

Next-Generation Secure Distributed Computing

Aniket Kate^{*1}, Julian Loss^{*2}, and Kartik Nayak^{*3}

- 1 Purdue University – West Lafayette, US & Supra Research – West Lafayette, US. aniket@purdue.edu
- 2 CISPA – Saarbrücken, DE. loss@cispa.de
- 3 Duke University – Durham, US. kartik@cs.duke.edu

Abstract

The fast-evolving space of distributed systems spans various different areas such as applied cryptography, distributed computing, and game theory. The purpose of this seminar was to bring together researchers from these respective fields and to provide a unique opportunity of fostering interdisciplinary discussions on various related topics.

Seminar September 1–6, 2024 – <https://www.dagstuhl.de/24362>

2012 ACM Subject Classification Security and privacy → Cryptography; Computing methodologies → Distributed computing methodologies; Theory of computation → Algorithmic game theory

Keywords and phrases blockchain, cryptography, distributed computing, game theory

Digital Object Identifier 10.4230/DagRep.14.9.22

1 Executive Summary

Julian Loss (CISPA – Saarbrücken, DE)

License  Creative Commons BY 4.0 International license
© Julian Loss

This seminar was dedicated toward improving our understanding of techniques and models in the quickly evolving space of distributed systems, in particular as found in the blockchain space. This space intersects in various different areas such as applied cryptography, distributed algorithms, and game theory. As such, our seminar brought together researchers with a background in one or more of these areas to discuss with and learn from each other. Over the course of five days, the seminar participants gave talks about their respective areas followed by a discussion. The talks ranged from presenting recent results to summarizing the progress of a field over several decades.

While the talks took place in the morning sessions, the afternoon sessions were dedicated to breakout discussions on topics related to the talks and chosen by the seminar participants. During these sessions, topics were discussed in greater detail, which led to new insights and research questions. At the end of each seminar day, one of the discussion participants was tasked with summarising the outcome of the discussion to the other seminar participants. This often led to further discussion on some specific points.

Broadly speaking, the topics of these discussions can be summarized as follows.

Day 1: The discussions for day one were on topics related to the blockchain space. Three different groups discussed recent advancements in the area of consensus protocols, decentralization, as well as a discussion on rational actors storing secrets in the context of blockchain protocols.

* Editor / Organizer



Except where otherwise noted, content of this report is licensed under a Creative Commons BY 4.0 International license

Next-Generation Secure Distributed Computing, *Dagstuhl Reports*, Vol. 14, Issue 9, pp. 22–44

Editors: Aniket Kate, Julian Loss, and Kartik Nayak



DAGSTUHL
REPORTS Dagstuhl Reports

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Day 2: The second day of discussions focused on topics related to new/non-traditional network and fault models, gossip layers for spreading messages in blockchain systems, as well as further discussion on game theoretic aspects in distributed systems (again with a focus on blockchains).

Day 3: There were no afternoon discussions on day 3. Instead, the participants enjoyed an excursion to Saarschleife and a dinner at a local brewery in the evening.

Day 4: On day 4, one group discussed the difference in consistency guarantees provided by primitives such as modern blockchain protocols and classic primitives such as byzantine broadcast. Another group discussed the purpose of and possible applications of trusted execution environments for storing secrets in blockchain systems.

Day 5: The seminar ended by noon, hence there were no afternoon discussions on day 5.

2 Table of Contents

Executive Summary

<i>Julian Loss</i>	22
------------------------------	----

Overview of Talks


Blockchain Sharding <i>Georgia Avarikioti</i>	26
On Adaptive Security in Distributed Cryptographic Protocols <i>Renas Bacho</i>	26
A study of fault-tolerant distributed systems in practice <i>Adithya Bhat</i>	26
Overview of Verifiable Secret Sharing and Distributed Key Generation <i>Sourav Das</i>	27
Key Results and Challenges in Asynchronous Consensus <i>Sisi Duan</i>	27
Introduction to Laconic Cryptography <i>Nico Döttling</i>	28
Towards practical and efficient performance robustness: QuePaxa and beyond <i>Bryan Ford</i>	28
Blockchain-based Consensus <i>Juan A. Garay</i>	29
BeeGees: Stayin' Alive in Chained BFT <i>Neil Giridharan</i>	29
Lessons Learned and Key Challenges from PoS Protocol Design In Ouroboros <i>Aggelos Kiayias</i>	30
The Economic Limits of Permissionless Consensus <i>Andrew Lewis-Pye</i>	30
Network-Agnostic Security: Importance and Challenges <i>Chen-Da Liu-Zhang</i>	31
Federated Byzantine Agreement <i>Giuliano Losa</i>	32
Brief Survey of Research on Topology Hiding Computation <i>Tal Moran</i>	32
Secret re-sharing in one message in the plain channels model, for any threshold <i>Matthieu Rambaud</i>	32
Universal Abstract Model of Consensus <i>Mohammad Sadoghi Hamedani</i>	33
How should a blockchain keep a secret? <i>Alin Tomescu</i>	33
New MPC Paradigms: YOSO, Fluid and SCALES <i>Sophia Yakoubov</i>	34

Marrying TEEs with Secure Distributed Computing <i>Fan Zhang</i>	34
Working groups	
Nuances in Substituting bulletin boards and broadcast channels by blockchains <i>Aniket Kate, Georgia Avarikioti, Adithya Bhat, Sisi Duan, Julian Loss, Kartik Nayak, and Matthieu Rambaud</i>	35
How can game theory be used for distributed computing? <i>Aggelos Kiayias, Georgia Avarikioti, Nico Döttling, Sisi Duan, Juan A. Garay, Chen-Da Liu-Zhang, and Sophia Yakoubov</i>	36
How to make a blockchain system decentralized? <i>Aggelos Kiayias, Sourav Das, Sisi Duan, Bryan Ford, Andrew Lewis-Pye, Kartik Nayak, Sravya Yandamuri, and Fan Zhang</i>	37
Efficient P2P Gossip/ Data Dissemination <i>Giuliano Losa, Adithya Bhat, Aniket Kate, Tal Moran, and Filip Rezabek</i>	38
Considering non-traditional Network Settings and Adversarial Models <i>Julian Loss and Kartik Nayak</i>	40
Can blockchains keep secrets using game theory? <i>Tal Moran, Nico Döttling, Juan A. Garay, Aniket Kate, Aggelos Kiayias, Chen-Da Liu-Zhang, Kartik Nayak, Alin Tomescu, and Sophia Yakoubov</i>	41
Can blockchains keep secrets using TEE? <i>Alin Tomescu, Sourav Das, Sisi Duan, Giuliano Losa, Filip Rezabek, and Fan Zhang</i>	42
Participants	44

3 Overview of Talks

3.1 Blockchain Sharding


Georgia Avarikioti (TU Wien, AT)

License  Creative Commons BY 4.0 International license
© Georgia Avarikioti

Sharding distributed ledgers is a promising on-chain solution for scaling blockchains but most designs lack formal grounds. In this talk, I provided an overview of how sharding works, some evident challenges, and a formal model that expresses what sharding is. Next, I presented several limitations of sharding: this approach cannot achieve linear scaling, it is not secure under fully-adaptive adversaries, and it always demands periodic succinct proofs of the blockchain state (e.g., checkpointing). Then, I presented a protocol abstraction for secure and efficient sharding, identifying sufficient components. Lastly, I presented common mistakes in existing state-of-the-art protocols that lead to security and/or efficiency losses under our model and concluded with some key observations and open research questions.

