TFTC 431

**Dan:** [00:00:00] We're going,

**Marty:** we're going. Dan Gould, welcome to the show.

**Dan:** Thanks for having me, Marty.

**Marty:** Well, thanks for joining. I'm very excited about what you're working on. You recently release the pay join development kit to make it easier to integrate, pay joins into Bitcoin Wallet software, just Bitcoin projects in general, and Pay Joinin is something, pay, join, pay to Endpoint, whatever you wanna call it, is something I've been writing about in the Bent for at least four years now, and something that we've talked about on Rabbit Hole Recap and Tftc many times throughout the year, the years, and I think it's something that's very important, but for some reason or another hasn't gotten a lot of adoption since it's been available to people.

So I guess before we jump into PayJoy and the development kit, That you're building. Why don't we learn a little bit about you, [00:01:00] your history in Bitcoin and why you decided to focus on this one area with page one specifically.

**Dan:** Sure thing. I started in Bitcoin development in late 2016. I was really fortunate, so I found Bitcoin on the internet and interested in it as money, of course, as we all are.

And when I was in college, I found tumble bit on Reddit and looked at the byline, which was Ethan Haman. So tumble bit's, one of these privacy protocols early days, 2016. And Ethan at the time was my teaching assistant. He was the teaching assistant for my probabilities class. So my mind was just totally blown that like real people actually worked on this stuff that I could go talk to.

So during winter break, I had the opportunity to go into the lab and work on tumble bit. I really wanted to contribute to open source and bitcoin. Captivated my attention and I just made a really crappy pr [00:02:00] uh, to change configuration settings on that. And that turned into this whole, you know, journey down the rabbit hole and multiple open source projects and focus on Bitcoin privacy.

So from there we went to Stratus in the uk they wanted to implement tumble bit, but this was before the 2019 SEN guidance on what you can and can't do with Bitcoin privacy. So that kind of ties into what's going on with ARC now and which privacy protocols work and don't work. So out of that group that was working on tumble bit came wasabi, uh, and Nicholas went on, he was working on that too.

And he went on to work on BTC pay. So I was using, I was traveling at the time and using iOS. For Bitcoin and was really disappointed. I couldn't use Wasabi for that. [00:03:00] So I forked Wasabi into Chain Case, which is an iOS app that does some coin joins. Kept thinking, okay, does this really solve the problem?

Does this really solve the problem? I don't think the application level solves the problem doing it in an iOS app. I don't think iOS users are the reason we don't have Bitcoin privacy is what it comes down to. So started collaborating with some of those developers that were working on wabisabi and looking at it more from a systems perspective.

When I realized Pay Join is a really simple protocol that gives you reasonable privacy, and it's incredibly convenient because it works just like a naive Bitcoin transaction would from a user's perspective, and started learning Rust and that turned into the Pay Join dev kit. And the point of that is that, Rather than create a piece of wallet software that has privacy, let's just bring privacy by default to all of the Bitcoin software and services that exist.[00:04:00]

**Marty:** Yeah. And you mentioned, uh, fiend guidelines that were clarified in 2019. What, what did those guidelines say explicitly? Like what, what type of structure did that give builders?

**Dan:** So I don't have it directly in front of me, and I'm not a lawyer, but the overarching theme was if you're not able to unilaterally spend someone's money, then you're not a money service business, which means you don't need to apply for licenses, and it means you're a networking business.

So you can run a regular software company and be a transaction coordinator, like a Atomic Swap coordinator or a Coin joint coordinator, and you're in compliance with the law. So that let all sorts of privacy innovations happen as long as you don't have that custody of user funds, which is awesome because it means that users can get this help from [00:05:00] the cloud, if you want to call it that, and their clients can still have really excellent privacy guarantees.

Mm-hmm.

**Marty:** And so let's take like a broader view on this, just like coin join more generally as sort of a transaction type to, to create better privacy assurances for users as they transact on the network. That's, I mean, I'm sure many people have listened to Tftc and Rabbit Hole recap, understand coin joins quite a bit.

But for anybody maybe new to the show, what is a coin join? How have, uh, implementations like join market? Wasabi and Samurai implemented it to date. And how does PayJoy differ in the way that they're doing it?

**Dan:** Yeah. I was listening to you and Matt on our HR yesterday, get into a little bit of how PayJoy might be a compliment to these things, and [00:06:00] it's not a replacement per se.

So when we're talking about Bitcoin transactions, we know that those transactions don't come from accounts. Your Bitcoin wallet isn't so much an account as it is a collection of keys that can spend entries in the Bitcoin ledger because anytime someone pays you, they're making a transaction and you get an output that your key can unlock.

So because it's not accounts, you have some pseudonymity, but there are naive ways that most wallets will spend Bitcoin. And when you do that, It's very easy to track all of your behavior, your balances, and you can put a target on your back, which is, you know, dangerous if someone wants your Bitcoin. So coin join, fix that problem partially by having multiple people [00:07:00] come together and make equal outputs at the same time so that someone looking through the history comes to a coin join and they're not sure which one of those equal outputs belongs to you or the other participants in the coin join.

There's a whole boatload of caveats about how you spend after that can deteriorate that privacy. But in general, if you look at the one coin join, the idea is that all of the participants have forward-looking privacy, meaning no one, once someone gets. Tracking back to that coin, join everything in front of that is unrelated to everything behind that.

**Marty:** Yeah. And you mentioned this in the, uh, the blog post announcing, uh, p dk, which is essentially what coin joins are trying to do. It's funny cuz you're using an output as an input in a new transaction. And so by [00:08:00] doing a coin join, you're essentially corrupting the common input ownership heuristic used by chain surveillance companies that try to track you across the ledger.

So if you buy from an exchange and send to a wallet, the wallet basically uses that common input ownership heuristic to assume that you control that wallet and then as you spend forward in time, um, from there it sort of use that to track you. And so Coin join sort of corrupts this

**Dan:** heuristic, correct.

Coin join corrupts the heuristic. And also because you have equal amounts, which is how join Market Wasabi and Samurai operate, you have not only ambiguity in where the inputs came from, like did they all come from the same user or not, but you also have that multiplied by a number of people that participate.

So your coin becomes one of all of the [00:09:00] participants in the coin join. And then assuming the implementation is done cleanly, the rest of all of the other people who do coin joins that spend from your source coin joint as it relates to join market. Join market's, cool. Because there's not fixed output amounts, so you can make a payment of your specified amount, and then all of the peers that join you in that coin joint, you pay them to participate.

