EPISODE 538

[INTRODUCTION]

[0:00:00.3] JM: Decentralized applications can be built on the Ethereum blockchain, just as the Bitcoin blockchain is a distributed append-only ledger of financial transaction history, Ethereum is a distributed append-only ledger of computational transaction history. New kinds of applications can be built on the Ethereum blockchain. Just like every new technology, we need an interface to bridge that new technology and our existing technology.

We can use a pure Ethereum browser, like Mist where we can use a Chrome extension like Metamask to turn our normal browser into an Ethereum interface. Dan Finlay is the lead developer of MetaMask. In today's episode, we explore why you would want to interface with decentralized applications and the different ways of doing so.

A few examples that we explore, simple transactions like transferring ether from one person to another, or transacting with a smart contract. My personal anecdote of using MetaMask recently for the first time was to fund a Gitcoin issue. Gitcoin is a way to put up financial rewards for people solving open source issues.

I locked up $42 in an Ethereum smart contract. It became the bounty of that issue. The issue was solved and I released the $42 from the smart contract to be sent to the developer who solved it. In this example, Ethereum served as a simple escrow service. To send my ether, I used the MetaMask plugin on my Chrome browser. If you're a little confused by all of these, don't worry, we explain it all in this episode with Dan Finlay, the creator of MetaMask.

[SPONSOR MESSAGE]

[0:01:48.3] JM: Software Engineering Daily is brought to you by ConsenSys. Do you think blockchain technology is only used for cryptocurrency? Think again. ConsenSys develops tools and infrastructure to enable a decentralized future built on Ethereum, the most advanced blockchain development platform.

ConsenSys has hundreds of web3 developers that are building decentralized applications, focusing on world-changing ideas like creating a system for self-sovereign identity, managing supply chains, developing a more efficient electricity provider and much more.

Listeners, why continue to build the internet of today when you can build the internet of the future on the blockchain? ConsenSys is actively hiring talented software developers to help build the decentralized web.

Learn more about consensus projects and open source jobs at consensys.net/sedaily. That's C-O-N-S-E-N-S-Y-S.net/sedaily. Consensys.net/sedaily. Thanks again, ConsenSys.

[INTERVIEW]

[0:03:04.4] JM: Dan Finlay is the lead developer of MetaMask. Dan, welcome to Software Engineering Daily.

[0:03:09.0] DF: Hi. Thanks for having me Jeffrey.

[0:03:11.2] JM: I did a show several years ago with another strong developer of MetaMask, who is – I believe his name was Aaron Davis, kumavis is his online handle. Back then, I wasn't really taking Ethereum very seriously. I was a bit perplexed by the whole project. Over time as I've learned more and more about it, I have really fallen in love and grown to appreciate the technical sophistication of Ethereum and what you're doing with MetaMask.

I want to just start off by saying I wish I would've recognized it sooner and I'm not even saying that from the investment fomo perspective. I'm saying it genuinely. I wish I would've started covering this space sooner, because it's really an important space.

[0:03:56.8] DF: Yeah. Well, I don't think that it's too late to get interested in the field. I'm not giving investment advice again on the fomo note. I hear people with fomo all day. Yeah, I mean in terms of actual usability, like I consider it so early. MetaMask seems like a little successful. When people call me and they tell me they think MetaMask is successful or they were using it,

I'm like, "Yeah." But in practice, like scalable applications, really secure applications, we're not remotely there yet.
There's so much interesting work to be done in this space still. Perhaps for having him on that early and for being – even curious now, because I know there is a lot of software engineers I look up to even, who they still have a almost snide perspective to it, because for many perspectives it just looks like a new derivative scheme or something like that. I really think there's so much more to it than that.

[0:04:49.0] JM: I know. I was at a conference about Kubernetes not too long ago and I was having a conversation with some really, really strong Kubernetes developers. These guys are distributed systems experts, they're leaders in the field of Kubernetes and yet, they were talking about, oh this Bitcoin thing is just a fad, this cryptocurrency thing is just a fad. I'm like, I don't understand. Is there some sort of psychological disconnect there – how are you a distributed systems expert and you can't see the importance of this technology? I guess, it just takes a real deep inspection to really internalize the importance of it.

[0:05:28.9] DF: Yeah. I think I may have gotten a head start just because of the problems I was interested in, happen to be well-suited to the blockchain. Because when you're learning a new thing, you're trying to come up with examples in your mind. When I was first learning about Ethereum, my big thing was like, I wanted digital voting, I wanted e-democracy, I wanted e-deliberation.

I was sick of these Facebook debates where they just keep – everyone is just having the same argument over and over again. Right about now, everybody is posting their five favorite gun control talking points or whatever. There's just a lot of ships passing in the night and I had this fantasy. I was like, "If we could keep track of what everybody thought and you could just look at the map of what all your different friends think and stuff like that."

