[

](https://imgflip.com/i/8o1r78)

Introduction

Welcome to the first edition of the Intents Newsletter

brought to you by some of the folks building Anoma. You may be thinking, "really, another newsletter? I can't even catch up with my reading list as is!" Fear not. The purpose of our publication is to provide signal. Indeed, we have a bias for long-form content. By filtering, we'll provide you with an aggregation of relevant research in and around intents, intent-centric protocols, and related literature.

The readers of our publication can expect one edition per month to start. This should provide enough time to work through material (if it is of interest) while not spamming your inbox so frequently that you unsubscribe. Also, we need to experiment, so this publication cadence could change. You are encouraged to provide feedback in the forum comments below. Tell us what you like, don't like, and where we can improve.

The newsletter will be broken down into different thematic categories which we find relevant to intent research. We'll provide the reader with a summary or abstract of the relevant work, along with a cited link to the source material.

Some months may include new categories while excluding categories from previous months. Indeed, it should also be noted, we are building an intent-centric protocol (Anoma) and we may feature our own material where relevant, but only if it's high quality (please forgive us in advance). Happy reading!

Acknowledgements

Thank you to @alberto, @graphomath, @cwgoes for feedback / review of different versions of this draft. Review ≠ endorsement. All errors, mistakes, and omissions, are my own

.

Thanks to @zach and @Emily_Heliax for insights on publication and distribution.

Kudos to the folks at Flashbots who produce The MEV Letter which we take inspiration from. It is a fantastic weekly publication.

Thank you to all the authors we've cited in this Newsletter

. Your contributions are significant, and we are grateful for them.

Table of Contents

Intents

[

](https://ibb.co/tLPN6bM)

Source

: Yulia Khalniyazova, Rise of the Resource Machines

Intents arise in the discourse as a term used to describe limit orders and cross-chain limit orders. While these are types of intents, they are very specific definitions. With Anoma our concept of intents is general. What are intents?

A colloquial definition of intents could be, intents are credible commitments to preferences and constraints over the space of possible state transitions. Mathematically, intents are atomic information flow constraints. A simple or more basic definition could be intents are signed messages specifying what actions users want without specifying the path of execution.

One interesting feature of intents with Anoma is the notion of intent-level composability. This means that applications written for Anoma can be composed at the intent level, not only the transaction level. Intents for different applications can be composed together and settled across domains, without any additional overhead or prior coordination on behalf of the application developers. In a sense, intent-level composability can help unify liquidity for various applications, assuming a shared P2P stack and node infrastructure.

Regardless of which definition resonates with you the reader (maybe none of these!) the concept has become increasingly popular not only in the discourse but for builders who are designing cross-chain / chain abstracted applications. In addition, intents are relevant to designers of intent-centric architectures, wallet developers and core protocol researchers. In this section, we'll highlight some of our favorite articles over the prior months that call out or are relevant to the concept of intents beginning with Christopher Goes' latest art Anoma as the universal intent machine for Ethereum

.

Resources

- [RFC][Draft]Anoma as the universal intent machine for Ethereum by Christopher Goes

Summary

: Anoma brings a universal intent machine to Ethereum, allowing developers to write applications in terms of intents, which can be ordered, solved, and settled anywhere in the Ethereum ecosystem.

An intent is a commitment to user preferences and constraints over the space of possible state transitions. The Anoma protocol brings a universal intent machine to Ethereum, allowing developers to write applications in terms of intents instead of transactions. Anoma is an interface, not an intermediary - it's not another MEV redirection device. Anoma provides a universal intent standard which does not constrain what kinds of intents can be expressed, but allows for built-in state, network, and application interoperability.

Intents and applications written with Anoma can be ordered, solved, and settled anywhere - on the Ethereum main chain, on EVM and non-EVM rollups, on Eigenlayer AVSs, on Cosmos chains, Solana, or any sufficiently programmable state machine. Anoma provides four key affordances: permissionless intent infrastructure, intent-level composability, information flow control, and heterogeneous trust - using three mechanisms: the resource machine, the heterogeneous trust node architecture, and languages for explicit service commitments. Anoma is compatible with any topology the Ethereum network chooses and lok something like this:

[

](https://ibb.co/6FpQg3s)

Source:

@apriori in Anoma as the universal intent machine for Ethereum

- Anoma Research Topic: Resource Machine Specification by Yulia Khalniyazova and 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.

