NetCamo: Camouflaging Network Traffic for QoS-Guaranteed Mission Critical Applications

Yong Guan, Xinwen Fu, Dong Xuan, Prashanth Umesh Shenoy, Riccardo Bettati, Wei Zhao

Abstract—**This paper presents the general approach, design, implementation, and evaluation of NetCamo, a system to prevent traffic analysis in systems with real-time requirements. Integrated support for both security and real-time is becoming necessary for computer networks that support mission critical applications. This study focusses on how to integrate both the prevention of traffic analysis and guarantees for worst-case delays in an internetwork. We propose and analyze techniques that efficiently camouflage network traffic and correctly plan and schedule the transmission of payload traffic so that both security and real-time requirements are met. The performance evaluation shows that our NetCamo system is effective and efficient. By using the error between target camouflaged traffic and the observed (camouflaged) traffic as metric to measure the quality of the camouflaging, we show that NetCamo achieves very high levels of camouflaging without compromising real-time requirements.**

Keywords—**Network Security, QoS, Traffic Analysis, Traffic Rerouting and padding, Anonymous Communication**

I. Introduction

In this paper, we discuss the design, implementation, and evaluation of NetCamo, a software toolkit that is capable of preventing traffic analysis while guaranteeing worst-case delays in computer networks for mission critical applications.

Traditionally, encryption has played an important role in network security. However, it is a misconception that to secure a network, one only needs to encrypt the traffic. With increasing amounts of traffic being encrypted and its contents therefore being beyond the reach of effective cryptanalysis, attention is shifting towards *traffic analysis*, and the prevention thereof. Traffic analysis is a security attack where an intruder observes network traffic in order to infer sensitive information about the applications and/or the underlying system. This form of attack is harmful because significant information about operation modes can be inferred by appropriately monitoring the pattern of traffic. It can, for example, uncover the location of command centers, determine the state of alertness of various units, or detect covert information flows to or from apparently non-involved parties [33]. In addition, effective traffic analysis is well known to greatly help the cryptanalysis efforts [33]. It is therefore important to develop means to render traffic analysis efforts ineffective.

Traffic analysis can be prevented by *camouflaging* the payload traffic, i.e., manipulating the traffic so that its pattern is not related, for an observer, to the operational status of applications. To achieve this, an integration of the following measures should be used:

- Traffic Padding. Additional packets (called padding packet) my need to be properly inserted into payload packet streams to camouflage them.
- Traffic Re-Routing. Usually, packets from one host to another are sent via one fixed path. In order to prevent traffic analysis, a stream of traffic between two hosts may need to be re-routed through multiple paths in order to camouflage the traffic.

The challenge of this study is that we deal with the problem of preventing traffic analysis in the context of mission critical system where the worst-case delay of payload packets needs to be guaranteed. This is not possible when the network is indiscriminately flooded by padding traffic.

Delay guarantees are realized by adopting a connection-oriented communication service model in conjunction with admission control during connection establishment. Before an application can transmit a flow of packets to another, a connection admission request is made, and the connection is admitted only if the delay requirements of both new and existing connections can be met. To achieve the prevention of traffic analysis in conjunction with delay guarantees, the design of our NetCamo system enhances the connection admission control module in order to perform both traditional admission control and traffic planning for camouflaging. We propose and analyze an efficient algorithm that is able to select a set of feasible routes for a connection so that (1) the delay requirements can be met and (2) the traffic can be correctly camouflaged.

After admission control and rerouting decisions have been made in the traffic planner, the traffic must be correctly *padded* to camouflage it. The padding algorithm is therefore critical for the performance of NetCamo, as it directly impacts the quality of traffic camouflaging. Furthermore, because it deals with each payload packet, the padding algorithm should not cause too much overhead We addressed these issues in the design of the padding algorithm used in NetCamo.

NetCamo has been realized in our laboratory in the Department of Computer Science at Texas A&M University. A thorough performance evaluation has been carried out. Performance data we collected indicate the system design objectives have been achieved:

- NetCamo effectively camouflages the traffic and hence prevents traffic analysis. We will show that the difference between targeted camouflaged traffic patten and real traffic patten is typically very small. In practice, this makes it extremely difficult for an observer to analyze the traffic and obtain status information of applications that are using the network.
- NetCamo does not compromise on delay guarantees. We measure the probability of real-time connections being admitted. Using this measure, we compare NetCamo with a system that does not do prevention of traffic analysis. The data shows that both systems have virtually identical performance in terms of admission capability.

The rest of this paper is organized as follows: Section II discusses the related work while Section III presents the network and traffic models we use. In Section IV, the methodologies

The authors are with the Department of Computer Science, Texas A&M University, College Station, TX 77843-3112. E-mail:{yguan, xinwenfu, dxuan, pshenoy, bettati, zhao}@cs.tamu.edu.

This work was partially sponsored by NSF under contract number EIA-0081761, by DARPA under contract number F30602-99-1-0531, and by Texas Higher Education Coordinating Board under its Advanced Technology Program.

we propose to use in NetCamo is discussed in detail. Implementation of NetCamo is presented in Section V. Performance evaluation is carried in Section VI. Section VII summaries the paper with a discussion on future extensions.

II. Related work

In this section, we survey previous work that is related to our study. Recently, there have been several reports on anonymous communication. The essence of anonymous communication is to hide sender identity and/or receiver identity from outside observers. The design, analysis, and implementation of systems that can provide untraceable email services [6], [18] or web browsing [1], [2], [17], [31], [37] anonymity have been reported. [13], [20], [30], [37] study how to provide anonymous connection oriented services. We note that the majority of these studies on anonymous communication do not directly address prevention of traffic analysis.

However, some of the solutions developed with anonymous communication can serve as a (partial) countermeasure for traffic analysis. For example, onion routing [13], [30] provides protection from traffic analysis between onion routers, however, it does not provide end-to-end protection since it does not protect from traffic analysis at end points. Real-Time Mixes [20] are able to hide the identities of senders and receivers. But, they do not prevent traffic analysis between two local exchanges (i.e., domains).

Theoretical issues on traffic analysis prevention have been investigated in [25], [26], [35], [2], [6]. The method proposed in [25], [26], [35] is to convert a given traffic pattern into so-called *neutral* traffic pattern. However, none of these studies considered the impact of rerouting and padding of the traffic to QoS requirements. Progress was made in [15] where the system is able to prevent traffic analysis and guarantee QoS. This work is limited to a static connection-oriented system where dynamic arrival of connection is not allowed.

Our study reported in this paper is significantly different from the previous work in that we do provide complete (end-to-end) prevention of traffic analysis while guaranteeing worst case delays of payload traffic. We address meeting both security and real-time requirements in an integrated manner.

III. Network and Traffic Model

In this section, we describe the model and define some terminology used in the following sections.

System Model A network consists of a number of host computers and network devices (e.g., routers or switches). They are connected by physical links on which packets are forwarded. For delay computation purposes, we model a router or a switch as having multiplexers and de-multiplexers at its output and input ports. The queueing delay of the packets happens at the multiplexer, where the packets compete for the output link. We denote C as the output link capacity, in bits per second.

We assume that an underlying routing subsystem determines a unique path between any pairs of hosts. We would, therefore, model the network as a fully connected, directed graph, $G = (V, E)$, with vertices in V being the hosts and the edges in E representing the paths from the source s to the destination t through some routers in the network. We assume that an observer can monitor the amount of traffic along each edge of this graph. Without precautions, this observer can determine the traffic pattern between each source-destination pair. Our work will be aimed at preventing such information from being released to the observer.

Traffic Model We adopt a connection-oriented model to provide communication services. A connection can be viewed as a flow of packets from a source to a destination. The defining characteristic of connection-oriented communication is the existence of a connection establishment phase preceding the actual data transfer. At establishment time, connection admission control (CAC) determines whether a connection request can be admitted or not and the necessary resources for the new connection are allocated if the connection can be accepted. During the lifetime, the connection is controlled by packet scheduling and policed to ensure that abnormal behavior of a connection does not affect other connections in the system.

Each connection is characterized by a connection identifier and a Quality-of-Service (QoS) specification. Typically, the connection identifier in an IP environment would be the five tuple consisting of source and destination address/port and the protocol identifier. In the following discussion, we assume that the QoS specification defines the traffic (in terms of average rate ρ and burst size σ) and the deadline requirement (D) of the connection. That is, the total amount of traffic during any time interval $[t, t + I)$ is bounded by $\min\{C * I, \sigma + \rho * I\}$.

