NSDI '11: 8th USENIX Symposium on Networked Systems Design and Implementation

Boston, MA
March 30–April 1, 2011

Opening Remarks and Awards Presentation

Summarized by Rik Farrow (rik@usenix.org)

David Andersen (CMU), co-chair with Sylvia Ratnasamy (Intel Labs, Berkeley), presided over the opening of NSDI. David told us that there were 251 attendees by the time of the opening, three short of a record for NSDI; 157 papers were submitted and 27 accepted. David joked that they used a Bayesian ranking process based on keywords, and that using the word "A" in your title seemed to increase your chances of having your paper accepted. In reality, format checking and Geoff Voelker's Banal paper checker were used in a first pass, and all surviving papers received three initial reviews. By the time of the day-long program committee meeting, there were 56 papers left. In the end, there were no invited talks at NSDI '11, just paper presentation sessions.

David announced the winners of the Best Paper awards: "ServerSwitch: A Programmable and High Performance Platform for Datacenter Networks," Guohan Lu et al. (Microsoft Research Asia), and "Design, Implementation and Evaluation of Congestion Control for Multipath TCP," Damon Wischik et al. (University College London). Patrick Wendell (Princeton University) was awarded a CRA Undergraduate Research Award for outstanding research potential in CS for his work on DONAR, a system for selecting the best online replica, a service he deployed and tested. Patrick also worked at Cloudera in the summer of 2010 and become a committer for the Apache Avro system, the RPC networking layer to be used in Hadoop.

Speed, Speed, and More Speed

Summarized by Colin Scott (cs@cs.washington.edu)

SSLShader: Cheap SSL Acceleration with Commodity Processors

Keon Jang and Sangjin Han, KAIST; Seungyeop Han, University of Washington; Sue Moon and Kyoungsoo Park, KAIST

Despite the importance of secure end-to-end communication, SSL deployment on the Internet is limited due to the heavy computational overhead it places on servers and the high cost of hardware accelerators necessary for achieving decent performance in practice. Working towards a vision of widespread SSL adoption, Keon Jang presented SSLShader, a proxy designed to provide high-performance SSL acceleration in a cost-effective manner.

Similar in vein to PacketShader—previous work from KAIST—SSLShader is based on the insight that commodity GPUs offer massively parallel computation at low cost; SSLShader uses opportunistic offloading to push parallelizable computation to a GPU during periods of high load. Combining batch processing, pipelining, and adapted cryptographic algorithms, SSLShader achieves better performance for RSA, AES, and SHA1 than high-end hardware accelerators. After implementing the aforementioned optimizations, the authors found that their system's performance was bottlenecked only by data copy overhead and inefficiencies in the Linux networking stack.

Grant Tarp (ACL), Paul Newman (VMware), and Jason Li (IA) asked questions related to whether SSLShader would perform similarly in different circumstances: running alternate cipher suites, using cheaper GPU cards, or servicing multiple Web sites. Keon answered affirmatively, speculating in turn that GPUs would offer acceleration for any cipher suites that use parallelizable algorithms; lower-quality GPUs would not strongly affect performance, since data copy rates are a bottleneck; and multiple Web sites could scale if they shared the same keys. Dan Halperin (University of Washington) incisively observed that many RSA optimizations are not used in practice, since they induce greater vulnerability to side-channel attacks. Finally, Aditya Akella (University of Wisconsin—Madison) asked Keon to identify the general lessons from this work that can be extended to other intensive applications. Keon replied that opportunistic offloading and batching are helpful in minimizing latency and maximizing throughput under varying loads.

ServerSwitch: A Programmable and High Performance Platform for Datacenter Networks

Guohan Lu, Chuanxiong Guo, Yulong Li, Zhiqiang Zhou, Tong Yuan, Haitao Wu, Yongqiang Xiong, Rui Gao, and Yongguang Zhang, Microsoft Research Asia

Awarded Best Paper!

Guohan Lu presented ServerSwitch, a fully programmable and high-performance network switch for prototyping datacenter network (DCN) designs. ServerSwitch comes out of four observations: rich programmability is required for the growing number of DCN designs that depart from traditional protocol formats; commodity Ethernet switching chips are becoming increasingly programmable; PCE-I interfaces provide high throughput and low latency communication between CPUs and I/O subsystems; and commodity multicore servers offer heavyweight packet processing capabilities.

By leveraging these observations, ServerSwitch explores the design space of integrating a high-performance packet-forwarding ASIC with a powerful fully programmable server. After implementing a software stack for exposing the configurability of their switching chip, the authors provided proofs-by-example that their platform supports many DCN designs. The examples were chosen to exhibit a wide range of functionality needs, including low-latency control-plane processing, support for arbitrary header formats, and in-network packet processing.

Given that several switch designs also use the PCE-I interface to connect CPUs and programmable switching chips, Sangjin Han (KAIST) wondered how ServerSwitch differs from previous work. Guohan clarified that ServerSwitch evaluated the design on various DCN designs. Anirudh Badam (Princeton) asked whether the PCE-I would be a bottleneck if other devices concurrently performed I/O. Guohan replied that existing motherboards provide many more PCE-I lanes than required for ServerSwitch's needs. Lastly, Chong Kim (MSR) questioned why a specialized switching chip was used rather than a simple TCAM.

TritonSort: A Balanced Large-Scale Sorting System

Alexander Rasmussen, George Porter, and Michael Conley, University of California, San Diego; Harsha V. Madhyastha, University of California, Riverside; Radhika Niranjan Mysore, University of California, San Diego; Alexander Pucher, Vienna University of Technology; Amin Vahdat, University of California, San Diego

Alex Rasmussen presented TritonSort, a record-breaking large-scale sorting system. TritonSort aims to improve the efficiency of increasingly prevalent data-intensive scalable computing (DISC) systems such as MapReduce or Dryad. In

addition to achieving impressive performance, TritonSort offers a number of lessons for balanced system design and scale-out architectures in general.

The authors observed that although DISC systems scale remarkably well, they are often afflicted by poor per-node performance. Consequently, the design of TritonSort was carefully defined in terms of the notion of balanced hardware/software architecture, where all resources are driven to nearly 100% utilization. With I/O-heavy workloads in mind, the team developed a staged, pipeline-oriented dataflow software system running on a static selection of hardware resources. In order to keep disks running at nearly optimal speeds, TritonSort implements an external sort that proceeds in separate routing and sorting phases. The team's delicate engineering ultimately resulted in a factor-of-six improvement in per-node efficiency over the previous sorting record holder.

Alex's talk encouraged a lively question-and-answer session. Siddartha Sen (Princeton) asked how the system balances system parameters both at the onset and in real time. Alex explained that both of these opportunities for improvement are currently being investigated. Anirudh Badam (Princeton) wondered how much DRAM could be decreased without sacrificing balance. Alex noted that memory size determines average write sizes, affecting throughput and variance. Steve Hand (Cambridge) inquired about the energy-efficiency of TritonSort. In response, Alex cited a previous sort competition where TritonSort performed very well in terms of energy efficiency, largely due to its sheer speed. Mike Freedman (Princeton) pointed out that modern disks exhibit high variance in disk throughput and wanted to know the extent of TritonSort reliance on homogeneous disk characteristics. Alex acknowledged that the disks used in TritonSort were fairly homogeneous, but noted that one could profile disks beforehand to handle heterogeneous disk performance.

Arkady Kanevsky (VMware) wondered how the mechanisms in TritonSort, which have been highly optimized for sorting, could be pieced together to play a role in a more general high-performance processing system. Alex noted that a version of MapReduce based on TritonSort has already been built. He also identified the logical disk distributor and the buffer manager as components that are widely applicable. Finally, Matai Zaharia (Berkeley) asked whether performance would be impacted if the system allowed disks to be written to and read from simultaneously. Although Alex couldn't provide any measurement results, he conjectured that the predictability of disk seeks would be negatively impacted by loosening the constraint.

Performance Diagnosis

Summarized by Andrew Ferguson (adf@cs.brown.edu)

Diagnosing Performance Changes by Comparing Request Flows

Raja R. Sambasivan, Carnegie Mellon University; Alice X. Zheng, Microsoft Research; Michael De Rosa, Google; Elie Krevat, Spencer Whitman, Michael Stroucken, William Wang, Lianghong Xu, and Gregory R. Ganger, Carnegie Mellon University

Raja Sambasivan presented Spectroscope, a request flow-based debugging tool for distributed systems. Spectroscope determines the cause of performance degradation by identifying changes in the timing and structure of request flows. The tool uses several heuristics to pinpoint persistent mutations, link them to their precursors, and rank them according to their overall performance impact.

Spectroscope begins with request-flow graphs built using end-to-end tracing tools such as Magpie (OSDI '04), X-Trace (NSDI '07), or Google's Dapper. The flow graphs are then binned into categories according to path structure and expected timing. This categorization step seeks to minimize intra-category variance and any difficulties in this process help to identify aspects of the distributed system which increase the variance of user-visible response time. To determine which categories actually contain mutations, Spectroscope employs hypothesis testing to compare each category's distribution of response times. After two distributions with a mutation are identified, Spectroscope iterates over the edges in the request flow to localize the mutation to a particular RPC or function call.

After the request flow mutations are identified, Spectroscope presents a user interface which assists the developer in investigating the problems. The interface ranks the mutations with a simple "greatest impact" heuristic: the number of requests affected by the mutation, multiplied by the slowdown in response time. The goal of this ranking is to direct the programmer's attention to the underlying cause, as there may appear to be more than one problem in the system, or one problem may yield many mutations.

Sambasivan presented the results of two evaluation case studies. In the first study, Spectroscope helped resolve four previously unsolved performance problems in a distributed file system, Ursa Minor (FAST '05). In the second, the tool was used to diagnose the cause of different benchmark results recorded by a new Google distributed system when tested in multiple datacenters. Spectroscope was able to identify the slow datacenter's communal BigTable instance as the culprit, acquitting the developers of the new system.

Rodrigo Fonseca (Brown University, session chair) asked if it is possible to identify problems using the system, rather than simply searching for causes of known problems. Raja answered that problems are identified by comparing recent paths against historical data, comparing the performance with strict SLAs, and relying on gut intuition.

Profiling Network Performance for Multi-tier Datacenter Applications

Minlan Yu, Princeton University; Albert Greenberg and Dave Maltz, Microsoft; Jennifer Rexford, Princeton University; Lihua Yuan, Srikanth Kandula, and Changhoon Kim, Microsoft

Today's cloud applications make extensive use of the network. With the growth of abstractions for network programming, not all application developers fully understand the network stack, particularly more complicated topics such as congestion control, delayed ACKs, and the need for Nagle's algorithm. This lack of understanding can lead to performance problems, which can be diagnosed after the fact by extensive logging. However, application logs are too specific to identify general network problems, switch-based logging is generally too coarse-grained, and network-wide packet sniffers can be very expensive.

