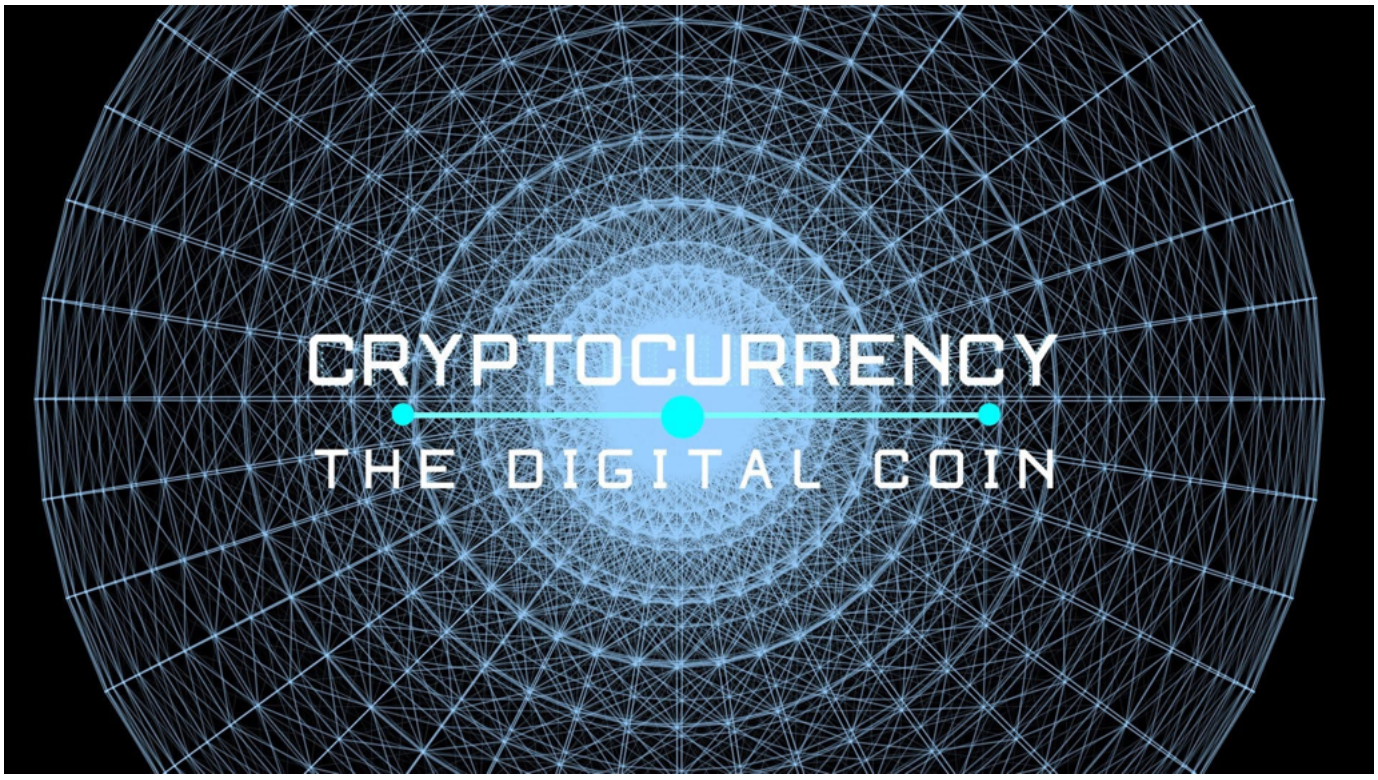
Bitcoin, Ethereum, Blockchain, Tokens, ICOs: Why should anyone care?

July 4th 2017 ★ 84,713 reads 4 



@preethikasireddy
Preethi Kasireddy

The Crypto market is gaining lots of steam. Gravity-defying price rallies…

Bitcoin Price Chart (1 year)

Ethereum Price Chart (1 year)

Top 8 Cryptocurrencies Price Chart (excluding Bitcoin & Ethereum)

…and multi-million dollar token sales are commonplace, as are front-page headlines from traditional news outlets discussing Ethereum, Bitcoin, ICOs, tokens, hard forks, and other technical topics.

image

Even my 13-year-old brother has been calling me up asking for explanations!

I've been personally invested in this space for a while now — most recently as an engineer for Coinbase — but even I'm surprised by how quickly the Crypto space has evolved in the past six months. If you want to understand why crypto is getting the spotlight, you have to understand the behind-the-scenes catalysts driving the market. Right now, that catalyst is the "token sale" or "Initial Coin Offering (ICO)" phenomena.

ICO vs IPO: The wild west of investment

What the heck is an ICO anyway? You may have heard of an “Initial Public Offering” (IPO) — when a company *goes public* by selling some of its shares to institutional investors, who in turn sell to the general public on the securities exchange. The public gets excited about IPOs because they let anyone with a brokerage account purchase shares of companies like Snapchat. Are ICOs the same thing? Yes and no. IPOs and ICOs are both used by companies to raise capital. The main (and really important) difference is regulation. IPOs are regulated by the SEC and have a set of legal requirements and a formal process for how they’re carried out. ICOs are currently unregulated and more of a “wild west” practice. Overall, there seems to be a lot of confusion and uncertainty when it comes to ICOs. Some argue that they have turned into a “perverse and unsustainable Keynesian beauty contest.” Supporters are optimistic and claim that it’s a new form of Venture Capital. With drastically opposing viewpoints like this dominating the conversation, most of us are left on the sidelines scratching our heads.

Necessary background knowledge

You can’t understand ICOs without understanding the underlying digital asset sold in an ICO. If you already know the basics of crypto, feel free to skip this section. For the rest of us… let’s start from the top!

