3rd USENIX Workshop on Hot Topics in Cloud Computing (HotCloud '11)

Portland, Oregon
June 14–15, 2011

Scheduling and Resource Management

Summarized by Byung Chul Tak (tak@cse.psu.edu)

Static Scheduling in Clouds

Thomas A. Henzinger, Anmol V. Singh, Vasu Singh, Thomas Wies, and Damien Zufferey, IST Austria

Damien Zufferey presented a job execution environment, called Flextic, for cloud applications. The goal of Flextic is to find a good interface between a cloud provider and customers. The user is expected to specify various job characteristics that will allow the provider to make efficient use of the datacenter resources. First, the user submits a program with information about resources, e.g., task durations and data size, using a custom language. This program is parsed to produce an execution plan. Then the scheduler generates several schedules for the given execution plan, with corresponding prices. The users select the preferred schedule, and this schedule is carried out by the execution platform. In selecting the most suitable schedule, the user can consider a price curve in which the price is high for shorter execution time and low for longer execution time. Choosing a longer time gives the provider more flexibility to optimize the global state of the datacenter; hence the price is lower. One challenge with this approach is that it requires solving large scheduling problems, so an abstraction technique is proposed to reduce the problem size (job model and infrastructure model) into a smaller one by exploiting regularities in the application.

The first question was whether it was fully implemented and what sort of tools were used. Damien said there was a proof-of-concept implementation. The scheduling part could scale well, but the execution part was still simple. Glenn Ammons (IBM) asked whether the technique was going to be used as an estimate or whether it would enable bidding. Damien replied that bidding would certainly be possible if people agreed on ways of describing the jobs. But the more fundamental question is whether the information needed in the technique is realistic. In the study, the main focus was scientific applications, but whether this technique is applicable to the more general class of application is important.

Migration, Assignment, and Scheduling of Jobs in Virtualized Environment

Seung-Hwan Lim, Pennsylvania State University; Jae-Seok Huh and Youngjae Kim, Oak Ridge National Laboratory; Chita R. Das, Pennsylvania State University

Why do we need to care about performance unpredictability in the cloud? Seung-Hwan Lim claimed that unpredictability creates a cascade effect in all the related jobs: a low-performing outlier dictates the overall performance of the entire application. In order to address this problem, virtual machine (VM) scheduling or reassigning to different physical machines has been considered. Amid VM scheduling, he mentioned that a set of VM migrations occur, and migration policy, in turn, determines the performance impact in reassigning VMs. He presented his measurement that showed that migration time could vary according to system configuration and how to group VMs for migration. He formulated an optimization problem that tries to minimize the total migration time when migrating a set of VMs while bounding the performance impact. This formulation allows him to estimate the completion time when multiple jobs contend for multiple resources. He also proposed performance slowdown as the metric of performance variance, which can be calculated from his formula.

How would this work handle cases in which jobs were dependent? This work assumed only independent cases, in order to ease the difficulty of calculating the probability of contention across multiple resources. The dependent case is more challenging and would be a direction for future work. Byung-Gon Chun asked how much accuracy degraded in estimating finish time when more than two jobs were considered. Lim said results showed about 15% accuracy degradation with up to four co-located workloads. How does this work compare with existing live migration work? Lim replied that many have considered the optimal state in terms of VM assignment, but this work looks at what happens during the state transition to optimal states. A live migration addresses migrating a single virtual machine, but they dealt with multiple VM migrations bringing a greater performance impact than a single VM migration.

Cloud Scale Resource Management: Challenges and Techniques

Ajay Gulati, Ganesha Shanmuganathan, Anne Holler, and Irfan Ahmad, VMware, Inc.

Ajay Gulati argued that resource management is critical for cloud deployments, both private and public. A desirable cloud management solution should provide high elasticity (i.e., scale) as well as high efficiency in terms of utilizing hardware resources. Current small-scale management solutions

such as VMware DRS have some difficulty scaling up to the cloud level, but cloud providers want to maximize system efficiency in order to achieve higher revenue. On the other hand, cloud providers such as Amazon EC2 do not provide a rich set of resource management controls to allow for better multiplexing and over-commitment of hardware resources.

DRS provides an abstraction called "Resource pools trees" that allows the specification of resource allocation in a hierarchical manner. The VMs are the leaves of this tree, and one can specify controls such as reservation, limit, and shares for each inner node of this tree hierarchy as well as VMs. These controls dictate the minimum, maximum, and dynamic resource allocation in case of contention. The benefit of this technique is that it allows you to specify the aggregate amount of resources instead of per individual VM, and actual resource allocation per VM can be dynamically controlled. In the case where some resource is idle, it is reassigned to the other VMs within the immediate group first before being made available to any higher grouping units.

However, when trying to scale DRS to cloud scale, there are several challenges. First, resources are heterogeneous, so some VMs cannot be hosted on some set of machines due to various constraints such as storage and network connectivity. Second, operations need to be carried out at high frequency. With this, the centralized scheme can suffer from lock contention issues, and distributed schemes need to make decisions based on partial information. Lastly, failures are common at cloud scale. In order to deal with these issues, three techniques are proposed: hierarchical, flat, and statistical scaling. Hierarchical scaling builds a load balancer on top of clusters that are managed by DRS. Flat scaling builds an overlay of virtualized hosts (the main difficulty of this being the lack of consistent views). In statistical scaling, a subset of the cluster is selected to form an eCluster, and DRS is run on this eCluster.

Andrew Warfield mentioned that the DRS algorithm in the paper seemed to be non-terminating in the review version and the final version had newer features that made it look different. Ajay responded that the final version of the algorithm included additional details and that the algorithm does terminate in all cases. Some parts, such as cost-benefit analysis, were intentionally left out due to space limitations. Michael Schwartz asked about the disadvantages of statistical approaches. Ajay replied that looking at a subset does not give you the best solution, because picking some random set of machines implies some loss of efficiency. But, given the result from the power-of-two choices, the loss of efficiency should be small.

Heterogeneity-Aware Resource Allocation and Scheduling in the Cloud

Gunho Lee, University of California, Berkeley; Byung-Gon Chun, Yahoo! Research; Randy H. Katz, University of California, Berkeley

The talk was about how to allocate resource and schedule jobs in a heterogeneous environment. Lee argued that the heterogeneity of a system environment made such tasks difficult, because some jobs require special hardware support such as a GPU. The proposed approach takes into account this need for specific hardware and, using the ProgressShare metric, provides fair scheduling of jobs. The traditional unit of scheduling was number of slots, but this does not work well in a heterogeneous environment. The proposed ProgressShare metric brings actual progress into the picture, and it can be used to schedule the jobs so that multiple jobs can make more progress than when using the SlotShare metric.

