2009 First International Conference on Advances in P2P Systems

Anomaly Detection in P2P Networks Using Markov Modelling

J. Díaz-Verdejo, G. Maciá-Fernández, P. García-Teodoro, J. Nuño-García
Department of Signal Theory, Telematics and Communications – CITIC-UGR
Faculty of Computer Science and Telecommunications - University of Granada
Granada (Spain)
e-mail: *jedv@ugr.es, gmacia@ugr.es, pgteodor@ugr.es, piripo-powa@gmail.com*

***Abstract*— The popularity of P2P networks makes them an attractive target for hackers. Potential vulnerabilities in the software used in P2P networking represent a big threat for users and the whole community. To prevent and mitigate the risks, intrusion detection techniques have been traditionally applied. In this work in progress, a Markov based technique is applied to the detection of anomalies in the usage of P2P protocols. The detector searches for two kinds of anomalies: those that appear in the structure, grammar and semantics of each of the messages in the protocol, and those associated to the sequence of messages (protocol sessions). Previous results from other protocols, as HTTP and DNS, confirm the potentialities of the approach.**

Keywords: Network and computer security; Intrusion detection; Anomalous behaviour; P2P networks; Markov modelling

I. INTRODUCTION

Society's current dependency on communication networks, especially Internet, together with the increasing complexity of networks and the associated protocols, makes it necessary to possess ever more robust and reliable security techniques to protect services and users. In this context, the popularity of P2P networks, with millions of computers connected through networks as eDonkey [1] or BitTorrent [2], makes them an attractive target for hackers. The ubiquity of associated software allows potential hackers compromising many systems if vulnerabilities in the programs were detected and exploited. For this reason, security tools are essential to mitigate the associated risks.

One of the tools available for this purpose are the Intrusion Detection Systems (IDS) [3][4]. First proposed in the 1980s, they aim at detecting intrusion events, i.e., actions that might put at risk the security of a given target environment. Despite its age, some limitations and challenges still remain, mainly related to the performance achieved by current IDS technologies [5]. In fact, this is an active research area at present.

IDS systems are usually categorized in two broad sets [6], depending on the strategy used for the detection. Signature-based IDSs (SIDS) try to detect intrusions by somehow comparing current events with a set of rules that define known attacks. On the other hand, anomaly-based IDSs (AIDS) model the normal behavior of a monitored system and its communications, and trigger an alert every time a significant deviation is observed in the analyzed behavior. Both kinds of approaches present advantages and drawbacks, the most relevant being the greater efficiency of SIDSs when detecting known attacks and also their inability to detect new attacks. A more detailed description of SIDS, AIDS and their capabilities can be found in [6][7][8].

In this work in progress, a new technique for anomaly detection previously developed by the authors is applied to P2P systems. The technique, named "Service Specification and Stochastic Markovian modeling" (S3M) [9][10], has previously been applied to two widely used protocols like HTTP and DNS, with satisfactory results. S3M is a mixed approach that combines the use of specifications and learning in a stochastic framework, by using probabilistic finite state automata (FSA).

The proposed technique can model most of the protocols used in network communications and it can be used at two levels. Each individual message of the protocol is modeled at the first level, while the second tries to model the sequences of messages (*i.e.*, protocol sessions). It is possible to combine both models to account for potential interactions among anomalies at messages and sessions. The detection at the individual messages level is useful for the detection of attacks targeted at exploiting vulnerabilities in the software, by using specially crafted messages or ill-formed messages. On the other hand, the detection at the sequence level can detect attacks to the protocol itself (flaws in the protocol design) or attempts to get advantage from it. An example of the latter could be the detection of worms propagating using P2P networks by examining the traces of the protocol.

Although it is still at the initial stages, the adaptation of the technique is promising and challenging. In this paper, we highlight the main challenges that we are facing during this research and the strategies followed to solve them. In a first phase, only eDonkey and BitTorrent networks are considered.

The rest of the paper is structured as follows. Some basics of the S3M technique are explained in Section II. Section III describes the main challenges found in the adaptation of the technique to P2P protocols, and the strategies followed for obtaining solutions. Finally, Section IV summarizes the conclusions of this paper.

II. THE MARKOV-BASED S3M TECHNIQUE

Most network protocols, especially those in the application layer, present a well defined structure for the messages and the sequences of messages exchanged by the involved service entities. This structure is given through corresponding protocol specifications, and makes it possible to use formal methods to describe both the way in which

978-0-7695-3831-0/09 $26.00 © 2009 IEEE
DOI 10.1109/AP2PS.2009.32

IEEE computer society

each message is generated and the allowed sequences of messages in the protocol [11]. In these cases, a FSA can be used to model the normal behaviour of a given protocol. An example is the well-known FSA describing the behaviour of the TCP protocol [12].

