

눈이 편안한 SSL 세미나

leeopop

대칭키 암호? 비대칭키 암호?

ABC 에다가 1씩 더해서 BCD를 만듭니다. = 키는 1
BCD 에다가 1씩 빼서 ABC를 만듭니다. = 키는 1

ABC에다가 $f(123)$ 을 취하여 AQD를 만듭니다. = 키는 123
AQD에다가 $f^{-1}(423)$ 을 취하여 ABC를 만듭니다. = 키는 423

저 신비한 f 함수에 수학적 원리가 담겨 있습니다.
(**123**과 **423**은 서로 **알기 힘든 수학적 관련성**이 있습니다)

다음 수학 개념이 필요합니다

- 법(modulo)
- 법에 대한 곱셈
- 법에 대한 곱셈의 역원

**법 : 예를 들어 100에 대한 법에서는
32 는 32, 132 도 32, -32 는 68**

8과 9를 곱하고 5으로 나눈 나머지는??
72를 5로 나눈 나머지 == 2

8을 5로 나눈 나머지 3과
9를 5로 나눈 나머지 4를 각각 곱하고
그 결과인 12를 5로 나눈 나머지 == 2

마찬가지로
 8^{16} 을 5로 나눈 나머지는
 8^8 을 5로 나눈 나머지의 제곱

뭐 이런식으로

법에 대한 곱셈의 역원

역원을 알려면 => 우선 항등원을 알아야제!

항등원은 그냥 1입니다...

$$5 * 1 \text{ mod } 3 = 5 \text{ mod } 3$$

자, 5의 7에 대한 역원은 무엇일까요??

$$5 * 3 = 15 = 14 + 1 = 1 \text{ mod } 7$$

오예

일일이 해 봐야 구할 수 있나요??

아뇨; **확장된 유클리드 호제법**으로 구할 수 있습니다.

뭐... 자세한건 넘어가고 어쨌든 잘 구할 수 있어요;

어디에 쓰냐고요??

$X * 5 = 6 \pmod{7}$ 입니다.
X는 무엇일까요??

양변에 5의 역원인 3을 곱해주면?

$$X * 5 * 3 = 6 * 3 \pmod{7}$$

$$\mathbf{X = 4 \pmod{7}}$$

다음 용어를 설명할 것입니다.

- 서명
- 해시

서명이란?

- 나만이 생성해 낼 수 있는 글자, 그림
- => 나만이 생성해 낼 수 있는 숫자
- 나는 3에는 삼, 6에는 육 이렇게 대답할 수 있고 이를 들은 다른 사람도 삼, 육을 따라 할 수 있지만,
- 임의로 주어진 17을 말해보라고 하면 따라 할 수 없다. => 전자서명

자 그림 실습!

- $Y = f(x)$ 를 아무거나 생각해 보세요
- $2x, 5x+2$ 등등
- 단, 100을 절대로 넘어서는 안됩니다.

해시란?

- 임의의 길이의 데이터를 나타내는 정해진 길이의 짧은 '지문'
- 어쩌다가 지문이 같은 데이터가 있을 수는 있지만, 지문이 다르면 데이터는 무조건 다르다.

고정된 길이의

누구나 생성할 수 있는 서명

예제

- 홀수면1, 짝수면 0
- 각 자리수의 합
- 각 자리수의 곱
- 기타 등등등

공개키 암호시스템의 개요

- 공개키 한 쌍이 있다(A,B)
 - 하나가 공개키면 다른 하나는 비밀키

평문 M을 A로 암호화 한 내용은 B로 열리고
평문 M을 B로 암호화 한 내용은 A로 열린다

(RSA의 특징, 전자서명에 사용)

어떻게??

합성수 M 을 A 로 나누면 B 가 된다

합성수 M 을 B 로 나누면 A 가 된다

뭐 대충 이런 느낌

소수 두 개를 준비합니다.

13과 17

두 수를 서로 곱합니다

13 곱하기 17 = 221

이게 공개키입니다(mod 221)

두 수에서 1을 뺀 수를 서로 곱합니다.

12 곱하기 16 = 192

공개키와 서로 소인 하나의 수를 고릅니다(7이 적절)

192에 대한 7의 곱셈의 역원을 구합니다 = 7 곱하기 55 = 192 곱하기 2 더하기 1

이 역원이 바로 비밀키

13*17 보다 작은 임의의 정수를 암호화 할 수 있습니다.

임의의 정수 m 의 7 제곱(아까 정한 적절한 서로 소인 수)을 하고 Mod 221을 해 주면 암호화가 완료됩니다!

암호화 된 수를 다시 비밀키 제곱 하고 mod 221을 해 주면 복호화가 완료됩니다!

해 봅시다.

공개키 암호 시스템의 개요

1. 서버가 암호화를 할 수 있는 "공개키"를 던집니다. (해커도 수신 가능)
2. 여러 사람들이 팬레터를 서버에게로 암호화를 해서 보냅니다.
(해커도 수신 가능)
3. 근데 그 내용을 알려면 공개키가 무슨 소수들로 이루어져 있는지 알아야 하는데 서버 말고는 아무도 모릅니다.
4. 해커는 사용자가 서버로 보낸 내용을 알 수 없습니다.

