Conference Reports

In this issue:

9th USENIX Symposium on Networked Systems Design and Implementation (NSDI '12)

San Jose, CA
April 25–27, 2012

Opening Remarks and Awards Presentations

Summarized by Rik Farrow (rik@usenix.org)

Steven Gribble introduced the workshop, saying that 30 papers out of the 169 submitted were accepted, and all submitted papers were shepherded as well. There were over 250 attendees on the opening day—not a record, but close to it. The Best Paper award went to Matei Zaharia and his co-authors for "Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing" (see their "Fast and Interactive Analytics over Hadoop Data with Spark" article in this issue of *;login:*). Steven announced a new award, the Community Award, for papers that make code available to the community. This went to "How Hard Can It Be: Designing and Implementing a Deployable Multipath TCP," by Costin Raiciu et al., with the Zaharia paper and "Serval: An End-Host Stack for Service-Centric Networking," by Erik Nordström et al., receiving Honorable Mentions.

Big Data

Summarized by Marcelo Martins (martins@cs.brown.edu)

CORFU: A Shared Log Design for Flash Clusters

Mahesh Balakrishnan, Dahlia Malkhi, Vijayan Prabhakaran, and Ted Wobber, Microsoft Research Silicon Valley; Michael Wei, University of California, San Diego; John D. Davis, Microsoft Research Silicon Valley

Mahesh Balakrishnan started by saying that flash drives create optimization opportunities in data centers that cannot be achieved with hard disks. He claimed that flash drives can be leveraged in a distributed environment without partitioning, hence keeping strong consistency, while guaranteeing cluster performance. Mahesh described how CORFU achieves this objective starting from its software and hard-

ware components. CORFU exposes a simple API to server applications that view the flash cluster as a single shared log supporting read and append operations. In parallel, CORFU utilizes custom-designed, network-attached flash units to reduce power draw and cost. That flash drives allow random reads with minimum overhead is the key to CORFU's scalability. Mahesh proceeded to explain the client-centric protocol used to access the shared log and how it guarantees ACID (atomicity, consistency, isolation, durability) conditions. Each client keeps a projection that maps logical entries into flash units and needs to request tokens from a sequencer machine to reserve an append operation to a specific unit. A write-once semantics is used to easily identify the tail of the log. Mahesh mentioned that the sequencer does not represent a single point of failure, as it is used only as an optimization asset. CORFU leverages chain replication for integrity and durability.

He then proceeded to describe how CORFU handle two types of failures: flash-unit and client failures. In the case of a flash-unit malfunctioning, reads are served by the remaining original drive, while new writes are posed to both original and substitute drives. In the background, the substitute drive replicates old data from the original drive. Projections storing the new mapping are interchanged between clients using the Paxos protocol. Mahesh claimed that the entire recovery procedure takes between 10 and 20 milliseconds for a 32-drive cluster. In the case of a client failure, if a client reserves a disk from the sequencer but crashes before completing its write, other clients can fill in the referred disk with invalid data to guarantee appending ordering.

Finally, Mahesh presented CORFU's performance evaluation . For throughput, CORFU reads scale linearly as more flash drives (up to 32) are added to the cluster. Appends also scale linearly with the number of flash drives, although the sequencer becomes the bottleneck when more than 31 drives are presented to the cluster. In its current implementation, the sequencer can serve up to 190,000 appends per second without performance degradation, although Mahesh believes that this number can go up to one million appends per second if the sequencer logic is implemented in hardware. In his final slide, he commented on the relationship between CORFU and Paxos. While Paxos-based protocols suffer from I/O bottlenecks due to space partitioning, CORFU stitches multiple flash-unit chains into a single virtual storage unit. The partitioning occurs over time, and therefore there is almost no I/O bottleneck.

David Andersen (CMU) asked about the influence of newer approaches to NVRAM technology (e.g., memresistors, PRAM) on CORFU's design. Mahesh explained that CORFU takes into account that current flash units are block-addressable, while the newer technology is byte-addressable. CORFU assumes a sequential-appending design given the write-block property of flash drives, but Mahesh believes that a byte-addressable flash storage could enable arbitrary appending and avoid data padding. John Dunagan (MSR) asked for clarification of the benefits of separating the flash units from the client. Mahesh explained that in CORFU, servers should communicate with all flash units in order to provide scalability, and therefore its design does not preclude data locality. Ben Reed (Yahoo! Research) questioned the validity of a global shared-log abstraction for different server applications. Mahesh agreed with Ben's position and clarified that CORFU does not make such an assumption and can provide a different log per application.

Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing

Matei Zaharia, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma, Murphy McCauley, Michael Franklin, Scott Shenker, Ion Stoica (University of California, Berkeley)

- *Awarded Best Paper!*
- *Awarded Community Award Honorable Mention!*

Matei Zaharia gave a lively presentation about the internals of Spark. The recipient of the Best Paper Award started by commenting on the inefficiency of the MapReduce programming abstraction in attending the data-sharing needs of particular "big data" applications (e.g., iterative processing and interactive queries). He gave examples of how Hadoop (an open-source implementation of MapReduce) does not scale properly, due to its design based on replication and network and disk I/O between stages. Matei then enticed the audience by asking how one could get in-memory data sharing and fault tolerance without replication across nodes to avoid I/O bottlenecks, and introduced his solution: Resilient Distributed Datasets (RDDs). RDDs work as an immutable, distributed collection of records that can only be recovered in a coarse-grained manner: the system only logs the deterministic operations between stages and reapplies the logged operations to the missing record sets in case of failure. To determine which record sets should be rebuilt, RDDs track the graph of transformations that builds them (*lineage*). High write throughput is guaranteed, as RDDs work close to memory-bandwidth limits.

Matei then presented Spark, a programming interface that uses RDDs as its abstraction for operations on large datasets. Matei proceeded with examples of how the Spark API can be used to solve real-world analytical problems. He showed how an administrator can use the API to solve a large log-mining problem and how the API requests translate into interactions between the master and workers in a

distributed environment. Spark, implemented in Scala, uses lazy evaluation to avoid unnecessary data operations. As the workers generate the first result sets, they store these sets in local memory. Subsequent queries take advantage of the cached results, leading to almost instantaneous responses. As evidence, Matei claimed that Hadoop takes about 170 seconds for responding to an on-disk search query from a 1 TB Wikipedia dataset, while Spark takes only 5–7 seconds. His next example showed the latency savings for an iterative machine-learning algorithm. Although the system pays the price of from-disk data loading in the first Spark query, subsequent iterations take advantage of in-memory shared data and respond quickly. Even in the case of failures, the recovery overhead is far from the full disk access (119s vs. 81s).

Spark is built from scratch, can load data from the same sources as Hadoop, including HDFS and S3, and is available as open-source software (http://www.spark-project.org). Matei also mentioned that RDDs can be used to easily express different parallel models, such as graph processing, iterative MapReduce, and SQL. Matei proceeded with a demonstration of his system running on a 20-node EC2 cluster. He showed how a user would use the Scala interpreter to run Spark queries and again emphasized the benefits from in-memory data sharing. He showed a live example of how a large Wikipedia query that took about 19 seconds to load data from disk only took 1 second for subsequent queries. Finally, Matei pinpointed the extensions to Spark that companies and other research projects have been developing since its open-source announcement as proof of its applicability to diverse scenarios.

Matei was asked about the possibility of memory leaks due to many RDD structures kept in memory as a result of intensive use of Spark. He replied that his system implements garbage collection to avoid leaks. John Dunagan (MSR) followed with a comment on the benefits of persistence for data sharing so that nodes will not lose sharing context once they switch tasks and asked how that would affect in-memory RDDs. Matei agreed with John's comment and said that his system leverages data locality via delay scheduling to keep the benefits of low I/O overhead. Rik Farrow asked whether he had considered persisting some of the RDDs instead of using LRU for replacement in case of low memory. Matei argued that one could take advantage of a hybrid solution that would consider both cases. He then showed one of his backup slides demonstrating the performance overhead relative to the amount of working set kept in memory (the missing data would be recomputed or retrieved from disk). In his scenario, for each 25% less working set kept in memory, the iteration time would increase by 15 seconds on average.

Camdoop: Exploiting In-network Aggregation for Big Data Applications

Paolo Costa, Microsoft Research Cambridge and Imperial College London; Austin Donnelly, Antony Rowstron, and Greg O'Shea, Microsoft Research Cambridge

Paolo Costa started with a brief recap of MapReduce and then focused on a particular problem his group at MSR was trying to solve: improving the performance of the shuffle and reduce phases of MapReduce. He argued that the all-to-all traffic pattern of the shuffle phase is an encumbrance to data-center networks and that the final results of a MapReduce job are usually much smaller than the intermediate results before the shuffle phase. As an example, Facebook reported that its usual final results were 5.42% the size of intermediate data. A few solutions exist to reduce the input size of shuffle and reduce phases. Combiners are limited to in-node aggregations, while rack-level aggregation trees easily create network bottlenecks on the switch links.

Camdoop's goal is to avoid this bottleneck, performing the combiner function as part of the network protocol by harnessing packet aggregation at each hop. Paolo then introduced CamCube, MSR's solution to perform in-network processing. Instead of using routers/switches, CamCube builds a 3D-torus topology, where servers are directly connected to each other and can intercept and process packets. The 3D topology facilitates server naming and fault-tolerance mapping. Paolo then explained the logic behind the topology choice and network design. Each server has the capability of processing packets and aggregating them before forwarding, avoiding the link-contention problem of rack-level aggregation trees.

In the second part of his talk, Paolo talked about Camdoop's innards. As a first goal, Camdoop does not modify the MapReduce model. Each server runs map tasks locally. The system keeps a spanning tree where combiners aggregate results from mappers and send them to their parents. The root node runs the reduce task. To guarantee network locality, the parent-children relationship is mapped to physical neighbors (using the 3D-topology naming). One problem with this solution is the load distribution. To overcome this issue, Paolo described a striping mechanism that distributes data across disjoint trees formed from the same cube topology. Not only load is distributed, but all possible links can be used if six disjoint trees are built.

The third part of Paolo's talk encompassed evaluation results and more implementation details. Paolo gave a quick overview of the 27-server CamCube testbed built by MSR and also a 512-server (8x8x8) packet-level simulator. One of the insights we can take from Camdoop is that the shuffle and

reduce phases run in parallel, resulting in a performance boost from the traditional MapReduce sequential model. Considering that all data streams are ordered, once the root of the spanning tree receives a packet, it can start the reduce function and no intermediate results need to be stored. For evaluation purposes, Paolo compared Camdoop with an implementation of MapReduce that runs shuffle and reduce in parallel, and a restricted version of Camdoop without in-network aggregation. First, he showed via benchmark applications that Camdoop can compete with the state-of-the-art Hadoop and Dryad. Next, he considered the effects of data-aggregation size and reduce-phase types (all-to-one vs. all-to-all) on task running time. For the all-to-one case, as we move from no aggregation (output size is equal to intermediate size) to full aggregation (output size is minimal compared to intermediate size), Camdoop presents up to a tenfold latency improvement over no-aggregation Camdoop. When comparing no-aggregation Camdoop with MapReduce, he pointed out that the aggregation size brings no improvement, since there is no actual data aggregation. Still, Paolo noted that there is a threefold latency improvement of no-aggregation Camdoop over MapReduce, due to other optimizations. Considering the full-aggregation scenario, Paolo also talked about the impact of the number of reduce tasks on running time. For both no-aggregation and MapReduce, as the number of reduce tasks increase, the running time becomes smaller. However, Camdoop returns the same performance numbers independent of the number of reduce tasks. Furthermore, Camdoop presents a tenfold and fiftyfold latency improvement over the other two solutions, respectively. Finally, Paolo showed that even for interactive applications (e.g., Bing Search, Google Dremel) which produce small-sized intermediate data, Camdoop is capable of taking advantage of in-network aggregation to quickly run MapReduce stages.

David Oran (Cisco) asked whether Paolo had any measurements on the effects of latency on Camdoop. Paolo replied that latency exists due to server-based forwarding on the order of hundreds of milliseconds, but this loss could be compensated by traffic reduction resulting from in-network aggregation. Rishan Chen (Peking University) asked about the scalability of Camdoop in terms of machines. Paolo answered that no matter the number of servers participating in the system, each node still maintains the same number of neighbors. What changes is the number of hop counts between nodes, which may increase latency, but this can be compensated for by on-path data aggregation. Ankit Singla (UIUC) asked for clarification of the differences between the full-bisection bandwidth topology and the one used by Camdoop. His impression was that in the former, all shuffle-phase transfers occur in parallel, and in Camdoop, parents have to wait for their children to finish before transferring data. Paolo partially disagreed with his observation, saying that transfers in the full-bisection topology do not occur as early as one might imagine and also involve waiting time.

