Modeling User Behaviors from FTP Server Logs

Yeming Hu, A. Nur Zincir-Heywood
Dalhousie University, Faculty of Computer Science
hu@dal.ca, zincir@cs.dal.ca

Abstract

In this paper, a modeling toolkit is proposed for modeling user behavior from FTP server log files. This toolkit can develop analytical models from the data at hand with minimum assumptions. Analytic models are intended to be data driven, which means users are not required to be experts on mathematics or statistics. Moreover, the toolkit provides a simple yet practical way to generate simulated traffic from the analytic models.

1. Introduction

Computer hackers are always one step ahead in finding security holes in current systems. Therefore dynamic defense mechanisms such as intrusion detection systems (IDSs) should be deployed to fortify the networks and the connected hosts. Current IDSs are far from being perfect; therefore systematic analysis and benchmarking of IDSs can provide researchers valuable information that can lead to improvements. Obtaining test data for IDS benchmarking is not a straightforward task. In general, two approaches are available: (1) First approach is to employ the traffic captured from a live network for benchmarking. However, due to disk space limitations and privacy concerns, captured data cannot be shared. Without the careful analysis of the captured data, benchmarking effort reveals very little about the performance of IDSs. (2) Second approach is to generate synthetic data by performing simulations. Simulations are based on analytic models; therefore researchers can conduct further analysis on the performance of the IDSs. Furthermore, since the synthetic data does not contain any sensitive information, benchmarking details can be publicly shared thus creating a certain degree of repeatability. Because of its advantages, majority of the IDS benchmarking efforts such as methodology developed by Puketza *et. al* [1], IBM IDS testing workbench [2] and MIT DARPA 99 corpus [3] adopt synthetic data approach. The most important shortcoming of these efforts is that benchmarking was regarded as a "black box" process and tools or frameworks employed in the benchmark were not made available to other researchers. In Kayacik *et. al.* [4] authors proposed an open framework for generating synthetic data. They believed that such a framework would be useful to IDS researchers to generate normal behavior to benchmark IDS systems. Our objective in this work is using Kayacik *et. al.*'s framework to develop a toolkit for modeling normal user behavior from File Transfer Protocol (FTP) server log files.

2. Modeling user behavior

The proposed framework in [4] composed of five components. In our work, we employed the five components as follows: First component constructs ftp session from the ftp server log files. Second component builds the probabilistic models of file download/upload transitions. Third component utilizes the probabilistic models to build synthetic sessions. Fourth component validates the synthetic sessions in terms of sequence similarity and transition delay similarity. Finally fifth component employs the synthetic sessions and uses an ftp server to generate traffic.

2.1. Discrete Markov Models

In this work, a Markov Model [5] based approach is employed to model the user behavior. Markov Models are useful to build the probabilistic models of event sequences evolving in time. In a first order Markov model, next state is dependent on only the current state. A Markov model is defined by the state space, I, transition probability matrix, P, and the probability distribution of the states, λ.

Let I be a countable set of states, where $i, j \in I$ represent states. If there are N states in I, a first order Markov model can be represented as a two-dimensional N x N matrix $P = (p_{ij} \mid i, j \in I)$. Let X_t define the state at step t. From the training data, probability of transition from state i to j p_{ij} can be calculated as follows;

$$p_{ij} = \frac{C(X_t = j, X_{t-1} = i)}{\sum_i C(X_{t-1} = i)}$$

$C(X_t = j, X_{t-1} = i)$ is the number of times state j follows state i in the training data. The i^{th} row of the P is the probability distribution of moving to all states from state i. This probability distribution is also called λ_i.

2.2. Methodology

Depending on the OS used and the software used, FTP log files can take different forms. In this work, we used

the FTP log files of the FTP server on one of the Dalhousie FCS Sun unix servers. In this case, we had two files, namely: (i) ftp_auth and (ii) ftp_trans.

ftp_auth file is used to collect the timestamp (session) and username information and ftp_trans file is used to get the operation (upload, download or delete) of the corresponding user at the given timestamp.

It should be noted here that this FTP server we modeled does not permit anonymous connections but instead each user has his/her username and password on the server and therefore, can only access his/her own space.

2.3. Building transition models

In this work, it is assumed that the order the users connect to the FTP server is random but the operations each user performs in a session have relationships with each other. Thus, Markov models are employed to model this relationship. In other words, each operation, a user performs, is considered as a state. In addition, a special purpose state is introduced to start and end a session.

Transition probabilities in matrices P and Q are utilized to generate sessions based on the HMM probabilistic models. The first matrix is composed of the probability of transferring from one state to another state. The other one is composed of the probability of transferring from one state to another with different transition delays. Each new session starts from the start state and the next state is selected stochastically until the end state is reached. Figure 1 summarizes the session generation algorithm.

```
State index i = 0
Set state s_i = START
Set delay d_i = 0
Loop Until s_i = END
    Increment i
    Probability proportional selection
    of s_i and d_i
End Loop
```

Figure 1. Pseudo-code session generation

Probability proportional selection in Figure 1 implies that if a state has a high probability of being the next state, it will have more chance. This is achieved by converting all probability distribution functions to cumulative distribution functions. A cumulative distribution function (CDF) is the summation of a probability distribution function (PDF). CDFs monotonically converge to 1 (i.e. it increases up to 1 but never decreases). After conversion, each state will have a probability slot allocated to it within the CDF; therefore slots will be allocated between 0 and 1. The higher the probabilities, the larger slots allocated. Probability proportional selection involves generating a random number between 0 and 1 and determining, which slot the random number belongs to. The state, which the slot belongs to, is selected as the next state. Process continues until the generated random number belongs to end state.

