;login:
OCTOBER 2013 VOL. 38, NO. 5
ELECTRONIC SUPPLEMENT

Conference Reports

HotCloud '13: 5th USENIX Workshop on Hot Topics in Cloud Computing

San Jose, CA
June 25-26, 2013

Keynote Address

Dystopia as a Service

Adrian Cockcroft, Netflix
Summarized by Rik Farrow (rik@usenix.org)

Adrian stated that they are engineers trying to solve hard problems and to fix things when they break. We all want perfect code on perfect hardware. But perfection takes too long, utopia is slightly out of reach. So they compromise, focusing on time-to-market instead of quality. At Netflix, they need to move fast so they can disrupt their competitors' OODA loop (observe, orient, decide, act).

And they do move fast: code features in days, 20 projects in parallel every week, get hardware in minutes, and incident response in seconds instead of hours. They can discover and stop problems in seconds.

Adrian suggested a list of books for inspiration: *Release It!*, Nygard; *Thinking in Systems*, Meadows; *Antifragile*, Taleb; *Drift into Failure*, Dekker; *Everything Is Obvious*, Watts; *The REST API Design Handbook*, Reese. These books range from economics, philosophy, to software design. Another eye-opening idea was to break things deliberately to see how your total system responds. If you watch the video, you'll find an even longer list of recommended books.

Netflix started out with a traditional architecture, running on mainframes with million dollar Oracle licenses. They still have one mainframe that runs the DVD business, but the streaming business runs Cloud Native, with the only non-cloud parts being the customers' devices and OpenConnect CDN boxes located within ISPs.

Part of making the transition from a traditional architecture to a cloud native one includes a change in how a business is organized. The traditional software business includes three silos: business, developers, and operators. For cloud, developers are their own Ops for the most part, and the few pure-Ops people work with the developers. Adrian listed four transitions from traditional to cloud native. First, management needs to integrate the traditional roles into a single organization, without silos. Second, work with NoSQL, where you figure out what you want first, then create a database that supports this; you do not create the schema first. Third, responsibility moves from Ops to Devs, with continuous delivery, which can be changed or scaled as needed. Creating new things is not the problem, retiring old things is. Finally, Devs provision hardware as they need it.

These are the steps to getting out of the way of innovation, said Adrian. Reducing costs tends to slow down development, and that's a death spiral. At Netflix, there is no process for applying for hardware; you just do it.

Adrian also described tools for antifragile testing; Chaos Monkey to make sure systems are resilient to individual failures; Chaos Gorillas, which shuts down entire zones, used once every three months; and Latency Monkey, which introduces extra latency into the system as well as error return codes.

Adrian described their basic design as a Web frontend with many services hidden behind it. They use Cassandra for data storage, accessed through a RESTful service called the Astyanax Cassandra Client. No application talks directly to Cassandra except Astyanax. Priam manages their clusters, and Netflix has 50 clusters in three regional DCs. All data gets stored within 10 seconds, and they currently are storing two petabytes on S3.

They do use Linux (CentOS or Ubuntu) with optional Apache frontends, memcached, and non-Java apps. Most code is in Java over JVM 6 or 7, with some Groovy, Closure, and Python scripts for maintenance. They wrap DNS with Denominators, which allows you to do everything you can do with DNS. UltraDNS, DynECT DNS, and AWS Route53 are all broken, claimed Adrian. With Denominators, if you lose an entire region, you can just switch everything to the other region.

There was no time for questions, as Adrian went nine minutes into the break.

I/O

Summarized by Zsolt Istvan (zsolt.istvan@inf.ethz.ch)

A Hidden Cost of Virtualization When Scaling Multicore Applications

Xiaoning Ding, New Jersey Institute of Technology; Phillip B. Gibbons and Michael A. Kozuch, Intel Labs Pittsburgh

With multicores being the norm in datacenters, most applications execute multiple threads to take full advantage of the parallel resources. When multithreaded applications are run in virtualized environments, lock exchanges between threads can suffer a significant performance penalty—and, surprisingly, this performance penalty has not been studied so far in detail. In the work presented by Xiaoning Ding, the authors not only explain the root of this problem, but also provide a solution that speeds up lock exchanges between threads.