To be fair, you can do this with a normal website and there's a cool site called Kialo that I'm excited about that's doing something like that. As soon as you start using those poll results to do anything serious, like if you were going to say like, "Okay, our poll results are getting pretty good. Let's make our corporation use these as our voting mechanism." Suddenly, the security requirements like live up dramatically.

Suddenly you're like, "Okay, what's the best way to stay secure and auditable and how do you make sure people aren't using other people's votes and they're only voting once, and the sys admin isn't manipulating the database and serving you the results they like?"

I think right away, the notion of applications that live in a digital comments like that appealed to me right away, like an impartial vote counter was I think the first use case that struck me and got me really excited.

[0:07:06.2] JM: Let's get into what you're building. MetaMask is a browser extension for interaction with the Ethereum blockchain. Why would I want to do that?

[0:07:14.6] DF: Yeah, that's a great question. Blockchains before, well basically the big one was Bitcoin and then everything was a bunch of clones of Bitcoin before Ethereum. Well, there were other ones. That's a generalization, but with Ethereum added was instead of just being a ledger for currency, it's a whole computer. Now we have this digital comments, this public computer or however you want to think of it. I mean, those are really the idealistic ways of thinking of it.

It's like best case scenario. It's a digital comment. It's a computer that runs in public. If we have that, well I was a web developer and I'm like, well the first thing I would want is I would want it as the backend for my website.

Right away, first thing, like I was saying, I was like, "Okay, let's see what it would take to make a voting app." What do you know, the Ethereum foundation's first three tutorials, it was like a hello world, it was a token machine and then it was a digital democracy, or I think there was a Kickstarter in there; peer-to-peer Kickstarter, nobody takes a fee.

Then voting. I was like, "Okay, great." I went through the tutorials. I made my voting; smart contracts, which is what they call the programs that run on the blockchain and right away we hit a wall, which is that to have a user interface to a blockchain application is very different from having a centralized service.

There's no database that just has your user ID and a hash of your password or whatever. Instead, a blockchain account it always involves cryptographic signatures. An account is essentially a key pair. This kind of pushed a need for adding key management to your user interface. For us being web developers we're like, "Okay, we need key management in the browser."

In retrospect, it just feels like why didn't we have key management in the browser? People keep on coming up with new creative uses of being able to sign challenges, because now when you visit a site with MetaMask, that site it can just issue a regular cryptographic challenge. It could say, "Do you mean to sign into this page?" There is a few sites experimenting with sign-in. That's very experimental. We need to add some features to make that very secure, but you've got cryptographic challenges and the user can read it and then they can hit approve.

We show a confirmation box. The keys are really in the user's control. It's not like an SSL keyword. Your browser generates one, or it doesn't really tell you about it. It just tells you this page is secure. This is a very hands-on key management. Now you have the power to approve signatures and the original pitch, approve Ethereum transactions. These are function calls. These writes to the public database. This could be casting your vote. These could be pledging to a crowd fund, etc.

[0:09:37.5] JM: One simple way of interacting with the Ethereum blockchain is to transfer money around. You could also do this on the Bitcoin blockchain with a Bitcoin wallet. MetaMask can be used as a wallet, it can also be used as many other things like sign-in. Let's just talk about the wallet, because that's almost like a base case use example. Why would it be useful to have an Ethereum wallet in my browser?

[0:10:06.5] DF: I mean, honestly if you're just using it as a browser, I don't think it's much better than any other wallet, but it is a wallet and it's right there in your browser. The thing that immediately distinguishes itself and really in some ways, I don't know why Bitcoin hasn't made an equivalent, or basically every blockchain could benefit from something like this and maybe someday well they can support.

We inject an API into every website, so now websites themselves can suggest the transaction to the user. For example, asking for a tip, there is a video tutorial site by always be coding, Jamie Lee, and you pay for the videos right there. It's all just client side. He's just got some JavaScript that doesn't show the video until you tip with MetaMask. He's not taking it too seriously, but he's made decent money with that.
He just got a little JavaScript and says web3 send the transaction to his account from your account, your account is revealed to the site when you login, and then asking for a certain value. Then MetaMask shows you what the site is asking for, and then you say, "Okay." It's just like a normal wallet and that you can send, you can manage your balance, you can do all these stuff. By including this API, we make it so much easier for websites to interact with you as a wallet-holding individual on the web.

[0:11:15.6] JM: When I install MetaMask for the first time, I create a vault and a vault is a set of wallets. Why would I want multiple wallets? What am I creating when I create that vault and what's going on during that first onboarding process with MetaMask?

[0:11:33.8] DF: Yeah, that's a great question. We basically borrowed some standards that were popular from the Bitcoin community when we started. We're using this thing called the seed phrase. We generate you 12 words, eventually we might up it to 24 just for security, but the 12 random words and this is your password, but it's more your super recovery secret. From these 12 random words, we've got enough random data that we basically just increment it a little bit and we can generate as many accounts as you want.

