

# Fingerprinting Tor's Hidden Service Log Files Using a Timing Channel

Juan A. Elices <sup>#1</sup>, Fernando Pérez-González <sup>\*.#2</sup>, Carmela Troncoso <sup>+3</sup>

<sup>#</sup> *Electrical and Computer Engineering Department, University of New Mexico  
University of New Mexico — Albuquerque, NM 87131 — USA*

<sup>\*</sup> *Signal Theory and Communications Department, University of Vigo  
University of Vigo — 36310 Vigo — Spain*

<sup>+</sup> *K.U.Leuven, ESAT/COSIC*

*Kasteelpark Arenberg 10, B-3001 Leuven-Heverlee, Belgium*

<sup>1</sup> jelices@ece.unm.edu, <sup>2</sup> fperez@gts.uvigo.es, <sup>3</sup> carmela.troncoso@esat.kuleuven.be

**Abstract**—Hidden services are anonymously hosted services that can be accessed over Tor, an anonymity network. In this paper we present an attack that allows an entity to prove, once a machine suspect to host a hidden server has been confiscated, that such machine has in fact hosted a particular content. Our solution is based on leaving a timing channel fingerprint in the confiscated machine's log file.

In order to be able to fingerprint the log server through Tor we first study the noise sources: the delay introduced by Tor and the log entries due to other users. We then describe our fingerprint method, and analytically determine the detection probability and the rate of false positives. Finally, we empirically validate our results.

## I. INTRODUCTION

Anonymous access to the internet has become a necessity for wide range of users: human rights activists, journalists, military, dissidents, bloggers, citizens in censored countries, etc. This need fostered the development of anonymous communication systems, such as Freenet [1], Tor [2], or JAP [3]. Among these, the most popular is Tor which, for instance, has had an important role in Iran and Egypt's dissident movements [4]. Anonymity is not only an issue for users, but it is also important for servers. The Electronic Frontier Foundation and Reporters Without Borders advise the use of hidden (anonymous) services to protect the safety of dissidents who have to overcome censorship. Unfortunately, hidden services are also used for illegal purposes such as distributing child pornography, or supporting terrorism.

In this paper we consider a scenario in which an entity (e.g., a law enforcement agency, or a censorship-prone government) has confiscated a machine suspicious of hosting a particular content, but this content has been deleted. This entity wishes to leave a fingerprint on the hidden server log that serves as proof that this particular machine actually hosted the targeted content. We propose to create this fingerprint by sending several HTTP requests to the server according to a pre-defined

schedule. The server logs the time when these requests are processed in the log file. Detecting the fingerprint consists on deciding whether the desired timing pattern is present in the log.

Finding such a pattern is not straightforward, mainly because of two issues. First, Tor achieves anonymity by relaying web traffic through (in general) three onion routers to ensure that no relay knows both the communication's originator and recipient. In the case of hidden services, both client and server use a three-relay path, and resort to a rendez-vous point to find each other [2]. Hence, the time when users send a request does not coincide in general with the time logged by the hidden service. Second, when detecting a fingerprint it is not possible to distinguish between our log entries and the ones resulting from other users' requests. We overcome these problems by estimating the log time from the date field included in the HTTP responses to our requests; and by statistically modelling the number of other users' entries in the log file. Our fingerprinting algorithm can be tuned to achieve a probability of misdetection as small as desired, while minimizing the probability of false positive. Further, we show that our fingerprint is difficult to recognize, and hence to remove, by the hidden service administrator.

The rest of this paper is organized as follows: Section 2 reviews previous approaches to the log fingerprinting problem. In Section 3 we formally describe our problem and introduce the notation. Section 4 studies the relation between the HTTP response date field and the requests' log time. Section 5 characterizes the distribution of the number of HTTP requests received by a server. In Section 6 we describe our fingerprinting method, and in Section 7 we analytically derive the probability of detection and the false positive rate, and empirically validate our theoretical results. Section 8 summarizes our contribution and provides future directions for our research.