Because the virtual machine manager's handling of interrupts of the virtual machine is done in software, so-called inter-processor interrupts (IPIs) are costlier than in native environments. The IPI is used by one thread to signal another thread when it has released a specific lock, and, as a consequence, exchanging locks in VMs becomes an expensive operation. Furthermore, VMMs often have to multiplex a large number of virtual cores on a smaller number of physical cores, and this introduces an additional scheduling overhead for virtual cores that are coming back from the idle state. The authors show that the previously mentioned two overheads significantly degrade the performance of applications which synchronize between threads with high frequency (tens of thousands of times per second). By running benchmarks with and without virtualization, they found that some applications can run up to five times slower when run in VMs than on the real machine.

As a solution to this problem, Ding proposed a runtime support design, called Gleaner, that mitigates the multicore virtualization penalty with idleness consolidation and IPI-free wake up. Idleness consolidation works by packing threads onto the same (virtual) core when they operate in an interleaved fashion—this helps reduce the number of virtual core state transitions, and always keeps the active threads on an active core. The idea behind IPI-free wake up is to utilize exceptions to signal other threads, instead of triggering a software interrupt in the VMM. When repeating the same benchmarks as before, the authors could reduce the performance penalty to less than 60% even in the worst case when adding Gleaner to the VM.

Abel Gordon (IBM Research) asked whether upcoming virtualization techniques in Intel CPUs, which make locking possible without IPIs, would solve the problem comparably well. Although this would help, it would not entirely remove the problem, Ding explained, because the performance penalty does not come only from IPIs, but also because the VMM must reschedule virtual cores that were idle. Andrew Sayler (U Colorado) asked whether modifying the VMM directly to reduce the cost of IPIs, instead of providing a user-space solution, would be a viable option. Ding said that the approach they chose solves the problem in a general way, which could, in turn, be applied to any type of VMM.

Priority I/O Scheduling in the Cloud

Filip Blagojević, Cyril Guyot, Qingbo Wang, Timothy Tsai, Robert Mateescu, and Zvonimir Bandić, HGST Research

Filip Blagojević presented a novel priority-based I/O scheduling technique that relies on the ICC-NCQ extension to the native command queueing (NCQ) interface of modern hard drives (http://sourceforge.net/projects/iccncq/). This work is motivated by scenarios in which two types of applications access the hard drive, with different requirements (e.g., a database and a background batch job). In today's systems, even though the operating system attaches priorities to disk accesses of different applications, this information may be lost when arriving at the drive, due to reordering optimizations. As a consequence, low-priority commands can get in the way of high-priority ones.

As a solution, Blagojević et al. implemented a system in which the OS passes the priority of commands to the disk directly. In the hard drive controller, commands are inserted into three different queues, part of the ICC-NCQ extension. For their experiments, the authors modified the Hadoop file system (HFS), but their approach is general and could be applied to other file systems as well: high-priority read requests are intercepted, and a special command is sent to the hard drive. As Blagojević pointed out, their approach needs the software stack to be somewhat modified, and the bypassing of the standard execution path for high-priority read commands means that these operations cannot benefit from caching.

To evaluate their solution, the authors ran the Yahoo Cloud Service Benchmark (YCSB) both on unmodified HFS instances and their custom ICC-NCQ-enabled software stack. The most important result is that when the priorities are correctly passed to the HDD, the average latency is effectively halved for the high-priority reads.

Because the work presented is still in progress, Blagojević mentioned several open questions, such as whether there is a threat of starvation of low-priority commands when operating with high contention on the HDD. Another interesting question was how many priority classes would be useful (or enough) for users to divide the available bandwidth effectively among applications.

Attendees asked first about how the authors' results compare to related work in SOSP '11. Blagojević explained that in contrast with other works which deal with priorities only on the OS level, their approach passes priority information all the way down to the hard drive. As a consequence, it provides more reliable behavior. Someone asked what happens when the upper layer logic aggregates requests to read ahead in bulk; does the increased size of data to be retrieved change the problem? Blagojević replied that, because the solution was mainly devised for workloads typical to clouds (e.g., applications running on HBase), the expected access pattern to disk is mainly small reads.

vPipe: One Pipe to Connect Them All!

Sahan Gamage, Ramana Kompella, and Dongyan Xu, Purdue University

Sahan Gamage said that many applications that are indispensable for providing Web-based services—such as Web servers, Hadoop, streaming servers, backup services, or Web proxy servers—often move data between network and disk directly with no computation. When these applications run in virtualized environments, the I/O overhead increases because every operation that involves the hard drive or the network passes through several layers of execution (virtual machine manager, virtual machine kernel space, and virtual machine user space) only to be sent right back down to the output device passing through the same layers again. Additionally, when multiple virtual machines share the same physical host, applications may suffer from increased I/O latency: if data is available, but the corresponding VM is not currently running, it has to be first rescheduled before the application can use the data.

