Poster: Client-Side Byzantine Attacks on Raft Algorithm in Blockchain

Donghee Kim
Korea University
dhkim@isslab.korea.ac.kr

Junbeom Hur
Korea University
jbhur@korea.ac.kr

***Abstract*—Raft is a widely used consensus algorithm in blockchain system ensuring each node in the system to agree upon the data in blocks and maintain their integrity. Raft has advantages of toleration for crash faults and high performance in terms of consensus time. In this study, we propose novel client-side Byzantine attacks on the raft algorithm. We then demonstrate the efficacy of our attacks, and discuss its root cause and mitigation strategies.**

I. Introduction

The blockchain system consists of two core technologies: consensus algorithm and distributed ledger technology. Raft [1] is one of the consensus algorithms for maintaining identical chain among participants and supporting verifiability of the block data in the blockchain system. Since the raft algorithm is operated by a single leader, it has the advantage of high speed in transaction processing [2] and toleration for crash faults as long as at least a half of the nodes are benign in the system. However, the raft algorithm is vulnerable to Byzantine attack by malicious nodes, especially manipulating data to violate data integrity [3], [4], [5]. In this paper, we describe novel Byzantine attacks focusing on the consensus protocol of a client transmitting transactions to the leader node, leading to consensus failure and performance degradation of the blockchain application. we will show effectiveness of our attacks through performance and security analysis.

II. Raft Consensus Algorithm

Raft is the leader-based consensus algorithm. In the algorithm, each node participaring in the consensus procedure is assigned one the following roles: leader, follower, and candidate [1]. The leader can receive transactions from the client, record the set of received transactions in the block, and broadcast the block to all participant nodes in the blockchain system. The follower is only able to receive blocks from the leader, and verify the liveness of the leader by checking heartbeat messages from the leader continuously. If one of the followers cannot confirm the liveness of the current leader, the follower changes it role into the candidate. It then increase its 'term' number which is the time sequence indicator, and triggers a new leader election procedure, assuming crash fault might happen on the current leader. To inform the leader election, the candidate broadcasts `RequestVote` message to the network. When multiple candidates trigger the election procedure at the same time, a candidate with the highest term number is selected by the followers; and the candidate is elected as leader via a majority voting method, and advertises the result. All of the nodes in the system then synchronize the log. To reach consensus and maintain data consistency, only the leader having the highest term can commit the transactions. Thus, if clients need to commit transactions, they need to know the leader first.

III. Byzantine Attacks on Raft

In the raft-based blockchain system, when a node receives transaction request messages from a client, it responds with different data to the client depending on its role. If it is a follower, it sends a reject message with information of the leader of the current term to the client on the request; if a candidate, it transmits just a reject message to the client, because there is no leader at the moment. However, if the client interacts with a faulty or malicious node, the current term's information that the client receives may be false, making it difficult to find the correct leader. Specifically, when Byzantine attackers send fake messages about current leader to the client, it will cause denial-of-service due to the delay of finding a leader as well as manipulation of transaction data.

A. Attack Scenario 1

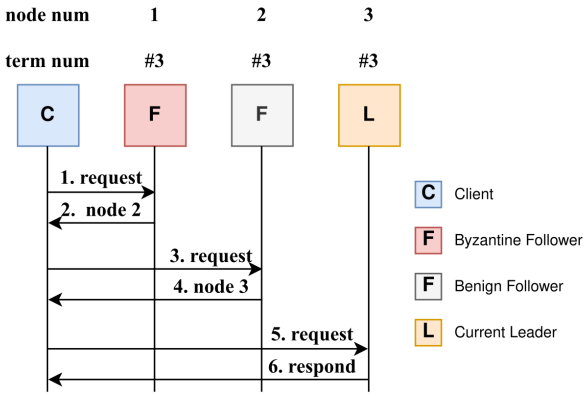


Fig. 1: Attack Scenario 1

This attack scenario considers the case that a malicious follower receives the transaction request message from a client, and deliberately informs the client of a fake node as the leader. As shown in Fig. 1, when node 1 who is the malicious follower receives the request from a client, it responds to the client that node 2 is a leader. Since node 2 is a follower in current term, it just notifies that node 3 is the leader, when it receives the request message from the client later. Even worse, if a new

leader is being elected while the client communicate with node 2, it may incur additional redirection on the client to find the current leader and request the transaction, causing latency to commit transactions. If the new leader election process begins, or the leader is changed (starting next term) before client figures it out, the client can hardly find the current leader quickly, resulting in lower throughput of the whole blockchain system.