When these FSAs are used for intrusion detection, a drawback of this approach is that it only considers the correctness of the message or sequence of messages, in terms of compliance with the specifications. The detection is *specification-based*. This is usually insufficient, as in many cases a single or several (a sequence) messages originating an attack still obey the rules specified by the FSA. Besides, many protocols are not fully specified, or leave certain details uncovered. This fact would allow a hacker to attack a protocol while no violation of its FSA is observed.

A more advanced approach is to build the FSAs as in the previous method in a first phase, and to evaluate the likelihood of the instances of the protocol by considering probabilities associated to the automata, *i.e.*, probabilistic finite state automata (PFSA). Here, each of the states of the FSA is viewed as a Markov source emitting symbols. The symbols may represent either certain fields of a single protocol message or types of messages, *e.g.,* ACK, HELLO, etc., depending on the level of description considered. This is the approach considered in the S3M modelling, which is briefly described next.

A. S3M modelling basics

S3M makes use of the Markov theory [13] to provide a production model consisting of a FSA and the associated probabilities for the observed events or transitions between states. A given sequence of events, p, can be evaluated by a model, λ, to provide a probability, $P(p|\lambda)$, of the events being generated by the model. From this probability, it is possible to define a *normality score*, $N_s(p)$, as

$$N_s(p) = P(p \mid \lambda)$$

Assuming that the model properly represents the normal behavior, $N_s(p)$ can be used to classify the events as either normal or anomalous, according to a given threshold, θ:

$$class(p) = \begin{cases} normal, & if \quad N_s(p) \geq \theta \\ anomalous, & otherwise \end{cases}$$

B. Estimation of the model

In S3M, the model for a protocol is derived from both the protocol specification and the observation of "*normal*" instances of the protocol in a monitored environment. The model is composed of two main elements: (*a*) a finite state automaton (FSA), defined by the states and the links connecting them (allowed transitions); and (*b*) a set of observable events with some probabilities associated with the different states, events and transitions. First, the topology of the FSA (states, links or transitions and final/initial states) is deduced from the specification of the protocol. Second, the transition probabilities, the set of events and their respective probabilities are calculated through a training process in which two complementary information sources are considered: (*i*) the observed legitimate messages or sequences of messages and (*ii*) the a priori allowed values of the different message fields. These sources are used for building a stochastic model that allows, in a posterior phase, the calculation of the probabilities $P(p|\lambda)$. For this purpose, it is necessary to assign observation probabilities for each event in each state.

The set of possible events in the FSA may be extracted from a specification or from any other source of information. In this case, it is hard to estimate the information related to the probability $P(p|\lambda)$, since usually there is only information about if a given event is allowed or not in each state. For this reason, it is necessary to apply a training procedure, in which traffic traces containing the events (messages, sequences of messages and field values within messages) are analyzed. In this case, the probabilities are estimated by an accounting procedure, considering the frequencies of appearance of each event (sequences of messages or field values within messages). Thus, the probability of observing an event e_k in a state s_i is estimated as

$$p(e_k \mid s_i) = \frac{count(e_k \mid s_i)}{\sum_{j=1}^{M} count(e_j \mid s_i)}$$

where M stands for the total number of events in the state i.

It is advisable to combine both sources of information, *i.e.*, specifications and traffic traces, as in many real services it is not feasible to acquire the whole set of events by an inspection of the specifications. This makes it essential to perform the training procedure.

Another significant advantage about the use of the training procedure must be indicated. It allows reviewing the topology of the FSA, including transition probabilities according to the observed data. Hence, the set of accepted transitions could be modified to incorporate that knowledge provided by the training data in two aspects: the relative frequency of occurrence of each transition (event), and the transitions themselves.

Moreover, many protocols are not completely specified, which results in different implementations, with few minor differences among them. The use of training data to estimate the allowed transitions and to assign probability values to them allows incorporating these differences into the model. Furthermore, if the set of training data is representative enough, there would be no need to initially establish the FSA manually from specifications, as it could be deduced from the training process. As it will be discussed below, this is an important advantage for complex protocols, as is the case in P2P protocols.

III. CHALLENGES AND STRATEGIES FOR THE USE OF S3M IN P2P PROTOCOLS

The aim of this work in progress is to use the S3M technique for the security analysis of P2P protocols. In this direction, we are currently working on the adaptation of the S3M technique in two levels. One model is being developed for the inner structure of each protocol message, and another for the sequences of messages (sessions) exchanged during the different operations of the protocol. In a first phase, we are considering only protocols used by eMule and BitTorrent, as they are widely deployed protocols.

Although it could seem that, based on the experience on HTTP and DNS, S3M could be applied to P2P protocols in a straightforward way, some new challenges appear when trying to afford this task, mainly due to the special features of P2P protocols. Some of these peculiarities and challenges are described in the following.