[

](https://ibb.co/tLz0Pw9)

Source

: Yulia Khalniyazova, Rise of the Resource Machines

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.

Solvers and Game theoretic analysis

[

](https://ibb.co/xDnMd4K)

Source

: Chitra, Kulkarni, Pai, Diamandis, An Analysis of Intent-Based Markets

Solvers are the agents that match intents. Users sign messages that specify what they want (intents) and solvers go and execute these intents on the user's behalf. While solvers can be thought of as a new intermediary in the transaction supply chain (Ethereum-centric topology), the role as it's understood often encompasses three distinct functions: liquidity provision, computational search, and transaction submission. These functions are distinct. While the same entity may often provide liquidity, compute and transaction submission, they may not always. Such choices affect incentives.

In the following section we provide three in-depth perspectives on Solvers, starting with the excellent game-theoretic analysis by Chitra, Kulkarni, Pai, and Diamandis which models incentives Solvers face in intent-centric markets.

The authors investigate the incentives faced by solvers using two models: (i)

a probabilistic auction model and (ii)

an optimization-based deterministic model. Their work implies that designers of intent markets need to thoroughly understand the various costs that solvers face. If the goal of an intent network is to maximize welfare while also allowing permissionless entry into the market, designers should be careful to construct auctions whereby bidders minimize contention (competition) in sourcing inventory.

In the Intent Machines report Anthony Hart and D. Reusche introduce the concept of an "intent machine", entities capable of processing user intents and transforming system state accordingly. In their most general form, intent machines can be defined as a kind of coalgebra. Later in the paper, the authors describe more practical instantiations of intent machines for the purposes of barter mechanisms. Ultimately, the paper lays the foundation for future work on the game theory of intent solving.

Finally, we close the section with a paper hot off the presses (Github) where Laurence E. Day explores the intersection of intent solving and fiduciary responsibility as it pertains to English law.

Resources

- An Analysis of Intent-Based Markets by Tarun Chitra, Kshitij Kulkarni, Mallesh Pai, and Theo Diamandis

Abstract:

Mechanisms for decentralized finance on blockchains suffer from various problems, including suboptimal price execution for users, latency, and a worse user experience compared to their centralized counterparts. Recently, off-chain marketplaces, colloquially called 'intent markets,' have been proposed as a solution to these problems. In these markets, agents called solvers compete to satisfy user orders, which may include complicated user-specified conditions. We provide two formal models of solvers' strategic behavior: one probabilistic and another deterministic. In our first model, solvers initially pay upfront costs to enter a Dutch auction to fill the user's order and then exert congestive, costly effort to search for prices for the user. Our results show that the costs incurred by solvers result in restricted entry in the market.

[

](https://ibb.co/nL6YfLv)

Source:

Chitra, Kulkarni, Pai, Diamandis, An Analysis of Intent-Based Markets

Further, in the presence of costly effort and congestion, our results counter-intuitively show that a planner who aims to maximize user welfare may actually prefer to restrict entry, resulting in limited oligopoly. We then introduce an alternative, optimization-based deterministic model which corroborates these results. We conclude with extensions of our model to other auctions within blockchains and non-cryptocurrency applications, such as the US SEC's Proposal 615

- Anoma Research Topic: Intent Machines by Anthony Hart and D. Reusche

Abstract:

Intents, abstract entities encapsulating a user's desires, are a promising approach to designing modern,complex decentralized financial systems. This paper presents a theoretical framework for abstracting intent-processing systems through the introduction of "intent machines", entities capable of processing user intents and transforming system state accordingly. Special cases of such machines implement mechanisms like automated auctions, barter systems, and counterparty discovery.

