Reference of Terms

Mechanism Labs

mechanismlabs.io

Summer 2018

Glossary

Adaptive Adversary An adversarial model in which an adversary has the ability to control nodes and change which nodes they control to maximize their likelihood of impeding network function; that is, they adapt their corruptions as circumstances change [4].

Asynchrony In asynchrony, messages sent by parties may be arbitrarily delayed and no bound is assumed on the amount of time that it takes for the messages to be delivered. However, to show that the protocol terminates after some finite amount of time, all messages are often assumed to be delivered eventually after some bounded but unknown amount of time. Some asynchronous protocols make the extra assumption of some synchronization point in the course of the protocol run [4].

Byzantine Failure/Fault A failure model in the distributed systems literature by which a node can generate messages and change state arbitrarily, without necessarily following the rules specified by the protocol. This arbitrary behavior includes explicit attacks on the protocol. [3].

Erasure This is a cryptographic assumption that strengthens honest parties. In this model, there is a way for an honest party to untraceably erase memory contents [8]. This may mitigate the problem of weak subjectivity in Proof of Stake.

Full Synchrony This model assumes that there is a known upper bound on message delay and all messages are received in the exact linear ordering in which they were sent. This assumption is often considered to be unrealistic in practice since it assumes the existence of a global synchronization clock which is extremely hard (practically infeasible) to have in a distributed

system. Full synchrony allows algorithms to operate in rounds, since this upper bound on message transmission exists [4].

Mildly Adaptive Adversary It takes the adversary at least t rounds to corrupt an honest node, where t is referred to as the agility parameter. If $0 < t < \infty$ then the adversary is called mildly adaptive [6].

Network Partition In the distributed systems literature, this constitutes a period of time in which all messages passing links connecting a node or a super set of nodes and the rest of the the nodes have been severed. In the cryptographic literature, this constitutes a period of time when the adversary has complete control on message delivery and messages may be delayed arbitrarily long [1].

Notarization In Dfinity, this refers to threshold signature from majority of nodes under a block created jointly by registered clients [2].

Optimistic Mode These refer to ideal conditions specified for a protocol in which some special behavior is achieved. For example, this mode defined in Thunderella in which a super majority of the committee (3/4) are honest and the leader is honest. [6].

Partial Synchrony This model assumes the existence of an upper bound on messages transmission delays or the relative speed of process execution, but this upper bound is not known a priori to any nodes in the protocol. This model assumes that the messages sent are received by their recipients within some fixed time bound. In other words, while the messages may be delayed arbitrarily, they are guaranteed to be delivered within the time bound. [4].

Posterior Corruptions This refers to an event where the set of users possibly holding majority of stake sometime in past, would sell their stake at some point and from that point onwards, they might be incentivized to act maliciously (eg. fork and double spend old money) [6].

Predictable In this model, you may or may not have access to have input but you can find out output, More precisely, for all i no efficient algorithm can predict the $i + 1$ bit for non-negligible value more than half [5].

Random Oracle Model This is a security proof framework where one provides all parties, good and bad alike, with access to a (public) random oracle; prove correct a protocol in this model; then replace the random oracle by an object like a hash function [9].

Robust Committee Reconfiguration This means that committees in Thunderella are chosen such that each remains honest (not necessarily online) until the honest chains are roughly the clock time for the current txn $+4*$ security parameter $(c + 4k)$, since notarization transactions are only considered legitimate if included in the blockchain by length $(c+2k)$. k is the security parameter [6].

Semi Synchrony This is an assumption on the existence of an upper bound not known a priori, however the distribution, variance and other statistical information about the upper bound is known [2].

Static Adversary This implies that the adversary chooses which players to corrupt before protocol begins. (It first chooses all nodes, after which the randomness of nodes are "set" or perfectly predicted by the adversary). The adversary is restricted to choose its set of dishonest parties at the start of the protocol and cannot change this set later on [4].

Strongly Adaptive It takes the adversary t rounds to corrupt an honest node, where t is referred to as the agility parameter. If $t = 0$ then the adversary is called strongly adaptive [6].

Threshold Relay This is the technique that Dfinity uses to randomly sample nodes into groups, set the groups up for threshold operation, chooses the current committee, and relay from one committee to the next is called threshold relay. [2].

VRF A verifiable random function is a psuedo-random function where each output is unpredictable given the knowledge of all prior outputs. Each output has publicly verifiable proofs of output correctness. [7].

References

[1] H. Attiya and J. Welch. *Distributed Computing: Fundamentals, Simulations, and Advanced Topics.* Hoboken, NJ: Wiley, 2004.

[2] T. Hanke, M. Mohavedi, D. Williams, DFINITY Technology Overview Series: Consensus System, DFINITY Stiftung, 2017.

[3] N. Lynch, *Distributed Algorithms.* Butterworth-Heinemann, 1 Mar. 1996.

[4] M. Zamadi, M. Mohavedi, "Crypto Reference." url: mahdiz.com/ crypto/basics/

[5] D. Wu, "CS 255 (INTRODUCTION TO CRYPTOGRAPHY" url: https://crypto.stanford.edu/ dwu4/notes/CS255LectureNotes.pdf

[6] R. Pass and E. Shi, "Thunderella," Cornell: 2017.

[7] S. Micali, M. Rabin, S. Vadhan, *Verifiable random functions.* In 40th Annual Symposium on Foundations of Computer Science, pages 120–130, New York, NY, USA, October 17–19, 1999. IEEE Computer Society Press.

[8] T-H. Chan, R. Pass and E.Shi, "Communication-Efficient Byzantine Agreement without Erasures," 2018

[9] M. Bellare and P. Rogaway "Random Oracles are Practical: A Paradigm for Designing Efficient Protocols". ACM Conference on Computer and Communications Security: 62–73 (1993).