3.2 On Adaptive Security in Distributed Cryptographic Protocols

Renas Bacho (CISPA – St. Ingbert, DE)

License  Creative Commons BY 4.0 International license
© Renas Bacho

Many distributed cryptographic protocols are analysed with respect to a static adversary that specifies the corruption set before the protocol execution. This state of affairs, however, is unsatisfactory, since in reality the adversary can corrupt parties at any point in time, which is covered by the notion of adaptive security. In this talk, we discuss the challenges arising in proving distributed cryptographic protocols, such as threshold signatures, adaptively secure and how we addressed these for some specific cases. Specifically, we revisit the adaptive security of the threshold BLS signature scheme that is widely used for distributed coin flipping.

3.3 A study of fault-tolerant distributed systems in practice

Adithya Bhat (Visa USA – Foster City, US)

License  Creative Commons BY 4.0 International license
© Adithya Bhat

In this talk, I talk about the gap between academic research on high-performance distributed systems and implementations by industry. I studied 4 open source implementations used by technology companies and open source community and presented 9 lessons that discusses design patterns, anti-patterns, optimizations and tips in implementing high-performance systems. The lessons are as follows:


1. Use real databases.
2. Write protocols in event-driven format.
3. Group related actions in contexts.

4. Boost performance with optimizations.
5. Write defensive code.
6. Effective usage of parallelism and concurrency.
7. Lower latency by analyzing critical path.
8. Write tests.
9. Synchronization problem and applications of clairvoyant caching

We discussed the 9 lessons with example code, references to real-world implementation, and present new techniques to write effective code.

3.4 Overview of Verifiable Secret Sharing and Distributed Key Generation

Sourav Das (University of Illinois – Urbana-Champaign, US)

License  Creative Commons BY 4.0 International license
© Sourav Das

My talk is divided into two parts: First, I will give an overview of the Verifiable Secret Sharing. In this overview, I will categorize 40+ years of research on three approaches based on the design technique. These approaches are (i) complaint-based, (ii) verifiable encryption-based, and (iii) share-resharing-based. I will then cover the core ideas of each of these approaches. I will then also present a new acknowledgment and reveal-based approach to VSS. I then conclude the first part of the talk, describing some open problems.

In the second part, I will talk about distributed key generation, covering the typical template of designing DKG protocols running n-parallel VSS scheme. I will cover the difficulties in naively adopting this approach in asynchronous networks due to FLP impossibility. I then summarize the state-of-the-art protocol for synchronous and asynchronous DKG protocols.

3.5 Key Results and Challenges in Asynchronous Consensus

Sisi Duan (Tsinghua University – Beijing, CN)

License  Creative Commons BY 4.0 International license
© Sisi Duan

Asynchronous Byzantine fault-tolerant (BFT) protocols are arguably the most appropriate solutions for building high-assurance and intrusion-tolerant systems in wide-area (WAN) environments, as these asynchronous protocols are inherently more robust against timing and denial-of-service (DoS) attacks that can be mounted over an unprotected network such as the Internet. In this talk, I reviewed the paradigms that can be used to build asynchronous BFT. Then I showed some recent research results on multiple asynchronous BFT primitives such as asynchronous common subset (ACS), direct acyclic graph (DAG) based protocols, and protocols that de-couple block transmission from consensus on the order. Finally, I discussed some research challenges and future works, such as scalability of the systems, verification of correctness, etc.

3.6 Introduction to Laconic Cryptography

Nico Döttling (CISPA – Saarbrücken, DE)

License © Creative Commons BY 4.0 International license
© Nico Döttling

Joint work of Nico Döttling, Sanjam Garg, Giulio Malavolta, Mohammad Hajiabadi, Pedro Branco, Chongwon Cho, Divya Gupta, Peihan Miao, Antigoni Polychroniadou

Main reference Chongwon Cho, Nico Döttling, Sanjam Garg, Divya Gupta, Peihan Miao, Antigoni Polychroniadou: “Laconic Oblivious Transfer and Its Applications”, in Proc. of the Advances in Cryptology – CRYPTO 2017 – 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part II, Lecture Notes in Computer Science, Vol. 10402, pp. 33–65, Springer, 2017.

URL https://doi.org/10.1007/978-3-319-63715-0_2

Communication efficiency is one of the central challenges for cryptography. Modern distributed computing techniques work on large quantities of data, and critically depend on keeping the amount of information exchanged between parties as low as possible. However, classical cryptographic protocols for secure distributed computation cause a prohibitive blow-up of communication in this setting. Laconic cryptography is an emerging paradigm in cryptography aiming to realize protocols for complex tasks with a minimal amount of interaction and a sub-linear overall communication complexity. This paradigm has led to the resolution of several long open problems in public key cryptography. In addition, techniques from laconic cryptography have been useful to build new notions such as registration-based encryption (RBE). In this talk, I will provide an introduction to the field and some of the main technical ideas used to construct laconic primitives.

3.7 Towards practical and efficient performance robustness: QuePaxa and beyond

Bryan Ford (EPFL Lausanne, CH)

License © Creative Commons BY 4.0 International license
© Bryan Ford

Leader-based consensus algorithms are fast and efficient under normal conditions, but lack robustness to adverse conditions due to their reliance on timeouts for liveness. We present QuePaxa, the first protocol offering state-of-the-art normal-case efficiency without depending on timeouts. QuePaxa uses a novel randomized asynchronous consensus core to tolerate adverse conditions such as denial-of-service (DoS) attacks, while a one-round-trip fast path preserves the normal-case efficiency of Multi-Paxos or Raft. By allowing simultaneous proposers without destructive interference, and using short hedging delays instead of conservative timeouts to limit redundant effort, QuePaxa permits rapid recovery after leader failure without risking costly view changes due to false timeouts. By treating leader choice and hedging delay as a multi-armed-bandit optimization, QuePaxa achieves responsiveness to prevalent conditions, and can choose the best leader even if the current one has not failed. Experiments with a prototype confirm that QuePaxa achieves normal-case LAN and WAN performance of 600k and 250k cmd/sec in throughput, respectively, comparable to Multi-Paxos. Under conditions such as DoS attacks, misconfigurations, or slow leaders that severely impact existing protocols, we find that QuePaxa remains live with median latency under 380ms in WAN experiments.

