A Transparent Learning Approach for Attack Prediction Based on User Behavior Analysis

Peizhi Shao[1], Jiuming Lu[1], Raymond K. Wong[1(✉)], and Wenzhuo Yang[2]

[1] School of Computer Science and Engineering, University of New South Wales, Kensington, Australia
wong@cse.unsw.edu.au

[2] School of Computer Science and Engineering, Nanyang Technological University, Singapore, Singapore

Abstract. User behavior can be used to determine vulnerable user actions and predict potential attacks. To our knowledge, much work has focused on finding vulnerable operations and disregarded reasoning/-explanations of its results. This paper proposes a transparent learning approach for user behavior analysis to address this issue. A user rating system is proposed to determine a security level of each user from several aspects, augmented with explanations of potential attacks based on his/her vulnerable user actions. This user rating model can be constructed by a semi-supervised learning classifier, and a rule mining algorithm can be applied to find hidden patterns and relations between user operations and potential attacks. With this approach, an organization can be aware of its weakness, and can better prepare for proactive attack defense or reactive responses.

Keywords: Transparent learning · Machine learning · User behavior analysis · Cybersecurity

1 Introduction

Cybersecurity (CS) is to study the processes and/or technologies that protect computers, programs, networks, and data from attacks, unauthorized access/change, or destruction. Cyber attacks are argued as the actions that attempt to bypass security mechanisms of computer systems [31]. A cyber attack detection is to identify individuals who try to use a computer system without authorization, or those who have access to the system but abuse their authorities [10]. Most attacks in general can be grouped into Denial of Service Attacks, Remote to Local Attacks, User to Root Attacks, and Probing [34].

Due to the increasing number of incidents of cyber attacks, CS has always been a critical issue concerned by every Internet user. Well-trained hackers make the traditional online security protection methods such as firewalls or virus detection software no longer effective. On the other hand, user behavior analysis (UBA) become a new area to detect online attacks and also perform real-time analysis based on user behavior and actions.

K.-Y. Lam et al. (Eds.): ICICS 2016, LNCS 9977, pp. 159–172, 2016.
DOI: 10.1007/978-3-319-50011-9_13

For example, email is a main intermediary for spreading virus and Trojan horse. The attackers will widely spread emails containing worm programs to infect computers and networks. These emails usually have attractive subject, which draw user's attention and allure them to open them. For example the widely spread Verona virus, also known as Romeo & Juliet virus, contain the words like *I love you* or *sorry*. So if a user is easily attracted by these words and opens these emails from unknown senders, it is likely that they will be affected and cause security threat to their systems. For cautious users with good online behavior, they may check the sender email address and distinguish if an email is safe to open. In many cases these users may just delete these emails and blacklist he sender address.

Meanwhile, big data analysis has been widely used in commercial applications such like product recommendation, and UBA is used for identifying target customers for certain products. Recently, UBA has gained tractions in CS [26]. Compared with the traditional attack detection methods (which detect the actions of a certain attack or the existence of virus software or Trojan horse), UBA focuses on determining abnormal user actions based on their usual, normal activities. After that, warnings can then be generated and countermeasure can be implemented. In general, it is difficult for an attacker to imitate the original user's behavior. UBA can also be used to detect insider attacks, for example, an employee may illegally transfer the company data to other competitors for profit.

Since UBA allows users to implement preventive measures instead of detecting the attacks, CS companies have started adding UBA into their products/services. However, to the best of our knowledge, most UBA approaches focus on finding vulnerable user operations, and do not provide explanation or reasoning of their findings. In this paper, we propose a novel approach that is based on the concept of transparent learning, in which a prediction of attacks can be reasoned with explanations. As a result, an organization can be aware of its weakness, and can better prepare for proactive attack defense or reactive responses.

The rest of this paper is organized as follows. Section 2 summarizes related work and Sect. 3 presents our proposed approach. Section 4 describes an example to illustrate our approach and finally Sect. 5 concludes this paper.

