Entropy estimation for real-time encrypted traffic identification

Peter Dorfinger[1], Georg Panholzer[1], and Wolfgang John[2]

[1] Salzburg Research, Salzburg, Austria
{peter.dorfinger, georg.panholzer}@salzburgresearch.at
[2] Chalmers University of Technology, Göteborg, Sweden
wolfgang.john@chalmers.se

Abstract. This paper describes a novel approach to classify network traffic into encrypted and unencrypted traffic. The classifier is able to operate in real-time as only the first packet of each flow is processed. The main metric used for classification is an estimation of the entropy of the first packet payload. The approach is evaluated based on encrypted ground truth traces and on real network traces. Encrypted traffic such as Skype, or encrypted eDonkey traffic are detected as encrypted with probability higher than 94%. Unencrypted protocols such as SMTP, HTTP, POP3 or FTP are detected as unencrypted with probability higher than 99.9%. The presented approach, named real-time encrypted traffic detector (RT-ETD), is well suited to operate as pre-filter for advanced classification approaches to enable their applicability on increased bandwidth.

Keywords: entropy estimation, real-time detection, traffic filtering

1 Introduction

During the last years the diversity of web based applications and their traffic patterns has increased enormously. This hinders network management activities which are to a substantial extent based on discriminative treatment of application traffic. Available systems for traffic classification are either based on fast but vague matching of IP addresses, transport protocol and ports or complex *deep packet inspection* (DPI) or statistical approaches. Especially applications that hide the traffic within encrypted communication are difficult to detect.

To detect such hidden applications often complex algorithms have to be performed. Due to the complexity such approaches have to inspect all packets on a link and consequently are unable to operate on high bandwidth links.

To overcome these shortcomings a fast and simple approach is needed that is able to pre-filter the traffic. The pre-filtering has the advantage to reduce the traffic which has to be further inspected to a reasonable amount. To keep the complexity of the approach low, it is a prerequisite to minimize status information. Thus our approach is designed such that pre-filtering of the traffic is performed upon arrival of the first communication packet (excluding TCP 3way handshake).

The classification based on the first packet solely has, beside the aspect of keeping status information low, the main advantage to be operational in real-time.

2 Related Work

In the late 1990s traffic classification was mainly performed on well known port numbers. Nowadays traffic classification is more and more getting complicated as an increasing number of applications try to hide its traffic from detection or classification. As hiding is often performed within encrypted network traffic, entropy-based classification algorithms have gained interest within the last years. Entropy-based approaches are often used to detect malicious traffic [1,2].

Lyda and Hamrock [1] use an entropy-based approach called bintropy that is able to quickly identify encrypted or packed malware. The entropy is used to identify statistical variation of bytes seen on the network.

Pescape [3] uses entropy to reduce the amount of data that has to be processed by traffic classification tools. Entropy is used as input for an advanced sampling approach that ensures that sensible information needed to get an appropriate model of the network traffic is still present. Packets not needed for an appropriate model are dropped.

An entropy-based approach which inspired the present work is presented in [2]. The N-truncated entropy for different encrypted protocols is determined. For example, for an HTTPS connection the byte entropy after 256 bytes of payload should be between six and seven. If the value for a specific connection is below this range, it is assumed that the connection is subverted.

In an earlier work [4] we concentrated on the possibility to pre-filter Skype traffic based on information gathered from the first packet of a flow.

3 Entropy and Entropy estimation

In 1948 Shannon [5] developed a measure for the uncertainty of a message. This measure is known as entropy in information theory. Shannon considers the case where we have a fixed number m of possible events $A_1, \ldots, A_m$ whose probabilities of occurrence $p_1, \ldots, p_m$ are known.

Entropy is defined as

$$H = -K \sum_{i=1}^{m} p_i \log(p_i), \tag{1}$$

Entropy can be used as a measure of the information content of a message. Equal probabilities lead to a maximum value for the entropy. Two concepts within data processing result in a high value of entropy. First data compression, as the bits needed for data representation should be minimized. Second data encryption, as any predictable behaviour available in the source data has to be removed. Both processing steps end with a data stream with equal probabilities for each event/symbol.