Protocol complexity. P2P protocols are more complex than DNS or HTTP in the sense that they consider many different types of messages, and the sessions in the protocols usually involve the exchange of large sequences of messages. As an example, the specification of the HTTP protocol defines two types of messages, *i.e.*, request and response, with only eight possible methods, *i.e.*, GET, POST, HEAD, OPTIONS, etc. However, in the eMule implementation of the eDonkey protocol, we have detected more than 12 different kinds of sessions between clients or between a client and an eMule server. Besides, the number of different messages in the protocol is higher than 30.

This means that building FSAs for P2P protocols result in really big automata when compared to those obtained for the HTTP or DNS protocols. For this reason, we have designed a methodology that allows the split of the automaton for the whole protocol in different *sub-automata,* representing diverse operations in the protocol. Dealing with these reduced size sub-automata becomes now a feasible task, especially if this automata should be trained as a previous step to a detection phase. As an example, in Fig. 1 we can see the sub-FSA used for the connection establishment in the eDonkey protocol.

Lack of strict implementations. As opposed to protocols like HTTP and DNS, which are well specified in publicly available documents, P2P protocols tend to be not well documented, or the specifications lack of information. In many cases, these specifications have been obtained through techniques like reverse engineering of the own software that implements the protocol.

Besides this fact, it is remarkable that even for a same protocol, *e.g.*, eDonkey, many different implementations appear (see Table I for implementations of the eDonkey protocol). Many of them contain certain bugs and others even consider certain extensions for the own application, *e.g.*, eMule extensions.

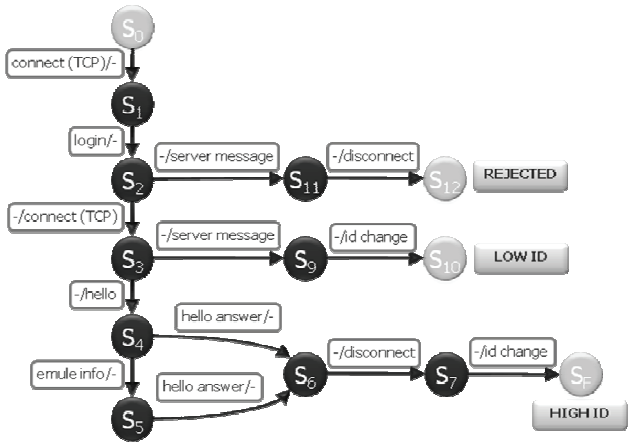

Figure 1. Example of sub-FSA for the connection establishment in eDonkey protocol. The initial/final states are coloured in yellow.

As a consequence of these facts, the process of building automata for the protocol becomes even harder when compared with the HTTP or DNS case. For this reason, in this ongoing work we have designed a methodology for building the automata as a dynamic (not manual) process.

We are currently evaluating an incremental approach that builds the sub-automata in several phases. The final objective is to develop a complete model for the protocol, starting from smaller sub-automata. In a first phase, a relatively small subset of sub-FSAs is manually generated by using one of the available specifications of the protocol, also identifying in this process a subset of the types of messages. Then, some traces are captured in a controlled environment. The available sub-FSAs are verified by using these traces, and the observed deviations are incorporated into the models. In this process, we are somehow training the sub-FSAs, as it is possible to redefine the allowed transitions between states and the available messages when doing these transitions. Next, those messages belonging to

TABLE I. PRESENCE OF DIFFERENT IMPLEMENTATIONS FOR THE EDONKEY PROTOCOL

Implementation	Presence	
	# of observed clients	*Percentage of observed clients*
eDonkey	12940214	14.23%
Old mldonkey	4708	0.01%
New mldonkey	87941	0.10%
Overnet	5844641	6.43%
eMule	70302372	77.32%
cDonkey	3212	0.004%
xMule	126601	0.14%
Shareaza	1289576	1.42%
aMule	325209	0.36%

Data extracted from [17].

protocol operations which have been understood by the available sub-FSAs are filtered out of the trace. The resulting dataset is used to infer new sub-automata and types of messages until all the traffic is in accordance with the models. Then, a new phase starts, in which a new dataset is recorded, now in a less restrictive environment. The same process as in the previous phase is followed until the whole set of sub-automata is established (or there is some confidence about that). Finally, the partial sub-automata are merged according to the observed sequences. The probabilistic nature of the FSA can be introduced at any point just by considering the relative frequencies of appearance.

Data representativeness. In order to build models that have a good representativeness of the real behavior of the protocols, we need to work with extensive traces taken from non-controlled environments. This is done in the last phase of our methodology. However, and additional challenge appear in this process. In order to use these models for intrusion detection, a "clean" training set [14] should be used, *i.e.*, data without attacks to train the models. In other words, as the system is attempting to model the normal behavior of the instances of the protocol, the training set must be representative of this normal operation and should not contain attack instances. On the other hand, the traffic should be real and not simulated, as the purpose is to model the normal operation of a real environment with real users [15]. As a consequence, if there is no control on the traffic from users, it is very difficult to obtain a trace with no attack instances. Various approaches to this problem have been proposed in the literature [14][16], but they all rely on the use of a S-NIDS to filter out the attacks in the captured traffic, which can be inaccurate due to false positives and to detection errors in the process.

