

An IXA-Based Network Measurement Node

Ramaswamy Ramaswamy, Ning Weng, and Tilman Wolf
 Department of Electrical and Computer Engineering
 University of Massachusetts Amherst

Abstract—Passive network measurement through packet trace collection is an important technique for understanding the fundamental behavior of increasingly complex computer networks. Our research aims at exploring a next generation of distributed and high-performance passive measurement infrastructure. The challenges of such a system lie in the large amounts of measurement data that are generated through monitoring of Gigabit links and the need for complex processing to extract relevant information. To alleviate these problems, we have developed a network processor based packet capture system that can preprocess packet traces on the measurement node before storing the trace in a database. The network processor implements several important functions: (1) variable-length protocol header parsing, (2) prefix-preserving address anonymization, and (3) online statistics collection. To support online anonymization at Gigabit link speed, we have developed a new algorithm that outperforms existing solutions by two orders of magnitude. This anonymization algorithm employs top-hashing and subtree replication to replace online cryptographic computations with simple lookup operations. We have implemented a prototype packet capture node on the Intel IXP2400 network processor and shown its operation at a Gigabit data rate in simulation as well as in hardware.

I. INTRODUCTION

The complexity of network systems and the heterogeneity of end systems will make networks increasingly difficult to manage. To understand the operational details of networks it is imperative that sufficient information on their behavior is available. This can be achieved through network measurement.

Passive network measurement systems typically collect packet traces that are then stored in trace databases. To extract information on the state of the network, the traces are searched and post-processed. In our work, we envision two extensions to this approach:

- **Distributed Measurement Nodes.** To provide a richer set of network management applications and traffic profiling capabilities, traffic is collected and correlated from multiple measurement nodes.
- **Preprocessing of Trace Data.** Scalability in distributed measurement is a key problem. The aggregate bandwidth of trace data from multiple measurement nodes can easily overwhelm a conventional database system. To alleviate this problem, we preprocess packet traces on the measurement node and perform simple statistics collection online.

The basic architecture of our measurement system is shown in Figure 1. In this paper, we discuss how to implement the packet capture and online preprocessing functions of this system, which can be performed on dedicated measurement systems or on network systems that also implement other networking functions. Since network processors are ideal platforms for the proposed measurement system, we use the IXP platform for this research.

The measurement node performs three functions:

- **Packet Capture and Header Parsing:** Each packet is parsed to determine the sequence of network headers that are present. This allows the consideration of nested protocol headers as well as different header sizes due to options.

- **Anonymization:** To ensure the privacy of network users, IP addresses are anonymized.

- **Online Queries and Statistics Collection:** Packet traces can be preprocessed on the measurement node to reduce the load on the centralized collection system. If traffic statistics match a particular query, a response is pushed from the measurement node.

The packet trace that is generated on the measurement node is sent to the centralized trace collection node to be stored in the trace database.

The remainder of this paper described in more detail the system aspects of this measurement node architecture. Section II discusses related work. Section III describes the IP address anonymization process, which is specially adapted for online operation on a network processor. The prototype implementation of our system is discussed in Section IV and future work concludes this paper in Section V.

II. RELATED WORK

The field of network measurement has a long history, and our work builds on results from a number of these efforts. Traditionally, two approaches have been taken towards network measurement: active and passive [2].

In the active approach, a sender and/or receiver measure and record the traffic that they send/receive, obtaining end-to-end (e.g., path) characteristics [14], [15], [23]. NLANR's AMP (Active Measurement Project) and Surveyor are large-scale measurement infrastructures that perform such active measurements. The performance metrics of interest for these and other measurement infrastructure are currently being defined by the IETF IPPM Working Group [7].

In the passive approach, measurements are taken at a given point in a network and are typically used to characterize local characteristics of the network and its traffic. For example, SNMP provides data from the SNMP-equipped devices that can then be used for management purposes (e.g., fault detection). Traces of packets passing through a passive measurement point can be analyzed for traffic mix (e.g., protocol or application) [4], [9], [17], [10], packet sizes [21], or “flow” size and burstiness [17], [1], [20]. The analyzed traffic is generated by users of the network and the privacy of their communications needs to be protected. These and other past efforts have provided tremendous insight into today's networks. The passive measurement projects that are most closely related to our proposed efforts here are Sprint's IPMON project [6], AT&T's Gigascope project [5] and CAIDA's passive measurement efforts [3]. The Windmill project combines passive and active measurement and integrates routing protocol information [8]. Sprint's IPMON project pioneered the development and use of a continuous-monitoring measurement infrastructure. None of these efforts, however,

