Builders Nights Singapore Keynote - Vitalik Buterin

(0:04 - 23:10)

How is everybody doing? Come on. Very cool, very cool. So exciting.

I think it's one of the most packed Builder Nights that we ever had in Singapore. And yeah, big shout out to you guys for coming, for hearing, and it's incredible to have such inspiration. I think everybody, until now, knows Builder Nights.

And yeah, so just a quick overview. Builder Nights is all about no shilling. We want to speak about technical, narrative.

We started with 4.3.7. We went to different kind of layer 2, infrascaling, and we're very excited to also have a lot of builders. How many developers do we have in the room? Yeah? Oh, cool. Nice, nice, nice, nice, nice.

Very cool. So sounds good. Without further ado, I think we can get started.

Big round of applause. Thank you so much, Vitaliy, for coming here and speaking about 4.3.7. And yeah, thank you so much. Always this light.

Sorry, the remote is too centralized. Come on, when's the transaction confirming? Should we be fine? Is this thing like running on chain on Bitcoin Satoshi's vision? There you are. Okay.

Okay. So great. So today we're talking about account abstraction and ERC 4.3.7 and ERC 7.7.0.2. So to start off, right, what is ERC 4.3.7? So ERC 4.3.7 is this culmination of a ten-year journey of figuring out how do we actually make account abstraction really work in the Ethereum ecosystem in a way that actually satisfies both all of the underlying goals as well as actually having the security and decentralization properties that we care about.

So actually attempts to do account abstraction in Ethereum started all the way back in 2015. So even when Ethereum launched, actually, for a while there was a plan to try to make the default wallet that users use be a smart contract wallet. And this was a serious expectation that we actually had for some amount of time and actually this was also the reason why we originally did not make logs cover ETH transfers between EOAs because we figured that people would quickly switch to smart contract wallets and smart contract wallets can just log everything.

So at the beginning the first account abstraction EIP was called EIP-86 and then after that it was followed up by EIP-208, it was followed up by many more things. And the core problem that it was trying to solve is basically that Ethereum is very good at enabling all of this general purpose computation for the execution part of a transaction. So for the

part of a transaction that actually does things once the transaction is already approved.

And what we want to do is we want to allow the validation part of a transaction to also similarly try to be general purpose. The specific motivating examples include, I mean one, multi-sig wallets, right? So I've been talking like everywhere actually since 2013 about how multi-sig wallets are the future and like we all need to switch to them. Then two is a quantum upgrade strategy.

Eventually quantum computers will come. According to Metaculous they'll come sometime in the early to mid 2030s. I've also talked to people who think that the Metaculous estimate is crazy and I've also talked to other people who think that the Metaculous estimate is crazy.

The problem is that they think that it's crazy in opposite directions. So you know we need to be prepared for quantum. Quantum computers will break ECDSA and so we will have to switch to some other mechanism.

Number three is being able to change the key that's securing your account, right? So key changes are a regular thing. They're a basic feature of any kind of mainstream computer security. So like PGP keys and like have like expiry keys in general have expiry dates and you're supposed to like rotate them out and switch the new ones because there's always a risk that over time a key gets leaked.

Then key changes, so multi-sigs, also custom validation conditions. So things like if you're sending a small amount of money require one key, if you're sending a larger amount of money require like four out of seven. So this is how the Ethereum foundation wallet works for example, right? So a whole bunch of these different use cases that try to implement more complicated logic on the validation step.

So ERC or EIP 86 and 208 started doing that and it started going down this rabbit hole but then we started figuring out more and more of the actual security issues, right? So for example, one of the problems that we ended up finding only a couple years in that's like a relatively subtle problem is this problem that what if you have a thousand accounts and each of those thousand accounts as part of their validation they call into one particular contract and then you have a thousand transactions that look like they're valid but then someone sends one transaction that flips a bit in that in that shared contract from one to zero and then suddenly all thousand of those contract transactions get kicked out of the mempool. This is a pretty bad denial of service vulnerability, right? And so this motivated the reasoning that hey, well we actually have to restrict the storage slots that individual transactions can access during the validation phase and then we discovered a couple of other things and that's where eventually EIP 2938 came in a few years later. Then Ethereum was finally entering the final stretch of switching to proof-of-stake and the entire internet was basically yelling at us and saying Ethereum is bad because we keep promising proof-of-stake and we keep never delivering and so we

felt like okay you know we have to like actually go off all cylinders and like actually get this proof-of-stake thing out there.

