Brandt’s Fully Private Auction Protocol Revisited

Jannik Dreier\textsuperscript{1}, Jean-Guillaume Dumas\textsuperscript{2}, Pascal Lafourcade\textsuperscript{1}

\\textsuperscript{1}Verimag and \textsuperscript{2}Laboratoire Jean Kuntzmann (LJK), Université Grenoble 1, CNRS, FRANCE

Africacrypt, Cairo, Egypt

June 23, 2013
Challenges in e-Auctions

- Competing parties:
  - Bidders/Buyers
  - Seller

- Many possible mechanisms: English, Dutch, Sealed Bid, ...
e-Auctions: Security Requirements

Security Requirements

- Fairness
- Verifiability
- Non-Repudiation
- Non-Cancellation
- Privacy
- Receipt-Freeness
- Anonymity
- Coercion-Resistance
1. Introduction

2. Brandt’s Fully Private Auction Protocol

3. Analysis & Results

4. Conclusion
Plan

1. Introduction
2. Brandt’s Fully Private Auction Protocol
3. Analysis & Results
4. Conclusion
Protocol by Brandt [Bra06]

- Completely distributed protocol, no authorities
- Distributed homomorphic n-out-of-n threshold ElGamal encryption
- Bidders compute function \( f \) where \( f_{ij} = 1 \) if bidder \( i \) won at price \( j \), \( f_{ij} \neq 1 \) otherwise.
- Each bidder \( i \) only learns “his” \( f_{ij} \), i.e. only if he won or lost
- Zero-Knowledge Proofs (ZKP) to protect against misbehaving parties
Protocol execution
Protocol execution

1. Distributed key setup
Protocol execution

1. Distributed key setup
2. Encrypted bids
Protocol execution

1. Distributed key setup
2. Encrypted bids
3. Hom. Computation of $f_{ij}$
Protocol execution

1. Distributed key setup
2. Encrypted bids
3. Hom. Computation of $f_{ij}$
4. Partial decryption
Protocol execution

1. Distributed key setup
2. Encrypted bids
3. Hom. Computation of $f_{ij}$
4. Partial decryption
5. Shares
Protocol execution

1. Distributed key setup
2. Encrypted bids
3. Hom. Computation of $f_{ij}$
4. Partial decryption
5. Shares
6. Missing shares for $f_{ij}$
Bid encoding, example

For a public constant $Y \neq 1$:

$$b_{aj} = \begin{cases} 
Y & \text{if } j = bid_a \\
1 & \text{otherwise}
\end{cases}$$

Example: $bid_1 = 3$, $bid_2 = 1$ and $bid_3 = 2$. Then

$$b_1 = \begin{pmatrix} 
b_{1,4} \\
b_{1,3} \\
b_{1,2} \\
b_{1,1}
\end{pmatrix} = \begin{pmatrix} 
1 \\
Y \\
1 \\
1
\end{pmatrix}, 
b_2 = \begin{pmatrix} 
1 \\
1 \\
1 \\
Y
\end{pmatrix}, 
b_3 = \begin{pmatrix} 
1 \\
1 \\
Y \\
1
\end{pmatrix}$$
Definition:

\[ \tilde{f}_{ij}(X) = \left( \prod_{h=1}^{n} \prod_{d=j+1}^{k} X_{hd} \right) \cdot \left( \prod_{d=1}^{j-1} X_{id} \right) \cdot \left( \prod_{h=1}^{i-1} X_{hj} \right), \quad f_{ij} = \left( \tilde{f}_{ij}(b) \right)^{r_{i,j}} \]

