Non-intrusive Tracing in the Internet

Alina Olteanu, **Yang Xiao** , **Jing Liu**

Department of Computer Science,

The University of Alabama, Tuscaloosa, AL 35487 USA

Emails: aolteanu@cs.ua.edu, yangxiao@ieee.org , jliu39@crimson.ua.edu

Thomas M. Chen

School of Engineering,

Swansea University, Swansea, Wales, UK SA2 8PP

Email: t.m.chen@swansea.ac.uk

C. L. Philip Chen,
Faculty of Science of Tech.,
University of Macau
Philip.Chen@ieee.org

Summary— Intruders which log-in through a series of machines when conducting an attack are hard to trace because of the complex architecture of the Internet. The thumbprinting method provides an efficient way of tracing such intruders by determining whether two connections are part of the same connection chain. Since many connections are transient, and therefore short in length, choosing the best time interval to thumbprint over can be an issue. In this paper, we provide a way to shorten the time interval used for thumbprinting. We then study some special properties of the thumbprinting function.

We also study another mechanism for tracing intruders in the Internet based on a timestamping approach which passively monitors flows between source and destination pairs. Given a potentially suspicious source, we identify its true destination. We compute the error probability of our algorithm

and show that its value decreases exponentially as the observation time increases. Our simulation results show that our approach performs well.

1. INTRODUCTION

Constant change is perhaps a guiding principle the Internet. Recent advances in technology have lead to significant growth of the Internet by factors of 10^3 and 10^6 in backbone speed and the number of hosts, respectively [7]. Since the public expansion of the Internet in 1990, many new challenges have surfaced; among them, operations between un-trusted end-points, more demanding applications, and less sophisticated users have caused severe stress on Internet requirements. Furthermore, the number of attacks on networked computer systems has been growing exponentially from year to year. When considering the task of tracing intruders in the Internet, three main challenges must be taken into account. First, attackers hide their origin by making use of the Internet's architecture. By using different hosts which belong to different countries and administrative domains to route malicious acts, intruders' actions become extremely difficult to trace. Second, the data collected from an Internet trace is usually incomplete or has missing values. For example, different domains of Internet service providers (ISPs) may not share data due to access issues, such as when they are in different counties. Finally, routes change frequently, packets are lost, and the network latency and time to convergence can be significantly increased due to routing instability.

To deal with such problems, two tracing mechanisms exist [4]: (1) methods of keeping track of all individuals and accounting for all activities, and (2) reactive tracing, in which no global accounting is attempted until a problem arises. In our view, the first mechanism is most related to network accountability and the second mechanism is most related to network forensics. Network-based tracing and host-based tracing are two common approaches for reactive tracing in connection chains. Host-based tracing involves one tracing system per network host [4], and a chain of connection hosts can be known via (1) host communications [3] or (2) reversing the attack chain by breaking the hosts in reverse order [5]. Host-based tracing schemes suffer when an extended connection crosses a host not running the system [4].

Network-based tracing has the advantage of not relying on hosts which can be untrusty or requiring host participation. Instead, network-based tracing uses the invariance of connections at higher protocol layers (such as the transport layer) to establish whether two connections are part of the same connection chain. We study two approaches for network-based tracing. The first approach is based on the idea of a connection thumbprint. A thumbprint is very similar to a checksum or the computed summary of the content of a connection [4], [2]. The thumbprint summarizes the content of a connection using only a small amount of data while simultaneously preserving the uniqueness of the connection. Therefore, thumbprints of related connections are similar and can lead to the construction of a connection chain. However, the thumbprinting method is dependent on the duration of the connection; so the time period used for thumbprinting is very important especially in the case of transient flows. In this paper, we provide a way to shorten the time interval required for thumbprinting. We study the tradeoff introduced by using a smaller time interval and its effect on basic thumbprint properties such as sensitivity and robustness.

Since thumbprinting relies on the content of the packages and therefore can be counterattacked by encryption, we study a second mechanism for tracing Internet intruders which involves passively monitoring flows between source and destination pairs in the Internet, a process similar to [1]. This approach is based on monitoring the transmission activities of nodes and does not interfere with network operations. A one-hop communication graph can be constructed by matching transmission timestamps and acknowledgements. Based on this graph, the nodes that are part of a connection chain are those which communicate at a sufficiently high rate. Our approach is different from the approach in [1] in that we make fewer statistical assumptions and we use clearer, simpler derivations. In addition, because we are working with the Internet, we face other challenges, like the availability of only partial information and the route instability mentioned above. To account for this, we assume that the information on every link has a certain probability of availability.

Our motivations in this paper can be derived from the above discussions as follows:

1. to find a way to improve current thumbprinting method for transient traffic flows by shortening the time interval required for thumbprinting;

2. to study the tradeoff introduced by using a smaller time interval and its effect on basic thumbprint properties such as sensitivity and robustness.

According to these motivations, relevant contributions are made as follows:

1. To found out the equation for sampling traffic in any time interval and the properties that can be used for identifying related flows so that we can shorten the time interval required for thumbprinting;
2. To obtain the error probability of our algorithm and show that its value decreases exponentially as the observation time increases.
3. to consider network background noise and redesigned a passive tracing scheme accordingly.
4. to provide numerical results and simulation results.

The remainder of the paper is organized as follows. Section 2 presents significant work related to our problem; Section 3 presents our results concerning minimizing the thumbprinting interval; Section 4 contains a detailed introduction to the tracing algorithm followed by an evaluation of its consistency in terms of error probability; Section 5 presents some graphical interpretations of the derivations; Section 6 provides simulations; Section 7 is the paper's conclusion.

2. Related Work

Paper [4] presents an IP traceback method based on the thumbprint idea. Often, attacks are conducted by logging through a chain of host machines. In this way, intruders make use of the Internet architecture to hide their true origin. The thumbprint technology, which is the summary of a connection, is used to compare connections and therefore trace the source of such an attack. A thumbprint is computed for each time interval, typically a minute, of each connection [4]. Thumbprinting has the advantage of preserving the characteristics of a connection while being cheap to compute and requiring little storage. Similar methods like checksum and message digest have two main disadvantages. Firstly, an error in the content leads to a different value being computed, and thus these methods are not robust. Secondly, they are not additive (i.e., two successive values cannot be combined to provide a new value for a longer interval). Other compression schemes require far more storage space than thumbprinting.

As the thumbprinting method is dependent on the duration of the connection, it is of importance to shorten the time interval used for thumbprinting, especially in the case of transient flows. This is the purpose of our derivation in Section 3.

However, fingerprinting techniques are vulnerable and may not be of interest when the attacker uses encryption at every hop in the chain. To deal with such problems, techniques based on packet timing information are more suitable.

Paper [1] presents a mechanism for tracing intruders in anonymous MANETs by passively monitoring flows between source and destination pairs. The algorithm is concerned with tracing the destination of a certain source which is considered to be suspicious according to a higher layer intrusion detection protocol. Because anonymity of nodes in the MANET is assumed and the monitoring is done in a non-intrusive way, this research only considers the timestamps of packets on adjacent links, and uses these to establish causal relationships between packets. A graph is used to model the communications among nodes. Based on traffic analysis and information carried in the packets, the graph is partitioned into two parts, one containing the set of possible destinations, and one with which the source does not communicate. The following assumptions are made about the distribution of traffic: the transmission activities on each link are assumed to follow a Poisson process, the rate of the flow to be traced is sufficiently high, and the duration of the observation is long enough. Therefore, the method will not work for low-rate or transient flows.