Wireless

Summarized by Marcelo Martins (martins@cs.brown.edu)

WiFi-NC: WiFi Over Narrow Channels

Krishna Chintalapudi, Microsoft Research India; Bozidar Radunovic, Microsoft Research UK; Vlad Balan, USC; Michael Buettener, University of Washington; Srinivas Yerramalli, USC; Vishnu Navda and Ramachandran Ramjee, Microsoft Research India

Krishna Chintalapudi started with a high-level overview of the differences between conventional WiFi radio and WiFi-NC. Conventional radio leverages one wide channel at a time for data transmission/reception and permits only one pair of communicating devices. Conversely, WiFi-NC proposes the simultaneous usage of several narrow channels for the same purpose and allows multiple simultaneous transmissions and receptions from several devices. Krishna said that despite the increase in WiFi data rates in the last 10 years, there was no corresponding increase in throughput. While the time taken to actually transmit data has decreased with higher transmission rates, the overhead associated with setting up the transmission has not changed at all. He explained that WiFi-NC increases the communication efficiency by slicing a wide data channel into lower data-rate channels and uses them to transmit data in parallel, which also parallelizes the associated overhead (CSMA, preamble, SIFS, and ACK). Krishna claimed that although the progress on the WiFi standard resulted in more bits being encoded per Hz, the current trend from new revisions of the 802.11 protocol to make channels wider does not always work. Access points operating in different frequencies must resort to the lowest frequency so that clients can coexist and avoid starvation. WiFi-NC can solve the coexistence problem by dividing the wide frequency channel into narrow channels and allowing independent data transmission and reception. Finally, he said that WiFi-NC can use the whitespace fragmented spectrum by creating low-rate channels that exist between other medium sources.

The key component behind WiFi-NC is its compound radio. Krishna said that the new radio design uses digital signal processing to create narrow channels without modifying the analog radio front end. To permit simultaneous transmissions and receptions, the compound radio should avoid radio interference. For transmissions and receptions, elliptic filters are the best for interference attenuation. He briefly went over other complex design challenges that should be over-

come before enabling simultaneous communication without interference. Next, Krishna briefly touched on how the NC radio selects the right operating point in the white spectrum using an optimization algorithm named TMax, although no details were provided during the talk.

The evaluation of WiFi-NC basically demonstrated that its design and implementation can overcome the coexistence, multiple simultaneous communication, and starvation-avoidance problems. Krishna produced graphs showing that the greater the number of narrow channels, the more efficient the communication is for a single link. For instance, by increasing the slicing of a 20 Hz channel from two to eight smaller channels, the efficiency of a 300 Mps link increased from 20% up to around 50%.

Aaron Gember (University of Wisconsin) asked whether one could assume that all nodes have the same white spectrum available to them and whether there would not be interference issues between nodes. Krishna said that there is no such assumption and that if nodes have overlapping visibility on parts of the spectrum, they will use narrower channels to fairly share this available spectrum. Suman Banerjee (University of Wisconsin) asked about the percentage of actual data transmission given the parallel transmissions using WiFi-NC compared to the amount of data related to radio control. Krishna first explained how his solution confronts preamble dilation on the physical layer resulting from the interference-avoidance algorithm. Later, he made a data transmission estimate of roughly 40%. Brad Karp (University College London) asked about the effect on the radio cost from using multiple components to enable greater communication efficiency, as in the case of WiFi-NC. Krishna replied that he is not a radio manufacturer and could not give an estimate. He mentioned that the front-end already comes with the filters, but we need to rely on Moore's Law to expect the prices to go lower. Finally, he added that the cost would be proportional to the number of transmitter/receivers.

Catching Whales and Minnows Using WiFiNet: Deconstructing Non-WiFi Interference Using WiFi Hardware

Shravan Rayanchu, Ashish Patro, and Suman Banerjee, University of Wisconsin Madison

Shravan Rayanchu asked how a user can determine the source of the WiFi disturbance that leads to his annoying video-buffering issue. It turns out there are many reasons for it, including weak signal and interference from WiFi and other devices (cordless phones, microwave ovens, Bluetooth receivers, etc.) that share the spectrum with the access point (AP). The idea behind WiFiNet is to construct a system capable of identifying the sources of WiFi interference due to non-WiFi devices and quantifying the impact on the link quality. He said that identifying the culprits of non-WiFi-to-WiFi interference is harder, since there is no simple way for a WiFi card to detect interference from, say, a cordless phone, and also there is no easy way to identify the non-WiFi device (cf. MAC addresses on WiFi cards).

WiFiNet leverages the power of AirShark, software running on access points capable of detecting non-WiFi devices. AirShark relies on WiFi cards capable of taking spectral samples from the communication medium and, with the help of a decision-tree classification algorithm, categorizing non-WiFi devices using tags. The classification algorithm works based on the signature of each spectral sample. Shravan's group at Wisconsin extended AirShark to perform interference quantification and localization. A WiFiNet controller coordinates the distributed view of different non-WiFi devices from access points. The first challenge of WiFiNet is to differentiate and identify non-WiFi samples. To do this, multiple synchronized APs observe the medium and collect samples that will later be analyzed. The samples can tell whether they belong to the same device if they share the same period and frequency range. Still, we need to separate different instances from the same device type. Shravan did not go into detail, but he said he used a clustering algorithm to identify multiple instances of a given device.

The next step is to localize the non-WiFi device. Assuming that the AP locations are known, one can implement several algorithms using the received signals. WiFiNet's choice is based on a propagation model from the devices. However, since the transmit power of a device cannot be estimated by the controller, WiFiNet uses the difference in the received power between pairs of APs to estimate the non-WiFi device position. Shravan followed with an evaluation of two different deployments (of 4 and 8 APs) that shows that the propagation model produces the lowest median error (less than 4 meters) compared to other algorithms (Fingerprint, Centroid, and Iterative). The last piece of WiFiNet is an estimation module for the impact of interference of detected devices on a wireless link. Given that each AP point is capable of identifying transmission overlaps between WiFi frames and non-WiFi pulses, the WiFiNet controller can correlate frame losses with the transmission overlaps to generate a relative number corresponding to the loss probability.

Shravan then made a few remarks on different experiments for testing the effectiveness of his proposed framework. The first experiment concerned the interference impact from a single non-WiFi device of various types on the link quality. It turns out that in most cases the WiFiNet estimate is very

close to the ground truth, with errors of less than 10% for more than 90% of the cases. For the scenario with multiple devices, WiFiNet is still capable of producing estimation values close to the ground truth, with errors within 15% for more than 85% of the cases.

Michael Nowlan (Yale University) asked whether WiFiNet used a dictionary to identify device signatures and, if so, whether it would require updating. Shravan replied that AirShark contains a dictionary of devices that only includes the most popular ones, and it does require updates. Srinivas Narayana (Princeton University) had two questions: (1) the reason behind the need to differentiate instances of the same device type, (2) the sensitivity of the estimation algorithm to the number of available access points. Shravan answered that identifying an instance could help one localize the culprit and proceed to mitigate the interference problem. Regarding algorithms, he said that the localization algorithm is actually more sensitive to the number of available APs, while the estimation could still be performed with a single AP. Still, both the estimation and localization are fairly accurate even with a low number of APs. Brad Karp (University College London) wondered about the possibility of using more advanced radio hardware and its impact on WiFiNet design. Shravan answered that sophisticated hardware definitely can improve the interference estimation problem and there are works that make use of it; however, WiFiNet's contribution was designed to harness current off-the-shelf WiFi cards and see to what extent they can be used to solve the interference-identification problem.

Content and Service-Oriented Networking

Summarized by Katrina LaCurts (katrina@csail.mit.edu)

RPT: Re-architecting Loss Protection for Content-Aware Networks

Dongsu Han, Carnegie Mellon University; Ashok Anand and Aditya Akella, University of Wisconsin—Madison; Srinivasan Seshan, Carnegie Mellon University

Dongsu Han addressed the problem of minimizing data loss in delay-sensitive communications. Minimizing loss is a particular challenge here because of time constraints; it often takes too long to retransmit data. Today, FEC (Forward Error Correction) is one of the most popular methods to protect against loss, but it's difficult to tune and is susceptible to bursty losses.

Content-aware networks, however, are equipped to do caching at the routers, which minimizes the bandwidth cost of redundancy (this is referred to as redundancy elimination, or RE). The RPT (Redundant Packet Transmission) system exploits this capability, introducing redundancy in a way that the network understands.

To retain robustness, sources in RPT send one copy of the original packet along with multiple "redundant packets." These redundant packets are very small and contain a reference to the original content in the cache. If the original packet is lost, the content can be recovered using the redundant packets and the cache.

Compared to FEC using Reed-Solomon, RPT has less overhead and a lower data loss rate. When tested on streaming video, the quality of the received video stream (measured via PSNR) was better with RPT than FEC. RPT has the added benefit that delay is decoupled from redundancy, so the user is able to more easily control both the quality of redundancy and the maximum delay incurred. One downside of RPT is that its flows get prioritized, which can lead to unfair bandwidth allocation when non-RPT traffic also exists in the network.

Zili Francisco asked for clarification on the redundant packets. Dongsu noted that these packets are not compressed packets—rather, they're deduplication packets that only contain a pointer to what the router should look up in its cache.

Fitz Nowlan (Yale) asked why the received quality of FEC (10,9) was so poor compared to other FEC schemes (Fig. 8 in the paper). This happens because there is a high chance that two or more packets are lost in the same batch, and if two or more packets are lost, you lose data.

David Nolan asked whether they had studied the amount of state that has to be received by each router when many RPT flows are coming into it, and whether they've looked at doing explicit expiration from the cache once a copy of the data has been successfully delivered. Dongsu noted that they're using RE as-is on content-aware networks, and so aren't actually modifying anything about the underlying network.

Serval: An End-Host Stack for Service-Centric Networking

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

► *Awarded Community Award Honorable Mention!*

Erik Nordström began by noting that the Internet of the 1970s was designed for accessing hosts. Today, we access services running in data centers replicated across many machines. Users are agnostic to the location/host and are only interested in the service.

Accessing a service involves locating a nearby data center, connecting to the service (establishing a flow, load balancing, etc.), and maintaining connectivity to the service, even through migration. Today's overloaded abstractions don't support these operations particularly well. Serval provides a new naming abstraction that gives clean role separation in the network stack by providing a service-level control/data plane split.

In Serval, the network layer stays unmodified, and a service access layer (SAL) is inserted above it. Applications now connect to services via ServiceIDs, which are allocated to service providers. Data for the service is sent over flows which are addressed by FlowIDs, which are then attached to particular endpoints (IP addresses).

The SAL contains a flow table and service table. The service table maps ServiceIDs to actions such as forward, delay, etc. In the control plane, the Service Controller uses this table and communicates with other service controllers, with DNS, with OpenFlow, etc. Serval can handle load balancing and flow migration and is incrementally deployable.

Performance-wise, having a single Serval TCP connection that never breaks saves hundreds of MBs of data, and is just slightly slower than TCP/IP (in their tests, roughly 933 vs. 934 Mbps). The throughput of the service table is a bit worse, but hasn't been optimized yet. Today, it can support 140K connections per second. Serval extends the SDN model to the network edge, while OpenFlow primarily focuses on layer-2/layer-3 abstractions.

Emin Gün Sirer (Cornell) asked about cross-layer interactions (e.g., how Serval interacts with congestion control). Serval can tell certain layers to freeze state, so bad cross-layer interactions can be avoided.

Reliable Client Accounting for P2P-Infrastructure Hybrids

Paarijaat Aditya, Max Planck Institute for Software Systems (MPI-SWS); Mingchen Zhao, University of Pennsylvania; Yin Lin, Duke University and Akamai Technologies; Andreas Haeberlen, University of Pennsylvania; Peter Druschel, Max Planck Institute for Software Systems (MPI-SWS); Bruce Maggs, Duke University and Akamai Technologies; Bill Wishon, Akamai Technologies

Paarijaat Aditya discussed hybrid CDNs. In these CDNs, clients download from peers as well as CDN servers. This gives us the scalability of P2P networks as well as the reliability/manageability of centralized systems. However, clients are untrusted, and the infrastructure can't observe P2P communication, leading to clients that can mishandle content, affect service quality, or misreport P2P transfers. CDNs need a mechanism to reliably account for client activity. This is where Reliable Client Accounting (RCA) comes in.

RCA looks at two types of attacks: misbehaving client software (e.g., deviations from correct protocol, collusion) and suspicious user behavior (e.g., repeatedly downloading content to drive up demand). In RCA, clients maintain a tamper-evident log of their network activity. These logs are based on those in PeerReview, but are able to take advantage of the CDN infrastructure. The logs are periodically uploaded to the infrastructure and verified, and anomalous clients are quarantined.

The overhead of RCA is primarily dominated by signature verification, which is on the order of the number of communicating client pairs. This overhead is lower than that of PeerReview, which is on the order of the number of messages. To verify client activity, plausibility checking verifies whether the log is consistent with a valid execution of software. To detect collusion and suspicious user behavior, RCA uses statistical methods. Prior work blacklists potentially misbehaving clients; RCA quarantines them, allowing the client to download but not upload. This enables the use of aggressive anomaly detectors.

The authors studied RCA in the context of Akamai NetSession, which is software used to distribute large files in CDNs. This software is bundled with a customer-specific installer. The authors attacked NetSession by having a client report fake downloads. The network overhead of RCA (as well as the CPU overhead) was less than .5% of the actual content downloaded; log storage accounted for around 100 KB/day. To handle log uploading and processing requires about 35 machines, compared to the 10 used now. RCA was able to catch all of the attacks the authors inserted into NetSession.

