Fair Computation
using Enclaves and Shared Ledger

Rohit Sinha, Siva Gaddam, and Ranjit Kumaresan

Open Source Enclaves Workshop 2019
Transparent Mint
Transparent Mint

Alice

Mint

Mint
## Transparent Mint

### User's TX Data

<table>
<thead>
<tr>
<th>Merchant ID</th>
<th>Date</th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>52544965</td>
<td>2014-06-03 13:37 PM</td>
<td>$23.00</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>12144989</td>
<td>2014-06-29 20:49 PM</td>
<td>$48.12</td>
</tr>
</tbody>
</table>

### Mint’s Proprietary Data

<table>
<thead>
<tr>
<th>Merchant ID</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>52544965</td>
<td>Restaurants</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>12144989</td>
<td>Gas Stations</td>
</tr>
</tbody>
</table>

Alice

Mint
Transparent Mint

<table>
<thead>
<tr>
<th>Merchant ID</th>
<th>Date</th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>52544965</td>
<td>2014-06-03 13:37 PM</td>
<td>$23.00</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>12144989</td>
<td>2014-06-29 20:49 PM</td>
<td>$48.12</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Merchant ID</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>52544965</td>
<td>Restaurants</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>12144989</td>
<td>Gas Stations</td>
</tr>
</tbody>
</table>

User’s TX Data

Mint’s Proprietary Data

Compute Provider

Report

Alice
## Transparent Mint

### User’s TX Data

<table>
<thead>
<tr>
<th>Merchant ID</th>
<th>Date</th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>52544965</td>
<td>2014-06-03 13:37 PM</td>
<td>$23.00</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>12144989</td>
<td>2014-06-29 20:49 PM</td>
<td>$48.12</td>
</tr>
</tbody>
</table>

### Mint’s Proprietary Data

<table>
<thead>
<tr>
<th>Merchant ID</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>52544965</td>
<td>Restaurants</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>12144989</td>
<td>Gas Stations</td>
</tr>
</tbody>
</table>

### Diagram

- Alice
- Compute Provider
- Mint
- Report

Alice's transactions are managed by a compute provider, Mint, which computes and reports the data.
Towards Transparency via Privacy and Fairness
Towards Transparency via Privacy and Fairness

**Privacy**: Only reveal $f(Alice\_txs, Mint\_db)$
Towards Transparency via Privacy and Fairness

**Privacy**: Only reveal \( f(Alice_{txs}, Mint_{db}) \)

**Fairness**: if anyone gets the output, then so must all honest parties
Towards Transparency via Privacy and Fairness

**Privacy**: Only reveal $f(Alice_{txs}, Mint_{db})$

**Fairness**: if anyone gets the output, then so must all honest parties

Impossible in Malicious Setting [Cleve86]
Fair Computation ➔ Fair Reconstruction
Fair Computation $\rightarrow$ Fair Reconstruction

(Unfair) Secure Computation
Fair Computation $\rightarrow$ Fair Reconstruction

(Unfair) Secure Computation

$$\text{(Unfair) Secure Computation}$$

Systems based on Intel SGX

VC3 for Map-Reduce [SCF+15]
Opaque for Spark [ZDB+17]
ObliDB [EZ17], StealthDB [GVG17] for SQL
Fair Computation → Fair Reconstruction

(Unfair) Secure Computation

Systems based on Intel SGX

- VC3 for Map-Reduce [SCF+15]
- Opaque for Spark [ZDB+17]
- ObliDB [EZ17], StealthDB [GVG17] for SQL
(Unfair) Secure Computation

Fair Reconstruction

Systems based on Intel SGX

VC3 for Map-Reduce [SCF15]
Opaque for Spark [ZDB17]
ObliDB [EZ17], StealthDB [GVG17] for SQL
Fair Computation $\rightarrow$ Fair Reconstruction

(Unfair) Secure Computation

$$\begin{align*}
\text{Systems based on Intel SGX} \\
\text{VC3 for Map-Reduce [SCF+15]} \\
\text{Opaque for Spark [ZDB+17]} \\
\text{ObliDB [EZ17], StealthDB [GVG17] for SQL}
\end{align*}$$

Fair Reconstruction

Fair n-party broadcast using $t < n$ TEE nodes and a shared ledger (corruption threshold $t$)
Fair Computation $\rightarrow$ Fair Reconstruction

(Unfair) Secure Computation

Fair Reconstruction

Systems based on Intel SGX

- VC3 for Map-Reduce [SCF+15]
- Opaque for Spark [ZDB+17]
- ObliDB [EZ17], StealthDB [GVG17] for SQL

Fair n-party broadcast using $t < n$ TEE nodes and a shared ledger (corruption threshold $t$)

[CGJ+17]: all $n$ parties need TEE
Fair Collaborative Computation
Fair Collaborative Computation

Collective Revenue Capture
Fair broadcast of model
Fair Collaborative Computation

