SmartTunnel: Achieving Reliability in the Internet

Yi Li, Yin Zhang, LiLi Qiu, Simon Lam
Department of Computer Sciences, University of Texas at Austin
{ylee, yzhang, lili, lam}@cs.utexas.edu

***Abstract*— Reliability is critical to a variety of network applications. Unfortunately, due to lack of QoS support across ISP boundaries, it is difficult to achieve even two 9s (99%) reliability in today's Internet. In this paper, we propose *SmartTunnel*, an end-to-end approach to achieving reliability. A SmartTunnel is a logical point-to-point tunnel between two end points that spans multiple physical network paths. It achieves reliability by strategically allocating traffic onto multiple paths and performing FEC coding. Such an end-to-end approach requires no explicit QoS support from intermediate ISPs, and is therefore easy to deploy in today's Internet. To fully realize the potential of SmartTunnel, we analytically derive near-optimal traffic allocation schemes that minimize loss rates. We extensively evaluate our approach using trace-driven simulations, ns-2 simulations, and experiments on PlanetLab. Our results clearly demonstrate that SmartTunnel is effective in achieving high reliability.**

I. Introduction

Many applications, such as Voice over IP (VoIP), video conferencing, streaming media, gaming, and online trading have stringent requirements on end-to-end reliability. Unfortunately, today's Internet does not even provide two 9s (99%) reliability [16], [10], [8]. This is considerably lower than what the public switched telephone network (PSTN) offers today (three to four 9s).

Achieving end-to-end reliability is hard in the Internet for several reasons. First, there lack both incentives and mechanisms for ISPs to cooperate. This implies that it is hard to provide reliability guarantees across ISP boundaries. As a result, while it may be possible to achieve high reliability within an individual ISP, the reliability of end-to-end paths, spanning over multiple ISPs, is significantly lower. Second, numerous measurement studies [26], [3], [28] have shown that Internet packet loss often exhibits burstiness (or temporal dependency). Bursty losses pose significant challenges to protect against. While forward error correction (FEC) coding is useful to protect against random losses, bursty packet losses significantly affect the effectiveness of FEC.

To address these challenges, we propose *SmartTunnel* to achieve reliability in today's Internet. A SmartTunnel is a logical point-to-point tunnel between two communicating end points that may physically span multiple Internet paths. It achieves reliability by properly allocating traffic across different paths and applying FEC coding. Such an end-to-end approach requires no explicit QoS support from intermediate ISPs, and can be directly applied to end hosts or Internet gateways. Therefore it is easy to deploy in today's Internet. Moreover, the simple tunnel abstraction can be naturally incorporated into applications and services such as virtual private networks to improve reliability.

Research sponsored by National Science Foundation CNS-0434515, CNS-0627020, CNS-0546720, CNS-0546755 and ANI-0319168

To realize the potential of SmartTunnel, we analytically derive near-optimal traffic allocation schemes that optimize end-to-end loss rates under bursty loss.

Our key contributions can be summarized as follows:

- We propose SmartTunnel abstraction to provide end-to-end reliability. This abstraction immediately supports diverse upper and lower network layer technologies without modification. It can also be easily incorporated into VPN services. Therefore SmartTunnel is easy to deploy in today's Internet.
- We analytically develop near-optimal traffic allocation schemes to optimize end-to-end loss rate in the presence of bursty packet losses. We further implement SmartTunnel using the click modular router [13].
- We extensively evaluate the effectiveness of SmartTunnel using trace-driven simulation, ns-2 simulations, and PlanetLab experiments. Our results show that SmartTunnel is effective in achieving high end-to-end reliability and multiple SmartTunnels can co-exist well.

The rest of the paper is organized as follows. In Section II, we survey related work. In Section III, we describe SmartTunnel architecture. In Section IV, we present algorithms that determine optimal traffic allocation. We describe our evaluation methodology and results in Section V. We conclude in Section VI.

II. Related Work

We broadly classify the related work into the following three areas: (i) measurement of Internet reliability, (ii) overlay routing and multihoming, and (iii) FEC based loss recovery.

Measurement of Internet reliability: Several measurement studies have reported that Internet reliability is quite limited. In particular, Paxson observed a routing pathology arises 1.5% - 5% during 1994-1995. Jiang et al. [10], and Gummadi [8] conducted large-scale measurement studies of Internet path failures, and reported that Internet reliability is often below two 9s (99%).

Overlay routing and multihoming: Several studies, such as [21], [24], [18], have shown that the default network path is often suboptimal in terms of latency, loss rate, and TCP throughput. To address these issues, a variety of overlay-based techniques have been proposed to improve network performance and resilience. For example, RON [18] allows distributed applications to recover network path failures by routing through alternative overlay paths. OverQoS [23] uses a controlled loss virtual link abstraction to bound the loss rate. Since OverQos uses only a single path, it cannot protect against highly bursty losses in a timely fashion. Zhang et al. [27] proposes mTCP, an end-to-end transport layer protocol, to enhance the robustness under path failures by using multiple paths. mTCP uses retransmission to recover packet loss and

is not suitalbe for real-time multimedia applications. More recently, scalable one hop source routing (SOSR) [8] proposes that upon path failures the source node randomly chooses four nodes as relay nodes to re-route traffic. Their results show that it can recover 20-56% failures. SOSR is a reactive approach and requires failure indications. Therefore it is suitable for recovering long-term failures, but not for recovering bursty loss, which is our main focus.

There are quite a few research studies on the design and evaluation of route control schemes for multihomed users. For example, Cao et al. [4] propose using hash functions to achieve load balancing among multiple links. Akella et al. [2] quantify the potential performance benefits of multihoming using real Internet traces. In their follow-up work, the authors implement a route control scheme based on either passive or active monitoring schemes. Their experimental evaluation show that their approach provides 15% to 25% performance improvement. The authors in [7] develop novel smart routing algorithms to simultaneously optimize cost and performance for multihomed users, and study the interactions between multihomed and single-homed users.

SmartTunnel can be applied to both overlay paths and multihoming paths. One of the fundamental differences between SmartTunnel and the existing work is that almost all the existing work uses only a single path (e.g., send traffic along the best performing path). Due to bursty packet losses in the Internet, using a single path yields limited reliability. In comparison, SmartTunnel can achieve high reliability by simultaneously using multiple paths.

FEC based loss recovery: Significant research work has been done on the design and evaluation of forward error correcting code, such as [17], [15]. Our work is orthogonal to the development of FEC coding algorithms, and can directly apply the existing systematic FEC coding schemes (i.e., FEC codes that include unmodified original data packets in the FEC group).

Jain *et al.* [9] study the problem of traffic allocation onto multiple paths to achieve high reliability. This is a pioneering work on this subject. It is also the work closest to ours. Different from our work, [9] targets delay tolerant networks, and the proposed approach is based on very different loss models from the Internet. In particular, they do not consider bursty loss. As a result, their approach does not work well for Internet paths, as we will show in Section V. Note that in order to cope with bursty loss, it is necessary to develop a completely different traffic allocation scheme (as opposed to a simple extension to [9]). We will further elaborate this point and compare SmartTunnel with [9] in Section IV-A.