Orna from Technion asked how they would measure the ProgressShare of all the jobs that would have run using all the slots. Lee replied that they could run it on a small set of machines as a micro-benchmark to estimate the progress and could also utilize the historical data. Orna also asked what happens when jobs report false progress numbers to gain some benefit. Lee said that jobs cannot benefit from lying. If one job says this machine is bad when it's actually good, then it will receive more bad machines. On the other hand, if the job says this machine is good when it's bad, it can prevent other jobs from using that machine, thereby harming other jobs.

Ion Stoica opened the panel discussion by asking how each author took into account different types of resources and whether they had plans to incorporate them into their work. Ajay said that in VMware they had hosts for VM placement and that they considered CPU and memory during the placement. Virtual disk placement is a separate problem. Virtual disks can be placed in the best data store and the user VM on the best host in terms of CPU and memory. Demian said that, concerning I/O, they did consider the location of files for better scheduling, but did not deal with network congestion. Lim added that when you share disks such as SSD, workload pattern would differ in each case because access patterns change.

Chit-kwan Lin asked about time granularity in managing the cloud infrastructure. According to Ajay, in VMware products it is about five minutes; running the load-balancer takes about a minute to finish at the current level of scalability. Any finer time granularity will cause too much overhead. If the scale grows, the time has to increase to maybe 15 or 20 minutes.

The session chair, Byung-Gon Chun, asked how often the VM migration happens in practice. Ajay said that it depends on the specific environment; some clients said that there were many migrations once they turned on the DRS algorithm. But migration definitely happens more frequently than people expect. Steven Ko from SUNY Buffalo asked why scheduling research is still active after being studied for so long. Demian said that some of the assumptions have changed in the cloud environment, so new heuristics may be required. One attendee pointed out that scheduling is important because H/W cost is high and providers would like to utilize the infrastructure to the maximum. Orna followed by saying that there are two new things about cloud: there are consumers who pay for execution where grid is a kind of best-effort thing, and virtualization introduces interference. She argued that we should adopt work from grid computing.

Economics

Summarized by Tian Guo (tian@cs.umass.edu)

To Move or Not to Move: The Economics of Cloud Computing

Byung Chul Tak, Bhuvan Urgaonkar, and Anand Sivasubramaniam, The Pennsylvania State University

Byung Chul Tak began with the benefits of cloud computing: cost saving, automatic scalability, and transparent redundancy. The focus of the talk was the cost-saving aspect of migrating an application to the cloud. Pay as you go and elasticity make it cheaper and easier to match the cloud more closely with demand. However, there is no consensus that the cloud is really saving money. Therefore, in this work, they tried to systematically investigate the conditions and variables affecting the benefits of clouds, studying two representative applications from which they could draw conclusions.

The cost assessment process of their framework involves specifying application properties and potential service time in order to calculate hardware configuration, identifying direct quantifiable costs (converting future cost into present cost for fair comparison by using NPV), and comparing these costs among five hosting options for high and low workload intensities. The conclusion they drew is thatcloud-based hosting is preferable for lower workload intensity with a smaller growth rate. He also said that data transfer cost would be significant and component-level partitioning can be costly. In the analysis of the effect of storage and software license, he further explained the importance of these two factors in decision-making. Last, he briefly mentioned the need for more accurate performance estimation of cloud-based application and economic study for scientific applications.

In the Q&A, Suman Jana (UT Austin) pointed out the work only considers steady increasing workloads and doesn't have a model for spiky workload. Byung Chul Tak said that they did consider the workload in a variant increasing speed, but the workloads are flattened over a long period of time. Andrew Warfield (University of British Columbia) asked whether they were assuming cheaper leasing prices over time. The speaker expressed his hope for reduced fees. Andrew asked whether the cloud-based results are more efficient. Byung Chul Tak answered yes.

Cutting MapReduce Cost with Spot Market

Huan Liu, Accenture Technology Labs

Huan Liu told us that the work is about how to save money with the spot market, and he explained why it is reasonable for a spot market to exist in the public cloud. There are unpredictable spiky workloads even in a large cloud provider like Amazon. By providing economic incentives, users will have the motivation to move around their demand and help to smooth out utilization for the providers. The problem with the spot market is its unpredictability. The Cloud MapReduce architecture employs a distributed approach implementing several queues, using Amazon's Simple Queue Services. He said that the data processed in Mapper will be sent to a reduce queue instead of stored locally. The reducer will pull all the messages in the reduce queue after receiving the pointers from the master reduce queue. Later on the reducer will generate output messages and send them to the output queue.

Huan Liu highlighted two important things: the checkpoints are stored in the SimpleDB and all the intermediate results get sent to reduce queue as soon as possible; there is streaming going on between MapReduce stages. Specifically, in one map node, there is a temporary buffer storing only one key-value pair. The idea is to gain some time before the cloud provider actually shuts down the instances, in order to flush good results to reduce queues and to check the assignments left in input queues by preventing the soft shutdown script from executing. Cloud MapReduce is the first ever implementation that works for spot market and saves a lot of money.

What happens if there is not enough time for flushing? Huan answered that the correctness of the results is guaranteed, even though some partial results will get lost because the partial results will never be committed in the SimpleDB. Is it more valuable to have a hybrid scheme with control nodes that are not spot market instances? Huan said yes, in the sense of performance guarantees. Andrew Warfield (University of British Columbia) asked how much money could be saved by using the Cloud MapReduce. Huan said they would

save more money if prices fluctuated frequently. Even if the client bid for the highest price in the spot market every time, there would still be a 50% saving.

