**Introduction:**  
 The exploratory research for the Winter 2024 ISEC 885 course aims to develop a problem direction for a doctoral research idea concept paper, leading to an idea paper, with the ultimate objective of contributing to the field of Asynchronous Consensus and Permission-less Systems**.**

In the realm of network protocols and by abstraction that of consensus protocols, the debate between synchronous and asynchronous approaches have been that synchronous protocols operate under the assumption of predictable timing, while asynchronous protocols allow for greater flexibility by not requiring strict timing assumptions. More specifically asynchronous consensus ensures that systems can reach an agreement even when there is no bound on the time it takes for messages to be delivered. Not knowing the bound on the time in an asynchronous setting directly contrasts synchronous consensus, which does know its bound on time, along with partially-synchronous consensus, which knows there is a bound on time, but doesn't know what that bound is (Miller et al., 2016). This research favors the asynchronous setting because of its decentralization, instantaneous progress, and resilience to network variability, which are favorable traits in trustless blockchain systems.

Permissioned and permission-less blockchains represent two different paradigms in the realm of distributed ledgers. When it comes to the Blockchain Trilemma, the advantage of having a permissioned system is that they typically make gains in security and scalability while giving up decentralization. On the other hand, permission-less systems typically have a higher decentralization factor with weaker scalability and security (Woznica & Kedziora, 2022). This research favors permission-less over permissioned because of its higher decentralization in the blockchain dilemma.

When it comes to asynchronous consensus using permissioned vs permission-less ledgers, the majority of asynchronous consensus research has involved permissioned ledgers. This being that most initial research in the field of asynchronous consensus is implemented with trusted dealers for Distributed Key Generation (DKG) protocols (Abraham et al., 2021; Das et al., 2020). Having a trusted dealer for key generation, and by definition making the blockchain permissioned, has been seen throughout the conception of asynchronous consensus, such as in the initial works of the HoneyBadgerBFT (HBBFT), Beat, Dumbo, and the Asynchronous Byzantine Fault Tolerant (ABFT) protocols (Miller et al., 2016, Duan et al., 2018, Guo et al., 2020, and Knudsen et al., 2021). Although these protocols lay the foundation for the field of asynchronous consensus, they do not enable a fully permission-less system and even call for the advancement of the field of permission-less asynchronous consensus without trusted dealers (Knudsen et al., 2021).

The lack of asynchronous consensus without a a trusted dealer led to the conception of the Aleph protocol. The same way HBBFT touts itself as the first practical asynchronous consensus protocol (Miller et al., 2016). Aleph touts its self as the first practical asynchronous consensus protocol without a trusted dealer (Gągol et al., 2019). Aleph not only lacks a trusted dealer to make it permission-less, but also contributes to the field of asynchronous consensus by using a Direct Acyclic Graph (DAG) for consensus instead of sequential Asynchronous Common Subset (ACS), thus providing a theoretical alternative for implementing asynchronous atomic broadcast in lieu of sequentially executing ACS (Guo et al., 2022). Aleph provides a valid protocol in this research environment because it executes consensus asynchronously without giving up its permission-less properties.

**Problem:** Problems and potential solutions identified from the excerpt:

1. Forking Attack:Problem: Malicious nodes can exploit the protocol's reliance on local validation to create a large number of valid but conflicting forks, overwhelming honest nodes and disrupting the network (Section H). This attack is particularly concerning because:

* It is relatively "practical," requiring minimal resources and control by the attacker (Section H.1).
* It can be adapted to various DAG-based protocols with different parent requirements (Section H.2, H.3).

2. Scalability of Communication Complexity Problem: The excerpt mentions that the communication complexity of the reliable broadcast protocol used in Aleph can be improved from N^2 log(N) to N^2 (Section F). This signifies a potential scalability bottleneck as the network grows.

3. Lack of Proof of Termination:

Problem: While not explicitly stated as a problem, the excerpt mentions the possibility of modifying the reliable broadcast protocol to include a proof of termination (Section F). This suggests that the current protocol might lack a mechanism to definitively inform nodes when the protocol has finished executing for a specific unit.

It's important to note that these are just potential solutions, and further research and evaluation are necessary to determine their effectiveness, feasibility, and potential trade-offs in the context of specific DAG-based protocols.

**Fork** **Bomb** **Goal**:

* Non-local validation: Implement mechanisms that go beyond local checks and involve communication or information sharing between nodes to verify the validity of units, making it harder for attackers to create valid-looking forks. This could involve techniques like reliable broadcast or verifiable random functions (VRFs). (Note: The excerpt mentions Aleph's use of reliable broadcast as an example.)
* Threshold signatures: Utilize threshold signatures where a unit's validity requires a certain number of honest nodes to sign off on it, making it more difficult for attackers to forge valid forks without compromising a significant portion of the network. (The excerpt briefly mentions this concept.)

