Byzantine Fault Tolerance for the Cloud

Hans P. Reiser

University of Lisbon Faculty of Science, Portugal*

Abstract. CloudFIT is an ongoing project that designs an architecture for intrusion-tolerant applications that can be deployed dynamically in the cloud. This position paper presents an outline of the architecture that is being developed in the project, and discusses the implications of the deployment in the cloud. We explore to what extent existing BFT algorithms can be used for increasing security and availability in the proposed architecture and what issues still need to be resolved in the future.

1 Motivation

Cloud computing has become a successful new paradigm in the IT industry. Computer infrastructure, platforms, and applications can dynamically be provisioned remotely over a network. In the future it is likely that many even mission-critical services with high security and availability requirements will move "to the cloud".

For over a decade, Byzantine fault tolerance (BFT) has been studied as a mechanism to improve availability and security of practical systems. It is appealing to use BFT for critical services deployed in a cloud. For example, a service might be hosted by multiple independent cloud providers, such that it tolerates faults in a subset of the clouds.

While improving the performance of BFT algorithms in practical systems has been at the center of interest of many researchers, three important aspects relevant in cloud-based systems are under-represented in previous work on BFT. First, BFT services need to be able to recover automatically from faults (both crashes and malicious faults). Second, a cloud-based system has a more complex architecture and potentially has multiple trust levels (for example, the trust in the cloud infrastructure can be different from the trust in the service instance itself). Third, the cloud makes it easy to dynamically change resources allocated to a service, and this mechanism might also be used to improve the service quality.

In the CloudFIT project[1], we investigate how to use BFT in order to develop fault and intrusion tolerant applications for the cloud. The goal is to define a modular architecture for BFT replication, with building blocks for BFT consensus (configurable for various trust settings), replica recovery, state synchronization, and resource allocation. This position paper presents the system model and general over-all architecture for secure and fault-tolerant cloud applications in CloudFIT and sketches the contributions that we want to make to BFT research.

2 Byzantine Fault Tolerance in CloudFIT

2.1 System model

Fault model. We assume that a service is replicated on multiple virtual machines deployed in the cloud, potentially across multiple cloud providers. Each replica may fail in arbitrary ways. The number of replicas that are faulty at a given time is bounded. Furthermore, in a first variant

* starting from 2011-03-01: Institute of IT-Security and Security Law, University of Passau, Germany

[1] `http://cloudfit.di.fc.ul.pt`

of the architecture we assume that the cloud infrastructure itself is trusted, i.e., that it fails only by crashing.

We are aware that assuming that the correctness of a virtualization infrastructure (typically, several 100'000s of lines of code [6]) is a strong, potentially unrealistic assumption. A revised variant of the architecture will use a model in which only part of the cloud infrastructure is trusted. We expect that using nested virtualization [1] it will become feasible to have a small, verified virtualization layer that is trustworthy, and on top of it run a traditional, fully-fledged cloud infrastructure that does not need to be trusted.

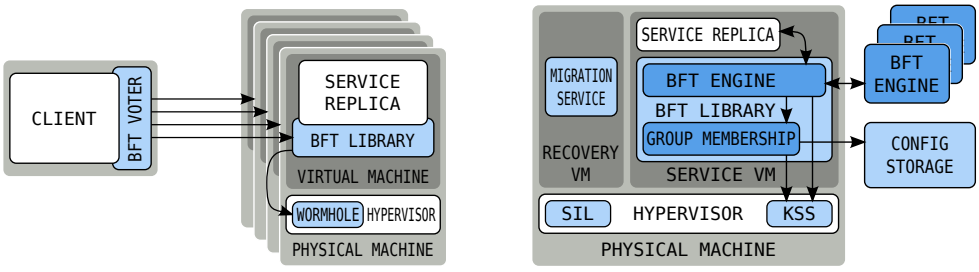
Dynamic replication groups. One advantage of cloud computing is the dynamicity of resource provisioning. Our architecture makes use of this advantage by enabling dynamic runtime modifications of replication groups. For example, in a replication group, increasing the number of replicas typically reduces the cost of read-only operations, while increasing the cost of modifying operations. Dynamically adapting the number of replicas may be used to increase service quality, but requires careful modifications to the BFT protocols in use.

Recovery. Replicas that fail need to be recovered, either in reaction to the detection of a fault, or proactively (e.g., triggered by time). One advantage of a computing cloud is that it is easy to dynamically allocate new resources for replacing a faulty replica. In previous work [4,2] we have shown that this mechanism can be used efficiently for instantiating new service replicas for proactive recovery.

A disadvantage of proactive recovery is that it adds timing assumptions to BFT replication. Removing timing assumption from BFT algorithm has been a key motivation of a great amount of research work in the area. However, it has been shown [5] that previously proposed systems with asynchronous proactive recovery do not achieve the goal of tolerating any number of failures over system lifetime. Our approach to this problem is (a) the separation between BFT replication and recovery and (b) weakening the timing assumptions used by the recovery component.

