Preliminary NS-evaluation of Secure Stochastic Routing*

Technical Report

Stephan Bohacek | João Hespanha | Junsoo Lee | Chansook Lim | Katia Obraczka

bohacek@math.usc.edu | hespanha@usc.edu | junsoole@usc.edu | chansool@usc.edu | katia@cse.ucsc.edu

August 6, 2001

Abstract

In this paper we illustrate how the game theoretical tools introduced in [1] can be used to develop secure network-layer routing policies. These policies address security issues such as packet interception and eavesdropping. We utilize `ns-2` simulations to evaluate the routing policies proposed.

1 Introduction

Current routing algorithms are vulnerable because the path over which a data packet travels is fairly predictable and easy to determine. This opens the door for *packet interception* and/or *eavesdropping* by an attacker. Consider, for example the network in Figure 1. Although there are multiple paths between the

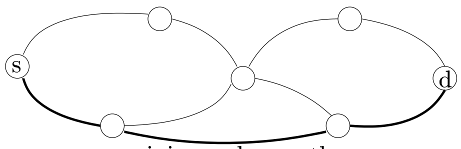

Figure 1: Multi-path network

two nodes marked as source (s) and destination (d), most routers currently use the path that minimize the number of hops. Even if several paths with the same number of hops exist, routing algorithms typically select one of the possible options and utilize that path all the time. Notable exceptions are Equal Cost Multi-Path (ECMP) [2] and OSPF Optimized Multi-Path (OSPF-OMP) [3]. However, these algorithms were developed to increase throughput and not to make routing robust to attacks. Although these algorithms make use of multiple paths (when these are available) they do not introduce unpredictability and therefore packet interception is fairly easy to achieve. Once the path that a packet follows in a data network is easy to predict, attackers that want to intercept packets—or reconstruct a file from packets eavesdropped in the network—can achieve their goals with a minimum number of resources. E.g., by breaking into just one of the routers or tapping into one of the links in the minimum-hop path.

In this paper, we present *routing algorithms that make packet interception and/or eavesdropping maximally difficult.* These algorithms explore the existence of multiple paths between two points of the network

*This research was supported by the Defense Advanced Research Projects Agency. The views presented here are those of the authors and do not represent the views of the funding agencies.

and route the packets in the network to minimize predictability. This forces an attacker to utilize a maximum number of resources. We focus on the development of routing algorithms for the network layer of the protocol stack but these ideas can be extended to other layers of the protocol stack by the use of overlays. In fact, routing can be viewed as a form of overlay that is implemented at the network layer.

In the following paragraphs, we review some of the existing approaches to address packet interception and eavesdropping and their limitations. We then proceed to explain how stochastic routing can be used to mitigate these security issues.

1.1 Packet interception

One of the most obvious attacks to a communication network is to prevent all or most of the data originating from one (or several) nodes o reach their destination. The Internet is particularly vulnerable to this type of attack because the data packets travel along a physical infrastructure that (i) is well known and easily probed (ii) is composed of thousands of elements (physical links, switches, routers, hosts, etc.), each one of them vulnerable to attacks.

The simplest form of packet interception is to physically disable a communication link ("cut-the-cable"). This possibility was at the heart of Internet's original design and this was taken into consideration both at the transport and network layer's of the protocol stack. At the network layer, each router constantly probes its immediate neighbors and, if one of them stops responding, the routing tables are updated to take into account the new topology of the net and find alternative paths to the destination. The transport layer guarantees that any data packets that may have been lost are retransmitted.

A more dangerous type of packet interception is to selectively intercept packets ("cut-the-logical-connection"), e.g., those associated with the communication between two strategic hosts in the network. An attacker could accomplish this by eavesdropping on a physical link and cause collisions with TCP packets, based on their source and destination. To prevent this, routers often exchange decryption keys and encrypt the TCP packets before sending them through a potentially unsafe connection. This prevents an attacker that is eavesdropping on the link to read the TCP header and determine its source and destination. Another alternative is to use dynamically varying IP addresses to prevent the attacker from isolating which packets should be intercepted.

