Our presentation is about a blockchain system called EOS.

This disclaimer basically says that we are NOT in any way associated with block.one, the company which has created EOS.

However, rest assured, this event has been coordinated with them, so hopefully we'll be talking sense.

The goal of this presentation is to make you consider building businesses on top of EOS.

OK, so before we talk about EOS, we’re going to briefly describe what we consider to be the main problems facing the crypto-space.

We’ll go through them one by one.

**Scalability**

This is a big one. If we want to host real-life businesses on the blockchain we need high performance. And what is high performance? These are the requirements of real industry:

* You need about 20k transactions per second just to do Visa & MasterCard, not to mention all the other payment systems out there.
* Facebook does 50k likes per second, not including all the posts and voting and other actions that the users take.
* And if you're in the financial industry and you're trying to do trading, it's sometimes a hundred thousand transactions per second.

So, imagine putting all of these on a single platform, so they can interoperate. Well, that would require millions of transactions per second. This is a massive scale.

And what can the blockchain currently offer? Bitcoin does 4 transactions per second. Ethereum – something between 15 and 30.

And those two systems are extremely expensive to run. Bitcoin and Ethereum shareholders are spending something like 6 billion USD a year just to be able to process less than 35 transactions per second. This gives us 50 USD just to process one transaction. This is mad.

You might say it's just a matter of time before scaling solutions are put in place. But actually, there are only two ways to go:

* You can go off-chain, which is basically the idea behind state channels. Will it work? Who knows. But even if it does, it only fixes the problem for basic payments, and does nothing for anything more complex than that.
* Or you can go into creating sub-chains, which is basically the idea behind Ethereum sharding and also Plasma. This helps a bit, if you want to run a lot of small apps within the same ecosystem. But if you want to compete with big businesses like Facebook, Uber or eBay, or if you're into currency trading - this solves nothing. For those apps you need enormous processing power on a single chain and oftentimes on a single CPU thread.

#### Transaction fees

Right now transaction fees are absurdly high. On average you pay 4 USD for a Bitcoin transfer and 30 cents for moving funds on Ethereum, and this will only get worse once Ethereum hits its capacity limits.

No business can survive this.

Even if we introduce a second level of transaction processing and the fees do get lower, they will still be totally unpredictable. They are decided by block producers, so you as a blockchain-based business have no say in it. In other words, any unexpected rise in transaction fees can kill your business overnight.

#### Private key security

People get hacked. And people lose their passwords. It's a fact. And if you're hacked on a blockchain app similar to Uber or eBay, you lose not only your money, but also you lose your identity and reputation.

There will be more and more unsophisticated users entering the space and they cannot be expected to perfectly protect their private keys. So there must have a way to recover access to a blockchain account, or otherwise this thing will never go mainstream.

#### Blockchain governance

How to make decisions in a decentralized environment? Obviously, a blockchain, like any other enterprise, needs to make decisions, in order to survive in a changing environment. If this process fails, we end up with a fork, which usually hurts everyone involved.

Blockchain governance is basically about what you do when you have a disaster similar to the infamous DAO incident, which happened last year. Or how you solve the block size issue in Bitcoin, where you have two camps of shareholders wanting a different outcome.

These can be tough problems, especially when nobody is in charge.

#### Smart-contracts running amok

One way to tackle this problem is trying to prevent it from happening in the first place. Which means one thing: formal verification of the code before it is deployed.

However, formal verification of a Turing complete programming language is really hard. And it just minimizes the risk, it does not eliminate it completely. Software development is an imperfect art, so no matter how much effort you put into checking the code, there will always be a black swan event waiting to happen.

As we can see from this screen-shot, smart-contract platforms do NOT consider this issue to be part of their business model. So, when things go wrong, you, as a business, are on your own.

#### High difficulty of app development

Where are we today? We have the decentralized computer and a bunch of people trying to build applications on top of it. But actually nothing is coming to market. All we have are just prototypes. The developers are trying to build those amazing apps, but they don't get to spend their time on building business logic and user interfaces. Instead, they are stuck on figuring out the low-level stuff, things like databases, data storage and inter-app communication.

And because everything is being implemented in the app layer, it's being done within relatively inefficient scripting environment. Which makes it really hard.

Furthermore, even if generic solutions eventually emerge this way, they will be expensive to use - it will cost gas each time your app needs to access them.