Connection oriented communication service is currently provided by two architectures: Integrated Service (IS) [4] and Differentiated Service (DS) [3]. In an IS architecture, different connections can have different QoS parameters (σ, ρ, D). This necessitates that information about every connection is kept by each router along the path for admission control and packet forwarding purposes. By explicitly computing delays for all connections, admission control makes sure that the deadline requirements for both the new connection and the existing connections can be met. An advantage of the IS architecture is its flexibility in connection QoS specification. However, IS is difficult to deploy in large-scale high-speed networks, as it doesn't scale well, for two reasons: first, high speed routers are required to maintain and schedule packets for a large number of connections. Second, as the number of connections increases, the runtime overhead (e.g., delay calculation) incurred for connection establishment and tear-down increases as well.

One step further, the DS architecture takes scalability issues into account. In DS, connections are partitioned into several predefined *classes*. QoS parameter and specification of the traffic carried by the connections are defined on a class-by-class basis. That is, the QoS of connections in the same class has the same mathematical representation. Packets of the same class are served according to a class-based scheduling policy. The result is that routers inside the network are aware only of aggregations of connections. In this fashion, the DS architecture makes the network scalable regardless of the number of connections.

Following the DS architecture, at each link, certain percentage of bandwidth is reserved for individual traffic class. Without explicit delay calculation, connection admission control can guarantee that the deadline requirements for both new connection and existing connections can be met by ensuring that the

bandwidth usage of individual classes does not go beyond the reserved portion. This is necessary to provide isolation among classes and hence to guarantee end-to-end delays to the connections in each class.

IV. Methodologies

We model the payload traffic between Host i and Host j as a random process $\{p_{ij}(t), t \geq 0\}$. Considering all the pairs of hosts in the system, we model the system payload traffic pattern by a 2-tuple (p, π), where matrix $p = (p_{ij}(t))_{n\times n}$ is called *payload traffic pattern matrix* and matrix $\pi = (\pi_{ij})_{n\times n}$ is called *payload traffic rate matrix*, where $\pi_{ij} = \sum_k \rho_k$ and ρ_k is the average rate of the connection k from Host i to Host j.

If an observer is able to monitor the payload traffic in the system and analyze the statistical characterizations of payload traffic, e.g., the matrix π, she or he can infer some sensitive information about operational mode of the system. We want to develop a countermeasure for this kinds of security attack.

To render such kinds of traffic analysis ineffective, we should deploy appropriate approaches to hide the statistical properties of the payload traffic. We achieve this by presenting the observer a distorted traffic pattern called *camouflaged traffic pattern* (CTP). Similarly to payload traffic, we can also model camouflaged traffic between Host i and Host j as a random process $\{c_{ij}(t), t \geq 0\}$ with time-independent mean γ_{ij}. Then, we can model the system camouflaged traffic using *camouflaged traffic pattern matrix* $c = (c_{ij}(t))_{n\times n}$ and *camouflaged traffic rate matrix* $\gamma = (\gamma_{ij})_{n\times n}$. Camouflaged traffic patterns can be either constant or random in general. In this study, our countermeasure to the traffic analysis is to make camouflaged traffic pattern as a periodic process with an average rate of $\gamma_{ij} = \alpha$. Even though an observer can learn the camouflaged traffic pattern, she cannot derive information on payload traffic pattern. Thus, the goal of preventing traffic analysis can be achieved.

As discussed earlier, hiding the payload traffic pattern is only one of our two objectives. The other is to provide QoS guarantees. That is, in mission-critical systems, many applications have real-time requirements, since delay-sensitive data are exchanged. We cannot deploy an approach that just hides the payload traffic pattern without taking applications' real-time deadline requirements into account. The key problem is how to manipulate the payload traffic such that the real traffic in the system appears to be the camouflaged traffic pattern and the QoS requirements of the applications are guaranteed as well.

Towards these purposes, we employ appropriate methodologies in three stages: system configuration phase, admission control phase, and run-time phase that together achieve our two objectives.

- During the system configuration phase, a suitable camouflaged traffic pattern needs to be determined. Once the camouflaged traffic pattern matrix is decided, *host-to-host delay* will also be analyzed.
- During the admission control phase, a traffic plan is generated for the newly arrived connection. The traffic plan specifies how to transmit the traffic of the connection into several routes without affecting the camouflaged traffic pattern and violating real-time requirements of the new and existing connections in the system.
- At run-time, we apply low-level *traffic control* (by sending additional padding traffic on communication paths) to compensate for fluctuations at connection level so that the real traffic pattern appears to be the camouflaged traffic pattern.

A. System Configuration Phase

At system configuration phase, a camouflaged traffic pattern needs to be determined. And then the host-to-host worst-case delay can be analyzed. In this sub-section, we will discuss how to determine the camouflaged traffic pattern and how to analyze the host-to-host worst delay.

A.1 Design of Camouflaged Traffic Pattern

As discussed above, the camouflaged traffic is modeled as a periodic process. The design of camouflaged traffic pattern is to select camouflaged traffic rate matrix γ. This is important, since the camouflaged traffic pattern has direct impacts to the performance of a network. With larger value of the element of matrix γ, more connections can be accepted under the constraint of camouflaged traffic pattern. However, it also results in larger delay to the payload packets. This may violate their QoS requirements. With smaller value of the elements of matrix γ, the packets may suffer smaller delay. But this may not allow us to hide the payload traffic in some cases.

In Integrated-service network, the camouflaged traffic pattern must be constrained by the link capacity. Assume that the link capacity is C. We have a system of inequalities: for every link L,

$$\sum_{L\in path_{ij}} \gamma_{ij} \leq C, \tag{1}$$

where $path_{ij}$ is the path from Host i to Host j determined by the underlying system. It says that the sum of the camouflaged traffic pattern between all the pair of hosts whose paths share the same link L cannot exceed the link capacity C. This system of inequalities specifies one of the correctness constraints that the camouflaged traffic pattern must satisfy.

On the other hand, in Differentiated-service network, the bandwidth (αC) provided to a specific class at each link is determined at system configuration time. That is, the percentage (α) of the link capacity assigned to a specific class is fixed during the system run-time. Thus, the camouflaged traffic pattern for DS network must satisfy the following system of inequalities, for every link L,

$$\sum_{L\in path_{ij}} \gamma_{ij} \leq \alpha C. \tag{2}$$

Obviously, depending on the type of system and the nature of applications, many constraints can be proposed. Thus, generally speaking, generating γ can be modeled as an optimization problem. For example, in DS network, a weight ϕ_{ij} may be assigned to the path from Host i to Host j, for all host pairs (say, in order to satisfy some requirement of applications in the system). The objective is to generate a matrix γ, which can maximize $\sum_{0\leq i,j<n} \phi_{ij}\gamma_{ij}$. We have the following optimization problem:

$$\text{Maximize} \quad \sum_{0 \le i,j < n} \phi_{ij}\gamma_{ij} \tag{3}$$

$$\text{Subject to} \quad \sum_{L \in path_{ij}} \gamma_{ij} \le \alpha C, \tag{4}$$

$$\text{and} \quad \phi_{ij} > 0, \tag{5}$$

$$\text{and} \quad \gamma_{ij} > 0, \ (i \ne j) \tag{6}$$

$$\text{and} \quad \gamma_{ij} = 0 \ (i = j) \tag{7}$$

Such an optimization problem may be solved using standard mathematical software package, such as [19], [24].

A.2 Host-to-host Delay Analysis

Once camouflaged traffic pattern is given, we can derive the host-to-host worst case delay, because the traffic transmitted between any pair of hosts is known (defined in camouflaged traffic pattern).

The scheduling policy at a link determines the order in which packets from a traffic flow are transmitted over the link. Hence, the link scheduling policy has a direct impact on the delays experienced by a traffic flow's packet over the link as well as on the distortion of the traffic within the network.

The worst-case delay can be calculated. Formulas for the delay of a variety of link scheduling disciplines exist. For links with a First-Come-First-Served (FCFS) discipline, for example, the worst case delay experienced by any packet at the link is the same for any traffic flow traversing it. For a network with FCFS link scheduling, the maximum delay at a link x is given by

$$d_{*,x} = \max_{I \ge 0}\left(\sum_{k=1}^{L_x} F_{k,x}(I) - I\right), \tag{8}$$

where $F_{k,x}(I)$ is the maximum number of packets that can arrive at Link x over its k-th input link during any interval of length I, and L_x is the number of links that feed into Link x at the switch. A wealth of delay formulas for other scheduling policies are available in the literature [9], [12], [14], [22], [29], [36], [38].

B. Admission Control Phase

Whenever a new connection comes, admission control determines whether it can be accepted or not. The new connection can be accepted if a traffic plan on how to transmit the packets of the new connection without violating the camouflaged traffic pattern and the deadline requirement can be generated. So the core of the admission control is traffic planning.

