# TRIPOD: An Efficient, Highly-available Cluster Management System

Paper #10


## Abstract

Driven by the increasing computational demands, cluster management systems (e.g., MESOS) are already pervasive for deploying many applications. Unfortunately, despite much effort, existing systems are still difficult to meet the high requirements of critical applications (e.g., trading and military applications), because these applications naturally require high-availability and low performance overhead in deployments. Existing systems typically replicate their job controllers so that these controllers can be highly-available and thus they can handle application failures. However, applications themselves are still often a single point of failure, leaving arbitrary unavailable time windows for themselves.

This paper proposes the design of TRIPOD, a cluster management system that automatically provides high-availability to general applications. TRIPOD's key to make applications achieve high-availability efficiently is a new PAXOS replication protocol that leverages RDMA (Remote Direct Memory Access). TRIPOD runs replicas of the same job with a replicas of controllers, and controllers agree on job requests efficiently with this protocol. Evaluation shows that TRIPOD has low performance overhead in both throughput and response time compared to an application's unreplicated execution.


## 1 Introduction

By simplifying provisioning and allowing multiple servers to be consolidated on a small number of physical hosts, virtualization has made low- and middle-end systems popular than ever. Nevertheless, the benefits of consolidation come with a hidden cost in the form of increased exposure to hardware failure.

Although recent advances in accommodating virtual machines with high availability are encouraging, existing works are still unable to ensure fault-tolerance in virtualized environment at low overhead. One typical technique is primary-backup based replication of virtual machines, including checkpoint-recovery based virtual machine replication and log-replay.

Log-replay records input and non-deterministic events of the primary machine and have them deterministically replayed on the backup machine to replicate the primary's state in case the primary node fails. However, deterministic replay is highly architecture-specific and introduces unacceptable performance degradation when applied in multi-core systems.

Checkpoint-recovery based replication of virtual machines is the other commonly used solution. It captures the entire execution state of the running VM at relatively high frequency so that changes can be reflected to the backup machine nearly instantly. However, it suffers from two problems. The first one is the network delay. In order to address output commit problem [34], output of the running VM needs to be held back until the primary receives the acknowledgement of the corresponding update from from the backup host. Apart from that, VM downtimes incurred by the checkpoint mechanism can be significant.

However, Paxos consensus is notoriously difficult to be fast. To agree on an input, traditional consensus protocols invoke at least one message round-trip between two replicas. Fortunately, Remote Direct Access Memory (RDMA) opens new opportunity.

## 2 Implementation

### 2.1 Syscall interception in Xen Hypervisor

A syscall aims at providing userland processes a way to request services from the kernel. This includes access storage operations, memory and network access, etc. Typical calls are open, close, `recv()`...

One way to perform a syscall is to use software interrupts. It consists of four steps: (1) User process copies the syscall number to execute in the CPU register EAX; (2) Execute int 0x80 to generate interrupt 0x80; (3) System calls handling routine executed in ring 0; (4) Return the result to the user process.

In order to intercept syscalls within the Xen Hypervisor itself, the first thing to do is to disable the Direct Trap optimization that allows the syscalls to bypass the hypervisor. Thus, syscalls are forced through the hypervisor and we will be able to catch them. Then, when is




*Submitted to APSys '16, August 4-5, 2016, Hong Kong, China.*

issued, the syscall will be intercepted and the information we are inerested in will be logged. Finally, we will keep on with the regular syscall execution.

## 3 Related Work

**Cluster management systems.** Cluster management systems [5, 6, 16, 17, 19, 36, 37, 42] are widespread because they can transparently support many diverse applications (e.g., Hadoop [15], Dryad [18], and key-value stores [33]). These existing systems mainly focus on high availability for themselves by replicating important components (e.g., controllers) within these systems, or focus on fault-recovery of applications [42]. To the best of our knowledge, no existing system provides efficient and general high-availability service to applications. Although TRIPOD's current design leverages an existing system Mesos [16], its general PAXOS protocol design can also be integrated in other systems.

**State machine replication (SMR).** SMR is a powerful, but complex fault-tolerance technique. The literature has developed a rich set of PAXOS algorithms [23, 24, 27, 30, 35] and implementations [7, 8, 27]. PAXOS is notoriously difficult to be fast and scalable [28]. To improve speed and scalability, various advanced replication models have been developed [13, 22, 26, 30]. Since consensus protocols play a core role in datacenters [4, 16, 41] and distributed systems [9, 26], a variety of study have been conducted to improve different aspects of consensus protocols, including performance [25, 30, 32], understandability [24, 31], and verifiable reliability rules [14, 40]. Although TRIPOD tightly integrates RDMA features in PAXOS, its implementation mostly complies with a popular, practical approach [27] for reliability. Other PAXOS approaches can also be leveraged in TRIPOD.

**RDMA techniques.** RDMA techniques have been implemented in various architectures, including Infiniband [1], RoCE [2], and iWRAP [3]. RDMA have been leveraged in many systems to improve application-specific latency and throughput, including high performance computing [12], key-value stores [10, 20, 21, 29], transactional processing systems [11, 38], and file systems [39]. These systems are largely complementary to TRIPOD.

