Tracking Anonymous Peer-to-Peer VoIP Calls on the Internet *

Xinyuan Wang
Department of Information and Software Engineering
George Mason University
Fairfax, VA 22030, USA
xwangc@gmu.edu

Shiping Chen
Center for Secure Information Systems
George Mason University
Fairfax, VA 22030, USA
schen3@gmu.edu

Sushil Jajodia
Center for Secure Information Systems
George Mason University
Fairfax, VA 22030, USA
jajodia@gmu.edu

ABSTRACT

Peer-to-peer VoIP calls are becoming increasingly popular due to their advantages in cost and convenience. When these calls are encrypted from end to end and anonymized by low latency anonymizing network, they are considered by many people to be both secure and anonymous.

In this paper, we present a watermark technique that could be used for effectively identifying and correlating encrypted, peer-to-peer VoIP calls even if they are anonymized by low latency anonymizing networks. This result is in contrast to many people's perception. The key idea is to embed a unique watermark into the encrypted VoIP flow by slightly adjusting the timing of selected packets. Our analysis shows that it only takes several milliseconds time adjustment to make normal VoIP flows highly unique and the embedded watermark could be preserved across the low latency anonymizing network if appropriate redundancy is applied. Our analytical results are backed up by the real-time experiments performed on leading peer-to-peer VoIP client and on a commercially deployed anonymizing network. Our results demonstrate that (1) tracking anonymous peer-to-peer VoIP calls on the Internet is feasible and (2) low latency anonymizing networks are susceptible to timing attacks.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and protection (e.g., firewalls)*; C.2.3 [**Computer-Communication Networks**]: Network Operations—*Network monitoring*

General Terms

Security

*This work was partially supported by the Air Force Research Laboratory, Rome under the grant F30602-00-2-0512 and by the Army Research Office under the grants DAAD19-03-1-0257 and W911NF-05-1-0374.

CCS'05, November 7–11, 2005, Alexandra, Virginia, USA.

Keywords

VoIP, Anonymous VoIP Calls, VoIP Tracing, Peer-to-Peer Anonymous Communication

1. INTRODUCTION

VoIP is a technology that allows people to make phone calls through the public Internet rather than traditional Public Switched Telephone Network (PSTN). Because VoIP offers significant cost savings with more flexible and advanced features over Plain Old Telephone System (POTS), more and more voice calls are now carried at least partially via VoIP. In fact, consulting firm Frost & Sullivan has predicted that VoIP will account for approximately 75% of world voice services by 2007.

For privacy reasons, people sometimes want their phone conversation to be anonymous and do not want other people know that they have even talked over the phone. The use of VoIP has made it much easier to achieve anonymity in voice communications, especially when VoIP calls are made between computers. This is because VoIP calls between peer computers have no phone numbers associated with them, and they could easily be protected by end to end encryption and routed through low latency anonymizing networks (e.g., Onion Routing [13], Tor [6], Freedom [3], and Tarzan [12]) to achieve anonymity. People intuitively think their computer to computer VoIP calls could remain anonymous if they are encrypted end to end and routed through some low latency anonymizing network.

On the other hand, law enforcement agencies (LEA) often need to conduct lawful electronic surveillance in order to combat crime and terrorism. For example, the LEAs need techniques to determine who has called the surveillance target and to whom the surveillance target has called. In a letter to FCC [8], several federal law enforcement agencies have considered the capability of tracking VoIP calls "of paramount importance to the law enforcement and the national security interests of the United States."

How to balance people's needs for privacy and anonymity and the security requirements of the law enforcement agencies has been a subject of controversy. In this paper, we leave the controversy between anonymity and security aside and instead focus on the technical feasibility of tracking anonymous peer-to-peer VoIP calls on the Internet. Our goal is to investigate practical techniques for the effective tracking of anonymous VoIP calls on the Internet and identify the weakness of some of the currently deployed anonymous communication systems.

We choose to investigate the popular Skype [28] peer-to-peer VoIP calls in the context of the anonymous VPN provided by findnot.com [11]. Skype offers free computer to computer VoIP calls based on KaZaa [16] peer-to-peer technology. Several properties of Skype have made it an attractive candidate for the investigation of tracking anonymous VoIP calls on the Internet:

- It is free and widely used. Since August 2003, there are over 100 million downloads of the Skype client. It is being actively used by millions of people all over the world. Skype is now included in Kazaa v3.0.
- All the Skype traffic is encrypted from end to end by 256-bit AES encryption.
- Skype can automatically traverse most firewalls and NAT (Network Address Translation) gateways with the help of intermediate peers.
- Skype intelligently and dynamically routes the encrypted calls through different peers to achieve low latency. This means that the route and the intermediate peer(s) of one VoIP call could be changed during a call.
- It uses proprietary peer-to-peer signaling protocol to set up the VoIP calls.

Since most Skype calls are carried in UDP, we can not directly use those anonymizing systems (such as Onion Routing [13], Tor [6] or anonymizer.com [1]), who do not support anonymization of all UDP flows, to anonymize Skype VoIP calls. We choose to use the anonymous communication services by findnot.com [11] that support anonymization of all IP protocols through point to point tunnel protocol (PPTP).

The key challenge in tracking encrypted VoIP calls across anonymous communication system is how to identify the correlation between the VoIP flows of the caller and the callee. Since all the traffic of the peer-to-peer VoIP calls are encrypted, no signaling information is available for correlation. To be able to track encrypted, anonymous VoIP calls across the Internet, we use the timing characteristics of the anonymized VoIP flow. Unfortunately, the original inter-packet arrival characteristics of VoIP flows are not distinct enough as the inter-packet timing arrival time of VoIP traffic is determined by the frame packetization interval used. This means that passive comparison of the original inter-packet timing characteristics of VoIP flows will not be able to distinguish different VoIP calls.

In order to uniquely identify the anonymous VoIP calls through inter-packet timing characteristics, we use an active approach to deliberately make the inter-packet timing of VoIP calls more distinctive. The idea is to embed a unique watermark into the inter-packet timing of the VoIP flows by slightly adjusting the timing of selected packets. If the embedded watermark is unique enough and robust enough, the watermarked VoIP flows could be effectively identified. By utilizing redundancy techniques, we can make the embedded watermark robust against random timing perturbation provided there are enough packets in the VoIP flow.

Our analytical and experimental results demonstrate that (1) tracking anonymous peer-to-peer VoIP calls on the Internet is feasible and (2) low latency anonimizing systems are susceptible to timing attack. Our VoIP tracking technique does not require the global monitoring capability, and it could be used to determine if party A is communicating (or has communicated) with party B via peer-to-peer VoIP even if the VoIP traffic is (or has been) disguised by low latency anonymous communication systems.

The rest of the paper is organized as follows. Section 2 formulates the problem of tracking anonymous peer-to-peer VoIP calls, and describes the overall tracing model. Section 3 presents the active timing based tracking method and analyzes its effectiveness. Section 4 describes our implementation of the high precision VoIP watermarking engine in real-time Linux kernel. Section 5 evaluates the effectiveness of our method empirically. Section 6 summarizes related works. Section 7 concludes the paper.

2. THE OVERALL MODEL OF TRACING ANONYMOUS PEER-TO-PEER VOIP CALLS

Given any two different Skype peers A and B, we are interested in determining if A is talking (or has talked) to B via Skype peer-to-peer VoIP. As shown in Figure 1, both Skype peers A and B have outgoing and incoming VoIP flows to and from the Internet cloud. The Skype peers could be behind firewall and NAT, and peer A and/or B could be connected to some low latency anonymizing network. Here we view the Internet cloud and any low latency anonymizing network as a black box, and we are interested only in the Skypy flows that enter or exit the black box. We assume that (1) we can monitor the Skype flow from the black box to the Skype peer; (2) we can perturb the timing of the Skype flow from the Skype peer to the black box.

