;login:
AUGUST 2013 VOL. 38, NO. 4
ELECTRONIC SUPPLEMENT

# Conference Reports

## NSDI '13: 10th USENIX Symposium on Networked Systems Design and Implementation

Lombard, IL
April 2–5, 2013

*Summarized by: Anish Babu Bharata, Weverton Cordeiro, Rik Farrow, Utkarsh Goel, Arpit Gupta, Imranul Hoque, Peng Huang, Murad Kaplan, Scott J. Krieder, Joshua B. Leners, Muntasir Raihan Rahman*

## NSDI '13 Opening Remarks and Awards

*Summarized by Rik Farrow (rik@usenix.org)*

Jeff Mogul, NSDI co-chair, opened the conference by telling attendees that there were 170 paper submissions, and each paper received three first-round reviews. About half the papers made it into the second round, and also received three or four more reviews. By the time the PC meeting occurred, there were 64 papers left. In total, each PC member performed 25.8 reviews on average. In the end, 38 papers were accepted, more than usual. Because of this, each presentation could only be 20 minutes, including questions at the end. And by the end of the first session, things were going well.

Jeff also pointed out that, with 258 attendees, this is almost the largest NSDI ever. Jeff then singled out three PC committee members for work above and beyond the "call of duty": Jeff Chase, James Mickens, and Renata Teixeira.

Glancing at the hot topics graphic, their most mentioned terms were: *network, data, system, applications, paper,* and *power.* Jeff magnified a small section of the chart, where the words "Looking for work" were visible as a significant topic. Finally, Jeff mentioned that Ratul Mahajan and Ion Stoica would be the co-chairs for NSDI 2014.

Nick Feamster, the other co-chair, took over for the presentation of awards.

The Best Paper awards went to: "Embassies: Radically Refactoring the Web," by Jon Howell, Bryan Parno, and John R. Douceur, Microsoft Research; and "F10: A Fault-Tolerant Engineered Network," by Vincent Liu, Daniel Halperin, Arvind Krishnamurthy, and Thomas Anderson, University of Washington.

The Community awards went to: "Composing Software Defined Networks," by Christopher Monsanto and Joshua Reich, Princeton University; Nate Foster, Cornell University; Jennifer Rexford and David Walker, Princeton University; and "Expanding Rural Cellular Networks with Virtual Coverage," by Kurtis Heimerl and Kashif Ali, University of California, Berkeley; Joshua Blumenstock, University of Washington; Brian Gawalt and Eric Brewer, University of California, Berkeley.

Howell et al. received their Best Paper award for being ambitious, thought-provoking and even controversial in their paper. Liu et al. got their award for applying simple effective insights to the co-design of network topology and protocols and evaluating them well.

Community awards were given to paper authors for contributions to the community above and beyond the typical paper. Monsanto et al. received their award for releasing the code for their software, which the award committee believed would be useful and built upon by the Software Defined Networking (SDN) community. Heimerl et al. received their community award on the basis that this will help with rural cellular deployments.

## Software Defined Networking

*Summarized by Weverton Cordeiro (weverton.cordeiro@inf.ufrgs.br)*

### *Composing Software Defined Networks*

*Awarded Community Award!*

Christopher Monsanto and Joshua Reich, Princeton University; Nate Foster, Cornell University; Jennifer Rexford and David Walker, Princeton University.

Joshua Reich began by arguing that one can build a robust, large, complex network system using OpenFlow, but it is going to be a cumbersome, time-consuming, and even an error-prone task. In a brief example, he described how complex it is currently to implement even simple sequential composition of logic such as simple load balancing and routing functions. The problem is that existing platforms for network programming only expose part of the network, lacking proper support for enabling the programming of various aspects in an integrated fashion. Building network applications using existing frameworks is pretty much like building complex software using assembly language, he said.

After drawing such a picture of the current state-of-the-art, Joshua presented Pyretic, a modular and intuitive language (and system) for enabling network programmers to build sophisticated network traffic management applications. Pyretic approaches the problem of designing complex network traffic management applications by providing three abstractions to programmers: policy abstraction, network abstraction, and packet abstraction. Policy abstraction enables programmers to use composition operators (sequential and parallel), which opens the door for doing all sorts of functional composition. The network abstraction lies on top of that, enabling programmers to decompose network soft-

ware logic based on topologies. Finally, the packet abstraction provides extensive headers, which form the foundation for the previous abstractions.

During the talk, Joshua navigated through a series of examples that highlighted the potentialities provided by each kind of abstraction, which now enables network programmers to focus on the meaning of their applications instead of concentrating on low-level details. "One single line, absolutely precise," he said. He also emphasized that Pyretic captures the full power of OpenFlow by providing one high-level policy for each natively supported policy, and that Pyretic can be used to implement both static and dynamic policies; a MAC-learning example described the power of these policies. Joshua concluded his technical discussion by sketching the implementation of the topology abstraction function in the context of a one-big-switch transformation example. Finally, he encouraged the audience to experience the full power of Pyretic by referring to the project Web site: http://www.frenetic-lang.org/pyretic.

Srimat Chakradhar (NEC Labs Princeton) asked about the things one can do with OpenFlow but cannot with Pyretic. In terms of capabilities, Joshua did not think there was anything one could implement with OpenFlow but not with Pyretic, though he noted that what one does give up is the ability to finely manage table resource usage, much in the same way that a Java programmer can no longer manage physical registers and memory. Chakradhar also asked about the impact on end-to-end latency. Joshua replied that the current prototype is mostly an interpreter, but with the micro-flow compiler that will be released shortly one would have the same performance. Rajesh Nishtala (Facebook) asked about the impact on queuing behavior. Joshua responded that they hadn't yet done this testing, but expected performance comparable to other systems.

### *VeriFlow: Verifying Network-Wide Invariants in Real Time*

Ahmed Khurshid, Xuan Zou, Wenxuan Zhou, Matthew Caesar, and P. Brighten Godfrey, University of Illinois at Urbana-Champaign.

Ahmed Khurshid started his talk by discussing the existing challenges for networking debugging: tons of devices running different protocols, from various types and vendors, and intricate relationships between devices, with several operators configuring them, among others. These challenges make it difficult for one to test every possible scenario. As a consequence, bugs may go hidden and affect production networks in a variety of ways (e.g., degrading their performance and/or making them more prone to attacks).

Ahmed enumerated some existing network debugging techniques, e.g., traffic flow monitoring and configuration verification. With regard to configuration verification, he emphasized a crucial problem: the only input taken is configuration scripts. Everything else (control-plane state, data-plane state, and network behavior) is predicted. To bridge this gap, Ahmed introduced a novel approach, data-plane verification, which considers data-plane state as input for the verification process, thus making it less dependent on predictions and closer to actual network behavior. He also emphasized running this verification task in real time in contrast to existing approaches such as FlowChecker, Anteater, and Header Space Analysis, as the network evolves.

This approach brings us to VeriFlow, a tool for checking network-wide invariants in real time. VeriFlow introduces a layer between the controller and devices in a Software Defined Network (SDN), thus enabling the verification of rules before they are deployed in the network. In summary, VeriFlow operates by intercepting new rules and building the equivalence classes associated to them, a set of packets that are affected by the rule. For each equivalence class computed, VeriFlow generates individual forwarding graphs, which model the forwarding behavior of the packets in the equivalence class through the network. Finally, VeriFlow runs custom queries to find problems. VeriFlow has a set of built-in queries to verify certain invariants; however, it also exposes an API for writing custom invariant checkers. During his talk, Ahmed extensively discussed the computation of equivalence classes (which use multi-dimensional prefix tries), always highlighting the aspects that make it an efficient and quick process. From the set of experiments carried out, one of the main takeaways was that VeriFlow checked 97.8% of the updates from a real BGP trace within one millisecond. Some updates took longer, however, because of the larger number of equivalence classes affected by new rules.

Sanjay Rao (Purdue University) asked about the coverage of error detection, the types of errors VeriFlow cannot detect. Ahmed argued that if the problem is visible at the data plane, it can be detected. Rao continued by asking about errors that span to multiple devices. Ahmed replied that yes, these type of errors can be captured as well, since VeriFlow has a global view of the network. Yan Chen (Northwestern University) asked whether VeriFlow is applicable to inter-domain routing. Ahmed replied that yes, VeriFlow can be used in this context as long as the controller is able to receive reports about changes. He also mentioned that the accuracy of the detection depends on receiving updates of network change in real time. Masoud Moshref (University of Southern California) asked whether one can reorder rules in order to improve VeriFlow's performance. Ahmed answered that they have not looked at this option yet, but they want to investigate it. He also said that as VeriFlow targets real-time processing of the

stream of rules coming from the controller, it may not have the liberty to reorder those. Takeru Inoue (Japan Science and Technology Agency) asked about VeriFlow's memory requirements for verification. Ahmed replied it is expensive; for a BGP experiment shown in the evaluation section, VeriFlow took 6 GB of memory.

### *Software Defined Traffic Measurement with OpenSketch*

Minlan Yu, University of Southern California; Lavanya Jose, Princeton University; Rui Miao, University of Southern California.

Lavanya Jose started her talk enumerating some questions that existing measurement technologies are either unable to answer, or would require a prohibitive amount of resources to do so: who is sending a lot of packets to a specific subnet? how are flow sizes distributed? is there anyone doing a port scan? etc. NetFlow cannot answer many of the questions posed. For example, NetFlow typically does not sample light flows, such as port-scanning flows. Increasing sampling rate is an option, but then it becomes resource-consuming. Streaming algorithms could be used as well, though each algorithm answers one question only.

Given the above, the question now is: what measurement architecture can answer all the questions? The answer is OpenSketch. OpenSketch is a software-defined traffic measurement architecture that separates the measurement control and data-plane functions, and uses sketches (Count Min Sketch) as building blocks to provide a generic and efficient measurement API. Lavanya discussed the basics of sketches, highlighting the tradeoff between memory consumption and accuracy of the resulting measures. The error can be estimated, which thus can indicate the degree of confidence one can have in the accuracy of the obtained measurements.

There is an issue with sketches, however: each one can estimate only one function. To solve this, Lavanya indicated a solution based on a three-stage pipeline that can support many sketches. The first stage in this pipeline is to hash the packet (based on the header), then classify it (based on the packet header and hash values), and finally update a set of counters based on the results of the previous stages. This pipeline can be configured in the controller plane in order to obtain the required measures and implement the measurement tasks to solve the questions initially posed. Lavanya then discussed possible strategies for implementing sketches with the pipeline, how one can provision the pipeline so that one can implement multiple, different sketches, and the evaluation results. The main takeaway? OpenSketch truly adheres to the SDN philosophy: separate the measurement control and data-plane functions, and make measurement in switches efficient and easy.

After the talk, Dejan Kosti (Institute IMDEA Networks) asked about the possibility of achieving throughput of 10 Gbps. Lavanya said it is possible, but sequentially updating the SRAM might become a bottleneck for tasks that update many counters per-packet. Dejan then asked about the limitations of OpenSketch. Lavanya replied that the data plane is somewhat limited so that OpenSketch can be made simple enough to implement with commodity hardware and operate at line rate. For example, some sketches cannot be implemented as they use more complex data structures (such as binary trees or heaps) not provided by the data plane.

## Pervasive Computing

*Summarized by Scott J. Krieder (skrieder@iit.edu)*

### *V-edge: Fast Self-Constructive Power Modeling of Smartphones Based on Battery Voltage Dynamics*

Fengyuan Xu, College of William and Mary; Yunxin Liu, Microsoft Research Asia; Qun Li, College of William and Mary; Yongguang Zhang, Microsoft Research Asia

Fengyuan Xu presented V-edge, a joint work aiming to improve the accuracy of battery prediction technologies. The idea is to consider power consumption and current system activities to accurately predict power runtime. By taking a snapshot of system activities the authors can determine CPU usage, backlight settings, and provide a calculation of power usage. A power model for smartphones needs to be an abstract concept, applying to the many different phone developers. Two of the most common ways to measure power consumption is through external metering and self-metering. This work uses self-monitoring, but applies unique algorithms for a fast and accurate measuring. The authors capture instantaneous current changes that lead to instantaneous output voltages. V-edge offers the advantages of accuracy and stability, and it's been tested on eight different batteries from two different smartphones. The reading is fast, as fast as the battery can provide an update rate. The V-edge solution consists of a training stage and an estimation stage. V-edge consists of an event-driven design, which provides a low overhead. After their power profiler runs, you will know which application consumes how much power and where.

Man Dong (Samsung Electronics) asked about the resistance in the battery pack, which is sensitive to temperature. Fengyuan Xu replied that in this case you don't need to know the absolute, as they work with the relative changes. Srimat Chakradhar (NEC Labs) asked that if they are sampling at one hertz, don't they miss quick events that only take a few milliseconds but happen with high frequency? Xu answered that V-edge only needs to provide power information during the training state. The event-driven design is quite accurate, comparable to fine-grained monitoring.

### *eDoctor: Automatically Diagnosing Abnormal Battery Drain Issues on Smartphones*

Xiao Ma, University of Illinois at Urbana-Champaign and University of California, San Diego; Peng Huang and Xinxin Jin, University of California, San Diego; Pei Wang, Peking University; Soyeon Park, Dongcai Shen, Yuanyuan Zhou, Lawrence K. Saul, and Geoffrey M. Voelker, University of California, San Diego"

Xiao Ma explained that the motivation for this work is abnormal battery drain (ABD). ABD is a condition on smartphone devices where the battery begins to drain at a significant rate without the authorization of the smartphone user. The authors developed an application called eDoctor which can identify the issue and suggest a solution to the user. ABD is often caused by an application update or software change, and eDoctor uses snapshots and installation logs to identify recent changes on the system. The application then suggests which apps to revert or remove to shift the battery drain back to normal consumption.

