A Large-Scale Exploratory Study on the Proxy Design Pattern in Ethereum Blockchain

Amir M. Ebrahimi  
Bram Adams  
Gustavo A. Oliva  
Ahmed E. Hassan
The Proxy Design Pattern: A cornerstone of conventional software design

The GoF Proxy Design Pattern
Programmable blockchains offer unique features to application development

- Ethereum introduced **smart contracts**, a software that runs on Ethereum blockchain.
- **Transactions** is the way to interact with a contract.

An Externally owned account represents either a user or developer.
Smart contracts are **not** monolithic applications

1. How prevalent are proxy contracts?

2. How are proxy contracts integrated into applications?
Proxy contracts: a dual-blade, enabling maintenance while fracturing blockchain immutability.

• How do developers upgrade smart contracts despite immutability?
  • Solution: Upgradeability proxy contracts

1. How prevalent are different types of proxies?
2. Tracking proxies is critical for security reasons
  • Lack of techniques that effectively detect proxies accurately, at scale and in a timely manner
### Dataset & research questions

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Ethereum’s dataset [Aug. 2015 to Sep. 2022]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>50M smart contracts</td>
</tr>
<tr>
<td></td>
<td>1.6B transactions</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Research Questions</th>
<th>RQ1: How prevalent is the proxy mechanism in the Ethereum ecosystem?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RQ2: What are different creational patterns for deploying proxy contracts?</td>
</tr>
<tr>
<td></td>
<td>RQ3: What are the different types and properties of proxy contracts?</td>
</tr>
</tbody>
</table>
Proxy Detection method: Our method efficiently identifies all proxy contracts in **under 15 minutes**

- A proxy has two signatures:
  1. It shall use the delegatecall
  2. The proxy contract shall have a similar interface to the actual serving contract’s ones.

- Our method matches the **behavior** of over 50M smart contracts against the two proxy signatures
Evaluation: Our method exhibits **perfect** precision and recall and **outperformed** a previous study

- Ground truth dataset (385 randomly labeled contracts)
  - Achieve 100% precision and recall

- Compared our performance with Salehi et. al’s work.
  - We detected 300K more proxy contracts
RQ1: How prevalent is the proxy mechanism in the Ethereum ecosystem?

- Prevalence metrics
  1. General ratio of proxy contracts
  2. Stakeholder adoption
  3. Smart contracts design
  4. Usage context
General ratio: 14% of all deployed contracts are proxies.

~50M deployed contracts

~7.2M (14%) are proxy contracts

Proxy contracts share of all deployed contracts
Stakeholder adoption: Over two-thirds of all EOAs who deployed a contract, at least deployed one proxy contract too.

The monthly cumulative ratio of EOAs who deployed a proxy contract.
**Design:** Proxy contracts are increasingly being used in the design of modular applications.

The monthly ratio of multi-contract transactions that involve at least one proxy contract.
RQ2: What are the different creational patterns for deploying proxy contracts?

**Method**

For each proxy we mined its transactions to figure out how they are deployed.
We found 12 creational patterns for deploying proxy contracts.

<table>
<thead>
<tr>
<th>Id</th>
<th>Creational Pattern</th>
<th>Deployment style</th>
<th>Prac. count</th>
<th>Proxy instance count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>EOA &gt; P</td>
<td>Off-chain</td>
<td>20,210 (94.66%)</td>
<td>50,174 (0.69%)</td>
</tr>
<tr>
<td>2</td>
<td>EOA &gt; FA &gt; P</td>
<td>On-chain</td>
<td>1,385 (6.49%)</td>
<td>6,618,012 (91.39%)</td>
</tr>
</tbody>
</table>

Reference for reading creational patterns

- Off-chain is the most popular deployment style among practitioners.
- Most proxy contracts are created using an on-chain style.

Item | Stands for
--- | ---
>    | Deployment operator
EOA  | Practitioner/Developer
P    | Proxy smart contract
FA   | Factory smart contract
### On-chain vs Off-chain deployment styles

<table>
<thead>
<tr>
<th>Off-chain</th>
<th>On-chain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smart contracts is deployed on blockchain</td>
<td>Smart contracts is deployed on blockchain</td>
</tr>
<tr>
<td>Deployment Infrastructure operates outside of</td>
<td>Deployment Infrastructure operates on</td>
</tr>
<tr>
<td>blockchain</td>
<td>blockchain</td>
</tr>
<tr>
<td>Deployment scripts are written in Web3,</td>
<td>A smart contract instantiates another smart</td>
</tr>
<tr>
<td>JavaScript</td>
<td>contract</td>
</tr>
<tr>
<td>Flexible</td>
<td>Less flexible</td>
</tr>
<tr>
<td>Less transparent &amp; secure</td>
<td>Transparent &amp; secure</td>
</tr>
<tr>
<td>Lower interoperability among contracts</td>
<td>Higher interoperability among contracts</td>
</tr>
<tr>
<td>Less likely for deploying proxy clones</td>
<td>More likely for deploying proxy clones</td>
</tr>
<tr>
<td>Often when a lower number of proxy contracts</td>
<td>Often when a lower number of proxy contracts</td>
</tr>
<tr>
<td>are deployed</td>
<td>are deployed</td>
</tr>
</tbody>
</table>
RQ3: What are the different types and properties of proxy contracts?

