Scalable Packet Classification using Distributed Crossproducting of Field Labels

David E. Taylor Jonathan S. Turner
{det3,jst}@arl.wustl.edu

Washington University in Saint Louis

Packet classification is an enabling function for a variety of applications including Quality of Service, security, monitoring, and multimedia communications. Such applications typically operate on packet flows; therefore, network nodes must classify individual packets traversing the node in order to assign a flow identifier, *FlowID*. Packet classification entails searching a set of filters for the highest priority filter or set of filters which match the packet. At minimum, filters contain multiple field values that specify an exact packet header or set of headers and the associated *FlowID* for packets matching all the field values. The type of field values are typically prefixes for IP address fields, an exact value or wildcard for the transport protocol number and flags, and ranges for port numbers.

Due to the complexity of the search, packet classification is often a performance bottleneck in network infrastructure; therefore, it has received much attention in the research community.The existing solutions explore various design tradeoffs to provide high search rates, power and space efficiency, fast incremental updates, and the ability to scale to large numbers of filters. There remains a need for techniques that achieve a favorable balance among these tradeoffs and scale to support classification on additional fields beyond the standard 5-tuple. We introduce *Distributed Crossproducting of Field Labels* (*DCFL*), a novel combination of new and existing packet classification techniques that can yield lookup performance comparable to Ternary Content Addressable Memory (TCAM) while scaling to additional filter fields and remaining memory, power, and update efficient. Like several existing approaches, we leverage observations of the structure of real filter sets in order to improve lookup performance and storage efficiency. Two key observations motivate our approach: the number of unique field values for a given field in the filter set is small relative to the number of filters in the filter set, and the number of unique field values matched by any packet is very small relative to the number of filters in the filter set. *DCFL* is also designed to take advantage of the millions of logic gates and hundreds of multi-port embedded memory blocks in the current generation of ASICs and FPGAs.

Using a high degree of parallelism, *DCFL* decomposes the multi-field packet classification problem and employs parallel search engines optimized for each filter field. Given that search techniques for single packet fields (longest prefix matching and range matching) are well-studied, the primary focus of this work is the development and analysis of a scalable technique for aggregating the results from each field search. Our approach is to perform a distributed set membership query using a network of aggregation nodes. Each query performs an intersection on the set of possible field combinations matched by the packet and the set of field combinations specified by filters in the filter set. We introduce several new concepts including field labeling, *Meta-Labeling* unique field combinations, *Field Splitting*, and optimized data structures such as *Bloom Filter Arrays* that minimize the number of memory accesses to perform set membership queries. By performing this aggregation in a distributed fashion, we avoid the exponential increase in the time or space incurred when performing this operation in a single step as in the seminal *Crossproducting* technique [2]. We also develop efficient encoding techniques for intermediate results allowing us to avoid the memory inefficiency suffered by similar techniques such as *Recursive Flow Classification* (RFC) [1].

Using a collection of 12 real filter sets and synthetic filter sets generated with the *ClassBench* tools [3], we provide an evaluation of *DCFL* performance and resource requirements for filter sets of various sizes and compositions. We show that an optimized implementation of *DCFL* can provide over 100 million searches per second and storage for over 200 thousand filters in a current generation FPGA or ASIC without the need for external memory devices.Furthermore, we show that *DCFL* retains its lookup performance and memory efficiency when the number of filters and number of fields in the filters increases. Scalability to classify on additional fields is a distinct advantage *DCFL* exhibits over existing decision tree algorithms and TCAM-based solutions. We continue to explore optimizations to improve the search rate and memory efficiency of *DCFL*. We also believe that *DCFL* has potential value for other searching tasks beyond traditional packet classification. Full technical report available at:
`http://www.arl.wustl.edu/~det3/`

SIGCOMM'04 Poster Session

1. REFERENCES

[1] P. Gupta and N. McKeown. Packet Classification on Multiple Fields. In *ACM Sigcomm*, August 1999.

[2] V. Srinivasan, S. Suri, G. Varghese, and M. Waldvogel. Fast and Scalable Layer Four Switching. In *ACM Sigcomm*, June 1998.

[3] D. E. Taylor and J. S. Turner. ClassBench: A Packet Classification Benchmark. Technical Report WUCSE-2004-28, Department of Computer Science & Engineering, Washington University in Saint Louis, May 2004.

On the Potential Threats of "Smart" Traffic Flooding Attacks

(Poster version: http://www.seas.upenn.edu/~yingx/Sigcomm2004Poster.pdf)

Ying Xu and Roch Guérin *
Multimedia and Networking Lab.
Department of Electrical and Systems Engineering
University of Pennsylvania
Philadelphia, PA, USA
yingx,guerin@ee.upenn.edu

Network Denial-of-Service (DoS) attacks can severely disrupt regular service delivery in the Internet. The substantial threats posed by DoS attacks have triggered the development of many *router-based* defense solutions. For example, *reactive* DoS defense systems such as the ACC-Pushback [1] proposal detect an ongoing attack based on monitoring *link loss rate*. If the measured loss rate exceeds a certain threshold, the ACC system is activated and starts searching for malicious traffic. *Proactive* defense systems have also been designed that continuously monitor the behavior of all users for conformance with a given criterion. For instance, the RED-PD proposal [2] tries to enforce TCP friendliness by sampling the incoming traffic using random packet losses. Any flow that loses more than the expected number of losses of a standard TCP flow during a certain period is deemed non-conformant and will be regulated via packet filtering. Since typical DoS attacks nowadays are mostly high rate attacks that flood the target host or network domain, they are usually accompanied with heavy losses and are likely to be identified and contained by either type of defense.

We expect that, as with other kind of "attacks" such as worms and viruses, DoS attacks will evolve to overcome defenses. Understanding if and when existing defenses can be defeated by more intelligent attackers, and what the implications are in designing better defense mechanisms is, therefore, an important topic. To gain a basic insight into this issue, we developed a number of "smarter" attacking schemes along what we felt were the most natural and promising directions, and evaluated their impact on existing defense schemes such as ACC and RED-PD. We explored two dimensions along which the intelligence (and complexity) of an attacking scheme increases. The first dimension is related to the level of sophistication of a *single* attacker. Specifically, we considered attackers that target either bandwidth or buffer space as the resource they are attempting to deprive other users from without being detected. When a single attacker was not successful, we turned to scenarios involving multiple attackers or more precisely, attackers with *multiple identities*. This accounts for cases when attackers have recruited multiple daemon hosts, or when a single attacker disguises itself using multiple source addresses. The overall complexity of an attacking scheme is then a function of the the number of entities involved, the complexity of each attacking entity, and the level of coordination between them. Our goal is to investigate whether the combination of increased intelligence and the use of multiple identities, can translate into greater attacking efficiency at a reasonable cost. Our main results are as follows:

*This work was supported in part through NSF grant ITR00-85930.

SIGCOMM 2004, Aug.30–Sep.3, 2004, Portland, OR, USA.

Single Attacker Scheme: We found that a *single* attacker employing a *rate adaptation* strategy can successfully evade the ACC system. This type of attacker *slowly* increases its rate and thus the system congestion level, until the link loss rate *reaches and stabilizes* at a level below the ACC activation threshold. This allows the attacker to greatly degrade the performance of TCP users without triggering the underlying defense. However, because of its high intrinsic bandwidth consumption and correspondingly large number of lost packets, a single such attacker will be detected by the RED-PD system. Hence, we considered another type of attacker that instead tries to increase the losses of TCP users while minimizing its own, by attempting to track the evolution of the target link queue, and alternating between bursts and silences in order to fill the queue and then stop before experiencing significant losses. We found that for a RED queue, such a periodic "blasting" strategy can *sometimes* succeed in making TCP users experience loss rates an order of magnitude larger than the attacker. However, the *absolute* amount of packet losses of the attacker is typically still high enough for the RED-PD system to detect and throttle it.

Multiple Identities Scheme: Since it is hard for a single attacker to elude the RED-PD defense, we further examined attacking schemes involving multiple attackers. We found that the attacking efficiency in this case is greatly influenced by how the multiple attacking entities were selected. When attackers are traffic sources residing on different host machines, the number of attackers required to launch a successful attack is large, primarily because it is hard to achieve efficient coordination among sources. In contrast, when the multiple attackers instead corresponded to multiple identities, i.e., multiple source addresses, of a single host, and were selected *in turn* during an attack, a much smaller number of attackers (distinct addresses) was needed to foil the RED-PD system. This is due to the sampling mechanism used by the RED-PD system, which relies on samples within only one scanning interval to detect a TCP-unfriendly user.

Our investigation has revealed that by adapting their behavior, DoS attackers can indeed defeat existing defense systems. We believe that this understanding is a first step in allowing us to devise more efficient and robust DoS defense mechanisms, which is the topic of our current research.

1. REFERENCES

[1] R. Mahajan, S. M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker. Controlling high bandwidth aggregates in the network. *SIGCOMM Comput. Commun. Rev.*, 32(3):62–73, 2002.

[2] R. Mahajan and S. Floyd. Controlling high-bandwidth flows at the congested router. In *Proc. 9th Intl. Conf. Netw. Protocols (ICNP'01)*, pages 192–201. IEEE Computer Society, 2001.

How *Real* Can Synthetic Network Traffic Be?

Félix Hernández-Campos F. Donelson Smith Kevin Jeffay

Department of Computer Science
University of North Carolina
Chapel Hill, NC 27599-3175
{fhernand, smithfd, jeffay}@cs.unc.edu
http://www.cs.unc.edu/Research/dirt

At some point, most networking researchers simulate or emulate networks in order to understand the behavior or performance of some piece of networking technology. In performing these experiments, there is a clear need to have the ability to generate realistic synthetic workloads; synthetic network traffic whose statistical properties match those of traffic observed on a real network link. In our work, we are developing a method of traffic modeling and synthetic generation that can realize a new level of realism in network experiments that is not possible with existing techniques.

Our approach follows the philosophy of using source-level descriptions of network traffic advocated by Floyd and Paxson [1]. We develop a new source-level model of the data exchange dynamics inside a TCP connection that is *application-independent,* and therefore amenable to modeling the complete mix of TCP traffic seen on a link. This is an advance over existing source-level modeling techniques that only model the traffic from a single application (*e.g.*, HTTP models [2]).

Our modeling of TCP connections distinguishes two cases. The first case is motivated by the observation that most applications communicate using a series of requests and responses. These connections can be described as a sequence of the form:

$$<(a_1, b_1, t_1), (a_2, b_2, t_2),\ldots, (a_n, b_n)>$$

where a_i and b_i represent the sizes of application-level data units flowing in the forward and the reverse directions of the connection respectively, and t_i represents the idle time between an exchange. The second case comes from connections wherein endpoints exchange data units concurrently. We represent these connections using two independent sequences of the form:

$$<(a_1, ta_1), (a_2, ta_2),\ldots, (a_n)>,\ <(b_1, tb_1), (b_2, tb_2),\ldots, (b_m)>$$

Our model, called the *a-b-t model*, is powerful enough to capture the essential nature of a connection at the source-level, and yet is simple enough to be populated directly from measurements.

We have developed a set of techniques and tools to convert an arbitrary packet header trace into a set of *connection vectors* of the forms shown above. These vectors represent the workload of TCP in terms of communication demand at the socket-level, and provide the foundation for *closed-loop* traffic generation of *entire traffic mixes*. This enables us to conduct experiments in which some network mechanism (*e.g.*, a TCP variant, an AQM scheme, *etc.*) is exposed the full range of source-level behaviors observed in one (or more) real network links.

The connection vectors are input to synthetic traffic generators we have developed for use in network simulators and testbeds. The most straight-forward traffic generation technique is to directly *replay* the source-level behavior captured by a set of connection vectors. In this case, each measured connection vector results in one synthetically generated connection with the relative start times between connections preserved. We are also investigating techniques for re-sampling and sub-sampling of connection vectors in order to derive new source-level traces that preserve the statistical variability of the original trace but result in different (and controllable) levels of load when used in a replay.

The direct replay of connection vectors has an important benefit: it enables us to compare the properties of a real trace and the ones of its source-level replay. This comparison provides not only a way of validating our modeling technique, but also a method for studying to the extent to which synthetic traffic is *realistic*, *i.e.*, the extent to which it can approximate the measured properties of real traffic. Our goal is to demonstrate the potential of our approach, and expose the limitations of synthetic traffic generation.

We are currently involved in a comprehensive experimental study to develop methods for tuning network-dependent parameters of the experimental environment (*e.g.*, round-trip times, window sizes, access bandwidths) to control the degree of realism in the synthetically generated traffic. Ultimately, the goal is to approach a high-fidelity reproduction of traffic observed on a network link. Our metrics for evaluating the degrees of realism in synthetic traffic include the following:

- The link load or throughput – the number of bits per second (including protocol headers) transmitted on a link.
- The statistical properties of the time series of counts of arriving packets and bytes on a link in an interval of time (*e.g.*, Hurst parameter estimates, wavelet spectra).
- The number of active TCP connections over an interval of time (typically one second).
- The distributions of TCP connections durations and rates.
- The distribution of packet sizes on the link.

Our cases studies include traces from several Internet links, including an OC-48 Abilene backbone link and a 1 Gbps Ethernet link connecting the UNC campus with its Internet service provider. The early results are promising. We have been able to demonstrate that through judicious control of network-dependent parameters, high-fidelity reproduction of traces of traffic from these links is possible in a network testbed.

REFERENCES

[1] S. Floyd, and V. Paxson, Difficulties in Simulating the Internet, *IEEE/ACM ToN*, vol. 9, num. 4, August 2001, pp. 392–403.

[2] P. Barford, and M. Crovella, Generating Representative Web Workloads for Network and Server Performance Evaluation, Proc. ACM SIGMETRICS, July 1998.

Empirical Modeling of Campus-wide Pedestrian Mobility: Observations on the USC Campus

Students: Deepankar Bhattacharjee, Ashwin Rao, Chintan Shah, Manan Shah, Faculty: Ahmed Helmy
Electrical Engineering Department, University of Southern California, Los Angeles, CA, 90089. (213)-821-1329
{dbhattac, ashwinvr, chintans, manansha, helmy @ usc.edu}

Mobility is one of the main factors affecting the performance of mobile ad hoc networks (MANETs). Specifically, in the absence of fixed infrastructure as in MANETs, mobile computing devices (such as handheld devices) may be used to route packets. Mobility of such devices may cause established links to break and consequently established paths between a sender and a receiver to become invalid. Obtaining realistic mobility models is thus essential to the proper design and evaluation of ad hoc protocol performance. Thus far, work on mobility modeling for ad hoc networks considered synthetic models and wireless network usage pattern measurement. In this work, we present a novel approach to model student mobility on campus using "mobility traces". Unlike previous measurement-based techniques, we actually trace user mobility (vs. network access patterns for W-LANs). This provides a new perspective on mobility modeling. We develop a methodology to capture movements of pedestrians (mobility traces) on the USC campus. We further propose various statistical metrics to capture spatio-temporal correlation among people. We then propose a hybrid parameterized mobility model, using the group, smooth random and obstacle mobility models. Parameters for such a hybrid model are estimated based on the trace. Such a model will be also useful for future ad-hoc networks, where an "ad-hoc enabled" device will be ubiquitous; especially in campus-like settings.

APPROACH & RELATED WORK

A wide variety of mobility models [1][2] have been proposed from analytic and simulation-based studies (so called "synthetic" models) on mobile nodes forming an ad-hoc network These are not based on actual measurements of mobility. Other wireless network-usage studies conducted at MIT[4], Dartmouth[3] and UC San Diego[5] capture user usage over access points in the campus. This however, does not model true movement behavior of users, although it does give useful insight for current networks. We capture snapshots of actual movements of people i.e. mobility traces by concentrating on 7 major intersections on the USC campus and divide the students into groups to cover regular time-windows for all weekdays, systematically. Collected data included number of people in groups, number of those who spilt in subgroups and this was collected for each path around the intersection. Subsequently, based on these "real" traces of movements, we obtain statistical parameters of user movements leading to a probabilistic description of user direction and speed. These descriptions are used in conjunction with parameters and attributes of existing synthetic models to obtain closed form expressions or distributions of the mobility, forming a hybrid mobility model.

SCENARIO AND SAMPLE RESULTS

The traces were collected from Feb. 26 to April 25, 2004. During this period, we observed 6389 people, 1758 groups of which 2382 subgroups were formed. About 60 students participated in the observations, and about 220 man hours of traces were collected.

A. Helmy was supported by grants from NSF CAREER, Intel and Pratt & Whitney.

We did not observe stationarity in the distribution of people in campus over different time slots. However, the temporal and the spatio-temporal distributions closely followed human behavior for example movement out of campus was observed during lunch hours. It also followed schedules of the campus such as that for classes or library hours.

Directional probabilities were also calculated from the wireless network usage in the campus and these were found to be very different from those obtained from our traces. The reason is that movement of people in this case tends to cluster around access points and also is restricted to people possessing laptops. Our traces did not have these limitations and as such the sample population observed in the two cases was very different.