The Cryptocurrency technology stack

Cryptocurrency technology stack

Bitcoin

Bitcoin is a decentralized digital currency that uses a peer-to-peer technology. Peer-to-peer essentially means that there isn't a central authority issuing new money or tracking transactions. Instead, these operations are managed collectively by the network. The transactions happen between users directly and are recorded on the *blockchain* (more on that below). The Internet is filled with great Bitcoin explainers, so I won't delve much farther

down the rabbit hole in this post. Instead, here are some starting points to get you up to speed:
Bitcoin Wiki, Wikipedia, What is Bitcoin, Bitcoin Magazine, Why Bitcoin matters.

Blockchain

A blockchain is a decentralized public database that keeps a permanent record of digital transactions. In other words, it's a logfile storing an immutable record of all the digital transactions. This decentralized database is not controlled by a central administrator, but instead is a network of replicated databases (meaning each node in the network stores its own copy of the blockchain) that is shared and visible to anyone within network. Every "block" in this blockchain contains a record of recent transactions, a reference to the block that came immediately before it, and an answer to a difficult mathematical puzzle, among other things.

A blockchain is collectively maintained by “miners”, who are members within the network that compete to validate Bitcoin transactions in each block by solving the complex algorithmic problem associated with the block. They do this by buying or renting lots of computing power to run these complex algorithmic problems on. The incentive for them to use their computing power to verify transactions is that they are rewarded with Bitcoin if they solve the problem and validate a Bitcoin block. The power of such a decentralized network is that economic value and governance are distributed among the network’s stakeholders (i.e. miners and

consumers) rather than concentrated in a single organization (e.g. banks, governments & accountants). Thanks to this setup, anyone can own and transfer assets digitally without the need for a third party.

Blockchain technology isn't limited to Bitcoin. It can be used to create any other cryptocurrency, such as Ethereum and Litecoin, which utilize their own blockchains. You can read more about bitcoin and blockchain at Wikipedia or watch this great explainer video.

Protocol layer

Next, we have the protocol layer. In general, a protocol is the special set of rules that nodes in a network use when they transmit information. These rules specify the interactions between the communicating entities.

One example of a protocol used in telecommunications is Transmission Control Protocol (TCP), which is a set of rules for exchanging messages at the information packet level on the internet. TCP guarantees that the data packets will be delivered and that they will be delivered in the same order in which they were sent. Another example of a protocol is Internet Protocol (IP), which is a set of rules to send and receive messages at the Internet address level — it essentially specifies the format of the data packets on the internet and the addressing scheme.

TCP/IP Protocol

When discussing blockchains, the term “protocol” refers to the “cryptoeconomic rules” that are enforced by a blockchain in order to maintain distributed consensus across the blockchain’s peer-to-peer network.

Cryptoeconomic rules are rules that govern a decentralized digital economy that:

(i) uses public key cryptography for authentication

(2) has economic incentives to ensure that the rules are followed

For example, in the case of Bitcoin’s blockchain, it has financial incentives that are provided to the miners for validating every Bitcoin transaction and in turn, securing the network.

What exactly are these financial incentives?

Tokens

Enter tokens.
The financial incentive for miners comes from the native token built on top of the Bitcoin blockchain — Bitcoin. The coin serves as a "carrot and stick" — miners who use their computing power to validate transactions are rewarded with a certain amount of coin.
In general, when you hear the term "cryptocurrency tokens" or simply "tokens", they are referring to tokens such as Bitcoin that are built on top of a blockchain and represent a digital asset which you own and can transfer to someone else.

There are various ways to create tokens on top of a blockchain. For example, the simplest tokens to understand are intrinsic tokens like Bitcoin, which is directly built on top of the Bitcoin blockchain. Or you can choose to fork the Bitcoin blockchain and build tokens on top — some examples include ZCash, Litecoin, Monero, and others. **Or you can build an entirely new blockchain technology and build a token on top of that — which is what Ethereum did**. The token on top of Ethereum's blockchain is "Ether". …you can even build tokens on top of Ethereum's blockchain itself. Gnosis (GNO) and Augur (REP) are examples of this. Perhaps confusing since "Ether" is the intrinsic token built on top of the Ethereum blockchain. I'll explain later in the post. For now, just accept the fact that it's possible to build other tokens besides the intrinsic token on the Ethereum Blockchain.

There's a helpful analogy here with traditional currencies — you can think of tokens as the currency itself (e.g. USD, EUR, etc.) and the blockchain protocol as the monetary policy. The main takeaway here is that **every token is based on some underlying blockchain** — whether it's Bitcoin's blockchain, Ethereum's blockchain, or some other forked/new blockchain.

Regardless of the cryptocurrency in question, tokens are valuable because the blockchain provides a backbone for asset manipulation that is immutable, decentralized, and impossible to counterfeit.

The application layer

So far, we've learned about Bitcoin and the underlying blockchain that enables it. We've also learned about the protocol that determines the rules of the blockchain, and the tokens built on top of it.
Together, these technologies have made us rethink our definition of money as something that is digital, easily transferrable, secure, and decentralized.
But the important part to realize is that money is just one application of the blockchain. Besides money, the reason so many of us in the crypto-world are nerding out about the blockchain is because it has revealed a potential future for **(1) protocols** and **(2) applications** in general.

(1) Protocols The ultimate dream of cryptocurrency developers is that we can take advantage of this blockchain technology to build new and improved communication protocols from the ground up. Protocols being developed for cryptocurrencies have the potential to solve problems with centralization that have plagued the Internet since the first dial-up modem whirred and beeped into action. What are examples of such protocols?