And they make your decided equal amount, which is great because it's kind of like a pay join and that you spend, and you don't have to worry. After that, you have this change amount because you're making equal amounts with a coin join, you end up with this change amount that's still linked to the original, uh, history and wasabi and samura, at least wasabi version one.

Just do something slightly different where everyone has the same output amount, so everyone gets change. [00:10:00] And wabisabi with wasabi v2 try to fix this by allowing any, not necessarily any amounts, but they break down the amounts in the outputs based on the input to try to reduce the amount of change. But all of these still suffer from some, both the knowledge that you know, that it's a, what's called a mix, meaning all of the inputs and all of the outputs are the.

Paying themselves. And with Join Market, you, you have all of those other takers who took the maker's payment amount. They don't have an incentive to keep that new amount. Their incentive is to join more, join market transactions, so you can kind of figure out who made the payment and who was a maker down the line.

There's not a ton of protection from that. Uh, and in [00:11:00] Wasabi two, even though you create all of these different amounts, the person who put the greatest input still has the greatest output set. So you can do, it's computationally expensive, but if someone were targeted, you could still target one of those participants and figure out a likely set of their outputs.

And over time, break that down. Uh, samurai tries to stay to the. Zero link. Everyone has the same amount. Everyone has these same spending conditions, which you still get that change, that their solution is just never spend the change. Or some people use swap services and stuff, which is expensive. Um, but really I think the key point is that they all still have this change problem, whether they say it or not.

And you don't have assurances for arbitrary amounts that your inputs and outputs are unlinked. And the, if you do multiple inputs, the [00:12:00] inputs are unlinked. And if you have multiple outputs, the outputs are unlinked. And this is where pay join really, uh, makes a difference. Pay join because you have a payment amount, you have the potential to break all of those links.

You're not depending on the equal output amount. You have a potential to get rid of all of it because you're making no change.

**Marty:** Yeah, that seems pretty massive because the change that you mentioned is commonly referred to as toxic change, uh, in the community because it's toxic to an extent because it can dox you if you co-mingle that that change with an output produced, um, while you're coin joining or at the end of the coin join.

And yeah, so like let's dive into how pay join really reduces or reduces that threat of the docs change cuz it's a [00:13:00] really cool way. And if you're successful, when you are successful, is that confident here in, uh, getting p dk to the point where it can easily be implemented in many wallets. Like the ideal scenario is that you're just spending and doing pay joins as you spend.

Um, and just unknowingly. Increasing your, your anonymity set as you spend your, your Bitcoin, maybe not your anonymity set, but your, you know, better privacy insurances as

**Dan:** you're spending. Anonymity set is a really interesting thing to talk about though, because that really comes from M nets, mix nets that comes from network privacy.

Back in the early aughts, this paper came out where they're working on anonymizing network traffic and network traffic doesn't have an amount associated with it. Really. Network traffic is just packets, and so if you have an anonymity set of people browsing the internet, for example, that's the number of people that the traffic could [00:14:00] potentially originate from.

With Bitcoin, at least with Chian coin joins, with zero link coin joins, this idea has been borrowed and adopted. So if everyone has the same amounts, we can think of this anonymity set in isolation in one transaction, but it kind of breaks down when you look. Across a bunch of transactions with this toxic change problem.

Um, so I don't think it's a harm, like I'm, I'm glad we are using some sort of measure, but I think it's imperfect. And what I think of page one, I don't really think of anonymity sets so much as ambiguous interpretations. Like how many interpretations could you imagine a certain output has? How many people it is, how many people this could have come from, but also like how many different trees of transactions, how many different histories is this associated with?

And because you don't have those equal amounts because you're [00:15:00] paying someone, so pay join is just two people, at least in its current iteration. Pay join BIP 78 pay Join is when a sender and receiver combined two transactions into one. You have the sender wants the privacy, and the receiver is open to consolidating.

Some of those outputs together. So they spend fewer outputs when they spend later on because they contribute input to the transaction with the sender. They consolidate what would otherwise be two outputs, their previous output and their incoming payment from the sender into one. And the payment amount, because it's combined, is hidden.

It's not explicit on the chain. The payment amount is the old input from the receiver plus the payment and some change. And you can set up the input amounts and the output amounts so that someone looking both doesn't know it's a [00:16:00] pay join. And they also don't know which one of the outputs includes a payment.

**Marty:** Yeah. And that's, again, pretty massive. If it can become, Widely used and popular and I guess is a good, good point to jump into. Why hasn't received as much as that adoption as many people like myself would like. And I imagine that revolves around the need for the receiver who needs to put up an input with the sender to have essentially a hot wallet and an always on server that enables them to essentially create inputs in these page joint transactions.

**Dan:** Yeah, having a hot wallet to always contribute an input and being available all the time to add that are definitely the biggest barriers. But I think we can get around both of them. [00:17:00] So because you're contributing an input right now, you need to have a server that's online that receives a proposal and can respond with.

An updated pay join, and this is kind of similar to Coin Joinin where everyone that's participating is online and talking to a central server to figure out everybody else's intent Pay join PIP 78, pay join Pay Joinin V1 solves this by having the receiver run a HTTP server that you can make those requests to and it can respond to.

Which is great because HTTP is everywhere. Um, we're using HTTP to have this call right now, it's ubiquitous versus something like Coin Join, which is a custom protocol that you need a whole network stack in your app to use. So you need kind of custom apps. It's a huge dependency to add. The only application besides a pay joint, sorry, a coin joint specific app that has this, as far as I know is, well there are two.

It was be btc, pay Server and [00:18:00] Sparrow. There are like different versions of some of the coin joint apps, but most of them are just. Coin join apps. Uh, because there's an incentive though for the coordinator. I think Coin Join has been able to skirt around this. They've been able to get teams together.

They've been able to raise money and they've been able to deploy these standalone wallet apps because there's that incentive where, okay, if we deploy this, we're going to make a lot of money and everyone wants to make money versus pay join. There is a small incentive for the sender who gets privacy and the receiver who can do that consolidation, basically batching to save fees later on.

But that hasn't really been enough to convince companies to take it on because it's difficult. There hasn't been any library software to do it, and I don't think anyone had explored before recently what that. Idea of [00:19:00] batching that can be done with coin Join really does is this whole idea of transaction cut through that Greg Maxwell came up way back in the day where if multiple people propose transaction intense before they post them and they can interact, then you can get rid of some steps in a transaction process, which gives you privacy, like how pay Join gets rid of the payment amount.