The probabilistic description of direction is modeled by a state machine very similar to the finite state machine **(FSM)** where the state is the user location in campus and the "transitional probability" is the probability of movement of a user in a particular direction (example: North). We implemented the FSM using a tool that operates on our traces to generate another trace file which is readable by the Network Simulator (NS). This trace file may be used for the purpose of simulating the mobility of nodes as if they are in the USC campus. This code was used by other teams and they obtained results very dissimilar to those obtained from the synthetic models. It was also established that the FSM or directional equation depends on a direction update factor which is a probability measure which in turn is a function of the user's current location and time. This is desirable in a mobility model.

CONCLUSION

We propose a new method for mobility modeling. The approach we adopted for the development of the proposed mobility model is very different from that used by previous synthetic models. Due to the use of real traces this model aligns closely to the actual movement of people on campus. Thus, it could be used to develop or better evaluate ad-hoc routing protocols, or in capacity planning. The model has been implemented by a FSM and our tool facilitates simulations studies for such protocol evaluations.

References

[1] F. Bai, N. Sadagopan, A. Helmy, "IMPORTANT: A framework to systematically analyze the Impact of Mobility on Performance of RouTing protocols for Adhoc NeTworks", *IEEE INFOCOM,* pp. 825-835, April 2003.

[2] A. Jardosh, E. Royer, K. Almeroth, S. Suri. "Towards Realistic Mobility Models for Mobile Ad hoc Networks." *ACM MobiCom*,, pp. 217-229, September 2003.

[3] D. Kotz, K. Essien, "Analysis of a Campus-wide Wireless Network", *ACM MobiCom*, pp. 107-118, September 2002.

[4] M. Balazinska, P. Castro. Characterizing mobility and network usage in a corporate wireless local-area network, *ACMMobiSy*s 03

[5] M. McNett, G. Voelker, "Access and Mobility of Wireless PDA Users", *ACM MobiCom*, poster, September 2003.

[6] Poster URL: nile.usc.edu/~helmy/mobility-trace/

Fault Diagnosis of Path-Vector Routing Through Accumulation of Topological Connectivity Information

Dan Pei, Mohit Lad, Dan Massey, and Lixia Zhang*

1. POSTER ABSTRACT

This paper presents a new approach to diagnosing routing faults (topology changes) in a path-vector routing protocol such as the Internet's BGP routing protocol. The *route diagnosis* problem takes a sequence of updates observed from a node as input, and identifies, as precisely as possible, the topology changes that triggered the updates, and that affected the new paths received by the node. Route diagnosis in BGP is of great interest to both operators and researchers. However, a path vector algorithm by itself fundamentally lacks sufficient information for effective route diagnosis. As a first step towards automatic BGP route diagnosis, we consider a simplified model of BGP where each AS is an atomic node, and nodes are connected by atomic logical links. Given these simplifying assumptions, we propose the following approaches to enhance path vector protocols to achieve effective route diagnosis.

Built-in Diagnosis and Root Causes. Existing path-vector protocols signal the *effects* of the topology changes through the announcement of a new best path. However, a wide range of different topology changes can all produce the same set of new best path updates. In other words, many different events can all result in the same signal (i.e. the set of updates). To achieve accurate diagnosis some (minimal) set of additional information must be added into the routing protocol to facilitate diagnosis.

Recent work including [1] has proposed adding a *root cause* notification attribute (RCN) in order to improve BGP convergence time. The RCN associates a sequence number with a link (or node). A node that detects a link failure will send out a routing update together with the link name and sequence number. The link name and sequence number are referred to as the *root cause* for this update. Any updates that are triggered by the same failure will propagate this root cause information. Analysis and simulations have shown that, by using RCN, routers can detect and avoid obsolete path and achieve substantial reduction in routing convergence time. This root cause attribute directly signals the triggering reason of each update, and we adopt it in our approach. The root cause explains why a previous best has gone, but does not necessarily explain why a node has chosen its new best path among many alternate paths. We propose the *Queued Root Cause Notification* approach to address this issue. In the Queued RCN approach, those root causes received by a node but didn't change this node's path will be stored in a queue because it explains why the alternate paths are gone. When this node's path does change later, the queued root causes are attached to the new path updates and propagated.

Fully Utilizing Topology Information. In our approach, a router can be attached an auxiliary device, called *monitoring station*, which conducts route diagnosis by passively collecting and analyzing the routing updates received at the router. During routing convergence periods, routers may explore a number of transient paths (both valid and invalid), revealing links and nodes that would not have been seen when the network is stable. Instead of discarding this path information after the network stabilizes, our approach uses the information to build a partial view of the topological connectivity. The root cause information provides us explicit information regarding the failure (or recovery) of some links. This combined information allows us to build an estimate of the network topology graph G.

The root cause information attached to updates directly signals the topology changes that have triggered the incoming updates. The network graph G plus the queued root causes help explain why nodes have chosen the new paths among alternate paths. One side benefit is that one can perform "path labeling" based on the topology graph and the root cause information. For example, if a path includes a failed link, one can classify such a path as "obsolete"; if a path appears to be the best available one according to the graph, it can be classified as "eventual". All this information can be combined to provide an operator or researcher with the information necessary to debug/deal with route changes.

Our experiments using real BGP updates show that the topology graph can be implemented with a reasonable overhead. Our proof-of-concept simulation also demonstrates the effectiveness of our approach. More detailed measure of diagnosis effectiveness is ongoing. We are also investigating the attack/misconfiguration detection based on the topology and root cause information. In summary, the accumulated topology with queued root causes open a wide door to the route diagnosis of path-vector protocols.

*Dan Pei, Mohit Lad, and Lixia Zhang are with UCLA. Email:{peidan,mohit,lixia}.cs.ucla.edu. Dan Massey is with Colorado State Univ. E-Mail: massey@cs.colostate.edu

2. REFERENCES

[1] D. Pei, M. Azuma, N. Nguyen, J. Chen, D. Massey, and L. Zhang, "BGP-RCN: Improving BGP Convergence Through Root Cause Notification," Tech. Rep. TR-030047, UCLA CSD, October 2003.

WISL: An Application That Enables User-Perceived Performance Measurement

Richard Liston and Ellen Zegura*

Large scale network measurement at the end-user can guide network architects and managers in activities like choosing application and transport layer parameters, provisioning and routing. They can also help answer such simple, yet elusive questions about the nature of network services provided to typical users. However, the design and implementation of the Internet makes such measurement studies difficult to perform, so studies are often limited in the number and location of points at which measurements are taken. The results of the measurements may not be representative of network performance as experienced by the end-user. Collecting user-perceived performance data requires users to execute measurement tools on their machines. Unfortunately, users have little incentive to participate in performance measurement activities.

To address the incentive problem we present an application called WISL[1], which stands for "What the Internet Sounds Like". The purpose of the application is twofold. At the lowest layer it provides an extensible platform on which to perform and report measurements of end-user perceived network performance. Researchers are able to plug measurement modules into the application with the expectation that the module will be executed at actual end-user locations throughout the world. Each module performs the measurements from its vantage point and the data can be collated for subsequent analysis.

The second, more salient feature of the application is that it plays sounds that are dynamically generated by network events as they occur and are detected by the measurement modules. The primary intent of this aspect of WISL is to provide a platform for composers to create musical compositions that are generated by events occurring on the network worldwide in real time. The music that is generated represents the composer's interpretation of network-level events. It belongs to a class called "aleatoric" music: music generated — at least in part — by chance events. This musical aspect WISL provides the incentive for end-users to execute the application. WISL is intended to be an interesting and compelling musical application in its own right.

We note in addition that the generality of WISL makes it possible to be used as a network monitoring tool. This is accomplished by mapping critical network events to sounds that act as audio alarms.

There are several criteria for the design of WISL. From the perspective of the end-user, WISL must be simple to download and execute. Also, since individual musical tastes vary the end-user must be able to choose among different SoundPalettes — the packages contributed by composers. Choosing a SoundPalette can be compared to selecting a particular radio station. There can be many different SoundPalettes for the user to choose from. Meeting these criteria will increase the likelihood of participation by end-users and therefore the quality of data collected.

Network researchers must be able to easily incorporate NetModules — the modules that perform the measurements. WISL must also allows for performing a wide range of types of measurements, and be able to accurately timestamp events. Researchers must be able to log data for subsequent analysis. These criteria will ensure that WISL will provide the most useful platform for network researchers. They will also help to attract network researchers to contribute NetModules that report on many different kinds of network events.

WISL must offer the composer the ability to create a representation in sound of a wide variety of events that may occur on the network. A rich source of network event detection will make for a varied and interesting platform upon which composers can create highly varied sound environments. It must also allow the composer to select and combine the events in different ways and at different time scales. The composer must not be bound to using events only at the times that the NetModule creator specifies.

For portability, WISL is implemented in Java and utilizes the JavaSound API. Although Java can be restrictive for some types of network measurement, we have been successful in implementing several interesting, as well as accurate, "atomic" NetModules. We also have created a complex NetModule called LandmarksModule. LandmarksModule collects data to use in evaluating different schemes for network topology mapping. LandmarksModule works by querying a fixed set of DNS servers — the thirteen gTLD servers — as network landmarks. It constructs a vector of RTTs and sends this to a central server which decides what other currently running WISL nodes are topologically close neighbors and sends this list to the WISL node.

Our primary goals in this poster session are to introduce the WISL application to the research community; to outline our criteria for WISL and to present its design in detail; and to discuss the kinds of measurements that WISL would be well-adapted for, as well as its limitations.

*Richard Liston is with Ursinus College and Ellen Zegura is with Georgia Tech, Email: liston@ursinus.edu, ewz@cc.gatech.edu

[1]For more information and download visit www.wisl.info

Landmark Guided Forwarding

[Extended Abstract for http://www.cl.cam.ac.uk/~mhl28/LGFwd.pdf]

M H Lim,J Crowcroft
University of Cambridge
Computer Laboratory
William Gates Building
15 JJ Thomson Avenue
Cambridge CB3 0FD
(mhl28,jac22)@cl.cam.ac.uk

A Greenhalgh
University College London
Computer Science
Department
Gower Street
London WC1E 6BT
A.Greenhalgh@cs.ucl.ac.uk

A Campbell
Columbia University
Department of Electrical
Engineering
1312 Seeley W. Mudd Bldg.
New York,NY 10027-6699
campbell@comet.columbia.edu

Many wireless Ad Hoc routing protocols have been proposed in recent years. However, none of these is ready for wide area deployment. An early survey paper by Elizabeth Royer categorized these protocols as table driven or source driven. In general, table driven protocols pro-actively gather topological routing information while source driven protocols reactively discover a route or routes to the destination as requested by the source. Pro-active routing protocols such as DSDV, Destination Sequenced Distance Vector, pro-actively exchange routing information between neighboring nodes. The associated routing state and the network traffic overheads is $O(n)$, where n is the number of nodes in the network, which does not scale well in large networks. Reactive routing protocols such as DSR, Dynamic Source Routing, use flooding techniques to discover new routes and repair existing routes. As the amount of traffic in the network increases or the diameter of the network increases, the cost of flooding increases. With reactive routing protocols the routing performance degrading under moderate mobility conditions.

An alternative approach to Ad Hoc routing is to take advantage of the physical location of nodes in the network and to do position based forwarding. An assumption made by protocols that take this approach is that every node knows its own geographical position. By limiting the exchange of positional information to only between adjacent nodes, the state and network overheads are reduced to $O(u)$, where u is number of adjacent nodes. GPSR, Greed Perimeter Stateless Routing, is a position based routing protocol that in general uses the geographically closest node to the destination as the next hop for the packet to be forwarded. This technique can result in a sub optimal path between source and destination. To address situations where a packet is required to traverse away from the destination to circle around a void, GPSR uses a perimeter forwarding scheme that uses the well known right hand rule on its planarised graphs. Although GPSR scales well and adapts to random topologies, suboptimal routing is still an issue to be resolved.

These observations lead us to the posistion that the lack of topological knowledge results in sub-optimal position based forwarding. We agreed with the reasoning of the distance effect from DREAM, which demonstrates that the update frequency from remote nodes can be reduced without compromising the routing accuracy. Moreover, connectivity in wireless Ad Hoc network is unpredictable hence it is therefore realistic to restrict the propagation of route information while considering scalability issue.

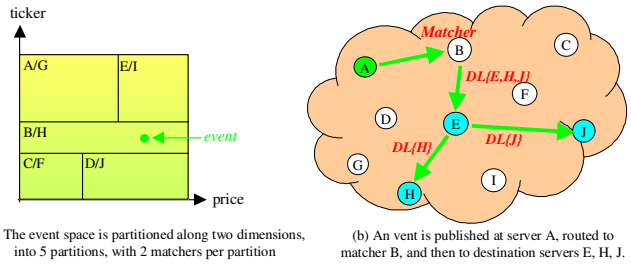
Based on these arguments, we propose a protocol that is a hybrid of position and topological based routing. Each device maintains distance vector and position information of other nodes within its local area. The size of the local area is dynamically scoped depending upon the node density and the transmission range of the device. Inside the local area packets are routed based on the distance vector information, outside the local area routing is based upon the position of the destination. A packet destined for outside the local area is routed using the shortest path algorithm to the node in the area that is physically closest to the destination. Upon receiving the packet, the node determines whether it is inside or outside its local area. If it is inside its local area it is routed to the final destination using the shortest path algorithm. If it is outside the area the process is repeated. Preliminary results show of our implementation scale better than DSDV. By extending the scoped topological view into position based routing, we envisage our routing protocol should have better foresight and performance than generic position based routing. Other protocols are yet to be investigated.

MEDYM: An Architecture for Content-based Publish-Subscribe Networks

Fengyun Cao Jaswinder Pal Singh
Princeton University
{fcao, jps} @cs.Princeton.edu

Content-based publish-subscribe (*pub-sub*) is an important paradigm for asynchronous communication between entities in a distributed network. Users *subscribe* to conditions of interest on future *events*, and are notified when events satisfying those conditions are *published* to the system. Subscriptions can be very expressive, specifying complex filtering criteria. Such timely, highly customized information delivery is valuable to many applications, such as personalized information dissemination, distributed system monitoring, alerting and notification, and application integration.

An Internet-scale pub-sub network consists of a set of pub-sub servers distributed over the Internet. Servers accept events as well as subscriptions from clients of the pub-sub network. We focus on the problem of efficient event delivery from the server at which the event is published to all servers with matching subscriptions. This is a challenging problem, as traditional group-based multicast techniques are not readily applicable. First, pub-sub communication is guided by content rather than network addresses. An event has to be matched with subscriptions to understand *where* it should be sent. Second, it is not clear *how* to route the event, even if the destinations are known. Due to the diversity of content-based subscriptions, different events can match subscriptions from different sets of servers. In the worst case, the total number of such sets can be $2^{\#servers}$, and it is impractical to maintain a multicast group for each such possible set.



(a) The event space is partitioned along two dimensions, into 5 partitions, with 2 matchers per partition

(b) An vent is published at server A, routed to matcher B, and then to destination servers E, H, J.

Figure 1. An example MEDYM network.

Based on the properties of content-based pub-sub communication, we propose a novel pub-sub network architecture called *MEDYM*: *Match Early with DYnamic Multicast*. Fig. 1 shows an example MEDYM network. The event space is partitioned into non-overlapping partitions with balanced load (see (a)). The partitioning algorithm can be data-type dependent (e.g. [5]) and can be combined with considerations of geographic locality, matching capability, etc. Every pub-sub server acts as a *matcher* for one or more partitions. Every partition has one or more matchers, for redundancy and sharing of matching load. Subscriptions are routed to all matchers that are responsible for partitions with which they overlap, so that a given event has to be matched at only one server, and early on so unncessary event propagation is stemmed. A matcher needs to store only those subscriptions that overlap with its event partition(s).

As shown in (b), a published event is first forwarded to a nearby matcher whose partition covers the event. The matcher generates a *destination list* (*DL*) of all *destination servers* with matching subscriptions. At this point, no more matching needs to be done, and the problem is converted to one of routing. In MEDYM, every event message carries a DL, a list of IDs of servers to which the receiver of the message is responsible for event delivery. Based on the DL and knowledge of network topology, a server receiving an event runs a *dynamic multicast routing algorithm*, which computes next-hop servers for the event and partitions the incoming DL into smaller DLs for each next-hop server. The event, with the corresponding new DLs, is then forwarded to the next-hop servers. In this way, a transient, stateless *dynamic multicast tree* grows to span all destination servers. Such distributed decision-making is highly flexible and robust, as servers can optimize routing decisions based on various routing policies and adapt to network environment changes and node or link failures in real time. Since the DL is partitioned at every forwarding step, the average DL size per message in a dynamic multicast tree is only O(log *#destination servers*).

Another major advantage of MEDYM is that events are routed through only the minimum set of servers that actually have matching subscriptions (except for the matcher), thus minimizing total event traffic load in the network, and distributing the load consistent with server self-interests. To the best of our knowledge, no other pub-sub network architecture has achieved this property.

We compared MEDYM with two major existing approaches, the Content-based Filtering (CBF) approach (simulated as in [1][1]) and the Channelization approach (simulated as in [4]). Detailed results can be found in our poster, at http://www.cs.princeton.edu/~fcao/poster.pdf.