Exertion-based Billing for Cloud Storage Access

Matthew Wachs and Lianghong Xu, Carnegie Mellon University; Arkady Kanevsky, VMware; Gregory R. Ganger, Carnegie Mellon University

Matthew Wachs said that the focus is on Infrastructure-as-a-Service in cloud accounting. In this setting, the providers want to recover their costs and the clients want to be charged fairly. Matthew focused on the storage consequences of accounting. He noted that providers only bill for capacity but fail to bill for access, which also incurs cost in buying more disks. He further pointed out that current metrics, such as IOPS or bytes transferred, are not directly proportional to time used and therefore are wrong. He later gave an example of billing for fixed I/Os, which is unsustainable in most cases. A few more alternatives were listed but none of them is ideal. Matthew said that charging for disk time is the fairest solution and that workload interference might affect the exertion required. The solution is to use performance insulation to avoid interference in the first place. He said that storage insulation could be achieved by preserving locality and providing predictable cache allocation. After using the insulation to limit the impact of other workloads, the exertion shown is close to ideal. Finally, he pointed out the importance of using disk time as a metric and performance insulation to guarantee fairness in billing for clients.

Zachary N.J. Peterson (Naval Postgraduate School) asked why providers are still using the metrics Matthew claimed to be wrong. Current metrics are easier for customers to understand and therefore they are more willing to pay. There are disadvantages with exertion-based billing, such as less transparency. Zachary asking about the mechanism that prevents providers from increasing the clients' disk time to create higher revenue. Matthew replied that billing for CPU time also has the same issue of provider trustworthiness; both need to be solved. Eyal de Lara (University of Toronto) questioned the advantages of this relatively complicated billing model compared to the current simple billing model. Matthew thought the sacrifice of simplicity is worthwhile, since clients are now paying much more money than they should. There was another question about why a simple billing model employed by a mobile company is not sufficient for cloud billing. Matthew pointed out there are a lot of available options in mobile markets and noted that the money paid for cell phone bills and for cloud service bills are not on the same scale. People would care more about precise accounting in the cloud environment, because of the amount of money involved.

John Wilkes (Google) asked how they would take latency of requests into account. Matthew said latency would not necessarily incur higher costs for providers. John again doubted the potential of the proposed billing model by giving an example of the potentially higher cost incurred by sequential accesses. Matthew concluded that disk time also belongs to opportunity costs with which we should motivate the billing models.

Panel

What do academics need/want to know about cloud systems?

Panelists: Chris Colohan, Google; Greg Ganger, Carnegie Mellon University; David Maltz, Microsoft; Andrew Warfield, University of British Columbia

Summarized by Sahil Suneja (sahilsuneja@gmail.com)

This was a post-lunch session and meant to instill enthusiasm and energy into the audience. The discussion truly lived up to expectations with the elements of fun and liveliness.

The panel started off with the industry people, David and Chris. David offered his list of things he wished everyone knew. A major point was that datacenter costs have unfortunately remained stable over time—the cost of a server today, irrespective of whether it is being used or not, is about $55 per month—so turning off servers isn't an attractive option. He also talked about the primary metric for a datacenter being profit, and that translates into minimizing costs by buying cheaper systems, increasing resource utilization, and reducing the cost of delivering power to the datacenters.

Chris emphasized the need for efficient and reliable datacenter design/layout. He raised questions regarding the means of estimating resource requirements for datacenters and the requirement of a theoretical basis for the correct size of clusters. Similar issues included the need to decide the number of machines per cluster vs. number of clusters, hierarchical cluster designs, decisions regarding uniformity/non-uniformity of machines, and hardware-software logic distribution, among other concerns. He talked about the lack of batch work to soak up available computational resources today, and the common error of using today's workloads to estimate future cluster sizes.

On the academic front, Andrew stressed the need to obtain realistic hints and technological constraints from industry

people for driving research. The more disclosure behind the internal functioning of commercial datacenters, the better the improvement opportunities for researchers. He highlighted the practical necessity of the industry people and academic researchers spending time together to keep innovating and building systems in interesting ways.

Greg tried to play the old curmudgeon, targeting Dave in particular. Like Andrew, he emphasized the importance of getting feedback from industry so that researchers are prevented from making imprudent assumptions. At the same time, he stressed the importance of academics questioning industry-provided assumptions as perhaps being the bottleneck for just one particular industry instance vs. a global phenomenon.

Michael Kozuch from Intel kicked off the discussion by inquiring how academics should think about scale. Dave and Chris confessed that they are confronted with this issue in industry as well. Dave dodged the question by saying that a thousand is small while a million machines is big, to which Greg playfully responded that these bounds are beyond what academics have access to! John Wilkes from Google teasingly advised the panelists to collaborate like physicists; Greg responded that physicists have an entire generation of students who work on constructing mechanisms that one day would allow the experiments to run, and that is impractical in computer science.

Tal Garfinkel from VMware steered the discussion away from scale in cloud computing and sought the panelists' thoughts on an autonomous self-serving collection of machines, data service, and applications that eliminates the need to involve IT and operations people. Chris diplomatically answered that the reason behind automating system administration could be the system scale that might make a robot a more economical option than actual people to manage the system. David agreed that this is a hard problem, as it requires the knowledge of what metric is to be optimized and how to obtain the input/output data. This needs both applications and people to find out edge cases, as Tal had mentioned in his question. Greg challenged the notion and argued that people have been working on automation and a lot of work is focused on problem diagnosis, which seems to be the hardest issue to deal with. He was surprised to see a response coming from industry to eliminate IT people. Chris took the opposite route and stressed the need for more operational staff at Google, which Greg criticized by commenting that it's the PhD students that are the operational staff at Google! The crowd joined in by arguing for the need to employ people!

As the session ended, John gave a lighthearted final remark—a suggestion that every academic to pursue an industry person for an interesting problem until he grasps the problem completely.

Security

Summarized by Tian Guo (tian@cs.umass.edu)

The HybrEx *Model for Confidentiality and Privacy in Cloud Computing*

Steven Y. Ko and Kyungho Jeon, University at Buffalo, The State University of New York; Ramsés Morales, Xerox Research Center Webster

Steven Ko talked mainly about how the Hybrid Execution (HybrEx) model fits into the context, instead of the details of system implementation. He put forward a general question about the trustworthiness of the cloud environment. The main focus of the work is to figure out how to utilize clouds with partial trust. The realities of people's distrust, the potential threats to the cloud, and the benefits of using the cloud make the problem worthy to explore.

In the current cloud environment, there are only two extreme options for people: complete trust or distrust of the cloud. Steven remarked that HybrEx is a solution for dealing with the unexplored middle ground. He explained that the main ideas of HyberEx are partitioning and tainting. For partitioning, HyberEx categorizes data as either public or private. Since there are just partial trusts for the cloud, the client will only deal the private data in a private cloud environment. In order to prevent information leakage, tainting is used to keep track of the data. After explaining the framework of HyberEx, Steven discussed the specific contexts it applies to, namely MapReduce and Bigtable. The popularity and feasibility of the two applications make them ideal for a good start. However, there are challenges such as finding the appropriate applications which will benefit from data partitioning, sanitizing data to enable private to public shifts, potential performance decrease due to higher communication cost, and the correctness of the computation.