III. SmartTunnel Architecture

SmartTunnel is a logical point-to-point tunnel between two communicating end points that may physically span multiple Internet paths. It sits at network-layer, and is transparent to applications. As shown in Figure 1, a tunnel source continuously monitors the network paths, and provides the performance of network paths to the controller. The controller applies the traffic allocation algorithm, described in Section IV, to distribute traffic onto multiple physical paths. On the data plane, data is first delivered to FEC encoder, which generates redundancy packets and hands over the resulting data and redundancy packets to the traffic distributor that stripes packets according to the controller's specification. The tunnel destination decodes and buffers data. Buffering is necessary to reduce packet reordering, which can degrade TCP performance. A packet is delivered to uppper layers either when all the packets before it have been received (or recovered) or when buffer is full.

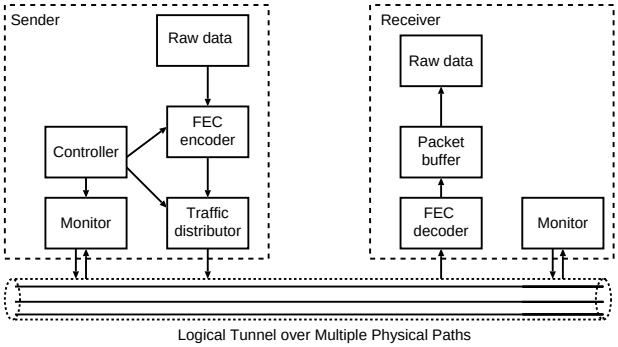

Fig. 1. SmartTunnel architecture.

SmartTunnel can be deployed either at the two communicating end-points or at their Internet gateways. SmartTunnel can also be easily integrated with current VPNs. Today's VPNs only provide reachability and security, but no reliability. By running SmartTunnel sender and receiver at the VPN servers of the two communicating points, network providers can offer value-added services.

IV. SmartTunnel Algorithms

To fully realize the potential of SmartTunnel, we need to address the following issues: (i) how to allocate traffic onto multiple paths to minimize loss rates under realistic Internet loss models, and (ii) how to measure path properties which can be used for the allocation schemes. In this section, we examine these issues in turn.

A. Allocation Problem Formulation

First we formulate the traffic allocation problem of minimizing packet loss rate. Table I summarizes our notations.

N	number of physical paths
D	number of data packets per FEC group
R	number of redundant packets per FEC group
G	FEC group size ($G = D + R$)
d_i	number of allocated data packets on path i
r_i	number of allocated redundant packets on path i
x_i	number of lost data packets on path i
y_i	number of lost redundant packets on path i
X	total number of lost data packets $X = \sum_{i=1}^{N} x_i$
Y	total number of lost redundant packets $Y = \sum_{i=1}^{N} y_i$
$\bar{X}_{\mathrm{FEC}}$	expected number of lost data packets after applying FEC: $\bar{X}_{\mathrm{FEC}} = \sum_{\ell=1}^{D} \ell \cdot Pr[X = \ell \wedge X + Y > R]$
$\bar{X}^{\approx}_{\mathrm{FEC}}$	continuous approximation of $\bar{X}_{\mathrm{FEC}}$ (see Section IV-B.1)

TABLE I
NOTATIONS

Consider a SmartTunnel from node s to node d. Traffic from node s to node d can take several different *physical paths*, which can be provided by either multihoming or overlay routing. Let N denote the number of physical paths available to the SmartTunnel. When a packet is transmitted along a physical path, it can get lost due to a variety of reasons, such as routing loops, failures, and network congestion. To achieve reliability, node s applies forward error correction (FEC) code

to protect packets that use the SmartTunnel. Specifically, for every D data packets, node s creates R redundant packets, which together with all the data packets form an FEC group of size $G = D + R$. The FEC code is designed to recover from R packet losses. That is, when node d receives any D out of G packets in the group, it can reconstruct the entire FEC group. If more than R packets are lost, however, node d can only deliver those successfully received data packets and cannot recover the lost data packets.

Given an FEC group with D data packets and R redundant packets, there are N^{D+R} different ways of allocating packets in an FEC group onto N physical paths. Different allocations can result in different numbers of packet losses after applying FEC. The goal of the SmartTunnel is to derive an optimal allocation that minimizes the expected number of packet losses after applying FEC. This problem can be formally specified as follows.

Definition 1 (Optimal Allocation Problem): Let d_i and r_i denote the number of data packets and redundant packets allocated on path i. Let random variables x_i and y_i denote the number of data packets and redundant packets that are lost on path i. Let random variables $X = \sum_{i=1}^{N} x_i$ and $Y = \sum_{i=1}^{N} y_i$ denote the total number of lost data packets and lost redundant packets, respectively. The optimal allocation problem is to determine an allocation $\{(d_i, r_i)\}$ under which the expected number of lost data packets after applying FEC is minimized. That is,

$$\text{minimize} \quad \bar{X}_{\text{FEC}} \equiv \sum_{\ell=1}^{D} \ell \cdot Pr[X = \ell \wedge X + Y > R] \quad (1)$$

where $\ell = 1, \cdots, D$ enumerates all possible values for X (*i.e.*, the total number of lost data packets), and the summation gives the expected number of lost data packets when FEC cannot reconstruct the entire FEC group (*i.e.*, when $X + Y > R$).

Modeling Temporal Loss Dependency in the Internet: To solve the above optimization problem, we first need to understand the behavior of Internet packet loss. Numerous measurement studies [26], [3], [28] have shown that Internet packet loss often exhibits burstiness. Burstiness affects the performance of FEC because when a large burst of packets are lost in an FEC group, FEC cannot recover the lost data packets. It is therefore important to explicitly model bursty loss in SmartTunnel.

A variety of models have been proposed to capture temporal loss dependency, including the 2-state Gilbert Model [6], the n-state Extended Gilbert Model, and the Markov Chain Model [26], [20], [3]. In this paper, we use Extended Gilbert Model. As shown in [20], the Extended Gilbert Model achieves a good balance between model accuracy and simplicity – it is much more accurate than the 2-state Gilbert Model, while only requires n parameters to be estimated (as oppose to n^2 in the General Markov Model).

