# Polymorphic Encryption and Pseudonymisation (PEP)

For privacy-friendly Personalised Medice

Bart Jacobs and the PEP team
bart@cs.ru.nl
Sept. 16, 2016



## Outline





## Where we are, sofar





## PEP overview

- We present a new, innovative set-up for encryption and pseudonymisation of sensitive personal data
- It's called polymorphic: the encryption and pseudonymisation can be "transcribed" to different recipients
- It can be applied in privacy-friendly cloud storage and identity management
  - here it will be described in a medical context
  - there are many other application areas, esp. in the Internet of Things — dealing with sensor / behavioural / surveillance data
- Here it will be described in a medical context
  - both encryption and pseudonymisation are highly relevant
  - data sources are e.g. medical equipment, doctor's input, or self-measurement devices
  - well within the constraints of European regulation



## Cryptographic basis

- Malleability of ElGamal public key encryption
  - ideas orginally developed by colleague Eric Verheul
  - they form the basis for complex protocols (team effort)
- The presentation contains two parts:
  - an informal one, to convey the main ideas
  - a formal one, for the underlying math and protocol hints



## Personalised medicine & PEP

- New development in healthcare: fine-grained personalised treatment based on statistical outcomes of large scale analysis of patient data
- In personalised healthcare one has to deal with:
  - identifyable medical data for the diagnosis and treatment of individual patients;
  - pseudonymised patient data for large scale medical research;
  - multiple sources of patient data, including in particular (wearable) self-measurement devices and apps.
  - the need to ensure confidentiality of patient data — and integrity, authenticity and availability too;
- The PEP framework is designed for this situation; it offers:
  - unprecented privacy-protection via encryption and pseudonymisation
  - support for the basic data-access functionality for research, and potentially treatment too, in personalised healthcare.

## New EU privacy regulation, and PEP

- Europe has recently (April 2016) adapted the GDPR
  - GDPR = General Data Protection Regulation
  - effective after a 2-year transition period
- It demands data protection by design and default
  - mandatory DPIA = data protection impact assessment
  - hefty fines for non-compliance
- The GDPR encourages innovation, as long as organisations implement appropriate safeguards
  - it allows for subsequent processing that is "compatible"

> Don't whine about the GDPR, but set-up proper protection!

This is where PEP comes in.



## Why "polymorphic"?

- In traditional encryption one encrypts for some chosen recipient
  - this recipient holds the cryptographic key for decryption
- In polymorphic encryption one encrypts in a generic manner
  - later on, this encryption can be transcribed/tuned so that any participant

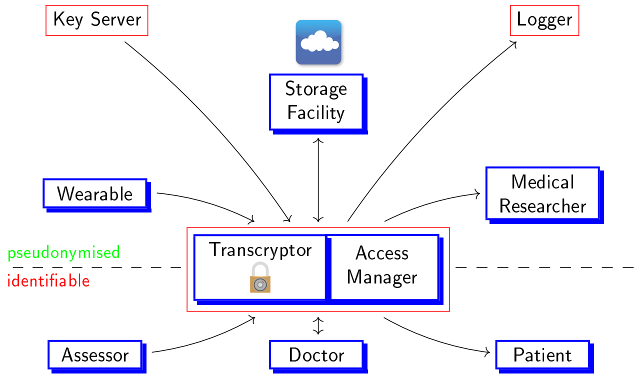
**Illustration in a medical setting:**

(1) any patient/user with a self-measurement device encrypts the collected (medical) data polymorphically

(2) later, access to the data can be transcribed specifically
- e.g. for selected doctors, or researchers, or service companies



## PEP overview picture: the "PEPcloud"






## Where we are, sofar

Introduction

Informal description, with pictures
- Polymorphic encryption
- Polymorpic pseudonymisation