Minlan Yu presented a system called SNAP, a Scalable Network-Application Profiler, which runs continuously on the end hosts in a datacenter. Yu argued that TCP stacks already gather many aspects of network-application interactions and is thus an appropriate layer at which to do diagnostic logging. For example, the flow-control algorithms rely upon how much data applications want to read and write, and the congestion-control algorithms build up on measurements of network delay and congestion. SNAP works by gathering the TCP-level statistics described by RFC 4898 and collected by many operating systems, such as number of bytes in the send buffer, congestion window size, number of dropped packets, etc. Yu noted that the CPU overhead of collecting these statistics for 1,000 connections every 500 ms was only 5%. Each end-host then transmits the collected statistics to a central server which combines the statistics with information about the network topology and the mapping of connections to applications to diagnose problems such as excessive retransmissions or undersized send buffers.

Yu presented results for running SNAP for a week in a datacenter with 8,000 machines running 700 applications. SNAP was able to identify TCP-level problems with 159 applications, including six which suffered from TCP incast. In one particularly egregious case, a record-processing application was able to process 1000 records per second if there were an even number of packets per record, but only five records per second if there was an odd number of packets per record. By using SNAP, the datacenter operators were able to tune the performance of the TCP stack.

During the question period, Yu was asked about SNAP's correlation model. SNAP assumes linear correlation across connections or a set of connections, which might be oversimplified. She was also asked about SNAP's application to non-TCP protocols, and responded that it could be extended to other protocols, but TCP was a good candidate because the gathering of statistics was already well-defined. Elias Weingärtner (RWTH Aachen University) asked if they could map TCP data to different applications running simultaneously on the same host. Yu said that they map data to each connection, so they can link it to the correct application.

Nothing but Net

Summarized by Brent Stephens (brents@rice.edu)

Efficiently Measuring Bandwidth at All Time Scales

Frank Uyeda, University of California, San Diego; Luca Foschini, University of California, Santa Barbara; Fred Baker, Cisco; Subhash Suri, University of California, Santa Barbara; George Varghese, University of California, San Diego

Frank Uyeda began this talk by describing the current state of datacenters. Big datacenters have thousands of hosts, a multitude of applications, and high-speed networks. Debugging and tuning network performance in a datacenter requires fine-grained information on network latency and bandwidth. This work focuses on how to identify short-lived bursts in a datacenter with low monitoring overhead.

Bursts can be identified by sampling utilization at fixed intervals. However, correctly determining the time scale is difficult, and a full network trace is needed in order to resample utilization at distinct time scales. The existing tools for sampling are not well suited for the task. Both tcpdump and NetFlow provide too much data, sampled NetFlow has the potential for false positives, and SNMP counters provide insufficient information and time resolution.

This work presents two techniques for supporting bandwidth reports at any time scale that can be generated after the fact without requiring a full packet trace: Exponential Bucketing (EXPB) and Dynamic Bucket Merge (DBM). These techniques scale down to microseconds, require orders of magnitude less storage than a full packet trace, and provide complex stats and visualizations. These techniques involve sampling at all end hosts in the network at the smallest time scale. EXPB allows for querying the max and standard deviation of bytes per interval. This is accomplished by keeping packet statistics in buckets growing in time by powers of two. The query is served using the closest computed time scale.

A drawback of EXPB is that it does not retain time domain information, which is required for median and percentile information. DBM performs time-series summarization, which supports queries regarding median, percentiles, and visualization of bandwidth over time. When DBM has more samples than buckets, buckets are merged together. Buckets may be merged by smallest byte count (DBM-mm) or smallest variance difference (DBM-mv) or by minimizing the difference between the highest and lowest samples (DBM-mr). DBM has a tunable tradeoff between accuracy and storage.

EXPB and DBM were evaluated by replaying 500 GB of traffic traces obtained from the TritonSort project. The base sampling time was 100 microseconds, and queries were performed at 52 time scales. Both EXPB and DBM require orders of magnitude less memory than storing packet traces, which use six bytes per packet. DBM-mr was found to be the most accurate variant of DBM and was best for time scales greater than 2 ms, with EXPB being better for time scales less than 2 ms. Both techniques can be combined for the best results. Future work includes performance culprit identification.

Wyatt Lloyd from Princeton asked if it is possible to collect stats in an event-driven manner instead of time-driven. Uyeda replied that he is not sure about the best sampling interval. Wyatt then asked if there are basic principles for other basic monitoring tasks and if Uyeda will provide a software package. Uyeda replied that this work is not fundamental to bandwidth. Any time series of data can use these techniques. Other folks have been working with latency, and this approach is complementary. As far as providing a package, they'd love to release that but are not on track to do that right now. Cheng Huang from MSR asked for any thoughts on what stats they can't capture with this technique; he feels that some are missing. Uyeda replied that there are some things you can't keep with EXPB, but DBM can be used to sample many. The question is how much error is induced.

ETTM: A Scalable Fault Tolerant Network Manager

Colin Dixon, Hardeep Uppal, Vjekoslav Brajkovic, Dane Brandon, Thomas Anderson, and Arvind Krishnamurthy, University of Washington

Colin Dixon explained that before he was a graduate student, he was an enterprise network administrator; the things that he wants from a network include a NAT, firewall, traffic prioritization, Web cache, VPN, and IDS/DPI system. These things require several proprietary middleboxes, but this solution is inconsistent and not scalable. For example, intrusion detection systems are deployed at the edge of the network. This makes it difficult to catch internal threats that do not cross the network edge. Increasing the deployment can fix this but requires coordination between the intrusion detection systems. Their work proposes a different approach: the end to the middle (ETTM). The idea is that we do not need physically centralized hardware to make logically centralized decisions. Instead, we can implement virtual middleboxes. Intrusion detection on ETTM gives pervasive monitoring that is not possible at the network edge.

ETTM makes many assumptions. ETTM is meant to be deployed on enterprise networks under a single administrative domain. Hosts must have trusted computing software, be multicore, and be virtualized. The switches in the network are required to provide ACL, such as can be provided by OpenFlow or 802.1X.

The goal of ETTM is to be able to extend network functionality in a pervasive, consistent, fault-tolerant way that automatically scales with network demand. The challenges with accomplishing this goal are being able to trust commodity PCs, performing consensus on commodity PCs, and allowing administrators to deploy new features. The PCs can be trusted by adding a trusted platform module to the hypervisor, extending 802.1X to use the TPM instead of keys, and creating an attested execution environment (AEE) to run the network tasks. Consistent decisions are accomplished with Paxos, and the AEE is exposed to allow developers to deploy services.

A micro-virtual router that invokes filters on packets is run in the AEE. The virtual router has two implementations: one in Open vSwitch that can perform line-speed packet header operations, and the other in iptables, which is slow but can perform general operations.

ETTM was used to implement a NAT, DPI/IDS, Web cache, traffic prioritization, worm scan detection, and firewall. Most features could be implemented in only 100s of lines of code. Consensus with Paxos can be performed with a group size of 20 in under 1 ms on a wired LAN, and 1700–8000 updates can occur per second. The NAT application is able to saturate a gigabit Ethernet link.

How does ETTM affect computing power? Enterprise networks have low bandwidth requirements, so ETTM should scale down well. Brad Karp from University College London asked about privacy and whether hooking an Ethernet hub up to the network allows you to snoop packets. Dixon replied that you would want something like WPA2 encryption for wired networks. Ashok Anand from the University of Wisconsin—Madison asked if ETTM could be simplified if the network was programmed instead of the end-hosts. Dixon replied that there is a whole bunch of work that does this. This work mostly focuses on path selection rather than the complete suite of things you want on your network. Everything they've done is related, and he intentionally stayed away from programming the network. How does mobility

change things? If all the hosts on the network are mobile, things change, but if it's a small set, then you could just tie the mobile hosts to the local AEE. Wojciech Golab from HP Labs asked why unsafe progress is allowed in Paxos in the case of a large-scale failure. Dixon replied that in general operation of the network he expects that this would never occur, and unsafe progress is allowed because two safe lists are easier to merge than a jumble of information.

Design, Implementation and Evaluation of Congestion Control for Multipath TCP

Damon Wischik, Costin Raiciu, Adam Greenhalgh, and Mark Handley, University College London

Awarded Best Paper!

Damon Wischik started his talk with some ancient history. Circuit switching used dedicated circuits for each flow, but this wasted excess capacity. Packet switching was introduced to allowed flows to utilize excess capacity to adapt to traffic surges. Multipath is packet switching 2.0. Packet switching allows for flows to utilize excess capacity on a single link, and multipath allows for a single flow to utilize excess capacity across multiple links. In circuit switching, flows cannot harm each other. To remove a circuit, you need a transport control protocol. If you want to aggregate links, then you need a new transport protocol to share the aggregate. MPTCP is a control plane for a collection of links that makes them behave like a single large pool of capacity. MPTCP is run at both the sender and receiver as a replacement for TCP. In this work, they formulated design goals and test scenarios and implemented a MPTCP that achieves these goals. There are further questions to be addressed with multipath TCP. How much of the Internet can be pooled, and what does this mean for network operators? How should we fit multipath congestion control to CompoundTCP or CubicTCP?

There are five other design goals for this work. Number 1 is that MPTCP should be fair to regular TCP at shared bottlenecks, and design goal 2 is that MPCP should use efficient paths. If each MPTCP sent its traffic on its least congested paths, this would be the most efficient allocation with respect to paths. Design goal 3 is that the throughput of MPTCP should be fair compared to TCP. In the situation where a user has a 3G network and WiFi access, design goal 2 could be satisfied by only sending on the 3G network, but 3G has a high RTT, which would give lower throughput. Design goal 3a is that MPTCP users should get at least as much throughput as a single-path TCP would on the best of the available paths. Design goal 3b is that MPTCP should take no more capacity on any link than a single TCP flow. At this point, design goal 1 has been subsumed by goals 2 and 3. Design goal 4 is to adapt quickly, and design goal 5 is not to oscillate.

MPTCP is based on TCP, so it is important to understand the basics of TCP. TCP maintains a congestion window which additively increases on every packet received and multiplicatively decreases on loss. MPTCP maintains a congestion window w for each path. For route r, w_r increases for each ACK, and w_r decreases for each drop by $w_r/2$. Specifically, on every ACK on subflow r, for each subset $S \subseteq R$ that includes path r compute:

$$\frac{\max_{s \in S} w_s/RTT_s^2}{(\sum_{s \in S} w_s/RTT_s)^2}$$

then find the minimum over all such S, and increase w_r by that much.