Within this work we are using entropy as a measure of uniformity. For an entropy-based test for uniformity an appropriate estimator for the entropy is needed. But as stated in literature [6] estimating the entropy based on a sample

is hard to retrieve, especially for $N < m$. Consequently we focus on a uniformity test which is influenced by entropy but does not need the estimation of the actual entropy.

Olivain and Goubault-Larrecq [2] present a work where, motivated by the problems of estimating the entropy based on a sample of length N accordingly, the N-truncated entropy is used instead. The N-truncated entropy $H_N(p)$ is defined as follows. Generate all possible words w of length N according to p. Estimate the entropy based on maximum likelihood (MLE) for all words w. The N-truncated entropy is then the average of the MLE estimates, i.e. sum up all MLE estimates and divide by the number of words to retrieve $H_N(p)$.

According to [2] $H_N(p)$, given that $p_i = 1/m$ for all i, which means that p follows the uniform distribution $\mathcal{U}$, $H_N(\mathcal{U})$ can be calculated by

$$H_N(\mathcal{U}) = \frac{1}{m^N} \sum_{n_1+\ldots+n_m=N} \left[\binom{N}{n_1 + \ldots + n_m} \times \left(\sum_{i=1}^{m} -\frac{n_i}{N} \log \frac{n_i}{N} \right) \right]. \quad (2)$$

The maximum likelihood estimator can be used as an unbiased estimator of H_N. Checking for uniformity is then straightforward. Based on a sample of length N estimate the entropy using MLE and compare the result to $H_N(\mathcal{U})$. The closer the estimated value is to $H_N(\mathcal{U})$ the more likely is it that the underlying distribution is uniform. Within our work we are using a Monte-Carlo method for estimating $H_N(\mathcal{U})$ and the confidence intervals.

For very short words this method for detecting uniformity fails. Paninski [7] states that for a uniformity test $N > \sqrt{m}$ samples are needed.

4 Classification

The proposed traffic classifier is based on a 2-stage approach, where the false-positive rate of the entropy estimation based classifier is reduced by the coding based classifier.

4.1 Entropy estimation based classification

The entropy estimation based classification is the core classification component of the whole approach. In contrast to other approaches like [2], only the first packet that transports payload is used to identify encrypted flows. While on the one hand this makes it more difficult to calculate an accurate estimate of the entropy it enables the utilization of this technique in an online fashion, i.e., to identify encrypted flows in live traffic without the need to buffer or delay packets.

The basic concept of our approach is to estimate the N-truncated entropy of the actual payload $\hat{H}_{MLE}(w)$ and compare the result to the estimated entropy of uniformly distributed random payload $H_N(\mathcal{U})$ of the same length. The difference between these two estimates is used to decide whether the flow is identified as being encrypted or not. In accordance to Paninski [8] we do not use the entropy estimation for payload with less than 16 bytes.

In order to preserve most of the encrypted flows we decided to use $\hat{H}_N(\mathcal{U}) \pm 3 \times \sigma_{\hat{H}_N(\mathcal{U})}$ as a suitable confidence interval. This should include $\hat{H}_{MLE}$ of approximately 99.7% of uniformly distributed random payload.

4.2 Coding based classification

We assume that for plain text messages the payload is encoded in ASCII or ANSI where values from 32 to 127 are used for printable characters. Based on the entropy-based approach text message may look random, consequently we defined an algorithm that identifies large text blocks within the payload.

The probability that a character from a random source will be in the range from 32 to 127 is about 37.5%. Especially if at the beginning of the packet a large fraction of characters is in this range the payload is most likely unencrypted. Consequently we added a check that if the fraction of bytes with values between 32 and 127 is greater than 75% we assume that the flow is unencrypted.

As we want to reduce the processing time we do not evaluate the full payload of the packet. For the coding based classifier only the first 96 bytes are evaluated.

5 Algorithm

This section presents the usage of the entropy estimation and coding based classifiers within a novel approach for traffic classification based on information solely gathered from content of the first packet of a flow. The first packet in this context is defined as the first packet sent by a UDP connection and for TCP the first packet is the one that follows the 3 way handshake. The term flow is defined as bi-directional flow based on the 5-tuple.