## 4 Conclusion

We have presented the design of TRIPOD, the first cluster management system design that automatically provides high-availability to general applications. TRIPOD leverages an RDMA-enabled PAXOS implementation to efficiently agree on job requests for critical applications across its controllers, and it replicates the same jobs to make these jobs highly available. TRIPOD also presents a design on achieving tail-tolerance with its fault-tolerance architecture. Initial evaluation shows that TRIPOD's protocol has reasonable overhead, which may be suitable for critical application with short response time requirements.

## References


[1] An Introduction to the InfiniBand Architecture. `http://buyya.com/superstorage/chap42.pdf`.

[2] Mellanox Products: RDMA over Converged Ethernet (RoCE). `http://www.mellanox.com/page/products_dyn?product_family=79`.

[3] RDMA iWARP. `http://www.chelsio.com/nic/rdma-iwarp/`.

[4] Why the data center needs an operating system. `cloud.berkeley.edu/data/dcos.pptx`.

[5] Tupperware. `https://www.youtube.com/watch?v=C_WuUgTqgOc`, 2014.

[6] E. Boutin, J. Ekanayake, W. Lin, B. Shi, J. Zhou, Z. Qian, M. Wu, and L. Zhou. Apollo: Scalable and coordinated scheduling for cloud-scale computing. In *Proceedings of the 11th USENIX Conference on Operating Systems Design and Implementation*, OSDI'14, pages 285–300, Berkeley, CA, USA, 2014. USENIX Association.

[7] M. Burrows. The chubby lock service for loosely-coupled distributed systems. In *Proceedings of the Seventh Symposium on Operating Systems Design and Implementation (OSDI '06)*, pages 335–350, 2006.

[8] T. D. Chandra, R. Griesemer, and J. Redstone. Paxos made live: An engineering perspective. In *Proceedings of the Twenty-sixth Annual ACM Symposium on Principles of Distributed Computing (PODC '07)*, Aug. 2007.

[9] J. C. Corbett, J. Dean, M. Epstein, A. Fikes, C. Frost, J. J. Furman, S. Ghemawat, A. Gubarev, C. Heiser, P. Hochschild, W. Hsieh, S. Kanthak, E. Kogan, H. Li, A. Lloyd, S. Melnik, D. Mwaura, D. Nagle, S. Quinlan, R. Rao, L. Rolig, Y. Saito, M. Szymaniak, C. Taylor, R. Wang, and D. Woodford. Spanner: Google's globally-distributed database. Oct. 2012.

[10] A. Dragojević, D. Narayanan, O. Hodson, and M. Castro. Farm: Fast remote memory. In *Proceedings of the 11th USENIX Conference on Networked Systems Design and Implementation*, NSDI'14, 2014.

[11] A. Dragojević, D. Narayanan, E. B. Nightingale, M. Renzelmann, A. Shamis, A. Badam, and M. Castro. No compromises: Distributed transactions with consistency, availability, and performance. In *Proceedings of the 25th ACM Symposium on Operating Systems Principles (SOSP '15)*, Oct. 2015.

[12] M. P. I. Forum. Open mpi: Open source high performance computing, Sept. 2009.

[13] L. Glendenning, I. Beschastnikh, A. Krishnamurthy, and T. Anderson. Scalable consistency in scatter. In *Proceedings of the 23rd ACM Symposium on Operating Systems Principles (SOSP '11)*, Oct. 2011.

[14] H. Guo, M. Wu, L. Zhou, G. Hu, J. Yang, and L. Zhang. Practical software model checking via dynamic interface reduction. In *Proceedings of the 23rd ACM Symposium on Operating Systems Principles (SOSP '11)*, pages 265–278, Oct. 2011.

[15] Hadoop. `http://hadoop.apache.org/core/`.

[16] B. Hindman, A. Konwinski, M. Zaharia, A. Ghodsi, A. D. Joseph, R. Katz, S. Shenker, and I. Stoica. Mesos: A platform for fine-grained resource sharing in the data center. In *Proceedings of the 8th USENIX conference on Networked Systems Design and Implementation*, NSDI'11, Berkeley, CA, USA, 2011. USENIX Association.

[17] M. Isard. Autopilot: Automatic data center management. *SIGOPS Oper. Syst. Rev.*, 41(2):60–67, Apr. 2007.

[18] M. Isard, M. Budiu, Y. Yu, A. Birrell, and D. Fetterly. Dryad: distributed data-parallel programs from sequential building blocks. In *EuroSys '07: Proceedings of the Second ACM SIGOPS/EuroSys European Conference on Computer Systems 2007*, pages 59–72, 2007.

[19] M. Isard, V. Prabhakaran, J. Currey, U. Wieder, K. Talwar, and A. Goldberg. Quincy: Fair scheduling for distributed computing clusters. In *Proceedings of the ACM SIGOPS 22Nd Symposium on Operating Systems Principles*, SOSP '09, pages 261–276, New York, NY, USA, 2009. ACM.

