R3 Reports

Top Ten Obstacles Along Distributed Ledgers' Path to Adoption

Sarah Meiklejohn

Contents

R3 Research aims to deliver concise reports on DLT in business language for decision-makers and DLT hobbyists alike. The reports are written by experts in the space and are rooted in practical experience with the technology.

10. Usability: Why use distributed ledgers? **1**
9. Governance: Who makes the rules? **2**
8. Meaningful comparisons: Which is better? **2**
7. Key management: How to transact? **3**
6. Agility: Which algorithms do we use? **4**
5. Interoperability: How to talk to each other? **4**
4. Scalability: Why store every transaction? **5**
3. Cost-effectiveness: What is the cheapest way? **5**
2. Privacy: How to protect data? **6**
1. Scalability: Do we need full agreement? **7**

For more Research, please visit R3's Wiki here.

Top Ten Obstacles along Distributed Ledgers' Path to Adoption

Sarah Meiklejohn

December 14, 2017

In January 2009, Bitcoin was released into the world by its pseudonymous founder, Satoshi Nakamoto. In the ensuing years, this cryptocurrency and its underlying technology, called the *blockchain*, have gone on a rollercoaster ride that few could have predicted at the time of its deployment. It's been praised by governments around the world, and people have predicted that "the blockchain" will one day be like "the Internet." It's been banned by governments around the world, and people have declared it "adrift" and "dead." The price of Bitcoin skyrocketed in late 2013 up to $1200 per bitcoin, only to spend the entire next year languishing at anywhere from $200 to $500 per bitcoin, before beginning a steady climb in 2016 that now has Bitcoin's price hovering close to its earlier peak.

After years in which discussions focused entirely on Bitcoin, people began to realize the more abstract potential of the blockchain, and "next-generation" platforms such as Ethereum, Steem, and Zcash were launched. More established companies also realized the value in the more abstract properties of the blockchain — resilience, integrity, etc. — and repurposed it for their particular industries to create an even wider class of technologies called *distributed ledgers*, and to form industrial consortia such as R3 and Hyperledger.

Amidst many unknowns, it is increasingly clear that, even if they might not end up quite like "the Internet," distributed ledgers — in one form or another — are here to stay. Nevertheless, a long path remains from where we are now to widespread adoption, and there are many important decisions to be made that will affect the security and usability of any final product.

In what follows, I'll present the top ten obstacles along this path, as well as what we as a community can do (and have been doing) to address them. All views are entirely my own but are informed by my role as a long-standing and active participant in this research community, and to avoid turning this document into a long book I have by necessity omitted mention of many interesting solutions. The problems are ranked very roughly from least to most urgent — or more accurately, from broadest to most specific — but widespread, productive, and long-lasting adoption is probably impossible without at least considering all of them.

10 Usability: why use distributed ledgers?

The problem. What does usability refer to, and how is it a problem? There are a few possibilities in this context; for example, one could say that currently deployed distributed ledgers do not have particularly nice interfaces and are difficult to use without expert knowledge. While this is certainly true, one can and should expect these problems with any new technology, and further expect them to be resolved with time and increased usage. (Here relational databases and the development of graphical, if imperfect, user interfaces into them might serve as a useful history.)

Thus, a perhaps more interesting 'usability' problem is: what do end users actually want from distributed ledgers? Most deployed cryptography that end users interact with today maps easily to processes and desires that have existed for centuries. If you want to hide the contents of your conversation with someone, use encryption. If you want to convince someone that it's really you

saying something, use a (digital) signature. But what is it that the full public verifiability (or accountability, immutability, etc.) of distributed ledgers really maps to in terms of what end users want?

Potential and developing solutions. To the best of my knowledge, there has been very little research into this thus far, with most research focusing on the — admittedly more urgent — architectural problems discussed later in this list. It may be that in fact most end users have no interest in the properties of distributed ledgers (just as they also likely have no interest in the properties of regular databases). Given the growing number of users who already interact with platforms like Bitcoin, however, and the discussed potential for distributed ledgers to — among other things — inform the choices of ethical consumers, it is still worth exploring which specific benefits people feel they can achieve by using these technologies.