## II. PREVIOUS WORK

Shebaro et al. [5] already studied the log fingerprinting problem. Their solution is based on sending  $k_i$  requests per

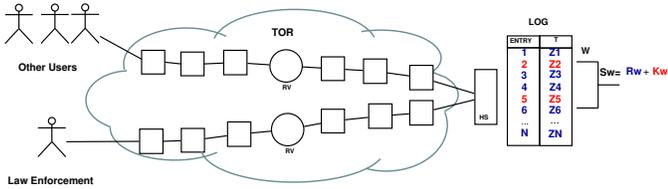


Fig. 1. System Model

minute representing a single fingerprint bit. Their method is not easily adjusted to meet a desired performance and needs a good statistical characterization of the number of requests that the server receives, an information that is rarely available at the client. Also their solution has the drawback of a high detectability at the server being fingerprinted. In contrast, our solution uses the HTTP response date field, which allows us to obtain more reliable results, and with a much lower detectability.

Liberatore et al. [6] also tackled the problem of tagging (fingerprinting) P2P clients' logs. They put their tag in the information that is stored, a CIDR block or a peerID. However, their algorithm cannot be used to fingerprint a webserver, as the only values of a log entry we can control are the time and the requested line. For reasons of detectability, we use only the time of the request. They mention the possibility of using the time between events, but they do not discuss how this could be implemented, nor provide a formal analysis of the underlying properties.

### III. PROBLEM DESCRIPTION

In this section we formally describe the problem and introduce the notation we use in the rest of the paper.

#### A. Channel Model

Figure 1 shows the basic scheme of our problem. We send  $L$  HTTP requests to a hidden server (HS) through Tor. These requests will appear in the server's log file mixed with other clients' requests (represented in blue and red in Figure 1). Note that in order to achieve our low detectability goal we do not tag the requests in any way.

We consider that the  $i$ th request we send appears on the HS's log at time  $Z_i = X_i + N_i$ ,  $i = 1, \dots, L$ , where  $X_i$  the moment when it was sent and  $N_i$  is random delay introduced by the Tor network Loesing et al. modeled  $N_i$  as a Fréchet distribution [7], but we choose not to use this approximation since an estimator  $\hat{Z}_i$  of the log time is available in the HTTP response, as we discuss later.

When it comes to deciding whether the  $i$ th request appears in the log or not, we look inside a window  $W_i$  containing  $\hat{Z}_i$ . We denote by  $E_{W_i}$  the number of log entries that fall inside this window. These entries can correspond to our fingerprinting requests that we denote as  $K_{W_i}$ , or to other clients' requests, whose number is denoted by  $R_{W_i}$ . More formally,  $E_{W_i} = K_{W_i} + R_{W_i}$ , where  $K_{W_i} = \sum_{j=1}^L I(Z_j \in W_i)$ , and  $I(\cdot)$  denotes the indicator function. The sequence

$\{R_{W_i}\}$  is modelled in Section V as a negative binomial (NB) distribution.

#### B. Detection Accuracy metrics

To measure the performance of our fingerprinting scheme, we use two metrics: the probability of detection ( $P_D$ ), which represents the probability of detecting the fingerprint when it is actually present in the log, and the probability of false positive ( $P_F$ ), which represents the probability of deciding that there is a fingerprint when the log has not been fingerprinted. Formally, we can express this problem via classical hypothesis testing with the following hypotheses:

- $H_0$ : The log has not been fingerprinted.
- $H_1$ : The log has been fingerprinted.

Then  $P_D$  is the probability of deciding  $H_1$  when  $H_1$  holds, whereas  $P_F$  is the probability of deciding  $H_1$  when  $H_0$  holds.

Typically, performance is measured using the so-called ROC (Receiver Operating Characteristic) curves, which represent  $P_D$  vs.  $P_F$ . In a practical setting, one fixes a certain value of  $P_F$  (that has to be very small if we want to achieve a high reliability and avoid accusing an innocent server) and then measure  $P_D$  (which we would like to be as large as possible).

In addition, we want to avoid that the presence of the fingerprint can be easily detected by any other party than the originator of such sequence, i.e., the sequence should have low detectability.

### IV. HTTP RESPONSE DATE INFORMATION

In this section we characterize the estimator of the log time,  $\hat{Z}_i$ . On the header of the HTTP response, there is the "date field", which we use as an estimator of  $Z_i$ . According to [8], in theory, this field represents the moment just before the HTTP response is generated. We define the estimation error as  $\varepsilon_i \doteq Z_i - \hat{Z}_i$ .

