[

intents-Newsletter

1016×550 323 KB

](https://europe1.discourse-cdn.com/standard20/uploads/anoma1/original/1X/8ab8f7176bf287beefd63fa16cb819249781e999.png)

Introduction

Welcome to V2 of the Intents Newsletter. We appreciate your patience. This issue is packed with four new Anoma Research Topics, hot off the presses including: Cross-chain integrity with Controller Labels and Endorsements

, Anoma Resource Machine Specification v2

, Heterogeneous Narwhal & Paxos

, Heterogeneous Paxos 2.0 specs

.

This month the newsletter leads off with distributed systems literature, heterogeneous trust in particular. We finish the topic with our recent paper on the Anoma P2P layer. The intents topic covers the new Resource Machine specification and revisits an excellent report on Anoma's Intent language, Juvix.

In the solver category, we revisit our research on private solving and our survey on constraint satisfaction problems. In the applications section, Christopher Goes' recent talk at DBA's Research Day, Scale-free Money: theory and practice (aka breaking the crypto wheel) is featured alongside the canonical collaborative finance literature, Liquidity-Saving through Obligation-Clearing and Mutual Credit: An Effective Monetary Innovation for SMEs in Times of Crisis

, which partially inspired Cycles Money.

In the final section we review both new and old research on TEEs including Ethan Buchman's recent presentation at Devmos 2024, Quartz: A Privacy Preserving Sidecar for CosmWasm

, as well as the new Autonomous TEEs Manifesto

by Poetic Technologies. The category also displays some foundational literature on TEEs including Ekiden: A Platform for Confidentiality-Preserving, Trustworthy, and Performant Smart Contract Execution

.

Acknowledgements

Thank you to all the brilliant researchers whose work is featured in this newsletter.

Table of Contents

Distributed Systems

In our section on distributed systems, we begin by reviewing a new Anoma Research Topic, Cross-chain Integrity with Controller Labels and endorsement.

A controller

is the authoritative state machine of a digital object. Controllers provide a new system for safely transferring mutable digital objects (resources) between mutually distrusting controllers. The trustworthiness of each object and promises made about

these objects by each controller is also tracked.

The proposed solution is based on taint tracking and endorsement from Information flow control research. The report introduces the concept of Controller tags and generalizes wrapped tokens to controller DAGs. By maintaining a Causal resource history (CRH) and a Consistent Controller Labels (CCL), the system allows applications to trust resources based on their authoritative controllers.

The Section also features previous work on heterogeneous trust. Distributed systems based on heterogeneous trust allow participants (e.g., blockchain validators) to make their own judgments about which other participants they trust. See the video recordings from this wonderful workshop Heterogeneous Trust in Distributed Systems at AFT 2023 for more insights.

We include the following reports: Heterogeneous Paxos: Technical report

, Distributed Protocols and Heterogeneous Trust: Technical Report

, Heterogeneous Narwhal and Paxos

, Heterogeneous Paxos 2.0 Specification

, and Blockchains Nodes, are Heterogeneous and your P2P Overlay should be Too: PODS

.

Cross-Chain Integrity with Controller Labels and Endorsement

By Isaac Sheff

Abstract:

In distributed systems, mutable digital objects typically require some state machine to decide on their definitive current state. This state machine can be replicated to enhance availability and fault tolerance. We call the authoritative state machine of a digital object its controller. Typical examples of controllers defining objects include a database storing a record, or a blockchain storing the current state of a smart contract. Without some kind of controller, different parties may have contradictory notions of what the state is, and no way to reconcile them. In a distributed system, some controllers may be Byzantine, and make duplicitous or incoherent statements about state. Here we design rules and procedures for a multi-state-machine ecosystem, featuring digital objects, or resources, with application-defined state-dependent rules for how they can be updated.

[

Controller-Salad

1294×748 109 KB

](https://europe1.discourse-cdn.com/standard20/uploads/anoma1/original/1X/8a050ec83570c508e3f8aab9ff6a1ef48fd080e7.png)

Source:

Sheff, Cross-Chain Integrity with Controller Labels and Endorsement

Each controller can express an authoritative state, including authoritative resource states. Each resource is also labeled with a controller identifier, whose state is definitive for this resource. Resources can transfer between controllers, and updates can depend on multiple resources, so resource labels also express a dependency graph detailing which controllers, if they were Byzantine, may have corrupted this resource. In a sense, these labels represent a distributed taint tracking or dynamic information flow control solution. One challenge is avoiding size explosion in this dependency graph: we enable removing unnecessary parts of history when, say, a resource transfers from $\boldsymbol{A}$ to $\boldsymbol{B}$ and back to $\boldsymbol{A}$ again. In information flow control terms, these operations require endorsement. Our resource controller operations generalize a number of techniques used in blockchain settings. We define rules and procedures for creating, updating, transferring, and tracking the state of labeled resources, and prove that our rules maintain safety properties including causal resource history and consistent controller labels.

