Encryptr: Easily Encrypt and Decrypt Sensitive Data with R

Cameron Fairfield

2019-Sep-13

Why we are waking up to the value of our own data

Scotsman, Aug-30, 2019

Data Governance

- Data is precious and sensitive

Data Governance

- Data is precious and sensitive
- Confidential data (when possible) should be anonymised or pseudonomysed

Data Governance

- Data is precious and sensitive
- Confidential data (when possible) should be anonymised or pseudonomysed
- Patients expect us to safeguard their data

Data Governance

- Data is precious and sensitive
- Confidential data (when possible) should be anonymised or pseudonomysed
- Patients expect us to safeguard their data
- Data can be minimised, deleted or encrypted

Research and Data

- Data governance approvals are key components of research approvals

Research and Data

- Data governance approvals are key components of research approvals
- GDPR (Europe) and HIPAA etc. (USA)

Research and Data

- Data governance approvals are key components of research approvals
- GDPR (Europe) and HIPAA etc. (USA)
- Data breaches are financially and reputationally costly

Research and Data

- Data governance approvals are key components of research approvals
- GDPR (Europe) and HIPAA etc. (USA)
- Data breaches are financially and reputationally costly
- Not all data can be removed from records

Encryptr

- Easy pseudonymisation by encryption

Encryptr

- Easy pseudonymisation by encryption
- RSA encryption with private / public key pair (asymmetric)

Encryptr

- Easy pseudonymisation by encryption
- RSA encryption with private / public key pair (asymmetric)
- Encryption of vectors, variables and files

Encryptr

- Easy pseudonymisation by encryption
- RSA encryption with private / public key pair (asymmetric)
- Encryption of vectors, variables and files
- Secure storage of confidential data (and allocation concealment / blinding)

Encryptr on CRAN / Github

```r
install.packages("encryptr") # CRAN
remotes::install_github("SurgicalInformatics/encryptr")
```

https://github.com/surgicalinformatics

```r
library(encryptr)

library(dplyr) # Used in presentation examples

# Encryptr comes with an example data set of GPs (Family Physicians)

gp
```

```
## # A tibble: 1,212 x 12
##   organisation_co~ name  address1 address2 address3 city  county
##   <chr>            <chr> <chr>    <chr>    <chr>    <chr> <chr>
## 1 S10002           MUIR~ LIFF RO~ MUIRHEAD <NA>     DUND~ ANGUS
## 2 S10017           THE ~ CRIEFF ~ KING ST~ <NA>     CRIE~ PERTH~
## 3 S10036           ABER~ TAYBRID~ <NA>     <NA>     ABER~ PERTH~
## 4 S10060           ABER~ TAYBRID~ <NA>     <NA>     ABER~ PERTH~
## 5 S10106           GROV~ 129 DUN~ BROUGHT~ <NA>     DUND~ ANGUS
## 6 S10125           ALYT~ NEW ALY~ ALYTH    <NA>     BLAI~ PERTH~
## # ... with 1,206 more rows, and 5 more variables: postcode <chr>,
## #   opendate <date>, closedate <date>, telephone <chr>,
```

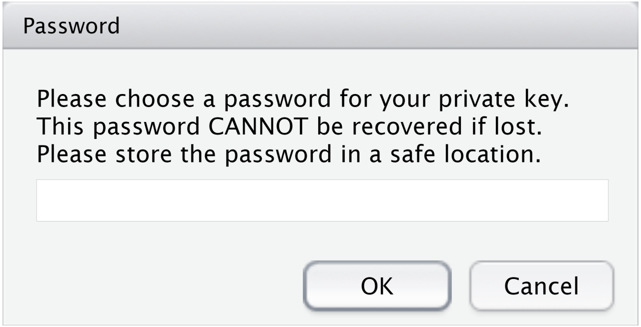
Public and Private Keys

```
genkeys()
```

```
## Private key written with name 'id_rsa'
## Public key written with name 'id_rsa.pub'
```

Default values are “id_rsa” and “id_rsa.pub”