The Extended Gilbert Model [20], as shown in Figure 2, captures changes in the loss burst length. Specifically, in an n-state Extended Gilbert Model, state i ($i = 0, 1, ..., n-2$) means that there are exactly i consecutive losses since the beginning of the current loss burst, whereas state $n-1$ means that $n-1$ or more consecutive losses have occurred. The corresponding loss probability vector is $L = [0, 1, \cdots, 1]$ (*i.e.*, $\ell_0 = 0$ and

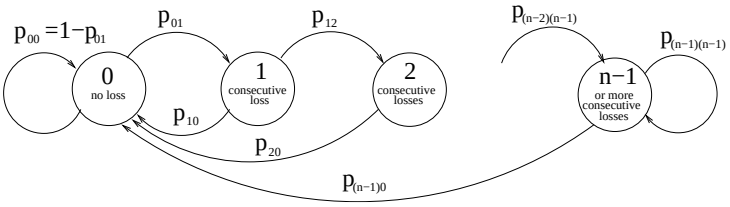

Fig. 2. The Extended Gilbert Model

for $\forall i > 0, \ell_i = 1$). Suppose the current loss burst length is i. For a newly transmitted packet, it will either get lost with probability $p_{i(i+1)}$ and cause the burst length to increment by one, or get through successfully with probability $p_{i0} = 1 - p_{i(i+1)}$ and reset the burst length to 0. No other state transitions are allowed. So the model is fully specified by n parameters p_{i0}, and the corresponding transition matrix has only $2n$ non-zero entries.

$$P = \begin{bmatrix} p_{00} & p_{10} & p_{20} & \cdots & p_{(n-2)0} & p_{(n-1)0} \\ p_{01} & 0 & 0 & \cdots & 0 & 0 \\ 0 & p_{12} & 0 & \cdots & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & p_{(n-2)(n-1)} & p_{(n-1)(n-1)} \end{bmatrix} \quad (2)$$

Why are new algorithms and techniques required? The problem of using redundancy to cope with failures has recently been considered in the context of Delay Tolerant Networks (DTNs) by Jain *et al.* [9]. They use a similar problem formulation to optimally allocate packets in an FEC group onto different paths. Given such similarity, one may be tempted to directly apply the techniques developed in [9] to the Internet. Unfortunately, several important reasons prevent us from applying these techniques to achieve high reliability in the Internet. Therefore, we have to develop new algorithms and techniques for use in SmartTunnel.

- *Burstiness in Internet packet loss.* Two different loss models are considered in [9]: independent packet loss (which exhibits no temporal loss dependency) and complete path failure (where a path delivers either all packets successfully or no packet at all). While these models may be useful in a DTN, neither model captures the commonly observed bursty loss behavior in the Internet. Since the goal of SmartTunnel is to minimize loss in the Internet context, it is essential to use models that capture the Internet loss behavior more accurately.
- *The use of different performance metrics.* Jain *et al.* [9] try to minimize the expected FEC *group* loss probability (as opposed to the *packet* loss probability), which is defined as the probability for FEC to be unable to recover an entire FEC group (*i.e.*, $Pr[X + Y > R]$). While this metric captures the performance of non-systematic FEC codes, it is too conservative for systematic FEC codes. As noted above, a major advantage of systematic FEC codes is that even if the entire group cannot be recovered (*i.e.*, $X + Y > R$), all the unmodified data packets that arrive successfully are still available. To capture the performance of systematic FEC codes, we use a new metric ($\bar{X}_{\text{FEC}}$), which is more difficult to optimize and calls for the development of new optimization algorithms (in Section IV-B).

Later in Section V, we will thoroughly compare our algorithms with the algorithms proposed in [9] along with several other

baseline algorithms, and show that our algorithms significantly out-perform the existing ones under bursty losses.

B. Allocation Algorithms

We decompose the original optimal allocation problem into the following two sub-problems:

1. *Given an allocation $\{(d_i, r_i)\}$, how to compute $\bar{X}_{\rm FEC}$, the expected number of lost data packets after applying FEC?* This is challenging because random variables X and Y are convolutions of random variables x_i and y_i (*i.e.*, the numbers of lost data and redundant packets on path i) and have no close form in general. In Section IV-B.1, we address the challenge by approximating the joint distribution of X and Y as a bivariate normal distribution.
2. *How to find an allocation $\{(d_i, r_i)\}$ that minimizes $\bar{X}_{\rm FEC}$?* The key challenge here is the enormous search space. Given D data packets and R redundant packets, there are N^{D+R} different ways of allocating them onto N different physical paths. For even moderate FEC group sizes, this is already a too big search space for a brute-force approach to work (e.g., 20-packet FEC group using 3 paths has 3486784401 combinations!). To address the issue, we develop an efficient dynamic programming algorithm to find an optimal allocation in Section IV-B.2.

1) Approximating $\bar{X}_{\rm FEC}$: Our first task is to estimate $\bar{X}_{\rm FEC}$, the expected number of lost data packets after applying FEC under a given allocation $\{(d_i, r_i)\}$. As noted above, our basic strategy is to approximate the joint distribution of X and Y with a bivariate normal distribution. Such normal approximation is reasonable when the number of independent paths is large and the allocation (d_i, r_i) on each path is relatively small compared to (D, R). In addition, our experience suggests that even when these conditions do not hold, the allocation obtained using the normal approximation tends to work well in practice. Similar positive experience has been reported in [9].

We use Extended Gilbert Model to model packet loss. Given allocation $\{(d_i, r_i)\}$, we can derive $E[x_i]$, $E[y_i]$, $V[x_i]$, $V[y_i]$, and $cov[x_i, y_i]$ from the transition matrix of the model. Because of space limitation, a detail derivation is shown in our technical report [14]. Our analysis takes into account the fact that different values of (d_i, r_i) can affect the burstiness of the packet loss observed on a path i. This phenomenon has been reported by several recent studies. For example, the authors in [22] show that burstiness of probing traffic significantly affect the burstiness in the observed loss rates. As the inter-arrival time of probing packets increases, the loss becomes less bursty.

Based on $E[x_i]$, $E[y_i]$, $V[x_i]$, $V[y_i]$, and $cov[x_i, y_i]$, we then compute the statistics of the total numbers of lost data and redundancy packets as follows. Let μ_X and σ_X denote the mean and standard deviation of X, respectively. Let μ_Y and σ_Y denote the mean and standard deviation of Y, respectively. Let $\sigma_{XY} = cov[X, Y]$ be the covariance of X and Y. In order to derive μ_X, σ_X, μ_Y, σ_Y, and σ_{XY} as functions of a given allocation $\{(d_i, r_i)\}$, we will assume that *losses on different paths are independent from each other*. Later in Section IV-E, we will discuss techniques that can be used to detect and remove paths with shared congestion. Under the independence assumption, we have

$$\mu_X = \sum_i^N E[x_i], \qquad \sigma_X^2 = \sum_i^N V[x_i]$$
$$\mu_Y = \sum_i^N E[y_i], \qquad \sigma_Y^2 = \sum_i^N V[y_i]$$
$$\sigma_{XY} = \sum_i^N cov[x_i, y_i]$$

.

We can then approximate the joint distribution of X and Y by a bivariate normal distribution with probability function

$$P(x, y) = \frac{1}{2\pi\sigma_X\sigma_Y\sqrt{1-\rho^2}} \exp\left[-\frac{z}{2(1-\rho^2)}\right], \quad (3)$$

where

$$z \equiv \frac{(x-\mu_X)^2}{\sigma_X^2} - \frac{2\rho(x-\mu_X)(y-\mu_Y)}{\sigma_X\sigma_Y} + \frac{(y-\mu_Y)^2}{\sigma_Y^2},$$

and

$$\rho \equiv cor[X, Y] = \frac{\sigma_{XY}}{\sigma_X\sigma_Y}$$

is the correlation of X and Y.

