Making malicious security orders of magnitude more efficient than previous semi-honest.

Tore Frederiksen\(^3\), Yehuda Lindell\(^1,2\), Valery Osheter\(^2\), Benny Pinkas\(^1\)

15 min vs. 41 sec
Introduction
Semi-honest construction
Malicious construction
Efficiency
Conclusion
INTRODUCTION – PUBLIC KEY ENCRYPTION

\[ \text{Gen}(\ell) \rightarrow sk \]

\[ pk \]

\[ m \leftarrow \text{Dec}_{sk}(c) \]

\[ c \leftarrow \text{Enc}_{pk}(m) \]
INTRODUCTION – DISTRIBUTED PKE

\[ m \leftarrow \text{Res}(m_A, m_B) \]
\[ m_A \leftarrow \text{Dec}_{sk_A}(c) \]
\[ m_B \leftarrow \text{Dec}_{sk_B}(c) \]
\[ m \leftarrow \text{Res}(m_A, m_B) \]
\[ c \leftarrow \text{Enc}_{pk}(m) \]
• *Sometimes* it can also be used for distributed signature schemes
  – Which is an end in itself
• Relevant for MPC protocols
  – CDN01, semi-homomorphic PKE
  – DPSZ12, somewhat-homomorphic PKE
• Cloud based key management
  – SEPIOR
  – UNBOUND
INTRODUCTION – RSA

• RSA:
  – Find \( \ell \) bit primes \( p \) and \( q \)
  – **Public key:** \( pq = N, e (= 3, 2^{16} + 1) \)
  – **Private key:** \( d \equiv e^{-1} \mod (p - 1)(q - 1) \)

• RSA is widely in use
  – TLS, PGP, ...

• Lots of previous work on the distributed setting
  – ..., [Gil99], [BF01], [ACS02], [DM10], [HMR+12]

• Challenging to solve efficiently
INTRODUCTION – DISTRIBUTED RSA

• Distributed RSA:
  – Find $\ell$ bit primes $p = p_A + p_B$ and $q = q_A + q_B$
  – Public key: $(p_A + p_B) \cdot (q_A + q_B) = N, e (= 3, 2^{16} + 1)$
  – Private key: $d_A + d_B \equiv e^{-1} \mod (p - 1)(q - 1)$

• Pick random $p_A, q_A, p_B, q_B$
• Do Rabin-Miller
• Repeat
• Candidate generation
  – Sampling random $p_A, q_A, p_B, q_B$ s.t. $p = p_A + p_B$ and $q = q_A + q_B$
• Construct modulus
  – Compute $N = (p_A + p_B) \cdot (q_A + q_B)$
• Verify modulus
  – Check that $N$ is the product of two primes
• Construct keys
  – Construct shares $d_A$ and $d_B$ s.t. $d \equiv e^{-1} \mod (p - 1) \cdot (q - 1)$
Candidate generation

Construct modulus

Verify modulus

Construct keys
OUTLINE

• Introduction
• Semi-honest construction
• Malicious construction
• Efficiency
• Conclusion
• \( p_A, p_B \in \mathbb{Z}_{2^{1024}} \) s.t. \( p = p_A + p_B \equiv 3 \mod 4 \)
• Trial division by small prime \( \beta \) [PS98]

\[
p_A + p_B \equiv 0 \mod \beta \\
p_A \equiv -p_B \mod \beta
\]

\( p_A \mod \beta = a \)

\( r_a \)

\( r_0, r_1, \ldots, r_{\beta-1} \in \{0,1\}^\kappa \)

\( b = -p_B \mod \beta \)

If \( r_a = r_b \) then \( p \) not prime
SEMISHONEST – CONSTRUCT MODULUS

- \((p_A + p_B) \cdot (q_A + q_B) = p_A \cdot q_A + p_b \cdot q_B + p_A \cdot q_B + p_B \cdot q_A\)
- Compute multiplication using OT [Gil99]

\[a \cdot b = c_A + c_B\]

\[m_i = (r_i + a_i \cdot b)\]

\[c_A = \sum_{i=1}^{\ell} m_i \cdot 2^i \bmod 2^i\]

\[c_B = -\sum_{i=1}^{\ell} r_i \bmod 2^{2\ell}\]

0: \(r_i \mod 2^{2\ell}\)