Both the mitigation techniques discussed above for "cut-the-logical-connection" attack are effective against attacks to the physical layer but they are ineffective against an attacker that infiltrated the network layer of a particular router. This is because the network layer must have access to the original TCP header (even if this is encrypted before being sent to lower layers of the protocol stack) and must also know the final destination of the packet (or at least have sufficient information to do the next-hop routing).

The only mechanism to prevent packet interception by an attacker that is able to infiltrate the network and/or the transport layer in some nodes is by exploring the existence of multiple paths between source and destination. Clearly, if an attacker is able to infiltrate at least one router in each possible path then no mitigation effort will succeed. However, this potentially requires many resources from the attackers. Moreover, this possibility is unlikely if a few high-security (but possibly low-throughput) nodes exist in key locations of the network.

The most obvious solution to explore multi-paths is to use (deterministic) flooding—i.e, to send one packet by each possible path—and rely on the transport layer to "filter-out" the duplicate packets. This solution is not practical in most networks because it would greatly increase the traffic. Perhaps, even more importantly, it significantly simplifies packet eavesdropping by an attacker. This issue will be further discussed shortly.

The approach we propose is the exploration of multi-paths through *statistical flooding*. In statistical flooding, packets are sent through all available paths but no packet is sent more than once at the network layer level (although error control at the transport layer may require a packet to be resent through a particular physical link). The flooding occurs "on-the-average" and it is randomized to minimize the probably of interception by an attacker. Note that statistical flooding does not prevent *every packet* from being intercepted. In fact, if one node in the path between source and destination is compromised, some packets will almost certainly be intercepted. However, the transport layer will ensure the retransmission of these packets and statistical flooding guarantees that these packets will eventually go through one of the no compromised paths. Statistical flooding can be achieved by *stochastic routing*, which will be described in detail later.

1.2 Packet eavesdropping

Another type of attack that should be addressed at the network layer is packet eavesdropping. Here, an attacker attempts to reconstruct a file (or other type of application layer object) from TCP packets eavesdropped from the network. Eavesdropping done at the physical layer (e.g., by tapping into the wire) can be made extremely difficult by utilizing router-to-router encryption. Dynamically varying IP addresses are also extremely effective against eavesdropping because the attacker will no longer be able to determine which packets are relevant. However, just as with packet interception, if an attacker is able to infiltrate a router at the network and/or transport layer, these mitigation techniques will not provide adequate protection.

While deterministic flooding is effective to prevent packet interception, it is unacceptable because of the security risks related to eavesdropping. In fact, to minimize the risk of eavesdropping, no more than one packet should be sent from source to destination. Moreover, the path traveled by this packet should be as unpredictable as possible to make eavesdropping maximally difficult. Statistical flooding also provides a solution to the eavesdropping problem since, as long as there exists one path between source and destination that has not been compromised, the probability that the attacker will eavesdrop all the packets can be made equal to zero be judicially selecting a stochastic routing algorithm.

2 Related work

Currently there are two popular approaches to IP layer security: VPNs [4] and IPsec [5]. VPNs, while appropriate for small enclaves, are not scalable to the whole network. IPsec, on the other hand, is useful for peer to peer communications. One drawback of IPsec is that it does not defend against packet replay. Another, perhaps more significant drawback of IPsec is that secret keys must remain secret. Thus, IPsec will not provide any security from an attacker with inside information. For example, if an attacker has the private key and if the attacker is able to intercept packets, then the entire message can be monitored. This paper develops techniques that can be used to defend against such attacks. The essence of the defense is to make the interception of packets difficult. For example, if the attacker is not able to intercept all packets, then a session key can be exchanged over the course of many packets. If this session key is not intercepted, then the attacker cannot decipher any of the message.

One approach to making the intersection of packets difficult is known as Onion routing [6]. Onion routing provides many layers of encryption. Each layer is used to encrypt the transmission between routers. Because of the many layers of encryption, the routers are unable to decrypt the data or even the destination address. All that a router can decipher is the next hop information. While Onion routing is very effective for security, it is not very efficient. For example, each connection must be built and torn down, the routers must encode and decode packets and memory intensive source routing is used. An alternative is to send the date packet along different paths. In this way, the attacker can only ensure that all packets are intercepted if she is monitoring all the paths used. The problem then is to determine which paths should be used and how much data should be sent through each path.

