Brief Announcement: When You Don't Trust Clients: Byzantine Proposer Fast Paxos

Keith Marzullo[1], Hein Meling[2], and Alessandro Mei*[3]

[1] Dept. of Computer Science and Engineering, University of California, San Diego
[2] Dept. of Computer Science and Engineering, University of Stavanger, Norway
[3] Dept. of Computer Science, Sapienza University of Rome, Italy

1 Introduction

State machine replication is a general approach for constructing fault-tolerant services, and a key protocol underlying state machine replication is consensus. The set of Byzantine failures is so large that it has been applied for masking the effects of compromised systems, and so Byzantine-tolerant consensus has been used to construct systems that are meant to ameliorate the effect of compromise (see [1] among others). In the Byzantine model, there is no trust among processes: any process can behave in an arbitrarily faulty manner. However, in multi-site systems, processes in the same administrative domain typically have a measure of mutual trust. This is because such processes share fate: for example, if a process in a domain is compromised, then other processes—perhaps all of them—can be compromised as well, and the local services they rely upon may be compromised. In [4], this observation was used to argue for the *Mutually Suspicious Domain* (MSD) model, in which there is mutual trust between processes in a domain, but no trust for inter-domain communication, i.e., processes within a domain must protect itself from possible uncivil behavior from processes in other domains.

In this paper, we propose a consensus protocol for state machine replication under the MSD model. We assume the typical Internet model, in which the servers are in the same administrative domain and replicated for increased availability, and clients are in other administrative domains. The protocol, which we call BP Fast Paxos, uses a hybrid failure model: processes within a domain assume a crash failure model while across domains they assume a Byzantine failure model. BP Fast Paxos is derived from Paxos [2], and provides low latency for client requests, can tolerate any number of (Byzantine) faulty clients, up to 1/3 (crash) faulty servers, and can protect itself against denial of service attacks.

The contribution of this paper is the insight that we can tolerate Byzantine clients, when we make the assumption that servers are benign faulty. We believe this is a realistic assumption, since client software is often more exposed, while servers are typically behind corporate firewalls and intrusion detection systems, and thus better protected from compromise.

* This work was performed while Alessandro Mei was a Marie Curie Fellow at the Computer Science and Engineering Department, University of California San Diego, USA. The fellowship is funded by the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n. 253461.

2 Contribution: BP Fast Paxos

In BP Fast Paxos consensus is reached by the interaction between proposers and acceptors, and learners that learn the consensus value. Given our failure model, the mapping of these roles to clients and servers have two obvious choices: (1) clients are proposers, and servers are acceptors and learners, and (2) proposers are separate from both clients and servers. In the latter case, proposers interact with clients, and can be placed at the edge of a data center, and is thus more exposed to compromise than servers inside the data center.

We develop BP Fast Paxos by modifying Paxos in three essential ways: (i) The ⟨PREPARE⟩ message sent by a proposer is replaced with a ⟨TRUSTCHANGE⟩ message sent by acceptors. The reason for this change is that the acceptors can then detect misbehavior of a Byzantine proposer, and simply change its trust to another proposer whom can conclude the protocol. (ii) We also require that ⟨ACCEPT⟩ messages (except those sent in round 0) contain a *proof* that it is legitimate. To construct this proof, a proposer must collect signed ⟨PROMISE⟩ messages from acceptors, so that acceptors can verify the proof against the value of the ⟨ACCEPT⟩ message. (iii) Finally, we introduce a mechanism to *detect equivocation* by the proposer for the current round. That is, if the current proposer sends different ⟨ACCEPT⟩ messages to different acceptors, then we can detect this using mechanisms similar to those used in Fast Paxos [3] to detect a collision in a fast round. This change requires that acceptors are also learners, enabling them to detect misbehavior through ⟨LEARN⟩ messages, and replace a misbehaving proposer through a *trust change*.

In the normal case behavior of the algorithm, where no agents are faulty, a single unique proposer will try to have its value accepted by the acceptors. This initial accept does not contain any signatures, and consensus will complete in two communication steps, if no failures occur. Misbehavior is detected by learners (and thus acceptors), in which case a trust change is required. In this case, also HMAC-based signatures are necessary. If misbehavior or a crash is detected, then more rounds are necessary.

Many protocols have been derived from Paxos, however, BP Fast Paxos is the first protocol to leverage the separation of agent roles to distinguish their individual failure assumptions. The protocol is two-step and is safe even when all proposers are Byzantine, and does not require signatures for the common case.

References

1. M. Castro and B. Liskov. Practical byzantine fault tolerance and proactive recovery. *ACM Trans. Comput. Syst.*, 20(4):398–461, 2002.
2. L. Lamport. The part-time parliment. *ACM Trans. on Comp. Syst.*, 16(2):133–169, 1998.
3. L. Lamport. Fast paxos. *Distributed Computing*, 19(2):79–103, 2006.
4. Y. Mao, F. Junqueira, and K. Marzullo. Towards low latency state machine replication for uncivil wide-area networks. In *Fifth Workshop on Hot Topics in Dependable Systems (HotDep'09)*, Estoril, Lisbon, Portugal, June 2009.