Automatic Derivation and Validation of a Cloud Dataset for Insider Threat Detection

Pamela Carvallo[1,2], Ana R. Cavalli[1,2] and Natalia Kushik[1]
[1]*SAMOVAR, Télécom SudParis, CNRS, Université Paris-Saclay, Évry, France*
[2]*Montimage, Paris, France*

Keywords: Dataset, Cloud Computing, Intrusion Threat, User Behavior, Synthetic Data Generation, Dataset Validation.

Abstract: The malicious insider threat is often listed as one of the most dangerous cloud threats. Considering this threat, the main difference between a cloud computing scenario and a traditional IT infrastructure, is that once perpetrated, it could damage other clients due to the multi-tenancy and virtual environment cloud features. One of the related challenges concerns the fact that this threat domain is highly dependent on human behavior characteristics as opposed to the more purely technical domains of network data generation. In this paper, we focus on the derivation and validation of the dataset for cloud-based malicious insider threat. Accordingly, we outline the design of synthetic data, while discussing cloud-based indicators, and socio-technical human factors. As a proof of concept, we test our model on an airline scheduling application provided by a flight operator, together with proposing realistic threat scenarios for its future detection. The work is motivated by the complexity of the problem itself as well as by the absence of the open, realistic cloud-based datasets.

1 INTRODUCTION

Cloud computing offers many benefits to the users and therefore, its security issues represent major concerns for those intending to migrate to the cloud. To reach its full potential, cloud computing needs to integrate solid security mechanisms that enhance trust in cloud services and infrastructures. Although numerous research has been carried out with the aim of detecting vulnerabilities and attacks, there are some types of threats that are more complex to detect and predict, such as malicious insider threats.

Such type of threats are more dangerous in a cloud environment than in a traditional IT infrastructure because the insider may gain access to data from other Cloud Service Clients (CSCs) hosted by the Cloud Service Provider (CSP). Many research works have been done addressing relevant indicators when trying to detect malicious insider threats (Costa et al., 2014; Nkosi et al., 2013; Kandias et al., 2010; Kandias et al., 2013; Claycomb and Nicoll, 2012; Duncan et al., 2015). However, to the best of our knowledge, very few publications (Claycomb and Nicoll, 2012; Duncan et al., 2015; Kandias et al., 2013) discuss their implications in a cloud environment.

Furthermore, when aiming to detect this malicious activity, the confidentiality and privacy of CSPs and CSCs concerning their internal organization and policies, create barriers for the collection and utilization of data for research purposes. Moreover, despite the predictions and possible creative attacks presented by researchers, we have little evidence of actual events involving the type of insider described in the CSA's (Cloud Security Alliance CSA, 2016) document (Claycomb and Nicoll, 2012). Additionally, addressing malicious activities also presents challenges since they vary according to the cloud service model, CSC characteristics such as services used, job types and organizational hierarchy.

All considerations described entail that it is sometimes preferable to proceed with synthetic data. This schema allows more extensible benchmarks as experiments are controllable and repeatable (Ringberg et al., 2008; Wright et al., 2010), even though synthetic data will only be realistic in a limited number of dimensions (Brian et al., 2014). Despite the significant dataset contributions (Kholidy and Baiardi, 2012; Brian et al., 2014) in the insider threat domain, their ability to consider realistic cloud-based environments against current and evolving scenarios, is a practical concern.

We also note that Cloud Computing (CC) security requirements are being usually implemented through a methodology in which threats are tackled and tested

Carvalllo, P., Cavalli, A. and Kushik, N.
Automatic Derivation and Validation of a Cloud Dataset for Insider Threat Detection.
DOI: 10.5220/0006480904800487
In *Proceedings of the 12th International Conference on Software Technologies (ICSOFT 2017)*, pages 480-487
ISBN: 978-989-758-262-2

incrementally. Therefore, synthetic data can be useful to confirm that a system detects a particular type of anomaly when that anomaly can be defined and measured. Consequently, by integrating the business logic to the generated model, a portion of the company's behavior is analyzed. To add *ground-truth* labeling to the dataset, a flight operator provided us with a real use case, namely a flight scheduling multi-cloud application. For this reason, we added insider activity based on several realistic threat scenarios into the generated data.