In the following, we will discuss a concept called host-based rerouting that is useful in our traffic planning. We then focus on the correctness and generation of a traffic plan.

discussion about what is a correct traffic plan and how to generate a correct traffic plan.

B.1 Host-based Rerouting

Intuitively, we may use the following algorithm to determine if a new connection can be accepted:

Step 1. Select the direct path from source to the destination.

Step 2. If there is a violation of camouflaged traffic pattern, then reject this new connection.

Step 3. If the real-time deadline requirement cannot be met, then reject it; else accept this new connection.

The above algorithm uses the direct path to send all the packets of the accepted connections from their sources to the destinations (hence called *direct-sending*). Due to the limitation that only direct path can be used, a connection may not be accepted because of no available capacity on the direct path due to the constraint of camouflaged traffic pattern.

Consider the following example with 3 hosts. Suppose that the chosen camouflaged traffic pattern enforces a stream of 3 MB/sec between any two hosts. This pattern can therefore be described as follows:

$$\gamma = \begin{pmatrix} 0 & 3 & 3 \\ 3 & 0 & 3 \\ 3 & 3 & 0 \end{pmatrix}. \tag{9}$$

Suppose that, at time instant t, the payload traffic rate matrix that indicates the rates of existing payload traffic is given by

$$\pi = \begin{pmatrix} 0 & 0 & 3 \\ 3 & 0 & 3 \\ 2 & 0 & 0 \end{pmatrix}, \tag{10}$$

where π_{ij} represents the average rate of the connection from Host i to Host j.

Now, let's assume that a new connection from Host 3 to Host 2 with bandwidth requirement of $\rho = 4MB/sec$ comes. By the simple direct-sending algorithm mentioned above, it is obvious that this new connection cannot be accepted, since 4MB/sec exceeds the camouflaged traffic pattern from Host 3 to Host 2 (defined by $\gamma_{32} = 3MB/sec$).

However, such a connection may be accepted if we reroute some of its traffic through some intermediate hosts from its source to its destination so that both the camouflaged traffic pattern and real-time deadline requirements can be met. We call this approach as *host-based rerouting*. If host-based rerouting is allowed[1], a traffic plan for this new connection can be generated as follows: two paths, a direct path (from Host 3 to Host 2) and a rerouting path (from Host 3 through Host 1 to Host 2), can be used to transmit the traffic of the new connection, where the amount of traffic of the new connections assigned to these two paths is $3MB/sec$ and $1MB/sec$, respectively. After this new connection is admitted, the new payload traffic rate matrix becomes:

$$\pi^{new} = \begin{pmatrix} 0 & 1 & 3 \\ 3 & 0 & 3 \\ 3 & 3 & 0 \end{pmatrix}. \tag{11}$$

The benefit of host-based rerouting is to increase the admission probability.

[1]Here we assumes that the deadline is large enough to do host-based rerouting.

B.2 Traffic Planning

A traffic plan for a new admitted connection specifies how its payload packets are transmitted. Formally, a traffic plan can be expressed by two vectors: $(p_{st}^1, p_{st}^2, \ldots, p_{st}^K)$ and $(q_{st}^1, q_{st}^2, \ldots, q_{st}^K)$, where s is the source and t is the destination, K is the total number of paths (including direct-path and host-based rerouting path) in the traffic plan, host-based rerouting path $p_{st}^i =< s, i_1, i_2, \ldots, i_m, t >$ specifies path i, starting with the source s, a sequence of the intermediate hosts $i_1, i_2, \ldots, i_m$, and ending with the destination t. The length of p_{st}^i can be 2, i.e., $p_{st}^i =< s, t >$, which means this path is the direct path. q_{st}^i specifies the amount of the payload traffic transmitted along Path i. We say, p_{st}^k *passes through* $< s, t >$, if $p_{st}^i =< s, \ldots, i, j, \ldots, t >$

B.3 Correctness of Traffic Planning

The acceptance of the new connection should not affect the camouflaged traffic pattern or cause the real-time requirements of the new and existing connections be missed. So a correct traffic plan for the new connection must satisfy a number of constraints, which we discuss in the following.

By *stabilization constraint*, we mean that the new and existing connections' payload traffic can be sent according to their traffic plan without the real traffic pattern exceeding the camouflaged traffic pattern. Formally, for any given host pair $< i, j >$,

$$\sum_{p_{uv}^k \text{ passes through } <i,j>} q_{uv}^k \leq \gamma_{ij}. \tag{12}$$

Conservation constraint ensures that the correct amount of traffic is rerouted at each node. If a node is the source or destination of the traffic, the aggregate output or input traffic must be equal to the outgoing or incoming payload traffic, respectively. Formally,

$$\sum_{k=1}^{K} q_{ij}^k = \pi_{ij} \tag{13}$$

at the source Host i, and

$$\sum_{k=1}^{K} q_{ij}^k = \pi_{ij} \tag{14}$$

at the destination Host j.

If a node is used as intermediate node in some rerouting path(s), the aggregate input rerouting traffic of this node must be equal to its aggregate output rerouting traffic. That is,

$$\sum_{p_{ij}^k \text{ passes through } <u,v>} q_{ij}^k = \sum_{p_{ij}^k \text{ passes through } <v,w>} q_{ij}^k, \tag{15}$$

where $v \neq i$ and $v \neq j$.

Delay constraint ensures that the real-time requirements are met, that is, that all traffic can be sent to its destination by its respective deadline. Formally,

$$d_{ij}^{WC} \leq DL_{ij}, \tag{16}$$

where d_{ij}^{WC} is the worst-case message delay experienced by the traffic flow from Host i to Host j considering rerouting and addition of padding traffic. Determining end-to-end delays d_{ij}^{WC} in this case is somewhat complicated by the fact that we do host-based rerouting and traffic padding. We can derive it based on the result of host-to-host worst case delay discussed in Section IV-A.2.

In order to determine the end-to-end delay of connections, we distinguish between the *direct path* of the connection, which is the path from the source host to the destination host as determined by the underlying routing subsystem (e.g., OSPF), and the *rerouted path*, which is the path assigned to the connection by the traffic planner. We denote the worst-case delays of messages along the two paths by $d_{ij}^{WC\_direct}$ and $d_{ij}^{WC\_reroute}$, respectively. Once the delays along the direct paths within the network are known, the value for $d_{ij}^{WC\_reroute}$ is determined by taking the maximum of the delays along all the host-based rerouting paths used to transmit the traffic between the two hosts. The delay along each rerouting path is computed by summing up the delays on the direct paths between host-based routers on the rerouting path, i.e. $d_{ij}^{WC\_reroute} = \max_{\forall k}(\sum_{m=0}^{|p_{ij}^k|-1} d_{i_m, i_{m+1}}^{WC-direct})$, where $|p_{ij}^k|$ is the length of the host-based rerouting path p_{ij}^k, $i_0 = i$ and $i_{|p_{ij}^k|} = j$. The end-to-end worst case delay d_{ij}^{WC} of a traffic flow is then formulated as:

$$d_{ij}^{WC} = \max(d_{ij}^{WC\_direct}, d_{ij}^{WC\_reroute}). \tag{17}$$

B.4 Traffic Plan Generation Algorithm

We have discussed the correctness constraints of a traffic plan. Now we address how to generate a correct traffic plan for the new connection.

We embed the traffic planning procedure into the connection establishment of the underlying network service. A framework for a Traffic Planning Algorithm is illustrated in figure 1. The traffic planner needs to determine whether admitting the new connection would violate either the camouflaged traffic pattern or the QoS requirements of a connection. If any of the two is violated, the connection establishment request must be rejected.

Selecting the path to transmit the traffic of the new connection is important, not only because it is related to the QoS requirement, but also because it has direct impact on the effectiveness of the algorithm (bad path selection can cause the lower admission probability). In this traffic planner, both direct path and rerouted path can be selected. The second step of the algorithm is to check if delay constraint can be met. In IS network, whenever a new connection comes, delay should be analyzed to make sure that the real-time deadline requirement of this new connection and other existing connections be met. Thus the delay constraints can be guaranteed. In DS network, the delay constraints can be easily guaranteed by checking the bandwidth usage of the class which the new connection belongs to does not exceed the pre-defined resource reservation for this class.