We use a few lists to store information. The SYNList stores all 5-tuples where we have received a packet with the SYN flag set. A 5-tuple is removed from the SYNList after receiving the first packet containing payload or receiving a packet where the FIN flag is set. Furthermore the 5-tuple will be removed from the SYNList if, 60s after the SYN packet, there has not been any data packet. This should prevent the SYNList from growing due to SYN flooding attacks.

A TCP or UDP flow which was detected as encrypted will be stored in the ENCRList. The 5-tuple will be removed upon receiving a packet where the FIN flag is set. Unencrypted UDP flows are stored in the UNENCRList. The UDP 5-tuples are removed from the lists after 300s inactivity of this 5-tuple.

The first block in Figure 1 ensures that the TCP 3 way handshake for all flows will be forwarded, as traffic classifiers often need the 3 way handshake to identify the start of a TCP connection. The 5-tuple of this flow is stored in the SYNList. This behaviour can be changed to drop the 3 way handshake. The following two blocks are responsible for forwarding/dropping of packets belonging to flows that have already been identified as encrypted or unencrypted respectively. If a UDP packet is not present in the ENCRList or in the UNENCRList it is the first packet of a flow. For TCP flows it is checked whether the 5-tuple of the packet is present in the SYNList, if so the packet is the first packet of a flow and has to be

evaluated. The encrypted check executes entropy and coding based classifier to determine whether the packet/flow is encrypted or not. If the flow is encrypted it is added to ENCRList and forwarded, otherwise to the UNENCRList and dropped. Detailed information about the implementation can be found in [9].

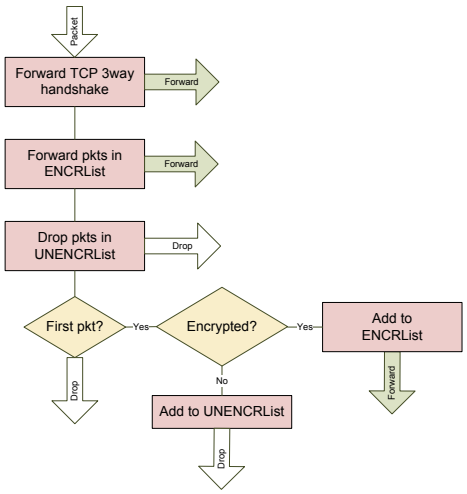

Fig. 1. Filtering flow chart

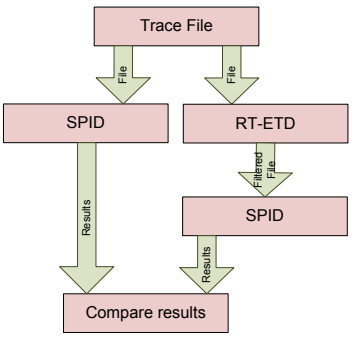

Fig. 2. Evaluation process

6 Evaluation

For the evaluation we used another traffic classification tool (SPID, Statistical Protocol IDentification), together with real network traces and traces where the ground truth is known.

SPID [10] utilizes statistical packet and flow attributes to identify application layer protocols by comparing probability vectors of these attributes to known protocol models obtained on controlled training data. As comparison measure, the Kullback–Leibler divergence together with a threshold is used. SPID is a hybrid technique, utilizing generic attributes, which include statistical flow features (e.g. flow and packet lengths) as well as packet payload characteristics (e.g. byte frequencies and offsets). With a balanced combination of attributes, SPID was shown to be very effective in differentiating between encrypted and obfuscated protocols considered hard to classify [10].

As ground truth traces with encrypted traffic we use a subset of the fully classified traces of encrypted traffic also used within SPID [10]. Session of encrypted eDonkey, MSE and Skype protocols (Table 1) have been collected at

a domestic network connection in Sweden by using Proxocket[3], which enabled efficient separation of network traffic on a per-application basis.

Real network traces have been collected in a network of a small cable network provider in a segment used by about 100 customers. Several traces have been collected in this network, for the evaluation we are using three of them. A 1h/2GB trace, a 7.5h/13GB trace and a 35h/48GB trace. Additionally a 1 hour trace with a total volume of 13GB from the network of a wireless provider used by about 1000 customers is used for the evaluation.