And you can also do a lot of cool stuff by reducing the number of steps it takes to open a Lightning channel, or even for services to batch transactions that are incoming. Instead of just batching a bunch of withdrawals, you can, you can do those and also pay them with incoming SATs without ever having to take those SATs into custody first.

There's like a whole unexplored territory. So yeah, basically we removed that hot wallet and server. Requirement, uh, through something. [00:20:00] There are a bunch of tricks we can use to do that, but also I think telling the story of how this fee savings can be so great could also really help pay joint adoption.

So these are all on the roadmap and mention in the blog post, but it has been a real difficulty.

**Marty:** Yeah, I mean, and the fee saving thing was, you mentioned that you listen to Matt and I discussing some rabbit hole recap, and you mentioned it in the, the blog post too. Is it really only fee saving for transactions like lightning channel openings or does it go beyond that for other use cases as

**Dan:** well?

It goes beyond that for other use cases. So the lightning one is probably the easiest one to. Understand because it gets rid of a whole transaction and a typical flow. To understand this, we gotta go into the lightning funding transaction flow. So let's say you, [00:21:00] your buddy sends you some SATs and like in a U T X O on chain and you wanna open a lightning channel, you would have to send those setss to your lightning node, wait for a confirmation, and then propose a channel open and open another transaction with a lightning peer.

With pay join, because it's interactive, you can propose to send some SATs to your lightning note like your buddy who pays you, and the response can include an output that is the channel open with your peer. It can just replace that original intent. So the only transaction that actually gets posted is funded by your buddy and opens a lightning channel.

This is like without even channel batching. This is just, you skip a whole transaction, but you can also batch transactions with that. That's the idea of transaction cut through. Does that make sense? Yeah.

**Marty:** Yeah. And so you, yeah, you're essentially cutting one step out of the Lightning channel setup process.

Cuz [00:22:00] if you're opening a channel in the first place, you're gonna wanna two-way liquidity. Um, so you would've made that transaction anyway, correct?

**Dan:** Yeah, you would always need to open. You always need a new output. That's the channel output. So either you take a single sig into your own custody first, or you get rid of that step by communicating with networking instead of communicating using the chain.

And this is, the idea isn't limited to lightning channels. That could be used for an exchange too. Like an exchange could queue up a withdrawal, say one of the users wants to. Take SATs outta the exchange. They could say that and the exchange could say, okay, that's gonna happen sometime during the day, and if they get a pay join coming in, rather than putting those SATs coming into the exchange, into the exchange's address, they can just put those SATs directly into their withdrawal and take the excess as change into the exchange.

[00:23:00] Because the pay join proposal includes like a fallback transaction that doesn't have to necessarily get posted. That can always be updated with any number of payments to a withdrawal. Or say they wanted to pay a bill or they wanted to open the lightning channels. You can really extend this cut through idea quite a bit, where both sides can now be batched instead of just an exchange batching a bunch of withdrawals, they can also batch their inputs, funding withdrawals, which is a huge problem when the block space market becomes expensive.

A lot of the Times Exchange is just run out of UT XOs that aren't confirmed to pay withdrawals. Yeah,

**Marty:** I mean that's massive. I mean you can make the argument like this is a great way to reduce the amount of times it's necessary to touch the change for service providers, whether you're running a liking company or an exchange.

Is this why after I posted that, [00:24:00] that newsletter earlier this week, Francis was getting all excited, I think, is that how he's thinking about implementing it in Bull Bitcoin if you Yeah,

**Dan:** he really gets it. He's gotten it for a long time, but like there hasn't been a page join receiver that works with their cipher node backend.

You know, there's been the BTC pay server one, but there hasn't been any way to connect it to Bitcoin D or connect it to l and d and that's what p DK allows. That's really, that's what improves it more than anything else. Um, we do need, and it's great for an exchange because they're running a server all the time anyway.

There's definitely a way to get rid of that server requirement. Um, or have someone else run a server for you. Kind of like how offline lighting payments would work. It's very similar, but for someone like an exchange, as long as you have the software that works, it is a great cost saving thing. And the other thing Matt mentioned was the difficulty of sending a transaction.

The cool thing is page join [00:25:00] version one was engineered around the Bitcoin U R I standard. So the experience from a user's end is the same. If the wallet supports it, it'll just send Page join. When you scan the QR or tap NFC or paste the U R I, there's no, you don't have to put your funds into Lightning and then use Lightning.

You don't have to put your funds into a Coin join app and then use Coin Joinin. It just works.

**Marty:** So the receiver essentially would initiate it and the end user may not even know that they're engaged in a pay join unless they are hyper cognizant of fees.

**Dan:** Is that correct? Right. The receiver adds a little more data to the address they would post anyway.

Mm-hmm. And the sender's wallet, if it supports it, can use that and otherwise can just ignore it. Yeah.

**Marty:** This is fascinating. And so turning back to like the server, which is really interesting, you [00:26:00] posted on the bitcoin dev mailing list earlier this year, I believe in January. If my memory, uh, serves me correctly here about solving this always on server for the receiver problem with something called a turn server.

A turn relay. What is, what is

**Dan:** this concept Y Yeah, I think linking to that, uh, mailing list might cause some confusion because that was like the first version of it where I put it on the mailing list to see if there was interest in the idea or not. Uh, turn is a very old way of getting too. Servers that, or two computers that don't host servers to talk to each other using a relay.

So this is assuming they're on all the time, and the receiver negotiates with a relay, and then that relay, if the sender connects to that relay, the relay can talk to the receiver. I don't necessarily think this is the best way to go forward, because there's a, there's actually huge [00:27:00] improvements in relays over the past couple of years.

Like Noster, obviously we relay media and that gets around censorship in a huge way. But also there's these two things, there's two pieces of technology. One is called oblivious http, which is kind of like to, without all of the complexity of a consensus network, and this idea called mask, which is like Turn Reimagined, it's a much easier way to do, uh, relay server.

And both of these are supported by iOS 17 and they're advertised as privacy features. So, Turn is like complex and a lot already supports it. But the cool thing about having O H C P and Mask in operating systems is that we can again, use this pay join dev kit in all of these different wallets and expect that these relay softwares will already exist.