Heterogeneous Narwhal and Paxos

By Tobias Heindel, Aleksandr Karbysheva, and Isaac Sheff

Abstract:

Blockchain interoperability often requires a mutually trusted layer (in the case of roll-ups or side-chains), or a third party (in the case of bridges). Heterogeneous Consensus protocols introduced the possibility for more direct, decentralized interoperability: chains can commit transactions together, given “enough” overlap in their trust assumptions. Like a bridge,

such protocols ensure atomicity under specific trust conditions.

[

Mempool-DAG

1332×658 52.6 KB

](https://europe1.discourse-cdn.com/standard20/uploads/anoma1/original/1X/346102ca7e7e2c4499c4fc559c0ab71d33dde37e.png)

Source:

Heindel, Karbysheva, and Sheff, Heterogeneous Narwhal and Paxos

Unlike a bridge, chains never relinquish control over the safety or liveness of their data. However, consensus is only part of the story: efficient mempool architectures like Narwhal can dramatically improve scalability and increase chain throughput. However, these mempools also rely on the chain's underlying trust assumptions. We generalize Narwhal for a Heterogeneous Trust model, and explain how to use it with Heterogeneous Paxos to commit cross-chain atomic transactions.

Heterogeneous Paxos 2.0: the Specs

By Aleksandr Karbyshev and Isaac Sheff

Abstract:

We introduce Heterogeneous Paxos 2.0, a modified version of Heterogeneous Paxos. Similar to the original, our assumptions and guarantees remain consistent; however, our new version simplifies the algorithm logic, reduces bandwidth usage, and enables a more efficient implementation.

Heterogeneous Paxos: Technical Report

By Isaac Sheff, Xinwen Wang, Robbert van Renesse, Andrew C. Myers

Abstract:

In distributed systems, a group of learners achieve consensus when, by observing the output of some acceptors, they all arrive at the same value. Consensus is crucial for ordering transactions in failure-tolerant systems. Traditional consensus algorithms are homogeneous in three ways:

- all learners are treated equally,
- all acceptors are treated equally, and
- all failures are treated equally.

[

HPaxos-Technical-Report

1318×348 65.1 KB

](https://europe1.discourse-cdn.com/standard20/uploads/anoma1/original/1X/b1ffaa121c7e01efe187641b85f931368d231072.png)

These assumptions, however, are unsuitable for cross-domain applications, including blockchains, where not all acceptors are equally trustworthy, and not all learners have the same assumptions and priorities. We present the first consensus algorithm to be heterogeneous in all three respects. Learners set their own mixed failure tolerances over differently trusted sets of acceptors. We express these assumptions in a novel Learner Graph, and demonstrate sufficient conditions for consensus. We present Heterogeneous Paxos: an extension of Byzantine Paxos. Heterogeneous Paxos achieves consensus for any viable Learner Graph in best-case three message sends, which is optimal. We present a proof-of-concept implementation, and demonstrate how tailoring for heterogeneous scenarios can save resources and latency.

Distributed Protocols and Heterogeneous Trust: Technical Report

By Isaac C. Sheff, Robbert van Renesse, Andrew C. Myers

Abstract:

The robustness of distributed systems is usually phrased in terms of the number of failures of certain types that they can

withstand. However, these failure models are too crude to describe the different kinds of trust and expectations of participants in the modern world of complex, integrated systems extending across different owners, networks, and administrative domains. Modern systems often exist in an environment of heterogeneous trust, in which different participants may have different opinions about the trustworthiness of other nodes, and a single participant may consider other nodes to differ in their trustworthiness. We explore how to construct distributed protocols that meet the requirements of all participants, even in heterogeneous trust environments.

[

Distributed-Protocols-and-Heterogeneous-Trust-Technical-Report

1266×1046 197 KB

](https://europe1.discourse-cdn.com/standard20/uploads/anoma1/original/1X/9e6addd8f6c09985691fc56f778de3eaed11cf80.png)

The key to our approach is using lattice-based information flow to analyse and prove protocol properties. To demonstrate this approach, we show how two earlier distributed algorithms can be generalized to work in the presence of heterogeneous trust: first, Heterogeneous Fast Consensus, an adaptation of the earlier Bosco Fast Consensus protocol; and second, Nysiad, an algorithm for converting crash-tolerant protocols to be Byzantine-tolerant. Through simulations, we show that customizing a protocol to a heterogeneous trust configuration yields performance improvements over the conventional protocol designed for homogeneous trust.

The Blocklace: A Universal, Byzantine Fault-Tolerant, Conflict-free Replicated Data Type

By Paulo Sérgio Almeida, Ehud Shapiro

Abstract:

Conflict-free Replicated Data Types (CRDTs) are designed for replica convergence without global coordination or consensus. Recent work has achieves the same in a Byzantine environment, through DAG-like structures based on cryptographic hashes of content. The blocklace is a partially-ordered generalization of the blockchain in which each block has any finite number of signed hash pointers to preceding blocks. We show that the blocklace datatype, with the sole operation of adding a single block, is a CRDT: it is both a pure operation-based CRDT, with self-tagging; and a delta-state CRDT, under a slight generalization of the delta framework.