3 Technical Approach—Stochastic routing

As mentioned above most routers in the Internet utilize a minimum-hop criterion to determine their routing policies. The resulting routing is deterministic in that all packets from one source to a destination follow the same path (unless there are topological changes in the network caused, e.g., by the disconnection of a router).

In each (deterministic) router, at the heart of the network layer there exists a next-hop routing table that associates each possible destination IP address with the address and physical interface of a neighboring router that is "one-step closer" to the destination or, in case the final destination is in the local subnet, the address and physical interface of the host.

Stochastic routing utilizes a distinct concept of routing table. *Stochastic routing tables*, map each possible destination IP address with a probability distribution over all possible next-hop addresses. From an end-to-end perspective, this type of routing results in data packets following random paths. The main challenges in stochastic routing is the determination of routing tables that ensure

1. *Delivery*, i.e., all packets will reach their desired destination with probability one
2. *Timeliness*, i.e., the delay is acceptable
3. *Statistical flooding* is achieved, i.e., all paths will be explored

With respect to 3, one should add that when different nodes/links in the network have different characteristics in terms of throughput, security, cost, etc., it might be desirable to have non-uniform statistical flooding.

In the following, we briefly provide some technical background on the computation of stochastic routing tables that satisfy the properties 1-3 mentioned above. The theoretical derivations can be found in [1].

3.1 Computation of stochastic routing tables

In [1], several routing games are proposed to compute stochastic routing that guarantee delivery, timeliness, and stochastic flooding. The key technical insight is to formalize the stochastic routing problem as an abstract game between two players: the designer of the routing algorithm and an attacker that attempts to intersect packets. We consider zero-sum games in which the designer wants to minimize the time it takes for a packet to be safely transmitted, and the attacker wants to maximize this time. To accomplish this, the adversary attempts to intercept the packet at particular links (or nodes) in the network. It is well known from the game theory literature that the solution to such games requires the use of mixed (randomized) policies. In practice, the mixed solution of the minimax problem provides the probability distributions needed for the stochastic routing tables. By formalizing the problem as an optimization with a time cost, we achieve both delivery and timeliness. Note that non-delivery would yield an infinite cost with positive probability and therefore a non-optimal solution. Statistical flooding is a consequence of the saddle solution.

In this paper, we consider *off-line routing games* [1] in which the adversary selects which link to be scanned before routing starts, but the player responsible for designing the routing policy does not know which link was selected. The cost to be minimized by the designer of the routing algorithm and maximized by the attacker is defined to be

$$J(\epsilon) := \mathrm{E}[\chi(\epsilon)], \tag{1}$$

where $\chi(\epsilon)$, $\epsilon \geq 0$, is a random variable that is equal to $(1+\epsilon)^{t-1}$ if the packet is intercepted at the tth hop and 0 otherwise. For $\epsilon = 0$, the random variable $\chi(\epsilon)$ is equal to 1 if the packet is intercepted and 0 otherwise and therefore $J(0)$ is simply the probability that the packet will be intercepted. The cost $J(0)$ assumes that all paths are equal and therefore all that matters is to make sure that the packet reaches its destination. However, for $\epsilon > 0$, $J(\epsilon)$ bias the solution sought by the player that designs the routing policy towards shorter paths since, when being caught is inevitable, it incurs in less cost if it is caught sooner than later. In fact, as $\epsilon \to \infty$, the burden of an extra hop is so large that the optimal solution will minimize the number of hops. We should emphasize that the abstract game described above is being used as an optimization tool to compute stochastic routing tables that guarantee delivery, timeliness, and stochastic flooding. Clearly, as formulated above, the game is too simplistic to model a realistic attack.