Here we do not intend to track all the peer-to-peer VoIP calls from anyone to anyone, nor do we assume the global monitoring and intercepting capability. Instead we focus on finding out if some parties in which we are interested have communicated via peer-to-peer VoIP calls anonymously, and we only need the capability to monitor and intercept IP flows to and from those interested parties. This model is consistent with our understanding of the common practice of lawful electronic surveillance by the law enforcement agencies.

Because the Skype VoIP flows are encrypted from end to end, no correlation could be found from the flow content. Given that the Skype VoIP flow could pass some intermediate Skype peers and some low latency anonymizing network, there is no correlation from the VoIP flow headers. Among all the characteristics of the VoIP flows, the inter-packet timing characteristics are likely to be preserved across intermediate Skype peers and low latency anonymizing network. This invariant property of VoIP flows forms the very foundation for tracking anonymous, peer-to-peer VoIP calls on the Internet.

A number of timing based correlation methods have been proposed, and they can be classified into two categories: passive and active. Passive timing based correlation approaches (e.g. [35], [34], [33] [7], [5]) correlate the encrypted flows based on passive comparison of their timing characteristics, and they have been shown to be effective when the timing characteristics of each flow are unique enough. However, the inter-packet timing characteristics of all VoIP flows are very similar to each other. The inter-packet arrival time of VoIP flows is determined by the voice codec and the corresponding packetization interval, and there are only a few commonly used VoIP packetization intervals (i.e.

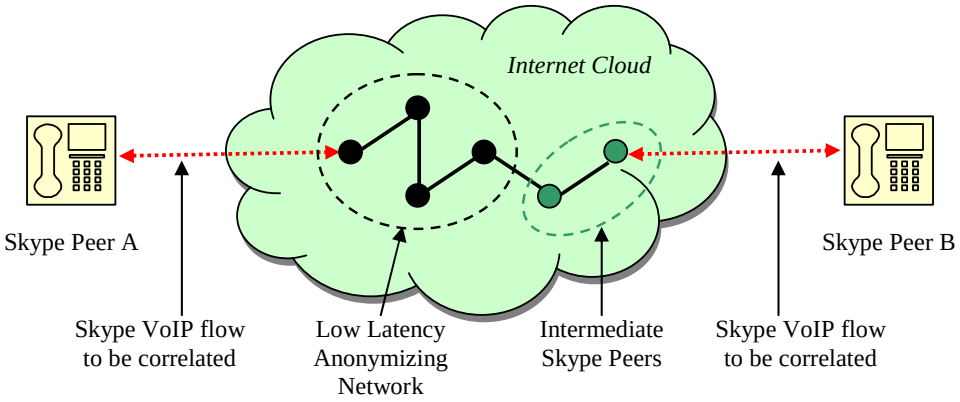


Figure 1: Anonymous Peer-to-Peer VoIP Calls Tracing Model

20ms or 30ms). Therefore, passively comparing the timing characteristics of VoIP flows will not be able to distinguish different VoIP flows.

Wang and Reeves [32] proposed the first active approach to correlate the encrypted flows. They suggested embedding a unique watermark into the inter-packet timing domain of the interactive flow through deliberate timing adjustment of selected packets, and correlating based on the embedded watermark. Because the embedded watermark could make an otherwise non-distinctive flow unique, their active method has the potential to differentiate flows with very similar timing characteristics. However, the method proposed in [32] can not be directly used to correlate VoIP flows due to the following reasons:

- The VoIP traffic has stringent real-time constraints, and the total end to end delay should be less than 150ms.
- The inter-packet arrival time of VoIP flows is very short (i.e. 20ms or 30ms). This requires any time adjustment on VoIP packet to be very precise and small.
- The watermarking method proposed by Wang and Reeves is based on the quantization of averaged Inter-Packet Delays (IPDs), and it requires packet buffering in order to achieve the even timing adjustment over different packets. The required buffering would be too long for the real-time VoIP flows.

To correlate anonymous VoIP flows with similar inter-packet timing characteristics, we use an active approach to deliberately yet subtly make the inter-packet timing characteristics of the VoIP flows more unique. This is achieved by embedding a unique watermark into the inter-packet timing domain of the VoIP flow in real-time.

To address the limitations of previous work [32], we use a new watermarking scheme that is suited for tracking anonymous VoIP traffic in real-time. The key challenge in tracking anonymous VoIP calls by the active approach is how to precisely adjust the packet timing without buffering and guarantee the even time adjustment of those selected packets.

3. ACTIVE TIMING BASED TRACKING OF VOIP FLOWS

We present the new watermarking scheme which guarantees the even time adjustment for embedding the watermark in real time and has all the theoretical strengths of work [32]. Unlike the watermarking scheme proposed in previous work [32], our new watermarking scheme is probabilistic in the sense that the watermark embedding success rate is not guaranteed 100%. In other words, the new watermarking scheme trades off the guaranteed 100% watermark embedding success rate with the guaranteed even time adjustment for embedding the watermark. By exploiting the inherent inter-packet timing characteristics of the VoIP flows, our new watermarking scheme achieves virtually 100% watermark embedding success rate with guaranteed even time adjustment for embedding the watermark.

3.1 Basic Concept and Notion

Given any packet flow $P_1, \ldots, P_n$ with time stamps t_1, $\ldots$, t_n respectively ($t_i < t_j$ for $1 \le i < j \le n$), we can independently and probabilistically choose a number of packets through the following process: (1) sequentially look at each of the first $n-d$ ($0 < d \ll n$) packets; and (2) independently determine if the current packet will be probabilistically chosen, with probability $p = \frac{2r}{n-d}$ ($0 < r < \frac{n-d}{2}$).

Here whether or not choosing the current packet is not affected by any previously chosen packets and it will not affect whether to choose any other packets. In other word, all the selected packets are selected independently from each other. Therefore, we can expect to have $2r$ distinct packets independently and randomly selected from any packet flow of n packets. We denote the $2r$ randomly selected packets as $P_{z_1}, \ldots, P_{z_{2r}}$ ($1 \le z_k \le n-d$ for $1 \le k \le 2r$), and create $2r$ packet pairs: $\langle P_{z_k}, P_{z_k+d} \rangle$ ($d \ge 1,\ k = 1, \ldots, 2r$).

