Vote Them Out: Detecting and Eliminating Byzantine Peers

Tuan Tran
Priyanka Mondal
atran18@ucsc.edu
pmondal@ucsc.edu
University of California, Santa Cruz
Santa Cruz, California

Roy Shadmon
Manthan Mallikarjun
rshadmon@ucsc.edu
mmallika@ucsc.edu
University of California, Santa Cruz
Santa Cruz, California

Peter Alvaro
Owen Arden
palvaro@ucsc.edu
oarden@ucsc.edu
University of California, Santa Cruz
Santa Cruz, California

ABSTRACT

Byzantine Fault Tolerant (BFT) protocols are designed to ensure correctness and eventual progress in the face of misbehaving nodes [1]. However, this does not prevent negative effects an adversary may have on performance: a faulty node may significantly affect the latency and throughput of the system without being detected. This is especially true in speculative protocols optimized for the best-case where a single leader can force the protocol into the worst case [3]. Systems like Aardvark [2] that are designed to maximize worst-case performance tolerate byzantine behavior without necessarily detecting who the perpetrator is. By forcing regular view changes, for example, they mitigate the effects of leaders who deliberately delay dissemination of messages, even if this behavior would be difficult to prove to a third party.

Byzantine faults, by definition, can be difficult to detect. An error of 'commission', such as a message with a mismatching digest, can be proven. Errors of 'omission', such as delaying or failing to relay a message, as a rule cannot be proven, and the node responsible for these types of omission faults may not appear faulty to all observers. Nevertheless, we observe that they can reliably be detected. Designing protocols that detect and eject nodes is challenging for two reasons. First, some behaviors are observed by a subset of honest nodes and cannot be objectively proven to a third party. Second, any mechanism capable of ejecting nodes could be subverted by Byzantine nodes to eject honest nodes.

This paper presents the Protocol for Ejecting All Corrupted Hosts (Peach, a mechanism for detecting and ejecting faulty nodes in Byzantine fault tolerant (BFT) protocols. Nodes submit votes to a trusted configuration manager that replaces faulty nodes once a threshold of votes are received. We implement Peach for two BFT protocol variants, a traditional pbft-style three-phase protocol and a speculative protocol, and evaluate its ability to respond to Byzantine behavior.

This work makes the following contributions:

(1) We present and prove a necessary and sufficient constraint on cluster membership guaranteeing that any nodes causing performance degradation via acts of omission will be detected.
(2) We present an agreement protocol, PEACHes, in which replicas pass votes about their subjective local observations of possible omissions to a TTP.
(3) We show how the separation of detection and effectuation allows fine-grained detection of malicious behavior that is compatible and easily integrated with existing systems.
(4) We present DecentBFT, an extension of BFT-Smart to which we added a speculative fast path (similar to Zyzzva) and integrated PEACHes.
(5) We show DecentBFT rapidly detects and mitigates a variety of performance attacks that would have gone undetected by the state of the art.

ACM Reference Format:
Tuan Tran, Priyanka Mondal, Roy Shadmon, Manthan Mallikarjun, Peter Alvaro, and Owen Arden. 2019. Vote Them Out: Detecting and Eliminating Byzantine Peers. In *ACM Symposium on Cloud Computing (SoCC '19), November 20–23, 2019, Santa Cruz, CA, USA.* ACM, New York, NY, USA, 1 page. https://doi.org/10.1145/3357223.3365442

SoCC '19, November 20–23, 2019, Santa Cruz, CA, USA

ACM ISBN 978-1-4503-6973-2/19/11.
https://doi.org/10.1145/3357223.3365442

REFERENCES

[1] Miguel Castro, Barbara Liskov, et al. 1999. Practical Byzantine fault tolerance. In *OSDI*, Vol. 99. 173–186.
[2] Allen Clement, Edmund L Wong, Lorenzo Alvisi, Michael Dahlin, and Mirco Marchetti. 2009. Making Byzantine Fault Tolerant Systems Tolerate Byzantine Faults.. In *NSDI*, Vol. 9. 153–168.
[3] Ramakrishna Kotla, Lorenzo Alvisi, Mike Dahlin, Allen Clement, and Edmund Wong. 2007. Zyzzyva: speculative byzantine fault tolerance. In *ACM SIGOPS Operating Systems Review*, Vol. 41. ACM, 45–58.