Collective Revenue Capture
Fair broadcast of model

Crowdsourced Machine Learning
Fair exchange of data and prediction
2-Party Fair Computation: Strawman
2-Party Fair Computation: Strawman
2-Party Fair Computation: Strawman

\[
\{ \text{out} \}_k y
\]
2-Party Fair Computation: Strawman

\[ \text{Enc}(\text{pk}_\text{Alice}, k_y) \]
2-Party Fair Computation: Strawman

\[ \text{Enc}(\text{pk}_\text{Alice}, k_y) \rightarrow \sigma_{\text{Alice}} \]

\[ \{ \text{out} \}_{k_y} \rightarrow \{ \text{out} \}_{k_y} \]
2-Party Fair Computation: Strawman

\[
\text{Enc}(\text{pk}_{\text{Alice}}, k_y) \xrightarrow{\sigma_{\text{Alice}}} \text{TEE} \xleftarrow{k_y} \{ \text{out} \} \]

\[
\{ \text{out} \} \]

2-Party Fair Computation: Strawman

\[ \text{Enc}(\text{pk}_A, k_y) \]

\[ \text{out} \]

\[ \text{TEE} \]

\[ k_y \]
Bulletin Board Abstraction

[CGJ+17]
Bulletin Board Abstraction

[CGJ+17]
Bulletin Board Abstraction

• post(x) returns (idx, σ)

[CGJ+17]
Bulletin Board Abstraction

- \text{post}(x) \text{ returns } (\text{idx}, \sigma)

[CGJ+17]


**Bulletin Board Abstraction**

- post(x) returns (idx, σ)
- getContent(idx) returns (x, σ)

[CGJ+17]
Bulletin Board Abstraction

- \text{post}(x) \text{ returns } (\text{idx}, \sigma)
- \text{getContent}(\text{idx}) \text{ returns } (x, \sigma)
- \text{getHeight()} \text{ returns } (\text{idx})
2-Party Fair Computation
2-Party Fair Computation

Ledger

\{ \text{out} \}_{k_y} \quad \{ \text{out} \}_{k_y}
2-Party Fair Computation

Ledger

\{\text{out}\}_{k_y}

\sigma_{\text{Alice}}

\{\text{out}\}_{k_y}
2-Party Fair Computation

Enc(pk_Alice, k_y) || σ_Alice
2-Party Fair Computation

\[ \text{Enc}(pk_{\text{Alice}}, k_y) \parallel \sigma_{\text{Alice}} \]

\[ k_y \]

\[ \sigma_{\text{Alice}} \]

\[ \{ \text{out} \} k_y \]
2-Party Fair Computation

Encoder (pk_Alice, k_y) || σ_Alice

TEE

Ledger

out

σ

k_y

σ_Alice

out

k_y
2-Party Fair Computation

\[ \text{Enc(pk}_\text{Alice, } k_y) \parallel \sigma_{\text{Alice}} \]

\[ \sigma \]

\[ k_y \]

\[ \sigma_{\text{Alice}} \]

Ledger

\{ out \} $k_y$

\{ out \} $k_y$
n-Party Fair Broadcast: Strawman
n-Party Fair Broadcast: Strawman
n-Party Fair Broadcast: Strawman
n-Party Fair Broadcast: Strawman

\[ Enc(pk_{Alice}, k_y) \parallel \sigma_{Alice} \]
\[ Enc(pk_{BankA}, k_y) \parallel \sigma_{BankA} \]
n-Party Fair Broadcast: Strawman

Enc(pk_Alice, k_y) || σ_Alice
Enc(pk_BankA, k_y) || σ_BankA

Ledger

TEE

σ_Alice

σ_BankA
n-Party Fair Broadcast: Strawman

\[ \text{Enc}(pk_{\text{Alice}}, k_y) \parallel \sigma_{\text{Alice}} \]
\[ \text{Enc}(pk_{\text{BankA}}, k_y) \parallel \sigma_{\text{BankA}} \]
n-Party Fair Broadcast: Strawman

Enc(pk_Alice, k_y) || σ_Alice
Enc(pk_BankA, k_y) || σ_BankA

Enc(pk_BankA, k_y)
n-Party Fair Broadcast
n-Party Fair Broadcast

\[ k_y \overset{\text{out}}{=} k_{\text{Alice}} \oplus k_{\text{BankA}} \oplus k_{\text{BankB}} \]
n-Party Fair Broadcast

\[ k_y = k_{\text{Alice}} \oplus k_{\text{BankA}} \oplus k_{\text{BankB}} \]
n-Party Fair Broadcast

k_{Alice}

\{ \text{out} \} k_y \equiv k_{\text{Alice}} \oplus k_{\text{BankA}} \oplus k_{\text{BankB}}