Scalability of Communication Complexity Goal:

* Employing RSA accumulators: The excerpt suggests replacing Merkle Trees with RSA accumulators in the reliable broadcast protocol to achieve better communication complexity. This could be further investigated and implemented to optimize network performance.

Lack of Proof of Termination Goal:

* Threshold signatures for termination: As mentioned in the excerpt, including a share of a threshold signature for the hash of the final message alongside the commit message could be explored. This would allow nodes to gather the required signatures and confirm the protocol's termination for a particular unit.
* Reputation systems: Implement mechanisms that track node behavior and penalize malicious actors, deterring them from launching fork attacks. However, designing effective reputation systems can be complex and requires careful consideration to avoid unintended consequences.

“We remark that N 2 log(N ) can be improved to N 2 in the communication complexity

bound by employing RSA accumulators instead of Merkle Trees in the RBC protocol.” aleph

Lacks the ability to provide proof

Additionally, one simple

modification can be done, which provides a proof that RBC has

terminated for each node which had received the output locally.

Namely, together with commit message, each node can send share

of a threshold signature of hash h with threshold 2f + 1. Then,

each node that gathered 2f + 1 is able to construct the threshold

signature and use it as a proof of the fact that RBC had finalized at

its end.

hence there is no way to roll it back. At this point we also remark

that a mechanism of banning nodes that are proved to be forking

(or malicious in some other way) is not sufficient to prevent this

attack.

We note that while the QuickAleph protocol (without the alert

system) satisfies the above condition, the Aleph protocol and the

QuickAleph extended with the alert system, do not.

Aleph uses reliable broadcast to disseminate units (which gives a

non-local validation mechanism) and similarly the alert system for

QuickAleph adds new non-local checks before adding units created

by forkers. On the other hand, protocols such as Hashgraph [4] or

Blockmania [23] satisfy this condition and thus are affected, if not

equipped with an appropriate defense mechanism.

the previous unit created by the same node (as in Hash-

graph) then one round of the attack can be realized in two

consecutive rounds.

***Not sure where this is going.***

***As this is presented it does not establish a clear and clean flow that builds a case for supporting a problem and need.***

***Aleph does not use a trusted dealer but is less secure. So it is not clear what the problem is as you have presented it other than to extend the Aleph work to address the need for non-trusted dealers in current permission-less blockchain environments and achieve a definable (from the literature ) level of security.***

Goal:  
 The goal of this exploratory research, conducted during the Winter 2024 ISEC 885 course, is to formulate a research idea paper focused on advancing the field of Asynchronous Consensus by developing a replicatable framework for deploying the Aleph protocol that is without a trusted dealer and is well suited for permission-less blockchains. This framework will establish a foundational benchmark for future studies, enabling researchers to compare and to build upon the original Aleph work effectively. While the existing Aleph paper offers comprehensive insights into the theoretical underpinnings of its consensus protocols, it lacks a practical framework that can be readily replicated for further investigation. Thus, the objective of this research is to furnish the Asynchronous Consensus community with a standardized test environment for Aleph, facilitating ongoing research and development endeavors.

**You need a goal / pair**

**You must establish a problem with existing frameworks to then focus on building one.**

References

Abraham, I., Jovanovic, P., Maller, M., Meiklejohn, S., Stern, G., & Tomescu, A. (2021). Reaching consensus for asynchronous distributed key generation. In Proceedings of the 2021 ACM Symposium on Principles of Distributed Computing (pp. 363–373).

Adam Gągol, Damian Leśniak, Damian Straszak, and Michał Świętek. 2019. Aleph:

Efficient Atomic Broadcast in Asynchronous Networks with Byzantine Nodes.

In Proceedings of the 1st ACM Conference on Advances in Financial Technologies.

214–228.

Baird, L., & Luykx, A. (2020). The Hashgraph Protocol: Efficient Asynchronous BFT for High-Throughput Distributed Ledgers. International Conference on Omni-layer Intelligent Systems, pp. 1-7. https://doi.org/10.1109/COINS49042.2020.9191430  
  
Boldyreva, A. (2002). Threshold signatures, multisignatures and blind signatures based on the gap-diffie-hellman-group signature scheme. In Public key cryptography–PKC 2003 (pp. 31–46). Springer.