Aditya Akella (University of Wisconsin—Madison) asked whether it is easier to track public and private data on a large scale. Steven said building a tainting system inside a framework like MapReduce might solve the problem. Since MapReduce is already well partitioned, the tagging of private and public data is relatively easy. Suman Jana (University of Texas—Austin) questioned the scalability of the taint tracking of data in a virtual machine level because of the significant overhead. Steven said the tainting is incorporated into the MapReduce level instead of the virtual machine level in order to lower the overhead. Aditya questioned the difference between data partitioning and vertical partitioning. Steven replied that the existence of the hybrid approach is not the

same as partitioning data into public and private. Following up, Aditya was curious about the cost-benefit analysis of hybrid execution. Steven thought that was a good direction for future work.

A Position Paper on Data Sovereignty: The Importance of Geolocating Data in the Cloud

Zachary N.J. Peterson, Mark Gondree, and Robert Beverly, Naval Postgraduate School

Zachary said that most people don't care about the location of data as long as it is accessible. However, it is a non-trivial problem, especially in the cloud environment, considering that some data should stay within political boundaries even when including data replication. Traditional data location doesn't represent the actual place where data is stored. The purpose is to efficiently locate some copies of data within certain boundaries. He stressed that tracking all copies is a hard and interesting problem.

Zachary explained that there are two techniques: geolocation of the host and possession of data. Simply combining the two techniques won't solve the problem, though. It only proves the existence of the host instead of the data. In addition, he pointed out that adversaries might purposely fake the data source by adding delay. For example, if some Web proxies cached subsets of the data and manipulate the delay measurements, people would gain incorrect information about the data location. He mentioned an important aspect of network measurement, which is that the server can only pretend to be outside the bounding area and never falsely pretend to be inside. An initial approach is leveraging MAC-PDP (a signed statement of Provable Data Position), which can be augmented with network delay measurement. In order to get the exact location of the data, multiple challengers should be used. The merits of the approach are it minimizes the latency without requiring the server-side computation, and it is easy to apply to existing infrastructure. However, higher communication costs are expected. Future directions include the evaluation of the initial idea and placement of landmarks.

Suman Jana (UT Austin) asked how to ensure the effectiveness of the mechanism if users want their data to be outside the boundaries instead of within. Zachary admitted that one-way verification will not help in this case but as a solution proposed doing computation that binds location. Chris Colohan (Google) questioned whether it is necessary to retain a copy of data locally in order to know the locations of other copies. Zachary responded that MAC-PDP protocol clients only need to store a MAC key k instead of the whole copy of data. It will recompute the copy to verify the authenticity. Someone asked about the legal framework regarding the goal of this work. Zachary said that it lies mostly on the IP side and made analogies to privacy violation and medical records. Aditya Akella (University of Wisconsin—Madison) mentioned that the extra copies outside the system are hard to track. Zachary agreed and pointed out the paper tries to show that the copies of data are within some boundaries.

Privacy-Sensitive VM Retrospection

Wolfgang Richter, Carnegie Mellon University; Glenn Ammons, IBM Research; Jan Harkes, Carnegie Mellon University; Adam Goode, Google; Nilton Bila and Eyal de Lara, University of Toronto; Vasanth Bala, IBM Research; Mahadev Satyanarayanan, Carnegie Mellon University

Wolfgang Richter started his talk by explaining the difference between introspection and retrospection for virtual machines. In introspection, we examine the live logs of a virtual machine during its execution. In retrospection, however, we can have access to all historical logs of all the virtual machines. He pointed out that we should treat VMs as big data instead of executable content. Retrospection is about deep search over historical VM data at a raw-data rather than metadata level while respecting privacy. For example, VM retrospection can be used as a unified interface to search all the historical data in a compromised VM for the root cause of the exploit. Searching a set of instances for privacy violation among different companies who use a similar cloud infrastructure would be another case. He said that the privacy goal of VM retrospection is achieved by data cryptography. So far they have explored the per-file, per-directory, and per-partition data encryption.

Wolfgang briefly described the design principle of their work. They provide on-demand search through the unified interface. Second, they offer VM owners the right to make the suitable retrospection policy. Last, they try to support generality of search. He concluded by talking about the implementation called Nanuk.

What level of VM data structure should be coupled with the implementation? Nanuk could query whatever data was available, regardless of the structure. What is the advantage of doing this work at the VM level? Wolfgang pointed to the potential security gain if the whole operating system is compromised. The snapshots of the VM guarantee the integrity of data even in the worst case. Aditya Akella (University of Wisconsin—Madison) asked about the trust model. Wolfgang replied that the trust model they explored provides as much as possible to the VM owners. The search of private data is possible only if the key is provided by the owners. Chris Colohan (Google) questioned whether it is necessary for VM users to take the snapshot after the data is encrypted. Wolfgang admitted that it is a question worth thinking about.

EVE: Verifying Correct Execution of Cloud-Hosted Web Applications

Suman Jana and Vitaly Shmatikov, The University of Texas at Austin

Suman Jana started his talk by providing a scenario of an interactive Web application running in the cloud. Once the application is submitted to the cloud, the correctness of the application is not visible to the owner. The incorrectness could be caused by things such as network failure, storage failure, or consistency failure. Knowing the detailed information about the application failure is crucial to owners, and running the applications in the cloud makes this a more challenging problem due to the low visibility of the cloud environment. He stressed that we should think more about consistency and partition tolerance compared to the availability of the service.

Suman gave an example of transient error in a tax application due to the low share of storage in the cloud. It is only possible to track the inconsistency if the owner monitors the application consistently, he commented. The focus of their work is to continuously verify the correctness of interactive Web applications. After analyzing the architecture of popular applications in the cloud, he concluded that the focus should be on verifying the consistency of data store operations. By checking consistency violations in the data store, faults would be easy to track. With EVE, witnesses keep logs of operations and send them to the verifier periodically for error detection performed by the streaming consistency verification algorithm. Finally, he talked about different scenarios where EVE could be useful, including checking the scalability of the application and comparing the quality of service among cloud providers.