Well, they could include protocols for payments, identity, domain name systems, cloud computing, reputation systems, and much more. Many of these systems today are highly centralized (e.g. Stripe, Paypal, Google, Amazon) and there's no such thing as defaults or standards for these things on the Web. Hence, in the long term, our hope is that the blockchain technology will enable decentralized, open, and secure protocols to be built with

use cases *far* outside cryptocurrency.

(2) Applications Blockchain enables what we call “decentralized applications”. Decentralized application, or “dApp”, is an application built on top of the blockchain. How does that work? Let’s consider the Bitcoin blockchain as an example. Bitcoin uses a scripting system for transactions that occur on the Bitcoin blockchain. A script is a simple list of instructions. So the Bitcoin’s scripting language enables us to write a script that is recorded with every transaction. The purpose of the script is to define the requirements the recipient must meet to gain access to the Bitcoins being transferred.

For a typical Bitcoin transfer, the script will define what the spender must provide:

1. a public key that, when hashed, matches the destination address included in the script
2. a signature to show evidence of the private key corresponding to the public key just provided

But the neat thing is that there's some flexibility in the parameters we can send with each transaction. For example, we can write a script that says "this transaction is only valid if it has two private keys". So essentially, this scripting language now lets us encode rules for how to move money, or more generally, any piece of information, around, without requiring us to trust some third party to follow a set of rules we care about. We simply trust the code and all is well.

Because Bitcoin has this scripting language, it's possible to use this language to build certain types applications that transact on the blockchain. **In other words, we can build applications that use Bitcoin transactions to communicate.**

For example, let's say we want to build a blockchain-based crowdfunding application. You might have a set of rules for how funds are transferred (or communicated) between one party

to another which you encode in the scripting language. Then users of the application can run a crowdfunding event that is governed by the blockchain.

This is the main idea behind dApps: a decentralized set of rules that define a specific application. This set of rules sits on a public and decentralized blockchain (instead of a central server owned by some large entity, such as Facebook or Amazon). This enables it to governed by autonomy and be resilient to censorship.

“Where are the apps?” dilemma

Many of us in the crypto world were under the impression that developers would immediately hop on the bandwagon and use Bitcoin’s scripting language to build decentralized applications on top. But fast forward eight years (Bitcoin was released in 2009), and Bitcoin has yet to become more than simply a store of value and a speculative investment. Sure, we’ve seen a handful of wallets and exchanges built. (Coinbase, Kraken, Poloniex, and GDAX, to name a few.) …And of course, we can’t forget Silk Road, the digital anonymous drug marketplace that processed over $1 billion in sales in 2.5 years and was shut down by law enforcement in late 2013. In some ways, Bitcoin *could* be considered the first

decentralized application since it runs on blockchain technology, is fully open-source, and runs without a central authority. But seriously, a lot of us are here still looking around and wondering, “Where are the *killer* apps?” Sadly, almost no one I know uses blockchain-based applications in their day to day. **Here are some factors holding these applications back from popularity** (note: these are my personal opinions):

1. Lack of developer friendliness and tooling

Programming applications using Bitcoin’s scripting language is not easy. Why? For one, the scripting language is too limited. A scripting language is a programming language where you can write code to perform some actions. An example of a scripting language widely used on the web today is JavaScript.

```
const greeting = (name) => "Hello, " + name + "!";
const add = (a, b) => a + b;
const subtract = (a, b) => a - b
```

Compare this to Bitcoin’s scripting language:

```
OP_DUP OP_HASH160 62e907b15cbf27d5425399ebf6f0fb50ebb88f18 OP_EQUALVERIFY OP_CHECKSIG
```

The JavaScript on top reads pretty much like English. Bitcoin’s scripting language, on the other hand, looks like machine code. Most developers are used to writing in expressive languages like JavaScript, Ruby or Python… not machine code. Bitcoin script is daunting for most developers. Secondly, developer tooling and great documentation goes a long way in gaining adoption among developers. Take React, for example, which is one of the most

popular front-end libraries today. One of the biggest reasons that React became so popular is because of how much effort the community has put into building a strong set of developer tools (e.g. IDEs, Babel, Webpack, boilerplates, Create React App, etc), documentation, and tutorials. Bitcoin's ecosystem, on the other hand, is the opposite of user-friendly. Lastly, Bitcoin's scripting language is not turing complete. A turing complete programming language is one that can be used to simulate any single-taped Turing machine. In other words, it can be used to solve any computation problem that a Turing machine can run given enough time and memory. (for more on this, read this Stackoverflow discussion). By not being turing complete, Bitcoin script restricts what you can do. **Overall, Bitcoin's scripting language has historically been limited, difficult to use and lacked adequate tooling and documentation.** As a result, it didn't encourage a developer community to form, which is the prerequisite to killer applications.

2. Building a decentralized application with strong network effects isn't easy

Many of the applications we use in our daily work (marketplaces, exchanges, social networks, etc) derive their value from their strong network effects. A network effect is when a product or service increases in value as more people use it. A classic example is Facebook. Every new user connecting to other users on the platform non-linearly increases the number of connections. Similarly, Venmo is useless if you're the only person on the platform. For every new friend that joins, the value of the product goes up because you can now pay and/or receive payment from this friend. Network effects help build better products and

services. However, building up this network is one of the hardest parts of building a successful product, classically known as the "chicken and egg" problem. So even if a developer were to make the effort to build a decentralized crowdfunding platform on top of Bitcoin's blockchain, getting users on both side of the platform (i.e. investors and product-builders) is an incredibly hard challenge. The blockchain provides the technological underpinnings to create decentralized applications, but it doesn't provide the framework or tools necessary to drive adoption of the network.

