How Do I Pay Thee? Let Me Count the Ways

Leveraging Smart Contracts to Facilitate Web Monetization Adoption

Sophia Lichterfeld and Garima Rastogi

MIT PRIMES CS/Bio Research Track

October 15th, 2023

Overview

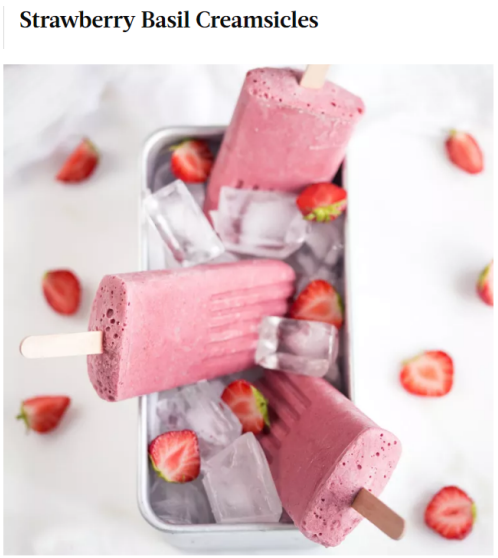
1. What is Web Monetization?
2. Background
   - Previous attempts of solving this problem
   - Smart contracts
3. Our proposed solution
   - Approach
   - Implementation
4. Future Work
5. Conclusions

What is Web Monetization?

Web Monetization

Web Monetization is how online creators can make money off of the content they produce.

Primarily through ads ☹️

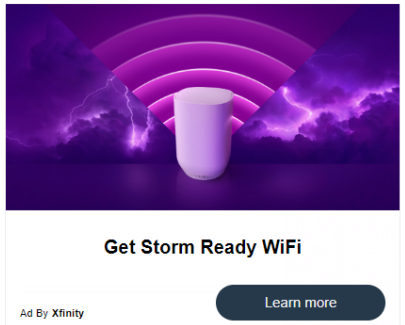


Advertisement

Advertisement

Advertisement

Lots of carrots make this the best carrot cake. I like to hand grate my carrots since I prefer the texture, but you can use your food processor or buy pre-grated carrots from the store. When we first tested the cake, we scaled the number of carrots back to two cups since three cups just sounded a bit extreme. After baking and letting it cool, we were a little disappointed. So if you're making the cake and start to second-guess the number of carrots called for in our recipe below, don't. You need all three cups. It is a carrot cake, after all.



- Alternative methods exist! 🥺
  - e.g. subscriptions to remove ads (Spotify, Youtube)
    - Not practical for small websites
  - Direct payment schemes; not well-addressed or well-supported 😥

Previous Trials

How do we enable Internet users to pay to view just one recipe without seeing ads?

Issue Resolution/**WM Scheme**	**Coil**	**Blendle**	**Axate**	**Brave**
Pay-Per-Content/Streaming Model	🌊	💧	💧	🌊
Currency Interoperability	😐	😐	😃	😞
Simpler Client-Side Adoption	😃	😃	😃	😃
After-The-Fact Owner-Side Opt-In	😞	😞	😞	😞

⋮

- Simpler client-side adoption well-implemented
  - e.g., browser integration to ease client-side payments

Previous Trials

How do we enable Internet users to pay to view just one recipe without seeing ads?

Issue Resolution/**WM Scheme**	**Coil**	**Blendle**	**Axate**	**Brave**
Pay-Per-Content/Streaming Model	💦	💧	💧	💦
Currency Interoperability	😐	😐	😀	😞
Simpler Client-Side Adoption	😀	😀	😀	😀
After-The-Fact Owner-Side Opt-In	😞	😞	😞	😞

⋮

- Website needs to adopt WM scheme for clients to be able to pay
  - Website can't be paid –> client not motivated to adopt WM scheme –> Website can't gauge client interest in the scheme
  - Our proposal: allow after-the-fact owner-side opt-in through **escrow**
- Eliminating initial owner-side opt-in will reduce this problem to just client-side

Background

W3C Web Monetization Project

Goal: standardize the implementation of web monetization schemes across the internet.