By carefully selecting the host-based rerouting paths and the amount of payload traffic transmitted along the selected paths, the traffic plan can satisfy the stabilization constraints. Conservation constraints can be satisfied by enforcing the intermediate

Notations:
s, t: the source and destination of the new connection
New connection's QoS requirement: bandwidth requirement ρ^{new} and deadline D^{new}
Camouflaged traffic rate matrix γ
AC_{uv}: the available capacity of the direct-path from Host u to Host v, Initially, $AC_{uv} = \gamma_{uv}$ for all the hosts u, v in the system
$CAP_{p_{st}^k}$: the capacity of the host-base rerouting path p_{st}^k, where $CAP_{p_{st}^k} = \min_{p_{st}^k \text{passes through} <u,v>} \{AC_{uv}\}$
$d_{p_{st}^k}^{WC}$: the worst case delay happened along path p_{st}^k
INPUT: ρ^{new}, D^{new}, and AC_{uv} for all u, v
1. Among the direct and rerouting paths, select a path p_{st}^k with the smallest delay path and $CAP_{p_{st}^k} \geq 0$ according to Shortest Path First (i.e.,smallest delay path)
2. If there is no available path
then reject this new connection.
3. If the real-time deadline requirement can not be met (i.e., $d_{p_{st}^k}^{W} C > D^{new}$)
then reject this new connection.
else do:
3.1 $q_{st}^k = \min\{\rho^{new}, CAP_{p_{st}^k}\}$
3.2 $\rho^{new} = \max\{0, \rho^{new} - q_{st}^k\}$;
3.3 If $\rho^{new} = 0$
then do:
3.3.1 For all the paths p_{st}^k selected, do $AC_{uv} = AC_{uv} - q_{st}^k$ for all $< u, v >$ which the path p_{st}^k passes through.
3.3.2 Accept this new connection
else go to step 1;

Fig. 1. Framework for Traffic Planner Algorithm

host(s) work in a work-conserving manner.

C. Run-time Phase

Traffic padding means that some necessary dummy packets are injected into the network in order to make the traffic pattern as camouflaged traffic pattern. The traffic padding process has direct impact on camouflaged traffic pattern. Two decisions need to be made beforehand in the traffic padding process: when the dummy traffic should be generated and how much dummy traffic is needed.

The padding process is illustrated in Figure 2. The result of the padding process is that a composition of padding traffic and payload traffic (including direct and rerouted traffic) is generated. We call the resulting traffic pattern *Real Traffic Pattern* (RTP). The real traffic pattern may not exactly be the same as the camouflaged traffic pattern due to some uncontrollable factors, for example, the periodic timer in the practical operating systems may not be activated periodically due to the CPU scheduling. The objective of the padding process is to make the real traffic pattern to match the camouflaged traffic pattern as closely as possible.

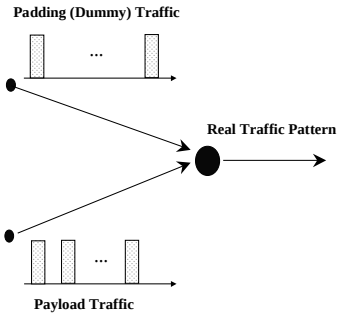


Fig. 2. Traffic Padding Process

C.1 Ideal Padding Algorithm

Figure 3 shows an ideal padding algorithm that generates Constant Bit Rate (CBR) traffic with a rate of α packets per time unit. The traffic pattern is one packet every time period $\frac{1}{\alpha}$. From this pattern, an observer cannot infer any information about the payload traffic pattern except that the payload traffic on each link does not exceed one packet every $\frac{1}{\alpha}$ time units.

Notations:
α: the traffic rate of camouflaged traffic pattern
INPUT: α
1. Every time period $\frac{1}{\alpha}$ Do:
1.1 if there is payload packet
1.2 then send it;
1.3 else send out one dummy packet;

Fig. 3. Ideal Padding Algorithm

C.2 Practical Padding Algorithms

The ideal padding algorithm in Figure 3 assumes that the system has a ideal timer. The timer can be set to a very high precision, and the triggered timer interrupt is served immediately. However, general-purpose operating systems have a limitation on timer period precision (say, 10 milliseconds in Windows NT and Solaris). This makes the ideal algorithm impractical, and better algorithms must be found. Below we describe two such algorithms which differ on how incoming packets are being handled.

So it is not a practical algorithm and then we develop the following practical padding algorithms.

Non-work-conserving algorithm In this algorithm (see Figure 4), a timer expires every τ time units. Whenever a new packet arrives, it is not sent out immediately, but is held until the next timer expiration. Thus the traffic pattern looks like a fixed number of packets that are transmitted whenever the timer expires. The disadvantage of this algorithm is that it holds the payload packets until the timer expires. This introduces unnecessary delay. In some cases, this delay may not be acceptable for real-time applications.

Work-conserving algorithm In this algorithm (see Figure 5), whenever a new packets comes, it is sent out immediately. Some (necessary dummy) packets are transmitted when the timer expires. With this algorithm, an average of $\alpha\tau$ packets are trans-

Notations:
α: the traffic rate of camouflaged traffic pattern
τ: the smallest system timer period
$|Q|$: Payload queue length
INPUT: α, $|Q|$, and τ
1. Every time period $r = \max(\tau, \frac{1}{\alpha})$ Do:
1.1 if there is payload packet in the queue
1.2 then send
x payload packets, where $x = \min(\alpha r, |Q|)$; and
y dummy packets, where $y = \max(0, \alpha r - x)$;
1.3 else send out αr dummy packets;

Fig. 4. Non-work-conserving Padding Algorithm

mitted every time period τ. In a particular timer period, however, the exact traffic amount being transmitted may not always be $\alpha\tau$ due to randomness of packet arrivals. That is, there may be an difference between the real traffic pattern and the targeted camouflaged traffic pattern. We will further analyze this in Section VI.

Notations:
α: the traffic rate of camouflaged traffic pattern
τ: the smallest system timer period
x: the number of payload packet sent out during the last time interval
INPUT: α, x, and τ
1. Whenever a payload packet arrives, it is sent out immediately and $x{+}{+}$;
2. Every time period $r = \max(\tau, \frac{1}{\alpha})$ Do:
send y dummy packets, where $y = \max(0, \alpha r - x)$;
reset $x = \max(0, x - \alpha\tau)$;

Fig. 5. Work-conserving Padding Algorithm

V. Implementation of NetCamo System

We have implemented the above proposed methodologies in a protype system called *NetCamo* on a Windows NT platform. In the following, we will first give an overview of the architecture of NetCamo, and then shortly elabrate on the implementation.

A. Architecture

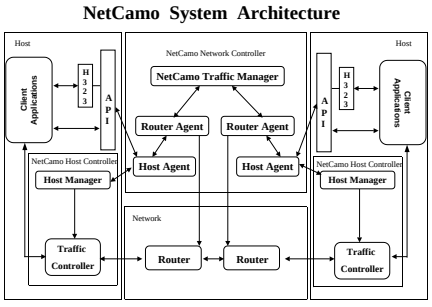


Fig. 6. Architecture of NetCamo

An overview of the architecture of a NetCamo system is illustrated in Figure 6. NetCamo system consists of two major components: the *NetCamo Network Controller* and the *NetCamo Host Controller*.

The NetCamo Network Controller contains the *NetCamo Traffic Manager*, the *Router Agent* and the *Host Agent*. The NetCamo Traffic Manager is responsible for delay calculation, real-time connection admission control and traffic planning. It generates a plan that specifies if and how the payload packets should be rerouted according to the connection's traffic specification and QoS requirement. The Router Agent controls and monitors the router in the network. The Host Agent accepts the connection admission and tear-down requests from the applications and forwards them to the NetCamo Traffic Manager. It also forwards the traffic plan information from the NetCamo Traffic Manager to the NetCamo Host Controller for policing and rerouting of the traffic.

NetCamo Host Controller resides on each host and contains two modules: the *Host Manager* and the *Traffic Controller*. The Traffic Controller is implemented as a kernel-mode driver for rerouting packets, sending necessary dummy packets, and forwarding rerouted packets from other hosts. The Host Manager forwards the traffic plan information from NetCamo Network Controller to the traffic controller and controls/monitors the activity of the traffic controller on each host.

Applications can call the *NetCamo API* to setup or teardown a connection. The connection admission or release request is submitted to a Host Agent and then to the NetCamo Traffic Manager for admission control. The NetCamo Traffic Manager notifies all the related the NetCamo Host Controllers about the new connection if this connection can be accepted and replies the application with accept or reject information. Applications that are H.323 [34] compliant (e.g., Microsoft NetMeeting) can submit the admission/teardown request via the H.323 Gatekeeper. The advantage is that our NetCamo implementation can be compatible with the application. It is very easy to be deployed. We call this upward-compatibility.