요는

"네가 이 수를 소인수분해할 수 있다면 뭐든지 해봐라" 입니다.

공개키 서명의 개요

1. "얘들아, 두 소수를 곱한 이 숫자가 내 서명이야~"
2. 음, 그걸 어떻게 믿지? 진짜 네 서명 맞아?
3. "그럼! 이 수를 이루는 두 소수는 나만 알거덩~"
4. 어디 테스트를 해 보자, 16을 암호화해봐
5. "응, 여기 71231234"
(이 결과를 해커가 들을 수는 있지만,
임의의 모든 응답에 대답할 수는 없습니다.)
6. 옷, 공개키로 복호화를 하니깐 딱 16이 나오네?
정말로 네가 비밀키를 가지고 있구나!

SSL의 개요

공개키로 암호화 한 정보는 서버를 제외한 누구도 볼 수 없습니다.

=> **비밀**

비밀키로 암호화 한 정보는 누구나 볼 수 있습니다.

=> **인증, 서명**

“우리가 앞으로 쓸 암호는 1234 이다” 라는 말을 서버에게 전달하고 싶은데
이 말 자체가 도청될까봐 걱정이에요

⇒ 이 말을 공개키로 암호화해서 서버에게 전송함

⇒ 이후 이들간의 통신은 1234 라는 키를 안전하게 공유하여 통신

SSL을 써야만 하는 이유

인터넷 공간으로 떠도는 정보는 누구나 볼 수 있습니다.
인터넷 공간으로 떠도는 정보는 누구나 볼 수 있습니다.
인터넷 공간으로 떠도는 정보는 누구나 볼 수 있습니다.

직접 보내는 사람과 최종적으로 받는 사람만이 봐야 합니다

우리가 보내는 패킷은 여러 컴퓨터들로 구성된 네트워크를 지나죠
이 컴퓨터들이 나쁜 마음을 먹으면 우리의 정보를 볼 수 있어요
(man in a middle attack)

또한 몇몇 라우터-스위치들은 멍청해서 패킷을 다른 곳으로 보낼 수 있어요
(spoofing)

SSL 과정

1. 서버야, 너 한번 12345 를 네 **개인키**로 암호화해서 줘봐
2. 유저야 여기 712345
3. 서버야 그리고 네 **인증서** 한번 보여줘봐
4. 유저야 여기있어
5. **내가 가지고 있는 믿음직한 공개키**로 풀어보니까 인증서가 올바른군
인증서에 명시되어있는 공개키 123으로 푸니까
12345가 나오는걸? 네가 진짜 서버가 맞구나!
저 서버에게 **4321**을 공개키로 암호화해서 53141234를 전송하자.
그럼 이제 우리 53141234을 **해독한 결과**로 통신하자. 무슨 숫자인지는
네 비밀키로 확인해봐
6. 음 해독해보니 4321이군! 이제 저 클라이언트랑은 4321
로 통신하면 되겠다.

Root CA

믿을 수 있는 놈들

애네가 배신때리면 답없어요...

강 CA들...

믿을 수 있는 놈이 믿을 수 있다고 한 놈

Root CA 들은 자신만의 비밀키를 가지고 있습니다.

웹 브라우저에는 이들의 지문(공개키)이 저장되어 있습니다.
그래서 내가 지금 접속하는 서버의 '인증서' 에 있는
'CA'의 서명을 확인합니다.

내가 CA역할을 할 수도 있습니다

“이 사이트는 내가 보장한다” 라고 보증해 줄 수 있습니다.
믿을지 안 믿을지는 브라우저 사용자의 선택

=> 자체 제작 인증서

usage: genrsa [args] [numbits]

- des encrypt the generated key with DES in cbc mode
- des3 encrypt the generated key with DES in ede cbc mode (168 bit key)
- aes128, -aes192, -aes256 encrypt PEM output with cbc aes

여기까지는 비밀키를 아무나 보지 못하도록 하는 부분

- out file output the key to 'file'
- passout arg output file pass phrase source

여기까지는 비밀키를 출력하는 옵션

- f4 use F4 (0x10001) for the E value
- 3 use 3 for the E value

여기까지는 public modulus

- engine e use engine e, possibly a hardware device.
- rand file:file:...
load the file (or the files in the directory) into
the random number generator

아마 졸업까지 사용하지 않을 듯한 옵션들

DES3로 암호화되고 F4를 지수로 사용하는 1024bit키를 private.key로 저장

CA님 인증좀....

req -new -key [아까 만든 키 파일] -out [요청 내용 파일 이름]

Common name에 사이트 주소를 적어주면 된다

생성된 파일을 베x사인, 등에 보내주면 내 파일에 서명을 해 준다....

내가 인증하고 싶을 때

내 비밀키를 내가 인증한다.

```
req -new -x509 -key ssl.key
```

역시 출력 포맷이나 암호화 알고리즘은 내가 설정할 수 있다.

아파치에 연동하기

a2ensite default-ssl

Available-modes의 ssl.conf 파일의 맨 끝에
SSLCertificateFile 내가 서명한 나의 요청 .cert
SSLCertificateKeyFile 에 나의 비밀키 .key