No Raw Text Password



Please choose a password for your private key.
This password CANNOT be recovered if lost.
Please store the password in a safe location.

```r
gp_encrypt = gp %>%
  encrypt(name)

gp_encrypt %>%
  select(organisation_code, name, address1)
```

```
## # A tibble: 1,212 x 3
##   organisation_code name                                address1
##   <chr>             <chr>                               <chr>
## 1 S10002            15f3778d917bb341acefe0fa4b7413739e~ LIFF ROAD
## 2 S10017            609e097052753986 58a5a9e01b05180243~ CRIEFF MEDICA~
## 3 S10036            39769353fd5ae7d8ab11a719b12cd69c30~ TAYBRIDGE ROAD
## 4 S10060            656ecb5d0db34a637e55103d9d5ae43add~ TAYBRIDGE ROAD
## 5 S10106            497109e512b08d0b0d3d2012bab1cfe69e~ 129 DUNDEE RO~
## 6 S10125            2fe49f4b3b9b8e26646cc4fb4067726b8d~ NEW ALYTH ROAD
## # ... with 1,206 more rows
```

```r
gp_encrypt %>%
  slice(1:2) %>%
  decrypt(name) %>%

  select(organisation_code, name, address1)
```

```
## # A tibble: 2 x 3
##   organisation_code name                    address1
##   <chr>             <chr>                   <chr>
## 1 S10002            MUIRHEAD MEDICAL CENTRE LIFF ROAD
## 2 S10017            THE BLUE PRACTICE       CRIEFF MEDICAL CENTRE
```

Encryptr Customisation

- Use look-up table - create object with encrypted output and ID variable on which to match

Encryptr Customisation

- Use look-up table - create object with encrypted output and ID variable on which to match
- Write a look-up file

Encryptr Customisation

- Use look-up table - create object with encrypted output and ID variable on which to match
- Write a look-up file
- Customise file names, key names, encrypt several variables

Encryptr Customisation

- Use look-up table - create object with encrypted output and ID variable on which to match
- Write a look-up file
- Customise file names, key names, encrypt several variables
- Use a publicly-available public key

Encryptr Customisation Examples

```r
# Creating a lookup table with specified name and filename

gp %>%
  encrypt(name, postcode,
          lookup = TRUE, write_lookup = TRUE,
          lookup_name = "new_lookup_name")



# Using a public key hosted at URL
gp %>%
  encrypt(name, public_key_path = "https://<some_url>/id_rsa.pub")
```

Encryptr File Encryption

```r
gp_encrypt %>% write_csv("gp_enc.csv")


encrypt_file("gp.csv")



# Encrypted file will have suffix: `.encryptr.bin`
decrypt_file("gp.csv.encryptr.bin", file_name = "gp2.csv")
```

Encryptr Ciphertexts Not Matchable

- Each repeat of encryption generates a unique number

Encryptr Ciphertexts Not Matchable

- Each repeat of encryption generates a unique number
- Prevents malicious, opportunistic use of public key

Encryptr Ciphertexts Not Matchable

- Each repeat of encryption generates a unique number
- Prevents malicious, opportunistic use of public key
- Alternative symmetric encryption outputs can be matched (and not always reversed)

Encryptr Ciphertexts Not Matchable

- Each repeat of encryption generates a unique number
- Prevents malicious, opportunistic use of public key
- Alternative symmetric encryption outputs can be matched (and not always reversed)
- Alternative methods need a “salt”

```r
encrypt_vec(c("a name", "a name", "a name"))
```

```
## [1] "415a3477ac62de74cd0716a56aeeaa59ebc616aa91815bde5ad4e247a11076853c8
## [2] "4868daf7746a128c9d4eeef0ea3dc6ce78bf731fd1bc11fe749e8899c6083982c4
## [3] "aeda24c161df36dc1594a5b95c917e68abdb7eae9693c0a01215a671dbb76ce956
```

Technical Aspects of Encryptr

- Wrapper around OpenSSL (and purrr)

Technical Aspects of Encryptr

- Wrapper around OpenSSL (and purrr)
- RSA asymmetric encryption for vectors (each component in vector $<$ 256 bytes)

Technical Aspects of Encryptr

- Wrapper around OpenSSL (and purrr)
- RSA asymmetric encryption for vectors (each component in vector $<$ 256 bytes)
- File encryption uses AES technique with symmetric session key which is in turn encrypted by RSA public key

What Encryptr Is Useful For

- Storing confidential data in trials, cohorts, service evaluation, etc.

What Encryptr Is Useful For

- Storing confidential data in trials, cohorts, service evaluation, etc.
- Retrieving individual data if needed

What Encryptr Is Useful For

- Storing confidential data in trials, cohorts, service evaluation, etc.
- Retrieving individual data if needed
- Blinding in RCTs

What Encryptr Is Useful For

- Storing confidential data in trials, cohorts, service evaluation, etc.
- Retrieving individual data if needed
- Blinding in RCTs
- Data governance considerations

Questions