Much like [1], we focus on tracing flows between pairs (source, destination), but we do so with the Internet. Unlike [1], we assume differences in data due to propagation delays, clocks being skewed during observation at multiple nodes at the same time, and the data which is either missing from observations or observed erroneously. We account for this by setting a probability at which information is available on every link. We also make fewer assumptions about the rate of the Poisson process, and the calculations are significantly simplified. Some other related work includes [12-46].

3. THUMPRINTING

The thumbprinting approach is based on the fact that, at higher protocol layers, the content of a connection is invariant at all points of the chain. The thumbprint summarizes the content of the connection using only a small amount of data while preserving the uniqueness of the connection. We first define the terminology and then present a method for shortening the time interval required for thumbprinting and some interesting properties of the thumbprint function.

3.1 Original Algorithm

A thumbprint is a function of a connection which preserves the unique characteristics of the connection. We use the notations and definition of a thumbprint from [4]. Consider a sequence of transmitted characters $a_1, a_2, \ldots, a_L$ which needs to be thumbprinted. Consider also the function $\alpha : \mathscr{A} \to \mathbf{R}^K$, which takes a character and returns a vector of K real numbers. Let $\alpha_k(\bullet)$ denote the k-th component of the function. Here K is a short fixed number representing the number of thumbprint components. If we consider L as the period of a connection, we can associate frequencies $\overline{f} = (f_1, f_2, ..., f_L)$ to our character sequence. Then the thumbprint is defined as [4]:

$$T_k = \sum_{i=1}^{L} \alpha_k(a_i) f_i, \quad k \in \{1,2,...,K\}. \qquad (1)$$

The thumbprint is thus a linear combination of the frequencies of characters and their corresponding weights. T_k represents the kth component of the K-dimensional thumbprint vector.

Each thumbprint component, $T_k(C,t)$, is a function of a specific connection C and the time interval t, in which the thumbprints have been computed. From [4], we also know that the comparison of two sets of thumbprints, $T_k(C,t)$ and $T_k(C',t)$, $k \in \{1,2,...,K\}$, for two different connections C and C', is given by the following formula:

$$\delta_t(C,C') = \log\left(\prod_{k=1}^{K} \left|T_k(C',t) - T_k(C,t)\right|\right) \qquad (2)$$

δ_t represents the logarithm of the product of component differences for two thumbprints in a specific time interval t. A large absolute value of δ_t implies that the two connections are related, while a small absolute value suggests that they are independent.

3.2 Minimizing The Thumbprinting Interval

In this section, based on the results in section 3.1 [4], we try to find the best period of time T, over which to thumbprint. The length of time divisions for the experiments in [4] is 1 time unit. However, many connections in the Internet are transient and therefore short in length; in these cases a shortened thumbprinting interval is needed. Hence, we need to find a new equation, rather than (1), to better describe the characteristic of each link or session in terms of more precise time duration.

We start with the thumbprint function given by (1), and we then account for the time in which the thumbprint has been computed. This gives us the following function:

$$\sum_{a,t} \alpha_k(a,t) f(a,t) . \qquad (3)$$

We will use the function given by (3) above as our thumbprinting function.

We now show the results of changes to variables $a = Ly$ and $t = T\tau$. This transforms our time interval from $[0,T]$ to $[0,1]$ and the samples from every time unit to every $1/T$ time units. Thus the thumbprinting function in (3) can be approximated by the following integral:

$$\iint_{[0,L]\times[0,T]} f(a,t)\alpha_k(a,t)\,da\,dt . \qquad (4)$$

We have:

$$\iint_{[0,L]\times[0,T]} f(a,t)\alpha_k(a,t)\,da\,dt \approx LT \sum_{a,t} \alpha_k(a,t) f(a,t) \Big/ (LT)$$

$$\overset{\substack{a=Ly \\ t=T\tau}}{=} LT \sum_{y,\tau} \alpha_k(Ly,T\tau) f(Ly,T\tau) \Big/ (LT)$$

$$= LT \iint_{[0,1]\times[0,1]} f(Ly,T\tau)\alpha_k(Ly,T\tau)\,dy\,d\tau$$

The time period has become interval $[0,1]$.

Since thumbprinting over a one unit time interval may not provide the best results, we further provide a way to transform interval $[0,1]$ into an arbitrary time interval $[p,q]$. In the following calculus, we use $y=(u-m)/(n-m)$ and $\tau=(v-p)/(q-p)$. We have:

$$LT\iint_{[0,1]\times[0,1]} f(Ly,T\tau)\alpha_k(Ly,T\tau)dyd\tau$$

$$=LT\iint_{[m,n]\times[p,q]} f(L(u-m)/(n-m),T(v-p)/(q-p))\cdot$$

$$\alpha_k(L(u-m)/(n-m),T(v-p)/(q-p))\left|\frac{\partial(y,\tau)}{\partial(u,v)}\right|dudv$$

$$=LT\iint_{[m,n]\times[p,q]}\frac{f(L(u-m)/(n-m),T(v-p)/(q-p))}{(n-m)(q-p)}\cdot$$

$$\frac{\alpha_k(L(u-m)/(n-m),T(v-p)/(q-p))}{(n-m)(q-p)}dudv\,. \qquad (5)$$

In equation (5), the time period has become interval $[p,q]$. This interval can be suitably chosen to accommodate connections with low/high data rates and transient connections in order to give the best performance possible under different scenarios.

3.3 Properties of the Thumbprinting Function

In the following, we describe a way to easily compute the value of the integral given by (4) by using the mean value theorem [10]. We further study some special properties of the thumbprinting function.

Consider the right-hand side of equation (5).

Let us choose u_0 and v_0 to be the points toward which we concentrate intervals $[m,n]$ and $[p,q]$, respectively. By making $m,n\to u_0$ and $p,q\to v_0$ (i.e. $n-m\to 0$ and $q-p\to 0$), we have:

$$\iint_{[0,L]\times[0,T]} f(a,t)\alpha_k(a,t)dadt = LT\iint_{[0,1]\times[0,1]} f(Ly,T\tau)\alpha_k(Ly,T\tau)dyd\tau = \frac{LT\iint_{[m,n]\times[p,q]} f\left(L\frac{u-m}{n-m},T\frac{v-p}{q-p}\right)\alpha_k\left(L\frac{u-m}{n-m},T\frac{v-p}{q-p}\right)dudv}{(n-m)(q-p)}$$

$$= LTf\left(L\frac{u_0 - m}{n - m}, T\frac{v_0 - p}{q - p}\right)\alpha_k\left(L\frac{u_0 - m}{n - m}, T\frac{v_0 - p}{q - p}\right)$$

$$= LT\, h\left(L\frac{u_0 - m}{n - m}, T\frac{v_0 - p}{q - p}\right). \qquad (6)$$

We have considered $h = f\alpha_k$ where h is a continuous function. Let $I = LT \iint_{[0,1]\times[0,1]} f(Ly, T\tau)\alpha_k(Ly, T\tau)dyd\tau$. If h is a C^n function where $n \geq 1$, then the set of points $\gamma = \{(u_0, v_0); I = LT(f\alpha_k)(u_0, v_0)\}$ represents a C^n curve which, according to the Implicit Function Theorem [6], ensures the existence of a function ϕ such that $v_0 = \phi(u_0), u_0 \in J$, where J is an interval included in $[m, n]$. This way we can express one of the variables u_0 and v_0 as a function of the other.

The last equality in (6) is true for any $(u_0, v_0) \in [m, n] \times [p, q]$ on the curve γ.