In order to characterize  $\varepsilon$ , we performed some experiments on an AMD Turion 64 X2 2GHz with 3GB of memory running Apache 2.2.15 over Windows Vista SP2. The experiments were selected to cover a wide range of server situations:

- 1) Normal situation: We request a 44 bytes file 2 times per second.
- 2) Large transmission time: We request a 7 MBytes file every minute.
- 3) Very loaded server: We request a 7 MBytes file 10 times per second.
- 4) Demanding requests: We request a dynamic file of around 80 bytes, for which the server needs at least 5 seconds to generate the response.

Table I shows the percentage of HTTP responses that we receive with a given response code. We can see that for valid HTTP responses (i.e., the ones with response code 200) the estimation error is  $\varepsilon = 0$ .

The above results were obtained using only one machine with one server software. In order to assess that those results can be generalized we performed a second experiment. We

TABLE I  
PERCENTAGE OF RESPONSES IN APACHE 2.2.15

	Code 200 with $\varepsilon = 0$	Code 200 with $\varepsilon \neq 0$	Code 5xx logged	Code 5xx not logged
Experiment 1	94.10%	0.00%	0.01%	5.89%
Experiment 2	93.60%	0.00%	0.91%	5.49%
Experiment 3	11.33%	0.00%	33.78%	54.90%
Experiment 4	100.00%	0.00%	0.00%	0.00%

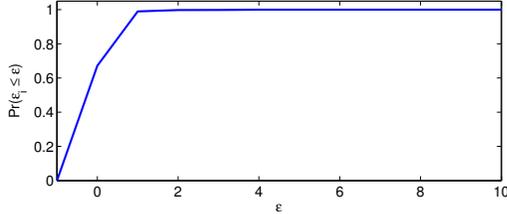


Fig. 2. Cumulative Distribution Function of  $\varepsilon$  in Apache 1.3.33

requested one petition per second during 24 hours to a web server at the University of Vigo. This machine is an AMD Athlon XP 2000+ with 512MB of memory that runs Apache 1.3.33 over Debian 4.0. Figure 2 shows the obtained cumulative distribution function of  $\varepsilon$ .

An important conclusion is that  $\varepsilon$  depends on the machine and its running HTTP server software. In this work we assume that the data from Figure 2 can be extrapolated and leave the characterization of this dependency for future work. We also assume that no HTTP requests triggering an error appear in the log.

In Figure 2 we observe that  $\varepsilon$  is always greater or equal to zero. We can conclude that  $W_i$  is reasonably given by  $W_i = [\hat{Z}_i, \hat{Z}_i + w]$ , for some integer  $w \geq 0$ . We define:

$$P_Z(Z_j, W_i) \doteq Pr(Z_j \in W_i) = Pr(Z_j - \hat{Z}_i \leq w),$$

where  $P_Z$  can be interpreted as the probability that the log time of the  $j$ th request falls within the reference window of the  $i$ th request.

## V. MODELING THE NUMBER OF LOG ENTRIES $R_W$

In the previous section we explained how to deal with the first noise source, i.e., the delay introduced by Tor. In this section we present a model for the number  $R_W$  of requests from other clients that fall within a window of width  $w$  seconds.

### A. Data Collection

The access logs for this research were obtained from seven different World Wide Web servers: a department-level web server at the University of Calgary (Department of Computer Science); a research group web server at University of Vigo (Signal Processing group); a campus-wide web server at the University of Saskatchewan; the EPA WWW server located at Research Triangle Park; the web server at NASA's Kennedy Space Center; the ClarkNet WWW server, an old commercial Internet provider in the Baltimore - Washington D.C. region; and the 1998 World Cup web site WWW server.

In Table II we show a summary of the different servers' logs. We see that the number of requests per day varies by several orders of magnitude, as we see in Section VII this has a great impact on the probability of false positive.

### B. Results

According to [9], the inter-time between requests follows an exponential distribution. This implies that the requests follow a Poisson distribution with parameter  $\lambda$  that can be estimated using the Maximum Likelihood Estimator (MLE) [10]:

$$f_{Poisson(\lambda)}(k) = \frac{\lambda^k e^{-\lambda}}{k!} \quad \text{and} \quad \hat{\lambda}_{MLE} = \frac{1}{N} \sum_{i=1}^N k_i,$$

where  $N$  is the number of considered samples.