The main contributions of this paper are:

- A dataset generation methodology that takes into account various issues and changes including statistical analysis as well as the creation of cloud-related user scenarios.
- A dataset validation criteria based on a set of predefined rules that include statistical evaluation.
- A design and presentation of a cloud-based proof-of-concept with malicious insider attacks.

The structure of the paper is as follows. Section 2 presents the related work. Section 3 presents of the threat of study, while Section 4 describes the dataset design and generation approach. Section 5 presents a proof of concept of our methodology. Finally, Section 6 concludes the paper and presents the future work.

2 RELATED WORK

Many insider-threat datasets have been proposed in the literature. Authors of (Kholidy and Baiardi, 2012) proposed a cloud-based dataset for masquerade attacks, i.e., where an attacker assumes the identity of an authorized user for malicious purposes. They utilized network and host traces from two machines of the DARPA dataset (Lincon Laboratory MIT, 2017), consisting in host-based audits from Windows NT and Unix Solaris, along with their corresponding TCP dump data. They correlated a seven week dataset and labeled the users from both machines into different roles according to their login session time and the characteristic of the user task (e.g., programmer, secretary, system administrator). Later they assigned every user to a labeled VM.

Additionally, non-cloud related literature on dataset generation shows a variety of approaches. RUU dataset was provided by (Salem and Stolfo, 2011), also concerning masquerade attacks. They built a sensor host for Windows OS that captured user's registry actions, process execution and window touches. They collected normal users and analyzed differences of masquerade users, following a controlled exercise. Carnegie Mellon's Computer Emergency Response Team (CERT) generated a collection of synthetic insider threat test datasets (Brian et al., 2014) to produce a set of realistic models. Authors of (Shiravi et al., 2012) proposed the ISCX dataset, under the notion of profiles that contained detailed descriptions of intrusions and abstract distribution models for applications, protocol and lower level network entities. The ADFA dataset (UNSW, Australian Defense Force Academy, 2017) was proposed by (Creech and Hu, 2013) with modern attack patterns and methodology. This dataset was composed of thousands of system call traces collected from a contemporary Linux local server, with six types of up-to-date cyber attacks involved.

The literature mentioned above raises many questions about the proper characterization of malicious insider threat and which features could adequately describe it for later detection techniques' analysis. Additionally, most of the presented datasets correspond to one-time implementations, which limits the generation and analysis to that particular testbed configuration. In this respect, we aim at an automatic dataset generation that can establish different scenarios with more dynamics taken into consideration. This feature also makes the analysis modifiable, extensible and reproducible. The following sections present this approach.

3 INSIDER THREAT

A malicious insider threat is defined by the CERT (Collins et al., 2016) as a "*threat to an organization occasioned by a current or former employee, contractor, or another business partner who has or had authorized access to an organization network, system or data. This action intentionally exceeded or misused the access in a manner that negatively affected the confidentiality, integrity, or availability of the organization information or information systems*".

This threat has been modeled in numerous studies. An ontology-based definition was provided by (Costa et al., 2014) using a real-world case data, considering factors such as: (i) Human behavior; (ii) Social interactions and interpersonal relationships; (iii) Organizations and organizational environments; and (iv) Information technology security. Another taxonomy was considered by (Kandias et al., 2010), composed of (i) System role (novice, advanced, administrator); (ii) Sophistication (low, medium, high); (iii) Predisposition (low, medium, high); (iv) Stress level (low, medium, high).

However this threat in cloud-environment in-

creases in complexity taking into account new actors involved, as well as their dependencies. Subsequently, a role-based categorization was proposed by (Claycomb and Nicoll, 2012), where an insider threat could be: (i) a malicious insider from the CSC, accessing cloud services or (ii) a malicious insider from the CSP, accessing to sensitive company data. In addition to these, more actors were presented by (Duncan et al., 2015): (i) Malicious insider from the Internet Service Provider (ISP) for each zones; (ii) External CSPs if resources are outsourced to other providers; and (iii) Cloud provisioning services (brokers).