1. Classify proxies based on their purpose

2. Classify proxies based on their implementation type
Most proxies (86%) are forwarders whereas 14% are for upgradeability purposes.

Labels
• Upgradeability proxy
• Forwarder proxy

Sample
• 385 random proxy contracts

Method
• Qualitative Study

Research gap:
Automatic approaches that detect upgradeability proxy contracts and monitor releases
**41.3% of proxies are minimal proxies, while 40.7% are unknown implementations.**

- **Tool**
  - Evm-proxy-identification

- **Reference implementations**
  - 7 known proxy reference implementations

- **Sample**
  - 16,602 random proxy contracts

![Graph showing the distribution of proxy implementations](image)
Implications to practice

- Proxies reduce transparency in marketplaces.

- Future studies should aim to study smart contracts release engineering via upgradeability proxies.
Effective and efficient method for detecting proxy contracts.

RQ1: How prevalent is the proxy mechanism in the Ethereum ecosystem?

RQ2: What are the different types and properties of deploying proxy contracts?
RQ3: What are the different types and properties of proxy contracts?
Effective and efficient method for detecting proxy contracts.
RQ1: How prevalent is the proxy mechanism in the Ethereum ecosystem?
RQ2: What are the different creational patterns for deploying proxy contracts?
RQ3: What are the different types and properties of proxy contracts?

A. 86% of proxies are forwarders, whereas 14% enable upgradeability

B. The majority of proxies (60%) are implemented according to one of seven reference implementations.

C. 41% of all proxies follow the Minimal Proxy (EIP-1167) reference implementation.
The proxy design pattern in smart contracts
Upgradeability proxy contracts
Prem-study: Is the proxy pattern a relevant practice in the domain of smart contracts?

1. Analyze the ratio of multi-contract transactions
2. Analyze the ratio of multi-contract transactions with different call types
3. Take a sample of multi-contract transactions with at least one delegatecall trace
4. Study the ratio of proxy usage in the sample
• A proxy increases
  • modularity and encapsulation

• If the proxy pattern is employed, then there must be some transactions in which different contracts interact with each.

The monthly ratio of multi-contract transactions
A proxy uses

- Delegatecall type of call to interact with the actual contract

- If the proxy pattern is employed, the multi-contract transaction must use delegatecall

The monthly ratio of multi-contract transactions that use various types of calls.
• Not every delegatecall is a sign of using proxy contracts, e.g.,
  • Library calls use delegatecall
  • Etc.

• Use Etherscan to analyze
  • 385 multi-contract transactions with at least one delegatecall operation
• Not every delegatecall is a sign of using proxy contracts, e.g.,
  • Library calls use delegatecall
  • Etc.

• Use Etherscan to analyze
  • 385 multi-contract transactions with at least one delegatecall operation

98.5% of the studied sample uses a proxy contract.
Prem-study: Is the proxy pattern a relevant practice in the domain of smart contracts?

Summary

1. An increasing trend of up to 33% monthly in the ratio of transactions involving multiple contracts.

2. Most importantly, 98.5% of multi-contract transactions with delegatecalls involve a proxy contract.
RQ2: What are the different creational patterns for deploying proxy contracts?

1. We found 12 creational patterns that are categorized into two major styles: i) on-chain and ii) off-chain deployment styles.

2. While the off-chain deployment of proxies is the most frequently chosen (94.6%) style, the majority (99.3%) of proxies are deployed automatically using the on-chain style.
**Obs #1:** We found 12 different creational patterns for deploying proxy contracts.

<table>
<thead>
<tr>
<th>Id</th>
<th>Creational pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>EOA &gt; P</td>
</tr>
<tr>
<td>2</td>
<td>EOA &gt; FA &gt; P</td>
</tr>
<tr>
<td>3</td>
<td>EOA &gt; PF &gt; P</td>
</tr>
<tr>
<td>4</td>
<td>EOA &gt; FA &gt; PF &gt; P</td>
</tr>
<tr>
<td>5</td>
<td>EOA &gt; FA &gt; FA &gt; P</td>
</tr>
<tr>
<td>6</td>
<td>EOA &gt; PF &gt; PF &gt; P</td>
</tr>
<tr>
<td>7</td>
<td>EOA &gt; FA &gt; PF &gt; PF &gt; P</td>
</tr>
<tr>
<td>8</td>
<td>EOA &gt; FA &gt; FA &gt; FA &gt; P</td>
</tr>
<tr>
<td>9</td>
<td>EOA &gt; FA &gt; FA &gt; PF &gt; P</td>
</tr>
<tr>
<td>10</td>
<td>EOA &gt; FA &gt; FA &gt; FA &gt; FA &gt; P</td>
</tr>
<tr>
<td>11</td>
<td>EOA &gt; FA &gt; FA &gt; PF &gt; PF &gt; P</td>
</tr>
<tr>
<td>12</td>
<td>EOA &gt; FA &gt; FA &gt; PF &gt; PF &gt; PF &gt; P</td>
</tr>
</tbody>
</table>

![Diagram showing creational patterns](image)
Obs #1: We found 12 different creational patterns for deploying proxy contracts.