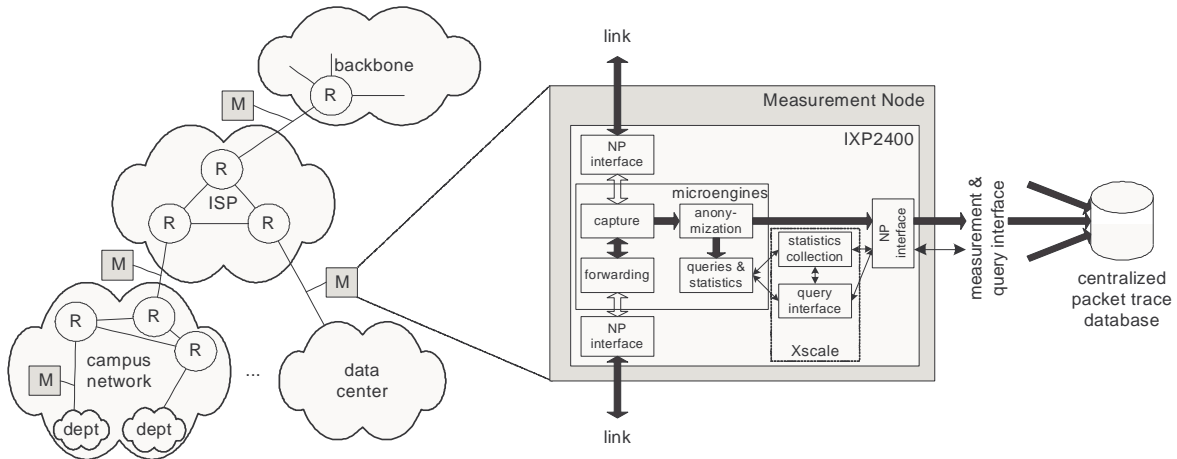


Fig. 1. Network Measurement Architecture.

leverage the use of network processors, which allow customized online queries. In this context the use of network processors is particularly crucial as complex centralized post-processing and storage of traffic traces can be off-loaded into the measurement node.

Our work develops a distributed network measurement infrastructure that uses network processors for packet capture and pre-processing (e.g., prefix-preserving IP address anonymization). This research extends the scope of network measurement such that it can be potentially integrated transparently on any IXA-based system.

III. IP ADDRESS ANONYMIZATION

Many personal and business transactions are performed over the Internet and it is imperative that the privacy of a network user is maintained. The IP addresses reveal the source and destination of each communication and it is a simple exercise to determine, for example, who is browsing which web servers.

To ensure that no private information is revealed in a network trace, sensitive header fields need to be sanitized. In most cases these fields are the IP source and destination address, which are the focus of this paper. However, IP source and destination addresses cannot simply be removed from the trace, as they are necessary to derive any useful networking statistic from the trace. Instead, IP addresses are “anonymized.” The anonymization operation is a one-to-one mapping between the original IP address as seen on the network and the anonymized IP address that is used in the trace. By employing cryptographic hash functions, this mapping cannot be guessed easily and thus is secure in a cryptographic sense. The main constraint on the anonymization algorithm is that it should be “prefix-preserving.” This means that if two original IP addresses have a common prefix of length l , then the anonymized IP addresses should also have a common prefix of exactly length l . This ensures that addresses that are “closely” co-located in a network (e.g., in the same subnet) are also closely co-located in the anonymized trace. Thus, some information on network-level characteristics of the measured traffic can be preserved across the anonymization step.

A. Existing Anonymization Approaches

A simple way of anonymizing IP addresses is to assign an arbitrary IP address mapping. One common technique is to assign 10.0.0.1 to the first IP address observed in a trace, 10.0.0.2 to the second unique IP address and so on. Recurrences of IP addresses are mapped to the same anonymized address as the first occurrence. This method allows the identification of packets that belong to the same flow, but correlations between flows get lost. From a network management and research point of view it is desirable to identify packets that originate from the same subnet (e.g., to identify correlations in DDoS attacks).