[

Block-Lace

1406×658 162 KB

](https://europe1.discourse-cdn.com/standard20/uploads/anoma1/original/1X/3f776bf364bb8ef4ce9ad408d29585263423e6d0.jpeg)

Source:

Shapiro, [Grassroots Social Networking: Where People have Agency over their Personal Information and Social Graph

](https://arxiv.org/abs/2306.13941)

Allowing arbitrary values as payload, the blocklace can also be seen as a universal Byzantine fault-tolerant implementation for arbitrary CRDTs, under the operation-based approach. Current approaches only care about CRDT convergence, being equivocation-tolerant (they do not detect or prevent equivocations), allowing a Byzantine node to cause an arbitrary amount of harm by polluting the CRDT state with an infinite number of equivocations. We show that a blocklace can be used not only in an equivocation-tolerant way, but also so as to detect and eventually exclude Byzantine behavior, namely equivocations, even under the presence of collusion. The blocklace CRDT protocol ensures that the Byzantine nodes may harm only a finite prefix of the computation

Blockchain Nodes are Heterogeneous and Your P2P Overlay Should be Too: PODS

By Naqib Zarin, Isaac Sheff, and Stefanie Roos

Abstract:

At the core of each blockchain system, parties communicate through a peer-to-peer (P2P) overlay. Unfortunately, recent evidence suggests these P2P overlays represent a significant bottleneck for transaction throughput and scalability. Furthermore, they enable a number of attacks. We argue that these performance and security problems arise because current P2P overlays cannot fully capture the complexity of a blockchain system as they do not offer flexibility to accommodate node heterogeneity. We propose a novel approach to address these issues: P2P Overlay Domains with Sovereignty (PODS), which allows nodes in a single overlay to belong to multiple heterogeneous groups, called domains.

[

P2-PPODSimage

1215×735 49 KB

](https://europe1.discourse-cdn.com/standard20/uploads/anoma1/original/1X/f70d2a9b2dae614e776808b0562601a340536cd9.png)

Source:

Zarin, [A brief introduction to Anoma's P2P layer

](https://anoma.net/blog/anomas-p2p-layer)

Each domain features its own set of protocols, tailored to the characteristics and needs of its nodes. To demonstrate the effectiveness of PODS, we design and implement two novel node discovery protocols: FedKad and SovKad. Using a custom simulator, we show that node discovery using PODS (SovKad) architecture outperforms both single overlay (Kademlia) and multi-overlay (FedKad) architectures in terms of hop count and success rate, though FedKad requires slightly less bandwidth.

Intents

At the heart of Anoma sits the Anoma Resource Machine. The Resource Machine is an alternative state architecture, it's a way to organize state, which is not the same thing as a virtual machine.

The Anoma resource machine implements intents without loss of generality. The RM provides state-level and intent-level information flow control, intent-level composability, and the programming framework for ordering/ storage/ compute services required for heterogeneous trust. Heterogeneous trust also requires cross-domain state synchronization and verification, which the resource machine's state architecture is designed to make simple and efficient.

Resources are the atomic units of state. They can be created once and consumed once. Resources are immutable. The current state of the system is made up of all created but not yet consumed resources. The resource model is somewhat of a generalization of the UTXO model. Unlike the UTXO model, the resource model is not limited to one context or currency case. The resource logic is a predicate associated with each resource which describes when you can create and consume a resource. The resource kind specifies the resources' fungibility domain. Resources of the same kind are fungible, resources of different kinds are non-fungible.

In particular, a resource R

is a composite structure which contains 8 components:

R = (l, label, q, v, eph, nonce, npk, r_s)

- l
- is a reference to the predicate of the resource logic
- label
- is the resource's fungibility domain
- q
- is the resource quantity
- v
- is fungible data, its extra but doesn't affect the fungibility of the resource
- eph
- flags the ephemerality of the resource
- nonce
- guarantes the uniqueness of the resource computable component
- npk
- the nullifier public key which references the nullifier key use to derive the resource nullifier.

- r_s
- references any randomness seed used

Also note that a resource has four computable components:

- r.cms
- commitment to the resource which is stored in global accumulator, a merkle tree in practice. H
- r.nf
- provides proof that a resource has been consumed.
- r.kind
- a hash of the resource logic reference and resource label. Somewhat of an identifier for the resource. Has a kind-distinctness (additively homomorphic).
- r. \Delta
- used to reason about the quantities of resource kinds during the balance check.

A transaction is a composite structure TX

TX = (rts, cms, nfs, \Pi, \Delta, extra)

, where:

- rts \subseteq \mathbb{F}_{rt}

is a set of roots of the commitment tree

- cms \subseteq \mathbb{F}_{cm}

is a set of created resources' commitments.

- nfs \subseteq \mathbb{F}_{nf}

is a set of consumed resources' nullifiers.

- \Pi: { \pi: ProofRecord}

is a set of proof records.

- \Delta_{tx}: \mathbb{F}_{\Delta}

is computed from \Delta

parameters of created and consumed resources. It represents the total delta change induced by the transaction.

- extra: {(k, d): k \in \mathbb{F}*{key}, d \subseteq \mathbb{F}*{d}}

contains extra information requested by the logics of created and consumed resources.