[

](https://ibb.co/cQFsZBK)

Source:

Hart and Reusche, Intent Machines

We present intent machines abstractly as a coalgebra; a nondeterministic state transition function capturing the machine's dynamics. This allows us to consider the most general notions of equivalence, composition, and interaction. We then provide several examples of intent machines, including a toy example where states are natural numbers and intents are weighted relations, and more practical instantiations for creating barter systems, utilizing auction bids as intents to distribute resources within a network. This research lays the foundation for future investigations into the game-theoretic aspects of intent processing, where the abstract formulation is intended to serve as a minimal formulation required for such work.

- For All Intents And Purposes: On The Agency And Duties Of Solversby Laurence E. Day

Abstract:

The Ethereum blockchain network is moving away from an expectation that its users interact with it by fully defining their will, and towards a model wherein users describe the rough shape of their desired outcome and defer the implementation (and execution thereof) to specialised third parties known as 'solvers.' This model - under active development at the time of writing - is known as the 'intent-based' framework, and at present enjoys limited retail use in applications such as asset swaps.

The primary question which this dissertation seeks to answer is whether a solver performing such a swap - having volunteered or otherwise been selected to execute an associated intent - becomes an agent for the user which made the request. If so, what authority is given or can be inferred? Moreover, whether such agency manifests, which obligations - if any - apply?

This dissertation describes the current approach to executing transactions within Ethereum, the general design of the intent-based model and how it can implement asset swaps. Thereafter we review the concepts of agency, authority, fiduciaries and fiduciary duties within English law before analysing the implications of the two systems colliding. In doing so, we earn a brief glimpse at some of the inevitable future battlelines concerning whether code is law within the realm of decentralised finance.

Information flow control

[

](https://ibb.co/hV4K8J4)

Information flow control is a declarative specification of what should be disclosed to whom under which conditions. Desired information flow control properties constrain, but do not fix, the choice of specific cryptographic primitives such as regular encryption, succinct zero-knowledge proofs, or partially or fully homomorphic encryption - as long as the primitives chosen preserve the desired properties, the choice can be made on the basis of implementation availability, computational efficiency, and cryptographic interoperability.

-Christopher Goes

Information flow control puts the power back into the hands of the user and developer. These parties can specify whom they are willing to reveal what information to. Information flow control would allow users and developers to reason precisely about what information actions taken in their application can be disclosed to whom. We strongly recommend the Viaduct Paper along with the other relevant literature.

In particular, in the context of Anoma there are three types of information flow control.

- State level information flow control - this describes what can be seen by whom after a transaction is created and executed. For example, shielded execution may reveal only the necessary proof invariants, where transparent execution would reveal all transaction data to any interested observers.

- Network level information flow control - describes what metadata sent around the network can be seen by whom. A user may wish to hide certain physical network addresses, for example.

- Intent level information flow control - this is at the counterparty discovery layer. What information can solvers, who match intents, see during the solving process, which solvers have access to some of the user's information.

Information flow control, while perhaps a new term in the blockchain development lexicon, has a rich history in the academic literature dating back to the late 1970s. We highlight the term because it is an important concept which gives users and developers the power to control where intents are sent and what information is revealed to whom. Information flow policies can be verified statically at compile time or dynamically during runtime.

This topic is also relevant to the broader community working on Chain abstraction when thinking about applications, permissions, policies and how they should interact and be enforced. Should the user just trust their application or should they have more fine-grained control over their intents, specifying solving and settlement requirements when and where desired.

Resources

- Viaduct: An Extensible, Optimizing Compiler for Secure Distributed Programs by Coşku Acay, Andrew C. Myers, Rolph Recto, Elaine Shi, Joshua Gancher

Abstract

: Modern distributed systems involve interactions between principals with limited trust, so cryptographic mechanisms are needed to protect confidentiality and integrity. At the same time, most developers lack the training to securely employ cryptography. We present Viaduct, a compiler that transforms high-level programs into secure, efficient distributed realizations. Viaduct's source language allows developers to declaratively specify security policies by annotating their programs with information flow labels. The compiler uses these labels to synthesize distributed programs that use cryptography efficiently while still defending the source-level security policy. The Viaduct approach is general, and can be easily extended with new security mechanisms.

Our implementation of the Viaduct compiler comes with an extensible runtime system that includes plug-in support for multiparty computation, commitments, and zero-knowledge proofs. We have evaluated the system on a set of benchmarks, and the results indicate that our approach is feasible and can use cryptography in efficient, nontrivial ways.