W3C (World Wide Web Consortium)

- Sets international standards for World Wide Web functionality/growth, in agreement with browser vendors and related parties
- Establishing standard for how **payment pointers** will be incorporated (through code)

```
<link rel="monetization" onload="console.log(event)" onmonetization="console.log(event)"
href="https://ilp.rafiki.money/yourName">
<meta name="monetization" content="$ilp.rafiki.money/yourName">
```

  - Benefit: browsers need to implement ONLY 1 payment pointer reader
  - Issue: if no payment pointer (form of pre-emptive owner opt-in), then browser can't send money anywhere

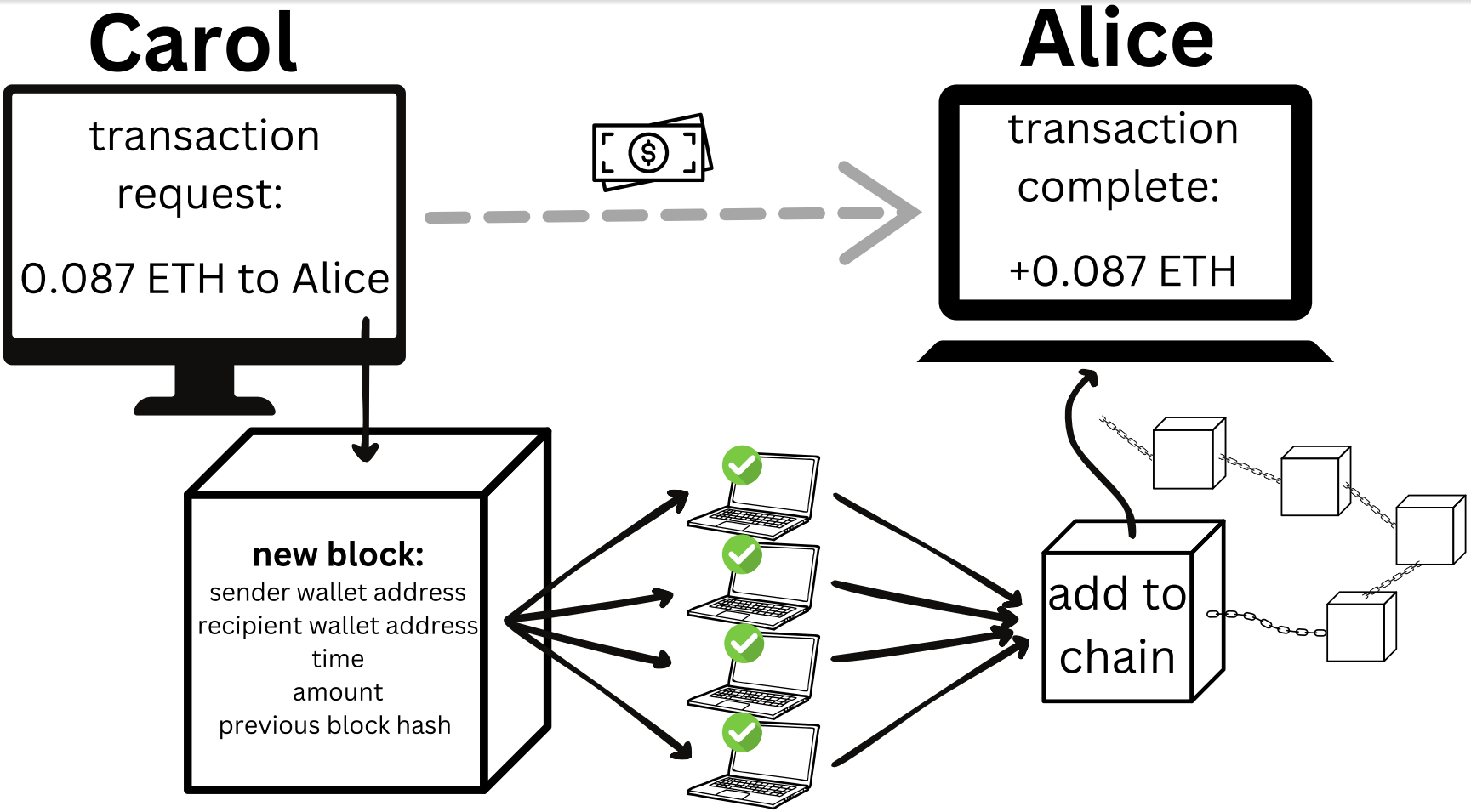
"Secondary" goal: increase/maintain currency interoperability.