That's nice, because if you can just write down these 12 very human comprehensible words, we can restore as many accounts as you want. You're asking, well what are those accounts? Why would I want multiple? Well, modern blockchain are like totally public. Not Zcash. You've got Zcash and you got Monero, there's some more privacy-centric chains and eventually there is talk that privacy features will certainly get rolled into more and more blockchain including Ethereum. For now, it's all just so public. You can go on a block explorer and you can see what everybody is doing all the time.

If you know somebody's account address, you see like – it's equivalent of seeing their whole bank account address, their balance and everyone they've sent to and everyone they received

from. The only what that there is even a semblance of privacy on a modern public blockchain like Ethereum or Bitcoin is by having many accounts.

It's not anonymous, but it's pseudonymous, like pseudonymous. You can generate a whole bunch of accounts, you can use them for different things. I use one account for my public identity and then I use another one for my cryptokitty collection and then you could have another one for your local community organization, or whatever.

[0:13:08.0] JM: MetaMask lets me associate account or multiple accounts with my browser session. This is in some ways like the different Google accounts that I might use in a Google Chrome session. An account identity is a public key. I have a different public key for each wallet and my public keys are also – there's a one-to-one mapping between a public key and an account, if I'm correct. Is that a good way of thinking about things?

[0:13:41.9] DF: Yeah. Then that actually that leads really nicely into another interesting point, which is that – yes, the normal kinds of accounts that MetaMask supports today, those are all key pairs. Every account, we're storing one private key and then we're exposing the public key to websites you visit.

There's a little bit of abstraction going on in there, but basically it's a key pair. Every time you send a transaction, the way that that transaction is valid to the public validators that are looking at every transaction is that your transaction is signed with your private key. That's one of the basic building blocks of the whole blockchain.

However, there is another kind of account also that we don't support yet, but actually some people have experimented with sometimes to some hazard. You could make your account be a smart contract itself. Long-term, we are very interested in doing this, it's just the thing that we have to be very careful about and we can talk about smart contract security in a little bit.

The idea would be that if you use a smart contract as your account, then your local keys they can just be permitted keys and they can be more transient. You could expire them, so the blockchain can be – like you were own access control list that you maintain. You can have other rules too. You could make multi-sig right out of the smart contract.

For example, you could say to send a transaction with this contract account, I need a signature from both my computer's key and the key on my phone, and maybe a key from the central service that does a two-factor check with me. You can build your own security using smart contracts. Once it's a smart contract, then you get a much more permanent address and that's really cool, because well right now private keys it's so scary. If you lose it, you lose everything. That's why it's like the Wild West right now. It's so dangerous. Once you have a more constant one and you can have recovery schemes. You could say like, if I get five of my eight closest family members together, they can generate me a new key. Once you have social recovery and things like that, these are features we're really working closely with uPort with. It's another consensus project. Once we have those, then you'll have a more permanent address on the blockchain and that will open us up to all sorts of identity and reputation things.

You'll be able to issue identities freely. You'll have identities that are maybe isolated for one purpose, like your gaming identity. Then you'll be able to associate it in confidence with a site as needed. You could go to a site that says, "Well, this game you have to be over 18 to play it." You could basically prove it, because the identity containers can have hash links and signatures and things like that that are not stored on the public blockchain, but you can disclose as desired.

[SPONSOR MESSAGE]

[0:16:24.2] JM: Sponsoring today's podcast is Brother International. The Brother QL-1100 label printers offers seamless integration of label printing functionality into your app development, requiring nominal coding. The SDK is well-documented, easy to use and works with most operating systems, including Windows, iOS and Android.

You can get the SDK at ptouch.com/ql1100. That's ptouch.com/ql1100. That's promo code is in the show notes for this episode. You can check out the Brother mobile app hackathon being held on March 23rd and 24th in Irvine, California. Brother is looking for teams of designers and developers to submit software apps, or prototypes that integrate with a Brother QL printer, or Brother rugged jet printer or labeler. $3,000 are offered in prices and you can sign up now through March 23rd. Get all the details at ptouch.com/ql1100.

Thanks to Brother International for being a sponsor of Software Engineering Daily.

[INTERVIEW CONTINUED]

[0:17:42.4] JM: In the simplest case, what would my login system look like? If I want to login to MetaMask, want to login to one of my accounts, what exactly is happening?

[0:17:53.6] DF: Yeah. Basically, MetaMask doesn't do this perfectly right now. I almost want to tell you where we're headed in the next month or so, because we don't have the best privacy story right now. Right now when you login with MetaMask, we basically put your public key onto the web3 API in every page you visit. That's not acceptable at all for a real production thing.

Basically, we've got an adoption a lot faster than we expected to. In the ideal case, you get to a site and they can have a login link, like a site has today. Then we'll show you a prompt and we'll say this site wants to see who you are, who do you want to appear as, and it could look a lot like your Google account list, and you should be able to know very clearly what details are publicly associated with those accounts and what details you are able to disclose that are associated with that account.