During a user study with 31 volunteers and 50 cases, the application was able to diagnose 47 of the cases accurately. This user study consisted of 6,000 device hours where a hidden bug was added to a user device. The bug was then activated and the users needed to use eDoctor to diagnose the problem.

Weverton Cordeiro asked whether they could identify bugs caused by updates to the Android OS system version. Ma said that they avoided telling users that a system update was the problem, as that is hard to undo. Cordeiro than asked about battery failure. Ma answered that most problems are software, and they can't detect hardware failures. Someone from Northeastern asked whether collecting data was required and was user privacy a concern. Ma said that they don't observe what the user does in an app, just the power utilized by an app. Also, everything is local on the phone.

### *ArrayTrack: A Fine-Grained Indoor Location System*

Jie Xiong and Kyle Jamieson, University College London

ArrayTrack is an indoor tracking system that can calculate a position down to a 23 cm level of accuracy when using access points (APs) with eight antennas. The motivation for this work is that GPS does not work indoors, and even in cases when it does, provides accuracy at the level of meters. Some future solutions include augmented reality on smartphones, or wearable glasses. The use cases for such a technology include trying to find a book in the library, an item in a supermarket, or a piece of art in a museum. Many works have tried to use WiFi, Infrared, and other technologies over the years, but none are able to calculate with the level of accuracy that ArrayTrack provides. The theory of operation requires there to be multiple antennas and radios at each AP. By locating a device and its rate of movement from the AP, the authors can calculate the angle of arrival from the device. The authors then calculate the distance based on the rate of arrival. Another benefit of ArrayTrack is that it can accurately calculate position and distance based on a single packet. Future work includes conducting studies on how the height of APs and clients affects results.

Phil Levis (Stanford) asked how many antennas the client has and what effect does MIMO have? Jie Xiong answered that they assume the client has only one antenna, and if the client has more than one antenna there will appear to be multiple locations. For MIMO (multiple input, multiple output), they did the processing at the MIMO site, and there they needed multiple antennas. Keith Winstein (MIT) asked whether they needed to know the exact location and orientation of each AP. Xiong replied that it did make a difference with linear-oriented antennas. If they used a circular array of antennas, they could avoid concern with orientation. Winstein then wondered whether they could bootstrap to find the orientation of the linear arrays, and Xiong said that he thought that might work.

### *Walkie-Markie: Indoor Pathway Mapping Made Easy*

Guobin Shen, Zhuo Chen, Peichao Zhang, Thomas Moscibroda, and Yongguang Zhang, Microsoft Research Asia

Guobin (Jacky) Shen explained that their research used WiFi client devices to generate maps of the given location. The motivation for their work is that mapping applications often assume that the map is predetermined or given. But in many cases this is not possible.

Consider the problem of generating electronic maps for a large number of buildings. Neither hiring someone to do this manually or requesting the floor plans for every building is realistic and both would be prohibitively expensive. Thus the authors set out to calculate maps dynamically based on where the WiFi client traffic was coming from throughout a building. The system then generates digital landmarks based on high traffic areas and connects these landmarks based on how client traffic traverses a building. Some challenges of this approach include noisy traffic that causes difficulty in the tracking data, the difficulty of tracking multiple users at a given time, and the need to handle user diversity as well as device diversity. The landmarks are easy to determine, but they simply need to be determined by the device not by the user.

The authors calculated maps by having users walk in a space for 20, 30, 50, and 100 minutes. This is where the work gets its name; the idea is that by walking you mark locations with WiFi landmarks. The maps improve with time, and future work is aimed to reduce the time it takes to collect an accurate map. One drawback to this approach is that the maps are

limited to where the clients are walking, and some areas such as closets or some routes that clients do not walk on will not be included in the map. Future work also seeks to improve data captures from lifts and stairwells.

Phil Levis, the session chair, pointed out that signal strengths vary over space. Shen replied that they don't need WiFi marks everywhere as they can still run their algorithms. Steve Tarzen (Vaporstream) asked whether random walks change the problem. Shen said that they split the readings into one-minute segments, and even with random walks, this works as long as they got data from a lot of users. Someone asked whether building materials could cause problems. For example, hospitals have lots of metal objects. Shen replied that it was the stability of the environment that was the key to their system working. Moving objects would affect the measurements, though.

## Network Integrity

Summarized by Arpit Gupta (agupta13@ncsu.edu)

### *Real Time Network Policy Checking Using Header Space Analysis*

Peyman Kazemian, Michael Chang, and Hongyi Zeng, Stanford University; George Varghese, University of California, San Diego and Microsoft Research; Nick McKeown, Stanford University; Scott Whyte, Google Inc.

Peyman started by discussing the importance of policy checking. Network state changes all the time, rules keep on changing, and, as a result, there is the potential for policy violations. Thus it is important to have policy checkers that are real-time for the entire network. Kazemian proposed NetPlumber, which improves on Header Space Analysis (NSDI '12) by enabling real-time checking of updates. NetPlumber is suited for SDN as it can tap into the centralized control data to perform real-time policy checking. Also, NetPlumber can be used for legacy implementations.

The heart of NetPlumber is the plumbing graph, which captures all possible paths of flows through the network. Nodes in the graph correspond to the rules in the network, and directed edges represent the next-hop dependency of these rules. This graph is a perfect tool to check policy in a network, react to violations, evaluate complex policies, etc. Peyman also talked about Flow Exp, which is a regular-expression-like language, to verify policies in complex networks. The work was evaluated over three real-world networks: Google WAN, Stanford's backbone network, and the Internet2 nationwide network. Rule-update verifications are on the order of sub-milliseconds, whereas link updates are a few milliseconds.

Someone asked Kazemian to compare/contrast NetPlumber with VeriFlow. Kazemian replied that performance-wise they're equivalent, as NetPlumber can handle any sort of flow and is independent of where wild cards are used. NetPlumber has more of an agnostic approach and is more generic. Though NetPlumber can't verify performance, it can check for loops, blackholes, etc., but not throughput guarantee. Ethan Katz-Bassett, the session chair, asked about the possibility of using NetPlumber for non-SDN networks. Kazemian said that you can use NetPlumber, but it will be using a snapshot and might miss things.

### *Ensuring Connectivity via Data Plane Mechanisms*

Junda Liu, Google Inc.; Aurojit Panda, University of California, Berkeley; Ankit Singla and Brighten Godfrey, University of Illinois at Urbana-Champaign; Michael Schapira, Hebrew University; Scott Shenker, University of California, Berkeley and International Computer Science Institute

Aurojit Panda started with the difference in time-scale of operations for control and data planes. Control-plane response to link failures is slow, and current solutions rely on precomputed backup paths. Such backup paths make sense for single link failures and are hard to generalize for multiple-link failures.

Panda suggested that the question to ask is whether we can push this to data plane. Such a solution will be impossible if we put constraints like no FIB (Forwarding Information Base) changes at packet rate and/or no additional data in packet headers. Their approach is to relax a few constraints—for example, allow changes to a few bits in the FIB at packet rates. Their solution is to take advantage of redundancy by extending routing tables with other paths to a destination, and to restore connectivity at data speeds using a strategy of reverse reconnect. Using reverse to reconnect means to start at the disconnected node to attempt to rebuild the DAG (Directed Acyclic Graph). Enabling reversals in the data plane means two challenges must be addressed, namely lost or delayed notification. A safe control plane is proposed,, which should not interfere with the data plane.

They evaluated their solution on WAN and datacenter topologies over NS3 to test for stretch, throughput, and latency. They also analyzed the effect of FIB update delays on latency and throughput, and end-to-end benefits of using DDC.

Omar Javed (University of Washington) asked how the concept of storing multiple links is different from the multiple path concept. Also, was it possible to enable link reversal for the wider Internet, where complex business agreements exist? Will it work for inter AS? Panda replied that link reversal is different from multipath as it is less restrictive. He also explained that current work focuses only on intra-domain routing and considers only a single AS.

Michael Freedman (Princeton) asked how to characterize the number of bits to change. Is the value three referred to

in the paper a hard limit, and why was a higher value not chosen? Panda replied that the current algorithm is a simple one and he is not sure whether higher values will work as expected.

### *Juggling the Jigsaw: Towards Automated Problem Inference from Network Trouble Tickets*

Rahul Potharaju, Purdue University; Navendu Jain, Microsoft Research; Cristina Nita-Rotaru, Purdue University

Rahul Potharaju explained that trouble tickets for network management are common and fixing them as soon as possible is currently the prime focus. What is missing is how to learn from such mistakes/problems/issues.

The main goal of this work is to analyze these tickets and extract three key features for summarization: problem, activities, and action. The information in a ticket has structured fields and free text. The goal is to use the free text to extract features. This choice is based on learning that structured field data is coarse grained and does not provide much information. A strawman approach of applying natural language processing (NLP) does not work because it is suited for well-written texts and ignores context. Their solution, NetSieve, combines NLP with semantics. In this context, knowledge is like a dictionary that needs to be built first before they choose to infer semantics for a ticket. It involves three steps: (1) repeated phrase extraction, i.e., extracting *n*-grams from words' pool-set, trading completeness for scalability; (2) knowledge discovery, which applies a pipeline of linguistic filters to determine domain-specific patterns; and (3) ontology modeling used to determine semantic interpretations. The evaluation was done for two standard metrics of accuracy percentage and F-score.

Yan Chen (Northwestern) asked about the status of the source code release because he envisioned using these techniques in other domains too, in analyzing security logs, for example. Rahul briefly replied that the IP for this project is owned by MSR, and there was a burst of laughter in the conference hall. Ahmed Khurshid (UIUC) asked what happens in cases where two errors are related to each other. Rahul replied that they found that such correlations showed improvements and are shown in use cases in the paper.

Abhishek Sharma (NEC) asked why the authors used area experts, and were other data mining (unsupervised) techniques used to compare results. Rahul affirmed exploration in that direction. He said they wanted to take a different approach with a semi-supervised approach; being domain-specific also helped in terms of performance.

## Data Centers

*Summarized by Imranul Hoque (ihoque2@illinois.edu)*

### *Yank: Enabling Green Data Centers to Pull the Plug*

Rahul Singh, David Irwin, and Prashant Shenoy, University of Massachusetts Amherst; K.K. Ramakrishnan, AT&T Labs—Research

Rahul Singh started his presentation on Yank by mentioning that applications hosted in modern datacenters assume always available stable servers. In order to guarantee server availability, datacenters employ a highly redundant power infrastructure, which is expensive. So applications often relax this strict stability assumption and compensate for it using low-cost high availability techniques in software. Singh then introduced a new abstraction of transient servers, which, unlike stable servers, have unpredictable availability; however, these servers receive advance warning either from the software (Amazon spot instances) or from the hardware (UPS units) prior to termination. Yank enables datacenters to use the mix of stable and transient servers transparently while maintaining application availability.

Singh presented two ways of supporting transient servers: by modifying individual applications and by providing system support. Yank adopts the latter approach because the former can be challenging for certain classes of applications. Upon receiving a warning, a transient server transfers its VMs to a stable server to ensure that the application is always available. The transfer has to be completed within the warning period to ensure that no state is lost. Singh mentioned two strawman approaches for transferring VM states: the live migration approach, which has low overhead but requires a large warning period, and the backup VM high availability approach, which supports low warning time but incurs high overhead. These two approaches fall on two ends of a spectrum. Singh pointed out that Yank covers the entire spectrum by adapting to the warning time. When the warning time is low, Yank is similar to the high availability approach. On the other hand, when the warning time is high, Yank behaves like the live migration approach. Singh then presented the high-level design of Yank, which consists of a snapshot manager that runs at each transient server and is responsible for sending VM states to backup servers; a backup engine that runs at each backup server and is responsible for storing multiple transient servers' snapshot; and a restoration service that runs at stable servers and is responsible for restoring VMs when the transient servers receive warnings.

Singh then presented their evaluation of Yank. First, he showed that when the warning time increases from five seconds to 20 seconds, the amount of data transferred from the transient to backup servers reduces by a factor of 70. This is because, with a higher warning time, the entire state of a program is small enough to be transferred after receiving the warning signal. Second, he showed that a five second

warning time is sufficient to bring down the client-perceived response time by a factor of 20. He also showed that a 4 GB backup server can support 15 transient VMs and claimed that powerful backup servers will be able to support hundreds of VMs. Finally, Singh showed that transition from stable servers to transient servers and vice-versa do not have any visible difference on the response time—thus, he claimed that Yank masks applications from transiency due to changing power availability.

Rajesh Nishtala (Facebook) asked what percentage of an application's memory is needed to reconstruct a live instance and why Yank moves states as opposed to reconstructing states, which can be done in a shorter time period. Singh replied that their experiments showed that the state of an application is much smaller than the total allocated memory. So their design is guided by this observation of small working sets. Dave Andersen (Carnegie Mellon University) asked about the benefit of Yank in terms of energy saving or carbon footprint reduction, because Yank assumes a grid power supply in addition to the renewable power supply. Singh answered that a datacenter can leverage the on-peak and off-peak electricity pricing in order to decide when to switch to renewable energy and back. For example, the energy cost can be reduced if, during an on-peak hour, all the workloads can be moved to transient servers. Andersen was not satisfied with the answer and said that he would continue the discussion later. He then asked whether the same techniques could be used to ensure VM migration within a bounded time. Singh answered affirmatively by saying that Yank gives users a knob to change the warning period and thus reduce the overhead of maintaining VM backups.