- i.e. allow clients to use whatever payment scheme they are most comfortable with
- Current protocol: Interledger (public blockchain)

Decentralized Finance

Objective

Fostering a more decentralized, open online transaction network



Smart Contracts

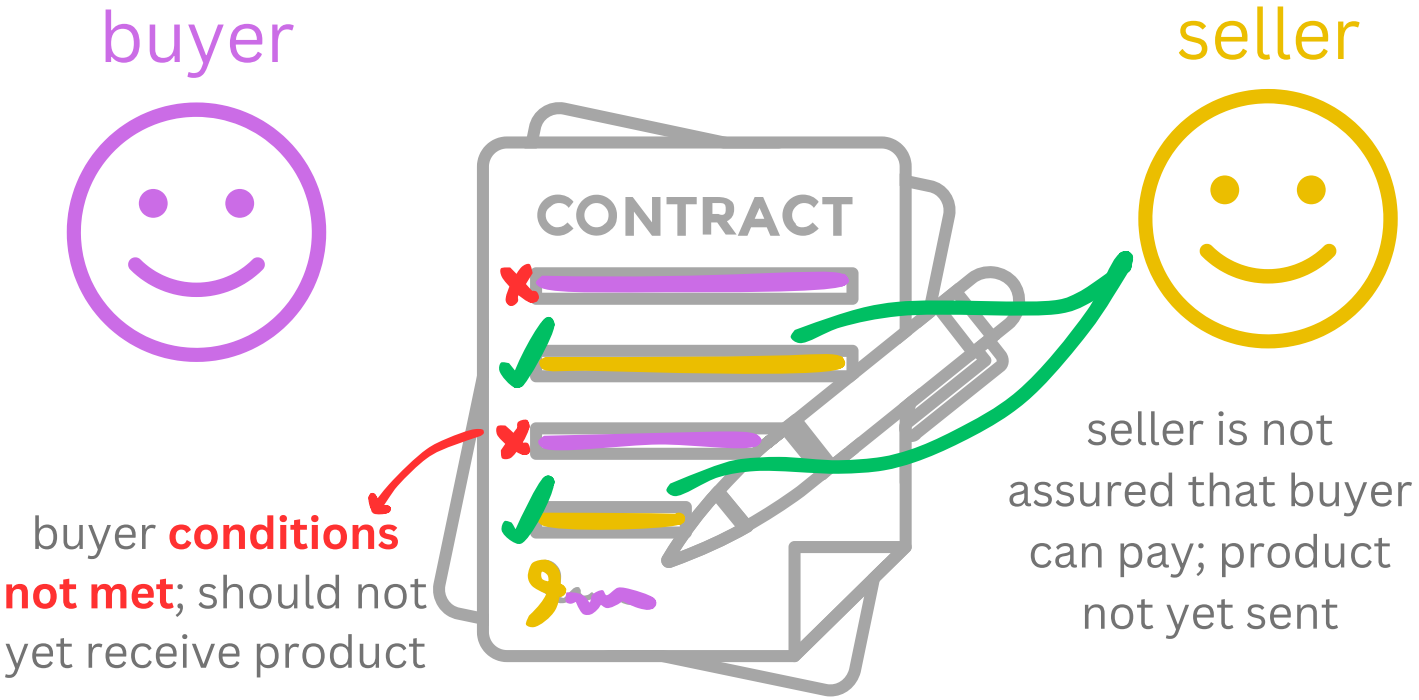
Definition

A section of code on the Ethereum blockchain that automatically executes—without the involvement of a third party—once both ends of an agreement have been upheld.

