NS-EVALUATION OF SECURE STOCHASTIC ROUTING
TECHNICAL REPORT

March 28, 2002

Stephan Bohacek* bohacek@math.usc.edu
João P. Hespanha⊥ hespanha@ece.ucsb.edu
Katia Obraczka+ katia@cse.ucsc.edu
Junsoo Lee* junsoole@usc.edu
Chansook Lim* chansool@usc.edu

* Department of Mathematics & Computer Science, Univ. of Southern California, Los Angeles, CA 90089-1113
⊥ Dept. Electrical & Computer Engineering, Univ. of California, Santa Barbara, CA 93106-9560
+ Computer Engineering Department, University of California, Santa Cruz, CA 95064

Abstract

Shortest path routing leaves connections at risk of interception and eavesdropping since the path over which data packets travel is fairly predictable and easy to determine. To improve routing security we propose a proactive mechanism we call *secure stochastic routing* that explores the existence of multiple routes and forces packets to take alternate paths **probabilistically**. In this paper, we investigate game theoretic techniques to develop routing policies which make interception and eavesdropping maximally difficult. Through simulations, we validate our theoretical results and show how the resulting routing algorithms perform in terms of the security/delay/throughput trade-off. We observed that a beneficial side-effect of these algorithms is an increase in throughput, as they make use of multiple paths. The Internet was designed to use redundancy to enhance reliability. We suggest here that, through stochastic methods, redundancy be used to increase security.

I. INTRODUCTION

One of the most obvious attacks to a communication network is packet interception which prevents data originating from one (or several) nodes to reach the destination. Eavesdropping can be thought as a "passive" form of interception, in which packets are "snooped" but not removed from the network. In "traditional" shortest-path routing protocols, the path over which a data packet travels is fairly predictable and easy to determine. Even if several paths with the same number of hops exist, routing algorithms typically select one of the possible options and utilize that same path for all packets. Indeed, a study by Zhang et al. [1] reveals that Internet routes are fairly persistent (e.g., often the same route between a source-destination pair persists for days; only 10% of the routes persist for a few hours or less). This makes IP networks vulnerable to *packet interception* and/or *eavesdropping* attacks. Notable exceptions to single-path routing schemes 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. In practice, they do not introduce unpredictability and therefore packet interception is fairly easy to achieve.

In this paper, we describe *secure stochastic routing*, or *SSR*, whose main goal is to make packet interception maximally difficult. These algorithms explore the existence of multiple paths between two network nodes and route packets to minimize predictability. Routers compute all possible paths between a source-destination pair and, according to a given probability distribution, assign some probability to each next-hop. The net effect is that data packets traverse random paths on their way from the source to the destination. We should point out that, unlike security mechanisms that are based on detection and response, we take a *proactive* approach to making routing less vulnerable to attacks. In other words, packets are always sent along multiple paths according to some probability.

Packet Interception and Eavesdropping

The simplest form of packet interception is to physically disable a communication link ("cut-the-cable"). However, this type of attack generally does not pose a serious concern because the Internet was designed to circumvent these types of (temporary or permanent) outages. A more insidious way of intercepting packets is to drop them at compromised routers ("infiltrate-a-router"). In deterministic shortest path routing, if interception is taking place on a link or router, potentially, all packets going over that link (or through the compromised router) can be intercepted. Furthermore, if the attacker does not intercept routing update packets, routers do not compute new routes in order to exclude the compromised link/router. This means that retransmitted data (generated by application- or transport-layer reliability mechanisms) will keep following the same route and will be subject to interception. A more "active" form of attack exploiting compromised routers is to have them generate spurious routing updates reporting very low costs to all or some destinations. As a result, traffic will be funneled through the compromised router(s) where it can be intercepted.