2.2 Over-all architecture

Fig. 1(a) shows the basic interactions in the architecture of our system. Clients access service replicas on the basis of a BFT library that is used for replication. As we assume that the virtualization infrastructure of the cloud provider is trustworthy, it is possible to use trusted functionality of the infrastructure (denoted "wormhole" in the figure).



(a) Basic interaction between clients and replicas

(b) Internal composition of low-level building blocks at each replica

Fig. 1: CloudFIT architecture

Fig. 1(b) illustrates in more detail the internal decomposition of a replica. The KSS (key storage and signing service) is one of the trusted components and corresponds USIG service [7] used in the EBAWA BFT algorithm. The config storage is a trusted component that is used to coordinate reconfigurations. The SIL (secure image launcher) is responsible for creating correct virtual machines, and is used initially and for all recovery operations. The recovery VM is used to access state of an old service VM during a recovery operation.

2.3 BFT for the cloud

One major goal of our work is to define a more flexible architecture for BFT replication and implement it as an extension to the BFT-SMaRt[2] library. We decompose the system into the following components:

- *BFT atomic multicast* is used for replication, on the basis of *BFT consensus*. Our current prototype is based on the EBAWA algorithm [7], which uses a trusted component (USIG – unique sequential identifier generator) and requires only $2f + 1$ replicas. We extend this algorithm for recovery and reconfiguration.
- *Initialization* is required on startup and on recoveries for distributing keys and other initial values. Specifically, for the EBAWA algorithm, an initial counter value for the USIG needs to be communicated to all replicas. This functionality is specified as a separate component.
- *Reconfiguration* is a component that coordinates the reconfiguration of the BFT replication library (number of nodes, algorithm, and internal parameters), controlled by a trusted *CONFIG STORAGE*, using an approach inspired by [3].
- *Recovery* is handled by a component that triggers proactive recoveries in a secure way, and optionally also handles reactive recoveries (using an external fault detector, which is not discussed here).
- *BFT state migration* handles the transfer of state between replicas during recovery operations, based on our previous work.

The key challenges that we address are the following: We want to find and formalize adequate abstractions for the components that are just outlined informally above. Having the right abstraction will make it easier to argue about the correctness of proactive recovery algorithms, and also apply a recovery strategy to multiple BFT algorithms. Ideally, the same specification of the recovery component can be used for multiple BFT algorithms.

Concluding, we are convinced that BFT is a key mechanism that can be used to increase the availability and security of services in the cloud. In addition, BFT systems can benefit from cloud computing through the cloud's dynamic properties and the possibility of providing a trusted component.

Acknowledgments

This position paper is based on joint work with Marcelo Pasin and Alysson Bessani (both University of Lisbon, Portugal) and Christian Spann (University of Ulm, Germany). Part of the work was supported by Fundação para a Ciência e a Tecnologia through the Carnegie Mellon Portugal Program and the project PTDC/EIA-CCO/108299/2008.

[2] `http://code.google.com/p/bft-smart/`

References

1. Muli Ben-Yehuda, Michael D. Day, Zvi Dubitzky, Michael Factor, Nadav Har'El, Abel Gordon, Anthony Liguori, Orit Wasserman, and Ben-Ami Yassour. The turtles project: Design and implementation of nested virtualization. In *OSDI '10: 9th USENIX Symposium on Opearting Systems Design and Implementation*. USENIX Association, 2010.
2. Tobias Distler, Rüdiger Kapitza, Ivan Popov, Hans P. Reiser, and Wolfgang Schröder-Preikschat. SPARE: Replicas on Hold. In Internet Society (ISOC), editor, *Proceedings of the 18th Network and Distributed System Security Symposium (NDSS '11)*, 2011.
3. Leslie Lamport, Dahlia Malkhi, Lidong Zhou, and Microsoft Research. Vertical Paxos and Primary-Backup Replication, 2009.
4. Hans P. Reiser and Rüdiger Kapitza. Hypervisor-Based Efficient Proactive Recovery. In IEEE, editor, *Proc. of the of the 26th IEEE Symposium on Reliable Distributed Systems - SRDS'07*, 2007.
5. Paulo Sousa, Nuno Ferreira Neves, and Paulo Verissimo. Hidden problems of asynchronous proactive recovery. In *Proceedings of the 3rd workshop on on Hot Topics in System Dependability*, Berkeley, CA, USA, 2007. USENIX Association.
6. Udo Steinberg and Bernhard Kauer. Nova: a microhypervisor-based secure virtualization architecture. In *Proceedings of the 5th European conference on Computer systems*, EuroSys '10, pages 209–222, New York, NY, USA, 2010. ACM.
7. Giuliana Santos Veronese, Miguel Correia, Alysson Neves Bessani, and Lau Cheuk Lung. Ebawa: Efficient byzantine agreement for wide-area networks. *High-Assurance Systems Engineering, IEEE International Symposium on*, 0:10–19, 2010.