9 Governance: who makes the rules?

The problem. The beauty of distributed ledgers is that no one entity gets to control the decisions made by the network; in Bitcoin, e.g., coins are generated or transferred from one party to another only if a majority of the peers in the network agree on the validity of this action. While this process becomes threatened if any one peer becomes too powerful, there is a larger question looming over the operation of these decentralized networks: who gets to decide which actions are valid in the first place? The truth is that all these networks operate according to a defined set of rules, and that "who makes the rules matters at least as much as who enforces them" [8].

In this process of making the rules, even the most decentralized networks turn out to be heavily centralized. In Bitcoin, the rate at which new blocks are added to the chain, the reward participants receive for sealing transactions into the ledger, and many other parameters were all handed down by Satoshi Nakamoto and have not changed since. Ethereum has now carried out four so-called "hard forks" of its network, in which participants are essentially mandated to switch to a new version of the software to comply with new rules created by a core set of developers.

One of the most interesting effects of these increasingly common collapses in the governance structures underlying cryptocurrencies is that they seem to be the first time many people even realize they exist. Thus, the problem is not just that we don't know how to govern these technologies, or that the governance structures are centralized, but that — somewhat ironically — we need more transparency around how these structures operate and who is responsible for which aspects of governance.

Potential and developing solutions. Several research papers have focused on ways to maintain decentralization in terms of the entities that enforce the rules, although in more general distributed ledgers it is still very much an open question how to incentivize participants to help maintain the ledger. (For example, it is unclear who, other than Google or other large certificate authorities, would want to expend the resources necessary to maintain a log server in Certificate Transparency, which provides no explicit reward to these entities.)

In terms of the entities that make the rules, very few solutions have been proposed. The "Satoshi Oath" [2] outlines a pledge that the authors deem necessary for anyone setting out to create a blockchain-based application, but it still serves only as a guideline. Broader distributed ledger projects such as R3 and Hyperledger have opted for a consortium-based approach, and other projects such as Certificate Transparency have opted for a fairly centralized approach. As in Item 8, however, it is unclear how these approaches compare to each other, or how these structures will evolve as these platforms gain in popularity.

8 Meaningful comparisons: which is better?

The problem. Bitcoin was the first cryptocurrency to be based on the architecture we now refer to as the blockchain, but it certainly isn't the last; there are now thousands of alternative cryptocurrencies out there, each with its own unique selling point. Ethereum offers a more expressive scripting language and maintains state. Litecoin allows for faster block creation than Bitcoin. Dogecoin has a cute dog! Looking beyond blockchains, there are numerous proposals for cryptocurrencies based on alternative consensus protocols — proof-of-stake, PBFT, two-phase commit,

etc. — and proposals in non-currency-related settings, such as Certificate Transparency, R3 Corda, and Hyperledger Fabric, that still fit under the broad umbrella of distributed ledgers.

The natural issue that arises in such an increasingly crowded landscape is how to distinguish between these solutions and pick the one that is best for a given application. Do you need a blockchain, or just a database? Maybe an Excel spreadsheet is sufficient for what you need to do? Related back to Item 10, what even are the properties that you want to satisfy? Do you need full public verifiability (and if so, why)? Even if someone could specify a list of necessary properties, it is not still not clear which current platforms support which properties, and to what extent.

Potential and developing solutions. Some recent research [6] has looked at the different parameters chosen by different cryptocurrencies (e.g., Bitcoin's 10-minute block generation vs. the 2.5-minute interval used by Litecoin) and the effect these parameters have on the security of the system, finding, for example, that the same level of security against so-called "selfish mining" attacks is achieved by 37 blocks in Ethereum as by 6 blocks in Bitcoin (due to the relative stale block rates of the two platforms). While insightful, this work focuses specifically on cryptocurrencies based on proof-of-work, and thus does not extend to more general distributed ledgers or cryptocurrencies based on alternative consensus protocols (see Item 3 for examples).

In considering distributed ledgers as a whole, recent research [3] explored the connections between the security properties provided by Certificate Transparency and Bitcoin, finding that the tradeoff between the two seemed to be between trust in a distributed set of authorities on the one hand and the need to (inefficiently) broadcast messages and engage in consensus protocols like proof-of-work on the other hand. This research again focuses on a specific underlying structure, so significant work remains to allow comparisons between other platforms, or more generally to understand the set of tradeoffs that one should consider in evaluating any particular choice.