Simulation results show that event routing in MEDYM is highly efficient. For example, an event that match subscriptions from 1% servers is routed through 6% servers in CBF, and 28% in Channelization, while only through the 1% matched servers in MEDYM. The maximum server bandwidth consumption of CBF is 58 times that in MEDYM, and of Channelization 37 times; the maximum link stress of CBF is 6 times that in MEDYM, and of Channelization 13 times. Content space partitioning also keeps subscription maintenance cost low, as subscriptions are only replicated and updated at matchers with overlapping partitions. The advantages of MEDYM are greatest for large networks and high subscription selectivity, exactly the scenarios where efficient content-based pub-sub is most valuable.

The overheads MEDYM introduces, such as the DL overhead and dynamic routing computation cost, turn out to be acceptable and more than outweighed by its benefits. For example, to route an event to 1,000 destinations, the average DL has only 18 server IDs and the routing algorithm we developed runs in under 0.5ms at the first server and decreases quickly thereafter. Parallelism can help even further.

We have implemented a prototype of MEDYM on PlanetLab and plan to deploy a publicly available pub-sub service using it. The current MEDYM architecture is expected to scale to at least a few thousand servers, which is more than adequate for most foreseeable applications. We are exploring the use of hierarchical structures or DHT techniques for further scalability. More information about our work can be found at http://www.cs.princeton.edu/DADI/.

REFERENCES

[1] A. Carzaniga, D. Rosenblum, and A. Wolf, "Design and evaluation of a wide-area event notification service," In *ACM TOCS.*, 2001.
[2] A. Carzaniga, A.L. Wolf, "Forwarding in a Content-Based Network". In Proc. of *ACM SIGCOMM* 2003.
[3] A. Carzaniga, M.J. Rutherford, and A.L. Wolf, "A Routing Scheme for Content-Based Networking". In Proc. of *IEEE INFOCOM* 2004.
[4] A. Riabov, Z. Liu, J. Wolf, P. Yu and L. Zhang, "Clustering Algorithms for content-based publication-subscription systems," In Proc. of *ICDCS* 2002.
[5] Y. Wang, L. Qiu, et. al, "Subscription Partitioning and Routing in Content-based Publish/Subscribe Networks". In Proc. of *DISC*'02.

[1] We observe that routing information described in [2,3] can result in redundant event routing, which was not addressed in [2,3]. Private correspondence with author A. Carzaniga confirms this observation, and we are working with him to understand the tradeoff between redundant routing and state needed, which is unclear at this stage. Therefore, we simulated [1] as representative for the CBF approach.

Understanding the interaction between overlay routing and Traffic Engineering *

H. Zhang, Yong Liu, Don Towsley †
{honggang, yongliu, towsley}@cs.umass.edu

Weibo Gong ‡
gong@ecs.umass.edu

Keywords
Overlay Routing, Traffic Engineering, Game Theory

ABSTRACT
We study the interaction between optimal routing overlay and Multiprotocol Label Switching (MPLS) Traffic Engineering (TE) in a single Autonomous System (AS). Our work is motivated in part by the work of Qiu *et al* [1], in which the interaction between overlay selfish routing and TE is brought up. However, our work is different in that [1] assumes each overlay user controls an infinitesimal amount of traffic demand and makes routing decisions independently. We study a single large scale and centrally controlled overlay network that controls a *non-negligible portion* of traffic demand and does optimal routing on application level. Akamai exemplifies this type of overlay. We further assume that the proportion of overlay traffic is significant enough to influence the routing decisions of TE.

We formulate this interaction as a two-player non-cooperative non-zero sum game, where TE's objective is to minimize the network cost as a whole, and an overlay optimizer is to minimize the overall delay for its own traffic on top of the routings set by TE. Strategies of overlay are the flow allocations of overlay traffic on logical (application) level, whereas, strategies of TE are the flow allocations of all traffic demand on physical level. The routing decisions (strategies) of overlay are essentially the inputs to TE (interpreted as traffic demands), and in turn, routing decisions (strategies) of TE will influence the future routing decisions made by overlay by affecting the costs or delays on logical links.

The key contributions and results are summarized as follows.

We focus on two types of games. In the first type of game, *Nash routing game*, overlay and TE are of equal status, and the interaction process is a *best-reply dynamics* in which each player takes turn to compute their optimal strategies based on the response of the other player in last round. Our conclusions of this game are that overlay routing will never improve the performance of TE if TE uses MPLS (analytically proved), and in most cases, TE's cost will be increased a lot in the interaction with overlay, and the cost increase of TE is a function of the percentage of overlay traffic. If overlay traffic is about 50% of total traffic, overlay's influence on TE's performance achieves the largest. These conclusions are confirmed in experiments in a 14-node tie-1 ISP topology. To illustrate the interaction process, for a simple network, we give an analytical proof on the existence, uniqueness and global stability of Nash equilibrium (NEP.) For general networks, we show experimentally that the selfish behavior of overlay can cause huge cost increase and oscillations to the whole network.

Even worse, we have identified cases, both analytically and experimentally, where the overlay's cost increases as the Nash routing game proceeds even though overlay plays optimally based on TE's routing at each round. Thus, it may not be wise for an overlay to always optimize its routes each time TE does physical routings. This observation is consistent with the inherent inefficiency characteristic of NEP and is of practical importance to an overlay routing structure.

In the second type of game, *Stackelberg routing game*, we propose that overlay play as a leader to completely eliminate oscillations and optimize its own performance. Since solving a static Stackelberg game (a bi-level programming problem) is NP-hard, we provide a gradient projection search heuristic to solve for Stackelberg strategy. Our preliminary results show that it is very promising to use this heuristic to solve for the approximate Stackelberg routing strategy for an overlay network.

Finally, we further study the games in which overlay has only limited information. We also discuss other issues on: The interaction between multiple overlays; Frequency and time scale of game playing process.

*A poster version of this paper is available at `www-net.cs.umass.edu/~honggang/overlay_poster.pdf`
A full version of this paper is available as a technical report at `www-net.cs.umass.edu/~honggang/overlay.pdf`

†Dept. of Computer Science, University of Massachusetts, Amherst, MA 01003. Honggang Zhang is a PhD student. Yong Liu is a senior postdoctoral research associate. Don Towsley is a professor.

‡Weibo Gong is a professor in Dept. of Electrical and Computer Engineering, University of Massachusetts, Amherst, MA 01003.

1. REFERENCES

[1] L. Qiu, Y.R. Yang, Y. Zhang, and S. Shenker. On selfish routing in Internet-like environments. In *Proceedings of the ACM SIGCOMM*, 2003.

A Systematic Simulation-based Study of Adverse Impact of Short-lived TCP Flows on Long-lived TCP Flows[1]

Shirin Ebrahimi-Taghizadeh
University of Southern California
sebrahim@usc.edu

Ahmed Helmy
University of Southern California
helmy@usc.edu

Sandeep Gupta
University of Southern California
sandeep@poisson.usc.edu

http://www-scf.usc.edu/~sebrahim/sigcomm04-poster.html

ABSTRACT

Best effort applications over non-TCP protocols, e.g., UDP, can be used in attacks that capture unfairly large share of bandwidth compared to TCP flows. While earlier studies may have pointed out that short-lived TCP flows (mice) may hurt long-lived TCP flows (elephants) in the long term, no insight was given as to developing scenarios leading to drastic decrease in throughputs of long-lived TCP flows. We have systematically developed TCP attack scenarios that differ from all prior research in that we use short-lived TCP flows to attack long-lived TCP flows. Our attacks are interesting since, (a) they are more difficult to detect, and (b) they point out the increased vulnerabilities of recently proposed scheduling, AQM and routing techniques that further favor short-lived flows.

We systematically exploit the ability of TCP flows in slow-start to rapidly secure greater proportion of bandwidth compared to long-lived TCP flows in congestion avoidance phase, to a point where they drive long-lived TCP flows into timeout. We use simulations, analysis, and experiments to systematically study the dependence of the severity of impact on long-lived TCP flows on key parameters of short-lived TCP flows – including their locations, durations, and number, as well as the intervals between consecutive flows. We derive the ideal durations of, as well as the ideal intervals between, attacking short-lived TCP flows. We show that targeting bottleneck links does not always cause maximal performance degradation for the long-lived flows. In particular, our approach illustrates the interactions between TCP flows and multiple bottleneck links and their sensitivities to correlated losses in the absence of 'non-TCP friendly' flows and paves the way for a systematic synthesis of worst-case congestion scenarios. Table 1 shows the percentage reduction in throughput of long-lived flows when attacked by UDP flows [3]. The table also shows that an attack by a carefully selected sequence of short-lived TCP flows (Figure 1) achieves nearly equal reduction in throughput.

Randomly generated scenarios (where the numbers, durations, and locations of, as well as the intervals between, short-lived flows are all selected randomly) cause less than 10% reduction in the throughput of long-lived flows. In contrast, the scenarios generated using our heuristic approach based on the above results provide greater than 85% reduction in throughput. Figure 2 depicts the frequency response of the long-lived TCP flows under such attacks. Similar scenarios achieve similar reductions for several TCP variants (Tahoe, Reno, New Reno, Sack), and for different packet drop policies (DropTail and RED). Our results demonstrate that even TCP friendly-flows, if carefully orchestrated, can severely degrade throughput of long-lived flows. They also demonstrate that scheduling, AQM and routing techniques and TCP variants designed to give higher preference to short-lived flows will make long-lived flows even more vulnerable to such attacks.

Table 1: Comparing UDP and TCP attacks

Type of malicious flows	Long-lived TCP flows throughput degradation
UDP constant bit rate flows	Up to 100%
UDP short bursts with P=1 sec	More than 90% [7]
Random mix of TCP short-lived and long-lived flows	Up to 10%
Specific pattern of TCP short-lived flows with P=1	>85%

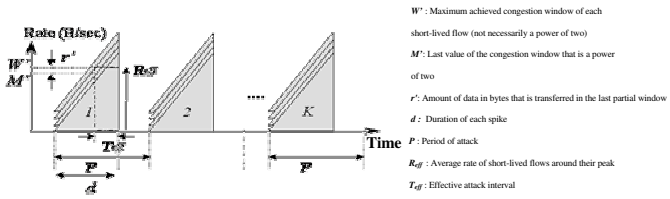


Figure 1: Effective stream of short-lived flows

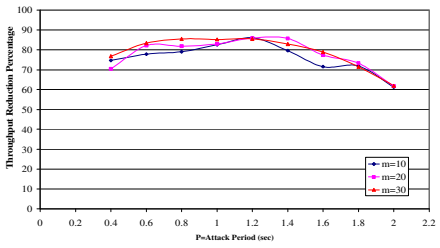


Figure 2: Frequency response of the long-lived TCP flows for *m* concurrent short-lived flows per attack interval

REFERENCES

[1] L. Guo and I. Matta , "The War between Mice and Elephants," Proc. of ICNP'2001, November 2001.

[2] A. Kantawala and J. Turner, "Queue Management for Short-Lived TCP Flows in Backbone Routers," Proc. of High-Speed Symposium, Globecom 2002.

[3] A. Kuzmanovic and E. Knightly, "Low-Rate TCP-Targeted Denial of Service Attacks," Proc. of SIGCOMM August 2003.

[1] This work was supported by grants from DARPA, NSF and NASA.

Removing Redundant Rules from Packet Classifiers

Alex X. Liu, Mohamed G. Gouda
Department of Computer Sciences
The University of Texas at Austin
Austin, Texas 78712-1188, U.S.A.

Redundancy Problem: Packet classification is the core mechanism that enables many networking services such as firewall access control and traffic accounting. A packet classifier consists of a sequence of rules whose function is to classify each incoming packet into one of predefined classes. A rule in a packet classifier is redundant if removing the rule does not change the decision of the packet classifier for each packet. For instance, in the following packet classifier, where each rule only checks one packet field F_1 whose domain is $[1, 100]$, rule r_3 is redundant and rule r_2 is redundant after removing r_3.

$$\begin{aligned} r_1: &\ F_1 \in [1, 50] \rightarrow accept \\ r_2: &\ F_1 \in [40, 90] \rightarrow discard \\ r_3: &\ F_1 \in [30, 60] \rightarrow accept \\ r_4: &\ F_1 \in [51, 100] \rightarrow discard \end{aligned}$$

How to detect and remove all redundant rules in a packet classifier is a new problem that has not been scientifically addressed previously.

Why Important: By removing all redundant rules from a packet classifier before a packet classification algorithm starts building data structures from the rules, both classification space and classification time are reduced.

Reducing memory space for packet classification algorithms is of paramount importance because a packet classifier on a core router must use very limited on-chip cache to store complex data structures. Fast packet classification algorithms need $O(n^d)$ classification space, where n is the total number of rules and d is the total number of packet fields that the classifier examines for each packet. On average a packet classifier has more than 15% redundant rules. Figure 1 shows the percentage of classification space that can be saved by removing all redundant rules versus the number of packet fields (assuming only 15% rules are redundant).

Reducing the amount of overlapping among rules or reducing the total number of rules reduces classification time. By removing redundant rules, while other non-redundant rules remain unchanged, both the amount of overlapping of rules and the total number of rules are reduced.

Previous Work: Gupta gave a sufficient but not necessary condition for identifying redundant rules in his PhD thesis. For example, rule r_3 and r_2 in the previous example are not identified as redundant rules by his definition.

Our Contribution: In this paper, we give a necessary and sufficient condition for identifying all redundant rules, based on which we categorize redundant rules into upward redundant rules and downward redundant rules. A rule r in a packet classifier is *upward redundant* if there are no packets whose first matching rule is r. For example, rule r_3 in the previous example is upward redundant. A rule r in a packet classifier is *downward redundant* if for each packet, whose first matching rule is r, the first matching rule below r has the same decision as r. For example, rule r_2 in the previous example is downward redundant after r_3 is removed.

[1]Corresponding author: Alex X. Liu (alex@cs.utexas.edu)

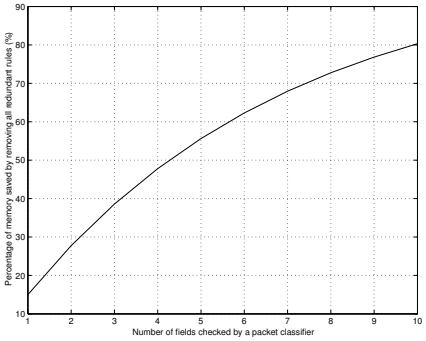


Figure 1: Number of fields vs. Memory Saved

In this paper, we present two efficient graph based algorithms for detecting and removing these two types of redundant rules. The data structure we use for detecting and removing redundant rules is called packet decision diagrams.

Experimental Results: We implemented the algorithms in this paper in SUN Java JDK 1.4. The experiments were carried out on one SunBlade 2000 machine running Solaris 9 with 1Ghz CPU and 1 GB memory. The average processing time for removing all upward and downward redundant rules from a packet classifier versus the total number of rules in the packet classifier is shown in Figure 2.

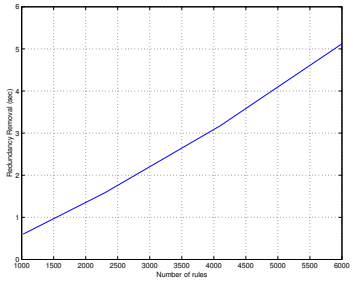


Figure 2: Average processing time vs. Total number of rules

Full Text Available:
http://www.cs.utexas.edu/ftp/pub/techreports/tr04-26.pdf

Aggregate Rate Control for TCP Traffic Management

Jae Chung and Mark Claypool
Computer Science Department
Worcester Polytechnic Institute, USA
{goos|claypool}@cs.wpi.edu

TCP, the de-facto Internet transport protocol, has a successful end-host congestion control mechanism that has largely been effective in managing Internet congestion. Yet, with TCP alone it is hard to achieve efficient congestion control mainly due to the inefficiency of congestion estimation at the end-host. Active queue management (AQM) promises to overcome the limitations of end-host only congestion control by providing congestion feedback information before router buffers overflow.

The most promising approaches view AQM as a feedback controller on a time-delayed response system and apply control engineering principles to design an efficient controller for TCP traffic [2, 3, 4]. In modern control systems, proportional integral derivative (PID) designs dominate due to their simplicity and effectiveness. Without exception, this applies to recent active queue management developments and has altered AQM research from basic framework design into detailed controller design and practical implementation issues.

Among PID principles, only the proportional integral (PI) feedback control approach is primarily considered for AQM since the effect of the derivative control is often insignificant under practical Internet environments. While the PI control approach seems promising, a critical deployment challenge is the configuration of PI control parameters in a time-delayed feedback system, i.e., the Internet. There are no simple and effective PI control parameter configuration available for time-delayed system [5]. The existing PI control-based AQM mechanisms such as the PI controller [3] or Adaptive Virtual Queue (AVQ) [4] lack complete configuration guidelines, making their practical deployment difficult.