User profiling research and recollection of real cases (Collins et al., 2016) present no common pattern with respect to subject's personal characteristics. However there are risk indicators (Greitzer et al., 2016) related to the motivational factors that may underlie malicious insider exploits, which are supported by studies indicating that most of these attacks (81%) are planned (Shaw, 2006).

From these representations of insider user profiling, we derive our model definition in the following section. Moreover, we propose a threat ontology with a probabilistic approach, where the motivational risk factors mentioned above are considered to occur with a given probability in time.

4 DATASET GENERATION

This section contains the methodology for deriving a configurable and automatic dataset for malicious insider threat in cloud environments.

We note that in thise case, the *anomalous* events are generated from a semantically distinct process than the generated from *normal* `Events`. For example, as mentioned in (Emmott et al., 2013), *anomalous* events should not just be points in the tails of a Normal distribution. Moreover, to provide heterogeneous data, they should be generated from different criteria.

4.1 Definition of Entities

We consider the following entities to generate patterns of activity, as shown in Figure 1. They are divided into two groups. On one hand, those are user-related entities, namely:

Profile is defined as an abstract representation of a person's attributes in an organization, to facilitate the reproduction of realistic behaviors. Each profile is composed of a `Factor`, a `Context` and a `Role`.

Factor is related to the human characteristics of a person. This category could add dynamic and static mid-term attributes such as the attitude for the given job, or more mid-term static factors such as a capability. Every `Factor` can be personalized and distinguished by its probability attribute.

Role is associated to the function of the `Profile` in a given organization. Moreover, we define the `Role` through the entity `Policy`, which is composed of a `Permission` related to an action in an `Asset` .

Context consists of attributes related to specific time and location conditions where the `Profile` is performing its `Role` (e.g., location, time of the day, IP from where jobs are being executed, cloud instance trying to access, among others).

Asset consists of any valuable hardware or software component, property of a CSP or CSC in the CC stack (e.g., physical servers, VMs, applications, databases, communication infrastructure), depending on SaaS, PaaS or IaaS models.

Permission is the type of authorization a `Role` has for a given `Asset` (e.g., read, modify).

On the other hand, the simulation also includes the event-related entities:

Sequence of actions is a list of sequential `Actions` performed by the same `Profile` under a given time interval. The approach generates three types of `Sequences`: (i) Pseudo-randomly selected `Actions`, following a specified distribution, (ii) Predefined *normal* and *anomalous* `Sequences` of `Actions`, under realistic scenarios, (iii) Hybrid sequences, composed by fixed actions in arbitrary positions, filled with pseudo random actions for the rest of the sequence.

Label represents the nature of the `Sequence` occurred (e.g., *normal*, *anomalous*).

Event consists in the tuple of attributes $\langle timestamp, profile, sequence, context, label \rangle$. This corresponds to an observation of a `Sequence` with a specific `Label`, performed by a `Profile` in a `Context` for a given time. These events can be generated with a given distribution (e.g., Uniform, Normal, Poisson), depending on the scenario of study, i.e., a Normal distribution could model rush-hours (e.g., patients arriving to a hospital, end-users accessing a website application).

This last group defines the sequences of actions for a `Profile` and gives them a label. Moreover, some `Roles` have predefined sequences.

