Privacy-Preserving Aggregation of Data from Multiple Sources

David Pointcheval
CNRS - ENS - INRIA

LIG - Grenoble
January 10th, 2019

The Cloud

Dropbox

iCloud Drive

amazon cloud drive

Google Drive

OneDrive

Anything from Anywhere

One can store

- Documents to share
- Pictures to edit
- Databases to query

and access from everywhere

Security Requirements

As from a local hard drive/server, one expects

- **Storage** guarantees
- **Privacy** guarantees
  - **confidentiality** of the data
  - **anonymity** of the users
  - **obliviousness** of the queries/processing

How to proceed?

Confidentiality vs Sharing & Computations

Classical Encryption allows to protect data

- the provider stores them without knowing them
- nobody can access them either, except the owner/target receiver

How to share the data?

How to compute on the data?

Broadcast Encryption

[Fiat-Naor - Crypto '94]

Broadcast Encryption

[Fiat-Naor - Crypto ‘94]

The sender chooses a target set

Broadcast Encryption

[Fiat-Naor - Crypto ‘94]

The sender chooses a target set

Broadcast Encryption

[Fiat-Naor - Crypto '94]

The sender chooses a target set

Users get **all-or-nothing** about the data

Broadcast Encryption

[Fiat-Naor - Crypto '94]

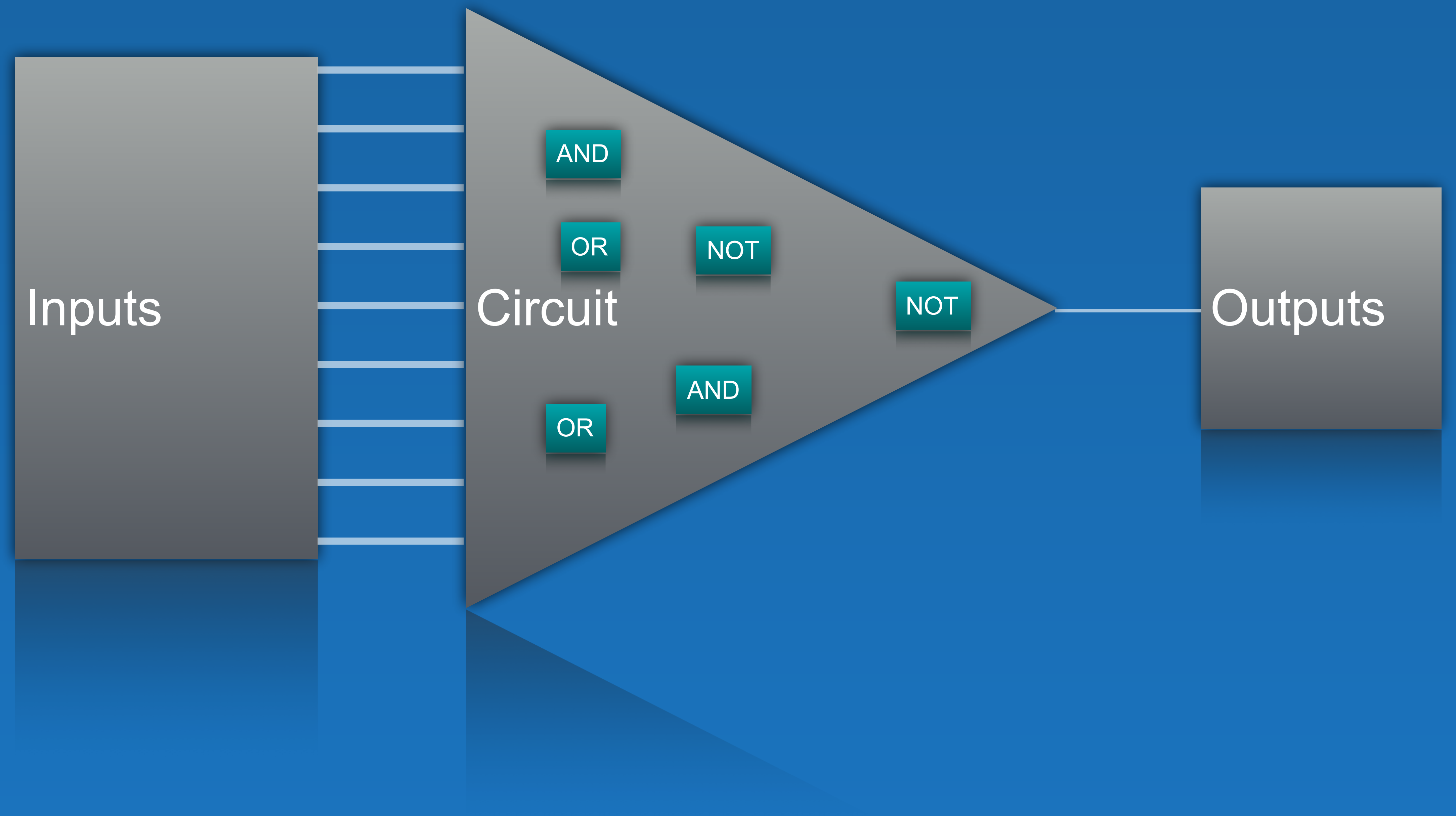
The sender chooses a target set

Users get **all-or-nothing** about the data

Sharing to a Target Set but No Computations!