3. Decentralization alone doesn't provide a 10x improvement

When we talk about decentralized applications built on top of the blockchain, we might think of transaction-based platforms, such crowdfunding, remittances, payments, coupons, etc. It might be a neat technical feat to have a decentralized version of these types of services, but the reality is, we already have existing apps that work perfectly fine for each of these use cases. For crowdfunding, we have Kickstarter. For remittances, we can use TransferWise. For payments, we can use Credit Cards, Paypal, Venmo, Square, etc. Peter Thiel's 10x rule is important to think about when we're considering how to get users to substitute existing solutions with the new decentralized ones. As of now, it's unclear what dimension these 10x advantages come from, so far as users go. Take WeiFund, for example, which is a decentralized crowdfunding platform. As a user, WeiFund's interface and user experience seems similar to conventional crowdfunding platforms such as Kickstarter or GoFundMe. The

main differences seem to be that they claim to have lower costs and that they use smart contracts to run the crowdfunding, allowing for more complex agreements. Is this enough to get users to make the effort to switch over (especially when the costs aren't *that* much lower)? By no means do I believe that decentralized applications have no benefits. In fact, I foresee a future where applications are 10x more secure, 10x cheaper, 10x more efficient, or 10x more on some dimension than the current ones. The point is that these benefits have not been proven yet, so there's little reason for users to consider using a decentralized application today.

The rise of decentralized applications… maybe.

Enter Ethereum.

Ethereum is a cryptocurrency launched in 2015 and built from the ground up using its own blockchain technology. It was designed to be a more generalized protocol than Bitcoin's blockchain, with the explicit goal of doing more than just creating and recording transfers of a blockchain network's native tokens.

As written in the Ethereum white paper:

> *The intent of Ethereum is to create an alternative protocol for building decentralized applications, providing a different set of trade-offs that we believe will be very useful for a large class of decentralized applications, with particular emphasis on situations where rapid development time, security for small and rarely used applications, and the ability for different applications to very efficiently interact, are important. Ethereum does this by building what is essentially the ultimate abstract foundational layer: a blockchain with a built-in Turing-complete programming language, allowing anyone to write smart contracts and decentralized applications where they can create their own arbitrary rules for ownership, transaction formats and state transition functions.*

Ethereum

In essence, Ethereum is simply a transaction-based state machine: we begin with a “genesis state” and incrementally execute transactions to transform it into some final state. The final state is what we accept as the canonical version of the current state of the world of Ethereum.

Ethereum transaction

While Bitcoin is the intrinsic token for Bitcoin's blockchain, Ether is the intrinsic token for Ethereum's blockchain.

Just like Bitcoin, the Ethereum blockchain contains a log of transaction-like events. Users send Ether to one another using the "log," and miners are incentivized to verify and secure these transactions within the network.

But it can also go way beyond that—the Ethereum blockchain can be filled with a wider variety of event information coming from any sort of computer program.

Let's look at a few of the core concepts that underly the Ethereum blockchain to understand why this is possible:

Ethereum basics

First is **accounts.** There are two types of accounts: Externally Owned Accounts and

Contracts Accounts. Both account types have an Ether balance.
The main distinction is that contract accounts have some piece of code associated with them, while externally owned accounts do not. Contract accounts, therefore, have the ability to perform any type of computation when its associated code is executed.
Next we have what are known as **transactions,** which are cryptographically signed data packages that store a message to be sent from an externally owned account to another account on the blockchain. When a transaction is sent to a contract account, the code associated with the contract account is executed by the “Ethereum Virtual Machine (EVM)” on each node (more on that below).
Finally, there are **messages**. Messages allow contract accounts to call one another. When a contract account send a message to another contract account, the code associated with the account is activated. Essentially, a message is like a transaction, except it’s produced by a contract account rather than an external account.

Let’s quickly explain the concept of the “Ethereum Virtual Machine (EVM)”. Remember how we learned that the protocol for the Bitcoin blockchain determines how transactions on the network get verified? Well, in Ethereum’s case, every node that is participating in the Ethereum network runs the EVM as part of this verification process.

Source: Coinspace.io

Let’s say we have a set of transactions that were started by some external accounts. These get accumulated into a block, and then the nodes in the Ethereum network go through the transactions listed in the block and run the code associated with these transactions within the EVM. It’s important to note that every node in the network runs the code and stores the resulting values. As you might guess, this tends to be computationally very expensive. To compensate for this expense and incentivize the nodes (or miners) to run these computations, the miners specify a fee for running these transactions. This fee is referred to as “gas” (you can read more on gas here). This is similar to how fees work in Bitcoin, where any fees attached to a bitcoin transaction go to the miner who mined the block that included the transaction.

Note: This is a very high level description of how the Ethereum blockchain works and it certainly skips a lot of details for purposes of brevity. I'll write more in-depth articles in the future.

Lastly, we have Ethereum's **programming language** for writing executable distributed applications and contracts. Unlike Bitcoin, Ethereum's programming languages (Solidity which is similar to JavaScript , Serpent which is similar to Python) don't look like machine code. It has the expressive power and functionality of languages that programmers are accustomed to developing on, like JavaScript or Python. Moreover, it lets you do pretty much anything an advanced programming language would let you do. Hence, it is "Turing complete".

The key takeaway from all this is that Ethereum stepped into the crypto-world and provided us with a generalized framework for running any type of code on the blockchain more easily. Because Ethereum's language is turing complete, stateful, and

developer friendly, the hope was to open up the benefits of the blockchain beyond just enforcing one particular ruleset (e.g. how digital money gets transferred) and enable a safe, open, highly available, autonomously governed, efficient, trustable and reliable mechanism to build any ruleset on top. This would enable developers to develop any type of application imaginable.

