background level.

$$f_t = \sigma(W_{fx}x_t + W_{fm}m_{t-1} + W_{fc}C_{t-1} + b_f) \tag{1}$$

$$C_t = f_t \bullet C_{t-1} + I_t \bullet g(W_{cx}X_t + W_{cm}m_{t-1} + b) \tag{2}$$

$$O_t = \sigma(W_{ox}x_t + W_{om}mt - 1 + W_{oc}C_t + b_0) \tag{3}$$

$$m_t = O_t \bullet h(C_t) \tag{4}$$

$$y_t = W_{ym}m_t + b_y \tag{5}$$

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{6}$$

Each block of LSTM contains an input gate, an output gate, and a forget gate. i_t, o_t, f_t are outputs of those three gates are respectively. c_t and m_t represents the

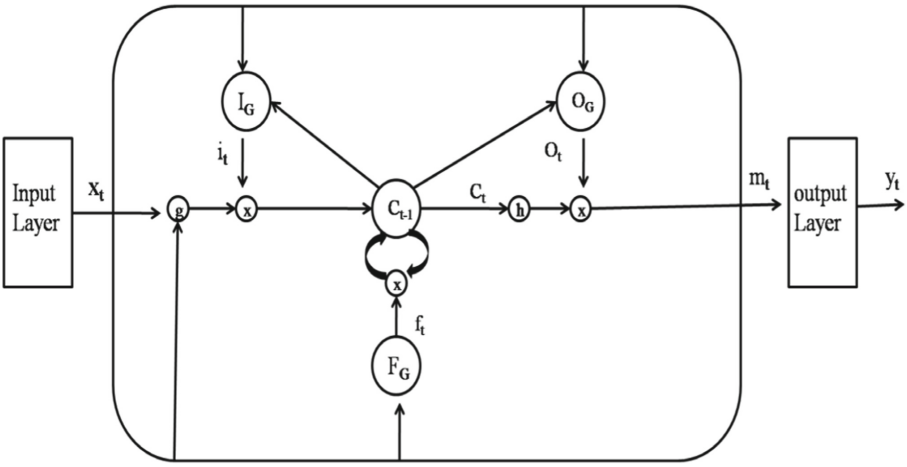

Fig. 2. LSTM cell archirtecture [23]

activation vectors and memory block for each cell. To build connection between the cell block, input and output layers bias vectors b and weight matrices W will be used normally.

• indicates the scalar product of two vectors, $\sigma(x)$ indicates logistic sigmoid function is denoted and centered logistic sigmoid function with range $[-2, 2]$ is denoted with g(x), centered logistic sigmoid function with range $[-1, 1]$ denoted with h(x) in Fig. 2. For detailed execution steps of LSTM cell architecture please refer Xiaolei work [24]. In our case, we need to remember all previous 6 states information while predicting the APT from a huge amount event codes which are continously generating in the real time. We have trained LSTM neural network to predict APT event code pattern. At the output cell all the errors are truncated, where these errors can flow back which makes error tends decay exponentially. In this way LSTM network in our case deals with long term dependencies of system event codes in order to detect exact APT pattern.

4 Implementation and Results

The processed output from the HIVE engine will be divided as 40% for training remaining 60% for the testing LSTM neural network. In our case, we have used open source machine learning library named Tensorflow which is developed by Google for performing dataflow programming of LSTM neural networks to predict the APT pattern from System Event code logs [25]. The output from the HIVE phase is segregated as two individual datasets i.e sample-1 and sample-2 respectivly. Sample-1 dataset is splitted as 60:40 ratio for training and testing phases of LSTM netowrk. Similarly, sample-2 dataset is splitted as 70:30 ratio for training and testing phases in our case. We have trained LSTM with these two different datasets and results are tabulated in Table 2.

