Software Licensure in the decentralized world

Today we had an encounter with the kujira whale.

And the weirdest thing was that the whale was not wrong. Because Notional is beginning to assist more and more groups with opening their work, I wanted to publish in a somewhat formal manner, my thoughts on proprietary code in the context of open decentralized systems. First of all, when Juno basically didn't exist – we made the decision to allow for permissionless deployment. And this means that we can't check contract code.

At that time, IBC was much younger and it wasn't fully clear what worked well and didn't. I still think that we made the right choice– two chain halts later, for that matter. Thing is, it was the right choice for our context. At that time, I barely understood CosmWasm's full potential. I recently spoke to some of the other early team members – and everyone says "ah, yeah, so that was Jake. (Hartnell). He saw things in it no one else did."

Our team at Juno has expanded dramatically (consider the wide view: at least 150, since we've 150 validators), and it's my belief that the lines defining core one who were who is not team and this idea of core two are all going to change and get super blurry. And that is sort of the defnition of success here– for Juno to become self-perpetuating and not depend on any of us as individuals.

value preserving software licensure

The other day someone asked me my take on the future of kujira, I told them I didn't really know, did not have enough information. I still don't, because I basically can't. The search code is not open. So when I think about that project, I know think about it with a dramatically increased risk profile. Basically I think that keeping these contracts closed first of all that's their decision and I respect it. Secondly, it increases the likelihood that their users will be subject to potentially a rug pole type event caused by the team, or potentially a rug pull type event caused by an external team like North Korea's Lazarus group: in the end the effect on users is the same. It seems to me that the could you're a whale actually recognizes these risks. So this document is not really about Kujira. This document is instead about unusual decentralized finance product blends.

Insecure open source software is still insecure software and if it is insecure, it is unlikely to pull together the kind of value that is necessary to make it secure. A few days ago I was working with a client to make the source code to their wallet available and I was successful and so were they:

```
1) their wallet is tied to their chain's community
2) their wallet cannot be legally copied
3) their wallet can be examined by their user community
```

From my standpoint, I'm satisfied, too. The project, btw, is Pylons, it's cool, and you should check it out! Not everything is fully baked yet, and it was important to them to really nail the notion of having a really nice mobile interface. It's written in flutter and the arcitecture is something I've wanted to build out for years. It's a delight to see all of these things coming together. Notional has refused similar work, where only one of those three variables above is missing: "their wallet can be examined by their user community."

A blockchain isn't a bank, and your crypto keys have no resemblance to a credit card. Losing those is permanent. Of course, the pylons wallet is a bit special: though it holds a seed phrase, it can also accept Visa, Mastercard (via stripe) as well as Google and Apple pay. It has taken the Pylons team a long time, a lot of money and even more hard work to build. They've done the legal. The wallet is not an exchange, it instead allows users to buy tickets and nfts and… many things (but not crypto) with regular old credit cards.

Notional's suggestion for projects wishing to preserve value

It is my personal and very strong opinion that the ability to audit must be preserved in order to make security claims. Crucially, this means that users must be able to audit, the end users. You cannot rely on a third-party auditing firm and simply post the results. A user cannot possibly be expected to understand the competitive marketplace that those firms exist in. To some degree, I do. Because of that I am quite sure that there are extremely real conflicts of interest and that those conflicts of interest could easily harm users, and have harmed users in the past. At Notional, and alongside our clients and the user community, we solicit cruel audits. That is to say, we have our own rubberstamps and if we wanted that, well those are very cheap anyhow.

A "cruel audit" is when the auditor understands that you want them at their most rigorous– not necessarily their fastest, or their easiest to deal with. These audits provide tremendous value, and their results can be verified but ONLY if the source code is available.

Security needs to be proven, and when it is not proven, value accrual is constrained, b because unproven security does not exist. The reason that unproven security claims constrain value accrual is that as a protocol or project gets larger it becomes a more and more attractive target for attack – and because the largest investor groups in fact do their own audits.

Therefore, it makes tremendous sense to work on bespoke licensing arrangements. Yes, in that case, if everybody can read the code, someone may attempt to copy your work, you may not be able to stop them particularly if they do not pursue a one to one copy. With that said, off the top of my head I can think of numerous projects all working on decentralized finance, who are harming their own growth through opacity. In this industry only extreme transparency will build trust.

As someone concerned with security, I must say that I unfortunately feel that it is completely correct and even the safest possible course to point it out when I see projects claiming to have secure, proprietary code. I do this because it protects users from harm, and I do it publicly because when the code can't be checked, there's no other way to verify security or even to make suggestions that improve security.

What projects who keep key bits invisible are really saying is, we are going to tell you that it is secure but we refuse to prove that.

Thing is, your code can simultaneously be visible and proprietary.

Our finding at Notional is that at some point these projects nearly universally regret this decision, because they eventually learn that they are the largest source of their own value accrual issues.

Summary

- It isn't necessary to allow others to use your code, the way the GPL stipulates. Use a different license, or make your own.
  - The MIT license is legally well accepted and allows you to place riders/restrictions on how the code is used. You can, for example, stipulate that your code can only be used on the chain kujira-1, with block 11905532 bearing the hash 0572D7104F53D48D9FE54152EE830168B8E4ECE1B60FFE31B4454BEB9D55AEB4 proposed by Danku_Zone w/ DAIC. I'd support Kujira in legally going after anyone violating your license terms, if Kujira's contracts were licensed that way.
- Audited proprietary code may not have been audited at all.
- It is srtictly necessary to make source code available for public review, to make security claims.