Michael Freedman from Princeton asked about the number of RTTs that short-lived flows take to converge compared to TCP. Wischik replied that MPTCP will converge very quickly to the correct bandwidth and use more RTTs to balance. Guohui Wang from Rice University asked how the end host knows how many paths there are in the network. Wischik replied that MPTCP assumes a different address for each path, but if you have ECMP in the network, you can open up multiple flows on different ports and hope to get different paths. The last question was what multiplexing 3G and WiFi in a very mobile network would look like. Wischik replied that in the simple case with 3G plus WiFi and WiFi disappears, it ought to converge in a few RTTs.

Data-Intensive Computing

Summarized by Wolfgang Richter (wolf@cs.cmu.edu)

Ciel: A Universal Execution Engine for Distributed Data-Flow Computing

Derek G. Murray, Malte Schwarzkopf, Christopher Smowton, Steven Smith, Anil Madhavapeddy, and Steven Hand, University of Cambridge Computer Laboratory

The goal of CIEL, a distributed execution engine, is to make distributed programming easier. CIEL's advantage over MapReduce and Dryad is that it can express unbounded iterative algorithms within the framework rather than breaking them up into multiple jobs, making it Turing-complete. CIEL is more efficient at representing such algorithms than MapReduce and Dryad—representing algorithms as a single job cuts down on per-job overhead, and the scheduler can use information from previous iterations to improve the schedule in later iterations.

CIEL creates Turing-complete expressiveness by providing a universal execution model which allows dynamic changes to the task graph. CIEL expresses a strict superset of the possible graphs and algorithms possible in MapReduce and Dryad. CIEL maintains tables of finished results ("complete" parts) and results that are unfinished ("future" parts). Tasks are able to spawn subtasks and delegate outputs. This is similar to how modern processor architecture is designed: pipelining and aliasing as much as possible.

CIEL still provides a single master model with multiple worker nodes; however, it has built-in support to have multiple hot-swappable masters running at the same time. CIEL also provides the "Skywriting" scripting language for rapidly producing CIEL distributed programs; the standard library includes an implementation of MapReduce. Skywriting is an interpreted, dynamically typed C-like language with runtime state stored as CIEL objects (in the CIEL tables). All of CIEL, including Skywriting, is written in 9,000 lines of Python code and is available online: http://www.cl.cam.ac.uk/netos/ciel/. There is also a *;login:* article about CIEL in the April 2011 issue.

Anirudh Badam (Princeton) opened the discussion by asking about deadlocks. Derek said that they used `select` rather than deadlock control. Someone else asked Derek to comment on optimizations that their model supports. Derek said that they took a different approach. Hadoop cannot support directed graphs. Their streaming works at low bandwidth but cannot support high flows. Something they did was take advantage of local data. They also support speculative execution. Theo Benson (Wisconsin—Madison) asked if their language makes it easier to deal with outliers, and Derek responded that their long-term strategy is to try speculative execution and shoot down the slower versions. Currently, they don't have any straggler detection, as they had no stragglers. Dave Maltz (Microsoft Research) asked if extra context switching resulted in having to move more data, and Derek said that they hadn't run into this in their experiments. In a k-means problem, that cost would be relatively minor.

A Semantic Framework for Data Analysis in Networked Systems

Arun Viswanathan, University of Southern California Information Sciences Institute; Alefiya Hussain, University of Southern California Information Sciences Institute and Sparta Inc.; Jelena Mirkovic, University of Southern California Information Sciences Institute; Stephen Schwab, Sparta Inc.; John Wroclawski, University of Southern California Information Sciences Institute

Current approaches to diagnosing networking problems in large systems require large amounts of domain-specific knowledge. For example, experts analyze packet dumps, race conditions in distributed software, etc.—there are too many random patterns for simple regular expressions to highlight issues. The semantic approach advocates creating a knowledge base consisting of abstract models of understanding the system extracted from experts which allow queries from users for problem diagnosis. This is a logic-based approach.

Arun Viswanathan described the key differences from prior logic-based approaches as the composibility of abstractions for expressiveness and expressive relationships for networks. They define "behaviors" as a sequence or group of one or more related facts which can relate to other behaviors. Behaviors capture the semantics of a system and are thus closer to users' level of understanding of the system. The key to capturing interesting behaviors lies in the variety of relationships that can be expressed. There are four broad categories of relationships relevant to networked systems that the modeling language can express: (1) temporal—causal etc., (2) concurrent—overlaps, (3) logical—combinations, and (4) dependency—between data attributes. Thus, encoding a model consists of (1) capturing simple behaviors, (2) relating simple behaviors to capture complexity, and (3) defining a behavioral model from which answers—facts satisfying the model—are obtained. More information on the semantic analysis framework can be found at http://thirdeye.isi.deterlab.net.

Dave Maltz pointed out that they had taken a top-down approach, and wondered if there was a bottom-up way of bridging prior approaches and theirs. Arun answered that they wanted to make sense of lots of data, and by encoding the knowledge they allow data mining to work. They can extract models that are much more meaningful instead of packets and values. Jason Li (Intelligent Automation, Inc.) asked what Arun meant by composibility here; does the model come from a human expert? Arun replied that composibility is completely manual for now, and that models do come from human experts.

Paxos Replicated State Machines as the Basis of a High-Performance Data Store

William J. Bolosky, Microsoft Research; Dexter Bradshaw, Randolph B. Haagens, Norbert P. Kusters, and Peng Li, Microsoft

Bill Bolosky explained that they wanted to build a high-performance data store from commodity parts. The surprising finding was that compromising on consistency or any of the guarantees of Paxos was not needed for efficiency. The building block of this distributed system is Paxos replicated state machines. Replicated state machines copy state and inputs across a distributed system to maintain consistency at each node. The consistency guarantee they are able to provide is that a reader sees the *latest* commit from *any* client.

There are three key observations the authors made while designing and implementing their data store. The first key observation was that they need not apply operations in order/one-at-a-time to their replicated state machines. This is similar to how an out-of-order processor works. The second key observation was that using a writeback cache prevents slow synchronous writes to disk—again, similar to innovations in processor design. The third key observation was that batching significantly increases throughput in their system, along with not sending writes to reading disks and vice-versa.

They implemented a distributed file system represented as the Gaios virtual disk in a Windows client. NTFS is layered on top of this virtual disk. Thus, access is completely transparent to the end user.

Steven Hand (Cambridge) asked if they did measurements in the presence of failure, and Bill responded that they killed a process on one of the nodes and got a glitch while the leader fails-over. This results in a bit of slowdown, but scaling for writes is pretty flat. Colin Scott (U. of Washington) asked about Paxos group management, and Bill said they referenced prior work. Someone asked if there was an open source version and Bill said no. Another person asked about logging operations in memory, and Bill said that you can write logs to disk (slow), write to flash, or just have a system that forgets when the power goes off. Failing in Paxos means forgetting your state. But if you are willing to count a crashed system as a Paxos failure, it would probably recover in 400 μs, and could be faster if the network was better tuned. Dave Maltz asked if you could have very large Paxos groups, and Bill said you could, although they would eat bandwidth.

Security and Privacy

Summarized by Hendrik vom Lehn (vomlehn@cs.rwth-aachen.de)

Bootstrapping Accountability in the Internet We Have

Ang Li, Xin Liu, and Xiaowei Yang, Duke University

Xiaowei Yang started by giving examples of how the Internet is vulnerable to traffic hijacking over IP prefixes or denial of service (DoS) attacks. These attacks are often disruptive and can be costly. Yang wondered whether it is possible to secure the Internet with low-cost and adoptable changes. It is important that such changes be gradually deployable and benefit early adopters. There are already several proposals to secure routing and mitigate DoS attacks. These approaches, however, miss a lightweight bootstrapping mechanism that securely binds an IP prefix to its owner's keys.

IP prefixes and AS numbers, the currently used identifiers, can both be spoofed. Yang wants to counteract this problem with a system called IP made accountable (IPA). The approach used in IPA is to derive the AS number of a network from the hash value of the corresponding public key and use DNSSEC and BGP as a public key infrastructure to bind an IP prefix to its owner's keys. By utilizing DNSSEC and BGP as public key infrastructure, no additional new infrastructure is needed. Reverse DNSSEC records are used as IP prefix ownership certificates. Additional advantages of this approach are that DNS and IP addresses have the same root of trust and that DNS is a very scalable system and can be used to support certification revocations and key rollovers.

In order to speed up the certificate validation of incoming routing announcements, Yang proposes including the corresponding certificates in an optional BGP field. As this adds overhead, IPA uses a caching mechanism such that a BGP router only sends the certificates it has not sent to a peer before in its BGP messages. The resulting system can be used for both origin authentication and path authentication. For functionality such as certificate revocation and key management, Yang mentioned several solutions and suggested that the audience refer to the paper for more details.

The goals of IPA were to make it a lightweight, secure, and adoptable system. Out of the evaluation results included in their paper, Yang showed that IPA introduces only moderate BGP overhead and can be easily deployed.

Rik Farrow raised the point that one of the difficulties with approaches like S-BGP is that signature checking causes too much overhead for the routers. Yang responded that this was indeed a problem for older routers, but that modern routers should be able to handle the additional load. Dave Oran of Cisco asked whether it would not be easier if RPKI databases were copied and checked in a back-end server once a BGP update arrives. Yang answered that this potentially introduces a dependency loop between routing and certificate distribution: before a router can reach the back-end server to obtain necessary certificates to secure routing, the route to the server must be secured first. In contrast, they try to bootstrap their own certificate distribution infrastructure without relying on other infrastructures.

Privad: Practical Privacy in Online Advertising

Saikat Guha, Microsoft Research India; Bin Cheng and Paul Francis, MPI-SWS

Saikat Guha began by discussing the drawbacks of current advertisement systems for the Web. Instead of trying to improve the quality of advertisements, emphasis is placed on producing large quantities of ads, and this slows systems. To get to know more about users, advertisement brokers set cookies on the users' computers, which allows them to obtain the user's browsing history of the Web sites that contain their advertisements. Thus, targeted ads have two sides: they make money, but they invade privacy as well.

In response, the authors built a system, Privad, that tackles the privacy problem, but nevertheless supports targeted advertisements. In order to be successful, the new system should support as much of today's model as possible, should just be private enough, should not trade off privacy for low-quality ads, and should be scalable. The underlying idea of the system they have built is that the best place for privacy is one's own computer. The system therefore makes use of a client software that runs in a sandboxed environment on the user's computer. It profiles the user, serves advertisements, and sends reports.

To preserve the user's privacy, the actual advertisements are downloaded to the user's computer in larger amounts than required. Furthermore, reports are not sent directly to a broker. Instead, a third party (called a dealer) serves as an anonymization proxy. Since the reports are encrypted with the broker's key, the dealer does not know the specific advertisement the client clicked on. The broker, on the other hand, does not know from which user the reports are. Through a report ID which the dealer inserts into each report, the broker can notify the dealer to block a user in case of clickfraud.