We propose a practical, rate-based AQM mechanism offering aggregated rate control (ARC) for TCP traffic. ARC is a reduced parameter PI controller, founded on classical control theory and a sound understanding of PI behavior for the Internet traffic control domain, offering easy configuration while keeping to proven system stability characteristics. We model a TCP-ARC feedback control system using a linear TCP model [3] and develop practical yet effective ARC configuration guidelines. The guidelines cover issues in choosing a target stable boundary system for ARC configuration and provide a method for selecting control parameters that help avoid system instability even when the system is out of the stability boundary. The guidelines also address the effects of the rate sampling interval on system stability, a consideration often neglected in other AQM studies.

Controllers with the same underlying principle design can result in noticeably different implementations, both in terms of complexity and performance depending on the way in which control information is obtained and feedback is processed. AQM requires information on the incoming traffic load to make accurate congestion control decisions, information which can be obtained in two different ways: derivation of queue samples or incoming traffic rate over the service rate. ARC takes a rate-based control information acquisition approach. For a small amount of data collection overhead, rate-based data acquisition reduces sampling noise that can significantly degrade the accuracy of congestion measurement. In addition, rate-based mechanisms can more effectively react to impending congestion, making control decisions before outbound queue buildup and can also be tuned to minimize queuing delay for a small loss in link utilization, allowing enhanced support for the quality of service (QoS) needs of various Internet applications.

Through an extensive simulation study, we have evaluated ARC and similar AQM mechanisms including the PI controller [3], AVQ [4] and SFC [2], and drop-tail queue management over a wide range of network and traffic conditions including Web flash crowd and multiple bottleneck cases. Our simulations show that ARC effectively handled network congestion in all the tested traffic conditions, outperforming other mechanisms in terms of queuing delay, link utilization, packet loss, and object response time for Web traffic. For details, see [1].

REFERENCES

[1] J. Chung and M. Claypool. Aggregate Rate Control for Efficient and Practical Congestion Managment. Technical Report WPI-CS-TR-04-03, CS Department, Worcester Polytechnic Institute, Feb. 2004. ftp://ftp.cs.wpi.edu/pub/techreports/pdf/04-03.pdf.

[2] Y. Gao and J. Hou. A State Feedback Control Approach to Stabilizing Queues for ECN-Enabled TCP Connections. In *Proceedings of IEEE INFOCOM*, San Francisco, CA, USA, Apr. 2003.

[3] C. V. Hollot, V. Misra, D. Towsley, and W.-B. Gong. On Designing Improved Controllers for AQM Routers Supporting TCP Flows. In *Proceedings of IEEE INFOCOM*, pages 1726–1734, 2001.

[4] S. Kunniyur and R. Srikant. Analysis and Design of an Adaptive Virtual Queue (AVQ) Algorithm for Active Queue Management. In *Proceedings of ACM SIGCOMM*, San Diego, CA, USA, August 2001.

[5] G. Silva, A. Datta, and S. P. Bhattacharyya. PI Stabilization of first-order Systems with time delay. *Automatica*, December 2001.

Lilith: an Interconnection Architecture Based on Label Switching for Spontaneous Edge Networks

Vincent Untz,
Student
LSR-IMAG Laboratory , Grenoble, France
Vincent.Untz@imag.fr

Martin Heusse, Franck Rousseau,
Andrzej Duda
LSR-IMAG Laboratory, Grenoble, France
{Martin.Heusse, Franck.Rousseau,
Andrzej.Duda}@imag.fr

Aims and Scope

We consider the problem of *spontaneous edge networks*. With this term we designate networks that interconnect hosts by means of different physical and link layer technologies and in which all or some of hosts are organized as a multihop ad hoc network. A spontaneous network can be connected to the global Internet or form an isolated group of hosts with internal connectivity. Such networks are becoming wide spread with the advent of various communicating devices at home and in offices as well as with the development of pervasive devices connected via different types of networks and integrated within the physical world.

Our goal is to define an interconnection architecture that supports TCP/IP applications without configuration or other technical effort from the user. The applications should work when a spontaneous network is connected to the global Internet or disconnected as well as when its topology changes due to host mobility or switching network interfaces. Moreover, the architecture needs to support different kinds of applications, especially those that generate time dependent traffic.

Layer 2.5 Spontaneous Networking

To address the issues raised in the case of spontaneous edge networks, we propose an interconnection architecture for spontaneous networks based on the following principles:

- we make all the hosts connected via different links to appear as one single IP subnet so that configuration protocols can use the subnet broadcast (IPv4) or the scoped multicast (IPv6) for all forms of discovery (addresses, names, services);
- we propose to interconnect hosts at layer 2.5, which enables us to easily integrate an ad hoc routing protocol;
- we use MPLS *(Multi Protocol Label Switching)*, the standard 2.5 layer and we establish LSPs (Label Switched Path) on demand;
- LSP paths transport packets between different links;
- the establishment of LSP paths is driven by a reactive ad hoc routing protocol.

We place interconnection at layer 2.5 because if we want to organize all hosts into one subnet, the interconnection should be done below layer 3 and above layer 2.

MPLS, the standard 2.5 layer allows us to leverage the existing expertise and implementations. We believe that when coupled with on demand establishment of LSP driven by an ad hoc routing protocol, MPLS provides several advantages: multiple paths for QoS traffic management and load distribution, possibility of testing LSP reachability and quality, and even QoS based routing.

Since we consider a spontaneous network as a single IP subnet, our architecture propagates layer 3 broadcasts or scoped multicast to all hosts. They can then be used for all configuration protocols. In particular, when the network is isolated from the global Internet, hosts can acquire addresses based on Auto IPv4 or stateless configuration IPv6 and use mDNS or LLMNR for name resolution. When connected to the border router, they benefit from DHCP service to learn the routable prefix. Moreover, protocols such as UPnP, SLP, or JINI can readily be used for service discovery.

An important aspect of wireless spontaneous networks is an ever-changing topology which causes routes to frequently appear and disappear. We address this issue by periodically exchanging the states of the LSP paths using the MPLS traffic statistics, thus enabling nodes to detect broken or error-prone paths. Emitters can then decide to try building a better path or use another one previously saved for backup.

To evaluate our approach, we have designed and implemented Lilith, a prototype of an interconnection node based on the Linux version of MPLS. It relies on a simple reactive ad hoc routing protocol for finding routes in a spontaneous network.

More materials

Additional information:
http://www-lsr.imag.fr/Les.Personnes/Vincent.Untz/lilith/

Sequoia – A Robust Communication Architecture for Collaborative Security Monitoring Systems

Xun Kang, Dayi Zhou, Dan Rao (Advisors: Jun Li, Virginia Lo)
Network Research Group, University of Oregon
{kangxun, dayizhou, rao, lijun, lo}@cs.uoregon.edu
Website: http://netsec.cs.uoregon.edu/research/sequoia.php

ABSTRACT

Our work involves the design, evaluation, and deployment of *Sequoia*, a robust communication architecture for distributed Internet-scale security monitoring systems. Sequoia supports a rich set of communication patterns for regional and global sharing of monitor observations, collaborative decision-making among monitors, and timely delivery of security information to monitors. Highly secure communication is achieved through a comprehensive set of security mechanisms for trust management of participating monitors and trust-based routing. In addition, Sequoia offers high-quality and reliable communication services using a scalable self-organizing structure that is resilient and adaptive.

The design of Sequoia is driven by our current research in open proxy blacklists and worm defense. Sequoia's communication architecture supports aggregation, integration, and dissemination of blacklists using a publisher-subscriber paradigm. We are also investigating distributed worm defense using Sequoia's infrastructure for collaborative consensus on worm signatures as well as for filtering and dissemination of worm information.

Sequoia comprises three key protocols through which monitors self-organize into a two-level hierarchy on which scalable, fast and trustworthy message delivery can be achieved:

The *Monitor Neighbor Discovery Protocol (MND)* is used to form a topology-aware flat overlay among monitors, with every monitor connected to nearby nodes as its neighbors. A monitor node joins the Sequoia monitor overlay by contacting known landmark nodes to obtain its coordinates, which are then used to query a directory server for a recommended list of nearby nodes. The monitor then chooses the closest neighbors based on round-trip measurements. Each node can further optimize and maintain its neighborhood relations through local gossiping.

The goal of the *Distributed Dominator Selection Protocol (DDS)* is to form a two-level communications hierarchy from the flat neighbor overlay constructed by *MND*. A monitor in the higher level of this hierarchy (*dominators*) must meet minimum requirements regarding trustworthiness and routing performance. A monitor can choose to apply for a Sequoia-certificate, or *S-certificate*, from a registry service, certifying this monitor's service type, trust level, public key, and other information. Each monitor in the lower level (*dominees*) eventually selects one or more higher level monitors; thus, each dominator acts as a hub for a group of dominee nodes to reach the rest of monitors. A dominator periodically advertises itself to its x-hop neighborhood, and presents its S-certificate and other qualifications to dominees. As needed, a dominee node can search in its y-hop neighborhood for dominators, selecting those it wishes to utilize based on the dominator's attributes. A caching mechanism is used to reduce message overhead. While improving scalability, the two-level structure ensures that untrusted nodes will not be able to forward security information for others, providing a robust communication structure.

Sequoia supports a rich set of communication modes among monitors, including unicast, multicast, broadcast, anycast, and aggregation. The *Communication Path Discovery Protocol (CPD)* discovers multiple delivery paths from one or more senders to one or more destinations, considering both efficiency and security constraints. This is achieved by mapping the highly trusted dominator nodes into a structured overlay network. Disjoint paths are found using node labeling properties associated with the overlay, while trusted paths are found using a distributed protocol that maximizes the trust rating of a path. Between each sender and each receiver, additional maximally disjoint paths can be established if stronger resiliency is desired.

The need for an architecture for security monitoring systems to gather, share, and deliver information in a large-scale system without centralized control has never been more compelling. Sequoia's use of a self-organized topology-aware structure to support rich, fault-tolerant, and secure communication is an important step towards this goal.

REFERENCES

[1] J. Li, P. Reiher, and G. Popek. Resilient self-organizing overlay networks for security update delivery. *IEEE JSAC*, 22(1), 2004.

[2] A. Rosenstein, J. Li, and S. Tong. Mash: The multicasting Archie server hierarchy. In *ACM Computer Communication Review*, 1997.

[3] A. Rowstron, A. Kermarrec, M. Castro, and P. Druschel. SCRIBE: The design of a large-scale event notification infrastructure. In *Proc. NGC 2001.*

[4] E. Anderson and J. Li. Aggregating detectors for new worm identification. In *USENIX 2004 WIP.*

[5] H. Kim and B. Karp. Autograph: Toward automated, distributed worm signature detection. In *USENIX Security 2004.*

Lower Space Bounds for Approximate Fairness

Abhimanyu Das, Debojyoti Dutta, Ashish Goel, John Heidemann, Ahmed Helmy
USC/Mahi Networks, USC/ISI, Stanford, USC/ISI, USC
abhimand@usc.edu, ddutta@usc.edu, ashishg@stanford.edu, johnh@isi.edu, helmy@usc.edu

Approximate max-min fair bandwidth allocation has been a very well studied problem. A low state scheme for the above problem will have interesting consequences for router design. In this paper we show that in the general case, if there are n flows, routers need to maintain $\Omega(n)$ state in order to give bounded fairness guarantees with low error. Our proof is independent of any assumptions about the internal algorithmic details. We are unaware of previous work in this direction.

An algorithm for bandwidth allocation among n flows is defined to be ϵ-fair if the bandwidth allocated to flow i, μ_i, is related to its demand λ_i, the error fraction ϵ, and its max-min fair rate f by: $(1-\epsilon)\min(\lambda_i, f) \leq \mu_i \leq (1+\epsilon)\min(\lambda_i, f)$. Unless stated otherwise, we use ϵ-fairness and approximate max-min fairness interchangeably. Thus, we have:

THEOREM 1. *Any algorithm that imposes ϵ-fairness among packets of n flows in a given window size W (ie provides fairness within every set of W consecutive packets) to within any constant relative error $\epsilon < \frac{1}{8}$ takes $\Omega(n)$ space.*

PROOF. Consider a sliding window of size W, and an algorithm, A, that provides ϵ-fairness within this sliding window. We provide a proof by contradiction. Assume A requires $g(n)$ state, where $g(n)$ is not $\Omega(n)$. Then we construct an encoder/decoder combination to transmit an n-bit string with an equal number of zeros and one's, using $g(n)$ bits. This violates information theory bounds for lossless coding.

Encoder Construction: If the i^{th} bit of the n-bit string is 0, we create a flow with rate r pkts/s (we call it a good flow), and if the i^{th} bit is 1, we create a flow with rate kr pkts/s (such a flow will be called a bad flow). Our construction uses constant sized packets. We choose k to be much greater than 2. The total sending rate of all the flows is $R = \frac{nr(k+1)}{2}$ pkts/s. The window W therefore corresponds to $\Delta = \frac{W}{R}$ seconds of the aggregate flow rate R. Let $x = r\Delta$. Thus, every window of W packets contains exactly $x = r\Delta$ packets of each of the $n/2$ good flows and $kx = kr\Delta$ packets of each of the $n/2$ bad flows. Set the capacity of the black box to be $C = 2nr$ pkts/sec. Thus, among the window W of packets, only $C\Delta = 2nx$ packets are accepted, and the rest are dropped. The max-min fair rate f of the router therefore corresponds to $3x$ accepted packets per flow within the window W. Now let us insert a total of W packets from these flows into A in a weighted round robin fashion, with weights 1 and k corresponding to good and bad flows respectively. The state of the algorithm, is sent as the encoded state. Thus the size of the code sent by the encoder is $g(n)$, which is not $\Omega(n)$ by assumption.

Decoder Construction: We make n copies of the algorithm A's state (received from the encoder) for decoding the n bits. For the i^{th} copy, we decode bit i as follows. We send x new packets labeled i through A. Then we observe how many of these packets are accepted, and decide the value of the bit accordingly.

Consider the last W packets seen at the encoder (before the x packets are inserted at the decoder, as above). Let us partition these W packets into two sets, M and P, such that $|M| = x$ packets and $|P| = W - x$ packets. Also, let the set of x new packets sent at the decoder be denoted by Q. By our assumption, the algorithm A provides ϵ-fairness within every window of W consecutive packets seen. In particular, this is true for each of the two windows formed by the consecutive sets MP and PQ, where $MP = M \cup P$ and $PQ = P \cup Q$. Now, we consider two cases.

Flow i is bad: We will find out the maximum number of accepted packets of flow i in Q denoted by maxQ_i. Consider the window MP. Now, the ideal fair share of this flow in the window MP is given by $\text{FS}_{\text{MP}}^{\text{bad}} = \frac{2nx - \frac{n}{2}x}{\frac{n}{2}} = 3x$. Also, the maximum possible number of packets of i accepted in M can be obtained by assuming that all packets of flow i in M are accepted; this is given by $\text{maxM}_i^{\text{bad}} = \left(\frac{k}{k+1}\right)\left(\frac{x}{\frac{n}{2}}\right) = \frac{2kx}{(k+1)n}$. Therefore the minimum possible number of accepted packets of i in P, $\text{minP}_i^{\text{bad}}$ is given by $\text{minP}_i^{\text{bad}} = \text{FS}_{\text{MP}}^{\text{bad}}(1-\epsilon) - \text{maxM}_i^{\text{bad}} = 3x(1-\epsilon) - \frac{2kx}{(k+1)n}$. Now consider the window PQ. The ideal fair share of this flow in the window PQ is given by $\text{FS}_{\text{PQ}}^{\text{bad}} = \frac{2nx - \frac{W-x}{k+1}}{\frac{n}{2}} = 3x + \frac{2x}{(k+1)n}$. Therefore, maxQ_i is given by the following equation: $\text{maxQ}_i^{\text{bad}} = \text{FS}_{\text{PQ}}^{\text{bad}}(1+\epsilon) - \text{minP}_i^{\text{bad}} = 6x\epsilon + \frac{2x\epsilon}{(k+1)n} + \frac{2x}{n}$.

Flow i is good: In this case, we will find out the minimum possible number of accepted packets of flow i in Q denoted by $\text{minQ}_i^{\text{good}}$. Consider the window PQ The maximum possible number of accepted packets of i in P can be obtained by assuming that all packets of flow i in P are accepted; this is given by $\text{maxP}_i^{\text{good}} = \frac{W-x}{(k+1)\frac{n}{2}}$. The ideal fair share of this flow in the window PQ is given by $\text{FS}_{\text{PQ}}^{\text{good}} = x + \frac{W-x}{(k+1)\frac{n}{2}}$. Therefore, $\text{minQ}_i^{\text{good}}$ is given by the following equation: $\text{minQ}_i^{\text{good}} = \text{FS}_{\text{PQ}}^{\text{good}}(1-\epsilon) - \text{maxP}_i^{\text{good}} = x(1-\epsilon) - \frac{(W-x)\epsilon}{(k+1)\frac{n}{2}} = x - 2x\epsilon + \frac{2x\epsilon}{(k+1)n}$. Now, if $\text{minQ}_i^{\text{good}} > \text{maxQ}_i^{\text{bad}}$, then we can clearly decode flow i as good or bad, based on the number packets that are accepted out of the x packets in Q. This simplifies to the following condition $x - 2x\epsilon + \frac{2x\epsilon}{n(k+1)} > 6x\epsilon + \frac{2x}{n} + \frac{2x\epsilon}{(k+1)n}$, or $1 - 2/n > 8\epsilon$. Thus, for large n, if $\epsilon < 1/8$, we can decode all the n bits without error. This violates the lower bounds in coding in the following way. The size of the minimum code is given by the entropy of the system. Now the entropy of the n-bit string is given by $\log C_{n/2}^n = \Omega(n)$. But by assumption, the code size is $g(n)$, which is not $\Omega(n)$. This is a contradiction. □

By manipulating the parameters, it is possible to improve the bounds on ϵ.