This talk generated much discussion. Vyas Sekar (Intel Labs) asked why Akamai was worried about these types of attacks if they control the software. Paarijaat clarified that since the clients are untrusted, they could modify client software and cause significant accounting errors. Srinivas Narayana (Princeton) asked about incentives for carrying out these attacks. A client might want to carry out click fraud against a particular customer, or undermine the CDN's reputation. Colin Dixon (IBM) pointed out that when RCA classifies a client as being malicious, the client is then served directly from the infrastructure. Clients might find this desirable (as the infrastructure usually provides better service), so what is the encouragement to not be malicious? Paarijaat pointed out that clients can opt out of P2P service if they so choose, and that one could be stricter and blacklist clients who were definitely malicious. Tudor Dumitraş (Symantec Research) asked whether RCA could help with an attack where a cli-

ent downloaded a file he wasn't supposed to have access to. These types of attacks should primarily be handled by the CDN via access control lists. Could RCA be used by clients to verify billing statements from the CDN? In theory, this is possible, as RCA is providing a mechanism to ensure honesty in reporting. Emin Gün Sirer asked about scalability, since RCA scales with O(n^2) rather than O(n*S), where n is the number of clients and S is the size of the content (admittedly very large). Paarijaat said that they had designed the system with scalability in mind.

Network Robustness

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

Header Space Analysis: Static Checking for Networks

Peyman Kazemian, Stanford University; George Varghese, UCSD and Yahoo! Research; Nick McKeown, Stanford University

Peyman Kazemian presented Header Space Analysis, a mathematical foundation for answering questions about a network's configuration. In this approach, packets are interpreted as points in a multi-dimensional binary space. That is, a packet with header of length L is represented by a point in an L-dimensional space, where each dimension can have the value zero or one, or be a wildcard; this space contains all possible packets. Networking equipment, such as switches and routers, is modeled as transfer functions from the two pairs: (packet/point, input port) to (packet/point, output port).

Armed with algebraic operators, Header Space Analysis can be used to answer questions such as: can host A talk to host B? does the network contain traffic loops? or is a network slice X fully isolated from slice Y? For example, to check for loops in a network, an all-wildcard packet is injected at every switch port, and the transfer functions from attached networking equipment are applied. If the transformed packet eventually returns to the starting port, and within the same header space, then a loop has been detected.

Header Space Analysis can be performed using a Python library, Hassel, available at https://bitbucket.org/peymank/hassel-public/. Hassel includes a Cisco IOS configuration parser which generates the packet transfer functions, and implements checks for reachability, loop detection, and slice isolation. Kazemian used Hassel to detect loops in Stanford's backbone network in less than ten minutes of runtime on a single laptop.

After the presentation, Kazemian was asked about modeling middleboxes that operate on a sequence of packets. He replied that it is difficult to do so currently because they require state, but such equipment can be over-approximated with a function modeling the largest header transformation it could make. Srinivas Narayana asked about the runtime complexity. Kazemian replied that some tests had complexity growing with the square of the number of VLANs in the network. Arjun Guha asked how the results are presented to the user. Hassel currently prints a textual representation of packet headers that match the query.

A NICE Way to Test OpenFlow Applications

Marco Canini, Daniele Venzano, Peter Perešíni, and Dejan Kostić, EPFL; Jennifer Rexford, Princeton University

Software Defined Networks (SDNs), such as those made possible by OpenFlow, allow the network to be managed by configuration programs running on general-purpose computers. These programs are often logically centralized, yet manage an inherently distributed system. This mismatch is a source of potential bugs, which can have serious consequences in production networks.

Marco Canini presented NICE (No bugs In Controller Execution), a tool which can uncover bugs in OpenFlow controllers written in Python for the NOX platform. NICE takes as input the unmodified controller program, a network topology, and a set of correctness properties (such as no forwarding loops or packet black holes). It then models the controller running on the network and uses model checking to identify packet traces that cause the controller to violate one or more of the properties.

However, the space of all possible packet traces is extremely large. To make NICE scalable, symbolic execution is applied to the controller's event handlers to determine equivalent classes of packets, and domain-specific knowledge is used to minimize the number of packet orderings that must be checked. The authors ran NICE on three previously published OpenFlow controllers, which identified eleven bugs. NICE is publicly available at https://code.google.com/p/nice-of/.

Ranveer Chandra requested references to the tested controller applications, which can be found in the paper. Ratul Mahajan asked why the controller needed to be symbolically executed during each state of the model checking. The reason is that some event handlers' execution depends on the state of the network; making this state symbolic as well would create a scalability challenge. Sambit Das wondered about the specificity of this work to OpenFlow. Canini replied that they make some OpenFlow-based assumptions about how the controller can interact with the switches to make NICE scalable.

Toward Predictable Performance in Software Packet-Processing Platforms

Mihai Dobrescu and Katerina Argyraki, EPFL, Switzerland; Sylvia Ratnasamy, UC Berkeley

Packet-processing platforms such as RouteBricks (SOSP 2009) and PacketShader (SIGCOMM 2010), which run on general-purpose hardware, offer flexibility unavailable in specialized network equipment. However, when multiple packet-processing applications, such as IP forwarding and traffic monitoring, are run together on a software router, the performance can degrade in unpredictable ways. These performance drops are due to contention for shared resources such as CPU caches and memory controllers.

Mihai Dobrescu presented the results of a benchmark study in which five different packet-processing applications ran in combinations inside a single contention domain. The results revealed that contention for the L3 cache was the dominant factor in the performance drop, as most of the profiled applications had few memory accesses.

The authors developed a synthetic application which allowed them to vary the number of L3 cache references each second. When the synthetic application contended for resources with each packet-processing application, the performance drops were within 3% of the corresponding drops caused by realistic applications. Therefore, this simple synthetic application can be used to predict offline the performance impact on an existing packet-processing application when it is run alongside a second one.

Dobrescu was asked by several attendees whether the results depend on the workload used for the benchmark. He responded that the results did, and that network operators generally provision for the worst case of a specific, assumed workload. Rishi Kapoor asked about other potential bottlenecks, such as the NIC, and was referred to analysis performed in the RouteBricks paper.

Privacy

Summarized by Will Scott (wrs@cs.washington.edu)

Detecting and Defending Against Third-Party Tracking on the Web

Franziska Roesner, Tadayoshi Kohno, and David Wetherall, University of Washington

Franzi Roesner presented her work looking at how companies track Web site visitors. The basic threat is that online trackers today can create profiles that capture a large fraction of user activity. Franzi started by laying out a classification system that captured the evolution of tracking Web tracking. Beginning with the use and outsourcing of analytics code, which only tracks activity on a single site, the taxonomy then looks at the different forms of cross-site tracking present on the Web today. In particular, it differentiates standard "vanilla" tracking from mechanisms that create profiles even when third-party cookies are disabled, those that don't rely on the client-side state at all but are included by other trackers, and social widgets that can associate Web browsing with additional profile information even when third-party cookies are blocked.

Franzi then presented a set of measurements showing that 91% of the top 500 domains include tracking code, and simulations of browsing from search logs showing that individual trackers could capture up to two-thirds of visited pages. These measurements also showed that the average domain included between four and five trackers, with DoubleClick being the most popular cross-site tracker. Also notable was that Facebook and Google social widgets were both present on approximately 30% of the top 500 domains.

Franzi then talked about potential ways to defend against Web tracking. She showed that blocking third-party cookies can dramatically reduce anonymous tracking, but doesn't impede social tracking. Additionally, Firefox's method of blocking third-party cookies is undesirable, because it breaks the functionality of embedded social widgets. ShareMeNot (http://sharemenot.cs.washington.edu) is a new alternative for Firefox and Chrome. In Firefox, it removes cookies from social widget requests until they are clicked, and in Chrome it replaces these buttons with local versions until clicked. Franzi showed that this browser extension is largely effective at preventing tracking by social widgets, but cautioned that the method was based on a blacklist and couldn't handle complex widgets such as Facebook comments.

The majority of questions after the talk focused on what additional profiling can't be detected by the client. Vyas Sekar from Intel Labs asked how the discovered trackers were grouped into administrative domain, and both Aaron Gember from University of Wisconsin and John Dunagan from Amazon asked about back-end correlation and side channels. Franzi responded that there likely was sharing of data and ownership between the trackers, and agreed that data handling would largely need to be tackled by legislative policy. The point of this work was to give users control before a longer-erm policy solution is found. Saikat Guha from MSRI asked why some sites would embed 43 trackers, and Franzi explained that those cases were largely due to ad networks, where a range of third parties could get pulled in.

Towards Statistical Queries Over Distributed Private User Data

Ruichuan Chen, Alexey Reznichenko, and Paul Francis, Max Planck Institute for Software Systems (MPI-SWS); Johannes Gehrke, Cornell University

Ruichuan presented Practical Distributed Differential Privacy (PDDP), a system allowing providers to analyze users, while allowing users to maintain exclusive control of their information. He began by explaining why previous work leaves something to be desired: previous attempts either can't scale, can't handle churn, or can't handle untrusted users. He then introduced PDDP, explained that the two key insights were requiring clients to only respond with binary answers and the addition of unknown noise at the proxy. By limiting the space of values and having the clients and proxy collaborate to create additional blind noise, answers remain confidential and malicious clients are not able to over-influence the results.

Ruichuan then went through an example of PDDP processing a query. In the PDDP system, analysts will send the honest but curious proxy SQL queries and bucketing specifications. The proxy will then select clients and forward the query. Clients will execute the query and, for each bucket, return a 1 or 0 encrypted with the analyst's key, using the Goldwasser-Micali system. The proxy next aggregates the results and then shuffles and adds blind noise to each bucket before returning the results to the analyst. The analyst is able to decrypt the binary values and tally the final, noisy result. The system is currently deployed as a Firefox plugin, allowing querying of online browsing activity. While performance is often a concern, PDDP's encryption is fast enough to respond efficiently. Ruichuan noted that they had observed tens of thousands of encryptions per second on desktop browsers, and more than 800 encryptions per second on smartphones. The system is also capable of widespread queries, with a query against one million clients taking a total of less than 30 CPU minutes and a total bandwidth and storage usage of 1.2 gigabytes.

The two points from the talk eliciting questions were the proxy design and the relationship between bucketing and noise. Jacob Lorch from Microsoft Research asked who was envisioned in control of the proxy, and Michael Freedman from Princeton asked if "honest but curious" was chosen because designing for malicious proxies would be too expensive. Ruichuan clarified that proxies could be controlled by a privacy advocate or a company with an external auditor and that the paper does cover design for a malicious proxy. Matvey Arye from Princeton asked if the differential noise could be reduced, due to the uncertainty introduced by bucketing, and another questioner asked who chose the number of buckets and their boundaries. Ruichuan responded that the analyst chooses the bucketing, and because of that control the differential noise needed to be orthogonal from the bucketing.

Koi: A Location-Privacy Platform for Smartphone Apps

Saikat Guha, Mudit Jain, and Venkata N. Padmanabhan, Microsoft Research India

Saikat Guha presented the design for Koi, a system providing a useful and secure interface to location data. He began by explaining that the reason so many mobile applications transmit location data is because the current abstraction of latitude and longitude presented by the operating system isn't useful. Instead, he proposed that applications should express their intent to the operating system and receive event notifications when locations they are interested in are reached. When building the system, they thought of six ways applications interact with location: advertising, content tagging, search, recommendations, social networking, and navigation, and they believe the Koi model will work with all of them.

Saikat then explained how the Koi model provides location privacy. In particular, in order to provide privacy but still answer personalized queries, Koi unlinks information, so while the cloud gets individual queries and locations, it doesn't know how they are linked together. To do this, the cloud system is split into two components: a matcher and a combiner. The matcher does most of the processing, while the combiner keeps information unlinked and can be provided by the user or a trusted organization. In addition, a model checker, ProVerif, has been used to formally prove that the protocol provides privacy guarantees. The specific proofs show that users can't be linked to their attributes, user attributes can't be linked together, and users with matching attributes can't be linked.

In addition, Saikat showed the flow of how a query takes place in the Koi system. Before the query is made, entities and users maintain their status by sending the matcher their identifier, and attributes (like category or location) encrypted first with the matcher's key, and then with the combiner's. The matcher forwards encrypted data to the combiner and gets the unencrypted data back, so it knows all of the information, while the combiner knows the links but not the actual data. A user can then ask the matcher for matching entities, without the cloud being able to link the match to the user. The benefit of this approach is that the matcher works on plaintext attributes (a user might query for entities within five blocks of a GPS position) and the phone can automatically update the location for all applications.

Michael Nowlan from Yale asked if companies would be willing to give up the ability to track location. Saikat agreed that many companies do want the data, but that the OS abstraction would allow significantly more transparency for the user about what applications are doing. Bryan Ford (Yale) asked what would happen if there was a malicious cloud component, or if two cloud components colluded. Saikat answered that the privacy guarantees only work if there isn't collusion, but the proofs of correctness provide some guarantee against a single malicious party. Marcelo Martins from Brown asked how this system would work for users with limited data plans. Saikat said that Koi reduces energy costs, because the OS can amortize requests from multiple applications.