Someone asked about the security of information on the user's computer. Guha explained that they do not attempt to solve this problem, but that the same problem exists for all kinds of other things (e.g., cached credit card information) as well. Aleksandar Kuzmanovic (Northwestern) asked about the compatibility with current models. Guha said their system supports cost per click, cost per impression, and cost per action. What are the incentives to install this kind of software if one does not want advertising? Guha said that this kind of software is definitely not for people who use Adblock, but that there are actually more people who install useless toolbars than who install Adblock. Kuzmanovic pointed out that people don't like advertising, and Guha replied that 12% of people use some form of ad blocking, while 21% will install anything. Wolf Richter of CMU wondered why he should feel safe installing massive Microsoft spyware on his computers. Guha replied that this is a blackbox broker that collects and holds privacy information, and it would be provided by someone else, like your AV provider. Guha suggested that people likely trusted their AV provider.

Bazaar: Strengthening User Reputations in Online Marketplaces

Ansley Post, MPI-SWS and Rice University; Vijit Shah and Alan Mislove, Northeastern University

Fraud is a big problem for many Web sites that connect buyers and sellers. These Web sites are based on identities and reputations that show up in a feedback profile. A problem, however, is that accounts are free and thus allow fraud through newly created identities. Each of the otherwise very successful Web sites has its own solution to prevent such types of fraud. These solutions range from account creation being made difficult, to in-person transactions and insurance services. All of these solutions, however, also come with drawbacks such as limited applicability or additional costs.

Alan Mislove presented a new approach, Bazaar, to tackle this problem. Bazaar works in conjunction with existing marketplace systems and is based on the insight that successful transactions represent a shared risk and thereby provide a bound on possible fraud. The core of Bazaar is a risk network in which identities are nodes and links represent the amount of successful transactions. The max-flow between buyer and seller is used as a metric to evaluate the risk of new transactions.

Mislove explained in more detail how the links are altered and why the max-flow is robust against known kinds of fraud. He then discussed how they dealt with certain challenges such as the delay of feedback, the inclusion of new users, and a scalable max-flow implementation.

To evaluate the system, they crawled the feedback on 8 million transactions on ebay.co.uk and used Bazaar to calculate a risk network based on the acquired data. Mislove explained that in this experiment it took only 1 to 6 seconds to check for fraud and that they obtained a false positive rate of 1% to 5% for the transaction history of only 90 days. He then concluded the talk by saying that, for this data, the use of Bazaar could have prevented fraud for an amount totaling $269,000.

What if the links between buyer and seller are not sufficient for a transaction? That kind of problem did not show up in the eBay graph, but one possible solution would be the use of an escrow service. Would this kind of system be an incentive to use compromised accounts to perform fraudulent transactions? That was possible, but Bazaar puts a bound on the amount of money that can be used.

Energy and Storage

Summarized by Brent Stephens (brents@rice.edu)

Dewdrop: An Energy-Aware Runtime for Computational RFID

Michael Buettner, University of Washington; Benjamin Greenstein, Intel Labs Seattle; David Wetherall, University of Washington and Intel Labs Seattle

Michael Buettner thinks that activity recognition for elder care is important. The goal is to track what and how objects are used to determine activities. One existing solution is to use cameras, which has privacy drawbacks. Another solution

is to use "Mote"-based sensor networks, which detect object use from accelerometers, but battery life, size, and cost limit the deployment of these sensor networks. Buettner's proposal is to use computational RFID, where an RFID reader sends power and commands to CRFID tags. Battery-free CRFID tags use radio signals to compute, sense, and communicate. Dewdrop is a runtime for CRFIDs. This enables CRFID tags to use scarce energy to run programs, which have varied and non-deterministic energy needs and whose input power can vary by two orders of magnitude.

Dewdrop is implemented on the Intel WISP. WISP has a 4m range, a 10uF capacitor, and a 3D accelerometer. WISP has been used for such applications as sleep monitoring, neural monitoring, and cold-chain undersea neutrino detection, but all of these applications evaluate within less than one meter from the reader, where energy is plentiful. The challenges of running on CRFID tags are that they have a minuscule energy store, differing energy needs, and inherent inefficiencies, and they harvest energy even while executing programs. The WISPs take hundreds of milliseconds to charge, and only tens of milliseconds to discharge. Executing too early causes the tag to hit a black-out threshold where all state is lost. Because energy on the tags is stored in capacitors, charging the capacitor is non-linear; the more energy stored, the more time it takes to store additional energy. Dewdrop needs to store enough energy to compute, but so much as to waste time.

Dewdrop adaptively finds the wake-up voltage that maximizes the execution rate for the program and the RF environment, under the constraint that the runtime must be simple because cycles are tight on the tag and there are no floating-point units, etc. Dewdrop uses the heuristic that total waste is minimized when the wasted time from failures and overcharging is equal. In practice, this works well. To save energy, the implementation of Dewdrop uses an exponentially adapted low power wake-up to periodically poll the capacitor voltage, and Dewdrop uses a low power voltage sampling technique that reduces the energy from sampling the voltage by a factor of 4. Dewdrop matches the performance of the existing hardware mechanism for light tasks, and it doubles the range for heavy tasks. Dewdrop finds a wake-up voltage within 0.1V of optimal and achieves greater than 90% of the maximum rate for all distances. Technology trends will increase the range and performance of CRFIDs. The WISP platform and tools are available to the community in the form of open source hardware and software.

How does Dewdrop handle harvesting dynamics when the tags are mobile? It depends on how fast you are moving. If you aren't moving that fast, you'll be close to the operating point, but Dewdrop is intended for static or slow-moving objects. Hari Balakrishnan from MIT asked if it makes sense to partition tasks that use lots of computation across many tags. Buettner replied that most of the energy cost on the current WISP is communication, which is done in software, and you can compute for a very long time for the cost of a single communication. He said that they are looking into this problem with WISP 5.0. Why, for a given task, isn't the rate of charging and energy consumption estimated online? It's really hard to estimate the rate at which you get energy because it varies quickly, and if you are sufficiently close to the reader, you don't need to store any energy. You also can't profile offline because running around other WISPs changes the charging profile.

SSDAlloc: Hybrid SSD/RAM Memory Management Made Easy

Anirudh Badam and Vivek S. Pai, Princeton University

Anirudh Badam explained that memory in network systems is used both as a cache to reduce disk pressure with tools like memcache and as an index for things like proxy caches, WAN accelerators, and in-line data-deduplicators to reduce disk pressure. However, memory density is not scaling well. The cost of DRAM only scales linearly up to 64 gigabytes, and disk capacity is scaling, but disk speed is still around 200 seeks per disk per second. One solution to this problem is high speed disk arrays, but these are expensive and use more rack space. Their proposal is to use flash to overcome DRAM limits. Flash devices are fast for random reads, achieving one million IOPS per drive, but writes are slow and destroy the device. Because flash has low latency, it is closer to main memory than it is to disks.

Current transparent tiering relies on the OS page daemon to transparently move pages to swap. However, even a flash-aware pager only delivers 30% of the SSD's raw performance. This also has the drawback that both writes and frees are writes, which has a negative impact on performance on SSDs. Non-transparent tiering is possible, but this requires intrusive modifications to the application. The goals of SSDAlloc are to work with unmodified applications, to use DRAM as an object cache, and to use the SSD wisely by having the object store be log-structured.

SSDAlloc is able to act as an object cache rather than a page cache by storing one object per page (OPP) in virtual memory. Physical memory is split into a page buffer so as to not be wasteful. A small set of the pages contain a single object, and the rest of the memory is a compact object cache where multiple objects are packed into a single page. Pages are materialized on demand from the RAM object cache and the SSD. The SSD is used as a log structured object store. For comparison against SSDAlloc-OPP, SSDAlloc also has the

option to implement a page cache, called SSDAlloc-MP. SSD maintenance is accomplished through object tables, which are similar to page tables. A garbage collector and log-writer copies and compacts objects in LRU order, using the object table to determine liveness. Objects that are written elsewhere and OPP objects that have been freed are treated as garbage.

SSDAlloc is implemented in 11,000 lines of C++ using the mprotect, mmap, and madvise system calls. The overhead of the SSDAlloc library is around 0.833 microseconds. For comparison, the latency of NAND is 30–50 microseconds. SSDAlloc is able to reach one million IOPS. Experiments were performed to evaluate SSDAlloc. SSDAlloc-OPP, SSDAlloc-MP, and SSD as swap were compared. SSDAlloc is able to write up to 32 times less data to the SSD. The performance of SSDAlloc-OPP is best at small object sizes, and SSDAlloc-OPP achieves a 2–4x improvement over SSDAlloc-MP. SSDAlloc is able to deliver 90% of the raw SSD random read performance.

In the Q&A, Bill Bolosky claimed that sweating about the log structured object store is not necessary, because the FTL on the SSD happens underneath you. Bolosky also asked about the performance of SSDAlloc for applications that actually fit into memory. Badam replied that you don't have to translate all of your malloc calls into SSDAlloc; you only should translate memory-intensive calls. Indices are stored with malloc; objects are stored with SSDAlloc. Aaron Gember from University of Wisconsin—Madison asked if performance is killed if the application has very small objects and the object table is larger than the size of memory. Badam replied that the object store is not kept in DRAM, and the object tables are implemented the same way as page tables. They have made optimizations to fit the object table into DRAM.

Debugging and Correctness

Summarized by Kristin Stephens (ksteph@cs.berkeley.edu)

Model Checking a Networked System Without the Network

Rachid Guerraoui and Maysam Yabandeh, EPFL

Maysam Yabandeh explained that there are two steps to dealing with bugs in a distributed system: testing and debugging. The system he presented focused on testing. Testing is done by exploring states, performing some transitions, and verifying user-specified invariants on the explored states—in other words, model checking (MC). The classical approach to model checking is to keep track of global states, which is a combination of system state and network state. Keeping things as only global state, however, means that global state changes following a change into any of the involved local states as well as the network state. This exacerbates the exponential explosion problem in number of possible states.

To alleviate the exponential explosion Maysam presented his idea of Local Model Checking (LMC). The global state is broken up into local states for each node with a shared network state. However, testing all possible combinations of local states and with all the different network messages could lead to invalid system states. Therefore, a soundness verification technique is used before reporting something as a bug. When a bug is found it is checked for soundness before being reported. Soundness verification looks at the partial order of predecessor states and transitions and sees if a total order can be found. If a total order can be found it could happen in a real run of the system. The evaluation of the system presented was testing a Paxos implementation with three nodes and one proposal. They were able to both rediscover previously known bugs and to find a new bug.