Hence:

\[ b_1 = \begin{pmatrix} 1 \\ Y \\ 1 \\ 1 \\ 1 \\ Y \\ 1 \\ 1 \end{pmatrix}, \quad \tilde{f}_1(b) = \begin{pmatrix} 1^* & 1^* & 1^* & 1^* & Y^* 1^* 1^* \\ Y^* 1^* & 1^* 1^* & 1^* 1^* & 1^* 1^* Y^* 1^* 1^* \end{pmatrix} = \begin{pmatrix} Y \\ Y \end{pmatrix} \]

\[ b_2 = \begin{pmatrix} 1 \\ Y \\ 1 \\ 1 \\ 1 \\ Y \\ 1 \\ 1 \end{pmatrix}, \quad \tilde{f}_2(b) = \begin{pmatrix} 1^* & 1^* 1^* Y^* 1^* Y^2 \\ Y^* & 1^* 1^* Y^* Y \end{pmatrix} = \begin{pmatrix} Y^2 \\ Y^2 \end{pmatrix} \]

\[ b_3 = \begin{pmatrix} 1 \\ Y \\ 1 \\ 1 \\ 1 \\ Y \\ 1 \\ 1 \end{pmatrix}, \quad \tilde{f}_3(b) = \begin{pmatrix} 1^* & 1^* Y^* 1^* Y^* 1^* \end{pmatrix} = \begin{pmatrix} Y^3 \\ Y^2 \end{pmatrix} \]
Plan

1. Introduction
2. Brandt’s Fully Private Auction Protocol
3. Analysis & Results
4. Conclusion
Observation: If $r_{ij} = 1$ for all $i$ and $j$, then $f$ is injective and efficiently invertible (proof in the paper).

$r_{ij}$ is jointly chosen by the bidders.

If malleable proofs of knowledge are used, a malicious bidder can set $r_{ij} = 1$.

Allows the seller to invert $f$ and obtain all bidders’ private bids.
How to set \( r_{ij} = 1 \)

When computing

\[
\gamma^a_{ij} = \left( \tilde{f}_{ij}(\alpha) \right)^{m^a_{ij}} \quad \text{and} \quad \delta^a_{ij} = \left( \tilde{f}_{ij}(\beta) \right)^{m^a_{ij}},
\]

wait until all other bidders published their \( \gamma^a_{ij} \) and \( \delta^a_{ij} \). Submit

\[
\gamma^\omega_{ij} = \left( \tilde{f}_{ij}(\alpha) \right) \cdot \left( \prod_{k \neq \omega} \gamma^k_{ij} \right)^{-1} \quad \text{and} \quad \delta^\omega_{ij} = \left( \tilde{f}_{ij}(\beta) \right) \cdot \left( \prod_{k \neq \omega} \delta^k_{ij} \right)^{-1}.
\]

Then \( r_{ij} = \sum_a m^a_{ij} = 1 - \sum_{a \neq \omega} m^a_{ij} + \sum_{a \neq \omega} m^a_{ij} = 1. \)
Proof of Knowledge of $x$:

<table>
<thead>
<tr>
<th>Peggy</th>
<th>Victor</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Secret</strong> : $x$</td>
<td></td>
</tr>
<tr>
<td><strong>Public</strong> : $g, v = g^x$</td>
<td>$g$</td>
</tr>
<tr>
<td>$z = g^r$</td>
<td>$1 : z$</td>
</tr>
<tr>
<td>$2 : c$</td>
<td>$c$</td>
</tr>
<tr>
<td>$s = r + c \cdot x$</td>
<td>$3 : s$</td>
</tr>
<tr>
<td>Check : $g^s \equiv z \cdot v^c$</td>
<td></td>
</tr>
</tbody>
</table>
How to fake the proofs

Proof of Knowledge of $x$:

<table>
<thead>
<tr>
<th>Peggy</th>
<th>Victor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Secret : $x$</td>
<td></td>
</tr>
<tr>
<td>Public : $g, v = g^x$</td>
<td>$g$</td>
</tr>
</tbody>
</table>

$z = g^r$

1: $z$  

2: $c$

$c$

$s = r + c \cdot x$

3: $s$

$g^s = g^{r+c \cdot x} = g^r \cdot g^{x \cdot c} = z \cdot v^c$
How to fake the proofs

Proof of Knowledge of \((1 - x)\) using Proof of Knowledge of \(x\):