B. Attack Scenario 2

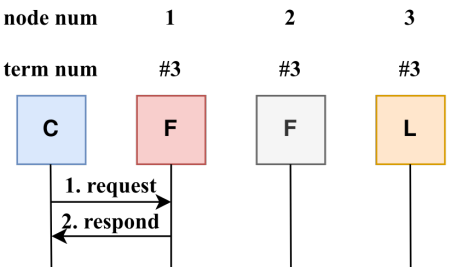


Fig. 2: Attack Scenario 2

This attack scenario considers the case that the attacker deceives the client that it is the current leader in order to be elected as a leader. As shown in Fig. 2, when the client requests a transaction to the Byzantine follower (node 1), node 1 responds to the client as it is the leader. If the client receives the response from node 1, the client sends set of transactions to node 1 until it detects any fault during processing the transactions. However, since the follower do not have permission to replicate and broadcast logs, such transactions cannot be appended to log and operated correctly. Therefore, the client has to request transactions to the actual leader again, causing additional latency to operate transactions of the client.

IV. Analysis

A. Performance

We first evaluate the performance of blockchain systems under the proposed Byzantine attacks in terms of transaction latency. As described in the attack scenario 1, if the leader changes while the follower delivers the manipulated response to the client, both the transaction latency and throughput would be deteriorated, because the client takes longer to find the real leader.

Transaction latency is defined as the time from the transaction is requested by the client to the transaction is committed from the actual leader. Formally, it is defines as the equation $t_l = t_c - t_r$, where t_l is transaction latency, t_c is transaction commit time, and t_r is transaction request time. Specifically, given the notations in Table I, t_c is computed as a follow on average:

$$t_c = t_r + RTT(1+q)(1+p),$$

where $t_l = RTT(1+q)(1+p)$. Therefore, the transaction latency would be increased in linear to the number of Byzantine attackers and attack success rate in the blockchain system.

TABLE I: Notations

Notations	Description
m	Number of all nodes
n	Number of Byzantine nodes
p	Percentage of Byzantine nodes ($= n/m$)
q	Attack success rate
RTT	Round trip time for transaction request/response message

B. Security

When a client receives a response about the leader from the node it communicates with, the client begins the transaction operations with the claimed leader by sending transaction data to the leader. If the leader is the malicious follower (e.g., attack scenario 2), the attacker is able to obtain transaction data from the client before the leader obtains it, which can be further abused to mount the other attacks such as blockchain folk.

V. Conclusion and Discussion

In this study, we represented client-side attacks exploiting diverse faults in raft-based blockchain systems, and evaluated their efficacy. The root cause for the vulnerability is the lack of authentication mechanism on the client side. One of the mitigation strategies is to adopt the authentication mechanism on the client, enabling it to verify whether the notified leader information is correct. How to authenticate the raft message on the client side in an efficient manner is an important future work for the secure raft-based blockchain system.

Acknowledgment

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the MSIT (Ministry of Science and ICT), Korea (No.2022-0-00411, IITP-2023-2020-0-01819, IITP-2022-2021-0-01810).

References

[1] D. Ongaro and J. Ousterhout, "In search of an understandable consensus algorithm," in *2014 USENIX Annual Technical Conference (Usenix ATC 14)*, 2014, pp. 305–319.

[2] L.-e. Wang, Y. Bai, Q. Jiang, V. C. Leung, W. Cai, and X. Li, "Beh-raft-chain: A behavior-based fast blockchain protocol for complex networks," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 2, pp. 1154–1166, 2020.

[3] G. Zhang and H.-A. Jacobsen, "Prosecutor: An efficient bft consensus algorithm with behavior-aware penalization against byzantine attacks," in *Proceedings of the 22nd International Middleware Conference*, 2021, pp. 52–63.

[4] H. M. Buttar, W. Aman, M. Rahman, and Q. H. Abbasi, "Countering active attacks on raft-based iot blockchain networks," *arXiv preprint arXiv:2204.00838*, 2022.

[5] W. Wang, S. Deng, J. Niu, M. K. Reiter, and Y. Zhang, "Engraft: Enclave-guarded raft on byzantine faulty nodes," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, 2022, pp. 2841–2855.