Sylvia Ratnasamy from Intel Labs Berkeley asked if Maysam had a sense of the general applicability of LMC. He responded that Paxos is known to be very complicated. The results from testing are a sign that it'll give good performance on other systems. However, the results do not necessarily mean that LMC's impact on other protocols would be as profound as its impact on Paxos.

Fate and Destini: A Framework for Cloud Recovery Testing

Haryadi S. Gunawi, University of California, Berkeley; Thanh Do, University of Wisconsin, Madison; Pallavi Joshi, Peter Alvaro, and Joseph M. Hellerstein, University of California, Berkeley; Andrea C. Arpaci-Dusseau and Remzi H. Arpaci-Dusseau, University of Wisconsin, Madison; Koushik Sen, University of California, Berkeley; Dhruba Borthakur, Facebook

We have entered the cloud era, where the use of thousands of commodity machines means that rare hardware failures become frequent. Or, as the speaker, Haryadi Gunawi, put it, our forecast is "cloudy with a chance of failure." With failures becoming common, failure recovery becomes important. However, this is hard to get right, because testing is not advanced enough and recovery is often underspecified. Haryadi said we need two advances: a way to exercise complex multiple, diverse failures and a way to write recovery specifications. He presented FATE (Failure Testing Service) and DESTINI (Declarative Testing Specifications) for each advance, respectively.

FATE exercises multiple diverse failures. By doing so, FATE faces the challenge of a combinatorial explosion of multiple failures; with a brute-force approach, tens of thousands of multiple failures require over 80 hours to exercise. To quickly explore failures and find bugs, Haryadi presented two prun-

ing strategies that improve testing time by an order of magnitude. DESTINI facilitates recovery specifications, which verify that the system under test is correct under failures. To make the specifications "developer friendly" (i.e., clear and concise), they experimented with Datalog, a declarative relational logic language. DESTINI only interposes I/O calls from which expectations of correct behavior and the actual behaviors can be deduced easily. FATE and DESTINI are written in around 6000 LOC in Java. For evaluation, they tested it on HDFS and found 22 new bugs, eight of which can only be found if multiple failures are injected. They also reproduced 51 known bugs.

Haryadi was asked about the news headlines he mentioned at the beginning of his talk, which showed various datacenters and companies losing data or having downtime. Did he have a sense of how many were from multiple concurrent failures? Haryadi responded that it was hard to tell from just the news. However, a lot of papers have explained that more bugs are found when injecting multiple failures.

SliceTime: A Platform for Scalable and Accurate Network Emulation

Elias Weingärtner, Florian Schmidt, Hendrik vom Lehn, Tobias Heer, and Klaus Wehrle, RWTH Aachen University

Elias Weingärtner started his talk discussing the pros and cons of network testbeds, network simulations, and network emulations. A network emulator, in particular, takes a real-world client and a discrete event-based network simulator. However, each has a different timing concept. Network simulation uses a series of discrete events, while real-world clients depend on continuous wall-clock time. The common solution is to bring the two times together by waiting between events, but many simulators are not real-time compatible, which causes time drift.

To prevent time drift, Elias presented the SliceTime system, which takes the approach of slowing down the client to match the simulator's speed. Clients are placed inside virtual machines and a barrier algorithm is used to limit time drift. SliceTime is implemented with a virtual machine for Linux and Windows, the ns-3 simulator, and a Xen hypervisor. The synchronizer implements the barrier synchronization algorithm and a modified sEDF scheduler is used to execute Xen domains for a time-slice duration. The event scheduler checks to see if the next event in the queue resides in the current time slice—if yes, it lets the event go, and if not, it blocks.

SliceTime was evaluated at different slice sizes, network configurations, and traffic setups. They found that it is resource-efficient, with low overhead for even 1 ms time slices. It is open source at http://www.comsys.rwth-aachen.de/projects/slicetime.

Jason Li from Intelligent Automation, Inc., asked if SliceTime was using an actual operating system or just an application connected to ns-3 through a socket. Elias replied that SliceTime does not make assumptions about what is put in the VM. Jason then asked whether the WiFi add-on was adapted for the ns-3 WiFi model or put in the VM. Elias said they implemented a device driver that gave the VM WiFi access into the simulation. Did every real client have a shadow node in the simulation and confirmation that the client never transmitted over any real interface? Yes. How does SliceTime compare to the optnet shadow model and hardware loop? It is similar, but the idea of network emulation was introduced 10 years ago. The key idea of SliceTime is that it slows down the simulation. Aaron Gember from the University of Wisconsin—Madison wondered what to keep in mind when choosing the size of a time slice. Elias said that the RTT is important and that the slice should be smaller than the network's RTT. How was time handled in the system—the simulator time is quantized, but real-world time is not, so does SliceTime attempt to quantize the client's time? Elias explained that the client time is not quantized and therefore does not execute in an entirely deterministic fashion. This also means that SliceTime cannot do absolute replicability.

Mobile Wireless

Summarized by Jeff Terrace (jterrace@cs.princeton.edu)

Accurate, Low-Energy Trajectory Mapping for Mobile Devices

Arvind Thiagarajan, Lenin Ravindranath, Hari Balakrishnan, Samuel Madden, and Lewis Girod, MIT Computer Science and Artificial Intelligence Laboratory

Arvind Thiagarajan presented a new method for providing accurate trajectory maps for mobile devices. Existing applications use a method that frequently polls the mobile unit's GPS device. This works well, but it quickly drains the battery of the unit, because GPS devices require a relatively large amount of power. There are also existing methods that use cell tower triangulation to attain point localization, but the low accuracy doesn't work well for trajectory maps.

The novel method proposed here, called CTrack, uses the low-power cell signal, combined with a unit's low-power accelerometer and compass to map these sensor readings to a trajectory. The method takes the recorded fingerprints from a device and maps them to grid locations using a dynamic programming algorithm. The results of this method produces trajectories with 75% precision at a small fraction of the power required for a GPS solution.

Responding to questions, Arvind stated that they did not measure the effect of road density; this method also does not

work well for unmapped trajectories (such as running in a field). Mapping the city of Boston required 125 hours of training data. When asked if the HMM could use accuracy level of the fingerprints as input, Arvind said that when they tried this, it actually reduced the flexibility of the HMM, producing worse results.

Improving Wireless Network Performance Using Sensor Hints

Lenin Ravindranath, Calvin Newport, Hari Balakrishnan, and Samuel Madden, MIT Computer Science and Artificial Intelligence Laboratory

Lenin Ravindranath presented a new wireless protocol for mobile devices such as smartphones and tablets. Existing wireless protocols work well when a device is mostly in a fixed position, but when a device is in motion, packet loss is bursty, resulting in poor throughput.

A novel algorithm for rate adaptation, specifically designed for devices in motion, called RapidSample, gets much better throughput than traditional algorithms when a device is in motion. The problem with RapidSample, however, is that it performs poorly when a device is not in motion.

The key insight of this work is that they leverage the sensors provided in many mobile devices (e.g., GPS, accelerometer, compass, or gyroscope) to give hints to the wireless protocol stack as to whether the device is in motion. An evaluation of mixed-mobility traces shows that the hint-aware algorithm outperforms all other methods, giving ideal throughput when a device is either static or mobile.

Jason Li asked about how useful RapidSample is when the movements are small, and about the dangers of using the tool. Lenin replied that the tool works with sensor hints, but at a rather coarse scale. Using GPS does require a lot of energy, and designing a wireless device to take advantage of these sensor hints is difficult. Someone asked about supplying mapping information, for example, in large offices, and Lenin replied that they wanted to work without maps using commodity appliances.

Poster Session

First set of posters summarized by Kristin Stephens (ksteph@cs.berkeley.edu)

Assuring Network Service with Bandwidth and Integrity Based Fairness

Fariba Khan (fkhan2@illinois.edu) and Carl A. Gunter, University of Illinois Urbana-Champaign

Authentication is a desirable property on the Internet. However, ISPs need direct incentive to install it on their networks. Fariba's poster presented the idea of using priority-based queuing to provide this incentive. Authenticated packets would be given higher priority over unauthenticated packets. Included in the idea is allowing fine-grained authentication based on how many addresses the authenticated mask covered, with the more addresses a mask covered, the lower the priority.

Predicting the Safety of Web Logins with Piigeon

Xiao Sophia Wang (wangxiao@cs.washington.edu) and David Choffnes, University of Washington; Patrick Gage Kelley, Carnegie Mellon University; Ben Greenstein, Intel Labs Seattle; David Wetherall, University of Washington

Many Web sites send their users' login information in the clear, without telling their users. Sophia's poster presented a project that had three goals: predict if the login is sent in the clear, inform the user before this happens, and understand how logins are protected in the wild. To do this, Web pages were analyzed using various heuristics. A Firefox plugin has been developed and can be found at http://piigeon.org.

Wide-Area Datacenter Selection and Traffic Engineering for Online Service Providers

Srinivas Narayana (narayana@cs.princeton.edu), Joe Wenjie Jiang, and Jennifer Rexford, Princeton University

This poster presented the observation that mapping nodes and datacenters do not exchange information with each other when it comes to load balancing and performance. This then begs the question, is it useful to share information? The poster explained that sharing path performance, routing information, and mapping information between mapping nodes and datacenters can help improve both mapping and routing decisions. Datacenters can use the shared information to choose which paths to use when they are multihomed. And the mapping nodes can use the shared information to choose which datacenter to send requests to. Guided by an optimization framework, the poster also describes an architecture in which these decisions are performed in a distributed fashion with intermittent information sharing.

Don't Settle for Eventual: Stronger Consistency for Wide-Area Storage

Wyatt Lloyd (wlloyd@cs.princeton.edu) and Michael J. Freedman, Princeton University; Michael Kaminsky, Intel Labs; David G. Andersen, Carnegie Mellon University

The CAP theorem states that a distributed data store can only have two of three properties: strong consistency (linearizability), availability, and partition tolerance, so most settle for eventual consistency. This poster presented a new consistency model for distributed systems that still permits

availability and partition tolerance. By combining causal consistency—related ops appear in the correct order—and per-key sequential consistency, they achieve consistency that is stronger than eventual. Wyatt Lloyd presented a system that realizes this new consistency model. It includes a modified key-value store and a client library with calls that track causality and a multiget operation that provides a consistent view of multiple keys.

dBug: Systematic Testing of Distributed and Multi-Threaded Systems

Jiri Simsa (jsimsa@cs.cmu.edu), Garth Gibson, and Randy Bryant, Carnegie Mellon University

This poster presents the design, implementation, and evaluation of dBug—a tool for systematic testing of concurrent programs. dBug repeatedly executes a test of a program while controlling the order in which concurrent events, such as intra- and inter-process synchronization and communications, execute. By doing so, repeated execution of a test can systematically explore different behaviors of the program and outcomes of the test. The implementation uses run-time interposition as a mechanism for transparently integrating the tool with existing programs. The evaluation used the tool to identify a number of concurrency errors such as deadlocks and incorrect usage of shared library API in unmodified distributed and multi-threaded programs.

vFlood: Opportunistic Flooding to Improve TCP Transmit Performance in Virtualized Clouds

Sahan Gamage (sgamage@purdue.edu), Ardalan Kangarlou, Ramana Rao Kompella, and Dongyan Xu, Purdue University

When multiple virtual machines (VMs) share a given CPU, VM scheduling latencies can be on the order of a few milliseconds, and may contribute to increasing the perceived round-trip times for TCP connections in sub-millisecond datacenter networks, causing significant degradation in throughput. vFlood is a lightweight solution which allows the VM to opportunistically flood packets to the driver domain and offloads the TCP congestion control to the driver domain, in order to mask the effects of virtualization. They found that this significantly improves the performance of small flows and is non-trivial for larger flows.