Das, S., Xiang, Z., & Ren, L. (2020). Asynchronous data dissemination and its applications. In Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security.  
Duan, S., Reiter, M., & Zhang, H. (2018). Beat: Asynchronous BFT Made Practical. In ACM SIGSAC Conference on Computer and Communications Security, pp. 2028–2041. <https://doi.org/10.1145/3243734.3243812>

Fischer, M. J., Lynch, N. A., & Paterson, M. S. (1982). Impossibility of distributed consensus with one faulty process (Technical Report No. MIT/LCS/TR-728). Massachusetts Institute of Technology, Laboratory for Computer Science, Cambridge.

Gao, Y., Lu, Y., Lu, Z., Tang, Q., Xu, J., & Zhang, Z. (2022). Dumbo-NG: Fast Asynchronous BFT Consensus with Throughput-Oblivious Latency. In Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security (CCS '22) (p. 1187–1201). Association for Computing Machinery. https://doi.org/10.1145/3548606.3559379  
  
Gennaro, R., Jarecki, S., Krawczyk, H., & Rabin, T. (2003). Secure Applications of Pedersen’s Distributed Key Generation Protocol. In Topics in Cryptology - CT-RSA 2003, The Cryptographers’ Track at the RSA Conference 2003, San Francisco, CA, USA, April 13-17, 2003, Proceedings (pp. 373–390). <https://doi.org/10.1007/3-540-36563-X_26>

Djari, A., Anceaume, E., & Tucci-Piergiovanni, S. (2022). An extensive agent-based simulation study of sycomore++, a DAG-based permissionless ledger. In Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing (SAC '22) (pp. 334–336). Association for Computing Machinery. <https://doi-org.ezproxylocal.library.nova.edu/10.1145/3477314.3507245>

Gennaro, R., Jarecki, S., Krawczyk, H., & Rabin, T. (2007). Secure Distributed Key Generation for Discrete-Log Based Cryptosystems. Journal of Cryptology, 20(1), 51–83. https://doi.org/10.1007/s00145-006-0347-3  
  
Guo, B., Lu, Z., Tang, Q., Xu, J., & Zhang, Z. (2020). Dumbo: Faster Asynchronous BFT Protocols. In Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS '20) (pp. 803–818). Association for Computing Machinery. https://doi.org/10.1145/3372297.3417262

Guo, B., Lu, Y., Lu, Z., Tang, Q., Xu, J., & Zhang, Z. (2022). Speeding Dumbo: Pushing Asynchronous BFT Closer to Practice. doi:10.14722/ndss.2022.24385  
  
Hood, K., Oglio, J., Nesterenko, M., & Sharma, G. (2021). Partitionable Asynchronous Cryptocurrency Blockchain. IEEE International Conference on Blockchain and Cryptocurrency, pp. 1-9. https://doi.org/10.1109/ICBC51069.2021.9461080

Kogias, E. K., Malkhi, D., & Spiegelman, A. (2020). Asynchronous Distributed Key Generation for Computationally-Secure Randomness, Consensus, and Threshold Signatures. In Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS '20) (pp. 1751–1767). Association for Computing Machinery. <https://doi.org/10.1145/3372297.3423364>

Knudsen, H., Li, J., Notland, J., Haro, P., & Ræder, T. (2021). High-Performance Asynchronous Byzantine Fault Tolerance Consensus Protocol. IEEE International Conference on Blockchain, pp. 476-483. https://doi.org/10.1109/Blockchain53845.2021.00073  
  
Miller, A., Xia, Y., Croman, K., Shi, E., & Song, D. (2016). The Honey Badger of BFT Protocols. Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, pp. 31–42. https://doi.org/10.1145/2976749.2978399  
  
Ramakrishna Kotla, Lorenzo Alvisi, Michael Dahlin, Allen Clement, and Edmund L. Wong. (2009). Zyzzyva: Speculative Byzantine fault tolerance. ACM Transactions on Computer Systems (TOCS), 27(4), 7:1–7:39. <https://doi.org/10.1145/1658357.1658358>

Silva, P., Matos, M., & Barreto, J. (2023). NimbleChain: Speeding up Cryptocurrencies in General-purpose Permissionless Blockchains. Distributed Ledger Technology, 2(1), Article 8. <https://doi-org.ezproxylocal.library.nova.edu/10.1145/3573895>

Woznica, A., & Kedziora, M. (2022). Performance and scalability evaluation of a permissioned Blockchain based on the Hyperledger Fabric, Sawtooth and Iroha. Comput. Sci. Inf. Syst., 19, 659-678.  
  
Zhou, Q., Huang, H., Zheng, Z., & Bian, J. (2020). Solutions to Scalability of Blockchain: A Survey. IEEE Access. Advance online publication. https://doi.org/10.1109/ACCESS.2020.2967218