#### Inter-blockchain communication

Right now blockchains are quite lonely creatures - they live in their own silos, separated from each other. Even apps on similar blockchains cannot talk to each other: a smart-contract on Ethereum is unable to communicate with a smart-contract living on Ethereum Classic, even though technically they are identical platforms.

OK, so we've painted this scary picture of blockchain space being overwhelmed with really big problems.

Now, what would be needed to overcome those problems? Let’s take a moment to describe what decentralized applications need, if they want to be successful:

* Your app needs to be scalable and cheap to run. That's pretty obvious. You need to be able to scale in order to have enough users to recover your initial costs.
* You need to have access to freemium models or different monetization strategies. If you want to charge your users that’s fine, but it should be the application choice, not the platform it's running on.
* Your app needs to be available via mobile phones and web interfaces, so that users don't have to download and install anything.
* Sometimes your users need privacy: if you build a financial app, users will expect confidentiality of their transactions.
* Your users need account names, no public keys, no fancy cryptographic stuff. And they need a procedure for account recovery, in case they are hacked or just lose their password.
* And you, as a business owner, probably need a rich development environment: lots of programming tools, and access to experienced developers.
* Your app needs to be upgradeable. You need to be able to improve your app to keep your business afloat. And finally, your app needs bug recovery, something that's often overlooked in this space.

OK, it’s time to introduce EOS to you, and find out how EOS can address all those problems & requirements.

What is EOS? The simplest way to put it would this: EOS is a general-purpose smart-contract platform, similar to Ethereum.

That's a fair definition, but in my view, it doesn't do justice to EOS. I prefer this one: EOS makes decentralized apps look and behave like conventional apps.

How does EOS work?

Let’s start with Ethereum. This is actually what Ethereum looks like. We have a bare-bone computer (in this case the EVM) and we have some apps trying to run on top of it.

They have a tough job because apart from some basic stuff, like the ability to send funds from one account to another and maybe store some data, they need to take care of pretty much everything else. It's like building a mobile app on a smart-phone without having Android or iOS at your disposal. It’d be pretty hard, I guess.

So what are we missing here?

Right, we need an operating system.

And this is what EOS brings to the table. EOS acts an operating system for running decentralized applications.

OK, so now let’s go through all the major features that make EOS unique.

We’ll discuss them one by one.

**Processing power**

We will start with a very bold claim: on day one EOS will be capable of at least 50 thousand transactions per second. Maybe more.

Let’s take a look how this fits into our market landscape.

How is EOS going to achieve that? It's gonna do it in two dimensions:

* Firstly, via enormous speed of sequential processing. EOS uses a solution borrowed from a currency exchange engine called LMAX, which is capable of millions of transactions per second on a single thread.
* On top of that we'll have parallel processing, made possible thanks to a clear separation between the things which can be done in parallel and things which cannot. This is an important distinction because not everything can be subject of parallel processing, e.g. managing an order book of a decentralized exchange always needs to be run on a single thread.

And what helps a lot to speed things up, is the fact that in EOS there is no such thing as gas – the processing power is free, we will talk about it in a moment. As a result, a lot of complexity is being eliminated, as you don't need to count all the tiny operations performed inside the VM.

Anther helping factor is the fact that in EOS nodes establish consensus over events, not over state, as is the standard in all other blockchains. So in EOS state is derived from events. At first this might look like an unnecessary complication, but it actually simplifies a lot of things inside EOS and it allows for powerful optimizations.

**Built-in governance**

* In the heart of EOS there is a consensus mechanism called DPOS, which stands for Delegated Proof of Stake. We don't have enough time to go into the details, so I'll just do a quick overview.

Basically, in DPOS there is a fixed number of block producers (around 20) and all of them are elected by the shareholders, each voting according to their stake.

The whole thing is powered by reputation: it’s hard to get elected but very easy to lose the job, especially if you misbehave or don’t contribute enough to the ecosystem.

As a result, DPOS allows for efficient decision making, while ultimate power always rests with the shareholders.

And one more thing: DPOS is cheap to operate – it doesn’t require several billions of dollars per year. As a result, the inflationary funds can be utilized for other goals.

* In EOS the assumption is that no matter what, bugs will be happening and handling them in an orderly manner is treated as integral part of the business.

To this end, block producers will have the ability to freeze & fix broken apps. Of course, only in case an app is unable to recover from a bug on its own. If block producers abuse this power, they will be voted out by the shareholders.