Smart Contracts

Definition

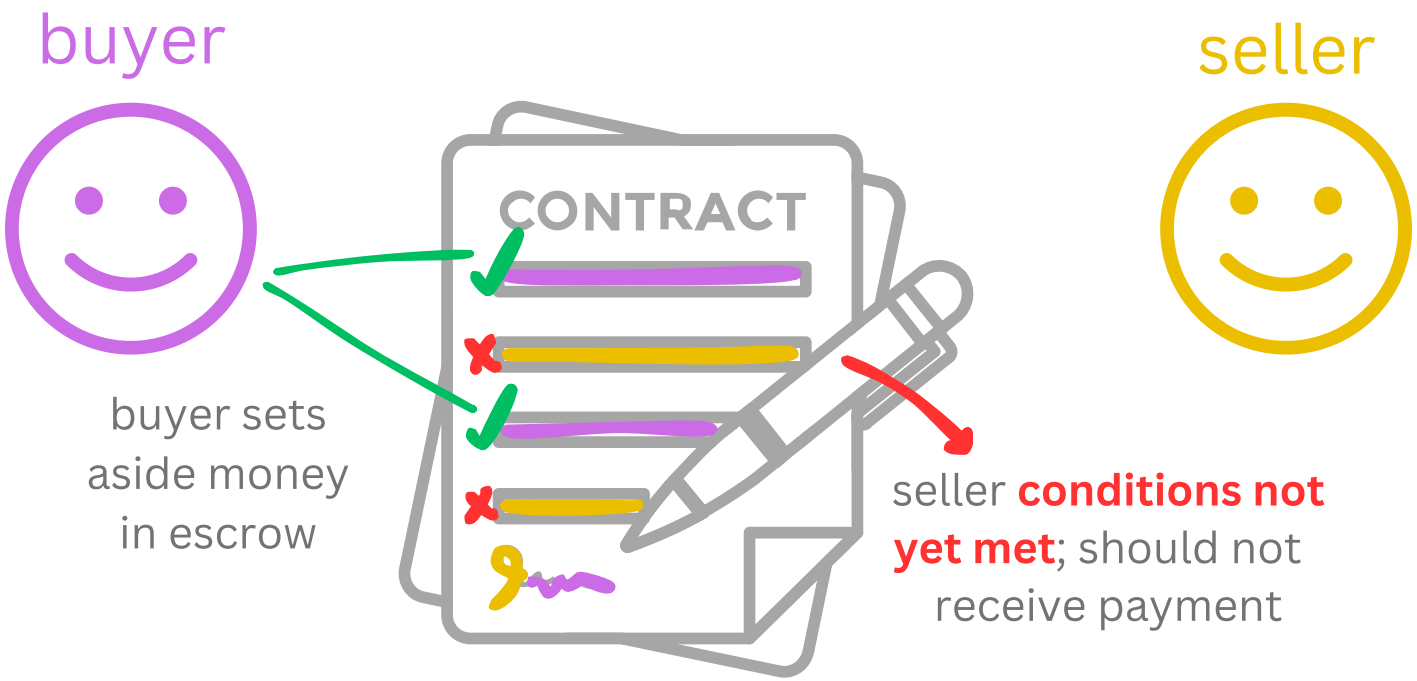
A section of code on the Ethereum blockchain that automatically executes—without the involvement of a third party—once both ends of an agreement have been upheld.



Smart Contracts

Definition

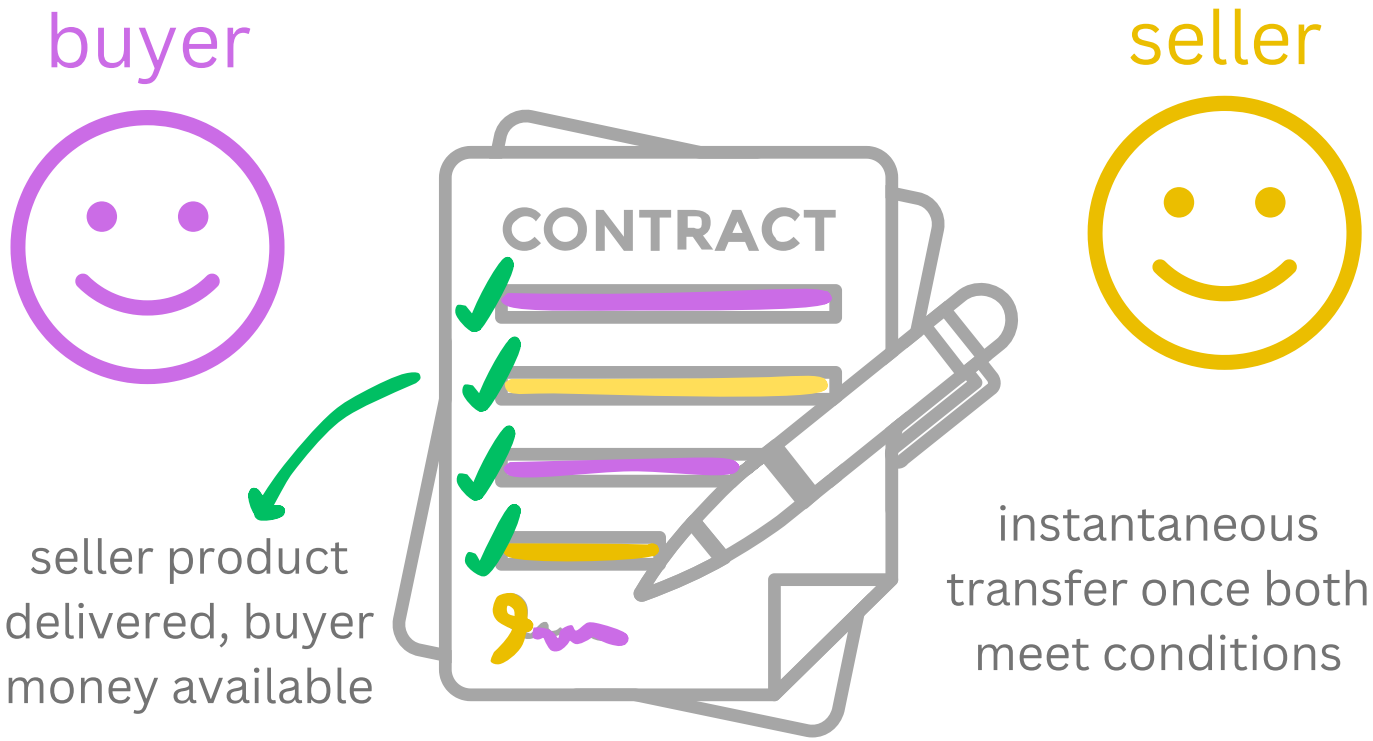
A section of code on the Ethereum blockchain that automatically executes—without the involvement of a third party—once both ends of an agreement have been upheld.