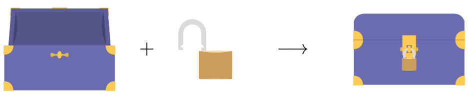
Formal description, mathematically

Conclusions

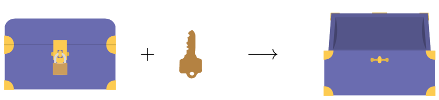


## Traditional (public key) encryption, pictorially

- **Encryption of data** : putting it in a locked chest



- **Decryption of data** : unlocking the chest

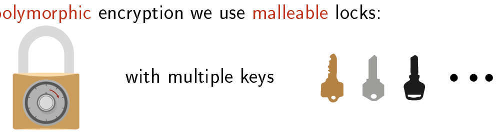


**Terminology**: = public key = private key

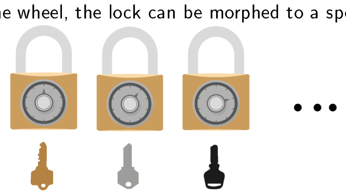


## Polymorphic locks

- Traditionally, only the owner of the private key can decrypt
- In polymorphic encryption we use malleable locks:

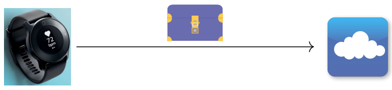



- By turning the wheel, the lock can be morphed to a specific key:

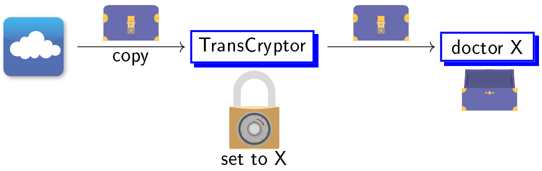


## Polymorphic encryption scenario (no pseudonyms yet)

- Sensitive device data are stored under polymorphic encryption



- Later on, device user gives doctor X access to the data:




The TransCryptor learns nothing about the data!



## Conclusions about polymorphic encryption

- Sensitive data can be encrypted, without a priori fixing who can decrypt
  - the data is inaccessible by the cloud storage provider
- At any later stage, the data can be made decryptable, for any participant
  - this works by blind transcription
  - it can be repeated at will



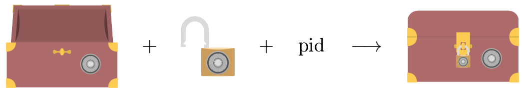
## Basic idea in polymorphic pseudonymisation

- Each user/patient $A$ has a unique identifier $\mathrm{pid}_A$ (= patient identifier)
  - e.g. social security number, like BSN in NL
- This pid can be "morphed" into pseudonyms, different per data handler
  - data handler means: doctor, researcher, assessor, ...
  - morphing into pseudonyms is done in a uniform manner
  - represented again as a wheel, that can be turned blindly
- We call the pseudonym for data handler $X$, generated from $\mathrm{pid}_A$, the local pseudonym of $\mathrm{pid}_A$ at $X$ — written later as $\mathrm{pid}_A@X$.
- The central TransCryptor can create these local pseudonyms — again in a blind manner



## Polymorphic pseudonyms, pictorially

- An encrypted pseudonym is a pid in a chest with an extra wheel:



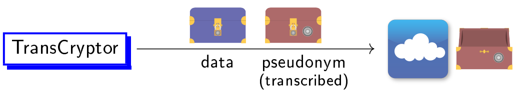

- This second wheel changes the content, in a blind manner
- The TransCryptor can set both wheels coherently, so that participant $X$ can decrypt and find the local pseudonym of pid at $X$
- There are now two chests:
  (1) one data-chest, as for polymorphic encryption 
  (2) one pseudonym-chest, with an extra wheel 



## Storage scenario, with pseudonyms

- The user (device) puts medical data in the data-chest, and his/her pid in the pseudonym chest, and sends both to the TransCryptor:


- The TransCryptor adjusts both wheels on the pseudonym-box — but does nothing with the data box!