So I'm not like sold on turn, turn is one way of having a relay. Um, [00:28:00] and that at that time was a good way to understand, for me to put into words that we're going to use a relay to get rid of the requirement to run a server, but it could be any number of different relay tech. It could be monster. You could do that.

**Marty:** That would be fascinating if you could and just like dive deeper into this, like what is leveraging this type of relay infrastructure, regardless of which one it ends up being. Like why does it make it easier?

**Dan:** Well, I think we should settle on one relay in a standard because then all the software can talk to all the other software.

When we're saying, just put a relay here, no one adopts it. No one has interoperability, and that's kind of what we have now in the original bib 78. There's actually this idea of an unsecured pay join server where [00:29:00] you make a pay join using a relay, but that relay is trusted not to share your information.

They can still see stuff and stowaway in Samurai and Sparrow also does relayed pay joins, but through tour and without a public specification that was built using a collaborative process. So it's just really hard for wallets to use those technologies because they'd need to build something proprietary that like an exchange could do it between their exchange app.

And their exchange. But beyond that, it would be really difficult to actually roll out. And the idea of serverless pay join would be to come up with a standard that all different wallets get implemented. I think mask is probably the best way to do that. Right now you create an application that the specific pay join relay understands and it's well specified.

So any client that wants to do serverless pay, join, relayed, pay, join, whatever it ends up being called, can agree [00:30:00] that, okay, we can talk over this relay and we're going to have encryption so that the server can't snoop and some other, you can set some other requirements so the server can't even do timing attacks, and that's why it's serverless, because you're not running a server and you're not really, depending on, you're not trusting someone with your privacy.

Yeah.

**Marty:** No, I completely agree. No, it's standardization. Mm. Coming to agreement on what the relay is gonna be that everybody uses is very important. Um, and it's been ENC encouraging to see like standards around like ps, BTS get adopted by the by wallets too. Um, which is ac I was actually talking to Rob Hamilton earlier today and told him I was gonna be discussing this with you and whether or not he had a question for you and, and it was, it revolved around like PSP ts like how, if at all [00:31:00] are PSP ts evolved in pay join or this infrastructure?

**Dan:** Before I answer that question, I want to, uh, say one thing about relays, which is, oh, people don't think about relays, but we use 'em all the time. And one of the standards that uses one that has just blown up is L N U R L L N U R L uses a relay so you can get a invoice whenever you want. And that one has just been huge for lightning adoption and making it possible for people to.

Receive without running a server all the time or being able to dedicate to, uh, a different service. Uh, but as far as Rob's question, PSB ts the page one protocol is really only two things. It's, it's an interactive PSB T coordinator, and then it's the specific network packets that you send back and forth to agree on fees and prevent attacks.

It's those two things. PS b t, coordination [00:32:00] and networking.

The protocol is so simple, I can really just explain it in three steps. So the first step is you generate that address that includes your server url, this is your Bitcoin, u r i with a pay join parameter, that's the receiver, does that. Then the sender constructs a transaction as they would normally, I call this the fallback transaction that pays to the address.

The receiver sends, but instead of just broadcasting that transaction, they create a complete valid P S B T and they send that to the receiver's server that they specified. So the receiver gets that, and if nothing else happens, they can always broadcast that transaction. They always have valid money that's gonna pay for whatever service.

So because they have that, they're comfortable contributing inputs that the sender might not know about [00:33:00] and returning a new P S B T with their inputs and the outputs change to include the value of that input. So it's really two PSBs that get sent back and forth and that's it.

**Marty:** Yeah. And as you're describing that process, like how long does that

**Dan:** take?

Uh, it's basically instant. It's just like sending a message. Sorry, I gotta plug in my computer here. I don't know what happened. I unplugged it, stepped on it. It's, uh, yeah, that's, that's the whole thing is it, it just takes as long as it sends as it takes to send a text message. As long as the server is online, you send a message and they respond.

And so, like

**Marty:** with that instantaneous interaction there, it's like me driving my mind towards like, how does the coin selection work on the receiver side in terms of utx as they're providing, as the inputs. Is it random? Um, obviously it's a [00:34:00] hot wallet to put particular utx. Is there, um, is there like a coin selection algorithm in the background?

**Dan:** This is another thing B DK does that other implementations haven't until I think earlier this year, B dk, was it wa, sorry, not b dk, uh, BTC pay server. Was, it wasn't picking random, but it was creating transactions that could sometimes be identified as possibly pay joins. So there's this thing called unnecessary input heuristic, where you are putting more inputs into the transaction than are strictly necessary to create one of the outputs that's assumed to be the payment output.

And there are different varieties of this. Some of them are identifiable as pay joins, some of them not. So the receiver does have to be cognizant of which inputs they choose. And the p DK has a function that you can just call on your inputs that will construct a pay join, preserving your privacy. That's not obviously [00:35:00] a pay join transaction.

And BTC pay server does that now. But if you use p dk, you just call that function and it'll be like, okay, either use a fallback transaction or construct a good, safe, secure, privacy preserving pay join.

**Marty:** So how does it determine if it's

**Dan:** good, safe, and secure? So like if, uh, if one of the inputs is obvious, I don't have the, there's a paper that came out last year that went into really great detail to identify these that we didn't know were identifiable.

But I think it's like, if there's a really big input and one of the outputs is also very large and the payment payment amount is trivial in comparison to that, then you can find out what the payment amount is. But if the inputs are relatively similarly sized and the payment is significant compared to those inputs, then it's more difficult to parse which one is the input and which one is the change.

Or it's like less likely [00:36:00] that if you make an assumption that the big input is also the big outputs, uh, owned by the, the big output then. Uh, because you don't, because it makes it more difficult to make that assumption, or it's less likely that that assumption is true. You have, you're said to preserve privacy.

**Marty:** So it's basically calculating something in the background that compares the payment that they're looking to receive with the U T X O that the sender's putting up and then going using coin selection to find the UT X O that sort of make it, uh, very hard to figure out what, what the actual payment was.

**Dan:** Yeah. You choose an input so that one of these varieties of unnecessary input heuristic is not created. That's, that's the idea. It's like [00:37:00] a subset, some problem. So you have to make sure, cuz you can use more than one input. So it's like the subset of sender inputs and the subset of. Receiver inputs won't be associated together and neither will the payment amount.

Mm-hmm.