The IPD (Inter-Packet Delay) between P_{z_k+d} and P_{z_k} is defined as

$$ipd_{z_k,d} \ = \ t_{z_k+d} - t_{z_k}, \ (k = 1, \ldots, 2r) \qquad (1)$$

Because all P_{z_k} ($k = 1, \ldots, 2r$) are selected independently, $ipd_{z_k,d}$ is independent from each other. Since each P_{z_k} is randomly and independently selected through the same process, $ipd_{z_k,d}$ is identically distributed no matter what inter-packet timing distribution the packet flow $P_1, \ldots, P_n$ may have. Therefore, $ipd_{z_k,d}$ ($k = 1, \ldots, 2r$) is independent and

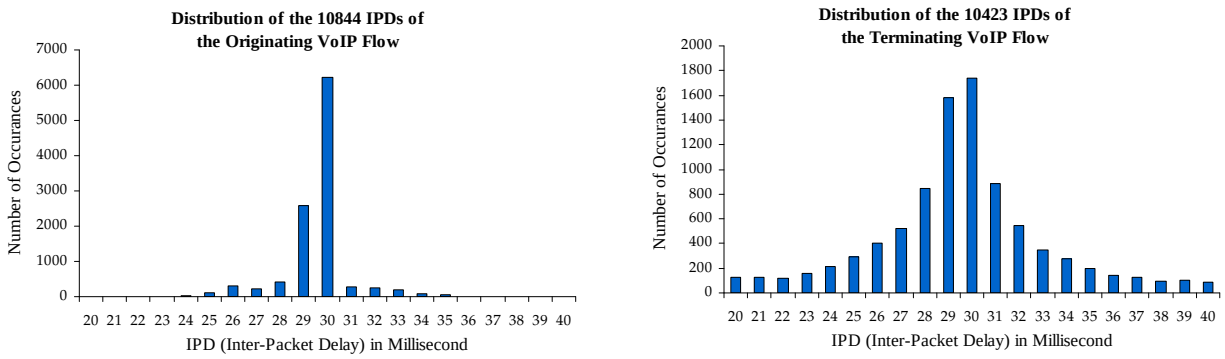


Figure 2: Distribution of IPDs of the originating and terminating Skype flows

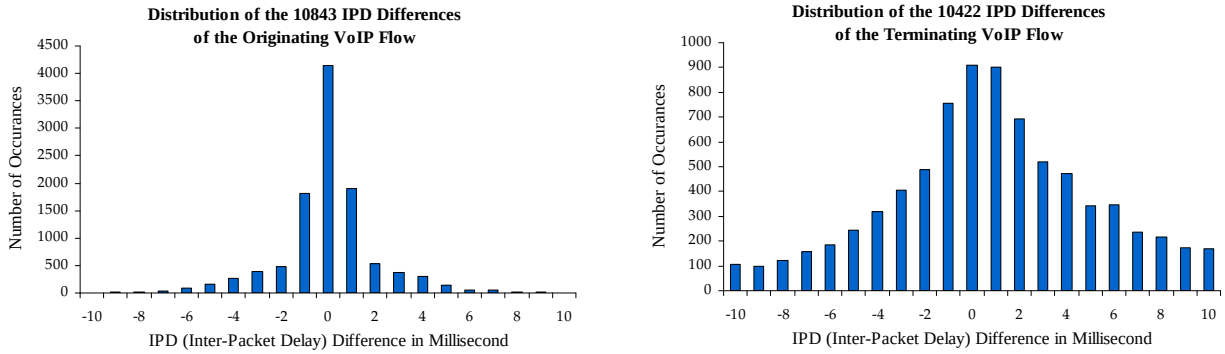


Figure 3: Distribution of IPD Differences of the originating and terminating Skype flows

identically distributed (*iid*).

We then randomly divide the $2r$ IPDs into 2 distinct groups of equal size. Let $ipd_{1,k,d}$ and $ipd_{2,k,d}$ $(k = 1, \ldots, r)$ denote the IPDs in group 1 and group 2 respectively. Apparently both $ipd_{1,k,d}$ and $ipd_{2,k,d}$ $(k = 1, \ldots, r)$ are *iid*. Therefore $\mathrm{E}(ipd_{1,k,d}) = \mathrm{E}(ipd_{2,k,d})$, and $\mathrm{Var}(ipd_{1,k,d}) = \mathrm{Var}(ipd_{2,k,d})$.

Let

$$Y_{k,d} = \frac{ipd_{1,k,d} - ipd_{2,k,d}}{2} \quad (k = 1, \ldots, r) \tag{2}$$

Then we have $\mathrm{E}(Y_{k,d}) = (\mathrm{E}(ipd_{1,k,d}) - \mathrm{E}(ipd_{2,k,d}))/2 = 0$. Because $ipd_{1,k,d}$ and $ipd_{2,k,d}$ are *iid*, $Y_{k,d}$ is also *iid*. We use $\sigma^2_{Y,d}$ to represent the variance.

We represent the average of r $Y_{k,d}$'s as

$$\overline{Y_{r,d}} = \frac{1}{r} \sum_{k=1}^{r} Y_{k,d} \tag{3}$$

Here $\overline{Y_{r,d}}$ represents the average of a group of normalized IPD differences, and we call r the redundancy number. According to the property of variance of independent random variables, we have $\mathrm{Var}(\overline{Y_{r,d}}) = \sigma^2_{Y,d}/r$. Because $\mathrm{E}(Y_{k,d}) = 0$ $(k = 1, \ldots, r)$, $\mathrm{E}(\overline{Y_{r,d}}) = 0$. Because $Y_{k,d}$ is symmetric $(k = 1, \ldots, r)$, $\overline{Y_{r,d}}$ is also symmetric. Therefore, the distribution of $\overline{Y_{r,d}}$ is symmetrically centered around 0.

To illustrate the validity of concepts of $Y_{k,d}$ and $\overline{Y_{r,d}}$, we collected two traces of the packet flows of a real Skype call from two communicating Skype peers that are 27 hops and over a thousand miles away (when the Skype call is routed through the commercial findnot.com anonymizing network). One trace is for the Skype flow that originated from one Skype peer, and the other trace is for the Skype flow that terminated at the other Skype peer. We call them the *originating flow* and the *terminating flow* respectively. The left and right chart of Figure 2 show the IPD histograms of the originating flow and the terminating flow respectively. They all have 30ms average IPD, which indicates that the packetization interval of Skype VoIP call is 30ms. While the IPDs of the originating Skype flow is more concentrated around 30ms, the IPDs of the terminating Skype flow is less clustered due to the network delay jitter. The left and right chart of Figure 3 show the histograms of $Y_{k,d}$ with d=1 (or equivalently $\overline{Y_{r,d}}$ with r=1 and d=1) of the Skype originating flow and the terminating flow respectively. They both confirm that the distribution of $\overline{Y_{r,d}}$ of Skype VoIP flows is indeed symmetric and centered around 0.

3.2 Embedding and Decoding A Binary Bit Probabilistically

Since the distribution of $\overline{Y_{r,d}}$ is symmetric and centered around 0, the probabilities of $\overline{Y_{r,d}}$ to be positive and negative are equal. If we decrease or increase $\overline{Y_{r,d}}$ by an amount $a > 0$, we can shift its distribution to the left or right by a so that $\overline{Y_{r,d}}$ will be more likely to be negative or positive. This gives us a way to embed and decode a single binary bit probabilistically.

To embed a bit 0, we decrease $\overline{Y_{r,d}}$ by a, so that $\overline{Y_{r,d}}$ will have > 0.5 probability to be less than 0. To embed a bit 1, we increase $\overline{Y_{r,d}}$ by a, so that $\overline{Y_{r,d}}$ will have > 0.5 probability to be greater than 0. By definition in equation (3), the decrease or increase of $\overline{Y_{r,d}}$ can be easily achieved by decreasing or increasing each of the r $Y_{k,d}$'s by a. By

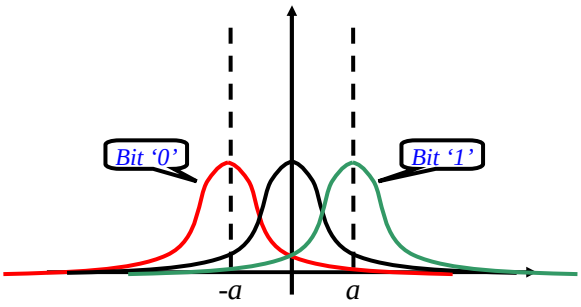


Figure 4: Embedding Binary Bit by Shifting the Distribution of $\overline{Y_{r,d}}$ by a to the Left or Right