3.8 Blockchain-based Consensus

Juan A. Garay (Texas A&M University – College Station, US)

License © Creative Commons BY 4.0 International license
© Juan A. Garay

The advent of blockchain protocols such as Bitcoin has ignited much excitement, not only for their realization of novel financial instruments, but also for offering alternative solutions to classical problems in fault-tolerant distributed computing and cryptographic protocols, as such consensus (aka Byzantine agreement). Underlying many of such protocols is a primitive known as “proof of work” (PoW), which for over 20 years has been liberally applied in the cryptography and security literature to a variety of settings, including spam mitigation, sybil attacks, and denial of service protection; its role in the design of blockchain-based protocols, however, is arguably its most impactful application. At a high level, the way PoWs enable such protocols is by slowing message passing for all parties indiscriminately, thus generating opportunities for honest parties’ views to converge, under the assumption that their aggregate computational power exceeds that of the adversary.

In this talk we show how to achieve consensus among n parties, up to t of which may be malicious and act arbitrarily, on PoW-based blockchains in the presence of a higher number of corrupted parties ($t < n/2$) and weaker setup assumptions (i.e., a common reference string [CRS] – the “genesis” block), something known not to be achievable in the “classical” consensus literature. We resolve this apparent contradiction with a new paradigm we call “Resource-Restricted Cryptography,” which hints at “how to do cryptography when cryptography is not possible.”

Further, all previous PoW-based blockchain protocols rely on the “random oracle” (RO) methodology, which models the hash function as an idealized random function. We also take the first steps in getting rid of this assumption, by showing how to achieve consensus based on “protocol-friendly” PoWs built from conjectures in fine-grained complexity, such as the Orthogonal Vectors problem, in the CRS and random beacon model – and no random oracles.

3.9 BeeGees: Stayin’ Alive in Chained BFT

Neil Girdharan (University of California – Berkeley, US)

License © Creative Commons BY 4.0 International license
© Neil Girdharan

Modern chained Byzantine Fault Tolerant (BFT) systems leverage a combination of pipelining and leader rotation to obtain both efficiency and fairness. These protocols, however, require a sequence of three or four consecutive honest leaders to commit operations. Therefore, even simple leader failures such as crashes can weaken liveness, resulting in high commit latency or lack of commit all together. We show that, unfortunately, this vulnerability is inherent to all existing BFT protocols that rotate leaders with pipelined agreement. To resolve this liveness shortcoming we present BeeGees, a novel chained BFT protocol that successfully commits blocks even with non-consecutive honest leaders. It does this while also maintaining quadratic word complexity with threshold signatures, linear word complexity with SNARKs, and responsiveness between consecutive honest leaders. BeeGees reduces the expected commit latency of HotStuff by a factor of three under failures, and the worst-case latency by a factor of seven.

3.10 Lessons Learned and Key Challenges from PoS Protocol Design In Ouroboros


Aggelos Kiayias (University of Edinburgh, GB)

License  Creative Commons BY 4.0 International license
© Aggelos Kiayias

This talk presents definitions and an overview of large scale ledger consensus as well as the threat model and setting suggested by the Bitcoin blockchain that includes dynamic availability and self healing from adversarial spikes. It introduces the “software only launch” problem for a bottom up launch of an IT system as a motivating use-case for the Bitcoin approach and threat model. It also motivates proof of stake (PoS) systems with the related “key airdrop launch” problem. It continues with an overview of the development of the Ouroboros protocol and its security argument. The discussion then moves to the benefits that the PoS approach brings to ledger consensus design allowing for a “best of both worlds” approach to be adopted. This approach has the system launching in a dynamic availability mode incentivizing participation and when the participation manages to exceed a threshold, high performance BFT style protocol operation can give further benefits such as such partition tolerance and responsiveness. The talk concludes with research directions, challenges and lessons learned.

3.11 The Economic Limits of Permissionless Consensus

Andrew Lewis-Pye (London School of Economics & Political Science, GB)

License  Creative Commons BY 4.0 International license
© Andrew Lewis-Pye

The purpose of a consensus protocol is to keep a distributed network of nodes “in sync,” even in the presence of an unpredictable communication network and adversarial behavior by some of the participating nodes. In the permissionless setting relevant to modern blockchain protocols, these nodes may be operated by a large number of unknown players, with each player free to use multiple identifiers and to start or stop running the protocol at any time. Establishing that a permissionless consensus protocol is “secure” thus requires both a distributed computing argument (that the protocol guarantees consistency and liveness unless the fraction of adversarial participation is sufficiently large) and an economic argument (that carrying out an attack would be prohibitively expensive for a potential attacker). There is a mature toolbox for assembling arguments of the former type; the goal of this paper is to lay the foundations for arguments of the latter type. For example, the Ethereum protocol is oft-claimed to be “more economically secure” after “the merge,” meaning in its current proof-of-stake incarnation relative to the (proof-of-work) original. What, formally, does this assertion mean? Is it true? Could there be alternative protocols that are “still more economically secure” than Ethereum? How do the answers depend on the assumptions imposed on, for example, the reliability of message delivery or the active participation of non-malicious players?

An ideal permissionless consensus protocol would, in addition to satisfying standard consistency and liveness guarantees, render consistency violations prohibitively expensive for the attacker without collateral damage to honest participants – for example, by programatically confiscating an attacker’s resources without reducing the value of honest participants’

resources, as is the intention for slashing in a proof-of-stake protocol. We make this idea precise with our notion of the EAAC (expensive to attack in the absence of collapse) property, and prove the following results:

- In the synchronous and dynamically available setting (in which the communication network is reliable but non-malicious players may be periodically inactive), with an adversary that controls at least one-half of the overall resources, no protocol can be EAAC. In particular, this result rules out EAAC for all typical longest-chain protocols (be they proof-of-work or proof-of-stake).
- In the partially synchronous and quasi-permissionless setting (in which resource-controlling non-malicious players are always active but the communication network may suffer periods of unreliability), with an adversary that controls at least one-third of the overall resources, no protocol can be EAAC. In particular, slashing in a proof-of-stake protocol cannot achieve its intended purpose if message delays cannot be bounded a priori.
- In the synchronous and quasi-permissionless setting, there is a proof-of-stake protocol with slashing that, provided the adversary controls less than two-thirds of the overall stake, satisfies the EAAC property.

Thus, while only “classical security” is possible in the dynamically available or partially synchronous settings, proof-of-stake protocols with slashing can obtain additional “economic security” in the quasi-permissionless and synchronous setting. All three results are optimal with respect to the size of the adversary.

With respect to Ethereum, our work formalizes the potential security benefits of proof-of-stake sybil-resistance coupled with slashing and the common belief that the merge has increased Ethereum’s economic security. Our work also provides mathematical justifications for several key design decisions behind the post-merge Ethereum protocol, ranging from long cooldown periods for unstaking to economic penalties for inactivity.

3.12 Network-Agnostic Security: Importance and Challenges

Chen-Da Liu-Zhang (Hochschule Luzern – Rotkreuz, CH & Web3 Foundation – Zug, CH)

License  Creative Commons BY 4.0 International license
© Chen-Da Liu-Zhang

In this talk we discussed about network-agnostic security, including an overview of the literature and recent results and techniques. This is a security model where a single protocol provides 1) high node resilience when the network is synchronous and 2) some node resilience when the network is asynchronous.