2 Related Work

2.1 Machine Learning

There are plenty of works focusing on using different machine learning (ML) methods/models for detecting abnormal operations and predicting potential attacks in CS. For example, artificial neural networks were used in Cannady [11] to classify user operations into different categories of user misuses. Lippmann and Cunningham [21] proposed an Anomaly Detection System using keyword selection via articial neural networks. Bivens et al. [8] presented a complete Intrusion Detection System including the following stages: preprocessing, clustering the normal traffic, normalization, articial neural networks training and articial neural networks decision. Jemili et al. [15] suggested a framework using Bayesian

network classifiers using nine features from the KDD 1999 data for anomaly detection and attack type recognition. Kruegel et al. [17] used a Bayesian network to classify events for OS calls. In Li et al. [20], an SVM classifier with an RBF kernel was used to classify the KDD 1999 dataset into predefined categories (Denial of service, Probe or Scan, User to root, Remote to local, and normal). Amiri et al. [2] used a least-square SVM to be faster son the same dataset. Some other ML models such as Hidden Markov Models [5] and Nave Bayes classifiers [35] have also been popular in CS, e.g., [3,16,25].

2.2 Rule Mining

Association rule Mining was introduced by Agrawal et al. [1] for discovering frequently appearing co-occurrences in supermarket data. Brahmi [9] applied the method to capture relationships between TCP/IP parameters and attack types. Zhengbing et al. [36] proposed a novel algorithm based on the signature apriori algorithm [14] to find new attack signatures from existing ones. Decision Trees such as ID3 [29] and C4.5 [30] are widely used for rule mining. Snort [24] is a well-known open-source tool using the signature-based approach. Kruegel and Toth [18] used DT to replace the misuse detection engine of Snort. Exposure [6,7] is a system using Weka J48 DT (an implementation of C4.5) as the classifier to detect domains that are involved in malicious activities. Inductive learning is a bottom-up approach that generates rules and theories from specific observations. Several ML algorithms such as DT are inductive, but when researchers refer to inductive learning, they usually mean Repeated Incremental Pruning to Produce Error Reduction [12] and the algorithm quasi-optimal [22]. Lee et al. [19] developed a framework using several ML techniques such as inductive learning, Association rule and Sequential pattern m ining to detect user misuses.

2.3 User Behavior Analysis

Researchers in [4] used UBA in CS. In [26], clustering algorithms such as Expectation Maximization, Density-Based Spatial Clustering of Applications with Noise, k-means have been used in UBA. A combination of these methods has also been discussed in [33]. Commercial products/services such as Lancope, Splunk, Solera have started including UBA in their offerings.

3 Our Approach

Unlike most existing UBA approaches, by following the concept of transparent learning, we use 2 independent modules working together. Firstly, a semi-supervised learning module is designed to rate a security risk of each user. The learning takes the rating that is a tuple of 3 scores that consider 3 different aspects, namely, constancy, accuracy and consistency, into account. Secondly, a rule mining module is used to identify hidden patterns between historic user operations and an attack, and is used to reason why the user is rated at a particular security level.

3.1 Transparent Learning

Transparent learning is a ML concept that aims at the transparency of ML models, algorithms and results. An ideal transparent learning technique is one that [32]:

– Produces models that a typical user can read, understand and modify.
– Uses algorithms that a typical user can understand and influence.
– Allows a user to incorporate domain knowledge when generating the models.

The main advantage of transparent learning is its interpretability. This is very important in CS, especially in understanding the reasons behind potential attack prediction. Most existing UBA systems can find potential vulnerabilities, but are unable to provide reasoning/explanations. This is because most of these systems are based on clustering or outlier detection, and many security experts may not understand why or how a prediction is made. To address this issue, we propose a transparent learning model containing 2 modules as shown in Fig. 1.