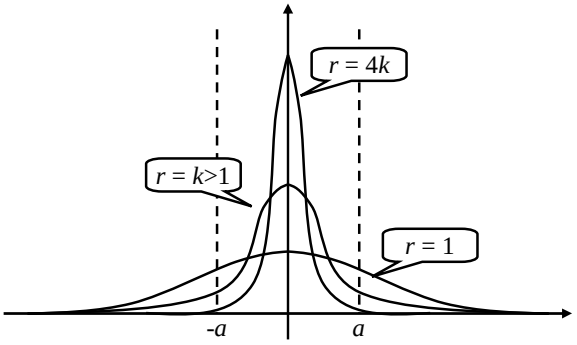


Figure 5: Probability Distribution of $\overline{Y_{r,d}}$ With Different r

definition in equation (2), the decrease of $Y_{k,d}$ by a can be achieved by decreasing each $ipd_{1,k,d}$ by a and increasing each $ipd_{2,k,d}$ by a; the increase of $Y_{k,d}$ by a can be achieved by increasing each $ipd_{1,k,d}$ by a and decreasing each $ipd_{2,k,d}$ by a.

After $\overline{Y_{r,d}}$ has been decreased or increased by a, we can decode the embedded binary bit by checking whether $\overline{Y_{r,d}}$ is less than or greater than 0. The decoding of the embedded binary bit is 1 if the value of $\overline{Y_{r,d}}$ is greater than 0, or 0 if the value of $\overline{Y_{r,d}}$ is less than or equal to 0. It is easy to see that probability of correct decoding is always greater than that of wrong decoding.

However, as shown in Figure 4, there is always a non-zero probability such that the embedded bit (with adjustment $a > 0$) will be decoded incorrectly (i.e. $\overline{Y_{r,d}} > a$ or $\overline{Y_{r,d}} < -a$). We define the probability that the embedded bit will be decoded correctly as the *bit embedding success rate w.r.t. adjustment* a, which can be quantitatively expressed as $\Pr(\overline{Y_{r,d}} < a)$.

Here the adjustment a is a representation of the watermark embedding strength. The larger the a is, the higher the bit embedding success rate will be. We now show that even with arbitrarily small $a > 0$ (or equivalently arbitrarily weak watermark embedding strength), we can achieve arbitrarily close to a 100% bit embedding success rate by having a sufficiently large redundancy number r.

Central Limit Theorem *If the random variables X_1, ..., X_n form a random sample of size n from a given distribution X with mean μ and finite variance σ^2,then for any fixed number x*

$$\lim_{n\to\infty} \Pr[\frac{\sqrt{n}(\overline{X_n} - \mu)}{\sigma} \leq x] = \Phi(x) \qquad (4)$$

where $\Phi(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} e^{-\frac{u^2}{2}} du$.

The theorem indicates that whenever a random sample of size n is taken from any distribution with mean μ and finite variance σ^2, the sample mean $\overline{X_n}$ will be approximately normally distributed with mean μ and variance σ^2/n, or equivalently the distribution of random variable $\sqrt{n}(\overline{X_n} - \mu)/\sigma$ will be approximately a standard normal distribution.

Applying the Central Limit Theorem to random sample $Y_{1,d}, \ldots, Y_{r,d}$, where $\mathrm{Var}(Y_{k,d}) = \sigma^2_{Y,d}$, $\mathrm{E}(Y_{k,d}) = 0$, we have

$$\Pr[\frac{\sqrt{r}(\overline{Y_{r,d}} - \mathrm{E}(Y_{k,d}))}{\sqrt{\mathrm{Var}(Y_{r,d})}} < x] = \Pr[\frac{\sqrt{r}\overline{Y_{r,d}}}{\sigma_{Y,d}} < x] \approx \Phi(x) \quad (5)$$

Therefore

$$\Pr[\overline{Y_{r,d}} < a] = \Pr[\frac{\sqrt{r}\overline{Y_{r,d}}}{\sigma_{Y,d}} < \frac{a\sqrt{r}}{\sigma_{Y,d}}] \approx \Phi(\frac{a\sqrt{r}}{\sigma_{Y,d}}) \qquad (6)$$

This means that the distribution of the probabilistic watermark bit embedding success rate is approximately normally distributed with zero mean and variance σ^2/r.

Equation (6) gives us an accurate estimate of the probabilistic watermark bit embedding success rate. It indicates that no matter what distribution $Y_{k,d}$ may be, no matter what variance $Y_{k,d}$ may have (as long as it exists), no matter how small the timing adjustment $a > 0$ (or the watermark embedding strength) might be, we can always make the watermark bit embedding success rate arbitrarily close to 100% by increasing the redundancy number r. This result holds true regardless of the distribution of the inter-packet timing of the packet flow.

Figure 5 illustrates how the distribution of $\overline{Y_{r,d}}$ can be "squeezed" into range $[-a, a]$ by increasing the redundancy number r.

Because the routers, intermediate Skype peers and the anonymizing network along the Skype VoIP call could introduce different delays over VoIP packets, we need to consider the negative impact of such delay jitters over the watermark decoding.

Let σ_d^2 be the variance of all delays added to all packets, X_k be the random variable that denotes the perturbation over $Y_{k,d}$ by the delay jitter, and $Y'_{k,d}$ be the random variable that denotes the resulting value of $Y_{k,d}$ after it has been perturbed by the delay jitter. We have the following quantitative tradeoff among the watermark bit detection rate, the defining characteristics of the delay jitter, and the defining characteristics of the original inter-packet timing of the VoIP flow, whose derivation can be found in the Appendix:

$$\begin{aligned} \Pr[\overline{Y'_{r,d}} < a] &\approx \Phi(\frac{a\sqrt{r}}{\sqrt{\sigma^2_{Y,d} + \sigma^2_d + 2\mathrm{Cor}(Y_{k,d}, X_k)\sigma_{Y,d}\sigma_d}}) \\ &\geq \Phi(\frac{a\sqrt{r}}{\sigma_{Y,d} + \sigma_d}) \end{aligned} \qquad (7)$$

Equation (7) gives us an accurate estimate of the watermark bit detection rate in the presence of delay jitters. The correlation coefficient $\mathrm{Cor}(Y_{k,d}, X_k)$, whose value range is [-1, 1], models any correlation between the network delay

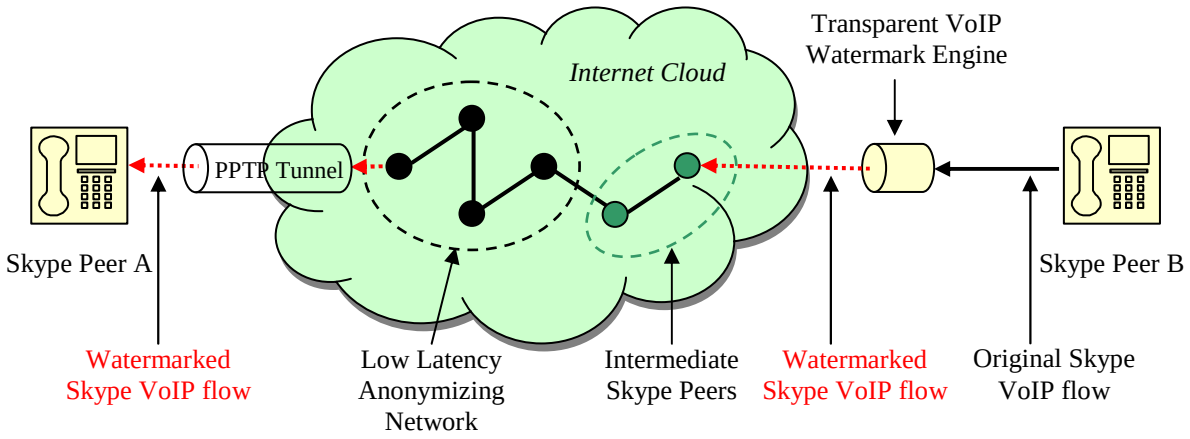


Figure 6: Experimental Setup for the Real-Time Tracking of Anonymous, Peer to Peer Skype VoIP Calls across the Internet

jitter and the packet timing of the original packet flow. In case the delay jitter is independent from the packet timing of the packet flow, $\mathrm{Cor}(Y_{k,d}, X_k)$ will be 0.