Another distribution commonly used to model counting processes is the negative binomial (NB) distribution [10]. We can see the NB as a Poisson distribution, where  $\lambda$  is itself a random variable, distributed according to  $\Gamma(r, \frac{p}{1-p})$ .

$$\begin{aligned} f(k) &= \int_0^\infty f_{Poisson(\lambda)}(k) \cdot f_{\Gamma(r, \frac{p}{1-p})}(\lambda) d\lambda \\ &= \int_0^\infty \frac{\lambda^k e^{-\lambda}}{k!} \lambda^{r-1} \frac{e^{-\lambda(1-p)/p}}{(\frac{p}{1-p})^r \Gamma(r)} d\lambda \\ &= \frac{\Gamma(k+r)}{k! \Gamma(r)} \cdot (1-p)^r p^k. \end{aligned}$$

To calculate the parameters we can use the MLE as follows [11]:

$$\begin{aligned} \hat{r}_{MLE} &= \left[ \frac{(\sum_{i=1}^N k_i)^2}{N \sum_{i=1}^N k_i^2 - (\sum_{i=1}^N k_i)^2} - N \sum_{i=1}^N k_i \right] + 0.5 \\ \hat{p}_{MLE} &= \frac{\sum_{i=1}^N k_i}{r \cdot N + \sum_{i=1}^N k_i}, \end{aligned}$$

where  $N$  is the number of considered samples.

TABLE III  
MLE PARAMETERS AND GOODNESS OF FIT

Log	Poisson	NB	Poisson	NB
	MLE Param.	MLE Param.	K-L Div.	K-L Div.
Calgary	$\lambda = 0.024$	$p = 0.023, r = 1$	0.9104	0.0053
Vigo	$\lambda = 0.050$	$p = 0.048, r = 1$	0.8843	0.0350
Saskatchewan	$\lambda = 0.130$	$p = 0.115, r = 1$	0.6956	0.0046
EPA	$\lambda = 0.557$	$p = 0.358, r = 1$	0.3638	0.0018
Nasa	$\lambda = 0.684$	$p = 0.406, r = 1$	0.3018	0.0002
Clarknet	$\lambda = 2.753$	$p = 0.579, r = 2$	0.1147	0.0020
World Cup	$\lambda = 15.266$	$p = 0.836, r = 3$	1.2198	0.0051

The estimated MLE parameters and the goodness of fit are shown in Table III. The goodness of fit is measured using the Kullback–Leibler divergence, that is a non-symmetric measure of the difference between two probability distributions: the observations and the model. From these results we can conclude that the negative binomial distribution is a better approximation than the Poisson distribution. This means that: i) the data is overdispersed, i.e., the variance is larger than the mean; and ii) the interarrival times are exponentially

TABLE II  
SUMMARY OF ACCESS LOG CHARACTERISTICS

Log	Calgary	Vigo	Saskatchewan	EPA	Nasa	Clarknet	World Cup
Access Log Duration	1 year	1year	7 months	1 day	2 months	2 weeks	8 days
Access Log Start Date	Oct 24/94	Jun 5/10	Jun 1/95	Aug 29/95	Jul 1/95	Aug 28/95	May 1/98
Access Log File (MB)	49.8	377	222	4.24	355	327	907
Total Requests	726,739	1,581,971	2,408,625	47,748	3,461,612	3,328,587	10,345,553
Requests per day	2,059	4,321	1 1,255	47,748	56,748	237,756	1,293,200

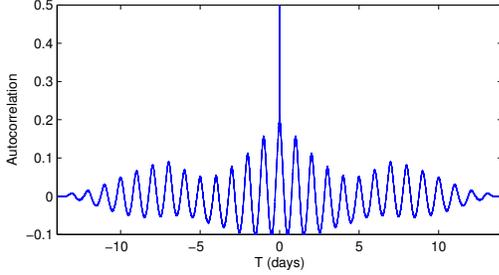


Fig. 3. Autocorrelation of ClarkNet Log

distributed but the rate is not fixed, but a gamma distributed random variable.

Some daily and weekly trends in the logs can be observed in Figure 3. Our method ignores them and considers  $\{R_{W_i}\}$  as a stationary sequence. Also, it can be seen that the peak is only one sample width. Therefore, the requests can be considered uncorrelated.

