International Journal of Computer Science and Engineering (IJCSE)
ISSN(P): 2278-9960; ISSN(E): 2278-9979
Vol. 3, Issue 3, May 2014, 13-22

International Academy of Science, Engineering and Technology
Connecting Researchers; Nurturing Innovations

CONDITIONAL NAIVE-BAYES TO DETECT MASQUERADES

K. VENKATESWARA REDDY[1] & N. PUSHPALATHA[2]

[1]Principal, Marri Laxman Reddy Institute of Technology & Management, Hyderabad, Andhra Pradesh, India

[2]Assistant Professor, Department of CSE, Marri Laxman Reddy Institute of Technology & Management, Hyderabad, Andhra Pradesh, India

ABSTRACT

An intruder masquerades as the legitimate user to obtain authorization on legitimate users account. These Masquerades has to be detected as early as possible, efficiently to reduce the malicious work done on that account. This paper proposes a novel approach for detecting the masquerades, which is not only efficient and also accurate than the earlier methods. Here we concentrate on the deviations of user profile with the trained profile, which are different base on duration of deviation for legitimate user and masquerades. Using this concept we extended the work done by Roy A Maxion [1] to device a Conditional Naïve Bayes method. Experiments on a standard datasets demonstrate that this Conditional Naïve Bayes classifier beats the previous best performing detector and reduces the missing-alarm rate and increases the hit rate.

KEYWORDS: Masquerades, Conditional Naïve Bayes Method, SEA Configuration and 1v49 Configuration

1. INTRODUCTION

One of the prime challenges to computer security is the masquerade attack, where an illegitimate entity poses as (and assumes the identity of) a legitimate entity. The illegitimate user, called masquerader, hides his/her identity by impersonating a legitimate user in a computer system or network and may maliciously damage the system. A masquerader can either be an insider with malicious intent, trying to hide his/her identity by impersonating other users, or an outsider, who generally tries to gain access to the account of the super-user. Masquerade attack can occur in a variety of ways, such as by obtaining a legitimate user's password, accessing an unattended and unlocked workstation, forging email address in messages and overtaking a computer via a network access. It is difficult to detect this type of security breach at its initiation, because the attacker appears to be a normal user with valid authority and privileges.

The detection of a masquerader relies on a user signature, in most cases, a sequence of commands collected from a legitimate user. The underlying assumption is that the signature captures detectable patterns in a user's sequence of commands. Each time, the current user's session is compared to this signature. A sequence of commands produced by the legitimate user should match well with patterns in the user signature, whereas a sequence of commands entered by a masquerader should match poorly with the user's signature. The detection becomes difficult when the masquerader perfectly mimics original user's behavior. There is also a chance that the legitimate user may be detected as a masquerader if the user's behavior changes, thereby causing annoying false alarms.

The user is not always stereotyped to consistently follow the behavioral pattern. He/she occasionally deviates from his/her own pattern but such deviations would only be temporary. On the other hand, a masquerader would be expected to deviate substantially from the behavioral pattern of the legitimate user. Thus the deviation would last for a

longer duration. Thus any detection technique that simply matches the user signature with the set of commands entered by the questionable user would invariably raise a false alarm even when the legitimate user deviates momentarily from his/her pattern.

This momentary deviation in user behavior has been termed in literature as user concept drift. We, in this paper, propose to identify a block of commands into three categories-legitimate, doubtful and masquerade. If a block does not match with the corresponding user signature then we designate the block as doubtful and wait for subsequent blocks. If the doubtful status continues for 2 to 3 consecutive blocks then it is justified to conclude that the deviation is due to masquerading.

Moreover, when one block is already doubtful, the subsequent block has to pass a more stringent legitimacy test. This is because once we know that a block of commands is a questionable block, the block following it would have a higher likelihood of being doubtful. The block size can be suitably defined in the context of the application so that the period of 2 to 3 blocks is not late enough to detect a masquerade. Based on this principle, we propose our algorithm, called the Conditional Naïve bayes Technique. We have experimented our algorithm with three standard datasets and demonstrate that the results of this method are the best reported so far.

