RSA

Ramki Thurimella
Public-Key Cryptography

- Symmetric cryptography: same key is used for encryption and decryption.
- Asymmetric cryptography: different keys used for encryption and decryption.
- Public-Key cryptography: an asymmetric cryptography scheme where the key used for encryption is made public.
History of Cryptography

- 1969, GCHQ (UK equivalent of NSA?): James Ellis proves the possibility of private-key generation over an public channel, but cannot find a way to implement it.
- 1973, QCHQ: Clifford Cocks invents PKC after thinking about it overnight. Classified and goes unused.
More History

- 1974, QCHQ: Malcolm J Williamson invents what is later becomes the Diffie-Hellman key exchange algorithm.
History

- 1977, MIT: Rivest, Shamir, and Adleman, based on difficulty of factoring large numbers
Many of the mathematical mechanisms for public-key cryptography were developed prior to their publication.

However, their utility was unclear to those that developed them.

“One-way functions” and complexity were not mathematically well-understood early on.
Chinese Remainder Theorem (CRT)

- Special case of this ancient theorem (Sun Zi, circa 300 AD).
- Let $p$ and $q$ be distinct primes and $n = p \cdot q$.
- For any pair $(a, b) \in \mathbb{Z}_p \times \mathbb{Z}_q$, there is a unique number $x \in \mathbb{Z}_n$ such that
  
  $a = x \mod p$ and
  
  $b = x \mod q$
CRT Example

- Let $n = p \cdot q = 3 \cdot 5$

<table>
<thead>
<tr>
<th>$(a,b)$</th>
<th>$x$</th>
<th>$(a,b)$</th>
<th>$x$</th>
<th>$(a,b)$</th>
<th>$x$</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0, 0)</td>
<td>0</td>
<td>(1, 0)</td>
<td>10</td>
<td>(2, 0)</td>
<td>5</td>
</tr>
<tr>
<td>(0, 1)</td>
<td>6</td>
<td>(1, 1)</td>
<td>1</td>
<td>(2, 1)</td>
<td>11</td>
</tr>
<tr>
<td>(0, 2)</td>
<td>12</td>
<td>(1, 2)</td>
<td>7</td>
<td>(2, 2)</td>
<td>2</td>
</tr>
<tr>
<td>(0, 3)</td>
<td>3</td>
<td>(1, 3)</td>
<td>13</td>
<td>(2, 3)</td>
<td>8</td>
</tr>
<tr>
<td>(0, 4)</td>
<td>9</td>
<td>(1, 4)</td>
<td>4</td>
<td>(2, 4)</td>
<td>14</td>
</tr>
</tbody>
</table>
Garner’s Formula

- \( x = (((a-b)(q^{-1} \mod p)) \mod p) q + b \)
- \( q^{-1} \mod p \) — constant dependent only on \( p \) & \( q \)
- What is \( 1/5 \mod 3 \)?
- What is \( -10 \mod 3 \)?
- What is \( x \) for \( (2,4) \)?
Finding $q^{-1}$

- Use Extended Euclid’s Algorithm
- Example: $p = 89, q = 107$
- Need to solve the diophantine equation
  
  \[107x + 89y = 1\]

  \[
  \text{GCD: (89,107) = (18,89) = (17,18) = (1,17) = (0,1)}
  \]

  \[
  \text{ExtGCD: } 1 = 18 - 17 \quad \text{-- (A)}
  \]
  
  \[
  18 = 107 - 89 \quad \text{-- (B)}
  \]
  
  \[
  17 = 89 - 18 \times 4 \quad \text{-- (C)}
  \]
  
  \[
  = 89 - 4 \times (107 - 89) \quad \text{substitute for 18 from B}
  \]
  
  \[
  = 5 \times 89 - 4 \times 107 \quad \text{simplify}
  \]
  
  \[
  1 = (107 - 89) - (5 \times 89 - 4 \times 107) \]
  
  \[
  \text{substitute for 18 & 17 from A,B}
  \]
  
  \[
  = 5 \times 107 - 6 \times 89 \quad \text{simplify}
  \]

  The coefficient of 107 in the final equation is the inverse of 107 in $\mathbb{Z}_{89}^*$
Why bother with CRT?

- CRT is useful for efficient modular exponentiation:
  
  $(x^s \mod n) = (a^s \mod p, b^s \mod q) = (a^s \mod (p-1), b^s \mod (q-1))$
**Totient Function**

- The totient function, \( \varphi(n) \), is the count of numbers \( k \) less than \( n \) such that \( k \) and \( n \) are relatively prime.
  - \( \varphi(p) = p - 1 \) for \( p \) prime
  - \( \varphi(pq) = (p - 1)(q - 1) \) for \( p, q \) prime with \( p \neq q \)

  **Proof:** From the numbers 1 through \( n-1 \), subtract multiples of \( p \) and \( q \). (Since \( n \) is a product of two primes \( p \) and \( q \), all other numbers in this range would have to be relatively prime to \( n \).)

  \[
  \varphi(n) = (n-1) - \left( \frac{n}{p} - 1 \right) - \left( \frac{n}{q} - 1 \right)
  \]

  Substituting \( n = p*q \) and simplifying
  \[
  = pq + 1 - (p+q) = (p-1)(q-1) = \varphi(p) \cdot \varphi(q)
  \]

  Note that if \( p \) can be equal to \( q \), the previous claim does not hold (Can be verified with \( p=q=3 \).)