- \Phi:

PREF

where PREF

= TX

→ [0, 1]

is a preference function that takes a transaction as input and outputs a normalized value in the interval [0, 1]

that reflects the users' satisfaction with the produced transaction. For example a user who wants to receive at least q = 100

of a resource kind resource_{.A}

for a fixed amount of resource kind resource_{.B}

might set the preference function to implement a linear function that returns 0

at q = 100

and returns 1

at q = q_{max} = |\mathbb{F}_{q}| - 1

A transaction is valid

with respect to a previous state if and only if:

- rts

contains valid commitment tree roots for the membership proofs

- nfs

contains valid, unique nullifiers for consumed resources

- input and output resources have valid resource logic and compliance proofs
- \Delta

is computed correctly

A transaction is balanced

if and only if \Delta = 0

. Applying a transaction updates the commitment Merkle tree and nullifier set. Transaction candidates

can access the current state and look up resources by kind for post-ordering state-dependent updates without conflicts.

Commitments and nullifiers

To create a resource, you publish a resource commitment (not defined by the resource machine). How the commitment is published depends on the concrete instantiation of the RM, but generally, it is a piece of data associated with the resource. This piece of data is enough to show that a specific resource was created, without necessarily showing the resource itself. To publish this commitment, you append it to a global commitment accumulator.

To consume a resource, you publish its nullifier (not fully defined) which depends on the instantiation of the Resource machine. You append the nullifier to the global nullifier set to count the resource as consumed.

In particular, the Anoma resource machine tracks all of these resources using two Merkel trees; the commitment tree and the nullifier set.

- The commitment tree tracks all resources which have been created. Every time we create a resource, we add a commitment to the commitment tree and we re-calculate the commitment tree root using normal Merkel tree methods
- Every time we consume a resource, we reveal a nullifier and that goes in the nullifier set.

Unbundling VMs

By way of comparison, the Ethereum Virtual Machine (EVM), along with other blockchain VMs like the SVM, combines three essential components: EVM (and SVM etc.) bundles three things that are conceptually separable:

- state architecture - how state is stored, how state is referenced, how you write to it, how you read from it and what the permissions are to changing it
- e.g. SSTORE

, SLOAD

, MSTORE

, MLOAD

,…

- e.g. SSTORE

, SLOAD

, MSTORE

, MLOAD

,…

- instruction set - how actual computation is performed, once you've read some data from state form tx calldata or w/e how do you perform operations on it, addition, multiplication, signature checks
- e.g. ADD

, MUL

, MOD

, EXP

,…

- e.g. ADD

, MUL

, MOD

, EXP

,…

- message-passing model - how different programs in all of these scenarios (multi-party interactions) different smart contracts or user programs talking to one another. This requires some kind of message passing.
- e.g., (address).call()

(address).delegatecall()

, (address).staticcall()

- e.g., (address).call()

(address).delegatecall()

, (address).staticcall()

The Anoma Resource Machine (ARM) specifies only the state architecture:

- State is split into resources, each associated with a resource logic which defines the permissions for that piece of state.
- Instruction set is an implementation choice which can be made differently for different resource logics. The resource machine only requires that your resource logic must define its own method of computation that we can verify and prove it.
- Message-passing model is determined at runtime, depending on how intents flow around the network. This will determine how messages get passed b/w different solvers computing solutions which satisfy different resource logics. All you need to produce at the end is something that satisfies everything whose state was changed.

The Resource Machine is flexible. For example, It's possible to make the resource machine compatible with the instruction set and the message passing model of the EVM. The resource machine is not a replacement for the EVM or other virtual machines, it's complimentary.

Indeed, we lead off this topic on Intents with the version two of the Resource Machine Specification. The specification includes application examples such as a token transfer with identity isolation, a counter application, and a proof-of-stake application. There are also transaction examples of two and three party exchanges. In particular, be sure to enjoy the diagrams made by Yulia Khalniyazova, co-author of the report.

Anoma Resource Machine (ARM) Specification v2

By Yulia Khalniyazova & Christopher Goes

Abstract:

The article explores the Anoma Resource Machine (ARM) within the Anoma protocol, providing a comprehensive understanding of its role in facilitating state updates based on user preferences. Drawing parallels with the Ethereum Virtual Machine, the ARM introduces a flexible transaction model, diverging from traditional account and UTXO models.

[

Three-Party-Exchange-Resource-Machine

1404×740 266 KB

](https://europe1.discourse-cdn.com/standard20/uploads/anoma1/original/1X/6ada71d0470ac1b6fd83a30b5ee89629d4adfde5.jpeg)

Key properties such as atomic state transitions, information flow control, account abstraction, and an intent-centric architecture contribute to the ARM's robustness and versatility. Inspired by the Zcash protocol, the ARM leverages commitment accumulators to ensure transaction privacy. The article outlines essential building blocks, computable components, and requirements for constructing the ARM, highlighting its unique approach to resource-based state management.