We note that adding negative binomial distributed random variables with the same  $p$  results in another negative binomial distributed random variable whose parameter  $r$  is the sum of the respective  $r$ 's [10]. In our problem we assume  $\{R_{W_i}\}$  to be a sequence of independent and identically distributed negative binomial random variables with parameters  $p_L$  and  $w \cdot r_L$ .

## VI. FINGERPRINTING METHOD

This section describes our algorithms to create and detect fingerprints.

### A. Creating the Fingerprint

Ideally we would like our fingerprint to be very difficult to detect by any other than its originator. Additionally, fingerprinting should be as fast as possible, in order to be able to read the fingerprint from a small fragment of the log file. We compute the rate of departures of our requests as  $\lambda_{FP} = d \cdot \hat{\lambda}_S$ , where  $d$  is the detectability factor, i.e., the increase in the number of requests the server will receive due to our fingerprint (reasonable values are 0.01, 0.05, 0.1), and  $\hat{\lambda}_S$  is a rough prediction of the rate of requests that the server receives.

When we want to fingerprint a server, we generate  $L - 1$  samples from an exponential distribution with rate  $\lambda_{FP}$ . These values are our interdeparture times, and the HTTP requests are sent to the server according to them. Therefore the expected time to fingerprint the server is  $(L - 1)\lambda_{FP}$  seconds.

Note that we need to store the HTTP response “date fields” of the correct responses (status code 200), since this

information will be later needed to recover the fingerprint. We denote the number of succesful responses as  $L'$ .

### B. Detecting the fingerprint

Detecting the fingerprint consists on deciding whether the log file contains the fingerprint. We say that the  $i$ th request from our fingerprint is present in the log when at least  $c_i$  entries in the log fall inside the window  $W_i$ . This can be mathematically expressed as  $d_i = I(E_{W_i} \geq c_i)$ , where  $c_i = \sum_{j=1}^L I(\hat{Z}_j \in W_i)$ . We denote the number of detected requests as  $S_{L'} = \sum_{m=1}^{L'} d_i$ . We consider that a fingerprint is present in the log when we find at least  $\Theta$  requests, i.e.,  $S_{L'} \geq \Theta$

We note that this is not the optimal decoder, as the distribution of  $\varepsilon$  is not used. However, when  $\varepsilon$  is small, as it is the case considered in this paper (see Sect. IV), the proposed decoder is nearly optimal.

## VII. ANALYSIS

In this section we calculate the theoretical probabilities of detection and false positives. Afterwards we do some experiments to validate the results.

### A. Probability of Detection

We recall from Section III that the number of requests in  $W_i$ ,  $E_{W_i}$ , is the sum of ours (i.e.,  $K_{W_i}$ ) and other users' (i.e.,  $R_{W_i}$ ) requests. Further, in Section V we showed that  $\{R_{W_i}\}$  can be modeled as an NB distribution with parameters  $p_L$  and  $w \cdot r_L$ .

Before diving into the analysis we must model  $K_{W_i}$ , that represents the number of fingerprint entries on the log that appear inside  $W_i$ .

We know that the  $j$ th request has a probability of  $P_Z(Z_j, W_i)$  of appearing inside  $W_i$ . Since  $K_W$  is the sum of Bernoulli random variables we can approximate it by a binomial distribution [12] with parameters:

$$n_k = \left\lfloor \frac{\left( \sum_{\hat{Z}_j \in W_i} P_Z(Z_j, W_i) \right)^2}{\sum_{\hat{Z}_j \in W_i} P_Z(Z_j, W_i)^2} + 0.5 \right\rfloor \text{ and}$$

$$p_k = \frac{\sum_{\hat{Z}_j \in W_i} P_Z(Z_j, W_i)}{n_k}.$$

1) *General Case*: here we study the probability of detection without making any assumption concerning  $\lambda_{FP}$ .