Privacy and integrity techniques (e.g., end-to-end encryption, Virtual Private Networks, secure tunnels, etc.) are quite effective in protecting against eavesdropping. However, they do not provide sufficient protection against many forms of interception attacks, or even eavesdropping when encryption keys have been compromised. We believe that the key to effectively protect a data transmission network against a wide range of attacks is a suite of mechanisms spanning several layers of the protocol stack: According to the end-to-end argument in system design [4], applications requiring security guarantees should use end-to-end security mechanisms (e.g., end-to-end encryption for privacy, message authentication for integrity/authenticity, etc.). However, these should be complemented by security mechanisms at other layers (e.g., link-layer encryption). SSR is one such mechanism (operating at the network layer) that *is not meant to replace end-to-end encryption* (or any other security mechanism) but to complement it by providing an added level of security (at little or no extra cost). For example, suppose that deterministic routing is utilized and an encryption key is stolen, then encryption fails to provide protection as packets can be sniffed by a single host in the minimum-hop path. Now suppose that for each transaction a session key is exchanged. If the key exchange is carried out over several packets, then SSR may send these packets over different paths making the key maximally difficult to intercept. In this way, the combination of encryption and SSR results in an effective two-layer defense.

Exploring Multiple Paths

The approach we propose here is to explore multiple 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 (although error control at the transport layer may require a packet to be re-sent). Flooding occurs "on-average" and is randomized to minimize the probability of interception by an attacker: as long as there exists one path between source and destination that has not been compromised, the probability that the attacker will intercept all the packets can be made equal to zero. Note that statistical flooding does not prevent *every packet* from being intercepted. Indeed, if one node in the path between source and destination is compromised, some packets will almost certainly be intercepted. However, reliable transport will make sure that the intercepted packets will eventually find a non-compromised path.

SSR also provides defense against other types of attacks. For example, an attacker could intermittently cut-a-link leading to problems in the convergence of routing tables, and, every time the link goes down, many packets would be lost until the next routing table update occurs. With SSR, some level of functionality is still provided because some packets avoid the compromised portion of the network. Of course that some level of degradation would still occur but SSR makes these attacks maximally difficult/minimally effective. SSR also provides some degree of protection against traffic analysis because traffic is spread across the network and is therefore more difficult to determine which nodes are originating most of the traffic carried through the network (at least if some form of encryption is used to hide the source and destination of the packets).

One question to ask is "where can SSR be effectively deployed?" Clearly, SSR will only be effective if there are multiple paths to explore. We see at least two domains in which this technique may prove useful: (1) Inside Internet service providers (ISPs) where there are often multiple independent paths to the exit point for that ISP. SSR would be part of the suite of tools that the ISP utilizes to provide secure networking to its clients. (2) Inside an organization that builds multi-path redundancy in its network to improve security (and also to augment throughput). An organization may utilize multiple connections to a single ISP, or even connect to multiple ISPs to connect to the outside network and employ SSR to spread data across independent paths provided by distinct ISPs.

II. Related Work

Virtual Private Networks [5], or VPNs, have been used as a way to securely interconnect a (typically small) number of sites. While private networks use dedicated lines, VPNs try to implement private networks atop a publicly-accessible communication infrastructure like the Internet. VPNs typically employ some combination of encryption, authentication, and access control techniques to allow participating sites to communicate securely. The emergence of IPsec [6] as a IETF standardized protocol has prompted VPN solutions to use IPsec as the underlying network-layer protocol. Secure BGP (S-BGP) [7] makes use of public key and authorization infrastructure, as well as IPsec to verify the authenticity and authorization of control traffic generated by the Border Gateway Protocol (BGP).

Onion routing [8] is another approach to security that focuses on hiding the identities of communicators. It uses several layers of encryption, where each layer is used to encrypt the transmission between routers on each end of a link. Because of the many layers of encryption, routers are unable to decrypt the data or even the source and destination addresses. All that a router can decipher is the next-hop information. While onion routing is very effective for anonymity, it is not very efficient: each connection must be built and torn down, routers must encode and decode packets, and memory-intensive source routing is used.

