Public Reviews of Papers Appearing at HotNets-III

Forward

Editors: Thomas Anderson and Nick McKeown

Welcome to HotNets-III! We would like to take this opportunity to thank the authors who submitted papers to the workshop, the program committee, the external reviewers, and especially the general chair, Alex Snoeren. We would not have been able to put on the workshop without their efforts. For those who keep track, we received 127 papers and accepted 24 of them.

As always, choosing papers for HotNets is a difficult task. Most of the submitted papers contained interesting ideas and approaches, and are worthy of further research. Our selection criteria included novelty, the ability of the paper to spark discussion at the workshop, and the likelihood that future research would benefit from early publication. For those papers not included, we look forward to seeing those ideas fleshed out in papers at future conferences.

As an experiment this year, we solicited a leading researcher to write a signed "public review" of each accepted paper. Our intent is to foster a dialogue between authors and readers. Too often, the context for understanding the relevance of a paper emerges only during the Q&A or hallway conversations at the workshop, and then disappears into the ether. The result is that each new reader must start afresh in deconstructing the paper.

We encouraged the public reviewers to be opinionated rather than factual, and to consider their canvas the broad set of topics raised by each paper. The opinions expressed in the public reviews are those of the reviewers alone, and intentionally do not reflect the views of the program committee or the chairs. The reviewers were selected independently of the program selection process and thus sometimes came to conclusions quite different from the program committee. The intent is to provide a singular point of view on the work, not the general consensus. A different reviewer might well come to a completely different set of conclusions about any given paper. Thus, the reader is cautioned that the public reviews are not the only or the best way to view the papers; they are intended to assist the reader, not to be the last word on the paper's eventual impact. Because of time constraints, the authors were not in general given an opportunity to respond to the public reviews.

The reviews are provided below, in the order the papers appear in the proceedings. Enjoy!

Overcoming the Internet Impasse through Virtualization

Larry Peterson, Scott Shenker, and Jon Turner

Public Reviewer: Nick McKeown

This paper is heretical and dangerous. In days gone by – as little as five years ago — these researchers would have been burned at the stake. They propose a Trojan horse (in the form of a multiplexed overlay network) in which new protocols, services and even complete alternatives to IP can be experimented with. But mark my words, this paper is not just about science experiments. It is a thinly veiled plan to allow the full-scale deployment of radical departures; whole new approaches that are not even backwardly compatible with the Internet of today. The authors question the very perfection of the Internet, even accusing it of being defaced by barnacles and unsightly outcroppings. They propose a scheme in which thousands of students can develop and deploy a myriad of different protocols, algorithms and services. It even threatens the authority of the IETF. In short, it will unleash a whole new era of chaos.

And that is exactly why you should read this paper. We need something like this to foster innovation in the Internet.

We all know that the Internet has become a victim of its own success. A research prototype and file transfer mechanism for physicists became a social necessity, and now it's difficult to make changes. Yet changes are needed; we should not be surprised that a network designed thirty years ago isn't perfect. If we had the chance to start again, we'd do things a little differently based on all the things learned along the way. Hindsight is a fine thing, and the Internet is a fantastic experiment to learn from.

Frustration about the difficulty of change in the Internet goes back several years. In 2001 the National Research Council report "Looking Over the Fence at Networks: A Neighbor's View of Networking Research" pointed out that:

> Success [...] has been a double-edged sword, for with it has come the danger of ossification, or inability to change, in multiple dimensions:
>
> * Intellectual ossification – The pressure for compatibility with the current Internet risks stifling innovative intellectual thinking. [...]
>
> * Infrastructure ossification – The ability of researchers to affect what is deployed in the core

infrastructure (which is operated mainly by businesses) is extremely limited. [...]

* System ossification – Limitations in the current architecture have led to shoe-horn solutions that increase the fragility of the system.

But even before 2001, researchers were looking for ways to allow innovation and experimentation in the Internet. This was a major goal of Active Networking, in which routers could be programmed to offer new services. To some, Active Networking provided a platform for flexibility and innovation. To others it was hair-brained, making routers too complex, less secure and lower performance; and it took us away from the end-to-end principle. As Active Networking withered, overlays sprouted, and now are part of the landscape, with Akamai possibly the first to establish a world-wide, permanent overlay network that ran over the existing Internet architecture.

The MBONE was one of the first research overlay networks, designed specifically to test the deployment of multicast. In 1998, Detour used measurements to demonstrate the benefits of routing over an overlay. The idea was not taken up, and the benefits are probably still there to be had six years later. Real change in the networking community came with the introduction of PlanetLab, with the help of Larry Peterson et al. and Intel Labs. PlanetLab provides an enormous world-wide sandbox for the testing of new ideas.

On the face of it, the paper by Peterson, Shenker and Turner is another paper about overlays. It doesn't make many technical innovations of its own. It slightly extends the already-successful notion of PlanetLab, adding proxies that permit the virtual testbed to support non-IP traffic.

But the value of the paper is not its technical cleverness; it's the proposal to build a large, scalable, realistic, geographically dispersed testbed that can be used for rapid prototyping of novel networking ideas.

Overlays are yet to fulfill their original promise. A series of one-offs means there is no common architecture to ease the incorporation of new ideas. And before PlanetLab came along, there was no widespread deployment of overlays. With the addition of proxies and a mechanism for endpoints to selectively subscribe to an overlay, the proposed virtual testbed might be the first time that overlays reach their potential. I, for one, would like to see it happen, and believe it is a more cost-effective approach than investing in costly physical testbed networks.

To the extent that a review should give you some guidance as to whether or not you should read the paper, my job is simple: This paper represents the very essence of what Hotnets papers should be about. It was the highest ranked paper by the Program Committee. Enough said.

Can ISPs take the heat from Overlay Networks?

Ram Keralapura, Nina Taft, Chen-Nee Chuah, and Gianluca Iannaccone

Public Reviewer: David Andersen

"Can ISPs Take the Heat from Overlay Networks" poses a timely and important question: Can the uncoordinated interactions between overlay networks and the underlying IP routing substrate result in poor performance for the entire network? The research in the paper is at a very early stage and leaves open numerous issues about the behavior of real-world overlays, but the questions it raises are serious, and deserving of further study.

Overlay networks have become increasingly popular in both research (e.g., DHTs, RON, Application-layer multicast, etc.) and in practice (Akamai, Kazaa, and so on). As I wrote this, a Reuters article came out pointing out that traffic from the BitTorrent peer-to-peer file sharing program now accounted for 35% of Internet traffic. At the same time, ISPs have developed more powerful tools for traffic engineering that let them balance the traffic load within their network. Overlay networks make the traffic mix dynamic and could "render it more difficult for ISPs to achieve their goals" by having "routing control in two layers, when each layer is unaware of things happening in other layers."

The authors study these questions using a simple simulator of a generic RON-like overlay that probes the paths between nodes every few seconds and routes traffic along the lowest-latency path. They present four potential problems:

1. Traffic matrix estimation
2. Load balancing
3. Oscillatory race conditions between overlays
4. Propagating intra-domain problems to other domains

These problems fall into two categories. The first two problems possibly pose policy problems for the ISPs; the second two address fundamental stability issues about the interaction between overlays, other overlays, and routing protocols.

I believe that load-balancing and assigning priority to traffic is primarily a policy issue within the ISP. When an organization hosts an overlay network node, the traffic that emerges from it *is* traffic from the organization, by definition. If unwanted traffic is being relayed through an organization, the organization probably has pressing security problems that obviate a small traffic engineering concern.

More seriously, the paper next points out that the independent routing performed by IGPs and by overlays is "a classic situation for race conditions that lead to traffic oscillations." In their simulations, overlays can oscillate because of interactions with other overlays when both converge to the same link, or when a single overlay changes traffic drastically enough to congest a single link. While the simulations present scenarios that could oscillate, there are a number of potentially mitigating factors that future studies

should include: First, most overlays use randomized probes and communicate between diverse subsets of nodes. Second, real networks have additional sources of randomization and synchronization that could make a large difference in the likelihood of oscillations. Finally, overlays that deal with high-bandwidth flows would need to use multi-path transfers, both to take advantage of network capacity and help reduce the magnitude of traffic swings.

Finally, the authors present the interesting observation that by responding to routing changes in a remote autonomous system, overlays can propagate traffic changes between ASes, something BGP expressly attempts to avoid. This raises an interesting question about the same effect from traffic engineering by ISPs.

Like a good HotNets paper should, "Can ISPs Take the Heat" ultimately leaves more questions trailing in its wake than it answers.

Supporting Spectators in Online Multiplayer Games

Ashwin R. Bharambe, Venkata Padmanabhan, and Srinivasan Seshan

Public Reviewer: Vivek Pai

If you are reading this paper, chances are good that (a) you played video games at one point, and (b) that you are no match for today's gamers on today's games. Take consolation in the fact that it has no bearing on your abilities as a computer scientist, but at the same time, realize it is not a good idea to challenge undergrads to a Quake deathmatch. Computer games are a big business – in 2002, games had total sales of $30 billion, while movies had roughly $20 billion. Electronic Arts has a larger market capitalization than Veritas and Siebel combined. So how can the game industry reach the undextrous masses and scale participation? The same way that the NFL and NBA do – letting people watch, cheer, and become observers of an event that feature players with much better skills than the viewers have.

This paper presents a scheme for allowing spectating and cheering in online multiplayer games. As the names imply, spectating is watching the action, possibly from multiple vantage points, while cheering is a means of providing audio feedback to the participants. The paper spends most of its time on spectating, which is understandable, given that it appears to have more interesting questions and opportunities than cheering.

The approach taken is interesting – use the same kind of infrastructure that you would for scalable streaming, namely End System Multicast, and use it to transmit information about the environment. If the viewers have the same playing engine as the participants, they can composite the images themselves, and so require much less bandwidth than receiving the full video streams. The approach to bandwidth adaptation is to weight how important the various elements in a scene are, and use that to determine what needs to be sent. Using Quake III data, the authors also show that delta encoding versus recent frames is a big win. The difficulty when compared to live streaming is that gaps in a video stream may be insignificant after a short period, but if the stream contains the appearance or disappearance of objects in the simulated world, then gaps would cause correctness problems. The authors solve this problem by forcing the messages that require reliable delivery to be marked appropriately. Between these problems and the other bandwidth adaptation strategies mentioned, the authors clearly demonstrate that this area has enough interesting problems to keep computer scientists busy.

So if we believe that this kind of information can be scalably distributed, and that the modifications necessary are within the realm of what games companies are willing to do, the ultimate question is whether this will matter. If sports viewing is any indicator, many people will gladly watch other people playing a game, whether in person or remotely. The purses for some competitions are in the range of $50k-$100k, which is significant, but still shy of professional sports. In comparison, LPGA tournaments tend to have purses over $1 million, and PGA purses are a few times larger. Time will ultimately tell, but if the purses keep growing and sales keep pace, spectatorship may become significant.

The biggest impediment, and perhaps the most obvious practical difficulty of this line of work, is how to address the issue of cheating. The authors briefly touch on this issue, while admitting it is not the focus of their paper, and state that delaying the spectating stream can alleviate problems related to cheating. While this claim may be true for games where only the recent past matters, the authors specifically mention that one of their target applications is real-time simulations, such as Age of Empires. In these games, strategies may take time to develop, and concealment from opponents is necessary. If large-scale viewing occurs, then presumably some trusted third-party can be employed to ensure fairness, but for small-scale informal play, no obvious route exists for ensuring that the spectators can not help the players. Given the amount of effort put into cheating tools, such as "aimbots" and the amount of effort companies have to put into making their network traffic hack-resistant, the emergence of spectator-based cheating is high.