B. Implementation Details

As mentioned above, NetCamo consists of two components: *NetCamo Network Controller* and *NetCamo Host Controller*. Furthermore, these two components are composed of several modules respectively: NetCamo Network Controller contains three modules *NetCamo Traffic Manager*, *Router Agent* and *Host Agent*, and NetCamo Host Controller contains two modules: *Host Manager* and *Traffic Controller*. Among these modoles, Traffic Controller plays a key role in the whole system. We take it as the example to introduce some details of NetCamo system implementation.

In principle, the traffic controller can be implemented in several different layers, such as data link, network, transport, or even application layer. We chose to implement it as a NDIS intermediate driver in Windows NT [23]. It is within the lower sub-layer of the network layer, just above the device driver layer, below the IP layer in the context of the TCP/IP protocol suite (Figure 7). An implementation at this layer performs better than realizations at higher layers and is easier to implement than at the data link layer. By this, downward compatibility can be achieved in the sense that no modifications to network drivers are needed.

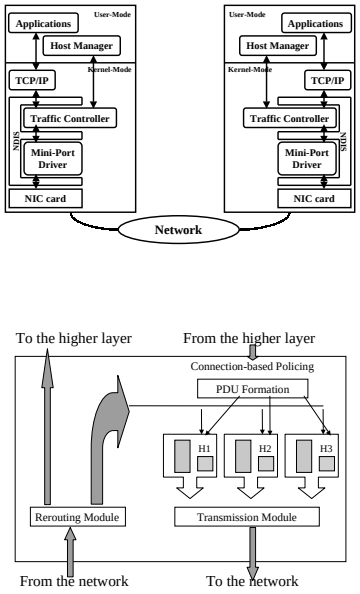


Fig. 8. Internal Architecture of Traffic Controller

Figure 8 also shows the internal architecture of the traffic controller. Whenever a packet arrives from the network, the traffic controller checks if the local host is this packet's true destination. If yes, the packet is forwarded to the higher layer in the protocol stack. If the packet is a dummy packet, it simply gets dropped. If the packet is intended to be rerouted, it is forwarded to the packet sending queue, where it gets sent back into the network. The traffic controller maintains two queues for each host (except itself) in the network (in Figure 8, H1, H2, and H3 are the other three hosts): the *packet sending queue* and the *dummy packet queue*. The transmission module acts as a static priority scheduler, in which the dummy packet queue has a lower priority, and the packet sending queue has a higher priority. When a packet comes from the higher layer, it is put into the packet sending queue of the traffic controller. Finally, the packets in the packet sending queue are sent to the network. In the meantime, dummy packets are injected to the network as needed. In this fashion, packets to every other host are sent out according to the rate defined in the camouflaged traffic rate matrix.

VI. Evaluation

In this section, we evaluate the performance of NetCamo. As discussed, the objective of NetCamo is to prevent traffic analysis while providing delay-guaranteed services. We use the following two performance metrics to evaluate NetCamo in terms of quality of the padding and in terms of utilization of available resources:

1. Padding Error Ratio (PER): This metric gives an indication about the quality of the padding by measuring the relative error between the real traffic pattern and the desired camouflaged traffic pattern. The smaller the PER is, the better the real traffic pattern matches the camouflaged traffic pattern.

Given that for the Non-Work-Conserving algorithm, its PER is zero, we focus on the error analysis for the Work-Conserving algorithm.

2. Admission Probability (AP): AP is defined as the probability of a connection being admitted. It can be estimated by

$$AP = \frac{C_a}{C_r}. \tag{18}$$

where C_a is the number of connections admitted and C_r the number of connections requested for admission in a session. The larger AP, the more QoS-guaranteed connections admitted and the better the system performance.

TABLE I
Notations

Term	Description
T	Observation period used by the observer.
τ	time-out value for the periodic timer.
α	the camouflaged traffic rate.
λ	Rate parameter of Poisson Traffic.

Table I is a list of notations we will use in the following analysis.

A. Padding Error Evaluation

The quality of a padding can be measured by comparing the real traffic pattern to the desired camouflaged traffic pattern. We compare these two patterns using the Padding Error Ratio (PER). The PER represents the relative error between the real traffic pattern and the desired camouflaged traffic pattern. The comparison is done over an observation period. Let T be the length of the observation period.

Let $Z(t-T,t)$ denote the difference between the real traffic and the desired camouflaged traffic pattern during the time interval from $t-T$ to (not including) t. Then we define the padding error ratio as:

$$PER = \frac{\sqrt{E[Z^2(t-T,t)]}}{T\alpha}, \tag{19}$$

where $T\alpha$ is the desired camouflaged traffic in time interval $[t-T,t)$.

In the following, we will derive a formula for PER and then validate this formula with a series of experiments.

A.0.a Theoretical Analysis. PER measures the quality of padding, i.e., how well the real traffic pattern matches the desired camouflaged traffic pattern. Quality of padding can be measured by the relative error between the real traffic pattern and the desired camouflaged traffic pattern. T is the observation period used by the observer.

In the theoretical analysis, we have the following assumptions:

1. Traffic padding algorithm uses an arbitrarily precise periodic timer.
2. For convenience of explanation, we assume that T is integral multiple of τ. It is easy to extend the following result to the general case.
3. The observation window starts in-phase with a timer period. In other words, the observation period starts when a timer expires, and ends when a timer expires as well.

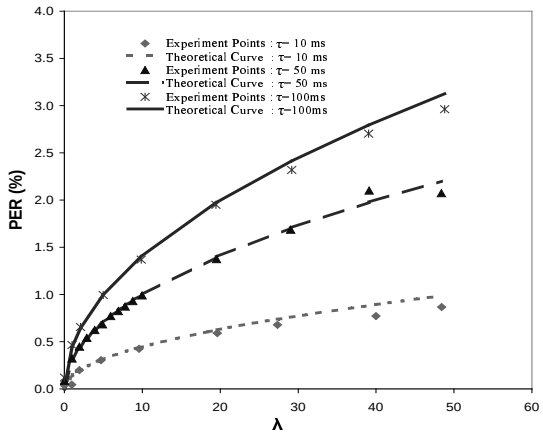


Fig. 9. Error analysis: $\alpha = 100 Packets/s, T = 1$ second

The observation window starts and ends at the time instants when timer is activated. In this way, the essence of padding error can be easily explored.

4. Without loss of generality, we assume that the arrival traffic follows a Poisson process. Although this model of traffic may not be accurate, it's sufficient for us to evaluate the padding algorithm.

These assumptions simplify our analysis. With these assumptions, we can derive Theorem 1.

Theorem 1: Under the assumptions above, the padding error ratio is

$$PER = \frac{\sqrt{2\lambda\tau}}{T\alpha}. \tag{20}$$

Detailed proof of theorem 1 can be found in Appendix.

A.0.b Experimental Results. The assumptions made in order to prove Theorem 1 may not hold in a real system. For example, in the practical systems, such as Solaris or Windows NT operating system, the periodic timer may not be exactly periodic. Our experimenal purpose is to verify whether the theoretical result is robust or not when some of these assumtions are violated.

Figure 9 shows both the theoretical value and the measures for PER as a function of traffic rate λ for three different timer periods. In the figure, the three curves show the theoretical results, while three sets of points show the experimental results at 3 different timer periods.

From figure 9, we can make the following observations:

1. The experimental results are very close to the corresponding theoretical values. This illustrates that the formula for PER in Theorem 1 is robust; that is, it can be used to precisely predict the padding error even though some assumptions are not true.
2. For a specific timer period, the greater λ, the greater PER. For Poisson traffic, λ reflects the magnitude of the variance. Thus, the difficulty of camouflaging increases as the traffic variance increases. This is consistent with intuition.
3. The greater the timer period, the greater PER. The reason is that the variance of the amount of payload traffic is larger with the timer period is larger.
4. We can see from Figure 9 that when the period of the periodical timer is 10ms, PER is less than 1%. In many practical operating systems, a timer with period of 10 millisecond is readily available. Thus, NetCamo can generate the real traffic pattern which is very close to the targeted camouflaged traffic pattern in the sense that the PER is very small.

B. Admission Probability Evaluation

In order to evaluate the amount by which camouflaging affects the capability to support QoS sensitive applications, we compare the admission probability of systems with and without camouflaging. We run a suite of experiments on three systems all of which provide QoS guarantees: (1) *Baseline_A* represents a system without traffic camouflaging. (2) *Baseline_B* represents a system that uses only traffic padding, with no rerouting in case of resource shortage. (3) *NetCamo* represents a system with both host-based rerouting and padding. All three systems employ our real-time delay analysis technology. In our evaluation, the first two systems will be used as the baseline systems to the third system, i.e. NetCamo.