[

](https://ibb.co/0JMvmZ6)

Source

: Viaduct, An Extensible, Optimizing Compiler for Secure Distributed Programs

- A Virtual Machine Based Information Flow Control System for Policy Enforcement by Srijith K. Nair, Patrick N.D. Simpson, Bruno Crispo, Andrew S. Tanenbaum

Abstract

:The ability to enforce usage policies attached to data in a fine grained manner requires that the system be able to trace and control the flow of information within it. This paper presents the design and implementation of such an information flow control system, named Trishul, as a Java Virtual Machine. In

particular we address the problem of tracing implicit information flow, which had not been resolved by previous run-time systems and the additional intricacies added on by the Java architecture. We argue that the security benefits offered by Trishul are substantial enough to counter-weigh the performance overhead of the system as shown by our experiments

- Paragon for Practical Programming with Information-Flow Control by Niklas Broberg, Bart van Delft, and David Sands

Abstract

:Conventional security policies for software applications are adequate for managing concerns on the level of access control. But standard abstraction mechanisms of mainstream programming languages are not sufficient to express how information is allowed to flow between resources once access to them has been obtained. In practice we believe that such control - information flow control - is needed to manage the end-to-end security properties of applications.

In this paper we present Paragon, a Java-based language with first-class support for static checking of information flow control policies. Paragon policies are specified in a logic-based policy language. By virtue of their explicitly stateful nature, these policies appear to be more expressive and flexible than those used in previous languages with information-flow support.

Our contribution is to present the design and implementation of Paragon, which smoothly integrates the policy language with Java's object-oriented setting, and reaps the benefits of the marriage with a fully fledged programming language.

- Certification of Programs for Secure Information Flow by Dorothy E. Denning and Peter J. Denning

Abstract

: This paper presents a certification mechanism for verifying the secure flow of information through a program. Because it exploits the properties of a lattice structure among security classes, the procedure is sufficiently simple that it can easily be included in the analysis phase of most existing compilers. Appropriate semantics are presented and proved correct. An important application is the confinement problem: The mechanism can prove that a program cannot cause supposedly nonconfidential results to depend on confidential input data.

Computer system security relies in part on information flow control, that is, on methods of regulating the dissemination of information among objects throughout the system. An information flow policy specifies a set of security classes for information, a flow relation defining permissible flows among these classes, and a method of binding each storage object to some class. An operation, or series of operations, that uses the value of some object, say x, to derive a value for another, say y, causes a flow from x to y. This flow is admissible in the given flow policy only if the security class of x flows into the security class of y.

- RIFLE: An Architectural Framework for User-Centric Information-Flow Security by Neil Vachharajani, Matthew J. Bridges, Jonathan Chang, Ram Rangan, Guilherme Ottoni, Jason A. Blome, George A. Reis, Manish Vachharajani, David I. August

Abstract:

Even as modern computing systems allow the manipulation and distribution of massive amounts of information,users of these systems are unable to manage the confidentiality of their data in a practical fashion. Conventional access control security mechanisms cannot prevent the illegitimate use of privileged data once access is granted.For example, information provided by a user during an online purchase may be covertly delivered to malicious third parties by an untrustworthy web browser. Existing information-flow security mechanisms do provide this assurance, but only for programmer-specified policies enforced during program development as a static analysis on special-purpose type-safe languages. Not only are these techniques not applicable to many commonly used programs, but they leave the user with no defense against malicious programmers or altered binaries.

In this paper, we propose RIFLE, a runtime information flow security system designed from the user’s perspective. By addressing information-flow security using architectural support, RIFLE gives users a practical way to enforce their own information-flow security policy on all programs. We prove that, contrary to statements in the literature, runtime systems like RIFLE are no less secure than existing language-based techniques. Using a model of the architectural framework and a binary translator, we demonstrate RIFLE’s correctness and illustrate that the performance cost is reasonable.