Of course as soon as you know we actually got close the narrative immediately flips to proof-of-stake being bad but you know that's fine that's how narratives work. So now Ethereum is proof-of-stake and you know it's proven to be robust and decentralized for over two years and that number will keep increasing. Actually yeah wait today is which day? It's the 17th.

Okay so I believe like two days ago was the second anniversary of the merge. So happy birthday proof-of-stake. So the point is that like we were busy on you know doing this whole proof-of-stake thing which is very important so that Bitcoin maximalists would laugh at us less.

Well actually no it's important it's important because like it's good for a Ethereum to stop consuming a quarter percent of the world's electricity and it's good for the chain to like actually have economic finality and be more secure and be more like client friendly and all of these benefits and like the other stuff is just a side benefit right. So the point is that we were busy right and so the change from an EIP to an ERC basically came about because we figured okay how do we actually keep moving account abstraction forward? Well how about we basically give the task to a part of the community that is separate from the core developers and so while the core developers are focused on the merge we can actually keep improving all of this account abstraction stuff in parallel, we can launch it, we can iterate on it and we can get it like really bootstrapped much more quickly in that way. And so that is part of where ERC 4.3.3.7 came from.

The other part of where 4.3.3.7 came from is that at that time there's been this community of people that were interested in use cases like letting applications sponsor transaction fees for their users, like letting people pay transaction fees in stable coins or in whatever asset they have right. So if you imagine you know you are some merchant in Argentina and like you know you have a restaurant and then someone pays you 23 Dai for a meal and then you decide that like you want to buy a coffee and then you just you suddenly need to spend like 3.5 Dai for the coffee but then guess what you don't have any ETH in your wallet then how do you pay right. So this is like another one of those things that somehow got into the label of being account abstraction you know totally not my fault but you know it's in there fine.

And so there's this like growing range of use cases and so I like I'm a later I'm gonna call it the convenience side of account abstraction right. So 4.3.3.7 is like basically a combination of turning 2.9.3.8 into an ERC instead of an EIP so something that works as a higher level primitive rather than being a directly a protocol feature and features like paymasters and all of these other convenience related things that users want. So this is the history.

Now what are the goals of ERC 4.3.3.7 right. So one is allowing users to specify custom validation functions so I talked about this already. Two is being friendly to decentralized mempools and not suffering from DOS attacks.

So this is really the thing that is hard right. If you're willing to be centralized then like actually you don't need any of this right. All you do is you just like have your smart contract wallet and then you just like send zero gas transactions and then you give it off to like a builder somewhere and then you know like if if the builder is worried about being DOSed then like the builder can like KYC you with a phone number and then like you just solve the problem and it's fine right.

Except for like the whole reason why we're doing this crypto thing in the first place. So if you actually want something that is open and decentralized it turns out that there's actually a lot of pretty tough challenges. And then finally extensions to support paymasters and also extensions to support signature aggregation which is another one of those topics that I think is going to become more and more important over time.

Basically as more and more applications start involving snarks, snarks are crazy expensive or on layer twos if you want to use BLS signatures to save data then like it's in order to actually get the data savings or in order to get the key computation verification savings in the snark case you have to do an aggregation step. You have to either add all the signatures together or you have to make a proof of the proofs and so you have to like have some mechanism for representing this idea that like end users come in with n signatures or n proofs and then you make like one proof of the proofs that proves that those n objects existing but without actually containing them and so you save on either data or computation or both. So supporting all of this stuff at the same time is the goal of ERC 4337 right and especially the DOS resistance requirements are one of the key reasons why ERC 4337 is hard right.

Basically if you want to resist DOS attacks you need to resist restrict what the validation phase of a transaction can do and ERC 4337 is the result of a decade of work figuring out how to build against these constraints right. So there have been many many instances of people coming in saying oh ERC 4337 is over complicated let me figure out something much much simpler in a hackathon and then they build a thing and then they realize like wait for this to work like you basically just require a centralized relayer and then that centralized relayer themselves is going to have to KYC everyone right. So like these are actually hard problems and like it always inevitably ends up getting into the point where 4337 just is roughly the class of solutions that you get.