Suppressing Malicious Bot Traffic Using an Accurate Human Attester

Muhammad Jamshed, Younghwan Go (yhwan@ndsl.kaist.edu), and KyoungSoo Park, KAIST

A new human attestation method to suppress malicious bots, Not-a-Bot, ensures that a user is human in a message by associating an arbitrary input event proof occurring within a specific time window (usually 1 second) during its message construction. This model is vulnerable to forgery by smart bots that can exploit this time interval. This poster presents a model where human use is more securely bound to a legitimate message by attaching proof of all relevant input events generated during message construction. It uses a TPM and processor's late-launch technology that is available in all modern machines. The results show that the overhead for generating proofs is tolerable for human use and practical for all applications.

On Lookups in Content-based Routers

Ashok Anand (ashok@cs.wisc.edu), Nilay Vaish, and Aditya Akella, University of Wisconsin—Madison

Content-driven networks are great, but many of these systems do not include the details of their implementation. This poster focused on the problem of frequent updates in a router. To speed up the process of updating they used several techniques: fingerprint packets, reduced communication between threads, a shared partitioned table, a lock at the level of table partition, batch jobs, and a particular table partition access sequence to reduce contention. Their results show this greatly speeds up updates.

Towards Transactional Cloud Resource Orchestration

Changbin Liu (changbl@seas.upenn.edu), University of Pennsylvania; Yun Mao, Xu Chen, and Mary F. Fernandez, AT&T Labs—Research; Boon Thau Loo, University of Pennsylvania; Kobus Van der Merwe, AT&T Labs—Research

Infrastructure as a service, cloud computing, and datacenter resource management are hard. This poster presented a framework that provided the ability to make cloud resource orchestration like a transaction. This means it had the properties of atomicity, consistency, isolation, and durability. This poster included a demo in which they live-migrated a virtual machine from one datacenter to another over a wide-area network.

Second set of posters summarized by Aaron Gember (agember@cs.wisc.edu)

SNEAP: A Social Network-Enabled EAP Method: No More Open Hotspots

Aldo Cassola, Tao Jin, Harsh Kumar, Guevara Noubir, and Kamal Sharma, Northeastern University

SNEAP leverages existing social networks to grant users more ubiquitous access to wireless connectivity. Unlike prior approaches which rely on keys provided by a special authentication service, SNEAP leverages a modified version of WPA-Enterprise to authenticate users and verify trust

relationships. Individuals who wish to share their wireless access point (AP) install a special SNEAP OpenWRT image on their router and register the router with the SNEAP radius server. When friends are in the vicinity and wish to connect to the AP, they authenticate using their social network, e.g., Facebook, credentials. The radius server verifies the credentials and ensures that a trust relationship, e.g., friendship, exists between the AP owner and the authenticating user. A simple demonstration showed the router registration process and user authentication process using a wireless router and two laptops (one serving as the owner's home computer and one as a client) with both tasks relying on a Facebook application and remote radius server.

Network Configuration Analysis

Theophilus Benson, University of Wisconsin—Madison; Hyojoon Kim, Georgia Institute of Technology; Aditya Akella, University of Wisconsin—Madison; Nick Feamster, Georgia Institute of Technology

Enterprise networks are difficult to configure, requiring thousands of lines of configuration commands distributed across thousands of devices to set up a wide array of protocols and standards. The objective of this work is to understand (1) how the network changes over time and (2) how operators interact with different devices, with the goal of designing better network management tools. The authors use a five-year history of configuration files from two enterprise networks to analyze the changes that occur. They observe that the most frequent change, which involves an interface and a route protocol, occurs to add a department. Other changes that occur frequently include adding a department with security controls, changing the control plane, and changing the addresses assigned to a department. However, the most prevalent configurations are not the most frequently changed configurations. Analysis also shows that the majority of configuration changes occur in routers, rather than in firewalls or switches. Lastly, most switch changes occur during the workday, while most firewall changes occur in the early evening.

BISMark: A Platform for Studying Home Networks

Walter de Donato, University of Napoli Federico II; Srikanth Sundaresan and Nick Feamster, Georgia Institute of Technology; Renata Teixeira, CNRS/UPMC Sorbonne Universités; Antonio Pescapé, University of Napoli Federico II

BISMark relies on instrumented home routers to measure the behavior of home networks and the last-hop broadband links that connect homes to the Internet. Two types of devices are being deployed in homes: a powerful NoxBox that runs Linux (16 of which have already been deployed) and a less-flexible but more stable Netgear router. With these devices, the authors can study the impact of ISP policies, the impact of certain home network configurations, e.g., WiFi, and the usage profiles of home network users. Active measurements include TCP/UDP throughput, last-mile latency, upstream/downstream jitter, packet loss, and DNS delay and availability. Passive measurements include per-application throughput (planned for the future), packet header captures, WiFi client and AP behavior, and DHCP requests. Measurements have analyzed an ISP policy termed PowerBoost—receiving a high burst throughput for a few seconds—explaining how it is implemented, how variable it is, and its impact on user perception of throughput. Analysis has also uncovered the presence of significant buffering in modems, which can cause up to multiple seconds of latency when the uplink is saturated.

Seattle: The Internet as a Testbed

Jeff Rasley, Monzur Muhammad, Alex Hanson, Sebastian Morgan, Alan Loh, and Justin Cappos, University of Washington

Seattle is a peer-to-peer version of PlanetLab, providing an environment for Internet-wide networking experiments. Seattle relies on real user nodes to provide a realistic testing environment. While PlanetLab provides experimenters with Linux virtual machines connected to Internet2, Seattle provides experimenters with a Python environment on machines with home broadband connectivity. Seattle has diurnal end-user availability, some mobile nodes and some limited access due to firewalls and NATs, and runs on servers, mobile phones, and laptops. The testbed has been used in the classroom for projects on link state routing and distributed hash tables based on Chord. Researchers have used Seattle for YouTube CDN mapping, tracking mobility and diurnal patterns, testing peer-to-peer encrypted file storage, and many other projects. Seattle is available for use now; users who donate N machines get access to 10^N machines. See http://seattle.cs.washington.edu for more information.

An Empirical Study on the Person-to-Public Distribution of Tor Bridges

Xiao Wang, Jinqiao Shi, Binxing Fang, Qingfeng Tan, and Li Guo, Institute of Computing Technology, CAS, China

Tor bridges, a technology designed to allow unrestricted Internet access in the presence of censors, are typically distributed in an authority-to-public or person-to-person fashion. However, users may sometimes post Tor bridge IP addresses online, eliminating some of the secrecy. In this work, the authors survey the status of potential Tor bridges located via Web searches for the phrases "Here are your bridge relays" and "bridge IP:Port". Addresses were found on a small fraction of Web sites. Of the 579 potential bridges the searches reveal, 64 IPs were still serving as bridges, 95 were Tor relays, and 420 were unidentifiable. Unfortunately,

such public bridges are a considerable portion of the available bridges, posting a threat to Tor's censorship-resistance.

Structured Comparative Analysis of Systems Logs Using Distalyzer

Karthik Nagaraj, Charles Killian, and Jennifer Neville, Purdue University

Distributed system logs contain lots of information that is useful to developers, but the size of these logs makes manual analysis infeasible. Distalyzer is a distributed system performance analyzer that automatically looks for differences between two sets of logs. Logs are gathered from two different systems run in the same environment, or from the same system run with different parameters. Systems provide two types of logs, state logs and event logs, each with a known structure. Distalyzer compares the occurrences of events or the values of states to identify differences between the two system runs. The discovered differences are ranked and presented to the developer to help identify performance leaks.

Work in Progress: Uncovering the Privacy Implications of Web Usage

Hendrik vom Lehn, Jó Ágila Bitsch Link, and Klaus Wehrle, RWTH Aachen University, Germany

Internet users visit a wide array of Web sites, providing personal information to many of these sites. For example, users may provide their name and email address to gain access to a site. Knowing what information a user has disclosed during their browsing sessions, enables a user to identify potential information leaks and make better decisions about their Web usage. Especially important is the detection of hidden information leakage, i.e., private information it is not obvious a user is disclosing. Gathering this information is facilitated by a browser plugin which receives a copy of all HTTP traffic. The traffic is analyzed, using various modules, to identify personal data a user has disclosed; the data is stored locally along with metadata from the HTTP exchange. Users can view the extracted information in a summarized form to better understand the level of privacy in their Web browsing, and future tools may automatically warn users of potential privacy concerns.

Armonia: A Highly Available Distributed Transactional Memory Store

Mehul A. Shah, Nathan Binkert, Stavros Harizopoulos, Wojciech Golab, and Indrajit Roy, HP Labs

Armonia improves on prior distributed transactional memory stores by using Paxos-based replication instead of a primary-backup design. The system has four goals: (1) scalability to 100s of terabytes of memory, (2) microseconds latency, (3) transactional consistency, and (4) five nines of availability. Armonia integrates transaction execution and commit protocols with Paxos to avoid extra network round trips and achieve low latencies. Using Paxos provides better availability because data is replicated more than once, compared to only two copies in a primary-backup design, and because Paxos does not need accurate failure detection.

InContext: Simple Parallelism for Distributed Applications

Sunghwan Yoo, Hyojeong Lee, Charles Killian, and Milind Kulkarni, Purdue University

InContext is an event-driven framework suitable for systems seeking parallel execution capabilities. With the InContext model, events can be global (enabling them to both read and write global service state), anon (enabling them to only read global system state), or none (providing no access to global service state). During execution, only one event at a time can be in the global state. Events desiring to enter the global or anon states must wait for existing events to commit or enter the anon state. Optionally, a commit may be deferred to keep a logical ordering of events. The authors modified Mace to support this parallel event model. During runtime, applications make an implicit upgrade to move from none to anon, and applications add an explicit downgrade call to move from global to anon. A model checker and simulator are provided as additional tools.