Security and Availability

Summarized by Will Scott (wrs@cs.washington.edu)

Aiding the Detection of Fake Accounts in Large Scale Social Online Services

Qiang Cao, Duke University; Michael Sirivianos, Telefonica Research; Xiaowei Yang, Duke University; Tiago Pregueiro, Tuenti, Telefonica Digital

Qiang Cao presented Sybil Rank, a system for detecting fake accounts in online social networks. Facebook believes 5–6% of accounts are fake, and these fake accounts are hard to detect automatically. Tuenti, the Spanish social network where this work was done, hand reviews 12,000 suspicious accounts every day. Only a small percentage of reviewed accounts are actually fake, so there is a need for a system that is both fast and effective at detecting these fake accounts in large social networks. Sybil Rank uses the landing probabilities of short random walks from known non-Sybil users to rank nodes. The intuition between these walks is that walks from real users are unlikely to enter Sybil regions.

Qiang then explained the details of how the ranking is computed. In order to handle multiple communities within large social networks, initial trust-seeds are picked from different communities. This process is performed by detecting communities and then hand-labeling a set of seed nodes in those communities as real or fake. Trust is then pushed along social links, although only a small number of iterations are performed, in order to produce the same effect as if nodes individually made short random walks.

Qiang concluded the talk with evidence that the method is effective. The algorithm was tested against the Stanford large network data set and remained effective even with a high number of attack edges. It is also deployed on the Tuenti network. In the Tuenti context they found several different structures of Sybil communities, and, more importantly, over 90% of the nodes that the algorithm ranks as least trusted are Sybil accounts.

Two questioners asked about the distribution of Sybils in the network, wondering if some would appear as a crust around the edge of the graph, or as disjoint nodes rather than as communities. Qiang responded that graph schemes tend to use attack edges to detect Sybils, and this approach would also likely miss Sybils which are well integrated with real users compared to other Sybils. Srinivas Narayana from Princeton asked how humans would classify nodes as fake. Qiang explained that they use lots of factors, including correlating photos with age and address, checking the posts an account has sent, and the IP addresses used for login. Another questioner asked whether the same mechanism would work in a peer-to-peer context, and Qiang answered that while this solution is meant for social networks, where there is a global view of the network, some of the techniques could also apply in the distributed context.

Don't Lose Sleep Over Availability: The GreenUp Decentralized Wakeup Service

Siddhartha Sen, Princeton University; Jacob R. Lorch, Richard Hughes, Carlos Garcia Jurado Suarez, and Brian Zill, Microsoft Research; Weverton Cordeiro, Universidade Federal do Rio Grande do Sul; Jitendra Padhye, Microsoft Research

Siddhartha presented the GreenUp service, which keeps desktop computers in an enterprise available, even when most of them are asleep. He began by laying out the problem space that GreenUp explores: desktop computers in companies use a lot of power and many are left on at night. There's a tension between the energy savings that accompanies putting machines to sleep, and the desire to have at-will connections or perform upgrades on machines. He then explained the motivation for GreenUp: requiring dedicated servers to manage machine state was expensive, and transitioning machines to VMs would require disruptive changes to the desktop environment.

Siddhartha then described the design of the service. In order to maintain accurate state of which machines are active and which are not, all active machines are set to randomly and repeatedly probe other machines on the subnet. By carefully choosing the number of probes to send, there is a high probability that all machines will get probed, and an awake machine will notice a state change soon after a computer goes to sleep. A resolution protocol resolves conflicts if multiple machines notice at the same time, so that only one awake machine will become the manager of the asleep machine. GreenUp takes care of several important points to make this work: machines use subnet broadcasts to distribute their IP and MAC addresses, and listening ports to other machines

so that they can become managers. Enough machines need to stay awake for the service to work effectively, and if a manager falls asleep another needs to take over for both it, and all the machines it was managing.

Siddhartha then covered the evaluation of the system. GreenUp was deployed at Microsoft on 1100 machines and was reliably able to wake up machines 99% of the time, with most errors caused by Wake-on-LAN issues. Of wake-ups, 87% took less than nine seconds, and 85% occurred before the incoming connection was closed (13% of connections gave up in under three seconds, and were likely port scanners or other automated systems). In their deployment, GreenUp never consumed more than 7% of CPU resources. They simulated larger management workloads and found that a manager could handle up to one hundred machines without noticeable CPU impact. Finally, they found that with GreenUp, machines were asleep an average of 31% of the time in their environment, but this was due to an IT-mandated sleep policy. In time, they believe GreenUp's availability guarantees will make more users comfortable, allowing their machines to sleep for increased savings.

After the talk, Bryan Ford from Yale asked if they needed to have a server for each subnet, rather than using VLANs to span larger areas. Siddhartha responded that in Microsoft's environment, the hundreds of separate subnets made this impractical, and they didn't want to send broadcast messages to large segments of the enterprise. With follow-up prompting from David Anderson at Carnegie Mellon, he added that in an ideal world an alternative solution to this problem would be for employee desktops to run in VMs that are migrated efficiently to awake servers when inactive. A second focus of questions was on the applicability of this method. Syed Amin from Yale asked about security issues, and Aaron Gember from the University of Wisconsin asked if it would work on home networks. Siddhartha responded that the focus had been on the enterprise domain and that the design isn't meant for an environment with malicious devices or only a small number of machines.

Data Center Networking

Summarized by Katrina LaCurts (katrina@csail.mit.edu)

Jellyfish: Networking Data Centers Randomly

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

Chi-Yao Hong presented Jellyfish, a new topology for data center networks. Old-school datacenter networks are constrained to tree topologies, which make it easier for spanning-tree protocols to operate. Today, however, we can build topologies with much more freedom. Data center topologies have two goals: high throughput and incremental expansion. The second goal is of particular interest in industry. Topologies with structure, e.g., hypercubes and fat trees, constrain this type of expansion, since they require a particular number of switches.

Jellyfish's solution is to forget about structure. Each switch uses some ports to connect to the servers, and the other ports to form a random graph with the other switches. To add nodes, it breaks an existing link and connects it to the new switch. This same procedure works for graph construction as well as incremental expansion. As a result, Jellyfish is cheaper and provides better incremental expansion than existing work. It also achieves higher throughput than fat-tree topologies, especially when there are many servers.

Jellyfish achieves high throughput by making the average path length shorter. When using all capacity, the problem of increasing throughput is equivalent to decreasing mean path length. In a fat-tree topology, most of the edges are used for redundancy rather than to decrease path length. In fact, Jellyfish is within 9% of the throughput for the best known degree-diameter graphs (those graphs represent the theoretically optimal throughput). These types of graphs are difficult to find and are not as expandable.

Jellyfish also addresses the problem of routing/congestion control and cabling. The authors found that using k-shortest paths for routing worked well combined with either TCP or Multipath TCP. For cabling, Jellyfish topologies actually have fewer switch-to-switch cables than fat-tree topologies. It even retains its gains when constrained to only using the same number of long cables as in a fat-tree topology. Long cables are of interest here because they're quite expensive.

In response to a question from Rohan Gandhi (Purdue), Chi-Yao noted that Jellyfish is quite robust, because if a switch fails, the topology is still a random graph. Siddhartha Sen (Princeton) had two concerns: one about the effect on routing table state (since we can no longer aggregate prefixes), and another about how Jellyfish affects multipath protocols. Chi-Yao pointed out that for the routing tables they can use dictionary lookup, but was unsure about the effect on state or on multipath protocols.

Ben Zhao (UCSB) asked what happens when the network is full, i.e., you've added all of the links and filled up all the slots, and have a full clique rather than random paths. Chi-Yao responded that most data centers aren't large enough for that to happen, but that if your budget allows for a full mesh, then that's actually great.

The last question came from Charlie Hu (Purdue). Since Jellyfish gives up hierarchy, how easy is it to make allocations? The authors are exploring this problem now and have found that you can impose a structured graph on top of the random graph with only a small loss of path length.

OSA: An Optical Switching Architecture for Data Center Networks with Unprecedented Flexibility

Kai Chen, Northwestern University; Ankit Singla, UIUC; Atul Singh, Kishore Ramachandran, Lei Xu, and Yueping Zhang, NEC Labs America, Inc.; Xitao Wen and Yan Chen, Northwestern University

Kai Chen presented OSA, an all-optical data center architecture. Today's data centers provide the infrastructure for many "big data" applications. Conventional three-tiered data centers can be problematic, causing communication bottlenecks due to oversubscription. Fat-tree and B-cube architectures provide high bandwidth but have high wiring complexity, use lots of power, and cost a lot. Hybrid (electrical + optical) efforts reduce complexity, power, and cost, but provide insufficient bandwidth.

The insight into the OSA work is that data center traffic exhibits regionality and some stability. By using optical link technology, OSA can dynamically change its top-of-rack (ToR) topology and link capacity to adapt to changing traffic matrices. OSA takes advantage of two technologies in particular: MEMS, switches that allow any input port to be connected to any output port (providing the flexible topology), and WSS, switches that allow for flexible link capacity.

As a result, OSA can achieve any k-regular topology with flexible link capacity among the ToRs. Currently, OSA is a container-sized data center, with a logically centralized control plane that can optimize the network to better serve the traffic. OSA can use systems such as Hedera to estimate traffic demand for the purpose of assigning links and wavelengths. In the prototype implementation, the bisection bandwidth is very close to the theoretical bandwidth. In simulation, OSA is almost non-blocking. Additionally, OSA costs less and uses less power and wiring than fat trees, and has better performance than hybrid architectures (and typically better cost/power/wiring as well). One caveat, however, is that OSA is not intended for all-to-all, non-stable traffic.

In response to a question from Srinivas Narayana (Princeton), Kai noted that they observed traffic stability in production networks on the order of seconds to minutes, which is appropriate, since OSA only takes ~100 ms to react to a change in traffic.

Less Is More: Trading a Little Bandwidth for Ultra-Low Latency in the Data Center

Mohammad Alizadeh, Stanford University; Abdul Kabbani, Google; Tom Edsall, Cisco Systems; Balaji Prabhakar, Stanford University; Amin Vahdat, Google and U.C. San Diego; Masato Yasuda, NEC Corporation, Japan

Mohammad Alizadeh presented HULL, an architecture for achieving both low latency and high bandwidth in data centers. Many data centers run low-latency applications today (e.g., high-frequency trading, RAMClouds). They would like predictable low-latency delivery of individual packets. With large-scale Web applications, where data is typically stored on machines separate from the ones containing the application logic, we have to communicate across the network to get data to an application, limiting the data access rate. Even the latency in the tail is important: the application won't respond if one operation is taking a long time. The goal of HULL is to reduce delay—in particular, queueing delay—to zero.

HULL first uses DCTCP (Data Center TCP) to allow the network to react to congestion in proportion to the extent of the congestion. DCTCP is helpful here, but does not get latency down as far as we'd like. The main idea in HULL is to use "phantom" queues, which signal congestion before queueing occurs. This approach sacrifices some bandwidth, but lowers latency. Since data center traffic often exhibits both mice and elephant flows, HULL can steal some bandwidth from the larger flows without severely impacting their performance. Unfortunately, because TCP traffic is bursty, to get down to near zero latency we must sacrifice a lot of bandwidth. To deal with this, HULL implements hardware pacers, which dynamically figure out at what rate to pace. These pacers bring the 99th-percentile latency down to almost zero.

HULL implements hardware pacing at the network edge, DCTCP in the racks, and phantom queues at the switches. In simulation and on a 10-server testbed, the authors saw a 93% decrease in latency for roughly a 17% increase in completion time.

John Dunagan (Amazon) asked how DCTCP interacts with workloads that don't typically run over TCP, as well as how much latency endhost software adds. Mohammad responded that, in principle, you could apply HULL to non-TCP flows, and that we could likely optimize out the overheads of software. Mark Handley (UCL) asked how long HULL takes to yield resources to a new flow. Mohammad showed that it's only a constant factor increase on top of TCP. Jeongkeun Lee (HP Labs) asked why HULL uses hardware pacing rather than software pacing, and Mohammad pointed out that software pacing requires disabling things such as LSO. In response to Costin Raiciu's (University Politehnica Bucha-

rest) question, "Why not just use two priority queues?" Mohammad noted that one usually assigns an application, not a specific flow, a priority. It is much harder to use priority queues at a flow level and to dynamically assign priorities.

Big Data (2)

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

PACMan: Coordinated Memory Caching for Parallel Jobs

Ganesh Ananthanarayanan, Ali Ghodsi, and Andrew Wang, University of California, Berkeley; Dhruba Borthakur, Facebook; Srikanth Kandula, Microsoft Research; Scott Shenker and Ion Stoica, University of California, Berkeley

Modern data parallel clusters are increasingly capable of storing large amounts of data in RAM. For analytic processing jobs, such as those executed by Hadoop or Dryad, caching file inputs can result in a significant performance boost. However, because of synchronization steps, or barriers, which occur after some execution stages, a single uncached input can eliminate the end-to-end latency improvement from caching other inputs in the same stage.

Ganesh Ananthanarayanan presented PACMan (Parallel All-or-nothing Cache Manager), a service for data parallel clusters that coordinates file caching across the cluster. PACMan uses a global view of the cluster to focus evictions on incompletely cached stages, stages for which not all compute nodes are able to cache the inputs. This approach ensures that stages with inputs fully cached across the cluster do not lose caching's benefits.

PACMan can use two metrics to determine which file to evict. The first is user-centric, minimizing jobs' completion times, and the second is system-centric, maximizing the utilization of the cluster. The service contains two cache eviction policies: LIFE, which is optimized for the first metric, and LFU-F, which is optimized for the second. Details of these policies can be found in the paper. PACMan has been evaluated using traces of Facebook's and Bing's production workloads, and reduced average job completion time by 53% and 51%, respectively.