<table>
<thead>
<tr>
<th>Peggy</th>
<th>Mallory</th>
<th>Victor</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Secret:</strong></td>
<td>(x)</td>
<td></td>
</tr>
<tr>
<td><strong>Public:</strong></td>
<td>(g, v = g^x)</td>
<td>(g, w = g v^{-1} = g^{1-x})</td>
</tr>
<tr>
<td>(z = g^r)</td>
<td>(1: z)</td>
<td>(y = z^{-1})</td>
</tr>
<tr>
<td>(\leftarrow 2: c)</td>
<td>(c)</td>
<td>(\leftarrow 2': c)</td>
</tr>
<tr>
<td>(s = r + c \cdot x)</td>
<td>(3: s)</td>
<td>(u = c - s)</td>
</tr>
<tr>
<td><strong>Check:</strong></td>
<td>(g^s \equiv z \cdot v^c)</td>
<td>(g^u \equiv y \cdot w^c)</td>
</tr>
</tbody>
</table>
How to fake the proofs

Proof of Knowledge of \((1 - x)\) using Proof of Knowledge of \(x\):

<table>
<thead>
<tr>
<th>Peggy</th>
<th>Mallory</th>
<th>Victor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Secret :</td>
<td>(x)</td>
<td>(x)</td>
</tr>
<tr>
<td>Public :</td>
<td>(g, v = g^x)</td>
<td>(g, w = gv^{-1} = g^{1-x})</td>
</tr>
</tbody>
</table>

\[ z = g^r \quad \rightarrow \quad 1 : z \quad \rightarrow \quad y = z^{-1} \quad \rightarrow \quad 1' : y \]

\[ 2 : c \quad \leftarrow \quad c \quad \leftarrow \quad 2' : c \quad \leftarrow \quad c \]

\[ s = r + c \cdot x \quad \rightarrow \quad 3 : s \quad \rightarrow \quad u = c - s \quad \rightarrow \quad 3' : u \]

Check :

\[ g^s \equiv z \cdot v^c \quad \quad g^u \equiv y \cdot w^c \]

\[ g^u = g^{c-s} = g^{c-r-c \cdot x} = g^{-r+(1-x) \cdot c} = g^{-r} \cdot g^{(1-x) \cdot c} = y \cdot w^c \]
How to invert $f$

- Bug in the $O(nk^2)$ algorithm in the paper, corrected version in $O(n^2k^2)$ in technical report [DDL12]
- With optimizations in $O(nk)$
- Prototype implementation:

![Graph showing runtime for different algorithms across varying bids](image)
Privacy, second attack

Exploit the lack of authentication:

- Target one bidder
- Impersonate all other bidders
- Resubmit the targeted bidder’s bid as their bids
- Impersonate the seller
- Obtain winning price = targeted bidder’s bid
Verifiability:

- No authentication of the bids, hence no verification who actually submitted the bids
- \( r_{ij} = 0 \) implies \( f_{ij} = 1 \), hence several “winners” possible
- Partial decryption phase: Need to prove the use of the correct key, otherwise “nobody wins”
Other attacks

- Non-repudiation: Lack of authentication
- Fairness: An attacker can impersonate all bidders, hence controlling winner and winning price.
Countermeasures against the identified issues:

• Use of non-interactive or non-malleable zero-knowledge proofs
• Authentication of all messages
• Bidders need to prove that the value $x_a$ they use to decrypt is the same they used to generate their public key
• When computing the $\gamma^a_{ij}$ and $\delta^a_{ij}$ the bidders can check if the product is equal to one – if yes, they restart the protocol using different keys and random values
Plan

1. Introduction
2. Brandt’s Fully Private Auction Protocol
3. Analysis & Results
4. Conclusion
• Analyzed Brandt’s Fully Private Auction Protocol
• Completely distributed protocol designed for high privacy
• However: No authentication of the messages
• Attacks on Verifiability, Privacy, Fairness and Non-Repudiation
• Malleable ZKPs allow for an efficient attack on privacy
• Corner cases can lead to unexpected results, but are detectable
• Proposed four simple fixes
Thank you for your attention!

Questions?

jannik.dreier@imag.fr
Felix Brandt.  
How to obtain full privacy in auctions.  

