2014-01-29.sagews

January 29, 2014

Contents

1 January 29, 2014: Public-key Cryptography, part 2 (RSA)

- screencast:
- whiteboard:

Approach: we will focus on the core number-theoretic ideas behind public-key crypto for now, completely ignoring the nice infrastructure that must be added on top in order to make it pleasant to use and more secure.

1.1 Whiteboard

(explain the theoretical idea behind RSA and how it works)

```
my_rsa = dict(rsa)  # make a copy
```

```
my_rsa['e']
29349401013494713972160676180756 7017373
```

```
my_rsa['n']
651790867435205665208048899946415370153
```

1.2 A live demo with somebody in the class

1.2.1 Part 1: receive a message

- I will put an e and an n into a chat.

- Then wait for a number back.
- Then we will decrypt the number.

1.2.2 Part 2: send a message

- Student will make up an e and an n and paste them into a chat.
- Then we will encrypt message to student.
- Student will decrypt it.

1.2.3 Part 3: sign a message

- Student will make up a message
- Student will encrypt message with their decryption key.
- Send it. We will verify the signature.

1.3 Crazy project idea crypto is a pain to use for communication, so make it easy!?

Create a small javascript library that a person can embed in a webpage, which provides a little textbox for encrypting a message to you in it. So, e.g., I could put it at http://wstein.org, and then even my mom could easily encrypt a message to me. She would type the message into a box, then click encrypt (or maybe even it would encrypt as you type), then take the output and paste it into an email to me.

The configuration for the javascript library would only include the public part of the key (n, e) and nothing else, so there are no security risks due to that. The main problem with this is that the person sending the message has to trust the website that contains the public key.

Note that the message never gets sent over the internet before encryption it stays in the browser.

There are many existing libraries written in Javascript that basically solve this problem, e.g., `http://www-cs-students.stanford.edu/~tjw/jsbn/`; so you dont have to worry about actually implementing big integer arithmetic, etc. This would be more a matter of making something look pretty and friendly and EASY.