To strike a balance between privacy and usefulness of traces, prefix-preserving anonymization has been proposed, which anonymizes IP addresses while preserving the prefix nature of IP addresses. This has been implemented in *tcpdpriv* [11] and Crypto-PAN [22]. In Crypto-PAN incremental cryptographic hash computations [19], [12] are used to determine the address mapping. In *tcpdpriv*, the anonymization involves pseudorandom functions that cause the address mappings to depend on traffic patterns and differ across traces. The (raw, anonymized) mapping pairs are stored in a table to maintain consistency. To avoid raw addresses being mapped to different anonymized addresses, this table needs to be distributed to all measurement nodes which is cumbersome and makes *tcpdpriv* unsuitable for multinode measurements. Moreover, the memory required by *tcpdpriv* depends on the number of entries in the table and grows larger as more raw addresses are seen in the trace. In [13], a high level environment which supports anonymization of both packet headers and payloads using a policy script is introduced. IP addresses are anonymized sequentially using a one-to-one mapping and are not prefix preserving. The methods proposed in that paper are suitable for offline anonymization of packet traces. A cryptography based solution to compress and anonymize packet traces is presented in [16]. However, this method is too computationally intensive to be performed online at a measurement node.

B. Precomputation

In a measurement system, it is desirable to perform trace anonymization as early in the collection process as possible.

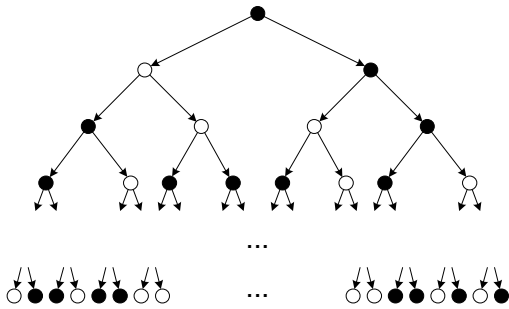


Fig. 2. Precomputation of Entire Anonymization Tree. Anonymization function is computed offline for any possible address.

By anonymizing header fields on the measurement node itself instead of external post-processing, it is less likely that unanonymized data is leaked. This requires the anonymization process to operate at a speed that can keep up with the link rates of the measurement node. This sort of *online* anonymization, however, cannot be achieved with current prefix-preserving anonymization algorithms.

The prefix-preserving anonymization presented in [22] requires up to 32 cryptographic hash (e.g., MD5) computations per IP address. In simulation on a RISC microprocessor, we have shown that this translates into 247,078 instructions. With link rates in the order of Gigabits per second it is not possible to implement the required processing in a cost-effective way. Even dedicated cryptographic hardware cannot easily achieve this processing rate. Thus, the incremental computation of the anonymization function is not a feasible approach to anonymizing IP addresses at Gigabit link speeds.

Instead, it is necessary to precompute the entire anonymization function. One way of visualizing the prefix preserving nature of anonymization it through an “anonymization tree.” The anonymization function can be seen as a function that takes a binary prefix tree as input and “flips” some of the nodes. The result of a flip is that the left child becomes the right and vice versa. This function can then be illustrated as a tree with flipped nodes in black and a non-flipped node in white [22] (see Figure 2). Note that any combination of black and white nodes in the tree yields a correct prefix-preserving anonymization function (however simplistic if all nodes are white).

With a precomputed anonymization tree, an address mapping can be determined by performing 32 lookups in the binary tree. No cryptographic computations are necessary, but the required memory is very large. While it is conceivable that next-generation measurement nodes could afford to allocate half a Gigabyte of memory to store the precomputed anonymization tree, it is still desirable to reduce the memory requirements. This is particularly important for high-performance routers with measurement capabilities because a copy of the anonymization tree is required on each port and needs to be implemented in faster, more expensive SRAM.

C. Top-Hashed Subtree-Replicated Anonymization

We have developed a novel prefix-preserving anonymization algorithm, called TSA (top-hash subtree-replicated anonymization), that addresses this problem by computing all necessary

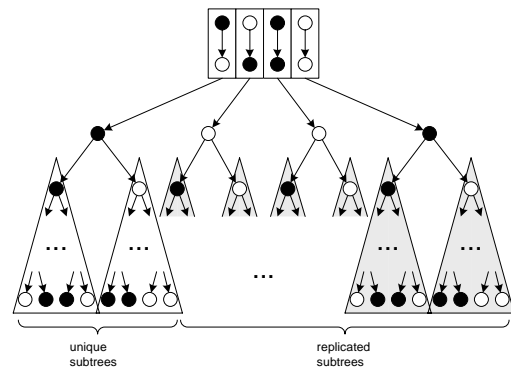


Fig. 3. TSA of Anonymization Data Structure. Tree nodes are computed offline for any possible address. Reuse of subtrees reduces memory requirements. Hashing of initial bits limits prefix preserving property to subtrees and increases resilience to attacks.