The important result here is that no matter what variance $Y_{k,d}$ may have (as long as it exists), no matter how large a variance the network jitter may have, no matter how small the timing adjustment $a > 0$ (or the watermark embedding strength) might be, we can always make the watermark bit detection rate arbitrarily close to 100% by increasing the redundancy number r. This result holds true regardless of the distribution of the network delay jitter.

4. TRANSPARENT WATERMARKING OF VOIP FLOWS IN REAL TIME

In order to be able to watermark any VoIP flows transparently, it is desirable to have a VoIP gateway which forwards the VoIP flows and watermarks any specified bypassing VoIP flows with specified watermarks. To embed the watermark into the inter-packet timing of a VoIP flow, we need a capability to delay specified packet of specified flow for specified duration. We choose to implement such a capability in the kernel of the Linux operating system.

One key challenge in implementing the transparent and real-time VoIP watermarking engine is how to precisely delay an outgoing packet in real-time. The inter-packet arrival time of normal VoIP flows is either 20ms or 30ms. This means that the delay of any VoIP packet must be less than 20ms. In order to hide the watermark embedding into the "background noise" introduced by the normal network delay jitter, the delay of any VoIP packet should be no more than a few milliseconds. To achieve packet delay of such a precision, the operating system must provide a hard real-time scheduling capability.

However, the standard Linux kernel lacks the hard real-time scheduling capability and it does not support time-critical tasks. Because the standard Linux is a time-sharing OS, the execution of any process depends on not only the priority of the process but also the current load in the OS, and there is no guarantee that a time-critical task will be processed and completed on time. In addition, the resolution of the software timer in the Linux kernel is by default 10ms, which is too coarse for our needs.

To achieve the guaranteed high precision, we choose to build our packet delay capability upon the Real Time Application Interface (RTAI) [23] of Linux. The following features of RTAI have made it an attractive platform for implementing the high precision packet delay capability:

- The hard real-time scheduling functions introduced by The RTAI coexist with all the original Linux kernel services. This makes it possible to leverage existing Linux kernel services, especially the IP stack components, from within the real-time task.
- The RTAI guarantees the execution time of real-time tasks regardless of the current load of non real-time tasks.
- The RTAI supports high precision software timer with the resolution of microseconds.

We built our transparent and real-time VoIP watermarking engine upon RTAI 3.1 in Linux kernel 2.6.8.1, and we implemented the VoIP watermarking engine as a RTAI kernel module. To facilitate the management of the kernel VoIP watermarking engine from user space, we also extended the netfilter/iptable mechanism in Linux kernel.

By integrating the RTAI hard real-time scheduling and the Linux kernel functionality, our real-time VoIP watermarking engine achieves the guaranteed delay precision of 100 microseconds over any specified packets of any specified flows despite the workload of the Linux kernel.

5. EXPERIMENTS

In this section, we empirically validate our active watermark based tracking of anonymous, peer-to-peer VoIP calls on the Internet. In specific, we conduct our experiments with real-time Skype peer-to-peer VoIP calls over the commercially deployed anonymizing system of findnot.com. Figure 6 shows the setup of our experiments. The Skype peer A is connected to some entry point of the anonymizing network of findnot.com via PPTP (Point to Point Tunnel Protocol) and all the Internet traffic of Skype peer A is routed through and anonymized by the anonymizing network of findnot.com. As a result, Skype peer B never sees the real

IP address of Skype peer A, and Skype peer A could appear to be some host of thousands miles away. In our experimental setup, the two communicating Skype peers are at least 27 hops away with about 60ms end to end latency.

We place our high precision VoIP watermarking engine between Skype peer B and the Internet and let it transparently watermark the VoIP flow from Skype peer B to peer A. We intercept the VoIP flow from the anonymizing network of findnot.com to Skype peer A, and try to detect the watermark from the intercepted VoIP flow.

While Skype VoIP call can use both TCP and UDP, we have found that it almost always use UPD. In our experiments, all the Skype calls happen to be UPD, and none of them has noticeable packet loss.

5.1 Watermarking Parameter Selection

Equation (7) gives us the quantitative tradeoff between the watermark bit detection rate, watermark embedding parameters and the defining characteristics of the network delay jitters.

To make the embedded watermark more robust against the network delay jitters and have high watermark bit detection rate, it is desirable to have larger watermark embedding delay a and bigger redundancy number r. However, a bigger watermark embedding delay means bigger distortion of the original inter-packet timing of the VoIP flow, which could potentially be used by the adversary to determine if a VoIP flow has been watermarked or not. Ideally, the delay introduced by the watermark embedding should be indistinguishable from the normal network delay.

To understand the normal network delay jitter as well as the hiding space for embedding our transparent watermark into the inter-packet timing of VoIP flows, we made a Skype call of 6 minutes long without watermarking, and collected the traces of the VoIP flows from both Skype peer A and B. We calculated the network delay jitter by comparing the timestamps of 10424 corresponding packets between the two VoIP flows. Figure 7 shows the distribution of the normalized network delay jitters. It indicates that there are about 50% chances that the network delay jitter will be equal to or bigger than 3ms. Therefore, it would be hard to distinguish any watermarked VoIP flow from unwatermarked ones if we embed the watermark with 3ms delay.

With watermark embedding delay a=3ms, we tried different redundancy numbers r to embed a 24-bit watermark into the Skype VoIP calls over the same anonymizing network of findnot.com. Figure 8 shows the average number of the error bits of the decoded watermarks of 10 Skype calls with a range of redundancy numbers. It clearly shows that the number of error bits can be effectively decreased by increasing the redundancy number r. With redundancy number r=25, the average number of error bits of the decoded 24-bit watermark is only 1.4.

In all of the following experiments, we use 24-bit watermarks with embedding delay a=3ms and redundancy number r=25. With this set of watermarking parameters, the watermarking of VoIP flow only requires 1200 packets to be delayed by 3ms. Given the 30ms packetization interval of Skype VoIP calls, the transparent watermarking can be applied to any VoIP calls that are as short as 90 seconds.

5.2 True Positive Experiments

We randomly generated 100 24-bit watermarks such that the Hamming distance between any two of them is at least 9. We then made 100 Skype calls of 2 minutes long and watermarked each of them with different watermark. We collected the originating and terminating watermarked VoIP flows from Skype peer B and A respectively, and decoded the 24-bit watermarks from them. We call any bit in the decoded 24-bit watermark that is different from the corresponding embedded bit as an *error bit*. Figure 9 shows the number of error bits of the 100 Skype VoIP calls and the watermark detection true positive rates given different numbers of allowed error bits. It indicates that very few of the 100 watermarked originating flows has 1 or 2 error bits, and a number of watermarked terminating flows has 1 to 6 error bits. If we require the exact match between the embedded watermark and the detected watermark, then we have 59% true positive rate. If the number of allowed error bits is increased to 4, the true positive rate becomes 99%. With number of allowed error bits being 6 or greater, we have 100% true positive rate.