Figure 2 shows a schematic representation of the evaluation process. In the first step the collected trace files are processed by SPID and RT-ETD. The results of SPID are used as 100% baseline within the individual traffic categories. The filtered output files from the RT-ETD are then processed by SPID. The results are then compared to the results based on the original file. The metric we are using here is the fraction of flows in each category that can still be detected in the filtered file and the fraction by which the size of the trace file was reduced by the RT-ETD. An optimal result would be if still 100% of the encrypted flows are present in the filtered files, and no unencrypted flows are present.

Table 1 shows the results based on the ground truth traces. The worst results we get for encrypted eDonkeyTCP traffic, where 2.3% of the missing 5.5% is dropped due to a packet length of the first packet that is shorter than 16 bytes. An evaluation of including packets where the length is shorter than 16 bytes is left open for future work.

	Flows	
Protocol	original	filtered
eDonkeyTCP encr.	398	94.5%
eDonkeyUDP encr.	828	99.6%
MSE	649	99.2%
SkypeTCP	91	97.8%
SkypeUDP	1973	98.0%

Table 1. Evaluation based on ground truth trace, note that more than 94% of the encrypted traffic is still present in the filtered file.

Traffic	amount
SYN + SYN/ACK flows	14.88%
VPN Key exchange	2.91%
VPN data	55.04%
Skype encr.	4.21%
SPID encr.	0.80%
AKAMAI	19.08%
unknown	3.08%

Table 2. Traffic shares of filtered file. The major fraction of the traffic belongs to encrypted protocols. SPID encr. includes all other encrypted protocols classifiable by SPID.

Based on the 1h/2GB real network trace we evaluated the usage for the coding-based classifier for TCP and UDP traffic. Using the coding-based classifier for UDP traffic does not influence the classification at all.

[3] Proxocket is a dll proxy for Winsock that dumps a copy of the network traffic to and from an application to a pcap file. (`http://aluigi.altervista.org/mytoolz.htm#proxocket`)

For TCP traffic the coding-based classifier removes about 900kByte from the output file without changing the classification results of SPID. The 900kByte are plain POP3 flows.

Table 3 shows the results of the evaluation of using our algorithm as pre-filter for SPID. We are showing results for two traces collected in the cable provider network, and results for the trace from the WLAN provider network. An empty entry indicates a 0 count, this category is completely removed, whereas 0% indicates that the fraction is below 0.01%.

Between 73% and 96% of the flows are dropped by our pre-filter. Unencrypted flows such as FTP, HTTP, IMAP or SMTP are almost completely removed, which is a strong indication that only a small fraction of unencrypted traffic is forwarded. For encrypted protocols that we take into account, eDonkey TCP/UDP encrypted, MSE, Skype TCP/UDP at least 76.7% (MSE) of the flows are still present in the filtered file. For eDonkey and Skype the fraction is above 93%. SSH and SSL are detected as unencrypted due to the usage of a plain connection establishment. Such protocols can be easily detected by state of the art filtering methods and are thus outside of the scope of our work.

	7.5h/13GB		35h/48GB		1h/13GB	
Type	original	filtered	original	filtered	original	filtered
Filesize [MB]	13531	3.36%	48527	2.94%	13157	12.5%
Sessions	242309	13.0%	1050206	4.37%	195243	27.1%
BitTorrent	64		8067		5061	
DNS	30946		91015		22166	
eDonkey					7169	
eDonkeyTCP encr.	**9**	**100%**	**36**	**100%**	**9653**	**96.2%**
eDonkeyUDP encr.	**44**	**93.2%**	**95**	**100%**	**21808**	**96.0%**
FTP	443		31		24	
HTTP	118226		210320	0%	46643	
IMAP	248		1861		26	
IRC			68		66	
ISAKMP	4		3	33%	227	
MSE	**30**	**76.7%**	**280**	**99.3%**	**1323**	**81.7%**
MSN	152		19		23	
POP	9770		25528		632	
SkypeTCP	**773**	**99.5%**	**1167**	**99.4%**	**456**	**94.1%**
SkypeUDP	**18945**	**99.0%**	**27675**	**99.2%**	**6079**	**98.7%**
SMTP	832		1075		53	
SpotifyServer	55	74.5%	90	94.4%	306	95.4%
SSH	946		60529		26	
SSL	10044		19203	0%	3210	
UNKNOWN	50778	23.3%	603144	2.8%	70292	21.2%

