

# The Real Web3: A simple proof-of-concept

February 28<sup>th</sup>, 2024  
John Rigler ([john@rigler.org](mailto:john@rigler.org))

## Summary:

We seem to have largely agreed on what **Web3** is supposed to be, yet today are experiencing something closer to **Web2.8**. This vanillaJS arbitrary data pinner shows both what “**The Real Web3**” might require and also provide to various use cases. The most obvious use case would be to create a censorship-proof message board where each user is known by their confirmed self-sovereign identity.

This product includes a small smart contract which have been implemented on the Polygon network. It also contains what appears to be a novel unspendable address hack to create a “Web3 Table of Contents” and native (fully on-chain) word search.

## Part One: A no-nonsense tech stack

My version of Web3 is build with the following rules in mind. You follow along and see the working prototype at: <https://rigler.org/web3> ← immediate redirect to IPFS.

### Rule One: No DNS or traditional web server (boot from Web3)

A truly distributed solution should be detached from any single choke point. For this reason, we boot from either IPFS or Arweave.

### Rule Two: No nodejs

We all love Nodejs, but for the purpose of this demo, we must reduce our server footprint to almost nothing. Lightweight Javascript is easy to audit. Since we are using content-based addressing, we can now audit and understand not only how code operated, but also that it won't change. Node frameworks can be transpiled, minified, and included in an IPFS or Arweave boot-up system, but this original project was conceived without it.

### Rule Three: Always store helpful public data on-chain

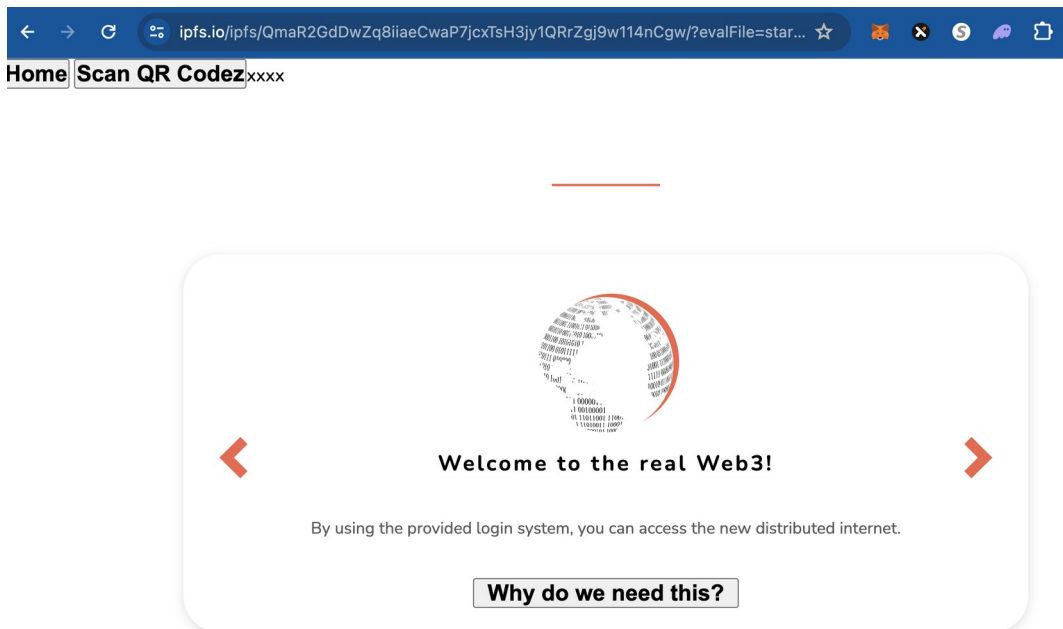
This is a big one. This demo aims to be simple and powerful. If you have enough crypto to pay gas fees, you should be completely in control of your actions. By writing everything to the ledger, this becomes obvious. You can also share your “boot up” Web3 index page with other people since you each could burn different sets of data into the ledger. Of course some temporary data solutions exist, but are a distraction from the very direct message of this paper.