More recently, motivated by constant distributed denial-of-service attacks (DDoS) to the Internet, several research efforts have focused on addressing routing-level security. Centertrack [9] uses an IP overlay network composed of Centertrack routers to log packets. Using the resulting logs, it is possible to reconstructed the path traversed by attackers packets. IP traceback [10] tries to accomplish the same goal of tracking DDoS attacks by tracing packets back to their source. The IP traceback mechanism uses probabilistic packet marking at routers which allows a victim to identify the path(s) attack traffic traversed without support from service providers/administrators.

Another related effort is the Resilient Overlay Networks (RON) project [11] whose goal is to improve the performance and robustness of network-layer routing. RON nodes monitor current routing paths and decide whether to choose other routes (by selecting alternate application-layer paths through other RON nodes) to meet application-specific performance requirements.

The approach developed here is, in some ways, similar to that presented in [12]. In [12], members of a group cooperate to maintain their anonymity to the server. Specifically, users send their request not directly to the server but to random users in the group. These users can either forward the request to the server or to another member of the group.

SSR was originally proposed in [13] where only link-level attacks were considered. Here, we extend these ideas to more general attacks that include, e.g., node-level attacks among others. Moreover, in [13] it was assumed that an attack would always succeed, whereas here the probability of an attack succeeding is a design parameter. Here, we also present simulation results that validate the effectiveness of SSR. These simulations show that SSR can lead to an increase in throughput when sending Constant Bit Rate (CBR) traffic. Moreover, with small modifications to TCP, this increase in throughput is maintained for TCP flows. However, we show that if TCP is not modified, the throughput can be drastically reduced due to out-of-order packets.

III. Stochastic Routing

At the heart of the network layer in each (deterministic) router, 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. SSR 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 SSR is the determination of routing tables that ensure: *Delivery*, i.e., all packets will reach their desired destination with probability one; *Timeliness*, i.e., the delay is acceptable; and *Statistical flooding* is achieved, i.e., all paths are explored. With respect to statistical flooding, 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.

A. Computation of stochastic routing tables

In [13] several routing games were proposed to compute stochastic routing tables that guarantee delivery, timeliness, and stochastic flooding. The key technical insight was 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. The solution to such games requires the use of mixed (randomized) policies [14]. In practice, the mixed solutions provide 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. Statistical flooding is a consequence of the saddle solution.

In this paper, we consider *off-line routing games* [13] 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 $J(\varepsilon) := \mathrm{E}[\chi(\varepsilon)]$, where $\varepsilon \geq 0$ is a design parameter and $\chi(\varepsilon)$ is a random variable that is equal to $(1+\varepsilon)^{t-1}$ if the packet is intercepted at the tth hop and 0 otherwise. For $\varepsilon = 0$, the random variable $\chi(\varepsilon)$ 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 $\varepsilon \neq 0$, $J(\varepsilon)$ 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 $\varepsilon \to \infty$, 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 used as an optimization tool to compute stochastic routing tables that guarantees delivery, timeliness, and stochastic flooding. Clearly, as formulated above, it is too simplistic to model a realistic attack.

We consider a data transmission network with nodes $\mathcal{N} := \{1,2,\dots,n\}$ 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. 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 a list $R := \{r_\ell : \ell \in \mathcal{L}\}$ that satisfies $\sum_{k:\vec{ik} \in \mathcal{L}} r_{\vec{ik}} = 1$, $\forall i \in \mathcal{N}$. Here, we restrict our attention to *stochastic cycle-free routing policies*. These are stochastic routing policies for which a packet will not return to a node where it has been before with probability one. We denote by $\mathcal{R}_{\text{nocycle}}$ the set of lists with this property. 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 need to determine a security policy $R^*(\varepsilon)$ such that