In the special case when $\rho = 0$ (*i.e.*, X and Y are independent), (3) can be further simplified into

$$P(x, y) = \frac{1}{2\pi\sigma_X\sigma_Y} \mathbf{e}^{-\frac{(x-\mu_X)^2}{2\sigma_X^2} - \frac{(y-\mu_Y)^2}{2\sigma_Y^2}}. \quad (4)$$

Our experience through extensive simulations suggests that the correlation of X and Y (*i.e.*, ρ) is often very small when multiple paths and sufficiently large FEC group size are used. For practical purposes, its effect can be safely ignored compared to the effects of μ_X, σ_X, μ_Y, and σ_Y. *We therefore will use (4) in the rest of the paper in the interest of simplicity*.

By replacing the discrete summation in (1) with a continuous integral, we can then approximate $\bar{X}_{\rm FEC}$ by

$$\bar{X}_{\rm FEC}^{\approx} \equiv \int_{x=0}^{D}\int_{y=R-x}^{\infty} x\, P(x, y)\; dy\; dx \quad (5)$$
$$= \int_{x=0}^{D}\int_{y=R-x}^{\infty} \frac{x}{2\pi\sigma_X\sigma_Y} \mathbf{e}^{-\frac{(x-\mu_X)^2}{2\sigma_X^2} - \frac{(y-\mu_Y)^2}{2\sigma_Y^2}}\; dy\; dx \quad (6)$$
$$= \int_{x=0}^{D} \frac{\mathbf{e}^{-\frac{(x-\mu_X)^2}{2\sigma_X^2}}}{2\sqrt{2\pi}\sigma_X} x \left(1 + \text{erf}\left[\frac{\mu_Y + x - R}{\sqrt{2}\sigma_Y}\right]\right) dx \quad (7)$$

where $\text{erf}(z) = \frac{2}{\sqrt{\pi}}\int_0^z \mathbf{e}^{-t^2} dt$ is the error function [25]. We can then numerically evaluate $\bar{X}_{\rm FEC}^{\approx}$ using standard software package such as Matlab.

2) Dynamic Programming Based Solution: Our second major task is to find an optimal allocation $\{(d_i, r_i)\}$ that minimizes $\bar{X}_{\rm FEC}^{\approx}$ as defined in (7). The main challenge here is that $\bar{X}_{\rm FEC}^{\approx}$ does not have close-form and cannot be easily transformed into simple objective functions. Fortunately, from (7) we can show that $\bar{X}_{\rm FEC}^{\approx}$ is monotonically decreasing with respect to both σ_X and σ_Y.

Our high-level approach to traffic allocation under no capacity constraints is as follows. We enumerate all possible values of μ_X, and use dynamic programming to find the data allocation that results in the minimum σ_X for each given

μ_X. Similarly, for all possible values of μ_Y, we determine the allocation of redundancy packets that minimizes σ_Y for each given μ_Y. Then we plug all $(\mu_X, \mu_Y, \sigma_X, \sigma_Y)$ into Equation 7, and find the allocation that minimizes $\bar{X}^{\approx}_{\rm FEC}$. The monotonicity of $\bar{X}^{\approx}_{\rm FEC}$ with respect to σ_X and σ_Y ensures that the final solution gives the best possible $\bar{X}^{\approx}_{\rm FEC}$. Later we will extend the idea to handle capacity constraints.

Below we first describe how to determine the allocation that minimizes σ_X and σ_Y given μ_X and μ_Y. Then we show how to use these solutions to solve the original allocation problem.

Subproblem: Variance Minimization. Let $E\{x_i|d_i = k\}$ and $V\{x_i|d_i = k\}$ denote the average and variance of data losses on path i when path i is allocated $d_i = k$ data packets. To apply dynamic programming, we need integer values of $E\{x_i|d_i = k\}$ and $V\{x_i|d_i = k\}$. So we scale them by λ. Let $ed[i,k] \equiv \lfloor E\{x_i|d_i = k\}\lambda \rfloor$ and $vd[i,k] \equiv \lfloor V\{x_i|d_i = k\}\lambda \rfloor$ be the scaled, discretized mean and variance of x_i, where $\lfloor \cdot \rfloor$ is the floor function (i.e., taking the largest integer no larger than the input). We can pre-compute $ed[i,k]$ and $vd[i,k]$ for $\forall 1 \le k \le D$ and $\forall 1 \le i \le N$.

We define a *data loss variance minimization* problem $\mathbf{dlvm}(d,p,e,c)$ as the problem of allocating d data packets onto the first p paths to minimize the total variance while satisfying capacity constraints c and the constraint that the total mean is e. Formally,

$$\mathbf{dlvm}(d,p,e,c): \text{minimize} \quad \sum_{i=1}^{p} vd[i,d_i]$$

$$\text{subject to} \quad \begin{cases} \sum_{i=1}^{p} d_i = d \\ \sum_{i=1}^{p} ed[i,d_i] = e \\ d_i \le c[i], \forall i \end{cases}$$

Let $opt(d,p,e,c)$ be the variance achieved by an optimal solution to $\mathbf{dlvm}(d,p,e,c)$. We have

$$opt(d,p,e,c) = \min_{\substack{0 \le k \le d \\ k \le c[p]}} \{vd[p,k] + opt(d-k, p-1, e-ed[p,k], c)\}$$

So we can apply a dynamic programming algorithm to solve $\mathbf{dlvm}(d,p,e,c)$ for all $0 \le d \le D$, $1 \le p \le N$, $0 \le e \le E_{\max}$, where E_{max} controls the granularity of the solutions. The complexity for this algorithm is $O(D^2 N E_{\max})$.

Similarly, we can define a *redundancy loss variance minimization* problem $\mathbf{rlvm}(r,p,e,c)$ as the problem of allocating r redundant packets onto the first p paths to minimize the total variance subject to the constraint that the total mean is e. We can solve $\mathbf{rlvm}(r,p,e,c)$ again using dynamic programming.

Traffic allocation under no capacity constraints. We are now ready to solve the original allocation problem. Let us first consider the unconstrained allocation problem, where $c_i = \infty$. In this case, data packets can be allocated independently from redundancy packets. So the final allocation is to allocate data packets such that $\mathbf{dlvm}(D,N,e_1,\infty)$ is minimized, and allocate redundancy packets such that $\mathbf{rlvm}(R,N,e_2,\infty)$ is minimized. The algorithm is illustrated in Figure 3. It searches through all possible values of μ_X and μ_Y, and determines the allocation that minimizes σ_X and σ_Y for each given μ_X and μ_Y. Note that the discretization may introduce some error. However, our experience from extensive simulations suggests that with sufficiently large $E_{\max}$ (*e.g.,* 500), the algorithm tends to perform close to optimal.

1. solve all $\mathbf{dlvm}(d,p,e,\infty)$ and $\mathbf{rlvm}(r,p,e,\infty)$. using dynamic programming
2. **for** $e_1 = 1$ **to** $E_{\max}$
3. $\{d_i\}$ = an optimal solution of $\mathbf{dlvm}(D,N,e_1,\infty)$
4. **for** $e_2 = 1$ **to** $E_{\max}$
5. $\{r_i\}$ = an optimal solution of $\mathbf{rlvm}(R,N,e_2,\infty)$
6. compute μ_X, μ_Y, σ_X, σ_Y under $\{(d_i,r_i)\}$
7. compute $\bar{X}^{\approx}_{\rm FEC}$
8. store $\{(d_i,r_i)\}$ if it is the best so far
9. **end**
10. **end**
11. **return** $\{(d_i,r_i)\}$ that gives the minimum $\bar{X}^{\approx}_{\rm FEC}$

Fig. 3. Traffic allocation under no capacity constraints.