We have thus shown that the value of the integral in (4) can be found by computing the value of the function $f\alpha_k$ at any point belonging to curve γ.

Furthermore, if we choose:

$$n = m + \varepsilon, u_0 = m + \varepsilon t, 0 < t < 1, q = p + \delta,$$
$$\phi(u_0) = p + \lambda(t)\delta,$$

we have:

$$I = LT(f\alpha_k)\left(L\frac{\varepsilon t}{\varepsilon}, T\frac{\lambda(t)\delta}{\delta}\right)$$

$$= LTf(Lt, T\lambda(t))\alpha_k(Lt, T\lambda(t)), t \in J.$$

We have shown that on the curve $\gamma = \{(u_0, \phi(u_0)); u_0 \in J\}$ we have:

$\alpha_k(u_0, \phi(u_0)) = \dfrac{I}{LTf(u_0, \phi(u_0))} = const.$. In order to determine the constant, we simply need to compute I. Therefore, α_k and f are inversely proportional on curve γ.

We now introduce an abstraction of the derivations used above by using a mean value operator. The use of this operator generalizes the procedure of associating the function h with its mean value. Such an operator can be defined as follows:

$$T(h)(0) = h(0) = \lim_{x \downarrow 0} \frac{\int_0^x h(t)dt}{x} = \lim_{x \downarrow 0} T(h)(x),\ h \in C([0,b])$$, for one variable and

$$T(h)(0,y) = \lim_{x \downarrow 0} \frac{\iint_{[0,x]\times[0,y]} h(t_1,t_2)dt_1 dt_2}{xy} = \frac{\int_0^y h(0,t_2)dt_2}{y},\ y \neq 0,$$

$$T(h)(x,0) = \frac{\int_0^x h(t_1,0)dt_1}{x},$$

$T(h)(0,0) = h(0,0)$, for two variables.

Operator T associates function h with another function that is equal at every point x to the mean value of function h on interval $[0,x]$. In Section 5, we will see the computation has been simplified by using this mean value approach.

4. Tracing the True Destination Of A Source

Since many attackers use encryption at every hop in the chain, the same packet will have different content on every link and fingerprinting techniques are not appropriate in this case. Therefore, we present a mechanism from paper [1] for tracing intruders in anonymous MANETs which based on using transmission timestamps to passively monitor the flows between source and destination pairs. Such an approach can be implemented by using sensors equipped with energy detectors to measure transmission timestamps and then distributing them over the field of interest. All measurements are fused and processed by a centralized monitor at a fusion center.

We first introduce the algorithm and its main features and then analyze the algorithm in terms of the error probability of detecting the true destination of a source. We show that the error probability decays exponentially with the observation time and also prove a similar result when the number of observed graphs goes to infinity.

4.1 Original Algorithm

Our derivation is based on a variation of the tracing algorithm from [1]. In [1], the approach for tracing the destination of a source is two fold. First, based on the transmission activities on adjacent links, it can be determined whether these links are part of the same flow. Then, a set of possible destinations is computed. Second, the intersection method is used: using the changes in topology, some nodes are eliminated, leading to a smaller set of possible destinations.

It is assumed that the transmission activities on each link follow a Poisson process $\boldsymbol{S}$, and its realization is denoted by $\boldsymbol{s}$. Therefore a realization of the transmission timestamps of data packets on link 1 will be $\boldsymbol{s}_1 = (s_1(1), s_1(2), s_1(3), \ldots)$. Here uppercase letters denote random variables, lowercase letters realizations, boldface letters vectors, and plain letters scalars.

In the following, we briefly introduce the idea behind the traffic analysis method and the intersection of different topologies method from [1].

For traffic analysis, consider two realizations $(\boldsymbol{s}_1, \boldsymbol{s}_2)$ of the transmission activities (timestamps) on two adjacent links. Let m and n be the indices of the current timestamps on links 1 and 2, respectively ($0 \le m \le \delta_1, 0 \le n \le \delta_2$). These sets of timestamps are matched against each other sequentially by assessing the difference $s_2(n) - s_1(m)$. If $s_2(n) - s_1(m) \le \Delta$, where Δ is a predefined maximum delay, and this difference is nonnegative, then the two timestamps match. The matching timestamps from the two connections form a pair of sequences $(\boldsymbol{f}_1, \boldsymbol{f}_2)$. Given the number of matching timestamps on a link, the empirical rate on that link can be now estimated. Specifically, if $\boldsymbol{f}_i$ contains $|F_i|$ timestamps over time T, then the empirical rate is $|F_i|/T$.

Let $\tau \ge 0$ be a given rate. Given a graph sourced at j, by repeatedly applying the matching timestamps algorithm for pairs of adjacent links, and selecting only the flows which support rates of at least τ, a subgraph of the initial graph can be obtained. This new graph will contain only nodes which node j can talk to at a rate higher or equal to τ. The method, called Trace Destination (TD), is applied for the graph sourced at O, and a set of possible destinations is obtained. However, node O can only be a relay node for some of these flows. In order to trace down the flows that are going through O, but did not originate at O, TD is applied for all immediate predecessors of O and the corresponding graphs sourced at these nodes. The set of obtained destinations is then subtracted from the previously obtained set of destinations of O.

To speed up the algorithm's convergence, a changing topology feature is used. Basically, the algorithm TD is applied for every communication graph that is observed. For each observed graph, a set of destinations if obtained. The final set of possible destinations is the intersection over all sets from all observed graphs. Out of this final set, the node which appears most frequently in all topologies is selected. If there are multiple possible destinations with the same number of appearances across all different topologies, then one node is selected at random.

4.2 Analysis

Our tracing algorithm is a variation of the one from Section 4.1, in which we account for the lack of information in the observations. We model the lack of information by considering that the information on each link is available with a certain probability. We denote this probability by $p_k, k \in \{1,2,...,K\}$, where K is the number of links in the considered path. In this modified algorithm, only flows which support rates of at least $r_k / p_k \geq \tau$ are selected from the initial graph.

It is now useful to study how well our variation of the algorithm from Section 4.1 converges. We thus study the probability P_e with which the algorithm finds the destination erroneously (called error probability) and analyze its asymptotic behavior over time T. We would like to mention that even though we assume Poisson distribution of packets on a link for our specific derivation, the algorithm and analysis here can be applied to different types of traffic.

Our error probability derivation is also inspired from [1]. Assume that the transmission activities on each link follow a Poisson process of rate $R < 1$. It is well-known that the Poisson distribution of the number of events in a time interval $(t, t+t']$ is given by the following relation:

$$P[(m_{t+t'} - m_t) = n] = \frac{e^{-Rt'}(Rt')^n}{n!},\ n = 0,1,...,$$

where $m_{t+t'} - m_t$ represents the number of events in time interval $(t, t+t']$ (see for example [11]).

Let O be the source whose destination we are trying to find and let θ be the true destination of O. We denote the destination rendered by our algorithm with $\hat{\theta}$. We then derive the probability that $\hat{\theta}$ is not the true destination of O. Let n_k be the number of timestamps taken in time interval (m_k, m_{k+1}), corresponding to link $(k, k+1)$. We denote the total number of timestamps taken for one flow by N_1.