### *Scalable Rule Management for Data Centers*

Masoud Moshref and Minlan Yu, University of Southern California; Abhishek Sharma, University of Southern California and NEC Labs America; Ramesh Govindan, University of Southern California

Masoud Moshref started his presentation on vCRIB by pointing out that datacenters use rules to implement management policies. Rules are saved on predefined fixed machines (hypervisors, switches, etc.). On one hand, machines have limited resources (i.e., they can support a limited of rules). On the other hand, future datacenters will have many fine-grained rules ranging from millions to billions of rules. So, it is necessary to offload rules, which creates a trade-off between resource and bandwidth usage. Moshref then pointed out some challenges of rule offloading. First, an offload scheme must preserve the semantics of overlapping rules. Second, the scheme must respect resource constraints. Third, the scheme must minimize traffic overhead. Fourth, and finally, the scheme must handle dynamics of traffic and rule changes and VM migration.

Next, Moshref presented the design of vCRIB, which is a virtual cloud rule information base. vCRIB provides a proactive rule placement abstraction layer to the operator. The input to vCRIB is a set of rules, and the output of vCRIB is a minimum-traffic feasible-placement. Moshref then pointed out how vCRIB addresses the challenges of rule-placement mentioned earlier in the talk. First, in order to ensure that the semantics of overlapping rules are unchanged, vCRIB uses a source partitioning with replication approach. Second, it uses a resource-aware placement algorithm known as First Fit Decreasing Similarity (FFDS) to find feasible placement of the rules. Third, vCRIB minimizes traffic overhead by refining the feasible placement using a traffic-aware refinement approach. Finally, vCRIB handles dynamism by re-running both the placement and refinement steps if the dynamics converted the feasible placement into an infeasible one. In the case when the placement is still feasible, vCRIB only re-runs the refinement step.

Finally, Moshref presented an evaluation of the vCRIB system. He compared vCRIB against source-placement, in which rules are saved at the traffic source. His simulation results revealed that vCRIB found low traffic feasible solutions. Additionally, adding more resources helps vCRIB further reduce traffic overhead. Moshref concluded his talk by mentioning several future works, which included supporting reactive placement of rules, splitting partitions when the number of rules becomes large, and testing for other rule sets.

Michael Piatek, the session chair, asked whether there is a way to limit the number of rule changes that happen at any point of time, because drastic rule changes may cause problems. Moshref replied that any such constraints will have to be set in the refinement algorithm by setting the appropriate budget. He also mentioned that it can be a good future step. Piatek followed up by asking whether setting a constraint will prevent the algorithm from finding a feasible placement. Moshref asked Piatek whether he was concerned about finding a feasible placement as opposed to refining for minimum traffic overhead. Piatek confirmed this and Moshref replied that in his algorithm he does not consider previous placements in order to find future placements; however, similar algorithms can be found for virtual machine placement and can be adopted for this scenario. Finally, Piatek asked what was causing the rule explosion in datacenters. Moshref replied this was happening because of the growing scale of datacenters and because of the way policies were written to regulate traffic within a datacenter (e.g., between pairs of virtual machines within a datacenter).

### *Chatty Tenants and the Cloud Network Sharing Problem*

Hitesh Ballani, Keon Jang, Thomas Karagiannis, Microsoft Research, Cambridge; Changhoon Kim, Windows Azure; Dinan Gunawardena and Greg O'Shea, Microsoft Research, Cambridge

Keon Jang focused on how to share the network in multi-tenant datacenters. He pointed out that multi-tenant datacenters consist of both intra-tenant (VM-to-VM) and inter-tenant (VM-to-storage) traffic. Because the network is shared by multiple tenants, network performance of tenants is interdependent. Jang mentioned three requirements for network sharing: a minimum bandwidth guarantee, upper-bound proportionality, and high utilization. Jang pointed out that no prior work satisfies all three requirements—they focus on intra-tenant traffic only; however, inter-tenant traffic accounts for 10–35% of the overall traffic. Additionally, guaranteeing minimum bandwidth and ensuring proportionality for inter-tenant traffic is harder.

Jang then presented an overview of Hadrian, which satisfies the above three requirements. Hadrian uses a hierarchical hose model. In this model, tenants can separately specify inter-tenant bandwidth requirement and communication dependency. This guides the placement of VMs across the datacenter. Hadrian uses a hose-compliant bandwidth allocation scheme, which ensures upper-bound proportionality and provides minimum bandwidth guarantees.

Jang then presented an evaluation of Hadrian through real-world deployments as well as large scale simulation experiments. He showed that job completion time is 3.6x faster using Hadrian. This is attributed to the better and predictable network performance as well as efficient resource utilization offered by Hadrian. His results also verified that Hadrian satisfies upper-bound proportionality. Finally, Jang pointed out that a Hadrian cluster accepts 37% more jobs compared to a non-Hadrian cluster. Thus, Hadrian enables providers to offer improved service at a lower price.

Rajesh Nishtala (Facebook) asked Jang to comment on the multi-tenancy of low-bandwidth, low-latency applications and high-bandwidth, high-latency tolerant applications coexisting in a single environment in the Hadrian model. Jang replied that latency is out of scope of their work; however, bandwidth reservation prevents the network from getting over-utilized, so latency remains low. Sanjay Rao (Purdue University) asked how easy it is to predict the inter-tenant bandwidth and whether it changes a lot over time. Jang replied that it would depend on the application. For example, in MapReduce, where the input data size is known a priori and where users can reason about the job deadline, bandwidth prediction is easy. Rao wanted to know whether they make the assumption that the bandwidth is specified by tenants. Jang answered affirmatively. Rao then queried about the complexity of dependencies among tenants and whether Hadrian uses the dependency relationships among tenants to make placement decisions. Jung replied that tenants only specify their total bandwidth requirements and Hadrian's placement decisions are solely guided by these bandwidth requirements, not by the dependency relationships. Michael Piatek (Google) commented that in case of all-to-all communication the placement problem is tricky. This is specifically true for services like storage providers. He asked whether providing some hint to the placement algorithm to make better placement decisions in case of services is possible. Jung replied that in this work they did not consider this approach but certainly these hints can help the placement engine to make better decisions.

### *Effective Straggler Mitigation: Attack of the Clones*

Ganesh Ananthanarayanan, Ali Ghodsi, Scott Shenker, and Ion Stoica, University of California, Berkeley

Ganesh Ananthanarayanan presented his work on effective straggler mitigation in the case of interactive data analytics. He pointed out that interactive jobs are common in today's clusters, and the number of interactive jobs is expected to grow further. These jobs are small in size, and low latency is crucial. For example, in Facebook's Hadoop cluster, 88% jobs operate on 20 GB of data and contain fewer than 50 tasks. These interactive jobs are sensitive to stragglers. Existing straggler mitigation techniques, such as blacklisting and speculation, are ineffective for these small jobs (6x–8x slower compared to the median task size).

Ananthanarayanan proposed proactively launching multiple clones of a job and picking the results from the earliest clone. This approach probabilistically mitigates stragglers. Because most of the small jobs use only a small fraction of resources in a cluster, to clone small jobs with only a few extra resources is feasible; however, cloning creates I/O contention. To avoid the contention, every clone should get its own copy of data. In MapReduce this data may either be the input data (replicated) or the intermediate data (not replicated). Ananthanarayanan considered the harder case of intermediate data and showed that job-level cloning is not effective in mitigating stragglers with a small number of clones; however, task-level cloning can solve the problem by using only a few (3) clones. Next, he presented two schemes for contention avoidance and straggler mitigation for intermediate data: contention-avoidance cloning (CAC) and contention-cloning (CC). He showed that CAC avoids contentions but increases vulnerability to stragglers. On the other hand, CC mitigates stragglers but creates contentions. He proposed solving the problem associated with CAC and CC by an approach called "delay assignment," where a small delay is assigned to get an exclusive copy before contending for the available copy. He

also mentioned that jobs are cloned only if sufficient budget (resource) is available.

Finally, Ananthanarayanan presented evaluations of his system (called Dolly) by using workloads from Facebook and Bing traces. He compared Dolly against two prior approaches called LATE and Mantri. His experiments showed that jobs are 44% and 42% faster compared to LATE and Mantri, respectively. Additionally, the slowest task is only 1.06x slower compared to the median task (down from 8x). He also showed that the proposed delay assignment technique is critical for achieving better performance, and with the increasing number of phases in jobs, the benefit of this technique increases further.

Christopher Stewart (Ohio State University) asked how a small job was defined in terms of the number of tasks in that job. Ananthanarayanan replied that they avoided defining and specifying which jobs were small. This notion was captured by the cloning budget. A specific number of clones for a job are created as long as resources are available. Stewart followed up by asking whether there was any insight on how to set the cloning budget. Ananthanarayanan mentioned that it would require sensitivity analysis, which is mentioned in the paper. The cloning budget depends on the knee of the power-law curve of the job size in a specific workload. Sanjay Rao (Purdue University) asked whether cloning could backfire, i.e., whether the proposed techniques depended on the fact that stragglers were random and there was no correlation between them. Ananthanarayanan replied that cloning may not be effective in case of data skew (i.e., when one task has more data to process than the other). Rao then refined his question by asking whether straggler nodes were correlated or not. Ananthanarayanan replied that according to their observation, stragglers were uncorrelated. For this reason, techniques such as speculation were effective; however, what Rao suggested might be interesting in order to make placement decisions for speculative clones. If a specific rack is known to be faulty, then avoiding that rack might be better. They did not explore these cases in their work.

## Substrate

*Summarized by Joshua B. Leners (leners@cs.utexas.edu)*

### *Wire Speed Name Lookup: A GPU-Based Approach*

Yi Wang, Tsinghua University; Yuan Zu, University of Science and Technology of China; Ting Zhang, Tsinghua University; Kunyang Peng and Qunfeng Dong, University of Science and Technology of China; Bin Liu, Wei Meng, and Huicheng Dai, Tsinghua University; Xin Tian and Zhonghu Xu, University of Science and Technology of China; Hao Wu, Tsinghua University; Di Yang, University of Science and Technology of China

Bin Liu explained that name lookup is an important problem, both in the context of networking and in domains outside of networking. This paper focuses on a particular use case of name lookup: routing in a content-centric network. Content-centric networking uses hierarchical names (e.g., /com/google/maps) of arbitrary length rather than fixed-length addresses to route packets. To be useful, content-centric networking needs a lookup mechanism that supports longest prefix matching, has high throughput (saturates a 100 Gbps Ethernet link), and low latency (100 microseconds per lookup).

Satisfying these constraints requires a carefully designed data structure that can work efficiently with GPU hardware. Some straightforward approaches won't work: character tries and state transition tables require too much memory. Aligned transition arrays (ATAs) can greatly compact the information of a transition table, but they don't support incremental update and are still inefficient (each character must be looked up separately). To address both of these concerns, the authors implemented multi-ATAs, which can use multiple characters in each transition and support incremental update.

There are some challenges to implementing multi-ATAs on a hybrid CPU-GPU platform: the PCI-e bus and GPU processor are limiting factors in achieving both high-throughput and low latency. Two techniques are used to improve performance. First, pipelining lookup and data transfer improves PCI-e bus and GPU processor utilization. Second, interweaving the memory layout of the input names in GPU memory reduces the memory accesses of threads, improving performance.

The implementation of the multi-ATA data structure performs well on the CPU-GPU platform. Using the multi-ATA requires two orders of magnitude less space than a baseline state transition table. The techniques to use the GPU efficiently allow up to ~70 million searches per second, an order of magnitude more than a state transition table. Furthermore, these techniques can saturate a 100 Gbps link with latencies of less than 100 microseconds per search.

Dong Zhou (Carnegie Mellon University) pointed out that in their evaluation, they worked only on a local machine but didn't actually transfer data onto a NIC. Zhou wondered whether this was a fair comparison. Bin Liu replied that they only worked within the context of a single machine and that they were looking at using the NIC in future work. Because using the NIC would take additional CPU cycles, Zhou then wondered, would competing with the CPU affect their results? Bin Liu replied that this was something they needed to address but thought that other hardware acceleration could help. Srimat Chakradhar (NEC Labs) wondered if the state tables get larger, would they still fit in the GPU. Bin Liu said that they currently had a 10 million-entry table on the GPU, which uses about 1/3 of the GPU's external memory

for a single GPU processor chip, and they had two processor chips on the GTX590 board. They estimated they could keep a 60 million-entry table on this kind of GPU. Newer GPUs had more space, and they thought they could keep a 120 million-entry table on the new hardware.

Michael Freedman (Princeton University) noticed that in the multi-ATA table it looked like collisions map into a second table. This seemed to imply that lookups would require multiple accesses. Bin Liu said that it didn't, and they have a proof in the paper. Freedman then asked whether they considered other hashing algorithms, such as cuckoo hashing, and Bin Liu said that they use a much simpler lookup algorithm. Gun Sirer (Cornell) suggested that they should be mining BitCoins with his GPUs, since, currently, their approach lacked security, as in self-verifying names. Lack of security as a first-class primitive plagues DNS today. Bin Liu said that security is future work.