cryptographic functions offline. In TSA, the anonymization of an IP address only requires 794 instructions and 4 memory lookups (using 8-bit tries). The ability to limit the amount of memory that is necessary for TSA to a few Megabytes makes TSA an ideal algorithm to be implemented on the IXP2400.

TSA is based on three additions to conventional prefix-preserving anonymization, which address the above challenges and makes TSA a practical solution for high-performance network trace collection engines. The data structure that is used for the TSA algorithm is illustrated in Figure 3. These additions are:

- **Precomputation:** The only way to avoid *online* cryptographic processing is to precalculate the entire anonymization function. In practice a measurement node can be expected to observe IP addresses from only a small fraction of the total address space. It is still necessary to precalculate the anonymization mapping for all addresses in the address space as traffic patterns are not known beforehand. This implies that the mapping for all 4 billion possible IPv4 addresses needs to be calculated and stored. We reduce this memory requirements by replicating subtrees in the anonymization tree.
- **Subtree Replication:** Most network traffic is limited in the number of distinct addresses that can be observed. This leads to the idea of replicating anonymization subtrees instead of having a distinct anonymization subtree for each prefix. This significantly reduces the memory requirement as only the unique subtrees need to be stored. The drawback of subtree replication is that the anonymization mapping can be revealed more easily. To make it harder to identify which subtrees are duplicates of each other, we consider a modification to subtree replication, called subtree replication with hashed mapping. In this scheme, the choice of which subtree to use is determined by a cryptographic hash function which is also precomputed.
- **Top Hashing:** From a security point of view, one of the major drawbacks of prefix preserving anonymization is that there is a correlation between the original IP addresses and the anonymized addresses. If two addresses share a prefix of a certain length, they share a prefix of the same length in the anonymized trace. This of course is desirable from a networking research point of view as the prefix relation gives clues of network traffic properties. The top hashing in TSA exploits this absence

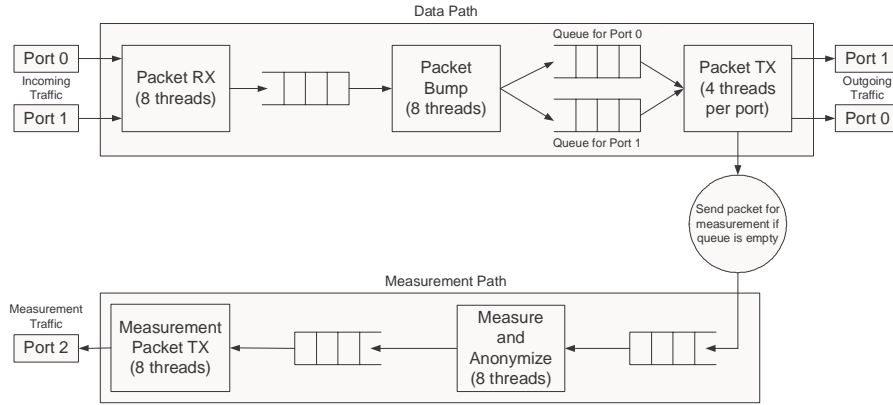


Fig. 4. Network Measurement Node on IXP2400.

of short prefixes to improve the resistance of the anonymization algorithm to attack. Instead of using an anonymization tree for the most significant bits of an address, a cryptographic hash function is used to anonymize these bits. The remaining bits are anonymized with the conventional anonymization tree. The benefit of this approach is that the hash function removes all correlation between the prefixes and thus a compromised address does not reveal information about other address prefixes. The cost of this approach is that the prefix nature of the most significant bits is not preserved, but due to the historic assignment of IP address blocks (and the resulting absence of prefixes shorter than eight bits), this has no practical impact on IPv4 network traces.

D. Anonymization Security

A major challenge that needs to be addressed in the context of anonymization is the issue of attacks on trace collection systems with the intent to compromise anonymized addresses. By injecting traffic with a particular pattern that can be identified by the attacker in the anonymized trace, the mapping between an original and an anonymized IP addresses can be established. The problem with prefix-preserving anonymization is that even a single compromised address reveals a significant amount of information about other IP address mappings.