This problem may also explain the relatively smaller amount of attention paid to cheering. If such cheering is intended to boost the morale of the players, it suffers from the time-delay necessary to ensure fairness. If it exists solely as a bonding mechanism for the spectators, then the delays are immaterial. The authors seem to pay attention to the former scenario, although it seems that the latter scenario is more likely given the need for delay. Also missing from the paper is the jeer, or taunt, which will undoubtedly use as much bandwidth as the cheer.

Some Foundational Problems in Interdomain Routing

Nick Feamster, Hari Balakrishnan, and Jennifer Rexford

Public Reviewer: David Wetherall

BGP has become a burning issue for networking researchers – there are three papers on BGP in this workshop alone! – and with good reason. It is the protocol used to select routes across and within the ISPs that make up the Internet. Nearly all that runs on top of the Internet depends on those routes and how well they work. And the evidence to date, much of which is catalogued in the paper by Feamster, Balakrishnan and Rexford, is that of a growing collection of problems with the stability, performance and security of BGP.

Starting roughly in the mid to late 1990s, measurement studies of routing behavior within the Internet began to show to moderate levels of instability (e.g., studies by Paxson, and by Labovitz). The finger pointed at BGP. Since that time, much more data on routing has become available thanks to resources such as the RouteViews project at the University of Oregon, ISP studies of their own networks, and improved inference techniques that deduce internal behavior from external observations. Measurements and knowledge of operational practices have opened a window on the black-box of Internet routing. And the result has been striking: a stream of papers on observed or potential pathologies and difficulties that have been found under almost every rock that has been turned over.

This is the context that surrounds Feamster's paper. The tide has now turned from identifying shortcomings in BGP towards finding solutions that improve Internet routing. Feamster's paper categorizes the known problems (and some suggested point fixes) according to whether they stem from policy or scalability considerations. As the authors point out, if we assume that we understand intra-domain routing (e.g., within an ISP using OSPF or ISIS), then the imposition of policy by separate parties and the hiding of information for scalability are the two fundamental factors that must be incorporated to enable inter-domain routing. It is these factors that complicate BGP and which have lead to a series of problems. The paper then poses questions that, when answered, will help to determine whether the problems are specific to BGP-4, general to path vector-based protocols such as BGP, or inherent in any scalable and policy-based interdomain routing protocol.

One possible criticism of this paper is that there is nothing new in it. The problems it describes and their solutions are all previously known in the literature and the questions are, well, questions. But I think this criticism misses the mark. There are enough known problems; the difficult issues concern how to fix these problems. And that is a substantial task, sufficiently large as to make it highly unlikely that a single paper can put forward a good solution to most of these problems along with an effective plan for its deployment. That being the case, the surest route to progress is for the community to cleave the overall problem into solutions for some of the sub-problems, and perhaps deployment strategies for solutions. But which sub- problems belong together? And how will they be solved, by tweaking BGP or defining new protocols? As yet I am not aware of community agreement on these questions. This is precisely the problem to which the authors direct our attention. The paper highlights our lack of understanding of the foundational aspects of existing problems, i.e., of whether they are intrinsic to BGP. It then argues that the razor of whether the problem is rooted in BGP-4 or scalable policy-based routing be used to group problems, and by implication their solutions.

Feamster et al. proceed by collecting known problems in one place and classifying them according to their cause. The descriptions of the problems are given in a manner that makes this paper a good introduction for researchers trying to grasp the overall area. The problems themselves range across oscillations caused by policy, the difficulty of remotely validating routing and verifying forwarding, and a series of defects caused by information hiding for scalability. However, the authors make no claim for completeness of BGP problems that are known to date. This is difficult to judge, as the authors do not explicitly state the goals of BGP, e.g., whether it is solely about reachability or includes performance considerations. From my own work, the main areas I would consider omissions are traffic engineering and providing any kind of QOS in the current system. Traffic engineering aims to make effective use of network resources. It is complicated with BGP today because ISPs have limited control over their incoming traffic (it tends to be directly controlled by their upstreams) and ISP choices may have follow-on effects that make the ultimate effects less predictable (the downstream may react, leading to instability in the worst case). The lack of predictable control precludes QOS guarantees at the protocol level; cross-ISP arrangements must be made by operators. A more minor omission is that, because of information hiding, BGP route selection is based on AS paths, which can lead to circuitous or poorly performing paths. These omissions are a little puzzling, as a more complete classification would make a better basis for understanding how to improve BGP.

For each set of problems, Feamster et al. then pose questions intended to tease apart whether the root cause of the problem is intrinsic to BGP-4 or an inherent design tradeoff in scalable policy routing. These questions are many and varied, and they are all intellectually interesting. What is less clear about them (besides their answer!) is what they mean for solving the BGP problem. A straightforward answer would be that if the root of a problem lies with BGP-4, then fix BGP-4, but if it is inherent in scalable policy routing, then fix it in a more general fashion. But what is this more general fashion? The paper acknowledges that this may be either restrictions on how BGP is used or a new protocol.

Thus in my view the paper is agnostic on whether to extend or constrain BGP as it exists, or to replace it with a new protocol. This was perhaps inevitable, but it would have been interesting to hear the authors' opinions on the alternatives, or at least the implication of different outcomes for each question.

More pragmatically, it may be difficult to conclusively answer the "impossibility result" style questions that the authors pose for a "foundational understanding" of BGP's problems. But do we need to answer these questions to make progress? We might instead ask why BGP has proved so problematic and tackle those factors directly. That is, we might turn to new, simplified protocols that try to solve some of BGP's ills by defining away the complex interplay of factors that gives rise to the problems in the first place. For example, it may be difficult to either demonstrate correct routing using redundant route-reflector schemes or prove it impossible to do so, as considered in 4.2.1. Instead, perhaps we can flood information between the routers of an ISP. This is a less scalable solution than is used today (with a greater cost in CPU and bandwidth) but it has the advantage of removing the information-hiding problems that stem from route reflectors. This road may lack a foundational understanding, but where it can be made viable it would seem an easier road to a solution.

The above points notwithstanding, the goal of a foundational understanding of BGP's problems is undeniably valuable. It will generate discussion as to which problems can be solved by tweaking BGP, and which are design tradeoffs that require more fundamental changes to interdomain routing. This dialog seems an essential part of working towards a solution – it is a dialog that someone needed to start.

Towards a Next Generation Inter-domain Routing Protocol

Lakshminarayanan Subramanian, Matthew Caesar, Cheng Tien Ee, Mark Handley, Z. Morley Mao, Scott Shenker, and Ion Stoica

Public Reviewer: Jennifer Rexford

The Internet's interdomain routing system is fraught with problems, ranging from persistent oscillation and slow convergence to security vulnerabilities and poor fault isolation. During the past few years, the Border Gateway Protocol (BGP) has become a popular subject of measurement, analysis, and (ultimately) sharp criticism. Yet, it is important to acknowledge that BGP is trying to solve an exceptionally difficult problem — providing end-to-end connectivity between the tens of thousands of separately-administered networks that comprise today's commercial Internet. Despite the substantial documentation of BGP's numerous failings, surprisingly few research studies have ventured to propose alternative designs that would solve the same challenging problem more effectively. This paper is a notable exception.

There are two complementary ways to read this paper — as a thought experiment and as a concrete proposal for replacing BGP. As a thought experiment, the paper's approach of optimizing the design of the routing system for the "common case" of how BGP is configured today is extremely appealing. For example, BGP allows each Autonomous System (AS) to configure complex local policies for selecting routes and advertising these routes to neighboring domains. However, in practice, two neighboring ASes typically have a customer-provider or peer-peer relationship, and these business relationships largely dictate the configuration of the BGP routing policies. In fact, the vast majority of AS pairs have a customer-provider relationship, leading to AS hierarchies where each AS prefers to direct traffic through a downstream customer rather than a peer or provider. Exploiting this observation, the paper proposes a hybrid design with link- state routing within a single hierarchy and path-vector routing across different hierarchies. (As an aside, the proposed hybrid design has some key similarities to the use of areas in OSPF and to the design of the EGP protocol that predates BGP, which would have been interesting to discuss in the paper.)

As protocol designers, we tend to shy away from designs that optimize for today's realities, lest they lead us to solutions that become increasingly brittle or inefficient for accommodating tomorrow's new realities. Yet, as researchers we desperately need to understand the fundamental cost of general solutions. In optimizing for the common case, the proposed architecture should enable a deeper understanding of the price we pay today (in convergence delay, configuration complexity, security vulnerabilities, etc.) for having a routing system with a design that exercises so little restraint. (That said, I think it would be extremely useful for the research community to establish a "negative result" — that the kind of generality that BGP provides is truly untenable — to solidify the argument that optimizing for some restricted or "common" cases is, in fact, necessary.) This is the main reason that I like the paper a great deal as a thought experiment (and a HotNets paper). In fact, I hope that networking researchers will conduct similar thought experiments for other Internet protocols.

My reluctance to view the new architecture as a concrete proposal for replacing BGP stems mainly from concern that the Internet's business relationships will continue to evolve. Already, mergers and acquisitions have led to many "sibling" relationships, where two neighboring ASes belong to the same institution and allow transit traffic. Many large Internet Service Providers (ISPs) have dozens of ASes spread throughout the world; even large enterprise networks often consist of multiple ASes to improve scalability or enable decentralized network management. In addition, concerns about reliability have led to "backup" relationships that sometimes violate the assumptions about the routing policies in customer-provider and peer-peer relationships. Also, some AS pairs do not have a single consistent business relation-

ship — rather, two ASes may have a peer-peer relationship in one part of the world and a customer-provider relationship in another. Even though these scenarios do not break the proposed architecture, they do lead to "exceptions" where the system gradually devolves to the properties of the existing BGP protocol.

The growing complexity of business relationships, and the likelihood of continued change in the coming years, makes it risky to have a routing system that optimizes for today's realities. Still, the proposal in the paper is intellectually stimulating and thought- provoking, and indirectly raises a very important question about where the economic factors belong in the routing architecture. Today, business relationships show up in the *configuration* of the routing protocol, but not in the design itself. Should these business relationships move deeper into the design and operation of the routing protocol, as proposed in this paper, or elsewhere into pricing and billing arrangements? For example, today an ISP configures its BGP policies to disallow transit traffic from one peer AS to another. Yet, if the ISP could announce (via the routing protocol, or perhaps some other mechanism) a *price* for sending transit traffic, there would be no need to disallow such traffic.

In addition to optimizing for the common case of customer-provider relationships, the proposed architecture exploits the fact that, although ASes often advertise multiple destination prefixes, these prefixes typically have the same AS path in BGP. This argues for advertising and selecting paths at the AS level, rather than the prefix level. The paper makes a nice point that the prefix-to-AS mapping changes very slowly and could be disseminated more efficiently (and securely) using other mechanisms. Despite the conceptual appeal, routing at the AS level raises a number of challenging issues. First, although measurement studies show significant commonality among paths for different prefixes, the paths are not necessarily the same from *every* vantage point in the Internet. Second, different paths may be necessary in some cases for flexible load balancing and control over backup paths. That said, the current practices for load balancing (e.g., by selective or customized advertisement of different subnets) are crude and ad hoc at best, so they are certainly ripe for redesigning. It would be great to find a way for interdomain routing to operate at the AS level while still satisfying the traffic engineering goals that drive prefix-level routing today.