Traffic allocation under capacity constraints. When network paths have capacity constraints, allocation of redundancy packets is dependent on the allocation of data packets to ensure the capacity constraints are satisfied. To incorporate capacity constraints, we modify the previous algorithm by introducing a new capacity constraint c' to ensure that the rate for sending redundancy packets on path i cannot exceed the residual capacity of path i (after sending its allocated data packet d_i).

The solution to the allocation problem with capacity constraints can be sub-optimal. Due to capacity constraints, the allocation of data packets is now coupled with the allocation of redundancy packets. Such coupling further increases the search space. For efficiency, we decouple the data and redundancy allocation by first optimizing data allocation (while ignoring the capacity consumed by redundancy packets), and then optimizing redundancy packet allocation based on remaining capacity. Such decoupling may result in sub-optimal solution. In practice, however, we find through extensive simulations that the solution we obtain tends to perform close to optimal.

1. solve all $\mathbf{dlvm}(d,p,e,c)$ via dynamic programming
2. **for** $e_1 = 1$ **to** $E_{\max}$
3. $\{d_i\}$ = an optimal solution of $\mathbf{dlvm}(D,N,e_1,c)$
4. $c'_i = c_i - d_i$, $\forall 1 \le i \le N$
5. solve all $\mathbf{rlvm}(r,p,e,c')$ via dynamic programming
6. **for** $e_2 = 1$ **to** $E_{\max}$
7. $\{r_i\}$ = an optimal solution of $\mathbf{rlvm}(R,N,e_2,c')$
8. compute μ_X, μ_Y, σ_X, σ_Y under $\{(d_i,r_i)\}$
9. compute $\bar{X}^{\approx}_{\rm FEC}$
10. store $\{(d_i,r_i)\}$ if it is the best so far
11. **end**
12. **end**
13. **return** $\{(d_i,r_i)\}$ that gives the minimum $\bar{X}^{\approx}_{\rm FEC}$

Fig. 4. Traffic allocation problem under capacity constraints.

3) Packet Spreading Algorithm: In Section IV-B.1 and Section IV-B.2, we derive the traffic allocation (i.e., $\{(d_i,r_i)\}$ for each path i). For the same allocation, different ways of assigning packets onto paths can result in different observed loss burstiness. The burstiness in the observed loss increases with the burstiness in traffic. So we should try to spread the packets allocated on the same path as evenly as possible. This reduces burstiness in experienced packet losses, and enhances effectiveness of FEC. To achieve this goal, we develop a credit-based scheme to allocate traffic. Each path is associated with a credit. The path with the largest credits is selected to transmit

the next packet. The credit of path i is updated as follows. Each time a new packet is transmitted, path i earns d_i/D credits. If path i is selected to transmits a packet, it consumes 1 credit. For a given allocation, $\{(d_i, r_i)\}$, the credit-based scheme determines exactly which packets in an FEC group should be allocated onto which paths so that the final loss rate is minimized. This allocation only needs to be computed once for a given $\{(d_i, r_i)\}$, and can be cached for future packet processing. More details are described in our technical report [14].

C. Estimating Parameters for the Loss Model

The effectiveness of the above traffic allocation scheme depends on the accuracy of the loss model estimation. In our evaluation, we use extended Gilbert loss model, and estimate its transition matrix (in Equation 2), as shown in [20].

$$p_{01} = (\sum_{i=1}^{n-1} m_i)/m_0$$

$$p_{(k-1)k} = (\sum_{i=k}^{n-1} m_i)/(\sum_{i=k-1}^{n-1} m_i)$$

where m_i denotes the number of loss bursts with length i, where $i = 1, 2, ..., n-1$ and m_0 denote the number of delivered packets. Since network path properties change over time, we predict future network performance using the measurements in the previous intervals.

D. FEC redundancy adaptation

We derive the allocation scheme given the number of redundant packets R per FEC group. In practice, we can do greedy search to find the minimum redundancy packets R which can satisfy target loss rate (e.g.,1e-6).

E. Handling Shared Congestion

In the previous discussion, we consider loss rates on the physical paths are independent. In practice, we may have some paths that share a common bottleneck. In this case, the loss rates on these paths are highly correlated. To handle such cases, we can apply an existing technique to detect shared congestion. A number of techniques have been proposed for this purpose, such as cross-correlation-based approach [19], entropy-based approach [11] and wavelet-based approach [12]. We then treat the set of paths that have shared congestion as one path, and apply our traffic allocation scheme to the merged paths.

V. Performance Evaluation

In this section, we first introduce our evaluation methodology and then describe evaluation results.

A. Evaluation Methodology

We evaluate the performance of SmartTunnels using the following three ways: (i) Internet trace-driven simulation, (ii) ns-2 simulation, and (iii) experiment on PlanetLab. These three evaluation methods are complementary to each other. Trace-driven evaluation allows us to extensively evaluate the performance of SmartTunnels under realistic Internet performance characteristics; ns-2 simulation allows us to study the interactions between multiple tunnels in a controlled environment; and real experiment allows us to understand the benefit and overhead of SmartTunnels in a real network.

We compare the following traffic allocation schemes:

- **SmartTunnel:** This is the algorithm we describe in Section IV.
- **Markowitz numeric (MkwNu):** This is the algorithm proposed in [9]. It maximizes the Sharpe-Ratio by solving a series of quadratic optimization problems.
- **Round robin (RR):** Traffic is assigned to multiple physical paths in a round robin fashion.
- **Greedy:** Traffic is assigned to the path that has the lowest loss rate. When multiple paths experience the same loss rate, one path is selected randomly among them.

B. Trace-driven Simulation

We collect Internet traces by sending 16-byte ICMP echo packets from 57 hosts on PlanetLab to 55 popular Web sites, selected from the 100 popular websites listed at [1]. We run zing and tcpdump concurrently on each PlanetLab host. To capture bursty loss behavior, zing is modified to generate ICMP echo packets with an inter-packet arrival of 2 ms. Tcpdump is used to captured ICMP echo-reply packets. In order to avoid PlanetLab hosts to drop packets when the probing traffic are too bursty. We introduce 1 second idle time every 1 second bursty traffic. Each measurement experiment lasts at least 800 seconds. In our measurement, 78.5% of paths have loss rates below 2%. The mean loss rates of these paths is 0.0175.

Each trace is divided into 20 intervals, so each interval is a 40-second trace. We apply different traffic allocation schemes on each 40-second interval. For all the evaluation, we use FEC group size of 40 packets (including data and redundancy packets), and adapt traffic allocation every interval. Two kinds of evaluation results are shown. One is oracle result in which we assume current network path performance is known and there is no prediction errors. The other one is prediction result in which current network path performance is predicted from previous intervals. Table II shows probabilities of SmartTunnel to achieve loss free reliability. When there is no prediction error, SmartTunnel can achieve loss free reliability with probability up to 0.9991 if it uses 6 paths. It can also achieve loss free reliability with probability 0.94 even with only 2 paths. If we consider prediction errors, SmartTunnel can also achieve loss free reliability with probability from 0.85 to 0.93.