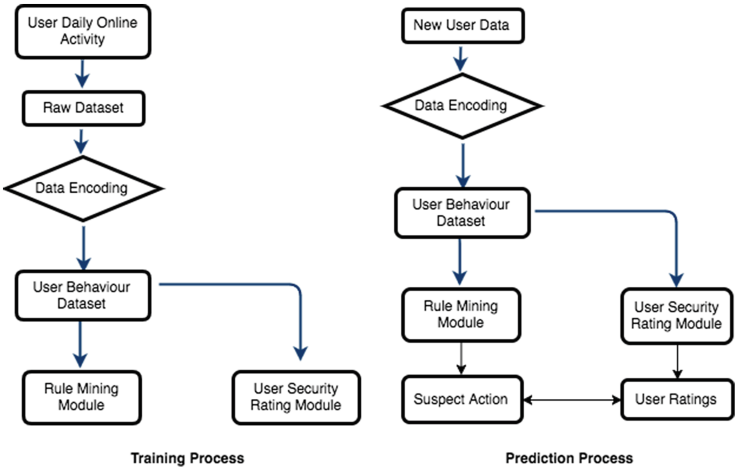

Fig. 1. The two main modules

3.2 User Security Rating Module

Based on user's daily online activities and the manner of using the computer, each user will be given a security rating. This rating indicates the probability that this user may lead to a threat to the system. This user behavior is a representation of one's personality and knowledge, and can be analysed from different aspects. Inspired by [13], we consider the following aspects when determining the rating:

Constancy – how long the user continuously maintains good online behavior. It indicates that the user has a good understanding of CS and/or the security policy of the company. It also illustrates the awareness of potential online attacks and will cautiously protect themselves from potential cyber attacks.

Accuracy – the frequency that a user makes security related mistake during online activities and the accuracy of the user to finish certain tasks. This shows the proficiency of the user. An experienced user is likely to have better security rating than a beginner.

Consistency – consistent usage patterns. If the user's online behavior is consistent. It is easy to be tracked and risk is also minimized.

3.3 Rule Mining Module

We base on an inductive learning algorithm called GOLEM [23] to develop a rule mining module. The algorithm works as follows. Firstly, it generates LGG [27,28] clauses from each pair of records. Then it picks the one that covering the maximum number of positive examples and considering reduction based on negative examples. Finally, it adds the reduced clause as a sub-rule into the final clause and mark the covered, positive examples. It repeats this process until all positive examples are covered.

GOLEM is a useful algorithm to find hidden rules between a set of facts and the results. However, in UBA, action frequency needs to be considered because frequent user actions should be more important than less frequent ones.

Algorithm 1. User Behavior Frequency GOLEM

Let ε^+ be a set of positive examples of user behavior records.
Let ε^- be a set of negative examples of user behavior records.
Let $M_h(\kappa)$ be an h-easy model of background knowledge κ.
Let s be a given sample limit.
Let $Pairs(s)$ be a random sample of pairs from ε^+.

Let $Lggs = \{C : \{\epsilon, \epsilon'\} \in Pairs(s)$ and $C = urlgg(\{\epsilon, \epsilon'\})$ wrt $M_h(\kappa)$ and C consistent wrt $\varepsilon^-\}$
Let e be a specific operation in $lgg(\{\epsilon, \epsilon'\})$.
Let E be the operation type for operation e.
Let $f(E)$ be the frequency of the operation type E.
Let $ubfc$ of pair $\{\epsilon, \epsilon'\}$ be the product of $cover(lgg(\{\epsilon, \epsilon'\}))$ and the minimum $f(E)$ of $lgg(\{\epsilon, \epsilon'\})$.
Let S be the pair $\{\epsilon, \epsilon'\}$ whose $urlgg$ has the greatest $ubfc$ in $Lggs$.

DO
  $\varepsilon^+(S)$ be a random sample of size s from ε^+
  Let $Lggs = \{C : \epsilon' \in \varepsilon^+(S)$ and $C = urlgg(S \cup \{\epsilon'\})$ consistent wrt $\varepsilon^-\}$
  Find ϵ' which produces the greatest $ubfc$ in $Lggs$.
  Let $S = S \cup \{\epsilon'\}$
  Let $\varepsilon^+ = \varepsilon^+ - cover(urlgg(S))$
WHILE covers more

Therefore, we extend the original GOLEM algorithm to User Behavior Frequency GOLEM (UBF-GOLEM). It is based on the notion of User Behavior Frequency Coverage (UBFC) - the product of the coverage of a specific operation and the frequency of this operation category. Then we change the LGG selection criteria from coverage based to UBFC based. The final algorithm is shown in Algorithm 1, and an example showing how GOLEM algorithm and UBF-GOLEM work is described in Sect. 4.

UBF-GOLEM is able to find hidden rules behind attack actions. For each attack action, we determine the most relevant operations that are related to an attack. After that, for attack prediction, vulnerable operations can be identified by comparing user operations with the generated rules.