Zachary Peterson (Naval Postgraduate School) asked about the efficiency of EVE's error detection for WordPress. Suman said that since WordPress doesn't employ an eventual consistency back-end database, no consistency violation can be detected. Zachary asked about the privacy issues of logs generated by the witnesses. Suman replied that the clients have the right to block some sensitive information and still have the potential to detect errors. Aditya Akella (University of Wisconsin—Madison) asked about the feasibility of mapping Web application operations to data-store operations. Suman admitted the importance of having a generic SQL-like interface, which will enable the portability of a variety of back-end infrastructures. Aditya wondered whether EVE could deal with things like quality degradation in the streaming service. Suman said that EVE is mainly designed to facilitate error detection in interactive Web applications. Also, it is not practical to keep logs of streaming service, due to the potential for growth in log file size.

Networking & Energy

Summarized by Byung Chul Tak (tak@cse.psu.edu)

Jellyfish: Networking Datacenters, Randomly

Ankit Singla and Chi-Yao Hong, University of Illinois at Urbana—Champaign; Lucian Popa, University of California, Berkeley; P. Brighten Godfrey, University of Illinois at Urbana—Champaign

Chi-Yao Hong presented Jellyfish, a technique for constructing datacenter networks that enables easy incremental expansion without sacrificing network bandwidth. He argued that one critical problem in datacenter networking is to enable incremental expansion and that current datacenter networks did not support this well. One commonly used fat-tree scheme allows bandwidth at a very coarse level limited by the available port count of switches. Upgrading switches in fat-tree schemes also requires full replacement. Other schemes suffer from similar drawbacks.

Jellyfish is based on the random graph, which makes it simple to expand the network to any desired size and provides high resilience from failures. Also, Jellyfish delivers more bandwidth than fat-tree structures in terms of bisection bandwidth. However, challenges remain. An unorthodox routing technique is required, since traditional techniques are mostly based on structural assumptions. He also discussed cabling issues. In order to connect N racks, he suggested using the square root of N as the number of rack clusters.

Ion Stoica asked about the impact of the square root of N on expandability; doesn't enforcing it require recabling other clusters of racks, which goes against the goal of easy expandability? And how does using the square root of N perform compared with a smaller random topology? Finally, in expanding servers, what is the impact of a large number of servers coming together? Hong responded that we do not have to stick with a square root of N option. It could be a starting point for cable configuration.

SilverLine: Data and Network Isolation for Cloud Services

Yogesh Mundada, Anirudh Ramachandran, and Nick Feamster, Georgia Tech

Yogesh Mundada reported that the recent series of data leakage incidents in major clouds made it difficult to adopt the cloud. Threats in the cloud can be classified into attacks on the shared resources and data loss/leakage. In order to address these problems, he proposed a technique called SilverLine to provide data and network isolation for VMs in the cloud environment.

SilverLine delivers data isolation by labeling data via information-flow tracking tools so that an enforcer module at the hypervisor level can check for any unauthorized access. If one malicious user tries to steal data through a SQL injection attack, the data will not be delivered to the attacker at the front end, because any data not owned by the attacker will be filtered by the cooperation of the declassifier and the enforcer. For network isolation, SilverLine makes use of IP address obfuscation and ping response normalization. This prevents attackers from identifying the location of victim VMs on the physical node.

Would introducing delays for network isolation have some performance impact? Yogesh said that there were no measurement numbers regarding performance impact, but he thought that it would be minimal.

Enabling Consolidation and Scaling Down to Provide Power Management for Cloud Computing

Frank Yong-Kyung Oh, Hyeong S. Kim, Hyeonsang Eom, and Heon Y. Yeom, Seoul National University

Frank Yong-Kyung Oh presented measurement studies of performance interference when VMs are consolidated. The goal of his study is to better understand the performance impact of VM consolidation so that it can be used for VM migration scheduling and consolidation in future studies. When several VMs with distinct characteristics are given, one of the goals is to consolidate them so as to minimize the effect on performance and the number of physical machines. This would allow us to turn off some servers, saving power consumption. They specifically looked at three effects: the effect of VM co-location, cache effect, and the effect of CPU thermal throttling. From studying the effect of VM co-location, they found that consolidating VMs that use different parts of resources in the system shows less performance interference; they also found that CPU and memory-intensive applications tend to consume more power than others. The cache effect revealed that disk-intensive VMs show better performance when pinned together with Dom-0 in Xen. And the insight gained from the thermal effect was that consolidating only CPU-intensive VMs may lead to unexpected performance degradation due to CPU thermal throttling. There were no clarification questions after the presentation.

The Data Furnace: Heating Up with Cloud Computing

Jie Liu, Michel Goraczko, Sean James, and Christian Belady, Microsoft Research; Jiakang Lu and Kamin Whitehouse, University of Virginia

Jie Liu presented Data Furnace, which proposes the use of server-generated heat as a household heating solution. He argued that energy can be more efficiently used by dispersing servers to homes or other buildings. This would provide the additional benefit of bringing computation closer to the user and some cost reduction from reusing already existing power infrastructure.

There were several interesting numbers from the presentation: home heating cost is about 10% of total home expenditure, and in the US, home heating is twice the IT energy cost. Another interesting figure was the estimate of number of servers required to properly heat a typical home in different regions of the US. San Francisco showed small variance of the number of servers, whereas Minneapolis showed large variance ranging from 140 servers to fewer than 10 servers.

Towards the end of the presentation, Jie outlined some of the questions regarding the usefulness of this idea, what the hidden costs are, thoughts on residential power capacity, some security issues, and performance concerns. In summary, this was an interesting idea with many challenges.

Ankit Singla asked whether this study considered the cost of moving data in such a distributed setting. Jie said that the cost of paying for the network, which was about $3000 for one server, was included in the calculation. Ankit asked again if this idea was part of realizing the green computing concept. Jie explained that the money is being spent for heating the houses anyway. The installation of a data furnace is a one-time cost at house construction time, just as when you would normally install a furnace. One attendee asked if there was any interest from national security folks. Jie said that this did not represent the official view of the organization and no comments from either the government or industry were received. Next, Orna asked how to compare this with the datacenter in Switzerland that heats all the office water near the datacenter. Jie said that, although similar, it would ultimately be cheaper to move computation around than energy or hot water. Whenever we can put computing near to where heat is needed, we can save costs in terms of transporting other things.

