Applied Cryptography and Computer Security
CSE 664 Spring 2020

Lecture 14: Security of RSA

Department of Computer Science and Engineering
University at Buffalo

Summary of RSA

- Key generation
  - choose prime p, q, and e; set $n = pq$
  - public key is $pk = (e, n)$
  - private key is $sk = d$, where $d \equiv e^{-1} \pmod{\phi(n)}$
- Encryption
  - given a message $0 < m < n$
  - encrypt as $c = E_{pk}(m) = m^e \bmod n$
- Decryption
  - given ciphertext c
  - decrypt as $m = D_{sk}(c) = c^d \bmod n$

Security of RSA

- The security of the RSA encryption schemes depends on the hardness of the RSA problem
- The RSA problem is widely believed to be computationally equivalent to factoring, but no proof is known
- Knowledge of the following is equivalent with public key (n, e), i.e., enables decryption
  - factors p and q
  - $\phi(n)$
  - private key d

Security of RSA

- Knowledge of n and $\phi(n)$ implies knowledge of factors p and q
  - given n and $\phi(n)$, we can compute

$$\phi(n) = (p-1)(q-1) = n - p - q + 1 = n - p - n/p + 1$$
$$p\phi(n) = np - p^2 - n + p$$

then
$$p^2 - np + \phi(n)p - p + n = 0$$
$$p^2 - (n - \phi(n) + 1)p + n = 0$$

  - the above equation has two solutions: p and q

Factoring Large Numbers

- For factoring a product of two primes, most effective algorithms are
  - quadratic sieve
  - number field sieve
  - elliptic curve factoring algorithm
- The best factoring algorithms run in sub-exponential time
- Hardness of factoring
  - 512-bit modulus has been factored in 1999
  - 768-bit modulus has been factored in 2009
  - 829-bit modulus has been factored in 2020
  - 1024-bit modulus may be factored soon

Security of RSA

- Plain RSA is very weak
- Attacks on plain RSA
  - short messages
  - brute force search
  - common modulus
  - small exponents e and d
  - timing attacks
- Improving security of RSA

Attacks on RSA

- Encrypting short messages with small e
  - often e can be very small such as 3
  - suppose that we encrypt a message $m < n^{1/3}$ and transmit ciphertext $c = m^e \bmod n$
    - any m can be encoded as an element of $\mathbb{Z}_n$ by treating it as integer and padding with 0s on the left
  - no modular reduction takes place and $m = c^{1/3}$ over integers can be easily computed
  - now how about encrypting 128-bit symmetric encryption key with a $>$ 1024-bit modulus?

Attacks on RSA

- Brute force key search
  - try possible keys hoping to find the correct one
  - infeasible to succeed unlike in case of symmetric encryption
- Brute force message search
  - the message space can be bounded by some value L
  - simply encrypt all messages
    - the encryption algorithm is public
  - when we see ciphertext c, simply compare it to our ciphertexts
  - attack takes time/space linear in L

Attacks on RSA

- Brute force message search
  - unfortunately, a much faster attack is known that runs in about $\sqrt{L}$ time
  - implications: decrypting a 128-bit key takes 2^{64} steps
  - algorithm:
    - we are given $c = \mathsf{Enc}(m) = m^e \bmod n$ for some $m < 2^\ell$
    - set $T = 2^{\alpha\ell}$ for $1/2 < \alpha < 1$
    - for $r = 1, \ldots, T$, set $x_r = c/r^e \bmod n$
    - sort the pairs (r, x_r) by the second value
    - for $s = 1, \ldots, T$, if $s^e \bmod n = x_r$ for some r, output $r \cdot s \bmod n$