5.3 False Positive Experiments

No matter what watermark we choose, it is always possible that an unwatermarked VoIP flow happens to have the chosen watermark naturally. We call this case as a *false positive* in correlating the VoIP flows.

We have shown that the true positive rate is generally higher if the number of allowed error bits is bigger. However, a bigger number of allowed error bits tends to increase the false positive rate. Therefore, it is important to choose an appropriate number of allowed error bits that will yield both high true positive rate and low false positive rate at the same time. To find the appropriate number of allowed error bits, we need to know the false positive rates under different numbers of allowed error bits.

Assuming the 24-bit watermark decoded from a random flow is uniformly distributed, then the expected false positive rate with $h \geq 0$ allowed error bits will be

$$\sum_{i=0}^{h} \left(\begin{array}{c} 24 \\ i \end{array} \right) (\frac{1}{2})^{24} \tag{8}$$

Because each of the 100 Skype calls is watermarked with different watermark, any of the 100 watermarked Skype flows has 99 uncorrelated watermarked Skype flows. Ideally, the number of different bits between the 24-bit watermarks decoded from different watermarked flows should be high.

Figure 10 shows the expected and measured numbers of different bits between the 24-watermarks decoded from the 9900 pairs of uncorrelated VoIP flows as well as the expected and measured watermark detection false positive rates under various numbers of allowed error bits. It indicates that the measured values are very close to expected value. This validates our assumption that the 24-bit watermark decoded from a random flow is uniformly distributed.

Out of the 9900 pairs of uncorrelated flows, no one has less than 6 different bits between the two watermarks decoded. There are 10 pairs of uncorrelated flows that have 6 different bits. Therefore, if we choose 5 as the number of allowed error bits, we would have 99% true positive rate and 0% false positive rate. If we use 6 as the number of allowed error bits, we would get 100% true positive rate and 0.1% false positive rate.

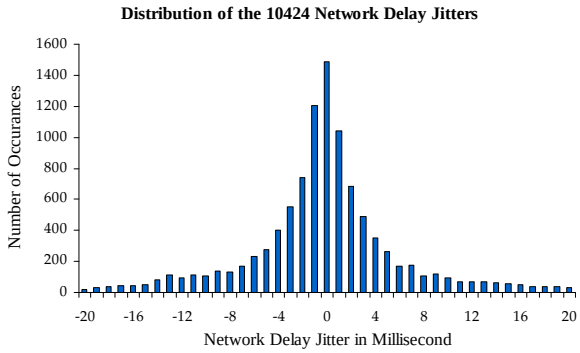


Figure 7: Distribution of the Network Delay Jitters of Skype VoIP Call

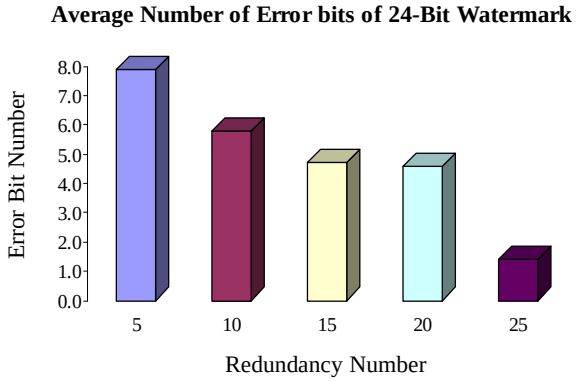
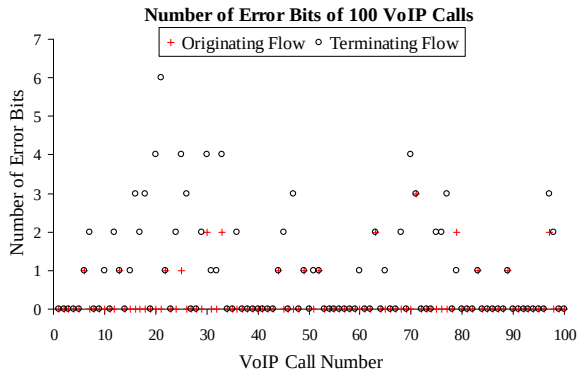


Figure 8: Average Number of Bit Errors vs the Redundancy Number r



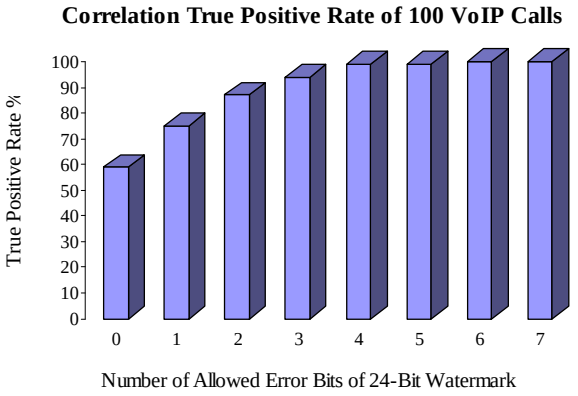
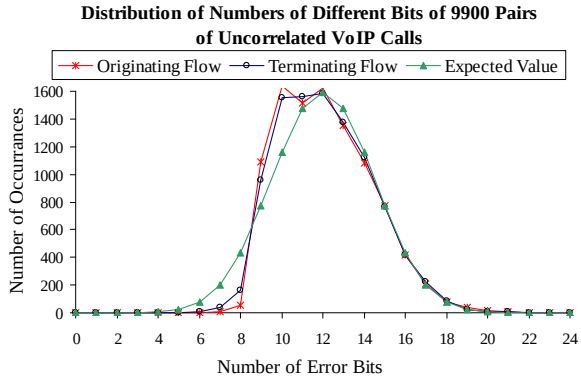


Figure 9: The Numbers of Error Bits and Correlation True Positive Rates of 100 Skype VoIP Calls



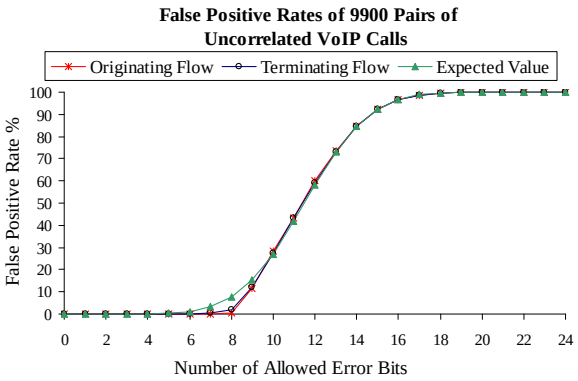


Figure 10: The Numbers of Error Bits and Correlation False Positive Rates of 9900 Pairs of Uncorrelated Skype VoIP Flows

6. RELATED WORKS

There have been substantial research works on how to trace attack packets with spoofed source address. Notably, Savage *et al.* [24] proposed IP traceback approach based on probabilistic packet marking (PPM), and Snoeren *et al.* [30] proposed logging based IP traceback approach. While both approaches have been shown to be effective in tracing the real source of large number of packets with spoofed source address, they can not be used directly to trace VoIP flows. Nevertheless, Savage's work demonstrated the potentials of active approach in tracing IP packets.

There are a number of works [34, 35, 33, 7, 32, 5] on how to trace encrypted attack traffic through stepping stones based on the inter-packet timing characteristics. Except Wang and Reeves' work [32], all other timing based approaches are passive. As the timing characteristics of VoIP flows are not distinct enough, passive examination of existing inter-packet timing of VoIP flows won't be able to distinguish different VoIP flows. Our proposed work differs from work [32] in that it does not require packet buffering to achieve the even time adjustment for embedding the watermark.