An example of an application that is incredibly simple to build on Ethereum is a "smart contract". A smart contract is a distributed contract that is represented in code and basically says "if this happens then do that". They can accept and store Ether and data, and can send that Ether to other accounts or even other smart contracts. Just like regular contracts (e.g. a property lease or an employment agreement), they are used to form agreements with people or entities, but unlike regular contracts, they act like autonomous agents that run entirely on the blockchain and remove the human out of the loop, making them automated, open, secure and trustless.

Another example of an application is a decentralized organization. A decentralized organization is a programmatic organization that runs based on rules encoded within smart

contracts. So instead of the typical hierarchical structure of an organization that is managed by humans, a decentralized organization encodes all its rules into a smart contract and then is completely managed by a blockchain.

Despite the fact that Ethereum has made it easy for us to now build applications on the blockchain, let's admit it, most of us, even including us crypto-nerds, are still living in a world where we don't use decentralized applications in our day to day.

Why is that? To explain, let's go back to my earlier hypothesis on why we ended up at the "where are the apps" problem, and see how Ethereum addresses each one.

Problem #1: Lack of developer friendliness

As we described above, Ethereum solves this problem by design through its expressive programming language and strong developer tooling.

Problem #2: Building up network effects is hard

With or without Ethereum, seeding and spinning the network effects is still a huge roadblock. Replacing existing network businesses who've built up huge networks effects is… as we said before, really HARD. If someone builds a decentralized Airbnb, they still need to convince both sides of the platform, the users and hosts, to come on board.

Problem #3: Doesn't provide 10x improvement

We learned earlier that most users wouldn't be willing to switch to a decentralized platform unless it's 10x better than an existing solution on some dimension.

Just because it's easier to build a decentralized application on Ethereum, doesn't mean it provides the 10x experience we're looking for. And so, the question we might ask is, are we still right back to square one? still stuck in the trenches?

Well, not really.

Because although Ethereum doesn't directly solve the network effects problem, nor the 10x problem, what it does do is enable the creation of a whole new set of applications that were never possible before. The clearest way to make a 10x improvement is to invent something completely new. **I believe Ethereum makes inventing something completely new possible by making it easy to build smart contracts.**

The rise of protocols, tokens, and applications

Why the big deal about being able to build smart contracts?

Protocols

Well, the beauty of being able to easily build smart contracts on Ethereum is that it enables anyone to easily build a new protocol on top of Ethereum. Remember that a protocol is simply a set of rules that nodes in a network use when they to transmit information. Smart contracts allow us to do exactly this — create an automated trustworthy set of rules between two or more parties.

Earlier, we mentioned how blockchain protocols have a intrinsic “token” associated with it, which is a digital asset that can be transferred between two users in the network without requiring the consent of a third party. In the case of Bitcoin’s blockchain, the intrinsic token is Bitcoin and in the case of Ethereum’s blockchain, the intrinsic token is Ether.
But just because the Ethereum and Bitcoin blockchain protocols have intrinsic tokens associated with it to drive the network, doesn’t mean a protocol built on Ethereum using a smart contract must have a token associated with it. Remember that the purpose of a protocol is simply to specify rules for communication between nodes.

So essentially, there's two types of protocols:

1. Ones which have a intrinsic token associated with it that help create economic incentives that drive a network
2. Ones which don't have a token that drives financial incentives and are simply used as a communication protocol between nodes (*Note: these types of protocols CAN still have a token associated with it — e.g. to represent membership in the network, shares on an open market, etc. The difference is that they are not used to drive some economic incentive.)*

For lack of better names, I'll call the first kind "**crypto-token-protocols**" and the second kind "**crypto-protocols**".

Tokens

Now onto tokens.
Just like Ethereum makes it possible to build new protocols on top of its blockchain, it also makes it possible to use smart contracts to build new tokens on top of its blockchain. Let’s call these types of tokens “non-intrinsic tokens”.

In this regard, broadly speaking, we can think of a token system as just a database with one operation: subtract X units from A and give X units to B, under the condition that: (i) A had at least X units before the transaction(ii) The transaction is approved by A

Ethereum makes it especially easy to implement such token systems. More specifically, ERC20 token interface provides a standardized way to develop a token that is compatible with the existing Ethereum ecosystem, such as development tools, wallets, and exchanges. What's more, these non-intrinsic tokens can exist as:

1. Standalone tokens built on Ethereum (as shown in the above diagram)

2. Be associated with any new underlying crypto-token-protocol built on Ethereum
3. Be associated with any new underlying crypto-protocol built on Ethereum

Protocols? Tokens? Protocols + Tokens? Why does this matter? Let's take a look.

Token sale (i.e. "ICO")

Launching a new cryptocurrency blockchain is not easy — it requires a massive bootstrapping effort in order to assemble the resources needed to get it up and running. But in the case of Ethereum, its intrinsic tokens were used to spin up their blockchains — in order to kickstart a

large network of developers, miners, investors, and other stakeholders, Ethereum created some Ether tokens and launched a presale of these tokens to the general public. It then used these funds to develop its blockchain.

Ethereum was not the first to do this. In 2013, when Ripple started to develop it's Ripple payment system, it created around 100 billion XRP token, and sold these tokens to fund the development of the Ripple platform.

This concept of fundraising via a token sale is sometimes referred to as an "Initial Coin Offering", or ICO. But the structure of this token can vary significantly (as we just saw in the previous section), whereas the term "ICO" makes it sound a lot more official and like an investment security, so let's stick to "token sale".