So convenience versus security goals of account abstraction is something that I think about a lot as like a key categorization right. So the security goals of account abstraction are the reasons that motivated me originally right. So multi-sig wallets, quantum resistance, social recovery, changing keys, especially revoking keys.

One of the newer things is wallets being secured by a trusted platform module so I have pass keys and you have a secp256r1 is the algorithm that gets supported by a lot of these and EOAs or secp256k1 which isn't. So security goals then you have convenience goals. Convenience goals are things like paying gas in ERC 20s, sponsored transactions, automation so like automatically making a payment every month and then there's like interesting things that are in the middle right.

So for example this idea of doing multiple operations in one transaction. This is actually a convenience goal and a security goal. The reason why it's a convenience goal is pretty simple right.

So who here has recently done a trade on just like any DEX? Okay cool people we actually use DEXs that's good. So in order to do a transaction on a DEX at least if you're doing it for the first time right you have to sign two transactions. It's like step one you approve and then step two after you've approved only then do you actually get to send the transaction.

This is annoying. So this is the convenience part. If you could do multiple operations within one transaction you could just send one transaction and it's much more convenient.

What's the security part? The security part is that like basically this ecosystem has come up with an entire zoo of protocols that have basically said well this approval stuff is annoying and so let's do a thing where you just like do a single approval and then that approval is just like an approval for infinite value and you try to approve as much as possible in one transaction so that after that you don't have to think about approvals. The problem is that if you do this then you're trusting an ever-growing amount of code some of which might have admin keys you have no idea and you're trusting that code with the ability to just like grab an unlimited amount of money from your wallet. This is bad.

And then the other bad thing of course is that if approving is like a thing that users are used to doing then what someone could do is like they could say hey you know I'm gonna do one of those things where I'm going to give you free airdrop tokens except to receive those tokens I need to verify your account and to verify your accounts you have to click approve and then you click approve and then you realise it just grabs all your ERC-20s. So doing multiple ops in one transaction can prevent a lot of those things and so it's also a security feature. But anyway so what's the point of this Venn diagram? Two technologies that people talk about as like sometimes I think talk about incorrectly as being like substitutes for account abstraction.

One is multi-body computation MPC and the other is you know what used to be 3734 now it's a 7702 and I think like this like I think it's good right and it's important but it's also important to understand why it's not full account abstraction right and basically

yeah what you notice is that MPC by itself it's I mean first of all like it cannot satisfy any of these convenience goals right and so the only thing that it's doing is it's satisfying the security goals but then even within the space of security goals it can't satisfy half of them right. So basically because like an MPC wallet it's like okay fine you're MPCing an ECDSA and that can give you multi-sig that can give you social recovery that will not give you a path to quantum safety that will not give you a key revocation that will not you give you TPM compatibility. Now if you have a trusted code co-signer it can give you granular access control okay so that's MPC right then if you think about 7702 it actually gives you all of the convenience features but it still doesn't give you a lot of the security features right.

So it's important to remember like basically what some of these intermediate technologies can do but also what they can't do and therefore why actually getting to the finish line of ideal account abstraction is so important. So okay I mean I've I guess I've shit-talked 7702 for two minutes but you know I yeah you know I'm a co-author and so you know 7702 came about for a reason right. So let's talk about what that reason is.

So what 7702 does is it gives existing EOAs the convenience features of account abstraction right. So basically everything in this the right side of the Venn diagram 7702 levels up EOAs to fit into that same standard. A key goal of 7702 is to allow the EOA ecosystem and the smart contract quality ecosystem to proceed as one ecosystem as they start employing all of these convenience features right.

So the intended explicit goal is that when for example Uniswap upgrade the you know the UI upgrades to let you do a proven trade in one single transaction the code path that enables that is exactly the same code path if you're coming in with a safe versus if you're coming in with an EOA right. Or sponsored transactions same thing right or you know like setting up some kind of social recovery ideally also the same thing. So proceeding as one ecosystem right and ideally yeah you're the 7702 wallet code can just be smart contract wallet code right.