## Rule Four: The Real Web3 is hacked out of an amoral and uncontrollable public blockchain ledger

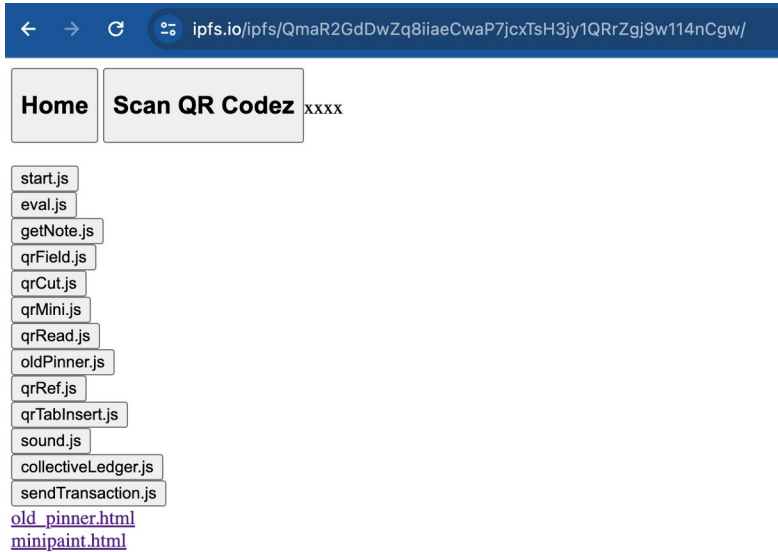
We can endlessly debate the relative merits or problems with etching data (ordinal inscription style) directly into a permanent record, yet we can only apply a sense of morality and “good practice” to our own actions. Because miners and validators do not scan content, no product offering ends up acting as a middleman. This means no censor. Without censorship, we can not enforce morality on others and thus we must come to the conclusion that **Web3 is beyond our control**. Once the places where it touches Web2 are removed, it takes on new, unexpected, powers. This doesn't mean that computer programmers shouldn't act with a strong sense of morality, but just that the rules of this new universe are not what they might have previously imagined.

### Part Two: The actual demo

The demonstration starts in pretty familiar territory. You can experience it by visiting <https://rigler.org/web3>. This page is one of a thousand that could lead to the same place. Anyone with IPFS knowledge could pin it into their library. If they first correct the intentional typos, the its content-based address would change. This code is fully functional and can create and read smart contract transactions.



Included is a simple table of contents (click Home to see):

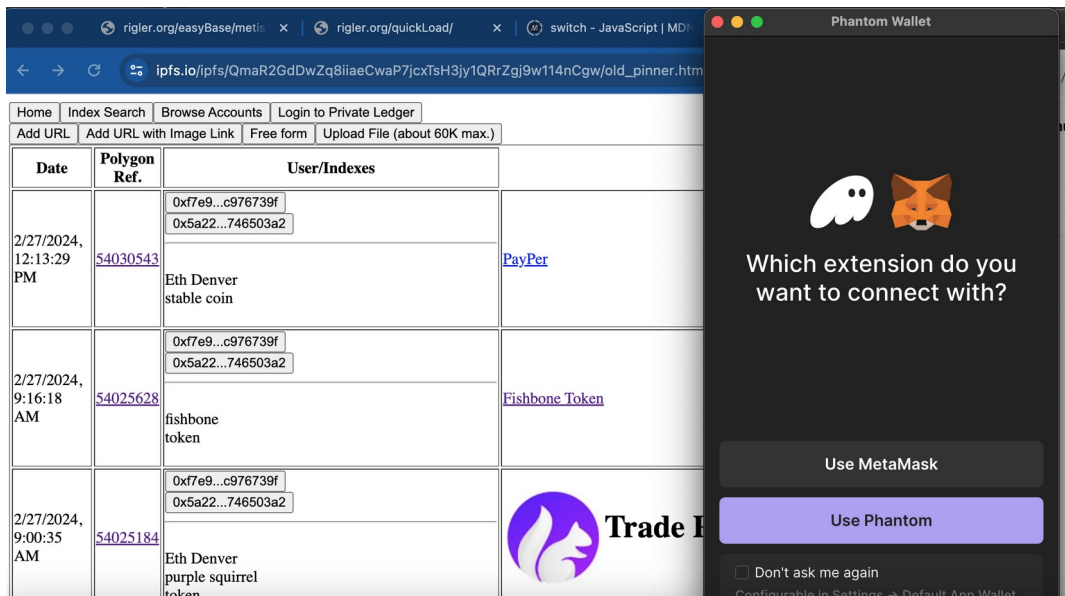


The

“old\_pinner.html” link is the actual demo, included in a larger set of tools.

The tool simply allows you to upload arbitrary data into the input field of a smart contract. It then relies on a block explorer to render each transaction and decode any data.

(Click on old\_pinner.html and you will see this.)



Below we see the “Add URL” option. This will simply create a hyperlink.