Because caching reduces disk accesses, Ananthanarayanan was asked about its importance relative to other resources, such as CPU. He replied that Map-style phases, which are generally disk-intensive, account for 60–70% of job completion times. Sriram Rao asked about the effect of adding more jobs to the cluster. Performance would continue to improve, but only to a limit. John Ousterhout asked about the ability to cache data parallel jobs over the long term. Ananthanarayanan replied that as long as job sizes continue to follow a power law–like distribution, there will be small jobs which can be cached effectively.

Reoptimizing Data Parallel Computing

Sameer Agarwal, University of California, Berkeley; Srikanth Kandula, Microsoft Research; Nico Bruno and Ming-Chuan Wu, Microsoft Bing; Ion Stoica, University of California, Berkeley; Jingren Zhou, Microsoft Bing

The performance of data parallel jobs running in Hadoop or Dryad clusters is heavily influenced by task parallelism, input data partitioning, and implementation and sequence of operators such as Map, Reduce, or Join. These settings are described by the job's execution plan, which can be determined manually, or automatically using frameworks such as Hive, Pig, or SCOPE. Unfortunately, configuration choices made at a job's start may not be optimal later.

Sameer Agarwal presented RoPE, a Re-optimizer for Parallel Executions, which uses statistics from recurring and similar jobs, as well as previous stages of the same job, to improve execution plans. RoPE captures both data properties, such as cardinality and common values, and code properties such as CPU consumption and peak memory utilization. The key challenge is to measure these properties with both high accuracy and low overhead.

RoPE provides statistics modules that can be integrated into a job's execution, as they are distributable, maintain sublinear state, and are restricted to a single pass over the data. The results are then collected and composed as inputs to an existing query plan optimizer. Using RoPE's improved inputs in Bing's production clusters yielded a 2x improvement in job latency at the 75th percentile while using 1.5x fewer resources, at a cost of only 2–5% increased overhead.

Agarwal was asked about applying RoPE's techniques to other frameworks such as DryadLINQ. He replied that the feasibility depends upon such frameworks' own plan optimizers, as the statistics gathered by RoPE are quite general. Matvey Arye asked about potential benefits from users annotating their operators with statistics. The presenter found that while SCOPE supports such annotations, in practice users do not have the necessary knowledge to make use of them. Rohan Gandhi wondered about the consistency of the collected statistics. The authors considered additional statistics, but restricted RoPE to those which remain stable.

Optimizing Data Shuffling in Data-Parallel Computation by Understanding User-Defined Functions

Jiaxing Zhang and Hucheng Zhou, Microsoft Research Asia; Rishan Chen, Microsoft Research Asia and Peking University; Xuepeng Fan, Microsoft Research Asia and Huazhong University of Science and Technology; Zhenyu Guo and Haoxiang Lin, Microsoft Research Asia; Jack Y. Li, Microsoft Research Asia and Georgia Institute of Technology; Wei Lin and Jingren Zhou, Microsoft Bing; Lidong Zhou, Microsoft Research Asia

Today's data parallel frameworks such as Hadoop and Dryad require shuffle steps during which output data from one

stage is re-sorted and re-partitioned to become the input data for a second. Because shuffle steps can require all-to-all communication between processing nodes, they are both time-consuming and resource-intensive. Zhenyu Guo presented statistics showing that shuffle operations cause more than half of all cross-pod communication in a production system at Bing.

To eliminate unnecessary shuffle steps between stages, the authors developed an optimization framework called Sudo, which has three components. First, Sudo tracks the partitioning properties of the data as it is being processed: for example, is the data hash-partitioned? is it range-partitioned? is it sorted within partitions? Second, Sudo determines which functional properties are required by the user code processing the data; perhaps the code expects the input data to be monotonically increasing. To determine these functional properties, Sudo performs program analysis on user-defined functions and can incorporate user-provided annotations as well.

Finally, Sudo determines the least-costly operation that will transform a stage's output data such that it will have the properties required for input to the subsequent stage. Ideally, data would not need any shuffling between stages. In other cases, Sudo may find that a local sort within a partition is necessary and sufficient. The paper contains further details about Sudo's reasoning and the properties present in various data-partitioning strategies. Experimental results show that Sudo can save up to 47% of disk and network I/O costs incurred by naive shuffling in Bing's production systems.

After the presentation, Mike Freedman asked how the data-partitioning and functional properties Sudo tracks were chosen. Guo replied that the data-partitioning properties were the ones provided by the underlying system, and the functional properties were the ones used in production.

Poster Session

Summarized by Karthik Nagaraj (knagara@cs.purdue.edu)

SNMiner: A Rapid Evaluator of Anomaly Detection Accuracy in Sensor Networks

Giovani Rimon Abuaitah and Bin Wang, Wright State University

Deployments of sensor nodes consist of many nodes constantly recording environmental measurements and later transmitting them to a central place for analysis. It is possible that some of these sensors are either returning faulty readings or have been compromised, leading to wrong readings. This work describes the design of a framework for identifying anomalous behavior in sensor networks, using statistical and data mining techniques. Historical data from nodes can help predict if a node has suddenly started exhibiting faulty behavior. This framework allows introduction of faulty readings within the network through different fault models, identifies faulty readings, and removes those from the collected data. The authors have developed a framework to collect all the sensor data so that different analysis techniques can be easily performed on the data and the detection accuracy of such techniques can be re-evaluated over time.

BlinkDB: Queries with Bounded Errors and Bounded Response Times on Very Large Data

Sameer Agarwal and Aurojit Panda, UC Berkeley; Barzan Mozafari and Samuel Madden, MIT CSAIL; Ion Stoica, UC Berkeley

With the recent explosion of data on the Web, many user decisions and activities can be improved, but they depend on the ability to process this data quickly. With such data quickly aggregating to terabytes and even petabytes, even the use of large compute clusters cannot easily produce query results within a few seconds. An interesting tradeoff is explored in this work between processing large volumes of data and producing results within deadlines, by tweaking the precision of the results. BlinkDB aims to produce slightly less accurate results within a time bound by running the queries across samples of the data. Using historical query information to gather usage patterns to create intelligent multi-dimensional stratified samples, BlinkDB achieves high accuracy with low storage overhead. It also caters to different query time bounds by using samples of different sizes. Some initial experiments show that BlinkDB can process 17 TB of data on 100 nodes in under 2 seconds, with 2–10% error.

PhoneLab: A Large-Scale Participatory Smartphone Testbed

Rishi Baldawa, Micheal Benedict, M. Fatih Bulut, Geoffrey Challen, Murat Demirbas, Jay Inamdar, Taeyeon Ki, Steven Y. Ko, Tevfik Kosar, Lokesh Mandvekar, Anandatirtha Sathyaraja, Chunming Qiao, and Sean Zawicki, University at Buffalo, The State University of New York

PhoneLab is the initiative, equivalent to PlanetLab, to enable testing of smartphone applications and protocols. Smartphones are ubiquitous—every other mobile phone user carries a smartphone and uses it to access the Internet, take pictures, play music, control other devices, etc. However, research and development of new technologies for smartphones is still limited to small testbeds with a handful of devices, limiting extensive testing of ideas under realistic workloads. PhoneLab aims to solve this problem by deploying Android smartphones to participating users who would use them normally. Researchers and developers, on the other hand, can provide new applications, tweak underlying operating system functionality and protocols, and collect

usage data or patterns. The use of these phones by real people provides ample opportunity to analyze new techniques and identify new research problems with smartphones. Two of the toughest challenges that hamper the setup of PhoneLab are funding for the devices themselves, and charting out guidelines for the use of the phones by participants.

Precise Anomaly Detection in Network Flows

Sriharsha Gangam, Purdue University; Puneet Sharma, HP Labs; Sonia Fahmy, Purdue University

Actively monitoring network flows is an important and challenging task for network operators, allowing them to verify the current state of the network and also identify faulty or malicious behaviors. With faster networks it is difficult to collect and analyze large volume of flow records at current line rates (of the order of millions of flows per minute). Existing monitoring solutions resort to sampling and sketching, which are lossy in nature and provide only approximate answers. This work advocates the use of co-located compute and storage units (blades) to the switches to provide a distributed passive monitoring infrastructure. With such in-network processing units, aggregation queries and network wide anomalies can be adaptively mined to reduce the communication overhead and improve the anomaly detection accuracy. Since anomalies can span multiple devices and a centralized approach is too expensive, this work provides techniques for anomaly detection in a decentralized manner. Briefly, the detection starts by aggregating coarse-grained flow summaries and iteratively converges to the solution by successively gathering finer-grained flow summaries on demand.

Scaling WiFi Performance for Large Audiences (poster and demo)

Jeongki Min, Arpit Gupta, and Injong Rhee, North Carolina State University

Arpit Gupta described how WiFi scales poorly at access points with many subscribers, mainly because traffic characteristics in such setups produce an asymmetry in sharing the physical medium. Analysis of traffic patterns in large audiences indicates that 76% of the traffic is incoming TCP data, from large request responses. In essence, access points need higher priority to transmit on the wireless medium, whenever download data from the wired network starts to backlog. This work explores the technique for dynamically boosting the access point priority as a logarithmic function of queue size. They further show that prioritization improves good-put of transmission and also reduces re-transmissions.

Non-Linear Compression

Michael F. Nowlan and Bryan Ford, Yale University

Compression is an integral part of many modern systems. However, the popular multicore and parallelization movement is at odds with traditional linear compression schemes, which impose sequential inter-block dependencies on applications. Unfortunately, serial processing significantly impacts parallelism on newer multi-core architectures and also forces network packets to be decompressed in order. Non-Linear Compression fills this gap by supporting both traditional linear compression similar to gzip and "modern" parallel and adaptive compression. NLC gives applications full control on where to form inter-block dependencies, structuring relationships as a parent-child hierarchy. This allows compression and decompression in parallel and independent of other blocks, so applications can make progress with only a subset of the data. The evolving history from all children can later be reconciled intelligently to avoid losing much compression ratio. This work pairs well with distributed storage as well as recent work on uTCP that allows developers to use unordered reliable transmission of packets, which can then be possibly compressed.

Performance Isolation and Fairness for Multi-Tenant Cloud Storage

David Shue and Michael J. Freedman, Princeton University; Anees Shaikh, IBM T.J. Watson Research Center

Cloud storage services are very popular among cloud users, because of high availability and reliability of such storage mediums. However, as the number of users increases, resource fairness gains serious concerns, as users would like to get performance equal to their expense. Currently, cloud providers strive to provide fairness between various virtual machines (VMs) executing on a single host but fail to provide guarantees on other shared infrastructures such as the storage service, network, etc. This work aims to provide weighted fair shares and performance isolation between users sharing a storage layer, such as Amazon S3, through a novel combination of four techniques. Using partition placement, weighting, replica selection, and fair queuing, storage requests are made more streamlined and amenable to performance fairness. Even in situations where client requests are unpredictable and may be bursty, these techniques keep fairness under control.

This same group also had a demo of Serval (presented as a paper), where they demonstrated playing a video on a smartphone without losing frames, while disabling different network types (WiFi, 4G).

Composable Reliability for Asynchronous Systems

Sunghwan Yoo and Charles Killian, Purdue University; Terence Kelly, HP Labs; Hyoun Kyu Cho, University of Michigan; Steven Plite, Purdue University

Distributed systems are traditionally implemented to handle failures using timeouts—a technique that treats really slow nodes as having failed, and restarting them. With the advent of more managed environments, such as data centers and campus setups, observed network delays are real delays, and nodes are quickly restarted if they fail. Therefore, if the state of the process were not lost, applications could be recovered in a cleaner and quicker manner. This work outlines the design and implementation of MaceKen, a transparent distributed systems programming framework that allows application developers to avoid dealing with process crashes and proceed on the assumption of slow nodes. MaceKen achieves this through carefully crafted techniques to persist the state of the application process and network messages sent or received, so that restarted nodes can resume execution from the last correct checkpoint. MaceKen marries the persistent protocol Ken with the popular Mace programming toolkit, so that applications can be easily developed in Mace and easily leverage the benefits of Ken.

New Architectures and Platforms

Summarized by Kevin Ngo (ngoke@onid.oregonstate.edu)

XIA: Efficient Support for Evolvable Internetworking

Dongsu Han, Carnegie Mellon University; Ashok Anand, University of Wisconsin—Madison; Fahad Dogar, Boyan Li, and Hyeontaek Lim, Carnegie Mellon University; Michel Machado, Boston University; Arvind Mukundan, Carnegie Mellon University; Wenfei Wu and Aditya Akella, University of Wisconsin—Madison; David G. Andersen, Carnegie Mellon University; John W. Byers, Boston University; Srinivasan Seshan and Peter Steenkiste, Carnegie Mellon University

Hyeontaek Lim introduced a new network architecture, eXpressive Internet Architecture (XIA). XIA provides a unique foundation for diverse communication styles. Lim first pointed out flaws in today's Internet architecture, specifically IP. IP is described as "the narrow waist of the Internet," an outdated model for host-centric communication that is still being used despite increasing demand for service and content-oriented communication. XIA seeks to support heterogeneous communication types (e.g., service, content, mobility, cloud) on a single Internet architecture rather than focusing on one communication type, all while being extensible for future communication types.