4 An Illustrating Example

This section presents a simplified example to illustrate how the two modules work. Assume some worm virus spread from a domain called "virus.com". We first prepare the encoded raw data to generate the user security rating via a semi-supervised learning method.

4.1 Training the User Security Rating Module

Raw data from user online activities, such as log-on data, Device connection data, file transfer data, Http access data and Email data, are collected. Sample of these data is shown in Fig. 2.

Data Encoding. Each user will be given a rating that contains 3 scores by considering the 3 aspects (constancy, accuracy and consistency) mentioned in the previous subsection. Based on these scores, we can train a classifier to rate and rank (based on the likelihood and risk) of a particular user to be attacked.

Training. After the collected data are encoded into multi-dimensional numeric features, we then cluster the data using an unsupervised algorithm such as k-mean and Expectation Maximization. We then manually select and labeled some data for supervised learning the security risk of a user. Typical supervised ML algorithms, such as SVM, artificial neural networks or DT, can be used to train a classifier. We can then use the trained classifier to predict the remaining unlabeled data.

4.2 Using Inductive Learning

In the rule mining module, we classify the user operations into different types. The checkpoint for each action type is as shown in Table 1.

Log-on Data

Log-on id	date	user	pc	activity
X0W9-Q2DW16EI-1074QDVQ	01/02/2010 05:02:50	WCR0044	PC-9174	Logon
C2O4-Z2RH12FQ-9176MUEL	01/02/2010 05:19:09	WCR0044	PC-9174	Logoff

Device Data

Device id	date	user	pc	activity
W7U9-D6EJ66QR-5998NVVQ	01/02/2010 07:44:23	KBD0201	PC-5997	Connect
K3K6-K6PY61IC-7281YJRT	01/02/2010 07:45:55	BHV0556	PC-6254	Connect

File Data

File id	date	user	pc	filename	content
I7H0-J1BK99KL-0235UCYA	01/02/2010 05:16:32	WCR0044	PC-9174	N1Y38G35.doc	conqu...
V6C5-R5SW99WN-5752LTXJ	01/02/2010 05:16:39	WCR0044	PC-9174	L9RS8KW5.doc	Today...

Http Data

Http id	date	user	pc	url	content
G5W9-X4ZB41IJ-1699KNGR	01/02/2010 06:54:37	ACM0931	PC-5571	http://newegg.com	alone on they t1517 residual m7 73 sample subsequently c...
F2P0-H3GU35KC-5660VIYU	01/02/2010 06:55:09	ACM0931	PC-5571	http://megaclick.com	parts dark would possibility 50 middle represent outer 3...

Email Data

Email id	date	user	pc	to	cc	bcc	from	size	attach	content
...	...	...	...	Ci..@dta...	Du...@dta...	El...@dta...	Re...@harr...	34113	3	...
...	...	...	...	Ca...@loc...			Ka...@dtaa..	46808	0	...

Fig. 2. Raw data sample

Example Data. For each user that has been attacked, we encode their data as discussed before. A snapshot of some positive examples is listed below:

```
[ Website_domain("virus.com"),
  Unusual_log_time(true),
  Website_domain("groupon.com"),
  Db_request_without_permission(false),
  Connect_too_long(false),
  Website_domain("google.com"),
  Get_file("1AFEEF45CB07F87C3D598ED8A5AA"),
  Website_domain("thepostgame.com"),
  Too_short_log(false) ]

[ Get_file("2351F8CD57177D4C1044D37460B"),
  Website_domain("imdb.com"),
  Receive_email_from("Glo34Hugh@hotmail.com"),
  Too_short_log(false),
  Db_request_without_permission(false),
  Connect_too_long(false),
  Connect_too_long(false),
  Send_email_to("ki77r089@gmail.com"),
  Send_email_to("mm63ld@gmail.com")]
```

```
[ Receive_email_from("Kora_Guro@virus.com"),
  Receive_email_from("Glo34Hugh@hotmail.com"),
  Too_short_log(false),
  Db_request_without_permission(false),
  Connect_too_long(false),
  Get_file("1AFEEF45CB07F87C3D598ED8A5AA"),
  Send_email_to("ki77r089@gmail.com"),
  Send_email_to("mm63ld@gmail.com")]

[ Unusual_log_time(true),
  Website_domain("groupon.com"),
  Website_domain("imdb.com"),
  Connect_too_long(false),
  Connect_too_long(false),
  Get_file("1AFEEF45CB07F87C3D598ED8A5AA"),
  Too_short_log(false)]

[ Website_domain("virus.com"),
  Get_file("2351F8CD57177D4C1044D37460B"),
  Website_domain("groupon.com"),
  Too_short_log(false)]

[ Get_file("2351F8CD57177D4C1044D37460B"),
  Receive_email_from("Kora_Guro@virus.com"),
  Unusual_log_time(true),
  Receive_email_from("Glo34Hugh@hotmail.com"),
  Too_short_log(false),
  Db_request_without_permission(false),
  Connect_too_long(false),
  Website_domain("google.com"),
  Get_file("1AFEEF45CB07F87C3D598ED8A5AA"),
  Website_domain("thepostgame.com")]
```

