2014 IEEE Security and Privacy Workshops

Asset Risk Scoring in Enterprise Network with Mutually Reinforced Reputation Propagation

Xin Hu, Ting Wang, Marc Ph. Stoecklin, Douglas L. Schales, Jiyong Jang, Reiner Sailer
IBM Research
{huxin,tingwang}@us.ibm.com, mtc@zurich.ibm.com, {schales, jjang, sailer}@us.ibm.com

Abstract—**Cyber security attacks are becoming ever more frequent and sophisticated. Enterprises often deploy several security protection mechanisms, such as anti-virus software, intrusion detection/prevention systems, and firewalls, to protect their critical assets against emerging threats. Unfortunately, these protection systems are typically "noisy", e.g., regularly generating thousands of alerts every day. Plagued by false positives and irrelevant events, it is often neither practical nor cost-effective to analyze and respond to every single alert. The main challenge faced by enterprises is to extract important information from the plethora of alerts and to infer potential risks to their critical assets. A better understanding of risks will facilitate effective resource allocation and prioritization of further investigation. In this paper, we present MUSE, a system that analyzes a large number of alerts and derives risk scores by correlating diverse entities in an enterprise network. Instead of considering a risk as an isolated and static property, MUSE models the dynamics of a risk based on the mutual reinforcement principle. We evaluate MUSE with real-world network traces and alerts from a large enterprise network, and demonstrate its efficacy in risk assessment and flexibility in incorporating a wide variety of data sets.**

I. Introduction

Mitigating and defending against ever more frequent and sophisticated cyber attacks are often top priorities for enterprises. To this end, a plethora of detection and prevention solutions have been developed and deployed, including anti-virus software, intrusion detection/prevention systems (IDS/IPS), blacklists, firewalls, and so on. With these state-of-the-art technologies capturing various types of security threats, one would expect that they are very effective in detecting and preventing attacks. In reality, however, the effectiveness of these systems often fall short. The increasingly diversified types of cyber attacks, coupled with increasing collection of applications, hardware configurations, and network equipments, have made the enterprise environment extremely "noisy". For example, IDS/IPS systems regularly generate over 10,000 alerts every day. Majority of them turn out to be false positives. Even true alerts are often triggered by low level of threats such as brute-force password guessing and SQL injection attempts. Although the suspicious nature of these events warrants the reports by IPS/IDS systems, they often only made the situation more noisy.

Digging into the haystack of alerts to find clues to actual threats is a daunting task that is very expensive, if not impossible, through manual inspection. As a result, most IPS/IDS alerts are often stored in a database merely for forensic purposes. These alerts are investigated only after significant incidents are discovered or critical assets are already severely damaged, e.g., security breaches, and data leakage. On the other hand, even some true positive alerts (e.g., device compromise, virus infection on a user's computer) are often too voluminous to become security analysts' top priority. This is because these alerts are considered as low level of threats and pose far less risks to the enterprises comparing with severer attacks, such as server compromise or sensitive data leakage. Evidently, more effective solutions require better understanding and ranking of potential risks to enterprise assets so that resources can be prioritized and allocated according to the severity of the risks.

In a typical enterprise environment, as shown in Figure 1, there are different sets of entities: servers, devices, users, credentials and (high-value) assets (e.g., databases, business processes). The connections between entities represent their intuitive relationships; for example, a user may own multiple devices. We note that the reputation of entities provide valuable indicators into its associated risks, and are important factors to rank security incidents, e.g., alerts, and anomalies. More importantly, the reputation of an entity and the risk it may produce are not confined to the entity. In fact, multiple entities are often tied together in a mutually reinforcing relationship.