We evaluate the above system in a *star* network with one router and four hosts. All hosts are connected to the router by links with the same capacity of 100 Mbps.

We evaluate the admission control behavior in a system where requests for connection establishment form a Poisson process with rate λ, while connection lifetimes are exponentially distributed with an average lifetime of 10 seconds for each flow. Source and destination hosts of connections are chosen randomly. All generated connections have a fixed burst size of 12,000 bits, and a flow rate of 360000bps. The end-to-end delay requirement of all connections is fixed at 20ms.

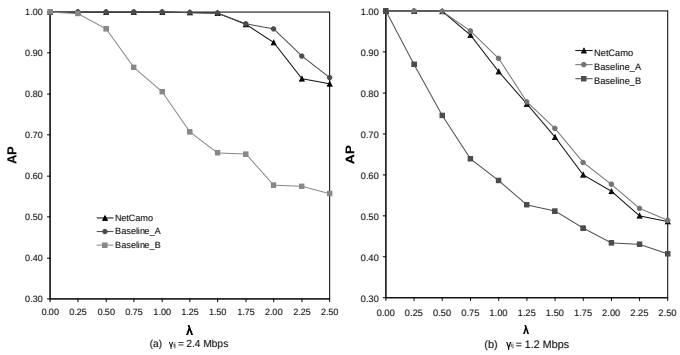


Fig. 10. Admission Probability Comparison of NetCamo, Baseline_A and Baseline_B

B.0.c Numerical Results and Observations. Figure 10 shows the admission probabilities in the three systems as a function of arrival rates. Figure 10 (a) shows the results of three systems in the experiment run with the camouflaged traffic pattern in which a bandwidth of 2.4 Mbps is reserved as camouflaged traffic pattern between any two hosts (i.e., the elements in the camouflaged traffic pattern matrix are all 2.4 Mbps). Similarly, Figure 10 (b) illustrates the results of three systems with a camouflaged traffic pattern of 1.2 Mbps between any two hosts. From these results, we can make the following observations:

1. As it should be, in systems with larger bandwidth in the camouflaged traffic pattern, more connections can be admitted, hence we get higher admission probabilities.
2. We found that Baseline_A system can always achieve the highest admission probabilities among the three systems. The reason is that without limitation of camouflaged traffice pattern,

in System Baseline_A, as along as there are enough link capacities along the path of the coming flow, the flow will be always admitted.

3. NetCamo can achieve higher admission probabilites than System Baseline_B. In most cases, it achieves a performance similar to System Baseline_A. For example, in Figure 10 (b), when λ is 1.0, the admission probability of NetCamo is 0.85 while the AP of System Baseline_B is 0.58. These observations can be explained by the fact that rerouting approach provides more flexibility to System NetCamo than System Baseline_B in selecting path for the coming connection.

VII. Summary

In this paper, we have discussed the methodologies used in our NetCamo system that is able to prevent traffic analysis and guarantee message deadlines. We take an integrated approach and realize it in three system phases:

- At the system configuration phase, correct camouflaged traffic parameters are derived in accordance to various constraints of the system. Delays along individual communication links are also analyzed.
- At the admission control phase, a traffic plan is generated for the newly arrived connection. The traffic plan specifies how to distribute the traffic of the connection into several routers so that both camouflaging and delay requirements are met.
- At run-time phase, traffic control is applied at a low layer by sending additional dummy traffic on communication paths to compensate for fluctuation of traffic at connection level so that the real traffic is consistent with desired camouflaged traffic pattern.

As mentioned earlier, our work is the first that addresses both problems of preventing traffic analysis and guaranteeing worst case delays in an integrated manner. Our NetCamo is the first *implemented* system that can offer this kind of secured real-time communication services in an open IP network. Comprehensive performance evaluation has been carried out. The performance data we collected demonstrates that our NetCamo system is effective and efficient. In NetCamo, the error between targeted time invariant traffic and real camouflaged traffic can be as small as one percent, and at the same time the probability to guarantee a real-time connection is still relatively high. These facts indicate that with our innovative camouflage and traffic planning technologies, the NetCamo system can effectively camouflage the network traffic without compromising real-time capability. The reader is referred to http://netcamo.cs.tamu.edu for the newest development on our NetCamo system.

Given the initial success of the NetCamo system, we are currently investigating several extensions. We are considering to let the network have multiple modes of camouflaged traffic patterns. We believe this will allow the network to mislead an intruder more effectively. For example, we may like to show a "peace time" status of the network while an alert has been set. Meanwhile, when the system is really in peace time, we may want it to appear to be in an alert status. Another extension we are working on is to decentralize NetCamo network controller in order to improve its efficiency when working in a large distributed environment. Finally, as we mentioned earlier, NetCamo effectively camouflages payload traffic pattern while traditional encryption technology camouflages the payload content. There are many other components that may need to be camouflaged in a communication system, including location and type of hosts, network topology, etc. We are studying various issues related to camouflage of these and other components that may be of interest to an intruder.

TABLE II
Notations used in proving Theorem 1

Notation	Description
$\mathbf{T}$	Observation period used by the observer
τ	The period of timer
α	The camouflaged traffic rate
λ	Rate of Poisson traffic
$\mathbf{P(t)}$	The total number of payload packets sent in time interval [0,t)
$\mathbf{D(t)}$	The total number of dummy packets sent in time interval [0,t)
$\mathbf{Z(t-\delta, t)}$	The difference between the real traffic and the camouflaged traffic in time interval $[t-\delta, t)$, where δ is any non-negative real number
$\mathbf{X(t-\delta, t)}$	The number of payload packets (coming from the high layer) sent in time interval $[t-\delta, t)$, that is, $X(t-\delta, t) = P(t) - P(t-\delta)$
$\mathbf{Y(t-\delta, t)}$	The number of dummy packets sent in time interval $[t-\delta, t)$, that is, $Y(t-\delta, t) = D(t) - D(t-\delta)$

Appendix

In this appendix, we prove Theorem 1 and derive a formula for the padding error ratio PER.

Table II gives a list of notations we will use here.

By using the definitions for $X(t-\delta, t), Y(t-\delta, t)$ and $Z(t-\delta, t)$, we can formulate the difference between the real traffic and the camouflaged traffic during an observation period as follows:

$$Z(t-T, t) = \alpha T - (X(t-T, t) + Y(t-T, t)). \quad (21)$$

Recalling the definition of PER in 19,

$$PER = \frac{\sqrt{E[Z^2(t-T, t)]}}{T\alpha},$$

we need the following three lemmas before proving Theorem 1.

Lemma 1:

$$\begin{aligned} \alpha n\tau &= Y(t - K_s\tau - (n-1)\tau, t - K_s\tau + \tau) \\ &\quad + X(t - K_s\tau - n\tau, t - K_s\tau), \end{aligned} \quad (22)$$

where both K_s and n are integers and $n > 0$.

Proof: According to the padding algorithm, at time instant t, dummy packets are sent out to pad the real traffic sent during $[t-\tau, t)$, so the payload traffic between $[t-\tau, t)$ and padding traffic between $[t, t+\tau)$ is equal to the camouflaged traffic during the interval of τ.

$$\alpha\tau = Y(t, t+\tau) + X(t-\tau, t). \quad (23)$$

In general, dummy packets during $[t-k\tau, t-k\tau+\tau)$ are sent to pad the payload traffic sent during $[t-k\tau-\tau, t-k\tau)$. So

$$\alpha\tau = Y(t-k\tau, t-k\tau+\tau) + X(t-k\tau-\tau, t-k\tau). \quad (24)$$

Substituting $k = K_s, K_s+1, \cdots, K_s+(n-1)$ into (24), we have

$$\begin{aligned}
\alpha\tau &= Y(t-K_s\tau, t-K_s\tau+\tau) \\
&\quad + X(t-K_s\tau-\tau, t-K_s\tau), && (25)\\
\alpha\tau &= Y(t-K_s\tau-\tau, t-K_s\tau) \\
&\quad + X(t-K_s\tau-2\tau, t-K_s\tau-\tau), && (26)\\
\alpha\tau &= Y(t-K_s\tau-2\tau, t-K_s\tau-\tau) \\
&\quad + X(t-K_s\tau-3\tau, t-K_s\tau-2\tau), && (27)\\
&\vdots \\
\alpha\tau &= Y(t-K_s\tau-(n-1)\tau, t-K_s\tau-(n-2)\tau) \\
&\quad + X(t-K_s\tau-n\tau, t-K_s\tau-(n-1)\tau). && (28)
\end{aligned}$$