**Marty:** Yeah, it's fascinating. So what is the current state of p dk, like, obviously you just released the announcement blog post earlier this week. Obviously you've been thinking about this for a while, and it's probably important to touch on, uh, something you mentioned in the blog post, which is you decide to make P D K, um, separate from something like B D K, uh, for, for, uh, specific reasons you, you did this intentionally.

Like why have a separate development kit and not just try to get this implemented into something like bdk

**Dan:** because it's a different thing really. Uh, so p dk right now is, [00:38:00] I'd say the sender is a late beta. So the sender works. It's been in the wild, it's been in Bitmask, it's been in Pay Join cli. It's been in no looking for quite some time, which are just different pay join apps and the receiver is early beta, so the receiver works, but we need more extensive user testing before we can call it a version one and have the interface be concrete.

Say that we're pretty sure that the interface won't change. So that's the status of p dk. I really started working on Pay, join and Rust to have this idea of a kit where you could put it anywhere. So instead of having an application code where you need to use Wasabi, you need to use samura, you need to use Join Market.

This can be drop in anywhere. And I was really lucky to get to work with Evan Lynn, who works on BDK very closely. Uh, when this was early before he was [00:39:00] granted by Spiral. So we were working on these Lightning Pay joins and. Because he knew that so well. He understood that BDK is really a wallet software.

Underneath that is this thing called Rust Bitcoin, which has been around for years and years. Ldk uses it and BDK use it. But that's all the primitives, that's all the consensus details. The consensus data structures where BDK manages the keys, BDK makes sure you're not spending the same U T X O multiple times without knowing it.

It's doing some, uh, coin selection for you and it manages output descriptors where Page join is doing that PSB t negotiation and the networking. So right now, will Owens is. An awesome summer of Bitcoin intern. He's being helped out by Steve Myers of BDK to get those together. One of the first integrations was with that [00:40:00] bitmask app, that's beta.bitmask.com.

It's a great web wallet that uses Lightning and Bitcoin using ldk and bdk, and they even have Rrgb assets. So we proved that it could be used with B D K, and we're working on finding what's common between different B D K pay JOININ combinations. So eventually, I think it will be possible to have pay joinin as a feature flag in B D K, but there's always a little bit more work that needs to be done to include pay join in your app.

And for that reason, I think it makes sense to have the focus on the pure protocol PS B T protocol. So that's really well tested, that's well understood. A lot of eyes are on that for that reason, and we can support that within BDK from that perspective.

**Marty:** Yeah, that makes a lot of sense. And so senders been in beta for a while, receivers just [00:41:00] getting beta.

Like what else, like what would make you happy in terms of the state of p dk, like in a, in a form where you'd be proud to send it out into the wild and have the people begin implementing it into their, their software?

**Dan:** Hmm. I'm proud of bdk. I'm proud of p DK already. I think they're, the rest implementation is suitable for people to implement and integrate and people are doing it.

We're certainly growing and after putting the blog post out and getting support from the community, there have been a lot of interest. The reason my voice goes up at the end, and I'm a little, I know it's not complete, is because it's still rust. It's not. Possible to use in any language you want or it's not easy.

So we've gotta work on bindings. And the cool thing about borrowing the architecture from L D K and BDK is that there's a bunch of people already working on a [00:42:00] sort of canon way to think about bindings. So no matter what project you're using, you can be familiar with it. And we have some basic Mathias from Trident has been working on Python bindings.

Thunder Biscuit has been a great help to unify the bindings between multiple languages. And Michael from Bolts has been really helpful in helping set up a roadmap for what Wasm bindings could look like for TypeScript and JavaScript. And once we have solid bindings, it's really possible to rely on all of the review and security that's gone into the sender and receiver.

So knowing your implementation is. Spect, right? Knowing that Coin selection algorithm is going to avoid the heuristics that are gonna identify you as pay joins and still be able to directly plug in using whatever language you want.

**Marty:** Yeah, that seems pretty massive. And Thunder Bisk just got a grant [00:43:00] from Spiral or the H R F correct to to focus

on

**Dan:** this full time.

Congrats to Thunder Biscuit. Yeah, he's been amazing. There's this thing called Uni Ffi from Mozilla that they use to write their, uh, password manager extensions once and then they use it in their browsers and they use it on all the different devices. So we're using, or Thunder Biscuit has led us to use the same Uni F F I Bindings technique, so we can write this library security focused library once and plug in in everywhere.

We don't have to worry about the security. It's a really good model. It's a secure model for this kind of software where people's SATs are at stake.

**Marty:** Yeah, that's fascinating. It's, um, always blows my mind talking to individuals like you working on this. Cuz sometimes personally I get, uh, I came into Bitcoin from like an economics perspective and got heavy in the mining like [00:44:00] five years ago.

So in recent years I've been really focused on mining and just like the, the economic side of bitcoins and its effects is a monetary good on the world. And it's always fun coming back and having conversations with people like you actually working on the protocol that make all this work and make it work better for individuals at the end of the day.

Um, so thank you for doing that, but I'd also like to like, if you're successful, p dk is successful when it is successful. Again, confidently speaking here, when it's successful and it starts getting wider adoption in wallets. I know I've had this conversation in the past and it's really hard to tell, but like, Or foresee, but like, like how much adoption of pay joint throughout the Bitcoin economy needs to happen before these common input ownership heuristic are completely boring.

**Dan:** I just got asked this question this morning. It's really great to talk with you as [00:45:00] well. I've been listening to the pod for a long time. You've had the pulse just on Bitcoin for so long, so to get the questions that are at the most, they're at the core of the community. Every facet of it. Everyone's asking these questions, so to get them all at once lined up is a great help for the Bitcoin community at large to understand what the heck is going on.

When I think about getting it out there to the world at large, I think about. Pay joins role in all of these technologies. So you have lightning, which is really for payments. And I heard Matt yesterday say, oh, you know, if a store supports lightning and pay join, of course I'm just gonna pay with Lightning.

Obviously it's cheaper, it's faster, it's well established. And you know, pay Join has a way to go until it's effectively breaking the [00:46:00] surveillance. And to that, I say to think about pay join as more of a settlement layer. Technology. So Lightning always settles on the main chain. You always have to make a new output to have a new channel.

And that's where Pay Join really comes in. It's about batching those settlements. Even when a receiver receives a very basic transaction, when they take that pay join, they're the best time to do that is when a fee market is low because they're doing a consolidation so that in a. High fee market environment, high block space, market environment, they can make a transaction with just one input instead of two.