First we study  $P_{d_i}$  that represents the probability that we detect the  $i$ th request of our fingerprint in the log. Given that our detection algorithm is such that this event happens when we have at least  $c_i$  entries inside  $W_i$  this probability is:

$$\begin{aligned}
P_{d_i} &= Pr(K_{W_i} + R_{W_i} \geq c_i) \\
&= \sum_{l=0}^{c_i-1} Pr(K_{W_i} \geq c_i - l) Pr(R_m = l) + Pr(R_{W_i} \geq c_i) \\
&\simeq \sum_{l=0}^{c_i-1} (1 - I_{1-p_k}(n_k - c_i + l + 1, c_i - l)) (1 - p_L)^{r_L(w+1)} \\
&\quad \cdot \binom{l + r_L(w+1) - 1}{l} \cdot p_L^l + I_{p_L}(c_i, r_L(w+1)),
\end{aligned}$$

where  $I_x(a, b)$  is the regularized incomplete beta function:

$$I_x(a, b) = \sum_{j=a}^{a+b-1} \binom{a+b-1}{j} x^j (1-x)^{a+b-1-j}.$$

Now we want to compute the probability that the number  $S_{L'}$  of fingerprint entries found in the log is above the threshold  $\Theta$ , this means to detect the fingerprint. The random variable  $S_{L'}$  is a sum of  $L'$  non-homogeneous dependent Bernoulli random variables, which we approximate by a binomial distribution [12] with parameters:

$$n_d = \left\lfloor \frac{(\sum_{i=1}^{L'} P_{d_i})^2}{\sum_{i=1}^{L'} P_{d_i}^2} + 1/2 \right\rfloor \quad \text{and} \quad p_d = \frac{\sum_{i=1}^{L'} P_{d_i}}{n_d}.$$

Now we can give an approximation of the probability of detection:

$$P_D = P(S_{L'} \geq \Theta) \simeq 1 - I_{1-p_d}(n_d - \Theta + 1, \Theta).$$

2) *Low  $\lambda_{FP}$  approximation:* When the probability that two successive requests from the desired user fall in the same window is very small, i.e.,  $Pr(\hat{Z}_{i+1} - \hat{Z}_i \leq w) \simeq 0$ , we can assume that  $c_i = 1 \forall i$ . This simplifies the resulting equations. This happens when  $\lambda_{FP}$  is several orders of magnitude smaller than  $1/w$  requests per second. In this case  $P_{d_i}$  takes the value:

$$P_{d_i} = P_Z(Z_i, W_i) + (1 - P_Z(Z_i, W_i)) \cdot I_{p_L}(1, r_L(w+1)),$$

and  $S_{L'}$  becomes a sum of  $L'$  homogeneous independent Bernoulli random variables. Therefore,  $S_{L'}$  is binomially distributed and the probability of detection becomes:

$$P_D = P(S_{L'} \geq \Theta) = 1 - I_{(1-P_{d_i})}(L' - \Theta + 1, \Theta).$$

### B. Probability of false positive

We want to achieve a very low false positive rate, in order to avoid accusing an innocent server.

1) *General Case:* here we study the probability of false positive without making any assumption.

The probability that  $c_i$  entries in the log appear in an interval of size  $w$  when no fingerprint is actually present is

$$P_{f_i} = Pr(R_{W_i} \geq c_i) = I_{p_L}(c_i, r_L(w+1)).$$

Again,  $S_{L'}$  is the sum of  $L'$  non-homogeneous dependent Bernoulli random variables, which can be approximated by a binomial distribution [12] with parameters

$$n_f = \left\lfloor \frac{(\sum_{i=1}^{L'} P_{f_i})^2}{\sum_{i=1}^{L'} P_{f_i}^2} + 1/2 \right\rfloor \quad \text{and} \quad p_f = \frac{\sum_{i=1}^{L'} P_{f_i}}{n_f}.$$

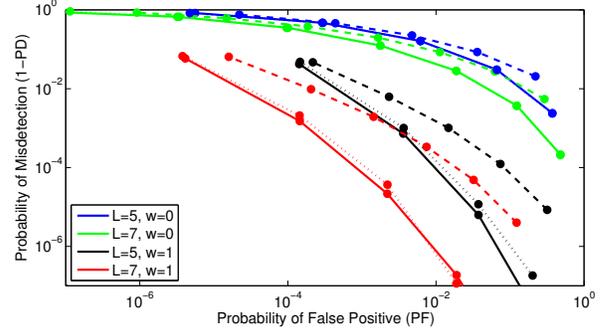


Fig. 4. ROC of the simulations for our research group's web server at the University of Vigo. Analytical (solid line), ideal conditions (dotted) and real conditions (dashed).