	2Path	3Path	4Path	6Path
Oracle	0.94	0.9852	0.996	0.9991
Prediction	0.85	0.88	0.91	0.9267

TABLE II
PROBABILITIES OF SMARTTUNNEL TO ACHIEVE LOSS FREE RELIABILITY

1) Oracle results: We compare different traffic allocation schemes by varying the number of available physical paths, redundancy level used in FEC, and quality of the paths. For each experiment configuration (e.g., a given number of paths to the same website and a given combination of path property), we conduct 20 random runs (i.e., selecting 20

different combinations of traces used for evaluation), and report the summary statistics from these runs.

To systematically study the performance, we categorize results into different scenarios based on the number of low loss paths selected. Low loss paths are paths whose loss rates are below 2%. This classification is used in [28]. Table III and Table IV compare percentages of intervals to achieve free loss reliability among different algorithms. Let K denote the number of redundant packets and G denote the number of low loss paths chosen. We make the following observations. First, in all cases SmartTunel is the best performing algorithm. Second, when all paths are low loss paths, SmartTunel can achieve loss free reliability for 94.47%-99.81% of time intervals. Third, SmartTunnel more effectively uses paths with high loss rates. For example, when all paths are high loss paths, SmartTunnel can achieve loss free reliability for around 6%-35% more time intervals compared to other algorithm. It is interesting that when high loss paths are selected, Greedy algorithm is almost the second best algorithm. Fourth, in those cases with three physical paths selected, the difference between various traffic allocation schemes becomes smaller when the number of redundancy packets increases. This suggests that the choice of traffic allocation is more important when the network bandwidth is limited and the number of redundancy packets is small.

Table V and Table VI show mean loss rates of SmartTunnel with different number of low loss paths selected and different number of redundant packets. We can derive the expected loss rate of N-Path SmartTunnel from these tables.

Let P denote the probability that a selected path has loss rate lower than 2%, $P_r(G = i)$ denote the probability that exactly i low loss rate paths are selected, and $R_N(G = i, K = j)$ denote the mean loss rate of a SmartTunnel when $G = i$, $K = j$, and N physical paths are used. In our measurement, $P = 0.785$. $R_N(G = i, K = j)$ can be lookup from Table V and Table VI. Then we can compute the expected loss rate $L_{N,K=j}$ of N-path SmartTunnel using K redundant packets as follows.

$$\begin{aligned} L_{N,K=j} &= \sum_{i=0}^{N} R_N(G=i,K=j) * P_r(G=i) \\ &= \sum_{i=0}^{N} R_N(G=i,K=j) * C_N^i * P^i * (1-P)^{N-i} \end{aligned}$$

For example, when N = 2 and K = 7, we can get $R_2(G = 0, K = 7) = 0.0101$, $R_2(G = 1, K = 7) = 0.0009$ and $R_2(G = 2, K = 7) = 0.0001$ from Table V. Based on the equation, we can get the expected loss rates of 2-Path SmartTunnel using 7 redundant packets $L_{2,K=7} = 8\times10^{-4}$. Similarly, the expected loss rate of 3-Path SmartTunnel can be as small as 5×10^{-5} when K is 14. Therefore we conclude that SmartTunnel can achieve around four 9s reliability with a small number of paths and redundant packets.

	K=7	K=10	K=14
G = 0	0.0101	0.0071	0.0054
G = 1	0.0009	0.0004	0.0003
G = 2	0.0001	2.6×10^{-5}	1.6×10^{-5}

TABLE V
MEAN LOSS RATES OF SMARTTUNNEL USING 2 PATHS

	K=7	K=10	K=14
G = 0	0.0067	0.0046	0.0031
G = 1	0.0003	0.0002	0.0001
G = 2	0.0001	2.6×10^{-5}	1.2×10^{-5}
G = 3	2.5×10^{-5}	9.8×10^{-6}	4.4×10^{-6}

TABLE VI
MEAN LOSS RATES OF SMARTTUNNEL USING 3 PATHS

2) Predictability of Path Properties: For a traffic allocation to work well, we should be able to predict future network path performance. In our technical report [14], we study the predictability of loss rates by applying Fisher exact probability test [5].

We find out that test results are not sensitive to the length of history traces. In practice, we prefer to use longer history trace to do the prediction because it is more stable. In the following evaluation, we use the loss transition matrix of previous 320 second trace (8 intervals) as the prediction of the current interval.

3) Trace-driven evaluation results: Table VII and Table VIII show trace-driven results with prediction. SmartTunnel out-performs the other schemes in all scenarios except one in which there are three low loss rate paths and K is 14. In this case, the reliablity of SmartTunnel reduces from 99.81 % (in oracle) to 97.64% due to prediction errors while prediction errors do not affect the performance of round robin.

C. NS-2 Simulation

In this section, we study the interactions between multiple SmartTunnels using ns-2 simulations. Figure 5 shows the network topology used in the evaluation. Both senders use 3 redundancy packets per FEC group, where each FEC group consists of 13 packets in total. Figure 6 shows the evolution of loss rates experienced by 2 SmartTunnels that share physical paths. To stress test, we initialize their allocation to both use path A-B-D. Due to poor initial allocation, both tunnels initially experience high loss rates before and after FEC. This also highlights the importance of appropriate traffic allocation on end-to-end reliability. Then the two tunnels continuously adapt their traffic allocation according to their monitored performance every time interval. At interval 7, both tunnels converge to low loss rates before FEC, and close to 0 loss rate after applying FEC. Figure 7 further plots the data allocation of two tunnels on these paths. As we can see, they converge to an even share of network resources, both allocating 5 data packets on two paths. Similar fair allocation is achieved for redundancy packets (not shown in the interest of space). Overall both reliability and fairness are achieved.

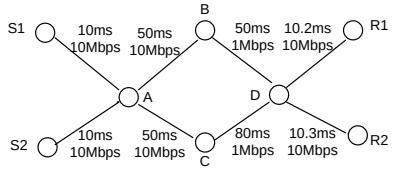

Fig. 5. Network topology used in n2-simulation. Two tunnels S1-R1 and S2-R2 share two physical paths. S1 sends 0.7 Mbps CBR traffic, and S2 sends 0.5 Mbps CBR traffic. Low-rate Pareto traffic is introduced as background traffic on links BD and CD. In addition, links BD and CD use Gilbert-loss models to drop traffic, where the loss transition matrix at B is [0.985 0.015; 0.45 0.55], and that at C is [0.99 0.01; 0.35 0.65].