Attacks on RSA

- Small encryption exponent e
  - suppose that the same $e = 3$ is used with different moduli (i.e., with different public keys)
  - suppose Alice wants to send the same message m to three different people using their moduli n_1, n_2, and n_3
  - she sends $c_i = m^3 \bmod n_i$ for $i = 1, 2, 3$
  - an eavesdropper Eve observes c_1, c_2, and c_3
  - Eve can use the Chinese Remainder Theorem to find a solution x $(0 < x < n_1 n_2 n_3)$ to the three congruences $x \equiv c_i \pmod{n_i}$
  - the solution is $x = m^3$ and m can be recovered by computing $\sqrt[3]{x}$ over integers (since $m < \min(n_1, n_2, n_3)$)

Attacks on RSA

- Common modulus attack
  - this attack deals with a common misuse of RSA
  - suppose for efficiency reasons a trusted party generates one modulus n and several key pairs (e_1, d_1), (e_2, d_2), $\ldots$
  - then each user has $pk_i = (e_i, n)$ and $sk_i = d_i$
  - this setup is trivially insecure
    - why?

Attacks on RSA

- Common modulus attack
  - now suppose that it is all right that all users know each others' keys
  - suppose Eve sees two ciphertexts that encrypt the same message

$$c_1 = m^{e_1} \bmod n \text{ and } c_2 = m^{e_2} \bmod n$$

  - $e_1 \neq e_2$ and it is likely that $gcd(e_1, e_2) = 1$
  - then Eve can use Extended Euclidean algorithm to compute x and y such that $e_1 x + e_2 y = 1$
  - Eve computes $c_1^x \cdot c_2^y \bmod n$ to recover m

Attacks on RSA

- Small decryption exponent d
  - the secret key d cannot be small
  - if $|d| \approx 1/4|n|$, there is an efficient algorithm for recovering d from public information (e, n)
  - thus, d should have roughly the same size as n
- Factors p and q close to each other
  - p and q cannot be chosen to be close to $\sqrt{n}$
  - if p and q are within a feasible comptutational effort from $\sqrt{n}$, a brute force search can find the factors

Attacks on RSA

- Timing attacks
  - measure decryption time hoping to recover the decryption key
  - exponentiation algorithm and the ciphertext are known
  - what we can do to prevent such attacks
    - use constant exponentiation time
    - add random delays
    - modify the values used in calculations by blinding

Attacks on RSA

- Are timing attacks practical?
  - the answer is yes
  - OpenSSL was discovered to be vulnerable in 2003
    - researchers discovered a remote timing attack on OpenSSL implementations that allowed to learn RSA keys
    - to secure it, turn RSA blinding on

Security of RSA

- Let's get back to security of RSA
  - RSA is not secure because it is deterministic
  - RSA leaks information
- To achieve security in the sense of indistinguishability, randomization and expansion are necessary
  - now the ciphertext will be longer than the message
  - suppose we want the computation effort of breaking the indistinguishability to be 2^k
  - the ciphertext must be at least k bits longer than the message

Security of RSA

- Simple padding scheme
  - idea: pad message m with random r and encrypt their concatenation
  - let $(n, e, d) \leftarrow \mathsf{GenRSA}(1^k)$ with $|n| = k$
  - let $|m| = \ell(k) \leq k - 1$
  - Gen: run $(n, e, d) \leftarrow \mathsf{GenRSA}(1^k)$ and output $pk = (n, e)$ and $sk = (n, d)$
  - Enc: given $m \in \{0, 1\}^{\ell(k)}$, choose random $r \leftarrow \{0, 1\}^{k-\ell(k)-1}$ and output

$$c = \mathsf{Enc}_{pk}(m) = (r||m)^e \bmod n$$

  - Dec: given $c \in \mathbb{Z}_n^*$, compute $m' = c^d \bmod n$ and output $\ell(k)$ least significant bits of it

Security of RSA

- Security of this simple padding scheme
  - if $|r|$ is not large enough, the scheme is not CPA-secure
    - e.g., $|r| = O(\log k)$
  - if $\ell(k) = c \cdot k$ for constant $c < 1$, the scheme can be conjectured secure
    - no proof based on the standard RSA assumption is known
  - if $\ell(k) = O(\log k)$, the scheme has been proven to be CPA-secure under the RSA assumption
    - is it satisfactory?

