MTAT.07.017
Applied Cryptography

Rakenduslik krüptograafia
Прикладная криптография

Juri Hudolejev
University of Tartu
Spring 2011

{

Practical course – theory is clear already

Using existing tools, libraries and applications – not reinventing the wheel

Solving (almost) real-world problems

Course Structure

15 practicals (2 + 2 hours) – weekly

- Fri 8:30 and 12:00 @ Liivi 2-205
- Start in class and complete at home

(optional) Project at the end of the course

(optional) Exam

Total: 6 EAP (6+ hours weekly)

Grading

Practicals + homeworks: **80%** – required

Project: **20%** – recommended

Exam – those who want to improve the grade

- Session time (June 2011)
- Will agree the time and form separately if needed

Practical Topics

Introduction, big numbers, secret sharing

Public-key cryptography

Hashing, randomness

ASN.1 (BER, DER)

X.509 (CRL, OCSP)

Trusted timestamping, digital signatures

Smart cards, EstEID

Key exchange protocols, SSL/TLS

Project

Individual or group work, depending on task

Will brainstorm and discuss topics later

Possible areas:

- Java/EstEID, timestamps, Python/ASN.1, …

Feel free to propose your own topics, anytime

Development Environment

Class:

- OpenSUSE 11 64bit / Sun Java 6.0 / BC 1.45

Testing:

- Ubuntu Maverick 32bit / OpenJDK 6.0 / BC 1.45

Other systems and languages:

- Installing: you are on your own
- Testing: make sure to agree beforehand!

Submitting the Homeworks

Deadline: Next Friday 08:00 EET/EEST

http://courses.cs.ut.ee/2011/appcrypto/Main/Homework/

Source code only!

Testing system will compile and run the code

If using external libraries:

- Provide **exact** version used
- Make sure it is easily and publicly available
- Prefer F/OSS

Submitting the Homeworks

Code have to pass all tests

- Tests will be made publicly available in class or some days later.

Suggestion: use code repositories

- GitHub, BitBucket, Google Code, own hosting

Deadline: Next Friday 08:00 EET/EEST

whoami

Juri Hudolejev

MSc in Computer Science
Tallinn University of Technology, 2009

Infrastructure Engineer at ZeroTurnaround

http://sayat.me/hudolejev
All feedback is welcome

Questions?

Discussion

Security goals

Security issues

Cryptographic methods

Case 1

A company website is under a DDoS attack from a botnet.

The company would like to keep this from happening in the future.

Case 2

Alice's laptop requires a password to log in.

Bob keeps on guessing the password and can access the laptop.

Alice would like to keep Bob from logging in under her account.

Case 3

Alice's laptop connects to a wireless network.

Bob uses a network sniffer to read Alice's traffic, finds her Facebook password, logs in and posts defamatory messages under her name.

Case 4

Alice did not log out when she went to the coffee machine.

Bob ran her email client which had a saved password and sent an insulting email to Alice's boss.

Case 5

Charlie is a developer of a web-based information collecting system where people can enter and track their sporting habits.

He thinks that by default, every user should only see his or her data, but every user can also select other users (coaches) who can see the results and give suggestions.

Case 6

Charlie makes backups of his software code on an optical disk.

He would like to keep others from accessing the software code in his backups.

Case 7

Charlie is writing an information system for the government.

The government knows that the officials have to access the personal data of citizens, but they want to keep the privacy damages to a minimum.

Case 8 (two more to go...)

Bob writes a document to confirm that he borrowed 100 EUR from Charlie and sends it to Charlie via email.

Charlie wants to be sure that the document came from Bob and it is enforceable.

Case 9 (almost done...)

Charlie is developing a door keycard system based on contactless RFID cards.

The customer wants to be sure, that the risk of using a stolen/missing card to access a room is lowered.

Case 10 (the last one!)

Charlie is a developer of an online casino with a lottery game.

Every day, the server generates ten numbers and users predict up to five of them. At 9 pm the server publishes the numbers it generated and the winners.

Charlie would like to give the users a guarantee that the server does not cheat – it does not change the numbers if it knows that someone would win.

