Cry – Project 1 (Project Plan): Presentation

2017-01-29

1. Motivation

2. Overview
   - Example Use Case
   - Functionality

3. Expectation

Section 1

Motivation

end-user

- Problem: An end-user wants to send and receive secure messages with other end-users.
- Solution: `Cry` lets end-users establish secure communication via built-in cryptosystems.

cryptographer

- Problem: As new technologies emerge, and as technology improves, new cryptosystems need to be developed.
- Solution: `Cry` allows cryptographers to easily prototype and benchmark their new cryptosystems.

Section 2

Overview

Situation

- *Alice* (sender) wants to confidentially send a message to *Bob* (receiver).
- *Eve* (eavesdropper) wants to know that message.

Procedure

1. *Alice* (sender)

```
$ cry encrypt --cryptosystem=rsa \
> --public-key=825 --plaintext=4692301804
The ciphertext is 1110003333 (took 1 second).
```

2. *Bob* (receiver)

```
$ cry decrypt --cryptosystem=rsa \
> --private-key=637 --ciphertext=1110003333
The plaintext is 4692301804 (took 1 second).
```

3. *Eve* (eavesdropper)

```
$ cry eavesdrop --cryptosystem=rsa \
> --public-key=825 --ciphertext=1110003333
The plaintext is 4692301804 (took 1 century).
```

In the previous example:

- `Cry` is the cryptographic framework.
- RSA is a cryptosystem implemented in `Cry`.
- The key-generation, encryption, decryption, and eavesdropping algorithms are specific to RSA.

In general, with Cry:

- an end-user can use an implemented cryptosystem to confidentially send and receive messages with others.
- a cryptographer can:
  - prototype her own cryptosystems where the cryptographic algorithms are either newly defined or reused from different existing cryptosystems.
  - test her cryptosystems for security and performance.

Section 3

Expectation

I am a frame title

I am a frame body. Copy/paste from report section.

title of frame 2

body of frame 2