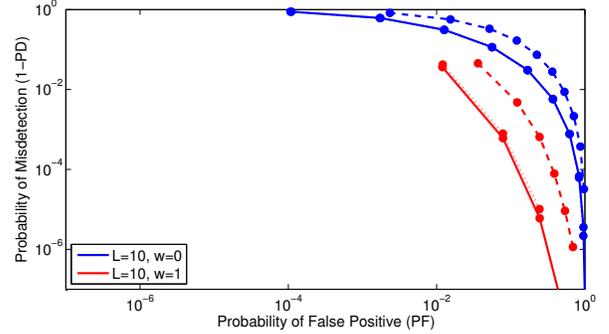


Fig. 5. ROC of the simulations for the web server at NASA's Kennedy Space Center. Analytical (solid line), ideal conditions (dotted) and real conditions (dashed).

Now we can give an approximation to the false positive probability:

$$P_F = P(S_{L'} \geq \Theta) \simeq 1 - I_{1-p_f}(n_f - \Theta + 1, \Theta).$$

2) *Low  $\lambda_{FP}$  approximation:* As before, when the rate of requests is small, we can approximate  $c_i = 1, \forall i$ . This implies that  $P_{f_i}$  takes the value:

$$P_{f_i} = I_{p_L}(1, r_L(w+1)).$$

Again  $S_{L'}$  becomes the sum of  $L'$  homogeneous independent Bernoulli random variables, which means it is binomially distributed and the false positive probability becomes:

$$P_F = P(S_{L'} \geq \Theta) = 1 - I_{(1-P_{f_i})}(L' - \Theta + 1, \Theta).$$

### C. Results

In order to validate our theoretical analysis we created a scenario where we can measure  $P_D$  and  $P_F$ . We implement a simulator which gives us the probabilities of detection and of false positive in two situations. The first is *ideal conditions* (i.e.,  $\{R_{W_i}\}$  is a sequence of iid NB random variables and  $\{\varepsilon_i\}$  is a sequence of iid random variables distributed according to Figure 2); the second is *real conditions*, where the log is taken from a web server and  $\{\varepsilon_i\}$  comes from the data of the last experiment in Section IV, preserving any existing autocorrelation.

Each experiment is simulated 10,000,000 times. We run this simulator for two different cases. The first one is the research group's web server at the University of Vigo where fingerprints are built by  $L=5$  and  $L=7$  entries. We consider two window sizes,  $w=0$  and  $w=1$  seconds. The second case is the web server at NASA's Kennedy Space Center. As it is a busier server, we make  $L = 10$  to bring the false positive rates to acceptable levels, and we also use windows of size  $w = 0$  and  $w = 1$  seconds.

The results are shown in Figures 4 and 5, where we can see that the analytical results closely match the simulated ideal conditions. This supports our theoretical analysis. We also see that only a discrete set of points (i.e., those marked with circles) are generated; this is due to the threshold  $\Theta$  only taking integer values from 1 to  $L$ . Note that in some cases some points are missing, because it is not possible to measure those false positive probabilities (since we have run 10,000,000 experiments, probabilities under  $10^{-7}$  could not be measured).

We can also see the performance loss due to non-ideal conditions. This is the gap between the dashed and the solid lines in Figures 4 and 5. The horizontal shift comes from assuming iid entries on the log and ignoring the trends, while the vertical shift comes from assuming independence on  $\{\varepsilon_i\}$ .

## VIII. CONCLUSIONS

This paper proposes a method to leave a timing fingerprint in the log of a hidden server. This fingerprint can be used as evidence that the server indeed hosted a particular content even after this content has been deleted. We note that, although our experiments have been carried on a Tor hidden server, the underlying principles of the attack are valid for any web server.

Our approach is based on sending HTTP requests to the hidden server and storing the "date field" of the responses we get. To detect the fingerprint we check whether there is a logged entry in a window around the time that appears in these responses. If we find a number of entries above a pre-defined threshold we decide there is a fingerprint in the log. We provide analytical expressions for the probabilities of detection and false positive. We further show that the resulting expressions admit a simplification when the sent requests are sparse, and finally provide an empirical validation of our results.

Ongoing research considers the observed differences between the HTTP response date field and the log time, which could be better modeled by including a study of their autocorrelation. This shall hopefully improve the analytical approximations and bring them closer to the empirical results.