The Core language of Juvix

By Lukasz Czajka

Abstract:

This report describes JuvixCore – a minimalistic intermediate functional language into which Juvix desugars. We provide a precise and abstract specification of JuvixCore's syntax, evaluation semantics, and optional type system.

[

Juvix-Comparison-Table

1082×544 56.3 KB

](https://europe1.discourse-cdn.com/standard20/uploads/anoma1/original/1X/04032993f9f68c3097725bfc71abbfab1208084b.png)

Source:

Czajka, The Core language of Juvix

The correspondence between this specification and the actual implementation is discussed and the role of JuvixCore in the Juvix compilation pipeline is explained. Finally, we compare the language features available in JuvixCore with those in Juvix and other popular functional languages.

Solving

Often times when the term solver arises in the discourse, different participants have different understanding of what a solver is. For example, in the recent paper [An analysis of Intent-Based Markets

](https://arxiv.org/abs/2403.02525) featured in the prior newsletter, a Solver is defined as an agent(s) who compete to satisfy user orders which may include complicated user specified conditions (Chitra, Kulkarnis, Pi, Diamandis 24').

Recently at our Intents Discussions event during EthCC we hosted a panel and discussed this definition with some folks who are practitioners or have experience in and around the solving arena. We began the panel by attempting to define what a Solver is from first principles. A couple more definitions:

- Solvers are self-interested actors who are financially incentivized and rewarded for their participation in optimization markets.
- A solver is a mechanism that you trust to outsource an optimization problem to. That mechanism could be a trusted compute source that computes the optimal outcome for you.

Keep these in mind as you work through the first article is an Anoma Research Topic which deals with exploring cryptographic approaches to enhance Privacy in Intent Solving or Private Solving for short. The second featured article is another Anoma Research Topic which provides a survey on constraint satisfaction problems including an overview of CSP tractability which aims to provide the reader with the limits of CSP solving.

Exploring Cryptographic Approaches to Enhance Privacy in Intent Solving

By Yulia Khalniyazova

Abstract:

In this report, we explore cryptographic strategies that could be used to enhance privacy during the phase of solving intents

in Anoma.

[

Private-Solving-Table

1078×418 43.9 KB

](https://europe1.discourse-cdn.com/standard20/uploads/anoma1/original/1X/4d586eae19d7f24740a76ad291db46adf2abc818.png)

We consider both well-known solutions like trusted execution environment, multiparty computations and homomorphic encryption and more exotic primitives like functional encryption, witness encryption, searchable encryption and collaborative SNARKs. We identify the limiting factors for each primitive and provide a summary of the results

Constraint Satisfaction Problems: A Survey for Anoma

By Anthony Hart

Abstract:

This paper serves as a broad introduction to Constraint Satisfaction Problems (CSPs) tailored for professionals working on Anoma, a decentralized financial framework. Our primary aim is to build a robust intuition for CSPs, preparing the Anoma community for its application in optimizing intents—defined as desires for future states of the financial system. We explore a wide array of CSP examples, including but not limited to Boolean Satisfiability and Integer Linear Programming, to equip readers with a general understanding of the field. We also include an overview of CSP tractability, aiming to give the reader an understanding of the limitations of solvability.

[

CSPcolor-Perms

1218×710 113 KB

](https://europe1.discourse-cdn.com/standard20/uploads/anoma1/original/1X/7f43f99605c4aa5a056ae69987fb05cf5f0f8af5.png)

A unique emphasis is placed on compositional methods, facilitating distributed problem-solving across multiple agents. Towards the end, we formalize the notion of ’intents’ within the framework of Constraint Optimization.

Applications

Applications have become a popular topic of discussion in the crypto discourse over the last couple of months in particular. Below we feature a recent talk from Christopher Goes given at DBA’s Research Day 2024 in New York. The talk conjectures a theory of money: Scale-free money. The curious reader maybe interested in reading Christopher’s previous post on the topic, Towards heterotopia - the prerequisite cultural and technological substrate for a return to a world of scale-free credit money

.

The second article featured in this section can be thought of as the canonical paper on collaborative finance. The paper shows that obligation-clearing is a very effective liquidity-saving method for providing relief in the trade credit market. The paper also demonstrates that when used with a complementary currency system such as mutual credit as a liquidity source, the effectiveness of obligation-clearing can be doubled. This paper has inspired some of the work of the builders of Cycles.Money. Cycles is a new application (an open clearing protocol) designed to clear the most debt for the most people with the least amount of money. The Cycles Money white paper is coming soon, but meanwhile, enjoy the classic Liquidity-Saving through Obligation-Clearing and Mutual Credit: An Effective Monetary Innovation for SMEs in Times of Crisis

Scale-free Money: theory and practice (aka breaking the crypto wheel)

By Christopher Goes

Excerpt:

The basic idea of scale-free money is pretty simple. It is a theory of how to use money. You start by defining some values or objectives, different things you want to see in the world and associating different tokens (denominations) with those values. You then use issuance of those denominations to fund progress towards whatever objectives you have delineated with different values. The holders of those tokens vote on what objectives of worth pursuing. One dynamic of scale-free money is that you always must make a hard decision about which values or codependent enough that you want to make them one thing, and which values are independent enough to make separate things. There will be as many denominations as distinct

objectives and values are worth representing as distinct. These denominations will be created, destroyed, sometimes they will merge split which makes sense in a world of flux.