Fully Homomorphic Encryption

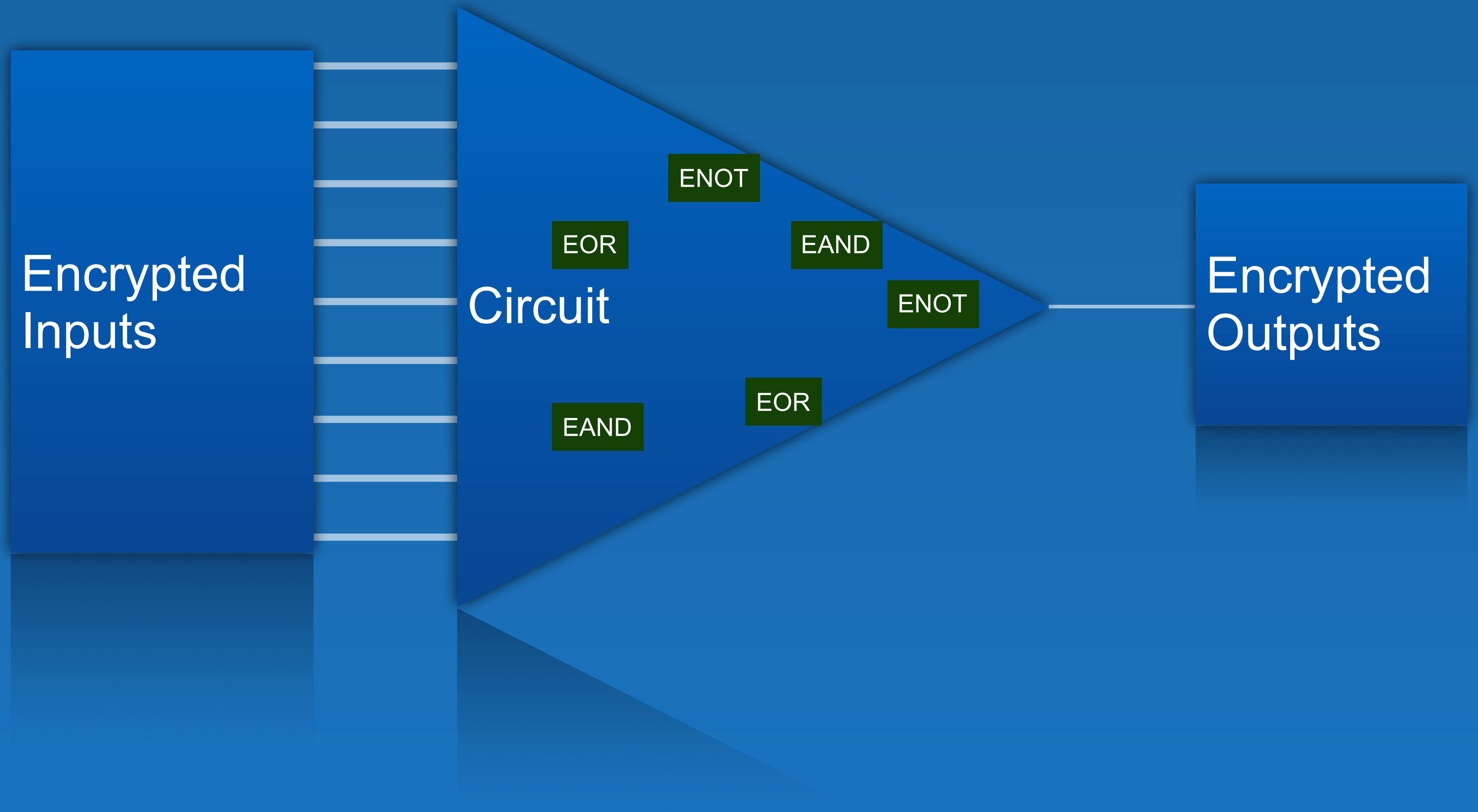
[Rivest-Adleman-Dertouzos - FOCS ’78]
[Gentry - STOC ’09]



Fully Homomorphic Encryption

[Rivest-Adleman-Dertouzos - FOCS '78]
[Gentry - STOC '09]