<a href="#">Home</a>	<a href="#">Index Search</a>	<a href="#">Browse Accounts</a>	<a href="#">Login to Private Ledger</a>
Title:	Orthoverse: Olwabarbia Token on LOOKSRARE		
URL:	https://looksrare.org/collections/0x118aed2606d02c2545c6d7d2d1021e567cc08922/910138533665E		
Search Words:	orthoverse	doom	olwabarbia 0x9F6C05F9F9803f6140
<a href="#">Next</a>	<a href="#">Clear Options</a>	<a href="#">Example</a>	
Date	Polygon Ref.	User/Indexes	
2/27/2024, 12:13:29 PM	<a href="#">54030543</a>	<div style="border: 1px solid gray; padding: 2px; margin-bottom: 2px;">0xf7e9...c976739f</div> <div style="border: 1px solid gray; padding: 2px; margin-bottom: 2px;">0x5a22...746503a2</div> Eth Denver stable coin	<a href="#">PayPer</a>
2/27/2024, 9:16:18	<a href="#">54025678</a>	<div style="border: 1px solid gray; padding: 2px; margin-bottom: 2px;">0xf7e9...c976739f</div> <div style="border: 1px solid gray; padding: 2px; margin-bottom: 2px;">0x5a22...746503a2</div> Fishbone Token	

Another option is allows for a similar hyperlink with an image instead of text. But since the real purpose of the application is to demonstrate what a pure Web3 implementation might look like, we simply provide you with a “free form” option:

<a href="#">Home</a>	<a href="#">Index Search</a>	<a href="#">Browse Accounts</a>	<a href="#">Login to Private Ledger</a>
<pre>&lt;pre&gt;&lt;h3&gt;Oh&lt;/h3&gt; Tell me what you want. What I really really want? Yea, tell me what you want. What you really really want. &lt;/pre&gt;</pre>			
Search Words:	<input type="text" value="wannabe"/>	<input type="text" value="spice girls"/>	<input type="text"/>
<a href="#">Next</a>	<a href="#">Clear Options</a>		

**Oh**

```
Tell me what you want.
What I really really want?
Yea, tell me what you want.
What you really really want.
```

<a href="#">Send</a>
----------------------

When you enter your choice and click “Next”, you are shown how your final product will look. Does the above look “good enough for forever”? Sure it does! Hit “send” to pop opened Metamask. Note that you can see the hex code of your transaction.

The screenshot shows a web browser window with the URL `ipfs.io/ipfs/QmaR2GdDwZq8iiaeCwaP7jcxTsH3jy1QRrZgj9w114nGcw/old_pinner.html`. The page content includes a text area with the text: "Yea, tell me what you want. What you really really want." Below this is a search bar with "wannabe" and "spice girls" entered, and a "Next" button. A "Send" button is also visible. A table below the search bar has columns for "Date", "Polygon Ref.", and "User/Indexes". The table contains one row with the date "2/27/2024, 12:13:29", the Polygon Ref. "54030543", and the User/Indexes "0xf7e9...c976739f" and "0x5a22...746503a2". A "PayPer" link is also present. On the right side, a Metamask transaction confirmation overlay is visible, showing the transaction details in hexadecimal format.

Date	Polygon Ref.	User/Indexes
2/27/2024, 12:13:29	54030543	0xf7e9...c976739f 0x5a22...746503a2

Once Metamask processes your entry, it will be on the blockchain (Polygon), and will show up at the top of the search.

## Quick Review: Web3 is a big deal

So let’s review what we know:


- With a Self-sovereign Ethereum Address ID, you are in complete control.
- No real Web2 homepage is need. This all lives in a new kind of distributed solution.
- Enter whatever arbitrary data that you want. No one will stop you.

And now lets add one more twist. A 100% Web3-native database is hacked into the transaction stack. Let’s look at the “Index Search” button:

The screenshot shows a web browser window with the URL `ipfs.io/ipfs/QmaR2GdDwZq8iiaeCwaP7jcxTsH3jy1QRrZgj9w114nGcw/old_pinner.html`. The page content includes a search bar with "Eth Denver" entered. A modal dialog box is open over the search bar, titled "ipfs.io says". The dialog box contains the text "Enter one search word:" and a text input field with "Eth Denver" entered. There are "Cancel" and "OK" buttons at the bottom of the dialog box. The background shows a table with columns for "Date", "Polygon Ref.", and "User/Indexes". The table contains one row with the date "2/27/2024, 12:13:29 PM", the Polygon Ref. "54030543", and the User/Indexes "0xf7e9...c976739f" and "0x5a22...746503a2". The text "Eth Denver stable coin" is also visible below the table.