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

where $J_{RD}(\varepsilon)$ denotes the value of the cost $J(\varepsilon)$ incurred when the routing policy R is used and the attacker selects the policy D. The policy $R^*(\varepsilon)$ guarantees a cost no larger than $J^*(\varepsilon)$ regardless of the policy used by the attacker. Moreover, it is the policy that minimized this "worst-case" cost. For a given $\varepsilon > 0$, consider the system of equations

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

Defining $x := \{x_\ell : \ell \in \mathcal{L}\}$, the equations above can also be written as $A(\varepsilon)x + \mu c = 0$, where $A(\varepsilon)$ is an appropriately defined matrix, and c an appropriately defined vector. The following was proved in [13]:

Theorem 1: For any $\varepsilon \geq 0$,

$$J^*(\varepsilon)^{-1} = \max_{\mu} \max_{x_\ell \in [0,1]: A(\varepsilon)x + \mu c = 0} \mu, \tag{1}$$

and a route-free security policy $R^*(\varepsilon)$ can be computed by $r^*_{\vec{ik}} := \frac{x^*_{\vec{ik}}}{\sum_{j:\vec{ij} \in \mathcal{L}} x^*_{\vec{ij}}}, \vec{ik} \in \mathcal{L}$, where $x^* := \{x^*_\ell : \ell \in \mathcal{L}\}$ maximizes (1) . Moreover, for $\varepsilon = 0$, $R^*(\varepsilon)$ maximizes the throughput from source to destination, assuming that all links have the same bandwidth.

Figure 1 show the routing policies obtained in a test network topology. For $\varepsilon = 0$ we get the policy that maximizes the flow from source to destination, whereas for $\varepsilon = 500$ we get the policy that minimizes the number of hops. For $\varepsilon = 1$ we get a compromise solution.

B. Extensions to Different Attack Scenarios

Nonhomogeneous Attack Probabilities. In [13] it was assumed that an attack is equally likely to succeed on each of the links However, nonhomogeneous attack probabilities may be desirable possible. For example, when connections pass through (less secure) public and (more secure) private networks. Furthermore, intrusion detection sensors could provide indications where an attack may be occurring. The homogeneous case presented above can be extended to the case where some links are more prone to attack than others. To this effect let $p_\ell \in [0,1]$ be the probability that an attack will succeed at intercepting packet on link ℓ. Then Theorem 1 holds with Equation (1) replaced by

$$J^*(\varepsilon)^{-1} = \max_{\mu} \max_{p_\ell x_\ell \in [0,1]: A(\varepsilon)x + \mu c = 0} \mu.$$

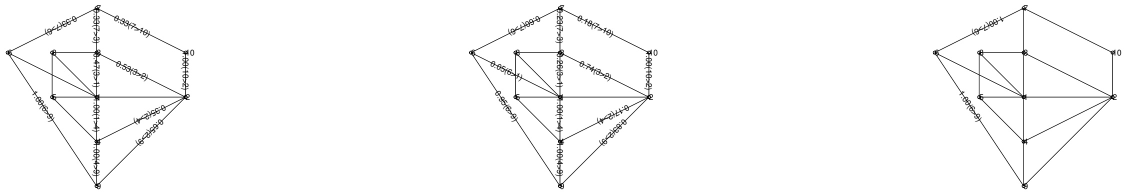


Fig. 1. Routing policies for three values of ε: 0 (left), 1 (middle), and (500) right. The source and destination nodes are 7 (the upper most node) and 9 (the lowest node), respectively. The label superimposed on each link is of the form $r_{\vec{ik}}(i > k)$. The lack of label on a link means that no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

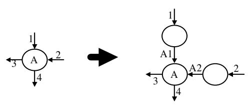
When $p_\ell = 0$, then there is no restriction on x_ℓ. Thus, links with low drop probabilities will tend to have larger utilization.