We have developed an analytic model of the security performance of conventional anonymization and TSA under a worst-case attack scenario to evaluate the tradeoffs between the different approaches [18]. Our analytic comparison of the susceptibility to attacks between conventional anonymization and our approach shows that TSA performs better for small scale attacks and comparably for medium scale attacks. For more details see [18].

IV. MEASUREMENT NODE PROTOTYPE

The prototype implementation of the proposed measurement system is based on the IXP2400 network processor platform that is provided to us through the IXA university program. The current implementation operates at links rates of $2 \times 1\text{Gbit/s}$. The data flow and allocation of tasks to the underlying NP components is shown in Figure 4.

Table I shows the statistics for the application obtained with 8 threads running on the microengine that is performing mea-

TABLE I
APPLICATION STATISTICS.

	Anonymization	Measurement
Instruction count	67	228
Avg. cycles per packet	794	1678
SRAM accesses per packet	2	2
DRAM accesses per packet	2	3

TABLE II
MICROENGINE UTILIZATION FOR MEASUREMENT APPLICATION.

Microengine	Active %
RX processing (0:0)	30.85
Packet bump (0:1)	40.19
TX processing (0:2)	59.82
Measurement (0:3)	67.20
Measurement TX (1:0)	41.66

surement processing. The anonymization part counts statistics over the code required for anonymizing a single IP address using TSA. The measurement part counts statistics for all other processing (such as parsing of headers, updating measurement statistics, and generation of measurement traffic) that are performed on the packet as it passes through the measurement node.

The application was simulated on the IXP simulator (Developer Workbench) which is available with version 3.51 of the software development kit. Simulation traffic consisted of unidirectional (incoming on Port 0 and outgoing on Port 1) 60 byte TCP/IP packets over Ethernet. We were able to sustain a transmit rate of up to 1120Mbps on the measurement port (Port 2). Table II shows the average utilization of the five microengines used in the application under these conditions.

The application was also run on the Radisys ENP-2611 hardware. Unidirectional UDP traffic consisting of 40 byte UDP datagrams was sent into Port 0 using *iperf*. The measurement node was observed to be functional at data rates of up to 65,000 packets per second. Further testing of the measurement node at higher data rates is in progress.

V. FUTURE WORK

We are currently exploring an extension to the current measurement prototype that allows collection of online statistics and the implementation of queries to the measurement node. The key research question is what traffic statistics to collect and how this information can be accessed through the query interface. This issue is closely related to the capabilities of the underlying hardware.

We propose to implement simple counting functions on the microengines and leave more complex processing to the Xscale control processor. In particular, we consider collection of packet counts, layer 3 and 4 protocol distributions, counts of packets with special significance (e.g., TCP SYN packets to identify DoS attacks), etc. These statistics can be further extended to gather more information through (1) per-flow statistics, (2) window-based statistics, and (3) multi-resolution counters. In all three cases there is a tradeoff between memory requirements and accuracy.

For the query interface we consider two possible types of queries. Queries that “pull” information from the measurement node are comparable to those done on conventional packet trace collections. The query is sent to the system and the appropriate information is retrieved and sent in response. With the online operation of our system another type of query is possible. “Push” queries are such that they continuously monitor the packet stream (or the statistics that are derived from it). When a particular condition is matched, a response is triggered.

Finally, it is necessary to obtain accurate time information for timestamping. We are currently in the process of integrating a GPS receiver with the IXP2400 to operate an NTP-style clock synchronization mechanism on the Xscale control processor.

The result of this work has the potential to make network measurement an easy to implement feature on network processor based systems. A wide-scale use of such an online measurement platform can expand the understanding of network operation as it goes beyond conventional network management protocols.