\mathcal{E}_d

k_{\text{BankA}}

k_{\text{BankB}}
n-Party Fair Broadcast

\[ k_y \equiv k_{\text{Alice}} \oplus k_{\text{BankA}} \oplus k_{\text{BankB}} \]
n-Party Fair Broadcast

\[ \{ \text{out} \} k_y \equiv k_{\text{Alice}} \oplus k_{\text{BankA}} \oplus k_{\text{BankB}} \]

\[ E_d \]

\[ k_{\text{BankA}} \]

\[ k_{\text{BankB}} \]
n-Party Fair Broadcast

\[ k_y \equiv k_{\text{Alice}} \oplus k_{\text{BankA}} \oplus k_{\text{BankB}} \]

\[ \mathcal{E}_d \]

1. \( k_{\text{BankA}} \)
2. \( k_{\text{BankB}} \)
\[ \pi_1 = \sigma_{\text{Alice}} \parallel \sigma_{\text{BankA}} \parallel \sigma_{\text{BankB}} \]
\[ \pi_1 = \sigma_{Alice} \| \sigma_{BankA} \| \sigma_{BankB} \]

\[ k_y \equiv k_{Alice} \oplus k_{BankA} \oplus k_{BankB} \]

Diagram: n-Party Fair Broadcast
\[ \pi_1 = \sigma_{\text{Alice}} \| \sigma_{\text{BankA}} \| \sigma_{\text{BankB}} \]

\( k_y \overset{\text{out}}{=} k_{\text{Alice}} \oplus k_{\text{BankA}} \oplus k_{\text{BankB}} \)
\[ \pi_1 = \sigma_{\text{Alice}} \| \sigma_{\text{BankA}} \| \sigma_{\text{BankB}} \]
\[ \pi_2 = E_{pk_{\text{Alice}}}(k_y) \]

\[ k_y = k_{\text{Alice}} \oplus k_{\text{BankA}} \oplus k_{\text{BankB}} \]
\[ \pi_1 = \sigma_{\text{Alice}} \parallel \sigma_{\text{BankA}} \parallel \sigma_{\text{BankB}} \]
\[ \pi_2 = E_{pk_{\text{Alice}}}(k_y) \]

\text{n-Party Fair Broadcast}

\begin{align*}
\text{1} & \quad k_{\text{Alice}} \\
\text{2} & \quad \pi_1 \\
\text{3a} & \quad \sigma_1 \\
\text{3b} & \quad \pi_2 \\
\text{4a} & \quad \pi_2
\end{align*}
\( \pi_1 = \sigma_{\text{Alice}} \parallel \sigma_{\text{BankA}} \parallel \sigma_{\text{BankB}} \)
\( \pi_2 = E_{pk_\text{Alice}}(k_y) \)

\( k_y \) is broadcast to all parties.

\( k_y \equiv k_{\text{Alice}} \oplus k_{\text{BankA}} \oplus k_{\text{BankB}} \)
\[ \pi_1 = \sigma_{\text{Alice}} \| \sigma_{\text{BankA}} \| \sigma_{\text{BankB}} \]
\[ \pi_2 = E_{pk_{\text{Alice}}}(k_y) \]
n-Party Fair Broadcast

\[
\begin{align*}
\pi_1 &= \sigma_{\text{Alice}} \parallel \sigma_{\text{BankA}} \parallel \sigma_{\text{BankB}} \\
\pi_2 &= E_{pk_Alice}(k_y)
\end{align*}
\]
LucidiTEE: Policy-Compliant Fair Computing at Scale

Rohit Sinha
Visa Research
rohit.sinha@visa.com

Sivanarayana Gaddam
Visa
sgaddam@visa.com

Ranjit Kumaresan
Visa Research
rakumare@visa.com

ABSTRACT
We seek a system that provides transparency and control to users by 1) enforcing agreed-upon policies on what functions can be evaluated over private data (even when the users are offline), and 2) enforcing the set of parties with whom the results are shared. For this level of control, the system must ensure policy compliance, and we demonstrate, using modern applications, the need for history-based policies, where any decision to compute on users’ data depends on prior use of that data. Moreover, the system must algorithmically ensure fairness: if any party gets the output, then so do all honest parties. It is an open research challenge to construct a system that ensures these properties in a malicious setting.

While trusted execution environments (TEEs), such as Intel SGX and Sanctum enclaves, offer partial solutions, they are at the mercy of

We observe that several services can be modeled as a stateful computation over data from multiple parties (comprising both the end users and the service provider). Moreover, a typical service performs computation over large datasets, on behalf of a large number of users, and allows users to go offline during the computation. For such a service to provide transparency and control to all parties, we need a system that (at the very least): 1) enforces agreed-upon policies on what functions can be evaluated over the joint datasets, along with an option for any party to revoke further use of their data, and 2) enforces the set of parties with whom the results are shared. Specifically, rather than relying solely on trust or legal recourse, protocols within the system must enforce policy compliance even when the input providers go offline during the computation, and ensure fairness towards the agreed-upon set

https://eprint.iacr.org/2019/178