Table 1. Data encoding

Online action	Checkpoint	Attribute data input
Log on PC	During work hours?	`Unusual_log_time(true/false)`
Log off PC	A short login session?	`Too_short_log(true/false)`
Web access	Website domain	`Website_domain(‘‘abc.com’’)`
Receive E-mail	Email sender	`Receive_email_from(‘‘xyz@abc.com’’)`
Send E-mail	Email receiver	`Send_email_to(‘‘xyz@abc.com’’)`
Login to database	Access permission?	`Db_request_without_permission(true/false)`
Log off database	Long login session?	`Too_long_log(true/false)`
Device connect	New device?	`Device_first_connect(true/false)`
Device disconnect	Long connection?	`Connect_too_long(true/false)`
File transfer from device	The file checksum	`Get_file(‘‘2351F8CD57177D4C1044D37460B’’)`
Download from website	The file checksum	`Get_file(‘‘1AFEEF45CB07F87C3D598ED8A5AA’’)`

Similarly, this is a snapshot of some negative examples from actions of users (that are not attacked):

```
[ Db_request_without_permission(false),
  Connect_too_long(false),
  Website_domain("google.com"),
  Get_file("1AFEEF45CB07F87C3D598ED8A5AA"),
  Website_domain("thepostgame.com"),
  Too_short_log(false)]

[ Website_domain("imdb.com"),
  Receive_email_from("Glo34Hugh@hotmail.com"),
  Too_short_log(false),
  Db_request_without_permission(false),
  Connect_too_long(false),
  Connect_too_long(false),
  Send_email_to("ki77r089@gmail.com"),
  Send_email_to("mm63ld@gmail.com")]

[ Receive_email_from("Glo34Hugh@hotmail.com"),
  Too_short_log(false),
  Db_request_without_permission(false),
  Connect_too_long(false),
  Get_file("1AFEEF45CB07F87C3D598ED8A5AA"),
  Send_email_to("ki77r089@gmail.com"),
  Send_email_to("mm63ld@gmail.com")]

[ Receive_email_from("Glo34Hugh@hotmail.com"),
  Too_short_log(false),
  Db_request_without_permission(false),
  Connect_too_long(false),
  Website_domain("google.com"),
  Get_file("1AFEEF45CB07F87C3D598ED8A5AA"),
  Send_email_to("ki77r089@gmail.com"),
  Send_email_to("mml63d@gmail.com"),
  Website_domain("thepostgame.com")]
```

And finally, this is a snapshot of some user actions that need to predict/determine if these are safe:

```
[ Get_file("2351F8CD57177D4C1044D37460B"),
  Website_domain("groupon.com"),
  Website_domain("imdb.com"),
  Receive_email_from("Glo34Hugh@hotmail.com")]

[ Receive_email_from("Kora_Guro@virus.com"),
  Unusual_log_time(true),
  Get_file("1AFEEF45CB07F87C3D598ED8A5AA"),
  Connect_too_long(false),
```

```
  Website_domain("google.com"),
  Send_email_to("mm63ld@gmail.com")]

[ Website_domain("virus.com"),
  Unusual_log_time(true),
  Receive_email_from("Glo34Hugh@hotmail.com"),
  Db_request_without_permission(false),
  Get_file("1AFEEF45CB07F87C3D598ED8A5AA"),
  Send_email_to("ki77r089@gmail.com")]
```