We consider a data transmission network with nodes $\mathcal{N} := \{1, 2, \ldots, n\}$ connected by unidirectional links. We denote by $\mathcal{L}$ the set of all links and use the notation $\vec{ji}$ to represent a link from node j to node i. Without loss of generality, we take the source and destinations nodes to be 1 and n, respectively. Because of this we do not need to consider links coming out of node n and for simplicity we will assume that no link in $\mathcal{L}$ exits node n. We consider here *stochastic routing policies.* Under such policies, whenever a packet arrives at node $k \in \mathcal{N}$, it will be routed through link $\vec{ki} \in \mathcal{L}$ with probability $r_{\vec{ki}} \geq 0$. As far as the routing is concerned, each routing policy is characterized by the list $R := \{r_\ell : \ell \in \mathcal{L}\}$, where the r_ℓ satisfy

$$\sum_{k:\vec{ik}\in\mathcal{L}} r_{\vec{ik}} = 1, \qquad \forall i \in \mathcal{N}.$$

In the sequel we denote by $\mathcal{R}_{\text{sto}}$ to the set of lists R with the above property. Here, we restrict our attention to *stochastic cycle-free routing policies.* These are stochastic routing policies for which $R := \{r_\ell : \ell \in \mathcal{L}\}$ is chosen so that a packet will not return to a node where it has been before with probability one. Formally, this means that there is no sequence of links $\mathcal{S} := \{\vec{i_1 i_2}, \vec{i_2 i_3}, \dots, \vec{i_{k-1} i_k}, \vec{i_k i_1}\} \subset \mathcal{L}$, with $r_\ell > 0$ for all $\ell \in \mathcal{S}$. We denote by $\mathcal{R}_{\text{nocycle}}$ the set of lists $R \in \mathcal{R}_{\text{sto}}$ with the above property. The distribution $\{r_{\vec{ik}} \in \mathcal{L} : k \in \mathcal{N}\}$ determines the entries of the stochastic routing table at node i, associated with the path from node 1 to node n. Note that with this type of stochastic routing, the routing table is actually a matrix, as the next-hop routing depends not only on the final destination but also on the source.

In offline games, an attacker stochastic policy $D = \{d_\ell : \sum_\ell d_\ell = 1, \ell \in \mathcal{L}\}$ simply consists of a distribution over the elements of $\mathcal{L}$, where d_ℓ is the probability that the adversary will attempt to intercept packets in the link ℓ. We denote by $\mathcal{D}$ the set of all distributions over $\mathcal{L}$.

Suppose now that, given a fixed routing policies $R := \{r_\ell : \ell \in \mathcal{L}\} \in \mathcal{R}_{\text{nocycle}}$ and a fixed attacker policy $D := \{d_\ell : \ell \in \mathcal{L}\} \in \mathcal{D}$, we denote by $J_{RD}(\epsilon)$ the value of the cost (1) when these policies are utilized. We are interested here in determining a security policy $R^*(\epsilon)$ defined by

$$J^*(\epsilon) := \min_{R\in\mathcal{R}_{\text{nocycle}}} \max_{D=\{d_\ell\}} J_{RD}(\epsilon). = \max_{D=\{d_\ell\}} J_{R^*(\epsilon)D}(\epsilon).$$

This policy guarantees a cost no larger than $J^*(\epsilon)$ regardless of the policy D used by the attacker. Moreover, it is the policy that minimized this "worst-case" cost.

For a given $\epsilon > 0$, consider the system of equations

$$\delta_{i1}\mu + (1+\epsilon) \sum_{j:\vec{ji}\in\mathcal{L}} x_{\vec{ji}} = \sum_{j:\vec{ij}\in\mathcal{L}} x_{\vec{ij}}, \qquad i \in \mathcal{N} \setminus \{n\}.$$

Defining $x := \{x_\ell : \ell \in \mathcal{L}\}$, the equations above can also be written as

$$A(\epsilon)x + \mu c = 0,$$

where $A(\epsilon)$ is an appropriately defined matrix, and c an appropriately defined vector. The following was proved in [1]:

Theorem 1 *For any $\epsilon, b > 0$,*

$$J^*(\epsilon) = \frac{b}{\max\limits_{\mu} \max\limits_{\substack{x_\ell\in[0,b]:\\ A(\epsilon)x+\mu c=0}} \mu}, \tag{2}$$

and a route-free security policy $R^*(\epsilon)$ can be computed by[1]

$$r^*_{\vec{ik}} := \frac{x^*_{\vec{ik}}}{\sum_{j:\vec{ij}\in\mathcal{L}} x^*_{\vec{ij}}}, \qquad \vec{ik} \in \mathcal{L},$$

where the $x^* := \{x^*_\ell : \ell \in \mathcal{L}\}$ maximizes (2). Moreover, for $\epsilon = 0$, the policy $R^*(\epsilon)$ maximizes the throughput from source to destination, assuming that all links have bandwidths equal to b.