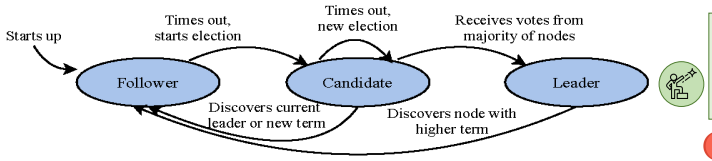
Poster: Client-Side Byzantine Attacks on Raft Algorithm in Blockchain

Donghee Kim
Korea University
dhkim@isslab.korea.ac.kr

Junbeom Hur
Korea University
jbhur@korea.ac.kr

INTRODUCTION

- Raft is a consensus algorithms for (1) maintaining identical chain among participants, and (2) supporting verifiability of the block data in the blockchain system [1],[2]
- Raft is vulnerable to Byzantine attack by malicious nodes, manipulating data to violate data integrity [3], [4], [5]
- We describe novel client-side Byzantine attacks focusing on the consensus protocol of a client transmitting transactions to the leader node
- Three roles in raft: leader, candidate, follower

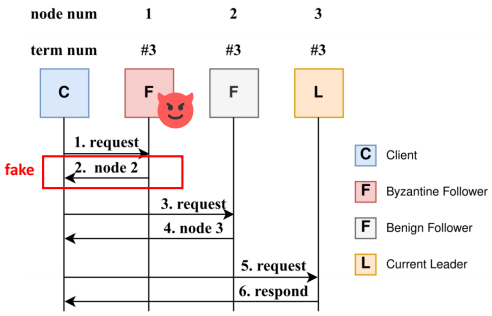


- Receive transactions from the client
- Record the set of received transactions in the block
- Broadcast the block to all participant nodes in the blockchain system

If clients need to commit transactions, they need to know the leader first.

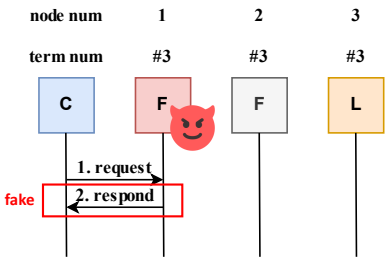
CLIENT-SIDE BYZANTINE ATTACKS ON RAFT

▪ **Attack Scenario 1**



- Attacker message: ***a fake node*** as the current leader
- Even worse, if leader change is triggered, **latency** to commit transactions is caused

▪ **Attack Scenario 2**



- Attacker message: ***it is the current leader***
- The client has to request transactions to the actual leader again, causing additional latency to operate transactions of the client

ANALYSIS

▪ Performance Analysis (Attack Scenario 1)

- If the leader is changed, both the transaction latency and throughput would be deteriorated
- Transaction latency : $t_l = t_c - t_r = RTT(1+q)(1+p)$

  (t_l = transaction latency, t_c = transaction commit time, t_r = transaction request time)
- t_c is computed as a follow on average: $t_c = t_r + RTT(1+q)(1+p)$

Notations	Description
m	Number of all nodes
n	Number of Byzantine nodes
p	Percentage of Byzantine nodes ($= n/m$)
q	Attack success rate
RTT	Round trip time for transaction request/response message

▪ Security Analysis (Attack Scenario 2)

- If the leader is a malicious follower, transaction data can be further abused to mount the other attacks such as blockchain folk

CONCLUSION AND DISCUSSION

- We represented client-side attacks exploiting diverse faults in raft-based blockchain systems
- The root cause for the vulnerability is the lack of authentication mechanism on the client side
- How to authenticate the raft message on the client side in an efficient manner is an important future work

REFERENCES

[1] D. Ongaro and J. Ousterhout, "In search of an understandable consensus algorithm," in 2014 USENIX Annual Technical Conference (Usenix ATC 14), 2014, pp. 305–319.
[2] L.-e. Wang, Y. Bai, Q. Jiang, V. C. Leung, W. Cai, and X. Li, "Beh-raftchain: A behavior-based fast blockchain protocol for complex networks,"
IEEE Transactions on Network Science and Engineering, vol. 8, no. 2, pp. 1154–1166, 2020.
[3] G. Zhang and H.-A. Jacobsen, "Prosecutor: An efficient bft consensus algorithm with behavior-aware penalization against byzantine attacks," in Proceedings of the 22nd International Middleware Conference, 2021, pp. 52–63.
[4] H. M. Buttar, W. Aman, M. Rahman, and Q. H. Abbasi, "Countering active attacks on raft-based iot blockchain networks," arXiv preprint
arXiv:2204.00838, 2022.
[5] W. Wang, S. Deng, J. Niu, M. K. Reiter, and Y. Zhang, "Engraft: Enclaveguarded raft on byzantine faulty nodes," in Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, 2022, pp. 2841–2855.

KOREA UNIVERSITY ISSLAB Information System Security Laboratory