1. INTRODUCTION

Network tomography is a promising technique to identify the location of IP faults. The goal of tomography is to infer the status of network internal characteristics based on end-to-end observations. In particular, binary tomography identifies the set of failed links from end-to-end path measurements. Upon detecting the failure of one or more of the monitored paths, a monitor sends its measurements to a central coordinator. The coordinator then runs the binary tomography algorithm, which takes as input the topology of the network and the status (i.e., up or down) of all monitored paths and finds the minimum set of links that explain the observations.

Unfortunately, obtaining accurate inputs for tomography algorithms is challenging on the Internet. First, active probes may interpret transient packet losses as persistent failures, which will result in incorrect inferences for the status of individual paths. Second, the network topology is dynamic, so it needs to be frequently re-measured. Third, different monitors may experience different network conditions, which will lead to inconsistencies in path status. Finally, a monitor’s report of the status of paths may never reach the coordinator. Inaccurate inputs can cause a tomography algorithm to report false alarms or inaccurate results. Our work studies techniques to address each of these issues.

In this abstract, we describe and evaluate a technique to distinguish persistent failures from transient losses. Our approach has two steps. First, we present an analytic model for transient losses in network links (Sec. 2). This model motivates a technique to distinguish transient losses from persistent failures using active probes. Second, because it is impossible to get ground truth on the Internet, we run experiments in an controlled environment on Emulab (Sec. 3). Controlled experiments allow us to validate our analytic model, and together, model and controlled experiments show that we can use the model to accurately parameterize our failure confirmation technique. We discuss our findings and directions for future work in Sec. 4.

2. MODELING TRANSIENT LOSSES

Transient packet losses make it difficult to determine the status of a path. If there were no transient losses, we could flag a path as down every time a probe was not answered. Unfortunately, this naive approach would result in many false alarms.

The usual strategy to differentiate persistent failures from transient packet losses and avoid false positives is to send confirmation probes if the answer to the original probe was not received. Suppose we send \( c \) confirmation probes on a path before declaring that a failure has occurred. If packet losses are independent, then we can estimate the probability of raising a false alarm, \( F \), as \( F = \gamma^c \), where \( \gamma \) is the path's loss rate.

On one hand, we want to increase \( c \) in order to decrease \( F \). On the other hand, we want to decrease \( c \) to decrease measurement overhead and the amount of time until the failure is detected. Given a target \( F \) (e.g., \( 10^{-6} \)), we can use an estimation of the loss rate in \( F = \gamma^c \) to calculate the minimum value of \( c \) that should be used to attain the given \( F \). This ensures that we minimize the number of probes sent. To minimize the time to detect a failure, we would also like to send the confirmation probes as fast as possible.

However, previous work has shown that packet losses on the Internet are bursty [3]. If we decrease the time between confirmation probes, \( \mu \), too much, then confirmation probes might experience the same loss burst, which would increase the number of false alarms and make the relation \( F = \gamma^c \) false.

To account for bursty losses on Internet links, we model transient packet losses using a Gilbert model. In a Gilbert model, links are either in a good state where all transmissions are successful or in a bad state where all packets are dropped. Links can change state every time a new packet arrives. We call the probability of staying in the bad state (i.e., the probability of losing a packet given that the previous packet was lost) the burst factor, and denote it by \( b \). Given a link’s \( \gamma \) and \( b \), we can calculate the probability of changing from the good state to the bad state (i.e., the probability of losing a packet given that the previous was transmitted).
In this model, the number of packets in a loss burst follows a geometric distribution. If two confirmation probes have \( k \) packets between them, we can use the geometric distribution to calculate the probability that both probes fall on the same loss burst. Similarly, if we want two confirmation probes to have a probability \( P \) of being in the same loss burst, we can invert the geometric distribution to calculate how many \( k \) packets should be between the probes. This is the first step in calculating the time between confirmation probes, \( \mu \).

The second step in calculating \( \mu \) is converting \( k \) to time units. This depends on how much time it takes for \( k \) packets to traverse the link. The number of packets per second can be calculated from the link speed, average packet size, and utilization. Note that these characteristics can vary by orders of magnitude depending on link type. For example, DSL links might need much higher values of \( \mu \) than fiber links.

In a real deployment, \( F \) and \( P \) should be small values (e.g., \( 10^{-6} \)). The loss rate \( \gamma \) on each path can be estimated from the monitor’s probes or with a loss-specific measurement technique [2, 3]. The burst factor \( b \) and the number of packets per second on a link are hard metrics to estimate; however, we can use conservative estimates and still obtain useful values for \( \mu \). Using this model we can calculate both \( c \) and \( \mu \), differentiating packet losses from failures quickly (small \( \mu \)) and minimizing the measurement overhead (small \( c \)).

3. CONTROLLED EXPERIMENTS

Here we show results obtained with experiments in the Emulab controlled environment. We modified Linux to inject random (i.e., Poisson) and bursty (i.e., Gilbert) losses on links. We vary the loss rates between 1% and 10%, and burst factors between 50% and 95%. We show results for a sequence of 25000 probes.

We analyze first a single link with random losses. In this scenario, confirmation probes are always independent and the interval between probes (\( \mu \)) does not impact the false positive ratio. Our experiments show that the false positive rate increases by 0.3% (i.e., almost no difference) if \( \mu \) equals zero instead of 0.2 seconds. Our empirical false positive rates obtained with the experiments are close those from the analytic model (\( F = \gamma^c \)), they fit with a R2 of 0.96.

Fig. 1 shows the false positive rate when varying the burst factor of the Gilbert model. We use the Abilene topology and a fixed per-link loss rate of 10%. Results for other loss rates are similar. When \( c \) is zero or one, the false positive rate is independent of the burst factor (i.e., a straight line) because there are not enough confirmation probes to get correlated in the same loss burst. For \( c = 2 \) and \( c = 3 \) we show results for \( \mu = 0 \) and \( \mu = 0.2 \) seconds. Again, results for \( \mu = 0.2 \) are independent of the burst factor because the probabil-

![Figure 1: Impact of Loss Burst Length on Confirmation Probability](image)

ity of a loss burst lasting 0.2 seconds in our setup is close to zero. However, for \( \mu = 0 \), the false positive rate increases with the burst factor, as the probability of confirmation packets being caught in the same loss burst increases. Note that it is better to send two spaced confirmation packets than three immediate ones.

These results validate our model and motivate specifying confirmation probes. They also show that the model can be used to find good values for \( c \) and \( \mu \).

4. DISCUSSION AND FUTURE WORK

We have shown how to avoid false positives due to packet losses when monitoring path failures. Our loss confirmation technique is motivated by a model that aids us in selecting the number of confirmation probes and the time interval between them. The model was validated with controlled experiments on Emulab.

Next, we plan to study other practical issues to the deployment of tomography techniques. Besides packet losses, we believe that topology changes due to rerouting is another significant challenge. We can reduce the amount of redundant traceroute probes used to map the topology [1], but how to efficiently update the topology as it changes is still an open problem. Finally, we want to study the use of spoofed probes in network tomography to diminish probing cost and improve identification of unidirectional failures.

Acknowledgements

This work was supported by the European Community’s Seventh Framework Programme (FP7/2007-2013) no. 223850 and by Thomson.

5. REFERENCES