Sharing secrets safely with goldeneye

Denwood, M.J.[1*], Reimert, M.M.[1], Gussmann, M.K.[1], Nielsen, S.S.[1], Calvo-Artavia, F.F.[2], Ewerlöf, I.R.[3] & Enright, J.[4]

[1] Department of Veterinary and Animal Sciences, University of Copenhagen, Denmark

[2] Danish Veterinary and Food Administration, Denmark

[*] *md@sund.ku.dk*

[3] National Veterinary Institute, Sweden

[4] School of Computing Science, University of Glasgow, UK

"Just because you're paranoid doesn't mean they aren't after you." -- Joseph Heller, Catch-22

Motivation

- Epidemiology frequently requires the use of large and potentially sensitive datasets obtained from external data providers
- We have a legal and moral responsibility to do everything we can to ensure that these data are always secure when under our care
- However, commonly implemented data security procedures are often either ineffective, cumbersome to use, or both
- **The Goldeneye sOLution for moDErN data safEtY procEdures (goldeneye[1,2,3]) leverages modern cryptographic methods in an easy-to-use R package wrapper, which ensures that routine storage of encrypted data can be achieved using a single R function**

How does it work?

We use a hybrid strategy of symmetric and asymmetric encryption, which are implemented using the sodium cryptography library [4]

1. Alec and Boris want to share data. They set up a new data-sharing group called "Janus" using the goldeneye package in R.
2. After setup, each user has a public-private keypair for use with asymmetric encryption.
3. Their public key is made available online so that all "Janus" users can download the key corresponding to each user.
4. Alec can then encrypt data for Boris using goldeneye with **`gy_save(data, file="secret.rdg", user="boris")`**, which:
    a. Encrypts the data using a randomly generated 256-bit symmetric encryption key for efficient and robust encryption.
    b. Encrypts the symmetric key separately for Boris and Alec using each of their public keys.
    c. Bundles the encrypted data and encrypted symmetric keys together as a goldeneye encrypted file.
5. Alec can safely store this file and/or send this file to Boris over insecure channels: it can only be decrypted using Boris's (or Alec's) private key, using the following R code: **`gy_load("secret.rdg")`**

Additional mechanisms are provided for facilitating automatic logging and/or a "kill switch" to remotely revoke decrypt authorisation

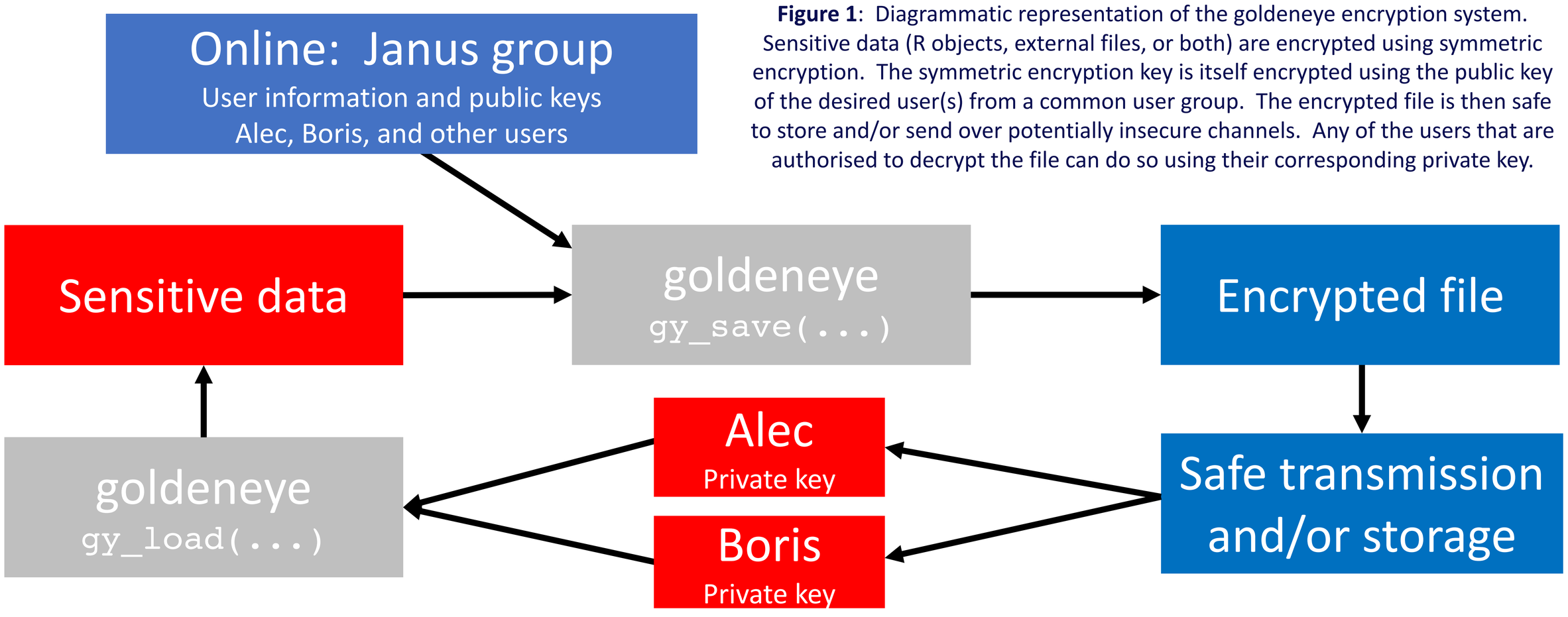

Figure 1: Diagrammatic representation of the goldeneye encryption system. Sensitive data (R objects, external files, or both) are encrypted using symmetric encryption. The symmetric encryption key is itself encrypted using the public key of the desired user(s) from a common user group. The encrypted file is then safe to store and/or send over potentially insecure channels. Any of the users that are authorised to decrypt the file can do so using their corresponding private key.

Try goldeneye for yourself!

- **We have set up a SVEPM2022 user group so that anyone interested in encryption can try out goldeneye**
- **All you need to do is set up a user account in R and send us your public key for upload to the SVEPM2022 group**
- **Follow the instructions at the link below (or the QR code opposite) to get started:**

https://www.costmodds.org/goldeneye/SVEPM2022

[1] https://en.wikipedia.org/wiki/Backronym, [2] https://en.wikipedia.org/wiki/Recursive_acronym, [3] https://www.costmodds.org/goldeneye, [4] https://CRAN.R-project.org/package=sodium

Acknowledgements: This work was carried out as part of the DigiVet project, which is funded by NordForsk: http://www.dcs.gla.ac.uk/~jenright/digivet_website/