Adding both sides of (25), $\cdots$, (28), we have

$$\begin{aligned}
\alpha n\tau &= \sum_{i=0}^{n-1} Y(t-K_s\tau-i\tau, t-K_s\tau-(i-1)\tau) + \\
&\quad \sum_{i=0}^{n-1} X(t-K_s\tau-(i+1)\tau, t-K_s\tau-i\tau). \quad (29)
\end{aligned}$$

From the definitions of $\mathbf{X}(\mathbf{t}-\delta, \mathbf{t})$ and $\mathbf{Y}(\mathbf{t}-\delta, \mathbf{t})$, it is obvious

$$\begin{aligned}
&\sum_{i=0}^{n-1} Y(t-K_s\tau-i\tau, t-K_s\tau-(i-1)\tau) = \\
&\qquad Y(t-K_s\tau-(n-1)\tau, t-K_s\tau+\tau), \quad (30)
\end{aligned}$$

and

$$\begin{aligned}
&\sum_{i=0}^{n-1} X(t-K_s\tau-(i+1)\tau, t-K_s\tau-i\tau) = \\
&\qquad X(t-K_s\tau-n\tau, t-K_s\tau). \quad (31)
\end{aligned}$$

Substituting (30) and (31) into (29), we have

$$\begin{aligned}
\alpha n\tau &= Y(t-K_s\tau-(n-1)\tau, t-K_s\tau+\tau) \\
&\quad + X(t-K_s\tau-n\tau, t-K_s\tau).
\end{aligned}$$

■

Lemma 2:

$$Z(t-T, t) = X(t-T-\tau, t-T) - X(t-\tau, t). \quad (32)$$

Proof: By (21), $Z(t-T,t) = \alpha T - X(t-T,t) - Y(t-T,t)$. Recall that we assume T is integral multiple of τ.

By definition of $X(t-T,t)$ and $Y(t-T,t)$, we have

$$\begin{aligned}
X(t-T,t) &= P(t) - P(t-T) \\
&= P(t) - P(t-\tau) \\
&\quad + P(t-\tau) - P(t-T), && (33)\\
Y(t-T,t) &= D(t) - D(t-T) \\
&= D(t-T+\tau) - D(t-T) \\
&\quad + D(t) - D(t-T+\tau). && (34)
\end{aligned}$$

Substituting $T = N\tau$, (33), and (34) into (21), we have

$$\begin{aligned}
Z(t-T,t) &= \alpha\tau + \alpha(N-1)\tau \\
&\quad - [P(t) - P(t-\tau) + P(t-\tau) - P(t-T)] \\
&\quad - [D(t-T+\tau) - D(t-T) \\
&\quad + D(t) - D(t-T+\tau)]. \quad (35)
\end{aligned}$$

By algebraic manipulation of (35), we have

$$\begin{aligned}
Z(t-T,t) &= \alpha\tau - [D(t-T+\tau) - D(t-T)] \\
&\quad - [P(t) - P(t-\tau)] + \alpha(N-1)\tau \\
&\quad - [P(t-\tau) - P(t-T) + D(t) \\
&\quad - D(t-T+\tau)] \\
&= \alpha\tau - Y(t-T, t-T+\tau) - X(t-\tau, t) \\
&\quad + \alpha(N-1)\tau - [X(t-T, t-\tau) \\
&\quad + Y(t-T+\tau, t)]. \quad (36)
\end{aligned}$$

Substituting $K_s = 1, n = N-1$ into (22), we have

$$X(t-T, t-\tau) + Y(t-T+\tau, t) = \alpha(N-1)\tau. \quad (37)$$

Substituting $k = N$ into (23), we have

$$\alpha\tau = Y(t-N\tau, t-N\tau+\tau) + X(t-N\tau-\tau, t-N\tau). \quad (38)$$

Because $T = N\tau$, then

$$\alpha\tau = Y(t-T, t-T+\tau) + X(t-T-\tau, t-T). \quad (39)$$

By algebraic manipulation of (39), we have

$$\alpha\tau - Y(t-T, t-T+\tau) = X(t-T-\tau, t-T). \quad (40)$$

Substitute (37) and (40) into (36), we have

$$Z(t-T,t) = X(t-T-\tau, t-T) - X(t-\tau, t).$$

■

Following Lemma 2, we can derive next lemma.

Lemma 3: The second-order moment of the difference between camouflaged traffic and real traffic is given by

$$E[Z^2(t-T,t)] = 2\lambda\tau. \quad (41)$$

Proof: By Lemma 2, $E[Z(t-T,t)] = X(t-T-\tau, t-T) - X(t-\tau, t)$. According to the definition of Poisson process and $X(t-\delta, t)$, $X(t-\delta, t)$ also yields obedience to Poisson distribution. Moreover, because there is no overlap between interval $[t-T-\tau, t-T)$ and $[t-\tau, t)$, $X(t-T-\tau, t-T)$ and $X(t-\tau, t)$ are independent from each other. Furthermore,

$$\begin{aligned}
E(X(t-T-\tau, t-T)) &= \lambda\tau, && (42)\\
E(X(t-\tau, t)) &= \lambda\tau, && (43)\\
E(X^2(t-T-\tau, t-T)) &= \lambda\tau + \lambda^2\tau^2, && (44)\\
E(X^2(t-\tau, t)) &= \lambda\tau + \lambda^2\tau^2, && (45)
\end{aligned}$$

Following the definition of $X(t-\delta, t)$, $Y(t-\delta, t)$ and $Z(t-\delta, t)$, we have

$$\begin{aligned}
&E[Z^2(t-T,t)] = \\
&\quad E((X^2(t-T-\tau, t-T)) + E(X^2(t-\tau, t)) \\
&\quad - 2E(X(t-T-\tau, t-T))\, E(X(t-\tau, t)). \quad (46)
\end{aligned}$$

Now, substituting (42), (43), (44), and (45) into (46), we have

$$E[Z^2(t-T,t)] = 2\lambda\tau.$$

■

Substituting (41) into (19), we have Theorem 1 proved.

Theorem 1

$$PER = \frac{\sqrt{2\lambda\tau}}{\alpha T}.$$

REFERENCES

[1] The Anonymizer, http://www.anonymizer.com/

[2] O. Berthold, H. Federrath, and M. Köhntopp, "Project Anonymity and Unobservability in the Internet," Proceedings of the tenth conference on Computers, freedom privacy: challenging the assumptions April 4 - 7, 2000, Toronto, ON Canada

[3] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for differentiated service," RFC 2474, Dec. 1998

[4] R. Braden, D. Clark, and S. Shenker, "Integrated services in the Internet architecture," RFC 1633, Jun. 1991

[5] A. Charny and J.-Y. Le Boudec, "Delay Bounds in a Network with Aggregate Scheduling," in Proceedings of QOFIS, Berlin, October 2000

[6] D. Chaum, "Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms," Communications of the ACM, v.24, n.2, pp.84-88, 1981

[7] B. Choi, D. Xuan, C. Li, R. Bettati and W. Zhao, "Scalable QoS Guaranteed Communication Services for Real-Time Applications," Proceedings of ICDCS'00, 2000

[8] J. Claessens, B. Preneel, J. Vandewalle," Solutions for anonymous communication on the Internet," in Proceedings. IEEE 33rd Annual 1999 International Carnahan Conference on Security Technology, Page(s): 298 -303, 1999

[9] R.L. Cruz, "A calculus for network delay," IEEE Transaction on Information Theory, 37(1):114-131, Jan, 1991

[10] R.L. Cruz, "A calculus for network delay, Part II: Network Analysis," IEEE Transaction on Information Theory,37(1):132-141, Jan, 1991

[11] R.L. Cruz, "Quality of service guarantees in virtual circuit switched networks," IEEE select. Areas Commun., 13(6):1048-1056, Aug. 1995.