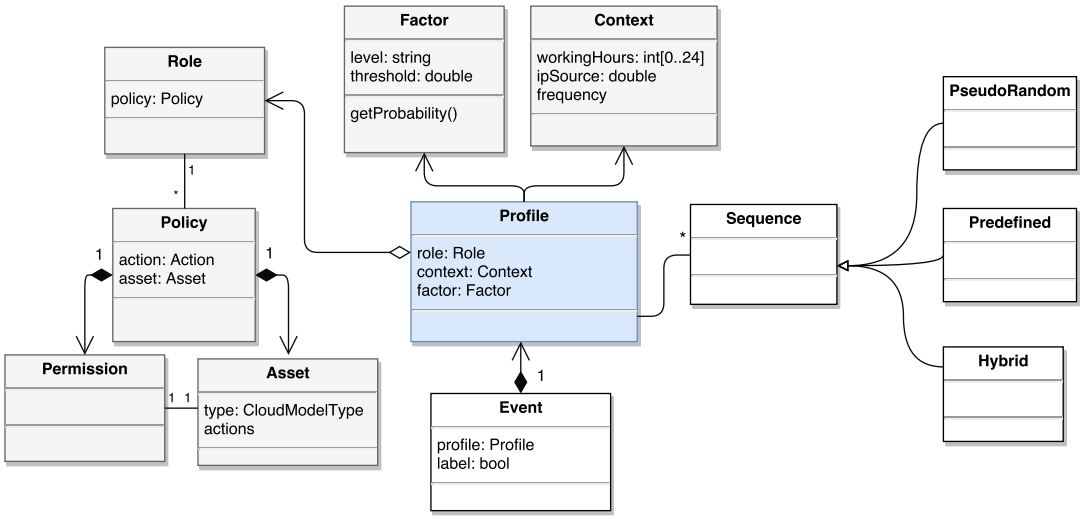

Figure 1: Class models for generation of cloud insider threat dataset

4.2 Generation Algorithm

The Algorithm 1 presents the dataset generation methodology. This process is initiated by a list of `Profiles`, the inter-arrival time between agents (*tba*) and the timeout (*to*). The *tba* parameter has relation with the amount of `Agents` to generate, and therefore with the `Event` generation process. Below, we provide more details related to the proposed algorithm referring to the corresponding lines of it.

Lines 2 to 7: Algorithm defines the `delay` for which it will create an `Event` from a pseudo-randomly chosen `Profile`. This generation is done with an exponential distribution, in order to model the pseudo-random generated sequences as a Poisson process (Line 6).

Lines 10 to 14: An agent function takes the form of a `Profile` assigned by a `Role` and Time Between Events (*tbe*) (e.g., Normal distribution with *mu* and *sigma* given by the `Context`).

Lines 17 to 18: The three types of events will be generated with a given probability in time. Each of these group of sequences is given by the `Policy` entity, which relates an `Action` to a given cloud `Asset`.

Lines 16 to 21: Under a certain probability given by the `Factor`, each `Profile` will have an *anomalous* behavior. The function `GenAnomaly` changes the `Context` attributes (e.g., source IP from where the `Sequence` was performed) introducing a single instance of such anomaly.

Algorithm 1: Dataset generator.

```
1:  function GENEVENTS(profiles, tba, to)
2:      while !to do
3:          profile ← PRNChoice(profiles)
4:          delay ← ExpDistrib(1/tba)
5:          AGENT(profile)
6:          WAIT(delay)
7:      end while
8:  end function
9:  function AGENT(b)
10:     role ← profile.role
11:     ctx ← profile.context
12:     tbe ← Distrib(ctx.mu, ctx.sigma)
13:     prob ← FACTORPROB(profile.factor)
14:     label ← "normal"
15:     for all pol in role.policies do
16:         if prob < MaliciousTresh then
17:             seq ← ChooseSeq(pol.PRNSeq,
                pol.predefSeq, pol.hybSeq)
18:             RUNEVENT(seq, probile, label)
19:         else
20:             GENANOMALY(profile)
21:             label ← "anomalous"
22:             RUNEVENT(seq, profile, label)
23:         end if
24:         WAIT(tbe)
25:     end for
26: end function
```

4.3 Dataset Validation

Defining suitable criteria for dataset validation is a complex process, since there are no general methodologies in the literature (Claycomb and Nicoll, 2012). To this end, we have defined three validation criteria. The first two (namely, items 1 and 2) add an a priori degree of realism to detect plausible attacks, as the result of consulting with the CSC use-case experts. The third (namely, item 3) rely on an a posteriori verifica-

tion and is used to prove the applicability of the proposed approach given the nature of the data for prediction or detection techniques:

1. Statistical similarity of events: the generated `Events`, should be statistically based in *realistic* behaviors for every `Profile` entity and each threat scenario. Such statistical data can be either provided by an *oracle* aware of the activities for each `Role`, or from traces of real case studies for later extrapolation.

One example could be a number of actions an employee should do on a monthly basis. In this case, an expert knows that a security administrator can initiate an action at any given time (e.g., 24x7 service availability) while a Database Administrator (DBA) `Role` should not initiate more than *N* `Events` per month, that consider a database back-up `Sequence`.