In this paper, we propose MUSE (**M**utually-reinforced **U**nsupervised **S**coring for **E**nterprise risk), a risk analysis framework that analyzes a large amount of various security alerts and computes the reputation of diverse entities based on domain knowledge and interactions between entities. Specifically, MUSE exploits the interactions as reflected in composite bipartite graphs where each pair of entity types (e.g., a user and a device) can form one bipartite graph. MUSE then applies an iterative propagation algorithm to utilize the mutual reinforcement between the connected entities and to derive their reputation and risk score simultaneously. Finally, with the refined risk scores, MUSE is able to provide useful information such as ranking of low reputation entities and potential risks to critical assets, allowing security analysts to make an informed decision as to how resources can be prioritized for further investigation. MUSE will also provide greater visibility into the set of alerts that are responsible for an entity's low reputation, offering insights into the root cause of cyber attacks. Our contributions include: 1) a mutual reinforcement framework to analyze the reputation and the risk of diverse entities in an enterprise network, 2) a scalable propagation algorithm to exploit the networking structures and identify potential risky entities that may be overlooked by a discrete risk score, 3) a highly flexible system that can incorporate data sources in multiple domains, and 4) evaluations with real network traces from a large enterprise to verify the efficacy of MUSE.

II. Risk and Reputation in a Multi-entity Environment

A. Problem Formulation

In a typical enterprise environment, there are multiple sets of connected entities as shown in Figure 1. Specifically, we consider five distinct types of entities: users $\mathcal{U}$, devices $\mathcal{D}$, credentials $\mathcal{C}$, high value assets $\mathcal{A}$, and external servers $\mathcal{S}$. These entities are often related in pairwise many-to-many relationships. For example, a device can access multiple external

DOI 10.1109/SPW.2014.18

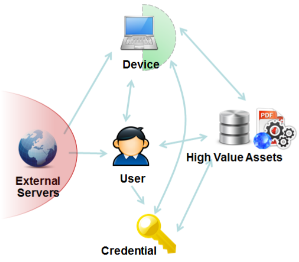

Fig. 1. Entities in a typical enterprise network

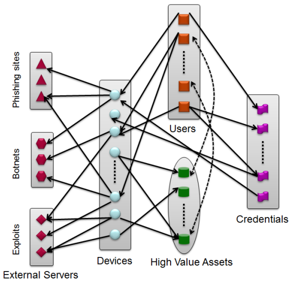

Fig. 2. Network of interactions among multiple entities

servers. A user may own several devices, while one device (e.g., server clusters) can be used by multiple users.

We model the inter-connection between entities as a composite bipartite graph $G = (V, E)$, schematically shown in Figure 2. In G, vertices $V = \{\mathcal{U}, \mathcal{D}, \mathcal{C}, \mathcal{A}, \mathcal{S}\}$ represent entities, and edges E of bipartite graphs represent their relationships: $E = \{\mathcal{M}_{\mathcal{DS}}, \mathcal{M}_{\mathcal{DU}}, \mathcal{M}_{\mathcal{DA}}, \mathcal{M}_{\mathcal{UC}}, \mathcal{M}_{\mathcal{DC}}\}$ where $\mathcal{M}_{\mathcal{DS}}$ is the $|\mathcal{D}|$-by-$|\mathcal{S}|$ matrix containing all the pairwise edges, i.e., $\mathcal{M}_{\mathcal{DS}}(i,j) > 0$ if there is an edge between device d_i and external server s_j. The value of $\mathcal{M}_{\mathcal{DS}}(i,j)$ denotes the edge weight derived from the characteristics of the relationship, such as the number of connections, the number of bytes transmitted, duration, and so on. Similarly, $\mathcal{M}_{\mathcal{DS}}$, $\mathcal{M}_{\mathcal{DU}}$, $\mathcal{M}_{\mathcal{DA}}$, $\mathcal{M}_{\mathcal{UC}}$, and $\mathcal{M}_{\mathcal{DC}}$ are the matrices of the pairwise edges representing the association of their respective entities.

Next we define the risk and the reputation of entities more precisely. We treat each entity as a random variable X with a binary class label $X = \{x_R, x_{NR}\}$ assigned to it. Here x_R is a risky (or bad) label, and x_{NR} is a non-risky (or good) label. A probability distribution P is defined over this binary class, where $P(x_R)$ is the probability of being risky and $P(x_{NR})$ is the probability of being non-risky. By definition, the sum of $P(x_R)$ and $P(x_{NR})$ is 1. We use this probabilistic definition because it allows a natural mapping between $P(x_{NR})$ and the general concept of reputation, i.e., an entity with a high probability of being good (or non-risky) is expected to have high reputation. In addition, it accepts different types of entities to incorporate specific domain knowledge into the reputation computation considering their respective characteristics, e.g.,