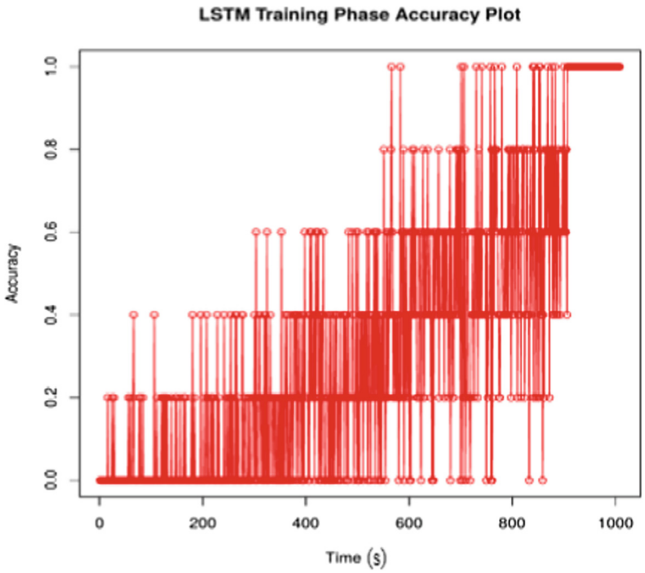

Fig. 3. LSTM network training phase accuracy plot for sample-1 dataset

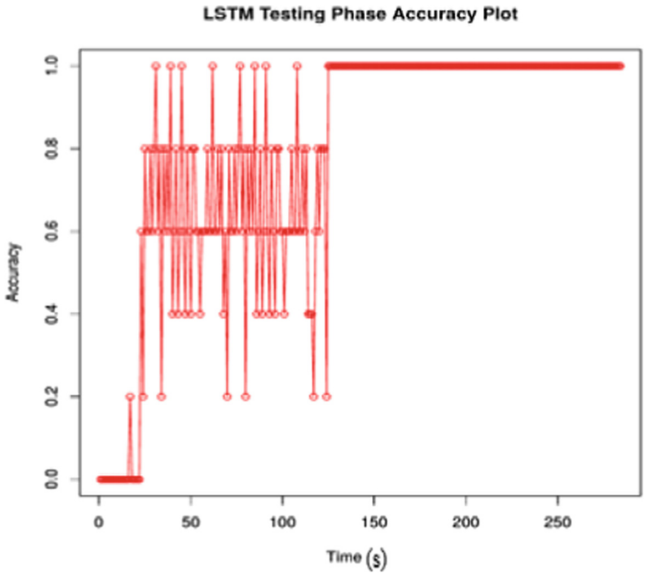

Fig. 4. LSTM network testing phase accuracy plot for sample-1 dataset

In both training and testing phases for LSTM neural network, we can clearly understand the way in which LSTM neural network is identifying the APT event code pattern by observing the accuracy value pattern in the Figs. 3 and 4.

Table 2. Implementation results for different data samples

Sample name	Phase (training/testing)	Percentage of data considered	Time taken for APT detection (sec)
Sample-1	Training	60	900 (∼15 Min)
	Testing	40	129 (∼2.2 Min)
Sample-2	Training	70	1028 (∼17.13 Min)
	Testing	30	117 (∼1.95 Min)

Initially Neural network accuracy is 0 is both the cases but eventually LSTM neurons will start learning and identifying the patterns so that we can see a gradual improvement in accuracy in both of the cases. In training phase, LSTM neural network nearly took 15 min and 17.13 min to identify the pattern for sample-1 as well as sample-2 data sets respectivly. But in the testing phase, for both data sets we can observe a drastic improvement i.e approximately 2 min and 1.95 min to identifying the APT event code pattern which will be highly suitable for the real time APT detection.

5 Conclusion and Future Work

In this paper, we propose a technique to detect APT in the real time with the help of LSTM neural network by analyzing large amount of SIEM system event log files which are collected from APT infected machine. From the Fig. 4. We can clearly identify that LSTM can able to detect the APT pattern with in a span of minutes which we can further optimize using high end processors at the hardware level. As an additional enrichment to this system, we can further train different event code patterns to this LSTM network so that we can build much robust system against next generation malware attacks. Although, many Intrusion detection systems and sandboxing techniques are available in market which operates at host and network layer, they are unable to cope up with the next generation APT techniques which are inbuilt AI components. In order to deal with evolving malware techniques, we can integrate our proposed model as a AI component level for traditional APT detection methods to build next generation intelligent and robust malware detection systems.