In section 2, we briefly outline the existing techniques of masquerade detection. In section 3, we discuss about the standard datasets, which are used for performance evaluation of this attack. Our new algorithm is outlined in section 4. Section 5 is concerned with the experimental details and results.

2. EARLIER WORK

In this section, we review different known techniques proposed for the purpose of masquerade detection. Schonlau et al., in [12], study various masquerade detection methods. These are Bayes 1-Step Markov, Hybrid Multi-Step Markov, Incremental Probabilistic Action Modeling (IPAM), Uniqueness, Sequence-Match, and Compression. Naive Bayes is popularly used in text classification and Maxion and Townsend [1] demonstrated that it is very efficient in this context as well. In [10] the same author shows that valuable information is lost when truncated command line data is used and proposes to use enriched command line data, which yields better results than the earlier dataset of truncated commands.

Table 1: Results from Previous Approaches to Masquerade Detection

Method	Hit Rate	False +ve
SVM (1v49 configuration)	94.8%	0.0%
SVM (SEA configuration)	80.1%	9.7%
Semi-Global Alignment	75.8%	7.7%
Recursive Data Mining	75.0%	10.0%
Bayes one-step Markov	69.3%	6.7%
Naive Bayes (no updating)	66.2%	4.6%
POMDP	63.8%	1.0%
Naive Bayes (updating)	61.5%	1.3%
Hybrid Markov	49.3%	3.2%
IPAM	41.1%	2.7%
Uniqueness	39.4%	1.4%
Sequence Matching	36.8%	3.7%
Compression	34.2%	5.0%

In [8], Coull et al. propose a novel technique based on pair-wise sequence alignment. In [15], it is proposed to profile a user by modeling his data exclusively, without using examples from other users. Recently, in [5], a new and efficient masquerade detection technique based on SVM is proposed which is based on two novel concepts of common commands and voting engine. Szymanski and Zhang [14] propose a method, Recursive Data Mining, which recursively mines a string of symbols by finding frequent patterns, encoding them with unique symbols and rewriting the string using this new coding. In [7], Lane proposes a model of intrusion detection based on semi-supervised learning. He adopts a partially ordered Markov Decision Process (POMDP) for user profiling. The performances of the methods discussed above are summarized in Table 1.

3. DATASETS

At this stage, it is apt to study the datasets that are considered to be standard for the study of command line based attacks. There are three popular datasets available for studying the characteristics of masquerade attacks. One is provided by Schonlau et al. [12], which is a truncated command dataset, commonly called as Schonlau dataset. Most of the techniques described above use this dataset. This data is collected by the UNIX acct auditing mechanism and consists of 15,000 "truncated" commands of 50 users. In addition to using first 5000 commands of all users as the training data and the remaining 10000 for testing, (termed as the SEA Configuration) Maxion [1] proposes another configuration called 1v49 configuration. In this configuration, the first 5,000 commands are used to build a model, and the first 5,000 commands entered by each of the rest of group, 49 users, are used as test data. This is contrary to SEA configuration where each user's last 10,000 commands containing simulated masquerade blocks are used as test data.

Lane and Brodley [6] generated another dataset, which contains 9 sets of sanitized user data drawn from the command histories of 8 UNIX computer users over the course of up to 2 years. The data is drawn from tcsh history files and has been parsed and sanitized to remove filenames, user names, directory structures, web addresses, host names, and other possibly identifying items. Another dataset collected by Greenberg [3], contains data of 168 different users of University of Calgary. The users are divided into 4 groups – Novice Pro- grammars, Experienced Programmers, Computer Scientists and Non-programmers according to their computer experience and needs. The complete command line data submitted by the user was captured as a chunk after it has been entered and processed by UNIX csh command interpreter. It also includes extra information known to csh, including the use of history and alias and the current working directory of the users. Login sessions are distinguished by a record that notes the start and end time of each session.