Rules Generated by GOLEM. Based on the example data listed before, the sub-rules generated from GOLEM are shown below. The coverage of each rule is also shown for each clause.

```
ROUND_1
Best_Rule:       [Get_file("2351F8CD57177D4C1044D37460B")]
Best_Cover:      3/6
2nd_Best_Rule:   [Website_domain("virus.com")]
2nd_Best_Cover:  2/6
ROUND_2
Best_Rule:       [Unusual_log_time(true),
                 Website_domain("groupon.com")]
Best_Cover:      2/3
2nd_Best_Rule:  [Website_domain("virus.com")]
2nd_Best_Cover:  1/3
ROUND_3
Best_Rule:       [Receive_email_from("Kora_Guro@virus.com")]
Best_Cover:      1/1
2nd_Best_Rule:  [Receive_email_from("Kora_Guro@virus.com")]
2nd_Best_Cover:  1/1
```

The rule generated for the attack is:

```
[Get_file("2351F8CD57177D4C1044D37460B")]
    OR
[Unusual_log_time(true) AND Website_domain("groupon.com")]
    OR
[Receive_email_from("Kora_Guro@virus.com")]
```

By applying the generated rule to the testing user actions, the result is shown in Table 2.

Table 2. GOLEM result

User	Potential attack	Vulnerable operations
User1	Virus	[Get_file(“2351F8CD57177D4C1044D37460B”)]
User2	Virus	[Receive_email_from(“Kora_Guro@virus.com”)]
User3	N/A	N/A

Rules Generated by UBF-GOLEM. Similarly to GOLEM, we apply UBF-GOLEM to the example data. There are 77 actions in all positive and negative examples. The top 3 frequent operations are: access website (16), send email (10) and get file (10). The rules are generated as follows:

```
ROUND_1
Best_Rule:      [website_domain("virus.com")]
Best_UBFC:      2/6 * 16/77
2nd_Best_Rule:  [Get_file("2351F8CD57177D4C1044D37460B")]
2nd_Best_UBFC:  3/6 * 10/77
ROUND_2
Best_Rule:      [Get_file("2351F8CD57177D4C1044D37460B")]
Best_UBFC:      2/4 * 10/77
2nd_Best_Rule:  [Receive_email_from("Kora_Guro@virus.com")]
2nd_Best_UBFC:  2/4 * 8/77
ROUND_3
Best_Rule:      [Receive_email_from("Kora_Guro@virus.com")]
Best_UBFC:      1/2 * 8/77
2nd_Best_Rule:  [Unusual_log_time(true),
                Website_domain("groupon.com")]
2nd_Best_UBFC:  1/2 * 7/77
ROUND_4
Best_Rule:      [Unusual_log_time(true),
                Website_domain("groupon.com")]
Best_UBFC:      1/1 * 7/77
2nd_Best_Rule:  [Unusual_log_time(true),
                Website_domain("groupon.com")]
2nd_Best_UBFC:  1/1 * 7/77
```

Furthermore, the rule generated for the attack is:

```
[website_domain("virus.com")]
    OR
[Get_file("2351F8CD57177D4C1044D37460B")]
    OR
[Receive_email_from("Kora_Guro@virus.com")]
    OR
[Unusual_log_time(true) AND Website_domain("groupon.com")]
```

Finally we can use the rules generated from UBF-GOLEM to determine the vulnerable actions from each user, as shown in Table 3.

UBF-GOLEM vs GOLEM. There are two advantages by using UBFC instead of a simple coverage for clause selection. Firstly, it considers the weights. The frequent operations should be more important as these operations have more chances to be attacked. Secondly, in GOLEM, if two clauses have the same coverage, the algorithm just chooses the first one. With UBFC, clauses with the same positive coverage can be ordered by UBF and this shall lead to more meaningful selection criteria.