FHE allows any computations on encrypted data

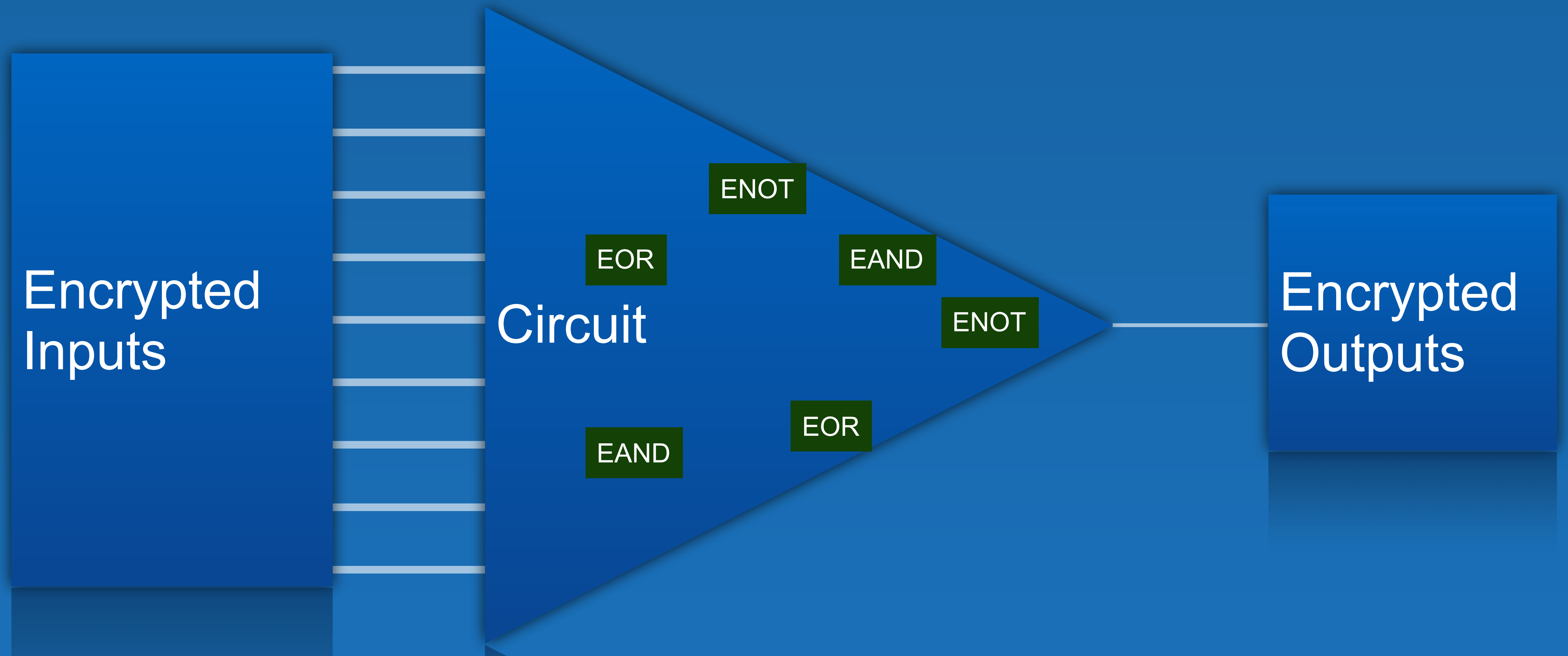


Fully Homomorphic Encryption

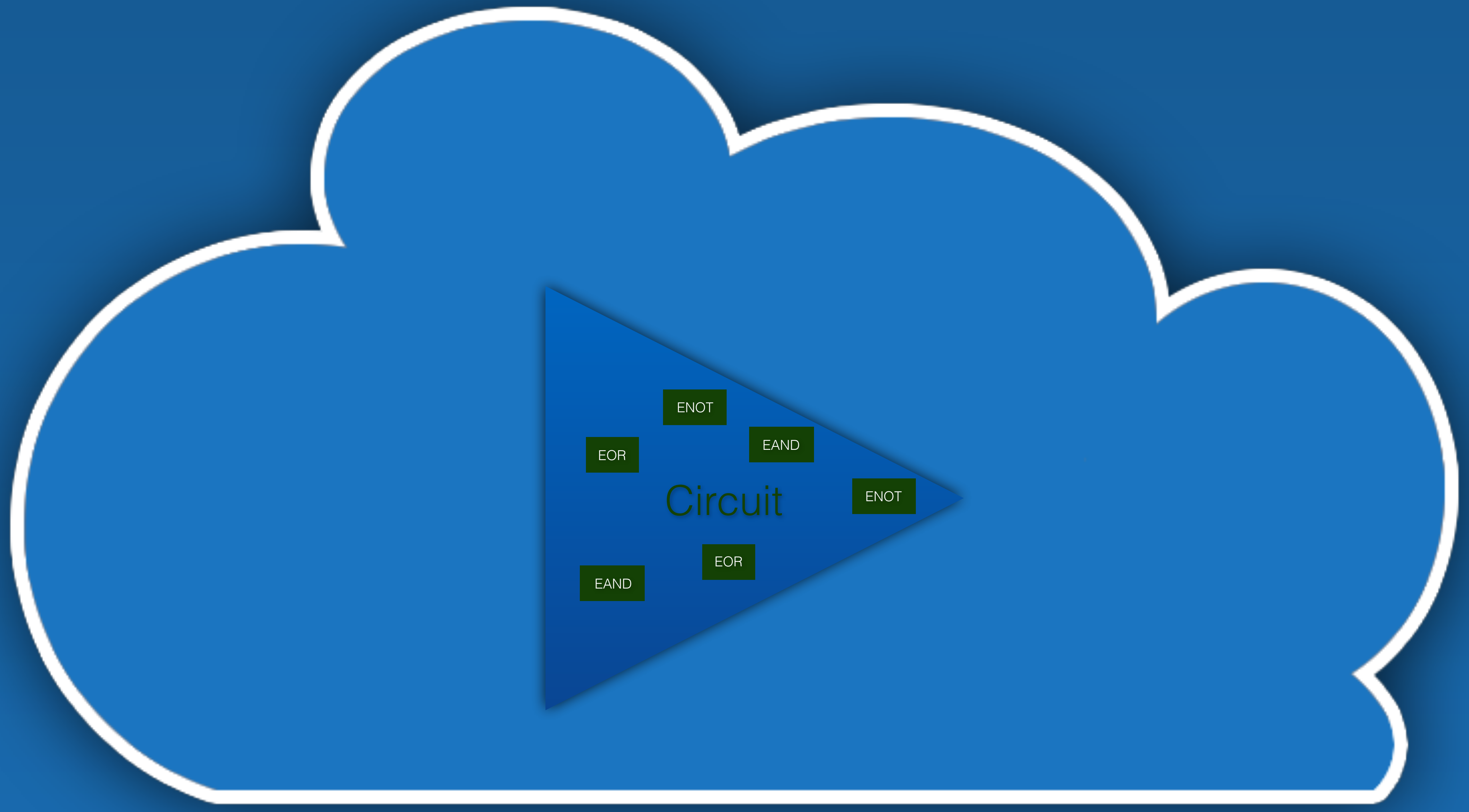
[Rivest-Adleman-Dertouzos - FOCS ’78]
[Gentry - STOC ’09]

FHE allows any computations on encrypted data

But the result is **encrypted** as the inputs!