	K = 7			K = 10			K = 14		
	G=2	G=1	G=0	G=2	G=1	G=0	G=2	G=1	G=0
SmartTunnel	94.47%	79.45%	31.12%	98.88%	89.62%	52.95%	99.09%	93.81%	63.30%
MkwNU	87.91 %	75.40%	25.51%	95.36%	84.58%	40.84%	98.20%	90.93%	53.40%
RR	82.82%	48.09%	16.73%	91.52%	64.51%	40.11%	92.04%	74.80%	49.53%
Greedy	88.86%	77.71%	27.85%	94.79%	85.87%	44.53%	97.80%	90.60%	53.21%

TABLE III

PERCENTAGES OF INTERVALS TO ACHIEVE LOSS FREE RELIABILITY UNDER 2 PATHS RANDOMLY CHOSEN FROM THE TRACES

	K = 7				K = 10				K = 14			
	G=3	G=2	G=1	G=0	G=3	G=2	G=1	G=0	G=3	G=2	G=1	G=0
SmartTunnel	98.71 %	96.80 %	88.20 %	48.82 %	99.61 %	98.82 %	93.58 %	65.86 %	99.81 %	99.62 %	97.25 %	77.84 %
MkwNU	96.35 %	93.84 %	84.96 %	33.09 %	98.80 %	97.26 %	90.76 %	47.73 %	99.67 %	99.02 %	94.78 %	64.79 %
RR	90.53 %	64.51 %	35.73 %	14.77 %	93.84 %	79.05 %	55.22 %	33.27 %	99.78 %	96.93 %	82.88 %	61.60 %
Greedy	96.00 %	94.49 %	86.08 %	42.82 %	97.73 %	96.60 %	90.43 %	53.23 %	98.92 %	98.44 %	93.94 %	61.08 %

TABLE IV

PERCENTAGES OF INTERVALS TO ACHIEVE LOSS FREE RELIABILITY UNDER 3 PATHS RANDOMLY CHOSEN FROM THE TRACES

	K = 7			K = 10			K = 14		
	G=2	G=1	G=0	G=2	G=1	G=0	G=2	G=1	G=0
SmartTunnel	90.46 %	77.25%	27.76%	96.18%	87.12%	50.21%	97.35%	91.25%	59.25%
MkwNU	83.66 %	73.53%	20.47%	91.27%	83.30%	37.47%	96.22%	89.25%	49.81%
RR	82.76%	48.19%	17.38%	91.36%	64.41%	39.89%	91.95%	74.45%	48.87%
Greedy	84.85%	76.18%	27.10%	90.82%	85.07%	42.63%	95.15%	89.22%	52.26%

TABLE VII

PERCENTAGES OF INTERVALS TO ACHIEVE LOSS FREE RELIABILITY UNDER 2 PATHS RANDOMLY CHOSEN FROM THE TRACES

	K = 7				K = 10				K = 14			
	G=3	G=2	G=1	G=0	G=3	G=2	G=1	G=0	G=3	G=2	G=1	G=0
SmartTunnel	94.42 %	92.31 %	85.46 %	41.77 %	96.39 %	95.39 %	90.49 %	58.18 %	97.64 %	97.06 %	94.14 %	71.08 %
MkwNU	91.32 %	89.31 %	82.61 %	25.77 %	95.03 %	93.61 %	88.79 %	40.68 %	97.22 %	96.28 %	93.31 %	59.67 %
RR	90.52 %	64.48 %	35.43 %	15.27 %	93.84 %	78.54 %	54.90 %	33.05 %	99.76 %	96.92 %	82.46 %	61.31 %
Greedy	91.37 %	90.62 %	84.69 %	39.36 %	93.82 %	93.37 %	89.03 %	49.23 %	96.09 %	95.56 %	92.42 %	57.14 %

TABLE VIII

PERCENTAGES OF INTERVALS TO ACHIEVE LOSS FREE RELIABILITY UNDER 3 PATHS RANDOMLY CHOSEN FROM THE TRACES

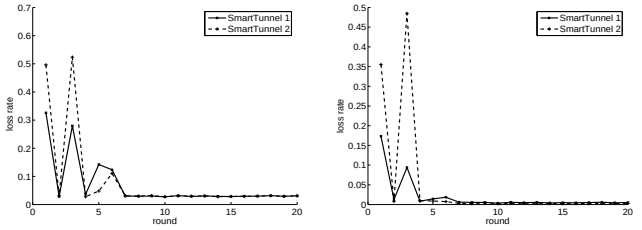

Fig. 6. (a) Loss rates before applying FEC. (b) Loss rates after applying FEC.

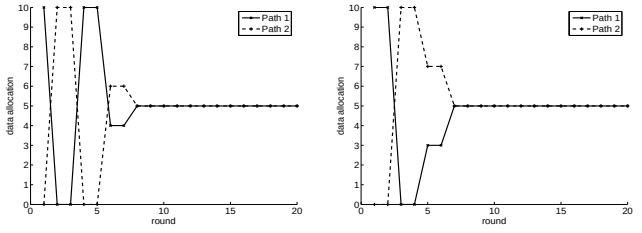

Fig. 7. (a) S1-R1 data allocation on 2 paths.(b) S2-R2 data allocation on 2 paths.

D. Experiments on PlanetLab

We implement SmartTunnel using click [13] on PlanetLab. The following elements are added to click to provide SmartTunnel functionalities: (i) monitors at both sender and receiver to cooperatively monitor network performance using either active probing or passive probing, (ii) a traffic distributor that stripes traffic according to the controller's specification, (iii) an encoder and decoder that apply FEC encoding /decoding and add/remove SmartTunnel header, and (iv) packet buffering. Figure 8 shows the diagram of different components in our implementation. ToSocket and FromSocket in the figure are the existing elements in click to provide sending and receiving functionalities, and the other elements in the figures are what we implement. At the sender side, the controller, implemented outside the click, coordinates with different click elements by specifying monitoring instructions, an FEC coding scheme, a traffic allocation scheme. The sender side receives and encodes data from the upper layer and stripes them onto multiple physical paths. The receiver logic is much simpler: it decodes and buffers packets before delivering them to the upper layer. The monitor at the receiver side responds to active probes from the sender, and also periodically sends back performance information for the paths that carry traffic.

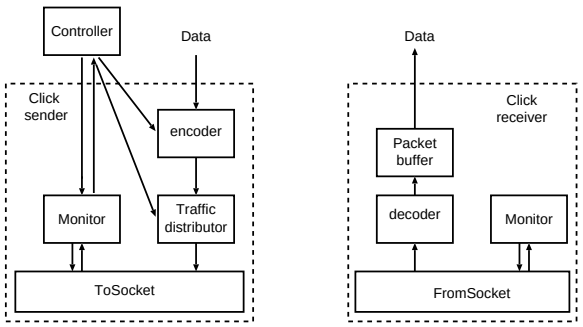

Fig. 8. SmartTunnel implementation in click.