Towards Coordinated Interdomain Traffic Engineering

Ratul Mahajan, David Wetherall, and Thomas Anderson

Public Reviewer: Jennifer Rexford

Interdomain traffic engineering is a black art where each Autonomous System (AS) adapts the BGP policies on its routers based on a local view of traffic, topology, and routing, with only a limited understanding of the effects on neighboring domains. As such, interdomain traffic engineering in today's Internet is inefficient and error-prone. This paper argues for greater coordination between neighboring ASes to make mutually beneficial routing decisions, and proposes building blocks for a solution. The main contribution is to formulate an interesting, important, and challenging problem, and to highlight the significant performance gains that could be achieved by a good solution.

The paper does a very nice job describing why interdomain traffic engineering is difficult, and why ad hoc techniques are unfortunately the state-of-the-art today. The difficulties arise from the lack of information to make sound routing decisions, and the limited control over which paths are used (especially for inbound traffic). The paper argues that two-way information exchange, route negotiation, and flow registration provide a way around these problems. At a high level, these seem like the natural building blocks for a solution. In presenting these building blocks, the paper arguably raises more questions than it answers (as a good HotNets paper should!), including:

* The paper argues that the coordination should lead to a stable outcome where no AS can improve its situation at a cost to others. Yet, the paper does not argue whether such a coordination scheme exists. Knowing if a negative result is hiding here is extremely important. (That said, even if no such scheme exists, other mechanisms (such as money) may exist to create an incentive for an AS to accept a less-favorable outcome to improve overall network efficiency.)

* A closely-related question is whether an AS would have an incentive to mis-represent its wishes to "game" the system. For example, an AS might advertise an incorrect utility to encourage the other AS to accept a different outcome. Even if the resulting outcome is less efficient for both ASes, one AS may suffer more than the other. If the two ASes are commercial competitors, this may be an acceptable (or even desirable) outcome. Finding a way to prevent this kind of gaming is crucial to making the system workable, and getting ASes to participate.

* The route negotiation process requires an AS to assign a numeric utility to each peering point and flow pair. Yet, the paper does not say how an AS would compute these utilities—and, particularly, how to assign utilities across the many sets of peering points and flows. For example, if an AS is trying to balance load on its links (e.g., to minimize a simple metric like maximum link utilization), what is the algorithm for setting the utilities?

* An AS may need to sacrifice on one flow to gain on another, or compromise in the present for a benefit in the future, as the paper nicely observes. Yet, the paper does not explain how two ASes would coordinate to reach these kinds of "deals." Even if a "win win" solution exists, how would the negotiation scheme manage to find it? More generally, what is the notion of "value" of an outcome to an AS? Hav-

ing a solution where each AS can have its own notion of "value" is appealing, and yet it makes it hard to understand how the cost and value of these compromises between two ASes over time would be tallied.

* In some cases, an AS may have routes to a destination through multiple neighboring ASes. For example, an Internet Service Provider (ISP) may have a peering relationships with two ISPs that share a common customer. In this situation, the AS has two sets of choices for peering points to send the traffic to this destination (or to receive traffic from this sender). How does the AS combine the information across the negotiation processes with the two ASes? The paper rightly focuses first on bi-lateral negotiations but it would be interesting to consider whether the right solution is two bi- lateral negotiations (with the AS in question making a local judgment of which offer to accept) or some sort of multi-lateral negotiation.

* Although flow registration is appealing, striking the right level of flow state and signaling is crucial to avoid the many problems that plagued earlier work on quality-of-service (QoS) routing. For example, the paper mentions that the flow state includes the estimated amount of traffic but it is not clear where this information would come from. Also, the paper is somewhat vague on how negotiations for a flow would propagate from one AS to the next as the path changes. Comparing and contrasting the flow registration proposal with QoS routing/signaling would be interesting, as the devil is in the details here to identify a solution that is effective without being cumbersome.

* The proposal would require some changes to today's routers and protocols, and perhaps some additional protocols on top of the existing routing system. Yet, the paper is not all that clear about what changes would be necessary. Exploring approaches that require changes (even fundamental changes) to the infrastructure is totally reasonable but it is helpful to understand what these changes might be. This would be useful for judging whether the gain in efficiency (e.g., as estimated in Section 5) from inter-AS negotiation offsets the potential cost in increased system complexity.

* The evaluation in Section 5 focuses on a semi-empirical approach, using inferred AS topologies, synthetic traffic demands, and a metric based on the degree of over-provisioning. Yet, it could have been interesting to show some toy examples to illustrate how inefficient today's uncoordinated traffic engineering can be, or to appeal to a more theoretical analysis for worst-case results (e.g., as in the paper "Routing and Peering in a Competitive Internet" by Ramesh Johari and John Tsitsiklis, MIT LIDS publication 2570).

Overall, the paper does a great job at formulating the problem of coordinated interdomain traffic engineering, and making the fundamental challenges accessible to a broad audience. It is rare to find a research problem with such rich intellectual scope and compelling practical importance. I believe that the paper will spawn other research work that attacks various aspects of the problem (including the questions listed above), both in terms of theoretical analysis and protocol design.

Network-Wide Decision Making: Toward A Wafer-Thin Control Plane

Jennifer Rexford, Albert Greenberg, Gisli Hjalmtysson, David A. Maltz, Andy Myers, Geoffrey Xie, Jibin Zhan, and Hui Zhang

Public Reviewer: David Clark

This paper raises an important and timely challenge: do we have the modularity of the IP control plane right? The answer is almost certainly that we do not. And what served us adequately in the past becomes less workable with the increasingly complex set of operational, engineering and policy requirements that networks must meet. So I support the set of questions raised in this paper.

Having said that, I would observe that the paper, perhaps due to the length requirements, presents a rather one-sided and optimistic view of the situation. It offers many advantages of centralized control, but provides only a short and rather incomplete assessment of the problems and risks of this approach, in the final section. It does make sense to start off with the positives – unless they are strong, the intrinsic pain of moving to a new scheme will doom the idea. But I would argue that to carry the argument to the next stage, we should catalog the benefits of what we have today and how these might erode with a different scheme, we should identify more completely the design problems to be solved and the required research agenda, and we should identify different design approaches that might be used to move in this direction, so that we can compare them.

Here are a few (very incomplete) examples to motivate such a discussion.

Dissemination failures: One of the advantages of the current scheme is that since the routing messages flow along the data paths, if any data path exists the routing computations are likely to find it. In other words, the current approach gives us a way to reason about its intrinsic robustness. A more centralized scheme seems to raise the possibility that due to failures of connectivity in the dissemination plane, the net fails to have valid routes even though there is data connectivity.

Responsiveness: The paper appeals to the precomputation of fall-back routes for quick recovery. I have always been doubtful about this, because of the number of different faults that might arise, and the possibility of multiple simultaneous faults, e.g. due to the cut of a fiber bundle. But if precomputation is useful, it remains to be argued that it is most practical in one or another sort of centralization. It seems to me that we might be able to find some fundamental time constants to real-time routing computations, and to discover if distributed computation or a central scheme with a cycle

of data gathering and dissemination is more responsive. I would argue that we should address this directly, separate from an exploration of precomputation.

Mixed vs. pure centralization: One approach for routing would be for the routers to be quiescent without direction from the central controller. A different approach would have the routers compute a default set of routes using a scheme similar to today, and have the central control modify or override these defaults. This might be more complex but more robust.

Hierarchy: The paper is not clear as to whether the "network" discussed is the whole Internet (are we talking about replacing BGP here) or an AS. But any scheme that can scale to the size of the Internet must appeal to hierarchy or some other mode of information hiding. Section 4 does not really address the set of issues that arise with scale, and I think there are deep research questions here.

Synchronization: Section 2.1 mentions two-phase commit (although it assigns this to the data plane, while I would expect it to be in the dissemination plane.) I think there is a lot more that has to be worked out about how the dissemination plane deals with partial failures, operation in the face of inconsistent connectivity, and so on. Two-phase commit is actually too robust, in that the system has to be able to move to a new state, in order to deal with a failure, exactly when the failure is preventing all the nodes from completing the commit protocol. The system has to be able to put itself into states of "controlled inconsistency", which the distributed schemes of today do intrinsically – routers that cannot be reached are simply ignored.

I suspect that the final form of this paper is a rather long research agenda. Given the six page limit, I do not fault the authors for not digging deeper. But I would like to see a richer catalog of issues, as well as benefits, to help get the discussion going.

Designing a Predictable Internet Backbone Network

Rui Zhang-Shen and Nick McKeown

Efficient and Robust Routing of Highly Variable Traffic

Murali Kodialam, T. V. Lakshman, and Sudipta Sengupta

Public Reviewer: James Roberts

Drawing on their experience in the design of router interconnection networks, the authors of these two papers have independently come up with a similar proposal for robust network design based on generalized load balancing. They provide complementary insights and collectively make a convincing case for re-examining the current approach to routing and capacity planning. In the following, we refer to the papers as [Kodialam] and [Zhang-Shen], respectively.

The traditional approach to designing an Internet backbone is based on the assumption that we know the matrix of demands between all pairs of ingress and egress routers. Network design can then be formulated as a multi-commodity flow problem. Routing and capacities must be selected to minimize some objective function while satisfying a range of constraints, derived from physical considerations and robustness requirements, for instance. IP shortest path routing implies the constraint that demands are routed over a single path, except for possible load balancing over equal cost paths.

The papers point to significant drawbacks with this approach. First, despite recent advances in the art of traffic matrix inference, there will always remain considerable uncertainty attached to demand forecasts for individual ingress-egress flows. This uncertainty derives both from the volatile nature of Internet traffic and from the instability of inter- and intra-domain routing. A second difficulty arises due to current lack of diversity in routing choices and the long convergence times that can arise in case of failures.

To address these issues, it is proposed to adapt the load balancing techniques that have recently been developed for the design of router interconnection networks. Equally splitting traffic over an initial load balancing stage enables guaranteed 100% throughput without the need for a detailed specification of individual source destination traffic demands. The authors recognize the earlier work by Valiant that applied the same techniques to processor interconnection networks and [Zhang-Shen] actually refers to their proposal as "Valiant load balancing".

The idea is to view the Internet backbone as a fully meshed network of N nodes with inter-node links created by some appropriate tunnelling technique. Traffic from node i to node j is first routed on the tunnel from i to some intermediate node k before being routed directly from k to j. Traffic is split over all possible two-hop routes, including routes $i-i-j$ and $i-j-j$. This splitting can be performed at flow level using a hash function of packet header fields, or at packet level if resequencing is performed at the destination. The tunnels need to be sized to accommodate all possible traffic matrices that satisfy only an upper bound on the total amount of incoming and outgoing traffic at each node.

[Zhang-Shen] considers a symmetric network where the traffic constraints are identical at each node and equal to r bits/s. This can represent a physical limit or a presumed reliable estimate of aggregate traffic. The authors show that this network can accommodate any traffic matrix when ingress-egress flows are balanced equally over all N paths and each link has capacity $2r/N$. The choice of a symmetrical network makes the parallel with interconnection networks more direct and aids understanding why this solution is both effective and efficient.