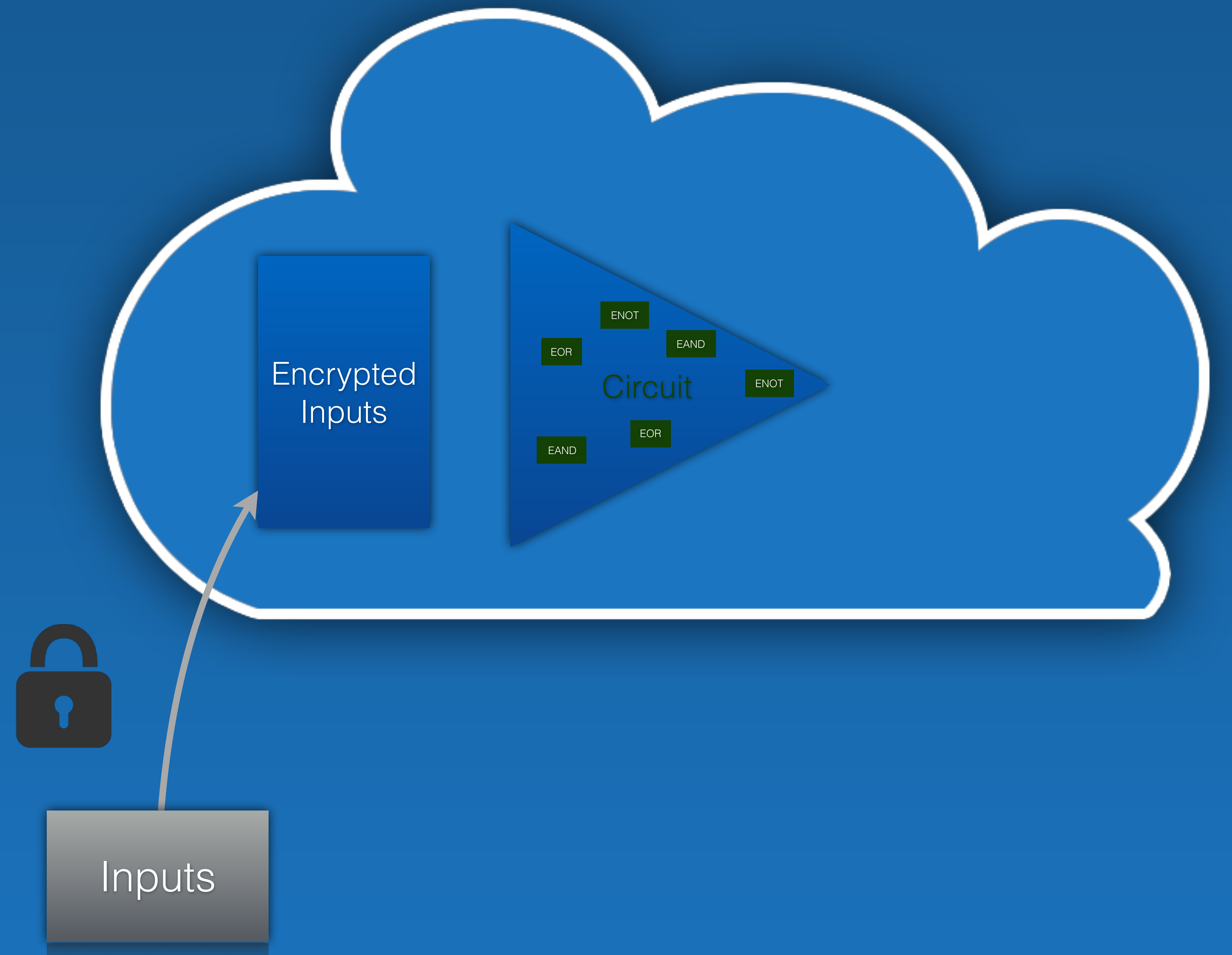


Outsourced Computations

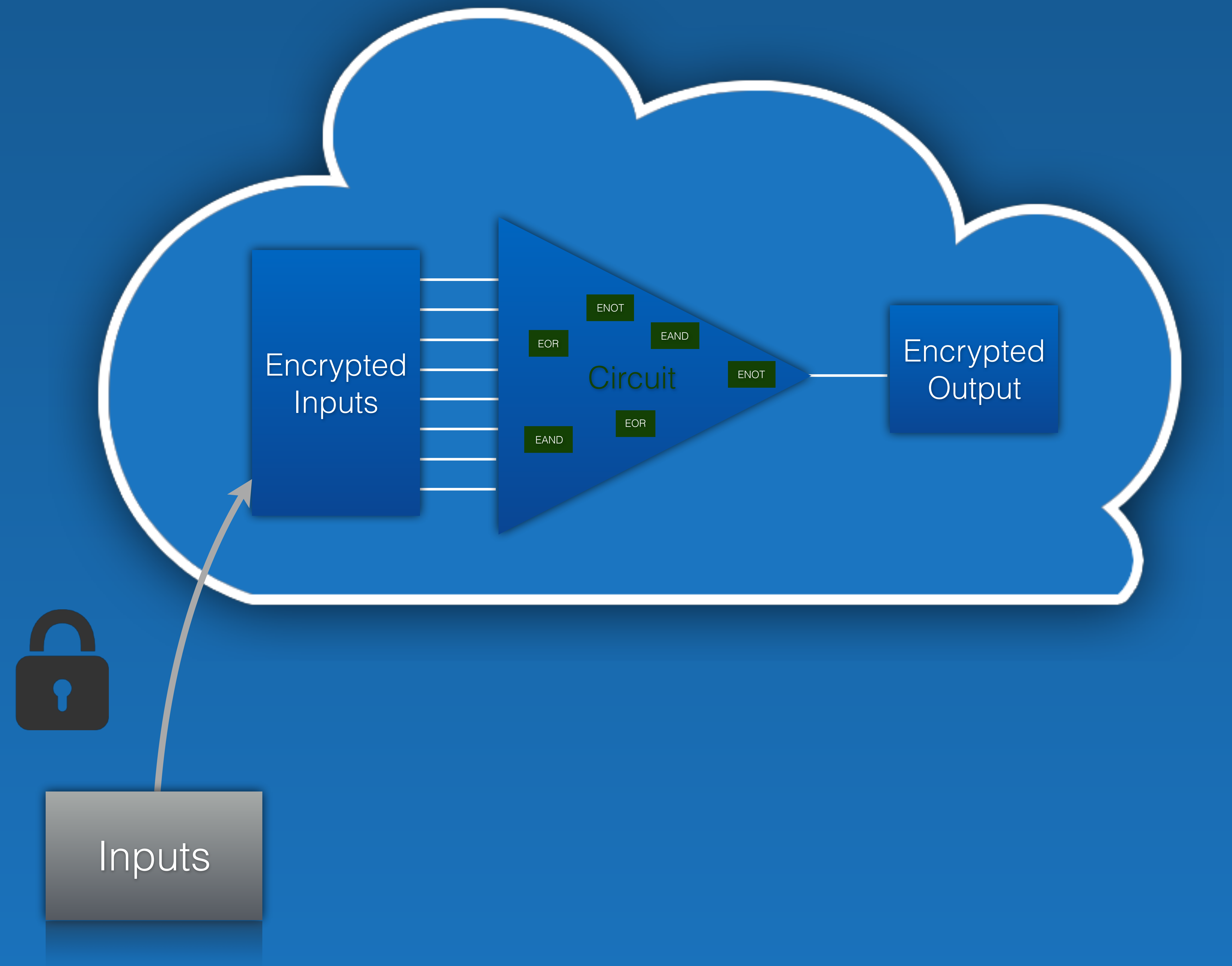


Inputs

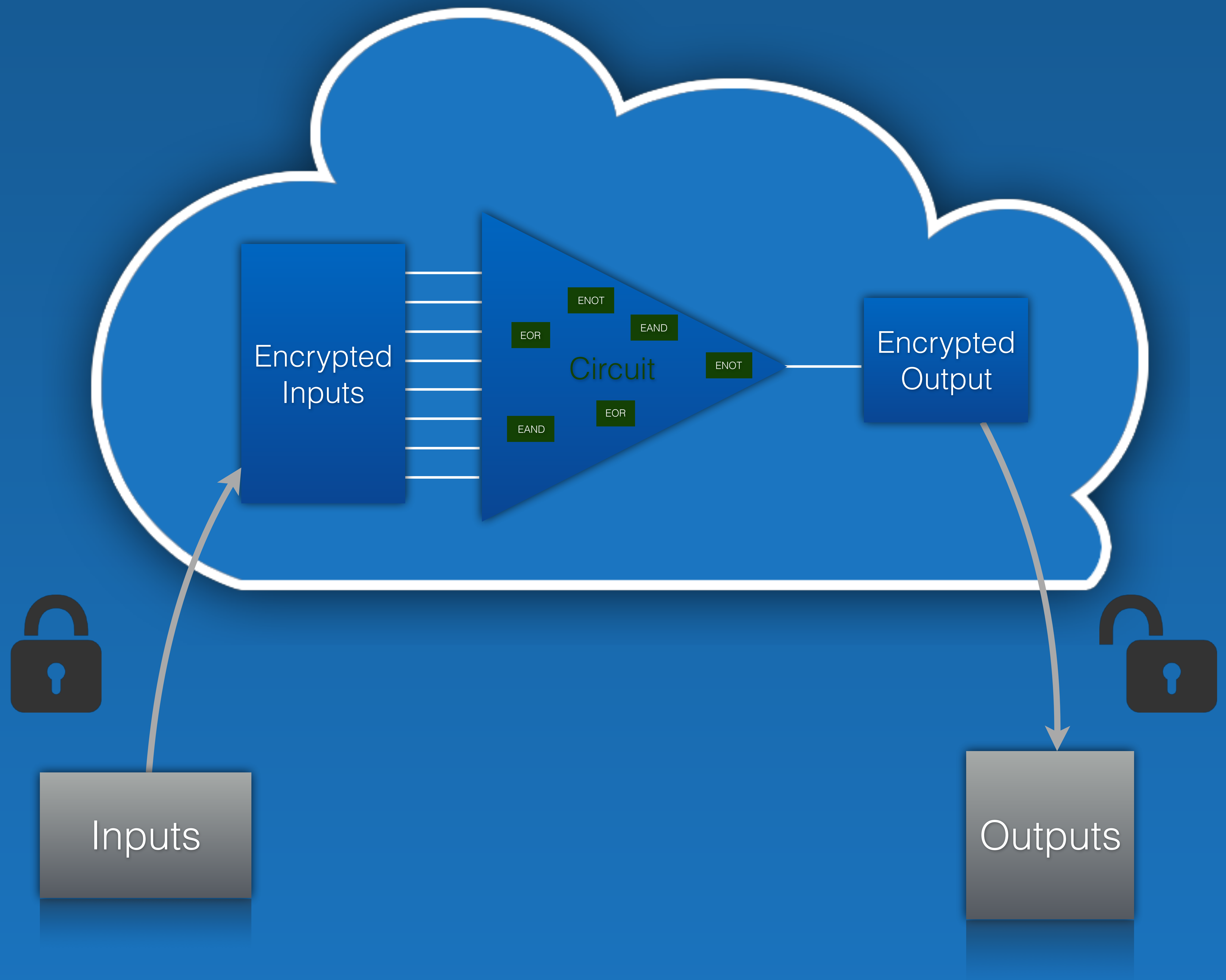
Outsourced Computations



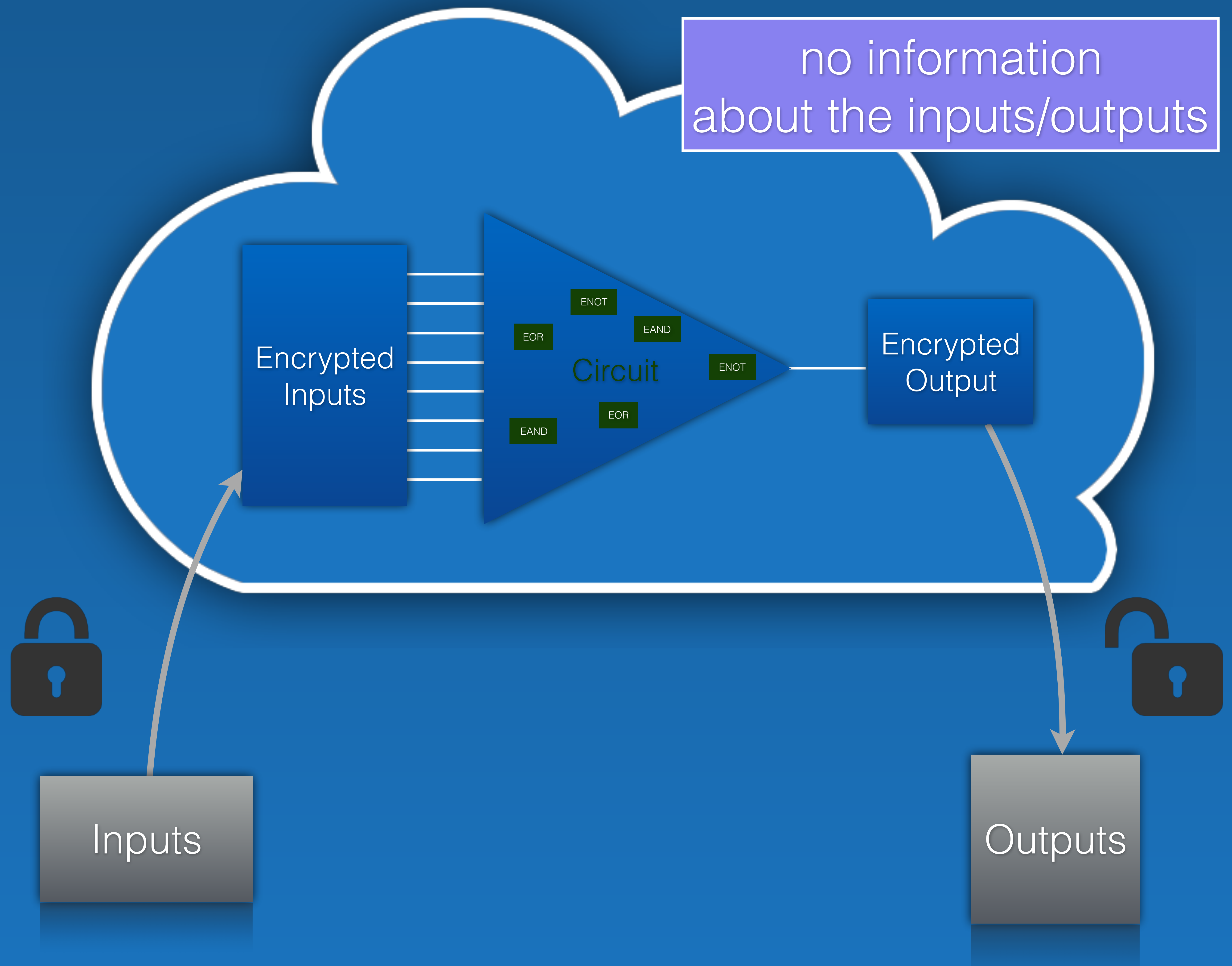
Outsourced Computations



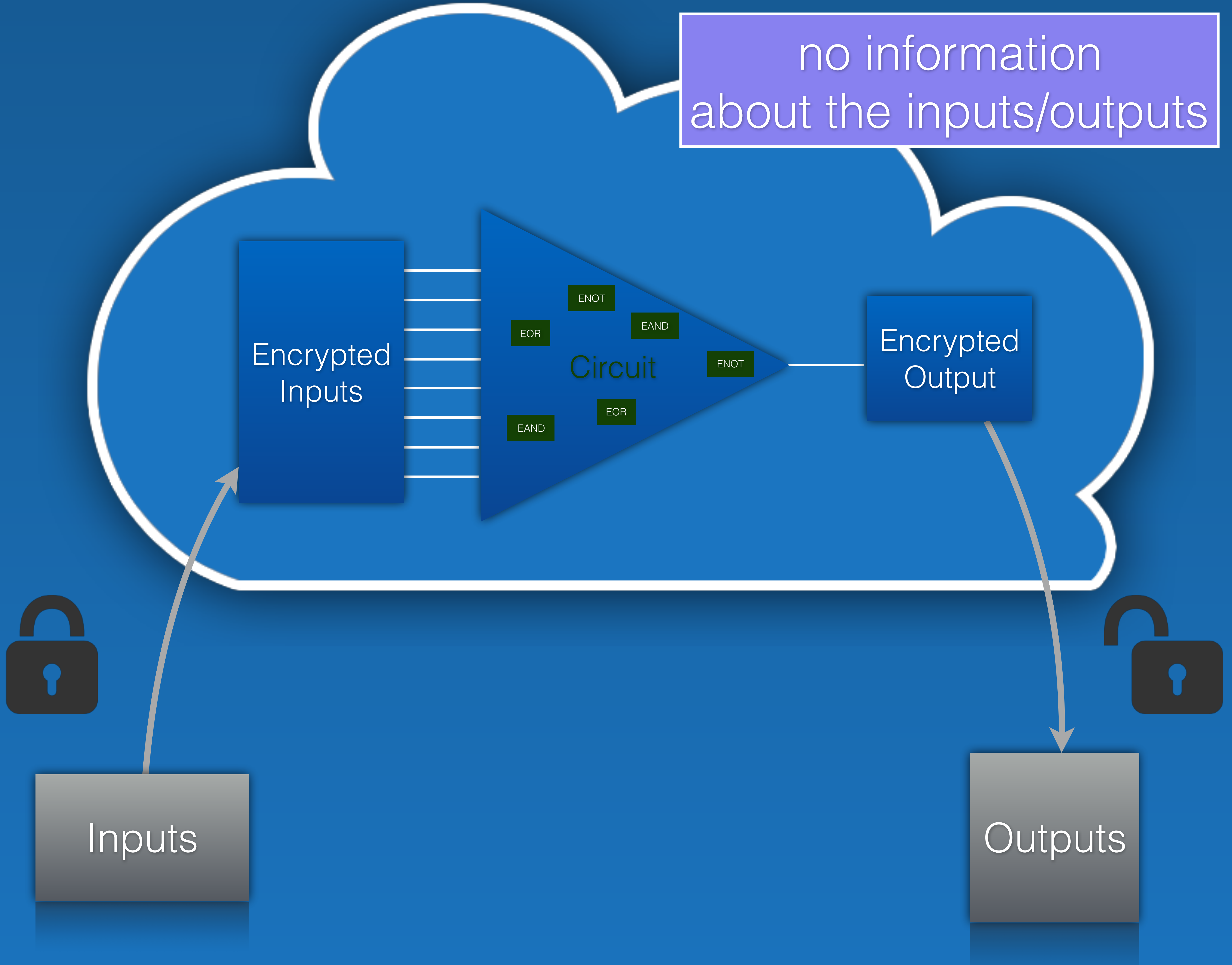
Outsourced Computations



Outsourced Computations



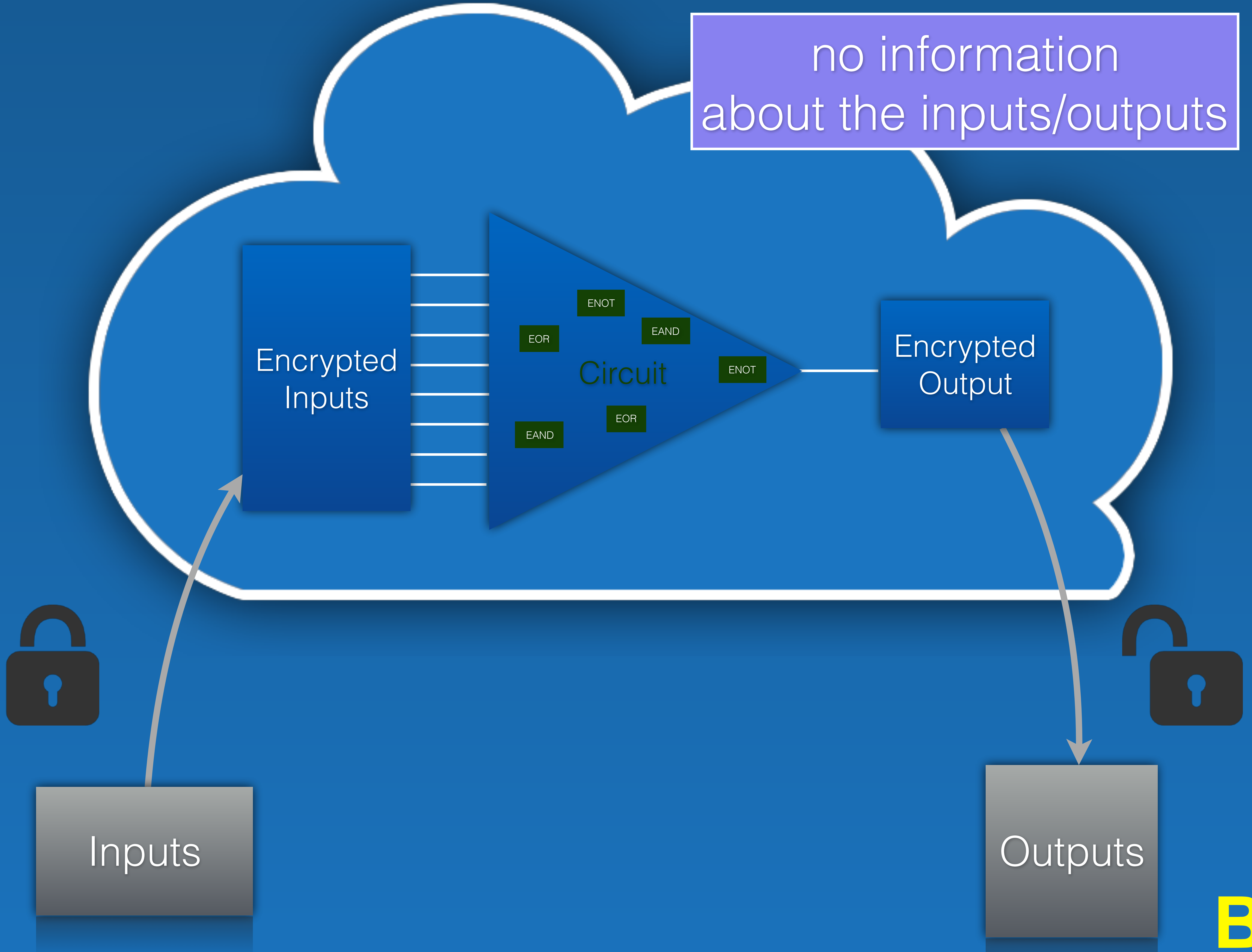
Outsourced Computations



FHE allows

- Any computation on private inputs
- Private « googling »

Outsourced Computations



FHE allows

- Any computation on private inputs
- Private « googling »

Computations