A token sale is when some party offers investors some units of a new cryptocurrency (i.e. token) for a certain price, that can then later be exchanged with other cryptocurrencies (i.e. tokens). The idea is that investors buy into these tokens, and the units of the token are fungible and transferable on cryptocurrency exchanges (e.g. Bitfinex, GDAX, Liqui, etc.) if there is demand for them.

While most token sales in the past have been restricted to building a new cryptocurrency (e.g. Ethereum, Ripple, etc), the smart contracts of Ethereum are now enabling startups to also to use token sales to fund development of various **protocols and applications** built on top of existing blockchains.

Before moving on, one important distinction to make is the difference between an application and protocol.

Application vs. protocol

An application can be built on one or more protocols. One example is Augur, which is a decentralized prediction markets application that is built on top of two protocols:

- Decentralized oracle protocol
- Exchange protocol

The decentralized oracle protocol is a “crypto-token-protocol” that has financial incentives to drive the network to form consensus around the outcomes of real-world events using Augur’s reputation tokens (REP). The exchange protocol, on the other hand, is a “crypto-protocol” and does not have a token associated with it to drive financial incentives, but instead is a set

of rules defined between buyers and sellers in order to move tokens between each other. But neither of these protocols need to be tied to a single application. Any application can in theory build on top of these underlying protocols.

Token sales for protocols vs. applications

Earlier, I mentioned how token sales can be used to drive development of a new protocol and/or to drive development of a new application. So in essence, a team can use an token sale to fund:

- a **blockchain** (e.g. Ripple)
- a **cryto-token-protocol** built on top of an existing blockchain
- a **crypto-protocol** built on top of an existing blockchain
- an **application** built on top of a **cryto-protocol**
- an **application** built on top of a **crypto-token-protocol**
- an **application** built on top of **crypto-protocol**(s) and **non-crypto-token-protocol**(s)
- an **application**

So, pretty much anything :)

The last one is interesting, because to do a token sale, the application doesn't even need to be built under a protocol. I can build non-profit organization and use tokens as a mechanism to fund the project. In this sense, a token sale simply becomes a new way to fund a traditional centralized application. A plain old crowdsale.

Okay, so investors buy these tokens and then what happens?

Depends. When a token is tied to a cryto-token-protocol, they look much more like intrinsic tokens like Ether and Bitcoin and are used to drive the development and network of a protocol. But when they are not, tokens simply represent something much more general. In fact, these tokens are flexible enough to represent a lot of different things.

For instance, let's say I want to build a decentralized storage service. I can build a storage protocol using smart contracts which serve as agreements between a storage provider and their client, defining what data will be stored and at what price.

I would then build a token for this protocol and do a token sale. If the protocol becomes widely used, then the protocol becomes more valuable, which in turn could increase the value of the token. Moreover, as a developer of this service, I could choose to make the tokens represent purchase rights to the services provided in the application.

What's important to note is that, broadly speaking, the mechanism for creating tokens are so flexible that they can represent lots of different things:

- Paid credits/membership within the decentralized application
- Entitlement to a share of profits and/or losses, or assets and/or liabilities
- Ownership or equity interest in the protocol (or project)
- Voting power in the company

- No function other than mere existence — simply a digital asset that can be freely traded on cryptocurrency exchanges
- etc..

There's a handful of projects which have successfully raise funds via a token sale, including as Augur, Antshares, Melonport, Gnosis, Antshare and many more. I'd suggest you read their respective white papers if you want to learn more.

Protocols, tokens, token sales, now what?

We're at a point where Ethereum has made it easy to not only build protocols that can power decentralized applications, but also to help get a network off the ground. Ethereum does this in two ways: **1. Money**

This one is obvious. As we already saw, a token sale now enables developers to easily release tradable tokens to raise funds for building a protocol and/or application. Using this money, the team could choose to invest in sales, marketing, etc. to drive the network.