4. ALGORITHM

We observe from the foregoing discussion that the performances of these algorithms are not satisfactory as the false positive rate is beyond the acceptable limits. Moreover, the performances of the algorithms are evaluated mostly on Schonlau data and on two models namely, SEA configuration and 1v49 configuration. Both the configurations evaluate the performance based on the command blocks. For instance, the Schonlau data provides 50 blocks of 100 commands each for each user for the training data, and 100 blocks of 100 commands each for the test data. Inserting an illegitimate block generates the masquerade data. Thus, in a sense the behavior of the algorithm is tested in a very specific fashion that is guided by the nature of the standard dataset.

We also know that the behavior of the masquerader cannot be identified just by one single command or instantaneously. Even a legitimate user may deviate from his normal behavior for sometime. Thus, it is more appropriate to

doubt a user who is consistently deviating from the normal signature rather than a user who deviates momentarily and returns to the normal pattern. We formalize this particular concept as follows. Let us assume that a block b1 of commands is detected as masquerade by any of the detection algorithms. Let us also assume that the block b0 preceding this block is detected as legitimate. When we are testing block b2 which immediately follows b1, it is justified to believe that b2 is more likely to be a masquerade, and hence we put more stringent condition on b2 to pass the legitimacy test. In case b2 successfully passes the legitimacy test, then based on our knowledge that b0 and b2 are legitimate, b1 is labeled as legitimate in spite of its failure to pass legitimacy test. Similarly, if b2 is identified as masquerade block (it fails to pass the stringent legitimate test), then the block b3 which immediately follows b2, is subjected to still harder legitimacy test.

In the context of Naive Bayes algorithm this condition can be implemented by changing the threshold progressively. One may argue that a block of 100 commands by itself is long enough and detecting a masquerade after two or three consecutive blocks (200 to 300 commands) can prove to be too late for online masquerade detection. But in the real life applications, the block size can be decided to be reasonably small depending on the context. In the case of real situation, if it is known that the smallest corrupted command string is of size k, then the block size can be chosen to be reasonably smaller than k i.e. k/4 or k/5. Motivated by this observation, we propose here a new algorithm for masquerade detection. We make use of Naive Bayes algorithm given in [8], and use a progressively varying threshold for consecutive masquerade blocks Let C be the set of unique commands. Let Cj be the set of unique commands used by the user uj during training and f(c, uj) be the frequency with which the command c is entered by the user uj during training. We define the following probability,

$$P(c,u_j) = \frac{f(c,u_j) + \alpha}{\sum c_j \in c_j f(c,u_j) + \alpha|C|}$$

Similarly, we define probability of use of command c for a group of users, U, as

$$P(c,U_{\neq j}) = \frac{\sum u_{k \in U \wedge k \neq j} f(c,u_k) + \alpha}{\sum u_{k \in U \wedge k \neq j} \sum c_{i \in c_j} f(c_i,u_k) + \alpha|C|}$$

Where α is a pseudo count, and | C | is the number of distinct commands in the data. The pseudo count can be any real number larger than zero (0.01 in this study), and is added to ensure that there are no zero counts. The lower the pseudo count, the more sensitive the detector is to previously unseen commands. The pseudo-count term in the denominator compensates for the addition of a pseudo count in the numerator.

For a given block of commands, b, let us define

$P(b, u_j) = \prod P(c,u_j)$

We then define self and non-self probabilities as

$P_{self}(b, u_j) = P(c,u_j)$

$P_{non-self}(b,u_j) = P(c, U_{\neq j})$

where U 6 =j is the set of all users except uj .

As in [8], the ratio of self and non-self is taken as

$$\text{Score}(b,u_j) = \frac{P_{self}(b, u_j)}{P_{non-self}(b,u_j)}$$

In traditional Naive Bayes method [8], a block b is determined to be masquerade or legitimate depending on whether Score(b, uj) falls below a threshold λ or exceeds the threshold respectively.

In the proposed algorithm, a block is detected as legitimate, doubtful or masquerade based on the corresponding status of two blocks preceding it. Let us assume that we are considering the block b2 and the two preceding blocks are b0 and b1. We determine the status of this block by the following set of tests. It may be noted that we often look back to revise the status of some of the earlier blocks.