[20] J. Jose, H. Subramoni, K. Kandalla, M. Wasi-ur Rahman, H. Wang, S. Narravula, and D. K. Panda. Scalable memcached design for infiniband clusters using hybrid transports. In *Proceedings of the 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (Ccgrid 2012)*, CCGRID '12, 2012.

[21] A. Kalia, M. Kaminsky, and D. G. Andersen. Using rdma efficiently for key-value services. Aug. 2014.

[22] M. Kapritsos and F. P. Junqueira. Scalable agreement: Toward ordering as a service. In *Proceedings of the Sixth International Conference on Hot Topics in System Dependability*, HotDep'10, 2010.

[23] L. Lamport. Paxos made simple. `http://research.microsoft.com/en-us/um/people/lamport/pubs/paxos-simple.pdf`.

[24] L. Lamport. The part-time parliament. *ACM Trans. Comput. Syst.*, 16(2):133–169, 1998.

[25] L. Lamport. Fast paxos. Fast Paxos, Aug. 2006.

[26] Y. Mao, F. P. Junqueira, and K. Marzullo. Mencius: building efficient replicated state machines for wans. In *Proceedings of the 8th USENIX conference on Operating systems design and implementation*, volume 8, pages 369–384, 2008.

[27] D. Mazieres. Paxos made practical. Technical report, Technical report, 2007. http://www. scs. stanford. edu/dm/home/papers, 2007.

[28] E. Michael. *Scaling Leader-Based Protocols for State Machine Replication*. PhD thesis, University of Texas at Austin, 2015.

[29] C. Mitchell, Y. Geng, and J. Li. Using one-sided rdma reads to build a fast, cpu-efficient key-value store. In *Proceedings of the USENIX Annual Technical Conference (USENIX '14)*, June 2013.

[30] I. Moraru, D. G. Andersen, and M. Kaminsky. There is more consensus in egalitarian parliaments. In *Proceedings of the 13th ACM Symposium on Operating Systems Principles (SOSP '91)*, Nov. 2013.

[31] D. Ongaro and J. Ousterhout. In search of an understandable consensus algorithm. In *Proceedings of the USENIX Annual Technical Conference (USENIX '14)*, June 2014.

[32] M. Poke and T. Hoefler. Dare: High-performance state machine replication on rdma networks. In *Proceedings of the 24th International Symposium on High-Performance Parallel and Distributed Computing*, HPDC '15, 2015.

[33] `http://redis.io/`.

[34] R. E. Strom, D. F. Bacon, and S. Yemini. *Volatile logging in n-fault-tolerant distributed systems*. IBM Thomas J. Watson Research Division, 1987.

[35] R. Van Renesse and D. Altinbuken. Paxos made moderately complex. *ACM Computing Surveys (CSUR)*, 47(3):42:1–42:36, 2015.

[36] V. K. Vavilapalli, A. C. Murthy, C. Douglas, S. Agarwal, M. Konar, R. Evans, T. Graves, J. Lowe, H. Shah, S. Seth, B. Saha, C. Curino, O. O'Malley, S. Radia, B. Reed, and E. Baldeschwieler. Apache hadoop yarn: Yet another resource negotiator. In *Proceedings of the 4th Annual Symposium on Cloud Computing*, SOCC '13, pages 5:1–5:16, New York, NY, USA, 2013. ACM.

[37] A. Verma, L. Pedrosa, M. Korupolu, D. Oppenheimer, E. Tune, and J. Wilkes. Large-scale cluster management at google with borg. In *Proceedings of the Tenth European Conference on Computer Systems*, EuroSys '15, pages 18:1–18:17, New York, NY, USA, 2015. ACM.

[38] X. Wei, J. Shi, Y. Chen, R. Chen, and H. Chen. Fast in-memory transaction processing using rdma and htm. In *Proceedings of the 25th ACM Symposium on Operating Systems Principles (SOSP '15)*, SOSP '15, Oct. 2015.

[39] G. G. Wittawat Tantisiriroj. Network file system (nfs) in high performance networks. Technical Report CMU-PDLSVD08-02, Carnegie Mellon University, Jan. 2008.

[40] J. Yang, T. Chen, M. Wu, Z. Xu, X. Liu, H. Lin, M. Yang, F. Long, L. Zhang, and L. Zhou. MODIST: Transparent model checking of unmodified distributed systems. In *Proceedings of the Sixth Symposium on Networked Systems Design and Implementation (NSDI '09)*, pages 213–228, Apr. 2009.

[41] M. Zaharia, B. Hindman, A. Konwinski, A. Ghodsi, A. D. Joesph, R. Katz, S. Shenker, and I. Stoica. The datacenter needs an operating system. In *Proceedings of the 3rd USENIX Conference on Hot Topics in Cloud Computing*, 2011.

[42] Z. Zhang, C. Li, Y. Tao, R. Yang, H. Tang, and J. Xu. Fuxi: A fault-tolerant resource management and job scheduling system at internet scale. *Proc. VLDB Endow.*, 7(13):1393–1404, Aug. 2014.