2. Sequences realism: the pseudo-random generated set of sequences for each event, should be validated to make sense in the context of the `Permission-Asset` tuple. In other words, independent actions to a given asset, for example, data elimination or tampering from a database, might not have a pre-defined order. In the case of other tuples, such as actions to a VM asset, it might be intuitive to generate `Sequences` with a given order (i.e., an action of `shuttingVMDown()`, this action cannot be followed by any other operation that assumes the VM is operational). The latter means that the set of invariants from the second group includes the ordering of the actions performed, i.e., action B in a `Sequence` cannot be executed by a given `Profile` unless the action A has been processed.

3. Anomaly detection techniques benchmarking: For an accurate prevention of this threat, proper anomaly detection benchmarking can be performed. For this matter, the dataset should contain "well distributed" labeled `Events` and its technique should recognize possible label imbalance (malicious `Events` are less frequent than *normal*). This is relevant at the moment of experimenting with detection techniques such as supervised machine learning models, as they can try to fit *anomalies* with *normal* events. Also, performance indicators such as AUC (Area Under the ROC curve) provide better understanding of the variability, where accuracy is divided into sensitivity and specificity, and better configurations can be chosen based on the balance thresholds of these values.

5 PROOF OF CONCEPT

The proof of concept is performed based on an airline scheduling cloud application provided by a CSC operator. Today's airlines need to permanently revise their flight schedule plans in response to competitor actions, while constantly maintaining operational integrity.

In this case study, we are particularly interested in researching how insider threats can attempt to perform malicious activities towards the Database Management System (DBMS) component, as it stores very relevant data about the flights and personal client information. The feasibility and effectiveness of our approach are evaluated through the validation criteria given in Section 4.3 and are presented in Section 5.2.

To perform our experiments, we have designed a testbed composed of two Linux instances in OpenStack, as depicted in Figure 2. These instances define the functionalities of an airline service, where the "Server Instance" consists of several application artifacts and system components, and communicates with the DBMS application. The created dataset contains generated events from five consecutive years. The chosen design criteria for the `Profiles` generated is outlined in Table 1. The description of the a five-month period is depicted in Figure 3, with the different nature of the *normal* and *anomalous* event generation i.e,. different frequency given by the `Factor` attribute for each `Profile`.

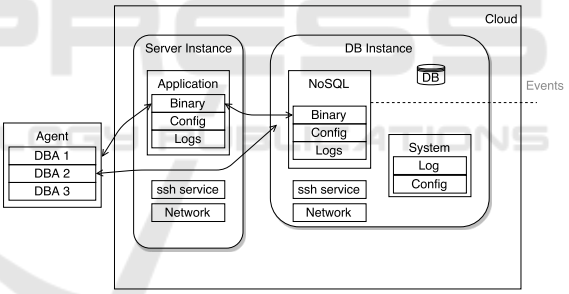

Figure 2: Proof of concept implementation.

5.1 Profile's Behavior Representation

As mentioned in the definition of entities, we represent each profile's behavior following a role-based approach. We utilize as an example a DBA, defined as a user in charge of administrative actions towards the database, such as installation, patching and upgrade of the database. This includes the ownership of all objects of the database and the ability to create and modify roles and users.

As defined in Section 4.1, each `Role` has pseudo-random, predefined and hybrid group of generated `Sequences` followed by a *normal* behavior. The following statements represent the examples of the events a DBA can perform and, therefore, define the DBA normal behavior:

- A *DBA* in a working day, logs into the DBMS and enrolls a new user with write permission over a database.
- A *DBA* regularly works remotely on Wednesdays, logging into the DBMS from $location_a$, while the rest of the week from $location_b$.

On the other hand, the following sequences can represent the examples of the generated `Sequences` for an anomalous DBA behavior:

- A *DBA* logs in from a public IP that does not belong to the company and performs a sequence of actions.
- A *DBA* outside Working Hours (WH), logs in and performs numerous sequences of actions on the database.