- If b_0 is legitimate then b_1 is marked legitimate only

  if Score(b_1,u_j) ≥ λ, else, b_1 is marked doubtful.

- If b_0 is legitimate and b_1 is doubtful, then b2 is marked legitimate

  if Score(b_2, u_j)≥ λ + λ/4 , or else, it is marked doubtful.

- If b2 is marked legitimate by the above test, then the marking of b1 is revised to legitimate.

- If b0 is doubtful and b1 is doubtful, then b2 is marked legitimate

  if Score(b_2, u_j) ≥ λ'+ λ'/4 , where λ' =λ + λ'4 , or else, it is marked masquerade.

- If b2 is marked masquerade by the above test then the markings of b0 and b1 are revised to masquerade.

- If b2 is marked legitimate by the above test then the markings of b0 and b1 are revised to legitimate.

5. EXPERIMENTAL RESULTS

We experimented our algorithm with three existing standard datasets namely Schonlau dataset [12], Lane dataset [6] and Greenberg dataset [3]. Experiments with different datasets are as follows.

5.1 Schonlau Dataset

We experimented our algorithm with the Schonlau dataset for different threshold values. Figure 1 summarizes the experimental results as the ROC curve. It can be seen from the ROC curve in Figure 1 that, except for two cases POMPD and Multinomial updating, points corresponding to all earlier experiments lie below the curve of the Conditional Naive Bayes Technique. Thus it can be seen that our algorithm out performs all the algorithms except these two methods. Even in these cases, the detection rate is much below 70% whereas the detection rate of our algorithm is above 80%. The best result that we obtained is with the detection rate of 83.9% with a false alarm rate of 8.8%. None of the earlier algorithms has achieved this level of accuracy so far.

Table 2: Results for Conditional Naive Bayes Method Taking Block Size of 100, 50 and 25

Test Configuration	Hit Rate	False +ve
SEA Configuration with block size 100	84.0%	8.8%
SEA Configuration with block size 50	85.3%	8.5%
SEA Configuration with block size 25	85.2%	10.6%
1v49 configuration	90.7%	1.0%

We have also tested our algorithm taking block sizes of 100, 50 and 25 commands, results of which is depicted in table 2. Figure 2 gives a comparison of the performances of our algorithm when tested using various block sizes. One interesting point is that although the block size is reduced to 50 and 25, the performance of the algorithm doesn't degrade alarmingly. In fact, the algorithm performs best when the block size is 50, as depicted in the figure. This means that now a masquerader can be identified in just the time of 150 commands as opposed to 300 when using a block size of 100. When using a block size of 25, though the performance is not as good as when using block size 100/50, it can be observed that the three curves are very close to each other. Thus, when using block size as 25, we can detect a masquerade in just 75 commands, which is quicker than what we can do using other methods.

We have experimented our algorithm with the 1v49 configuration also. In this configuration, the algorithm could detect masquerades at 90.7% accuracy with a false alarm of 1.0%.

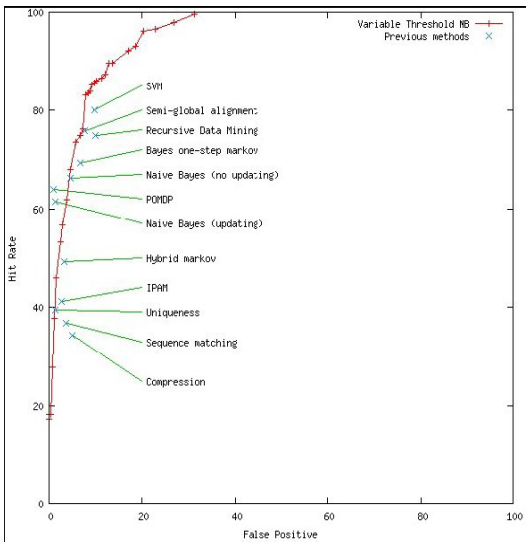

Figure 1: ROC Curve of the Conditional Naïve Bayes Algorithm and Comparisons with Other Algorithms

Lane Dataset

As described in section 3, this contains normal data of 9 users only. As the number of users is less, we go for a similar kind of approach as 1v49 configuration. Each user's own data is used for training and all the other users' data are used for testing. In this case we got a detection rate of 99.68% with 0% false positive. As the numbers of commands of each user are different, so we have taken only 4891 commands of each user (minimum number of commands available in each user's data) into consideration.