But No Controlled Sharing!

Functional Encryption

[Boneh-Sahai-Waters - TCC '11]

Functional Encryption

[Boneh-Sahai-Waters - TCC '11]

The authority generates functional decryption keys dk_f according to functions f

Functional Encryption

[Boneh-Sahai-Waters - TCC '11]

The authority generates functional decryption keys dk_f according to functions f

Functional Encryption

[Boneh-Sahai-Waters - TCC '11]

The authority generates functional decryption keys dk_f according to functions f

- From $C =$ **Encrypt**(x), **Decrypt**(dk_f, C) outputs $f(x)$

Functional Encryption

[Boneh-Sahai-Waters - TCC '11]

The authority generates functional decryption keys dk_f according to functions f

- From C = **Encrypt**(x), **Decrypt**(dk_f, C) outputs $f(x)$
- This allows **controlled sharing of data**

Result in clear
for a Specific Function
for Specific Users

Functional Encryption is Powerful

Functional Encryption allows access control:

- with $f_{\mathsf{id}}(x \,||\, y) = (\text{if } y = \mathsf{id}, \text{ then } x, \text{ else } \perp)$: **identity-based** encryption
- with $f_{\mathsf{G}}(x \,||\, y) = (\text{if } y \in \mathsf{G}, \text{ then } x, \text{ else } \perp)$: **broadcast** encryption