### *SoNIC: Precise Realtime Software Access and Control of Wired Networks*

Ki Suh Lee, Han Wang, and Hakim Weatherspoon, Cornell University

Ki Suh Lee said that measuring and controlling interpacket delays can enable better understanding of network characteristics and new applications. For example, measuring these interpacket delays can give better characterization of network traffic, and controlling the interpacket delays can create new covert channels. Unfortunately, current techniques are insufficient to measure these interpacket delays precisely; however, the precision of network measurements could be improved with access to the PHY layer: counting the number of PHY layer idle characters between packets can give network research applications sub-nanosecond precision.

Manipulating idle characters requires accessing the PHY, which is currently a black box that hides information (including idle character counts). One approach could use something like BiFocals, which uses physics equipment to measure the PHY layer. Unfortunately, this equipment is expensive ($500,000) and only works offline. Because there is limited access to hardware, the authors propose using software to count idle characters.

The authors implement their approach as a new platform called SoNIC, which ports some of the PHY layer functionality of a 10 Gbps Ethernet link into software. Specifically, SoNIC ports all of the functionality that manipulates bits into software (everything above the scrambler of the physical coding sublayer), but keeps the functions that translate bits into signals into hardware. This split requires high-performance software, which SoNIC implements using three techniques: (1) SoNIC software threads are pinned to an individual CPU core; (2) polling and an optimized DMA, rather than interrupts, are used to interface with the network device; and (3) software is tightly optimized (e.g., by replacing loops with bitwise operators).

These techniques give SoNIC precise measurement and control of interpacket delay and interpacket gaps, which allowed the authors to implement several functions that were previously impossible (including a new timing channel with data rates up to 250 Kbps that is undetectable by current techniques).

Dong Zhou( Carnegie Mellon University) asked whether there are any limitations to applications they can support, because SoNIC appears to require a lot of CPU. Lee replied that the applications must be able to work with data faster than 10 Gbps using the remaining resources of the system. Nikita Borisov (University of Illinois Urbana-Champaign) thought that it's cool that their timing channel attack works even if there are unmodified routers along the path, and wondered whether they had considered cross traffic while evaluating this attack. Lee said that they had, but their paper uses only unloaded routers to demonstrate feasibility. Hongyi Zeng (Stanford University) wondered about the requirements for an FPGA to implement SoNIC. Lee replied that the FPGA must have transceivers that can support more than 10.3 Gbps. Zeng then asked about the use of a CDF graph that showed variations in the interpacket gap, wondering why hardware would exhibit these variations. Lee said that SoNIC has errors because it's timestamping within the network stack, and within the kernel there's a lot of overhead: other tasks, interrupts, etc. Zeng asked about hardware timestamps, and Lee replied that hardware clocks have lower resolution than their techniques. What's cool about SoNIC is that they get really precise timing from counting the idle characters. Junda Liu (Google) noticed that every port had five dedicated kernel threads, and asked whether that required five cores. Lee answered yes, that they pinned each thread to its own core, but the other cores were shared. More CPU-intensive applications (e.g., full packet capture that requires disk access) are impossible right now because of application requirements (must handle > 10 Gbps of data).

### *Split/Merge: System Support for Elastic Execution in Virtual Middleboxes*

Shriram Rajagopalan, IBM T. J. Watson Research Center and University of British Columbia; Dan Williams and Hani Jamjoom, IBM T. J. Watson Research Center; Andrew Warfield, University of British Columbia

Shriram Rajagopalan noted that elasticity, the ability to scale a Web service dynamically to meet demand, has been well-studied in the context of Web applications; however, these applications often depend on middleboxes, such as firewalls, intrusion detection systems, and load balancers, which are not well suited to dynamic scaling, because they maintain

state. Because middleboxes are hard to provision dynamically, scalable Web services over-provision their middleboxes or stop using them entirely.

An important insight is that middleboxes are flow-oriented: most processing deals with a single flow in isolation, and that the state associated with these flows is partitionable. Using this insight, the authors classified middlebox state into three categories: partitionable (e.g., flow state), coherent (e.g., counters), and ephemeral, state that is local to a middlebox instance (caches, etc.).

To leverage this insight, the authors implemented FreeFlow, a VMM runtime for middleboxes that can dynamically provision middleboxes. To use FreeFlow, a middlebox developer must annotate the middlebox's state as partitionable, coherent, or ephemeral. At runtime, FreeFlow uses OpenFlow to migrate flows (and their state) correctly to dynamically allocated middleboxes, without disrupting existing traffic. To prevent coherent state from becoming a bottleneck, FreeFlow uses looser consistency semantics for keeping such state in sync. The lessened consistency is not problematic for most coherent state, such as counters for reporting statistics, monitoring thresholds, etc.

Using its Split mechanism, FreeFlow is able to keep latency low by dynamically allocating new middleboxes in the face of increased load. FreeFlow also keeps utilization high with its Merge mechanism: combining middlebox replicas when load decreases. FreeFlow has end-to-end benefits with existing middleboxes. For example, FreeFlow allows Bro, an intrusion detection system, to scale dynamically, performing as well as over-provisioning, with only a minor performance hit after a burst in load.

Masoud Mosharef (USC) asked whether they assumed that the flow entries were not dependent. Rajagopalan replied that they assumed that flows were independent. Mosharef then asked whether they had any suggestions for selecting middleboxes (e.g., to balance network traffic). Rajagopalan answered that implementing policy on top of their mechanism is future work. Anthony Nicholson (Google) pointed out that their implementation requires modifying code, and wondered whether FreeFlow could be made to work with unmodified middleboxes. Rajagopalan replied that, currently, modifying the code is necessary. The idea is that people are already moving away from custom hardware middleboxes to software-based middleboxes that can be deployed and scaled in the cloud. We have the opportunity to get things right the first time by designing these middleboxes properly. Srimat Chakradhar (NEC) asked whether they were assuming locking across middle boxes for coherent state. Rajagopalan answered yes, if the coherent state requires strong consistency. Yan Chen (Northwestern) pointed out that in Bro, there are multi-dependent flows and wondered whether they tried to evaluate the state explosion in keeping these flows together. Rajagopalan said that there is no state explosion as such. There are two ways of tackling this scenario: partition state at larger granularity or abstract the dependencies into a coherent state and synchronize as needed. There is definitely a tradeoff in the granularity of partitioning vs. the ability to finely balance load. They didn't find many interflow dependencies in their evaluation, so there wasn't much problematic state.

Steve Tarzia (Vaporstream) asked why not make the middleboxes stateless, since they were already re-architecting them. Rajagopalan replied that stateless Web applications can afford to look up session state from external systems like databases or key-value stores because they handle thousands of requests per second; however, middleboxes handle millions of packets per second. The latency requirements on middleboxes make accessing an external database impractical.

## Wireless

*Summarized by Arpit Gupta (agupta13@ncsu.edu)*

### *PinPoint: Localizing Interfering Radios*

Kiran Joshi, Steven Hong, and Sachin Katti, Stanford University

Currently, interference is the major cause of poor performance for WiFi networks, and localizing these interfering devices is desirable. Most of the previous work in this area required extensive pre-deployment calibration. Steven Hong presented their solution, which requires minimum calibration and is built on top of existing AP architecture.

Steven also emphasized that this solution cannot only be used for locating interfering devices but also for location-based advertising, indoor navigation, real-life analytics, etc. PinPoint can differentiate between multiple interfering signals, compute line-of-sight angle-of-arrival (LoS AoA) in a non-line-of-sight (NLoS) multipath environment, and aggregate and process noisy data from APs. Computation of LoS AoA in an NLoS multipath environment with multiple antenna requires usage of angle of arrival for interference signals, but the presence of multipaths obscures such an approach. Steven than explained how angle of arrival estimation techniques work in general. Also, LoS detection naively won't work due to a NLoS scenario. Their solution is to use feature vectors and arrival time differences. LoS signals arrive first, enabling identification of LoS signal and AoA eventually.

For evaluations, they used AoA+ CSSI (cyclic signal strength indicator) information and compared it against performance of RSSI-based techniques and MUSIC. Pinpoint leverages existing WiFi infrastructure, provides a better algorithm for

LoS AoA estimates, and is capable of differentiating various interference sources. This makes PinPoint a better candidate for interference localization than existing solutions.

Sarthak Grover (Georgia Tech) asked about the CSSI approach, whether it is possible to differentiate between two WiFi signals. Steven replied that it works better if the interfering signals are of different protocols. In the case of two WiFi signals, CSSI information won't be important but AoA will surely be important. A researcher from UCL London said that MUSIC is not suited to indoor scenarios and asked whether theirs was a fair comparison. Steven admitted that the objective for MUSIC is different, and they do focus on the strongest multipath components.

### *SloMo: Downclocking WiFi Communication*

Feng Lu, Geoffrey M. Voelker, and Alex C. Snoeren, University of California, San Diego

Feng Lu started with statistical figures to emphasize the dominance of WiFi radios in consuming power for energy limited smart devices. He then explained how WiFi sleep works and about the focus of researchers to develop better sleep policies in recent years. Most apps are real time and chatty in nature. Feng explained that the data rates for such apps are small, but these apps stay connected more than 62% of the time. In order to identify opportunities to save energy, knowing where energy is spent is important. WiFi radio goes to idle before sleep, which is almost the amount of energy of the transmit (Tx) and receive (Rx) states. As apps require a smaller data rate, but available rates are higher, time spent in Tx/Rx is small and most of the energy is spent in idle state.

Their solution is to downclock the WiFi radio, saving 30–40% energy. Clock rate is gated by the sampling rate, which is higher following the Nyquist principle for WiFi signals. Recent advances in compressive sensing allows them to cheat when the information rate is much less than the signaling rate (11 times). The simple idea is to sample groups of chips rather than a single one, enabling downclocking for WiFi radio. The solution enables downclocking for both transmission and reception.

SloMo is implemented over the Sora (SDR) platform and doesn't require any modification to WiFi APs, with full backward compatibility. Evaluation reveals that there is not much difference for cases where SNR is good and when SNR is poorer. As an example, SloMo energy consumption for the Angry Birds app goes up for Tx and Rx, but significant energy savings are observed for idle times. Similarly, apps such as Skype benefit significantly. They observed that the increase in airtime is less than 13%, ensuring that SloMo is useful and relevant for energy saving.

Philip Levis from Stanford pointed out that this solution works fine at the link level but wondered how it's going to play out at the MAC level. For example, if the receiver is downclocked and AP is sending RTS/CTS, it is possible that the receiver won't be able to decode these RTS/CTS signals. Feng replied that if RTS/CTS are sent at 1 or 2 Mbps, decoding them easily is possible, but for other data rates we may not be able to decode correctly. For the cases when RTS/CTS was sent at 1–2 Mbps, SloMo was able to achieve 80–90% detection rate for low SNR values, so it is not much of a concern. Men Dong (Samsung Labs) was curious how the authors determined the energy breakdown in their experiments. Also, the 700 mW power consumption mentioned in earlier slides does not match the specifications for Qualcomm or TI WiFi chips. Feng explained that the WiFi node sends a null packet to the AP for sleep activities, and they used that knowledge to determine when the AP went to sleep and vice versa. Finally, they combined the power models based on real smartphone measurements to map power consumed for different modes of operation. The energy consumption values mentioned are for data transmissions/receptions. Arpit Gupta (NC State University) asked about scenarios in which multiple apps (data-intensive ones along with chatty ones) concurrently use the network interfaces; what should be the mechanism to switch downclocking ON/OFF for such scenarios? Feng claimed that apps do not actually use network interfaces concurrently, and users interact with one app at a time. Masoud (USC) asked how downclocking affects user experience or the performance of applications like Skype. Feng said that experiments were carried out with Skype and no perceivable impact was observed.

### *Splash: Fast Data Dissemination with Constructive Interference in Wireless Sensor Networks*

Manjunath Doddavenkatappa, Mun Choon Chan, and Ben Leong, National University of Singapore

Manjunath Doddavenkatappa talked about the importance of data dissemination and the critical nature of its completion time, which for existing protocols is of the order of a few minutes attributable to contention resolution. The proposed solution, Splash, eliminates the need for contentions and therefore minimizes completion time.

Manju explained how Splash eliminates the need for contention resolution through the use of constructive interference and channel diversity, covering the network with fast, parallel paths (tree pipelining). He further explained how Splash utilizes transmit density diversity, opportunistic overhearing, channel cycling, and XOR coding techniques to strengthen reliability. He emphasized that any missing data is recovered locally and the fact that 90% of nodes have full objects makes local recovery practical.

Evaluation of this work was carried out over two testbeds: Indriya at NUS and Twist at TU Berlin. Various experiments strongly demonstrated that Splash reduced the data dissemination completion time by an order of magnitude. Contributions of individual techniques were also presented giving better insight to various factors responsible for this performance improvement.

Philip Levis complimented the work for the significant gain demonstrated in the paper and asked about the implications for the physical layer, whether there was a need for new physical chips. Manju replied that it depends on the modulation techniques. Masoud (USC) asked whether devices needed to be synchronized and what data rate was required to keep that accuracy. Manju agreed that there is a need for synchronization and that the time difference between different data transmissions should be less than .5 μs to result in constructive interference. Session chair Brad Karp asked about how constructive interference will scale out as you increase the density, because the heuristic used about leaf and non-leaf nodes might not work if the network is too dense and non-leaves are nearly the entire network. Manju said that it is totally random because of the capture effect; it depends on the placements of nodes, and it is difficult to find an optimal number of receivers.