Greenberg Dataset

As described in section 3, it consists of 168 users' data. We process the raw data to get clean dataset for each user, a sample of which is shown in table 3. If alias is present then <#> is used as the separator between them. The total of 168 users are divided into two sets as victim (V) and masquerade (M) as follows.

Initialize: V = {}, M = {}

for j=1 to 168

if (#commands(Uj) ≥ 2000 and

#commands(Uj)< 5000) then

V = V [{Uj}

else

M = M [{Uj}

end if

end for

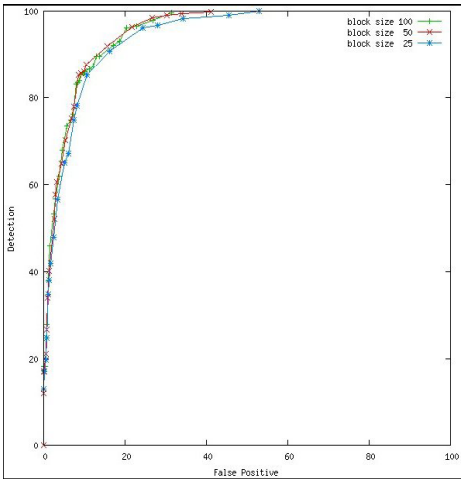

Figure 2: ROC Curve of Conditional Naïve Bayes Method with Comparison Using Various Block Sizes

Coincidentally, the number of users in the victim set comes to be 50, which is same as the number of users in the Schonlau dataset. To have a greater diversity, all the remaining 118 users (=|M|) are taken as source of masquerader commands. The data for the victim users are truncated to 2000 commands out of which 800 commands are used for training purpose. The remaining 1200 commands are divided into 40 blocks of 30 commands each. 10 blocks of masquerade data is inserted into the test data, for which, 10 places are selected randomly from the possible 41 places. For a victim, if same place is selected more than once then it indicates consecutive occurrence of masquerade blocks. Selecting a continuous stream of commands from the same masquerade user represents occurrence of consecutive masquerade blocks. Masquerade blocks are randomly selected from the command pool of M and inserted at the selected places. So at the end, the test data consists of 1500 commands logically grouped into 50 blocks.

Table 3: Shows the Raw Data and its Corresponding Processed from the Greenberg Dataset

Raw Data	Processed Data
C bye D /user/cpsc500/101b91/**** A logout H NIL X M 09	Bye<#>logout
C rwho | more D /user/cpsc500/101b91/**** A NIL H NIL X NIL	Rwho | more

As our method delays the declaration of masquerade by 3 blocks, we take the block size to be 10 (1/3rd of the original block size) during the testing phase. So the new scenario is, the test data consists of 150 small blocks out of which 30 are masquerades. Our algorithm achieves a hit rate of 84.13% with a false positive of 9.4%. We have also experimented our algorithm with 1v49 configuration on this dataset, which gives a detection rate of 97.38% with a false positive of 0%.

CONCLUSIONS

In this paper, we propose a new algorithm for masquerade detection. Our approach is based on a kind of deferred detection, which marks a block to doubtful, and once we get a few doubtful consecutive blocks we detect these as masquerade blocks. Moreover, as we encounter more and more doubtful blocks we impose more and more stringent condition of legitimacy test. We show that our method gives exceptionally promising accuracy. For SEA configuration its performance is better than all other known methods. For the 1v49 configuration, it is comparable with one of the recent methods [5], which uses SVM for detection. Our method is extremely simple to implement and more efficient. Moreover, our method is more realistic in the sense that it does not generate false alarm for any momentary deviations, which may be a accounted for user concept drift.