Date	Polygon Ref.	User/Indexes
2/27/2024, 12:13:29 PM	54030543	0xf7e9...c976739f 0x5a22...746503a2

One of the “search terms” is “Eth Denver”. We have been adding these all week!  
Here is the results page:

1709061209 54030543 0xf7e9857afd27ba03493bdc57f2588e71c976739f MethodID: 0x3740071f Eth Denver stable coin
<a href="#">PayPer</a>
1709049635 54025184 0xf7e9857afd27ba03493bdc57f2588e71c976739f MethodID: 0x3740071f Eth Denver purple squirrel token
 <b>Trade Purple Squirrel Token today!</b>
1709038919 54020181 0xf7e9857afd27ba03493bdc57f2588e71c976739f MethodID: 0x3740071f Eth Denver proof-of-life

And here is a snippet from the polygonscan reference for one of the transactions:

From:

0xF7e9857AFD27Ba03493BdC57F2588E71C976739f

To:

0x5a2220d56f56db9C9F5B

0x4574682044656e7665720000000000000000

000

Transfer 1 wei From 0x5a2220...746503a2 To 0x457468...00000000

Transfer 1 wei From 0x5a2220...746503a2 To 0x70726f...00000000

Notice the target address of the two internal transactions. These are not spendable addresses. You can tell because of all the zeroes. But these also aren't hashes. They

are simply a straight hex conversion of the search strings:

E t h D e n v e r  
0x4574682044656e76657200000000000000000000

Did you see the hack here? Let's look a bit closer. It is just a simple ascii to hex conversion.

E	t	h	(space)	D	e	n	v	e
45	74	68	20	44	65	6e	76	65

This allows you to retool these “fake” or “unspendable” addresses as search indexes. Just search the API for internal transactions that match the word. Site such as Etherscan or Polygonscan will always index these as part of their basic offering as a block explorer. **This is huge.** Tokens are usually restricted to their own closed garden, smart contracts usually are too, but with this address hack, you can search the whole ledger for someone else sending the same signal.

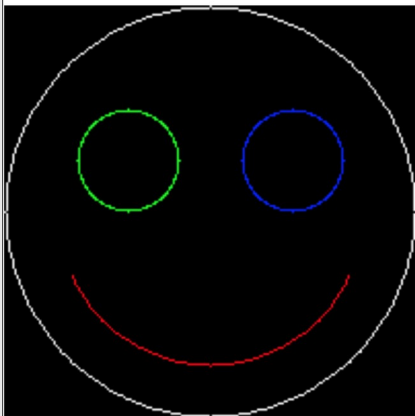
So now you not only have a way into the system where you know:

- no one is spying on you
- the “website” can't easily be blocked since it is in IPFS or Arweave
- you can “say” whatever you want with only Metamask
- core-level indexing is hacked right into the transaction for easy searchability

### Oh yea, these are also like small Ordinal Inscriptions

In this version, we use a small bit of PHP to decode the on-chain image. Soon, a completely on-chain image viewer will be included.

Let's look at that transaction a bit closer:

8/28/2023, 12:37:14 PM	46866724	0x36ef...9cb29a4c	
		0x5a22...746503a2	
		#png happy face	
		0x36ef...9cb29a4c	

🔍 Input Data:

```

9515c3a930c296c2ae211bc398c2b35453c2956e30c2a40bc2b81dc38bc2b2c2a47ec3
97c2963dc3be1dc3a9c2bbc2a458c38717c293c293c292622dc2be23c2bbc2ba7c6dc3
91c2aac2a75bc289c28d247513c39f21c39d00c38bc3910734c3ae0a7978c28c74c29d
c3aa0339c2b90ec296c3ba1dc3b27305523608c3b63a01c2993a0b19c3bb03c3bac3af
32c3a4c3ad23c2a4c3a626c2b4c3a51e32c3b22064c3b20b12c3b138c39d1bc2b5c2bb
7e63c29ac3a6c2b6c2a96c5fc39ac2b56e2fc2b5c391c2b4c38876c2bb1ec392c2a078
c38ec28bc38bc393c2a66c4bc39754c295c28d525528c39b2b392952c28b22326ac291
c2bec39573475f1dc393c3aac3bc67776800131857c3ba24c2ab51c3ba25c2bc13692a
c2952650c3b826c3812050c28f0f3ec3b36cc3adc398c2bcc2833cc28cc2abc3ac2856
50312638c28f561dc2b5c2a2c2817b3c524dc3b658c2a00ac3a3c2aa1ec3b135c28dc2

```

Below we see the PNG tag. Above we view the middle of the hex blob:

🔍 Input Data:

```

7@ #png happy face
A PNG

IHDRÈÈ ":9É
pHYs Ä Ä + IDATxíÛr«JDáôpÿ_æ<[[[ÆÉpÛôÛ<: [Z[16x[ eYtCHÃ@<äv[çyð
ÅóÛ=±öiÏaa[wÿu³Z[µ[©ñ@a[[[é0@!:[Ø³TS[n0[ , È²[~x[=pé»XÇ[[[b-
¾#>> |mÑª§ [[[$u[ß!ÝËÑ[4î
yx[t[ê[9¹[[[úòs[R6ö: [[[ú[ú[í2äí#æ&´â[2ð dð[[ñ8Ý[µ]~c[æ[©l_Úµn/µÑ
´Èv»[0 xÎ[ËÓ; lKxT[[RU(Û+9)R["2j[¾ÔsG_Óêügwh[[Wú$«Qú%¼[i*[[&Pø&Á

```

If you want to take a look yourself, this is polygon tx:

**txid:**

0xbf38024190bbca9ae2a73c85019042b1c9aa993cb75a09e408741dd5185229cb

**block height:**

46866724

**position in block:**

33

<https://polygonscan.com/tx/0xbf38024190bbca9ae2a73c85019042b1c9aa993cb75a09e408741dd5185229cb>

### The Origins (and future) of all this...

I consider this system to be series of clever little hacks of the data capabilities of EVM. But even before this, other hacks were possible. Here is an example with a



dogecoin like currency called Digibyte:

<https://digibyteblockexplorer.com/address/DBxYoUTUBEvCoMzzzzzzzzzzzzzzzzzzzzZ31xMU>

In this version. Addresses can be hacked into the ledger with my “unspendable” function. This works for various Gen. 1 currencies.

The screenshot displays a list of transactions in a ledger. Each entry consists of a recipient address, a label 'To', and a transaction amount in DGB. The amounts are shown in small boxes with a close icon (X). The transactions are as follows:

To	Amount (DGB)
DAxCAKEzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzWyGosU	0.00001
DMXwpMF9goRr8QTd1z2U8MuyzAEMT9HzYV	14.03404389
DBxYoUTUBEvCoMzzzzzzzzzzzzzzzzzzzzZ31xMU	0.00001
DCxTHExDiSTANCEzzzzzzzzzzzzzzzzzzzzWvWwSd	0.00001
OP_RETURN (https://www.youtube.com/watch?v=F_HoMkkRHv8)	0

The full meanings of DAxCAKEzzzz or DCxTHExDiSTANCEzzzz is beyond the scope of this paper, but their logic and placement was an earlier non-EVM search hack. These were built with:

<https://github.com/johnrigler/unspendable>





Also, in the BSV chain, the data (Gen. One/OP\_RETURN) field has no limit. I was able to etch a picture into that ledger and actually use the block explorer to read it back (see on next page). Inscriptions pre-date BTC-ordinals.

The screenshot shows the WhatsonChain website interface. At the top, there's a navigation bar with 'APIs', 'About', and 'Classic View' on the left, and 'English' and 'Mainnet' on the right. The main header features the 'WhatsonChain' logo and a search bar. Below the header is a table of transactions. Transaction #2 is selected, showing a total input of 0.10039692 BSV and a total output of 0.10004476 BSV. The selected output is labeled '#1 bitcom - B'. Below the transaction details, there are buttons for 'Decode', 'Download', and 'Report', and a dropdown menu for output format: 'ASCII', 'SCRIPT', and 'HEX'.

<https://whatsonchain.com/address/1FgJok3sLLbP4hmiCtURKLkARKt49W98JA>





The “Decode” button above reads the special “bitcom -B” format and renders as an image:


← → ↻ whatsonchain.com/address/1FgJok3sLLbP4hmiCtURKLkARKt49W98JA

Default Decoder    

Plugins Lab Store

Plugins [learn more](#)

-  WOC Default Decoder
-  Bico decoder
-  3D Model Viewer
-  STAS Viewer



So some of these ideas are novel, but just under the surface, Web3 had been in the works for a long time. The difference between the image above and the happy face drawn in Polygon is that this BSV image involved me running and using a huge core server , and the polygon version was done only with metamask and the new Web3 tool.

## Conclusion

This tool was designed to be simple, small, and hopefully readable. My main goal was to prove to myself that a **RealWeb3** application could be created under a strict set of requirements. The ideas expressed in this paper are general in nature and should not be patented. I do truly believe that this is “more Web3” than some server-based tool. This difference is more than just academic. The Media Scholar Marshall McLuhan is famous for saying “The medium is the message”. I believe that the lack of gatekeeper granted by applications such as this are the foundation of a new medium-- which I call Web3.