[Kodialam] considers the more general case where each

node i has possibly different incoming traffic R_i and outgoing traffic C_i. In this scheme, a proportion α_k of traffic R_i is first routed to node k before continuing to its destination. The proportions α_k are the same for all nodes. It is shown that the amount of traffic on every tunnel then depends only on vectors R, C and α, and not on the individual elements of the traffic matrix. The routing is thus robust to variations in traffic for *any* choice of the α_k. The authors proceed to explain how the latter can be chosen to optimize some objective function (required capacity or maximum utilization, for instance). The problem is formulated as a linear program and can therefore easily incorporate additional linear constraints. The scheme in [Zhang-Shen] characterizes the optimal solution in one special case.

[Zhang-Shen] focuses on the performance of Valliant load balancing under failure conditions. By spreading traffic over many paths, the network is clearly less vulnerable to quality degradation in case of failure. Perhaps unsurprisingly, it is shown that to compensate for 5 failed links in a 100 node network, it is sufficient to overprovision links by around 5%. Importantly, in case of node or link failures, there is no routing protocol convergence time penalty since routers continue to load balance over all remaining paths.

[Kodialam] investigates the cost of restricting load balancing to two-hop paths and to identical sharing for all nodes. They argue persuasively that the penalty of their approach with respect to a hypothetical optimal solution of the multi-commodity flow is hardly more than a few percent more capacity. They also show how their approach can be extended to allow splitting parameters that depend on source and destination.

The latter two evaluations effectively demonstrate certain clear advantages of Valiant load balancing. However, there may also be disadvantages that are not explored. In particular, neither paper compares the cost of their solution with that of a well-planned network for which the traffic matrix is assumed known. Since load balancing systematically routes flows twice, irrespective of the distance between nodes, we can expect the resulting network to require twice as much capacity as a network where known demands follow the shortest path. There is thus considerable scope for some intermediate solution where imprecision in the estimated traffic matrix can be compensated for by some more limited, less indiscriminate form of load balancing.

Two papers in the literature are relevant to the last remark. Prasanna and Vishwanath, *Traffic constraints instead of traffic matrices: Capabilities of a new approach to traffic characterization*, Proceedings of ITC 18, Elsevier, 2003, and Ben-Ameur and Kerivin, *New economical virtual private networks*, Comm. ACM, June 2003, independently propose to solve the multi-commodity flow problem without precise knowledge of the traffic matrix. The latter is replaced by a set of linear constraints satisfied by its elements including, for example, the row sum and column sum constraints, R_i and C_i, considered by [Kodialam] and [Zhang-Shen]. Additional constraints can account for other knowledge about the matrix such as estimates of some large point to point flows or the fact that known traffic with a neighbouring domain can only use one of a small set of egress points.

A further interesting possibility is briefly discussed by [Zhang-Shen]. This is the option to use load balancing over multiple paths only for traffic that cannot be accommodated on the direct route thus reducing latency for most user flows. This does not appear to be an obvious extension of the present proposal since, to this reviewer's understanding, it would be necessary to perform adaptive routing based on the status of the direct route measured in real time. This may be possible and indeed would have a beneficial impact on both cost efficiency and resilience. There are parallels with the introduction of dynamic routing in the phone network some 20 years ago: calls that are blocked on the direct route overflow to one of many possible two-hop paths.

Rethinking the Service Model: Scaling Ethernet to a Million Nodes

Andy Myers, T.S. Eugene Ng, and Hui Zhang

Public Reviewer: Radia Perlman

The paper rightfully explains that Ethernet, as originally envisioned, does not scale. However, the paper does not really define what "Ethernet" is. Once Ethernet is modified, is it still Ethernet? What is intrinsically different between Ethernet and layer 3? In fact, this is not something that is well defined, and so the burden is on the authors to come up with some crisp differentiator. Perhaps they mean that the data packet format is Ethernet, the addresses are EUI-48, and that it supports broadcast.

I am confused by statements like "Growing existing Ethernets into a multi-site enterprise network is a more natural and less complex evolutionary path than alternatives such as building a layer 3 IP VPN." I can't imagine why that would be true. People interconnect sites all the time with layer 3. Perhaps people prefer the autoconfiguration of bridges, but that's not what that statement is saying.

The paper concludes that it is the need to support broadcast that makes traditional Ethernet not scale, and makes temporary loops an issue. However, even unicast has problems when one is forwarding a packet without a hop count, and which may spawn multiple copies at each hop (which is equally true of unicast and multicast/broadcast).

The paper should contrast what they are doing with CLNP areas, together with the ES-IS protocol. What they are proposing is basically the same, perhaps with proposed extensions for advertising services. But IPX used IS-IS (the same routing protocol as for CLNP), services advertised on the local link, and the Designated Router on each link advertised the services it learned on each link in its link state information. This is what the paper is proposing...an explicit pro-

tocol (like ES-IS) between endnodes and routers on a link so that the DR can advertise the membership of its link to other routers.

The measurement of how much ARP traffic there is at CMU's link is interesting, but they should say how many nodes are on that link.

The authors talk about the Rapid Spanning Tree Protocol, but it is really essentially the same as the traditional Spanning Tree Protocol. The only difference is how temporary loops are avoided. In the original variant, it was done by waiting for some amount of time. In RSTP it's done through some sort of complex negotiation, from what I glean from this paper. If indeed RSTP does not avoid temporary loops, this is important, and there should be a separate paper where the authors concentrate on "these are the changes from original to RSTP and RSTP does not actually prevent temporary loops", if that is indeed the case. I did not follow the example given in figure 2, partially because the description of RSTP in the paper is mostly just a description of the heart of the spanning tree protocol, which did not change between STP and RSTP. And I think might be some errors. I think the terminology "designated port" is a non-root port for which this bridge is Designated Bridge, i.e., one that has been selected by the spanning tree algorithm (STP or RSTP). The paper says "a port on a bridge that is connected to a bridge that is further from the root is called a designated port". You can have a neighbor further from the root than you, but still not be DB because there might be another neighbor who is closer to the root than you. Likewise I think there is an error in the description in the paper that implies that the only ports that might be blocking are alternative root ports. Also "loop terminates when the old root's Message AGe reaches MAX-AGE, which happens after the BPDU has traversed MaxAge hops in the network". Actually, the age counts while it sits in memory, not just when it gets forwarded. One of the changes in RSTP that I don't think makes a difference in any functional way is that I heard that they combined the "age" and "cost" fields into one field, and require that cost=hops, so you can't have different costs on different links. I haven't checked the spec to see if that's the case, but usually a standards body doesn't remove functionality (the ability to make different costs for different links).

"Not having to deal with the complication of broadcast frees the network to employ a more robust link-state routing protocol." This isn't the case. Ethernet uses spanning tree routing even for unicast. And as the RBridge paper shows, you can support broadcast/multicast with link state.

"The choice of a link state protocol ensures quick route convergence when network components fail." Actually, the spanning tree algorithm isn't slow to converge. The problem is that it is dangerous to have temporary loops as long as the packet format has no TTL and as long as forwarding spawns extra copies (since no "next hop" is specified, and bridges forward onto multiple ports when the destination is unknown). So the slowness was because of the 30 second (which is a parameter) wait to make sure that anyone who needs to turn off will do so. Even if it were a link state protocol, if the forwarding were done with an Ethernet header, it would be safer to wait some time before forwarding because temporary loops would spawn multiple copies, and not go away for a long time (with no hop count). The paper sort of says this, but with the wrong conclusion "Temporary routing loops are still possible, but since broadcast is not supported, a data packet can at worst persist in a network loop until the loop is resolved; there is no danger of exponential duplicate packet proliferation." As I said, unicast packets can proliferate, and at gigabit speeds, a packet can loop an awful lot of times, slowing down routing convergence, without a hop count.

In summary, the solution in this paper is not functionally different from either CLNP/IPX with IS-IS or RBridges, both of which have DRs figure out the membership of their link and pass around this information to other routers in link state information. I think the extra encapsulation done by CLNP and RBridges is important (in order to specify both next hop and ultimate destination, and to have a hop count) so that temporary loops are not a problem.

I would like to see the authors write a paper describing the changes from original spanning tree to RSTP and critique them. But even if RSTP accomplished what it was hoping to do (replace the one piece of STP in which temporary loops were avoided by waiting some amount of time by something that would be able to know the exact amount of time it was safe to turn on a link), the basic algorithm is the same between RSTP and STP, and has the same properties of suboptimal pt-to-pt routes, not being able to spread traffic around, etc.

On the Scalability of Fair Queueing

A. Kortebi, L. Muscaliello, S. Oueslati, and James Roberts

Public Reviewer: Thomas Anderson

A fundamental assumption underlying almost all modern congestion control research is that fair queueing is impractical because per-flow state is infeasible at Internet speeds. Yet a simple cost-analysis shows that this assumption is no longer valid. I was part of a design team at DEC SRC that over a decade ago built a commercially viable ATM switch with gigabit links, per-flow state and approximate fair queueing. Hardware has only gotten cheaper since then. A gigabit Internet connection, when used, costs the end user about half a million dollars per year. Suppose we store every packet header that traverses the gigabit connection in SRAM, and keep them for sixty seconds. Clearly, we don't need to be this profligate, but what if we were? At current SRAM prices, the storage would cost only a few hundred dollars; even at every hop along the path, it would still be less than 1% of

the end user bandwidth cost. This gap between WAN bandwidth prices and memory is widening over time; memory is getting cheaper than bandwidth at a rate of 5-10x every decade. Thus, if we look backwards, it should be no surprise that the Internet eschewed per-flow state; it would have been prohibitively expensive to add per-flow state to the Internet equipment of the early 70's. There is no reason for us today to continue to be yoked to the optimizations of the past.

Kortebi et al. add another plank to bolster the argument that fair queueing is practical to implement. First, they observe that a fair queueing implementation can discard previous state whenever the link goes idle; in this case, all flows can be scheduled immediately on arrival. Thus the time/space complexity for implementing fair queueing is proportional to the number of flows with packets that arrive during continuous busy periods; on the lightly to moderately loaded links typical of today's networks, these busy periods are likely to be short.

Second, they observe that only flows requesting more than their fair share need to be tracked; an important result of Internet traffic measurement is that most flows are small but most packets are due to large flows. To be effective in most practical situations today, one only needs to do fair queuing on large flows, a small subset of total flows. Various techniques can be used to recognize these large flows with minimal state.

Of course, fair queuing might become a victim of its own success, by creating the conditions for network equipment to be more effectively used than it is currently. (Much of the benefit of the Kortebi et al. work would disappear if links were completely busy all the time.) It should seem a bit odd that WAN bandwidth is so relatively expensive, yet most links are kept intentionally overprovisioned. One plausible explanation is that TCP congestion control has failed at its mission of managing bottlenecks; since a single large flow can cause packet loss on unrelated small flows, ISPs intentionally overprovision links to prevent this from happening. They would no longer need to do this if we had fair queueing. But ISPs would probably continue to overprovision to some degree, to provide acceptable performance after failures, to provide better end user response time, and to maximize profit, since the bandwidth pricing model for most ISPs is based on burst usage. Thus it is likely that much of the effect Kortebi et al. identify would still hold, even in a future network with ubiquitous fair queuing.