References

1. Kaspersky Lab: The Great Bank Robbery: The Carbanak APT (Detailed Investigation Report) (2015). https://securelist.com/the-great-bank-robbery-the-carbanak-apt/6873/
2. McAfee Labs Threats Report, June 2018. https://www.mcafee.com/enterprise/en-us/assets/reports/rp-quarterly-threats-jun-2018.pdf
3. Messaoud, B.I.D., et al.: Advanced persistent threat: new analysis driven by life cycle phases and their challenges. In: International Conference on Advanced Communication Systems and Information Security (ACOSIS). IEEE (2016)

4. DeepLocker: How AI Can Power a Stealthy New Breed of Malware (2018). https://securityintelligence.com/deeplocker-how-ai-can-power-a-stealthy-new-breed-of-malware/
5. Kharitonov, D., Ibatullin, O.: Extended security risks in IP networks. arXiv preprint arXiv:1309.5997 (2013)
6. Kaspersky Security Bulletin (2015). https://securelist.com/kaspersky-security-bulletin-2015-overall-statistics-for-2015/73038/
7. The Big Bang APT (2018). https://research.checkpoint.com/apt-attack-middle-east-big-bang/
8. Marchetti, M., et al.: Analysis of high volumes of network traffic for advanced persistent threat detection. Comput. Netw. **109**, 127–141 (2016)
9. Zhao, G., et al.: Detecting APT malware infections based on malicious DNS and traffic analysis. IEEE Access **3**, 1132–1142 (2015)
10. Kayacik, H.G., et al.: Detecting Anomalous Hypertext Transfer Protocol (HTTP) Events from Semi-Structured Data. U.S. Patent Application No. 15/420,560
11. Sai Charan, P.V.: Abnormal user pattern detection using semi-structured server log file analysis. In: Satapathy, S.C., Bhateja, V., Das, S. (eds.) Smart Intelligent Computing and Applications. SIST, vol. 104, pp. 97–105. Springer, Singapore (2019). https://doi.org/10.1007/978-981-13-1921-1_10
12. Rot, A., Olszewski, B.: Advanced persistent threats attacks in cyberspace. Threats, vulnerabilities, methods of protection. In: 2017 Federated Conference on Computer Science and Information Systems, vol. 13 (2017)
13. Brickell, E.F., et al.: Method of improving computer security through sandboxing. U.S. Patent No. 7,908,653, 15 March 2011
14. IBM QRadar (The Intelligent SIEM). https://www.ibm.com/security/security-intelligence/qradar
15. NetIQ. https://www.netiq.com/de-de/
16. Jasek, R., Kolarik, M., Vymola, T.: APT detection system using honeypots. In: Proceedings of the 13th International Conference on Applied Informatics and Communications (AIC 2013), WSEAS Press (2013)
17. Ali, P.D., Gireesh Kumar, T.: Malware capturing and detection in dionaea honeypot. In: 2017 Innovations in Power and Advanced Computing Technologies (i-PACT). IEEE (2017)
18. Anastasov, I.: DancoDavcev.: SIEM implementation for global and distributed environments. In: 2014 World Congress on Computer Applications and Information Systems (WCCAIS). IEEE (2014)
19. Apache-Hadoop. http://Hadoop.apache.org
20. Apache-Hive. https://hive.apache.org/
21. Armour, D.J., Kalki, J.: Determining computer system usage from logged events. U.S. Patent No. 8,185,353, 22 May 2012
22. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
23. Hochreiter, S., Schmidhuber, J.: LSTM can solve hard long time lag problems. In: Advances in Neural Information Processing Systems (1997)
24. Ma, X., et al.: Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. Transp. Res. Part C: Emerg. Technol. **54**, 187–197 (2015)
25. Tensorflow. https://www.tensorflow.org/