At the very least exactly the same solidity file. So if we do that then we basically keep these two ecosystems together and we level up the convenience side of the entire ecosystem to give people all these features that we need and then in parallel we can keep improving the security features we could keep making the security side of 4337 more and more viable more and more secure cheaper and cheaper and then through upgrading or through conversion like we merge the ecosystems at some point later. So this basically is the intended roadmap.

So what are the next steps right. So this is one of the ways in which I think about it basically yeah if you think about the two axes as being like one is functionality and the other is efficiency and compatibility with you know existing applications in the existing ecosystem. You can think of EOAs as being in one corner and you can think about smart

contract wallets as being in the opposite corner right.

And so either you have an EOA and your EOA does not have these convenience features it does not have these security properties but on the other hand if you have a smart contract wallet which does have these properties then your smart contract wallet like it has a whole bunch of other problems right. So for example if you have a smart contract wallet you would also need to have an EOA that would that would do the paying transaction fees and you would be incompatible with a bunch of stuff and so on and so forth. The thing that 7702 does is it fulfills this short-term EOAs upgrades bucket right.

Basically yeah it moves EOAs up the functionality stack and then that gives you the convenience and then eventually we can have either an account upgrade mechanism or potentially we can convince users to just switch accounts entirely. This is a debate whichever one happens unfortunately we don't have to resolve it tomorrow then we can actually add the security features on top and we can get to ideal smart contract wallets and then at the same time we can take account abstraction and then we can start enshrining it and we can start adding layer one protocol features to make it more and more powerful and we can basically maneuver smart contract wallets into being the exact same thing as upgraded EOAs and then eventually the only thing that we have is just very efficient and very powerful smart contract wallets. So this is the long-term goal here right.

So next steps right. So EIP-7702 this is of course you know planned for Pectora and you know it's going to be great and EOA users will start getting access to really nice convenience features. I mean it definitely don't expect to be able to do one click trades on Uniswap at like fork time plus 30 minutes because I think this stuff needs to be developed securely it needs to be deployed with care and you know all of those things but these kinds of functionalities will start coming online.

At the same time ongoing work on refining smart contract wallets so there's this entire smart contract wallet ecosystem there's a lot of discussions around doing modular smart contract wallets, what pieces can we standardize, how do we do more verification of all the different pieces. So continued improvements on that whole ecosystem in parallel to this there's this whole conversation about enshrining like features of 4337 in layer one and this doesn't have to mean like literally taking all of 4337 and just turning it into a layer one feature right. Actually yeah the 4337 team so you know people some people from 4337 team are here.

Hi Yoav, say hi to everyone. So we you know we yeah so you know we always already like you know Yoav and the team have already yeah taken the original 4337 in like turning into an L1 feature EIP and turn it into a collection of EIPs that like put in specific features right and so you can you can take this piece by piece and the specific value that you get by doing it piece by piece is basically like one is that there are certain

inefficiencies that are unavoidable when something is a smart contract right. So for example one very simple one is like what if you have a bundle like you have to load the code you have to load a bunch of storage slots.

(23:10 - 28:34)

This has an extra fixed cost of about a hundred thousand gas and this is an expense and the problem is that if you have a market that has a high fixed cost and then like that adds a market inefficiency right. So that's one problem. Another problem is that there has been a lot of discussion around inclusion lists and around improving the Ethereum network's transaction inclusion guarantees so that users can reliably see transactions included even if the you know builders become super centralized and the builders decide that they want to screw around with and delay certain people and so in order for that to happen what we want is not just EOA based transactions but also account abstracted user operations to be able to benefit from those properties.

In order for that to happen because inclusion lists are a protocol level feature some portion of ERC 4337 the concept of validation execution separation and the ability of validation to be something that is a smart contract code needs to be enshrined in some form right and so this is also a bunch of work that needs to be done and then the hope is that over the next few years we can get to the point where the parts that really need to be enshrined get enshrined and then we work toward some kind of convergence toward an end state for this ecosystem. Making dApps more smart contract wallet friendly. This is super important for all the dApp builders out there right so there's a bunch of assumptions that there are some dApps and there are some protocols that are currently used that will stop being true in a smart contract wallet world.

So one simple one is like there's some of these like permit ERCs. One of the problems that they have is basically that they make the assumption that if you have a public key that hashes to an address then that public key is authorized to control the address. Pretty simple right? In an EOA world this is guaranteed to be true.