So anytime you're doing settlement, having the option to pay join and having that option ideally decided for you by software that's making predictions, will save money over the long, the long haul. As far as when it becomes effective, it's [00:47:00] effective immediately. So if you're making pay joins and some of your transactions are pay joins and someone watching you knows that, then they're, for all of those pay join transactions, the common input ownership heuristic is wrong.

Their assumption that they'd make is just wrong as it becomes a more widespread than any transaction they look at could have that likelihood to be a paid join. So the example I gave in Discord this morning was, Say you don't know any information about the people you're tracking, but 5% of transactions, you know, break common input ownership heuristic.

Let's say 5% of transactions are page joins. There are other transactions or are other reasons people make transactions that don't conform with that heuristic, but let's say they're page joins. Then if someone were following your transaction in isolation, they would say, okay, I have a 95% probability that these inputs are both [00:48:00] owned by the same person.

And then if you they have another transaction, they have that probability again. So as these are chained together, they're all independent variables, then the probability that they can follow some input to its output goes down substantially. If you have one 95% chance probability times another 95% chance probability, it just gets weaker and weaker as time goes on.

So you don't need that much to make tracking Bitcoin over a few. Different transactions. Very unreliable.

**Marty:** Yeah, that makes sense. And then,

so at that point, like a chain surveillance company wouldn't have confidence in your, in whether or not you control certain utx loads. It may still have confidence about other individuals not engaged in page doing transactions outside of that. Um, but [00:49:00] there's a point, sorry, there's a plane flying above me now since the back, the back porch Studios always fun.

But is there a point where actually, uh, I don't know, I th my assumption is there, it's isolated to people in page income, uh, transactions, the, the privacy that's provided them, like there, is there any critical tipping point where. I'd say like 50% of the network is engaged in PayWay transactions. The other 50% is afforded benefits of, of that individual privacy.

People transacting in the PayWay fashion. They're getting.

**Dan:** This is a question I really wish I had a better grasp of because I'm not, because of course the surveillance companies act like they know exactly what's going on. That's how they make money. They tell people, oh, we know we can track with certainty, but I don't know [00:50:00] where the hammer falls on that, where I think we'll find out in this bit fog case.

Mm-hmm. How reliable the public thinks this tech is. Um, but if someone analyzing your history, knows you use pay, join, or knows that the wallet you use supports it, then they can really no longer. Make that assumption. So you get that safety just by having your wallet support it, even if you're not using it, especially because it's something that is automatic in terms of user experience is not something you have to opt into really and manually do as a sender.

Um, and even as a receiver with serverless pay join, so if wallet's supported, you have that assumption across the board. And then even if someone can't identify what wallet you're using, say you're just using something that supports newer [00:51:00] technology like SegWit or Taproot, then there probably is a chance at least after an exchange pays you that you could be using a wallet, supporting pay join and these common input heuristics used to track you aren't really valuable anymore.

Yeah,

**Marty:** yeah. The bit fog case will be very interesting. What worries me about that case is it's becoming pretty clear that it seems like chain analysis is, or chain analysis is, um, basically over advertising their abilities. But you can see a case where, God dammit it, a helicopter coming right over me.

Gimme a second

**Dan:** technology.

I'm really curious what a jury will be convinced of is in that case, cuz that's what it's gonna come down to. It's [00:52:00] how the experts that, uh, you know, everyone looks to, to trust, can convince a jury. And I think it's, it's pretty obvious that their techniques aren't perfect. That there is a chance that every step along the way, even without pay join, has a high degree of ambiguity and could not be what they're saying it is, but yeah, we'll see.

**Marty:** Yeah. Well and that's like the, the conversation I've been having in my mind too, cuz you could see a case where the jury has no idea what's going on. They can convict, um, this gentleman, I forget his name, uh, off the top of my head, uh, with bad information. And then chain analysis can continue to use those bad techniques to just essentially buddy up with the government to throw people they don't like in jail.

But at the end of the day, on chain, like really, [00:53:00] like the truth is they don't actually know what's going on. So like what, like is that I a silver lining to all this? Like, it's weird that chain analysis and the government can go bully people and throw them in jail with bad information. But we also have the knowledge that the truths of the situation is that they actually can't identify people on the chain.

I don't know if I'm articulating this correctly, but do you see

**Dan:** if there's, yeah, I mean, they definitely can throw people in jail. Unfortunately, before a, a trial, this dude is in jail. He is suffering. Um, but I have a lot of, I do have faith in the American justice system. I like to think I do that. Even if it got through one jury trial at some point.

There's appeals, there's an appeals process, and it's quite plain that what they're doing is heuristic analysis. They're making assumptions to come [00:54:00] to a conclusion about what someone did or did not do. And this is, I mean, all law enforcement is heuristic analysis. Um, if you're dressed up wearing a nice shirt, if you have a.

Particular ethnic background, it tends to be the outcome of a traffic stop might be different for you than someone else, but that doesn't necessarily mean the justice system when all is said and done will not be able to see through that.

**Marty:** Yeah. Yeah. It's fascinating times. So you seem pretty passionate about this right?

To privacy, particularly as you transact with Bitcoin. Like do you, do you see a potential timeline where Bitcoin, maybe it doesn't fail, but isn't as successful as it potentially could be due to a lack of privacy assurances for individuals? [00:55:00]

**Dan:** Hmm.

The biggest issue with privacy in financial markets to me is that the people that. Sensor that make the decisions of what's okay and what's not are not necessarily elected. They're not chosen through a, a democratic process, and they can be there for a long time. So I think markets would be better off for people to not have the opportunity to bully others and make rules without being elected and being held accountable.

And Bitcoin gives me a lot of hope that that's possible. Uh, the fact that this privacy issue was left in the white paper is exciting to me because it's like, oh, nice. Satoshi gave us some directions into what's possible, what we can fix. Um, it gives me something to do. So there's one line. [00:56:00] There's like, I think there's one section about privacy in the white paper, which talks about common input ownership.

Puristics Satoshi says, if. Some inputs are spent in the transaction, then they can be assumed to come from the same person, even though addresses are pseudonymous. And you can get a degree of anonymity by receiving payments using a fresh address. So because we can pay join, because we can make these transactions using interaction that break that heuristic, we can bust that last open privacy problem.

Like it's a long path ahead and it will be iterative. But I, I have faith that it's possible to solve the problem. And it's not so much that I think Bitcoin would fail because it didn't have privacy. To me it's more, it's more a matter of when it happens. That is the thing that's holding adoption back.