Applications

[

](https://imgflip.com/i/8o1qww)

The articles in this section discuss applications that can be built with intent-centric architectures. This includes grassroots protocols, which can loosely be defined as serverless peer-to-peer networks not reliant on centralized operators, blockchains, or intermediaries of any kind. The motivation for building these protocols is to return freedom, agency, and privacy back to the populace and allow them to participate in digital democracies of the future. Ehud Shapiro and his collaborators take a scientific approach in defining the problem space, motivations (including combatting surveillance capitalism), and protocols to bring these next generation of coordination tools to life.

This relates to Anoma because the protocol is designed intentionally for applications concerned with coordination of socio-economic processes dealing with resources, distributed capabilities (freedom of action, and control over resources), and agreement between agents (users) under conditions of heterogeneous trust.

The current economic network and organizing paradigm restricts autonomy and limits freedom. Communities have two options: to give up autonomy for the sake of interoperability - use infrastructure, protocols, and currencies operated and controlled by someone else, and thereby participate in wider economic networks - or to give up interoperability for the sake of autonomy - opt out of the shared infrastructure and produce everything themselves. Anoma aims to offer a third way - one which preserves both autonomy and interoperability.

Some examples of Anoma applications:

- Public Signal - double/sided version of Kickstarter implemented with intents. Possible to implement Alex Tabarrok’s dominant assurance contracts.
- Multichat - a distributed and encrypted chat network without any specially designated server operators.
- Promise Graph - a language and structured accounting logic for making and managing promises in and across distributed organizations.
- Scale-free money - a hypothetical monetary system design which allows anyone to create arbitrary denominations of money at any time for any purpose (inclusive of Multilateral Trade-Credit Set-off)

Please see this thread for more details.

Resources

- Enabling the Digital Democratic Revival: A Research Program for Digital Democracyby various authors

Abstract:

This white paper outlines a long-term scientific vision for the development of digital-democracy technology. We contend that if digital democracy is to meet the ambition of enabling a participatory renewal in our societies, then a comprehensive multi-methods research effort is required that could, over the years, support its development in a democratically principled, empirically and computationally informed way. The paper is co-authored by an international and interdisciplinary team of researchers and arose from the Lorentz Center Workshop on Algorithmic Technology for Democracy

(Leiden, October 2022).

- Grassroots Flash: A Payment System for Grassroots Cryptocurrencies by Andrew Lewis-Pye, Oded Naor, and Ehud Shapiro

Abstract:

The goal of grassroots cryptocurrencies is to provide a foundation with which local digital economies can emerge independently of each other and of global digital platforms and global cryptocurrencies; can form and grow without initial capital or external credit; can trade with each other; and can gradually merge into a global digital economy. Grassroots cryptocurrencies turn mutual trust into liquidity, and thus could be a powerful means for 'banking the unbanked'.

[

](https://ibb.co/tXCpJ6M)

Source

: Lewis-Pye, Naor, Shapiro, Grassroots Flash: A Payment System for Grassroots Cryptocurrencies

Grassroots cryptocurrencies have not been provided yet with a payment system, which is the goal of this paper. Here, we present Grassroots Flash, a payment system for grassroots cryptocurrencies that employs the blocklace – a DAG-like counterpart of the blockchain data structure. We analyze its security (safety, liveness, and privacy) and efficiency, to prove that it is indeed grassroots.

- Grassroots Social Networking: Where People have Agency over their Personal Information and Social Graph by Ehud Shapiro

Abstract:

Offering an architecture for social networking in which people have agency over their personal information and social graph is an open challenge. Here1 we present a grassroots architecture for serverless, permissionless, peer-to-peer social networks termed Grassroots Social Networking that aims to address this challenge. The architecture is geared for people with networked smartphones—roaming (address-changing) computing devices communicating over an unreliable network (e.g., using UDP). The architecture incorporates (i) a decentralized social graph, where each person controls, maintains and stores only their local neighborhood in the graph; (ii) personal feeds, with authors and followers who create and store the feeds; and (iii) a grassroots dissemination protocol, in which communication among people occurs only along the edges of their social graph.