The first question in the session panel discussion was about information visibility between VMs and the cloud infrastructure. Presentations seem to assume that underlying infrastructure needs certain information about VMs in order to work. How much do VMs (or applications) need to tell the infrastructure, will they be able to tell the infrastructure, and how much can the infrastructure trust the information? Yogesh responded that they have built their technique at the VM's OS level, which made things simpler. If they had to go down to the VMM-level, they would have more control over the data items but would also need to understand the higher-level abstractions, which would be non-trivial.

Ion Stoica argued that installation of a Data Furnace at home would require a large maintenance effort. Jie said that one

maintenance concern of the server is perhaps to provide operation without harming the environment; some have studied reliability vs. environmental conditions and have found that conditions were tolerable. Another maintenance concern would be replacing a failed part, swapping disks, and so on, which would require someone to actually go in and take action. Those tasks could be handled by over-provisioning.

One interesting discussion took place about the incremental scalability of datacenters. When current datacenters expand their hardware equipment, they buy servers in large numbers and configure them for a relatively long operation time until the next upgrade takes place. Studies such as Jellyfish and Data Furnace allow the incremental expansion of datacenters: the cloud provider could bring in new servers frequently and in much smaller numbers.

John Wilkes asked why cloud providers would pay to have servers installed in homes when there are not enough workloads to utilize even the current hardware resources in the datacenter. Jie said that the Data Furnace approach would make sense if workload was overcommitted. However, more opportunity arises from content caching near to where contents are created and needed. If data is located where it is needed, it can be served faster. Someone brought up the issue that current network speed is not fast enough. Jie said his analysis included the cost of installing fast fiber network, and its cost did not end up dominating other cost factors. Nevertheless, networking would be the greatest challenge.

Poster Session

No reports are available for this session.

Joint ATC, WebApps, and HotCloud Keynote Address

An Agenda for Empirical Cyber Crime Research

Stefan Savage, Director of the Collaborative Center for Internet Epidemiology and Defenses (CCIED) and Associate Professor, UCSD

See the USENIX ATC '11 reports for a report on this session.

OSes and Frameworks ("There is an OS/App for that!")

Summarized by Henrique Rodrigues (hsr@dcc.ufmg.br)

Unshackle the Cloud!

Dan Williams, Cornell University; Eslam Elnikety and Mohamed Eldehiry, King Abdullah University of Science and Technology, Thuwal, Saudi Arabia; Hani Jamjoom and Hai Huang, IBM T. J. Watson Research Center; Hakim Weatherspoon, Cornell University

Dan Williams said that IaaS providers are mainly focused on giving virtual machines to users. Nowadays, none of the features at the infrastructure level, such as VM migration and CPU boosting, are exposed to the user. In addition, the research community is continuously proposing innovations at the hypervisor level. The problem in the current scenario is that users don't have control over any of these features. To overcome this limitation the group proposes xClouds. The goal of xClouds is to provide extensibility to IaaS-based resource provisioning. Unlike current public clouds, xClouds gives users the ability to leverage their own set of hypervisor-level modules.

Dan presented some design alternatives to implement an extensible cloud. Among the three options, two of them, VMM extensions and exposing the hardware through the VMM, depend on the adoption of a new VMM by the provider. The third option is to use nested virtualization, which doesn't need provider cooperation. This latter option was adopted by xClouds, which was implemented using Xen and tested on an EC2 instance. Dan also presented an evaluation of xClouds, comparing I/O performance between single and nested virtualization setups using a combination of Xen/KVM/HVM.

Himanshu Raj from Microsoft asked if Xen needs any modification to be run on top of EC2 instances as a nested VMM. Dan said that there are some changes required to support the paravirtualized device operations. Muli Ben-Yehuda from Technion/IBM Research commented that both KVM and Xen, in the next release, will have hardware support for nested virtualization, so the performance of xClouds will be better in the near future.

The Datacenter Needs an Operating System

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

For many, the datacenter is like a big computer, where users can run their applications and process their data, either interactively or in a batch-processing fashion. To deal with the growing range of applications and users, Matei Zaharia claims, the datacenter will need an operating system.

The datacenter operating system is not a replacement for the Linux host OS but is software that acts as an operating system at the level of the whole datacenter. Matei listed some features that an operating system should provide to its users: resource sharing, debugging, monitoring, programming abstractions, and, most importantly, enabling independently developed software to interoperate seamlessly. Some platforms that took steps towards providing some of these abstractions are Hadoop, Amazon Services, and Google Stack. The problem with current solutions is that they are all narrowly targeted and are not general/longer-term solu-

tions. In the last part of his presentation, Matei discussed the problems that should be solved to have a practical implementation of a datacenter OS and how researchers can help in this process.

Someone from the University of Toronto asked about the difference between existing cluster OSes and a cloud OS. Matei replied that the main difference is the diversity of users and applications using datacenters. Why did Matei think the idea of building a cloud OS would succeed? Matei said that some approaches are already successful and pointed some of them out.

Large-scale Incremental Data Processing with Change Propagation

Pramod Bhatotia, Alexander Wieder, İstemi Ekin Akkuş, Rodrigo Rodrigues, and Umut A. Acar, Max Planck Institute for Software Systems (MPI-SWS)

Pramod Bhatotia began by discussing the advantages of incremental computation on large-scale datasets. The main idea of incremental computation is to leverage previous processed results to enable more efficient computation of recently updated data. Computing the page rank of a recently crawled URL is an example of a good use case for incremental computation. Two systems for incremental processing, Google Percolator and Yahoo! CBP, were presented. Pramod pointed out that the main disadvantage of current approaches is the need to rewrite existing applications in new programming models using dynamic algorithms, which are harder to design.

Current large-scale applications are developed using static algorithms and well-known programming models. The goal of Pramod's work was to make these applications as efficient as the ones that make use of incremental computation and dynamic algorithms. His presentation focused on how to achieve this goal in a MapReduce-based application. Their approach was to take an unmodified program and automatically make it incremental by (1) dividing computation into sub-computations, (2) keeping track of input dependencies between each sub-computation, and (3) recomputing only the computations affected by input changes.

Pramod then discussed some of the challenges to building such a solution for MapReduce-based applications. To evaluate the performance gains of their solution, the runtime speedup was compared against an increasing input dataset.