7 Key management: how to transact?

The problem. Here's a fun confession: in early 2013, I was using Ubuntu on my desktop at work. For the previous four years I had been performing upgrades periodically, but — as happens frequently — the system had reached a point where there were enough dangling threads to affect its functioning, so I decided to do a clean re-installation of the operating system. Thinking that all I had on my desktop were repositories under version control anyway, I performed the installation without backing up first. Not five minutes after the installation completed did I remember the horrible fact that I had my Bitcoin wallet on my desktop. Luckily I had very few bitcoins stored on it, but over the years many others have reported similar incidents in which they lost far more — 7,500 bitcoins in the worst publicly reported incident — and the point is that once the wallet is lost or your money is stolen, the irreversibility of both the underlying cryptography and the transactions mean that there's no way to recover. In one very public example, after an attack exploited a loophole in the code of the smart contract behind The DAO, the community was essentially powerless to do anything except watch them steal all the funds stored inside (until, that is, some of the governing Ethereum developers discussed in Item 9 created and advocated for a hard fork to undo the damage).

While these incidents obviously have serious financial implications for the individuals involved in them, they should in some sense not be particularly surprising to these individuals given the unregulated "Wild West" world of cryptocurrencies. As people are proposing more mainstream uses of cryptocurrencies — e.g., integrating Bitcoin wallets into Linux distributions — that will put these wallets in the hands of less experienced users who may be less prepared for these types of risks, however, this will lead to wider and more serious consequences. We thus need robust solutions that are able to better tolerate both the loss and the theft of keys.

Potential and developing solutions. A commonly used Bitcoin technique that has the potential to mitigate this issue is a *multisignature*, in which multiple parties join their public keys together to create an address for which some subset of their signatures is sufficient to spend its contents. For example, in a 3-of-3 multisignature address all three parties would need to sign, and in a 1-of-3 address any one of the signatures would suffice. A 2-of-3 multisignature address arguably provides the best defense against key loss, as funds stored in that address are still accessible even if one key is lost (and as a bonus, an attacker needs to access two separate keys to steal from it).

One could similarly argue that secret sharing, in which the key is split among some number of friends (or devices) who can be trusted to not collude but to provide their shares if and when key loss occurs, could also help. With both of these solutions, however, we must understand if the threat model in which they work is realistic for the scenario (going back to Item 10) in which we expect to be distributing, storing, and using these keys. It is also still an open problem to identify solutions in which one could regain access to lost funds with the same ease with which users currently regain access to accounts for which they have forgotten the password.

6 Agility: which algorithms do we use?

The problem. As discussed in Item 9, in many cryptocurrencies the rules seem to have been handed down from on high: Bitcoin uses ECDSA over curve `sec256k1`, even if it seems like a weird choice. Bitcoin addresses are computed by taking the ECDSA public key, performing SHA-256, performing extended RIPEMD-160, performing SHA-256 again, and again, rearranging the bytes of this output and the output of the extended RIPEMD-160 hash, and converting the result into a base58 string. Once the governing developers have decided on a good proof-of-stake protocol (more on that in Item 3, that will replace Ethash as the consensus protocol in Ethereum).

Aside from the governance issues raised by these rigid specifications, cryptographic primitives break, and they do so frequently. Eventually, computers may become powerful enough that SHA-256 will not be considered secure. Maybe quantum computers will come along soon and allow anyone who has one to forge ECDSA signatures. While there will likely be sufficient warning before either of these events to give the developers time to switch to more secure alternatives (and anyway the problem would go far beyond distributed ledgers!), there is an argument to be made for providing users with multiple options even today, just as is done with systems such as TLS. As with TLS, however, agility can enable dangerous attacks, so significant caution is needed in both how it is implemented and in which cryptographic primitives are supported.

Potential and developing solutions. Almost all cryptocurrencies achieve no agility whatsoever: for each cryptographic primitive, there is one valid instantiation. One arguable exception is Ethereum, in which one could technically encode any cryptographic primitive — even something that is completely insecure — into a smart contract by including a custom library within the contract code. Nevertheless, the underlying cryptography (i.e., the checks that peers in the network make to verify individual transactions) are just as rigid as in other cryptocurrencies.

