COMP 3704 Computer Security

Christian Grothoff
christian@grothoff.org

http://grothoff.org/christian/
RSA

Pick $p, q$ prime and $e$ such that

$$GCD((p - 1)(q - 1), e) = 1 \quad (1)$$

- Define $n = pq$,
- compute $d$ such that $ed \equiv 1 \bmod (p - 1)(q - 1)$,
- Let $c \equiv m^e \bmod n$,
- then $m = c^d \bmod n$!
Proof

\[ c^d \equiv (m^e)^d \mod n \]  \hspace{1cm} (2)

\[ \equiv m^{ed} \mod n \]  \hspace{1cm} (3)

\[ \equiv m^{k(p-1)(q-1)+1} \mod n \]  \hspace{1cm} (4)

\[ \equiv mm^{k(p-1)(q-1)} \mod n \]  \hspace{1cm} (5)

\[ \equiv m \mod n \]  \hspace{1cm} (6)
RSA Summary

- Public key: $n, e$

- Private key: $d = e^{-1} \mod \phi(n)$ where $\phi(n) = (p - 1) \cdot (q - 1)$

- Encryption: $c = m^e \mod n$

- Decryption: $m = c^d \mod n$
RSA Facts

- \( D_{A_{\text{priv}}} \left( D_{B_{\text{priv}}} \left( E_{A_{\text{pub}}} \left( E_{B_{\text{pub}}} (M) \right) \right) \right) = M \)

- \( e \) is usually small prime \((3, 17, 65537)\)

\( \Rightarrow \) Encryption (significantly) faster than decryption!

\( \Rightarrow \) Signature verification (significantly) faster than signing!
Chinese Remainder Theorem

Let \( n = \prod_{i=1}^{t} p_i \) where \( p_i \) prime, \( p_i \neq p_j \) for \( i \neq j \). Then the system of equations (for \( i \in \{1, \ldots, t\} \))

\[
x = a_i \mod p_i
\]  

has a unique solution \( x \mod n \).
Chinese Remainder Theorem and RSA

Suppose we kept $p$ and $q$ and calculated $u = q^{p-1} \mod n$ and $v = p^{q-1} \mod n$. Then we can compute $m = c^d \mod n$ using:

$$m_1 = c^d \mod (p-1) \mod p$$  \hspace{1cm} (8)

$$m_2 = c^d \mod (q-1) \mod q$$  \hspace{1cm} (9)

$$m = m_1 \cdot u + m_2 \cdot v.$$  \hspace{1cm} (10)
Re-using the Primes

Can we re-use $pq = n$ with a different $e$ to generate a second key pair? Suppose we have $(d_1, e_1)$ and $(d_2, e_2)$ and encrypt the same message $m$:

$$c_1 = m^{e_1} \mod n \quad (11)$$
$$c_2 = m^{e_2} \mod n \quad (12)$$

Can the adversary recover $m$?
Common Modulus Attack

Given \( n, e_1, e_2, c_1 \) and \( c_2 \) the adversary can compute \( r < 0 \) and \( s \) such that:

\[
re_1 + se_2 = 1 \tag{13}
\]

Use again the extended Euclidean algorithm to compute \( c_1^{-1} \mod n \). Finally:

\[
(c_1^{-1})^{-r} \cdot c_2^s \equiv m \mod n \tag{14}
\]
Low Encryption Exponent Attack

- $e$ is known
- $M$ maybe small
- $C = M^e < n$?
- If so, can compute $M = \sqrt[n]{C}$

$\Rightarrow$ Small $e$ can be bad!
Padding and RSA Symmetry

• Padding can be used to avoid low exponent issues (and issues with \( m = 0 \) or \( m = 1 \))

• Randomized padding defeats chosen plaintext attacks (dictionary!)

• Padding breaks RSA symmetry:

\[
D_{A_{priv}}(D_{B_{priv}}(E_{A_{pub}}(E_{B_{pub}}(M)))) \neq M
\]

(15)

• PKCS#1 / RFC 3447 define a padding standard
ElGamal Signatures

- Calculate $y = g^x \mod p$ for $p$ prime. $x$ is the private key.
- Select $k$ such that $GCD(k, p-1) = 1$, compute $a = g^k \mod p$.
- Solve $M = (xa + kb) \mod (p - 1)$ using the extended Euclidian algorithm.
- Signature is $(a, b)$. Verified using $y^a a^b \mod p = g^M \mod p$. 
Proof

\[ y^a a^b \equiv g^{ax} g^{kb} \mod p \quad (16) \]
\[ \equiv g^{ax+kb} \mod p \quad (17) \]
\[ \equiv g^{M+(p-1)\cdot t} \mod p \quad (18) \]
\[ \equiv g^M \cdot (g^{p-1})^t \mod p \quad (19) \]
\[ \equiv g^M \mod p \quad (20) \]
Diffie-Hellman Key Exchange

Generator $g$ and prime $p$ are known to everyone.

1. Alice calculates $a \equiv g^x \mod p$ for random number $x$, sends $a$ to Bob.

2. Bob calculates $b \equiv g^y \mod p$ for random number $y$, sends $b$ to Alice.

3. Alice computes $K = b^x$.

4. Bob computes $K = a^y$. 
ElGamal Encryption

• Calculate $y = g^x \mod p$ for $p$ prime. $x$ is private key.

• Select $k$ such that $GCD(k, p - 1) = 1$, compute $a = g^k \mod p$.

• Calculate $a = g^k \mod p$ and $b = y^k M \mod p$, $C = (a, b)$

• Decrypt using $M = b/a^x \mod p$

• Really just Diffie-Hellman
Questions
Assignment

Implement RSA using libgmp.

Research PKCS#1 block type 2 padding.\(^1\)

\(^1\)A good starting point is the source of libgcrypt.