REFERENCES

1. Maxion, R. A., Townsend, T. N.: Masquerade Detection Using Truncated Command Lines. In Proceedings of the International Conference on Dependable Systems and
2. Davison, B. D., Hirsh, H: Predicting sequences of user actions. In Predicting the Future: AI Approaches to Time-Series Problems, papers from the 1998 AAAWorkshop,27 July 1998, Madison, Wisconsin, pp. 5-12. AAAI Technical Report WS-98-07, AAAI Press, Menlo Park, California, 1998
3. Greenberg, S.: Using Unix: Collected traces of 168 users. Technical Report88/333/45, Department of Computer Science, University of Calgary, Calgary, Canada, 1988
4. Killhourhy, K. S., Maxion, R. A.: Investigating a possible flaw in a masquerade detection system. Technical Report CS-TR: 869, School of Computing Science, University of Newcastle, Nov 2004
5. Kim, H.-S., Cha, S.-D.: Empirical evaluation of SVM-based masquerade detection using UNIX commands. Computers & Security, Vol. 24, 160-168, March, 2005

6. Lane, T., Brodley, C. E.: Temporal Sequence Learning and Data Reduction for Anomaly Detection. In Proceedings of the Fifth ACM Conference on Computer and Communications Security, 3-5 November 1998, pp. 150-158, San Francisco, California
7. Lane, T.: A Decision-theoretic, semi-supervised model for intrusion detection. Technical Report TR-CS-2004-16, University of New Mexico, 2004.
8. Coull, S., Branch, J., Szymanski, B., Breimer, E: Intrusion detection: A bioinformatics approach. In 19th Annual Computer Security Applications Conference, LasVegas, Nevada, December 8-12 Networks (DSN-02), 219-228, 23-26, June 2002 Washington, D.C. IEEE Computer Society Press, Los Alamitos, California
9. Maxion, R. A., Townsend, T. N.: Masquerade detection augmented with error analysis. IEEE Transactions on Reliability, 53(1), 124-147, March 2004
10. Maxion, R. A.: Masquerade Detection Using Enriched Command Lines. In International Conference on Dependable Systems and Networks (DSN-03), San Francisco, CA, USA, June 2003
11. McCallum, A., Nigam, K: A comparison of event models for Naive-Bayes text classification. In AAAI-98 Workshop on Learning for Text Categorization, 1998, Madison, Wisconsin
12. Schonlau, M., DuMouchel, W., Ju, W., Karr, A. F., Theus, M., Vardi, Y: Computer Intrusion: Detecting Masquerades. Statistical Science, 16(1): 58-74, February 2001
13. Schonlau, M., Theus, M.: Detecting masqueraders in intrusion detection based on unpopular commands. In Information Processing Letters, 76(1-2):33-38, November 2000
14. Szymanski, B. K., Zhang, Y.: Recursive data mining for masquerade detection and author identification. In Proceedings of the 5th IEEE System, Man and Cybernetics Information Assurance Workshop, West Point, NY, June 2004

AUTHOR'S DETAILS

Dr K Venkateswara Reddy is a principal of MLRITM, Hyderabad, received his M. Tech and Ph.D from JNTU and Osmania University. He checquered a dynamic career in reputed engineering colleges as a professor, Head and Vice-principal and is found to be disciplinary and dynamic personality in academic and administrative spheres. He bagged several national and international journals to his credit in 20 years length of his service. He is life member of ISTE. Presently he is working in Cloud Computing, Network Security, MANET and other emerging fields of computer science.

N. Pushpa Latha working as a Assistant professor in Marri Laxman Reddy Institute of Technology and Management, Hyderabad. She has 6+ years teaching experience and good knowledge in computer subjects. She completed master degree in computer science and engineering dept. from University College of Engg, JNTU Campus, Kakinada. Presently pursuing Ph. D from JNTU, Hyderabad.