[1] Since, in general, the x^* that maximize (2) are not unique, the security policy R^* is also not unique. However, it is always possible to choose an x^* that leads to a cycle-free R^*.

Figure 2 show the routing policies obtained using this algorithm for a test network topology. All links in this topology are bidirectional. This figure shows the policies for three values of the ϵ parameter: 0, 1, and 500. For $\epsilon = 0$ we get the policy that maximizes the flow from source to destination, whereas for $\epsilon = 500$ we get the policy that minimizes the number of hops. For $\epsilon = 1$ we get a compromise solution.

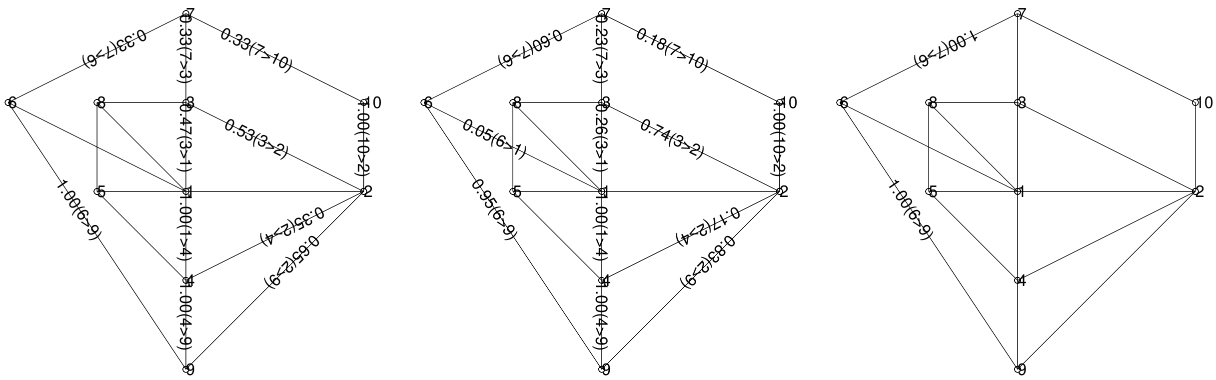

Figure 2: Routing policies for three values of ϵ: 0 (left), 1 (middle), and (500) right. The source and destination nodes are 7 and 9, respectively. The label superimposed on each link is of the form $r_{\vec{ik}}(i > k)$. The lack of label on a link means that no packets will be routed through that link.

4 Simulation Results

To test the routing algorithm proposed, we simulated the network in Figure 2, using the `ns-2` network simulator [7]. In all simulations presented here, all link have a propagation delay of 20ms, a bandwidth of 10Mbps, and all packets are 1000bytes long.

We were interested in determining the effect of stochastic routing on security, throughput, and packet transmission delay. To evaluate how secure a particular routing policy is we determine the maximum percentage of packets that an attacker could see by eavesdropping on one, two, or three links. We assumed here that the attacker chooses the set of link that maximizes the percentage of packets seen (i.e., the "worst" case scenario). Figure 3 show the simulation results obtained for several values of the parameter ϵ. Constant bit rate (CBR) traffic was used in all these simulations. As expected, routing is most secure for $\epsilon = 0$ since in this case all paths are equally good and the packets will be spread the most across all paths. As ϵ increases, more packets are routed through the shorter paths and therefore an attacker eavesdropping on those will see a larger percentage of packets. In fact, for $\epsilon = 500$, we essentially have minimum-hop routing and by attacking a single link it is possible to see every packet. Because in the network tested there are only three independent paths, when the attacker is allowed to eavesdrop on three link she will be able to see every packet, regardless of which type of routing is used.