Security of RSA

- PKCS #1 v1.5
  - it is a widely used and standardized version of RSA from RSA Laboratories
  - it uses the above idea and requires $|r|$ to be at least 8 bytes
  - $\ell(n)$ is at most $k/8 - 11$ bytes
  - $|r|$ is $k/8 - \ell - 3$ bytes
  - encryption is formed as

$$c = (00000000||00000010||r||00000000||m)^e \bmod n$$

  - no byte of r is allowed to be 0
  - the construction is believed to be CPA-secure, but no formal proof under the RSA assumption is known

Towards Higher Security Standards

- An example padding for achieving CPA-security
  - we are given an encryption scheme $\mathcal{E} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ and a cryptographic hash function h
  - to encrypt message m, generate a random number r
  - compute the ciphertext (c_1, c_2) as

$$c_1 = \mathsf{Enc}_k(r) \text{ and } c_2 = h(r) \oplus m$$

  - to decrypt a message given (c_1, c_2), compute $h(\mathsf{Dec}_k(c_1)) \oplus c_2$

Semantic Security of RSA

- This padding scheme with RSA:
  - public key is (e, n) with 1536-bit modulus
  - encryption of m is $(r^e \bmod n, h(r) \oplus m)$ for a random $r \in \mathbb{Z}_n^*$
  - to decrypt a ciphertext (c_1, c_2), compute $m = h(c_1^d \bmod n) \oplus c_2$
  - for 256-bit messages, the size of ciphertexts is $1536 + 256$
- Why is this solution secure?
  - it relies on randomness of h and one-way nature of Enc_k
  - to learn something about m from $h(r) \oplus m$, one has to know $h(r)$
  - but since $h(r)$ is random, you cannot recover r
  - recovering r from $\mathsf{Enc}_k(r)$ is also infeasible

OAEP

- In 1994 Bellare and Rogaway proposed an optimal asymmetric encryption padding (OAEP) method for encoding messages
  - it is the basis of PKCS #1 v2.0 and later
  - it uses encryption $\mathcal{E} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ (formally modeled as one-way trapdoor permutation)
  - it also uses two hash functions $h : \{0,1\}^\ell \to \{0,1\}^t$ and $g : \{0,1\}^t \to \{0,1\}^\ell$

OAEP

- OAEP algorithms
  - to encrypt an ℓ-bit message m:
    - choose an t-bit random r
    - compute the ciphertext as

$$\mathsf{Enc}_k(m \oplus g(r) || r \oplus h(m \oplus g(r)))$$

  - to decrypt ciphertext c:
    - after applying Dec_k to c, parse the content in two parts $c_1 || c_2 \leftarrow \mathsf{Dec}_k(c)$
    - to recover m from $m \oplus g(r)$, we need to find r as $c_2 \oplus h(c_1)$
    - finally, set $m = c_1 \oplus g(r)$

OAEP

- Security of OAEP
  - if h and g are modeled as random oracles and the RSA problem is hard
  - RSA-OAEP is proven to be CCA-secure for certain types of public exponents e (including common $e = 3$)
  - OAEP is designed in such a way that the only way to find m is to explicitly choose m and r and try them

OAEP

- Size of parameters in OAEP
  - ciphertext has size k (e.g., 1536 for RSA)
  - t should be such that 2^t work is infeasible and success is negligible
    - e.g., $t = O(k)$
  - the plaintext size ℓ can be up to $k - t$
  - e.g., with $k = 1536$ and $t = 128$, message size is up to 1408 bits
  - expansion is optimal

Summary

- Security of RSA
  - many attacks have been discovered over the year
  - attacks on plain RSA can be very damaging
  - countermeasures for implementation-based attacks exist
- CPA-security of RSA
  - can be added by using padding
  - OAEP achieves an optimal expansion and is provably secure