A number of low-latency anonymizing systems have been proposed to provide various levels of anonymity. Notably, Onion Routing [13] and its second generation Tor [6] aim to provide anonymous transport of TCP flows over the Internet. ISDN mixes [21] proposed a technique to anonymize the phone calls over the traditional PSTN. Tarzan [12] is an anonymizing network layer based on peer-to-peer model. Unlike most other anonymizing systems, Tarzan introduces cover traffic in addition to encrypting and relaying the normal traffic.

Felton and Schneider [10] identified a web caching exploiting technique that would allow malicious web site to infer whether its visitors have visited some other web pages, even if the browsing is protected by anonymizing services. Murdoch *et al.* [20] have recently investigated timing based attack on Tor with the assumption that the attacker controls a corrupt Tor node. Levine *et al.* [18] investigated passive timing based attack on low-latency anonymizing systems with the assumption that the attacker controls both the first and the last mix in the anonymizing network. However, none of these timing based approaches can be directly used to track VoIP calls.

7. CONCLUSIONS

Tracking encrypted, peer-to-peer VoIP calls has been widely viewed as impossible, especially when the VoIP calls are anonymized by the low latency anonymizing system. The key contribution of our work is that it demonstrates (1) tracking anonymous, peer-to-peer VoIP calls on the Internet is feasible; and (2) low latency anonymizing system is susceptible to timing based attack.

Our technique for tracking anonymous, peer-to-peer VoIP calls is based on subtle and deliberate manipulation of the inter-packet timing of selected packets of the VoIP flow. Our experiments of the real-time peer-to-peer VoIP calls over a commercially deployed anonymizing system show that the encrypted and anonymized VoIP flow could be made highly unique with only 3ms timing adjustment on selected packets. This level of timing adjustment is well within the range of normal network delay jitters. Our results also show that our watermark based tracking technique can be effectively applied to any peer-to-peer VoIP calls that are at least 90 seconds long.

8. REFERENCES

[1] Anonymizer. URL. http://www.anonymizer.com

[2] M. Arango, A. Dugan, I. Elliott, C. Huitema and S. Pickett. *RFC 2705: Media Gateway Control Protocol (MGCP) Version 1.0*. IETF, October 1999.

[3] A. Back, I. Goldberg, and A. Shostack. Freedom 2.1 Security Issues and Analysis. Zero-Knowledge Systems, Inc. white paper, May 2001

[4] S. A. Baset and H. Schulzrinne. An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol. Columbia Technical Report CUCS-039-04, December 2004

[5] A. Blum, D. Song, and S. Venkataraman. Detection of Interactive Stepping Stones: Algorithms and Confidence Bounds. In *Proceedings of the 7th International Symposium on Recent Advances in Intrusion Detection (RAID 2004)*. Springer, October 2004.

[6] R. Dingledine, N. Mathewson and and P. Syverson. Tor: The Second Generation Onion Router. In *Proceedings of the 13th USENIX Security Symposium*, August 2000.

[7] D. L. Donoho, A. G. Flesia, U. Shankar, V. Paxson, J. Coit and S. Staniford. Multiscale Stepping Stone Detection: Detecting Pairs of Jittered Interactive Streams by Exploiting Maximum Tolerable Delay. In *Proceedings of the 5th International Symposium on Recent Advances in Intrusion Detection (RAID 2002): LNCS-2516*, pages 17–35. Springer, October 2002.

[8] FBI. Letter to FCC http://www.askcalea.com/docs/20040128.jper.letter.pdf

[9] Federal Communications Commission. Notice of Proposed Rulemaking (NPRM) and Declaratory Ruling RM-10865, ET Docket No. 04–295, FCC 04–187. In *Federal Register at 69 Fed. Reg. 56956*, August, 2004.

[10] E. W. Felton and M. A. Schneider. Timing Attacks on Web Privacy. In *Proceedings of the 7th ACM Conference on Computer and Communications Security (CCS 2000)*, pages 25–32. ACM, November 2000.

[11] Findnot. URL. http://www.findnot.com

[12] M. J. Freedman and R. Morris. Tarzan: A Peer-to-Peer Anonymizing Network Layer. In *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS 2002)*, pages 193–206. ACM, November 2003.

[13] D. Goldschlag, M. Reed and P. Syverson. Onion Routing for Anonymous and Private Internet Connections In *Communications of ACM*, volume 42(2), Febrary 1999.

[14] M. T. Goodrich. Efficient Packet Marking for Large-scale IP Traceback. In *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS 2002)*, pages 117–126. ACM, October 2002.

[15] ITU-T Recommendation H.323v.4 Packet-based Multimedia Communications Systems. November 2000.

[16] Kazaa. URL. http://www.kazaa.com/

[17] T. Kohno, A. Broido and K. Claffy. Remote Physical Device Fingerprinting. In *Proceedings of the 2005 IEEE Symposium on Security and Privacy*, IEEE, 2005.

[18] B. Levine, M. Reiter, C. Wang, and M. Wright. Timing Attacks in Low-Latency Mix Systems. In *Proceedings of Financial Cryptography: 8th International Conference (FC 2004): LNCS-3110*, 2004.

[19] J. Li, M. Sung, J. Xu and L. Li. Large Scale IP Traceback in High-Speed Internet: Practical Techniques and Theoretical Foundation. In *Proceedings of the 2004 IEEE Symposium on Security and Privacy*, IEEE, 2004.

[20] S. J. Murdoch and G. Danezis. Low-Cost Traffic Analysis of Tor. In *Proceedings of the 2005 IEEE Symposium on Security and Privacy*, IEEE, 2005.

[21] A. Pfitzmann, B. Pfitzmann and M. Waidner. ISDN-MIXes" Untraceable Communication with Small Bandwidth Overhead. In *Proceedings of GI/ITG Conference: Communication in Distributed Systems, Mannheim*, Informatik-Fachberichte 267, pages 451–463, Springer-Verlag, 1991.

[22] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. R. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. *RFC 3261: SIP: Session Initiation Protocol.* IETF, June 2002.

[23] RTAI. URL. http://www.rtai.org

[24] S. Savage, D. Wetherall, A. Karlin, and T. Anderson. Practical Network Support for IP Traceback. In *Proceedings of ACM SIGCOMM 2000*, pages 295–306. ACM, September 2000.

[25] H. Schulzrinne. Internet Telephony. In *Practical Handbook of Internet Computing*, CRC, 2004

[26] H. Schulzrinne and J. Rosenberg. A Comparison of SIP and H.323 for Internet Telephony. In *Proceedings of International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV 1998)*, pages 83–86, Cambridge, England, July 1998.

[27] H. Schulzrinne and J. Rosenberg. Signaling for Internet Telephony. In *Proceedings of The 6th IEEE International Conference on Network Protocols (ICNP'98)*, October 1998.

[28] Skype - the Global Internet Telephony Company. URL. http://www.skype.org

[29] S. Snapp, J. Brentano, G. V. Dias, T. L. Goan, L. T. Heberlein, C. Ho, K. N. Levitt, B. Mukherjee, S. E. Smaha1, T. Grance, D. M. Teal, and D. Mansur. DIDS (Distributed Intrusion Detection System) - Motivation, Architecture, and Early Prototype. In *Proceedings of the 14th National Computer Security Conference*, pages 167–176, 1991.