### *Community Award! Expanding Rural Cellular Networks with Virtual Coverage*

Kurtis Heimerl and Kashif Ali, University of California, Berkeley; Joshua Blumenstock, University of Washington; Brian Gawalt and Eric Brewer, University of California, Berkeley

*Awarded Community Award!*

Kurtis Heimerl mentioned that his talk was about cellular networks, which is not very common at networking conferences. Interestingly, he compared the invention of cellular networks with the light bulb to demonstrate the impact of this technology. This set the tone for the entire talk. Kurtis explained the reasons why rural areas lack cellular network coverage: the investment cost is high and user density is not enough to recover costs. He revealed that half the cost of running a cellular tower in remote areas is power related. Thus, to enable wider coverage for rural areas, power draw must be reduced. Kurtis explained various components of a rural base station tower and the costs for each of its components, making it clear that the lack of power infrastructure was responsible for higher costs.

The power amplifier draws 130 W constantly and is always turned on. Because the number of users is low in rural areas, most of the time nothing goes on and energy is wasted keeping the power amplifier constantly running. Their simple idea is to enable sleep for base stations when not in use. Implementation of this idea is relatively tricky for the user side. Kurtis spoke about two solutions for this problem: wake-up radios and wake-up phones. The fundamental change is to involve users for power provisioning, which is shown to be very common in rural areas. The proposed solution when compared to traditional schemes saves around 84% power.

Alex Snoeren (UCSD) asked about power consumption of receivers for scanning activities, as in the case of WiFi networks; scanning is a power hungry activity, so how does this work impact power consumption for handsets? Alex also wondered about the cost of maintaining armed guards for rural areas. Kurtis replied that scanning is a common activity for cellular networks and results in no overhead for this solution. Labor cost is trivial in these areas, so hiring guards is not much of a cost issue. Rajesh Nishtala (Facebook) asked whether this kind of solution could be used for data activities like checking email. Kurtis said that there is nothing about this work which limits it to calls; it can be used for data activities, too. Though, as these things are asynchronous in nature, some periodic synch activities need to be planned around such a service, but it is surely doable. Josh Reich (Princeton) asked about potential attacks on such a service. Kurtis replied that validation mechanisms make sure that such attacks are not a problem. Josh asked why the authors didn't considered power cycling. Kurtis replied that usage for such a service is mostly for emergency situations and thus timing is an important factor to consider. Daniel Turner (UCSD) asked an economics-based question: who will sponsor such a service, government or rural entrepreneurs? Kurtis said that GSM operation requires a license, and its usage by rural entrepreneurs would require policy changes.

## Big Data

*Summarized by Muntasir Raihan Rahman (mrahman2@illinois.edu)*

### *Rhea: Automatic Filtering for Unstructured Cloud Storage*

Christos Gkantsidis, Dimitrios Vytiniotis, Orion Hodson, Dushyanth Narayanan, Florin Dinu, and Antony Rowstron, Microsoft Research, Cambridge

Christos Gkantsidis presented research on optimizing data analytics in the cloud. Public cloud infrastructures are breaking the locality of data and computation by introducing separate storage and computation infrastructures. This approach has many advantages, but comes at the cost of network transfer. The paper has measurements that characterize the amount of data transfer required for this, and the author mentioned that it is quite significant.

The key insight is that most jobs only operate on a subset of the input data. So filtering the input data before transferring from storage to compute can yield significant performance gains. The authors propose generating network filters that get rid of unnecessary data before starting the network

transfer. The filter has to be correct and transparent; however, to filter the data, we need some structure, whereas the input data is usually unstructured. The authors propose using static analysis of job byte-code and extracting row and column filters to discover the data that is actually used in computing. Filters are opportunistic, conservative, and transparent.

Christos also briefly outlined the system design, especially the construction of row and columns filters. A row filter discovers rows that generate some output data. Then column filters identify which substring of that row is of interest. One problem is that some MapReduce programs use state, whereas filters cannot rely on mutable state. The solution is to tag all mutable fields as output. On the other hand, column filters use abstract interpretation and tokenization methods to find interesting columns within a row.

The experimental setup was Hadoop on Windows Azure. The author presented results for eight jobs in the same datacenter; however, implementing filters on Amazon storage is not possible. So the authors took the data, filtered it using Rhea, and compared the job results with the pure input and the filtered input. The authors also observed that the overhead of Rhea is linear in the number of cores. They observed that the filter performance was bottlenecked by string I/O. They saw 30–80% improvement in runtime using Rhea. These numbers turn out to be lower than job selectivity in Hadoop.

Following the talk, Ryan McElroy (Facebook) asked whether the authors tested the filters directly, where storage and compute are already collocated, and whether there were any benefits. Christos replied that they did some experiments using local storage and machines for testing the filter, which means that storage and compute are collocated, and they saw some benefits. The session chair, George Porter, then asked whether declarative cluster computing frameworks such as Spark, Pig, or Hive could help with the static analysis required in Rhea. In a nutshell, George was asking about generalizing Rhea from pure MapReduce programs written in Java to more functional programs like Spark, which is written in Scala. Christos first clarified by saying that in declarative languages, select and project operators are explicit in the code, which Rhea had to discover for MapReduce programs. But even in that case, the user will still need to do static analysis on user-defined functions.

### *Robustness in the Salus Scalable Block Store*

Yang Wang, Manos Kapritsos, Zuocheng Ren, Prince Mahajan, Jeevitha Kirubanandam, Lorenzo Alvisi, and Mike Dahlin, The University of Texas at Austin

Yang Wang presented Salus, a scalable and robust block store. The first priority of any storage system is not to lose data. The problem is exacerbated by remote storage used by most users. Salus adds robustness on top of scalable remote storage systems such as Amazon elastic block store (EBS). There are existing systems that provide strong protection against arbitrary failures; however, these techniques do not go well with online scalable storage systems. This is where Salus comes in; it matches well with scalable block stores. Salus inherits scalability from existing systems, while providing strong robustness with low overhead. The robustness guarantees are that a client never reads corrupted data, and the system remains durable and available despite a fixed number of failures.

Yang then described the architecture of existing scalable systems. The first component is a metadata server, which is rarely accessed to avoid being a bottleneck. Scalability is achieved via parallel access to data servers, whereas the availability and durability guarantees come from replication; however, these systems lack ordered write guarantees and have single points of failure. A block store needs ordering guarantees for implementing barrier semantics. A simple checksum does not suffice to resolve the corruption of data by a compute node.

Salus overcomes these problems by introducing end-to-end verification at the block driver level to prevent corrupt data reads by clients. Also, Salus uses pipelined commits to the servers to implement barrier semantics. The single point of failure is alleviated through active storage in the replication protocol. Yang then went into details about pipelined commits and active storage. A naive approach of waiting for all writes to finish does not work for barrier semantics, since it loses all parallelism. Two-phase commit also falls short because it cannot provide ordering among multiple transactions. Pipelined commit solves this issue by forcing a transaction leader to wait for an acknowledgement from the previous leader. So the pipelined commit still does the first phase of 2PC in parallel but executes the second phase sequentially for barrier semantics. The performance is still good because the message overhead in the second phase is much smaller than in the first phase. Salus handles active storage by decoupling safety and liveness. Safety is still achieved via f+1 replicas, whereas liveness is guaranteed by only storing soft state in compute nodes. Surprisingly, active storage helps with performance because the soft state compute nodes can now be collocated with storage nodes. Evaluation results show that Salus is always safe, and remains live when the number of computing node memory failures doesn't exceed two. The experiments also showed that the overhead of Salus doesn't grow as the system scales.

George Porter asked whether Salus can support multiple concurrent writers. Yang answered that it was still an

open problem, but it could be achieved by sacrificing either linearizability for causal consistency or scalability. Salus can handle independent random failures. As a follow-up, George asked whether Salus can handle correlated failures. The author said that they did not experiment with correlated failures and it was left as future work.

### *MemC3: Compact and Concurrent MemCache with Dumber Caching and Smarter Hashing*

Bin Fan and David G. Andersen, Carnegie Mellon University; Michael Kaminsky, Intel Labs

Bin Fan presented MemC3, which improves upon the basic Memcached via low overhead caching and better hashing. The goal of Memcached is to reduce space overhead and improve performance. The state-of-the-art systems improve performance by sharding, which eliminates inter-thread communication; however, this only works for uniform workloads. For skewed workloads, sharding can lead to hotspots and memory overhead. Instead, the authors try to use space-efficient concurrent data structures for their system via smart algorithmic tuning. Their system has 3x throughput improvement and requires 30% less space compared to the original Memcached system.

Before diving into the details of MemC3, Bin gave a brief overview of the original Memcached architecture and the typical workload for the system. MemC3 optimizes for this typical workload of small objects with high throughput requirements. The core data structures used in Memcached is a key-value index implemented via a chaining hash table, and a doubly linked list for LRU eviction. MemC3 replaces these data structures with efficient concurrent data structures without global locks for improved single-node scalability and reduced space overhead. For hashing, the authors use optimistic cuckoo hashing, which is space-efficient and highly concurrent. For cache eviction, they use CLOCK-based LRU eviction. The rest of the talk focused on optimistic cuckoo hashing.

The default Memcached architecture uses a chaining hash table, which has low cache locality and high pointer cost. Another alternative could be linear probing, which is cache friendly but has poor memory efficiency. Instead, the authors use cuckoo hashing, where each key has two candidate buckets, and lookups read both buckets in parallel. The value is actually stored in just one of the buckets, and this scheme has constant amortized insertion cost. Memory efficiency is further improved to 95% with increased set-associativity. Even though the system only supports single write, multiple reader concurrency, the problem is still hard due to false misses. This happens if a read happens while a recursive insertion is going on. The authors solved this using a two-step insert process. In the first step, the system finds a path to an empty bucket without editing buckets. In the next step, the empty slot is moved back through the reverse path. The only required guarantee is that each swap is atomic. The authors use optimistic locking for the atomic swap, which is optimized for read-heavy workloads. The system also avoids concurrent writes by serializing all inserts, which performs well with read-heavy workloads. Micro-benchmarks reveal that their hash table increases throughput from 1.74 to 21.54 M lookups/sec for all hit lookups, with further improvement for all miss lookups. They also showed that their system scales much better with increased number of threads.

Following the talk, Srimat Chakradhar (NEC Labs) asked whether the authors considered the impact of network performance on MemC3, since the network can be the main bottleneck. He commented that the experiment was masking the true network delay, and that any amount of local computation improvement cannot bypass the network delay. Bin wanted to answer this question offline. Arash Molavi (Northeastern University) asked whether the hash path could end up in a loop. Bin responded that they handled it with a fixed upper bound on retry attempts, and that if a lookup fails after 500 attempts, the table is most likely full. Arash then asked how the authors came up with the magic number of 500. Bin's response was that they used experimental parameter tuning to figure out the optimal number of retry attempts. Sid Sen (Princeton University) suggested that further improvement could be obtained by using overlapping instead of disjoint set-associativity for the hash table. The speaker responded that they tried that idea, but it did not work since they are only storing part of the key for performance reasons.

### *Scaling Memcache at Facebook*

Rajesh Nishtala, Hans Fugal, Steven Grimm, Marc Kwiatkowski, Herman Lee, Harry C. Li, Ryan McElroy, Mike Paleczny, Daniel Peek, Paul Saab, David Stafford, Tony Tung, and Venkateshwaran Venkataramani, Facebook Inc.

Facebook is the largest user of Memcache in the industry space, and Rajesh Nishtala began by listing Facebook's key system infrastructure requirements: real-time communication and aggregation from multiple sources, access and update of popular content, and huge scalability. This means that any such system needs to support large read-heavy workloads, must be geo-replicated, and has to support evolution of products; however, Memcache does not need to worry about persistence, which helps it scale. The basic building block for Facebook's infrastructure is Memcached, which is basically a network attached in-memory hash table with LRU-based eviction policies. The rest of the talk gradually explored larger and larger scale usage of Memcached at Facebook.

The first phase was a single front-end cluster that dealt with a read-heavy workload, wide fan-out, and fault tolerance. Before delving into the design, Rajesh briefly mentioned that with the pre-Memcached architecture, a few shared databases sufficed to serve the workload. Obviously, this did not scale, as evident from an example of data fetching for a simple user request that Rajesh pointed out. So the first step was to add a few Memcached servers (around 10) to alleviate the load (about a million operations per second). The typical workload had about twice as much reads as writes, which required higher read capacity. The solution was to use Memcached as a demand-filled look-aside cache, with high probability of lookup success. Updates usually invalidate Memcache entries. Facebook prefers idempotent deletes rather than updates because they can be repeated without loss of data; however, the look-aside cache can lead to stale sets where the Memcache and the database are not consistent. This was resolved with a lease. This still leaves a problem called "thundering herds," where a huge number of Web servers flood the database that stores a popular item once the Memcache entry is invalidated through an update. This issue is resolved by using Memcache as an arbiter to decide who gets access to the database.

Next, Rajesh described the scale increasing to around 100 servers and about 10 million operations per second. Obviously, this required even more read capacity. At this scale, items are distributed using consistent hashing; however, this leads to all-to-all communication between Web servers and Memcache servers, which is bad for the network. One such problem is incast congestion, where multiple Memcache servers reply to the client at the same time, which overloads the client network connection. The solution is a simple sliding window protocol to control the number of outstanding requests.

In the second phase, Rajesh went on to talk about scaling with multiple front-end clusters, which introduced issues like data replication control and consistency. At this scale, there are thousands of servers and hundreds of millions of operations per second. Here the main problems are to keep each cluster of Memcache servers consistent and to manage over-replication of data. The solution is a simple push-based approach, where the database pushes invalidation updates to all Memcache clusters. The network overhead of this broadcast approach is handled using a middle layer of Memcache routers.

