`Cry` (cryptographic framework)
Project 1 (Project Plan): Report

Daniel Dunning, Michael Degraw, Vu Phan

2017-01-29

Contents

1 Motivation

- For end-users:
  - 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.
- For cryptographers:
  - Problem: New cryptosystems need to be developed.
    - Reason: Most existing cryptosystems are secure only if their assumptions are true.
    - Example: The RSA cryptosystem's security relies on the assumption that integer factorization is hard. It is currenly unknown whether a polynomial-time factorizing algorithm exists.

    - Solution: `Cry` let cryptographers easily prototype and test their new cryptosystems.

2 Overview

2.1 Example Use Case

Situation:

- *Alice* (sender) wants to confidentially send a message to *Bob* (receiver).

- *Eve* (eavesdropper) wants to know that message.

Procedure:

1. Each person downloads the binary file `cry` of the `Cry` cryptographic framework.

2. *Bob* publishes his choice of cryptosystem: RSA (Rivest, Shamir, Adleman).

3. *Bob* generates his keys:

   ```
   $ cry getkeys --cryptosystem=rsa
   The public & private keys are 825 & 637 (took 1 second).
   ```

4. *Bob* publishes his public key (and hides his private key).

5. *Alice* obtains *Bob*'s published public key.

6. *Alice* encrypts her message (say, her phone number):

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

7. *Alice* publishes the encrypted message.

8. *Bob* obtains *Alice*'s published encrypted message.

9. *Bob* easily decrypts the message with his private key:

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

10. *Eve* struggles to eavesdrop the message without *Bob*'s private key:

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

2.2 Functionality

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.

3 Expectation

- Usability:
  - `Cry` enables end-users to establish secure communication via simple command-line options.
  - Being a modular object-oriented framework, `Cry` lets cryptographers easily define and benchmark new cryptosystems.