You might choose to say your name, or you might choose to only reveal your age, or your location. The goal would be that we have an account list, and from that list you can compose the minimum required login credentials. In the current form, it will just be a list of ideas that you've nicknamed, but over time we're going to have richer and richer identities associated with these; pictures, names, associations.

Once you click login, then that account's address is going to be revealed to that site. Since that's a blockchain address, they now have the entire blockchain, this huge public database to scan and scour for anything they want to know that's public about that address. If you've logged in before, they instantly know everything about you. If you sign a challenge, they can really verify that it's you.

[0:19:26.6] JM: Fascinating. You could imagine a decentralized Twitter, where all of your tweets are on the Ethereum blockchain. By logging in, the web application would be able to scan the

history of the Ethereum blockchain and find all your tweets, so that it could give you a history of all your tweets.

[0:19:48.3] DF: Yeah. That's literally already a project. It just doesn't really scale right now. It's called Leeroy, Leeroy.io. Not safe for work. Somehow it's been largely taken over by the porn industry.

[0:20:00.1] JM: Somehow.

[0:20:01.2] DF: Yeah. It's a public blockchain. Long-term, I think public blockchains are as dependent on client side filtering as they are on all these other security, because it's a public ledger. If you're letting anyone post there and it's sensorship proof, then the only sensorship or filtering you have is client side. Leeroy hasn't quite nailed that yet and it's not doing anything for scaling. Right now, it's all every tweet is on the blockchain.

It's expensive. I think it's like – I don't know, it's probably 25 cents a tweet right now, but it's globally verifiably you and nobody can stop it. That's cool. Yeah, there's obviously a lot of scaling challenges going forward to make something like that really let's say contend with Twitter. Because yeah, Twitter is free, it's so easy. The occasional ads and maybe the suspicion of sensorship is not remotely enough to keep people away from it at the moment. Yeah, the user experience of blockchain stuff has to improve a lot before it's actually contending.

[0:20:54.1] JM: There is a reasonably large design space of different approaches you can take to building a client that interacts with a blockchain. The two basic areas of the designed space are you have light clients and full nodes. Full nodes have an entire copy of the blockchain. Light clients just have perhaps the block headers, so it's a condensed format of the blockchain, just enough information such that you can verify transactions.

It's like being a consumer of a bit torrent network, but not necessarily being a full contributor to the bit torrent network. You're not necessarily a seed on the bit torrent network. MetaMask is actually neither of those, because if I understand correctly, you use today at least there is a remote client that is a copy of either – I'm not sure if it's a remote light client, or a remote full

node that you're using as your reference to the blockchain. Maybe you could just tell me the MetaMask approach to light client, versus full node, versus what you're doing.

[0:22:08.3] DF: Yeah, great question. You're completely right about architecture. MetaMask's approach I think was we came to Ethereum, we immediately saw this total show-stopping user experience bug, of just not being able to login, or have a user interface. We decided to work on that first. That meant, we compromised on some of the cool security that's possible from a blockchain early on.

For example, yeah we're not even validating block headers or anything like that. When you browse a site with MetaMask by default, we're drawing the data from Infura, that's our backend provider. They run very full node. They run the fullest kind of node you can run, called archived nodes. They keep around stale data that nobody even needs to keep around anymore, just so that you can do historical lookups through MetaMask.

The long-term, we're continuously looking at further decentralizing. We didn't let our current browser-based light client research stop us from iterating on the user experience, but we do have a project in the works for basically continuously further decentralizing, adding what they call the trustlistness of the blockchain into the MetaMask.

A lot of these is going to get a lot easier as Ethereum moves on to proof of stake too, because today if you want to iterate over the block headers, you just need to iterate down as many block headers as you want to statistically insure yourself against. There's really no finality, so there's some uncertainty of what's really a safe client.

With proof of stake, there is finality. If you have a current list of validators, it's like having a list of a bunch of different people whose response are going to cross-check against each other. From them, you're able to get a much more current block header that's current. You don't have to download as many block headers to synchronize.

We're also working on doing some basically pure gossip in the background of the browser. We're just now starting to do some work with the protocol labs team to experiment with a mesh

network behind – in the background of the browser, so it's going to start – we're just going to do a little ping pong experiments, everyone is to check their resilience of this network.

We're a network now of a million peers, so that's actually more peers than are on the Ethereum network today. By having bootstrapped the user experience, there's some hope that we might actually create a more resilient peer-to-peer network right out of the gates once the technology catches up with it.

[0:24:33.8] JM: I got a crazy question and then we'll get back to the MetaMask basics, but this is something that I've been thinking about a little bit. When the IPFS networks gets fully rolled out, could you have the idea of a full node become decentralized and broken up into the IPFS network, such that the overall space required to contribute to the full node verification process is less intense per node? That you could still do a proof of work system, but the proof of work would be across a distributed instance of the blockchain, a distributed tamper-proof instance of the blockchain that's broken up across IPFS. Has anybody looked at that?