Gamage proposed a solution that delegates the task of moving data between two I/O devices to the VMM. Instead of bringing data all the way up to the user code and then sending it back down through the same layers, vPipe sends a request to the VMM to perform the data transfer on the behalf of the user code. For this to work, both the guest operating system and the driver domain of the VMM must be extended. The current prototype of vPipe is implemented on the top of the Xen hypervisor, running Linux, and supports file-to-socket data transfer. The support for other combinations, such as file-to-file, is currently in progress. The prototype was tested with a simple application that reads files from disk and sends them to the network, and uses the lighttpd Web server. In both cases, the overhead of pushing data through several layers is removed, and the CPU usage is significantly reduced.

Konrad Miller (KIT) asked whether it would be possible to create an operation that is semantically similar to the sendfile command, but operates at the VMM level. (The sendfile command in the Linux operating system moves data between two I/O devices by delegating the task to the kernel.) Unfortunately, this is not quite possible due to the way network sockets are created and accessed by the VMs, explained Gamage. Another question related to the positive impact a lighter-weight device emulation could have on data movement overhead. Although the vPipe approach provides high performance, it also necessitates changes to the guest OS, which would not be necessary if the emulation costs were much smaller, said the presenter.

Livio Soares (IBM Research) asked whether they anticipate any difficulties with implementing the file-to-file transfer in vPipe. Gamage answered that the scenario when the destination file does not exist yet indeed poses an implementation challenge. Gamage explained that this functionality is currently being implemented and that they will have to modify a file system function to make it possible.

Security, Mobile, and Big Data

Summarized by Varun Prakash (vsprakash@uh.edu)

Jobber: Automating Inter-Tenant Trust in the Cloud

Andy Sayler, Eric Keller, and Dirk Grunwald, University of Colorado, Boulder

A confident and enthusiastic Andy Sayler started with an introduction to some of the questions his research aimed to answer. The safety, reliability, and improved efficiency of datacenters is the focus of the group's research. Some problems, such as the isolation of virtual machines, was projected. To address these issues, they proposed a dynamic network security architecture called Jobber.

Some of the features that are unique to Jobber, such as the idea of Introduction-based Routing, were discussed. A framework was presented along with some of the applications in which the framework could be most useful, such as sensor networks. Andy presented, on similar lines, three different architectures, some of which were futuristic but still served the purpose of providing secure network within a datacenter.

There were four questions.

Towards Secure and Convenient Browsing Data Management in the Cloud

Chuan Yue, University of Colorado, Colorado Springs

Although browsers are some of the most commonly used softwares in our day-to-day life, little has been known about browser vulnerabilities. Chuan Yue started his talk with an introduction to browsers and the sort of browsing data that is usually generated in the case of personal use. The data can be categorized as (1) user data, which includes passwords, bookmarks and autofills; (2) browser data, which includes history and cookies; and (3) settings data, such as language preferences. The realities faced by users in terms of browsing data insecurities and inconvenience problems are at the core of the research and the presentation.

Some of the browsers that provide a server synchronization feature for data and system consistency were tested for performance. To add weight to the claims, each of the most commonly used browsers were tested for vulnerabilities using specific benchmarks designed by the researchers to take most of the cases of data security mentioned earlier into consideration. As a result, an elaborate table of some of the requirements of the solution and how the current state-of-the-art manages to satisfy these policies was given. Also, the importance of system architecture on the vulnerabilities was highlighted. Questions were kept offline due to shortage of time.

Clone2Clone (C2C): Peer-to-Peer Networking of Smartphones on the Cloud

Sokol Kosta, Vasile Claudiu Perta, and Julinda Stefa, Sapienza-Università di Roma; Pan Hui, The Hong Kong

The marriage of cloud computing and mobile computing has led to a wide range of capabilities to the already advanced mobile computing platform. The task of executing computationally and space-extensive application, which was previously a concern for mobile operating systems developers, is now considered an easy-to-tackle problem. The authors presented their work, which includes the building of a virtual peer-to-peer platform using the cloud, where each peer is recognized as a clone of a virtual machine.

Some of the clones that were considered for experimentation include the Xen, QEMU, and Virtual box. An assessment of performance in terms of CPU read I/O and write I/O were performed when these clones were run on Amazon and private clouds.