Cryptographic System

Computer system that involves cryptography

(definition acceptable for secondary school only)

Set of algorithms sufficient to perform particular encryption and decryption operations

- Key generation (k, k+, k-, etc.)
- Encryption (plaintext → ciphertext)
- Decryption (ciphertext → plaintext)

Keys

Symmetric algorithms: 80 .. 128 bits

Public key crytpography: 1024 .. 3072 bits

Elliptic curve cryptography: >= 163 bits

```
java.lang.Integer
  public static final int MAX_VALUE: 2^31-1
java.lang.Long
  public static final long MAX_VALUE: 2^63-1
```

Arbitrary-Precision Arithmetic

C/C++:

- `long long` aka `quad` (64 bit)
- extarnal libs: OpenSSL BN, GMP, Crypto++

C#

- `System.Numerics.BigInteger`

Java

- `java.math.BigInteger, java.math.BigDecimal`

Python

- `long` (merged with `int` in Python 3)

Exercises

Compute the square of maximum `long` value.

Plaintext: any human-readable string you like

Convert it to decimal number

Generate random key of the same bit-length

Compute ciphertext: `plaintext XOR key`

Restore plaintext: `ciphertext XOR key`

Do plaintexts match?

Secret Sharing

Split a secret into pieces so that

- Individual pieces reveal no info about the secret
- All (or some) pieces together can be used to restore the initial secret

http://en.wikipedia.org/wiki/Secret_sharing

Shamir Secret Sharing Scheme

- (K, N) scheme – only some of the pieces are needed to reconstruct the secret

Shamir Secret Sharing: Example

Secret: 101

K = 3

N = 4

A = 17 (random)

B = 42 (random)

Polynomial:

$17x^2 + 42x + 101$

Shares (x_i:y_i):

1:160

2:253

3:380

4:541

$$p_1 = \frac{(x-x_2)(x-x_4)}{(x_1-x_2)(x_1-x_4)} = \frac{(x-2)(x-4)}{(1-2)(1-4)} = \frac{x^2}{3} - 2x + \frac{8}{3}$$

$$p_2 = \frac{(x-x_1)(x-x_4)}{(x_2-x_1)(x_2-x_4)} = \frac{(x-1)(x-4)}{(2-1)(2-4)} = -\frac{x^2}{2} + \frac{5}{2}x - 2$$

$$p_4 = \frac{(x-x_1)(x-x_2)}{(x_4-x_1)(x_4-x_2)} = \frac{(x-1)(x-2)}{(4-1)(4-2)} = \frac{x^2}{6} - \frac{x}{2} + \frac{1}{3}$$

$$160\left(\frac{x^2}{3} - 2x + \frac{8}{3}\right) - 253\left(\frac{x^2}{2} - \frac{5}{2}x + 2\right) + 541\left(\frac{x^2}{6} - \frac{x}{2} + \frac{1}{3}\right)$$

$$\frac{160}{3}x^2 - 320x + \frac{1280}{3} - \frac{253}{2}x^2 + \frac{1265}{2}x - 506 + \frac{541}{6}x^2 - \frac{541}{2}x + \frac{541}{3}$$

$$17x^2 + 42x + 101$$

Task 1.1 (2 points)

Implement trivial secret sharing scheme using XOR

man http://en.wikipedia.org/wiki/Secret_sharing

Suggested method signatures:

```
BigInteger[] share(BigInteger secret, int n);
BigInteger restore(BigInteger[] shares);
```

Run:

```
java TrivialSecretSharing share <secret> <N>
java TrivialSecretSharing restore <s1> .. <sN>
```

Task 1.2 (3 points)

Implement Shamir's secret sharing scheme (K, N)

Use `ID:Value` format for shares, e.g. `6:100500`

Restore command should accept exactly `K` shares

man http://en.wikipedia.org/wiki/Shamir%27s_Secret_Sharing

Run:

```
java ShamirSecretSharing share <secret> <K> <N>
java ShamirSecretSharing restore <X:sX> .. <Y:sY>
```

Questions?

Go Code!

Deadline: Friday Feb 18 08:00 EET

Submit at:
http://courses.cs.ut.ee/2011/appcrypto/Main/Homework

Good luck!

}

Course info:

http://courses.cs.ut.ee/2011/appcrypto/

Contact:

Juri Hudolejev <juri@ut.ee>

Next lab session:

Friday 2011-02-18 **08:30** EET @ Liivi 2 - 205