  - Euler’s Theorem: \( a^{\varphi(n)} = 1 \mod n \)
Encryption and Decryption

- Let \( p, q \) be primes
- Let \( n = p \cdot q \) (between 1024 & 4028 bits)
- Pick random \( e, 1 < e < \phi(n) \), that is relatively prime to \( \phi(n) \)
- Pick \( d \) such that \( e \cdot d = 1 \mod \phi(n) \)
- Let \( 0 \leq m < n \) be the message
Encryption and Decryption

- Let $c = m^e \mod n$ be the ciphertext.
- Then $c^d = m^{ed} = m \mod n$
  - Follows from Euler’s Theorem, since $e*d = 1 \mod \varphi(n)$.
- Make $e$ public, and keep $d$ secret
RSA Example from Wiki

1. Choose two distinct prime numbers, such as $p = 61$ and $q = 53$
2. Compute $n = pq$ giving $n = 61 \cdot 53 = 3233$
3. Compute the totient of the product as $\varphi(n) = (p - 1)(q - 1)$ giving $\varphi(3233) = (61 - 1)(53 - 1) = 3120$
4. Choose any number $1 < e < 3120$ that is coprime to $3120$. Choosing a prime number for $e$ leaves us only to check that $e$ is not a divisor of $3120$. Let $e = 17$.
5. Compute $d$, the modular multiplicative inverse of $e(\text{mod } \varphi(n))$ yielding $d = 2753$

- The public key is $(n = 3233, e = 17)$. For a padded plaintext message $m$, the encryption function is $m^{17(\text{mod } 3233)}$.
- The private key is $(n = 3233, d = 2753)$. For an encrypted ciphertext $c$, the decryption function is $c^{2753(\text{mod } 3233)}$. 
Why RSA works

- “Easy” to compute $m^e \mod n$ and $c^d \mod n$
- “Hard” to determine $d$, even given $e$ and $n$!
- We *think* of $f(x) = x^e \mod n$ as a one-way function.
Why RSA works

- Factoring $n$ is the same thing as revealing $d$: Let $\langle n,e \rangle$ be an RSA public key. Given the private key $d$, one can efficiently factor of $n$, and given the factorization of $n$, one can efficiently determine $d$. 
Attacks on RSA

- Elementary Attacks
  - common modulus
  - blinding
- Attacks on the actual RSA cryptosystem
  - low private exponent
  - low public exponent
- Attacks of implementations of the RSA cryptosystem
  - timing attacks
Elementary Attacks

- Poor configuration: A different $n$ must be used for all users in a system, since if you know your own public, private key pair you can factor $n$

- Blinding: can fool principals into providing signatures on any message $m$ by multiplying by $r^e$, where $r$ is random
Low Private Exponent

- Wiener developed an attack that works effectively when $d$ is sufficiently small.
- Let $n = pq$ with $q < p < 2q$
- Let $d < (1/3) n^{1/4}$
- Given $\langle n, e \rangle$ where $ed = 1 \mod \varphi(n)$, then Eve can efficiently recover $d$
Low Private Exponent

- The details of this are involved. But there other ways of providing faster decryption, in particular the CRT method. (currently used by OpenSSL for modular exponentiation)
To reduce encryption time, use a small $e$. This is not known to lead to any total breaks of RSA. Most attacks use Coppersmith’s Theorem.

Let $n$ an integer & $f$ a monic poly of degree $d$. Let $X = n^{(1/d)-\varepsilon}$. Then there is an efficient way to determine all $|x_0| < X$ such that $f(x_0) = 0 \mod n$
Franklin-Reiter: if $e$ is low and Eve has ciphertexts for $m$ and $f(m)$ for some publicly-known polynomial $f$, then there is an attack to recover $m$ and $f(m)$. 
More Public Exponents

- Coppersmith’s Short Pad Attack: Alice sends a message to Bob that has been randomly padded on the end to it. Eve intercepts it and prevents it from reaching its destination. Alice sends the same message to Bob again with a different random padding on the end. Eve can determine $m$ given these two ciphertexts.
Small Public Exponents

- Partial Key Exposure: if \( e < \sqrt{n} \), then Eve can determine all of \( d \) from knowing a percentage of the bits of \( d \).

- Theorem: If \( n \) is \( b \) bits, then you only need \( b/4 \) bits of \( d \) to reveal all of \( d \). (holds for all \( e \))

- When \( e = 3 \), the cryptosystem leaks half of the bits of \( d \)!
Implementation Attacks

- Common method of modular exponentiation: repeated squaring.

\[ z = m \]
\[ c = 1 \]
\[ \text{for } i = 0 \text{ to } n \]
\[ \text{if } d_i = 1 \text{ then } c = c \times z \mod n \]
\[ z = z^2 \mod n \]
\[ \text{end} \]

- At the end, \( c = m^d \mod n \)
Timing Attacks

- To determine $d$, Eve generates a large number of random messages and observes how long it takes to compute their ciphertexts.
- By observing how much time it takes to compute everything combined with the physical specifications of the computational device it is possible to determine the bits of $d$. 
Timing Attacks

- Once a quarter of the bits of d have been discovered, Eve can factor n
- Similar attacks exist for other implementations of modular exponentiation. Implementing blinding prevents this attack since Eve no longer knows what message is being decrypted
Conclusion

- RSA seems to be a secure public-key cryptosystem, despite our best efforts.
- Still no proof that it is theoretically safe.
- Most attacks on RSA come from poor configuration or bad implementations.