One interesting direction would be to study the lower bounds under different models. Our on-going work also includes the study of the communication complexity of distributed schemes to impose approximate global max-min fairness. For more details, see

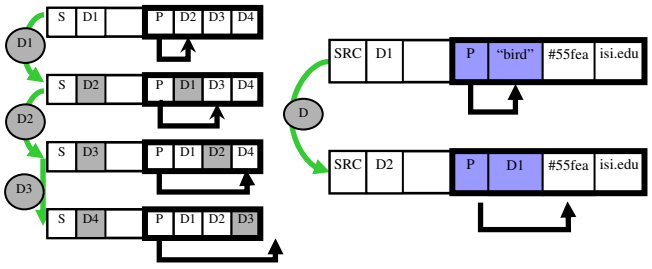
http://netweb.usc.edu/ddutta/research/fairness.

DataRouter: A Network Layer Service for Application-Layer Forwarding

Venkata K. Pingali, Runfang Zhou, Joseph D. Touch
USC Information Sciences Institute
{pingali, rzhou, touch}@isi.edu
http://www.isi.edu/datarouter

Application overlays in the Internet such as Chord and CAN enable interesting and useful content-directed forwarding at a significant performance and complexity cost. Hop-by-hop connections used in such systems necessitate application-layer mechanisms to ensure end-to-end reliability and resequencing. Custom re-implementations of these transport-level functions within the application make it difficult to share code across overlay systems and hinder interoperation. Existing network-level overlays lack separate application endpoint identifiers and forwarding based on them. DataRouter is an open, generic packet forwarding facility that uses string rewriting. It augments the traditional, numeric address in an IPv4 or IPv6 header with application-provided identifier strings, embedded as tags in an enhanced IP Loose Source Route (LSR) option. Applications explicitly provide the forwarding information in the tags. A DataRouter indexes a forwarding table to identify the rewriting rule and rewrites the tag.

DataRouter's application-directed forwarding at the network layer has several advantages. Forwarding in kernel is much more efficient than forwarding at the user level (see Figure 2). Existing protocols for transport (TCP, UDP etc.), routing (RIP, BGP etc.) and security (IPsec) can be reused, which eliminates the need for reinventing the corresponding functionality in the application layer. The DataRouting facility and the code base can be shared by multiple overlays at the same time. By concatenating tags belonging to different overlay systems, it is possible to integrate different overlay systems without needing a gateway.



(a) IP LSR processing **(b) DataRouter tag processing**

Figure 1 Loose Source Route processing with DataRouter tags

DataRouter tags are embedded in an enhanced IP Source Route option (see Figure 1). The handling of the source route of tags entries is similar to IP Loose Source Route except that the DataRouter tags go through a lookup and rewrite procedure to determine the DataRouter next hop and the final content of the option. DataRouter tag properties include (1) Routing class identifying the overlay (e.g., song hashes, www URLs), and (2) Rewriting rule matching policy: exact, longest, first, range etc. and (3) Value: the tag string itself. Applications can set or read the tags. The tag rewriting rules are stored in a kernel forwarding table and can be manipulated by the application.

A preliminary implementation of the DataRouter has been completed in FreeBSD 5.0, using a new IPv4 option. It includes a preliminary API for inserting and reading tags and configuring rewriting rules. Applications can set one or more tags through *setsockopt()* system call or through a user-level library. The source route option containing the tag(s) is inserted into all packets originating in that socket. Rewriting rules are stored in a kernel forwarding table using the *droute* command.

On a dual-processor 2.4 GHz Xeon PC with 64-bit/66 MHz PCI gigabit Ethernet cards and running the FreeBSD 5.0, IPv4 packets are forwarded at 332K packets/sec, indicated as "IP/reg" in Figure 2. Forwarding the same packets on a kernel with DataRouter extension support, i.e., where the DataRouting capability is present but not used, does not affect performance measurably. DataRouted packets forwarded based on an exact match of a 32-bit hash decreases performance to around 271K packets/sec (Hash/DR), and forwarding based on a regular expression (in this case, "*.(isi|usc).edu") results in 149K packets/sec. (RE/DR). These are simple baseline experiments, in which all packets for a test have the same header, and the forwarding tables have only one entry of each type (regular longest-prefix, hash, and regular expression).

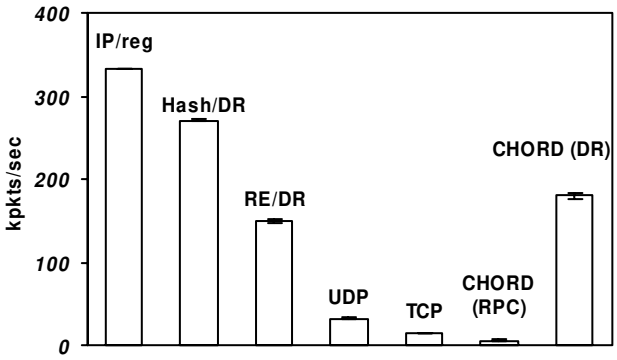


Figure 2 Forwarding rates for IP packets and Chord messages

A preliminary integration of Chord and DataRouter has been completed. Chord is able to forward 5,500 messages/sec using UDP as the transport protocol on the above mentioned platform, indicated as "CHORD (RPC)" in Figure 2. Chord integrated with DataRouter (CHORD (DR)) is 30x faster, at 180K messages/sec.

Analysis of Gradient-based Routing Protocols in Sensor Networks

Jabed Faruque[1] Konstantinos Psounis Ahmed Helmy[1]

Department of Electrical Engineering
University of Southern California, Los Angeles, CA 90089, USA
{faruque, kpsounis, helmy}@usc.edu

Sensor networks may be most widely used for habitat and environmental monitoring where the attached tiny sensors sample various physical phenomena. Moreover, many of the physical phenomena follow the diffusion property with distance, i.e., $f(d) \propto \frac{1}{d^\alpha}$, where d is the distance from the point having the maximum effect of an event, $f(d)$ is the magnitude of the event's effect and α is the diffusion parameter that depends on the type of the effect; e.g., $\alpha = 2$ for light, and $\alpha = 1$ for heat. The routing protocols used in the sensor networks for habitat and environmental monitoring applications can exploit this natural information gradient to efficiently forward queries towards the source.

Real life sensors are not perfect and are subject to malfunction due to obstacle or sensor/node failures. Also, the characteristics of the sensor nodes, i.e., limited battery life, energy expensive wireless communication and unstructured nature of the sensor network, make data-centric routing protocols based on the information gradient a challenging problem. Several routing protocols have been proposed to exploit the information gradients in the sensor networks. These protocols use greedy forwarding and can be broadly classified as:

1. *Single path approach*[1, 3, 4], where the query reaches to the source from the sink through a single path.
2. *Multiple path approach*[2], where the query uses multiple paths to reach to the source.

In this poster, we do not aim to design new routing protocols per se. Rather, the objective of the research is focused on the evaluation and the analysis of the general approaches to route a query using the natural information gradient in the sensor networks. In the analysis we are interested in two metrics:

1. *Reachability*, i.e., the success probability, which is the probability that the query initiated from the sink will reach to the source.
2. *Overhead* in terms of average energy dissipation, which is the number of transmissions required to forward the query to the source and to get the reply using the reverse path.

For simplicity of the analysis, we use a simple grid topology. Using probability tools and combinatorics we develope simple analytical models of the reachability and the overhead for both approaches. Further, we simulate the protocols which are designed based on these two approaches in more realistic scenarios. Through simulation, we also investigate the *path quality* in terms of the path-length increasing factor. We only consider sensor networks with static nodes, which is usually the case for environmental monitoring. Last, we assume that the queries are triggered after the event's occurrence.

Comparison of both approaches using analytical and simulation results are presented in the poster. From the results it is found that in our model the multiple path approach is energy efficient when the source is 25 hops away from the querier; otherwise, the single path approach is preferable, though the reachability reduces. Also, the multiple path approach results in shorter paths than the single path approach, and the resulting paths are quite close to the shortest possible paths.

[1]Partially supported by Pratt & Whitney UTC Institute for Collaborative Engineering(PWICE) project, Intel and NSF CAREER awards.

1. REFERENCES

[1] M. Chu, H. Haussecker, and F. Zhao. Scalable information-driven sensor querying and routing for ad hoc heterogeneous sensor networks. *Int'l J. High Performance Computing Applications*, Fall 2002.

[2] J. Faruque and A. Helmy. RUGGED: RoUting on finGerprint Gradients in sEnsor Networks. In *ICPS*, July 2004.

[3] Q. Li, M. DeRosa, and D. Rus. Distributed algorithms for guiding navigation across a sensor network. In *MobiCom*, September 2003.

[4] J. Liu, F. Zhao, and D. Petrovic. Information-directed routing in ad hoc sensor networks. In *WSNA*, September 2003.

RFID Wake-up in Event Driven Sensor Networks

Primož Škraba
Student
Dept. of Electrical Engineering
Stanford University
primoz@stanford.edu

Hamid Aghajan
Faculty
Dept. of Electrical Engineering
Stanford University
hamid@wsnl.stanford.edu

Ahmad Bahai
Faculty
Dept. of Electrical Engineering
Stanford University
bahai@stanford.edu

Wireless sensor networks (WSN) are most often envisioned with a large number of nodes and their typical range of communication is on the order of tens of meters. By definition, WSNs consist of nodes monitoring their environment and taking action when some event is detected. The action could range from reporting this change to some central or mobile node, to implementing a local control loop through local actuators, either at the nodes themselves or at other local nodes. How the data is aggregated and what is done with it are both problems that are application specific. The common factor in these problems is that they are event driven. It is changes in the environment that cause the nodes to change from local monitoring points to a network of information.

Nodes are generally small and cheap with extremely limited energy reserves. All the energy for sensing, processing, and communication must come from either small batteries or energy scavenging. The energy capacity of batteries is increasing slowly and in most cases, energy scavenging techniques do not supply a steady power source. A major source of power consumption in nodes is communication. It has been found that over short distances, a radio listening to the channel consumes power on the same order of magnitude as the actual receiving or transmitting. The result is that idle listening becomes the dominant power consumer in the system.

Thus, when no changes are detected and there is nothing to tranmsit, it makes sense to turn off the node radios. However, this creates problems with communication. Often it is necessary for nodes that are not directly involved with sensing an event to be involved in communication, such as when acting as a multihop relay node, or to do some local computation or data aggregation. Thus the network must be able to enable communication with nodes that have not sensed any events.

It would be ideal if it were possible to send a message to a node without the penalty of having to listen to the channel. Radio frequency identification tags (RFID) can provide this functionality, but require higher transmit powers. RFIDs are also becoming increasingly common in commercial inventory management systems and are proving both robust and effective. In the context of this paper, they are used as the basis of an energy efficient out-of-band wake up mechanism. A distinction of the proposed RFID scheme is the lack of a high-power reader system, since our scheme is based on an ad-hoc scheme of employing RFID devices, which does not rely on hearing back from the called tags and hence the initial call transmit power is much lower than in regular RFID calls.

In this work, we present an analytical model for both the RFID wake-up scheme and sleep schedules/random scheduling. Using a novel event-driven traffic model, the RFID scheme is compared to both random unsynchronized scheduling and synchronized sleep scheduling. The average power consumption and the delay of establishing either a point-to-point or multicast communication link is found in terms of the event rate and scope.

It was found that for low event rates, even under very low duty cycles, for sleep schedules/random scheduling, the idle listening power dominates the communication power consumption. An optimization is then performed, over the active duty cycle length and total duty cycle length, to find the operating curve of random scheduling in terms of power and delay.

The analysis for the RFID wake-up scheme illustrates the behaviour of RFID wake-up in terms of power and delay with different system and network parameters. In particular, two versions of RFID are examined. The first is a simple RFID, which only gives the capability to broadcast a wake-up signal to all of a node's neighbors. The second is a more advanced wake-up mechanism, where specific subset of nodes can be woken. Performance for all variants is found for different levels of robustness in the RFID system and the network density.

The poster, this abstract, and a list of references can be found at the following url:
http://www.stanford.edu/∼primoz/sigcomm04/

Exploiting Distance-Indexed IP Traceback Schemes

Lin Cai Jianping Pan Sherman Shen

Denial-of-Service (DoS) attacks and their distributed variants (DDoS) have become a serious threat to the healthy proliferation and growth of the Internet. Most Internet service providers, including many high-profile ones, have suffered severe DoS/DDoS attacks, and have sustained a considerable loss of service capabilities, customers, and revenues. Besides other causes, the lack of *source accountability* in the TCP/IP protocol stack is a major concern that sometimes even encourages these attacks. Attackers can easily forge their identities (usually the source IP address, protocol identifier, and port number of their outgoing packets) when they have no intention to obtain services from DoS/DDoS victims, but just want to prevent legitimate users from doing so. The so-called *source spoofing* does not affect the destination-oriented Internet routing fabric that successfully transports both attack and legitimate flows to victims.

Source traceability is a victim-oriented approach to achieve source accountability. With the assistance of anomaly and intrusion detection tools, victims or their agents first identify attack flows, and then initiate a request that traces back toward the real sources of these flows. Traceback can occur in higher layers (e.g., by correlating SMTP server signatures in email headers), but the traceability of IP packets is essential, due to the fact that many DoS attacks do not exchange application-layer data at all. However, IP-level traceback is much more challenging, since each IP packet is self-contained and can carry different source identities even from the same attack source. An *ideal* traceback scheme should correlate attack packets efficiently, identify or isolate attack sources effectively, and more importantly, allow an incremental deployment. In addition, such a scheme should be lightweight and only impose minimal changes to existing Internet infrastructures (especially in core routers).

Many IP traceback schemes [1] have been proposed in the last few years. In terms of how traceback characteristics are extracted and where the information is stored, most schemes follow one of the following two approaches: *router stamping* and *packet stamping*, which emulate the traceback techniques in postal and telephone systems, respectively. In router-stamping schemes (e.g., [2]), a router identity (or its fraction) is stored in packets when they travel through routers; victims collect these stamped packets, and after having enough stamps, recover a reverse path toward attack sources, which is identified by the stamps of traversal routers. In packet-stamping schemes (e.g., [3]), routers keep a copy (or a digest) of forwarded packets for a while; victims should initiate a traceback request within a certain time-period, which is facilitated by a traceback authority consulting routers that still have the matching packet stamps. In general, packet stamping incurs higher computation and storage overhead in routers; therefore, router stamping appears more attractive for IP traceback schemes.

In this poster, we focus on the router-stamping approach. We first discover an overlooked *buffer overflow* vulnerability that is intrinsic to many distance-indexed IP traceback schemes. When there is limited space in packets to store stamps, traversal routers only stamp packets with a given probability, in order to preserve the already-inscribed stamps made by upstream stamping routers. Also, these stamps are indexed by the incremented distance to the packet destination, in order to allow victims to *independently* recover a *reverse* path from the destination in a *hop-by-hop* manner. To promote an incremental and favorable deployment, these schemes have to follow open protocols and adopt well-known parameters, which are also available to attackers. Substantiated by extensive efficacy analysis and numerical results, we design *extension*, *split*, *branch*, and *synthesized* exploits that can actually take advantage of the buffer overflow vulnerability by creating different types of forged reverse paths in a very efficient manner when compared with the traceback effort attempted by victims. These forged paths are statistically indistinguishable from the genuine ones; in addition, with unconstrained exploits, even genuine paths appear to be forged from the viewpoint of victims. Consequently, we show that the design goal of these traceback schemes can be seriously compromised in practice.

Moreover, we discuss the distance-related vulnerability in a general context relevant to network protocols, and examine a few possible alternatives. We also point out that the overflow vulnerability in distance-indexed IP traceback schemes cannot be effectively eliminated by some *quick fixes* (e.g., dropping or flagging overflowed packets) without interfering with their stateless, low-overhead, and incrementally-deployable design, *unless* the distance increment procedure becomes overflow-resistant. We still agree that IP-level traceback schemes are essential and promising to circumvent ever-increasing DoS/DDoS attacks, if these schemes are properly designed, developed, and deployed; therefore, it is vitally important to identify and understand their intrinsic weaknesses, if any, at the earliest possible stage. This poster is an attempt to serve this purpose.

L. Cai and S. Shen are with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada. E-Mail: {cai,xshen}@bbcr.uwaterloo.ca. J. Pan is now with NTT MCL, Palo Alto, CA, USA. E-Mail: panjianping@acm.org.

REFERENCES

[1] A. Belenky and N. Ansari. On IP traceback. *IEEE Communications Magazine*, 41(7):142–153, 2003.