Harnessing TCP's Burstiness with Flowlet Switching

Shan Sinha, Srikanth Kandula, and Dina Katabi

Public Reviewer: Thomas Anderson

This paper presents a novel idea: can we use TCP's burstiness for something useful? Recall that packet sends in TCP are bursty because of ack clocking. With ack clocking, packets are usually sent only in response to successfully received acks. This has the positive benefit of causing the sender to immediately stop if the network becomes so overloaded that acks stop returning, but it has the downside that it tends to keep the packets from a given sender clustered together, particularly on high bandwidth paths. This burstiness causes all sorts of problems, from increased variability in queue delay to increased loss rates. As a result, many researchers, including myself, have explored or advocated smoothing, or "pacing", TCP sends to smooth out this burstiness.

Sinha et al. take an opposite tack. They observe that TCP induced clustering of packets has a notable positive side effect; all the packets in a flow sent within a round trip time will normally be clustered together in time. They call this cluster of packets a "flowlet." If a particular flow is re-routed only in the periods between flowlets, and the difference in latency between the two paths is small compared to this period, then it is likely that the receiving host will continue to receive packets in the order that they were sent, despite the re- routing. Since TCP treats out of order arrivals as a potential signal of packet loss, this prevents the receiving host from misinterpreting the routing change as something more serious.

Of course, there are many details to work out, such as whether a router can reliably recognize a flowlet in real-time, what happens when flows are throttled through thin access links (spreading their packets out), etc. But suppose it did work. Would it matter? For today's Internet, probably not. One common practice in the Internet today is called "equal cost splitting" – where packets from a single flow are split across parallel links connecting the same routers. Theoretically, this could cause problems: because the Internet is store and forward, it is easy to observe short packets racing ahead of large packets along these parallel links. However, since most TCP connections don't send interspersed long and short packets, this rarely causes confusion at the receiver.

What I like about this paper is that it may help to put to rest a longstanding myth that fine-grained traffic engineering is impractical. In general, the only way to achieve balanced network utilization is to do some form of fractional routing – to split packets between the same ingress and egress (source and destination) along multiple paths. But if an ISP were to do this, e.g., using MPLS tunnels, the end hosts would treat the resulting chaotic ordering of arriving packets as a signal that packet loss had occurred, completely negating any potential benefit to more balanced traffic engineering. Of course, there are proposals to implement fractional routing using a hash function on flows – by keeping flows routed along a single path, one can avoid reordering at the endpoints. If some flows are huge (the so-called elephants vs. mice), however, it may not be possible to achieve both balance and single path routing per flow, without some form of dynamic re- routing of large flows to achieve the target

bandwidth fraction. The paper shows that it may be possible to do this fine-grained re-routing without triggering the loss recovery mechanisms in TCP.

An open question, unanswered by the paper, is whether this re-routing must happen on every round trip or whether it could be done less frequently and still be effective. Re-routing large flows arguably should be done conservatively, because they might have a large impact on bottlenecks along the new path. Thus, it may be better, not worse, for the network if re-routing to cause spurious packet loss detection and a temporary reduction in rate. Finally, flowlet re- routing may be too coarse-grained – suppose you have a single large flow, and need to split it along several paths. Flowlet splitting will cause it to flop back and forth, sending all the packets from the flow during a single round trip along a single path, achieving balance only at the wrong time scale. These questions can be answered by future research.

Any architects reading this are of course cringing. Wouldn't it be better to fix the endpoint TCP stacks to be more robust in the face of packet reordering? Of course. I would argue, however, that there is still a value in understanding techniques to decouple design decisions. It is a non-starter to simultaneously change both ISPs traffic engineering practices and endpoint TCP behavior. We will have an easier time fixing TCP if ISPs do fine-grained traffic engineering, and we will have an easier time fixing ISP traffic engineering if we don't have to worry about its impact on TCP.

Can we contain Internet worms?

Manuel Costa, Jon Crowcroft, Miguel Castro, and Antony Rowstron

Public Reviewer: Vern Paxson

At a high level, the current state of the art in research on combating Internet worms tells a story something like the following. First, random-scanning worms are "solved" (from a research perspective) by the development of network-based detectors that with high accuracy can suppress worms that spread by probing randomly-selected Internet addresses. Second, for worms that spread by other means, such as "topological" worms for which infectees locate potential new victims using information located on the infectee itself (e.g., address books for email-based worms), the story is much less complete. While there has been considerable research activity - schemes that look for common content in network traffic, techniques for hardening hosts to resist infection in the first place, efforts to deploy and monitor "honeypots" that deliberately become infected, prospects for detecting "behavioral signatures" that discriminate between worms and benign activity - many large hurdles remain.

One major hurdle concerns the problem of *trust*. Because worm propagation can span the entire Internet, there is great interest in using deeply distributed sensors for detection. But how can your site trust alerts sent to it by my site? I might be an adversary wanting to steer you into an anti-worm reaction that will itself inflict damage on your site. Or, I might be an innocent party whose sensors themselves have been compromised by a propagating worm that seeks to muddle global-scale detection.

The core contribution of this paper is an intriguing idea for how you can trust alerts you receive regardless of their pedigree. The notion, which comes from that of proof-carrying code, is that "Self-Certifying Alerts" include with them an audit trail that demonstrates the path of execution used by the worm's exploit. By confirming the correctness of the audit trail, you can know that there definitely is a corresponding exploit. While you don't know for sure that it is actually being used by a propagating worm, you likely won't go wrong fixing the vulnerability revealed by the exploit (by automatically patching the corresponding code, or installing some sort of filter to prevent access to it).

This approach raises a number of issues, which the "Vigilante" system presented in the paper only partially addresses. One of the more well developed of these is how to detect the exploit in the first place. Vigilante can accommodate a wide range of mechanisms for doing so, which is part of the appeal of its architecture. The authors discuss one scheme in particular, a conceptually simple mechanism for detecting a broad class of exploits that work by altering a program's execution, either directly (executing injected instructions) or by modifying its control flow, for example by overwriting a return address on the stack in order to call existing code with injected arguments. The basic idea is to track "dirty" data, i.e., data that arrives from an untrusted source. Picture associating a bit with each word in memory (and each register) which, if set, means the data in that word was ultimately derived from an untrusted source such as network input. Machine instructions that use a dirty word as an operand propagate the dirtiness to values they produce. Instructions fetched from dirty memory locations, or clean themselves but which use dirty operands for control flow (such as branching indirectly through the contents of a dirty word), generate fatal faults.

This mechanism is not a panacea. For example, it cannot detect instances where dirty data alters control flow by being the operand tested by conditional branches, such as for shell-escape exploits (or, more generally, logic exploits). But it can thwart a whole class of attacks, which includes the majority of those used by worms today.

The basic notion is quite old (1970s). Two key issues with using it in practice concern performance overhead and "false positives" in terms of the mechanism interfering with benign operations (for example, downloading a patch for a running program). The implementation in the paper is done entirely in software, and for an unoptimized implementation incurs a high performance overhead (50x). The authors make an interesting but only half-convincing argument that overhead on this order is acceptable, which I absorbed as follows: be-

cause the nature of worms is to exploit large populations, generally there will be potential victims that tend to be idle, and thus can afford to burn cycles on executing the expensive protection mechanism. It's not clear to me that such machines will be available and operating in this fashion for the services that future worms attack - but the argument is still stimulating to think about. In addition, concurrent with the authors' work, Crandall and Chong developed hardware that implements essentially the same scheme (see [4]). This greatly reduces the performance impact, and the hardware is simple enough that it seems plausible it may in fact be implemented for future CPUs.

The paper's treatment of other issues with Self-Certifying Alerts (SCAs) is less satisfying, however. To keep SCAs small, the authors state that information "not essential to trigger the vulnerability" can be removed, but it is unclear how such information is identified, or even whether the problem is decidable. Another issue concerns the growing "arms race" in which malware uses mechanisms to detect when executing in altered environments such as virtual machines, in order to thwart analysis (already a significant problem for virus detection). Then there's the question of just how a host receiving an SCA replays it to verify its correctness. While the paper states that techniques from fault tolerance can be used to replay execution, it is not clear with what complexity or expense. What if the audit trail requires network activity or altering state stored on disk? What if it takes trillions of cycles, or simply doesn't terminate? In general, can worms propagate faster than the SCAs can be generated, propagated, verified, and acted upon?

While the paper acknowledges the basic problem of adversaries devising problematic SCAs, and discusses a few such angles (e.g., flooding of SCAs), the paper would have been significantly stronger had it included more thorough enumeration and exploration of the possible issues. While there is only so much one can express in 6 pages, I wonder whether a sense of HotNets requiring "big ideas" got in the way here. To my thinking, the paper unnecessarily spends time on trying to claim very broad significance. This manifests mainly in discounting network-based worm detection techniques, which are criticized as limited to known attacks or suffering from undue false positives. Disclaimer - these techniques are one of the areas in which I work - but I find these comments both inaccurate, and, more to the point in terms of the authors best telling their story, an unneeded distraction (as is the paper's vague, all-encompassing title).

Instead, tell us more about the interesting, perhaps-very-hard research still to be done for us to achieve truly workable SCAs. And, come to think about it, how this approach might integrate with network-based defenses, rather than competing against them.

Toward a Framework for Internet Forensic Analysis

Vyas Sekar, Yinglian Xie, David A. Maltz, Michael K. Reiter, and Hui Zhang

Public Reviewer: Alex Snoeren

This paper proposes an elaborate monitoring system to determine the source of Internet attacks. Rather than focus on a particular type of attack or infestation, the authors attempt to provide a generic technique, leveraging a simple feature common to all Internet- based attacks: communication must occur between the attacker and the victim. Of course, as attackers become more and more sophisticated, the communication is becoming increasingly indirect and difficult to distinguish from benign traffic. Unlike previous traceback systems, which only attempt to isolate the origin of a particular packet or attack flow, the presented scheme attempts to go a step further, identifying the entry point—or patient zero—of an attack or infestation. Doing so basically requires solving two extremely difficult problems: determining the origin of individual packet flows, and correlating these packet flows to infer communication patterns that pinpoint indirection points and stepping stones in an attack.

Given my background in the area, I can't help but view this work as a combination of stepping-stone techniques (first advanced, to my knowledge, by Staniford-Chen and Heberlein) with per-flow traceback (such as my own work on hash-based traceback at BBN). In the current incarnation, however, the authors obviate the need for flow traceback by implicitly assuming that traffic is not spoofed—that is, traffic sources can be identified from IP headers. Hence, the authors argue that simple flow records can suffice for tracking communication between hosts. Given the proliferation of bot-nets and zombies, this may currently be a reasonable assumption. One might argue, however, that as soon as stepping-stone detection techniques become sufficiently powerful to trace through indirection nodes, the attackers will return to spoofing source addresses to further obfuscate their location.

A larger problem, however, arises from the construction and maintenance of these flow records. It's easy to point at the massive storage requirements and dismiss the scheme out of hand; I'm more inclined to believe in the power of Moore's law and the resources of motivated ISPs. I worry instead about the liability issues. If implemented as described, Tier-1 ISPs would be in charge of maintaining logs of all flows that cross their network: who talked to whom, and when. Imagine the value of that information if it found its way into the wrong hands (and here I'm not just worried about "attackers," but even entities who purport to be looking out for the citizens' well-being). Obfuscating these traffic logs was one of the key motivations behind our hash-based traceback work: given the legislative environment of the time (circa 2000), network operators indicated it would

be politically impossible to even admit the existence of a system that could provide such logs, lest its implementation be mandated and the records subsequently subpoenaed.