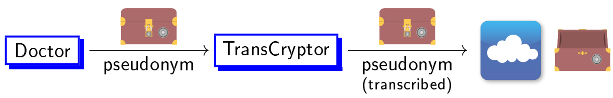



- The encrypted data are stored under the local pseudonym of pid for the Storage Facility
  - the same happens with data from other sources

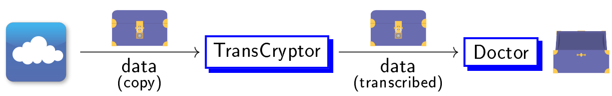


## Retrieval scenario, with pseudonyms

- Doctor $X$ wants to get stored data for a patient
  - she knows pid and sends it in a pseudonym box




- The Storage Facility finds his local pseudonym for pid in the chest, and sends all associated (encrypted) data back:



## Where we are, sofar





## ElGamal basics

Let $G$ be an additive (elliptic curve) group with generator $g$ of prime order $p$ (so $p \cdot g = 1$)

- **Keys:** $x \in \mathbb{F}_p$ private key, $y = x \cdot g$ is associated public key
  - recall discrete log problem, for hiding $x$
- **Encryption** of $M \in G$

  $\langle r \cdot g,\ r \cdot y + M \rangle$ where $r \in \mathbb{Z}_p$ is random
  - This $r$ randomises the ciphertext.
- **Decryption** of $\langle b, C \rangle$ using private key $x$

  $C - x \cdot b$
- **Correctness:** decryption-after-encryption is identity:

  $(r \cdot y + M) - (x \cdot (r \cdot g)) = (r \cdot (x \cdot g) + M) - (r \cdot (x \cdot g)) = M.$



## ElGamal manipulations

We introduce explicit notation, retaining the public key $y$

$$\mathcal{EG}(r, M, y) = \langle r \cdot g,\ r \cdot y + M,\ y \rangle$$

We describe three operations on ElGamal ciphertexts:

(1) **re-randomise**: to change the appearance, but not the content

(2) **re-key**: to change the target, who can read the ciphertext ( )

(3) **re-shuffle**: to raise the plaintext to a certain power ( )

These operations will be defined as three functions $\mathcal{RR}, \mathcal{RK}, \mathcal{RS}$ each of type, independent of any encryptions

$$G^3 \times \mathbb{F}_p \longrightarrow G^3.$$



## (1) Re-randomisation

**Definition (of $\mathcal{RR}\colon G^3 \times \mathbb{F}_p \to G^3$)**

Define re-randomisation with $s \in \mathbb{F}_p$ as:

$$\mathcal{RR}(\langle b, C, y \rangle, s) \stackrel{\text{def}}{=} \langle s \cdot g + b,\ s \cdot y + C,\ y \rangle$$

**Lemma**

*This re-randomising is an encryption of $M$ with random $s + r$, that is:*

$$\mathcal{RR}(\mathcal{EG}(r, M, y), s) = \mathcal{EG}(s + r, M, y)$$

**Proof**:
$$\begin{aligned}
\mathcal{RR}(\mathcal{EG}(r, M, y), s) &= \mathcal{RR}(\langle r \cdot g,\ r \cdot y + M,\ y \rangle, s) \\
&= \langle s \cdot g + r \cdot g,\ s \cdot y + r \cdot y + M,\ y \rangle \\
&= \langle (s + r) \cdot g,\ (s + r) \cdot y + M,\ y \rangle \\
&= \mathcal{EG}(s + r, M, y).
\end{aligned}$$



## (2) Re-keying (wheel on lock )

**Definition (of $\mathcal{RK}\colon G^3 \times \mathbb{F}_p \to G^3$)**

Define re-keying with $k \in \mathbb{F}_p$ as:

$$\mathcal{RK}(\langle b, C, y \rangle, k) \stackrel{\text{def}}{=} \langle \tfrac{1}{k} \cdot b,\ C,\ k \cdot y \rangle$$

where $\frac{1}{k} \in \mathbb{F}_p$ is the inverse of $k$.