The mechanism you need to make this work in practice is an infrastructure that makes it easy to separate medium of exchange, unit of account, and store of value so people don't have to agree on denominations in order to interact. Also important that people don't pay any kind of UX penalty for disagreeing on denominations as long as they agree on the parameters of the specific interaction. Crypto is already pretty good at this, and is mostly going in this direction. In this world of scale-free money the units of play will be invisible to humans most of the time. Rather than just having tokens which are discussed on twitter, we may have tokens created between peer to peer nodes for network bandwidth, storage, compute which are changing hand autonomously and are created and destroyed all of the time. Only some of these tokens will arise to surface level consciousness.

[

Scale-free-Money

1790×780 32.2 KB

](https://europe1.discourse-cdn.com/standard20/uploads/anoma1/original/1X/ec9fe0edad7ce6fdea2ab2b06db3f9d3360e8388.png)

One particular useful characteristic of this type of money is that its scale neutral. If we look in the world today at coordination structures and monetary control, we see that we have an asymmetric graph (left) of scale vs. control over some kind of locus of money. We have some local currencies, crypto and small scale stuff on the left. On the right we have hundreds of millions of people involved in some governance structure represented as fiat currency. Everything in the middle is basically unserved, there is no locus of monetary control at that level. If you take a first principles approach where you try and associate denominations of money with different values, you would see a graph that looks more like the one of the right. Here you have many different locuses of control, different denominations, different governance functions for issuance, difference things being funded, and different values/objectives. These things are more evenly balanced vs. scale.

Liquidity-Saving through Obligation-Clearing and Mutual Credit: An Effective Monetary Innovation for SMEs in Times of Crisis

By Tomaž Fleischman, Paolo Dini, and Giuseppe Littera

Abstract:

During financial crises, liquidity tends to become scarce, a problem that disproportionately affects small companies. This paper shows that obligation-clearing is a very effective liquidity-saving method for providing relief in the trade credit market and, therefore, on the supply-side or productive part of the economy. The paper also demonstrates that when used in conjunction with a complementary currency system such as mutual credit as a liquidity source the effectiveness of obligation-clearing can be doubled. Real data from the Sardex mutual credit system show a reduction of net internal debt of the obligation network of approximately 25% when obligation-clearing is used by itself and of 50% when it is used together with mutual credit.

[

MTCS

1112×1210 595 KB

](https://europe1.discourse-cdn.com/standard20/uploads/anoma1/original/1X/a3c31c11858f1dcdb1732c22df8bb1bd3ffe4fd1.jpeg)

Source:

Fleischman, Dini, and Littera, Liquidity-Saving through Obligation-Clearing and Mutual Credit:An Effective Monetary Innovation for SMEs in Times of Crisis

These instruments are also relevant from the point of view of risk mitigation for lenders, based in part on the information on individual companies that the mutual credit circuit manager can provide to banks (upon the circuit member's request) and in part on the relief that liquidity-saving provides especially to NPL companies. The paper concludes by outlining recommendations for how even greater savings could be achieved by including the tax authority as another node in the obligation network.

Trusted Execution Environments

- The curious reader should start by watching this talk by Ari Juels which informed the analysis herein.

Trusted Execution Environments (TEEs) have recently gained significant popularity in the crypto discourse. TEEs provide

powerful protections for program execution, ensuring integrity and confidentiality within an isolated environment known as an enclave. A TEE can act like a trusted third party.

- TEEs provide integrity. When a program runs inside an enclave, it is protected from tampering by the creator of the underlying software, the host owner, the operating system, and other processes. This ensures that only the intended program continues to execute without interference.
- TEEs provide confidentiality. The private state of a program running within an enclave is shielded from the operating system and other processes, ensuring that sensitive data remains secure and inaccessible to unauthorized parties.
- TEEs can provide attestations. With attestation, a TEE can remotely prove to a party that a specific program is running correctly. Attestation enables users to verify the exact program they are interacting with. In particular in an SGX there are two ways to get remote attestations: Enhanced Privacy ID (EPID) & Data Center attestation primitive (DCAP). (See this post on X by Pratyush Ranjan Tiwari for more details)
- EPID - is Intel's proprietary technology for secure, anonymous device authentication. It is widely deployed but relies heavily on Intel's centralized infrastructure for attestation. While EPID allows devices to prove their authenticity without revealing their specific identity, it requires an internet connection to Intel's servers.
- DCAP - is Intel's solution for flexible, scalable attestation in enterprises. It enables organizations to create and manage their own attestation systems, providing full control and removing dependence on Intel. DCAP is well-suited for decentralized environments and applications using TEEs - it allows attestation without relying on internet access to Intel's servers.
- EPID - is Intel's proprietary technology for secure, anonymous device authentication. It is widely deployed but relies heavily on Intel's centralized infrastructure for attestation. While EPID allows devices to prove their authenticity without revealing their specific identity, it requires an internet connection to Intel's servers.
- DCAP - is Intel's solution for flexible, scalable attestation in enterprises. It enables organizations to create and manage their own attestation systems, providing full control and removing dependence on Intel. DCAP is well-suited for decentralized environments and applications using TEEs - it allows attestation without relying on internet access to Intel's servers.