REFERENCES

- [1] S. Bhattacharyya, C. Diot, J. Jetcheva, and N. Taft. Pop-level and access-link-level traffic dynamics in a tier-1 POP. In *Proc. of the First ACM SIGCOMM Internet Measurement Workshop*, pages 39–53, San Francisco, CA, Nov. 2001.
- [2] S. Bhattacharyya and S. Moon. Network monitoring and measurements: Techniques and experience. In *Tutorial at ACM Sigmetrics 2002*, Marina Del Rey, CA, June 2002.
- [3] CAIDA (Cooperative Association for Internet Data Analysis). *Characterization of Internet Traffic Loads, Segregated by Application*. <http://www.caida.org/analysis/workload/byapplication/>, 2002.
- [4] k. claffy, G. Miller, and T. Kevin. The nature of the beast: Recent traffic measurements from an internet backbone. In *Proc. of 1998 INET Conference*, Geneva, Switzerland, June 1998.
- [5] C. Cranor, Y. Gao, T. Johnson, V. Shkapenyuk, and O. Spatscheck. Gigascope: High performance network monitoring with an SQL interface. In *Proc. of the 2002 ACM SIGMOD International Conference on Management of Data*, page 623, Madison, WI, June 2002.
- [6] C. Fraleigh, C. Diot, B. Lyles, S. B. Moon, P. Owezarski, D. Papagianaki, and F. A. Tobagi. Design and deployment of a passive monitoring infrastructure. In *Passive and Active Measurement Workshop (PAM2001)*, Amsterdam, Netherlands, Apr. 2001.
- [7] IETF Internet Protocol Performance Metrics Working Group (ippm). <http://www.ietf.org/html.charters/ippm-charter.html>.
- [8] G. R. Malan and J. Farnam. An extensible probe architecture for network protocol performance measurement. In *Proc. of ACM SIGCOMM 98*, pages 215–227, Vancouver, BC, Sept. 1998.
- [9] S. McCreary and k. claffy. Trends in wide area ip traffic patterns: a view from Ames Interent Exchange. In *PITC Specialist Seminar on IP Traffic Modeling, Measurement and Management*, Sept. 2000.
- [10] T. McGregor, H.-W. Braun, and J. Brown. The NLANR network analysis infrastructure. *IEEE Communications Magazine*, 38(5):122–128, May 2000.
- [11] G. Minshall. *TCPDPRIV*. <http://ita.ee.lbl.gov/html/contrib/tcpdpriv.html>, 1.1.10 edition, Aug. 1997.
- [12] National Institute of Standards and Technology. *Advanced Encryption Standard (AES)*, Nov. 2001. FIPS 197.
- [13] R. Pang and V. Paxson. A high-level programming environment for packet trace anonymization and transformation. In *Proceedings of the ACM SIGCOMM Conference*, pages 339–351, Karlsruhe, Germany, Aug. 2003.
- [14] V. Paxson. End-to-end routing behavior in the internet. *IEEE/ACM Transactions on Networking*, 5(5):601–615, Oct. 1997.
- [15] V. Paxson. End-to-end internet packet dynamics. *IEEE/ACM Transactions on Networking*, 7(3):277–292, June 1999.
- [16] M. Peuhkuri. A method to compress and anonymize packet traces. In *Proc. of First ACM SIGCOMM Internet Measurement Workshop*, pages 257–260, San Francisco, USA, Nov. 2001.
- [17] D. Plonka. Internet traffic flow size analysis. Technical report, University of Wisconsin, <http://net.doit.wisc.edu/data/flow/size/>, 2000.
- [18] R. Ramaswamy and T. Wolf. High-speed prefix-preserving IP address anonymization for passive measurement systems. *under submission*.
- [19] R. L. Rivest. The MD5 message digest algorithm. RFC 1321, Network Working Group, Apr. 1992.
- [20] S. Sarvotham, R. Riedi, and R. Baraniuk. Connection-level analysis and modeling of network traffic. In *Proc. of the First ACM SIGCOMM Internet Measurement Workshop*, pages 99–103, San Francisco, CA, Nov. 2001.
- [21] K. Thompson, G. J. Miller, and R. Wilder. Wide-area internet traffic patterns and characteristics. *IEEE Network Magazine*, 11(6):10–23, Nov. 1997.
- [22] J. Xu, J. Fan, M. H. Ammar, and S. B. Moon. Prefix-preserving ip address anonymization: Measurement-based security evaluation and a new cryptography-based scheme. In *Proc. of 10th IEEE International Conference on Network Protocols (ICNP'02)*, pages 280–289, Paris, France, Nov. 2002.
- [23] M. Yajnik, S. B. Moon, J. Kurose, and D. Towsely. Measurement and modeling of the temporal dependence in packet loss. In *Proc. of IEEE INFOCOM'99*, pages 345–352, New York, NY, Mar. 1999.