A Service Access Layer, at Your Service

David Shue, Matvey Arye, Prem Gopalan, and Erik Nordström, Princeton University; Steven Y. Ko, SUNY, Buffalo; Michael J. Freedman and Jennifer Rexford, Princeton University

The Internet was designed for host-to-host communication, but recent trends are toward a service-centric Internet. In the service-centric architecture proposed by the authors, a service access layer is added between the transport and network layers. The service access layer enables applications to use topology-independent service names instead of topology-dependent addresses. The transport layer is purely responsible for data delivery; the service access layer deals with resolving a service, initiating and terminating connections to a particular instance, and maintaining affinity to that instance across network address changes such as during VM migration or client mobility events. When an application desires to establish a TCP connection to a specific service, the service layer intercepts the initial SYN and forwards the packet to an authoritative service router that recursively resolves the packet to an instance of the service, which replies directly to the source with a SYN/ACK. After the connection is established, all further communication occurs directly between a user and the selected service instance. In

the demo, an Android client broadcast a request for a specific service, and the phone was served by the server (running on each laptop) whose SYN-ACK was received first.

Third set of posters

Block-based Bitrate Control for Wireless Networks

Xiaozheng Tie (xztie@cs.umass.edu), Anand Seetharam (anand@cs.umass.edu), Arun Venkataramani (arun@cs.umass.edu), Deepak Ganesan (dganesan@cs.umass.edu), and Dennis L. Goeckel (goeckel@ecs.umass.edu)

The poster presented BlockRate, a wireless bitrate control algorithm designed for blocks, or large contiguous units of transmitted data, as opposed to small packets. Recent trends in research as well as in practice (e.g., 802.11n) suggest significant overhead amortization benefits of blocks. Yet state-of-the-art bitrate algorithms are optimized for adaptation on a per-packet basis, so they can either have the amortization benefits of blocks or high responsiveness to underlying channel conditions of packets, but not both. To bridge this disparity, BlockRate employs multiple bitrates within a block that are predictive of future channel conditions. In each feedback round, BlockRate uses a history-based scheme to predict the SNR for packets within the next block. In slow-changing scenarios, such as indoor mobility, BlockRate uses a simple linear regression model to predict the SNR trend over the next block. In fast-changing scenarios, such as vehicular mobility, BlockRate uses a path loss model to capture more significant SNR variations. They have implemented a prototype of BlockRate in a commodity 802.11 driver and evaluated it via deployment on an indoor mesh testbed as well as an outdoor vehicular testbed. The evaluation shows that BlockRate achieves up to 1.4x and 2.8x improvement in goodput under indoor and outdoor mobility, respectively.

LocalFlow: Simple, Local Flow Scheduling in Datacenters

Siddhartha Sen (sssix@princeton.edu), Sunghwan Ihm, Kay Ousterhout, and Michael J. Freedman, Princeton University

LocalFlow is a completely local approach to flow routing in datacenter networks. It addresses the problem of large flows, which can significantly degrade network utilization and starve other flows if routed through oversubscribed links. LocalFlow is an efficient bin-packing algorithm that runs locally on each network switch. It uses two key ideas. First, it proactively splits and rate-limits flows to ensure that they are small enough before collisions occur. This guarantees provably optimal, max-min fair routing in standard fat-tree networks. Second, it uses a splitting technique that leverages wildcard rules in upcoming programmable commodity switches to group contiguous packets into "flowlets" to minimize end-host reordering. Previous flow-routing algorithms rely either on centralized schedulers or on end-host control of multiple paths. The first approach lacks parallelism and thus scalability, while the second approach cannot predict the paths of flows and thus only works if the flows-to-paths ratio is high. They presented the design and theoretical analysis of LocalFlow, as well as preliminary simulation results that demonstrated the practicality of splitting. For example, based on packet traces from a university datacenter switch, LocalFlow splits less than 4.3% of total flows on average, using approximate splitting to within 5% on a 1024-host fat-tree network.

A Memory-Efficient, High-Performance Key-Value Store

Hyeontaek Lim (hl@cs.cmu.edu), Bin Fan, and David G. Andersen, Carnegie Mellon University; Michael Kaminsky, Intel Labs

This work develops a memory-efficient, high-performance key-value store. It focuses on indexing techniques in fast key-value stores as the indexing data structure is one of the main sources of the memory consumption in key-value store systems (e.g., 4 bytes/item), while DRAM is becoming a scarcer resource, as flash/disk's capacity per dollar is growing much faster than DRAM's. This work proposes three basic key-value store designs based on new indexing data structures (partial-key cuckoo hash tables and entropy-coded tries) and combines these basic stores to build a full key-value system; by inserting new data to a write-optimized basic store and gradually moving the data to a very memory-efficient basic store, this system achieves approximately 0.7 bytes/item, while a data retrieval requires only 1.01 flash reads, which allows nearly full utilization of flash drive random read performance.

KARMA: Trade Your Idle Resources Now for Elastic Scale Out Later

Shriram Rajagopalan (rshriram@cs.ubc.ca), Dhirendra Singh Kholia (dkholia@cs.ubc.ca), Mohammad Shamma (mshamma@cs.ubc.ca), and Andrew Warfield (andy@cs.ubc.ca)

Virtual machine (VM) utilization in the cloud seldom exceeds 10–12%, as these were, in an earlier life, underutilized physical servers. But these VMs are charged for resources like CPU and memory even though they remain idle during low load periods. This work proposes a system called KARMA that will leverage techniques used by grid systems such as Condor, BOINC, etc., to pool idle resources of VMs belonging to a community of users in the cloud. KARMA will use container virtualization techniques such as OpenVZ to create low overhead application containers (sandboxes). During peak loads, instead of scaling in cloud by launching additional short-lived VMs (and paying for them), users could scale their applications by launching application containers (for free) that draw upon the resources

in the KARMA pool. Container virtualization offers both performance and isolation from the host VM's perspective. To ensure information privacy from the guest application's perspective, they propose using "malwarized applications" that leverage malware creation techniques such as packing and code obfuscation to deter the host VM from tampering with the guest application. Performing resource sharing from within virtual machines (VMs) has the potential to reduce user costs at the expense of global resource optimization. Their intention is to provide a short-term benefit for some users, that (antagonistically) motivates a longer term optimization of how resources are accounted and charged.

DARD: Distributed Adaptive Routing for Datacenter Networks

Xin Wu (xinwu@cs.duke.edu) and Xiaowei Yang, Duke University

From http://www.cs.duke.edu/events/?id=00000001331

Datacenter networks typically have many paths connecting each host pair to achieve high bisection bandwidth for arbitrary communication patterns. Fully utilizing the bisection bandwidth may require flows between the same source destination pair to take different paths to avoid hot spots. However, the existing routing protocols have little support for load-sensitive adaptive routing. This work proposes DARD, a Distributed Adaptive Routing architecture for Datacenter networks. DARD allows each end host to adjust traffic from overloaded paths to underloaded ones without central coordination. They use an OpenFlow implementation and simulations to show that DARD can effectively use the network's bisection bandwidth. It out-performs previous solutions based on random flow-level scheduling, and performs similarly to previous work that assigns flows to paths using a centralized scheduler but without its scaling limitation. They use competitive game theory to show that DARD's flow scheduling algorithm makes progress in every step and converges to a Nash equilibrium in finite steps. The evaluation results suggest its gap to the optimal solution is likely to be small in practice.

WebCloud: Enabling More Direct Content Exchange Between Web Clients

Fangfei Zhou (youyou@ccs.neu.edu), Liang Zhang (liang@ccs.neu.edu), and Eric J. Franco, Northeastern University; Richard Revis, Jandrell, Pearson & Revis Ltd.; Alan Mislove (amislove@ccs.neu.edu) and Ravi Sundaram, Northeastern University

From http://www.northeastern.edu/expo/view_abstracts/abstract.php?sid=1814

We are at the beginning of a shift in how content is created and exchanged over the Internet: today, individual users, powered by devices like digital cameras and services like online social networks, are creating content that represents a significant fraction of Internet traffic. As a result, compared to content shared over the Internet just a few years ago, content today increasingly is generated and exchanged at the edge of the network. Unfortunately, the existing techniques and infrastructure that are still used to serve this content, such as centralized content distribution networks, are ill-suited for the new patterns of content creation and exchange, resulting in a mismatch of infrastructure and workload.

In this work, they take a step towards addressing this situation by introducing WebCloud, a content distribution system that enables more direct content sharing between users in existing online social networks. WebCloud works by adding a small amount of JavaScript to a social network's Web pages, locally storing content that each user views. When another user browses the content, the JavaScript fetches it from one of the user's online friends instead of directly from the social networking site. The result is a more direct exchange of content between users; essentially, WebCloud leverages the storage and bandwidth resources of social networking users to help serve content. Because WebCloud is built using techniques already present in many Web browsers, it can be applied today to many online social networking sites. They demonstrated the practicality of WebCloud with simulations and a prototype deployment.

Seeking Efficient Data-Intensive Computing

Elie Krevat (ekrevat@andrew.cmu.edu) and Tomer Shiran, Carnegie Mellon University; Eric A. Anderson, Joseph Tucek, and Jay J. Wylie, HP Labs; Gregory R. Ganger, Carnegie Mellon University

From http://www.cs.cmu.edu/~ekrevat/

New programming frameworks for scale-out parallel analysis, such as MapReduce and Hadoop, have become a cornerstone for exploiting large datasets. However, there has been little analysis of how these systems perform relative to the capabilities of the hardware on which they run. They have developed a simple model of I/O resource consumption and applied it to a MapReduce workload to produce an ideal lower bound on its runtime, exposing the inefficiency of popular scale-out systems. Using a simplified dataflow processing tool called Parallel DataSeries (PDS), they demonstrated that the model's ideal can be approached within 20%. Current research explores why any DISC system built atop standard OS and networking services faces a gap between ideal and actual performance. They have found that disk stragglers and network slowdown effects are the prime culprits for lost efficiency in PDS. They are also building up PDS into a more feature-rich system (e.g., to support fault tolerance), to understand all areas where efficiency is lost at scale.

Datacenters Learning to Share

Summarized by Andrew Ferguson (adf@cs.brown.edu)

Mesos: A Platform for Fine-Grained Resource Sharing in the Datacenter

Benjamin Hindman, Andy Konwinski, Matei Zaharia, Ali Ghodsi, Anthony D. Joseph, Randy Katz, Scott Shenker, and Ion Stoica, University of California, Berkeley

Recent attention to the problem of performing large-scale computation on commodity clusters has yielded numerous computational frameworks, such as Hadoop, Dryad, and CIEL. Organizations that wish to use several frameworks, or even several versions of the same framework, in isolation must currently use multiple clusters, which can require extensive data duplication and yield idle resources.