External Server Reputation $\boldsymbol{p_s}$ $= P_s(x_{NR})$ indicates the server's probability of being malicious and infecting the clients connecting to it. A low reputation p_s indicates high probability of being malicious. **Device Reputation $\boldsymbol{p_d}$** $= P_d(x_{NR})$ represents the probability that a device may have been infected or compromised. **User Reputation $\boldsymbol{p_u}$** $= P_u(x_{NR})$ indicates how suspiciously a user behave, e.g., an unauthorized access to sensitive data. **Credential Reputation $\boldsymbol{p_c}$** $= P_c(x_{NR})$ denotes the probability that a credential may have been leaked to the adversaries and thus making any servers associated with the credential vulnerable. **High Value Asset Reputation $\boldsymbol{p_a}$** $= P_a(x_{NR})$ denotes the asset's probability of being risky, such as unauthorized accesses, data extrusion, and so on.

Since there is a natural correlation between reputation and risk (e.g., less reputable entities generally pose high risks), we define an entity's risk as $P(x_R) = (1 - P(x_{NR}))$ weighted by the importance of the entity, such that a high value asset will raise a high risk even with a small decline in its reputation.

With these definitions, the goal of MUSE is to aggregate large amounts of security alerts, determine the reputation of each entity by exploiting structural relationships among entities, and finally output the rankings of risky entities as candidates for further investigation. In the next section, we will describe the mutual reinforcement principle [1] that underlies MUSE.

B. Mutual Reinforcement Principle

Intuitively, one can see that entities' reputation and risk are not separated; instead, they are closely correlated and dependent. While interacting with each other, an entity's reputation can impact on the risk associated with its neighbors, and at the same time, the entity's risk can be dependent on the reputation of its neighbors. For example, a device is likely to be of low reputation 1) if the websites it frequently visits are listed in blacklists and are considered as suspicious/malicious, 2) if the users using the device have bad reputation, and 3) if the credentials used to log into the device have high risks of being compromised, leaked, or even used by an unauthorized user. Similarly, a credential's risk of being exposed will increase if it has been used by a less reputable user and/or on a device yielding suspicious behavior patterns. Similarly, a user will have low reputation if the user owns several low-reputation devices and credentials. Last but not least, a high value asset or the sensitive data stored in internal servers is likely to be under a significant risk if it is accessed by multiple low-reputation devices. We describe these mutually dependent relationships more formally in our multi-layer mutual reinforcement framework, using the following set of equations governing the server reputation p_s, device reputation p_d, user reputation p_u, credential reputation p_c, and high-value asset reputation p_a.

$$\begin{aligned}
p_d &\propto \omega_{ds} \sum_{d \sim s} m_{ds} p_s + \omega_{du} \sum_{d \sim u} m_{du} p_u + \omega_{dc} \sum_{d \sim c} m_{dc} p_c \\
p_u &\propto \omega_{du} \sum_{d \sim u} m_{du} p_d + \omega_{uc} \sum_{u \sim c} m_{uc} p_c \\
p_c &\propto \omega_{uc} \sum_{u \sim c} m_{uc} p_u + \omega_{dc} \sum_{d \sim c} m_{dc} p_d \\
p_a &\propto \omega_{da} \sum_{d \sim a} m_{da} p_d + \omega_{ua} \sum_{u \sim a} m_{ua} p_u + \omega_{ca} \sum_{c \sim a} m_{ca} p_c ,
\end{aligned}$$

where $d \sim s$, $d \sim u$, $\ldots$ represent edges connecting device d with server s and user u, $\ldots$, $\propto$ means "proportional to", ω_{ij} indicates the weights associated with edges and reputation types and m_{ij} is the value in the connectivity matrices. We exploit this mutual reinforcement principle in the bipartite graph network to simultaneously estimate the reputation and the risk using the propagation algorithm described in the next section.

C. Reputation Propagation Algorithm

Due to the tight correlation and connection between entities in an enterprise network, we employ the principle of belief propagation (BP) [12] on the large composite bipartite graph G to exploit the link structure and efficiently compute the reputations for all entities. Belief propagation is a message passing algorithm on general graphs and has been widely used in many graph inference and labeling tasks [2] such as social network analysis, fraud detection, and computer vision.