This is an interesting theoretical notion which could also be important in practice. The idea here is that if there is a small chance that a synchronous network becomes asynchronous (meaning that some messages are in fact not delivered in time within the assumed delay bound), and independently there is also a small chance that there is a high corruption, then the protocol remains secure as long as it is not the case that both bad events occur at the same time. This approach can in principle give a much higher security in practice than a standard protocol which assumes that neither of the events happen.

3.13 Federated Byzantine Agreement


Giuliano Losa (Stellar Development Foundation – San Francisco, US)

License  Creative Commons BY 4.0 International license
© Giuliano Losa

Federated Byzantine Agreement allows creating a quorum system in a permissionless environment, without resorting to proof-of-work or proof-of-stake. In this talk, we review the Federated Byzantine Agreement model and we discuss how to solve the consensus problem in this model.

3.14 Brief Survey of Research on Topology Hiding Computation

Tal Moran (Reichman University – Herzliya, IL)


License  Creative Commons BY 4.0 International license
© Tal Moran

A distributed computation in which nodes are connected by a partial communication graph is called topology-hiding if it does not reveal information about the graph beyond what is revealed by the output of the function.

In this talk, I give a brief survey of the research landscape on topology-hiding computation (THC), and give some examples of both positive and negative results. Finally, I discuss some of the open questions in this area.

3.15 Secret re-sharing in one message in the plain channels model, for any threshold

Matthieu Rambaud (Télécom Paris, FR)

License  Creative Commons BY 4.0 International license
© Matthieu Rambaud

Dynamic re-sharing protocols enable an n -sized committee of parties, holding shares of a secret, to distribute new secret-shares of the same secret to a new n' -sized committee. It should guarantee secrecy beyond t and t' corruptions in both committees, where t and t' are parameters called “thresholds”, and termination, a.k.a. “robustness,” as soon as $t + 1$ and $t' + 1$ parties are honest in both committees. Resharings are claimed to be executed by Fireblocks and Coinbase to secure the storage of threshold signing keys: every minutes-long intervals or after every signing. In such applications, known as “proactive refresh,” it is mostly the case that $n = n'$ and $t = t'$ and that an old party is run by the same machine as a new party. If such a machine is corrupt during the resharing, then it counts in both the corruption budget (t for the old committee, t' for the new one). Hence corruptions count twice during the resharing, thus the system is particularly vulnerable during this time frame.

We present “HRL”: the first resharing that operates in the plain synchronous channels model and allows any threshold. As a bonus, it is truly “YOSO”, i.e., parties do speak once (no hidden Byzantine broadcast (BC) implementation, nor need to talk to a trusted PKI ahead of the execution). The format of the secret sharing allows non-interactive evaluations of linear maps, hence, allows non-interactive threshold BLS or Schnorr signing.

The techniques of previous works could not reach these properties, due to a structural impossibility. Namely, all previous resharings allowing $n/3 \leq t \leq 2n/3$ require a bulletin board PKI setup, which is impossible to implement in the plain channels model. In more details, a PKI was needed to implement a common coin (CC) and/or a BC. A PKI was a fortiori required in public-key encryption-based resharings (Eurocrypt'21, Dfinity's 2021, Usenix'23). However, a PKI has been known for 45 years to be unimplementable in the plain channels model. Worse, these black-box tools required by previous resharings (publication on a PKI and BC/consensus) yielded a resharing time much longer than HRL, which takes only one message delay.

The paper and slides will be made publicly available in October.

3.16 Universal Abstract Model of Consensus

Mohammad Sadoghi Hamedani (University of California – Davis, US)

License © Creative Commons BY 4.0 International license
© Mohammad Sadoghi Hamedani

The inception of Bitcoin and blockchain has renewed the vision of a democratic and decentralized computational paradigm, that is, to ingrain integrity, transparency, and accountability into the very fabric of the computational model. These fundamental concepts and the technologies behind them—a generic ledger-based data model, cryptographically ensured data integrity, and transparent and accountable consensus-based replication—prove to be a powerful and inspiring combination. Arguably, the resilient consensus protocol is at the heart of this paradigm shift. In this presentation, we aim to provide an insightful overview of the core structure of the consensus protocols. We will further offer the theoretical intuitions behind our ongoing work, including the speculative consensus model, concurrent consensus with a wait-free property, geo-scale meta-consensus, consensus with weaker consistency models and isolation semantics, as well as a variety of sharding and cross-chain protocols through our novel reliable communication primitives. Furthermore, we provide practical realization of ongoing work in building a consensus-based blockchain called Apache ResilientDB (Incubating).

3.17 How should a blockchain keep a secret?

Alin Tomescu (Aptos Labs – Palo Alto, US)

License © Creative Commons BY 4.0 International license
© Alin Tomescu

In this talk, we survey results & challenges around generating, maintaining & using a shared secret amongst the validators of a proof-of-stake (PoS) blockchain.

In the first part, we discuss techniques for generating a secret. We start with secret sharing in the threshold setting and then in the weighted setting that arises in PoS blockchains. We then introduce publicly-verifiable secret sharing (PVSS), explaining why it could be an ideal primitive to build distributed key generation (DKG) protocols from. Lastly, we discuss the new “silent setup” setting (previously known as “ad hoc groups” in the literature) for bootstrapping threshold cryptosystems without a DKG or any explicit secret sharing.


In the second part, we discuss the threat of collusion attacks in the PoS attacks, where validators stand to profit by exposing the shared secret (or a function of it; e.g., a VRF). We present three different collusion attacks, which are all detectable-yet-unpunishable. We then give a TEE-based approach that could prevent collusion.

In the third part, we discuss some new techniques used to speed up threshold cryptosystems. We begin by reminding practitioners that Lagrange interpolation in threshold cryptosystems can and should be done via an optimized quasilinear time algorithm, instead of quadratic. Then, we present new results on threshold cryptosystems that use group elements as secret key. Lastly, we present an exciting new direction on batching threshold cryptosystems so that communication during aggregation is independent of the batch size.

Overall, we highlight important research problems in both the theory and the practice of threshold cryptography.

3.18 New MPC Paradigms: YOSO, Fluid and SCALES


Sophia Yakoubov (Aarhus University, DK)

License  Creative Commons BY 4.0 International license
© Sophia Yakoubov

There are compelling motivations for using MPC protocols where computing nodes need not speak more than once: (1) nodes might not be able to commit their resources for long, and (2) it is desirable to maintain communication efficiency in large-scale computations in the presence of an adaptive adversary. This talk surveys recent results and open problems in this space.

3.19 Marrying TEEs with Secure Distributed Computing

Fan Zhang (Yale University – New Haven, US)

License  Creative Commons BY 4.0 International license
© Fan Zhang

Main reference Michael Rosenberg, Maurice Shih, Zhenyu Zhao, Rui Wang, Ian Miers, Fan Zhang: “ZIPNet: Low-bandwidth anonymous broadcast from (dis)Trusted Execution Environments”, 2024.