Aggregate Attacks on Non-serial Elements. In some cases it is possible to attack many links simultaneously. Suppose, for example, that two ISPs lease a portion of the same physical link and each ISP represents the link as a distinct logical link. Hence, attacks on the physical link will result in an aggregate of logical links being attacked. The general problem of aggregate attacks is complicated by the fact that a single attack may have the opportunity to intercept a packet multiple times. However, this only occurs only when there is, in effect, a routing cycle. As discussed above, in order to make the problem more tractable, routing cycles have not been consider. Thus, we assume that the aggregate attack does not attack network elements that could be traversed by a packet in a serial fashion. This simplification has no impact when only one link is attacked, however, it can be difficult to verify when multiple network elements are attacked. Fortunately, even when it is not satisfied, the effect of the incorrect assumption is small when the probability that the attack succeeds is small. With this assumption in place, aggregate attacks can be solved as follows: Let $\mathcal{A}$ denote the set of possible attacks (perhaps each one corresponding to interception on a particular link that is vulnerable) and let I_i, $i \in \mathcal{A}$ be the set of links where interception will occur when attack i is executed. Define the matrix B such that $B_{i,\ell}$ is equal to one if $\ell \in I_i$, $i \in A$ and zero otherwise. Then Theorem 1 holds with (1) replaced by

$$J^*(\varepsilon)^{-1} = \max_{\mu} \max_{Bx \leq 1, x \geq 0 : A(\varepsilon)x + \mu c = 0} \mu.$$

In the case of nonhomogeneous attack probabilities, B should be defined such that $B_{i,\ell} = p_i$ if $\ell \in I_i$ and $B_{i,\ell} = 0$ otherwise, where p_i is the probability that attack i will succeed.

Node Attacks. While Section III-A discusses attacks at links, attacks at nodes are also possible. The problem of developing node attack defenses can be transformed into a problem of developing a defense against link attacks by transforming the network graph so that each link entering a node is transformed into a node and a link as shown in Figure 2. Attacks on the node are consider as simultaneous attacks on the links entering into the node. Note that these added links (labeled A1 and A2 in Figure 2), are not serial elements unless a routing cycle exists. Therefore, the method developed in Section III-B can be used.



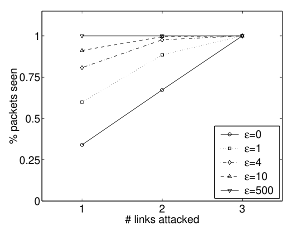


Fig. 2 Fig. 3

Fig. 2. Converting a Node Attack to a Link Attack.

Fig. 3. Worst-case percentage of packets an attacker could intercept by compromising one, two, or three links (CBR traffic).

IV. Simulation Results

To evaluate the routing algorithm proposed in Section III-A, we simulated the network in Figure 1, using the `ns-2` network simulator [15]. In the simulations presented, all links have propagation delay of 20ms and bandwidth of 10Mbps. Each queue implements drop-tail queuing discipline with maximum queue size set to 200 packets for the case of the CBR simulations and 100 packets for the TCP simulations. All packets are 1000 bytes long. The simulation time for each trial was 100 seconds. Experiments were performed using either CBR or TCP-SACK traffic. In all experiments, the security parameter ε is fixed during a trial but varies from trial to trial.

We were interested in determining the effect of SSR on security, throughput, and packet transmission delay. To evaluate how secure a particular routing policy is, we determined the maximum percentage of packets that an attacker could intercept by compromising one, two, or three links. We assumed here that the attacker chooses the set of links that maximizes the percentage of packets seen, i.e., the worst-case scenario. Figure 3 shows the simulation results obtained for several values of the parameter ε. Constant bit rate (CBR) traffic was used in the simulations shows, but similar plots are also obtained with TCP-SACK traffic (even taking into account the retransmissions that occur with this protocol). As expected, routing is most secure for $\varepsilon = 0$, since packets will be spread the most across all paths. As ε increases, more packets are routed through the shorter paths and therefore an attacker can intercept a larger percentage of packets. In fact, for $\varepsilon = 500$, 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 compromise three carefully chosen links she will be able to intercept every packet, regardless of which type of routing is used.