Smart Contracts

Definition

A section of code on the Ethereum blockchain that automatically executes—without the involvement of a third party—once both ends of an agreement have been upheld.



Proposed Approach

3 Potential Current States of WM

- Case 1: existing legible payment pointer
  - website already implements WM

3 Potential Current States of WM

- Case 1: existing legible payment pointer
  - website already implements WM
- Case 2: existing illegible payment pointer
  - notify owner that current payment pointer is invalid

3 Potential Current States of WM

- Case 1: existing legible payment pointer
  - website already implements WM
- Case 2: existing illegible payment pointer
  - notify owner that current payment pointer is invalid
- Case 3: nonexistent payment pointer

3 Potential Current States of WM

- Case 2: existing illegible payment pointer
  - notify owner that current payment pointer is invalid
- Case 3: nonexistent payment pointer

Cases 2 and 3 can be handled identically; in either instance, the browser fails to extract a payment destination address

Aims

Purpose

Allowing users to implement WM even before websites have adopted it yet
→ handle instances with mangled tag or no tag (*i.e.* cases 2 and 3)

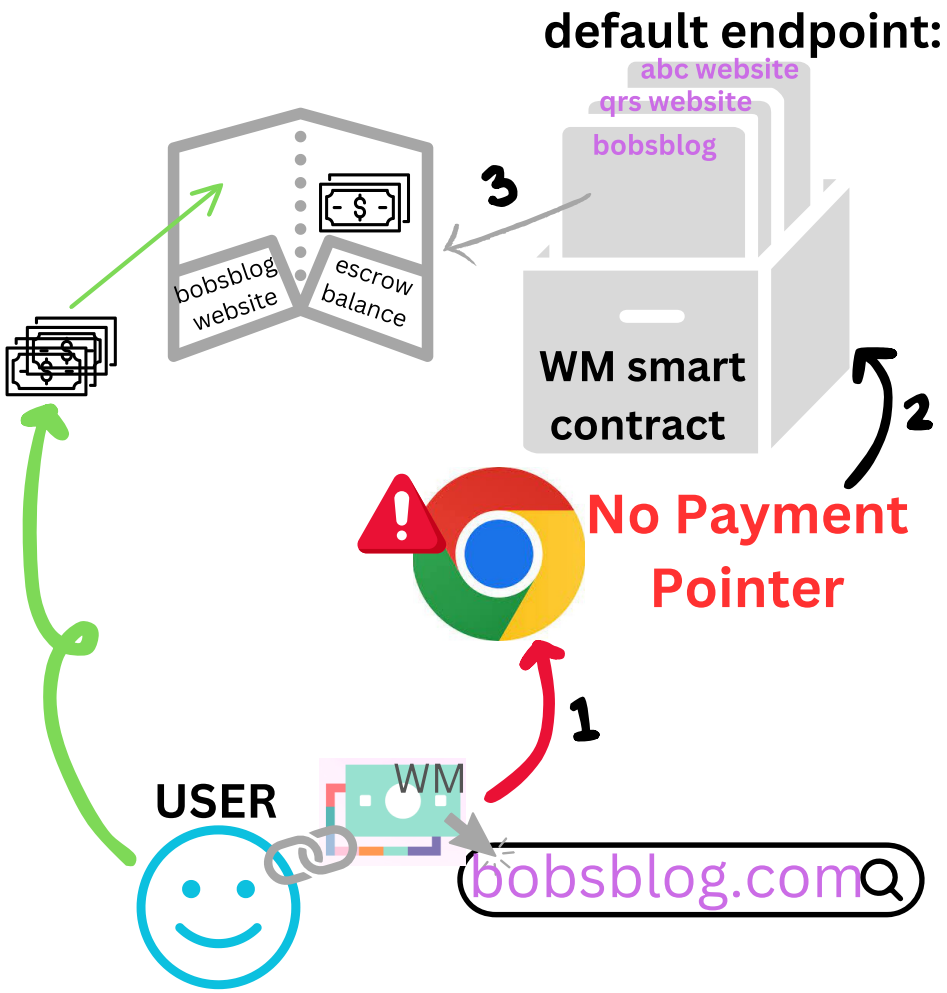
Requirements

1. System does not hinge on preemptive website adoption
2. Owner—and only owner—can collect money accumulated from WM at a later time point
   - respecting standard notions of website ownership: owner is the person who can edit website
3. Money is returned to user after "expiration" date

Payments from Clients to Contracts

Hold WM money temporarily in escrow in smart contract:

- smart contract address as default payment destination
  - user can implement WM even before the website owner has set up their end
- one smart contract for all websites
  - hard-code WM payment destination address
  - "subfolders" with WM revenue for individual sites (to preserve privacy)