In a world where accounts can upgrade this is not guaranteed to be true and so you have to do things in a way that assumes that any wallet can become a smart contract wallet. So this is one example. Another example is lots of dApps require off-chain signed messages and off-chain signed messages there is an ERC for making them smart contract wallet friendly which is ERC 1271 and then on top of that there is another ERC.

I forget the number it's like somewhere in the high six thousands but the point of it is basically it makes it extends 1271 to cover a 6492. So I think you should just like think of 6492 as being part of 1271 right? So if someone ever says 1271 just like assume it includes 6492 right? Which basically extends it to cover like what we call counterfactual contracts. Contracts whose code has not yet been deployed on chain right? Which is a really important use case right? Because like your account receives money for the first

time before it sends a transaction for the first time right? Because you need to already have money to send a transaction.

So that's like dApps need to support that. There's also like certain conceptual changes. So for example you can no longer assume that if you make a signature once that signature will stay valid for the same wallet forever and like that's fundamental right? Because you want people to be able to change and revoke their keys.

So there's a bunch of these changes. Another big one is like there used to be dApps. I think people have fortunately already stopped doing this.

That assume that if an account has no code then it's an EOA and so it's controlled by a person and they do this to try to like basically prevent smart contracts from interacting with their dApps and like they're trying to do like soulbound token type use cases or royalties type use cases and things like that and that's something that unfortunately you'll probably just like have to stop doing and find some other way to do anyway. So all important next steps. Finally a whole other rabbit hole is cross-layer two smart contract wallets right? So one of the things I talked about in a blog post last year is this idea of the three transitions.

Ethereum ecosystem needs to simultaneously upgrade in three directions. One of them is security and convenience through account abstraction. The second is scalability through layer twos and the third is privacy through stealth addresses and various cryptographic protocols.

Now the challenge comes when you have to do multiple of these things at the same time. For example if you have an EOA that EOA is going to have the same address on every chain that has any version of the EVM. Actually it doesn't even need to have the EVM right? If you like literally just take Bitcoin and then you just swap out SHA-256 and RIPEM-D160 for Ketchack and you swap out the address format then like you would be able to have Ethereum EOAs on Bitcoin right? Now if you go to smart contract wallets then like suddenly it depends you might end up depending on not just the full EVM and like then there's like a difference between being EVM compatible and being almost EVM compatible and so there's more of a premium on like actually having the full EVM.

(28:34 - 30:22)

Then you have the fact that you have different EVM versions and so if you know you have some EVMs that support EOF and others don't then that creates an issue and then finally yeah you are going to have chains where you're not able to guarantee having the same address on every chain right? And this is like one of the motivations of the whole concept of chain-specific addresses right? Basically that you make the address like the human readable address include not just the 20 byte account ID but also include the chain so L1 or L2 or whatever that that particular account is living on and then that

address would be something that your wallet would just automatically give you and then when you give it to someone that then they would already know like this is the network this is the network that you're supposed to be sending them to and then if they're they're currently on Optimism but your wallet is on Arbitrum then they would know the wallet would know to do a cross L2 transfer instead of like trying to transfer on Optimism right? And so the whole concept of like accidentally sending someone coins on Ethereum when they should have sent on Polygon will just stop being a problem right? But in order for this to actually happen then like you actually need chain-specific addresses. Another big issue is if a user changes their keys then you will need a way for those key changes to propagate across layer twos and for that the short-term solution is basically making key change messages that are replayable. The medium-term solution is a keystore wallets so the scroll team has been working on this a bunch of other teams have been working on this where the key information lives on layer one and then the long-term solution is keystore wallets where the key change information lives on layer two.