By definition,

$$P_e := P_e(TD) = P(\theta \neq \hat{\theta})$$
$$= P(\theta \text{ is not correctly detected by algorithm TD}).$$

According to [1], Theorem 4.4, the fact that the algorithm does not find the true destination is due to one of three possibilities:

A) θ is not detected because the empirical rate along the path from O to θ is less than τ; we will denote the probability of this event by $P(A)$;

B) θ is mistaken for a relay node because there is some node j which is a successor of θ, for which the empirical rate from O to j is greater than τ. We denote the probability of this event by $P(B)$;

C) θ is incorrectly detected as being the destination of a flow originating in some predecessor of O. We denote the probability of this event by $P(C)$;

In cases B and C, all the empirical probabilities are greater than or equal to τ. So

$$P_e \leq P(A) + P(B) + P(C). \qquad (7)$$

In the following, we establish an upper bound for $P(B)$. The empirical rate on the path from O to j is given by $N_1(T)/T$.

$$P(B) \leq Ke^{-\frac{N_1(T)}{T}T},$$

where K is a constant related to $\mathrm{card}(G)$.

$$\frac{N_1(T)}{T} \geq r_k \geq p_k \tau \text{ for all } k \;\; => \frac{N_1(T)}{T} \geq \tau \frac{\sum p_k}{\# \text{ of links}} = \tau\, \tilde{p}.$$

Using the fact that $-\frac{N_1(T)}{T} \leq -\tau\, \tilde{p}$, we obtain:

$P(B) \le Ke^{-N_1(T)} \le Ke^{-\tau T\,\tilde{p}}$.

Similarly, for $P(C)$we have: $P(C) \le Ke^{-\tau T\,\hat{p}}$.

For $P(A)$, the assumption is that there is at least one link along the path from O to θ with an empirical rate which is less than τ . Suppose j_0 is the first node on the path connected to a link whose rate is less than τ . Therefore all links along the path up to j_0 have rates greater than or equal to τ . Let N_1 be the total number of timestamps taken during the observation period T .

We denote by λ the rate of the Poisson process defined by the recorded timestamps. Then the rate on an arbitrary interval $[m_k, m_{k+1}]$ is $\lambda(m_{k+1} - m_k)$.

$$P(A) \le \prod_{k=0}^{j_0-1} \frac{[r_k(m_{k+1} - m_k)]^{n_k}}{n_k!} e^{-r_k(m_{k+1}-m_k)} \prod_{k' \in N^+(j_0)} e^{-r_{k'}(m_{k'+1}-m_{k'})}.$$

$N^+(j_0)$ denotes all successors of node j_0 on links with rates of at least R . Notice that in the first product $r_k \ge p_k\tau$ for all k , while all $r_{k'}$ from the second product are less than $p_k\tau$. On the other hand, $r_{k'} \ge (N_1(T) - N(j_0))/(T - m(j_0))$. We thus have:

$$\begin{aligned} &P(A) \le C_{j_0} \exp[-\tau(\sum_{k=0}^{j_0-1} p_k(m_{k+1} - m_k)] \exp[-\frac{N_1(T) - N(j_0)}{T - m(j_0)} \\ &(T - m(j_0)] = C_{j_0} \exp(N(j_0) \exp[-\tau(\sum_{k=0}^{j_0-1} p_k(m_{k+1} - m_k)] \\ &\exp(-N_1(T)) = K_1(j_0, \tau) \exp(-N_1(T)) \\ &= K_1(j_0, \tau) \exp(-T\rho(T)), \end{aligned}$$

where

$$C(j_0) = \prod_{k=0}^{j_0-1} \frac{r_k(m_{k+1} - m_k)^{n_k}}{n_k!} \text{ and}$$

$$K'(j_0, \tau) = C(j_0) e^{N(j_0) - \tau m_{j_0}} .$$

Here we let $\rho = \lim_{T \to \infty} N_1(T)/T > 0$.

Therefore $P(A)$ decreases exponentially with the observation time.

Using equation (7), we now have:

$$P_e \le K'(j_0,\tau)e^{-\rho T} + \mathrm{card}(\mathrm{G})e^{-\tau T\tilde{p}} + \mathrm{card}(\mathrm{G})e^{-\tau T\hat{p}}$$
$$= e^{-T\tau\min\{\tilde{p},\hat{p}\}}\left(K'(j_0,\tau)+2K\right).$$

We can now investigate the convergence rate of our algorithm by showing the asymptotic decay rate of the error probability as time increases.

By logarithming the above inequality we obtain:

$$\lim_{T\to\infty}\left(-\ln P_e/T\right) \ge \tau\min(\tilde{p},\hat{p}) > 0 .$$

Compared to the upper bound for the error of algorithm TD, which is τ (corresponding to the case $p_k = 1$ for all k), the modified algorithm has upper bound $\tau\min(\tilde{p},\hat{p})$.

The minimum is strictly positive as the minimum of a finite number of strictly positive values.

We can see that the algorithm has an exponential convergence rate on error probability over time *T*. That means, in our work, the error rate of finding correct connection pairs will decrease exponentially while the observing time goes up.

4.3 Generalization

We now prove a generalization of the previous result regarding the error probability when the number M of observable graphs goes to infinity.

If in the mobile case $\theta \ne \hat{\theta}$ for all M cases, then in each observable graph G_i, $i=\overline{1,M}$, we have $\theta \ne \hat{\theta}$. If we consider these M events independently, we have:

$$P_e(T,M) \le e^{-2TM}\left[2\mathrm{card}(G)e^{\ln T+\ln 2} + K(j_0,2)e^{-\tau T+\ln T+\ln 2}\right]^M$$
$$= e^{-2TM}\left[2\mathrm{card}(G)\tau T + K(j_0,2)e^{(\tau-\rho)T}\right]^M,$$
where $\rho(N,T) > 0$.

If we consider T to be fixed and sufficiently large, and the number M of graphs going to infinity, the error probability again decays exponentially. We have:

$$P_e(T,M) \le \left(\mathrm{card}(G)e^{-T\min(\rho,\tau\tilde{p},\tau\hat{p})}\right)^M$$
$$= \left(K'(j_0,\tau)+2\mathrm{card}(G)\right)^M e^{-MT\tau\min(\rho,\tilde{p},\hat{p})}.$$

So P_e decreases with the speed of $e^{-\tau TM}$. We then have:

$$\lim_{M\to\infty}\left[-\ln\left(P_e(T,M)/M\right)\right]\geq$$
$$\lim_{M\to\infty}\left[\tau TM\min\left(\widetilde{p},\hat{p}\right)+M\ln\left(K'(j_0,\tau)+2\mathrm{card}(G)\right)\right]/M$$
$$=\tau T\min\left(\widetilde{p},\hat{p}\right)+\ln\left[K'(j_0,\tau)+2K\right]>0.$$

We have obtained a bound on P_e which is strictly smaller than 1 and thus non-trivial. It means given a sufficient observation time, no matter how many the observation objects are, our tracing scheme still can get the correct connection pairs in a low error rate. In another words, it is scalable.

5. Numerical Evaluation

Given two connections *C* and *C'*, Fig. 1 represents the frequency of occurrences of the same character a_i at every minute in the two connections over a period of time L. f denotes the frequency of a_i in connection *C* and g represents the frequency of the same character in *C'*. The area between the graphs of f and g approximates the sum of the differences between the frequencies of characters in the two connections from Section 3: *B*: $\sum_{i=1}^{L}\left(f(a_i)-g(a_i)\right)p(a_i)/L$.