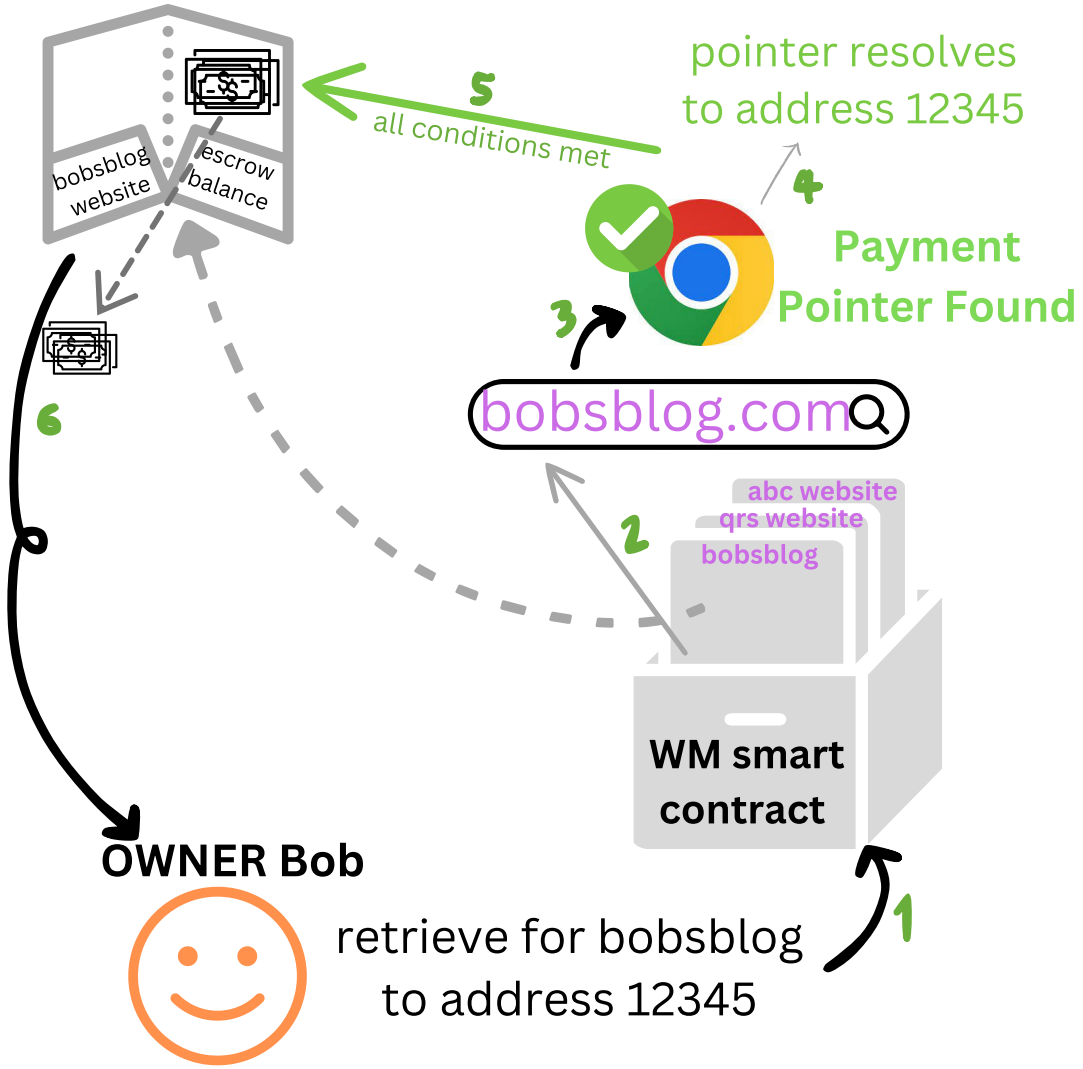


Payments from Contract to Website Owners

Conditions to retrieve money:

1. website has valid pointer
   → *website has adopted WM*
2. website pointer matches provided wallet address
   → *the person requesting to retrieve is owner*

Money accumulated in escrow is transferred to owner's account; future WM uses legible payment pointer now in HTML head



Implementation

Link to GitHub: https://github.com/s-lichterfeld/PRIMES-2023

Escrow Collection in a Single Smart Contract

Inputs

$URL \leftarrow$ URL of the website
$amtStreamed \leftarrow$ micropayment value from this payment session
$rootDomain \leftarrow$ extracted root domain from URL
$websiteID \leftarrow$ one-way hash of $rootDomain$ mapped to $inEscrow$ in $websites$

Algorithm 1

extract(URL) $\rightarrow rootDomain$ ▷ www.my.pg.com/a → pg.com
hash($rootDomain$) $\rightarrow websiteID$ ▷ one-way hash
if $websites$!contains $websiteID$ **then** ▷ existing escrow folder?
  $websites[websiteID] \leftarrow 0$ ▷ initially $inEscrow = 0$
end if
$inEscrow += amtStreamed$
transact: $websites[websiteID] \leftarrow inEscrow$

Link to GitHub: https://github.com/s-lichterfeld/PRIMES-2023

Cashing Out Smart Contract

Inputs

address ← wallet address of user attempting to cash out
URL ← URL of the website
pointer ← payment pointer found by browser in the HTML header of the website with the given *URL*, via Oracle