2.4. Validating sessions

In this work, we develop a toolkit in order to generate synthetic data that represents a given normal behavior (i.e. a server log file). Once, such a data set is generated, it can be used for training/testing machine learning based intrusion detection systems as well as benchmarking any intrusion detection system.

As discussed in section one, there are other efforts in the literature in order to generate such benchmarking data sets. However, the lack of validation methods is one of the shortcomings of those previous intrusion detection system benchmarking efforts. Thus, in this work, in order to validate the synthetic data we generate, we will compare it against the original data based on the operation characteristics, i.e. session counts and session lengths.

3. Results

In our experiments, we used the ftp server log files of one of the Unix hosts at Dalhousie University, Faculty of Computer Science. Since this file was small with 1700 user sessions. Only 1700 synthetic sessions were generated synthetically for this data set. Five criteria have been used to express results: Mean, median, number of actions, number of sessions and session duration. Number of actions is further divided into three parts: Number of Upload actions, Number of Download actions and Number of Deletion actions.

As seen in figure 2 and table 1 synthetic data represents the original data set very similarly. However, it should be noted here that there is one user on this ftp server, which creates half of the actions in the log file. Thus, this affects some of the statistics of the original data file.

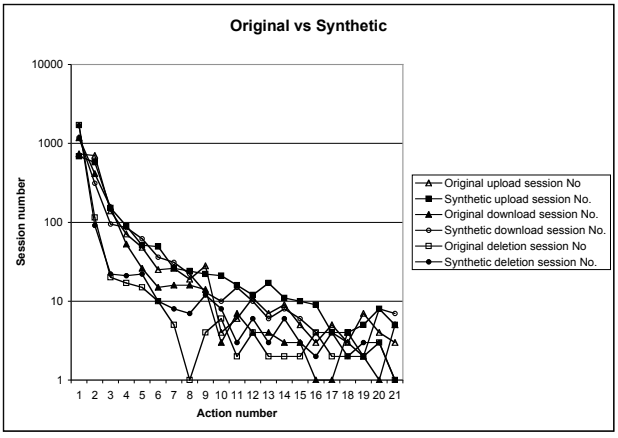

Figure 2. Comparison of original data vs. synthetic data in terms of actions (log-scale)

Table 1. Comparison of the original data set vs. the synthetic data set in terms of mean and median

	Original median	*Original mean*	*Synthetic median*	*Synthetic mean*
Session counts	2	58.8203	3	55.9622
Session icounts	1	52.6432	1	50.9137
Session ocounts	0	3.49413	0	2.55079
Session dcounts	0	2.683	0	2.4977
Session duration	0	165.375	1	161.334

In both the figure and the table above, Session counts means the number of actions in one session; Session icounts is the number of upload actions in one session; Session ocounts is the number of download actions in one session; Session dcounts is the number of deletion actions in one session; and Session duration is the length of the session (in seconds).

4. Discussion

In this paper, we presented a probabilistic modeling approach to build user models from the ftp server log files. Our contribution is the distinctive use of Discrete Markov models to model the user behavior from the ftp server logs. To the best of our knowledge, this is the first attempt to model ftp server log files. The set of components that we developed within the framework comprise a comprehensive ftp toolkit, which can develop and validate usage models and generate synthetic data. Furthermore, the results show that this toolkit can be used to model user behavior from ftp log files since the synthetic sessions are comparable to the sessions in the original data. Thus, our system can be used to generate normal behavior in order to train/test machine learning based IDSs or used as a benchmarking data set for any given IDS. Given the difficulty of obtaining real traffic for benchmarking IDSs, we believe this work performs a valid contribution. For future work, we want to expand the toolkit so that it can be used on any type of ftp server log file as well as test it on more data sets.

References

[1] Puketza N.J., Zhang K., Chung M., Mukerjee B., "A Methodology for Testing Intrusion Detection Systems", In IEEE Transaction on Software Engineering v 22 no 10, pp 719-729, Oct 1996. http://citeseer.ist.psu.edu/puketza96methodology.html

[2] Debar, H., Dacier, M., Wespi, A. and Lampart, S. 1998. "An experimentation workbench for intrusion detection systems", Res. Rep. RZ 2998 (#93044) (Sept.). Research Division, IBM, New York, NY.

[3] Haines J. W., Lippmann R. P., Fried D. J., Tran E., Boswell S., Zissman M. A., "1999 DARPA Intrusion Detection System Evaluation: Design and Procedures", MIT Lincoln Laboratory Technical Report, http://www.ll.mit.edu/IST/ideval/pubs/2001/TR-1062.pdf

[4] Kayacik H. G., Zincir-Heywood A. N., "Generating Representative Traffic for Intrusion Detection System Benchmarking", Proceedings of the IEEE CNSR'2005, pp.112-117 Halifax, Canada, May 2005.

[5] Rabiner L. R., Juang B. H., "An Introduction to Hidden Markov Models", IEEE ASSP Magazine, vol. 3, issue 1, pp.4-15, Jan 1986.