Table 3. UBF-GOLEM result

User	Potential attack	Vulnerable operations
User1	Virus	[Get_file("2351F8CD57177D4C1044D37460B")]
User2	Virus	[Receive_email_from("Kora_Guro@virus.com")]
User3	Virus	[Website_domain("virus.com")]

In the example above, GOLEM generated rules ignore the potentially important sub-rule which is [website_domain("virus.com")]. This is because, for each sample incidence which is covered by more than one clause, only the best-covered one will be selected. For example, [website_domain("virus.com")] is the second best in the first two rounds, but it is ignored because all of its covered examples have been marked as covered by other sub-rules. However, [website_domain("virus.com")] is the most important sub-rule generated by UBF-GOLEM. This is because the website has the highest frequency of user operations, which makes its UBFC value larger than those from other sub-rules.

4.3 Discussions

Compared with other CS systems, our proposal focuses on supporting reasoning and determining relationships between an attack and user operations. Our approach determines the relationship between a threat and the user actions that may cause this threat. Based on the frequency and variety of these user operations, companies may consider adjusting their security policies accordingly. In our two module approach, the user security rating module provides each user a security rating, and the rule mining module is to determine potentially vulnerable operations performed by each user. This gives an idea on which individual or user group and also what operations need to be considered. When an attack happens, it will also be easier to locate and faster response to the attack.

5 Conclusions

User behavior is useful to predict potential attacks based on vulnerable user actions. Much work to date has focused on finding vulnerable operations instead of providing reasoning/explanations of its findings. In this paper, we have presented a transparent learning approach for UBA to address this issue. A user rating system is proposed to determine a security level of each user, with explanations of potential attacks based on his/her vulnerable user actions. A detailed example has been presented to illustrate how approach works. We believe that, with justifiable reasoning from our proposed approach, an organization can be aware of the weakness of its current system, and can better prepare for proactive attack defense or reactive responses.

References

1. Agrawal, R., Imieliński, T., Swami, A.: Mining association rules between sets of items in large databases. In: ACM Sigmod Record, vol. 22, pp. 207–216. ACM (1993)
2. Amiri, F., Yousefi, M.R., Lucas, C., Shakery, A., Yazdani, N.: Mutual information-based feature selection for intrusion detection systems. J. Netw. Comput. Appl. **34**(4), 1184–1199 (2011)
3. Ariu, D., Tronci, R., Giacinto, G.: HMMPayl: an intrusion detection system based on hidden Markov models. Comput. Secur. **30**(4), 221–241 (2011)
4. Asenjo, P.E.R.: Web user behavior analysis. Ph.D. thesis, Universidad De Chile (2011)
5. Baum, L.E., Eagon, J.A., et al.: An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology. Bull. Amer. Math. Soc. **73**(3), 360–363 (1967)
6. Bilge, L., Kirda, E., Kruegel, C., Balduzzi, M.: EXPOSURE: finding malicious domains using passive DNS analysis. In: National Diabetes Services Scheme (NDSS) (2011)
7. Bilge, L., Sen, S., Balzarotti, D., Kirda, E., Kruegel, C.: EXPOSURE: a passive DNS analysis service to detect and report malicious domains. ACM Trans. Inf. Syst. Secur. (TISSEC) **16**(4), 14 (2014)
8. Bivens, A., Palagiri, C., Smith, R., Szymanski, B., Embrechts, M., et al.: Network-based intrusion detection using neural networks. Intell. Eng. Syst. Artif. Neural Netw. **12**(1), 579–584 (2002)
9. Brahmi, H., Brahmi, I., Ben Yahia, S.: OMC-IDS: at the cross-roads of OLAP mining and intrusion detection. In: Tan, P.-N., Chawla, S., Ho, C.K., Bailey, J. (eds.) PAKDD 2012. LNCS (LNAI), vol. 7302, pp. 13–24. Springer, Heidelberg (2012). doi:10.1007/978-3-642-30220-6_2
10. Buczak, A.L., Guven, E.: A survey of data mining and machine learning methods for cyber security intrusion detection. IEEE Commun. Surv. Tutor. **18**(2), 1153–1176 (2015)
11. Cannady, J.: Artificial neural networks for misuse detection. In: National Information Systems Security Conference, pp. 368–81 (1998)
12. Cohen, W.W.: Fast effective rule induction. In: Proceedings of the Twelfth International Conference on Machine Learning, pp. 115–123 (1995)
13. Digman, J.M.: Personality structure: emergence of the five-factor model. Annu. Rev. Psychol. **41**(1), 417–440 (1990)
14. Han, H., Lu, X.L., Ren, L.Y.: Using data mining to discover signatures in network-based intrusion detection. In: Proceedings of International Conference on Machine Learning and Cybernetics, vol. 1, pp. 13–17. IEEE (2002)
15. Jemili, F., Zaghdoud, M., Ahmed, M.B.: A framework for an adaptive intrusion detection system using Bayesian network. In: ISI, pp. 66–70 (2007)
16. Joshi, S.S., Phoha, V.V.: Investigating hidden Markov models capabilities in anomaly detection. In: Proceedings of the 43rd Annual Southeast Regional Conference, vol. 1, pp. 98–103. ACM (2005)
17. Kruegel, C., Mutz, D., Robertson, W., Valeur, F.: Bayesian event classification for intrusion detection. In: Proceedings 19th Annual Computer Security Applications Conference, pp. 14–23. IEEE (2003)
18. Kruegel, C., Toth, T.: Using decision trees to improve signature-based intrusion detection. In: Vigna, G., Kruegel, C., Jonsson, E. (eds.) RAID 2003. LNCS, vol. 2820, pp. 173–191. Springer, Heidelberg (2003). doi:10.1007/978-3-540-45248-5_10