In the grand finale, Rajesh went on to the largest scale, that is, multiple regions, where data consistency issues really kick in. At this scale, there are billions of operations per second. The main problem is to handle writes to slave regions. This is resolved by only writing to the master, which works for read-heavy workload, but there can still be race conditions. This is handled via a remote marker, which is a special flag that indicates whether a race is likely to happen. This works because Memcache ensures that misses are rare. Again, the read-dominance of the workload is the reason why this works.

Rajesh concluded with three lessons learned from building massive scale systems. First, pushing as much complexity to the client as possible always helps. Second, operational efficiency is a first-class citizen. And third, separating cache and persistent storage allows systems to scale.

Anirudh Badam (MSR) asked about bottlenecks in a single server. Rajesh answered that their systems are always provisioned for the worst case, which means that single server bottlenecks rarely surface. Badam then asked about using flash-based storage for Memcached. Rajesh said that they use flash mainly for colder data, as opposed to using Memcached for hot data. Dave Andersen (CMU) asked whether the large scale usage of Memcache at Facebook was due to legacy issues. Rajesh answered that wasn't the case. He said scaling Memcache has worked well at Facebook and there is no reason to replace it. Dave then asked about using Memcached outside Facebook. As Bin pointed out in the last talk, the typical Facebook workload has a lot of reads of small data items, so Rajesh reiterated that as the dominant workload that would drive Memcache deployment outside Facebook. Amar Phanishayee (MSR) asked about the size of the cache and the average utilization. Rajesh answered that it depended on the popularity of the item, which is basically the principle of caching. Marcos Aguilera (MSR) asked for a clarification about pushing complexity to the client. Rajesh clarified that the complexity is actually pushed to the client-side library.

## Posters

*Summarized by Utkarsh Goel & Anish Babu Bharata*
*{utkarsh.goel, anishbabu.bharata}@cs.montana.edu*

### *Demystifying Page Load Performance with WProf*

Xiao Sophia Wang, Aruna Balasubramanian, Arvind Krishnamurthy, and David Wetherall, University of Washington

Many techniques that focus on reducing the Web page load time (PLT) exist. Even though these techniques aim to provide a low PLT, they are still difficult to be identified due to the complexity of the page load process. The authors have abstracted a dependency graph of the activities that make up a page load and have developed a lightweight in-browser profiler, WProf, to produce this graph. The results obtained from WProf shows that synchronous JavaScript plays a significant role by blocking HTML parsing, and the computation is a significant factor that makes up as much as 35% of the critical path.

### *Towards A Secure Controller Platform for OpenFlow Applications*

Xitao Wen, Yan Chen, Northwestern University, Chengchen Hu, Xi'an Jiaotong University, Chao Shi, Northwestern University, Yi Wang, Tsinghua University

The OpenFlow architecture suffers from trust issues as it embraces third-party development efforts. Most of the apps are highly flexible for defining network behavior, yet the openness and the absence of security enforcement make it difficult to keep such trust on the OF controller, especially on the third-party modules. To provide an intuitive impression, authors have envisioned four classes of attacks: (1) direct intrusion from control plane into data plane,( 2) leakage of sensitive information, (3) manipulation of OF rules, (4) deactivation of other apps. As a part of the implementation, the authors have ported the app hub to evaluate the additional controller latency and throughput. Their results show that there were no additional latencies, and even the throughput was reduced by only 1.3%.

### *Syndicate: A Scalable, Read/Write File Store for Edge Applications*

Jude Nelson, Larry Peterson, Princeton University

To preserve data consistency and its durability, coordination between cloud storage, network caches, and local storage is a must. The authors' work is mainly based on following design goals: decouple caching from consistency, decouple storage policy from implementation, and decouple read performance from durability. The authors' early work consisted of implementation of the user gateway as a FUSE file system, the replica gateway as a cloud-hosted process, and the metadata service as a Google AppEngine service.

### *Dynamic Layer Instantiation as a Service*

Flavio Espositox, Yuefeng Wangx, Ibrahim Matta, John Day, Boston University

Current Internet architecture lacks scoping of control and management functions. This results in a challenge to deliver a communication service with required characteristics when the ranges of operations are wide. This deficiency, together with the desire to offer virtualized network services, has compounded existing network service management challenges. The main contribution of the authors is that RINA is capable of enabling private networks to be instantiated dynamically. This is done by customizing network management policies into a single layer, without the shortcomings of the TCP/IP architecture.

### *WASP: A Centrally Managed Communication Layer for Smart Phone Networks*

Murad Kaplan, Chenyu Zheng , Eric Keller, University of Colorado

With the rapid increase in the use of smartphones-based interactive applications, overloading off the cellular network's capacity can be observed, which may quickly expend the limited bandwidth cap. In order to overcome these problems, the authors have proposed WASP, a communication layer within a smartphone that couples cellular connections with direct phone-to-phone links (over WiFi direct). The poster presented by the authors presented the design and implementation of WASP and discussed the design space that was enabled by this new SDN-based model. Their demonstration consisted of three parts: (1) they collected data from a small scale collection of phones, (2) they collected data from a larger scale simulation of the Android application within NS-3 (Network Simulator 3), and (3) they conducted a mixed-mode simulation involving real Android phones participating with the NS-3 based simulation.

### *Characterizing Broadband Services with Dasu*

Zachary S. Bischof, Mario A. Sanchez, John S. Otto, John P. Rula, Fabián E. Bustamante, Northwestern University

The authors presented a crowd-sourced approach to broadband service benchmarking from end-systems. In the context of Dasu, the authors have described its prototype implementation. Dasu is a measurement experimentation platform for the Internet's edge based on the observation that, by leveraging minimal monitoring information from the local host and home routers, one can achieve the scale of "one-time" end host approaches with the accuracy of hardware-based solutions. The demonstration of Dasu consisted of broadband characterization functionality and its user interface. The results observed by the authors showed that by using an end-host approach, the large-scale view allowed them to capture the wide range of service performance as experienced by users over different times of day and across different geographic locations.

### *Auditable Anonymity*

Sonia Jahid and Nikita Borisov, University of Illinois at Urbana-Champaign

The cloud is a common platform these days for storing and sharing data. Deploying applications in clouds brings some new challenges in security and privacy. The authors' work enables the data owner to get logs for data accesses, while hiding the information from the cloud provider at the same time. This is done at the data owner's end by decrypting the audit log and then getting access to all the logs provided by data access.

### *Self Tuning Data-Stores for Geo-Distributed Cloud Applications*

Shankaranarayanan Puzhavakath Narayanan, Ashiwan Sivakumar, Sanjay Rao, and Mohit Tawarmalani, Purdue University

Most of the applications using the Internet share user data in an interactive manner. Moreover, these applications have stringent service-level agreements that are responsible for placing tight constraints. These constraints affect the performance of underlying geo-distributed datastores. The main challenge here is the deployment of such systems in the cloud as application architects have to maintain consistency between multiple replicas, minimize access latency, and ensure high availability. In this poster, the authors have adopted a systematic approach in which they were able to capture the performance of a datastore by developing analytical models. These datastores are based on application workload and are capable of building a system for optimal performance by automatically configuring the datastore.

### *A Declarative Framework for the Verification of Network Protocols*

Jiefei Ma and Alessandra Russo, Imperial College London; Jorge Lobo, Universitat Pompeu Fabra; Franck Le, IBM T. J. Watson Research Center

Verifying network protocols is a challenging problem. In this poster, the authors have proposed a declarative framework that builds upon the recent realization that with simple extensions, database-style query languages can be used to specify and implement network protocols. This framework consists of three components: a protocol model, a communication model, and an analysis model. The authors have observed the following results: in a network running a link state protocol, the presence of persistent forwarding loops was revealed by using this framework, and the framework detected flaws in a MANET protocol that was designed for finding disjoint paths.

### *Natjam: Prioritizing Production Jobs in the Cloud*

Brian Cho, Samsung Inc.; Muntasir Rahman and Indranil Gupta, University of Illinois at Urbana-Champaign; Cristina Abad, University of Illinois at Urbana-Champaign and Yahoo! Inc.; Tej Chajed, University of Illinois at Urbana-Champaign; Nathan Roberts, Yahoo! Inc.; Philbert Lin, University of Illinois at Urbana-Champaign

This project presents the Natjam system, which does both efficient resource utilization and better job run times during overload. This system introduces job and task eviction policies, thus achieving the co-existence of high priority production jobs and low priority research jobs on the Apache Hadoop cluster. This system is well tested, and they presented data that shows Natjam is better than existing systems and relatively close to an ideal outcome.

### *Consistent Packet Processing—Because Consistent Updates Are Not Enough*

Peter Perešíni and Maciej Kuźniar, École Polytechnique Fédérale de Lausanne and Technische Universität Berlin/T-Labs; Nedeljko Vasić, École Polytechnique Fédérale de Lausanne; Marco Canini, Technische Universität Berlin/T-Labs; Dejan Kostić, Institute IMDEA Networks

This project explores the needs for consistent packet processing in software-defined networking controllers and describes the situations in which performance and scalability issues arise even in a straightforward OpenFlow deployment. The project proposes the use of transactional semantics within the controller and explains the benefits it poses over consistent packet processing.

### *IRate: Initial Video Bitrate Selection System for HTTP Streaming*

Ricky K.P. Mok, Weichao Li, and Rocky K.C. Chang, The Hong Kong Polytechnic University

This project proposes IRate, which offers lightweight, fast, and yet accurate decision-making for selecting the best initial video bit rate for a given network condition. As one of its benefits, this system just needs modification on the server and not on any hosts. The results show greater accuracy of the IRate quality oracle against the duration of the probe kit measurement.

### *Software Defined Measurement for Data Centers*

Masoud Moshref, Minlan Yu, and Ramesh Govindan, University of Southern California

Accurate detection of large flow aggregates and choice of better routes for these flows are needed for performing traffic engineering. For reducing latencies of a partition/aggregate, there is a need for identifying short traffic bursts. These measurement tasks vary according to what traffics are required to be measured, where to measure them, and when to measure. For automatically distributing the measurement tasks across all the switches, the authors have proposed leveraging a centralized controller in datacenters. The key challenge here is to identify the right division of labor across the controller and switches. Another challenge is that monitoring a source IP prefix is a collaborative task of multiple switches because a datacenter topology consists of several switches and no switch sees all the traffic.

### *IP2DC: Making Sense of Replica Selection Tools*

Anish Bharata, Mike P. Wittie, and Qing Yang, Montana State University

Most of the cloud-based applications deliver the same level of responsiveness as stand-alone software, which leads to user dissatisfaction and slow adoption. In order to avoid poor user experience due to end-to-end delay or lag, back-end logic and application data are usually deployed across geographic

locations. These distributed servers then direct all the user requests to the closest server. The challenge lies in the accurate selection of a server closest to a user, or a group of communicating users. Even though earlier studies have proposed a number of tools for identifying the closest replica server to a client IP, these tools suffer from incomplete coverage of the IP space and can make predictions based on stale network measurements. Hence, which of these tools makes the most accurate prediction in most cases remains unclear. The authors of this poster are interested in their coverage of the IP space and their accuracy in determining the closest public cloud datacenter to a given IP, relative to direct probing.

## Reliability

*Summarized by Peng Huang (ryanhuang@cs.ucsd.edu)*

### *F10: A Fault-Tolerant Engineered Network*

Vincent Liu, Daniel Halperin, Arvind Krishnamurthy, Thomas Anderson, University of Washington

*Awarded Best Paper!*

Vincent started by stating that multi-rooted tree topologies such as Fat Tree are preferred in today's datacenter networks for bisection bandwidth and cost concerns. Vincent proceeded to talk about failure detection and recovery problems inside these networks. The way they deal with failure is through heartbeat detection and centralized controller for recovery by exploiting path redundancy. But the detection and recovery are slow, which can lead to suboptimal flow assignment: DC networks use commodity switches that often fail and the particular topology doesn't allow local recovery. Vincent described the basic approach of F10, which achieves fast failure detection and local, fast, and optimal failure recovery by co-designing topology, routing protocol, and failure detection.

Fat Tree is not good at failure recovery primarily because there is no redundancy on the way down and the alternative routes are many hops away. They stem from the symmetries in the topologies: each node is connected to the same set of parents as the siblings and the same set of grandparents as their cousins. F10 breaks these symmetries. To this end, it investigates two types of subtrees, one with consecutive parents (type A) and the other with strided parents (type B). But type A and type B alone is essentially Fat Tree. The authors propose to mix them as an AB Fat Tree: half of the subtrees are type A and half are type B. With this topology, more nodes can have alternatives that are one hop away. Then F10 adds a cascaded failover protocol through local rerouting, notification, and a centralized scheduler.

Vincent went on to explain the failure detector in F10 in detail; it's faster than current failure detection because it looks at the link itself, monitoring incoming bit translation and rerouting the next packets. The key point is that rerouting is cheap in F10.

F10 was evaluated with both testbed (Emulab) and simulation on traces. Results show that failure recovery can be under a millisecond and with only 14% of the congestion loss compared to Fat Tree.