As a side benefit, for $\varepsilon = 0$ we can achieve maximum throughput (from a sender's perspective) since we make use of all independent paths to the 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 $\varepsilon = 500$, drops start occurring at sending rates around 10Mbps, whereas, for $\varepsilon = 0$, drops only become significant for sending rates higher than 30Mbps. In general, the increase in throughput depends on the redundancy in the topology and the amount of cross-traffic. The price to pay for security comes in terms

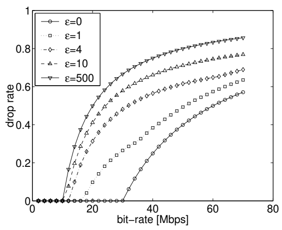


Fig. 4

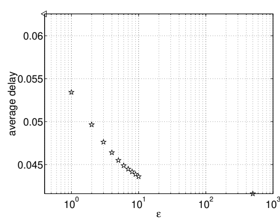


Fig. 5

Fig. 4. Drop-rate vs. sending rate.

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

of the average latency per packet. This is because to achieve high security one explores all the paths from source to destination, including those with high latency. Figure 5 shows that for large ε we indeed have minimum average delay and as ε decreases the delay increases.

Incurring higher per-packet average delay is not problematic for "lock-step" protocols like the Trivial File Transfer Protocol (TFTP) [16]. By buffering data at the receiver, even some multimedia streaming applications can handle the increased delay and its variation. However, other problems arise when SSR is used in conjunction with standard TCP. Figure 6 shows the average throughput of a long-lived TCP-SACK flows (triangle $\triangleright$ and circles $\circ$) operating under SSR. For the TCP-SACK, the throughput for $\varepsilon = 500$—essentially minimum-hop routing—is roughly six times larger than that of other value of ε (cf. circle $\circ$ at the right with others at the left). This is because packets sent through longer paths are 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 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. The fact that standard TCP performs poorly when used in conjunction with multi-path routing has been observed in [3], [17].

To address this problem, we developed TCP-MP, which is a modification of TCP that is able to increase the throughput under multi-path routing. TCP-MP does not assume a packet is dropped when out-of-order sequence numbers are observed, but only when a time-out occurs. Figure 6 (triangle $\triangleleft$ and crosses $\times$) shows that in the case of $\varepsilon = 0$, the throughput of the TCP-MP is larger than all other configurations. Thus recovering the results observed for CBR traffic. Figure 7 shows the drop rate for the different TCP implementations and different ε. Figures 6 and 7 show that TCP-SACK slightly out-performs TCP-MP when $\varepsilon = 500$ (i.e., using shortest path routing). In this case, TCP-SACK has slightly higher throughput and slightly fewer drops. However, for all other value of ε, TCP-MP continues to perform well—in the sense that its throughput is high and the number of drops is smaller than that of min-hop routing—whereas the performance of TCP-SACK is greatly degraded. Thus, it can be concluded that, if the proposed routing methods are implemented and TCP-MP is utilized, there is no substantial loss in performance (and even some possible increase in throughput) but security can be greatly improved. Work remains to be done before TCP-MP can be safely deployed. A significant concern is the fairness between TCP-MP and other versions of TCP. Furthermore, since only timeouts are used to detect drops, there are important issues related to the selection of the timeout period.

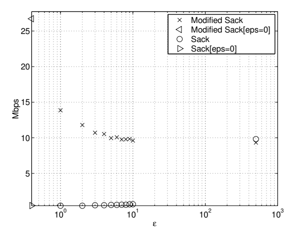


Fig. 6

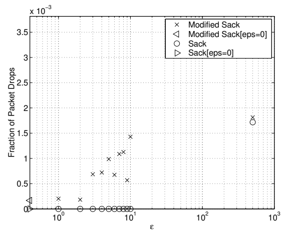


Fig. 7

Fig. 6. Transmission rate of TCP and a modified version of TCP for different security levels ε. Since the x-axis is in logarithmic scale, the point $\varepsilon = 0$ is indicated with an triangle on the y-axis.

