PAYMENT PROJECT

Principal Investigator: Prof. Steven Murdoch

Research Fellow: Aydin Abadi

UCL
PAYMENT PROJECT
Progress So Far

PAYMENT

Authorised Push Payment (APP) Fraud
Privacy Enhancing Technology
User Authentication
Cryptocurrencies

Payment with Dispute Resolution
Developed Attacks on Private Set Intersection (PSI)
Developed New Updatable PSI
Developing New PSI with Secure Payment
Two Factor Authentication With Forward Security
Fair Payment System for Cloud’s Storage

Paper
Paper
Paper
Paper
Paper
• “Authorised Push Payment” (APP) fraud:
  • Definition: An APP fraud is a type of cyber-crime where a fraudster tricks a victim into making an authorised online payment into an account controlled by the fraudster.
  • It is called “authorised” because the victim authorises the payment.
  • The APP fraud has various variants, such as:
    • romance
    • investment
    • CEO
    • invoice
The amount of money lost due to APP frauds is substantial

- Only in the first half of 2021, a total of £355 million was lost to APP frauds.

- APP fraud is a global issue.

- According to the FBI’s report, victims of APP frauds reported at least a total of $419 million losses, in 2020.

- Recently, Interpol warned its member countries about a concerning variant of APP fraud called investment fraud via dating software.
Although the UK’s regulators (unlike other countries) have provided specific guidelines to financial institutes to prevent APP frauds occurrence and improve victims’ protection, these guidelines are:

- ambiguous
- open to interpretation

There exists no mechanism in place via which honest victims can prove their innocence.

To date, the APP fraud problem has been overlooked by the information security and cryptography research communities.
To facilitate the compensation of APP frauds victims, we:

1. proposed a **new protocol** called “Payment with Dispute Resolution” (PwDR).

2. **formally defined** PwDR.
   
   • Identified its core security properties:
     
     (i) security against a **malicious victim**.
     
     (ii) security against a **malicious bank**.
     
     (iii) **privacy**.

3. **formally proved** the security of PwDR.
PAYMENT PROJECT
Authorised Push Payment (APP) Fraud

Our Solution’s Features

• The PwDR offers **transparency** by
  
  (1) **accurately formalising** reimbursements’ conditions
  
  (2) **offering traceability**
  
  (3) **providing an evidence-based final decision**

• The PwDR offers **accountability**, as it is equipped with **auditing mechanisms** that help identify the party liable for an APP fraud loss.
  
  • The auditing mechanisms themselves are accompanied by our lightweight **privacy-preserving threshold voting** protocols.
    
    • Our voting protocols let auditors **vote privately** without having to worry about being retaliated against, for their votes.
The PwDR is **efficient**: 

- We analysed the PwDR’s cost via both:

  - asymptotic analysis
  
  - concrete evaluation

- our analysis indicates the protocol is highly efficient.
## PAYMENT PROJECT
### Protecting Victims of APP Frauds
#### The PwDR Protocol's Cost

### Asymptotic cost analysis

<table>
<thead>
<tr>
<th>Party</th>
<th>Setting</th>
<th>Computation Cost</th>
<th>Communication Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer</td>
<td>$e = 1$</td>
<td>$O(1)$</td>
<td>$O(1)$</td>
</tr>
<tr>
<td>Bank</td>
<td>$e = 1$</td>
<td>$O(1)$</td>
<td>$O(1)$</td>
</tr>
<tr>
<td>Arbiter $D_1, \ldots, D_{n-1}$</td>
<td>$e = 1$</td>
<td>$O(1)$</td>
<td>$O(1)$</td>
</tr>
<tr>
<td>Arbiter $D_n$</td>
<td>$e = 1$</td>
<td>$O(n)$</td>
<td>$O(1)$</td>
</tr>
<tr>
<td>Dispute resolver</td>
<td>$e = 1$</td>
<td>$O(n)$</td>
<td>$O(1)$</td>
</tr>
</tbody>
</table>

### Concrete cost analysis

<table>
<thead>
<tr>
<th>Party</th>
<th>Time in millisecond</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arbiter $D_n$</td>
<td>n = 6</td>
</tr>
<tr>
<td></td>
<td>$e = 1$</td>
</tr>
<tr>
<td>Arbiter $D_n$</td>
<td>0.019</td>
</tr>
<tr>
<td>Dispute resolver</td>
<td>0.001</td>
</tr>
</tbody>
</table>

---

### Footnotes

- n: number of arbiters
- e: threshold
- *: Denotes an outlier
The PwDR Protocol's building blocks:

- Commitment scheme
- Digital signature
- Smart contract and blockchain
- Pseudorandom function
- Bloom filter
- Threshold voting protocols
PAYMENT PROJECT
Protecting Victims of APP Frauds
The PwDR Protocol's Workflow

• The PwDR Protocol involves two main phases:
  • Payment
  • Dispute resolution
PAYMENT PROJECT
Protecting Victims of APP Frauds
Extension and Further analysis of PwDR

• We have been:

1. extending the PwDR’s functionality
   • developing new privacy-preserving analytics.