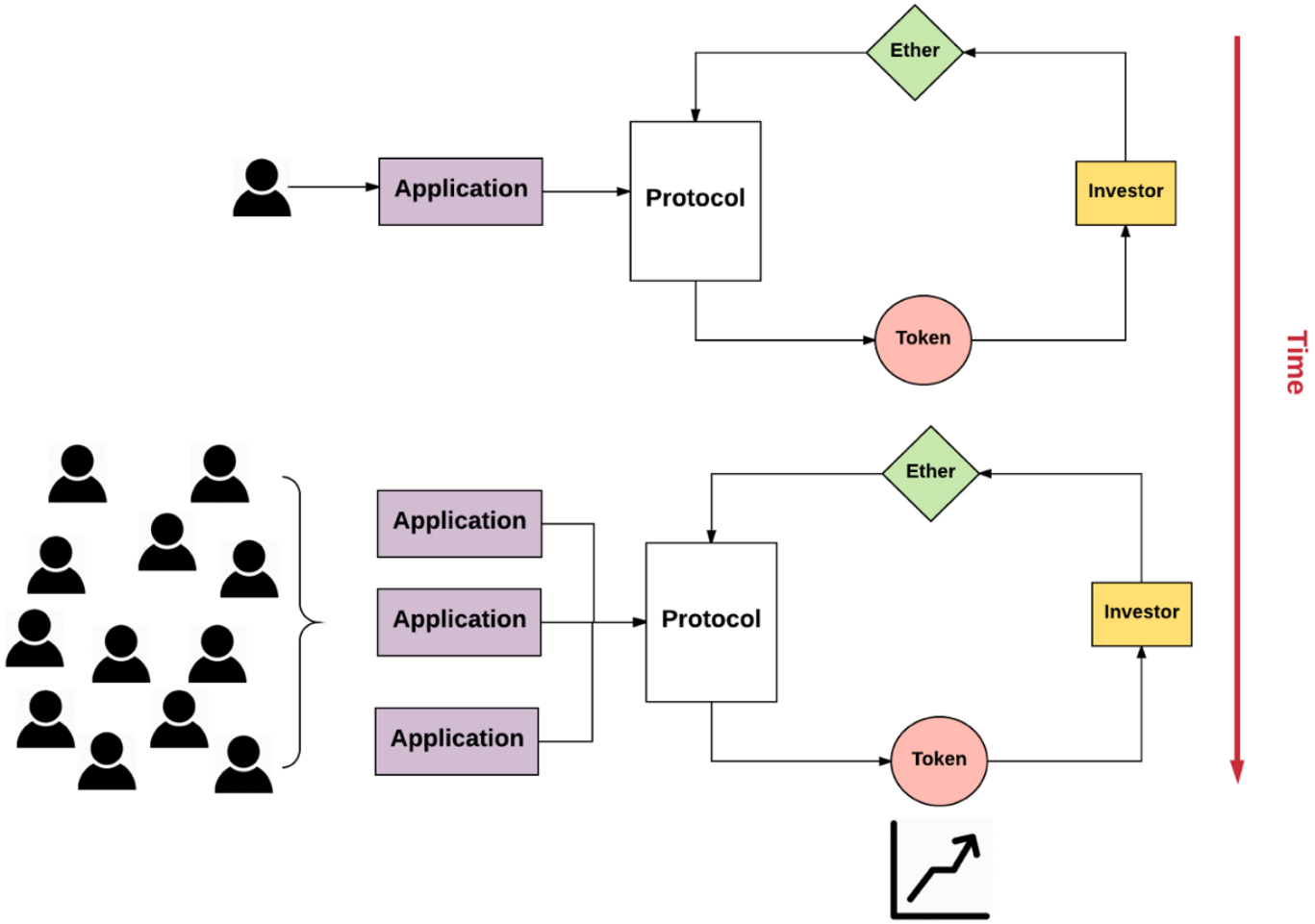
2. Users

This is the more interesting piece of the puzzle. **Protocols and decentralized applications**

can solve the network effects problem by using a token sale as a mechanism to get early contributors and adopters. Early adopters who believe in the protocol or application have an incentive to buy the token because there is potential for that token to be worth more in the future.

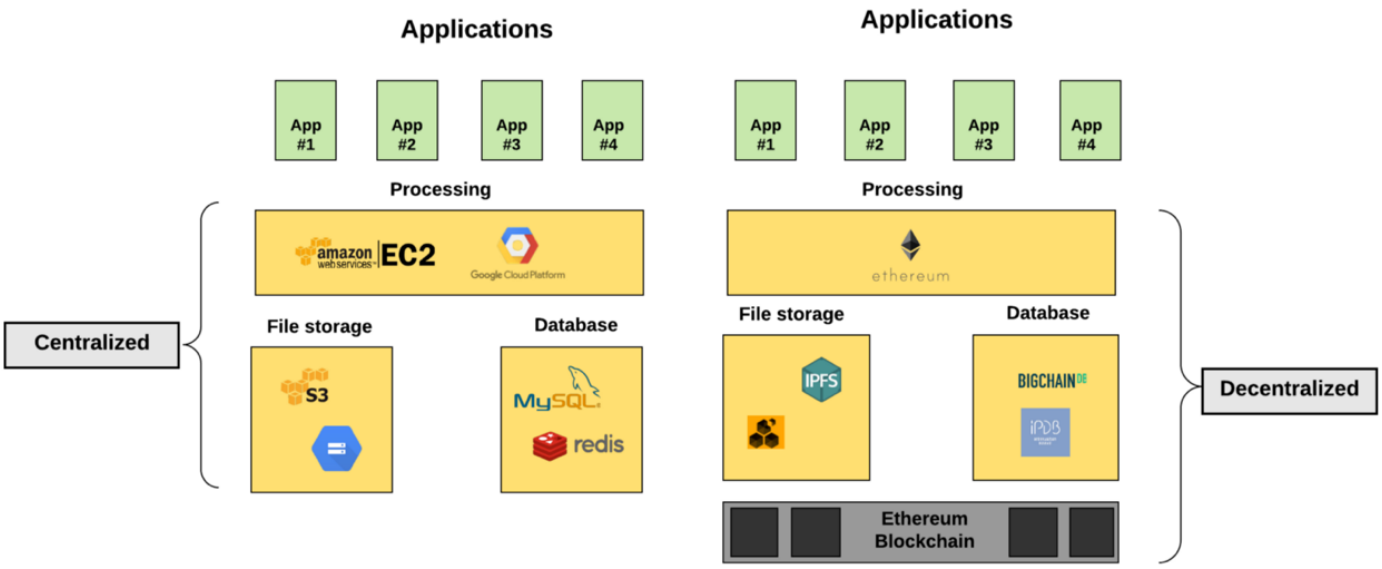
So in essence, tokens could help bootstrap a network of early adopters because the incentives of the early adopters and the development team line up perfectly.

Let's say you want to build a new file sharing protocol. You can launch a token sale through which you gain some early adopters, investors, and entrepreneurs who become interested in "buying in." They might be simply speculating or they might truly believe in the product. At that point, they become stakeholders in the protocol itself and are financially invested in its success. Then some of these early adopters either become users of the products built on top of the protocol or build products and services around the protocol themselves, with the incentive to drive the success of the protocol further in order to increase the value of their tokens. As the protocol gains adoption, it increases the value of the tokens, which further draws more attention from more investors, application builders and users, which leads to more applications, and so on.



What Ethereum has done is create an incredibly flexible system to innovate at the protocol level and application level. We'll likely see a lot of experimental and innovative protocols and applications being built over the coming years. Many of these will fail, just like a lot of startups fail. But over time, it's likely that some core set of protocols and associated networks will successfully drive mainstream adoption.