* What else have we got? There'll be a built-in governance mechanism, including a legally binding constitution, an arbitration system for resolving disputes, and an entire system of shareholders voting to decide on things like the level of inflation and the way the income from inflation is being distributed.

So for example the constitution will guarantee that the maximum level of inflation will be say 5%, and shareholders voting will decide how to distribute this income.

* Basically, EOS itself acts like a fully autonomous decentralized app whose business is hosting other decentralized apps.

**Infrastructure for apps**

* In EOS developers only need to write code for what's unique for their application. All common features, including low-level stuff, are provided by the blockchain.

Those features include: account permissions, account recovery, scheduling, authentication, inter-app communication, biometric 2nd factor validation.

Account recovery might be especially interesting here. It works like this: if you get hacked, and lose control of your account you can get it back, provided you know the previous password, before it was changed by the hacker. And it gets even better: if you weren’t hacked but just lost your password, you can still regain access to your account via EOS social network feature.

And biometric 2nd factor validation basically means that your iPhone or Android phone can act as your hardware wallet. Just like Trezor.

* Furthermore, EOS provides every application with its own private database.
* The system has a built-in storage solution based on IPFS and maintained by block producers. And this storage solution is free to use by your app, provided you're a token holder.

**No transaction fees**

* This is my favorite. Your EOS tokens are never consumed, as there is no such thing as transaction fee or gas payment for running a smart-contract.
* The system doesn’t need any fees as spam protection is achieved in an alternative way.
* If you're a shareholder, you can use the system for free. The deal is very simple: if you own 1% of the tokens, you own 1% of the network, including all its bandwidth and all its resources.
* However, if you still prefer the Ethereum pay-as-you-go model, you can rent EOS tokens, instead of buying them, and this way use the platform on pay-as-you-go basis.
* The point is, you have a choice. As a result, you can monetize your app any way you want.

**Publish source code, not machine code**

* This one looks quite innocuous but actually it's a big deal. Firstly, it means that human intentions are part of the blockchain consensus.
* What it also this means is that the source code of your app can be recompiled in the future.
* It also means that EOS can have multiple VMs and your contract can switch from one to the other.
* And finally, your smart-contract becomes upgradable, so you are no longer stuck on the very first version of your app.

**Asynchronous communication**

* EOS works like a giant email server, as everything based on sending and receiving messages. Which means that the entire communication in EOS is asynchronous. This is the opposite of all other smart-contract platforms, where smart-contracts talk to each other in a fully synchronous manner.
* And it just happens that asynchronous communication is the only way to arrange communication between blockchains. As a result, EOS apps can communicate with each other NOT only within the same blockchain, but also across blockchains.
* So, if we had multiple EOS blockchains, both public and private, everything could be fully interconnected

Here is the list of EOS main features again.

And now I'll tell you a little bit about EOS origin.

There are two ways of building a smart-contract system:

* Either you start by creating the damn thing and then try to use it to run concrete apps on top of it. We might call it a top-down approach.
* Or you start with building a couple of concrete, non-trivial apps, make some mistakes when doing it, learn from those mistakes, and as a next step try to figure out what's common between your apps, and only then start building the abstraction layer. So it’s a bottom-up approach.

You can make your own judgement which one is more likely to be more effective.

Unfortunately, the prevailing approach in the crypto-space is the first one, top-down. Teams of very talented people are creating abstract smart-contract systems, and then they are confronted with tough reality when they try to build something useful on top of those abstract systems.

EOS goes the other way, bottom-up. The team behind EOS has already built some of the most successful blockchain apps in the crypto-space, and now they have set out to leverage their experience by building a generic platform for hosting such apps.

And when I say most successful apps, I don't mean market capitalization, I mean how popular those apps are among actual users.

This is what it looks like. BitShares and Steem were created by the same developers who are now behind EOS.

And those two apps are very different: BitShares is a decentralized exchange, whereas Steem is a social media platform.

As you can see, Steem is the most used blockchain app in the entire space. BitShares is pretty popular, too.

This is what their UIs look like. Those are very complex apps. They still have their deficiencies, but they work smoothly enough to trick you into thinking that they are just normal web apps. They have a couple of amazing features, e.g. account recovery was first introduced in Steem.