As a side benefit—and not surprisingly in light of Theorem 1—for $\epsilon = 0$ we can achieve maximum throughput from source to destination. This is supported by the data in Figure 4, where we plot the drop-rate as a function of the source's sending rate. For $\epsilon = 500$ drops start occurring at sending rates around 10Mbps, whereas for $\epsilon = 0$ drops only become significant for a sending rates of 30Mbps.

The price to pay for security and throughput comes in terms of the average transmission delay per packet. This is because to achieve high security one needs to explore all the paths from source to destination, including those with many hops. Figure 5 shows that for large ϵ we indeed have minimum average delay and, as ϵ decreases the delay increases.

It turns out that problems arise when stochastic routing is used in conjunction with TCP-Reno. Figure 6 shows the average throughput of a long-lived TCP source operating under stochastic routing. We can see

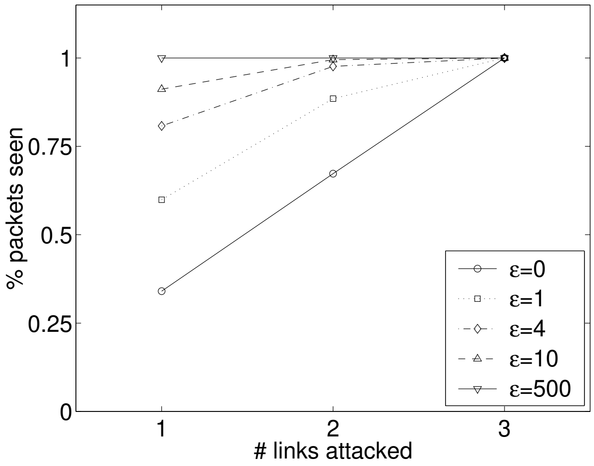

Figure 3: Worst-case percentage of packets seen by eavesdropped on one, two, or three links (CBR traffic).

that the throughput for $\epsilon = 500$ (essentially minimum-hop routing) is roughly six times larger than that for any other value of ϵ. This is because packets sent through longer paths are very often assumed dropped by TCP, as they only arrive after several packets that were sent later but traveled through shorter paths. Because fast-retransmit is only triggered when three duplicate acknowledgments are received, TCP-Reno is able to handle some degree of out-of-order packets. However, in all our simulations this proved insufficient to handle packets traveling through paths with distinct propagation times. It is interesting to note that the large penalty in throughput does not arise because of a large number of retransmissions. It happens instead because TCP assumes that there is congestion and keeps the window size very small, resulting in gross underutilized of the available bandwidth. This difficulty arises in order routing algorithms that explore multi-path and had been noticed, e.g., in [3, 8].

5 Conclusions

In this paper we illustrates how the game theoretical tools introduced in [1] can be used to develop secure network-layer routing policies. These policies address security issues such as packet interception and eavesdropping. We utilize `ns-2` simulations to evaluate the routing policies proposed. The routing policies proved efficient in achieving statistical flooding at the expense of some increase in the average package transmission delay. A beneficial side effect of these policies is an increase of the available bandwidth, as they explore multi-path. However, TCP congestion control interpreted the out-of-order packets as a sign of congestion, resulting in a very low throughput. We are currently exploring methods to overcome TCP's low performance. A technique that seems promising is to adjust the number of duplicate acknowledgements that trigger TCP's fast-retransmit. One should note that most of the techniques proposed for other multi-path routing techniques are not applicable here as they attempt to force all packets from a flow to follow the same path [3, 8]. Another direction for future investigation is the use of the stochastic routing policies arising from the online games in [1]. Some of these policies do not require routing "matrices," because the next-hop routing only depends on the final destination. The advantages/disadvantages of these approaches needs to be investigated.

References

[1] J. P. Hespanha and S. Bohacek, "Preliminary results in routing games," in *Proc. of the 2001 American Contr. Conf.*, June 2001.