John Wilkes (Google) asked if there is any restriction on implementing the reduce task in order to make it divisible into sub-computations. Pramod said that developers should use MapReduce combiners to achieve a fine-grained division of the reduce task. Christopher Colohan (Google) pointed out that on large MapReduce computations usually the smaller you make your task, the higher the management overhead is, which is the opposite of the results shown. Pramod said that when you are making use of incremental computation, this overhead only affects the performance in the first computation.

TransMR: Data-Centric Programming Beyond Data Parallelism

Naresh Rapolu, Karthik Kambatla, Suresh Jagannathan, and Ananth Grama, Purdue University

Naresh Rapolu explained why current data-centric processing models such as MapReduce or Dryad are unable to deal with the side effects of parallelized algorithms that present data dependencies. To exemplify the problem, he used a simple MapReduce-based word count application. The limitation is mainly due to the deterministic replay-based fault tolerance model adopted by these programming models. This fault model assumes that the application semantics won't be affected by re-running a computation task in the case of task failure. However, not all algorithms have a data-parallel implementation compliant with this fault tolerance model.

To support parallel algorithms' side effects and overcome the limitation of current data processing frameworks, they proposed a transition-based MapReduce programming model. The key ideas of their approach are to develop every data-centric operation as a transaction and to use a distributed key-value store as the shared memory abstraction accessed by all operations. The concurrency model is based on two operations: put and get. With optimistic reads and buffered writes they could build a programming model that doesn't require any locks.

For evaluating their programming model, Naresh presented the speedup of two algorithms implemented on their current prototype, which uses Hadoop and HBase as the key-value store. The algorithms are the Boruvka's Algorithm for finding a graph's minimum spanning tree and the Push-Relabel algorithm to find the maximum flow of a graph. Both experiments resulted in a speedup close to four times for 16 nodes.

In the panel discussion, Christopher Colohan (Google) asked whether we are able now to design an API that will be widely accepted and that will last for a long period of time. Matei answered that we can at least try starting from lower-level primitives and that future extensions or changes to this API would not be a problem, because even standard OSes have had to make changes to support newer technologies. Dan added that if we want this API to last for a long time, it needs to be user-centric. Ion Stoica asked Naresh if he had any idea how to improve their system in order to achieve linear scal-

ing? Naresh said that there are a lot of parameters the user can tune to have better performance, but most of them are application-dependent.

Steve Ko (SUNY Buffalo) asked how we can deal with the size of a datacenter while designing an operating system for it. Matei replied that most of the scalability problems still have to be solved for the systems being built today. The interesting thing about designing a datacenter operating system is that once one problem has been solved, it is possible to incorporate the solution into the datacenter OS and, therefore, we won't need to keep solving the same problem for each independent platform.

Rodrigo Fonseca (Brown University) commented about possible optimizations for xClouds and also pointed out that some of them will depend on provider cooperation. Dan replied that the deployability of xClouds among multiple vendors was more important than specific optimizations for individual providers.

Glenn Ammons (IBM Research) asked Pramod if his group had evaluated their framework using real-world applications. Pramod replied that he didn't show the results because of the time constraint but that they tested their framework using the Apache Mahout library.

Rodrigo asked why Matei didn't mention the word "cloud" in his presentation and whether he thinks that there are differences between a public datacenter and a private datacenter. Matei said it was because they think that the datacenter operating system should be generic enough to run on both private and public datacenters, regardless of the differences between them.

Performance

Summarized by Sahil Suneja (sahilsuneja@gmail.com)

Modeling the Parallel Execution of Black-Box Services

Gideon Mann and Mark Sandler, Google Inc.; Darja Krushevskaja, Rutgers University; Sudipto Guha, University of Pennsylvania; Eyal Even-Dar, Final Inc.

Mark Sandler presented work in which the goal is to estimate the impact of a change, deep down in the call stack following a user request, on the latency of a higher-level service in the hierarchy.

Call trees do not encode the parallelism structure among multiple calls, and this acts as a hindrance to accurate latency estimation up the stack. The proposed approach automatically reconstructs the flow of a service by looking at the nature of overlapping between multiple RPCs and combining multiple invocations of the service to generate a consistent call graph. Since a service can have multiple flows (e.g., different RPCs depending upon cache hit/miss), two different schemes are employed for deciding whether two invocations follow the same flow-clustering invocations having identical control flow graph (better) vs. invocations calling identical sets of lower-order services. During the training phase, using either of these two approaches, flows are constructed, and average latencies at each node are recorded using actual traces. Then, in the testing phase, for a given set of latencies at child services, best matching flow is found and simulated and its latency at the parent is reported.

This approach can allow modeling service dependencies and aid in detecting potential problems caused by changes down in the stack.

No questions were raised at the end of the talk.

CloudSense: Continuous Fine-Grain Cloud Monitoring with Compressive Sensing

H.T. Kung, Chit-Kwan Lin, and Dario Vlah, Harvard University

Chit-Kwan Lin emphasized the relationship between performance and monitoring—the more the available information about the state of a cloud, the better the decision-making with regard to its management. With increasingly interactive applications, finer-grained status information would prove beneficial in improving application performance.

However, the major challenge to fine-grained monitoring is the bottleneck at the collection point. The example used was that of MapReduce straggler detection, where the sooner a straggler is detected, the earlier the job can complete. This requires global relative comparisons and, in turn, global status collection. To overcome the collection bottleneck, the status stream can be compressed in the network. But since distributed compression is hard, a compressive sensing technique could be used which increases reporting granularity via in-network distributed compression, so the largest anomalies could be detected first and with few reports. The proposed solution is a switch design for compressive sensing called CloudSense. For a single rack, status from each node is collected into the signal vector. Random projections are computed onto low-dimensional space, generating measurement vectors which are sent to the master. After recovering the original signal vector, the master solves for L1 minimization by linear programming. In the two-rack case, an aggregation switch is added that summarizes the measurement vectors so that the data sent over links does not increase.

Rodrigo Fonseca from Brown University inquired about the signals to which the proposed technique was applicable. Chit Kwan's response was to use this primarily for performance counter reporting similar to MapReduce task progress

reporting. He also hinted at obtaining inputs from industry colleagues. Another question sought clarification on means to identify when the values of two of the design parameters, signal sparsity and number of measurements, were sufficiently large. Chit-Kwan said that since signal sparsity is a static system property, it isn't supposed to be set dynamically. But for the number of measurements, there is a tradeoff with regard to loss in decoding accuracy. If it is too large, mistakes could be made, but the model gives very few false positives, while the false negatives could be dealt with easily—the more measurements are obtained, the better the decoding sensitivity is.