Perhaps because they are unencumbered by the ideological mindset frequently encountered in cryptocurrencies, industry-led distributed ledgers are much more agile. In Corda, for example, an individual user can pick from a selection of available algorithms when forming a contract. In Hyperledger Fabric, participants can essentially plug in their own consensus protocol. Here then, the question is not necessarily whether or not it can be done at all, but how these different options can compose (e.g., what does it mean if two parts of the ledger have been agreed upon by two distinct consensus protocols?) and what effect these options have on the overall security of the system.

5 Interoperability: how to talk to each other?

The problem. Some people believe that the future will contain one single ledger (like "the Internet"), and in fact general-purpose platforms could in theory — i.e., if the other issues on this list were solved — support most known applications of distributed ledgers. The more likely scenario, however, is that different companies will gravitate toward different ledgers based on their particular requirements. For example, financial applications requiring consensus between a fixed set of banks are more likely to adopt a platform such as Corda or Hyperledger Fabric, whereas applications that require fully open participation are more likely to adopt a platform such as Bitcoin or Ethereum.

To achieve the much-discussed potential of distributed ledgers to eliminate (or at least significantly open) proprietary silos, it is thus essential to achieve some kind of *interoperability*, or a set of methods that allow these disparate ledgers to talk to each other.

Potential and developing solutions. Within the realm of platforms based on blockchains, the idea of a *sidechain* provides a simple way to translate actions from one blockchain into another;

i.e., to have transactions published in one ledger have an effect on another ledger. The security of these sidechains has been relatively unstudied to date, however, and while they have attracted significant attention they have not yet seen much adoption.

Beyond this, there have been a number of attempts to provide interfaces between distributed ledgers and real-world data, such as Town Crier [11] and Oraclize.it. Otherwise, both Corda and Hyperledger Fabric attempt to be modular in their choice of protocols (see Item 6), which means that they are designed to interoperate across different usages of the overall system, but to the best of my knowledge there has been little attempt amongst existing solutions to interface with each other.

4 Scalability: why store every transaction?

The problem. Scalability can mean a lot of things. Item 1 discusses the need to scale the time it takes to process individual transactions with the computational power that is added to the network. Another aspect of scalability is that, as the system becomes more popular, it should not significantly increase the storage load placed on users of the system, as this deters participation and increases the barrier to entry. For integrity purposes, however, it is important in systems in which we expect full public verifiability (discussed further in Item 1) to not delete entries from the ledger. The main problem lies in how to provide a balance between these two (seemingly contradictory) requirements.

In certain applications, it may be sufficiently important to provide full auditability (for example, to satisfy regulatory requirements) that we cannot help but increase the storage load of participants. Especially in consumer applications, however, a relatively compelling case can be made that is is not actually necessary to store every transaction, as — for example — we are unlikely today to want or need to meaningfully examine someone's coffee purchases from early 2010. In fact, retaining such information over a long period of time is clearly at odds with the privacy of users (Item 2).

Additionally, in a slightly altered trust model, more "casual" users may be willing to offload the work of auditing the system to more "archival" users; thus, while archival users would be required to store the entirety of the ledger, casual users could store only the information needed to check the validity of their own personal transactions.

Potential and developing solutions. The main innovation that allows Bitcoin wallets to run on smartphones — crucially, without storing the 100GB that currently constitutes the full Bitcoin ledger — is the idea of an SPV client, which is analogous to the casual user mentioned above. These clients retrieve from archival peers and store only the headers of blocks along the blockchain, rather than their full contents, and rely on these headers to check for double spending (SPV stands for Simplified Payment Verification). While such clients have gained wide adoption, their security — and in particular their privacy — is imperfect and somewhat poorly understood.

Another major development that would help prevent excessive transactions from being included in the ledger is the Lightning network, and more generally a recent (and increasingly active) line of research that allows two parties to open a *payment channel*. Using such a channel, a single pair of transactions can be placed in the ledger — that, respectively, serve to open and close the channel — but many individual payments can take place. Again, while this approach seems promising and has attracted significant attention, the security and privacy aspects of using such channels are still not well understood.

Finally, another promising approach is the idea of *sharding* the ledger, as discussed in Item 1. If implemented in a certain way, participants do not need to store (or even hear about) transactions that are irrelevant to them; e.g., I don't need to store everyone's morning coffee purchases, just my own. While this significantly reduces the storage requirements of an individual user, it is still important to understand the practical costs of storing what is ultimately a monotonically growing ledger.