[

](https://ibb.co/0G0gt5L)

Source

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

The architecture realizes these components using the blocklace data structure – a partially ordered conflict-free counterpart of the totally ordered conflict-based blockchain. We provide two example Grassroots Social Networking protocols—Twitter-like and WhatsApp-like—and address their security (safety, liveness and privacy), spam/bot/deep-fake resistance, and implementation, demonstrating how server-based social networks could be supplanted by a grassroots architecture.

zkVMs

[

](https://imgflip.com/i/8o28wr)

A zkVM is generally defined as an emulation of a CPU architecture into a universal circuit that can compute and prove any program from a given set of opcodes. zkVMs arise partially because writing ZK circuits is hard, meaning that it requires expertise and its easy-to-make mistakes. In addition, different circuits require different verifiers, and involve a preprocessing step per circuit.

The primary reason for the emergence of zkVMs is that they provide a seamless developer experience for writing “ZK applications” without any exposure to cryptography - the cryptography is abstracted away. For example, zkVMs allow for any program to be compiled to a defined set of opcodes, which can easily be audited.

In particular, developers can write programs in high-level languages like Rust or Juvix and compile them to ZK circuits, allowing for the use of succinct proofs and/ or zero knowledge in their applications. For Anoma and other intent-centric protocols, this is critical because it will allow satisfaction of intents to be verified on any state machine. zkVMs can also support function privacy, allowing for privacy in counterparty discovery and settlement. zkVMs have significant implications for the future of privacy-preserving applications and interoperability.

In the cited material below, we highlight recent research by Alberto Centelles where he explores different zkVMs.

Resources

- Anoma Research Topic: Compiling to zkVMs by Alberto Centelles

Abstract:

With the advent of non-uniform folding schemes, the lookup singularity, generalized arithmetizations such as CCS and the application of towers of binary fields to SNARKs, many of the existing assumptions on SNARKs have been put into question, and the design space of zkVMs has opened.

Although zkVMs provide a friendly developer experience, their proving time is still significantly (around a million times) slower than direct compilation to circuits due to the overhead of their abstractions (stack, memory, execution unit, etc).

[

](https://ibb.co/MCCqNxg)

Source

: Alberto Centelles, Compiling to zkVMs: Parallelization, decoupling and abstractions

One of the causes of their poor performance is that existing zkVMs are still program-agnostic; their provers haven’t leveraged the structure of a program. Compilers have a long history of optimizing computations by identifying patterns in their structure. We take advantage of the fact that a program is generally executed before it is proven, so the prover of a zkVM is aware of execution trace before establishing a proving strategy. We explore different ways zkVMs may benefit from identifying identical sub-circuits (data-parallel circuits) in programs by analyzing techniques such as the GKR protocol, uniform compilers and proof-carrying data (PCD).

Formal Methods

[

](https://imgflip.com/i/8o1rld)

While we’re busy building Anoma, there is an ongoing effort to make Anoma and its components more formal. In particular, we want to make things more mathematically precise as this precision is essential for understanding and implementing Anoma.

Why are formal methods important? Formal Methods involve using math to contribute to the reliability and robustness of the software (code) and / or hardware design specifications. Formal models are useful for creating specifications, verifying the desired properties of the system, creating documentation, and guiding the implementation process.

For example, TLA + is a formal specification language developed by Leslie Lamport for the purpose of designing, modeling, writing documentation, and verifying programs. TLA+ is particularly useful for concurrent systems and distributed systems.

In the context of Anoma we are using Isabelle, a higher-order logic (HOL) theorem prover written in Standard ML and Scala. Isabelle can be seen as an IDE for formal methods, which we are using to create a formal specification of the Anoma protocol.

Resources

- Formalizing Concurrent Programs; the generalities by Anthony Hart

Summary:

One of the core goals when undertaking this project was to understand how to take a concurrent program and formalize it mathematically. In general, what does this look like? Our expectation was that things would differ significantly from more usual algorithm formalizations, but, actually, it ended up not being as novel as we thought. In this post, we want to describe

the general idea of formalizing concurrent programs, some different approaches, and some connections to more abstract mathematics. For this post, we will not explain HPaxos in any detail, but we will use some of its implementation in enough detail to demonstrate some broad points. You don't need to understand distributed consensus to understand this post; it's just my inspiration.

Feedback & Conversation

[

](https://imgflip.com/i/8o26q6)

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 tointentnewsletter@proton.me

E-mail Signup

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