Functional Encryption allows computations:

- any function f: in theory, with *iO* (Indistinguishable Obfuscation)
- concrete functions: inner product

FE: Concrete Case

Student Name	English		CS		Math	
	Written	Spoken	Theory	Practice	Algebra	Analysis
Year 1						
Year 2						
Year 3						

FE: Concrete Case

Student Name	English		CS		Math	
	Written	Spoken	Theory	Practice	Algebra	Analysis
Year 1						
Year 2						
Year 3						

- For each student: transcript with all the grades

FE: Concrete Case

Student Name	English		CS		Math	
	Written	Spoken	Theory	Practice	Algebra	Analysis
Year 1						
Year 2						
Year 3						

Name	English	CS	Math
Year 1			
Year 2			
Year 3			

Student Name	English		CS		Math	
	Written	Spoken	Theory	Practice	Algebra	Analysis
Total						

Name	Total
Year 1	
Year 2	
Year 3	

Name	Total
3Years	

- For each student: transcript with all the grades
- Access to partial information for each student

FE: Concrete Case

Student Name	English		CS		Math	
	Written	Spoken	Theory	Practice	Algebra	Analysis
Year 1						
Year 2						
Year 3						

Class	English	CS	Math
Year 1			
Year 2			
Year 3			

Class	English		CS		Math	
	Written	Spoken	Theory	Practice	Algebra	Analysis
Total						

Class	Total
Year 1	
Year 2	
Year 3	

Class	Total
3Years	

- For each student: transcript with all the grades
- Access to partial information for each student
- And even global grades for the class

FE: Inner Product

[Abdalla-Bourse-De Caro-P. - PKC ’15 - EPrint 2015/017]

Cells of derived tables are linear combinations $\overrightarrow{a_i}$ of the grades $\overrightarrow{b}$ from the main table:

$$c_i = \sum_j a_{i,j} b_j = \overrightarrow{a_i} \cdot \overrightarrow{b}$$

- $\overrightarrow{b}$: vector of the private grades, encrypted in the main table
- $\overrightarrow{a_i}$: vector of the public coefficients for the cell c_i, defines f_i
- With ElGamal encryption:
  - computations modulo p
  - if grades, coefficients, and classes small enough: DLog computation

ElGamal Encryption

[ElGamal - IEEE TIT '85]

- ElGamal Encryption on $\mathbb{G} = \langle g \rangle$:
  Secret key: $s \in \mathbb{Z}_p$
  Public key: $h = g^s$
  Encryption: $c = (c_0 = g^r, c_1 = h^r \cdot m)$
  Decryption: $m = c_1 / c_0^s$
- Semantically secure under DDH in $\mathbb{G} = \langle g \rangle$
- Multiplicatively homomorphic
- Additive variant: m is replaced by g^m
  but requires discrete logarithm computation
- Encryption of vectors:
  with many h_i and the same randomness

FE: IP with ElGamal

[Abdalla-Bourse-De Caro-P. - PKC '15 - EPrint 2015/017]

Parameters:	a group $\mathbb{G} = \langle g \rangle$ of prime order p
Secret key:	$\vec{s} = (s_j)_j$, for random scalars in $\mathbb{Z}_p$
Public key:	$\vec{h} = (h_j = g^{s_j})_j$
Encryption:	$c = g^r$ and $\vec{C} = (C_j = h_j^r \cdot g^{x_j})_j$
	$D = \vec{f} \cdot \vec{C} = \prod_j C_j^{f_j}$
	$= g^{r \sum_j f_j s_j} g^{\sum_j f_j x_j} = g^{r \cdot \vec{f} \cdot \vec{s}} g^{\vec{f} \cdot \vec{x}}$
Functional key:	$dk_f = \sum_j f_j s_j = \vec{f} \cdot \vec{s}$
Decryption:	$D = c^{dk_f} \cdot g^m \longrightarrow m = \log_g(\vec{f} \cdot \vec{C} / c^{dk_f}) = \vec{f} \cdot \vec{x}$

FE: Limitations

- one key limits to one function on any vector 🙂
- a malicious player could ask many functional keys
  - n different keys reveal x
  - for the indistinguishability between two sets of vectors, the adversary is not allowed to ask keys that trivially tell them appart
    $\Rightarrow$ if n vectors in the sets, the adversary cannot ask any key! 😞
- a unique sender only can encrypt all the inputs 😞
  - Multi-Input Functional Encryption (MIFE) 🙂

[Goldwasser-Gordon-Goyal-Jain-Katz-Liu-Sahai-Shi-Zhou - Eurocrypt '14 - EPrint 2013/727 - EPrint 2013/774]

Multi-Client Functional Encryption

- Client $\mathbf{C}_i$ generates c_i = $\mathbf{E}(i,\boldsymbol{\lambda},x_i)$ for a label $\boldsymbol{\lambda}$ (or a time period)

  $\Rightarrow$ only one ciphertext for each index i and each label $\boldsymbol{\lambda}$

[Goldwasser-Gordon-Goyal-Jain-Katz-Liu-Sahai-Shi-Zhou - Eurocrypt ’14 - EPrint 2013/727 - EPrint 2013/774]