3 Cost-effectiveness: what is the cheapest way?

The problem. People like to complain a lot about Bitcoin and its proof-of-work-based consensus algorithm. They compare the electricity it uses to the electricity used by whole nations, or at least by large power plants. They call it an "environmental disaster." While many of these arguments

are quite overblown, the fact remains that proof-of-work is indeed very expensive, and it would of course be a good thing if the same result could be achieved in a cheaper way.

To this end, there have been a huge number of alternative consensus protocols proposed, and even used in alternative cryptocurrencies [1, Section VI]. Peercoin, Nxt, and BlackCoin all use a version of *proof-of-stake*, and Ethereum has a planned transition (i.e., hard fork) to their own version of proof-of-stake, Casper, within the next two years. Intel's Sawtooth Lake platform uses proof-of-elapsed-time (PoET), which relies on its SGX architecture. Permacoin [10] proposes repurposing the mining process to perform useful work, such as archiving important information. One of the most popular Bitcoin-based proposals, Bitcoin-NG [5], proposes isolating the usage of proof-of-work to elect a periodic "leader," who can then quickly (i.e., without performing large amounts of computation) bear witness to individual transactions.

If one expands outwards to general distributed ledgers, the list of alternative consensus protocols grows and begins to include more familiar algorithms: two-phase commit, Raft, PBFT, etc. These tend to be significantly more efficient than the "proof-of-X" protocols used in cryptocurrencies, but with the tradeoff that they require a fixed set of known participants. As in Item 8, what is needed is a way to understand this tradeoff, provide meaningful comparisons within this growing landscape of consensus protocols, and understand the benefits that each provides in a given setting.

Potential and developing solutions. In the distributed ledger research community, this seems to be the issue that is being explored most actively. In addition to the protocols mentioned above, there are dozens more articles with their own proposals, both for the protocols mentioned above (especially proof-of-stake, which seems to be a sort of "holy grail" as of this writing) and for new and increasingly exotic ones. Thus, of all the items on this list, this seems to be the problem for which I most expect to see a viable set of solutions in the near future.

2 Privacy: how to protect data?

The problem. While significant research has focused on the *anonymity* of users in cryptocurrencies (i.e., protecting the identity of participants), little research has focused on their *privacy*. For example, Bitcoin users are technically "pseudonymous," as their on-chain identities are not inherently linked to their real-world identity, but the details of each transaction — e.g., the amount of bitcoins being sent — are still completely transparent. If we consider more expressive platforms and more exotic use cases, the desire to store health records on a distributed ledger means transactions would contain information about a patient's name (i.e., their real-world identity), their age, the nature and justification of medical procedures, etc. While a naïve solution would be to encrypt this data and provide decryption keys only to the necessary parties, the fact is that encryption schemes, as with all cryptographic primitives, break (see Item 6) or are compromised, so this does not provide a long-term solution for privacy.

Even with respect to anonymity, there's still a lot of work to do. A long line of research has demonstrated the limitations of Bitcoin's anonymity [1, Section VII], and while emerging platforms such as Monero and Zcash promise improvements, they also likely have their own weaknesses that will emerge with increased adoption and patterns of usage that the protocol specification did not consider.

Potential and developing solutions. One of the simplest solutions is to send transactions to only those participants with whom you trust the associated details; this is akin to the sharding-based solutions we explore in Item 1. Even here, however, it is still important to consider privacy, as transactions may be shared beyond the initial set of participants, or it may be necessary to hide certain details from one participant but reveal them to another.

One general solution that has been proposed is Hawk [7], in which users can hide the details of their transactions but still convince the rest of the network that the transactions are valid. While useful, Hawk is specific to Ethereum and makes use of fairly advanced cryptography. Thus, it is still very much an open question how to achieve privacy in a lightweight and flexible manner for general distributed ledgers.

1 Scalability: do we need full agreement?

The problem. Arguably the biggest hurdle for fully distributed ledgers is the insistence that every node in the network needs to agree on the full state of the entire ledger. Aside from the issues with this approach raised earlier (for example, in Item 4), this means that distributed ledgers cannot and do not scale in terms of their ability to process growing numbers of transactions (*throughput*) while still ensuring that users do not have to wait for their transactions to be included (*latency*). In other words, the more computational power that joins the network, the worse it will perform in terms of throughput and latency. This is precisely because of this requirement that every node must agree on every transaction, as it means the more transactions are in the system, the longer the nodes must wait for them to flood the network.