As such, practitioners in the community continue to explore potential applications of TEEs in enabling verifiable and private computation. TEEs are not without tradeoffs. Various SGX attacks have been discovered and documented at sgx dot fail, over the years. TEE enthusiasts often argue for TEEs used for defense in depth purposes. If you add TEE to a system correctly, you make the system strictly stronger. For example, you could run an MPC protocol and put the nodes in enclaves. To break the MPC protocol, an adversary would have a more difficult time compromising the nodes now.

In this section, we begin by featuring a recent presentation given by Ethan Buchman at Devmos 2024. The presentation reviews Informal Systems work on Quartz, a privacy preserving sidecar for CosmWasm.

Quartz: A Privacy Preserving Sidecar for CosmWasm

By Ethan Buchman

Exceprt: Cycles is an open clearing protocol designed to clear the most debt for the most people with the least amount of money. Cycles is derived from our thinking about what is needed from a monetary perspective in order to take blockchains to the next level and use them for what they are useful for, atomic multilateral settlement. In order to make Cycles work we need to do operations on encrypted data. The talk is about why we are doing that, and why we chose the technology we happened to choose. We think the framework we are building for Cycles can actually be useful for everyone building CosmWasm applications. This would allow them to have privacy preserving smart contracts.

Many people just turn to zero-knowledge proofs for private computation and say this is all we need for privacy. But it turns out, ZK is not enough. The talk will take you through an argument about these existing cryptographic primitives and explain why they won't work for us in Cycles and in many other applications. Ultimately, we need Trusted Execution Environments (TEEs).

[

quartzoverview

2230×1044 147 KB

](https://europe1.discourse-cdn.com/standard20/uploads/anoma1/original/1X/3e3695a915b379b5a82e09c1206af18135ab4a30.jpeg)

Source: Buchman, Quartz: A Privacy preserving Sidecar for CosmWasm

1. Users submit encrypted intents, encrypted to the public key of the enclave (which is registered onchain during a handshake process).

2. The intents are retrieved.
3. Intents are decrypted in the enclave. Compute you want is executed and encrypted.
4. Encrypted results are published on-chain
5. The chain verifies the attestation from the enclave & ZKP (if desired) and updates the state
6. Users can read the encrypted results that are relevant to them

Autonomous TEE Manifesto

By Poetic Technologies

Abstract:

A spectre is haunting the world — the spectre of crypto. All the powers of the old world have entered into a holy alliance to exorcise this spectre. Two things result from this fact:

I.

Crypto is already acknowledged by all state power to be itself a stateless power. The promise of crypto, however, is at stake: both censorship resistance and decentralization have been compromised by centralizing forces in protocol design and in legacy institutions.

II.

It is high time that cypher-punks should openly, in the face of the whole world, publish their views, their aims, their tendencies, and meet this nursery tale of the Spectre of Crypto with a manifesto of the movement itself, to change the tides of time.

To that end, cypher-punks from different territories have assembled to sketch the following manifesto, to be published as a free and open-source document. The following document outlines a vision of an open-source software and hardware architecture for confidential computation, particularly Trusted Execution Environments or TEEs. In what follows, we outline our vision for each of the different parts of the technological substrate necessary to bring autonomous TEEs into being.

Ekiden: A Platform for Confidentiality-Preserving, Trustworthy, and Performant Smart Contract Execution

By Raymond Cheng, Fan Zhang, Jernej Kos, Warren He, Nicholas Hynes, Noah Johnson, Ari Juels, Andrew Miller, Dawn Song

Abstract:

Smart contracts are applications that execute on blockchains. Today they manage billions of dollars in value and motivate visionary plans for pervasive blockchain deployment. While smart contracts inherit the availability and other security assurances of blockchains, however, they are impeded by blockchains' lack of confidentiality and poor performance. We present Ekiden, a system that addresses these critical gaps by combining blockchains with Trusted Execution Environments (TEEs). Ekiden leverages a novel architecture that separates consensus from execution, enabling efficient TEE-backed confidentiality-preserving smart-contracts and high scalability. Our prototype (with Tendermint as the consensus layer) achieves example performance of 600x more throughput and 400x less latency at 1000x less cost than the Ethereum mainnet.

[

Ekiden-Architecture

1418×1300 140 KB

](https://europe1.discourse-cdn.com/standard20/uploads/anoma1/original/1X/55a12de589e8fd470bf1ddb51fd1f13c318f9995.png)

Source:

Cheng, Zhang, Kos, He, Hynes, Johnson, Juels, Miller, and Song, Ekiden: A Platform for Confidentiality-Preserving, Trustworthy, and Performant Smart Contract Execution

Another contribution of this paper is that we systematically identify and treat the pitfalls arising from harmonizing TEEs and blockchains. Treated separately, both TEEs and blockchains provide powerful guarantees, but hybridized, though, they engender new attacks. For example, in naive designs, privacy in TEE-backed contracts can be jeopardized by forgery of blocks, a seemingly unrelated attack vector. We believe the insights learned from Ekiden will prove to be of broad

importance in hybridized TEE-blockchain systems.