In our approach, for dealing with such issue, during the non-controlled environment phase, given the fact that preliminary FSAs are built in the first phases, we try to get advantage of this information to filter possible attack instances that appear in the traces. For this task, we are evaluating the use of different non-supervised techniques for pattern recognition, *e.g.*, clustering techniques. The rationale behind this idea is that protocol procedures or messages that appear in the trace will only be incorporated into the model if they appear a considerable number of times or they sufficiently resemble a known procedure in the model.

IV. CONCLUSIONS

This paper presents the general guidelines of a work in progress aimed at developing an intrusion detection system for detection of anomalies in P2P protocols. It is based on the adaption of a technique based on Markov FSA previously designed by the authors for detection of anomalies in HTTP and DNS protocols.

The main conclusion from the preliminary work already done is that it is feasible the adaptation of the technique to the peculiarities that P2P protocols present. The main challenges have been identified and some strategies have been proposed to solve these problems. We are currently evaluating the effectiveness of these solutions.

ACKNOWLEDGMENTS

This work was partially supported by the Spanish National Research Program of the MEC, under project TEC2008-06663-C03-02 (70% FEDER funds).

REFERENCES

[1] Y. Kulback, and D. Bickson (2004): The eMule Protocol Specification. Available at http://www.cs.huji.ac.il/labs/danss/p2p/resources/emule.pdf

[2] B. Cohen (2008): The BitTorrent Protocol Specification. Available at http://www.bittorrent.org/beps/bep_0003.html.

[3] J.P. Anderson (1980): "Computer Security Threat Monitoring and Surveillance". Technical report, James P Anderson Co. Fort Washington, Pennsylvania.

[4] E.D. Denning (1987): "An Intrusion-Detection Model". IEEE Transactions on Software Engineering, Vol. 13, N. 2; pp. 222-232.

[5] T.S. Sobh (2006): "Wired and Wireless Intrusion Detection System: Classifications, Good Characteristics and State-of-the-art". Computer Standards & Interfaces, Vol. 28; pp. 670-694.

[6] P.N. Tan, M. Steinbach, and V. Kumar (2006): "Introduction to Data Mining". Addison-Wesley.

[7] A. Patcha, and J.M. Park (2007): "An Overview of Anomaly Detection Techniques: Existing Solutions and Latest Technological Trends". Computer Networks, Vol. 51; pp. 3448-3470.

[8] P. García-Teodoro, J.E. Díaz-Verdejo, G. Maciá-Fernández, and E. Vázquez (2009): "Anomaly-based Network Intrusion Detection: Techniques, Systems and Challenges". Computers & Security, Vol. 28; pp. 18-28.

[9] J.M. Estévez-Tapiador, P. García-Teodoro, and J.E. Díaz-Verdejo (2004): "Measuring Normality in HTTP Traffic for Anomaly-based Intrusion Detection". Computer Networks, Vol. 45, N. 2; pp. 175-193.

[10] J.M. Estévez-Tapiador, P. García-Teodoro, and J.E. Díaz-Verdejo (2005): "Detection of Web-based Attacks Through Markovian Protocol Parsing". 10th Symposium on Computers and Communications, pp. 457-462.

[11] W.A. Shay (2004): "Understanding Communications and Networks". Third ed., Thomson.

[12] Stevens, W.R., Fenner, B., and Rudoff, A.M. (2004). Unix Network Programming, the sockets networking API, volume 1. Addison Wesley Profesional, 3rd Edition.

[13] W. Feller (1968): "An Introduction to Probability Theory and its Applications. Vol. I". Third ed., John Wiley & Sons.

[14] M. Bermúdez-Edo, R. Salazar-Hernández, J.E. Díaz-Verdejo, and P. García-Teodoro (2006): "Proposals on Assessment Environments for Anomaly-based Network Intrusion Detection Systems". LNCS (Critical Information Infrastructures Security) 4347; pp. 210-221.

[15] J. McHugh (2000): "Testing Intrusion Detection Systems: a Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory". ACM Transactions on Information and System Security, Vol. 3, N. 4; pp. 262-294.

[16] N. Athanasiades, R. Abler, J. Levine, H. Owen, and G. Riley (2003): "Intrusion Detection Testing and Benchmarking Methodologies". Proc. 1st IEEE International Workshop on Information Assurance IWIA; pp. 63-72.

[17] http://mldonkey.sourceforge.net/HowManyDonkeys. Last visited May, 2009.