At the high level, the algorithm infers the properties of a node (in our case, the reputation of an entity) in the graph based on two sources of information: 1) the prior knowledge

$\psi(x_i, x_j)$	$x_i = x_{NR}$	$x_i = x_R$
$x_j = x_{NR}$	$0.5 + \epsilon$	$0.5 - \epsilon$
$x_j = x_R$	$0.5 - \epsilon$	$0.5 + \epsilon$

TABLE I. EDGE POTENTIAL FUNCTION

about the node itself, and 2) information about the neighbor nodes. The inference is achieved by iteratively passing messages between all pairs of nodes n_i and n_j. Let $m_{i,j}$ denotes the "message" sent from i to j. The message represents i's influence on j's reputation. One could view it as if i, with a certain probability of being risky, passes some "risk" to j. Additionally, the prior knowledge about i (e.g., importance of the assets, and a user's anomalous behavior) is expressed by *node potential function* $\phi(i)$ which plays a role in determining the magnitude of the influence passed from i to j. In details, edge $e_{i,j}$ is associated with message $m_{i,j}$ (and $m_{j,i}$ if the message passing is bi-directional). The outgoing message from i to neighbor j is updated at each iteration based on the incoming messages from i's the other neighbors and node potential $\phi(i)$ as follows.

$$m_{i,j}(x_j) \leftarrow \sum_{x_i \in \{x_R, x_{NR}\}} \phi_i(x_i)\psi_{ij}(x_i, x_j) \prod_{k \in N(i) \backslash j} m_{k,i}(x_i),$$

where $N(i)$ is the set of i's neighbors, and $\psi_{i,j}$ is the *edge potential* which is a transformation function defined on the edge between i and j to convert a node's incoming messages into its outgoing messages. Edge potential also controls how much influence can be passed to the receiving nodes, depending on the properties of the connections between i and j (e.g., the number of connections, and volume of traffic). $\psi(x_i, x_j)$ is typically set according to the transition matrix shown in Table I, which indicates that a low reputation entity (e.g. a less reputable user) is more likely to be associated with low reputation neighbors (e.g. compromised devices). The algorithm runs iteratively and stops when the entire network is converged with some threshold T, i.e., the change of any $m_{i,j}$ is smaller than T, or a maximum number of iterations is done. Convergence is not theoretically guaranteed for general graphs; however the algorithm often does converge for real-world graphs in practice. At the end of the propagation procedure, each entity's reputation is determined by the updated messages and normalization constant k.

$$p(x_i) = k\phi_i(x_i) \prod_{j \in N(i)} m_{j,i}(x_i); \qquad x_i \in \{x_R, x_{NR}\}$$

D. Incorporating Domain Knowledge

The main challenge of adapting a BP algorithm is to determine its parameters, in particular, the parameters in the node potential and the edge potential function. In this section, we briefly discuss how we leverage the available data sources in a typical enterprise network and incorporate specific domain knowledge (unique to each entity type) to infer the parameters.

Characteristics of External Servers $\mathcal{S}$: We develop an IDS system that leverages several external blacklists to inspect all the HTTP traffic. It flags different types of suspicious web servers to which internal devices try to connect. We classify suspicious servers into the following five types. *1) Spam websites* are flagged by external spam blacklists like Spamhaus. *2) Malware websites* host malicious software including virus, spyware, ransomware, and other unwanted programs. *3) Phishing websites* pretend to be popular websites, such as bank websites to lure unsuspecting users to obtain their login credentials. Recently, spear phishing attacks, which exploit personal information about the target victims to increase the probability of success, become one of the major threats to enterprises. Because of its potential to cause severe damage, we assign a high risky value to its node potential. *4) Exploit websites* host exploit toolkits, such as Blackhole and Flashpack, which are designed to exploit vulnerabilities of the victims' web browsers and install malware on victims' machines. *5) Botnet C&C servers* are connected with bot programs to command instructions, update bot programs, or to extrude confidential information. If an internal device makes an attempt to connect to any known botnet C&C servers, the device is likely to be compromised. In addition to blacklists (e.g., Zeus Tracker), we also design models to detect fast fluxing and domain name generation botnets based on their distinct DNS request patterns. Using the categorization of suspicious servers, we determine initial node potential values according to the severity of their categories. We assign $(\phi(x_R), \phi(x_{NR})) = (0.95, 0.05)$ for the high-risk types, such as botnets and exploit servers; while we assign $(0.75, 0.25)$ and $(0.6, 0.4)$ for the medium- and low-risk types, respectively.