**Lemma**

*This re-keying is an encryption of $M$ with public key $k \cdot y$, that is:*

$$\mathcal{RK}(\mathcal{EG}(r, M, y), k) = \mathcal{EG}(\tfrac{r}{k}, M, k \cdot y)$$

*It can be decrypted with adapted private key $k \cdot x$.*

**Proof**:
$$\begin{aligned}
\mathcal{RK}(\mathcal{EG}(r, M, y), k) &= \mathcal{RK}(\langle r \cdot g,\ r \cdot y + M,\ y \rangle, k) \\
&= \langle \tfrac{1}{k} \cdot r \cdot g,\ r \cdot y + M,\ k \cdot y \rangle = \mathcal{EG}(\tfrac{r}{k}, M, k \cdot y).
\end{aligned}$$



## (3) Re-suffling (wheel on chest )

**Definition (of $\mathcal{RS}\colon G^3 \times \mathbb{F}_p \to G^3$)**

Define re-shuffling with $n \in \mathbb{F}_p$ as:

$$\mathcal{RR}(\langle b, C, y \rangle, n) \stackrel{\text{def}}{=} \langle n \cdot b,\ n \cdot C,\ y \rangle$$

**Lemma**

*This re-shuffling with $n$ is an encryption of $n \cdot M$ with random $n \cdot r$:*

$$\mathcal{RS}(\mathcal{EG}(r, M, y), n) = \mathcal{EG}(n \cdot r, n \cdot M, y)$$

**Proof**:
$$\begin{aligned}
\mathcal{RS}(\mathcal{EG}(r, M, y), n) &= \mathcal{RS}(\langle r \cdot g,\ r \cdot y + M,\ y \rangle, n) \\
&= \langle n \cdot r \cdot g,\ n \cdot (r \cdot y + M),\ y \rangle \\
&= \langle (n \cdot r) \cdot g,\ (n \cdot r) \cdot y + n \cdot M,\ y \rangle \\
&= \mathcal{EG}(n \cdot r, n \cdot M, y).
\end{aligned}$$

## Some algebraic properties

(1) Re-keying and re-shuffling commute:

$$\mathcal{RK}\Big(\mathcal{RS}(\langle b, C, y\rangle, n), k\Big) = \mathcal{RS}\Big(\mathcal{RK}(\langle b, C, y\rangle, k), n\Big)$$

(2) Re-randomisation is a group action, of $\mathbb{F}_p$ on $G^3$

$$\mathcal{RR}(\mathcal{RR}(\langle b, c, y\rangle, s), s') = \mathcal{RR}(\langle b, c, y\rangle, s' + s)$$
$$\mathcal{RR}(\langle b, c, y\rangle, 0) = \langle b, c, y\rangle$$



## Polymorphic encryption via re-keying

- There is a master private key $x \in \mathbb{F}_p$, with public key $y = x \cdot g \in G$.
  - only the trusted key authority has $x$, stored in a HSM
- Each participant $A$ has a diversified private key $x_A = K_A \cdot x$.
  - only the TransCryptor knows the table of pairs $(A, K_A)$, in a HSM
  - $A$'s public key is: $y_A = x_A \cdot g = K_A \cdot x \cdot g = K_A \cdot y$.
- Polymorphic encryption of $D$ is $\mathcal{EG}(r, D, y)$, with master public key $y$
  - anyone can encrypt her data $D$ in this way, and put it in storage
  - if needed, the TransCryptor can re-key this ciphertext to participant $A$
  - via: $\mathcal{RK}(\mathcal{EG}(r, D, y), K_A)) = \mathcal{EG}(\frac{r}{K_A}, D, K_A \cdot y)$
    $= \mathcal{EG}(\frac{r}{K_A}, D, y_A)$
  - then $A$ can decrypt this, since $y_A = K_A \cdot y$ is her public key
- This only describes the bare essentials
  - proper authentication, authorisation and logging must be added