1: \(r_i + b \mod 2^{2\ell}\)
SEMI-HONEST – VERIFY MODULUS

- Biprimality test [BF01]

\[ \gamma \in_R \mathbb{Z}_N^* : \left( \frac{\gamma}{N} \right) = 1 \]

\[ \gamma_A = \gamma^{\frac{N+1-p_A-q_A}{4}} \mod N \]

\[ \text{If } \gamma_A \cdot \gamma^{-\frac{p_B-q_B}{4}} \equiv \pm 1 \mod N \]

Then \( \tau = \top \) else \( \tau = \bot \)

Repeat
SEMI-HONEST – CONSTRUCT KEYS

• Easy local computation [BF01]

• Compute
  – $w = N + 1 - p_A - q_A - p_B - q_B \mod e$
  – $v = w^{-1} \mod e$

• Alice outputs $d_A = \left\lfloor \frac{-v \cdot (N+1-p_A-q_A)+1}{e} \right\rfloor$

• Bob outputs $d_B = \left\lfloor \frac{-v \cdot (-p_B-q_B)}{e} \right\rfloor$
MALICIOUS – IDEA

- Allow adversary to fail good candidates
- Accepted key must be “good” without leakage

- Selective failure prevention
- Input consistency
- Correctness of biprimality
MALICIOUS – STEPS

• Selective failure prevention
  – Do OT on random, linear encoding
  – Use linearity to obtain correct product
  – Randomness ensures leakage on encoding does not leak on input

• Input consistency
  – Commitments based on AES encryption
  – Zero-knowledge of correct encryption
  – Very efficient commit-many-open-few

• Correctness of biprimality (zero-knowledge)
  – Almost standard proof-of-knowledge of discrete log
  – Few “commitments” on top to ensure composability
MALICIOUS – CONSISTENCY

- “Commitment” by encrypting using AES
- Efficient commit-many-open-few

\[ C_{K_A} = \text{Com}(K_A) \]

\[ K_A, C_{K_A} \rightarrow ZK \]

\[ C_{K_A} =? \text{Com}(K_A) \]

\[ T / \perp \]

\[ C_{p_A} = \text{AES}_{K_A}(p_A) \]
MALICIOUS – VERIFY MODULUS

\[ \gamma \in \mathbb{Z}_N^* : \left( \frac{\gamma}{N} \right) = 1 \]

\[ \gamma_A = \gamma^{\frac{N+1-p_A-q_A}{4}} \mod N \]

If \( \gamma_A \cdot \gamma^{\frac{-p_B-q_B}{4}} \equiv \pm 1 \mod N \)

Then \( \tau = \perp \) else \( \tau = \perp \)

\[ v = b \cdot (p_A - q_A) \]

\[ \gamma^v \mod N = ? \]

\[ \overline{\gamma_A} \cdot \gamma_A^b \cdot \gamma^{\frac{-b \cdot (N+1)}{4}} \mod N \]
Zero-knowledge

\[ C_{p_A} = ? \text{AES}_K(p_A) \land C_{q_A} = ? \text{AES}_K(q_A) \land \{C_t = ? \text{AES}_K(v + b \cdot (p_A + q_A))\} \]
OUTLINE

• Introduction
• Semi-honest construction
• Malicious construction
• Efficiency
• Conclusion
EFFICIENCY – IMPLEMENTATION 2048 RSA

- AES-NI for AES and PRG
- [KOS15] for OTs (seed OTs using [PVW08])
- [NP99] for 1-out-of-$\beta$ OTs
- ZK using garbled circuits using [JKO13]
- Primitives based on OpenSSL
IMPLEMENTATION – EXPERIMENTS

- Azure using multi-threaded Xeon machine
- Single-thread min 56, max 598, average 182 seconds
- 8-thread, average 41 seconds
- Best previous 15 minutes for semi-honest [HMR+12]

<table>
<thead>
<tr>
<th>Phase</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Candidate generation</td>
<td>10</td>
</tr>
<tr>
<td>Construct modulus</td>
<td>55</td>
</tr>
<tr>
<td>Verify modulus</td>
<td>6</td>
</tr>
<tr>
<td>Zero-knowledge</td>
<td>16*</td>
</tr>
<tr>
<td>Other</td>
<td>13</td>
</tr>
</tbody>
</table>
OUTLINE

• Introduction
• Semi-honest construction
• Malicious construction
• Efficiency
• Conclusion
CONCLUSION

• New protocol for malicious distributed RSA generation
  – Malicious security almost for free
  – No specific number theoretic assumptions
  – Implementation

• New efficient commit-many-open-few protocol

• Effective selective failure prevention for multiplication using OT
Thank you for your attention!

Tore Frederiksen
Cryptography Engineer
tore.frederiksen@alexandra.dk