We have outlined three types of `Profiles` (DBA_1, DBA_2 and DBA_3) with the same DBA `Role`, differentiating them by a created `Factor` named "skill level" as described in Table 1. This `Factor` defines the time taken to perform a `Sequence` of actions with low, medium and high skills and prompts the sequences' length. We also have modeled the three profiles with the `Factor` of being malicious, with different probabilities or discontent. Such cases are depicted in Figure 3, where profile generation is performed under the realistic constraints given by the use case scenario experts. Additionally, in our example the generation of `Events` is treated by the "No. Monthly events" which we have settled at 10 events per day.

Table 1: Chosen parameters for DBAs.

Parameters	DBA_1	DBA_2	DBA_3
IP 192.168.1.	.100	.110	.120
Location	Germany	Germany	Germany
WH	9am-6pm	9am-6pm	9am-6pm
Skill level	30 (high)	60 (med)	120 (low)
Malicious Distrib.	Constant rate	Poisson	Random
No. Daily Events	10	10	10

5.2 Results

The validation criteria presented in Section 4.3, allow us to determine the usability of the generated dataset. We verify these criteria with the given proof-of-concept in the following steps. For the first criteria, we refer to Figure 3 and Table 2 for the average and standard deviation of the generated `Events`.

The second criteria studies the *realism* of the sequences performed by each `Profile`, with respect to their `Role` in the company. We have validated our generation methodology by using the sequence alignment algorithm and obtained scores for our predefined sequences i.e., Create (C), Read (R), Update (U), Delete (D), treated as sub-sequences among all the generated sequences. This other group of sequences and their pseudo-random generation, derive in a heterogeneous dataset with the length showed in Table 3.

Table 2: Criteria 1: Statistical similarity of events per day.

Profile	Max. Criteria	Normal Events		Malic. Events	
		Avg.	S.D	Avg.	S.D
DBA_1	10	7.23	2.60	1.58	1.11
DBA_2	10	7.13	3.28	1.79	3.61
DBA_3	10	7.73	3.28	0.55	1.01

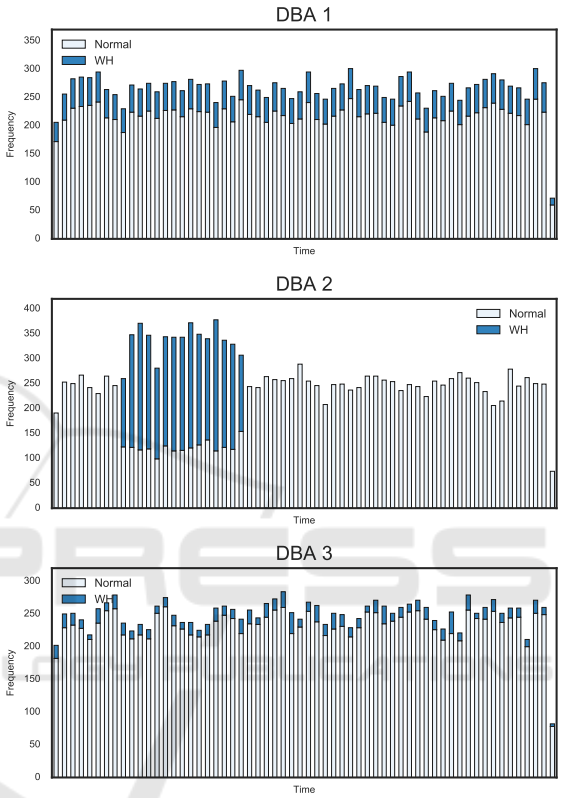

Figure 3: First year histogram of anomalies per month for Profile with with malicious probability of fixed rate (1 every 10 `Events`, Poisson and Uniform distributions.

The Table 4 shows the results for the last item of the validation criteria, where we addressed the well-known machine learning classifier Support Vector Machine (SVM) to estimate the quality of the generated dataset with respect to their classification performance: Recall ($TP/(TP+FN)$ where TP and FN are True Positive and False Negative values, respectively) and Precision, which shows the ability of the

Table 3: Criteria 2: Example of sequences realism for the DBA role.