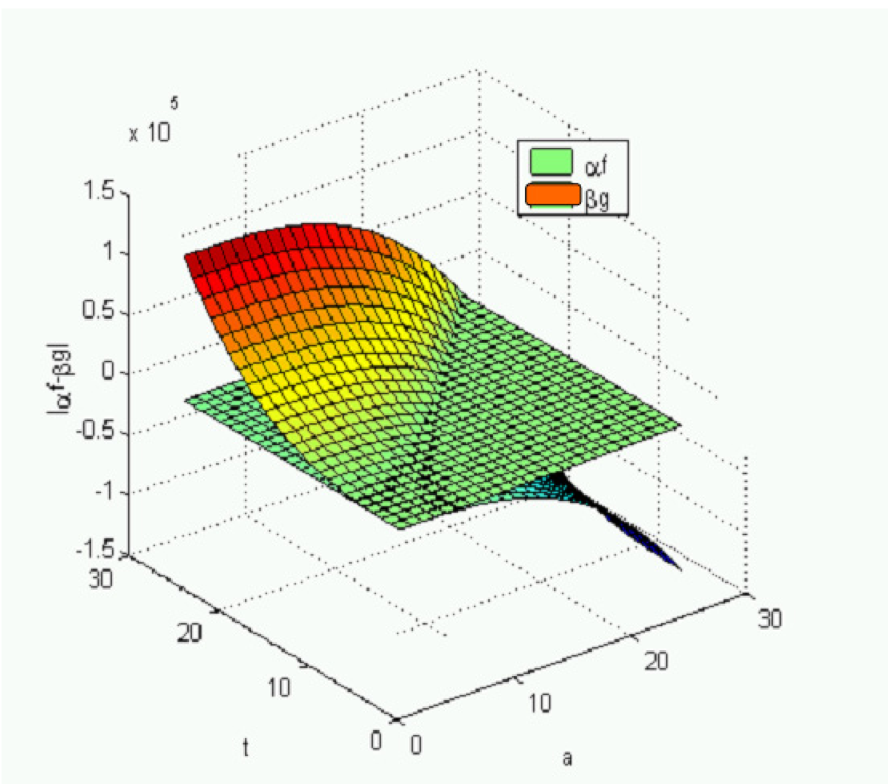

Fig. 1 Maximum difference between f and g in absolute value.

Fig. 2 represents the volume as given by the integral $\iint_{[0,L]\times[0,T]} f(a,t)\alpha_k(a,t)\,dadt$. The volume of the solid in Fig. 2 is bounded above by $f(a,t)\alpha_k(a,t)$ and below by $[0,L]\times[0,T]$ in the xy plane.

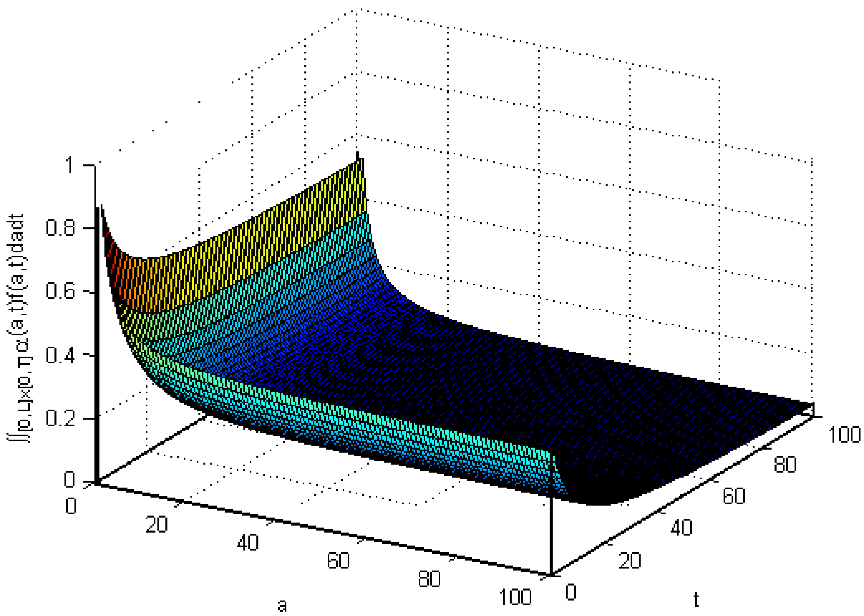

Fig. 2 Volume of the solid that lies below function $f(a,t)\alpha_k(a,t)$ on $[0,L]\times[0,T]$.

The volume of the solid in Fig. 3 is bounded above by $f(Ly,T\tau)\alpha_k(Ly,T\tau)$ and below by $[0,1]\times[0,1]$ in the xy plane.

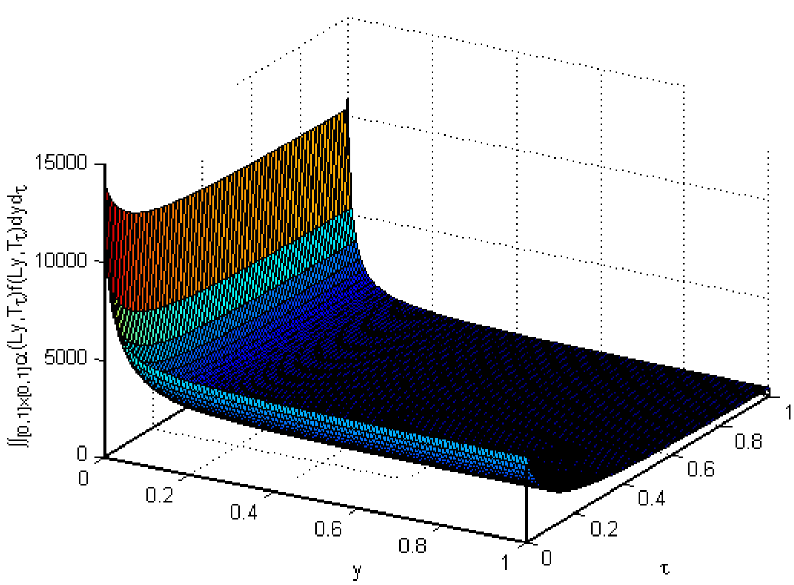

Fig. 3 Volume of the solid that lies below function

In Fig. 3, the observation time L used for comparing two connections has been shrunk by a factor m. However, the number of samples (measured frequencies) taken is the same. Instead of measuring the frequency of a character every minute, we measure every $1/m$ minutes. The emphasized region determined by the graphs of *f* and *g* in Fig. 3 has the same volume as the region in Fig. 2. We can see how after the change of variable the height of the figure increase as its base decreases, while the volume remains constant (Figs. 2 and 3).

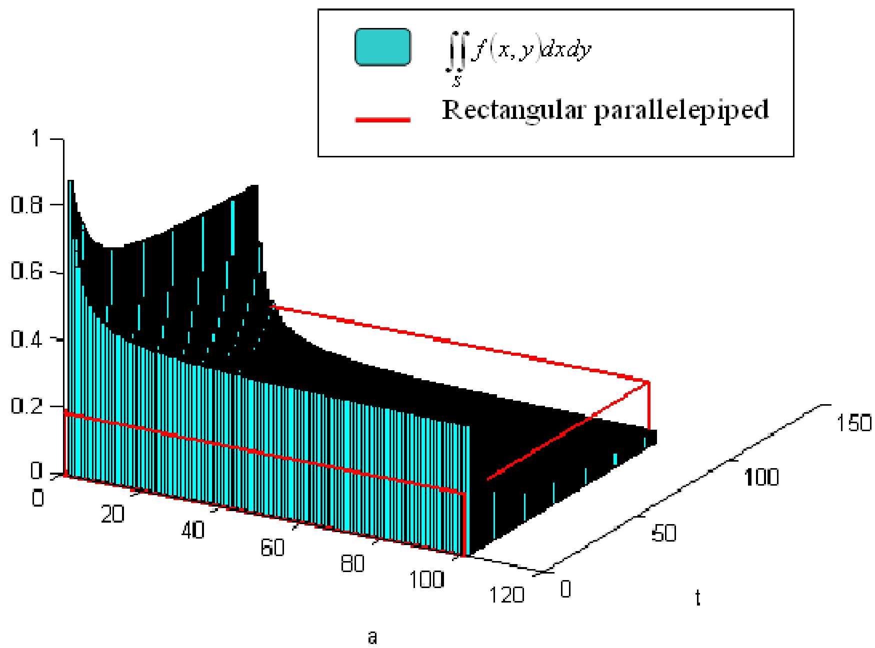

Fig. 4 Approximating the volume defined by the double integral. The volume of $\iint_S f(x,y)dxdy$ is equal to the volume of the rectangular parallelepiped drawn in red.