Algorithm 2

```
pointer ← Oracle(URL)
if pointer is found && address == pointer then
    extract(URL) → rootDomain
    hash(rootDomain) → websiteID
    inEscrow ← websites[websiteID]          ▷ retrieve inEscrow value
    transact: address ← inEscrow            ▷ owner gets money from WM
end if
```

Link to GitHub: https://github.com/s-lichterfeld/PRIMES-2023

Future Work

Goal 1: Incorporate Interoperability

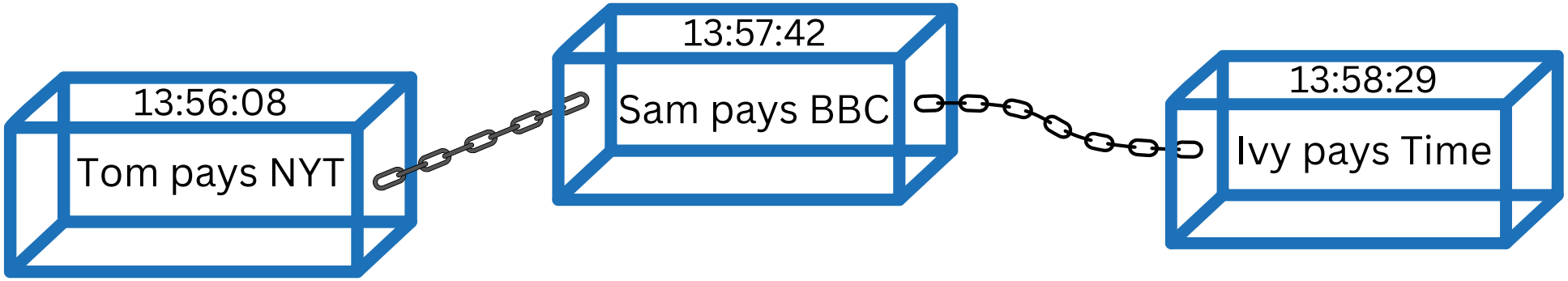
- Our project should be independent of money-transfer protocols
  - Reduces adoption barrier
- NOT reliant on Ethereum: used because good model for escrow

Goal 2: Minimize/Eliminate Transaction Fees (and overall costs)

- Such fees associated with all payments e.g. credit cards
- Web payments on scale of "micropayments," usually smaller than fees
- Also, smart contracts are insanely expensive due to Ethereum's decentralization
- Possible solutions:
  - Create own private blockchain (like Brave did)
    - Centralizes blockchain, thus reduces smart contracts' price
  - Utilize microtransaction solutions (e.g. side chains, lightning networks)
    - Allow transactions to stand off on side of blockchain during processing

Blockchain Transaction History

By default, **public** blockchains are highly transparent to enhance security:



Concern

With a blockchain-based system, we produce a permanent record of transactions that, combined with WM, reveals browsing history.