Fig. 7. Drop rate of TCP-SACK and a modified version of TCP for different security levels ε. For $\varepsilon < 10$, the TCP-SACK flow experienced no drops. Since the x-axis is in logarithmic scale, the point $\varepsilon = 0$ is indicated with an triangle on the y-axis.

V. Conclusions

We investigated the use of SSR and demonstrated through simulations that it improves security. By proactively forcing packets to probabilistically take alternate paths, SSR mitigates the effects of interception, eavesdropping, and traffic analysis attacks. The routing policies proposed proved efficient in achieving statistical flooding at the expense of some increase in average package transmission delay. A beneficial side effect is an increase in throughput. However, when used with TCP, this protocol needs to be modified to achieve this. This is a topic of current research.

The routing computation algorithms presented in this paper require each node to have full topology information, implying the use of a link-state routing protocol. We are currently investigating if these or other stochastic routing algorithms can be implemented under distance-vector routing. Another implementation approach is to use source routing whereby the end-hosts define the path to be taken by each packet and is responsible for the randomization of paths. The use of source routing to enhance security has suggested in [18]

References

[1] Y. Zhang, V. Paxson, and S. Shenker, "The stationarity of internet path properties: Routing, loss, and throughput," tech. rep., ACIRI, May, 2000.

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

[3] C. Villamizar, "Ospf optimized multipath (ospf-omp)," *draft-ietf-ospf-omp-03*, p. 46, June 1999.

[4] J. Saltzer, D. Reed, and D. Clark, "End-to-end arguments in system design," *AMC Trans. on Computer Systems*, vol. 2, no. 4, pp. 195–206, 1984.

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

[6] S. Kent, "Security architecture for the internet protocol." Request for Comments (RFC) 2401, 1998.

[7] S. Kent, C. Lynn, J. Mikkelson, and K. Seo, "Secure border gateway protocol (s-BGP) - real world performance and deployment issues," in *Proceedings of NDSS 2000*, 2000.

[8] 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, Jan. 2000.

[9] R. Stone, "CenterTrack: An IP overlay network for tracking DoS floods," in *9th USENIX Security Symposium*, 2000.
[10] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Practical network support for IP traceback," in *Proceedings of the 2000 ACM SIGCOMM Conference*, (Stockholm, Sweden), pp. 295–306, August, 2000.
[11] D. G. Andersen, H. Balakrishnan, M. F. Kaashoek, and R. Morris, "Resilient overlay networks," in *Proc. 18th ACM SOSP*, (Banff, Canada), 2001.
[12] M. K. Reiter and A. D. Rubin, "Crowds: Anonymity for Web transactions," *ACM Trans. on Information and System Security*, vol. 1, pp. 66–92, 1998.
[13] J. P. Hespanha and S. Bohacek, "Preliminary results in routing games," in *Proc. Of the 2001 American Control Conference*, June, 2001.
[14] S. D. Patek and D. P. Bertsekas, "Stochastic shortest path games," *SIAM J. Contr. Optimization*, vol. 37, pp. 803–824, 1999.
[15] The VINT Project, a collaboration between UC Berkeley, LBL, USC/ISI and Xerox PARC, "The ns manual (formerly ns Notes and Documentation)." http://www.isi.edu/nsnam/ns/ns-documentation.html, Oct. 2000.
[16] K. Sollins, "The TFTP protocol." RFC 1350, 1992.
[17] D. Thaler and C. Hopps, "Multipath issues in unicast and multicast next-hop selection," *RFC 2991*, Nov. 2000.
[18] R. Perlman, *Network Layer Protocols with Byzantine Robustness*. PhD thesis, MIT, 1988.