[0:25:23.5] DF: Yeah. Actually, I think some of the first work on that was done by the IPFS team themselves. Or I heard that wanderer contributed and then why are you sleeping contributed. Right now, I think MetaMask has what we're calling like a sub-spoke, it's a sub-project led by our very own Herman Young. He's calling it muskatala and it's all about gossiping Ethereum state data over IPFS.

The short answer is that it doesn't fundamentally make block data more available than if everyone was running a normal Ethereum client, because a normal Ethereum client is already generous with data. It's always saying, "Oh, you want that block? Here's that block?" In many ways, it's like what IPFS would be doing. One of the big differences for why we're pursuing it is that IPFS uses a different transport, lib P2P instead of dev P2P, which is it's a transport layer and it's transport agnostic. We can run it in the browser.

The Ethereum basic protocol doesn't work with web RTC, so basically right now any Ethereum client in a browser needs to talk to an HTTP client, which makes it very centralized. As soon as we were able to talk over web RTC, we actually have a browser peer-to-peer network. We're experimenting with using lib P2P to gossip all that state data.

Yeah, you can still do all the same stuff proving the work backwards. It's just a matter of you get a recent block and then every block references the last block and then you go, you request that and you request that and so forth.

IPLD has some really cool methods for requesting deep into a Merkel tree, so you can say like, "Give me that leaf and it will give you all the chunks leading up to it." That's really cool. One of the weird parts about doing Ethereum or blockchain stuff over IPFS is that blockchains fundamentally have – well, they need a gossip layer where you get the latest block. IPFS doesn't by default have a way to say latest. Everything is by hash, so you have to know about it already to request it. They also have a pub-sub system, so we're going to surely sort through that too.

Essentially yeah, it's just another transport. It's just another peer-to-peer transport for gossiping data. Yeah, once you teach it how to speak the language of a different hash tree, yeah you can use it for pretty much any blockchain.

[0:27:30.2] JM: Yeah. Maybe I wasn't crisp with my question, but I was thinking more in terms of could you imagine a world where instead of having these full nodes that are centralized in China, or wherever you've got the cheapest electricity and we've got these giant server farms, because they can afford to run the full nodes, we could have perhaps light clients doing mining and their transaction verification instead of scanning their local disk, they're scanning a distributed instance of the blockchain that's distributed throughout IPFS. I don't know. I guess, this is so far flung. It's not even worth speculating about.

[0:28:11.6] DF: Yeah, it's interesting. I mean, one of the basic problems would just be the latency of asking for that data. There's no reason theoretically it couldn't work, it's just a latency thing. Usually miners want the lowest latency, because it would give them more time to propose a block. As we go to proof of stake, less of that block time is going to be dedicated to mining, so you could give that to the light client thing, but then you lose one of the great benefits of getting proof of stake, which is shorter block times and just lower latency network in general.

I do think that there is going to be a lot of value to looking for ways to make it more and more distributed. As we have things like sharding – with sharding, my understanding is that actually it's going to be assumed that even what you just call a full node – I actually just got a presentation on this. I'm not an expert on it, but sharding is a strategy for scaling the blockchain.

Since you imagine scaling, the point of sharding is that you have many, many of these instances of the Ethereum virtual machine. A single validator, they don't know which instance they're going to be block to block, so they get a preview a block or two ahead. Then they have to gather all the data that will be required to propose a new valid block.

It actually maybe that in the future, maybe even in the year or so, full nodes are actually doing a little bit like what you described, they can't be aware of all data. It maybe that actually when somebody broadcast a transaction, they actually attach the state data with the proof of its validity with that transaction.

Then a miner actually doesn't need a full copy of the state. It's just anyone who wants to interact with state and needs to be able to prove that that state is valid, so actually now that you mention it, you're reminding me that I think that's on the roadmap.

[0:29:47.8] JM: All right. We will see. Back to the present, one thing I used recently that I was actually blown away how cool this was. This was my first really big wild moment with Ethereum was I used MetaMask to setup a Gitcoin issue. Gitcoin is this project where you can put bounties on Github issues. For example, we've got the Software Engineering Daily open source project, which is this platform that there's a bunch of open source contributors working on and we want a forum built.

People who are listeners to the show can come and hang out and interact with our forum and we want to write it from scratch, because whatever, why not? I posted a Gitcoin issue, which if I understand it correctly I'm setting up a smart contract that I deposit some amount of money into, some amount of ether and the Gitcoin system will release that money to the person who fulfills that Github issue.

Somebody picked it up. Somebody from Vietnam right now is working on that issue that I created to create a forum for software daily. When they're done, assuming they do finish, I will click yeah, they did the job. They set up the forum and they will get my payment. I just think that is incredible. If I understand it correctly, explain what is going on on the backend why is MetaMask important to this process.