Finally, once the protocols begin to take shape and standardize, we'll see a whole host of decentralized applications being built on top.



It’s not a happy ending yet

Token sales are providing the fuel needed to drive development of protocols built on top of the blockchain, and to further drive developer interest in building applications on top of these protocols.

Of course, this isn’t the perfect happy ending.

For one, getting a bunch of early adopters isn’t enough. You also need to work hard to sustain the growth of the network effects, just like traditional businesses do. That means putting in years of hard work to building a useful application and driving adoption.

Secondly, another trend I’ve noticed is that most of the token sales we are seeing

today are being used to drive network effects around specific applications rather than open and decentralized protocols. Since tokens are so flexible, dApp developers are creating tokens that are coupled to the dApp, instead of a standardized underlying protocol that can be shared among applications. This could lead to fragmentation in protocols.
Third, the initial growth of the token value is mostly driven by speculation (since it takes some time for the platform being built to become valuable). Hence, there will likely have high volatility. It's unclear if and how we can mitigate this, and if we can even figure out a mechanism to get token prices to stabilize over time. Overall, there's a lot of open questions around the viability of a token's value over time. **Ideally, we want the token's value to be tied to the value of the protocol or application, similar to how a public company's stock is tied to the company that issued it, or to represent some valuable digital right to a service. But as of today, the value of these tokens is still mostly speculation.**
Fourth, the market for token sales incredibly frothy right now. Because securities regulations makes it difficult to sell tokens (which are unregistered securities) as equity (remember that a token can represent anything, including equity within the protocol or application), developers are not doing it. Instead, they are structuring them as crowdsales. While there are some highly respectable projects raising much needed capital in this manner without the hassle of regulation, there's a long tail projects that are simply taking advantage of the high demand in the ICO market to raise millions of dollars in capital with very little to show for it — some of which have even turned out to be outright scams that absconded with the funds collected during the process. We want these crowdsales to benefit the groups of people gathering together to build a common public good, but not the scammers. How do we achieve that?
Besides these issues, there are still lots of unanswered questions that need to be figured out before token sales become a viable form of funding:

- What is the right design and structure for a token sale?
- In what cases does it make sense to structure tokens as an investment security? How should such token sales be regulated?
- What criteria should an individual investor use to evaluate token sales? (team, business model, etc)
- How is an investor guaranteed that the sale process in being managed in a safe and legal

manner?

- Does there need to be a mechanism for distributing a portion of the liquidation value of the

- Could it be more efficient to port the smart contracts of a token to use Ether directly, instead of creating a new token per protocol?
- What are the tax implications for the investor's gains or losses?
- If selling tokens become a new way to raise capital for startups, what happens to traditional Venture Capitalists?
- … and SO MUCH MORE

If these questions interest you, you're in luck — I'll write about some of them in upcoming posts!

Conclusion

I've tried my best in this article to articulate my views on token sales and clear up some of the confusion around blockchain development in general.

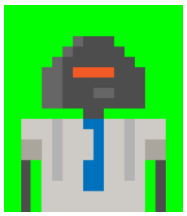
Talking about cryptocurrency and blockchain development is like trying to take a picture of a running cheetah. The space is moving at breakneck speed, and any attempt to pin it down results in a blurry picture. Regardless, I still believe it's important that we educate the broader community on cryptocurrency topics.

If you feel that I've made any overreaching assumptions in this walkthrough, please share commentary below! I'd love to talk more and learn from each other.

We need everyone's input to figure out the right path towards a healthy and sustainable cryptoeconomic future.

Enter the Blockchain Writing Contest

The Blockchain Writing Competition is sponsored by Tatum, the ultimate blockchain development platform. Submit #blockchain stories and get a chance to win prizes worth $1000 EVERY MONTH!!!!



by Preethi Kasireddy @preethikasireddy.

Read my stories

Comments

Signup or Login to Join the Discussion

TAGS

#blockchain #ethereum #bitcoin #cryptocurrency #venture-capital

RELATED STORIES

Blockchains don’t scale. Not today, at least. But there’s hope.

Published at Aug 22, 2017 by preethikasireddy #blockchain

Stablecoins – The T0000000.1 Problem

Published at Aug 12, 2022 by oceanfling #stablecoin

Top 5 FAQs on Smart Contract Auditing

Published at Aug 12, 2022 by davidhenry #smart-contract-audit

CBD & Crypto – the Best Team Up Since Riggs & Murtaugh?

Published at Aug 12, 2022 by ghostd09 #cryptocurrency

What is a Cryptocurrency Crowdsale and How Does It Work?

Published at Aug 11, 2022 by wanetaj #cryptocurrency

Meet the Writer: HackerNoon's Contributor Bader Youssef, Web3 Developer

Published at Aug 11, 2022 by crackTheCode #blockchain

THIS ARTICLE WAS FEATURED IN...

Cryptocurrencyfacts | Sfox | Dieteticienne-nutritionniste-paris | Campaign-archive

MENTIONED IN THIS ARTICLE

COINS

XRP Litecoin Monero Zcash

COMPANIES

AirBnB Amazon BUNCH Coinbase Dapp

Different Facebook fees Few Funding

Google Oracle Paid Stripe YouTube

READ

Archive
Leaderboard
Noonification
Signup
Tech Brief
Tech Tags
Top Stories

WRITE

Distribution

Editor Tips
Guidelines
New Story
Perks
Prompts
Why Write

PARTNER

Billboard
Brand Publishing
Case Studies
Contests
Niche Marketing
Newsletter
Writing Contests