Characteristics of Internal Entities $\mathcal{D}, \mathcal{U}, \mathcal{C}, \mathcal{A}$**:** For internal entities, e.g., devices, users, credentials, and assets, rich information can be obtained from the internal asset management systems and IPS/IDS systems. Available information include device's status (e.g., OS version, patch level), device behavior anomalies (e.g., scanning), suspicious user activities (e.g., illegal accesses to sensitive data, multiple failed login attempts), and credential anomalies (e.g., unauthorized accesses). We adjust node potential for the internal entities based on the information by assigning a severity score to each type of suspicious activities exemplified above. Since entity i can be flagged multiple times for several different types of suspicious behaviors, we first sum the severities of all suspicious activities to get total severity S_i. To avoid being over-shadowed by a few outliers, we transform the severity using sigmoid function $P_i = \frac{1}{1+\exp(-S_i)}$. The node potential for i is then calculated as $(\phi(x_R), \phi(x_{NR})) = (P_i, 1 - P_i)$. Another benefit of using the sigmoid function is that if no alerts have been reported for an entity, its initial node potential will automatically be $(0.5, 0.5)$ implying that no prior information exist for the particular entity.

Although the parameters in MUSE require some level of tuning and manual assignment, it is valuable to security analysts in several aspects. First, the output of MUSE is the ranking of high-risk entities whose absolute risk values are less meaningful. As long as the parameters are assigned based on reasonable estimation of the severities of different types of alerts, MUSE will provide useful information to help analysts prioritize their investigation. Second, MUSE offers flexibility to incorporate diverse entity types and can be easily adapted for a wide variety of other domains. On the other hand, it is possible to automatically learn the appropriate parameter values through machine learning techniques provided that proper labeled training sets are available, which is our plan for future exploration.

III. EVALUATION

A. Data Sets

We evaluated MUSE with data sets collected from multiple data sources in a large enterprise network for over one week. The data sources included DNS messages from local DNS servers, network flows from edge routers, proxy logs, IPS alerts, and HTTP session headers (for categorization of websites). The volume of the raw data per day was about 200 GB. The resulting graph consists of 11,790 nodes and 44,624 edges. The number of different entity types[1] are shown in Table II.

Server					Device	User	Assets
Spam	Malware	Phishing	Exploit	Botnet			
5500	124	10	26	16	3823	2191	100

TABLE II. NUMBER OF DIFFERENT ENTITIES

B. Experiment Results

We applied MUSE to the graph, and our algorithm converged at the 5th iteration. We manually inspected top ranked entities (i.e., with higher $P(x_R)$) in each entity type, and confirmed they were indeed suspicious entities. Due to the space constraint, we discuss one example of user reputation among our findings. We selected 5 top risky users based on the output of MUSE. Figure 3 shows their risk values at each iteration. Note that all the users started with neutral score $(0.5, 0.5)$, meaning that these users had not been flagged by anomalous behaviors. However, due to their interaction with low reputation neighbors, their associated risks increased. Further investigation showed that user332755 owned 5 devices which made 56 times of connections to spam websites, 4 times of connections to malware websites, and 2 times of connections to exploit websites during our monitoring period. user332755 inherited low reputation from his neighbors including the user's devices, causing his risk to quickly rise to the top.

We also measured the running time of MUSE at each iteration against the size of the graph in terms of the number of edges. As shown in Figure 4, the running time quadratically increased, which can be a bottleneck to handle large graphs. However, belief propagation can be distributed and executed in parallel, providing scalability to process large graphs [10].