[0:31:24.2] DF: Right. MetaMask is important there for the same reason any Ethereum website uses MetaMask. It's just because when you're posting that bounty, you need some way to post it. Now if you're going to do with, let's say it was a Bitcoin smart contract which is – it's more difficult to code in my opinion, but Bitcoin people would say it's possible. The site would have to do. They'd say, "Okay, well here is a Bitcoin transaction that we'll do this broadcasted."

They might give you a QR code, which would be the easiest thing that Bitcoin came up with. Then you get your phone out and you scan the QR code. That's not that bad, but by integrating the wallet directly in a browser, we made it so you can click fund this issue and MetaMask just asks you.

We're trying to make it as natural as possible. People are used to say the Paypal pop-up saying, "Do you want to send this?" We want to harness those patterns and they will say, "Okay, you want to fund this?" Over time, we're going to be able to make it more and more comprehensible. We're going to say like, "Oh, you're interacting with Gitcoin. It's verified." You could use an alternative UI if you wanted, right?

You can see this is a Gitcoin bounty and then it says exactly how much you're going to send and you say accept and you never have to trust anyone with your account, no one can freeze your account. You're in total control of those funds. You have a cryptographically provable history of all the funds you've given out, which is what I think is one of the cool things about Gitcoin. Anybody could just post a bounty and then not pay off on it.

If you've got a blockchain account and you've got a history, somebody can actually look at that history and they can just say, "Oh, look at that. They love the bounties they've posted. They paid out most of them. I can read the discussions on the one they didn't and it looks fair to me." You

have just this more auditable trail, which establishes a little more trust and there's not really a middle person taking a fee, because it's just a smart contract holding your funds.

The only fee is just the network fee that you're paying the peers for validating your transaction. I love Gitcoin. We started using it at MetaMask and we seriously – there were ones where we put a trivial bounty on some issues, almost like – I don't know, it felt like a joke. I was hoping other people would pitch in on it. I thought the community would say – they would use Gitcoin to vote on issues, but instead they were just people picking up tiny bounty things.

I felt a little bad about it. Yeah, and then returning quality work is a little amazing. I look forward to seeing how much we can get done this year. Yeah, some of our backlog that we weren't getting to, we've started tightening up with Gitcoin.

[SPONSOR MESSAGE]

[0:33:55.6] JM: Today's sponsor is Datadog, cloud-scale monitoring and analytics platform. Datadog integrates with more than 200 technologies, so you can gain deep visibility into every layer of your stack and other data that you're interested in tracking as well. For example, you can use Datadog's restful API to collect custom metrics from your favorite data sources and analyze trends in Ethereum prices over time.

Start a 14-day free trial land as a bonus, Datadog will send you a free t-shirt. You can go to softwareengineeringdaily.com/datadog to get that free t-shirt. Thank you to Datadog for being a continued sponsor. Get that free t-shirt at softwareengineeringdaily.com/datadog.

[INTERVIEW CONTINUED]

[0:34:54.1] JM: I'll be fascinated to see how much we can get out of it too. I mean, the optimistic vision is like basically you put money in and you get good code out. That's magical. That's unbelievable. Talking a little bit more about the different ways that people can interact with the Ethereum blockchain, with smart contracts. You've got MetaMask, which is a Chrome extension. There's also Mist, I just interviewed Fabian from Mist.

That's slightly different, because Mist has a full node running under the Ethereum browser that is Mist. Mist is an Ethereum browser. I know there's some other approaches to this. What are the pros and cons of that? What's the advantage of being a browser extension versus being a entire browser built upon an Ethereum full node?

[0:35:47.9] DF: Yeah. Mist was the original thing we were cloning. When we originally pitched MetaMask we're like, "It's like Mist, but it's like Mist lite or something." We almost felt like we were like not as good as Mist, because Mist is a full client. For the crypto-idealists, running Mist was the cool thing to do, because you've got a full local blockchain client and you're doing all these.

On the other hand, we always let MetaMask users point a local blockchain client if they want. You actually can basically do the same thing. For all basic intents, they were the first comer and they set the bar for what this wallet browser was going to be like. Yeah, we just reduced the friction a whole lot by adding it right into your normal browser. Now you can just hop in upon sites that integrate MetaMask and take advantage of them right away.

Then actually, recently it's come to our attention, there's also the security benefits to not building your own browser. By just being baked into the browser somebody is already using, we benefit from all the security updates that that browser is just naturally enjoying.

For example, Mist is built on electron. Electron it's built on chromium, but it's built on an older version of chromium. It's not like an urgently updated copy of chromium, the way your chrome browser is. Mist is considerably behind, just because behind chromium updates. Just because electron is basically – they're updating it there. You've got a dependency tree and it's just not updating as fast.

