Fast and Accurate Signature-Generator for Detecting P2P Traffic

Cruz Alonso Bejarano, Luis A. Villa Vargas, Marco A. Ramírez, Oscar Camacho

Center for Computing Research of the National Polytechnic Institute, México

e-mails: {alonsob, lvilla, mramirez, oscarc}@cic.ipn.mx
Phone: +52-55-57-29-6000, Ext.56519

Abstract. Over the last few years, sharing information using P2P systems is widely used. Music, videos, software and documents contents are some of the information more requested by P2P users. However, the traffic generated for these applications have increased considerably. Moreover, pornography, no legal software distribution and virus propagation, have found in this systems an ideal platform for propagation. Detect P2P traffic is difficult. Current applications uses camouflaging techniques to avoid be detected: dynamic port numbers, port hopping and encrypted payload among others. In this paper we present a new approach to detect P2P traffic. We focus our work on the automating generation of signatures.

1 Introduction

Over the last few years Peer-to-Peer (P2P) application has come up as one of the main traffic generator over Internet bandwidth. Some studies revels that much of the ISP's traffic is caused for P2P application [10, 11]. Most of the traffic content related to P2P is Music, videos, software and documents. Moreover, it is likely that P2P is used as a favorite communication route for pornography. Because of this fact, proposing network infrastructure techniques to identify P2P traffic is attracting much attention within research circles. The accurate P2P traffic identification is indispensable for traffic controlling and blocking. The methods more used to identify P2P traffic are based on: port-based analysis, signature matching and heuristics methods. The port-based studies focus their strategy in identify the well-known port numbers [1,8,9]. It is due to most application use a very know port numbers for communications, and in this way traffic classification it is a trivial task. However, ports assignment in new P2P applications is dynamic, that is, using arbitrary ports. These new mechanism enable the possibility of avoid firewall-blocking mechanism. Heuristic methods use rules combinations created with information extracted from the transport layers: ports numbers, IP address, protocols, among others.

Signature mashing techniques extract strings for payload that can be used for identification [6,7]. Two major drawbacks are attached to these techniques: these do not work without payload information; either without signatures (does not work if the traffic was not previously analyzed). In this paper we present a new and accurate

A. Gelbukh, S. Suárez, H. Calvo (Eds.)
Advances in Computer Science and Engineering
Research in Computing Science 29, 2007, pp. 195-201

Received 11/07/07
Accepted 19/10/07
Final version 20/10/07

approach for P2P traffic-detection. We have developed a platform which dynamically can extract signatures from packets, the SiGeSy *Signature-Generarator-System*. This fast platform creates signatures from network traffic traces that can be synthesized in an FPGA-based system. The rest of the paper is organized as follows. Section 2 discusses the related work in P2P traffic detection. In section 3 a brief description of our selected P2P applications set are presented. The design and description of our signatures-generator platform is presented in section 4, and finally in section 5 our conclusions are presented.

2 Related Work

The P2P traffic detection has engaged the attention within research circles. Transport layer information like port numbers, IP address, payload-data, packet inspection among others are the mostly used in related works to identify P2P traffic [1,2,3,4,5]. Multiprotocol detection system has been analyzed also for P2P [6]. In this work the port usage, connection life times churn rates and overlay technology is analyzed as well. Detect P2P traffic based in the hosts behavior is analyzed in [8]. Motivated by the slow process of manual signature generation, some researchers have recently given attention to automating the generation of signatures mainly for virus and worms detection [7].

3 P2P Applications Analyzed

In order to prove the accurately of our Signatures detector platform, this work analyzes three different P2P applications: *eMule*, *Ares* and *Kazaa*. We have selected these applications due to these are the most popular in the University Campus. The Figure 1 shows the results survey of the P2P applications most used in our campus. More than 20% of people involved in the opinion poll use *Ares* and *Kazaa*, and 14.2% use *Limewire* and *eMule*.