Matei Zaharia presented Mesos, a layer for sharing common resources such as CPU cores, memory, and data blocks over which diverse computational frameworks can run. Besides enabling multiple frameworks to fairly share the same cluster, Mesos makes it easier to build and deploy new, specialized frameworks for more unique computations. To illustrate this point, Zaharia worked with his group to develop Spark, a lightweight system for machine learning which achieves significantly higher performance than Hadoop.

The core of Mesos is a small microkernel-like engine which makes resource offers to the frameworks running above it. These resource offers describe slots of CPU and memory that are available on particular cluster nodes. The frameworks accept or reject the offers and employ their own scheduling logic on accepted offers. Mesos uses dominant resource fairness (see summary below) to choose which resource offers to make. Because of this design, applications running on Mesos perform best when they have many fine-grained tasks to fill the offers. However, this is not a requirement; Zaharia presented examples of Mesos clusters simultaneously supporting Hadoop, MPI, Torque, and Spark applications.

Mesos consists of about 20,000 lines of C++ and is available as an open-source Apache Incubator project. The Mesos masters have only soft-state, and they provide high availability failover using Apache ZooKeeper. It is currently in use at Twitter, Conviva, and by researchers at UCSF.

George Porter asked how Mesos shares the network bandwidth between competing applications. Zaharia said that Mesos does not do anything in particular but that the next presentation addresses this issue. Srikanth Kandula asked if the predictability of job completion times suffered. Zaharia replied that completion times depend upon the intra-framework scheduling policy, and jobs may finish faster. Arkady Kanevsky (VMware) asked about the mechanism used to isolate frameworks running on the same nodes, and Zaharia said that Mesos uses Solaris zones and Linux containers.

Sharing the Datacenter Network

Alan Shieh, Microsoft Research and Cornell University; Srikanth Kandula, Microsoft Research; Albert Greenberg and Changhoon Kim, Windows Azure; Bikas Saha, Microsoft Bing

Today's datacenters occasionally suffer from poor network performance due to application interference. A single application may monopolize a shared resource with many TCP flows, use a more aggressive variant of TCP, or simply use UDP; malicious users can launch denial of service attacks against other virtual machines or entire racks.

Alan Shieh presented Seawall, a client-side solution to sharing the datacenter network by decoupling network allocation from applications' traffic profiles. Seawall sits in the hypervisor of each datacenter node and establishes a single tunnel between each source and destination VM. It determines per-link rate limits and converts them to per-tunnel rate limits.

Seawall also makes use of periodic congestion feedback (e.g., percentage of lost packets) and ECN marks to adapt the rate of traffic permitted through each tunnel. Shieh describes this as "link-oriented congestion control," and it can employ standard congestion control loops such as AIMD, CUBIC, and DCTCP. A heuristic is used to convert path-congestion feedback into link-level congestion feedback, because a congested link will result in path congestion on many tunnels. Path feedback is combined in proportion to the amount of traffic on that path.

Shieh presented two evaluations of the Seawall system. In the first experiment, an application attempted to use a UDP flood to deny service over a particular link. In the second, an application attempted to gain more than its fair share of the network by using many TCP flows. In both cases, the Seawall system appropriately isolated and limited the malicious traffic. Finally, Shieh compared Seawall with related works such as SecondNet, Gatekeeper, and CloudPolice.

Steven Hand (Cambridge) asked how this could be scaled to an Internet-wide topology, and Shieh answered that it would be difficult because the necessary topology information is not readily available. Ye Wang asked how this interacts with the virtual machine's existing network stacks. Shieh indicated that they modified the TCP stacks in the VMs to respect the congestion control parameters passed up from the Seawall system.

Dominant Resource Fairness: Fair Allocation of Multiple Resource Types

Ali Ghodsi, Matei Zaharia, Benjamin Hindman, Andy Konwinski, Scott Shenker, and Ion Stoica, University of California, Berkeley

Many scheduling approaches, such as weighted fair-queueing, round-robin, and Hadoop and Quincy's task schedulers employ (weighted) max-min fairness to fairly allocate resources, for several reasons. The first reason is that it provides a share guarantee: each of n users will get at least $1/n$ of the scheduled resource. Furthermore, if one user requests less of the shared resource, the newly available resource is split evenly by the other users. Secondly, max-min fairness is strategy-proof—users are not incentivized to lie, and would not improve their performance by requesting more resources than needed. Finally, max-min fairness provides the flexibility to implement a variety of scheduling policies, such as proportional sharing, fixed or dynamic priorities, and reservations.

Ali Ghodsi presented dominant resource fairness (DRF), a strategy for extending max-min fairness to the domain of many users making heterogeneous demands on multiple resources. The development of this strategy is important for today's datacenters, in which not all tasks require the same ratio of CPU to memory. In the DRF model, all allocated resources are divisible, and users record their task requirements using a demand vector, such as <1 CPU, 4 GB of RAM>. The DRF algorithm then applies max-min fairness to the user's dominant resource—the resource in the cluster which is most tightly constrained.

Ghodsi compared DRF with two alternative policies: asset fairness, which equalizes each user's sum of resource shares, and Competitive Equilibrium from Equal Incomes (CEEI). CEEI gives each of n users $1/n$ of each resource, which can then be freely traded in a market. The speaker showed that, unlike DRF, asset fairness does not guarantee that each user will get at least $1/n$ of each resource, and that CEEI is not strategy-proof. Finally, Ghodsi presented the results of a simulation of Facebook's Hadoop workload and showed that DRF-based scheduling could outperform the existing Fair Scheduler.

Derek Murray (Cambridge) wanted to know if the performance bottleneck was perhaps disk access and not memory or CPU. Ghodsi responded that they did not have disk access statistics, but confirmed that they had observed clusters hitting their CPU and memory limits. Michael Freedman asked if it was possible to make Amazon EC2 exchange strategy-proof; the speaker did not think it possible.

Wireless and More

Summarized by Hendrik vom Lehn (vomlehn@cs.rwth-aachen.de)

PIE in the Sky: Online Passive Interference Estimation for Enterprise WLANs

Vivek Shrivastava, Shravan Rayanchu, and Suman Banerjee, University of Wisconsin—Madison; Konstantina Papagiannaki, Intel Labs, Pittsburgh

Vivek Shrivastava started with an introduction to the use cases of interference estimation. In enterprise WLANs with a set of access points, a central management entity can be used to dynamically control the transmission power and channel assignment. The link interference ratio (LIR) is a helpful metric that is often derived from bandwidth tests. Such bandwidth tests, however, require downtime and are not scalable.

Shrivastava presented a new system for passive interference estimation (PIE) that is able to estimate the interference in a passive way in real time. Access points are equipped with a traffic sniffer and a clock synchronization mechanism. Reports from these traffic sniffers are sent to a wireless LAN controller, which then calculates the LIR based on the calculated station isolation loss rate and the interference loss rate.

They evaluated how fast PIE converges depending on the report periodicity and the occurring traffic patterns. Their measurements showed that 100 ms is enough in case of saturated traffic. An evaluation with real wireless LAN traces showed that in case of lighter traffic approximately 700 ms is required. Shrivastava then presented results of an application of PIE to data scheduling. The results showed that PIE outperforms existing approaches, especially in mobile scenarios. Shrivastava finished by telling the audience that measurements in production systems showed that hidden terminals and rate anomalies are indeed a problem and describing how PIE fits into related work and some limitations of PIE.

Dan Halperson (U. Washington) asked about data rate selection. Shrivastava answered that the caused interference depends on the chosen data rate of clients, but that they focus on interference estimation. Based on this, it would be possible to perform a more intelligent data rate selection. Shrivastava was asked if the numbers given earlier are for hidden terminal problems between access points. He explained that the numbers are for everything which the access points sees and thus also include transmissions from clients.

SpecNet: Spectrum Sensing Sans Frontières

Anand Padmanabha Iyer, Krishna Chintalapudi, Vishnu Navda, Ramachandran Ramjee, and Venkata N. Padmanabhan, Microsoft Research India; Chandra R. Murthy, Indian Institute of Science

Anand Iyer said that the wireless spectrum is currently underutilized and that new rules of the FCC make it possible to opportunistically access licensed spectrum. Current studies on the utilization of spectrum are, however, all static and cover only a few places around the world. In order to get better measurements, they decided to build a PlanetLab-like system for spectrum analyzers.

After presenting the goals and implementation challenges, Iyer presented the SpecNet architecture. A master server manages all requests and forwards them to slave servers which in turn are connected to the actual spectrum analyzers. Users can access the system using XML-RPC. The API provides both low-level access and more abstract commands through a uniform interface. After the request has been performed, the master server stores all results in a database.

Before explaining key challenges, Iyer gave a brief overview of how spectrum analyzers work. In order to decrease the required scan time, SpecNet supports several mechanisms which allow distribution of the scanning among several spectrum analyzers. Iyer presented two example applications of SpecNet: performing simple scans for a given area, and detecting and localizing violators, which requires coordination between the sensing devices. He presented a simple scan with a live demonstration. After that, he explained that expensive devices, the attenuation of buildings, and security concerns are current limitations of the system. Iyer said that they are looking for people with spectrum analyzers who want to participate in SpecNet: http://bit.ly/SpecNet.

How many sites are running the experiments? Currently, 10; they hope that more people will sign up.

Towards Street-Level Client-Independent IP Geolocation

Yong Wang, UESTC and Northwestern University; Daniel Burgener, Marcel Flores, and Aleksandar Kuzmanovic, Northwestern University; Cheng Huang, Microsoft Research

Aleksandar Kuzmanovic explained that GPS and WiFi are good localization methods for the end user, but not for a server that wants to know where the user is, which, for example, would be required for street-level online advertising. One way to determine the location of an Internet host is to use active vantage points that measure the delay and thereby approximate the host's location. More advanced approaches that also take the network topology and demographics information into account reach an accuracy of about 35 km (22 miles), which is insufficient for the desired applications.

Kuzmanovic said that the system they developed is much better in this regard. Their system is based on two insights: many Web sites provide the geographical location of the host that is serving them and relative network delays from different landmarks can be used instead of absolute delays. Their system uses multiple steps to select the landmarks which are used to determine a host's location. In the last step, the hosts location is approximated using the landmark with the minimum delay to the targets. This mechanism has the advantage that it avoids the so-called last-mile inflation.

They evaluated their system using three datasets as ground truth: one from PlanetLab, a collected residential dataset, and the locations that search engine visitors searched for. Kuzmanovic explained that for these datasets they could decrease the median error (compared to existing approaches) from 35 km to about 1.5 km. Important factors that influence the resulting quality are the density of landmarks, the population density, and the type of access network through which the host is connected.

How easily could this approach be transferred to other parts of the world? They currently have data for 200,000 landmarks in the US, but one can do the same for other regions as well. Is this going to be an open system? They will make it a commercial system.