Intel SGX Explained

By Victor Costan and Srinivas Devadas

Abstract:

Intel's Software Guard Extensions (SGX) is a set of extensions to the Intel architecture that aims to provide integrity and confidentiality guarantees to security sensitive computation performed on a computer where all the privileged software (kernel, hypervisor, etc) is potentially malicious. This paper analyzes Intel SGX, based on the 3 papers that introduced it, on the Intel Software Developer's Manual (which supersedes the SGX manuals), on an ISCA 2015 tutorial, and on two patents. We use the papers, reference manuals, and tutorial as primary data sources, and only draw on the patents to fill in missing information.

[

Intel-SGXexplained

1092×1158 62.5 KB

](https://europe1.discourse-cdn.com/standard20/uploads/anoma1/original/1X/bf80faaa5243b253c818630fc640c1b9a34bc6bc.png)

Sourece:

Costan and Devadas, Intel SGX Explained

This paper does not reflect the information available in two papers that were published after the first version of this paper. This paper's contributions are a summary of the Intel-specific architectural and micro-architectural details needed to understand SGX, a detailed and structured presentation of the publicly available information on SGX, a series of intelligent guesses about some important but undocumented aspects of SGX, and an analysis of SGX's security properties.

SGXonerated: Finding (and Partially Fixing) Privacy Flaws in TEE-based Smart Contract Platforms Without Breaking the TEE

By Nerla Jean-Louis, Yunqi Li, Yan Ji, Harjasleen Malvai, Thomas Yurek, Sylvain Bellemare, and Andrew Miller

Abstract:

TEE-based smart contracts are an emerging blockchain architecture, offering fully programmable privacy with better performance than alternatives like secure multiparty computation. They can also support compatibility with existing smart contract languages, such that existing (plaintext) applications can be readily ported, picking up privacy enhancements automatically. While previous analysis of TEE-based smart contracts have focused on failures of TEE itself, we asked whether other aspects might be understudied. We focused on state consistency, a concern area highlighted by Li et al., as well as new concerns including access pattern leakage and software upgrade mechanisms. We carried out a code review of a cohort of four TEE-based smart contract platforms. These include Secret Network, the first to market with in-use applications, as well as Oasis, Phala, and Obscuro, which have at least released public test networks.

[

Tee-Architecture-High-level

1138×850 106 KB

](https://europe1.discourse-cdn.com/standard20/uploads/anoma1/original/1X/1bf07f7da03e755ac458aa6d06de2b4da5307756.png)

Source:

Jean-Louis, Li, Ji, Malvai, Yurek, Bellemare, and Miller, SGXonerated: Finding (and Partially Fixing) Privacy Flaws in TEE-based Smart Contract Platforms Without Breaking the TEE

The first and most broadly applicable result is that access pattern leakage occurs when handling persistent contract storage. On Secret Network, its fine-grained access pattern is catastrophic for the transaction privacy of SNIP-20 tokens. If ERC-20 tokens were naively ported to Oasis they would be similarly vulnerable; the others in the cohort leak coarse-grained information at approximately the page level (4 kilobytes). Improving and characterizing this will require adopting techniques from ORAMs or encrypted databases.

Second, the importance of state consistency has been under-appreciated, in part because exploiting such vulnerabilities is thought to be impractical. We show they are fully practical by building a proof-of-concept tool that breaks all advertised

privacy properties of SNIP-20 tokens, able to query the balance of individual accounts and the token amount of each transfer. We additionally demonstrate MEV attacks against the Sienna Swap application. As a final consequence of lacking state consistency, the developers have inadvertently introduced a decryption backdoor through their software upgrade process. We have helped the Secret developers mitigate this through a coordinated vulnerability disclosure, after which their state consistency should be roughly on par with the rest.

Feedback and Conversation

And that's a wrap. We covered a lot of ground. Thanks for reading! If you found some of the material interesting and want to chat with us about it, connect with us at:

- research.anoma.net

If you are interested in collaborating on an Anoma Research Topic (ART), like the one's featured in this newsletter, we'd be more than happy to collaborate. As a refresher, what is an ART?

- ART is an open-access, lightweight, peer-reviewed index of reports, primarily written and reviewed by Anoma's researchers and engineers, but open to anyone who wishes to participate

.

- The ART collection covers a diverse set of topics including distributed systems, cryptography, compilers and blockchains. These reports help map the theoretical foundations upon which Anoma is built.

Visit art.anoma.net for more details.

Content Submissions for future newsletters

Given that we intend to publish the newsletter on a monthly cadence, we are open to and encourage readers to submit content. Please submit content for consideration for future issues to intentnewsletter@proton.me

E-mail Signup

If you'd like this monthly newsletter delivered to your inboxsign-up here!