Lim introduces two primary design pillars: "principal types" and "fallbacks." Principal types allow the defining of ad hoc communication models in XIA. The current Internet architecture has one principal type, the IP address. In XIA, a principal specifies a type and a type-specific identifier (e.g., service type and hash of service's public key). Each principal has type-specific semantics to allow XIA routers to logically process it. XIA routers do not have to support all principal types, although they must support host-based communication. However, there must be a way of supporting legacy routers that would not understand these new principal types.

Fallbacks are mechanisms that allows incremental deployment of new communication types and backwards-compatibility. With direct acyclic graph (DAG) addressing, intent is encoded into packets. If necessary, the architecture can fall back to different routing choices if the intent of the packet is not understood by a router. DAG addresses, routes with indicated possible fallbacks, are encoded into packet headers (the encoding method can be found in the paper). Fallbacks can even be nested. With a click-based implementation on commodity hardware with 351,000 table entries based on a Route Views snapshot, they found that with XIA routers, forwarding provides throughput comparable to that of high-speed IP routers with minimal slowdown for small packets with three fallbacks. The prototype of XIA is hosted on GitHub at https://github.com/XIA-Project/xia-core, which supports LAN, XIA-over-IP, and GENI.

An attendee wasn't clear whether the source address in XIA was the legacy source address or a DAG. Lim replied that there is a symmetry between the destination address and the source address, and that the source address can be a service to allow flexible communication between any pair of entities in the network. Another attendee noted that the original Internet was intended to be evolvable and extensible but over time that idea was lost, for several reasons, and asked how, if deployed, XIA would avoid the noted pitfalls that TCP and IP fell into. Lim answered that XIA had a security feature, intrinsic security, by which the host identifier can be used to identify the source of a packet, which can be used to amend some problems on the Internet. The attendee followed up with the reservation that people would end up adopting only a select few principal types and asked if the architecture could prevent that. Lim responded that they had not addressed that at this point. David Oran remarked that content-networking people preferred no source addresses at all and asked how Lim would answer that. Lim supported source addresses for security reasons and noted that they do not hinder flexibility in content networks. Oran also asked why they did not use partial orders rather than DAGs. Lim did not find much difference between them and found no side-effect of using DAGs over partial orders.

Design and Implementation of a Consolidated Middlebox Architecture

Vyas Sekar, Intel Labs; Norbert Egi, Huawei; Sylvia Ratnasamy, UC Berkeley; Michael K. Reiter, UNC Chapel Hill; Guangyu Shi, Huawei

Vyas Sekar talked about a new approach for building and managing middlebox deployments. As networks evolve, new applications and devices appear, along with evolving threats to exploit these applications. Then, as companies move these applications to be Internet-enabled, there is a question about policy compliance. Looking at an enterprise network, Sekar's group found that rather than changing routers and switchers, the current standard was to add specialized appliances, or middleboxes. Middleboxes are often shipped with narrow interfaces that can be frustrating to manage. As network requirements grow, the only option for network operators is to buy more of these middleboxes, which is not only costly but limits extensibility and flexibility. This work seeks to consolidate the building of individual middleboxes and the managing of a network of middleboxes.

CoMb is top-down design with a logically centralized middlebox controller. On the management layer, the goal was to balance the load across the network and to assign processing responsibilities across different middleboxes. However, reuse and policy dependencies made this difficult. The need for explicitly capturing these dependencies is eliminated if all applications relating to a given session run on the same node. Sekar introduced hyperapps, which take the logical union of all the actions that need to be run on a given packet, so that common actions are not duplicated. With hyperapps in place, the management problem becomes a simple and near-optimal program. Sekar went on to explain the design in depth, which consists of a policy enforcement layer that logically routes packets.

Sekar then showed that consolidation has a low overhead for existing applications, takes little time to index a network, has five times the throughput of VM-based consolidation, sharply reduces maximum load many-fold, and cuts provisioning costs by about half. Addressing isolation, he noted that CoMb currently relies on process-level isolation and leverages user-space for high-performance networking. He also addressed the problem of changing vendor business models and said that consolidating middleboxes is already happening in the form of virtual appliances and that with the benefits it's likely that someone will adopt it.

An attendee had reservations about using CoMb in the presence of topology constraints. Sekar noted that the paper addresses those cases. Another attendee from Cambridge asked about hidden policies in middleboxes. Sekar responded that the policies are not coming from the middlebox vendor but, rather, from the network operator running an optimization. The attendee then plugged a project from Cambridge, Mirage, which addresses an isolation-reuse problem, which pleased Sekar. John Dunagan asked why they did not run everything on every server and simply choose the number of servers, which is the alternative he saw becoming adopted. Sekar replied that in some sense that is the kind of resource management they are doing by removing the physical coupling. Dunagan followed up and remarked that if every server provided uniform functionality, rather than policy specification, an easier approach to sizing would be to route one person to the traffic to measure CPU utilization. Sekar responded that policy specification was more for deciding what kinds of traffic needed what kinds of processing than for resource management. Aaron Gembar asked whether there was a way to get expected resource consumption. Sekar affirmed and said they had thoughts on more fine-grained fairness issues. Michael Sirivianos referenced an earlier point and asked why different middleboxes had different peak times. Sekar said different boxes have different types of functionality, which causes non-overlapping traffic. Lastly, Dongsu Han wondered whether they accounted for vendor-specific hardware optimizations and, if so, whether that created issues. Sekar responded that if an application has an affinity for a particular sort of hardware, it will be run on that hardware.

An Operating System for the Home

Colin Dixon, IBM Research; Ratul Mahajan, Sharad Agarwal, A.J. Brush, Bongshin Lee, Stefan Saroiu, and Paramvir Bahl, Microsoft Research

Colin Dixon presented HomeOS, a home operating system that eases extensibility and management by providing a PC abstraction for home technology. The motivation was to bring to life the long-envisioned smarthome, where devices are able to come together to cater to wants and needs such as climate control and energy monitoring. They talked to homeowners of existing smarthomes and found that although the smarthomes were convenient, they had poor extensibility and were frustrating to manage. It was difficult to add new devices and configure each individual device with access control. Dixon presents two existing abstractions for technology within the home: a network of devices with interoperability protocols, and fixed appliances with fixed tasks. A network of devices is easily extensible but difficult to manage, whereas appliances are easier to manage, due to their closed nature, but extensibility is difficult. To achieve both extensibility and management, Dixon urged us to view the home as a computer where adding devices consists of plugging in a peripheral and adding tasks consists of installing an application.

Applications run on top of HomeOS and communicate with the devices using drivers. This way, users can interact with HomeOS's centralized high-level interface rather than interacting with individual devices. Dixon presented three challenges with HomeOS in the field. First, non-technical users suddenly have to become network managers. Second, heterogeneity can make it difficult for developers to build applications. Lastly, it can be difficult for HomeOS to keep up with new classes of devices and applications that arrive frequently. HomeOS respectively addresses these challenges with management primitives that align with the user's mental models, with protocol-independent abstract APIs for devices, and with a kernel agnostic toward device functionalities. These goals are mapped onto a stack with layers for device connectivity, device functionality, management, and applications. This stack respectively covers the heterogeneity of topology, devices, control, and tasks.

Dixon then did a live demo of HomeOS on a laptop. He installed an application from a list of compatible applications from the HomeStore. Through a wizard, two cameras were slowly added and configured, and the video from the cameras were shown on-screen. Dixon's group evaluated HomeOS based on usability and extensibility, using field experiences and controlled experiments. They found that the field experiences were positive; users could manage their HomeOS deployments, and developers found the programming abstractions and layering to be natural. Still, users found it hard to diagnose faults, interoperability protocols could be fragile, and not all device features were exposed over the network. In the controlled experiments, they found that most non-technical users were able to complete management tasks without training and found that developers were able to very quickly write one of two realistic applications.

Where would HomeOS physically be located within the home? HomeOS would run on a single Windows PC, where it would have access to the network to communicate with devices. What about conflicting devices? There was little to do to stop two devices conflicting. Would there be configuration management for devices? There would not be.

Cloud Performance

Summarized by Advait Abhay Dixit (dixit0@purdue.edu)

Structured Comparative Analysis of Systems Logs to Diagnose Performance Problems

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

Karthik Nagaraj started with a performance comparison of two implementations of BitTorrent: Azureus and Transmission. Transmission was about 20% slower than Azureus in all executions. For this performance problem, logs did not contain any error messages. Due to non-determinism and concurrency in the system, a line-by-line comparison of logs from Azureus and Transmission is not possible. To address such performance problems, the authors developed Distalyzer, a tool that does a comparative analysis of logs to find the root cause of a performance problem.

Distalyzer automatically analyzes large volumes of logs using machine-learning techniques. Distalyzer first extracts features (such as states and events) from the logs that represent characteristics of the execution. Then it uses predictive modeling to extract those features that diverge between the two sets of logs. Divergence in features may not impact performance, so Distalyzer identifies dependencies between system components and performance using dependency graphs. It then adds weights to the edges by exploiting the underlying statistical model and weighs the node sizes based on divergence between the two sets of logs. After pruning edges with low weights, the root-cause graph is output to the application developer for analysis.

The presenter briefly described case studies on three mature distributed systems, which uncovered six performance problems. They uncovered a constant delay (sleep) in Transmission, which, although it was written in C, was slower than Azureus (Vuze), which was written in Java, because of this added delay.

Distalyzer is available at: http://www.macesystems.org/distalyzer/.

Srinivas Narayana (Princeton University) asked for guidelines for developers on how to record state and event transitions in logs. Karthik responded that the amount of logging to add is a known hard problem. Adding too many logs increases runtime overheads, but adding too few logs does not tell you enough about the system. There is no good guideline on this, but Distalyzer allows developers to use their existing high-performance logging infrastructure. Later, developers can use a parser to translate their logs for analysis by Distalyzer, and that worked well for them. Fitz Nowlan (Yale University) asked if they have a graph for Transmission and Azureus after they fixed the performance problem. Karthik displayed the graph. Transmission and Azureus had identical performance.

Orchestrating the Deployment of Computations in the Cloud with Conductor

Alexander Wieder, Pramod Bhatotia, Ansley Post, and Rodrigo Rodrigues, Max Planck Institute for Software Systems (MPI-SWS)

Alexander Wieder listed the choices available to a user who wants to run a MapReduce application using Amazon's cloud service. The user has choices for storage (Amazon's S3 storage, local storage in EC2 nodes) and computation (EC2,

other private infrastructure). Choosing a strategy for deploying cloud applications becomes challenging because of the variety of services and providers available to the user, where each one has different performance characteristics, pricing models, and interfaces.

Conductor is a system that chooses a strategy for deployment of applications in the cloud. It optimizes resource allocation with a set of user goals (e.g., a deadline or reducing monetary costs) and resources available for use. Conductor optimizes by generating a formal linear programming–based execution model and feeding it to a LP-solver along with the user goals. Conductor also provides a resource abstraction layer which sits between the MapReduce framework and the actual resources. The abstraction layer provides abstractions for storage and computation. This allows the cloud application to use different computation and storage services simultaneously. Once the application is deployed, Conductor monitors the execution and changes the deployment to adapt to performance variations or changes in price.

Using EC2 and S3, Wieder described an experiment that showed how Conductor finds the optimal execution plan for a given job completion deadline. The experiment also illustrated how Conductor adapts the execution (by allocating more EC2 nodes) when the observed performance of EC2 nodes is lower than the estimated performance.

Dongsu Han (Carnegie Mellon University) asked about the Amazon policy of having different pricing for different regions and whether Conductor handles this case. Alex said that they can handle this case. This adds to the diversity of products available. In the model, each instance type can be considered as a different service in each region, with different prices but the same performance characteristics. Dongsu Han then asked if computation costs generally dominate over bandwidth costs, and Alex said that it is very hard to come up with realistic cases of what people usually do with a cloud. He couldn't really say whether jobs are computationally bound or I/O bound.

Transport

Summarized by Advait Abhay Dixit (dixit0@purdue.edu)

Fitting Square Pegs Through Round Pipes: Unordered Delivery Wire-Compatible with TCP and TLS

Michael F. Nowlan, Yale University; Nabin Tiwari and Janardhan Iyengar, Franklin and Marshall College; Syed Obaid Amin and Bryan Ford, Yale University

Janardhan Iyengar began by saying that currently, applications have a choice of just two transport protocols: UDP and TCP. While UDP offers no guarantees, TCP guarantees in-order delivery at the cost of performance. This is an all-or-nothing choice. The authors' goal is to break up the functionality of the transport layer to pieces that can be composed by the application. The architecture, called Minion, uses TCP, UDP, and TLS as a substrate to ensure compatibility with middleboxes. The kernel component of Minion, called uTCP for unordered TCP, stores untransmitted data in priority queues on the sender side. The application can determine the order in which data is transmitted by specifying the priority in the write system call. On the receive-side, uTCP delivers unordered byte fragments to the application but guarantees that all bytes will be eventually delivered. Minion's uCOBS protocol provides a datagram delivery service atop uTCP. Janardhan spoke briefly about uTLS, which provides encrypted out-of-order delivery using TLS wire format. He illustrated the application-level benefits of Minion through an experiment in which uTCP provides low latency for high priority data.