19. Lee, W., Stolfo, S.J., Mok, K.W.: A data mining framework for building intrusion detection models. In: Proceedings of the 1999 IEEE Symposium on Security and Privacy, pp. 120–132. IEEE (1999)
20. Li, Y., Xia, J., Zhang, S., Yan, J., Ai, X., Dai, K.: An efficient intrusion detection system based on support vector machines and gradually feature removal method. Expert Syst. Appl. **39**(1), 424–430 (2012)
21. Lippmann, R.P., Cunningham, R.K.: Improving intrusion detection performance using keyword selection and neural networks. Comput. Netw. **34**(4), 597–603 (2000)
22. Michalski, R.S.: A theory and methodology of inductive learning. In: Michalski, R.S., Carbonell, J.G., Mitchell, T.M. (eds.) Machine Learning. Symbolic Computation, pp. 83–134. Springer, Heidelberg (1983)
23. Muggleton, S., Feng, C., et al.: Efficient Induction of Logic Programs. Turing Institute (1990)
24. Norton, M., Roelker, D.: SNORT 2.0: Hi-performance multi-rule inspection engine. Sourcefire Network Security Inc (2002)
25. Panda, M., Patra, M.R.: Network intrusion detection using naive bayes. Int. J. Comput. Sci. Netw. Secur. **7**(12), 258–263 (2007)
26. Pfleeger, S.L., Caputo, D.D.: Leveraging behavioral science to mitigate cyber security risk. Comput. Secur. **31**(4), 597–611 (2012)
27. Plotkin, G.: Automatic methods of inductive inference. Ph.D. thesis, The University of Edinburgh (1972)
28. Plotkin, G.D.: A further note on inductive generalization. In: Machine Intelligence, vol. 6, pp. 101–124. Edinburgh University Press (1971)
29. Quinlan, J.R.: Induction of decision trees. Mach. Learn. **1**(1), 81–106 (1986)
30. Quinlan, J.R.: C4. 5: Programs for Machine Learning. Elsevier, Amsterdam (2014)
31. Raiyn, J., et al.: A survey of cyber attack detection strategies. Int. J. Secur. Appl. **8**(1), 247–256 (2014)
32. Reiss, F.: Transparent Machine Learning for Information Extraction: State-of-the-Art and the Future (2015). http://www.emnlp.2015.org/tutorials/15/15_OptionalAttachment.pdf
33. Udantha, M., Ranathunga, S., Dias, G.: Modelling website user behaviors by combining the EM and DBSCAN algorithms. In: 2016 Moratuwa Engineering Research Conference (MERCon), pp. 168–173. IEEE (2016)
34. Uma, M., Padmavathi, G.: A survey on various cyber attacks and their classification. Int. J. Netw. Secur. **15**(5), 390–396 (2013)
35. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann, Burlington (2005)
36. Zhengbing, H., Zhitang, L., Junqi, W.: A novel network intrusion detection system (NIDS) based on signatures search of data mining. In: First International Workshop on Knowledge Discovery and Data Mining (WKDD), pp. 10–16. IEEE (2008)