[2] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Practical network support for IP traceback," *Proc. of 16th ACM SIGCOMM (SIGCOMM'2000)*, pp. 295–306, 2000.

[3] A. Snoeren, C. Partridge, L. Sanchez, and C. Jones, "Hash-based IP traceback," *Proc. of 17th ACM SIGCOMM (SIGCOMM'01)*, pp. 3–14, 2001.

Exploiting the Transients of Adaptation for RoQ Attacks on Internet Resources*

Mina Guirguis
Dept. of Computer Science
Boston University
Boston, Massachusetts
msg@cs.bu.edu

Azer Bestavros
Dept. of Computer Science
Boston University
Boston, Massachusetts
best@cs.bu.edu

Ibrahim Matta
Dept. of Computer Science
Boston University
Boston, Massachusetts
matta@cs.bu.edu

Motivation and Overview of Work: Over the past few years, Denial of Service (DoS) attacks have emerged as a serious vulnerability for almost every Internet service. An adversary bent on limiting access to a network resource could simply marshal enough client machines to bring down an Internet service by subjecting it to sustained levels of demand that far exceed its capacity, making that service incapable of adequately responding to legitimate requests. In that sense, DoS attacks can be regarded as exploiting the system's steady-state capacity. In this work, we turn our attention to unorthodox adversarial attacks, which we term Reduction of Quality (RoQ) attacks, that exploit the transients of a system's adaptive behavior, as opposed to its limited steady-state capacity. Modern computing and networking systems rely on adaptation to drive the system into quiescent regions of operation that would maximize the overall system's gain, in addition to being stable, fair and efficient. In this work, we analytically capture the effect of RoQ attacks that would deprive an Internet element from reaching steady state by knocking it off whenever it is about to stabilize. For instance, an attacker can continually disturb the stability of a router by affecting the congestion signals (prices) fed back to the rate-adaptive sources. We formalize the notion of attack "potency", which exposes the tradeoff between the "damage" inflicted by an attacker (*e.g.*, waste in bandwidth) and the "cost" of the attack (*e.g.*, average attack rate). Moreover, our notion takes aggressiveness into account(*i.e.*, the level of exposure risk that the attacker is willing to take), enabling us to identify different families of DoS attacks based on their aggressiveness. We give examples of RoQ attacks on a number of common adaptive components currently incorporated in computing and networking systems. But below, we only focus on network adaptation, keeping in mind the bigger range of applicability of RoQ attacks on other systems.

Network Adaptation Mechanisms and Vulnerabilities: End system protocols (*e.g.,* TCP) rely on feedback mechanisms to adapt their sending rates to match their "fair share" of network resources. Buffer management schemes play an important role in the effectiveness of transmission control mechanisms as they constitute the feedback signal (by marking or dropping packets) to which such mechanisms adapt. Active Queue Management (AQM) techniques have been developed that try to maintain the queue size at a target level and employ probabilistic dropping. Stabilizing the queue at a low target guarantees efficiency while minimizing jitter and round trip time in general.

Attack Definition: We focus on attack techniques that would hinder an AQM from stabilizing its queue, and hence resulting in a noisy feedback signal to the end-system transmission controllers, which in turn would lead to high jitters due to oscillations, unfairness as well as inefficiencies due to queue drainage, *i.e,* the input rate can't saturate the link capacity. For simplicity, we consider an attack comprising a burst of M packets transmitted at the rate of δ packets per second over a short period of time τ. This process is repeated every T units of time.

Attack Goal: We define Π, the *attack potency*, to be the ratio between the *damage* caused by that attack and the *cost* of mounting such an attack. Clearly, an attacker would be interested in maximizing the damage per unit cost—i.e., maximizing the attack potency.

$$\text{Potency} = \Pi \quad = \quad \frac{\text{Damage}}{\text{Cost}^{\frac{1}{\Omega}}}$$

The above definition does not specify what constitutes "damage" and "cost". In this work , we consider various instantiations of these metrics(e.g., bandwidth, delay jitter, etc..). Ω is introduced to model the aggressiveness of the attacker. Our results confirm that RoQ attacks can degrade the performance of any AQM scheme, degenerating them to Drop-Tail, while injecting the minimum attack traffic. Moreover, RoQ attacks can achieve higher potency than "shrew" (targeting timeouts in TCP) and DoS attacks (brute-force).

Vulnerability Assessment: We used a control-theoretic model to underline the complex interplay between the efficiency-load behavior of a resource and the adaptation mechanisms of both the resource and its consumers. The adaptation is modeled as an optimization process driving the system to a quiescent stable operating point. An optimized RoQ exploit would then keep the system oscillating between different states, in presence and absence of the attack traffic. We developed associated metrics to quantify the system's vulnerabilities. We present numerical and simulation results, which we validate with observations from real Internet experiments. We are currently investigating adaptation mechanisms that are more resilient to these new forms of attacks by exploiting tradeoffs between performance and vulnerability. Our plan is to develop efficient techniques for the detection of RoQ exploits when they do occur as well as invoking proper counter-measures.

URL: `http://cs-people.bu.edu/msg/research/roq`

Reference: M. Guirguis, A. Bestavros and I. Matta. *Exploiting the Transients of Adaptation for RoQ Attacks on Internet Resources*, To appear in ICNP 2004, Berlin, Germany.

*This work was supported in part by NSF grants ANI-0095988, ANI-9986397, EIA-0202067 and ITR ANI-0205294.

An Analysis of Location-Hiding Using Overlay Networks

Ju Wang (student)
Department of Computer Science and Engineering
University of California, San Diego
jwang@cs.ucsd.edu

Andrew A. Chien (faculty)
Department of Computer Science and Engineering
University of California, San Diego
achien@ucsd.edu

ABSTRACT

Overlay networks have been proposed as a means to achieve application location-hiding. In particular, overlay networks are used as proxies which mediate communication between applications and their users without revealing application IP addresses. The capability to communicate without revealing IP addresses is also known as location-hiding or application hiding, and its essence is indirect communication. This capability can support anonymous communication, protect applications and hosts from direct attacks, and supporting physical infrastructure from Denial-of-Service (DoS) attacks. For example, many researchers [1-4] exploit this location-hiding capability to protect Internet applications from DoS attacks on application's supporting physical infrastructure.

However, fundamental questions about such proxy networks remain unanswered, especially in the presence of intelligent attackers: Can proxy networks achieve location-hiding via indirection? If so, under what circumstances can they stably withstand attacks? How long will it take attackers to penetrate a proxy network and reveal application location?

To shed light on these questions, we develop a generic framework for proxy network approaches to location-hiding, which encompasses most of the proposed approaches. We also develop a stochastic model to characterize the dynamic behavior of the system — exploring in particular how attacks, defense mechanisms and correlated host vulnerabilities affect stability (the ability to resist an attack). Based on this framework and model, using theoretical analysis combined with simulation techniques, we analyze the behavior of proxy network systems and characterize when location hiding is feasible and when it is not.

We focus on classes of attacks that exploit the connection structure of the proxy networks, and use directed penetration in an attempt to reveal the application's hidden location. Such attacks focus on elements that are present in all proxy network approaches.

Our specific research contributions include:

- design of a generic framework and analytic model for proxy network approaches to location-hiding,
- based on the model, we characterize several fundamental properties of the proxy network-based location-hiding, showing that existing approaches employing static structure against host compromise attacks are infeasible,
- based on the model, proxy-network-based location-hiding can be successful if it includes proactive defenses employing proxy network reconfiguration and migration. The proxy network depth and reconfiguration rates are the critical factors for achieving the effectiveness.
- using simulation techniques, we explore cases where host vulnerabilities are correlated, showing that in many cases this distinction makes proxy-based location-hiding infeasible.
- finally, we show that with intelligent exploitation, only limited host (OS/software) diversity is needed to mitigate the negative impact of correlated vulnerabilities and location-hiding can be achieved.

These results provide both deeper understanding of the location-hiding problem and guidelines for proxy network design. The generic framework provides a foundation to understand proxy networks' capability of location-hiding, to formally analyze the behavior of such systems, to rigorously reason about how proxy networks should be designed, and serves as a foundation for future studies employing more complex and realistic models.

References

1. Adkins, D., et al., *Towards a More Functional and Secure Network Infrastructure*. 2003, Computer Science Division, UC Berkeley: Berkeley.
2. Adkins, D., et al. *Taming IP Packet Flooding Attacks*. in *HotNets-II*. 2003.
3. Andersen, D.G. *Mayday: Distributed Filtering for Internet Services*. in *4th Usenix Symposium on Internet Technologies and Systems*. 2003. Seattle, Washington.
4. Keromytis, A.D., V. Misra, and D. Rubenstein. *SOS: Secure Overlay Services*. in *ACM SIGCOMM'02*. 2002. Pittsburgh, PA: ACM.

Related publication and project information can be found at http://www.cs.ucsd.edu/~jwang/

Impact of Routing Protocol and Routing Policies on Internet Resilience

Feng Wang and Lixin Gao
Department of Electrical and Computer Engineering, University of Massachusetts
Amherst, MA 01002
{fewang, lgao}@ecs.umass.edu

Studies have shown that the Internet has experienced the widespread reachability problem [1] [2]. Most of them are transient failures. In Figure 1, we show failure durations at four ASs in August 2003. The data was collected from Oregon RouteView [3]. From this figure, we find that more than half of failures last for less than 90 seconds. In this paper, we study the potential causes of transient failures.

One potential cause of transient failures is that Border Gateway Protocol (BGP) is a path vector routing protocol, in which each AS advertises only its best path to its neighbors. When an AS's neighbor uses the best route sent by the AS, the AS cannot get the best route from the neighbor due to *loop avoidance property* of BGP protocol. For example, Figure 2(a) represents a simple example, in which each node represents an AS, while the edge represents the connectivity between a pair of ASs. In Figure 2(a), AS2's neighbor AS1 selects the path (1 2 0) via AS2 as the best path to the destination d. As a result, AS2 does not know the path (1 0) at AS1. This property can lead to transient failures. For example, if the link between AS0 and AS2 fails, both AS2 and AS3 will experience transient failures. Such failures will last until AS2 and AS3 learn the path (2 1 0) from AS1 and (3 2 1 0) from AS2 respectively. This example shows that the loop avoidance property of BGP protocol could be one potential cause of transient failures.

Another potential cause of transient failures is that BGP is a policy-based routing protocol, in which each AS determines whether to announce a route to its neighbor according to routing policies. The decision is usually determined by AS commercial agreements. An AS typically does not carry provider to provider or peer to peer traffic, which is known as no-valley policy [4] [5]. For example, Figure 2(b) represents an AS graph, in which the edge between a pair of ASs represents their AS relationship. AS0 announces prefix d to both providers, AS1 and AS4. AS4 chooses the path (4 1 0) as the best path. The path (4 0) in AS4 is invisible to AS1 because of loop avoidance property of BGP protocol. Furthermore, AS4 cannot propagate the path (4 1 0) to its peers AS2 and AS3 due to the no-valley policy. As a result, AS2 and AS3 have only one path to d. The no-valley policy can make certain paths in an AS invisible to its neighbors. If the link between AS1 and AS0 fails, AS1, AS2 and AS3 will experience a transient failure until they learn the alternate path from AS4. So routing policies could be a potential cause of transient failures as well.

Additionally, the examples in Figure 2 show that transient failures can be propagated to other ASs. For example, in Figure 2(a), the transient failure occurred at AS2 can cause AS3 to experience a transient failure. However, the failure duration at AS3 could be longer than that at AS2. The reason is that AS3 is further away from AS1 than AS2, and AS1 provides the alternate path. Therefore, some ASs could experience longer transient failures than others.

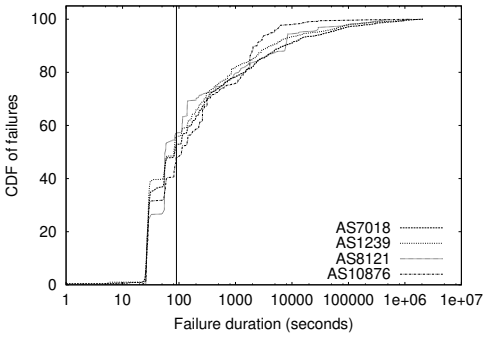


Fig. 1. Distribution of failure durations.

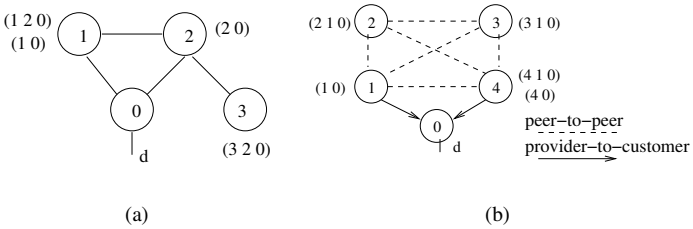


Fig. 2. Examples of transient failures. The paths beside each node indicate available paths to d, and the first one is the best path.

Our study shows that BGP routing protocol and routing policies play a critical role in ensuring the robustness of the Internet. BGP routing protocol and routing policies could cause certain paths invisible to some ASs. They could be the potential causes of transient failures. Furthermore, widespread anomalies such as worm attacks can exacerbate the extent of transient failures. During worm attacks, a large number of BGP sessions in the Internet are reset, which can result in a significant number of routing dynamics. Those dynamics will invoke even more transient failures. Our work can lead to the guidelines for setting routing policies to reduce transient failures in the Internet.

REFERENCES

[1] C. Labovitz, G. Malan, F.Jahanian. Internet Routing Instability. Proc. of SIGCOMM, September 1997.
[2] C. Labovitz, A. Ahuja, F. Jahanian. Experimental study of Internet stability and wide-area network failures. Proc. of Fault Tolerant Computing Symposium, June 1999.
[3] http://www.oregon.com
[4] L. Gao, On Inferring autonomous system relationship in the Internet. IEEE/ACM Transactions on Networking, December 2001.
[5] L. Gao, J. Rexford, Stable Internet routing without global coordination. ACM SIGMETRICS, 2000.

Robust Forwarding in Structured Peer-to-Peer Overlay Networks

Wang-kee Poon
Department of Computing
The Hong Kong Polytechnic University
cswkpoon@comp.polyu.edu.hk

Rocky K. C. Chang
Department of Computing
The Hong Kong Polytechnic University
csrchang@comp.polyu.edu.hk

ABSTRACT

Structured peer-to-peer overlay networks are widely exploited to build large-scale decentralized applications. Robustness to malicious nodes is an important and fundamental objective of designing an overlay network. The presence of even a small percentage of malicious nodes can present a serious threat to the stability of an overlay network. Existing approaches mainly utilize multiple paths to increase the probability of delivery. In this paper, we propose *Fence* that allows a source to diagnose possible forwarding faults injected by malicious nodes. Based on the feedback information, the source is able to identify malicious nodes and to achieve very robust message forwarding.

1. BACKGROUND

Structured peer-to-peer (p2p) overlay networks, such as CAN, Chord, Pastry, and Tapestry, provide distributed lookup services for large-scale decentralized applications. A major function of the overlay networks is to efficiently lookup the node corresponding to a key. While many lookup mechanisms are quite efficient, they are vulnerable in the presence of malicious nodes. Existing implementations do not address this problem directly, except for providing a time-out based retransmission scheme.

2. PROBLEM

A malicious node can disturb a normal delivery of a lookup message by message dropping, delayed forwarding, or incorrect forwarding. Only a small percentage of such malicious nodes is needed to seriously degrade the forwarding performance. In a simulation study of Pastry, the probability of a successful lookup is dropped to 65% when there are 10% malicious nodes [1]. Existing solutions mainly involve forwarding a message through multiple paths, with the hope to increase the probability of delivery. In this paper, we take another approach that involves first identifying the malicious nodes, and then performs appropriate actions to bypass them.

3. APPROACH

In this paper we propose *Fence* to diagnose *forwarding faults* on a forwarding path that are injected by malicious nodes. Forwarding faults include undesirable delay, message dropping, and forwarding to undesirable nodes. Once a source discovers a fault through Fence, it can perform immediate actions, such as fast recovery, fault isolation, and message blocking.

Fence allows a source to monitor each forwarding step of its own lookup messages. Using Fence, a source expects to receive from each forwarding node a *proof* for each successful forwarding of its messages. Based on the proof, the source can learn 1) the duration that the message has stayed at each forwarding node, 2) whether the forwarding is successful, and 3) the identity of the next forwarding node. It turns out that these three pieces of information are sufficient to allow the source to identify forwarding faults.

Based on the inference on the status of its messages, a source can perform the following actions to provide a very robust forwarding service in the presence of malicious nodes.

- **Fast Recovery** A source can roll back to the previous hop to continue forwarding immediately after identifying a forwarding fault.
- **Fault Isolation** A source, after identifying a list of malicious nodes, may inform other forwarding nodes about this list, so that these malicious nodes can be eventually isolated from the network.
- **Message Blocking** A node may ignore lookup messages sent from malicious nodes. In this case, the malicious nodes may include selfish nodes which do not dutifully provide forwarding service. Therefore, blocking the selfish nodes' messages would serve as an incentive for a more nonselfish behavior.