(30:23 - 32:40)

So that's the kind of long-term roadmap there right? But the whole cross layer two discussion is like one that's actually very tightly connected to the switch from EOS to smart contract wallets and then actually if you start taking privacy into account then you start talking about stealth addresses and you start talking about how do we make keystore wallets themselves be zk'd and there's a lot of really interesting stuff there and so like that ends up taking things even further right? So there's still quite a long way to go but I think also this year has been a really pivotal year for this whole ecosystem to really become much more mature right? So we have EIP 7702 coming soon. We also have zk email now on testnet and you know soon will become available in wallets so like I know I've seen like sole wallet has been integrating it. I've seen I believe like okx has been like there's been a bunch of wallets starting to do zk email and I expect like this concept of zk wrapping institutional guardians like this also will be one of the big unlocks that makes the whole concept of multi-sig and social recovery like actually really usable for regular users right? So this year is a very good year for that and I think it's particularly powerful because it creates this really nice spectrum where beginning users who are joining for the first time and users who are like very cypherpunk and wants to hold their own all of their own keys and users that wants to do things like me and have multi-sigs and like do social recovery with various other individuals like can all be part of the same ecosystem and can all use the same dApps and all use the same wallets right? So I think this is something that's really starting to become accessible as well right? So really trying to be friendly to new users and at the same time being friendly to the values that make the crypto space be what it is at the same time and really trying to do a good job of that.

(32:41 - 50:19)

So you know this is where account abstraction is this year but still a long way to go and best of luck to all of us building to make that happen. So I think I think we have like a couple of questions so let's do just raise your hand I don't know yeah let's go on the back here. Yeah hi Vitalik, first of all I would just like to say thanks for like all the jobs you've created everyone in Singapore is here because of you.

My question is instead of doing this on the base layer on L1 why can't we do this on some L2 and kind of try it out there and then if it all works out then come to L1? It's a good question I mean I think my answer there is basically that like the Ethereum ecosystem as a whole has some pretty powerful network effects right and like we've seen examples of people trying to create do their own layer two and do their and make wallets that target specific layer twos and basically say you know we're going to create this like sub ecosystem and it just it feels like all the examples of that that I've seen just end up failing right because ultimately like people don't want to join Ethereum to use like just your own two or three bespoke applications they want to use the entire set of Ethereum applications and so for that reason like you want compatibility across a broad range of layer twos. I mean I think one other like version of this that gets brought up is like should we enshrine 4337 in layer twos first and then move it to layer one later and I think that's something that can happen right I mean there is a multiple growing L2 focused teams in the Ethereum foundation there's the roll call group there's a lot of people that have been really trying to like do this movements to try to coordinate layer 2 EVM improvements together and I think it's good to like it is good to do that though at the same time right like there's still a lot of stuff that lives on layer one and so like the nice thing about the 4337 approach is that if you do the enshrines thing on layer two but then you also just have an ERC on layer one or even if you have it enshrined on like five L2s but then everywhere else you have an ERC version that's like maybe crappier has worse censorship resistance it costs twice as much but at least you can use it like that's still already something that's much better right so I think like there there is a lot of value in like actually bringing over the existing user base and all of the existing network effects but some kind of like staggered deployment is something that where there's definitely a lot of value in it as well in one other arguments that I used to be strong but got a little bit weaker recently right is that the argument used to be that like oh 4337 is expensive so let's only use it on layer 2 where on layer 2 the fees are trivial right and what we've seen happen now is basically yeah like layer 1 itself has gotten much cheaper and then also layer 2s like the data has gotten cheaper right so like actually the data computation balance on layer 2s has changed somewhat right and so it feels like it's just actually fine to like even do some of this stuff on on layer 1 more and then you know hopefully by the time layer 1 he like really heats up again I guess wise like we'll actually have some enshrinements and all this stuff will be cheaper on layer 1 as well hello I'm Sasha I work for 1inch and my question related to swaps because we are here for the representing the swaps and you shared the user experience for the smart wallets and there was a long way for the swap there was a p2p DAXs DAX aggregators then DAX aggregators Uniswap