There is some basic vulnerabilities. I don't know the exact details on all of them, but yeah, like I saw some security nerds saying MetaMask was actually the more secure option for some of these browser issues. That was a wake-up call, because I think for a long time we thought of it as like, "For people who aren't taking it seriously," but it might turn out that browser extensions are a great way to add peer-to-peer networking into the web experiences.

Actually, we're seeing more participation from browser vendors to invite more peer-to-peer technologies. Firefox released a couple additions to the web extension protocol, that almost seem like gifts to us. They didn't contact us and say, "Hey, MetaMask this is for you." They added a couple features, like the ability for an extension to provide IPFS dat and secure scuttlebutt URLs.

They're not including those clients in the browser, but they're leaving it open that a web extension could bake in one of those clients and then allow people to visit sites off those peer-to-peer protocols. That was like, "Oh, we're really all on the same team here." This is an old web, new web. This is like –

[0:38:19.1] JM: We did a show about scuttlebutt a while ago. That's yet another one of these things where I was like, This seems really cool." Well scuttlebutt I actually took a little more seriously than when I did my original shows about Ethereum, just because I had seen some peer-to-peer stuff start to take off. I was primed to take it more seriously. Scuttlebutt, that's a peer-to-peer social network. You could think of it as like a peer-to-peer Slack. You're saying that Firefox built-in support directly for scuttlebutt?

[0:38:50.0] DF: It's not the full client and protocol. What they did is – a web extension has access to the URL bar, or it can ask for permission to access the URL bar. You might notice some extensions, they'll be like search enhancements or something. You'll type something into the search bar and it will go to the extension or something. I don't have a good example of that offhand, but that's part of the extension API.

They added the ability for you to do SSB://. They gave the ability for extensions to take those and do something with them, basically inviting a web extension that is based around enabling a peer-to-peer web. It feels like our wheelhouse. I mean, our plates are very full. When we think about what we want MetaMask to be, we want it to be a peer-to-peer web extension. Maybe it doesn't all have to be in one extension.

It would be amazing to see a secure scuttlebutt extension come out and then see people interact with both protocols. Because you could load peer-to-peer data off secure scuttlebutt, which is in my opinion actually a great protocol for a basic decentralized Twitter. I wouldn't base

my funding model off of its likes, because it doesn't have that double spend resilient. For peer-to-peer data integrity that's identity associated, it's great. It's free transactions and stuff and really great peer-to-peer caching and stuff.

You could be loading your peer-to-peer Twitter off of secure scuttlebutt and then you could be tipping over the Ethereum network with MetaMask, and that sounds so cool. That's just using existing browser extension APIs. That's a Firefox beta, but I think it will shoot.

[0:40:17.8] JM: I want to return to a basic example, because just knowing the listeners there are some people who are listening who are new to the space and I try to give people every chance to catch up, and myself I love to have sort of introductory refreshers all the time.

Let's talk about basic transaction on MetaMask and perhaps the most basic transaction might be sending ether from one wallet to another, or calling a smart contract. Let's just say we're sending ether from one wallet to another. This illustrates some Ethereum basics, as well as how MetaMask works specifically.

Again, I can login with my public key onto one of my accounts on MetaMask and I want to send ether to you for example for a cup of coffee or something like that. How do I do that in MetaMask and what is happening under the covers?

[0:41:14.1] DF: Okay. The most basic one, you're not even using a website. MetaMask provides normal wallet functionality right in it. You'll see a little fox in your browser bar, you click it, you see your current account and its nickname and you see your balance and it's equivalent in your local currency. Then you have a big send button. When you click that, you get a little send form and you can either paste in your friend's address. Or if you're using the Ethereum name service, sorry for getting a little advanced, but it's a smart contract that does name resolution.

If they have an ENS name, you can type their ENS name and we'll resolve to that too. Then you just type how much you want to send to them. You hit next and then we've got – maybe the scariest part for new users about an Ethereum transaction, I'd love to say that you just hit send and forget it. There's this one other thing that we haven't completely abstracted away and it's gas price.

Every transaction on a blockchain today has a notion of a transaction fee. That transaction fee, it's basically you're like bidding for the peers to process your transaction next. Normally, the blocks are not full so a low bid is fine and it will get processed, but sometimes something popular is happening, whether there's a big popular crowd still at the moment, or a new app just launched.

Actually, I should be more acquainted with the moment we're on. It seems to be working well and have I – I know when something is going very bad, but things seems quiet lately. We try to recommend a gas price that will be mined in the next couple minutes, basically. We try to recommend the cheapest price that will get mined soon, but you can adjust it.

Actually, we're working on adding a graph for projecting like how long are you going to wait if you pay this much less? This is black magic, because you're estimating the results of an auction that hasn't happened yet. We've got some decent algorithms that have a decent track record, so that if we're a little conservative with our estimates, we'll be able to say, "Okay, for 10 cents you will have your transaction mined in the next three minutes probably."

