RSA Co-Design Project

Final project guidelines
Outline

- Introduction to cryptography
- RSA
- Application (motivation)
- Field arithmetic
- Project
- Tasks
Symmetric Key Cryptography

Source: Cryptography Course (Vincent Rijmen)
Public Key Cryptography

Source: Cryptography Course (Vincent Rijmen)
RSA use case – Application scenario - Public-key cryptography

**Diffie-Hellman Key Exchange**

<table>
<thead>
<tr>
<th>Alice</th>
<th>Public Key</th>
<th>Evil Eve</th>
<th>Bob</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1:</strong> Alice generates a random number: $X_A$&lt;br&gt;$X_A = 6$ (Secret)</td>
<td></td>
<td>Evil Eve sees $G = 7$, $P = 11$</td>
<td>Bob generates a random number: $X_B$&lt;br&gt;$X_B = 9$ (Secret)</td>
</tr>
<tr>
<td><strong>Step 2:</strong> $Y_A = G^{X_A} \pmod{P}$&lt;br&gt;$Y_A = 7^6 \pmod{11}$&lt;br&gt;$Y_A = 4$</td>
<td>[Diagram]</td>
<td>$Y_B = G^{X_B} \pmod{P}$&lt;br&gt;$Y_B = 7^9 \pmod{11}$&lt;br&gt;$Y_B = 8$</td>
<td></td>
</tr>
<tr>
<td>Alice receives $Y_B = 8$ in clear-text</td>
<td>Evil Eve sees $Y_A = 4$, $Y_B = 8$</td>
<td>Bob receives $Y_A = 4$ in clear-text</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4:</strong> Secret Key = $Y_B^{X_A} \pmod{P}$&lt;br&gt;Secret Key = $8^6 \pmod{11}$&lt;br&gt;Verified Secret Key = 3</td>
<td></td>
<td>Secret Key = $Y_A^{X_B} \pmod{P}$&lt;br&gt;Secret Key = $4^9 \pmod{11}$&lt;br&gt;Verified Secret Key = 3</td>
<td></td>
</tr>
</tbody>
</table>
Factorisation records
2009: 768 bits or 232 digits

1 digit ~3.3 bits

768 bits
512 bits

Source: Cryptography Course (Vincent Rijmen)
RSA – Key generation

- Random primes $p$ and $q$
- $n = p \cdot q$ (1024 bit)
- $e = \text{random co-prime to } \varphi(n) = (p-1) \cdot (q-1)$
- $d = e^{-1} \mod \varphi(n)$
RSA

- $c = m^e \mod n$
- $m = c^d \mod n$
- Large number exponentiation
Field arithmetic

14.79 Algorithm Left-to-right binary exponentiation

INPUT: $g \in G$ and a positive integer $e = (e_t e_{t-1} \cdots e_1 e_0)_2$.
OUTPUT: $g^e$.

1. $A \leftarrow 1$.
2. For $i$ from $t$ down to 0 do the following:
   2.1 $A \leftarrow A \cdot A$.
   2.2 If $e_i = 1$, then $A \leftarrow A \cdot g$.
3. Return($A$).
Small Example – Binary Exponentiation

Example: $5 \ ^ {10}$

e = (1010)\_b

<table>
<thead>
<tr>
<th>Operation</th>
<th>Bit</th>
<th>Result (A)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assignment</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>Square (S)</td>
<td>0</td>
<td>$5^{2}$</td>
</tr>
<tr>
<td>Square (S) &amp; Multiply (M)</td>
<td>1</td>
<td>$(5^{2})^2 \times 5 = 5^5$</td>
</tr>
<tr>
<td>Square (S)</td>
<td>0</td>
<td>$(5^5)^2 = 5^{10}$</td>
</tr>
</tbody>
</table>
14.79 **Algorithm** Left-to-right binary exponentiation

INPUT: $g \in G$ and a positive integer $e = (e_t e_{t-1} \cdots e_1 e_0)_2$.
OUTPUT: $g^e$.

1. $A \leftarrow 1$.
2. For $i$ from $t$ down to 0 do the following:
   2.1 $A \leftarrow A \cdot A$.
   2.2 If $e_i = 1$, then $A \leftarrow A \cdot g$.
3. Return($A$).

14.94 **Algorithm** Montgomery exponentiation

INPUT: $m = (m_{t-1} \cdots m_0)_b$, $R = b^t$, $m' = -m^{-1} \mod b$, $e = (e_t \cdots e_0)_2$ with $e_t = 1$, and an integer $x$, $1 \leq x < m$.
OUTPUT: $x^e \mod m$.

1. $\tilde{x} \leftarrow \text{Mont}(x, R^2 \mod m)$, $A \leftarrow R \mod m$. ($R \mod m$ and $R^2 \mod m$ may be provided as inputs.)
2. For $i$ from $t$ down to 0 do the following:
   2.1 $A \leftarrow \text{Mont}(A, A)$.
   2.2 If $e_i = 1$ then $A \leftarrow \text{Mont}(A, \tilde{x})$.
3. $A \leftarrow \text{Mont}(A, 1)$.
4. Return($A$).
Project

- Implement RSA Encryption & Decryption using Hardware/Software co-design.
- New test vectors will be given.
- Decisions are all yours - what to design in software, what in hardware
- Connect hardware and software using memory-mapped and port-mapped interfaces
- Take care of synchronization aspects
Optimization goal

- Flexibility
- Area
- Time
How to measure performance?

- Time = number of clock cycles in GEZEL
- Area = number of slices on FPGA
- Flexibility?
Flexibility

- You cannot measure!
- But you can argue
  - Sell your design
- There is no right/wrong answer
  - But you need a good reasoning for design

- Be creative
  - stay reasonable
Tasks

- Implement RSA encryption and decryption
  (no key generation)
- Write a report
  - 3-5 pages text not including figures, citations, listings, ....
  - Limit listings!
- Make a presentation
  - Present your project (follow report)
  - Skip general introductions
  - 10min per group