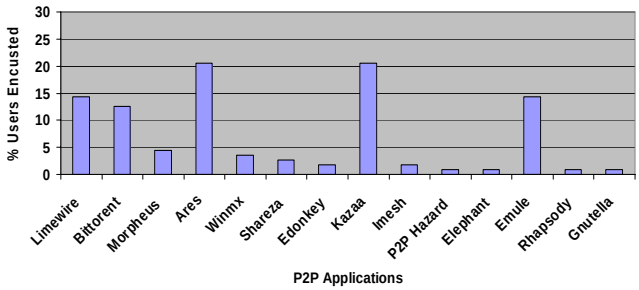

Fig. 1. Opinions pull of the most P2P application used in the University Campus.

The *eMule* system is maintained as an open source application. It is connected mainly to the *eDonkey* P2P network, although it is also connected to other networks achieving a large user base. Its main drawback is the slow download files speed. A faster P2P application is *Ares*. This application is attractive due to its simplicity and also for the fast connection offered. However, the most popular P2P program has been *Kazza*. Although seemingly at the present time *Kazza* popularity is declining.

4 SiGeSy - Signature Generator System

4.1. Architecture Overview

The Architecture is organized in four main structures: trace-creation, string-generator, string-matching and signatures-filtering.

4.2. Trace-Creation

We have created and analyzed different packet traces (showed in Table 1), grouped by P2P application. A total of 30 traces were created using the Ethereal ver. 0.99, running with WinPcap version 3.1. Our traces-set were generated in a controlled fashion and in isolated way, launching instances of each of the desired P2P application and collecting all the packets exchanged. Three different activities were monitored: during the application start-up, File-searching and file-downloading. Clean configuration are traces that were created when any P2P application were running.

Table 1. The set of traces used to generate signatures

Appli.	Start-Up	File-Search	File-Download
Ares	A1,A2,A3	As1,As2,As3	Ad1,Ad2,Ad3
Kazaa	K1,K2,K3	Ks1,Ks2,Ks3	Kd1,Kd2,Kd3
eMule	e1,e2,e3	es1,es2,es3	ed1,ed2,ed3
Clean	C1, C2, C3		

4.3. SPG - String Process Generator

Because the fact that the location of strings in the packets-payload are not deterministic, is important to have a system with the ability to recognize strings of different lengths, where these strings can be situated in different locations into the packet. The figure 2 shows the string-creating system. Using our pre-generated *start-up* traces (we use *Ares* for the system description) as an input (Table 1), the system reads a data stream packet at rate of one byte at once. The SPG system showed in the figure 2, has a string-data granularity from 2 up to 16 bytes (W_{min}=2 and W_{max}=16). In this way, fifteen sub-strings between W_{min} and W_{max} are generated having the same 16 bytes in the window; and after that data stream can be advanced byte for byte.

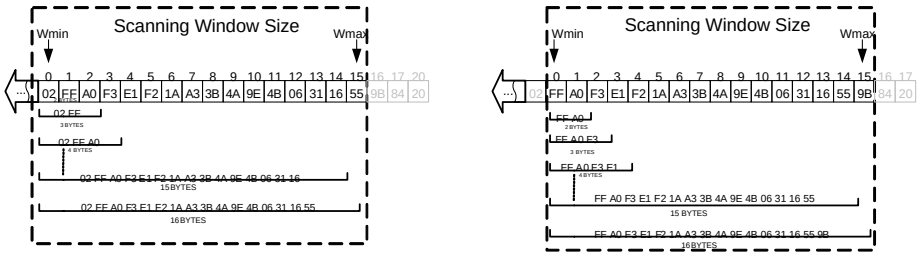

Fig. 2. The SPG produce fifteen strings using 16 bytes window size.

By generating signatures in this way, eventually all the possible strings of length from W_{min} byes to W_{max} bytes in every packet are scanned. This procedure is followed with each group of traces. The strings created with the SPG are written in a MySQL database file. Because of the fact that we have three start-up traces for each P2P application, we have also three SQL-files. Once the three files with strings are created, all the information is processed in the SMP (String Matching Process) as showed in the figure 3.