In the experiment, we construct a SmartTunnel on top of

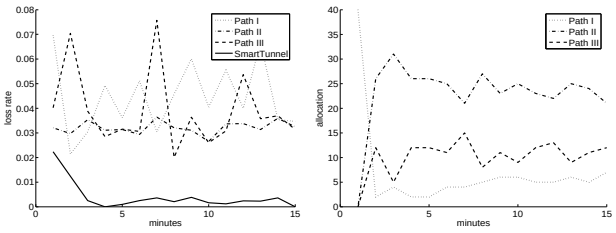

Fig. 9. (a)Evolution of loss rates before and after FEC (b) Evolution of traffic allocation across three paths

three overlay paths between two hosts. The SmartTunnel uses 10 redundancy packets per FEC group (each group with 40 packets). Figure 9 (a) shows 15 minutes time series of path loss rates before FEC and SmartTunnel loss rate after FEC. As we can see, these three paths experience substantial loss rates. Figure 9 (b) further shows the traffic allocation on these paths over time. Initially all traffic are allocated on Path I. FEC does not work well because the loss is too bursty. Every one minute, SmartTunnel computes the new traffic allocation based on its observed performance. Among these three paths, Path III has lowest loss rate and smallest loss variance. SmartTunnel put around 60 % traffic on it. Instead of putting all traffic on the best path, SmartTunnel also uses worse paths (i.e, put around 13% traffic on Path I). After 4 minutes, SmartTunnel achieves almost full reliability.

E. Summary

To summarize, in this section we evaluate the performance of SmartTunnels using trace-driven simulation, ns-2 simulation, and PlanetLab experiments. Our results show that SmartTunnel can achieve high reliability over a diverse set of scenarios. Moreover our initial study of interactions between multiple smart tunnels suggests that they can co-exist well. We plan to further investigate their interactions more thoroughly in the future.

VI. Conclusion

In this paper, we propose *SmartTunnel*, an end-to-end approach to achieving high reliability. It applies FEC and allocates traffic onto multiple physical paths to minimize loss rates under realistic Internet loss models. Using extensive simulation and real implementation, we demonstrate that SmartTunnel is effective in achieving high reliability.

As part of our future work, we are interested in applying SmartTunnel to wireless networks. An increasing number of wireless devices have multiple interfaces. Effectively utilizing these interfaces simultaneously has the potential to significantly improve reliability in wireless networks. Wireless link loss characteristics differ significantly from those of wireline links. In particular, network paths involving wireless links may experience extended outage periods (e.g., due to mobility or environmental changes). Such outages may span multiple FEC groups, and significantly reduce the effectiveness of FEC. We are interested in extending SmartTunnel to handle such outages in addition to bursty losses.

References

[1] 100 top web sites. `http://www.100hotsites.com`.

[2] A. Akella, B. Maggs, S. Seshan, A. Shaikh, and R. Sitaraman. A measurement-based analysis of multihoming. In *Proceedings of ACM SIGCOMM '03*, Karlsruhe, Germany, Aug. 2003.

[3] J.-C. Bolot. End-to-end packet delay and loss behavior in the Internet. In *Proceedings of ACM SIGCOMM '93*, San Francisco, CA, Sept. 1993.

[4] Z. Cao, Z. Wang, and E. Zegura. Rainbow fair queueing: Fair bandwidth sharing without per-flow state. In *Proceedings of IEEE INFOCOM '00*, Tel Aviv, Israel, Mar. 2000.

[5] Fisher exact probability test. `http://home.clara.net/sisa/fishrhlp.htm`.

[6] E. N. Gilbert. Capacity of a burst-noise channel. *Bell System Technical Journal*, 39:1253–1266, Sept. 1960.

[7] D. K. Goldenberg, L. Qiu, H. Xie, Y. R. Yang, and Y. Zhang. Optimizing cost and performance for multihoming. In *Proceedings of ACM SIGCOMM '04*, Portlan, Oregon, Aug. 2004.

[8] K. P. Gummadi and H. V. Madhyastha. Improving the reliability of internet paths with one-hop source routing. In *Proc. of OSDI*, Oct. 2004.

[9] S. Jain, M. Demmer, R. Patra, and K. Fall. Using redundancy to cope with failures in a delay tolerant network. In *Proc. of ACM SIGCOMM*, Aug. 2005.

[10] W. Jiang and H. Schulzrinne. Assessment of voip service availability in the current internet. In *Proc. of PAM*, Apr. 2003.

[11] D. Katabi, I. Bazzi, and X. Yang. A passive approach for detecting shared bottlenecks. In *Proc. of 10th IEEE International Conference on Computer Communications and Networks*, Oct. 2001.

[12] M. S. Kim, T. Kim, Y. Shin, S. Lam, and E. Powers. A wavelet-based approach to detect shared congestion. In *Proc. of ACM SIGCOMM*, Aug. 2004.

[13] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek. The click modular router. *ACM Transactions on Computer Systems*, pages 263–297, Aug 2000.

[14] Y. Li, Y. Zhang, L. Qiu, and S. S. Lam. Smarttunnel: A multipath aproach to achieving reliability in the internet. Technical TR-06-38, Dept. of CS, Univ. of Texas at Austin, Jul 2006. `http://www.cs.utexas.edu/ftp/pub/techreports/tr06-38.pdf`.

[15] M. Mitzenmatcher. Digital fountains: A survey and look forward. In *Information Theory Workshop*, 2004.

[16] V. Paxson. End-to-end routing behavior in the internet. In *IEEE/ACM Transactions on Networking*, 1997.

[17] L. Rizzo. Effective erasure codes for reliable computer communication protocols. *ACM Computer Communication Review*, Ap. 1997.

[18] RON measurement data. `http://nms.lcs.mit.edu/ron/data/`.

[19] D. Rubenstein, J. Kurose, and D. Towsley. Detecting shared congestion of flows via end-to-end measurement. In *IEEE/ACM Transactions on Networking*, pages 381–395, Jun. 2002.

[20] H. Sanneck, G. Carle, and R. Koodli. A framework model for packet loss metrics based on loss run lengths. In *SPIE/ACM SIGMM Multimedia Computing and Networking Conference*, Jan. 2000.

[21] S. Savage, A. Collins, E. Hoffman, J. Snell, and T. Anderson. The end-to-end effects of internet path selection,. In *Proc. of ACM SIGCOMM*, Aug. 1999.

[22] J. Sommers, P. Barford, N. Duffield, and A. Ron. Improving accuracy in end-to-end packet loss measurement. In *Proc. of ACM SIGCOMM*, Aug. 2005.

[23] L. Subramanian, I. Stoica, H. Balakrishnan, and R. H. Katz. OverQoS: An Overlay based Architecture for Enhancing Internet QoS. In *Proc. of NSDI*, Mar. 2004.

[24] H. Tangmunarunkit, R. Govindan, and S. Shenker. Internet path inflation due to policy routing. In *Proc. of SPIE ITCOM*, Aug. 2001.

[25] E. W. Weisstein. Erf. From *MathWorld* – A Wolfram Web Resource. `http://mathworld.wolfram.com/Erf.html`.

[26] M. Yajnik, S. Moon, J. Kurose, and D. Towsley. Measurement and modelling of the temporal dependence in packet loss. In *Proc. of INFOCOM 99*, Mar. 1999.

[27] M. Zhang, J. Lai, A. Krishnamurthy, L. Peterson, and R. Wang. A transport layer approach for improving end-to-end performance and robustness using redundant paths. In *USENIX Annual Technical Conference*, 2004.

[28] Y. Zhang, N. Duffield, V. Paxson, and S. Shenker. On the constancy of Internet path properties. In *Proceedings of Internet Measurement Workshop 2001*, San Francisco, CA, Nov. 2001.