Profile	Seqs. Similarity		Seqs. Length	
	Predef.	Ratio	Avg.	S.D.
DBA_1	[C, R]	0.56	1.81	0.52
DBA_1	[R, U]	0.55	1.71	0.61
DBA_1	[R, D]	0.55	1.72	0.64

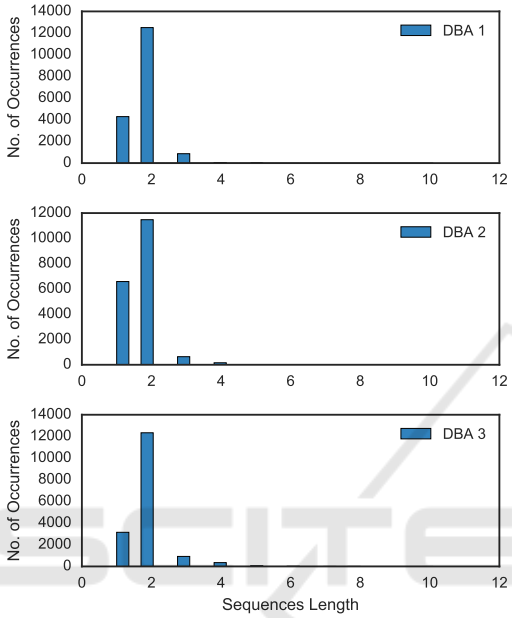

Figure 4: Histogram of length for the Profile with high, medium and low skills.

classifier not to label as positive a sample that is negative ($TP/(TP+FP)$, where FP is False Positive). Additionally, the AUC score is a plot of the TP rate versus FP rate for a binary classifier as its discrimination threshold varies. The area under the ROC curve (AUC) therefore reflects the relationship between sensitivity and specificity. Below, we present a 3 experiential setup, namely Exp. 1, 2, and 3. For the Exp.1. we fixed the malicious probability and we used different values for the skill attribute. The Exp.2. was performed using equal values for skill, while different malicious probability. Exp.3. relies on different values for both, skill and malicious probabilities.

As can be seen from the tables, the skill factor does not essentially affect the SVM prediction. However, the malicious probability distribution in time significantly influences, for example the FP rate. Other parameters and factors need to be taken into consideration in order to estimate their correlation with the SVM prediction score. The results can also be analyzed from the performance of the models using the receiver operating characteristic (ROC) curve.

Table 4: Average Detection Performance for SVM classifier.

Exp. Setup	SVM Metric	DBA_1	DBA_2	DBA_3
Exp. 1.	Recall (%)	98.21	97.48	93.68
	Precision (%)	82.26	80.69	76.56
	AUC Score	0.93	0.92	0.89
Exp. 2.	Recall (%)	51.06	95.58	50.23
	Precision (%)	90.31	73.57	71.42
	AUC Score	0.74	0.91	0.75
Exp. 3.	Recall (%)	85.36	95.04	46.89
	Precision (%)	73.01	71.85	54.24
	AUC Score	0.89	0.91	0.72

A higher AUC indicates better overall performance. Given the experimental results, one can conclude that the SVM technique shows good performance for the malicious insider dataset being derived.

6 CONCLUSION

In this paper, we have presented a methodology for dataset generation and a validation approach including several criteria such as statistical evaluation, sequence realism and benchmarking of detection techniques. Even more, we have implemented a proof-of-concept with anomalous malicious insider attacks, which has been validated on a realistic use-case. As a conclusion, we can say that the results obtained will be very useful for intrusion detection techniques and, in particular, for these working on malicious insider threats.

As a future work, we plan to extend the results presented in this paper by the addition of social engineering factors to our model and the benchmarking of new scenarios, to evaluate the impact of data variability. Another future research will be to use the cloud-based dataset we have created to compare the detection ability of different anomaly-based intrusion detection techniques. Finally, we will intend to propose novel prediction techniques for insider threats, following the capability of dynamically deploying different use-case scenarios.

ACKNOWLEDGEMENTS

The work presented in this paper has been developed in the context of the MUSA EU Horizon 2020 project (MUSA, 2017) under grant agreement No 644429.

REFERENCES

Brian, L., Joshua, G., Mitch, R., and Kurt C., W. (2014). Generating test data for insider threat detectors. *JoWUA*, 5(2):80–94.

Claycomb, W. R. and Nicoll, A. (2012). Insider threats to cloud computing: Directions for new research challenges. In *2012 IEEE 36th Annual Computer Software and Applications Conference*, pages 387–394.