An interesting consequence of the mean theorem ([10]) is shown in Fig. 4. There exists a point $(x_0, y_0) \in S$ for which the volume of the integral $\iint_S f(x,y)dxdy$ is equal to $f(x_0, y_0)\text{Area}(S)$.

$f(x_0, y_0)$ gives the height of the rectangular parallelepiped whose volume approximates the value of the double integral $\iint_S f(x,y)dxdy$. This is particularly useful for finding the volumes of integrals that are hard to compute.

6. SIMULATIONS

In order to compare the performance of our approach to the original algorithms in [1], we adopt the same simulation method and settings in our experiment. Specifically, we deploy 25 nodes in an area of 750 *m* × 750 *m*. Each node has a communication range of 250 *m*. All of the nodes move according to a slot-based random walk model. Within each slot (of length T seconds), every node will first randomly pick a point in the range of (0, 10] *m* based on the previous position and then move toward it directly. Node 0 is always communicating with θ, and the other nodes randomly select a partner within each slot. We use the Dijkstra algorithm for path routing. Different paths may share one link and remain unchanged until the next slot. Every node generates traffic following a Poisson process at rate R. When the traffic passes by an intermediate node, it will be added with *i.i.d.* uniform delays drawn from $[0, \Delta]$ (the traffic order may be permuted due to queuing mechanism). After generating all the traffic, each link is padded with independent Poisson chaff noise to rate λ. We simulate the network for 1000 slots. In the simulation, $\lambda = 1$ packet/second, $\Delta = 1/18$ second, and $R = 1/9$ packet/second.

We have modified the TD and RT algorithms in [1] for our approach. The only difference between ours and the original is the selection of the support link. As we discussed in Section 4, assuming that the lack of information (p_k) on each link is already known, the support link k is selected only if $r_k / p_k \geq \tau$. The modified algorithms are referred to as TD2 and RT2, accordingly. We simulate TD, TD2, and SIM [1] under various thresholds and observation times. In order to find the best threshold τ, we first evaluate TD2 versus different τ values. As shown in Fig. 5, for $M = 1$, we notice that the error probability of TD2 decreases with τ until it stabilizes around $\tau \in [0.19, 0.23]$ and then starts to increase sharply. By this observation, we set $\tau = 0.2$ in the subsequent tests to minimize error probability.

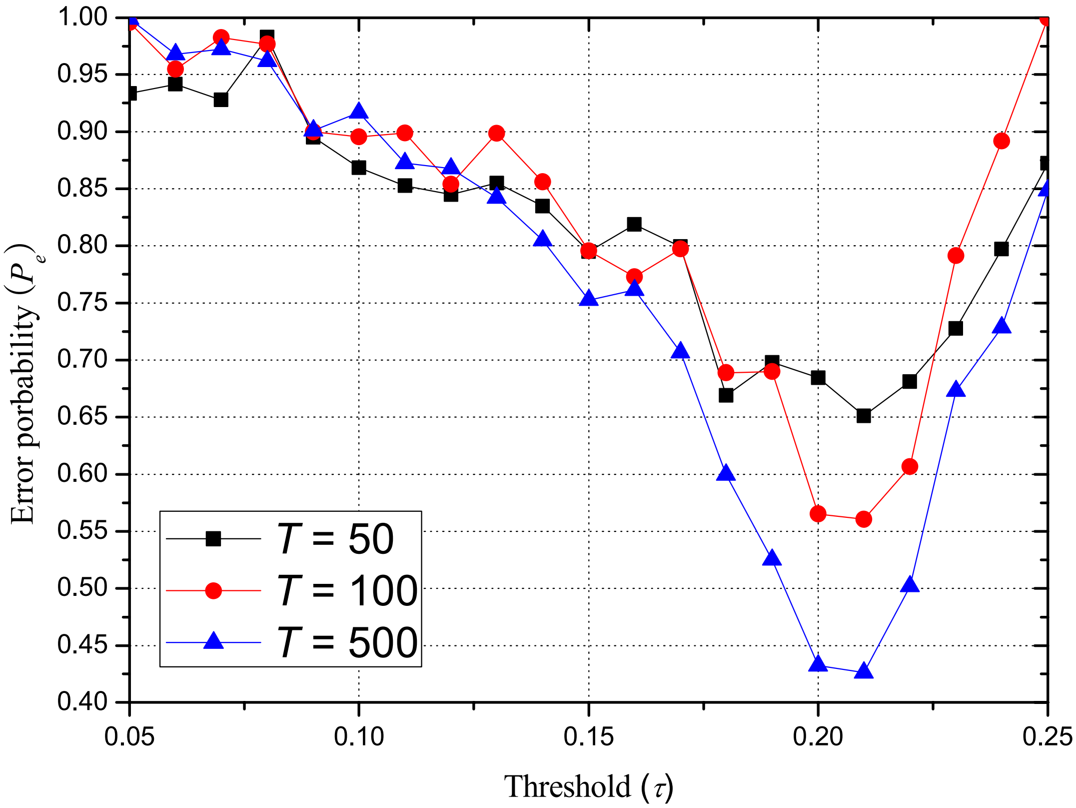

Fig. 5. Error probability P_e of TD2 vs. threshold τ.

We then simulate TD2 and SIM [1] for various total observation times MT to evaluate their performance affected by the observation time. As we can see from Fig. 6, just like the result of TD in [1], the error probability of TD2 also decays subexponentially. When M and T increase, the error probability of TD2 decreases. Since the SIM is not affected by the observation time, its error probability stays constant over range (0.55, 0.75).