basically released their standard 60 76 83 if I don't remember and I saw your comments and we've just released another vision which is atomic swaps white paper and I just want to see like and now I saw your talks and I really I hear that you do understand correctly that you think that chain abstraction and basically interoperability between chains should be solved not through the um any standards for the swaps or on the layer on the layer net I don't know aside how do you see the basically the swap future I mean okay I guess uh the two things that matter to me the most right one is uh standardization at the ux layer like I think no matter what we need to have the property that you can take an address that address contains an account id and the chain and you can put that into a two field and then you can click send and if it's on the same layer one and layer two and then it does like a regular send and if it's cross layer two the wallet figures out like whatever the bat whatever is the best uh like safe cross layer swapping method that you that it can that that's going to happen quickly right so that so that's the first part right so from a ux perspective making like within a chain and cross chain feel the same the second thing that I personally care about is I believe that at the very least there needs to be some kind of like fully credibly neutral decentralized like you know like no governance no governance token a kind of backstop mechanism so that like we it's just it's just this thing that sits there and always provides a guarantee that there is at least this uh one way to do a trade that like that you're all that any wallet will always be able to do and that it will be able to do like even uh 10 or 15 years from now right and then individual wallets can uh obviously optimize on top of that and then uh individual layer twos and layer two ecosystems can optimize on top of that and and you can uh create like even uh individual swapping platforms and I mean they like to me yeah so the one one benefit of your the ERC 7683 approach in particular right is basically that it's like extremely low infrastructure right because as a user all you need to do is just send a transaction on the source layer two right you don't have to like go and talk to any IP address you don't need any off-chain peer-to-peer network you don't need like anything that can break right so like that so that's why like I see 7683 as being like a really powerful like direction to go like especially for this kind of backstop I mean the the the atomic swap approach right it generally does involve some kind of off-chain communication right and uh I mean I think if it's more efficient then like that's fine right I think uh like wallets should integrate it and it's all its job to try to give like the best experience and the lowest slippage to their users right and they should incorporate that right so like that's good right I want to see a world where users have I mean like the lowest possible slippage I also want to see a world where you know like if some totally crazy thing happens in the world and like you know like two-thirds of all the servers stop working and then another two-thirds stop working because a bunch of companies shut down the code that gets written 10 years should still be usable right so like I think our ecosystem should be satisfying both of those goals at the same time yes yeah thank you for all your hard work thanks a lot for your presentation interesting as always um so with account abstraction you would get um addresses that are specific for blockchain would that be negative if you for example send it on the wrong blockchain to that address sorry the the addresses that are specific

for the blockchain yes would it be negative for you would it be negative why would it be negative well if you have one address on ethereum but you send to that address on arbitrum by mistake right sorry so when I mean like chain specific addresses I mean the idea that like the address so like the thing that right now is like 0x whatever like that would contain both the account id which is what it is now and it would also contain the chain right so from a ux perspective like it would be like basically physically impossible unless you like start intentionally fiddling with digits and like there's even ideas to make the chain id be part of the check summing mechanism and so like you would not even be able to edit it manually unless you go into python or whatever um and and so like from a user experience perspective like you would just have this like one atomic object that just represents the idea of like this particular account on this particular chain right and so the whole concept of like sending to the right address on the wrong chain is not something that would like even be possible anymore oh let's do a couple of questions there one there test test so um you have a slide about erc 7702 where you it's it's written give existing ua the convenience feature of account abstraction but how can you ensure that ua's have all the possible feature that could exist because abstraction is also you have different type of smart wallets you know like safe have and you have and you have probably many more how how do you how do you fix how do you solve that how do you think about that yeah so 7702 was definitely meant to so it gives the convenience features right because what 7702 gives you is it basically lets you have something that is a smart contract wallet from the perspective of like an execution but ultimately it's still controlled by an eoa and so it cannot be more secure than an eoa but aside from that it can do everything right and so for that reason right 7702 is like it just is by definition able to unlock whatever like convenience account abstraction features that you want and the other thing that it could give you right is it could give you like a wallet that has a second key right so like for example you could have an ecdsa key that's inside of a hardware wallet and then at the same time you can like authorize a second key but then you can give it a spending limit right so like you could do things like that but like ultimately the security is upper bounded by the fact that there is still one single key that controls the entire account and so like that just is the fundamental limitation of 7702 right 7702 is about increasing the functionality of wallets within that box if you want to go beyond that box then what you need is you need to either switch to another smart contract wallet today or you need to like wait for potentially like if the community decides to do it a future eip that would allow you to convert an eoa or a 7702 wallet into a full smart contract wallet cool i just want to mention something on twitter side like two things super quickly what feature are important to you that make a wallet perfect that i remember it was quite viral and the other thing is that you public mentioned the stage one and i think like you want to expand on these two things because i think super interesting for the audience here sure so i mean features that make a wallet like great for me are i mean so first of all right i think like i wants to see a really good and very user-friendly implementation of the whole multi-second social recovery concept and you know the concept of having like different permissions for potentially different accounts or