The product of the research is named CloneDoc, a group of securely collaborating systems for a smartphone platform. The authors looked at another platform for non-mobile computing, called SPORC, for inspiration. The system was tested on a smartphone testbed that consisted of a diverse set of phones. As a result, the authors presented statistics in terms of system responsiveness, networking, and energy consumption. A comparison with SPORC showed CloneDoc's performance was much higher on all types of phones. There were three questions.

Transparent and Flexible Network Management for Big Data Processing in the Cloud

Anupam Das, University of Illinois at Urbana-Champaign; Cristian Lumezanu, Yueping Zhang, Vishal Singh, and Guofei Jiang, NEC Labs; Curtis Yu, University of California, Riverside

Cristian Lumezanu discussed an infrastructure's capabilities for processing data in the cloud. Their proposal, called FlowComb, aims to address some of the shortcomings of some of the infrastructures running commonly used cloud platforms. Their goal was to better predict resource demands. To achieve this, the flow of data within the infrastructure is rerouted to ideal areas to avoid congestion and bottlenecks, a task that is taken up by the decision engine. The authors provided some preliminary results that show an increase in computation performance and a reduction in associated delays. There were three questions.

Reliability

Summarized by Shekhar Gupta (shkhrgpt@gmail.com)

The Case for Limping-Hardware Tolerant Clouds

Thanh Do, University of Wisconsin–Madison; Haryadi S. Gunawi, University of Chicago

Haryadi Gunawi focused his presentation on performance degradation of cloud infrastructure in the case of limping hardware rather than persistent faults. He started with an overview and examples of hardware failure in clouds. He also provided some facts to show the existence of limping hardware in real cloud infrastructure. He used an HDFS example to show how limping hardware can be worse than crashed hardware and how it can lead to cascading effects. He concluded his talk with the questions, "Is crashing better than limping?" and "How can we anticipate limping in algorithm design to avoid limping effects?"

Alysson Bessani (University of Lisbon) asked a question about the applicability of timeout in Zookeeper, as it doesn't rely on timeout. Gunawi agreed with Alysson, but added that there are other problems with Zookeeper. Livio Soares asked a question about the analogy of software limping with hardware limping. Gunawi replied that Hadoop does take care of software limping. Rakesh Bobba suggested using aggressive heartbeat protocols to tackle limping hardware.

Cloud Computing for the Power Grid: From Service Composition to Assured Clouds

György Dán, KTH Royal Institute of Technology; Rakesh B. Bobba, George Gross, and Roy H. Campbell, University of Illinois Urbana-Champaign

Rakesh Bobba started with an overview of electric power systems and also provided an introduction to the North American electric grid and its regulations. He said that such a grid needs a cyber-physical infrastructure with high reliability. He mentioned the usage of datacenters for smartmeter data. He added that the extension of such datacenters for the power grid is risky. Failure of the power grid leads to the failures of all its dependent sectors. In conclusion, he underscored the need for research to be done in the area of isolation, security, and fault tolerance to use clouds for power grid.

George Porter (UCSD) asked why not tighten network rules to take care of real-time problems. Rakesh replied that such rules are already there but can't be trusted. Hayadi asked about the size of the data. Rakesh said data is generated every second and so there's a lot. Shankari commented that moving existing infrastructure to the cloud could lead to new research directions and Rakesh agreed. Alysson suggested using private clouds instead of public. Rakesh said people are trying to build private clouds for the grids.

Towards a Fault-Resilient Cloud Management Stack

Xiaoen Ju, University of Michigan; Livio Soares, IBM T.J. Watson Research Center; Kang G. Shin, University of Michigan; Kyung Dong Ryu, IBM T.J. Watson Research Center

Livio Soares described cloud management stacks and mentioned that people want to build their own cloud management infrastructures. He then gave an overview of OpenStack, which they analyzed, finding that problems in the cloud stack are not trivial and that not many people are looking into the fault-resilience aspects of cloud stacks. He provided his definition of fault resilience. The authors used execution graphs to inject faults in cloud stacks to show how bugs adversely affect fault resilience. He also talked about periodic checks for monitoring. He concluded with a remark that cloud resilience must be improved and asked the question, "Can we transit from the current state to the fault resilient mode?"

Julinda Stefa (Sapienza University of Rome) asked about the overhead to implement the tool. Livio replied that logging is lightweight, but running and monitoring produce lots of overhead. Someone wondered about how to change a culture where no fixing is needed after development. Livio agreed that it's a problem with software development in general, as there is always a high cost involved to write bug-free software. In the follow-up to the same question, Rakesh pointed out that time to market is also a big constraint and therefore there are always possibilities of bugs.