Virtual Machine Images as Structured Data: The Mirage Image Library

Glenn Ammons, Vasanth Bala, Todd Mummert, Darrell Reimer, and Xiaolan Zhang, IBM Research

Just as a VM image puts application configuration in one place, similarly an image library collects all enterprise configuration together. This aids in simplifying maintenance operations such as patching and security scans, allows version and access control, and permits offline analyses, search, mining, and comparisons. Glenn Ammons presented the Mirage virtual machine image library: while the hypervisor provides an unstructured VM image, Mirage presents the image as more structured data, allowing faster image deployment by indexing the file system structure.

The task of converting all images from Xen to KVM while using RC2 (Research Compute Cloud) was the first use case presented. This is inherently an iterative process—find a bug, fix it, and try again. The version control features of Mirage are especially useful in this scenario, allowing for rollback and comparisons for debugging. The second use case involves employing the IBM Workload Deployer, which deploys images to machines in an enterprise while providing an enterprise configuration in an all-in-one-place view. Its customers typically have complicated workflow environments: for example, the OS team creates an image while the middleware team installs the middleware to create the product consumed by the application team where apps are installed. When the OS team updates the OS, Mirage allows computing the difference between the different versions, among other things, and creates new middleware automatically.

During the Q&A, Glenn elucidated the fact that manual labor is still needed, even with the automated version controlling and patching, although it is reduced to just verifying the automated result. Someone asked about policies to deal with concurrent changes to the images: for example, when the middleware team changes the system configuration as a result of their work and the OS team upgrades the version, their change might conflict with the current configuration. Glenn clarified the need to verify the results, as the method employed is opportunistic in nature.

Accelerating the Cloud with Heterogeneous Computing

Sahil Suneja, Elliott Baron, Eyal de Lara, and Ryan Johnson, University of Toronto

Elliott Baron presented the idea of leveraging the AMD Fusion kind of heterogeneous processors, which combine the GPU and CPU on the same chip, for accelerating and offloading cloud management tasks at the hypervisor level. The on-chip architecture allows low-latency memory access to the GPUs, overcoming the traditional bottleneck of access over the PCI bus.

Various use cases were presented—memory scrubbing, memory deduplication, memory compression, virus scanning, batch page table updates, etc. A case study of hashing-based page sharing was presented, indicating significant speedups over the CPU versions, as expected. Even with the Fusion architecture, memory copying between CPU and GPU was exposed, even though the two cores share memory. Also, hardware management for sharing the on-board GPUs between guest VMs and the hypervisor was discussed. The idea of incorporating time and space multiplexing was proposed.

Chris Colohan from Google asked whether GPUs could perform I/O operations on the storage stack. That could enable accelerated scrubbing of hard drives. The need for data to be in memory before the GPUs could access them was flagged as the hindering factor by Elliott, who acknowledged the benefits of the proposed idea.

Ion Stoica (UC Berkeley) suggested keeping data compressed in memory with on-demand decompressing. Elliott had already hinted at this in his talk.

Matei Zaharia (UC Berkeley), who hadpresented his work on OS for datacenters in the pre-lunch session, asked about the use cases of Mark's work. In Mark's opinion, the main use case is to actually detect the root cause of your problem. This is because latencies can be very different even when datacenters are running identical code. In this case, his technique allows one to try different hypotheses and figure out what's the most likely reason for the variability.

Chris Colohan from Google asked a question combining the ideas from two of the talks—monitoring and GPGPU computing. He wondered if GPGPU could be used for monitoring applications—security monitoring and intrusion detection. The other speakers answered yes, but Elliott, while agreeing that the proposed idea seemed very logical, made a playful comment regarding Windows users not liking the antiviruses

using their CPUs, and hence using GPUs to offload the CPU for this task made even more sense!

John Wilkes from Google wondered whether all this could be generalized. What could be present in a general framework for monitoring and what would the standard libraries offer? Chit-Kwan's opinion was that of a datacenter-wide bus with APIs at the bus level and no higher, where every status stream would be represented by a type that could be published along with the granularity. Elliott said that his work was not specifically in the monitoring game, but he mentioned one important feature for his general framework—to keep GPU interaction out of the hypervisor but in dom0 space.

Michael Kozuch from Intel put forward an open-ended question—the importance of performance and its ranking in the hierarchy of scale, reliability, security, etc. Elliott rated security above performance, while Chit-Kwan considered performance more important than monitoring. Glenn's view was that more important than performance is the visibility into performance. Mark considered performance as being of primary importance.

John Wilkes raised a debatable issue by stating that emphasis on performance should be to a lesser degree in academics. In his opinion, performance is relatively easier to add, and there are much more interesting things to be found outside the performance space. While Glenn agreed with this notion, Elliott clarified that his project is not so much about getting some performance points—it's about utilizing the new architecture that's hitting the markets. In Mark's view it depends on the problem being addressed—for example, number crunching performance is more important than serving user requests. Eyal de Lara (University of Toronto) said that people are still working in the performance space and it is important for the datacenters—not that a particular optimization adds some small percentage improvement, but definitely that an idea can reduce datacenter size by half. If the return is a small delta improvement, then the original comment made sense, but from there to assuming that we have all the performance we need and we don't really need to improve on it is not correct.

Byong Gon continued on the last discussion and inquired about predictable performance. Mark jokingly answered by contradiction—he was more comfortable in answering what could make performance unpredictable. He believed there was no single answer to the original question—perhaps having sufficient resources. In his opinion, consistent performance was more important. Going back to the last discussion, he agreed that 10% performance improvement was not very important, but 10x was definitely important.

Michael Kozuch put forward the final question, which dealt with hardware innovation possibilities. Elliott believed heterogeneity was an important step forward, with network processors and FPGAs finding their way onto chip in the future. Glenn lightheartedly routed the question to the Google guys to put forward their demands and requirements to the academics, with reference to the panel discussion on day one.

Cloud Computing and Data Centers (Joint Session with ATC)

See the USENIX ATC '11 reports for a report on this session.

Invited Talk (Joint Session with ATC)

Helping Humanity with Phones and Clouds

Matthew Faulkner, graduate student in Computer Science at Caltech, and Michael Olson, graduate student in Computer Science at Caltech

See the USENIX ATC '11 reports for a report on this session.