Brad Karp (UCL) asked how the solution compared to multipath TCP. Vincent said they didn't evaluate this, but as pointed out in the paper, they are targeting slightly different cases, where you aren't necessarily able to affect end-hosts, and you want them to use their own OS. Vincent speculated that even with multipath TCP, there would still be performance degradation in that you will loose one of the paths, which potentially could be worse than F10. Siddhartha Sen (Princeton) suggested adding more links, like VL2, so they have a fully connected topology. Vincent replied that he thought that was a Clos topology and that F10 will further optimize that. Even with a traditional datacenter network which has all-to-all connectivity, F10 probably would have a 50% - benefit over Fat Tree. Someone from USC asked whether the solution had impact on the nice load balancing properties of Fat Tree. Vincent replied that F10 doesn't change that. You can do the same thing in F10.

Someone from MSR asked two questions: when a node fails, will reacting too quickly to links cause more problems? Vincent answered that the actual number of messages broadcast is not that high. Also, F10 will do exponential back-offs. Does the global scheduler make an implicit assumption that traffic is stable? Vincent explained that previous work (Hedera and MicroTE) shows that there are LFN (Long Fat Network) flows that are predictable. Even if things are unpredictable, the scheduler can still provide benefits.

### *LOUP: The Principles and Practice of Intra-Domain Route Dissemination*

Nikola Gvozdiev, Brad Karp, and Mark Handley, University College London

Nikola started with users' expectations of Internet reachability as the transport has become reliable and routing systems adaptive. Many real-time applications, such as VoIP and interactive gaming, become intolerant of brief interruptions. Routing is a major source of the unreachability. Nikola then explained the big picture of routing and the roles of iBGPs. Although previous work has looked into gateway protocols, the fundamental behavior of intra-AS route propagation is still unanswered. Typically, iBGP fails when updates of iBGP causes transient loops, which in turn cause collateral damage.

The authors propose new, clean-state intra-domain route dissemination protocols, SOUP and LOUP, that are loop free. These protocols use reverse forwarding tree (RFT) and forwarding activation (FA), require minimal configuration, and can be fully distributed. They can provably avoid loops when the underlying topology is stable. Nikola showed several cases of transient loops and how the protocol can avoid them.

The evaluation of SOUP an LOUP was done on synthetic topologies, and simulation was on a simulated publicly available network topology. Result shows LOUP causes no loops or blackholes.

Ethan Katzbatha (USC) asked what should be done for packets going to destinations that are being withdrawn. Nikola said there is a fundamental tradeoff between a loop and a blackhole. If you don't do anything, the packets will be dropped at the border. There can be smarter solutions if there is already a tunnel set up. The border router knows where the packet should be going to, and the router can forward the packet using this tunnel. Michael Freedman (Princeton) noted that even with the tell-me-when shortcut, there might still be strange issues in that the propagation of reverse activation where the announcements go to different parts of the network. Nikola said that there is one more mechanism that is not described in the presentation that fixes the issue.

### *Improving Availability in Distributed Systems with Failure Informers*

Joshua B. Leners, Trinabh Gupta, The University of Texas at Austin; Marcos K. Aguilera, Microsoft Research Silicon Valley; Michael Walfish, The University of Texas at Austin

Joshua described the importance of failure handling in distributed systems. The typical method of timeouts doesn't tell what and why. For example, consider the ICMP "destination unreachable" error message; it's unclear why something's unreachable: is it a problem with the destination or network? is it permanent or transient? The same error message is delivered for all these types of failures. The reasons behind this is that the network was designed to hide find-grained information as described in a SIGCOMM 1988 paper on design philosophy of the DARPA Internet protocol. Joshua revisited this classic network design choice and argued that this choice is mismatched to today's requirements. Today's network environment is built with a large number of commodity machines and switches and, as a result, failures are common and diverse. Applications, on the other hand, lack failure information. To them, the diverse failures look similar and so they choose universal recovery mechanisms.

Joshua then provided an example where knowing the type of failure can help applications to act more efficiently for failure recovery. The thesis of the paper is to expose the failure types to application.

One potential problem of exposing failure types to applications is that it may be burdensome to applications. For example, an application now needs to understand the semantics of each failure type, and if the interface changes, the application code also needs to change. The approach they use to tackle this problem is to group failures: e.g., process crash, host crash, and host reboot can be grouped to represent the case where the target stops permanently.

The authors built a service, Pigeon (5400 LOC C++), which implements their new interfaces. Pigeon collects local failure information with sensors, transports the information to end-hosts, and provides an interpreter for applications to understand the information.

Evaluation of Pigeon on a 12-host Fat Tree-topology network connected by 16 physical routers with injected failures shows distinguishing the host and network failures reduces unavailability by 4x, from 6.9 seconds to 1.6 seconds. Also, enhancing Cassandra with Pigeon helps Cassandra optimize replica selection. Pigeon also helps RAMCloud avoid unnecessary recovery.

Dave Levin (University of Maryland) asked whether Pigeon can handle Byzantine failure. Joshua answered it's targeted for traditional crash failures. Someone from MSR asked how to ensure the information sent to application reflects the ground truth. Joshua said for stop conditions, Pigeon confirms that the process crashed by, for example, looking at the process table. In the case of network failure, the current prototype fetches the topology from the gateway. Michael Freedman (Princeton) asked whether using this in a large distributed system, where nodes might see the same failure as different types, would cause conflicting reports and problems for Pigeon. Joshua said no.

## Applications

*Summarized by Murad Kaplan (murad.kaplan@colorado.edu)*

### *BOSS: Building Operating System Services*

Stephen Dawson-Haggerty, Andrew Krioukov, Jay Taneja, Sagar Karandikar, Gabe Fierro, Nikita Kitaev, and David Culler, University of California, Berkeley

Stephen Dawson-Haggerty started by presenting facts on energy consumption in the US, how a big percentage of this consumption is related to building and how this is a computer science problem. As many buildings have digital controls that include sensors and actuators that run many applications, Stephen asked the big question: can we write portable building monitoring and control applications that help to reduce the energy consumption caused by the digital control? He pointed to a number of applications used in today's buildings and asked if researchers can be forward looking in the way these application use resources and energy. These applica-

tions break down into a couple of different categories. The challenge is how to integrate existing applications using the existing physical infrastructure, and how researchers can make these kinds of applications that pop up all the time widely deployable in a way that saves more energy. The other challenge is how researchers can incorporate these applications in wider scale control to better manage their energy consumption.

From a systems point of view, researchers face several challenges; the first is portability—how do we take these applications, which are pretty much demo work today, and make them run anywhere in the building? Second is the failure modes we have been introduced to in these applications and how we can deal with them. And third, accessing the control systems in the buildings, exposing the existing hardware to the world and building all our stuff on top of it. Researchers want to scale to a large number of sensors and to large buildings, and they want these applications to be portable and easy to use.

To accomplish these goals, Stephen and his team built BOSS, distributed building operating system services that together allow these applications and control processes to sit on top of and interact with physical sensors and actuators in the building, and that offer authorization, optimization, and personal comfort services.

Stephen described the BOSS architecture by presenting portability as a part of the abstraction layer. He also presented applications that were built on BOSS to use its services to perform analysis, provide personal control, and personalize the micro-environment and optimization. The team installed these applications in a number of buildings to make these measurements.Stephen concluded that while BOSS provides a lot of opportunities, there are also a number of issues since applying computer system design to buildings involves a lot of pieces. BOSS was able to provide a 30% saving in electricity and steam and a 60% saving in lighting in its test apps. These apps can be found at http://smap.cs.berkeley.edu.

Someone asked how BOSS would provide a good handle for coordination in the case of a multiple control system. Stephen answered that there is a tradeoff in the way that you handle these services; right now they are statically configured. Kurtis Heimerl (University of California, Berkeley) was interested in the user programming environment and asked how far along BOSS was in doing that. Stephen responded that it will be a cool idea to let people program but the key challenge is the really dynamic binding.

### *Stochastic Forecasts Achieve High Throughput and Low Delay over Cellular Networks*

Keith Winstein, Anirudh Sivaraman, and Hari Balakrishnan, MIT Computer Science and Artificial Intelligence Laboratory

Keith Winstein started by highlighting some special characteristics of current cellular networks. He explained how cellular networks have probably became the means by which a majority of users access the Internet, but that cellular networks differ in many ways from the traditional Internet. One of these differences is that cellular networks are highly variable in link speed. Using Verizon LTE cell phones, they measured the downlink and uplink as they kept them busy and kept recording how fast packets crossed each link. Keith presented a graph that showed how extremely variable the network was. Keith also pointed out another difference from the "old fashioned" networks is that they are too reliable. We talk about best-effort network delivery, but in fact it is insane-effort network delivery. Another graph of one TCP download showed how there is a big delay in the round trip time.

Then Keith presented a graph showing Skype performance over the Verizon LTE network. This is to show when Skype is not using all the bandwidth available because Skype is being very conservative. Also when Skype is not conservative and sends higher than the available bandwidth, the result is a huge delay. To overcome this problem, Keith presented their protocol, Sprout. Sprout is a transmit protocol designed for real-time interactive applications over these variable, semi-reliable networks. Sprout has the goal of providing the application with the most throughput possible, but the higher priority is to bound the risk of accumulating a large delay in gateway queues.

In order to control the risk of delay, the authors have separate parts of their algorithm. At the Sprout receiver, the authors infer the current link speed and they do that with Bayesian inference on a stochastic model. In part two, the receiver tries to predict what is going to happen to the link speed in the future. It makes a "cautious forecast" and sends this to the sender as part of its acknowledgments. In part three, the sender uses this forecast to control an evolving window that compromises between the desire for more throughput and for controlling the risk that a packet will spend more than 100 ms in a queue.

Keith later compared Sprout with other applications, such as Skype, Google-Hangout, and Facetime. Sprout performed better in term of throughput and delay. Keith said that a Stanford networking class had reproduced these plots, and the two students who chose Sprout were awarded the best project in the class. MIT students were challenged to beat Sprout and came up with 3,000 different algorithms with different compromises between throughput and delay. Keith

indicated several limitations in their measurements in the conclusion of his talk: Sprout was only evaluated in the case of video conferencing, and all measurements were made in Boston. The source code and directions to use it are at : http://alfalfa.mit.edu/.

Phil Levis (Stanford University) said that he sees the authors are using LEDBAT (Low Extra Delay Background Transport), but did they think they should replace LEDBAT with Sprout? Keith answered that LEDBAT tries to coexist with TCP on the same bottleneck queue to scavenge available bandwidth. Although there are similarities between LEDBAT and Sprout such as counter filter and one-way delay estimation, they did not test it in competition with TCP on the same queue, so they don't want to say that they do better than LEDBAT in this situation.

Kurtis Heimerl (University of California, Berkeley) asked if the authors had thought about using geographical knowledge to help in this. There are falls in throughput traffic when a user is switching towers, for example, or driving a car, so how would Sprout use that to make better prediction? Keith answered that there is previous work in NSDI on this kind of thing, and he is not sure if speed and location may affect his results or not. Kurtis pointed out that with cellular networks, there is a whole circuit switch piece of technology that essentially is designed to solve this problem for voice traffic; how is that related to the problem Sprout is trying to solve? Is it something Sprout could just solve by using that circuit switched network? Keith replied that circuit switching networks in cellular systems have admission controls that give users 64k bits per second and they don't have that for large bit rate flows that need to vary in speed. The problem is that the link quality varies, which you can't just solve with better policy. Someone asked, looking at the graph with the available bandwidth even if the mobile was stationary, whether they had tested Sprout with Sprout. Keith replied that they did that by testing six phones in the same cell.

### *Demystifying Page Load Performance with WProf*

Xiao Sophia Wang, Aruna Balasubramanian, Arvind Krishnamurthy, and David Wetherall, University of Washington

Xiao Sophia Wang demonstrated that the Web is everywhere and it is a critical part of the Internet. Unsurprisingly, page load is critical. Studies suggest that Amazon can increase its revenue by 1% by reducing page load time by 100 ms. Wang showed her measurement studies of page load time of about 200 Web pages. The results show that the median page load time is three seconds. And few Web pages take more than 10 seconds to load. Because Web page load time is slow, there are many techniques aimed to reduce this time. These optimization techniques include using CDN, small pages, etc. While these techniques can help some pages, they may have no effect on others. And, in fact, they may even harm performance. The reason is that the page load process is poorly understood. Because of that, the authors' goal in this paper was to understand the page load process. Wang explained how difficult it is to understand page load time because there are many factors that affect the page load. She showed an example on how a page's structure affects the page load time.

Wang presented the process of demystifying Web page performance. The authors modeled the page load process. And they presented WProf to implement the set of dependencies inside browsers to study in real Web pages. To identify the bottlenecks of page load, the authors applied critical path analysis on top of the dependences graph that WProf generated.

Their methodology includes three elements. First, they designed test pages to systematically uncover dependencies across browsers. They also examined documents as well as browser code whenever the source code was available. For designing the test pages, the authors started by thinking how Web objects are organized in Web pages. They found four categories of dependency policies: (1) flow dependencies, which is the natural order of the activities acquired; (2) output dependencies, which ensure the correct execution of multiple processes with access to the same resources; (3) lazy and eager binding, which are tradeoffs between data downloads and page load latencies; and (4) resource constraints resulting from limited computing power or network resources.

Wang reviewed some experimental results where they loaded real pages using WProf. They ran the experiments at the campus network using WProf in Chrome, looking at 150 Web pages. Their conclusion was that most object downloads are not critical, and JavaScript blocks parsing on 60% of top pages. Caching eliminates 80% of Web object loads, but it doesn't reduce page load time as much. Wang concluded her talk by pointing out the most important elements of WProf: it automatically extracts dependencies and analyzes critical paths. WProf can be used to understand performance of page load times and to explain the behaviors of current optimizations.

