프로젝트 #4 (개인)

소프트웨어학부 암호학

2022년 10월 6일

문제

키의 길이가 64 비트인 미니 RSA 알고리즘을 구현한다. 공개키 암호방식의 국제표준 알고리즘인 RSA는 현재 키의 길이가 최소 2048 비트가 되어야 안전하다. 이 과제에서는 실세계에서 활용하기에는 안전하지 않지만 RSA의 기본 원리를 이해하기에 충분하고, 구현이 까다롭지 않은 미니 RSA를 선택하였다.

RSA 키의 길이

RSA 키의 길이는 RSA 모듈러스 n의 길이를 의미한다. 구체적으로는 n 값을 이진수로 표현하기 위한 최소 비트의 수를 뜻한다. 예를 들어, n이 15이면 4 비트로 15를 표현할 수 있으므로 길이가 4 비트가 된다. 따라서 RSA 키의 길이가 64 비트이면 n 값이 $2^{63} \le n < 2^{64}$ 사이에 있어야 한다는 뜻이다.

카마이클 함수 $\lambda(n)$

표준시스템에서는 RSA 키를 생성할 때 오일러 함수 $\phi(n)$ 대신에 계산양을 줄여주는 카마이클 함수 $\lambda(n)$을 사용한다. 미니 RSA도 $\lambda(n)$을 사용하여 키를 생성한다. $\lambda(n)$은 다음과 같이 정의된다.

$$\lambda(n) = \mathrm{lcm}(p-1, q-1) = \frac{(p-1)(q-1)}{\gcd(p-1, q-1)}$$

함수 구현

외부에서 보이는 전역 함수를 아래 열거한 프로토타입을 사용하여 구현한다. 각 함수에 대한 요구사항은 다음과 같다.

- `void mRSA_generate_key(uint64_t *e, uint64_t *d, uint64_t *n)` – 길이가 32 비트 내외인 임의의 두 소수 p와 q를 생성한 다음, 키의 길이가 64 바이트인 RSA 공개키 (e, n)과 개인키 (d, n)을 생성한다. RSA 모듈러스 n 값은 $2^{63} \le n < 2^{64}$을 만족해야 한다. 두 소수 p와 q의 길이가 비슷할수록 더 안전하다는 점을 참고한다.
- `int mRSA_cipher(uint64_t *m, uint64_t k, uint64_t n)` – $m \leftarrow m^k \bmod n$을 계산한다. 계산 중 오류가 발생하면 0이 아닌 값을 넘겨주고, 없으면 0을 넘겨준다. $m \ge n$이면 m이 값의 범위를 넘었으므로 오류로 처리해야 한다.

지역 함수

내부에서만 사용하는 지역 함수는 지난 과제에서 구현한 것을 각자 필요에 맞게 사용한다. 다음에 열거한 것은 이번 과제에 필요한 함수 목록이다.

- `static uint64_t gcd(uint64_t a, uint64_t b);`
- `static uint64_t umul_inv(uint64_t a, uint64_t m);`
- `static uint64_t mod_add(uint64_t a, uint64_t b, uint64_t m);`

- `static uint64_t mod_mul(uint64_t a, uint64_t b, uint64_t m);`
- `static uint64_t mod_pow(uint64_t a, uint64_t b, uint64_t m);`
- `static int miller_rabin(uint64_t n);`

arc4random 함수

arc4random 계열의 함수는 암호학적으로 안전한 의사난수를 생성하기 위해 개발되었다. 기존 라이브러리에 있는 `rand()` 함수는 안전을 고려하지 않고 설계되었기 때문에 과제에서는 사용하지 않는다. 많이 쓰이는 몇 가지 함수에 대한 용법은 아래와 같다.

- `uint32_t arc4random(void)` – 32비트 난수를 생성하여 넘겨준다.
- `void arc4random_buf(void *buf, size_t nbytes)` – 크기가 `nbytes`인 난수를 생성하여 `buf`에 담아 넘겨준다.
- `uint32_t arc4random_uniform(uint32_t upper_bound)` – 0과 `upper_bound-1` 사이의 32비트 난수를 넘겨준다.

골격 파일

구현이 필요한 골격파일 `mRSA.skeleton.c`와 함께 헤더파일 `mRSA.h`, 프로그램을 검증할 수 있는 `test.c`, 그리고 `Makefile`을 제공한다. 이 가운데 `test.c`를 제외한 나머지 파일은 용도에 맞게 자유롭게 수정할 수 있다.

제출물

과제에서 요구하는 함수가 잘 설계되고 구현되었다는 것을 보여주는 자료를 보고서 형식으로 작성한 후 PDF로 변환하여 `이름_학번_PROJ4.pdf`로 제출한다. 여기에는 다음과 같은 것이 반드시 포함되어야 한다.

- 본인이 작성한 함수에 대한 설명
- 컴파일 과정을 보여주는 화면 캡처
- 실행 결과물의 주요 장면과 그에 대한 설명, 소감, 문제점
- 프로그램 소스파일 (`mRSA.c`, `mRSA.h`) 별도 제출
- 프로그램 실행 결과 (`mRSA.txt`) 별도 제출

평가

- Correctness 50%: 프로그램이 올바르게 동작하는 지를 보는 것입니다. 여기에는 컴파일 과정은 물론, 과제가 요구하는 기능이 문제없이 잘 작동한다는 것을 보여주어야 합니다.
- Presentation 50%: 자신의 생각과 작성한 프로그램을 다른 사람이 쉽게 이해할 수 있도록 프로그램 내에 적절한 주석을 다는 행위와 같이 자신의 결과를 잘 표현하는 것입니다. 뿐만 아니라, 프로그램의 가독성, 효율성, 확장성, 일관성, 모듈화 등도 여기에 해당합니다. 이 부분은 상당히 주관적이지만 그러면서도 중요한 부분입니다. 컴퓨터과학에서 중요하게 생각하는 best coding practices를 참조하기 바랍니다.

HK