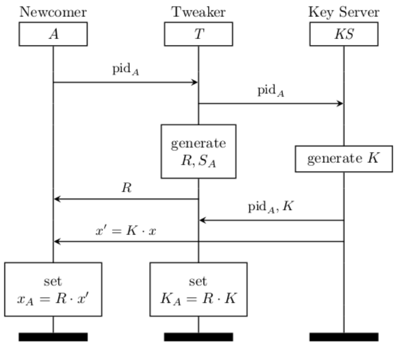


## Polymorphic pseudonymisation via re-shuffling

- Each patient $B$ has personal identifier $\mathrm{pid}_B \in G$
- $B$'s local pseudonym at $A$ is $\mathrm{pid}_B@A = S_A \cdot \mathrm{pid}_B$
  - only the TransCryptor knows these pairs $(A, S_A)$
  - $B$'s polymorphic pseudonym is $\mathcal{EG}(r, \mathrm{pid}_B, y)$
- All $B$'s data (for storage) is sent to the TransCryptor with this PP
  - the TransCryptor re-shuffles and re-keys PP to the local pseudonym $\mathrm{pid}_B@SF = S_{SF} \cdot \mathrm{pid}_B$ of the Storage Facility
  - Via: $\mathcal{RK}(\mathcal{RS}(\mathcal{EG}(r, \mathrm{pid}_B, y), S_{SF}), K_{SF})$
    $= \mathcal{EG}(\frac{S_{SF} \cdot r}{K_{SF}}, S_{SF} \cdot \mathrm{pid}_B, K_{SF} \cdot y) = \mathcal{EG}(S_{SF} \cdot r, \mathrm{pid}_B@SF, y_{SF})$
  - SF decrypts and uses this local pseudonym $\mathrm{pid}_B@SF$ as database key to store the (polymorphically encrypted) data of $B$
- If doctor $A$ wants to retrieve $B$'s data:
  - $A$ sends PP $\mathcal{EG}(r, \mathrm{pid}_B, y)$ to the TransCryptor, who re-keys and re-shuffles it to $SF$, who obtains his local pseudonym of $B$, and looks up and returns the requested data, which gets re-keyed to $A$



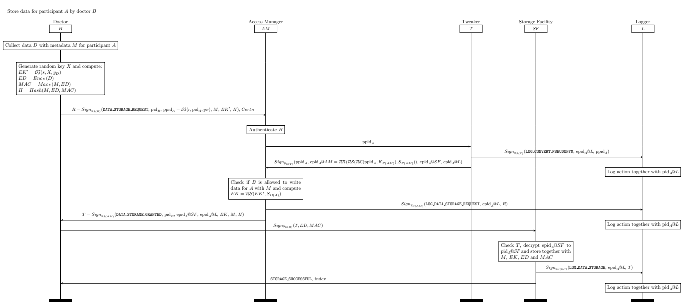
## Simplified example: key distribution protocol




Neither Key Server nor TransCryptor (Tweaker) learns $A$'s private key $x_A$.



## Actual protocols are a bit more complicated





## Where we are, sofar

## Main points (PEP is PET)

- Polymorphic encryption & pseudonymisation give unprecedented privacy protection
  - while retaining basic functionality for personalised healthcare
  - in line with GDPR's data protection by design
- The PEP-technology can become a new NL/EU/. . . standard
- Province of Gelderland has given 750K€ support for development
- Nijmegen's Digital Security group is:
  - elaborating and documenting the design
  - developing prototype implementations
  - making the open source software freely available
  - working with interested parties to get this up and running
- PEP forms the backbone of new Parkinson studies, using data from multiple sources, including wearables
- See PEP whitepaper at http://eprint.iacr.org/2016/411