URL <https://eprint.iacr.org/2024/1227>

The first part of this talk surveys recent novel applications of Trusted execution environments (TEEs) in the context of secure distributed computing and applications. However, all known TEEs have been broken at some point. How can we rely on TEEs if they will inevitably be broken? The second part presents recent works on tolerating TEE failures in-protocol.

4 Working groups

4.1 Nuances in Substituting bulletin boards and broadcast channels by blockchains

Aniket Kate (Purdue University – West Lafayette, US & Supra Research – West Lafayette, US), Georgia Avarikioti (TU Wien, AT), Adithya Bhat (Visa USA – Foster City, US), Sisi Duan (Tsinghua University – Beijing, CN), Julian Loss (CISPA – Saarbrücken, DE), Kartik Nayak (Duke University – Durham, US), and Matthieu Rambaud (Télécom Paris, FR)

License © Creative Commons BY 4.0 International license
© Aniket Kate, Georgia Avarikioti, Adithya Bhat, Sisi Duan, Julian Loss, Kartik Nayak, and Matthieu Rambaud

The group discusses the relationship between broadcast channels and blockchains implemented as the atomic broadcast primitive.

A broadcast channel (BC) is a primitive extensively employed in the cryptographic literature that assumes that a protocol proceeds in rounds with known time bounds and that every message broadcasted by an honest party in a round reaches all designated receivers within the same round. Moreover, if receivers expect a message from a particular sender in some round and do not receive the message, then the receiver can count that sender to be adversarial.

While some efforts in the literature suggest that we can realize a broadcast channel (or closely related primitives) using a blockchain, the group discussed that it can be far from reality. Indeed, almost all blockchains in practice will fail to realize a broadcast channel: Reasons for this range from the Internet behaving non-synchronously at times to blockchain getting overburdened with messages to censorship and front-running attacks. The Blockchain literature, as a result, often models blockchains as state machine replications (SMR) or atomic broadcasts (ABC) that ensure eventual liveness/termination and prefix ordering of messages. The group extensively discussed the effect of replacing the BC assumption with an SMR/ABC as well as the overall what can we achieve using SMR/ABC in a black-box fashion.

For example, the group observes that the non-interactive primitives such as publicly verifiable secret sharing as well as distributed key generation mechanisms and multi-party computation (MPC) based on that may function similarly when we replace a BC with an SMR/ABC; however, that is not true for many broadcast-based MPC primitive. Moreover, the group explores lower-bound results in the SMR/ABC-assisted models. For example, Byzantine agreement with strong unanimity cannot be implemented for an honest majority $t < n/2$ of players assuming blockchain access to SMR/ABC, even when players have furthermore access to asynchronous P2P channels. On the other hand, asynchronous primitives such as Agreement on a common subset of nodes come free and can be implemented for any $t < n$.

4.2 How can game theory be used for distributed computing?

Aggelos Kiayias (University of Edinburgh, GB), Georgia Avarikioti (TU Wien, AT), Nico Döttling (CISPA – Saarbrücken, DE), Sisi Duan (Tsinghua University – Beijing, CN), Juan A. Garay (Texas A&M University – College Station, US), Chen-Da Liu-Zhang (Hochschule Luzern – Rotkreuz, CH & Web3 Foundation – Zug, CH), and Sophia Yakoubov (Aarhus University, DK)

License © Creative Commons BY 4.0 International license
 © Aggelos Kiayias, Georgia Avarikioti, Nico Döttling, Sisi Duan, Juan A. Garay, Chen-Da Liu-Zhang, and Sophia Yakoubov

The group discusses the topic of how game theory can be used for distributed computing. The discussion involves two parts. In the first half, the group discusses some basic concepts about game theory and the two typical approaches to modeling the blockchain protocol (using Bitcoin as a particular example) using game theory. In the second half, the group discusses one of the approaches and in particular discusses rational protocol design (RPD). It was concluded that using RPD can benefit the modeling of rational behavior of blockchain consensus. Further investigation and discussion might be needed to learn what results may be impossible in the Byzantine/cryptographic settings but possible in a game theory setting.

In the first half, the group discusses basic concepts of game theory such as model, solution concept, and mechanism design. Towards the discussion on what things are impossible in the Byzantine/crypto settings but possible in the game theory setting and what things are impossible in the game theory settings but impossible in the Byzantine/crypto settings, the group uses Bitcoin as an example to think about the problem. In particular, consider a set of n participants adopting the protocol strategies $\langle \Pi_1, \Pi_2, \dots, \Pi_n \rangle$ and for each one some associated utility u_i . Applying the strategies and allowing the system to operate over time results in an outcome for each player that can have a different utility. As a result, the utility function can be thought of as a mapping $\langle \Pi_{Btc}, \Pi_{Btc}, \dots, \Pi_{Btc} \rangle \rightarrow R$, the set of real numbers. For instance the sequence of strategies $\langle \Pi_{Btc}, \Pi_{Btc}, \dots, \Pi_{Btc} \rangle$ is when all players decide to follow the Bitcoin protocol. We can then assess whether this strategy profile is a Nash equilibrium. The challenge of standard Nash equilibrium is that it does not consider collusions so some other solution concepts can be more relevant than consider coalitions. In all cases, of utmost importance is how to define the utility function.

The group then discusses some applications such as fair exchange and multiparty computation (MPC) and uses these applications to think about how to model the utility. In general, there are two approaches: 1) using an external party (trusted party or blockchain) to define utilities; 2) Define utilities internally from parties inside the system. The conclusion was that the set of strategies in the typical modeling of internal parties (aka type 2) might sometimes be overly simplified.

In the second part, the group discusses Rational Protocol Design (RPD) [FOCS 2013] and uses the Eurocrypt'18 paper by Garay et al. as an example to discuss type 1 of the approach. The original RPD model considers game theory for MPC in general and considers a metagame between a ‘protocol designer D’ and ‘adversary A’. Motivated by RPD, the Eurocrypt'18 paper uses the model to consider selfish mining in Bitcoin. The model considers that the protocol (i.e., PoW-based consensus) is fixed and that violations of security notions (i.e., safety and liveness) are formally captured in a suitably modified ledger ideal functionality. The adversary may control a mining pool with arbitrary size at the beginning of the execution. Using a simulation argument, it was demonstrated that Bitcoin achieves an equilibrium such

that rational adversaries are not willing to launch selfish mining if $n > t$ or any other attack for that matter. It was agreed that the approach might be useful for other problems and settings as well.

After the summary of the discussion, Ittai also pointed out that secure multiparty computation with covert adversaries is another related direction worth considering in this context.