Further information can be accessed from:
http://www.comp.polyu.edu.hk/∼cswkpoon/fence

4. REFERENCES

[1] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D. S. Wallach. Security for structured peer-to-peer overlay networks. In *5th Symposium on Operating Systems Design and Implementaion (OSDI'02)*, Dec. 2002.

Double Sense Multiple Access for Hidden Terminal Avoidance in Wireless Ad Hoc Networks

Feiyi Huang, *Student*, and Yang Yang, *Supervisor**

Poster URL: http://www.brunel.ac.uk/∼eestyyy/dsma.htm

Double Sense Multiple Access

In wireless ad hoc networks, it is a challenging problem to design an efficient media access control (MAC) protocol that can completely avoid the interference from the hidden terminals, which are defined as the terminals beyond the communication range of the transmitter but within that of the receiver. In [1], Haas and Deng have proposed the dual busy tone multiple access (DBTMA) protocol where the transmitter and the receiver can clear other packet transmissions in their communication ranges by broadcasting two different busy tone (BT) signals. To avoid the still possible packet collisions due to the propagation delay, a mandatory waiting time is inserted between the successful reception of a ready-to-send (RTS) packet and the transmission of a data packet. In addition, all terminals are required to keep sensing the BT signals, even while they are transmitting packets.

In this paper, we propose and analyse a novel dual channel random access protocol, called "Double Sense Multiple Access" (DSMA), for solving the hidden terminal problem. Two time-slotted channels are used for transmitting control (e.g. RTS) and data packets separately. Under DSMA, a transmitter will sense the BT signals twice before sending the data packet to the receiver. In doing so, the transmitter can identify if it is the intended transmitter for sending a data packet and, therefore, the possible data packet collisions due to propagation delay can be completely avoided. Compared with DBTMA, our DSMA protocol has the following advantages: (1) higher channel efficiency by completely avoiding the collisions between control and data packets; (2) higher power efficiency by setting lower data rate, i.e. lower transmission power, in the control channel; (3) shorter transmission delay by removing the "mandatory waiting time" in DBTMA; and (4) less complexity and cost by reducing the "keep sensing" requirement in DBTMA to "sense twice" in DSMA.

Throughput Analysis

The analytical results given in [1] are for a fully connected network wherein no hidden terminal exists. In contrary, we consider a realistic non-fully connected network and have derived the throughput S of DSMA as follows.

$$S = \frac{G \cdot \delta e^{-\gamma G}}{(\delta + 4) \cdot G e^{-\gamma G} + e^{(\gamma - 1) G}},$$

where γ and δ denote, respectively, the lengths of control packet and data packet. G is the offered traffic.

Fig. 1 and Fig. 2 show the throughput S versus offered traffic G with γ and δ as parameters, respectively. The analytical results shown in solid lines are perfectly verified by the simulation results in markers ($\gamma = 3$ and $\delta = 20$). As expected, the throughput performance is closely related to the ratio between γ and δ. The smaller the ratio γ/δ, the higher the throughput curve we can obtain. These curves are very useful for traffic sizing, especially when the radio channel effects are taken into account.

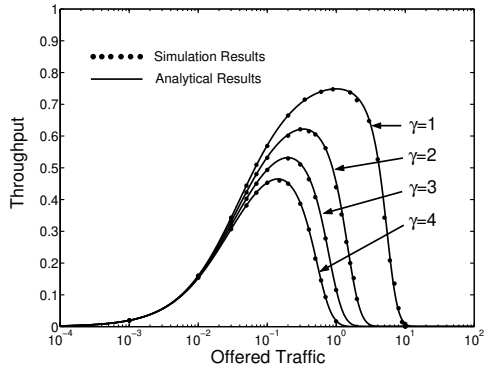


Figure 1: Throughput, $\gamma = 1, 2, 3, 4$, $\delta = 20$.

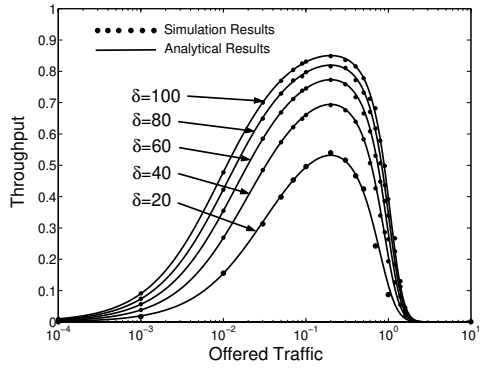


Figure 2: Throughput, $\gamma = 3$, $\delta = 20, 40, 60, 80, 100$.

We are currently studying the delay and stability performance of DSMA and the results will be available soon.

References

[1] Z. J. Haas and J. Deng, "Dual busy tone multiple access (DBTMA) – a multiple access control scheme for ad hoc networks," *IEEE Transactions on Communications*, vol. 50, pp.975–985, June 2002.

*The authors are with the Department of Electronic and Computer Engineering, Brunel University, Uxbridge, London, UB8 3PH, U.K. (e-mail: *dc03ffh@brunel.ac.uk* and *yang.yang@brunel.ac.uk*).

Strong Cache Consistency Support for Domain Name System

Xin Chen, Haining Wang, Shansi Ren, and Xiaodong Zhang
College of William and Mary
{xinchen,hnw,sren,zhang}@cs.wm.edu

***Abstract*— Caching is critical to the performance and scalability of Domain Name System (DNS). However, DNS only supports weak cache consistency by using Time-To-Live (TTL) mechanism. Without strong cache consistency among DNS servers, it is not only cumbersome and inefficient to invalidate out-of-date cache entries, but also highly likely to lose connection services due to cache inconsistency. The loss of service availability is a much more serious problem than the service degradation. We propose an active DNS cache update protocol, called *DNScup*, to maintain strong cache consistency. Our trace-driven simulation and prototype implementation demonstrate that *DNScup* achieves the strong cache consistency of DNS, and hence, significantly improves its availability, performance and scalability.**

I. Motivation

DNS performance has been well studied. However, none of previous work focuses on the DNS cache consistency. The DNS cache inconsistency may induce the loss of service availability, which is much more serious than performance degradation. To investigate the dynamics of domain-name-to-IP-address (DN2IP) mapping changes, we have conducted a wide range of DNS measurements. The purpose of our DNS dynamics measurement is to answer the question about how often a DN2IP mapping changes. In general, a mapping change may cause two different effects. If the original DN2IP mapping is one to one, the change may lead to the loss of connection services. We call this kind of changes as physical changes. However, if the original DN2IP mapping is one to many, the change may be anticipated to balance the workload of a web site as CDN does. We classify these changes as logical changes.

II. Measurement

Since the DNS is predominately used by Web sessions to resolve the IP addresses of Web sites, our measurements are focused on the dynamics of the mappings between Web domain names and their corresponding IP addresses. We collected the Web domain names from the recent IRCache proxy traces between November 6, 2003 to November 12, 2003. Based on its TTL value, each domain name in our collection is periodically resolved to check if the mapping has been changed. What we have found are summarized as follows:

- While physical mapping changes per Web domain name rarely happen, the probability of a physical change per minute within a large number of Web domains is certain to one;
- Compared with the frequencies of logical mapping changes, the corresponding TTLs are set to much smaller values, resulting in redundant DNS traffic;
- The TTL value of a Web domain name is independent of its popularity, but its logical mapping change frequency does depend on the popularity of the Web domain.

Based on our measurements, we conclude that maintaining strong cache consistency is essential to prevent the potential losses of service availability, especially for critical or popular Internet services. Furthermore, with the strong cache consistency support, CDNs may provide fine-grained load-balance with reduced DNS traffic.

III. DNScup

To reduce the storage overhead and communication overhead, we introduce the dynamic lease technique for maintaining DNS cache consistency. A DNS cache keeps track of the local query rate of a cached DNS record. The authoritative DNS name server grants the lease of a DNS record to the DNS cache on-the-fly based on its query rate. The lease duration is determined by the record's DN2IP change frequency. Overall, the activities of *DNScup* include: (1) the query rate estimation at the cache-side; (2) the dynamic lease granting at the server-side; and (3) the communication between the DNS cache and the authoritative DNS server. We use trace-driven simulations to evaluate the effectiveness of dynamic lease of DNScup.

As an extension to DNS Dynamic Update protocol that maintains a consistency among a master DNS sever and its slaves within a single zone, we build a *DNScup* prototype with minor modifications on top of BIND 9.2.3. The components of *DNScup* implementation include the detecting module, the listening module, the notification module and the track file. In order to deal with the wide area DNS cache update propagation, we define a new type of message called `CACHE-UPDATE`. The interactions among all components are illustrated in Figure 1.

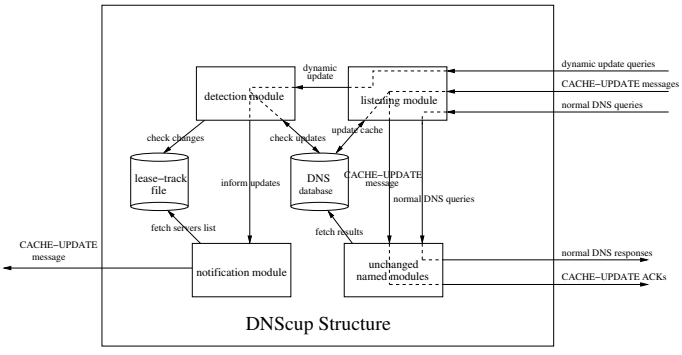


Fig. 1. The Structure of *DNScup* Prototype

Our trace-driven simulation and prototype implementation demonstrate that *DNScup* achieves the strong cache consistency in DNS and significantly improves its availability, performance and scalability.
URL: *http://www.cs.wm.edu/∼xinchen/DNScup.html*

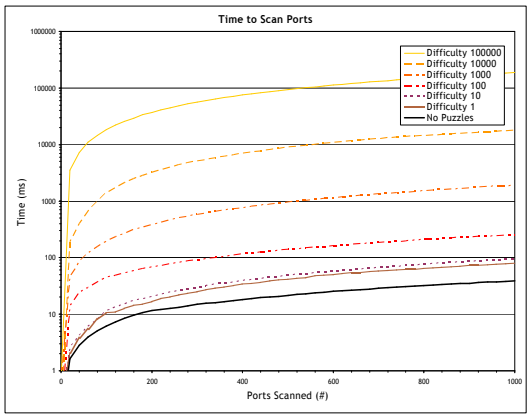
Reducing Malicious Traffic With IP Puzzles *

Ed Kaiser Wu-chang Feng Wu-chi Feng Antoine Luu
OGI@OHSU ENSEIRB
{*edkaiser, wuchang, wuchi*}*@cse.ogi.edu* *luu@enseirb.fr*

Amidst the traffic of the Internet is an enormous amount of undesirable communication. Currently there is no significant disincentive for clients who contribute to this flood of undesirable communication. A mechanism for punishing only the malicious is required in order to discourage clients from behaving badly. The standard response has been to disconnect clients exhibiting suspicious behavior from the rest of the network using a binary filter. Ideally though, the mechanism should be analog to allow falsely identified clients to prove that they are legitimate, so that service to them can be reinstated. Client puzzles have been proposed in several protocols as a mechanism well suited for this task; clients do all the work involved in proving their legitimacy.

Until now, client puzzles have been used only as an application layer defense against flooding attacks. Yet, client puzzles in the application layer can be thwarted if any adjacent or underlying protocol does not provide a similar defense. For example, DoS-resistant authentication protocols can be thwarted by basic UDP or IP flooding. Implementing client puzzles in TCP offers no protection from those flooding attacks either. Clearly, the network layer is the lowest layer vulnerable to distributed network flooding attacks. We argue that the network layer is the most defensible layer against flooding and other forms of distributed attacks. This poster describes the design and implementation of our *network layer* client puzzle protocol.

In addition to distributed flooding attacks, network layer puzzles can defend against attacks which have been undefendable until now. As an example, in-network reconaisance attacks such as port scans cannot be defended at higher layers; by the time any higher layer becomes aware of the attack, the attacker has obtained all the information sought. It is important to stop reconaisance attacks since the information gathered is used by worms to create the most potent attack topology possible, which means the difference between a severe network outage or a brief network congestion. We show that network puzzles can effectively throttle port scans.



When deploying a protocol in the network layer, it is important to be flexible about which devices must participate. Our protocol, which can be implemented within the fast path of network hardware, gives every network device along the path from the client to the server the choice of participating as a puzzle issuer or not. Being able to place puzzle issuers arbitrarily close to the client allows quenching undesirable traffic closer to the source, wasting fewer resources deeper within the network.

Another issue with puzzles is that the work load required for each puzzle must be adjusted to meet the real-time needs of a network. Throttling a malicious client cannot be done simply during the establishment of a connection; the need for puzzles may start and stop at any point in a flow's lifetime. Further, the puzzle difficulty required to keep services at maximum utilization changes frequently during the lifetime of any flow.

This poster describes our protocol as well as an `iptables` implementation that addresses these challenges. More information about the project can be found at:
`http://www.cse.ogi.edu/sysl/projects/puzzles`

*This material is supported in part by the National Science Foundation under Grant ANI-0230960 and the generous donations of Intel Corporation. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation or Intel.

Power Management schemes in LAN Switches

Maruti Gupta, Satyajit Grover, Suresh Singh
Computer Science Department
Portland State University
{mgupta,satyajit,singh}@cs.pdx.edu

We examine the feasibility of introducing power management schemes in network devices in the LAN. Currently, such schemes exist to minimize the power consumption on devices such as desktops, notebooks and a number of other portable devices. However, no such dynamic power management schemes are available for network devices such as routers and switches. In this work, we take a look at the feasibility of introducing power management schemes in LAN switches and address questions arising from putting various components on switch interfaces to sleep.

We choose to begin with LAN devices and in particular LAN switches for several reasons. Firstly, LAN switches comprise the bulk of network devices in the LAN and they also consume the largest percentage of energy (see Table I in [1]) among them. Second, a majority of interfaces on LAN switches are directly connected to hosts, that intuitively would have longer periods of low traffic activity than on those interfaces that carry aggregated traffic from many hosts. Our approach to power management in switches here involves powering off or putting to sleep LAN switch components, interfaces, or entire switches. However, powering devices back into normal or higher power consuming modes causes a spike in energy consumed as well as takes a certain amount of transition time. Hence, the decision to put devices to sleep must be made such as to not result in an overall increase in power consumption or cause performance degradation due to loss of network connectivity and packet drops. Another side effect of putting ports or switches to sleep is that protocols running at layer 2 and above may be negatively affected.

In our first step, we collected and examined traffic data from our campus network LAN to see if there are enough periods of inactivity at various switch interfaces to justify sleeping. Our data shows that there are significant number of intervals during low as well as high activity periods when individual switch interfaces can be put to sleep to intervals ranging from more than 20 seconds to about 1 second. To address the question on when an interface should be put to sleep and for how long, we developed an abstract sleep model where we define three different sleep states (other than ON and OFF) for switch interfaces based on the *functionality* of the interface in each of these states.

- *Simple Sleep:* In Simple Sleep, the interface sets a sleep timer, and only wakes up when the timer expires. All packets arriving during the sleep period are lost.
- *HAS:* In (Hardware Assisted Sleep), an incoming[1] packet wakes up the interface but is lost since it is not buffered.
- *HABS:* In (Hardware Assisted Buffered Sleep), an incoming packet wakes up the interface and is buffered, thus requiring the input buffer to remain powered on to buffer incoming packets.

We then used our sleep model to develop algorithms for sleeping. The sleeping algorithm makes the decision on whether to sleep or not depending upon the traffic activity seen so far. In the case of Simple Sleep, we observed that Simple Sleep would be best used on interfaces connecting hosts to switches, especially if the ACPI implementation on these hosts were modified to inform the switch of their sleeping decisions. HABS is suited to use for all interfaces, especially those that cannot tolerate packet loss such as interfaces connecting switches to other switches and routers. To make a decision on when to sleep, we used a traffic estimation scheme where we used the exponentially weighted moving average filter (EWMA) to derive estimated inter-activity intervals as well as knowledge of the periodic behavior of the protocols. We used the estimated value obtained to make decisions to sleep based on whether sleeping would save energy or not. Using energy values and transition times obtained from wireless LAN cards for idle, active and low power modes, we implemented our sleeping algorithms on trace data obtained for several interfaces and observed significant energy savings. Our results show that sleeping is indeed feasible in the LAN and in case of the HABS model, with little or no impact on other protocols. However, we note that in order to maximize energy savings while minimizing sleep-related losses, we need hardware that supports sleeping.

1. REFERENCES

[1] M. Gupta and S. Singh, "Greening of the Internet", *ACM SIGCOMM'03.*

[1] A incoming packet includes those coming from the wire as well as the packets switched to a particular switch port from other ports

Applying Repeated Games to Networking Problems

Mike Afergan (afergan@mit.edu)

1. OVERVIEW

In our research, we examine the impact of applying repeated game models to networking problems. Repeated interactions are prevalent in networking applications and well-studied in Economic literature. Here, we present the results of applying repeated game models to two well-known results. The first is the celebrated strategyproof routing mechanism of Feigenbaum *et al* (FPSS) [1]. The second is application overlay application trees. In both we find, through counter-examples, that the traditional models are unsatisfactory. However, from this negative result we find two *positive results* in the repeated framework. As such the contributions of our work are:

1. The application of repeated games to two well understood problems and recognized results.
2. The conclusion that the FPSS model is not strategyproof in a repeated environment, a significant problem for a routing protocol.
3. Analysis of the equilibrium conditions, which provide an understanding of the relationship between system parameters and the outcome. This in turn provides useful insights into how such systems should be built.