## ACKNOWLEDGMENTS

Research supported by the European Union under project REWIND (Grant Agreement Number 268478), the European Regional Development Fund (ERDF) and the Spanish Government under projects DYNACS (TEC2010-21245-C02-02/TCM) and COMONSENS (CONSOLIDER-INGENIO 2010 CSD2008-00010), and the Galician Regional Government under projects Consolidation of Research Units 2009/62,

2010/85 and SCALLOPS (10PXIB322231PR), and by the Iberdrola Foundation through the Prince of Asturias Endowed Chair in Information Science and Related Technologies. This work was supported in part by the Concerted Research Action (GOA) Ambiorics 2005/11 of the Flemish Government and the IAP Programme P6/26 BCRYPT.

C. Troncoso is a research assistant of the Fund for Scientific Research in Flanders (FWO). The authors are really grateful to the people who made their logs available: R. Fridman, J. R. Troncoso-Pastoriza, E. Fogel, L. Bottomley, J. Dumoulin, S. Balbach, M. Arlitt and C. Williamson and to the ACM SIGCOMM for making them available.

Thanks to all the people that have reviewed this paper or have helped in any other way: N. Mathewson, R. Dingeldine, B. Shebaro, P. Jamkhedkar, M. Limon.

## REFERENCES

- [1] I. Clarke, O. Sandberg, B. Wiley, and T. Hong, "Freenet: A distributed anonymous information storage and retrieval system," in *Designing Privacy Enhancing Technologies*, ser. Lecture Notes in Computer Science, H. Federrath, Ed. Springer Berlin / Heidelberg, 2001, vol. 2009, pp. 46–66.
- [2] R. Dingeldine, N. Mathewson, and P. Syverson, "Tor: the second-generation onion router," in *Proceedings of the 13th conference on USENIX Security Symposium - Volume 13*, ser. SSYM'04. Berkeley, CA, USA: USENIX Association, 2004, pp. 21–21.
- [3] O. Berthold, H. Federrath, and M. Köhnstopp, "Project anonymity and unobservability in the internet," in *Proceedings of the tenth conference on Computers, freedom and privacy: challenging the assumptions*, ser. CFP '00. New York, NY, USA: ACM, 2000, pp. 57–65.
- [4] W. J. Sullivan, "2010 free software awards announced," May 2011. [Online]. Available: <http://www.fsf.org/news/2010-free-software-awards-announced>
- [5] B. Shebaro, F. Perez-Gonzalez, and J. R. Crandall, "Leaving timing-channel fingerprints in hidden service log files," *Digital Investigation*, vol. 7, no. Supplement 1, pp. S104 – S113, 2010, the Proceedings of the Tenth Annual DFRWS Conference.
- [6] M. Liberatore, B. N. Levine, and C. Shields, "Strengthening forensic investigations of child pornography on p2p networks," in *Proceedings of the 6th International Conference*, ser. Co-NEXT '10. New York, NY, USA: ACM, 2010, pp. 19:1–19:12.
- [7] K. Loesing, W. Sandmann, C. Wilms, and G. Wirtz, "Performance measurements and statistics of Tor hidden services," in *The 2008 International Symposium on Applications and the Internet*. Turku, Finland: IEEE, July 2008, pp. 1 – 7.
- [8] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee, "Hypertext Transfer Protocol – HTTP/1.1," RFC 2616 (Draft Standard), Internet Engineering Task Force, Jun. 1999, updated by RFCs 2817, 5785. [Online]. Available: <http://www.ietf.org/rfc/rfc2616.txt>
- [9] M. F. Arlitt and C. L. Williamson, "Web server workload characterization: the search for invariants," in *Proceedings of the 1996 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, ser. SIGMETRICS '96, 1996, pp. 126–137.
- [10] N. Johnson, A. Kemp, and S. Kotz, *Univariate discrete distributions*, ser. Wiley series in probability and mathematical statistics. Applied probability and statistics. Wiley, 2005.
- [11] L. J. Simon, "Fitting negative binomial distributions by the method of maximum likelihood," in *Proceedings of the Casualty Actuarial Society*, vol. XLVIII, Arlington, VI, 1961, pp. 45–54.
- [12] S. Y. T. Soon, "Binomial approximation for dependent indicators," *Statistica Sinica*, vol. 6, pp. 703–714, 1996.