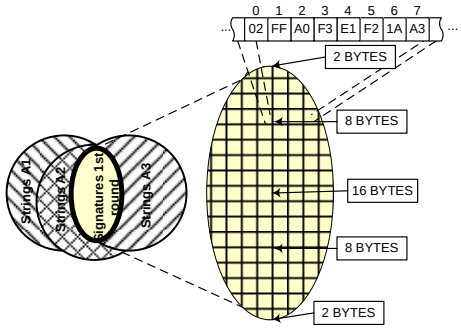

Fig. 3. The String Matching process.

The intersection between strings from A1, A2, and A3, create the first round signatures information file (***Signatures_1st***). Most of these signatures can be used for detect *Ares*, however, there are many signatures that are not *Ares*-exclusive information: NTF access, firewall access, IDS server access, among others. In order to have a more *Ares*-exclusive signatures file, we most filter the ***Signatures_1st***.

4.4. The SFP-Signatures Filtering Process

The strategy used to filter the *Ares* ***Signatures_1st*** is showed in the figure 4. We use the 27 traces no used to create the ***Signatures_1st***. The idea behind the SFP is to remove all signatures that matching with *Kazaa*, *eMule* and *Clean* strings generated with same SPG process. As we can see, in figure 4, all the signatures that not match with the ***Signatures_1st*** (*Ares*), make up the ***Signatures_2nd*** which is a more pure *Ares* signature file.

After repeat the whole process for *Ares*, *Kazaa* and *eMule*, three signatures-files were produced (one for each application). The content of these files is presented in the next section.

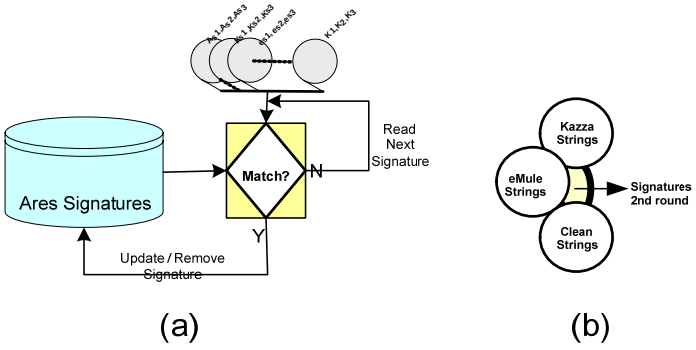

Fig. 4. (a) The SFP - Signatures Filtering Process. (b) A *signature_2nd* file generated after the SFP conclusion.

5 SiGeSy Results

In this section we present the signatures generated with SiGeSy. The Table 2 shows the packets reduction achieved. It is very important because for *Ares* the initial trace has 935 packets, which could contain signatures and on the other hand, SiGeSy shows that the signatures which can be used to detect *Ares* are only located in eleven packets.

Table 2. Packets reduction achieved by SiGeSy

P2P Application	Ini-Packets	End-Packets
Ares	935	11
Kazaa	632	18
eMule	729	5

It is important to note that, *Kazaa* payloads are encrypted in ten of the eighteen packets. That means that the other eight packets include the information useful to extract signatures. Although applications as *Kazza* use more advanced camouflaging techniques, we ha observed that this type of mechanism are not applied until the initial connection has been setup and therefore the SiGeSy get to find the packets with the adequate information.

The Table 3 shows a selection of signatures-set extracted from the **signatures_2nd** files. Although each file contains from 90 to 150 signatures, a visual analysis shows that some of them are clearly identified as the packets transmitted during the initial connection.

Table 3. A selection of signatures extracted with the SiGeSy

P2P Applications	Signatures Selected
Ares	GET /ares/hme2.php http://www.youtube.com
Kazaa	1.kazaa.com1%0#.* Host: images.kazaa.com Hermannet- works.com1%0#
eMule	http://emule-project.net DonkeyServer N01

6 FPGA Based IDS Design

In this section we present the signatures generated with SiGeSy. The Table 2 shows the packets reduction achieved. It is very important because for *Ares* the initial trace has 935 packets, which could contain signatures and on the other hand, SiGeSy shows that the signatures which can be used to detect *Ares* are only located in eleven packets.