[30] A. Snoeren, C. Patridge, L. A. Sanchez, C. E. Jones, F. Tchakountio, S. T. Kent, and W. T. Strayer. Hash-based IP Traceback. In *Proceedings of ACM SIGCOMM 2001*, pages 3–14. ACM, September 2001.

[31] S. Staniford-Chen and L. Heberlein. Holding Intruders Accountable on the Internet. In *Proceedings of the 1995 IEEE Symposium on Security and Privacy*, pages 39–49. IEEE, 1995.

[32] X. Wang and D. Reeves. Robust Correlation of Encrypted Attack Traffic Through Stepping Stones by Manipulation of Interpacket Delays. In *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS 2003)*, pages 20–29. ACM, October 2003.

[33] X. Wang, D. Reeves, and S. Wu. Inter-packet Delay Based Correlation for Tracing Encrypted Connections Through Stepping Stones. In *Proceedings of the 7th European Symposium on Research in Computer Security (ESORICS 2002), LNCS-2502*, pages 244–263. Springer-Verlag, October 2002.

[34] K. Yoda and H. Etoh. Finding a Connection Chain for Tracing Intruders. In *Proceedings of the 6th European Symposium on Research in Computer Security (ESORICS 2000), LNCS-1895*, pages 191–205. Springer-Verlag, October 2002.

[35] Y. Zhang and V. Paxson. Detecting Stepping Stones. In *Proceedings of the 9th USENIX Security Symposium*, pages 171–184. USENIX, 2000.

APPENDIX

Let D_i $(i = 1, \ldots, n)$ represent the random delays added to packet P_i by the adversary, let $D > 0$ be the maximum delay the adversary could add to any packet, and let σ_d^2 be the variance of all delays added to all packets. Here we make no assumption about the distribution of random delay the adversary could add to each packet except that the delay is bounded. For example, D_i and D_j $(i \neq j)$ could be correlated to each other and/or have different distributions. This models all the possible bounded random delays the adversary could add to a packet flow.

Given the assumption that the adversary does not know how and which packets are selected by the information hider, the selection of information embedding packet P_{z_k} $(k = 1, \ldots, 2r)$ is independent from any random delay D_i the adversary could add. Therefore, the impact of the random delays by the adversary over randomly selected P_{z_k} is equivalent to randomly choosing one from the random variable list: $D_1, \ldots, D_n$. Let b_k $(k = 1, \ldots, 2r)$ represent the impact of the random delays by the adversary over the k-th randomly selected packet P_{z_k}. Apparently the distribution of b_k is a compound one that depends on the probability that each D_i would be selected. Since each P_{z_k} is randomly selected according to the same probability distribution over $P_1, \ldots, P_n$, each b_k has the same compound distribution. Furthermore, because each P_{z_k} is selected independently, b_k is also independent from each other. In other words, the impact of any random delays by the adversary over those independently and randomly selected information bearing packets is independent and identically distributed (*iid*), and it is essentially an *iid* random sample from the random delays the adversary added to all packets.

Let $x_{1,k}$ and $x_{2,k}$ be the random variables that denote the random impact over $ipd_{1,k,d}$ and $ipd_{2,k,d}$ respectively. Apparently both $x_{1,k}$ and $x_{2,k}$ are *iid*. It is also easy to see that $x_{1,k}, x_{2,k} \in [-D, D]$, $\mathrm{E}(x_{1,k}) = \mathrm{E}(x_{2,k}) = 0$, and $\mathrm{Var}(x_{1,k}) = \mathrm{Var}(x_{2,k}) = 2\sigma_d^2$. Let $X_k = (x_{1,k} - x_{2,k})/2$, then X_k is *iid*, $\mathrm{E}(X_k) = 0$, and $\mathrm{Var}(X_k) = \sigma_d^2$.

Let $Y'_{k,d}$ be the random variable that denotes the resulting value of $Y_{k,d}$ after it is perturbed by $x_{1,k}$ and $x_{2,k}$, then we have

$$\begin{aligned} Y'_{k,d} &= [(ipd_{1,k,d} + x_{1,k}) - (ipd_{2,k,d} + x_{2,k})]/2 \\ &= (ipd_{1,k,d} - ipd_{2,k,d})/2 + (x_{1,k} - x_{2,k})/2 \\ &= Y_{k,d} + X_k \end{aligned} \quad \text{(A-1)}$$

Therefore, $\mathrm{E}(Y'_{k,d}) = 0$. Since $Y_{k,d}$ is *iid* and X_k is *iid*, $Y'_{k,d}$ is also *iid*.

$$\begin{aligned} \mathrm{Var}(Y'_{k,d}) &= \mathrm{Var}(Y_{k,d}) + \mathrm{Var}(X_k) + 2\mathrm{Cov}(Y_{k,d}, X_k) \\ &= \sigma^2_{Y,d} + \sigma^2_d + 2\mathrm{Cor}(Y_{k,d}, X_k)\sigma_{Y,d}\sigma_d \\ &\leq \sigma^2_{Y,d} + \sigma^2_d + 2\sigma_{Y,d}\sigma_d \\ &= (\sigma_{Y,d} + \sigma_d)^2 \end{aligned} \quad \text{(A-2)}$$

Let $\overline{Y'_{r,d}}$ be the random variable that denotes the resulting value of $\overline{Y_{r,d}}$ after it is perturbed by $x_{1,k}$ and $x_{2,k}$, then we have

$$\overline{Y'_{r,d}} = \frac{1}{r}\sum_{k\,=\,1}^{r} Y'_{r,d} \quad \text{(A-3)}$$

According to the property of variance of independent random variables, $\mathrm{Var}(\overline{Y'_{r,d}}) = \mathrm{Var}(Y'_{k,d})/r$. It is also easy to see that $\mathrm{E}(\overline{Y'_{r,d}}) = 0$.

By applying the Central Limit Theorem to random sample $Y'_{1,d}, \ldots, Y'_{r,d}$, where $\mathrm{E}(Y'_{k,d}) = 0$, and $\mathrm{Var}(Y'_{k,d}) = \sigma^2_{Y,d} + \sigma^2_d + 2\mathrm{Cor}(Y_{k,d}, X_k)\sigma_{Y,d}\sigma_d$, we have

$$\begin{aligned} \Pr[\frac{\sqrt{r}(\overline{Y'_{r,d}} - \mathrm{E}(Y'_{k,d}))}{\sqrt{\mathrm{Var}(Y'_{k,d})}} < x] &= \Pr[\frac{\sqrt{r}\overline{Y'_{r,d}}}{\sqrt{\mathrm{Var}(Y'_{k,d})}} < x] \\ &= \Phi(x) \end{aligned} \quad \text{(A-4)}$$

Therefore

$$\begin{aligned} \Pr[\overline{Y'_{r,d}} < a] &= \Pr[\frac{\sqrt{r}\overline{Y'_{r,d}}}{\sqrt{\mathrm{Var}(Y'_{k,d})}} < \frac{a\sqrt{r}}{\sqrt{\mathrm{Var}(Y'_{k,d})}}] \\ &\approx \Phi(\frac{a\sqrt{r}}{\sqrt{\mathrm{Var}(Y'_{k,d})}}) \\ &= \Phi(\frac{a\sqrt{r}}{\sqrt{\sigma^2_{Y,d} + \sigma^2_d + 2\mathrm{Cor}(Y_{k,d}, X_k)\sigma_{Y,d}\sigma_d}}) \\ &\geq \Phi(\frac{a\sqrt{r}}{\sigma_{Y,d} + \sigma_d}) \end{aligned} \quad \text{(A-5)}$$