[12] B. Devalla, A. Sahoo, Y. Guan, C. Li, R. Bettati and W. Zhao, "Adaptive Connection Admission Control for Mission Critical Real-Time Communication Networks," Proceedings of MilCom'98, Vol. 2, pp. 614-620, 1998

[13] D. Goldschlag, M. Reed, and P. Syverson, "Onion Routing for Anonymous and Private Internet Connections, " Communications of the ACM, v.42, n.2, pp.39-41, 1999

[14] P. Goyal, S.S. Lam, and H. Vin, "Determining end-to-end delay bounds in heterogeneous networks," Proceedings of 5th Int. Workshop on network and and op. sys. support for digital audio and video, pp. 287-298, 1995

[15] Y.Guan, C.Li, D.Xuan, R.Bettati, and W.Zhao, "Preventing traffic analysis for real-time communication networks," Proceedings of IEEE MILCOM1999, vol. 1, pp. 744-750, Nov. 1999

[16] Y.Guan, X. Fu, R.Bettati, and W.Zhao, "Efficient Traffic Camouflaging in Mission-critical QoS-guaranteed Networks," Proceedings of IEEE Information Assurance and Security Workshop, pp. 143-149, West Point, June 2000

[17] E. Gabber, P.B.Gibbons, Y.Matias, and A.Mayer, "How to make personalized web browing simple, secure, and anonymous," Proceedings of Financial Cryptography'97-LNCS 1318. Springer-Verlag, 1997

[18] C. Gülcü and G. Tsudik, "Mixing Email with Babel, " Proceedings of the 1996 Symposium on Network and Distributed System Security, 1996

[19] IMSL, "IMSL MATH/LIBRARY Fortran Subroutines for Mathematical Applications User's Manual," IMSL Inc., 1989

[20] A. Jerichow, J. Müller, A. Pfitamann, B. Pfitzmann, and M. Waidner, "Real-Time Mixes: A Bandwidth-Efficient Anonymity Protocol, " IEEE Journal on Selected Areas in Communications, v.16, n.4, pp.495-509, 1998

[21] S. Kent and R. Atkinson, "Security Architecture for the Internet Protocol,", RFC2401, 1998

[22] C. Li, A. Raha and W. Zhao, "Stability in ATM Networks," Proceedings of IEEE INFOCOM, vol.1, pp. 160-167, 1997.

[23] Microsoft, "Windows NT DDK Documentation," MSDN online library, http://msdn.microsoft.com/library/default.htm.

[24] NAG, "The NAG Fortran Library Introductory Guide, Mark 16," NAG Ltd., 1993

[25] R. E. Newman-Wolfe, B. R. Venkatraman, "High Level Prevention of Traffic Analysis," Seventh Annual Computer Security and Applications Conference, San Antonio, Texas, Dec 2-6, pp. 102-109, 1991

[26] R. E. Newman-Wolfe, B. R. Venkatraman, "Performance Analysis of a Method for High Level Prevention of Traffic Analysis," 8th Annual Computer Security and Applications Conference, San Antonio, Texas, pp. 123-130, Nov 30-Dec 4, 1992

[27] A.K.J.Parekh and R.G.Gallager, "A generalized processor sharing approach to flow control in integrated services networks: the single-node case," IEEE/ACM Transaction on Networking, vol. 1, no.3, pp. 344-357, 1994

[28] A.K.J.Parekh and R.G.Gallager, "A generalized processor sharing approach to flow control in integrated services networks: the multiple-node case," IEEE/ACM Transaction on Networking, vol. 2, no.2, pp. 137-150, 1994

[29] A. Raha, S. Kamat, W. Zhao, "Guaranteeing End-to-End Deadlines in ATM Networks," Proc. Of International Conference on Distributed Computing System, pp. 60-68, 1995

[30] M. Reed, P. Syverson, and D. Goldschlag, "Anonymous Connections and Onion Routing, " IEEE Journal on Selected Areas in Communications, v.16, n.4, pp.482-494, 1998

[31] M. K. Reiter and A.D.Rubin, "Crowds: Anonymity for Web Transactions, " ACM Transactions on Information and System Security, v.1, n.1, pp.66-92, 1998

[32] P. Srisuresh, "Security Model with Tunnel-mode IPsec for NAT Domains," RFC2709, 1999

[33] R.C.Summers, "Secure Computing," McGraw-Hill, 1997

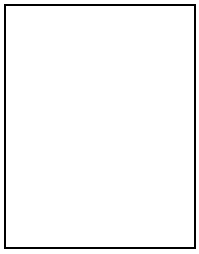
[34] G.A.Thom, "H.323: the multimedia communications standard for local area networks," IEEE Communications Magazine, vol. 34, no. 12, pp. 52 -56, Dec. 1996.

[35] B. R. Venkatraman, R. E. Newman-Wolfe, "Performance Analysis of a Method for High Level Prevention of Traffic Analysis Using Measurements from a Campus Network," Tenth Annual Computer Security and Applications Conference, Orlando, Florida, pp. 288-297, Dec. 1994

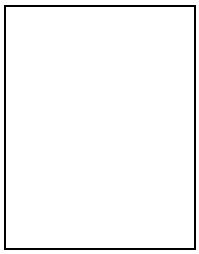
[36] D.Wrege, E.Knightly, H.Zhang, and J.Liebeherr, "Deterministic delay bounds for VBR video in packet-switching networks: Fundamental limits and practical tradeoffs," IEEE/ACM Transaction on Networking, 4(3), pp. 352-362, 1996

[37] Zero-knowledge Systems, http://www.zeroknowledge.com/

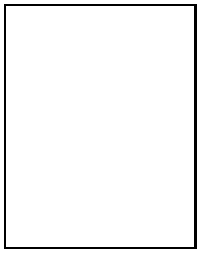
[38] H. Zhang, "Providing End-to-End Performance Guarantees Using Non-Work-Conserving Disciplines," Computer Communications: Special Issue on System Support for Multimedia Computing, 18(10), Oct 1995



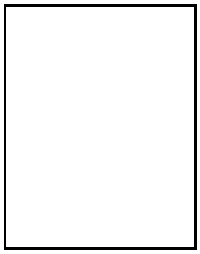
Yong Guan currently is a Ph.D. Candidate in the Department of Computer Science at Texas A&M University. He obtained both his M.S (1996) and B.S (1990) degrees in Computer Science from Peking University, China. His research interests are QoS provision for high-speed networks, Network Security and distributed systems. He is a student member of the IEEE.



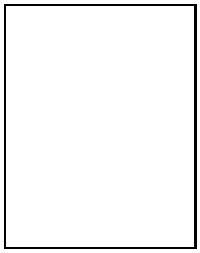
Xinwen Fu currently is a Ph.D. student in the Department of Computer Science at Texas A&M University. He obtained his M.S (1998) and B.S (1995) degrees in electrical engineering from Graduate School of University of Science and Technology of China and Xian Jiaotong University, China, respectively. His research interests are Network Security and distributed systems.



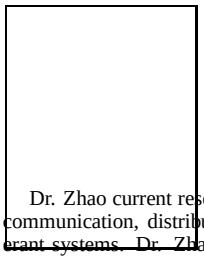
Dong Xuan received his B.S. (1990) and M.S. (1993) degrees in Electronic Engineering from the Shanghai Jiao Tong University, China. He was on faculty of Electronic Engineering at SJTU from 1993 to 1997. In 1997, he worked as a research/teaching assistant in City University of Hong Kong. He is presently a Ph. D student in the Department of Computer Science, Texas A&M University. His main research interests are real-time computing and communication and distributed systems.



Prashanth Umesh Shenoy received his M.S (2000) degree in computer science from Texas A&M University, B.S (1998) degree in Computer Engineering from Mangalore University, India. He is currently working for Cisco corporation.



Riccardo Bettati received his Diploma in Informatics from the Swiss Federal Institute of Technology (ETH), Zuerich, Switzerland, in 1988 and his Ph.D. from the University of Illinois at Urbana-Champaign in 1994. From 1993 to 1995, he held a postdoctoral position at the International Computer Science Institute in Berkeley and at the University of California at Berkeley. Since 1995 he is Assistant Professor in the Department of Computer Science at Texas A&M University. His research interests are in real-time distributed systems, real-time communication, and network support for distributed applications.



Wei Zhao received his B.Sc. of physics from Shanxi Normal University, China, M.Sc. and Ph.D degrees in computer and information science from the University of Massachusetts at Amherst in 1983 and 1986, respectively. In 1990 he joined the Department of Computer Science at Texas A&M University where he is currently a full professor and department head. He is an IEEE fellow.

Dr. Zhao current research interests include secured real-time computing and communication, distributed operating systems, database systems and fault tolerant systems. Dr. Zhao was an Editor of the IEEE Transactions on Computers. He served as guest editors special issues on real-time operating systems for ACM Operating System Review and real-time middle ware for the International Journal of Real-Time Systems. Currently, he is an editor for the International Journal of Real-Time Systems and a member of the executive committee of IEEE Technical Committee of Real-Time Systems. He was the Program and General Chairs of The IEEE Real-Time Technology and Applications Symposia in 1995 and 1996, respectively. He was the Program and General Chairs of the IEEE Real-Time Systems Symposia in 1999 and 2000, respectively. In 2001, Dr. Zhao will serve as the general chair of IEEE International Conference on Distributed Computing Systems.

Wei Zhao has published over 170 papers in journal, conference and book chapters. He is an inventor of two US patents. Dr. Zhao and his students received the Best Paper Awards in the IEEE International Conference on Distributed Computing Systems and in the IEEE National Aerospace and Electronics Conference.