Like the reason people like gold and hold it in their treasuries is because they have the rock. They [00:57:00] know they have the rock. They know no one saw them receive the rock. And I think that's possible for Bitcoin and that's going to. Make it grow when it's possible and people are going to have a lot more eyes and trust in the system when they know it has that.

Because when people say, oh, Bitcoin's private, I think a lot of people know that's bs. It's just not, they. They get intuitively that that can't be the case. But if you can prove that's the case, people will be a lot more comfortable to use Bitcoin and allow it to be far more widespread.

**Marty:** Agreed. And that's why I'm really excited about what you're working on too, because there's a lot of, uh, a lot of people believe that Bitcoin can never attain a sufficient level of privacy assurances without upgrading with something like confidential transactions or zero knowledge proofs.

Like we'd need a massive protocol upgrade to get the privacy of people [00:58:00] really one at the end of the day, and I'd never. I believe that to be true due to the fact that coin join exists and things like Pay Join exists. It's just more effort that's needed on the software side of things to make user experience such that people are, are using Bitcoin and Coin Selection and PSTs with multiple inputs

**Dan:** smartly.

I see no reason you couldn't coordinate a pay join that involved more than two individuals. I think you could get the coin join process where multiple people contribute and where you have ambiguity from payments in the user experience of scan a QR or share your address u r I with someone to signal your intent.

And then the result of that is a transaction that like pay join doesn't stick out as something weird. It just looks like a batch. Okay. And gives you the unlink ability from your inputs and outputs to the point where [00:59:00] even the person you're paying doesn't know what input you paid with, which they do in page one.

Version one.

**Marty:** Yeah. So let's flesh that out. How would, uh, like a multi receiver paid join work

**Dan:** when the time comes? I'll release some in depth, uh, uh, documents, but I think it could work with a coordinator. Um, you could use an ECA system. This is what like Max Hillebrand has suggested on the mailing list in the past where the, you register inputs and outputs and then somehow you split those outputs based on the other inputs and outputs.

So there's a degree of ambiguity, how that all comes together, how it fits within the QR code paradigm. Is yet to be seen, but I'm convinced it's possible.

Why are you convinced it's

**Marty:** possible? [01:00:00]

**Dan:** Uh, because all the basic tech is, is there. You really just need a way to coordinate. It's a matter of sending enough messages and blinding the sensitive information, and we know how to do both of those things.

**Marty:** Yeah. So just more time needed to be spent on this problem to actually figure out how to

**Dan:** Yeah.

So you need, you need engineering resources, you need smart minds. You need people who collaborate and you need people who don't give up on the problem. Yeah.

**Marty:** That actually brings up a good point in something. I've had discussions with some core devs, uh, over the last year, particularly like this concept of like burnout, particularly within Bitcoin core.

Like what are, what are your thoughts on the state of just developing on Bitcoin right now? It seems like there is. A fair amount of burnout and, um, the core specifically, it's really hard to get stuff done that tends to frustrate people. [01:01:00]

**Dan:** Yeah, I think it's actually getting better than it has been. I think we had a low point between the bear market and legal trolls, but we're seeing more and more support in the world of grants.

I think we're seeing people be quite generous and realize the need. And I think because it's financial technology, the companies that rely on it are more likely to support it than they are plain data or networking open source stuff. Because their money is secured on it. They can say, okay, we can spend this percent of our budget to make sure it's secure by supporting the people whose responsibility it is to secure this thing.

So I, I have, I'm optimistic in that regard. Um, I, I know a lot of people, it's really difficult. It's been difficult for me to get grants. Of course. I'm not like, Super well funded. I've sacrificed a lot to be able to work on Bitcoin and persevere through tough times with it. Um, it's [01:02:00] not easy to maintain this kind of development when you're not, you know, venture backed or working for a corp who's paying your salary.

And those opportunities are super appealing, especially before the past year to go and work at Facebook the year you graduate and make a know 300 grand a year, just a couple years later is super appealing. I understand why people wanna do it. Um, and the other thing about the grant programs that I would like to call for change for is I think a lot of these programs focus very solely on a non-profit, which is different from.

Sinco land. I think a lot of the grants are conditional on, okay, you're not gonna make any money doing this and that, that makes projects that are unsustainable. Like even if you had a nonprofit, it, it, it's not necessarily a bad thing that, [01:03:00] that makes money to sustain the project. And in fact, I think that's what you want to encourage.

I think you want to encourage business models that work that way those business models can support more and more development and you get a spiral upward. So I hope that, I hope that can change. I hope some of these grant giving organizations can say, okay, just because there's a business model associated with this does not mean it's ineligible for funding.

And in fact, that could solve the problem that we keep running into where our grant money runs out at the end of a year because this person's work is expensive and is not sustainable without a continuous donations.

**Marty:** What are some examples of like projects that could make money that. Aren't getting grasp.

**Dan:** One really good one that succeeded is the Bolts Exchange atomic swaps. That's been like a huge success story, I think is that got, I guess, that got some support from Human Rights Foundation. It's not like huge support of course, but they're able to turn that [01:04:00] into a real business that at least can sustain.

I don't want to get into, you know, stuff that, uh, hasn't been funded because I don't want to complain. I think by and large, all of the granting has been overwhelmingly positive, directed in precisely the right direction. Um, I don't think much of it has been misguided at all. I think the people giving these grants are making very good decisions and supporting the developers that need it.

I just ask that they, you know, consider to open their mind a little more. Yeah,

**Marty:** I think that's a fair ask. Makes a lot of sense. And, uh, technology is important. Like you mentioned it, like it's been hard for you, but what keeps you, what keeps you going? Like,

**Dan:** it's meaningful, man. It's, I, I feel like I'm, uh, doing something that can help people and I see a light at the end of the tunnel.

We're working on [01:05:00] Bitcoin. Like I'm not working on some esoteric, uh, zoology that really requires this constant funding to sustain. We're working on Bitcoin. This is, we're working on financial markets. Having expanding knowledge into this realm is worth something tangible. So I, I can keep going. It's not a problem.

**Marty:** Yeah. Well, again, thank you for doing the work that you're doing. I think it's extremely important. Um, and

**Dan:** I'm very happy. It's fun. Everyone in this has very high, uh, well, not literally everyone, but I think on average the people that I work with in Bitcoin have a high degree of. Humility and collaboration, uh, being able to work on network protocols in particular where so much of the culture is inspired by this idea of rough consensus and decentralization is really easy to fall in love with.