Definition

1. Does the transaction recipient know the sender's identity?
2. Do the remaining nodes know the sender's identity?

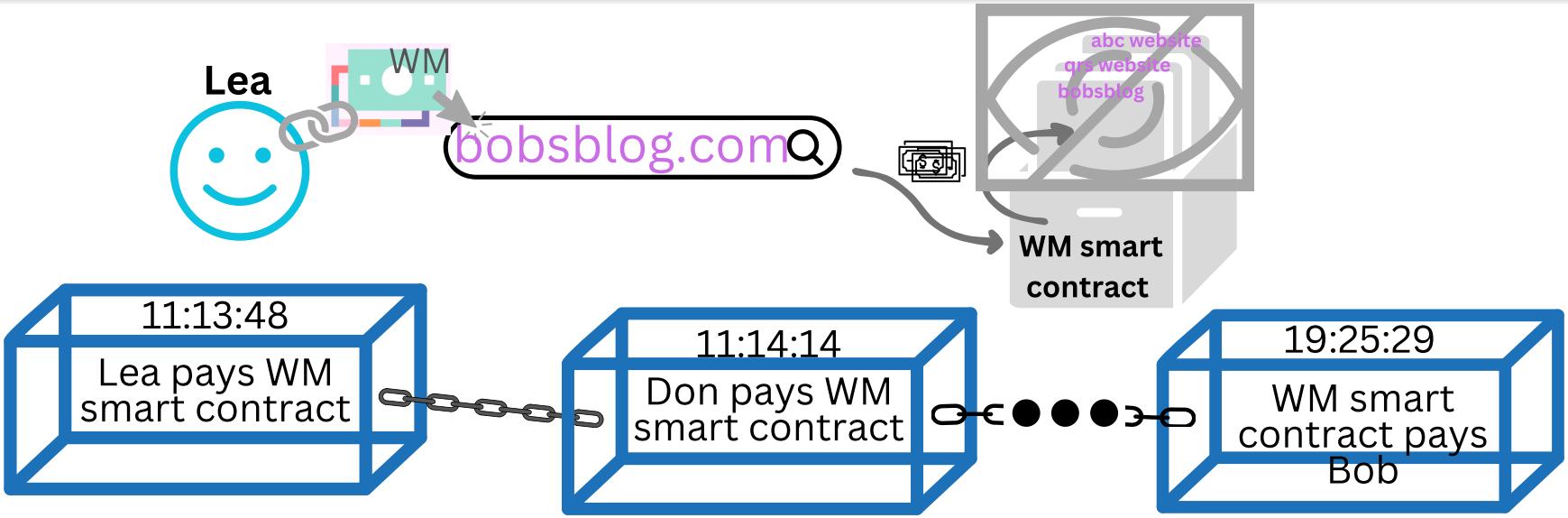
Ensuring Privacy

Private blockchains (*i.e.* Z-Cash) exist, but:

- no private smart contracts
- limit widespread adoption

Blinding Website Paid

Transactions by all users to any website stored in same WM smart contract → public blockchain only is informed that transaction was made into the contract as a whole



Conclusions

Impact

Aim

Driving the adoption of WM to provide users with an alternative to ads in the future.

Method

We implement a novel approach leveraging the escrow-holding ability of smart contracts. Our proposed scheme encourages widespread WM utilization by having website owners adopt the API standard to cash out.

Potential

- user- and creator-friendly avenue to increased WM adoption
- opportunity for users to directly support content creators
- financial incentive as stepping stone on the pathway toward a more decentralized online financial environment based on WM

Link to GitHub: https://github.com/s-lichterfeld/PRIMES-2023

Acknowledgements

We would like to thank

- our mentor, Ms. Kyle Hogan;
- MIT PRIMES: Dr. Srini Devadas, Prof. Pavel Etingof, Dr. Slava Gerovitch, Ms. Mayuri Sridhar, and Mr. André Lee Dixon;
- our parents