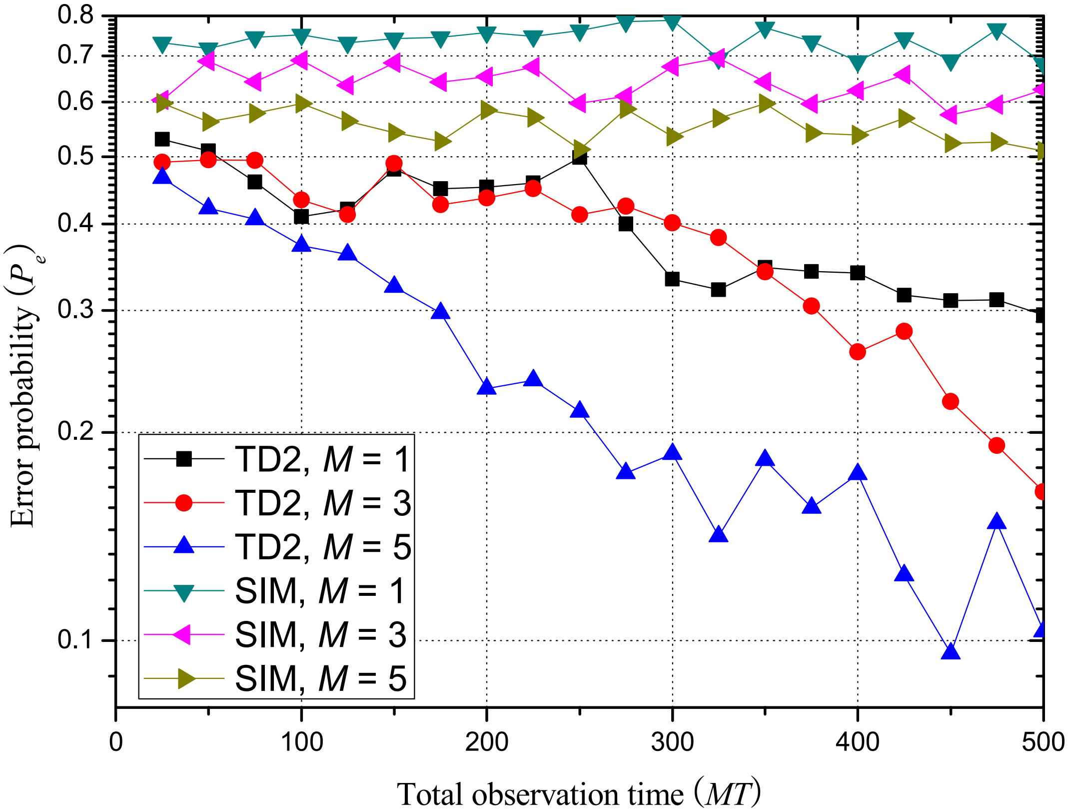

Fig. 6. Error probability P_e of TD2 and SIM vs. total observation time MT .

In fact, as long as we have set up a good value for the threshold, the result of TD2 should have the same characteristics of that of TD. Hence, the only major difference between TD and TD2 is the selection of threshold. Finally, we compare the performance of TD2 to the original TD in [1] under different threshold values. In Table 1, we can see that a good threshold for TD ($\tau \in [0.12, 0.14]$) will lead to a high error probability (nearly 83%) for TD2, while a good threshold for TD2 ($\tau \in [0.19, 0.21]$) will lead a high error probability (nearly 60%) for TD. We therefore can not tell which one is better based on different thresholds which are predefined values. However, TD2 can achieve the lowest error probability when $\tau = 0.21$ and $T = 500$, which is about 4 percent lower than TD's best result.

Table 1. Comparison on performance between TD and TD2

τ	$T = 50$		$T = 100$		$T = 500$	
	TD	TD2	TD	TD2	TD	TD2
0.12	0.5617	0.8451	0.5576	0.8539	0.4832	0.8679
0.13	0.5067	0.8551	0.4041	0.8987	0.4623	0.8422
0.14	0.6155	0.8348	0.4582	0.8559	0.4751	0.8048
0.19	0.6344	0.6979	0.5891	0.6899	0.5543	0.5252
0.20	0.6014	0.6844	0.5892	0.5652	0.5674	0.4325
0.21	0.6922	0.6511	0.5985	0.5604	0.5810	0.4263

Note that the simulation procedure is an abstraction, varying from a simplified simulation to very detail emulation. The most detail one is an implementation. The chosen level of abstraction in simulation should be just enough to capture the essentials of the studied method. If the simulation is too detail (such as like a implementation), it not only costs a lot time, but also obscure the effect of the studied methods due to the complexity of the system, which is treated as noise. If the simulation is too simple, it cannot study the method well. Therefore, a good approach is to choose an appropriate level of abstraction, neither too simple nor too detail, to study the method. We believe that our chosen simulation already has been already good enough to evaluate the studied methods. In order to better illustrate the methods, the above deployed scenarios involve all the valuable information: random movement of each node, regular communication process, and background noise. In such context, we may reasonably draw the conclusion that, our algorithm is an alternative way to trace targets in anonymous networks, and that it is better than the algorithm in [1].

7. Conclusion

We provided a way to shorten the time interval used for thumbprinting and tune it suitably depending on network conditions. We have found interesting properties of the thumbprinting function using the mean value and provided a general method to compute the value of the function. The method works for any function that satisfies certain properties. We have also studied another mechanism for tracing intruders in the Internet by passively monitoring flows between source and destination pairs. We computed the error probability of our algorithm and showed that its value decreases exponentially as the observation time increases. Our simulation shows that our algorithm is an alternative way to trace targets in anonymous networks, and that it is better than the algorithm in [1].

Acknowledgement

This work is supported in part by The U.S. National Science Foundation (NSF), under grants: CNS-0716211, CCF-0829827, CNS-0737325, and CNS-1059265.

References

[1] T. He, H.Y. Wong and K-W. Lee, “Traffic Analysis in Anonymous MANETs,” in Proc. IEEE MILCOM, San Diego, November, 2008.

[2] L. T. Heberlein, K. Levitt, and B. Mukherjee, “Internet Security Monitor: An Intrusion-Detection System for Large Scale Networks,” in Proc. 15th National Computer Security Conference, pp. 262-271, Oct. 1992.

[3] H.T. Jung et al., “Caller Identification System in the Internet Environment,” in Proc. of 4th Usenix Security Symposium, 1993.

[4] S. Staniford-Chen and L. T. Heberlein, “Holding Intruders Accountable on the Internet,” in Proc. the 1995 IEEE Symposium on Security and Privacy, (Oakland, CA), pp. 39-49, May 1995.

[5] S. Wadell, Private Communication. 1994.

[6] Implicit Function Theorem. (2007, October). [Online]. Available: http://en.wikipedia.org/wiki/Implicit_function_theorem.

[7] Internet. (2008, March 26). [Online]. Available: http://en.wikipedia.org/wiki/Internet.

[8] Jensen's inequality. (November, 2008). [Online]. Available: http://en.wikipedia.org/wiki/Jensen's inequalit.

[9] Least Squares. (October, 2008). [Online]. Available: http://en.wikipedia.org/wiki/Least_squares.

[10] Mean Value Theorem. (December, 2008). [Online]. Available: http://en.wikipedia.org/wiki/Mean_value_theorem.

[11] Poisson process. (2008, March 17). [Online]. Available: http://en.wikipedia.org/wiki/Poisson_process

[12] A. Olteanu, Y. Xiao, J. Liu, and T. M. Chen, "Studying Non-intrusive Tracing in the Internet," Proceedings of The 7th International ICST Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness (Qshine 2010).

[13] I. Hamadeh and G. Kesidis, "A taxonomy of internet traceback," International Journal of Security and Networks, Vol. 1, No.1/2, pp. 54 - 61, 2006.

[14] X. Wang, "The loop fallacy and deterministic serialisation in tracing intrusion connections through stepping stones," International Journal of Security and Networks, Vol. 1, Nos.3/4 pp. 184 - 197, 2006.

[15] J. Pan, L. Cai, and X. Shen, " Vulnerabilities in distance-indexed IP traceback schemes," International Journal of Security and Networks, Vol. 2, Nos. 1/2, pp. 81 - 94, 2007.

[16] T. Korkmaz, C. Gong, K. Sarac, and S. G. Dykes, "Single packet IP traceback in AS-level partial deployment scenario," International Journal of Security and Networks, Vol. 2, Nos. 1/2, pp. 95 - 10, 2007.