While flow identification and storage is clearly problematic, it appears to be the simpler of the two problems. The authors provide sketches of several schemes to infer causal communication patters, but it's unclear how well they work with perfect flow information; the problem seems far harder with incomplete or noisy communication data. With many attacks consisting of only a handful of packets, its easy to imagine many flows will escape detection if packet monitoring is conducted in the core. Conversely, many forms of malware, like email worms, use communication channels that are well-established. For example, an email virus will proceed from a client to a mail server just as any other email would. It would seem to require a high degree of coverage to separate virus mail from normal email using only content-agnostic statistical techniques.

Regardless of how you come down on the practicality of the proposed scheme, the problem is obviously of critical import, and the framework seems to be a plausible direction of research. The staggering resource requirements of this and similar forensic schemes, however, beg the question of whether we should focus on determining the cause of an attack, or if we'd be better off preventing its spread to begin with. Unfortunately, I'm not sure that's an entirely technical question.

Spam-I-am: A Proposal for Spam Control Using Distributed Quota Management

Hari Balakrishnan and David Karger

Public Reviewer: Jon Crowcroft with Richard Clayton and Christian Kreibich

If there is one topic that will start a debate amongst a set of networkers, it is Spam. I have encountered three platforms in such debates in the corridors, mail lists, Blogs and Wikis of the world:

1. Technocratic Optimism

2. Socio-economic Pessimism

3. Hybrid Realism

This paper by Balakrishnan and Karger out of MIT belongs in the first category. Since one is sure to find occupants of the other positions at Hotnets, it is highly likely to provoke a lively discussion.

Spam is relatively hard to come up with a purely technical definition which is part of the challenge. Spam is not simply unsolicited e-mail: Much email I deal with as an academic is unsolicited, but, for example, perfectly reasonable queries about research or student places on courses. There will be legitimate reasons for unheralded missives for many people in many walks of life. Hence closed user group approaches (white lists) are not going to suffice, although they may be a useful component of the solution space. Spam is not merely marketing: I am subscribed to many services that make targeted recommendations, (advertisements) by e-mail. These are not Spam. Spam is not merely a scam. Sometimes, it actually is relevant to me! Much Spam is a scam, but not most.

The problem of Spam has really been perceived in the way that it is generated. Lists of targets (victims) are constructed without regard for the role of the recipient, simply because it is cheap to do so. To transmit Spam at a level that reaches large numbers of thoughtlessly composed mailboxes requires non-trivial resources. Together with the fear of being caught, this has led to the trend of enlisting of zombied machines around the net (often via worms, viruses and other hacks) Spam generators. Notice then that Spam is a many-to-many phenomenon; it is not always sourced from bogus locations, but it is often not intended to come from those locations.

The content of Spam, as worms and viruses, is frequently polymorphic, adapting itself more or less, to the alleged recipient ("My Very Dear Esteemed Crowcroft"). This makes simple rule based filtering ineffective. Indeed, the more we put in the Spammers way, the more they differentiate the content by receiver.

The problem of unsolicited e-mail (the application layer equivalent of DDoS to some people's way of thinking) is widespread and familiar to the lay public as it is to the networker. It is a reasonably recent phenomenon, almost post ".com bubble". I have all my e-mail since 1976 in a set of indexed files (much of it now uploaded to some gmail accounts), and if I run one of the common Spam filter tools on the data, I can see that there was very little before 1999. It has since then grown at a rate that exceeds the Internet host count! Today, I count around 50would regard as Spam, consistent with many surveys, some cited in this paper. This makes it a significant problem: When I am on vacation for more than a week, my mail box often exceeds my (generous) file system quota! However, for the last 2 years, I see almost none of this Spam. This is because my organisation uses a Spam filtering mechanism which, provided it is updated frequently removes (or in my case by request, marks specially with a rank) Spam. Of course, this consumes resources such storage, CPU, network capacity and people However, with a couple of staff and 1 big server, 30,000 mail boxes are largely filled only with things they were meant to be. This scales reasonably for a large technical organisation, who need to protect resources in a variety of ways with IDS and firewalls as well, in any case, but not for the smaller outfit, although many ISPs will offer such a service and it can be quite effective: My wife's company is an online recruitment agency matchmaking engineers and architects to posts on large projects all by web and e-mail; they see very little Spam: even though a lot is sent to them, it is filtered

by their ISP. The problem with this is that resources are still consumed, and the management is repeated at all the different sites. Thus a pure end-to-end technical approach such as filtering, even with very rapid evolution and dissemination of filter rules, is not going to be good enough on its own.

Some countries have attempted to legislate against Spam. Such approaches depend on catching and fining the culprits. That requires the ability to find the culprit, a jurisdiction that covers the sender, and proof of the identity of the sender and other admissible evidence. This does not work in a networked globe. Even if open SMTP relays are removed, traceback often leads to a pile of dusty PCs in a remote corner of some Internet Cafe that have been subverted by the wily Spammer. So legal remedies are not quite enough.

This paper's technical contribution comes from the technocrat platform. The authors propose a resource management approach to the problem. The approach is really an extension of an existing one from Burrows et al at Microsoft. The idea came from computational economics. Some researchers have proposed the idea of billing per E-mail as a solution. Charging as a means of resource management is popular in many areas today: Telecom folks have often said that the Internet would have far less DOS attacks, if only we had connection-oriented network layer protocols with admission control, and per session per volume charges. This sort of skates over the existence of much telephony hacking (phreaking) and junk faxing. Worse: if one were to subvert a number of cell phones, one could do all the same things a Spammer does (indeed junk SMS is now a fact of life) and incur a bill for the poor unsuspecting owner of the zombied device! Thus purely economic medicine will just not do the job. In fact, any cost high enough to deter Spammers, will also mitigate very unfairly against poorer internet users: e.g. people in developing countries will no longer be able to enquire about PhD scholarships in your University! Many such schemes open up attacks on users' purses through identity theft.

The Penny Black idea is to virtualise the "charge" to senders. To deliver a message to a receiver, a sender must first obtain a capability, and this capability needs to be unravelled. The process of getting the obfuscated key is designed to cost something non-monetary that will take time, e.g. CPU cycles. The most recent variant of this is memory-bound; this deals with one objection to the proof of work through CPU puzzles, which is getting the amount of work (much like setting the price point) right- there's a couple of orders of magnitude of variation in CPU speed in deployed systems on the net, whereas there is much more uniformity of memory speeds. Again, as with the spam filter rule-bases, the Penny Black project has a central repository of such problems.

In the Spam-i-am proposal, what we have is really a cousin of the Penny Black ideas, but decentralised. By making use of the ubiquitous computing idea of Distributed Hash Tables, we can store, retrieve and check the "rights to send" efficiently for very large numbers of participants. This can be provided as a third party service, only needing modification to Mail User Agents at send and receive time, although easily also adapted to batching and applied to SMTP relays (MTAs) too. The scheme in the paper is elegant, and actually seems to me to be one of the first times I have seen DHTs used for something directly applicable as well as useful!

However, as with the previous approaches, this solution on its own is not sufficient. As with any purely technical solution, this too will be gamed by the Spammers. In particular, the trend towards stealing service will accelerate (and some estimates now put the volume of spam sent through hijacked machines at over 75%). The authors dryly observe that there will now be some financial incentive to fixing security holes. But will end users be satisfied to be out of pocket for $40 just because they clicked on a Trojan and infected their machine. Why indeed will the system ever get off the ground when you are asking people to spend money to make other people safe from their indiscretions. Also the authors assume that this scheme might be slow to take off, with only some email being checked. In fact, one can safely predict that as with SPF the early adopters will be the spammers – and they'll be sending stamped email long before anyone else, trading on the fact that initially very little will be checked so that they can re-use stamps. The obvious result of this prediction is that truly adaptive filters will be treating a stamp as evidence of wickedness and will have to be forcibly re-educated as the scheme came into wider use! The authors admit that maillists are hard to accommodate in this scheme: there are legitimate one-to-many mail shots such as CERT advisories:-) Most technically worrying for me is that while this use of DHTs is very appropriate and elegant, P2P system security itself is a very new area. P2P systems designers have many difficult problems yet to solve, particularly, as the authors themselves observe, the introduction of malicious nodes that modify in-flight data, or swamp the system with bogus content. Remember, Spammers have huge numbers of systems that they have already undermined, and use to source Spam: adding multiple keys and senders that appear legitimate is an easy step for them in this anonymous world, and attacking the DHT is very likely.

Some other niggling questions I would ask in the session: the authors ask us to disinfect our machines thoroughly, which is pretty tricky today; one would expect to retain the ability to send 9/11 E-mails, something that undermines the zero tolerance of exhausted quota; users will want to monitor their quota, as will system administrators which may lead to all sorts of markets in quota; finally, there is the potential for enormous loss of privacy in this system: it seems an interesting challenge to see if one can introduce resource management but retain anonymity; indeed, one of the earliest peer-to-peer proposals, the Eternity System by Ross Anderson, was aimed at solving precisely this problem.

To conclude, I think this is a good proposal for a technical part of the solution space. This must needs be combined with other pieces which I have mentioned above, and others,

before we return back to the Halcyon days of the Spam free Internet that I recall in 1980.

I am tempted to finish with a response to their paraphrase of Dr Seuss: In a true network eco-system, only Green Spam would be permitted, and technical, economic and legal means would be devised that align together to remove the incentives that only Eggs 'em on.

Revisiting MAC Design for an 802.11-based Mesh Network

Bhaskaran Raman and Kameswari Chebrolu

Public Reviewer: Robert Morris

The attraction of this paper is that it describes new ideas spurred by experience with a production mesh network. While there are many energetic community mesh networking projects underway, they tend not to communicate their experience to the networking research world. Wireless networks are notoriously hard to model or simulate accurately due to subtle interactions among radios and between the radios and the propagation environment; for this reason fresh infusions of real-world experience are welcome.

The authors' network uses directional antennas to form point to point links. Each node has multiple such links, with one radio per link. Local regulations limit the network to a single channel shared by all the links. Side-lobes in the antenna radiation patterns mean that when a single node sends and receives simultaneously on different links, it loses many packets. Ensuring that a node only transmits or only receives at any given time sounds like a good potential solution, but could turn out to be impractical depending on antenna sensitivity patterns at receivers, carrier sense at the senders, and the ability to impose a global send/receive schedule. The paper demonstrates experimentally that antenna sensitivity and carrier sense need not prohibit this solution. Finally, the paper proposes a practical distributed scheduling scheme to assure consistent send/receive decisions by all adjacent nodes.

Perhaps the most interesting question left open by this work is whether it is needed in areas fortunate enough to enjoy multiple non-interfering 802.11 channels. Most areas have three such channels. Even in the context of a single network, the channels may not all be available for directional links: often each node will reserve one channel for an omnidirectional antenna for local clients. Thus a node with as few as three directional links might benefit.

No Long-term Secrets: Location-based Security in Overprovisioned Wireless LANs

Daniel B. Faria and David R. Cheriton

Public Reviewer: Robert Morris

The basic idea in this paper is to use the limited range of 802.11 radios as an easy way to implement a geographic network access policy. Such a policy might sound like "anyone physically inside our building is allowed to use our wireless network." The core technical challenge is making the border precise, to deny access to someone just outside the building's walls.