Jannik Dreier, Jean-Guillaume Dumas, and Pascal Lafourcade.  
Attacking privacy in a fully private auction protocol.  
Let $G_q$ be a multiplicative subgroup of order $q$, prime, and $g$ a generator of the group. We consider that $i, h \in \{1, \ldots, n\}$, $j, \text{bid}_a \in \{1, \ldots, k\}$ (where $\text{bid}_a$ is the bid chosen by the bidder with index $a$), $Y \in G_q \setminus \{1\}$. More precisely, the $n$ bidders execute the following five steps of the protocol:

1. **Key Generation**
   Each bidder $a$, whose bidding price is $\text{bid}_a$ among $\{1, \ldots, k\}$ does the following:
   
   - chooses a secret $x_a \in \mathbb{Z}/q\mathbb{Z}$
   - chooses randomly $m^a_{ij}$ and $r_{aj} \in \mathbb{Z}/q\mathbb{Z}$ for each $i$ and $j$.
   - publishes $y_a = g^{x_a}$ and proves the knowledge of $y_a$’s discrete logarithm.
   - using the published $y_i$ then computes $y = \prod_{i=1}^{n} y_i$. 


1 Bid Encryption

Each bidder $a$

- sets $b_{aj} = \begin{cases} Y & \text{if } j = bid_a \\ 1 & \text{otherwise} \end{cases}$
- publishes $\alpha_{aj} = b_{aj} \cdot y^{r_{aj}}$ and $\beta_{aj} = g^{r_{aj}}$ for each $j$.
- proves that for all $j$, $\log_g(\beta_{aj})$ equals $\log_Y(\alpha_{aj})$ or $\log_Y\left(\frac{\alpha_{aj}}{Y}\right)$, and that $\log_Y\left(\prod_{j=1}^{k} \alpha_{aj}\right) = \log_g\left(\prod_{j=1}^{k} \beta_{aj}\right)$.

2 Outcome Computation

- Each bidder $a$ computes and publishes for all $i$ and $j$:
  \[
  \gamma_{ij}^a = \left( \prod_{h=1}^{n} \prod_{d=j+1}^{k} \alpha_{hd} \right) \cdot \left( \prod_{d=1}^{i-1} \alpha_{id} \right) \cdot \left( \prod_{h=1}^{i-1} \alpha_{hj} \right)^{m_{ij}}
  \]
  \[
  \delta_{ij}^a = \left( \prod_{h=1}^{n} \prod_{d=j+1}^{k} \beta_{hd} \right) \cdot \left( \prod_{d=1}^{j-1} \beta_{id} \right) \cdot \left( \prod_{h=1}^{i-1} \beta_{hj} \right)^{m_{ij}}
  \]
  and proves its correctness.
1 Outcome Decryption

- Each bidder $a$ sends $\phi_{ij}^a = (\prod_{h=1}^n \delta_{ij}^h)^{x_a}$ for each $i$ and $j$ to the seller and proves its correctness. After having received all values, the seller publishes $\phi_{ij}^h$ for all $i$, $j$, and $h \neq i$.

2 Winner determination

- Everybody can now compute $v_{aj} = \frac{\prod_{i=1}^n \gamma_{aj}^i}{\prod_{i=1}^n \phi_{aj}^i}$ for each $j$.
- If $v_{aw} = 1$ for some $w$, then the bidder $a$ wins the auction at price $p_w$. 
Timings I

Parallel Brandt with OMP on an Intel Xeon E5-4620, 32x2.2GHz

- 32 cores Brandt-16 bidders
- Sequential Winner-16 bidders
- Sequential Attack-16 bidders
- Counter Attack-16 bidders
Timings II

Parallel Brandt with OMP on an Intel Xeon E5-4620, 32x2.2GHz

- 32 cores Brandt-32 bidders
- Sequential Winner-32 bidders
- Sequential Attack-32 bidders
- Counter Attack-32 bidders
Timings III

Parallel Brandt with OMP on an Intel Xeon E5-4620, 32x2.2GHz

- 32 cores Brandt-64 bidders
- Sequential Winner-64 bidders
- Sequential Attack-64 bidders
- Counter Attack-64 bidders