[17] A. L. Burt, M. Darschewski, I. Ray, R. Thurimella, and H. Wu, " Origins: an approach to trace fast spreading worms to their roots," International Journal of Security and Networks, Vol. 3, No. 1, pp. 36 - 46, 2008.

[18] K. R. Schrader, B. E. Mullins, G. L. Peterson, and R. F. Mills, "An FPGA-based system for tracking digital information transmitted via Peer-to-Peer protocols," International Journal of Security and Networks, Vol. 5, No.4 pp. 236 - 247, 2010.

[19] D. S. Wong and X. Tian, "E-mail Protocols with Perfect Forward Secrecy," International Journal of Security and Networks, Vol. 7, No. 1, 2012

[20] L. Vespa, R. Chakrovorty, and N. Weng, "Lightweight Testbed for Evaluating Worm Containment Systems," International Journal of Security and Networks, Vol. 7, No. 1, 2012.

[21] F. Kandah, Y. Singh, W. Zhang, and T. Wang, "A Misleading Active Routing Attack in Mobile Ad-hoc Networks," International Journal of Security and Networks, Vol. 7, No. 1, 2012.

[22] K. Alsubhi, Y. Alhazmi, N. Bouabdallah, and R. Boutaba, "Security Configuration Management in Intrusion Detection and Prevention Systems," International Journal of Security and Networks, Vol. 7, No. 1, 2012.

[23] Z. Xiao and Y. Xiao, "PeerReview Re-evaluation for Accountability in Distributed Systems or Networks," International Journal of Security and Networks, Vol. 7, No. 1, 2012.

[24] Z. S. Al-Salloum, "Defensive Computer Worms - An Overview," International Journal of Security and Networks, Vol. 7, No. 1, 2012.

[25] J. Cao, X. Jia, and L. Shu, "Editorial," International Journal of Sensor Networks, Vol. 11, No. 1, 2012. pp. 1-2.

[26] R. Silva, J. Silva, J. M. L.P. Caldeira, and J. J.P.C. Rodrigues, "Mobile multimedia in wireless sensor networks," International Journal of Sensor Networks, Vol. 11, No. 1, 2012. pp. 3-9.

[27] S. Medjiah, T. Ahmed, and A. Hamid Asgari, "Streaming multimedia over WMSNs: an online multipath routing protocol," International Journal of Sensor Networks, Vol. 11, No. 1, 2012. pp. 10 - 21.

[28] W. C. Chia, L. W. Chew, L. Ang, and K. P. Seng, "Low memory image stitching and compression for WMSN using strip-based processing," International Journal of Sensor Networks, Vol. 11, No. 1, 2012. pp. 22 - 32.

[29] L. W. Chew, W. C. Chia, L. Ang, and K. P. Seng, "Low-memory video compression architecture using strip-based processing for implementation in wireless multimedia sensor networks," International Journal of Sensor Networks, Vol. 11, No. 1, 2012. pp. 33 - 47.

[30] G.U. Gamm, M. Kostic, M. Sippel, and L.M. Reindl, "Low-power sensor node with addressable wake-up on-demand capability," International Journal of Sensor Networks, Vol. 11, No. 1, 2012. pp. 48 - 56.

[31] D. Li, L. Liu, and H. Du, "An approximation algorithm for dominating nodes selection in multi-channel multi-radio wireless sensor networks," International Journal of Sensor Networks, Vol. 11, No. 1, 2012. pp. 57 - 65.

[32] D. Takahashi, Y. Xiao, and K. Meng, "Virtual Flow-net for Accountability and Forensics of Computer and Network Systems," (Wiley Journal of) Security and Communication Networks, DOI: 10.1002/sec.407, accepted.

[33] Y. Xiao, S. Yue, B. Fu, and S. Ozdemir, "GlobalView: Building Global View with Log Files in a Distributed / Networked System for Accountability," (Wiley Journal of) Security and Communication Networks, DOI: 10.1002/sec.374, accepted.

[34] Y. Xiao, K. Meng, and D. Takahashi, “Accountability using Flow-net: Design, Implementation, and Performance Evaluation," (Wiley Journal of) Security and Communication Networks, Special Issue on Security and Privacy in Emerging Information Technologies, Vol.5, NO. 1, pp. 29–49, Jan. 2012.

[35] Y. Zhang, Y. Xiao, K. Ghaboosi, J. Zhang, and H. Deng, "A Survey of Cyber Crimes," (Wiley Journal of) Security and Communication Networks, Vol. 5, No. 4, pp. 422–437, Apr. 2012.

[36] Y. Zhang, Y. Xiao, M. Chen, J. Zhang, and H. Deng, "A Survey of Security Visualization for Computer Network Logs," (Wiley Journal of) Security and Communication Networks, Vol. 5, No. 4, pp. 404–421, Apr. 2012.

[37] Y. Xiao, D. Takahashi, J. Liu, and H. Deng, and J. Zhang, "Wireless Telemedicine and M-Health: Technologies, Applications, and Research issues," International Journal of Sensor Networks (IJSNet), Vol. 10, No. 4, 2011, pp. 202- 236.

[38] L. Liu, Y. Xiao, J. Zhang, A. Faulkner, and K. Weber, "Hidden Information in Microsoft Word," International Journal of Security and Networks (IJSN), Vol. 6, Nos. 2/3, 2011, pp. 123- 135.

[39] J. Liu, and Y. Xiao, “Temporal Accountability and Anonymity in Medical Sensor Networks,” ACM/Springer Mobile Networks and Applications (MONET), Special Issue: Wireless and Personal Communications, Vol. 16, No. 6, pp. 695-712, Dec. 2011.

[40] D. Takahashi, Y. Xiao, Y. Zhang, P. Chatzimisios, and H.-H. Chen, "IEEE 802.11 User Fingerprinting and Its Applications," Computers and Mathematics with Applications, Vol. 60, No. 2, July 2010, pp. 307-318.

[41] J. Liu, Y. Xiao, K. Ghaboosi, H. Deng, and, J. Zhang, "Botnet: Classification, Attacks, Detection, Tracing, and Preventive Measures," EURASIP Journal on Wireless Communications and Networking, Volume 2009, Article ID 692654, 11 pages, doi:10.1155/2009/692654

[42] Y. Xiao, "Flow-Net Methodology for Accountability in Wireless Networks," IEEE Network, Vol. 23, No. 5, Sept./Oct. 2009, pp. 30-37.

[43] K. Meng, Y. Xiao, and S. V. Vrbsky, "Building a Wireless Capturing Tool for WiFi," (Wiley Journal of) Security and Communication Networks, Vol. 2, No. 6, Nov./Dec. 2009, pp. 654 – 668.

[44] M. Lei, Y. Xiao, S. V. Vrbsky, and C.-C. Li, “Virtual Password Using Random Linear Functions for On-line Services, ATMs, and Pervasive Computing,” Computer Communications Journal, Elsevier, Special Issue on Secure Multi-Mode Systems and their Applications for Pervasive Computing, Vol. 31, No. 18, Dec. 2008, pp. 4367–4375.

[45] D. Takahashi and Y Xiao, “Retrieving Knowledge from Auditing Log Files for Computer and Network Forensics and Accountability,” (Wiley Journal) Security and Communication Networks, Vol. 1, No. 2, pp. 147-160, March/April 2008.

[46] Y. Xiao, "Accountability for Wireless LANs, Ad Hoc Networks, and Wireless Mesh Networks," IEEE Communication Magazine, special issue on Security in Mobile Ad Hoc and Sensor Networks, Vol. 46, No. 4, Apr. 2008, pp. 116-126.