2. LEAST COST PATH ROUTING

The FPPS mechanism for inter-domain routing is strategyproof – each network bids truthfully. The mechanism pays a node k, on the Least Cost Path (LCP) from s to t, *(Cost_of_LCP(s,t)_Without_k - Cost_of_LCP(s,t)_With_k)+ $cost_k$*. For example, in Fig. 1, node A, on the LCP from s to t_1, is paid: $(1 + 10) - (1 + 1) + 1 = 10$. However, this mechanism is *not strategyproof in the repeated game*. In Fig. 1, A and B both can (implicitly) collude to bid higher, for example, 20. Now each is paid $(20+10)-(20+1)+20 = 29$.

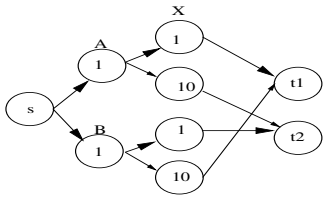
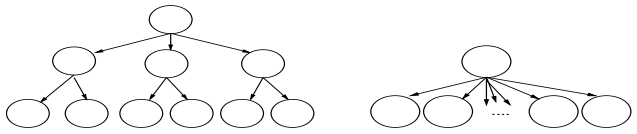


Figure 1: FPSS not Strategyproof when Repeated

To better understand this overcharging, we develop a formal model of the game and analyze its equilibrium. This requires us to examine several modeling questions involving equilibria notions and classes of protocols. The model then enables us to deterime the impact of the properties of the protocol on the resulting price. Several of the conclusions, summarized in Table 1 are quite surprising. For example, it matters if we use Mega*bits* or Mega*bytes* – and further that decreasing the period of the protocol may increase price.

Table 1: Impact of Parameters on Overcharging

As *[Variable]* Increases	Impact on Overcharging
N : Number of players	Decreases
b : Minimum bid size	Increases
d : Period of the protocol	Decreases
D : Stability period of topology	Increases



(a) Efficient Topology (b) Selfish Users

Figure 2: Sel sh Users Lead to a Different Topology

3. APPLICATION OVERLAY TREES

Application overlays have been proposed as a way of achieving the benefits of IP multicast. In practice, as discussed [2], the utility of a node decreases as it moves away from the source and also as it supports more children. Instead of topologies as in Fig. 2a, sufficiently greedy users will produce Fig. 2b.

In our research, we build a model of this interaction where users' greed is tempered by their desire for the network to continue to exist. With this model we solve for the equilibrium conditions and simulate the interactions on a BRITE-generated network topology. In our model, users' greed is tempered by their desire to ensure the network's continued existence, which in turn is a function of its efficiency. This allows us to determine the relationship of parameters such as the cost of adding children or δ, the patience level, on network efficiency and maximum size. In turn we are again able to make recommendations for system design.

4. REFERENCES

[1] J. Feigenbaum, C. Papadimitriou, R. Sami, and S. Shenker. A BGP-based mechanism for lowest-cost routing. In *Proceedings of the 2002 ACM Symposium on Principles of Distributed Computing*, 2002.

[2] L. Mathy, N. Blundell, V. Roca, and A. El-Sayed. Impact of simple cheating in application-level multicast. In *Proc. of IEEE Infocom*, 2004.

A Comparative Study of Mobility Prediction Schemes for GLS Location Service

Shshank Sharma, Venugopal Alatzeth, Gurpreet Grewal, Saurabh Pradhan, Ahmed Helmy

Department of Electrical Engineering
University of Southern California
Los Angeles, USA
{shshanks, alatzeth, ggrewal, sapradha, helmy}@usc.edu, http://nile.usc.edu/~helmy/GLSPrediction/grp1.html

Performance of geographic routing suffers from mobility-induced location errors. Location errors can also occur at location servers due to infrequent and/or lost updates, especially when the nodes are highly mobile. We performed an extensive study of the reasons for query failure in GLS [1] for different mobility models, namely the Random Way-Point (RWP), Reference Point Group Mobility (RPGM), Freeway (FW), and Manhattan (MH) using ns-2. We classified query failure reasons as in [2] namely, RLOOP, NRTE, NOSRVF and TTL. Specifically, we studied the impact of node velocity on query failure rates and reasons. In the case of RWP mobility model we observed that although only 8% of the total queries failed at 10m/s, the query failure rate increases to a high of 40% at 50m/s. Primary reason for this increase is mobility induced location errors such as RLOOP and TTL. In a more realistic mobility model such as Manhattan, this trend is more pronounced. 85% of the query failures are due to RLOOP at speeds higher than 20 m/s. The number of query failures also increased from a low of 3% at 10m/s to a high of 25% at 50m/s. We also observe that other query failure reasons such as inability to find a location server (NOSRVF) and voids (NRTE) do not show an increase with increasing mobility of nodes. In the case of freeway mobility model (in which geographic restriction is stricter compared to MH) the query failure rate is around 15-20% and the impact of failures due to location errors is less. Inability to find a location server was the major reason for query failures in FW. In RPGM model, we observed query failure rates of about 60%. At speeds of 30m/s and higher, TTL expiry is the major reason for query failures. We also observe that the communication pattern (inter group or intra group) has a major impact on the query failure rate since a query to find a target node in the same group is more likely to be successful than, that for a node in a different group.

We believe that mobility prediction by the location servers themselves can improve the query success rate in the Grid/GLS framework. Earlier work, such as Node Location Prediction (NLP) and Destination Location Prediction (DLP) [3], advocates location prediction by the forwarding node, and is more useful when routing packets hop-by-hop. Problems may arise if the location information maintained in the location server itself is in error, due to low location update frequency or high node mobility. We address this issue at the query point (location server) even before a destination location is given to start the routing.

Schemes proposed to predict node location can be broadly classified into ones that use movement history and frequency of visits, movement patterns, or constant velocity/direction. We implemented three prediction schemes in GLS: linear velocity based (LVP), weighted average based (WVP) and O (1) Markov recency-based (OMP) [4]. LVP uses the two most recent locations of a node to calculate the speed of the node and predict the current location. In WVP, we take running weighted average of node velocity. The history-based scheme (OMP) makes use of patterns in node movement to predict future locations. The evaluation metrics are control packet overhead, storage requirement, query success rate and prediction accuracy. These prediction schemes involve no communication overhead and require minimal increase in storage requirements (12 bytes per node) at location servers. In ns simulation runs, LVP scheme predicted a more accurate location than original GLS 58-70% of the prediction attempts. We observed that the effectiveness and accuracy of a particular prediction scheme in turn depends on the node mobility model. Geographic restrictions (such as in Manhattan and Freeway models) and node speed also impact the prediction accuracy.

We expect greater improvements in prediction accuracy by incorporating mechanisms that take into account changes in node speed and direction of movement. Validating predicted locations using maps (geographic restriction knowledge) is expected to further improve the accuracy of these prediction schemes, especially in mobility models such as Manhattan and Freeway models. We are currently investigating the performance of such schemes.

REFERENCES

[1] J. Li, J. Jannotti, D. Couto, D. Karger, R. Morris, A Scalable Location Service for Geographic Ad Hoc Routing (GLS/Grid), ACM Mobicom 2000

[2] Käsemann M, Hartenstein H, Füßler H, Mauve M, A Simulation Study of a Location Service for Position-Based Routing in Mobile Ad Hoc Networks. Technical Report, Reihe Informatik, Universität Mannheim, June 2002

[3] D. Son, A. Helmy, B. Krishnamachari, "The Effect of Mobility-induced Location Errors on Geographic Routing in Ad Hoc Networks: Analysis and Improvement using Mobility Prediction", IEEE Wireless Communications and Networking Conference (WCNC), March 2004.

[4] Libo Song, David Kotz, Ravi Jain, Xiaoning He. Evaluating location predictors with extensive Wi-Fi mobility data. Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), March 2004.

A. Helmy was funded by NSF CAREER, Intel and Pratt&Whitney

Nontransitive Dice Paradox in Networking *

[Poster Abstract]

Chi-kin Chau
Computer Laboratory, University of Cambridge
Chi-kin.Chau@cl.cam.ac.uk

Here is a simple trick that you can use to try to dupe a friend out of his money with high probability. Bet with him on the outcome of three specially numbered dice labelled X, Y, and Z each with six faces marked as: X (3 3 5 5 7 7), Y (2 2 4 4 9 9), Z (1 1 6 6 8 8). Each person chooses a different die. The larger number wins when the dice are rolled. Strangely, whichever die your friend has chosen, you can always outwit him with another die of greater winning probability; in fact $\mathbb{P}(X > Y) > \mathbb{P}(Y > X)$, $\mathbb{P}(Y > Z) > \mathbb{P}(Z > Y)$, and $\mathbb{P}(Z > X) > \mathbb{P}(X > Z)$. Therefore, one can legitimately arrange the order of the dice as $Z \succ X \succ Y \succ Z$, creating a cyclic preference.

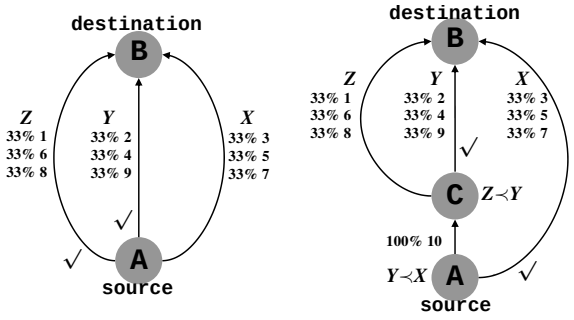
This is an amusing paradox that also arises in decision problems and elections, where the relation induced by pair-wise comparisons on random variables, unlike on deterministic values, may violate transitivity. Indeed, there is an implication in networking, and this poster shows that the behaviour of distributed decision making in the presence of stochastic metrics gives rise to situations resembling the nontransitive dice game with counter-intuitive outcomes that may trap unwary system designers and network administrators.

Networks are plagued with uncertain metrics. Internet traffic is inherently stochastic due to both fluctuating demands and varying payload sizes, leading to stochastic metrics such as available bandwidth, queuing delay, throughput, and jitter. Link states are not always static, for instance, due to changes of network topology caused by system failures and misconfigurations. Furthermore, connections over novel network infrastructures, such as overlay networks and wireless ad hoc networks, give rise to more varying and dynamic connectivity. Often, routers will be supplied with incomplete or out-dated knowledge of the network, owing to delayed, infrequent or batched link state advertisements. In addition, aggregation of link states in hierarchical networks, in order to help scalability, but presents imprecise information to identify the available resources.

It is feasible to acquire the information about path characteristics such as available bandwidth locally by using a variety of end-to-end measurement tools, based on active probing or passive inference. For example, in overlay networks such as RON or Planetlab, information about the available bandwidth of the paths is useful for making local routing decision. Given that the metrics for deciding the best path are stochastic, distributed decision making is not as simple as in the deterministic case.

Consider an overlay network, with three paths X,Y,Z with available bandwidth distributed as the nontransitive dice. Drawing on the nontransitive dice game, one can see that different arrangements of comparisons will yield different paths of maximum bandwidth. If all three paths are put together as in the left diagram below, then Y and Z are the optimal paths with the most likely maximum bandwidth since $\mathbb{P}(\max\{X,Y,Z\} = Y) = \mathbb{P}(\max\{X,Y,Z\} = Z) > \mathbb{P}(\max\{X,Y,Z\} = X)$.

However, with a different topology as in the right diagram below, contrary to A's intent, it has to choose the *least* likely maximum bandwidth path X. It is because the intermediate node C selects path Y as it is stochastically larger than Z (i.e. $\mathbb{P}(Y > Z) > \mathbb{P}(Z > Y)$). Hence, A is forced to choose path X because $\mathbb{P}(X > Y) > \mathbb{P}(Y > X)$. Here arises the paradox.



Motivated by the above paradox, we explore the consequences of network algorithms in the presence of stochastic metrics. We identify several more paradoxical cases, including a Braess-like paradox where the addition of link capacity by a well-meaning network administrator causes everyone to be worse-off. Braess' paradox is due to selfish users collectively optimising a different objective from the social journey time, so the addition of a link can possibly lead to worse social journey time. Our paradox, however, is due to the nontransivity of random variable comparisons where a better link in the global sense may not be the better one in the local sense.

In the full paper, we present other possible consequences of stochastic metrics such as sub-optimality in route selection and minimum spanning tree construction, divergence in routing policy, and oscillations in resource reservation, all of which are (disguised) forms of the cyclic preferences arising in the nontransitive dice game.

*A full paper is available at http://www.cl.cam.ac.uk/∼ckc25/dice

Towards a Composable Transport Protocol: TCP without Congestion Control

Roland Kempter
Department Electrical and Computer Engineering
University of Utah
kempter@eng.utah.edu

Bin Xin
School of Computing
University of Utah
xinb@cs.utah.edu

Sneha Kumar Kasera
School of Computing
University of Utah
kasera@cs.utah.edu

1. INTRODUCTION

The reliable transport layer protocol of the Internet, TCP, provides a variety of features including error control, congestion control, flow control, and in-order delivery. Given the diversity of applications and the underlying network, some of these features are not required for certain applications and networks. In fact, their presence could result in reduced performance. We believe that these features should be offered as composable units rather than all tied together. Composability is also likely to help in adding new features or for upgrading. The eventual goal of our research is to design a fully composable TCP. In this poster, we focus on separating out the congestion control feature such that it could be added and removed without affecting the other features.

We first ask ourselves, where could we use a TCP without congestion control. TCP without congestion control could be useful in scenarios where packet losses are due to reasons other than network congestion, and where fair access to the network resources is achieved through other means. Wireless networks are good examples. Wireless networks have two interesting characteristics. First, they experience a fair amount of loss due to random channel errors[1]. Second, the fairness in accessing the wireless link is handled at the link layer. In these scenarios, TCP congestion control could become a burden.

2. EXPERIMENTATION

While the majority of efforts towards a composable protocol have been using specialized frameworks or APIs (e.g., [Kohler 99]), we investigate the feasibility of constructing a composable TCP based on the widely deployed, highly optimized TCP New Reno in FreeBSD 4.7. TCP New Reno in FreeBSD 4.7 has around 10K lines of code (LOC) with comments. The congestion control code is spread across several files and intermingled with the code implementing other features. We separate out the congestion control code using C preprocessors. In our modified version of TCP New Reno, congestion control amounts to about 250 LOC. We also investigate the use of the TCP Selective Acknowledgment (SACK) feature, that allows a TCP receiver to acknowledge out of order segments selectively rather than cumulatively acknowledging the last correctly received in-order segment (in the case of earlier version of TCP including TCP New Reno). TCP SACK amounts to about 1K LOC and is already available as a composable feature. We compose four different flavors of TCP: TCP New Reno without SACK (BASE), TCP New Reno with SACK (SACKBASE), our modified experimental TCP with neither congestion control nor SACK (EXP), and TCP with SACK but without congestion control (SACKEXP). Without congestion control, TCP send rate is determined by the receiver window and the bandwidth-delay product.

Instead of running synthetic simulations, we choose to follow the more realistic **emulation** approach to measure the performance of these four variants of TCP. We set up the test network, comprising four hosts, two routers and a single shared link, in Emulab [Emu]. The FreeBSD kernels with these variants of TCP are compiled and data for each variant is collected using `ttcp` and `tcpdump`. The data are then analyzed off-line using Ethereal [Eth]. We evaluate the four TCP variants by measuring goodput and efficiency, which is defined as the ratio of goodput to throughput.

Our results show that by simply excluding the congestion control feature from TCP New Reno, EXP improves goodput significantly over BASE up to 280%, but drastically reduces link efficiency down to 13%. This is because without the congestion control feature, the sender window is bigger resulting in higher goodput but at the same time the *Go-back-N* nature of the protocol wastes bandwidth. However, with SACK, we find that the **goodput** of SACKEXP over SACKBASE lies anywhere from **184%** (link loss rate p=0.01) to **730%** (link loss rate p=0.2), while **efficiencies** of SACKEXP over SACKBASE are between 60% and 95%. This interesting result shows that removing congestion control improves goodput but adding SACK also improves efficiency. In fact, we also observe that the goodput improvement of SACKEXP over SACKBASE can be much higher than that of EXP over BASE. We suggest using SACKEXP in the wireless scenario described above, although we need to examine this more.

In future, we will investigate the composability of other TCP features. Our initial efforts indicate that separating the error control feature is likely to be much harder (also corroborated by [Kohler 02]).

A full version of this poster is available at `http://www.cs.utah.edu/~xinb/poster-final.pdf`.

[1]It is possible to locally repair link errors at the link layer and there are several existing studies examining this issue. Here, we only point out that TCP is still necessary for end-to-end reliability.