It's easy to imagine situations in which a geographically based access policy would be convenient. A research lab might have frequent guests to whom it is happy to extend wireless service, but not want to give away service to inhabitants of an apartment building across the street. A cafe might want to extend free service to customers sitting inside but no-one else. In both cases the management overhead of distributing per-user keys or adding MAC addresses to access lists would be high for transient users. Equally important, the penalty for incorrectly granting access is low. The proposed system, which would be convenient though perhaps error prone, seems appropriate for these examples.

The paper is neat because it takes advantage of the properties of radios rather than trying to mask them or abstract them away. It proposes a security policy that's unusually intuitive, since it's based on physical perimeters rather than abstract authentication logic. The paper provides a sketch of an implementation using models of the relationships among signal-to-noise ratio, distance, and packet error rates. Finally, the evaluation properly reflects the system's overall goal of reducing management overhead by estimating the reduction in capital and operating costs.

The key question the authors will have to answer as they continue this work is how tight a bound they can place on the location of clients in a real deployment. The more densely one places access points, the lower transmit power they can use, and thus the closer clients have to be to an access point to use it. However, a building with dense low-power access points might therefore have a good deal of natural network access control, without the techniques proposed in this paper. Thus the proposed system must out-perform access-list techniques economically, and provide more accurate explicit localization than the localization implicit in low-power access points.

Smart-Phone Attacks and Defenses

Chuanxion Guo, Helen J. Wang, and Wenwu Zhu

Public Reviewer: James Kurose

The long-promised "convergence" of telephone and data networks may indeed be just around the corner. While this convergence will undoubtedly bring new benefits, it will also bring new challenges, one of which – the infection/compromising of smart phones and the consequent ability to attack the phone network – is the topic of this interesting paper. The basic premise of the paper is that smart phones, which download/run applications and connect to many networks (wireless data networks, cradle-synch to a PC, cellular), are subject to many possible infection vectors. Moreover, once infected, these smart devices may attack a phone network in ways previously not possible when phones were passive, "dumb" devices.

As the authors themselves note, the purpose of the paper is to "alert the community on the imminent dangers of potential smart phone attacks against [the] telecomm infrastructure." A number of possible DDOS scenarios (base station DDOS, call center DDOS, spamming, and identity theft) are outlined. Some initial approaches towards addressing the problem: (i) hardening the systems in several ways, including attack surface reduction, e.g., by turning off unused functions by default (ii) detecting attacks by monitoring traffic statistics (the authors indicate that telephone traffic is "highly predictable and well-managed" and so detecting attacks is easier than in the Internet), and (iii) checking end-point devices for needed security patches or shielding (e.g. rate limiting).

As is the case with many Hotnets papers, the paper raises as many questions as it answers. Indeed, the authors note that much of the space is yet to be explored. Many interesting questions would seem to involve differences between the telephone network, and the Internet. For example, from an architectural standpoint, does the well-defined user/network boundary in the telephone network make it less susceptible than the Internet to attacks, or limit attack severity? While a compromised PC can attempt to inject any traffic at any rate it desires, a compromised phone, must signal before making its (typically single) fixed-rate call, and can not signal (or send voice) at an arbitrarily high rate. In wireless phone networks, does the HLR/VLR architecture make it easier to check for and detect known compromised smart phones? Is the telephone network's signaling plane (which is accessible only through the well-define user/network signaling interface) subject to attacks that are different from those launched on the Internet's control infrastructure (e.g., DNS, routing)? How are attacks on the signaling infrastructure different from the DDOS attacks that can be launched on the data plane infrastructure? The different charging models may also aid the detection of compromised phones. Even if a zombie phone is not immediately identified, telephone calls are logged (and often billed for), making the smartphone owner more likely notice anomalous behavior in comparison to the compromised PC owner, who has little/no knowledge of outgoing traffic.

In summary, the paper alerts the community to number of important and timely issues, and identifies avenues for future research. Perhaps the longer-term result will be a deeper understanding of two very different network architectures (the telephone network, and the Internet) – how susceptible these architectures are to attack, and how easy/hard they are to defend.

Reduced State Routing in the Internet

Ramakrishna Gummadi, Nupur Kothari, Young-Jin Kim, Ramesh Govindan, Brad Karp, and Scott Shenker

Public Reviewer: Ellen Zegura

There has been a recent surge of interest in fixing the myriad problems with BGP. Indeed, you will find several such proposals in this workshop alone. Some efforts in this area are wholesale redesigns, starting with a clean slate and redesigning interdomain addressing and routing, with the issue of deployment typically set aside. Others are incremental and designed with backwards compatibility as a primary concern. Against this backdrop, this paper occupies an interesting point in the design space. It does not propose to fix the problems with BGP. Instead, it solves a relatively narrow and focused problem: reducing the size of forwarding tables while explicitly keeping current BGP and IGP functionality. (The reader will remember from Networking 101 that routing tables and forwarding tables are different, and that forwarding tables are critical to router performance because they are accessed on the fast path for IP lookup.)

The key idea to accomplish the reduction in forwarding table size is to adapt the geographic routing techniques from the wireless world, as well as rewriting address labels at the ingress to each domain. The research community has seen many instances of adapting wired solutions to fit the characteristics of wireless, but it is relatively rare for ideas to flow the other way. This is an interesting twist. The paper specifically addresses the challenges of maintaining the intradomain traffic engineering and inter-ISP policy routing capabilities of today's Internet, while also allowing the use of geographic routing and the accompanying reductions in core router forwarding state and processing.

Several questions come to mind in reading this paper. How amenable are the ideas to incremental deployment? Would network providers be willing to reveal the geographic location of routers and end- systems, or would that information be considered too sensitive to make available via DNS? Will the ideas make sense in the context of proposals to change interdomain routing, either partially or completely?

The SoftRouter Architecture

T. V. Lakshman, T. Nandagopal, Ram Ramjee,
K. Sabnani, and Thomas Woo

Public Reviewer: Nick McKeown

Have you ever wondered why routers are so complicated? After all, they don't seem to do much. In most cases, they accept a packet, do some header checks, lookup the next-hop address, decrement the TTL, update the checksum and send it on its way. The requirements of an IPv4 router are contained in a single RFC, and a typical networking class covers the requirements in a single slide. So how come a backbone router is seven feet tall, consumes 10kW, and costs a couple of million dollars? And how come a 10Gb/s OC192 linecard contains 30million logic gates, occupies several square feet of board space, has 300Mbytes of buffering and costs over $100,000? And finally, how come a router's operating system has ten million lines of source code (almost as much as a 5ESS telecom switch), and is less reliable than most laptops?

Routers are complex, unreliable and expensive monsters. They have to support multiple flavors of multicast, security, load-balancing, routing protocols, and so on, that are specified in thousands of RFCs. Internally, they require complicated algorithms (lookup algorithms, crossbar schedulers, buffer managers, packet schedulers, classification algorithms), and specific support for monitoring, security, reliability, fault-tolerance and robustness. Plenty of problems to keep thousands of engineers busy at networking companies, and a few architects and researchers awake at night. It's not surprising they need frequent reboots.

Solutions to these problems can sometimes be borrowed from other fields, such as software engineering or computer architectures; but several interesting problems are unique to the design of routers and switches. And as technology changes, new problems emerge, such as the large random access time of DRAM, and power consumption of fast circuits.

Despite the number of interesting problems, there is a general lack of research on the overall architecture of Internet routers. For example, what architectural approaches lead to simpler implementation, greater reliability, higher performance, or more scalability? Compare the number of papers on router reliability with the number on computer system reliability, and you get the picture. This is quite surprising given how much we collectively rely on the network functioning correctly. Part of the problem lies in the proprietary nature of the network industry. A router vendor only needs to conform to Internet standards at the external interfaces of the box; they don't need to tell us how they do it, and so in practice they keep their techniques closely guarded secrets. Contrast this with the computer industry that has to tell us how their systems work so we can program them. This has lead to an academic openness and pride with each new generation of computer; and in the networking industry, it often leads to reinvention of the wheel, and kludgy implementations.

But patterns have emerged, and a small number of architectures are now commonplace. For example, routers usually consist of multiple linecards arranged around a switch fabric. The switch fabric might be a simple backplane, or could be a crossbar switch and scheduler. There are usually one or more route processor cards that run routing protocols and manage the system. The datapath – or forwarding path – of a router is typically built from custom-designed ASICs, while the control plane is based on a standard microprocessor. The datapath is on the linecard, and the controller is housed on another card in the same rack.

The self-respecting architect will have wondered - at some time or other - if the control microprocessor really needs to be in the same rack. Couldn't a controller (or several redundant controllers) manage multiple routers in multiple racks of equipment? And then can't we use the network to connect the controllers to the datapath, and possibly make the controllers remote — somewhere else in the network. It seems this might lead to a centralized and more manageable control strategy, and simple lightweight datapath switches. Furthermore, if the protocol that runs between the controller and the datapath is an open standard, we can decouple the development of the controller from the datapath, and even introduce competition inside the system. Perhaps a single, network-wide controller developed by one company could manage and control a variety of hardware forwarding elements from another vendor.

It was thinking along these lines that led to the SoftSwitch concept in telephony, and later the Multiservices Switching Forum (MSF) for ATM switches and IP routers. More recently, ForCES has been introduced in the IETF as a way to separate control elements from datapath elements in routers. And this was the starting point of the paper by Lakshman et al., and the proposed SoftRouter.

Others have tried to decouple routing from forwarding, to limited degrees and with varied success. Software routers such as routed, zebra and now XORP do this; in fact, XORP has a well defined FEA or forwarding element abstraction layer, so that it can support multiple forwarding paths (Click, Linux, FreeBSD etc). We tried something similar in a tool for a "Networking Hardware" class at Stanford, called VNS/NetFPGA: we pulled the control element out and run it from a PC many hops away. The most comprehensive work in this area was probably DCAN at Cambridge, and the associated work on switchlets.

The SoftRouter decouples the control element (CE) and the forwarding elements (FEs) by pulling the CE out onto the network, and allows the creation of big forwarding elements by arranging a number of adjacent smaller ones. While similar to ForCES, the SoftRouter approach is more radical and more interesting, particularly by allowing dynamic binding of controllers to forwarding elements.

Separating the control elements from the forwarding ele-

ments introduces as many questions as it answers. What is good about this paper is that it identifies the main issues that need to be discussed, collecting them together in one place and creating a framework for discussion and debate. This is what the Hot Nets workshop is all about, and I'll look forward to the discussion.

There are several interesting technical challenges: Bootstrapping (how does a forwarding element find and trust its controller? And what happens when one of them dies?); deciding what functions belong in each entity (for example, where are unusual IP options or error conditions processed?); and what happens when things fail and the network topology changes.

The paper doesn't shed much light on these problems, but hopefully it will motivate others to provide answers. For example, the bootstrap problem is non-trivial. If a forwarding element in one part of the network is to be controlled by a controller in a remote location - perhaps thousands of miles away - how does it find the controller? Does it have to know a priori which controller to trust? If so, what happens when the controller fails? How can it authenticate the controller? It seems that we need a simple, secure and robust protocol that works in arbitrary and unreliable topologies. The paper doesn't offer us a solution. And perhaps it will take so much complexity in the forwarding element as to throw the whole technique into question. Maybe.