at least i mean like different classes of assets or applications right so i want something like like so you know how when you go to safe and you can set up a new wallet one thing why not instead of just letting people put in ethereum addresses also let people put in an email address like you have guardian number one zero x one two three four five blah blah guardian number two can be like sir baba lot.eth guardian number three can be like you know like george at gmail.com and then under the hood it would automatically generate a zk email address that is only controllable by george at gmail.com right and then you could do that for zk email you could do that for a non adhar you could do that for like zk-wrapped miner card like basically any of these zk-wrapper institutional guardians right or and then if you have i mean like let's say yeah any any one of these blockchain-based social media or messengers whether it's farcaster or status or something else then like one other kind of like even like further goal is like you should be able to just like basically link your social to your wallet and it would automatically suggest a set of guardians to you based on who your frequent contacts are based on you know like who actually yeah like maybe has had access to their account for a long time and who actually yeah like has a wallet that you can link to um so like the one implementation of social recovery i remember seeing in the wild outside of crypto right is wechat like in wechat's account recovery it had this feature where basically in order to recover your account like you have to ask two of your contacts to send us you a particular six digit code right and like the reason why that account recovery works is because like you don't have to manually choose who your trusted contacts are it's just like based on your existing contact list right so a really good implementation of social recovery that basically like guides the user toward what choices make sense for a newbie what choices make sense for someone with like 30 million dollars and what choices make sense for someone who is you know like very paranoid and wants to trust themselves and like every kind of points in between right so that's feature number one uh feature number two is uh everything i've talked about like cross layer two a user experience cross layer two compatibility number three is built in light client so use helios to actually verify the ethereum layer one so anytime you do a call a transaction like a balance get even a transaction simulation you you ask for like client proofs and then you use helios and like you actually verify them and you do that for layer one and then you also do that exact same thing for any layer two that exposes how its like state root mechanism works so that's number two number three deep integration of privacy protocols so if you send if you create a new address and you move coins from one address to another address the default way to do that move should be through a privacy protocol if you have a wallet and within that wallet you have five different addresses the wallet should treat that as being five different profiles that you are trying to keep separate from each other this is not perfect but i think less information is always better than maximum information right you want to try to like uh look we want to have as much control and as much choice over the flow of information and which information gets publicized in which information does not as possible right so that's uh so that's number three um number four is uh i mean of course uh all of the like protection uh protection

features around preventing scams around preventing like detecting thefts detecting bad applications like that whole list of things and done in a way that's maximally privacy preserving um and then number five is uh i think this is another one of those things that's a little bit more far out is instead of like just being a metamask style thing that assumes daps are in a browser you actually try to fix the security prop flaws that are inherent in the concept of daps being web pages itself right so when your dap is a web page you're grabbing the dap contents from a server you have no idea whether or not that server was just hacked two minutes ago and the entire dap replaced with one that's going to steal all your money right so what we need at the very least is uh daps need to have code uploaded to ipfs and you should your what your browser should be fetching it from ipfs directly now brave does this but like the implementation is crazy slow and so even using brave i end up using dot eth dot limo so you want an implementation that's actually fast and then also you want you want like full supply chain verification right so you want something where if you have some way of knowing let's say i'm using ave then like you should be able to say okay well ave is ave dot eth or whatever their dot eth is and then that dot eth should itself directly control the ipfs hash and so you would actually need like a dao to approve every single commit to the uh to the ui right so you actually have this kind of chain of trust and then if the ui was upload was changed within some let's say 24 hours then like your wallet would even would even give you a warning of that so i basically really have this a strong end to end chain of trust for dap interfaces and then on top of that like actually go in and start thinking about things like oh you know can we try to make a programming language to try to make dap uis automatically yeah you know like more guaranteed to like actually do what the user intends to do then we can think about privacy protection right so basically attempting to access an outside server should be viewed as a sensitive operation in just the same way as sending a transaction should be viewed as sensitive operation right okay so yeah you know like there's basically this like really far rabbit hole that i think that i think you can go if you want to create an ideal wallet and i would love to see people like you know like really yeah try to take this challenge on so thank you