7 Conclusions

This paper focused on the automating generation of signatures to detect P2P traffic. We have developed a platform which dynamically can extract signatures from packets. This fast platform creates signatures from network traffic traces. Our results shows that our system reduce the number of packets that most be analyzed to find an accurate set of signatures. For example, the relation for packets reduction is from 925 packets to 11 for *Ares* application. We have demonstrated that SiGeSy can extract signatures even when more sophisticated techniques are used for the P2P applications like *Kazaa* (payload encryption). Finally we have observed that these sophisticated mechanisms are not applied until the initial connection has been setup and therefore the SiGeSy get to find signatures. Finally we have reported a prototype experiment to demonstrate the portability of our platform. The Prototype was synthesized in a Field Programmable Gate Array- FPGA.

References

1. Subhabrata Sen, Oliver Spatscheck, and Dongmei Wang, "Accurate, Scalable In-network identification of P2P Traffic Using Application Signatures", 3th International World Wide Web Conference, New York City, May 2004, pp. 512 – 521.

2. Sarang Dharmapurikar, Praveen Krishnamurthy; Sproull, T.S.; Lockwood, J.W., "Deep packet inspection using parallel bloom filters", Micro, IEEE, Volume 24, Issue 1, Jan.-Feb. 2004 Page(s): 52 – 61
3. Sen, S.; Jia Wang, "Analyzing peer-to-peer traffic across large networks", IEEE/ACM Transactions on Networking, Volume 12, Issue 2, April 2004, pp. 219 – 232.

4. Madhukar, A. Williamson, C., "A Longitudinal Study of P2P Traffic Classification", 4th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, Sept. 2006, pp. 179- 188.

5. Sun-Myung Hwang, "P2P Protocol Analysis and Blocking Algorithm", Springer Berlin / Heidelberg Lecture Notes in Computer Science, Volume 3481/2005, pp. 21-30.

6. Holger Bleul Erwin P. Rathgeb Stefan Zilling, "Advanced P2P Multiprotocol Traffic Analysis Based on Application Level Signature Detection", Telecommunications Network Strategy and Planning Symposium, 2006. NETWORKS 2006. 12th International, Nov. 2006, pp. 1-6.

7. Newsome, J. Karp, B. Song, D., "Polygraph: automatically generating signatures for polymorphic worms", IEEE Symposium on Security and Privacy, May 2005, p.p 226-241.

8. Fivos Constantinou, Panayiotis Mavrommatis, "Identifying Known and Unknown Peer-to-Peer Traffic", Proceedings of the Fifth IEEE International Symposium on Network Computing and Applications, 2006, pp. 93 - 102.

9. T. Karagiannis, A. Broido, N. Brownlee, Kc Claffy, and M. Faloutsos. Is P2P dying or just hiding? In Globecom, Dallas, TX, USa, November 2004.

10. Krishna P. Gummadi, Richard J. Dunn, Stefan Saroiu, Steven D. Gribble, Henry M. Levy, John Zahorjan, "Measurement, modeling, and analysis of a peer-to-peer file-sharing workload", Proceedings of the nineteenth ACM symposium on Operating systems principles, Bolton Landing, NY, USA, October 2003, pp. 314 – 329.

11. Stefan Saroiu, P. Krishna Gummadi, Steven D. Gribble, "Measurement Study of Peer-to-Peer File Sharing Systems", Proceedings of the Multimedia Computing and Networking (MMCN), San Jose, January, 2002, pp. 156-170.

12. www.xilinx.com. Spartan-3 Starter Kit Board User Guide. UG130 (v1.1) May 13, 2005

13. Julien Lamoureux , Steve Wilton, HDL Conversion Tools. University of British Columbia, November 11, 2005.