Table 3. Results for SPID pre-filter. The filter reduced the filesize by a factor of about 20. Well known encrypted protocols are removed, whereas a large fraction of encrypted flows is still present

For the 1h/2GB real network we performed a detailed analysis on the filtered file. The trace is reduced to a size of 39.63MByte. Table 2 gives an overview of the categories that are still present in the filtered file. Almost 15% of the traffic is SYN+SYN/ACK traffic without any data communication. 63% of the traffic is encrypted traffic, where for the detection of the Skype traffic we are using the Adami Skype detector [11]. 19% of the traffic belong to a single flow where a binary request, invoked by a flash player, requests content from the AKAMAI distribution network.

7 Summary and Conclusions

In this paper we have presented how to use information from the first packet of a flow to identify encrypted traffic. The algorithm consists of two classifiers and can be used as real-time traffic filter as only the first packet of a flow has to be evaluated. The core classifier is based on payload entropy estimation, where entropy is used as a measure for uniformity which is an indication for encryption. The algorithm is refined by a further classifier, which takes into account the coding range used by the ASCII code. The main strength of the approach is its simplicity and accuracy. Evaluation based on encrypted ground truth traces and real-world network traces shows that more than 94% of the encrypted traffic is detected as encrypted, and more than 99% of the unencrypted traffic as unencrypted. Well known unencrypted protocols such as SMTP, HTTP, FTP, IMAP, POP3 and DNS are detected as unencrypted with probability as high as 99.9%.

A typical use case for our real-time traffic filter could be to pre-process data for L7 classifiers[4] where the focus is on encrypted flows, or detecting hidden traffic within encrypted flows. Using our approach the traffic volume that has to be handled by these classifiers can be reduced by a factor of about 10 to 50, depending on the traffic matrix.

References

1. Lyda, R., Hamrock, J.: Using entropy analysis to find encrypted and packed malware. IEEE Security & Privacy **5**(2) (2007) 40–45
2. Olivain, J., Goubault-Larrecq, J.: Detecting subverted cryptographic protocols by entropy checking. Research Report LSV-06-13, Laboratoire Spécification et Vérification, ENS Cachan (2006)
3. Pescape, A.: Entropy-based reduction of traffic data. IEEE Communications Letters **11**(2) (2007) 191–193
4. Dorfinger, P., Panholzer, G., Trammell, B., Pepe, T.: Entropy-based traffic filtering to support real-time Skype detection. In: IWCMC, Caen, France. (2010) 747–751
5. Shannon, C.E.: A mathematical theory of communication. Bell System Technical Journal **27** (1948) 379–423, 625–56
6. Schürmann, T.: Bias analysis in entropy estimation. Journal of Physics A: Mathematical and General **37**(27) (2004) L295–L301
7. Paninski, L.: A coincidence-based test for uniformity given very sparsely sampled discrete data. IEEE Transactions on Information Theory **54**(10) (2008) 4750–4755
8. Paninski, L.: Estimation of entropy and mutual information. Neural Computation **15**(6) (2003) 1191–1253
9. Dorfinger, P.: Real-Time Detection of Encrypted Traffic based on Entropy Estimation. Master's thesis, Salzburg University of Applied Sciences, Austria (2010)
10. Hjelmvik, E., John, W.: Breaking and improving protocol obfuscation. Tech. Rep. 2010-05, Computer Science and Engineering, Chalmers University of Technology (2010) Online: `http://www.iis.se/docs/hjelmvik_breaking.pdf` (28.01.2011).
11. Adami, D., Callegari, C., Giordano, S., Pagano, M., Pepe, T.: A Real-Time Algorithm for Skype Traffic Detection and Classification. In: Smart Spaces and Next Generation Wired/Wireless Networking. (2009) 168–179

[4] Classifiers mainly utilizing costly regular expressions on packet payloads.