Cloud Security Alliance CSA (2016). The Treacherous 12 - Cloud Computing Top Threats in 2016.

Collins, M., Theis, M., Trzeciak, R., Strozer, J., Clark, J., Costa, D., Cassidy, T., Albrethsen, M., and Moore, A. (2016). Common sense guide to mitigating insider threats. Technical Report CMU/SEI-2016-TR-015, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA.

Costa, D., Collins, M., Perl, S. J., Albrethsen, M., Silowash, G., and Spooner, D. (2014). An ontology for insider threat indicators: Development and application. In *Proceedings of the Ninth Conference on Semantic Technology for Intelligence, Defense, and Security, Fairfax VA, USA, November 18-21, 2014.*, pages 48–53.

Creech, G. and Hu, J. (2013). Generation of a new ids test dataset: Time to retire the kdd collection. In *2013 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 4487–4492.

Duncan, A., Creese, S., and Goldsmith, M. (2015). An overview of insider attacks in cloud computing. *Concurrency and Computation: Practice and Experience*, 27(12):2964–2981.

Emmott, A. F., Das, S., Dietterich, T., Fern, A., and Wong, W.-K. (2013). Systematic construction of anomaly detection benchmarks from real data. In *Proceedings of the ACM SIGKDD Workshop on Outlier Detection and Description*, ODD '13, pages 16–21, New York, NY, USA. ACM.

Greitzer, F. L., Imran, M., Purl, J., Axelrad, E. T., Leong, Y. M., Becker, D. E., Laskey, K. B., and Sticha, P. J. (2016). Developing an ontology for individual and organizational sociotechnical indicators of insider threat risk. In *STIDS*.

Kandias, M., Mylonas, A., Virvilis, N., Theoharidou, M., and Gritzalis, D. (2010). *An Insider Threat Prediction Model*, pages 26–37. Springer Berlin Heidelberg, Berlin, Heidelberg.

Kandias, M., Virvilis, N., and Gritzalis, D. (2013). *The Insider Threat in Cloud Computing*, pages 93–103. Springer Berlin Heidelberg, Berlin, Heidelberg.

Kholidy, H. A. and Baiardi, F. (2012). CIDD: A Cloud Intrusion Detection Dataset for Cloud Computing and Masquerade Attacks. In *2012 Ninth International Conference on Information Technology: New Generations (ITNG)*, pages 397–402. IEEE.

Lincon Laboratory MIT (2017). Darpa intrusion detection evaluation. https://www.ll.mit.edu/ideval/data/index.html.

MUSA (2017). MUSA H2020 project. http://www.musa-project.eu/. (Retrieved May 2017).

Nkosi, L., Tarwireyi, P., and Adigun, M. O. (2013). Insider threat detection model for the cloud. In *2013 Information Security for South Africa*, pages 1–8.

Ringberg, H., Roughan, M., and Rexford, J. (2008). The need for simulation in evaluating anomaly detectors. *SIGCOMM Comput. Commun. Rev.*, 38(1):55–59.

Salem, M. B. and Stolfo, S. J. (2011). Modeling user search behavior for masquerade detection. In *Proceedings of the 14th International Conference on Recent Advances in Intrusion Detection*, RAID'11, pages 181–200, Berlin, Heidelberg. Springer-Verlag.

Shaw, E. D. (2006). The role of behavioral research and profiling in malicious cyber insider investigations. *Digit. Investig.*, 3(1):20–31.

Shiravi, A., Shiravi, H., Tavallaee, M., and Ghorbani, A. A. (2012). Toward developing a systematic approach to generate benchmark datasets for intrusion detection. *Computers & Security*.

UNSW, Australian Defense Force Academy (2017). Adfa ids datasets. https://www.unsw.adfa.edu.au/australian-centre-for-cyber-security/cybersecurity/ADFA-IDS-Datasets/.

Wright, C. V., Connelly, C., Braje, T., Rabek, J. C., Rossey, L. M., and Cunningham, R. K. (2010). *Generating Client Workloads and High-Fidelity Network Traffic for Controllable, Repeatable Experiments in Computer Security*, pages 218–237. Springer Berlin Heidelberg, Berlin, Heidelberg.