<table>
<thead>
<tr>
<th>Id</th>
<th>Creational pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>EOA &gt; P</td>
</tr>
<tr>
<td>2</td>
<td>EOA &gt; FA &gt; P</td>
</tr>
<tr>
<td>3</td>
<td>EOA &gt; PF &gt; P</td>
</tr>
<tr>
<td>4</td>
<td>EOA &gt; FA &gt; PF &gt; P</td>
</tr>
<tr>
<td>5</td>
<td>EOA &gt; FA &gt; FA &gt; P</td>
</tr>
<tr>
<td>6</td>
<td>EOA &gt; PF &gt; PF &gt; P</td>
</tr>
<tr>
<td>7</td>
<td>EOA &gt; FA &gt; PF &gt; PF &gt; P</td>
</tr>
<tr>
<td>8</td>
<td>EOA &gt; FA &gt; FA &gt; FA &gt; P</td>
</tr>
<tr>
<td>9</td>
<td>EOA &gt; FA &gt; FA &gt; PF &gt; P</td>
</tr>
<tr>
<td>10</td>
<td>EOA &gt; FA &gt; FA &gt; FA &gt; FA &gt; P</td>
</tr>
<tr>
<td>11</td>
<td>EOA &gt; FA &gt; FA &gt; PF &gt; PF &gt; P</td>
</tr>
<tr>
<td>12</td>
<td>EOA &gt; FA &gt; FA &gt; PF &gt; PF &gt; PF &gt; P</td>
</tr>
</tbody>
</table>

A metamodel that summarizes proxy creational patterns
Proxy Detection Approach & Evaluation

1. Detect proxy contracts
2. Construct the evaluation ground truth
3. Evaluate detection approach
i. Ground truth dataset (385 random contracts)
   • 90 proxy contracts
   • 295 others

ii. Compared our performance with Salehi et. al’s work.
- Ground truth dataset (385 random contracts)
  - 90 proxy contracts
  - 295 others

<table>
<thead>
<tr>
<th>Label</th>
<th>Precision</th>
<th>Recall</th>
<th>F1-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proxy</td>
<td>100%</td>
<td>68.9%</td>
<td>81.6%</td>
</tr>
<tr>
<td>Other</td>
<td>91.3%</td>
<td>100%</td>
<td>95.5%</td>
</tr>
</tbody>
</table>

The performance of our proxy detection approach.
- Ground truth dataset (385 random contracts)
  - 90 proxy contracts
  - 295 others

<table>
<thead>
<tr>
<th>Label</th>
<th>Precision</th>
<th>Recall</th>
<th>F1-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proxy</td>
<td>100%</td>
<td>68.9%</td>
<td>81.6%</td>
</tr>
<tr>
<td>Other</td>
<td>91.3%</td>
<td>100%</td>
<td>95.5%</td>
</tr>
</tbody>
</table>

![Confusion Matrix Diagram]
• Ground truth dataset (385 random contracts)
  • 90 proxy contracts
  • 295 others

<table>
<thead>
<tr>
<th>Label</th>
<th>Precision</th>
<th>Recall</th>
<th>F1-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proxy</td>
<td>100%</td>
<td>68.9%</td>
<td>81.6%</td>
</tr>
<tr>
<td>Other</td>
<td>91.3%</td>
<td>100%</td>
<td>95.5%</td>
</tr>
</tbody>
</table>

28 inactive proxy contracts

Non-Scope

295 Other

62 Proxy

28 Proxy

0 Other
• Compared our performance with Salehi et al’s work.
  • Both approaches are behavioral
  • Sep-05-2020 to Jul-20-2021
• Compared our performance with Salehi et. al’s work.
  • Both approaches are behavioral
  • Sep-05-2020 to Jul-20-2021

<table>
<thead>
<tr>
<th>Approach</th>
<th>Number of proxy contracts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Salehi et. al. approach</td>
<td>1,427,215</td>
</tr>
<tr>
<td>Our approach</td>
<td>1,723,309</td>
</tr>
</tbody>
</table>

We detected
~ 300K more proxies
RQ1: How prevalent is the proxy mechanism in the Ethereum ecosystem?

**Summary**

A. Our approach
   i. Efficiently mines active proxy contracts (in under 20 min)
   ii. Achieve perfect precision and recall
   iii. Improves upon previous studies

B. Over 14% of contracts are proxies.

C. Our assessment from the three viewpoints of usage context, stakeholder adoption, and smart contracts design shows that the tendency for using proxy contracts is growing.