4.3 How to make a blockchain system decentralized?

Aggelos Kiayias (University of Edinburgh, GB), Sourav Das (University of Illinois – Urbana-Champaign, US), Sisi Duan (Tsinghua University – Beijing, CN), Bryan Ford (EPFL Lausanne, CH), Andrew Lewis-Pye (London School of Economics & Political Science, GB), Kartik Nayak (Duke University – Durham, US), Sravya Yandamuri (Duke University – Durham, US), and Fan Zhang (Yale University – New Haven, US)

License © Creative Commons BY 4.0 International license

© Aggelos Kiayias, Sourav Das, Sisi Duan, Bryan Ford, Andrew Lewis-Pye, Kartik Nayak, Sravya Yandamuri, and Fan Zhang

The group discusses the topic of how to make a blockchain system decentralized. The discussion began with a discussion on what decentralization means and then briefly touched on the topic of how to detect the tendency of a system to become less decentralized.

To begin with the definition of decentralization, a few considerations are made. First, we need to consider a relative permissionless model for the definition of decentralization. In the notion used by Budish, Lewis-Pye, and Roughgarden [EC 2024] during the presentation of the day, the models include fully permissionless; dynamically available settings (e.g., longest chain); quasi-permissionless setting (Algorand, system with a set of IDs), but not the permissioned setting. Second, we need to take a layered view of the blockchain to fully consider the problem (e.g., application layer, relay layer). The group discusses different layers to consider in the big picture as well as the Edinburgh Decentralization Index (EDI) that aims to measure the decentralization of various blockchain systems. The different layers the paper considers include Governance, client, tokenomics, consensus, network, software, and hardware.

In terms of defining the decentralization of the system, the group converges on the idea of identifying resources of interest in different aspects or layers of the system and then examining the way that such resources are distributed to various entities and stakeholders. In order to measure the way the resources are distributed between participants metrics such as the Gini coefficient or the Nakamoto coefficient can be used to model decentralization.


In terms of combining information from different layers, the group considered a model that includes an attacker with certain objectives and a budget and/or resources that is willing to invest in order to reach her objectives. The resources can be monetary but might not be limited to money, e.g., they can involve insiders or computational power. The objectives we considered include goals such as censorship of transactions, safety violations in the settlement of transactions, and manipulating the value of the system (e.g., the value of the digital asset that is maintained and used by the system).

In terms of detecting the tendency of less decentralization, there was no clear conclusion. But there might be multiple approaches to quantify it. The rich-get-richer problem was discussed and identified as worthy of further research. Proofs of personhood were mentioned as potential building blocks for enabling system policies with increased egalitarianism and engagement.

Transparency of ownership was also identified as a possibly weaker goal but related to decentralization. It was remarked that a system without transparency of ownership might still be decentralized. A system with transparency of ownership allows users to engage with the system in a more informed fashion without necessarily requiring a high degree of decentralization. It was discussed that transparency of ownership would be hard to achieve in existing systems and similar tools such as those used to measure decentralization can be applied.

4.4 Efficient P2P Gossip/ Data Dissemination

Giuliano Losa (Stellar Development Foundation – San Francisco, US), Adithya Bhat (Visa USA – Foster City, US), Aniket Kate (Purdue University – West Lafayette, US & Supra Research – West Lafayette, US), Tal Moran (Reichman University – Herzliya, IL), and Filip Rezabek (TU München – Garching, DE)

License  Creative Commons BY 4.0 International license
© Giuliano Losa, Adithya Bhat, Aniket Kate, Tal Moran, and Filip Rezabek

The participants discussed how to efficiently disseminate data in blockchain systems, possibly using peer-to-peer gossip techniques.

Most blockchain systems need to disseminate at least two types of data items: lists of transactions called blocks, which are usually of size in the order of a megabyte, and consensus messages, which are usually much smaller (maybe a few kilobytes). Consensus messages are often produced in an all-to-all fashion, also called parallel broadcast (PBC), for example, when every participant votes and we must disseminate the votes of well-behaved participants to all well-behaved participants. In DAG-based protocols, block dissemination is also an all-to-all problem. However, in leader-based protocols, blocks may originate from a unique block producer at a time. In both cases, data items must be disseminated reliably despite the malicious behavior, also called Byzantine failure, of some of the participants.

Malicious participants may try to perform so-called eclipse attacks, in which one or more well-behaved participants become isolated from the rest of the network and do not receive the data items broadcast by other participants or send their own data items to other participants. In most cases, this not only poses a liveness problem but also a safety problem. For example, in longest-chain systems, an isolated participant may not receive the longest chain in the system and instead follow a shorter attacker chain. In proof-of-stake systems, isolating a participant may lead to a stake bleeding attack in which isolated, well-behaved participants gradually lose stake due to inactivity penalties imposed by the system, until the attacker is able to secure a super-majority stake that allows a successful double-spend attack.

Malicious participants may also try to perform denial-of-service attacks by taking advantage of the fact that many dissemination protocols amplify the messages to be broadcast, or they may try to censor particular blocks that contain transactions that they do not like.

We can initially categorize existing work in the area along two dimensions:

- Whether the set of participants in the system is known by all participants, e.g. as is the case in most proof-of-stake systems, or whether participants have only limited knowledge of other participants.
- When participants know each other, whether it is feasible or desirable for a participant to communicate directly with all others, or whether communication with a small neighborhood is preferred.

When all participants are known, a fixed fraction (or a weighted fraction) is assumed malicious, and data items are large, we can use erasure coding to obtain efficient dissemination protocols in which, to broadcast one data item of size S , each participant sends data directly to all participants but, in total, only sends a number of bits linear in S (regardless of the size of the system, as long as the data item is big enough). Efficient use of this building block in consensus protocols often requires verifiable information dispersal (VID), as introduced by (Cachin and Tessaro 2005). Recent examples of consensus protocols making use of VID include (Shoup 2023; Yang et al. 2021).

When sending to all directly is undesirable, knowledge of the set of participants allows the creation of Erdős–Rényi graphs, or other exponential-expansion graphs, of logarithmic degree. (Liu-Zhang et al. 2022a; Liu-Zhang, Matt, and Thomsen 2024) apply this idea to the weighted setting. An additional challenge is to tolerate adaptive corruption when the corruption budget of the adversary is more than logarithmic. (Matt, Nielsen, and Thomsen 2022) solve this problem under delayed adaptive corruptions (which they also formalize), while (Tsimos, Loss, and Papamanthou 2022) exploit the all-to-all nature of parallel broadcast to hide topology information from an adaptive adversary while keeping message complexity low.

When participants only have a partial view of other participants, we can resort to peer-sampling protocols in order to create a connected overlay graph over the system (which can then be used by higher-level data-dissemination protocols). Early works in the Byzantine setting include (Bortnikov et al. 2008) (Brahms) and (Jesi, Montresor, and van Steen 2010), and more recent works include Basalt (Auvolat et al. 2023). Roughly speaking, in those works each participant maintains a set of known peers (its view) and periodically exchanges its view with other participants to produce a stream of peers that must uniformly sample the system. Assuming an initially connected view-graph and a limit on the rate at which a malicious participant can communicate with other participants, Brahms guarantees connectivity and convergence to a uniform sample using views of size cubic root of n (where n is the number of participants). The assumption that malicious participants are rate-limited implies a way to prevent Sybil attacks. This may be realized using some form of proof-of-work, or, as in (Pilet, Frey, and Taïani 2020), using network addresses as a Sybil-resilient resource.