The paper also begs the question at to whether the Control and Forwarding should be co-located or geographically separated. If they are separated, they could be in adjacent cities, hundreds of miles (and many milliseconds) away. Do delays in configuring and monitoring the forwarding element affect its performance? Certainly the CE-to-FE protocols must be simple and lightweight. Is the forwarding element smart enough to be autonomous during emergencies? The FE probably needs its own processor for managing its resources even if it weren't doing routing. Does this then detract from the simplicity of the approach? Perhaps instead the controller and forwarding elements should be in the same POP. But then again, the benefits of the SoftRouter architecture (reliability, scalability, security, flexibility) are also available through the building of larger more scalable routers. Indeed the larger router architecture also overcomes the interconnect problem.

A unique potential benefit is "decoupling the innovation curve of control and forwarding elements". But it will be a challenge to define a protocol that is fluid enough to allow the control and forwarding elements to evolve and improve. It might only work if the interface between the control and forwarding elements is static and the functionality of the forwarding engine evolves slowly. It will take a lot of commitment and hard-work within the IETF to design a protocol that is static enough to build products, and dynamic enough to allow innovation. But if it happens, the idea has some merit. Else, it's an academic exercise.

Finally, decoupling might increase management and configuration challenges for the user. Separating the OS vendor from the computer vendor did not improve reliability, it created more efficient markets and lower procurement costs.

In summary, this paper provides the framework for a discussion worth having, and will motivate further research. Can a SoftRouter-like architecture transform the data networking industry in the same way the SoftSwitch is transforming the voice telecom industry? If you think it will simplify routers, and provide a path to small, simple routers built around a Linux box and a bunch of network interfaces, then think again. It might even create yet another layer of complexity that makes routers bigger and even less reliable than they are today. Either way, it's a good debate for the networking research community to have.

Customizable Routing with Declarative Queries

Boon Thau Loo, Joseph M. Hellerstein, and Ion Stoica

Public Reviewer: Ramesh Govindan

Database techniques have recently been applied to a variety of networking problems. Stream processing techniques can provide rapid and accurate traffic estimation for detecting anomalies. Implementing traditional database operators, perhaps with looser consistency semantics, on a large network can potentially simplify programming Internet and sensor network based applications.

This paper continues the latter thread of research, showing how database techniques can be used to "program" route computation in networks. It asserts that declarative techniques can be used to let network users specify customized routes to destinations. Specifically, Datalog (a Prolog-like language) is shown to be expressive enough to capture routing paradigms beyond pure shortest path routing: source-routing, policy-based routing etc. Furthermore, the paper explains how traditional database query optimization techniques can be applied to improve the efficiency of such a routing system.

The observation that tuple routing for query optimization resembles network routing has been made in passing before, but this paper clearly grounds that intuition by presenting a specific instance of this relationship. In many ways, this paper is a perfect prototype for a good Hotnets paper: it presents a provocative position, makes a significant intellectual leap, and for these reasons is likely to generate significant discussion at the workshop.

Like any good Hotnets paper, it raises more questions than it answers. Isn't this just Active Networks in another guise, the authors' explanations notwithstanding? On a more philosophical level, do we really need this level of flexibility in user control? Can such customizable routing be implemented at Internet-scale? The query optimizations presented in the paper describe, in a rigorous way, performance tweaks well

known to the routing community, but does this framework provide intuition for designing performance optimizations in future routing implementations? As with Active Networks based approaches, this framework gives rather more control to the user than an ISP would like, so how can we ensure the stability and robustness of the overall network? Given these questions, I think of the paper's contribution as being the novelty and audacity of its thesis, not so much the specific scheme the paper presents.

Verifying Global Invariants in Multi-Provider Distributed Systems

Sridhar Machiraju and Randy H. Katz

Public Reviewer: Thomas Anderson

Ask any technical leader at an ISP, and they will tell you that it is crucially important to prevent the unnecessary disclosure of internal information about their network. Proposals that expand the amount of information being disclosed, for example to avoid unintended hotspots after re-routing, are met with deep suspicion. ISPs operate in a competitive environment, they say, and their competitors can use information about network capabilities, workloads, and bottlenecks to poach customers.

This poses an interesting question: what is the minimal amount of data that must be disclosed across ISPs to keep an interdomain routing system working? Today's Internet was not designed with this goal in mind, and if radical information isolation were possible, it might prove irresistible to ISPs. One might believe that the ISPs are only interested in preventing unilateral disclosure, allowing more information to be disclosed than absolutely necessary, as long as there is a level playing field. In practice, though, if ISPs have the capability of reducing their disclosure unilaterally, we might still see a race to the bottom.

This paper doesn't answer the question of the minimal design, but it does provide some techniques that might prove useful to this effort. For example, suppose an ISP wanted to check whether its neighbor could accommodate some proposed change in routes, without incurring congestion. If everyone knew everything, this would be easy – it would be a simple matter of calculating whether the proposed workload sent by the upstream ISP would fit into the available link capacities of the downstream ISP. This paper goes much farther, though, by showing that it is possible to compute the feasibility of a proposed routing change without either the upstream disclosing its workload, or the downstream disclosing its topological constraints, using homomorphic encryption. With homomorphic encryption, arithmetic can be performed on two encrypted values, so that the result, when decrypted, yields the same result as if the plaintext values had been directly manipulated.

This approach is certainly worthy of further research, but there is reason to be skeptical as to whether it will prove useful over the long term. Network measurement researchers have become extremely sophisticated over the past few years at inferring hidden quantities from external measurements. Even if ISPs turn off most or all of the tools these researchers have been using (e.g., traceroute), it remains true that ISPs must necessarily leak information about their internal state to their users, simply from the relative timing data of how long packets take to get from place to place, their jitter patterns showing shared bottlenecks, etc. Faced with an army of determined researchers attempting to reverse engineer the internals of their networks, will ISPs persist in pursuing the fig leaf of confidentiality? Only time will tell.

A Wakeup Call for Internet Monitoring Systems: The Case for Distributed Triggers

Ankur Jain, Joseph M. Hellerstein, Sylvia Ratnasamy, and David Wetherall

Public Reviewer: Dina Katabi

This paper articulates an interesting technical problem for network researchers to work on. Several prior proposals make the case for an Internet monitoring infrastructure, where geographically distributed nodes monitor, store, and track the state of a network. Such systems usually adopt a query-driven approach; they collect data either periodically or whenever a query arrives. In contrast, this paper argues for a data-driven approach to Internet monitoring, i.e., monitors collaborate to ensure certain global invariants are satisfied and trigger an alarm when they are violated. A successful implementation of a distributed trigger mechanism allows timely detection of interesting events and obviates the need to repeatedly polling the monitors.

The notion of triggers is hardly new. It has been extensively researched in the field of databases. However, the widely distributed environments of Internet monitoring and the need to deliver timely information while minimizing communication overhead create new challenges. Triggers are naturally defined in terms of the /aggregate /behavior of a collection of nodes (e.g, rate-limit the total traffic from PlanetLab to any destination IP). But, Individual nodes need to detect and react to changes in aggregate behavior using local observations.

To me, distributed triggers are just one aspect of a more general problem; how do we intelligently manipulating large collections of distributed network data. For instance, how do intrusion detection boxes combine their information to answer more sophisticated questions about the security of the Internet? How do we use BGP updates at different peer routers to describe the state of inter-domain routing and detect anomalies and exploits? How do we combine the views from a large number of performance monitors to describe the performance of the Internet as a whole? Much of Inter-

net data is highly distributed and reflects views of the same system as seen in different parts of the network. As Internet research shifts focus from performance optimization to creating robust and self-managed networks, there is a need to support sophisticated queries over this distributed data to extract meaningful high- level information.

Probably the most serious criticism to this paper is the lack of a convincing technical solution. The paper presents a simple implementation of a trigger that is raised whenever the total traffic generated from PlanetLab nodes to a specific destination exceeds a threshold. This simple example helps the reader understand and appreciate the usefulness, challenges and choices involved, in building distributed triggers. The proposed solutions, the analysis, and the discussion may seem overly simplistic. But, this paper, like many other HotNets papers, focuses on framing a fairly novel interesting problem and should lead to useful debate.

Providing Packet Obituaries

Katerina Argyraki, Petros Maniatis, David Cheriton, and Scott Shenker

Public Reviewer: Alex Snoeren

In this paper, the authors argue for the ability to detect where in the network a packet is dropped, or, in the absence of a packet drop, to generate a return receipt. This facility is ostensibly motivated by the desire of end hosts to intelligently select between multiple routes to a destination, avoiding portions of the network with high loss rates. Currently, determining where in the network forwarding service breaks down is a tricky task: the state of the art involves sending active probes from multiple vantage points and correlating the results. The problem is hard enough for long-lasting path outages; detecting and isolating transient failures (or attacks) in such a manner is nigh on impossible. Of course, in an overlay or edge-routing context, an end user is likely unconcerned with precisely where a path failure occurs; she just wants to be able to avoid it. Hence, the suggestion made here is for each packet to confirm its delivery or report its last successful AS hop. In essence, every packet conducts a passive AS-level 'traceroute.'

As described, the functionality is useful only to the (currently small) set of end hosts prepared to leverage the information to explicitly select ASes to route through, or those with a (likely misguided) interest in debugging the Internet. The authors attempt to bolster their case by pointing out that ISPs might use such a system to enforce SLAs with their peers, but it's unclear how to enforce compliance. Viewed in a larger context, however, this work can be seen as providing additional motivation for the various traceback and network forensic techniques proposed both at HotNets and elsewhere. A large fraction of the criticism of forensic techniques stem from their relative cost: in most cases, audit trails are kept on an on- going basis, yet they are rarely consulted. If Argyraki and her coauthors are to believed, there are far more common uses for at least limited, general-purpose packet auditing facilities.

The obvious question, then, is how much information at what cost. With the recent clamor for "knowledge planes," "Internet weather services," and "underlays," the discussion seems quite timely. The paper strikes one, slightly awkward balance, but it's not entirely clear that the authors are completely wed to their current position. In particular, while the straw-man scheme is fairly well developed, the authors readily admit the existence of several technical challenges facing deployment. The provided bandwidth overhead calculations seem optimistic: realistic values are likely to ride high with per-packet protocol overheads, and I have serious doubts about any reduction in digest size for real networks. Similarly, the 'companion packet' solution to MTU limits and the short-term entry expiration scheme seem problematic.

Ignoring the warts of the particular scheme, however, the authors manage to bring together the needs of a number of research proposals and present them from a different perspective. In my opinion, this proposal presents a refreshing new cut at the type of information that's important about a packet's path. From a performance perspective, an AS-level audit trail seems like it might be the right level of detail. At the same time, however, the costs of the scheme seem to be on-par with traceback schemes that collect far more information; it would have been nice to see some more quantitative evaluation about the cost/benefit equation. I infer (perhaps incorrectly) that the authors feel full traceback is somehow unpalatable, but it would have been nice to seem them argue that point explicitly.

Another interesting aspect of this work is the notion that, in the face of increasing end-host intelligence, ASes may begin to aggressively resist active probing of their internal network structure and performance. The inaccuracy of ICMP-based probing is already well-known, and that would seem to be without an explicit effort on the part of ISPs to deter measurement. If we contemplate an increasingly black-box network, it seems prudent to consider what types of information are essential to efficient operation.