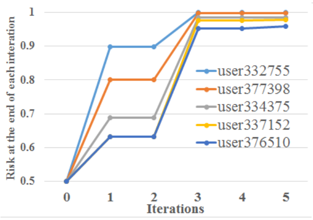

Fig. 3. Risk scores of top 5 risky users at the end of each iteration

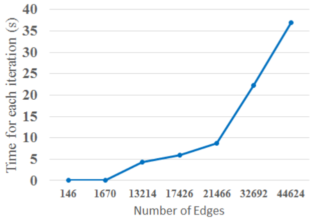

Fig. 4. Running time for each iteration with varying sizes of graphs

IV. Related Work

With the cyber threats rapidly evolving towards large-scale and multi-channel attacks, security becomes crucial for organizations of varying types and sizes. Many traditional intrusion detection and anomaly detection methods focused on a single entity and applied rule-based approaches [3]. They were often too noisy to be useful in practice [11]. Our work is inspired by the prominent research in the social network area that used a link structure to infer knowledge about the network properties. Previous work demonstrated that social structure was valuable to find authoritative nodes [6], to infer individual identities [5], to combat web spam [4], and to detect security fraud [8]. Among various graph mining algorithms, the belief propagation algorithm [12] has been successfully applied in many domains, e.g., detecting fraud [9], accounting irregularities [7], and malicious software [2]. For example, NetProbe [9] applied a BP algorithm to the eBay user graph to identify subgraphs of fraudsters and accomplices.

V. Conclusion

In this paper, we proposed MUSE, a framework to systematically quantify and rank risks in an enterprise network. MUSE aggregated alerts generated by traditional IPS/IDS on multiple data sources, and leveraged the link structure among entities to infer their reputation. The key advantage of MUSE was that it derived the risk of each entity not only by considering its own characteristics but also by incorporating the influence from its neighbors. This allowed MUSE to pinpoint a high-risk entity based on its interaction with low reputation neighbors, even if the entity itself was benign. By providing risk rankings, MUSE helps security analysts to make an informed decision on allocation of resources and prioritization of further investigation to develop proper defense mechanisms at an early stage.

Acknowledgment

This research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defense and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the author(s) and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defense or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

[1] J. Bian, Y. Liu, D. Zhou, E. Agichtein, and H. Zha. Learning to recognize reliable users and content in social media with coupled mutual reinforcement. In *Proceedings of WWW '09*, 2009.

[2] D. H. Chau, C. Nachenberg, J. Wilhelm, A. Wright, and C. Faloutsos. Polonium: Tera-scale graph mining and inference for malware detection. In *SIAM International Conference on Data Mining*, 2011.

[3] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Maciá-Fernández, and E. Vázquez. Anomaly-based network intrusion detection: Techniques, systems and challenges. *Computers & Security*, 28(1):18–28, 2009.

[4] Z. Gyöngyi, H. Garcia-Molina, and J. Pedersen. Combating web spam with trustrank. In *Intl Conference on Very Large Data Bases*, 2004.

[5] S. Hill and F. Provost. The myth of the double-blind review?: Author identification using only citations. *ACM SIGKDD Explorations Newsletter*, 5(2):179–184, 2003.

[6] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.

[7] M. McGlohon, S. Bay, M. G. Anderle, D. M. Steier, and C. Faloutsos. Snare: A link analytic system for graph labeling and risk detection. In *ACM Conference on Knowledge Discovery and Data Mining*, 2009.

[8] J. Neville, O. Simsek, D. Jensen, J. Komoroske, K. Palmer, and H. Goldberg. Using relational knowledge discovery to prevent securities fraud. In *ACM Conference on Knowledge Discovery and Data Mining*, 2005.

[9] S. Pandit, D. H. Chau, S. Wang, and C. Faloutsos. Netprobe: a fast and scalable system for fraud detection in online auction networks. In *International Conference on World Wide Web*, 2007.

[10] A. G. Schwing, T. Hazan, M. Pollefeys, and R. Urtasun. Distributed message passing for large scale graphical models. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[11] J. Viega. *Myths of security*. O'Reilly Media, Inc, 2009.

[12] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Exploring artificial intelligence in the new millennium. chapter Understanding Belief Propagation and Its Generalizations. 2003.

[1] Due to the privacy issues, we were not able to include authentication logs to incorporate user credentials in our experiments.