Then once you've set your gas price to what you like, then you hit send and it will say pending for a little while. Once it says it succeeded, it means we've seen a block processed that had that transaction, which basically means it's very unlikely to be reverted. If you wait another minute past that, it's basically locked in stone.

[0:43:22.7] JM: I know we're near the end of our time. Gas price and gas limit, these are terms that I think could use even more disambiguation. Can you dive a little bit deeper into how gas prices calculated and what people need to know about gas?

[0:43:37.4] DF: Yeah. That's a great question, because gas is one of the fundamental innovations that made Ethereum possible. Basically, gas is the smallest indivisible unit of computation cost. The reason this was necessary where it wasn't in Bitcoin is in Bitcoin, you're just sending a transaction. Pretty much you're just sending Bitcoin every time. Every transaction is basically the same cost.

Ethereum though, you're running a computer program. You've got the whole halting problem. I actually remember Aaron talking about this when he was on your podcast, but the halting problem being that when you started computer program, there is no way to know when it will end other than just running it.

If you're running some computation as a service for people, you need some way of basically saying, "Okay, look. That's more than we agreed to." That's what gas limit is. The gas limit is the maximum amount of computation that you're willing to pay for when you send a transaction. Then gas price is how much you're willing to pay per unit of gas. If I'm willing to pay 2 units of gas and this computation takes 20 gas units of computation, it's going to cost me 40 basically ether units, let's say. The finest unit is called a way, but you don't really need to know that.

We usually refer to a gas price in G way today, that's giga way. That's just because that's around the order of magnitude that's popular to pay today. When you send a transaction with MetaMask, we automatically calculate what is likely the gas limit that's correct. We're actually thinking of hiding it, because it usually just confuses people.
We're able to simulate a transaction and tell you a gas limit that's reasonable for that transaction. When you send a transaction, you're not going to overpay for it if there's an error. When there's an error, they can just burn up all your gas limit. That's a concern. The gas price – that remains the more user interesting parameter and that is how much you're willing to pay. The miners who are in charge of deciding the order of all transactions, you know the rational ones anyways, they just look at the most expensive transactions first. They look at the ones with the highest gas price.

They don't know how long it's going to take to compute it, but they want the best thing for their buck. They want the most money for every step of computation they execute. For the most part, gas price it's a bid and an auction where you're currying favor among the miners, or validators to mine your transaction sooner.

This could be important when there's something that's time sensitive, like there's an auction. This happens all the time when there's a crowd sales. Some crowd sales they structure themselves on a time limit and that forces people to be in a big rush. I really hate it when people do that, like please come up with more creative structures for your crowd sales.

When that happens, people basically just have to crank up their gas prices. I've seen people spend ungodly amounts of money on their transaction fees just to get in sooner. If you think a purchase is going to be worth a million dollars to you, then maybe $50,000 is worthwhile as a transaction fee.

I don't think so. I think that if you're very informed about the gas market, you can pay barely enough to get a fast process without overpaying. I think people still overpay a lot just because they're afraid of the gas market. If you go to a site like Eth gas station, you can see some pretty good metrics and stats about it.

I also have a Github project I call Eth gas price visuals. It's just a scatterplot of gas prices. Even that, I think helps give a tangible sense of where the gas market is, because really you just want to pay just enough to definitely be in and you don't want to overpay.

[0:46:57.4] JM: All right Dan, well I know we're up against time so I'll let you go. We should definitely have you again in the future. I'd love to talk more about MetaMask and other developments. I'm sure as you've said and after talking to you it's very early days and there is plenty more developments coming down the pipe. I'm looking forward to talking again in the future.

[0:47:14.3] DF: Yeah. It's definitely moving really fast. I don't even know what we would have to talk to about next time. I'm sure it would be interesting, so I would be happy. Anytime.

[0:47:21.8] JM: Okay. Sounds good, Dan.

[END OF INTERVIEW]

[0:47:27.9] JM: If you are building a product for software engineers, or you are hiring software engineers, Software Engineering Daily is accepting sponsorships for 2018. Send me an e-mail jeff@softwareengineeringdaily.com if you're interested.

With 23,000 people listening Monday through Friday and the content being fairly selective for a technical listener, Software Engineering Daily is a great way to reach top engineers. I know that the listeners of Software Engineering Daily are great engineers, because I talk to them all the time. I hear from CTOs, CEOs, Directors of engineering who listen to the show regularly. I also hear about many newer, hungry software engineers who are looking to level up quickly and prove themselves.

To find out more about sponsoring the show, you can send me an e-mail or tell your marketing director to send me an e-mail jeff@softwareengineeringdaily.com. If you're a listener to the show, thank you so much for supporting it through your audienceship. That is quite enough, but if you're interested in taking your support of the show to the next level, then look at sponsoring the show through your company.

Send me an e-mail at jeff@softwareengineeringdaily.com. Thank you.

[END]