Charles Killian (Purdue University) asked whether previously sent messages are canceled. Janardhan said that, due to middleboxes, the hard constraint is keeping the format on the wire unchanged. They can, however, suppress messages that are in TCP's queue.

How Hard Can It Be? Designing and Implementing a Deployable Multipath TCP

Costin Raiciu, Universitatea Politehnica Bucuresti; Christoph Paasch and Sebastien Barre, Université Catholique de Louvain; Alan Ford; Michio Honda, Keio University; Fabien Duchene and Olivier Bonaventure, Université Catholique de Louvain; Mark Handley, University College London

▶ *Awarded Community Award!*

Costin Raiciu shared his experience in designing and implementing a deployable multipath TCP. Mobile devices with multiple interfaces that have different characteristics and coverage are becoming common. In data centers, there are multiple paths between end hosts. Content providers have multi-homed servers for robustness, but while networks are becoming multipath, TCP has remained single-path.

The authors' goal was to develop a deployable multipath TCP that works with unmodified applications and without changing network hardware and configurations. The authors used TCP options to set up the MPTCP connection and send MPTCP protocol-specific acknowledgments. This design ensures that packets sent over the wire do not differ from other TCP packets in any way other than the TCP options that they carry. Thus, MPTCP works with middleboxes that randomize sequence numbers or modify IP addresses and port numbers. However, if middleboxes remove TCP options, MPTCP reverts back to TCP. Costin also described

an optimization to improve the performance of MPTCP over multiple interfaces that have a large difference in round-trip times (such as MPTCP over Ethernet, WiFi, and 3G). Finally, Costin demonstrated a streaming video over MPTCP running on a host with Ethernet, WiFi, and 3G. Code is available at http://mptcp.info.ucl.ac.be/.

Vyas Sekhar (Intel) said that MPTCP makes sense in data centers, but they do not have middleboxes. So do we need it in the real world, or can we just have multiple connections? And can we do things differently for data centers? Costin replied that using multiple TCP connections instead of MPTCP will still require changing end-host applications. Even then, MPTCP behaves better during handovers. For data centers, we still need separate sequence number spaces for each connection. MPTCP has been shown to perform better than bonding multiple ports as well. Janardhan Iyengar (Franklin and Marshall College) wondered if it is possible to avoid deadlock by reserving some receive window space for the data acknowledgment. Costin said that middleboxes might remember advertised receive windows and hold or drop data that exceeds the receive window, so data acknowledgments cannot be sent if there is a receive window full of unacknowledged bytes. However, if some part of the receive window is reserved for data acknowledgments, this is possible. Bryan Ford (Yale University) asked what happens when a path and middleboxes change during the lifetime of a subflow—for example, when a network link fails. Costin replied that if things change during the lifetime of a subflow, that subflow is killed if there are other subflows available. If there are no other subflows, MPTCP falls back to TCP. This is a one-way switch, meaning MPTCP never tries to switch back to open multiple subflows again.

The TCP Outcast Problem: Exposing Unfairness in Data Center Networks

Pawan Prakash, Advait Dixit, Y. Charlie Hu, and Ramana Kompella, Purdue University

Pawan Prakash described an unfairness problem that may arise when switches employ drop-tail queues and rely on TCP to fairly allocate bandwidth to flows at a bottleneck link. Pawan described a scenario where flows from two input ports at a switch drain to the same output port. One of the input ports carries a large number of TCP flows, while the other has only a few flows. In such a scenario, the flows from the port with fewer flows do not receive their fair share of bandwidth at the output port. The happens because of a temporal phenomenon which the authors call "port blackout," wherein many consecutive packets from one of the input ports are dropped due to synchronization of packet arrival times. Either of the two input ports may experience port blackout. However, the smaller set of flows suffers more as a result of port blackout and hence receives a smaller share of the bandwidth at the output port. The authors observed the problem and validated their hypothesis on a testbed of 16 hosts connected in a fat-tree topology.

Ashok Anand (Bell Labs India) asked if they experimented with DCTCP where buffer utilization is low. Pawan answered that they do not have ECN support in their testbed and hence could not experiment with DCTCP, but they feel that the problem will not happen with DCTCP. Dongsu Han (Carnegie Mellon University) asked why consecutive packet drops matter. Pawan replied that flows in the large bundle respond differently to consecutive packet drops from flows in the small bundle. When the large bundle of flows sees consecutive packet drops, there is a smaller reduction in throughput. However, when the flows in the small bundle see the same number of consecutive packet drops, they suffer timeouts that result in a severe reduction in throughput. Mark Handley (University College London) pointed out that the problem is phase lock between packets and asked whether randomizing packet sizes to break phase lock would help. Pawan replied that it is something they hadn't tried, but they plan to.

5th USENIX Workshop on Large-Scale Exploits and Emergent Threats: Botnets, Spyware, Worms, New Emerging Threats, and More (LEET '12)

San Jose, CA
April 24, 2012

Opening Remarks

Summarized by Rik Farrow (rik@usenix.org)

Engin Kirda (Northeastern University) opened the workshop by saying that they had decided on combining speakers from industry with academic researchers this year, with six industry and seven academic position or research papers accepted. He asserted that he wanted LEET to retain its workshop format and that people should ask questions throughout each presentation unless the speaker asked that questions be held until the end.

Engin then listed types of attacks and pointed out that social networks provide a huge new set of targets. Fabian Monrose (UNC), obviously rising to the invitation to ask questions, challenged Engin's list, saying that there is nothing new there. Engin responded that he is not trying to predict the future, except to say that malware will become more sophisticated. He also pointed out that with virtualization and data

moving into the cloud, there will be more loss of control over data, and it will be more difficult to determine the correctness of computations.

New Challenges

Summarized by Rik Farrow (rik@usenix.org)

Challenges in Network Application Identification

Alok Tongaonkar, Ram Keralapura, and Antonio Nucci, Narus, Inc.

Alok Tongaonkar explained that Narus is a subsidiary of Boeing that focuses on traffic classification by categorizing traffic. Narus has its own research staff, as well as being a source of funding for external research into network attacks. They fund 20 research fellows.

Alok described the focus of their talk as traffic classification. In the past, most tools used the server port number as a hint for determining application protocol. But this works poorly now, because applications like BitTorrent can use any port. Another type of classification uses header features to identify the protocol—for example, similarities in packet length. Most commercial tools today use deep packet inspection (DPI), using strings like EHLO or HTTP to guess the protocol. But now we have thousands of apps (think games running within Facebook or a smartphone, which are tunneled within HTTP). And they need a way to learn about applications automatically, as there are thousands of new applications.

Rik Farrow pointed out that it's easy to hide one application within another, like steganography. Vern Paxson (UCB) suggested that Alok provide a threat model. Alok said that they want to be able to identify applications even when encrypted. It is true that applications hide within other applications and that they consider this a real problem. Niels Provos said, just to be clear, they are not trying to identify traffic where people are trying to hide their intentions. Alok said that network operators need to identify known traffic before they can even begin to consider uncovering covert channels.

Fabian Monrose asked what percentage of traffic, from Narus's point of view, they have no way of analyzing. Alok said they used tstat to analyze cellular traffic, and it was only able to identify 70% of the traffic, and most of that was HTTP, which is not good enough, as HTTP is used to tunnel many other apps. Vern Paxson wondered if this isn't a losing battle. How much of the traffic is over SSL? Alok said he was just worried about cleartext traffic, with hopes that their packet header analysis may work for encrypted traffic too. Most cellular data traffic is not encrypted, except for authentication.

Alok then said that one of the big challenges was how to generate signatures for 0-day applications. Paul Ferguson thought it was inappropriate that they use 0-day to identify new applications, and Alok agreed. They just want to identify new apps in traffic. But given flow sets, how do they automatically learn to match traffic for new applications. There are lots of challenges here.

Vern Paxson asked whether crowdsourcing, such as is used to create Wireshark recognizers, might work well. Alok wondered if you should trust signatures developed by volunteers for a commercial system. Vern thought that network operators would appreciate something over nothing. Also, classification is different from intrusion detection, so there are really two different types of applications. Alok agreed that there really are two types. Fabian said he liked Vern's suggestion about crowdsourcing. They looked at the traffic on two campuses and found that 13% of streams used ports that indicated encryption was in use, and another chunk was object streams, which would be opaque to analysis. Alok said that analyzing object streams is an interesting problem, but their header analysis might work there as well.

Sherlock Holmes and the Case of the Advanced Persistent Threat

Ari Juels and Ting-Fang Yen, RSA Laboratories

Ting-Fang Yen started by listing several well-publicized APT attacks: Shady RAT, Operation Aurora Google in China 2009, and against RSA in 2011 to gain access to SecureID. What makes these attacks different is that the attackers have access to lots of tools and can create new attacks; they remain in the victim's network; they are as well resourced as nation-states; and they are strongly motivated. She then pointed out the differences between traditional attackers and APT attackers, where APT attackers have different goals (espionage), objectives, and targets (individual users or systems). APT attackers are not bound by a playbook, although well-known attacks do follow a pattern that starts with spearphishing.

Ting-Fang then used four Sherlock Holmes stories to suggest other techniques that could be used: surround a victim in a more general attack that hides the attack specifics; expropriate data using an indirect method rather than simply uploading the data; conceal unauthorized communications within a commonplace object; and use other methods to bypass a physical perimeter (e.g., a robot through ductwork or digital photoframes). For example, a large botnet might include a bot within a target organization, and an attacker could simply pay to install probe software. Ting-Feng ended with three

points that might help in detecting attacks: behavior profiling, defensive deception, and information sharing.

Engin asked about the role of education in stopping APT. Ting-Feng responded that, given enough time, you can get anybody to open an email. Manuel Egele (UCSB) said that banks would make good targets and wondered why we never hear about them. Ting-Feng said there are a lot of attacks we won't hear about. Jason Brett of UCSB asked whether they had seen instances through RSA of the use of bots. Ting-Feng said she had not, but does not know all that is going on in RSA.

Fabian Monrose wondered what RSA does since they had been attacked, based on her last three points. Ting-Feng said she focuses on log analyses, and RSA collects lots of logs but hasn't done anything with them; she wants to focus on working with these logs. Fabian then asked what the research community might do to help. Ting-Feng said that logs are messy, missing fields, and hard to process. Also, there are many false alarms. Niels Provos wondered whether the logs they had collected would have been sufficient to detect the attacks in the past. She responded that they are not sufficient in themselves, as they don't have total visibility. Niels asked what else they might need. Ting-Feng said the Windows security logs would be useful. Tudor Dumitraş of Symantec Research asked about the role of 0-days in detecting attacks. Ting-Feng said that since they were impossible to detect (by definition), they really weren't of much use in detection.

Let's Parse to Prevent Pwnage

Mike Samuel and Ulfar Erlingsson, Google

Ulfar Erlingsson said that Mike had done most of the work, and since this was a position paper, they hadn't finished it until recently. Ulfar provided an opaque link (via tiny-url.com) and mentioned that you might want to be cautious about following such links. Instead of opening the PDF file pointed to by the link, you might allow Google to process the PDF and present it to you as HTML. In the same way, you could take advantage of a third party to process photos instead of trusting your own system to correctly parse a photo format that comes from an unknown source.

Vulnerabilities can be caused by inconsistencies in processing by different software applications. Ulfar named this data confusion. For example, processing JPEG and PNG files has resulted in vulnerabilities in just about everything since 1998, including a bug in libpng in 2012. He suggested performing lowering, using protocol buffers. As another example, antivirus doesn't work because of data confusion. AV companies cannot write correct parsers for all file formats. But we can prevent data confusion, and we have technical work appearing in the paper. Ulfar said that rather than have parsers for all content (d) and contexts (c) (O(c*d)), you just need O(c+d).

Niels Provos pointed out that there are thousands of protocols, and Ulfar responded that the problem is linear, and that they actually did this work for a Web browser. Then Ulfar showed a demo of a thin client, with all protocol parsing done in a sandbox, and the results displayed in the thin client. Vern Paxson said that work had been done before on thin clients, and Ulfar agreed that the client doesn't have much functionality. Vern wondered if the greed for rich interaction will kill this idea, and Ulfar agreed that this approach by itself is not enough, but parsing using annotated grammars is required. Google already needs to understand protocols and has been doing this. Vern argued that sooner or later they will have to parse a blob of bytes, and Ulfar responded that they already plan on only allowing normalized blobs, such as a single bitmap format.

Ulfar said they have developed one grammar for HTML, CSS, URI, and JSON, plus grammars for languages such as C, Java, and Python. They use lower encoders, sanitization (stripping out scripts), and contextual templating, doing the processing and computation securely and separately from the display. Niels again said that this sounds like a lot of work. Ulfar answered that Mike Samuels has been doing this for years. Fabian Monrose wondered whether they will ever reach a point where everything is sound. Ulfar responded that you don't get to a point where you resolve everything, but you do resolve a lot. Fabian then asked about getting programmers to buy in, and Ulfar said that they want this to happen automatically, within libraries, so that the programmers don't have to do anything.

Emerging Threats

Summarized by Engin Kirda (ek@ccs.neu.edu)

Observations on Emerging Threats

Paul Ferguson, Trend Micro, Inc.