Finally, (Cohen, Loss, and Moran 2023) assume that views (which they call local neighbor sets) are given and propose efficient BA protocols based on gossip over the view graph. In particular, their protocol defends against attacks in which malicious participants try to flood the system with bogus traffic thanks to the amplifying effect of gossip.

It would be useful to assemble known lower and upper bounds (in communication complexity, number of rounds, etc.) in the various settings that we described above, but the discussion group did not have time to go into the details.

Avenues for further work include considering rational participants who seek to maximize some utility function (as in (Li et al. 2006; Mao et al. 2020)), where tit-for-tat or market-based approaches might be useful.

4.5 Considering non-traditional Network Settings and Adversarial Models

Julian Loss (CISPA – Saarbrücken, DE) and Kartik Nayak (Duke University – Durham, US)

License  Creative Commons BY 4.0 International license
© Julian Loss and Kartik Nayak

This report will focus on the exposed parties model, mixed fault models, network agnostic models, and beyond.

The motivation for this discussion was to develop a better understanding of various network and fault models that are used in the design of distributed protocols, e.g., for multi-party computation (MPC) or consensus.

We focused on six different models:

- The classical synchronous model. Here, parties have shared clocks and know an upper bound on the time it takes to deliver a message between any two parties. In this model, it is possible to design protocols that withstand a large number of faulty parties that try to derail the protocol.
- The classical asynchronous model. In this model, parties have no notion of common time and there are no bounds on the delivery times of messages. Protocols for MPC and consensus in this model inherently tolerate a smaller number of faults than their synchronous counterparts and require randomization. On the other hand, the asynchronous model is a far more realistic approximation of large-scale networks like the Internet.
- The network agnostic model. This is a more recent model that has recently drawn a lot of interest from the community. In the network agnostic model, one does not fix either of the above network types. Instead, the goal is to design algorithms and protocols that achieve the best possible fault tolerance depending on the type of network they are run in. In this manner, a network-agnostic algorithm can tolerate more faults than a purely asynchronous one when the network happens to be synchronous, but will also retain some fault tolerance in asynchronous networks (this is typically not the case for purely synchronous algorithms). It can be shown that there is a trade-off between the number of faults that such an algorithm can tolerate in an asynchronous network and a synchronous network. This allows a protocol designer the flexibility to choose the fault thresholds for each type of network so as to minimize the probability of failure with respect to certain events.
- The mixed fault model. The models discussed so far treat every party as fully malicious (a.k.a. byzantine) meaning that no guarantee on their behavior is assumed. While this allows us to make strong security guarantees, it is often excessively pessimistic for settings where not all faults display worst-case behavior. The mixed fault model therefore takes a more tailored approach in which different types of faults are carefully distinguished according to their exact behaviour. In this manner, it is possible to obtain protocols that tolerate a larger number of faulty parties than would otherwise be possible. Examples of benign failures are parties that run the protocol code honestly, but may crash or for which certain protocol messages are lost in transit.
- The exposed parties model. This relatively unexplored model is a consequence of using cryptography in the design of distributed protocols. If parties locally store key material to perform cryptographic tasks such as signing or encrypting messages, it is possible for these keys to become exposed to the adversary over the course of the protocol execution. Once this happens, an adversary can perform these operations on behalf of the parties whose keys have been exposed. However, it may still be meaningful to treat such parties

as otherwise remaining honest in the protocol and ask whether one can guarantee that they obtain the proper output at the end of the protocol execution. In this manner, it once again becomes possible to tolerate a higher number of faults that would usually be treated as fully malicious.

Our discussion began with getting everyone on the same page on these models. We then discussed their implications and applicability to practical scenarios such as, e.g., blockchain protocols. During the last part of the discussion, we thought about ways to extend or reinterpret some of these models. Two of the most interesting discussion points here were about the exposed parties model and the mixed faults model.

- For the exposed parties model, we discussed why it would make sense that parties' whose keys have become known to the adversary could still authenticate themselves to each other. (This assumption is necessary in order to achieve any meaningful statements). We concluded that this indeed is well-motivated by practical scenarios, where signing keys from protocols are usually separate (and stored separately) from long-term authentication keys such as, e.g., TLS keys.
- For the mixed fault model, we determined that it could be interpreted as a version of the synchronous model where the synchrony guarantees are relaxed. Namely, in the synchronous model, any party who cannot send or receive messages within the synchrony bounds of the model is treated as fully malicious. By modeling such parties, instead, as benign faults as discussed above, it is possible to obtain protocols with a higher resilience.

4.6 Can blockchains keep secrets using game theory?

Tal Moran (Reichman University – Herzliya, IL), Nico Döttling (CISPA – Saarbrücken, DE), Juan A. Garay (Texas A&M University – College Station, US), Aniket Kate (Purdue University – West Lafayette, US & Supra Research – West Lafayette, US), Aggelos Kiayias (University of Edinburgh, GB), Chen-Da Liu-Zhang (Hochschule Luzern – Rotkreuz, CH & Web3 Foundation – Zug, CH), Kartik Nayak (Duke University – Durham, US), Alin Tomescu (Aptos Labs – Palo Alto, US), and Sophia Yakoubov (Aarhus University, DK)

License © Creative Commons BY 4.0 International license
 © Tal Moran, Nico Döttling, Juan A. Garay, Aniket Kate, Aggelos Kiayias, Chen-Da Liu-Zhang, Kartik Nayak, Alin Tomescu, and Sophia Yakoubov

This report will focus on the discussion on Sep 2 and Sep 5 around game-theoretic aspects of the problem.

The motivating setting for this discussion group is a blockchain where users would like to “delegate” secrets to the blockchain, for example in order to encrypt a message that is guaranteed to be decrypted iff some smart-contract condition is met, or having a smart contract generate a signature.

This seems to be a classic use of threshold secret sharing. The problem, however, as noted by Alin Tomescu, is that in the blockchain setting parties are *rational* rather than honest, and can be bribed into divulging their shares (potentially even using the blockchain itself to do so).

The main idea to combat this is to punish the shareholders who leak their shares. We tried to model the question a bit more formally and to come up with constructions that could potentially detect and punish unauthorized leaks.

Model. The model we came up with isn’t exactly the classic secret sharing; we call it “rational threshold secret keeping,” and it works roughly as follows, with n parties and threshold t :

- There is a single dealer and n parties.
- The dealer wishes to share n secrets, such that secret s_i will “belong” to party i .
- The dealer does this by sending each party a share (party i receives x_i)
- Every subset of t parties should be able to reconstruct (at least) one of the secrets.

Cryptographic guarantees.

- Any subset of $t - 1$ parties has no knowledge about any of the secrets.
- A coalition of all $n - 1$ parties except for party i has no knowledge of secret s_i .