2. further analysing the PwDR’s performance (in collaboration with Dr. Partha Das Chowdhury form the University of Bristol):
   • Implemented the PwDR’s smart contracts.
   • Analysed its costs.
PAYMENT PROJECT
Protecting Victims of APP Frauds
Implementation of smart contracts

Now Dr. Partha Das Chowdhury

will discuss the implementation of the smart contracts
PAYMENT PROJECT
Protecting Victims of APP Frauds

Architecture

Smart Contracts
1. SAP – Key Management
2. Add Payee
3. Generate Payment Request
4. Make Payment
5. Generate Compliant Request
6. Verify Key Agreement
7. Resolve Complaint

SevenOrgsChannel:
<configtx.yaml>

Consortium:
SampleConsortium
• ↔: *ChannelDefaults
• Application:
• ↔: *ApplicationDefaults
• Organizations:
  • -*Org1 - Bank
  • -*Org2 - Account Holder
  • -*Org3 - FCA
  • -*Org4 - Which
  • -*Org5 - Arbitrator
  • -*Org6 - Arbitrator
  • -*Org7 - Arbitrator
• Capabilities:
  • ↔: *ApplicationCapabilities

Docker Containers

Hyperledger Fabric – 2.2.3
Ubuntu 20.04.3 LTS - AWS
PAYMENT PROJECT
Protecting Victims of APP Frauds

Commands

./network.sh up

./network.sh createChannel -c drchannel -verbose

./network.sh deployCC -c drchannel -ccn sap -ccl go -ccv 0.1 -ccp /home/ubuntu/dispute-resolution/sap -ccep 
"AND(\"Org2MSP.peer\")"

./network.sh deployCC -c drchannel -ccn payee -ccl go -ccv 0.2 -ccs 1 -ccp /home/ubuntu/dispute-resolution/payee -
ccep "AND(\"Org1MSP.peer\")"

./network.sh deployCC -c drchannel -ccn payment -ccl go -ccv 0.1 -ccs 1 -ccp /home/ubuntu/dispute-
resolution/payment -ccep "AND(\"Org1MSP.peer\")"

./network.sh deployCC -c drchannel -ccn complaint -ccl go -ccv 0.1 -ccs 1 -ccp /home/ubuntu/dispute-
resolution/complaint -ccep "OR(\"Org1MSP.peer\",\"Org5MSP.peer\",\"Org6MSP.peer\",\"Org7MSP.peer\")"

./scripts/invoke-fcn.sh drchannel
./scripts/query-fcn.sh drchannel complaint
# Lines of Code (LoC)

<table>
<thead>
<tr>
<th>Sl No</th>
<th>Particular</th>
<th>LoC – Without Privacy</th>
<th>LoC - Privacy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SAP - Chaincode</td>
<td>-</td>
<td>346</td>
</tr>
<tr>
<td>2</td>
<td>Payee - Chaincode</td>
<td>291</td>
<td>376</td>
</tr>
<tr>
<td>3</td>
<td>Payment - Chaincode</td>
<td>393</td>
<td>478</td>
</tr>
<tr>
<td>4</td>
<td>Complaint - Chaincode</td>
<td>690</td>
<td>775</td>
</tr>
<tr>
<td>5</td>
<td>Helper Functions – common for every Chaincode</td>
<td>85</td>
<td>85</td>
</tr>
<tr>
<td>6</td>
<td>Encryption – common for every Chaincode, except SAP</td>
<td>-</td>
<td>117</td>
</tr>
<tr>
<td></td>
<td>Total LoC</td>
<td>1559</td>
<td>2177</td>
</tr>
</tbody>
</table>
async submitTransaction() {

    const complaint = queue.nextComplaint()
    const resolveType = helper.getRandomResolveType()
    const K1 = helper.sapKeys.K1
    const K2 = helper.sapKeys.K2

    let args = {
        contractId: 'complaint',
        contractVersion: '1.1',
        contractFunction: 'ResolveComplaint',
        contractArguments: [complaint.ID, resolveType, K1, K2],
        timeout: 60,
    };

    - label: Resolve Complaint
      txDuration: 120
      rateControl:
        type: fixed-load
        opts:
          transactionLoad: 100
      workload:
        module: benchmarks/dispute-resolution-encrypted/ResolveComplaint.js

Smart Contract

Test Iterations
# PAYMENT PROJECT

Protecting Victims of APP Frauds

## Test Report

### Caliper report

#### Summary of performance metrics

<table>
<thead>
<tr>
<th>Name</th>
<th>Succ</th>
<th>Fail</th>
<th>Send Rate (TPS)</th>
<th>Max Latency (s)</th>
<th>Min Latency (s)</th>
<th>Avg Latency (s)</th>
<th>Throughput (TPS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generate Payment Request</td>
<td>1918</td>
<td>0</td>
<td>15.6</td>
<td>9.33</td>
<td>0.43</td>
<td>4.56</td>
<td>15.5</td>
</tr>
<tr>
<td>Make Payment</td>
<td>1918</td>
<td>0</td>
<td>16.5</td>
<td>10.09</td>
<td>0.44</td>
<td>3.90</td>
<td>16.1</td>
</tr>
<tr>
<td>Generate Complaint</td>
<td>2016</td>
<td>0</td>
<td>16.6</td>
<td>8.26</td>
<td>0.41</td>
<td>4.02</td>
<td>16.3</td>
</tr>
<tr>
<td>Resolve Complaint</td>
<td>2117</td>
<td>107</td>
<td>17.7</td>
<td>7.57</td>
<td>0.26</td>
<td>3.73</td>
<td>17.7</td>
</tr>
<tr>
<td>Verify Agreement</td>
<td>18700</td>
<td>0</td>
<td>156.8</td>
<td>1.02</td>
<td>0.01</td>
<td>0.26</td>
<td>156.8</td>
</tr>
</tbody>
</table>
The end