It's just fun. Yeah. Yeah. [01:06:00]

**Marty:** And you mentioned you see the light at the end of the tunnel, so I'm not gonna put words in your mouth, but, um, stoked. A question in my mind is like, do you, obviously there's always gonna be need, there's always gonna be the need to do maintenance on the protocol, but are there like a few things that are in the docket that may or may not get merged into Bitcoin or added to the, uh, areas above the protocol layer that, if added emerged, would get Bitcoin to a state that you think is okay to run?

Sort of at that state into perpetuity.

**Dan:** I'm not sure I know exactly what you mean. Like I see some sort of change that's coming that's going to make Bitcoin private. Or are you talking about something else?

**Marty:** Like just in general, like if we got, like some people think like if we get L [01:07:00] two, um, uh, and a couple other upgrades like Bitcoin at the protocol level at least will be at a place where it's like, all right, this is good enough.

Like if we had ossification, like this would be okay. Um, is there a combination of things like that in your mind that if the protocol were to get to, you'd be like, all right, this is pretty robust, pretty resilient? Hmm.

**Dan:** I don't know. I focus very narrowly on the privacy problem sometimes where I feel like I'm missing things that are happening on the mailing list or at the protocol level.

It seems like there's a lot in terms of P2P privacy that's happening that I'm really excited for and probably my optimism comes from support like this of P DK and the idea that pay join can make a real difference. And some of the research I've done that [01:08:00] convinces me that we can get reliable privacy in Bitcoin at a systems level instead of just with specific applications like that mindset, I think is really what changed it more than anything because when we were all focused on use this application, get privacy, I just never saw how that could expand to all of Bitcoin.

But thinking of protocols like Lightning and Pay, join as kind of. Like in the operating system model, when you're doing systems programming, you think of the operating system and then you have drivers on top of that to use your peripherals, which are kind of, you know, to run your audio and whatnot. And I think of lightning and pay join and other systems like device drivers, you're not using a device necessarily, but it gives you a feature.

And because they're engineered to be systems that run in all these different wallet softwares, they [01:09:00] become, become ubiquitous. And I think we're getting to a point with privacy where I see that as becoming possible. And that's where my optimism comes from, is like this systems perspective rather than the application's perspective.

Hell yeah.

**Marty:** Yeah, that makes a lot of sense. And for anybody listening right now that wants to help push P DK forward, maybe contribute, where

**Dan:** should we send them? Page Joinin dev kit.com. Sorry, page join dev kit.org is the place to go first. Yeah, I know. I don't even know my own url. Um, so the GitHub and the Discord are linked there.

We're on Twitter too as page joinin dev kit.org. If you wanna just an overview of page join page, join.org is that, uh, we're quite consistent in Discord. There's a bunch of different projects you can get involved in. So there are a couple designers shout out to Bob Space for supporting this work and willing, being willing to collaborate.

And [01:10:00] BDK of course is Steve Meyers and Summer of Bitcoin are doing a huge help to get p DK out there. And then if you want it in your specific application, the bindings are growing and if we can work with you to get a implementation into your wallet, there's a number of people who are willing to get that done.

**Marty:** What are, uh, the top wallet on your list to get it implemented in?

**Dan:** Um, I mean, whatever bull Bitcoin's developing that is on the way. We've got some drafts for that and I, I think getting in an exchange is gonna really change the game bolts is another kind of receiver exchange that could be super functional and helpful.

The BDK cli of course. Um,[01:11:00]

I think, yeah, I don't know who wants to step up to add serverless pay join first. There are a few people, of course, like the people I mentioned that are willing, but as far as large scale rollouts, it's hard to say who's going to be first. Even Voltage has, because of the Lightning Pay join. They've reached out and talked about, okay, if we can introduce this in the dashboard and you can just fund with a pay join, that's so much easier than, you know, opening all these channels separately.

They could just open the inbound channel and the batched outbound channels you wanted in one funding transaction. So yeah, I just get a, I get kind of overwhelmed with all the different places this could go. Yeah. What could go

**Marty:** wrong with the serverless implementation?

**Dan:** The serverless implementation? I think the biggest risk is that we build something with either too many [01:12:00] dependencies or that's in too inflexible, so people don't want to integrate it.

And I think this is what's happened with some of the ones I mentioned previously, like where you have a spec and it works and it's great, and people that use it love it, but it only works with one application. So getting that early feedback, which I've gotten a lot of, fortunately, I've gotten a lot of collaboration.

People want this is critical to having a successful rollout. And I think that's what the BIP process is for. Uh, it gives you a clear way to post on the mailing list first, get people like Marty to scratch their head and say what's turn and want to talk about it and then get other developers to come in and say, that's wrong.

Don't do that. This might be a better idea. And come to that rough consensus where you can address everyone's concerns, build implementations, test it out, and have something that's robust and future-proof. Yeah, that makes a lot of

**Marty:** sense. It's, well keep crushing it. Dude, this is awesome. Very, uh, very [01:13:00] excited and feel fortunate that an individual like yourself has dedicated their life to solving this problem.

Cause like I said earlier, I think it's, uh, a big problem to solve and would do the world a lot of good if it is solved. So, Thank

**Dan:** you. Thank you. It's happening. We're winning.

**Marty:** We are winning. We're not gonna win. We're winning. We just gotta keep winning. You always have been. We always have been. Winning just doesn't always seem like it.

That's the truth. So Dan, thank you. We will, um, hopefully we can do this again and there's more, more to talk about when this is widely adopted throughout the space. I'd love to do like a, a group discussion with you and Francis. Um, cause I love Francis, number one and number two. I think he is underrated in terms of like how he's thought about [01:14:00] the infrastructure of his exchange and all the stuff that he's built and open sourced for others to, to leverage.

And I think. Like you said, bull Bitcoin getting in this and like adding it to their stack, their open source stack could be massive for

**Dan:** people. I think other exchanges could see what's possible when someone takes a chance. And Francis has been on the pay join bug for a long time. I think the tech is just catching up, but all the potentials you mentioned, yeah, I'm interested.

It sounds awesome. Let's do it. Yeah.

**Marty:** All right. Well you go enjoy your 4th of July weekend. I'm gonna go do the same and uh, I guess I'll see you on the internet.

**Dan:** Thanks Marty. See you on the internet. All right, see you

**Marty:** Dan. Peace of love freaks king.