This means that if we can prove that s_i was involved in an unauthorized computation, we can punish party i .

Construction. The scheme we came up with involved secret sharing with an access structure that has two types of shares: each party receives a “big” party-specific share for its own secret, and $n - 1$ “small” shares for each of the other secrets. Reconstruction of a secret requires the big share and $t - 1$ small shares.

(We note that while this scheme does give the guarantees we require from the model, in post-Dagstuhl discussions we realized that it doesn’t have very good “rationality” properties – nailing down the “right” definition in terms of rational parties is ongoing work.)

4.7 Can blockchains keep secrets using TEE?

Alin Tomescu (Aptos Labs – Palo Alto, US), Sourav Das (University of Illinois – Urbana-Champaign, US), Sisi Duan (Tsinghua University – Beijing, CN), Giuliano Losa (Stellar Development Foundation – San Francisco, US), Filip Rezabek (TU München – Garching, DE), and Fan Zhang (Yale University – New Haven, US)

License © Creative Commons BY 4.0 International license
© Alin Tomescu, Sourav Das, Sisi Duan, Giuliano Losa, Filip Rezabek, and Fan Zhang

We started from the premise that, if we are to rely on trusted-execution environments (TEEs) to prevent collusion in decentralized secret-sharing infrastructures, this must not worsen the current privacy of these infrastructures. For example, it would be unacceptable for a break in the TEE to lead to the shared secret being revealed. Therefore, this excludes naive solutions that try to avoid a distributed key generation (DKG) by simply storing the shared secret in the TEE.

We then quickly realized an inherent trade-off in relying on TEEs to prevent collusion: if the TEE is to prevent colluding validators from reconstructing the secret when they are not supposed to, then it follows that if the TEE is unavailable / crashed / buggy, then the underlying secret sharing infrastructure will lose its liveness. Otherwise, if it did not, that would imply colluders could reconstruct the shared secret by (say) crashing the TEE.

Note that, in live systems such as blockchains where the secret sharing infrastructure may be used for generating randomness via threshold verifiable random functions (tVRFs), TEEs could crash. If so, losing the liveness of the blockchain would be very problematic. But we quickly realized that there may be a way to mitigate even against this (to be described later).

We proposed a simple architecture. Currently, the validators maintain a t-out-of-n secret sharing $(s_1, s_2, \dots, s_n) = \text{Share}(s, t, n)$ of a secret s . Therefore, t or more validators can collude to reveal s , which would be bad. To prevent this, we assume each validator will have a TEE. Then, we assume the TEEs can establish their own, independent, t-out-of-n sharing $(s'_1, s'_2, \dots, s'_n) = \text{Share}(s', t, n)$ of a different, independent, secret s' . As a result, the final secret will be $s + s'$, instead of just s' . This means that, if we can restrict the usage of the TEE-secured s' secret, we could make it impossible for colluding validators to reveal $s + s'$ or any function of it, such as a (threshold) VRF, which we focus on hereafter.

Note that this already prevents collusion where validators try to pre-agree on the shared secret s outputted by the DKG protocol, since the TEE will effectively “randomize” the final shared secret as $s + s'$.

To make sure a validator i cannot “trick” its TEE to reveal a VRF under its share s'_i , we would require the TEE to maintain an append-only view of the blockchain’s consensus (e.g., keep track of the latest block header) and only produce a VRF share under s'_i if “it is time to,” according to this view. This ensures that the TEE will never compute its VRF share “too early,” which would make the randomness predictable.

Note that once the TEE correctly computes its VRF share under s'_i , it can be combined with the validator’s VRF share under s_i . This yields a “collusion-free” VRF share under $s_i + s'_i$ and if enough validators combine their “collusion-free” shares, this yields a VRF under $s + s'$, which is what is needed in the higher-level randomness beacon protocol.

We conceptualized this as storing “half” of a validator’s secret key share (i.e., s'_i) inside the TEE, and the other “half” (i.e., s_i) inside the validator. Note that the “TEE half” is to prevent collusion while the “validator half” is to prevent leakage of the final secret $s + s'$ when the TEEs are all compromised (which is not unfathomable anymore).

We also had many other ideas and thoughts around this:

- It may be possible to reduce reliance on TEEs: e.g., when using a PVSS-based DKG, if PVSS transcripts from more than $> 2/3$ of validators are required, then need $> 1/3$ of validators to have TEEs to make sure one validator contributes honestly.
- The necessary TEE functionality should be abstracted, since it may come from one of many providers (e.g., Intel SGX, AWS Nitro, AMD)
 - Need attestation: i.e., enclave signs PVSS transcript it deals
 - Need sealing: i.e., enclave persists its share of the secret key share
- Need to consider how resharing an existing secret might complicate the story, if at all
- Another interesting requirement is upgradeability: e.g., the block format/signatures might change and the TEE needs to be upgraded.
 - May need to put enclave hashes on a chain to allow for upgradeability
 - For example, the blockchain would expect that the PVSS be signed together with the so-called “PVSS-dealing-enclave-hash”
 - Would require reproducible builds that give the same enclave hash to all validators
 - May need “re-sealing” functionality inside the TEE, to allow it to re-encrypt its share for a new enclave hash that is stored on-chain
 - Interestingly, we get tamper-evidence because the enclave hashes are posted on the chain, should the validators post malicious enclaves.

Participants

- Ittai Abraham
Intel – Petah Tikva, IL
- Georgia Avarikioti
TU Wien, AT
- Renas Bacho
CISPA – St. Ingbert, DE
- Adithya Bhat
Visa USA – Foster City, US
- Sourav Das
University of Illinois –
Urbana-Champaign, US
- Nico Döttling
CISPA – Saarbrücken, DE
- Sisi Duan
Tsinghua University –
Beijing, CN
- Bryan Ford
EPFL Lausanne, CH
- Juan A. Garay
Texas A&M University –
College Station, US
- Neil Girdharan
University of California –
Berkeley, US
- Aniket Kate
Purdue University – West
Lafayette, US & Supra Research –
West Lafayette, US
- Aggelos Kiayias
University of Edinburgh, GB
- Andrew Lewis-Pye
London School of Economics &
Political Science, GB
- Chen-Da Liu-Zhang
Hochschule Luzern – Rotkreuz,
CH & Web3 Foundation –
Zug, CH
- Giuliano Losa
Stellar Development Foundation –
San Francisco, US
- Julian Loss
CISPA – Saarbrücken, DE
- Tal Moran
Reichman University –
Herzliya, IL
- Kartik Nayak
Duke University – Durham, US
- Matthieu Rabaud
Télécom Paris, FR
- Filip Rezabek
TU München – Garching, DE
- Mohammad Sadoghi
University of California –
Davis, US
- Alin Tomescu
Aptos Labs – Palo Alto, US
- Sophia Yakoubov
Aarhus University, DK
- Sravya Yandamuri
Duke University – Durham, US
- Fan Zhang
Yale University – New Haven, US