Since this insistence on full replication thus violates one of the most basic properties of distributed systems, we might naturally ask ourselves: why do it in the first place? One of the primary benefits of cryptocurrencies, enabled by this requirement, is the idea of full public verifiability: any participant can verify for themselves the correct functioning of the system by, for example, replaying all transactions and ensuring that any agreed-upon rules haven't been violated. If only certain nodes agree on certain parts of the ledger, then there is no one participant that can satisfy this notion of verifiability. To allow for increased throughput and decreased latency, one must therefore provide a balance between avoiding full replication — which is often accomplished using a technique called *sharding*, in which each participant sees transactions only within a given shard — and enabling at least some degree of openness and verifiability.

Potential and developing solutions. This topic has received significant attention, and a number of academic proposals adopt some form of sharding [4, 9]. Many industrial proposals similarly adopt a type of sharding; e.g., in Corda, participants need to achieve consensus only on transactions that are directly relevant to them, and in Certificate Transparency there is similarly no global consensus on the contents of the ledger. While these approaches achieve significantly better scalability, they raise questions about verifiability that fully decentralized solutions do not. For example, if only certain participants see certain transactions, how can other participants tell that their transactions obey the global set of rules? In the absence of such a global set of rules, what meaningful notions of integrity can we even satisfy? As with all the items on this list, we again see that each platform does not provide one uniquely perfect solution, but rather a set of tradeoffs that — to come full circle back to Item 10 must ultimately be balanced according to the individual use case in which the technology will be used.

References

[1] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten. Research perspectives and challenges for Bitcoin and cryptocurrencies. In *Proceedings of the IEEE Symposium on Security and Privacy*, 2015.

[2] J. Brekke and E. Haase. Satoshi Oath, 2016. `http://ipfs.b9lab.com:8080/ipfs/QmXysWEAexXQqYZhTGpECvksnaBkSEWHdGhM7vNeHxue2g/`.

[3] M. Chase and S. Meiklejohn. Transparency overlays and applications. In *Proceedings of ACM CCS 2016*, 2016.

[4] G. Danezis and S. Meiklejohn. Centrally banked cryptocurrencies. In *Proceedings of NDSS 2016*, 2016.

[5] I. Eyal, A. E. Gencer, E. G. Sirer, and R. van Renesse. Bitcoin-NG: a scalable blockchain protocol. In *Proceedings of NSDI 2016*, 2016.

[6] A. Gervais, G. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun. On the security and performance of proof of work blockchains. In *Proceedings of ACM CCS 2016*, 2016.

[7] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In *Proceedings of the IEEE Symposium on Security and Privacy*, 2016.

[8] V. Lehdonvirta. The blockchain paradox: Why distributed ledger technologies may do little to transform the economy, 2016. `http://blogs.oii.ox.ac.uk/policy/the-blockchain-paradox-why-distributed-ledger-technologies-may-do-little-to-transform-the-economy/`.

[9] L. Luu, V. Narayanan, K. Baweja, C. Zheng, S. Gilbert, and P. Saxena. A secure sharding protocol for open blockchains. In *Proceedings of CCS 2016*, 2016.

[10] A. Miller, A. Juels, E. Shi, B. Parno, and J. Katz. Permacoin: Repurposing Bitcoin work for data preservation. In *Proceedings of the IEEE Symposium on Security and Privacy*, 2014.

[11] F. Zhang, E. Cecchetti, K. Croman, A. Juels, and E. Shi. Town Crier: an authenticated data feed for smart contracts. In *Proceedings of CCS 2016*, 2016.

R3 is an enterprise software firm using distributed ledger technology to build the next generation of financial services infrastructure.

R3's member base comprises over 80 global financial institutions and regulators on six continents. It is the largest collaborative consortium of its kind in financial markets.

Consortium members have access to insights from projects, research, regulatory outreach, and professional services.

Our team is made of financial industry veterans, technologists, and new tech entrepreneurs, bringing together expertise from electronic financial markets, cryptography and digital currencies.

corda is an open source, financial grade distributed ledger that records, manages and executes institutions' financial agreements in perfect synchrony with their peers.

Corda is the only distributed ledger platform designed from the ground up to address the specific needs of the financial services industry, and is the result of over a year of close collaboration between R3 and its consortium of over 80 of the world's leading banks and financial institutions.