Paul Ferguson from Trend Micro talked about the emerging threats they have been seeing. Trend Micro's Threat Research group is specially tasked with looking forward on the threat landscape. He talked about how the toolkits the attackers are using are getting more sophisticated, and he suggested that mobile threats are probably the next big thing. After the presentation, Tudor Dumitraş from Symantec commented that the problem is that kits make it easier for cybercriminals, because they create work for the defenders. Fabian

Monrose from UNC asked how we know that nation-states are carrying out targeted attacks. Do we assume things? Do we have a pointer for this? Paul's answer was that there is a lot of ad hoc evidence out there and although there are no scientific measurements, there is general awareness and knowledge in the community that such attacks are indeed taking place.

W32.Duqu: The Precursor to the Next Stuxnet

Eric Chien and Liam OMurchu, Symantec; Nicolas Falliere

Nicolas Falliere from Symantec gave a talk on the Duqu analysis they recently conducted in the company. After Nicolas mentioned that Duqu was based on the codebase of Stuxnet, Vern Paxon from UC Berkeley asked what it means that the code is based on Stuxnet. Nicolas answered that there is evidence that the attackers had full access to the source code of Stuxnet. Vern followed up by asking if the Duqu was targeting a specific country. Nicolas said that unlike Stuxnet, Duqu was not targeting a specific nation. One of the questions asked was whether there was authentication in the Duqu command and control mechanism. Nicolas said that the creators of Duqu had not built in any authentication mechanisms.

Botnets and DDoS Threats

Summarized by Manuel Egele (maeg@cs.ucsb.edu)

So You Want to Take Over a Botnet...

David Dittrich, University of Washington

Dave Dittrich discussed what should be taken into consideration when a botnet is to be taken over or down. First, he addressed liability questions that arise when people demand that compromised command and control (C&C) infrastructure should be used to remove malware from infected machines. Dave said that this is a slippery slope and the liabilities are not clear in case that harm is done.

While discussing possible legal approaches, which could include declaring the possession of malicious software illegal, Engin Kirda asked for clarification of the expression "malicious software possession," which David clarified as attack tools and not malware samples per se.

Dave highlighted nomenclature as a big problem. Additionally, reported numbers are often inflated and unverifiable, as the counting methodologies are commonly not disclosed. This includes missing information on timing, or error rates. Dave described the takedown of the Kelihos.B botnet, which turned out not to be exhaustive enough, as the attackers were able to establish the modified Kelhios.C botnet very quickly after the takedown effort. This led Dave to ponder whether the five most recent takedowns were all successes or just one big failure.

Dave then said that the size estimates of botnets vary hugely, to which Vern Paxson responded that the frequency at which DHCP leases are reissued is dependent on the individual ISPs, and thus it is hard to get the size right. However, Dave pointed out that sometimes bots use a unique ID for each installation and the count can be quite reliable. Engin Kirda wondered why a displayed graph did not indicate novel infections, and Dave replied that the graph shows information gathered after the C&C was sinkholed, and thus no new infections took place.

Dave did mention that relying on the legal process effectively describes the involved ethics as a by-product. That includes the identification of all stakeholders, a detailed analysis of harm and benefits, the likelihood of success, the intentions for the requested action, and automatic external review by the court. This description raised the question of whether the courts are technically competent enough to base their decisions on solid grounds. Dave agreed that it is challenging to put such technical matters into lay terms and that it can be hard to find courts and lawyers that are technically savvy enough.

Dave was then asked what would be done to alleviate the problems that arise from the bad naming choices made in the industry. Apparently, there are efforts on the way in Europe and throughout the industry to come up with a taxonomy to define the necessary terms.

Classification of UDP Traffic for DDoS Detection

Alexandru G. Bardas, Loai Zomlot, Sathya Chandran Sundaramurthy, and Xinming Ou, Kansas State University; S. Raj Rajagopalan, HP Labs; Marc R. Eisenbarth, HP TippingPoint

Alex Bardas explained why one has to consider UDP-based distributed denial-of-service attacks as well as those carried out over TCP. Readily available software packets and free tools such as the Low Orbit Ion Cannon (LOIC) rely on UDP to perform their flooding attacks. Alex then described their detection mechanism that implements a counting approach for packets grouped by source IP address. Although many attacks use the same payload for all packets, this is not a necessity. The fact that UDP is stateless makes detection challenging.

The underlying assumption is that during an attack there are almost solely incoming packets but no outgoing packets that would suggest legitimate communication. The authors call this observation the Proportional Packet Rate

Assumption for legitimate UDP conversations. This inherent two-way communication results mainly from applications implementing their own acknowledgment mechanisms for the unreliable UDP protocol. Thus, the proposed detection system calculates packet ratios of sent and received packets for each sender at the enterprise level. The proposed system was evaluated on synthetic and production networks such as departmental networks, and non-DNS packet captures from industry partners (e.g., ISPs, universities, and financial institutions).

The authors used traffic generated by the LOIC tool to evaluate their approach. The results indicate that there is no silver bullet to detect all UDP-based DDoS attacks. However, by adapting the thresholds for their methods according to each network independently, the authors were able to achieve good results.

The first post-talk question was about understanding how the involved thresholds are chosen. Alex mentioned that the thresholds are best set by observing the network under consideration for a while (i.e., training) and then choosing the corresponding thresholds. He acknowledged that a continuous reevaluation of these thresholds might be necessary, as the optimal values for thresholds can change over time, depending on how the network is used. The second questioner, from the University of British Columbia, tried to analyze the possible impact of spoofed IP addresses, and whether an attacker could disrupt the communication between two peers by spoofing one of the IP addresses. Alex agreed that their system would take down the connection between these two clients in this case but also mentioned during the talk that most ISPs perform ingress filtering and thus IP spoofing (e.g., from dial up connections) is not that easy to accomplish. Furthermore, Alex clarified that their approach relies on observing all traffic entering the network. That is, asymmetric routes where not all used routes are covered pose a problem to the proposed approach.

Tracking DDoS Attacks: Insights into the Business of Disrupting the Web

Armin Büscher, Websense Security Labs; Thorsten Holz, Ruhr University Bochum

Armin Büscher provided some insights into the business behind DDoS attacks. Like the preceding speaker, he mentioned that readily available tools such as the LOIC allow attackers such as botmasters and hacktivists to attack a variety of targets. The authors extracted the motives of these cyber-crooks by evaluating underground forums and identified blackmail, the disruption of competition and adversaries, and political protest as the main areas of motivation. Armin then went on to present a more detailed analysis of the DirtJumper Crimeware Kit attack tool, which is sold and pirated on different underground forums. In their study, the authors analyzed 274 bots which received 1,968 unique target URLs during the observation period, with the majority of the attacked services being HTTP and MySQL database installations. An analysis of the target URLs indicates that they cover a wide variety of businesses such as shopping, gambling, and adult entertainment sites. The authors were thinking about probing the victims of the observed DDoS attacks to evaluate whether the ongoing attacks were successful. However, the probe requests would of course contribute to the attack, so the authors decided against taking this step.

Niels Provos wondered how the authors could evaluate whether an attack was successful. Armin named etrade.com.au, which was attacked by DirtJumper and as a result was taken offline for four days. This information was publicly disclosed and acknowledged by etrade.com.au around Christmas 2011. Armin mentioned that they are aware of at least two other attacks which were also successful, but he was not at liberty to discuss any details about these incidents. Another person asked whether the motivation to attack banks is blackmail. Armin said they had interviewed banks, and blackmail was indeed the motivation for the attacks. An unanswered follow-up question was whether attackers could expect higher success rates for phishing attacks during a DDoS attack, as the legitimate banking Web site would be unavailable to customers.

Fabian Monrose wondered how the landscape is changing, as he had a flashback to HotBots five years ago when the community discussed similar topics. Armin stated that the Internet and how businesses use the Internet have significantly changed in recent years. Today, we have businesses whose business model relies solely on their Web site. Thus, these companies are prime targets for such attacks.

Mobile Security

RGBDroid: A Novel Response-Based Approach to Android Privilege Escalation Attacks

Yeongung Park, Dankook University; ChoongHyun Lee, Massachusetts Institute of Technology; Chanhee Lee and JiHyeog Lim, Dankook University; Sangchul Han and Minkyu Park, Konkuk University; Seong-Je Cho, Dankook University

No reports are available for this session.

Studying Cyber-Criminals

Summarized by Rik Farrow (rik@usenix.org)

Clustering Potential Phishing Websites Using DeepMD5

Jason Britt, Brad Wardman, Dr. Alan Sprague, and Gary Warner, University of Alabama at Birmingham

Jason Britt gave some background about phishing sites and said that producing a good-looking phishing site takes time. Even though they may look different, they often use the same support files. In their research, they collected MD5s of support files to group phishing sites. Jason then mentioned phishing kits, tools that are sold to help people create phishing sites. Their goal is to go after the bigger players, the people who make or sell phishing kits by looking at the support files found on phishing sites.

They collected data over five months, mostly financial phish sites, but also security and private companies. They checked 265,611 potential sites: 38% were manually confirmed to be phishing sites. If the hash of the top-level page doesn't match a known phish kit page, they collect hashes of support files. Then they use a clustering technique (SLINK) with a minimum similarity threshold of .8 to find sites that are likely to come from the same phishing kit. They did find phishing kit similarities, but they would like to have more assurance that they have really found clusters.

Someone from Paypal asked why there were so many singleton clusters. Jason said that they felt they had enough data, but innocuous files confuse their analysis technique. The same person asked about sites that were unreachable when they did their analyses after the five months was up, and Jason said that they had not looked at that. Would it work better to look at the DOM structure rather than using hashes, which are easy to change? Jason said that their syntactical analysis might solve that problem. Did Jason feel that phishing would be a problem in 10 years? He expects things will change, with more targeted attacks.

Ask WINE: Are We Safer Today? Evaluating Operating System Security through Big Data Analysis

Tudor Dumitraş and Petros Efstathopoulos, Symantec Research Labs

Tudor Dumitraş started with the notion of a worm that could infect all vulnerable machines in 15 minutes. He then pointed out that malware variants are multiplying exponentially, even as OS protection has improved, with DEP and ASLR commonly used in Windows. Tudor presented actual data collected using antivirus telemetry (100 million reports from 6 million hosts over 21 months) and intrusion detection telemetry (65 million reports from 5 million hosts over 31 months). Vern Paxson asked whether this is the actual number of reports, or was this data sampled. Tudor answered that this information is after filtering, not all the data collected. Vern then asked whether this was background noise or something evil. Tudor said that this was malware detection or network packets that triggered IDS.

Tudor wanted to provide some answers to questions such as: do the new OS protections make it less likely that malware will be created for newer platforms? does it help if the platform software is completely updated? are certain platforms less susceptible to attacks? The answers come from the Worldwide Intelligence Network Environment (WINE), a too,l recently developed by Symantec, that is available to researchers in academia. Someone asked how the data is available, and Tudor replied that this is the same data Symantec uses internally, but this is also a filtered sample designed for data-intensive experimentation (SQL, MapReduce, R). WINE can only be accessed on Symantec's premises (in Culver City, CA, and Herdon, VA).

Someone asked about the difference between AV and IDS. Tudor answered that both products are operating on the hosts, and network attacks that trigger the IDS have already bypassed the platforms' firewalls. Tudor then pointed out that these reports indicate that AV or IDS has blocked the attack, so it was a detected, but not successful, attack. The attacks are on various Windows platforms, as that is the most common platform for running Symantec software. Although most viruses are reported variants, they wanted to provide the most specific identification possible, so they present distinct viruses, identified by non-heuristic signatures.

Their first finding is simple: the more hosts running a particular platform, the more distinct virus signatures will be found on that platform. The OS versions with the highest numbers of distinct virus signatures are the ones with the newest security technologies, including DEP/SafeSH, NX bit support, ASLR, User Account Control, etc. He then addressed Mac OS X, where there is a similar trend, but the numbers are three orders of magnitude smaller, so there is not enough data for a statistically significant result. He next looked at how the passage of time affects the number of intrusions or distinct viruses; the number of intrusions seems to increase over time, and the number of viruses does not start from zero even for new releases, but it was not clear that the rate increases over time. Tudor mentioned that finding a virus on a host did not mean that the malware could successfully execute on that platform. In another graph, the introduction of a security technology like DEP, NX bit, and firewall enabled by default

had no detectible effect on the detection of viruses on that platform.

Tudor said that their study has several biases: their data (1) only comes from hosts where the owners actually installed and maintained AV or IDS, (2) is almost entirely from Windows, and (3) does not include servers in their analysis. WINE includes other data sets, such as binary reputation (signatures of all binaries downloaded to hosts around the world), historical URL reputation, samples of spam, and a large collection of malware samples. In conclusion, the production of malware is driven by the number of target platforms deployed, and the introduction of new defenses does not mean a reduction in the number of attacks seen.

Fabian Monrose wondered if there was a correlation between the number of attacks they see and vulnerability disclosures. Tudor said they are doing something very similar, looking at 0-day attacks. One of the things they have observed is that, after disclosure of a vulnerability and 0-day attack, the number of attacks grows a lot.

Threats in Social Networks

Key Challenges in Defending Against Malicious Socialbots

Yazan Boshmaf, Ildar Muslukhov, Konstantin Beznosov, and Matei Ripeanu, University of British Columbia

Adapting Social Spam Infrastructure for Political Censorship

Kurt Thomas and Chris Grier, University of California, Berkeley; Vern Paxson, University of California, Berkeley, and International Computer Science Institute

No reports are available for this session.