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?
What is Web Monetization?

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

Primarily through ads 😞

- 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 😞
What is Web Monetization?

Previous Trials

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

<table>
<thead>
<tr>
<th>Issue Resolution/WM Scheme</th>
<th>Coil</th>
<th>Blendle</th>
<th>Axate</th>
<th>Brave</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pay-Per-Content/Streaming Model</td>
<td>🌈</td>
<td>🌊</td>
<td>🌊</td>
<td>☁</td>
</tr>
<tr>
<td>Currency Interoperability</td>
<td>😞</td>
<td>😞</td>
<td>😊</td>
<td>😞</td>
</tr>
<tr>
<td>Simpler Client-Side Adoption</td>
<td>😊</td>
<td>😊</td>
<td>😊</td>
<td>😊</td>
</tr>
<tr>
<td>After-The-Fact Owner-Side Opt-In</td>
<td>😞</td>
<td>😞</td>
<td>😞</td>
<td>😞</td>
</tr>
</tbody>
</table>

- 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?

<table>
<thead>
<tr>
<th>Issue Resolution/WM Scheme</th>
<th>Coil</th>
<th>Blendle</th>
<th>Axate</th>
<th>Brave</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pay-Per-Content/Streaming Model</td>
<td>🌊</td>
<td>🛠️</td>
<td>🛠️</td>
<td>🌊</td>
</tr>
<tr>
<td>Currency Interoperability</td>
<td>😞</td>
<td>😞</td>
<td>🤖</td>
<td>😞</td>
</tr>
<tr>
<td>Simpler Client-Side Adoption</td>
<td>😊</td>
<td>😊</td>
<td>😊</td>
<td>😊</td>
</tr>
<tr>
<td>After-The-Fact Owner-Side Opt-In</td>
<td>😞</td>
<td>😞</td>
<td>😞</td>
<td>😞</td>
</tr>
</tbody>
</table>

- 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)

```html
<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

Carol
- TRANSACTION REQUEST:
  - 0.087 ETH TO ALICE

new block:
- sender wallet address
- recipient wallet address
- time
- amount
- previous block hash

Alice
- TRANSACTION COMPLETE:
  - +0.087 ETH

add to chain
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.

- **buyer**
  - buyer *conditions not met*; should not yet receive product

- **seller**
  - seller is not assured that buyer can pay; product not yet sent
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.

buyer

seller

seller product delivered, buyer money available

instantaneous transfer once both meet conditions
Proposed Approach
3 Potential Current States of WM

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

Payment Pointer Found
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

Sophia Lichterfeld and Garima Rastogi
How Do I Pay Thee? Let Me Count the Ways
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

Payment Pointer Not Found
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
Proposed Approach

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
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)
Leveraging Escrow in Smart Contracts

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.

Sophia Lichterfeld and Garima Rastogi

How Do I Pay Thee? Let Me Count the Ways
Leveraging Escrow in Smart Contracts

Implementation

Link to GitHub: https://github.com/s-lichterfeld/PRIMES-2023
Escrow Collection in a Single Smart Contract

Inputs

URL ← URL of the website

amtStreamed ← micropayment value from this payment session

rootDomain ← extracted root domain from URL

websiteID ← one-way hash of rootDomain mapped to inEscrow in websites

Algorithm 1

\[
\text{extract}(\text{URL}) \rightarrow \text{rootDomain}  \quad \triangleright \quad \text{www.my.pg.com/a} \rightarrow \text{pg.com}
\]

\[
\text{hash}(\text{rootDomain}) \rightarrow \text{websiteID}  \quad \triangleright \quad \text{one-way hash}
\]

\[
\text{if websites !contains websiteID then}  \quad \triangleright \quad \text{existing escrow folder?}
\]

\[
\quad \text{websites[websiteID]} \leftarrow 0  \quad \triangleright \quad \text{initially inEscrow} = 0
\]

\[
\text{end if}
\]

\[
\text{inEscrow} + = \text{amtStreamed}  \quad \triangleright \quad \text{transact: websites[websiteID] \leftarrow inEscrow}
\]

Link to GitHub: https://github.com/s-lichterfeld/PRIMES-2023
Cashing Out Smart Contract

Inputs

\[ \text{address} \leftarrow \text{wallet address of user attempting to cash out} \]
\[ \text{URL} \leftarrow \text{URL of the website} \]
\[ \text{pointer} \leftarrow \text{payment pointer found by browser in the HTML header of the website with the given URL, via Oracle} \]

Algorithm 2

\[ \text{pointer} \leftarrow \text{Oracle}(\text{URL}) \]
\[ \text{if pointer is found} \land \text{address} = \text{pointer} \text{ then} \]
\[ \text{extract(\text{URL})} \rightarrow \text{rootDomain} \]
\[ \text{hash(rootDomain)} \rightarrow \text{websiteID} \]
\[ \text{inEscrow} \leftarrow \text{websites[websiteID]} \quad \text{// retrieve inEscrow value} \]
\[ \text{transact: address} \leftarrow \text{inEscrow} \quad \text{// owner gets money from WM} \]
\[ \text{end if} \]

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
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
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