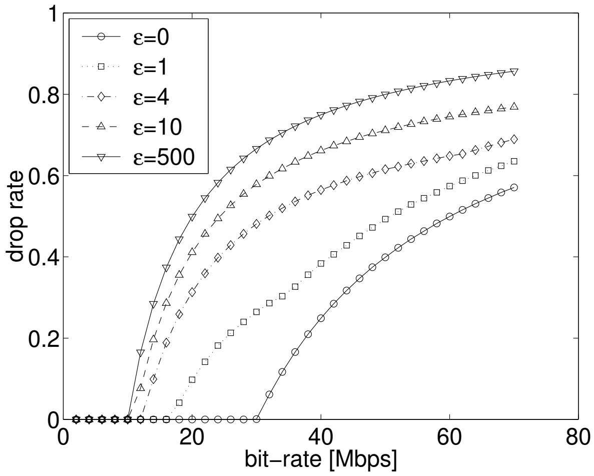

Figure 4: Drop-rate vs. sending rate.

[2] C. Hopps, "Analysis of an equal-cost multi-path algorithm," *RFC 2992*, Nov. 2000.

[3] C. Villamizar, "OSPF optimized multipath (OSPF-OMP)," *Internet-Draft (draft-ietf-ospf-omp-03)*, June 1999.

[4] A. Emmett, "VPNs," *America Networks*, May 1998.

[5] IP Security Working Group, "IPsec." Available at *http://www.ietf.org/html.charters/ipsec-charter.html*.

[6] P. F. Syverson, M. G. Reed, and D. M. Goldschlag, "Onion routing access configurations," in *DISCEX 2000: Proceedings of the DARPA Information Survivability Conference and Exposition*, vol. I, pp. 34–40, IEEE CS Press, Jan. 2000.

[7] The VINT Project, a collaboratoin between researchers at UC Berkeley, LBL, USC/ISI, and Xerox PARC, *The ns Manual (formerly ns Notes and Documentation)*, Oct. 2000. Available at http://www.isi.edu/nsnam/ns/ns-documentation.html.

[8] D. Thaler and C. Hopps, "Multipath issues in unicast and multicast next-hop selection," *RFC 2991*, Nov. 2000.

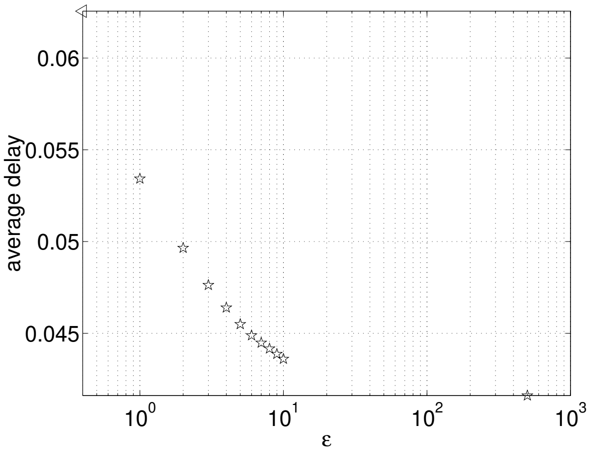

Figure 5: Average packet transmission delay (CBR traffic). The triangle over the vertical axis corresponds to $\epsilon = 0$ (which could not be represented in log-scale).

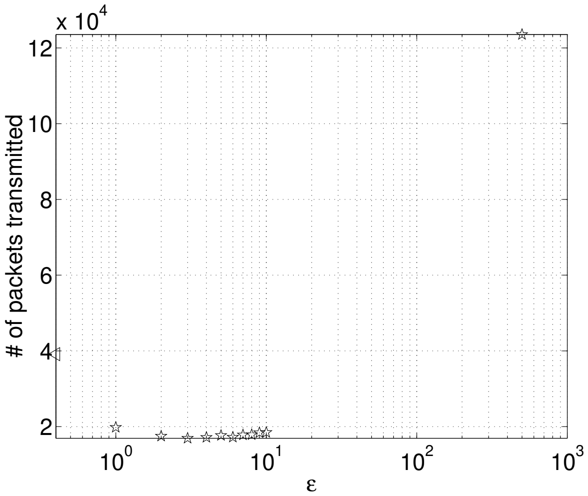

Figure 6: Number of packets transmitted (TCP traffic). The triangle over the vertical axis corresponds to $\epsilon = 0$ (which could not be represented in log-scale).