And one more interesting thing: while those apps are quite popular among users, in reality none of them could run on any existing smart-contract platform, including Ethereum.

Why is that?

Firstly, because those two little apps alone would eat up all of Ethereum processing power.

Secondly, Steem business model requires that all user actions are free of charge (as otherwise nobody would upvote anything). Ethereum is unable to offer that and most probably never will.

EOS strong points:

* EOS has a very efficient team with a proven track-record. Steem was invented, created and launched in just four months. And this is probably the most complex blockchain app currently in production.
* EOS is using Web Assembly as a VM for compiling & running smart-contracts. Web Assembly is a big deal - it's an emerging industry standard for web applications supported by Google, Microsoft, and Apple. You should look it up, but basically it's a universal compile target for several programming languages, including C++, C#, Java, JavaScript and possibly even Ethereum's Solidity.
* Regarding EOS financial supporters, we have some big names there: Brock Pierce and Bo Shen. Interestingly, both of them were also early investors of Ethereum, and still are.
* And last but not least, we have a declaration from EOS founders to spend about 1 billion USD on creating an ecosystem around EOS. Apparently, this is where most funds from the currently running ICO will go - not on EOS code development (as this is already fully funded) but on those of you who will try to do something useful with EOS.

Unsurprisingly, EOS has also got its weak points.

* DPOS, the consensus mechanism used by EOS, is perceived as being not decentralized enough.

This is a bit strange, as actually the opposite is true: DPOS is the most decentralized system out there and here is a chart that proves this. It shows the only rational measure of decentralization: how many unique entities are involved in producing most of the blocks. This is the data for BitShares & Steem, both of which are using DPOS.

However, despite these facts, DPOS suffers from being quite unappreciated in the crypto-space. It’s a problem because human perception does matter.

* The ecosystem around EOS has not really emerged yet. It might be hard for EOS to catch up with the high number of developers currently working for Ethereum-based projects.
* Track-record of poor documentation. Unfortunately, this is true. Those guys behind EOS are brilliant, but their projects have always lacked good documentation. Hopefully it will change this time around.
* Using C++ is right now the only way to create smart-contracts on EOS. This will surely change, as Web Assembly gets upgraded, but unfortunately at this early stage this might be a big turn-off for developers who are unfamiliar with C++.
* And it's not live yet, but it will be in June next year.

This is the roadmap. Currently we are at the MVP stage.

You can already play with the system on your local machine and start building smart-contracts with it, provided you have some basic knowledge of C++.

In a couple of days there'll be a subsequent release, called Dawn 2.0, and there’ll be a public test-net up & running.

And now let's revisit our list of requirements for decentralized apps. As you can see EOS is doing a pretty good job here.

The only potentially weak points for EOS are those two: the lack of rich development environment and no special features for privacy protection.

And we can also revisit our crypto-space problems. Again, as you can see, EOS features nicely match just about all of them.

I hope all of this is self-explanatory.

To wrap it up:

* What sets EOS aside, is not just the technology. It's actually the way it's going to operate, e.g. the unique role of the token.
* EOS builds on solutions which have already been battle-tested in real-life blockchain apps. For example, the entire concept of DPOS, elimination of transaction fees, account recovery – all of them have been proven to work.
* EOS takes the notion of smart-contracts to the next level, as it is extremely business oriented: you can reliably build businesses on top of it, businesses with predictable cost and ones which will scale.
* Can other systems (like Ethereum or Neo) copy EOS solutions? Not really. EOS is quite different from anything else on a very fundamental level. To follow EOS path, other platforms would have to backtrack a lot regarding the fundamental elements of their technology. So, it's very unlikely.

And finally, this graph shows the conclusion we have reached when observing how blockchain technology evolves.

Back in 2008 we started with a simple payment system, which then evolved into a smart-contract system, which now is about to evolve into a fully blown operating system, not just for smart-contracts, but for entire blockchain-based apps.

Will EOS replace Ethereum? I don't think so, at least in the predictable future. But what will probably happen is EOS playing the same role for Ethereum, as Ethereum is playing for Bitcoin.

And now very briefly about our company:

* Our focus is on blockchain-based fundraising and digital asset management solutions.
* We are in the process of setting up a software house dedicated to building dApps, both on Ethereum and EOS.
* And we are aiming to be elected as an EOS block producer. We have some experience with that, as one of our advisors is already a witness for Steem.