Someone asked whether WProf can predict what the benefit from parallelization will be. Wang replied that HTML parsing and CSS evaluation can be parallelization, but JavaScript evaluation is hard to parallelize. Someone from University of California, San Diego, asked if the critical path measurement can be used for a specific object, not only the entire page. Wang replied that their tool can do that. Brighton Godfrey (University of Illinois at Urbana Champaign) asked, assuming their results are based on the minimum over five trials for each page, would their conclusions change if they looked at the median and outliers? Wang replied that because there is large variation, they only looked at the median page.

### *Dasu: Pushing Experiments to the Internet's Edge*

Mario A. Sánchez, John S. Otto, and Zachary S. Bischof, Northwestern University; David R. Choffnes, University of Washington; Fabián E. Bustamante, Northwestern University; Balachander Krishnamurthy and Walter Willinger, AT&T Labs—Research

Mario Sánchez talked about having measurement experimentation platforms located at the edge of network. The rapid growth of the Internet has been driven in part by advances in broadband technology and affordability which made the Internet not only larger but also spread out and diverse. Although many of the distributed systems are located at the edge of network, when we measure them from the code, they look different. The truth is that researchers like measuring platforms at the right scale, vantage point, and location in order to provide sufficient control for them to conduct needed experimentation. While there are platforms like PlanetLab that provide great fine-grained control, they don't scale very well. Also, their vantage points are located in research and academic networks. Other platforms, like Dimes, provide the potential for scale, and the actual view that researchers want, but they also provide little flexibility in term of control and experimentation. So there is a need for a platform that combines the best of both worlds. But for such a platform to be useful, the safety of the volunteered nodes need to be guaranteed, and researchers need both to be able to control the impact of having the experiment in their network and to have a way to share resources between multiple experiments and multiple experimenters.

To overcome these challenges, Mario presented Dasu, which is a prototype for such a platform. Dasu is a software measurement experimentation platform that is hosted by end users located at the edge of the network and that relies on the direct incentive of using broadband measurement for end users. Dasu was designed with two purposes: to allow end users to characterize the service from their broadband network and to support experimentation from the edges. By considering these design purposes, Dasu aligns the objectives of end users and experimenters in three different ways. First, both end users and experimenters can benefit from having large coverage in order to capture network and service diversity. Second, both users and experimenters benefit from having large availability in order to be able to catch changes in policies and scheduled events. Third, users and experimenters also benefit from being at the edge and extensible, in the case of end users in order to remain unaffected in the face of changes in ISP policies, and for experimenters in order to be able to expand the new platform with new measurements.

At the same time, Mario pointed out that researchers face a lot of challenges; one of them is the lack of dedicated resources, which means they cannot run arbitrary experiments on these machines. For instance, some experiments related to censorship might be out of the question because these may get end users in trouble. Mario presented the design implementation of Dasu and how it addresses these challenges.

To protect volunteer nodes, Dasu runs the experiments inside its own engine inside a sandbox. Also, Dasu's clients run a resource profiler to continuously monitor the resources consumed by the system, and there is also a watchdog timer that triggers client shut down if anything goes wrong. Mario presented the dynamic of the system by showing experiments run by clients using Dasu.

Dasu is currently an extension to BitTorrent and soon will be stand-alone in a DNS resolver. Mario also pointed out that Dasu has over 90,000 users in over 150 countries. Dasu's main components are the Dasu client and the infrastructure service that supports it.Indranil Gupta (University of Illinois at Urbana Champaign) asked how Dasu would stop a runaway experiment. Mario said they do have the resource profiler that looks at the CPU usage if anything goes wrong, and then it will automatically shut it down. Mike Weighty (Montana State University) asked if the authors have any plan on extending Dasu to mobile devices. Mario said there are other students in his lab who focus on wireless, and he is working on something related.

## Security and Privacy

*Summarized by Weverton Cordeiro (weverton.cordeiro@inf.ufrgs.br)*

### *πBox: A Platform for Privacy-Preserving Apps*

Sangmin Lee, Edmund L. Wong, Deepak Goel, Mike Dahlin, and Vitaly Shmatikov, The University of Texas at Austin

Sangmin Lee pointed out a recent study which has shown that only 17% of users pay attention to access permission requests when installing applications on their mobile devices. More alarmingly, just 3% of them actually fully understand these requests. But even though users read and fully understand these requests, there is no guarantee that apps will actually comply with them—in fact, there is evidence that some apps out there are misusing users' private data! The user may choose to believe that app publishers will not misuse their private data; but in which one of the 300,000+ publishers can they actually trust? According to Sangmin, such an obscure scenario for users' privacy strongly suggests that existing countermeasures to protect users' privacy are not working.

Sangmin proposed to shift the users' trust from the 300,000+ app publishers to a few major brands, such as Google, Microsoft, Apple, and Amazon. In addition to the fact that one already has to trust these companies to use their device anyway, they also have a reputation to maintain, and thus more incentive to work correctly. The key to realize the proposed

shift? πBox: a framework that combines app confinement through sandboxes and control channels to provide explicit and useful privacy guarantees.

Sangmin explained that πBox provides a per-user, per-app sandbox confinement. All users' data stays inside the sandbox along with an app instance, and the sandbox prevents the app from leaking the data. The sandboxes use a model which makes them span the users' device and the cloud—and thus use resources from both the device and the cloud. Private vaults (which are in both the device and the cloud) and content storages are provided for storing user and app-specific data, respectively. Both are secured to prevent privacy violation. The aggregate channels provide communication capabilities between the app and its publisher, and use differential privacy to introduce random noise and thus bound information leakage. Finally, The sharing channel allows one to have control of "when" and "with whom" to share, but provides weak guarantees on "what to share." However, it makes it difficult for an adversary to access users' private data directly. Sangmin mentioned, though, that shared channels are not entirely secure (for example, photo sharing remains subject to steganography).

Along with a three-level classification model (green, yellow, and red), πBox enables fine-grained control over which security features will be enabled for each app, and provides explicit privacy guarantees users can rely on. Sangmin closed the talk arguing that real, existing applications can benefit from ϖBox, and highlighting that overhead is low: only a few lines of code needed to be changed in existing, popular apps; and the measured information throughput was marginally degraded.

Fengyuan Xu (College of Willian and Mary) appeared concerned about the burden πBox will place on users (who will have to decide what, when, and with whom contents will be shared, for example). Sangmin argued that there will be not much difference from how it is currently done, and that apps will provide support for that decision process. In a follow up question, Fengyuan asked if πBox is vulnerable to attacks that use covert channels. Sangmin replied that it depends on whether the implementation of sandboxes and the design of πBox is orthogonal. He added that if there are any improvements on building better sandboxes, πBox can benefit from it by using it. Fengyuan then asked if πBox is something like a platform, in which you can plug in other schemes to make it more secure. Sangmin simply replied that yes, it is. Yan Chen (Northwestern University) asked what the aggregation channels can support. Sangmin replied that one big goal of aggregate channels is to provide information to app providers without violating users' privacy. He mentioned some examples of apps and how they could use aggregate channels to export information to app publishers. Yan then asked if ϖBox does protect social network information (i.e., the list of friends to whom one is sharing photos) from potential spammers. Sangmin replied that yes, it does.

### *P3: Toward Privacy-Preserving Photo Sharing*

Moo-Ryong Ra, Ramesh Govindan, and Antonio Ortega, University of Southern California.

Moo-Ryong Ra started his talk by mentioning that cloud-based photo sharing service providers (PSPs) are very popular, in spite of the various privacy concerns on the way these providers provision their services. For example, shared photos may be over-exposed (either accidentally or because of poor PSP system design). The PSP could also use inference techniques to extract information from shared photos. In other words, he argued that one can trust the (mobile) device but not everything else (e.g., the network and the PSPs).

On one hand, these privacy concerns Moo-Ryong mentioned are real. He showed several news headlines that reported privacy issues with Photobucket, Facebook, and Instagram. On the other hand, PSPs provide valuable and useful services to users: a shared photo may be adapted to the various types of devices that access these photos (smartphones, computers, etc.) and perform image quality processing. The only question is, can we protect users' privacy while still performing cloud-side image transformation? The answer: yes, we can. In the remainder of his talk, Moo-Ryong described P3, a privacy-preserving photo encoding algorithm. The algorithm extracts and encrypts a small part of the photo that contains significant visual information (the "secret part"), while preserving the larger part (the "public part"), which contains no significant information, in a standards-compatible format. With P3, PSPs can still provide their services without being required to redesign their systems.

P3 concentrates on JPEG images, and exploits the fact that DCT (Discrete Cosine Transform) coefficients of natural images are sparse, with a few coefficients holding most of the useful information; put in an intuitive way, P3's "secret part" and "public part" correspond to the most significant bits and least significant bits of the DCT coefficients, respectively. There is a challenge, however: how to reconstruct the original image out of the secret and public parts? If the public part has not been modified, a straightforward series of linear operations can reconstruct the image. In case it has been modified, the PSP must also send the linear operator used for processing it. During the talk, Moo-Ryong discussed the (easy-to-deploy) architecture of P3, provided evidence regarding its privacy-preserving aspect (through a set of examples and evaluation experiments), and showed that it causes minimal storage overhead (in the worst case, the image file-size increased at most 20%). Regarding the

privacy-preserving aspect, Moo-Ryong showed that P3 virtually broke face recognition algorithms: the number of successful face recognitions on the public part was decreased significantly.

Nikita Borisov (University of Illinois at Urbana-Champaign) asked if one can obtain the original picture solely with the secret part. Moo-Ryong answered negatively, saying that one needs both parts to obtain the original picture. In a follow up, Nikita posed the situation in which the P2P is fully malicious. Moo-Ryong said that in this particular case P3 is broken. However, it was emphasized that in this case it is a functionality problem rather than a privacy problem—the P2P will not be able to do anything malicious without the secret part anyway. The session chair (Krishna Gummadi, Max Planck Institute for Software Systems) asked for the high-level intuition as for why the smallest part of the image contained most of the visual information, whereas the larger part of the image contained almost no information. Moo-Ryong replied that this is related to why JPEG works, and revisited the discussion about the process of extracting the secret part from the image. Finally, Lakshminarayanan Subramanian (New York University) asked about the privacy guarantees of P3. Moo-Ryong replied that they are still trying to prove privacy guarantees of P3 using mathematical tools. However, they are unaware of any techniques that could break the algorithm.

### *Awarded Best Paper! Embassies: Radically Refactoring the Web*

Jon Howell, Bryan Parno, and John R. Douceur, Microsoft Research.

*Awarded Best Paper!*

Jon Howell started his talk by stating that the Web promises us a really nice isolation model, in which users are protected from malice even though they click on "dangerous links." The reality is quite different, however, as browsers have vulnerabilities that might compromise the system, if explored by a malicious Web app. In this scenario, what kind of advice should one give to ordinary Web users? Do not click on "dangerous links"?

According to Jon, these security weaknesses are not caused by poor Web API implementation; even a correct browser implementation cannot protect users from external protocol flaws. "The API itself needs to be changed," he said. Why? He first explained that the Web API has a subset of methods that specify the behavior of the application when it arrives at the client; that subset he called Client Execution Interface (CEI). To provide isolation, CEI must be as simple as possible and have a well-defined semantics, so that it can be implemented correctly; in other words, it should pursue minimality. He also explained that the same API has served as a Developer Programming Interface (DPI), and developers have always wanted this interface to be richer and richer, so that fancier applications can be built with less and less code; in other words, it has also been pursuing richness. "This API has been pulled off in two opposite directions," he said. To solve this problem, he proposed to refactoring the Web API, separating the roles of CEI and DPI.

Jon argued that the CEI should provide Web applications an execution environment having the same philosophy (and isolation guarantees) of a multi-tenant datacenter but in the client machine—a "pico-datacenter," as he defined it. In his refactored API, binary code would be the core of the CEI, thus accepting binary programs from the vendor. In this case, Web developers would be able to develop their applications using the libraries that best suit their needs (e.g., libraries for HTML rendering, JavaScript processing, etc.); not only is browser incompatibility gone in this "Embassies" model, but users would also be able to run applications such as Gimp in their browsers. To run an application in the pico-datacenter, the client would just need to load the required libraries. From an experimental evaluation using a prototype, the Embassies model has been shown to introduce some overhead. However, Jon strongly emphasized that this overhead comes in exchange for a truthful Web promise of a nice isolation model. "Dangerous links"? No more!

Nikita Borisov (University of Illinois at Urbana-Champaign) mentioned he saw some similarities between the Embassies model and the one that does exist in the mobile platform, and then asked Jon if the two models would eventually become one. Jon replied that he definitively sees convergence there, since the mobile world is being pushed towards the right user model. However, he thinks that not all the opportunities of this model are being explored, since mobile apps do not depend on a clean and narrow execution interface. Someone asked about the effectiveness of Web-indexing in this new model and the burden it would cause to vendors such as Google and Yahoo!. Jon answered that this is a reasonable aspect to be worried about, but said that while Web app vendors can go anywhere they want (by using any shared libraries they please), they would still attach to popular transit protocols, and also expose interactions one would be able to sniff on. Lakshminarayanan Subramanian (New York University) proposed taking the work in a different direction, and asked if there is a problem with how Web pages should be designed. Jon suggested that a way one can look at this work is to ask why the Web API developers use to design their pages must be bound to the browser, instead of being just software libraries that they are free to choose. Jon said it is inevitable there will be complexity in the stack used to design Web pages. The advantage is that Web developers will be free to make choices over that complexity.