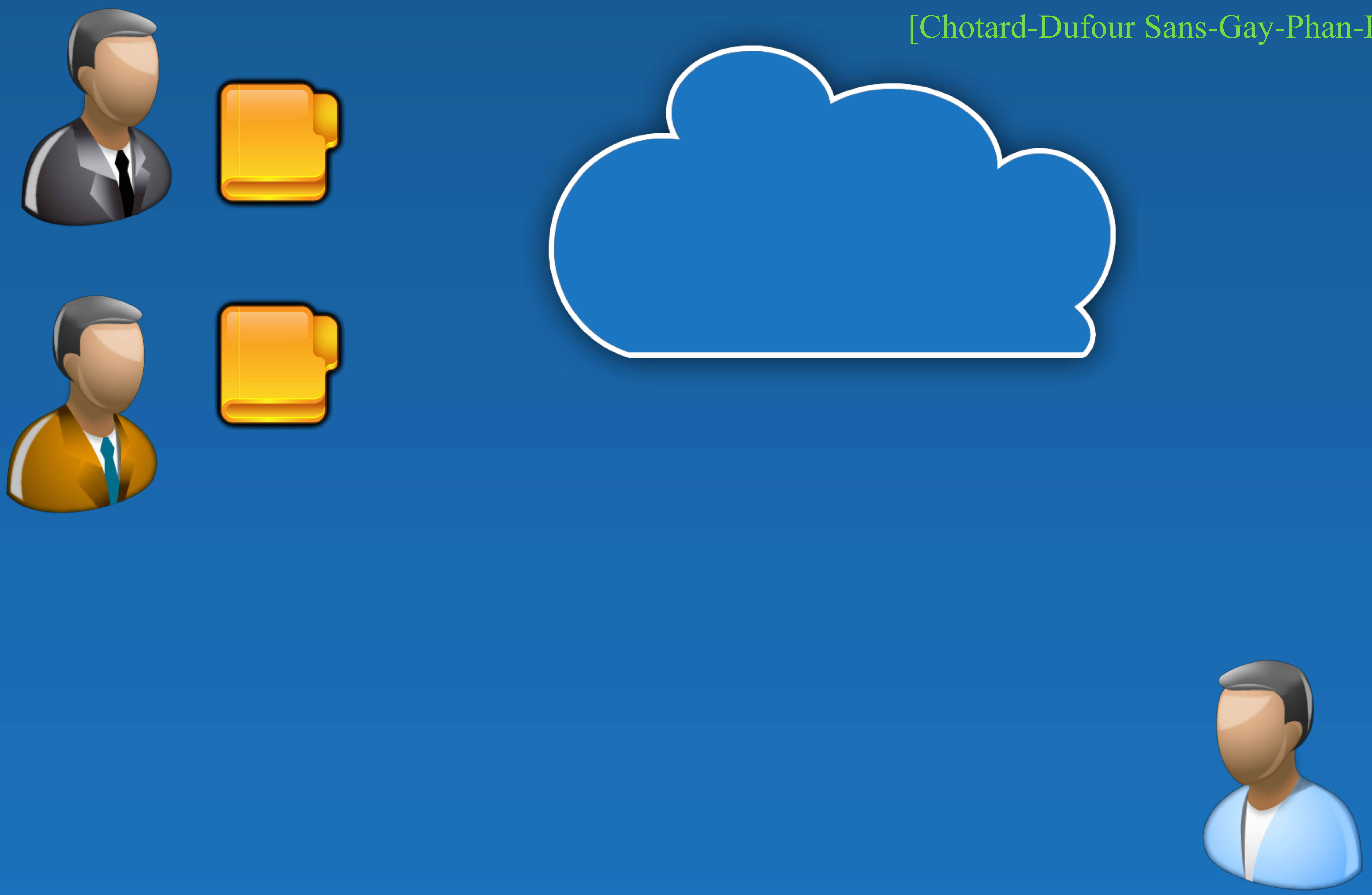
- Multi-User Inputs
- Private encryption limits attacks
- More reasonable security model 🙂
- But still a unique authority for the functional key generation 😞

Independent and Untrusted Clients

- Senders $(\mathbf{S}_i)_i$ provide sensitive inputs x_i (e.g. financial data) in an encrypted way under secret encryption keys ek_i
  $\rightarrow c_i = \mathbf{E}(ek_i,\lambda,x_i)$ for a label λ (or every time period)
- For some function f, an aggregator proposes, as a service, to communicate the aggregation $f(x)$ for every label λ, thanks to a functional decryption key dk_f
- The senders want to keep control on f
  $\rightarrow dk_f$ is generated by the senders

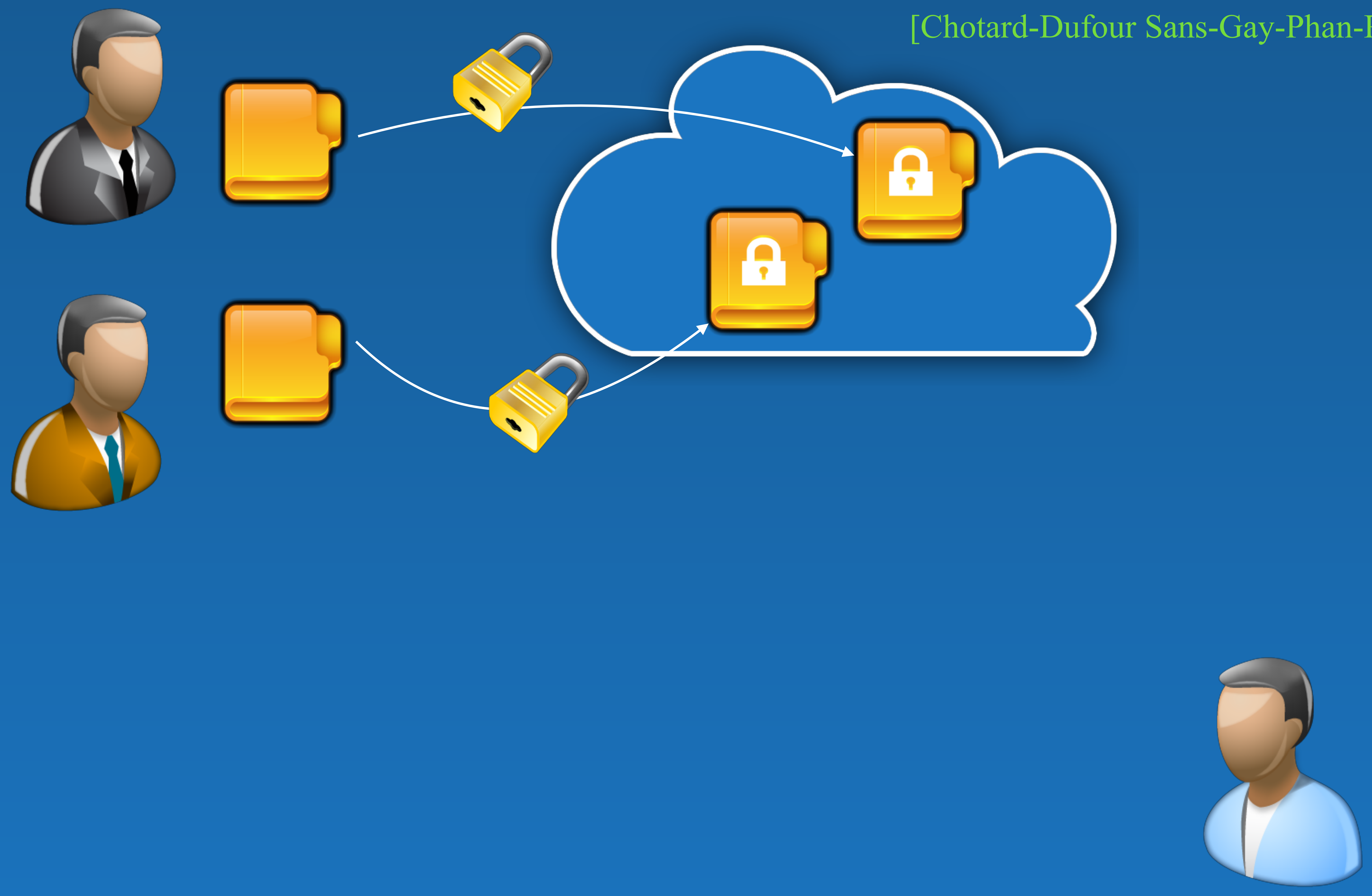
Decentralized MCFE

[Chotard-Dufour Sans-Gay-Phan-P. - Asiacrypt ’18 - EPrint 2017/989]



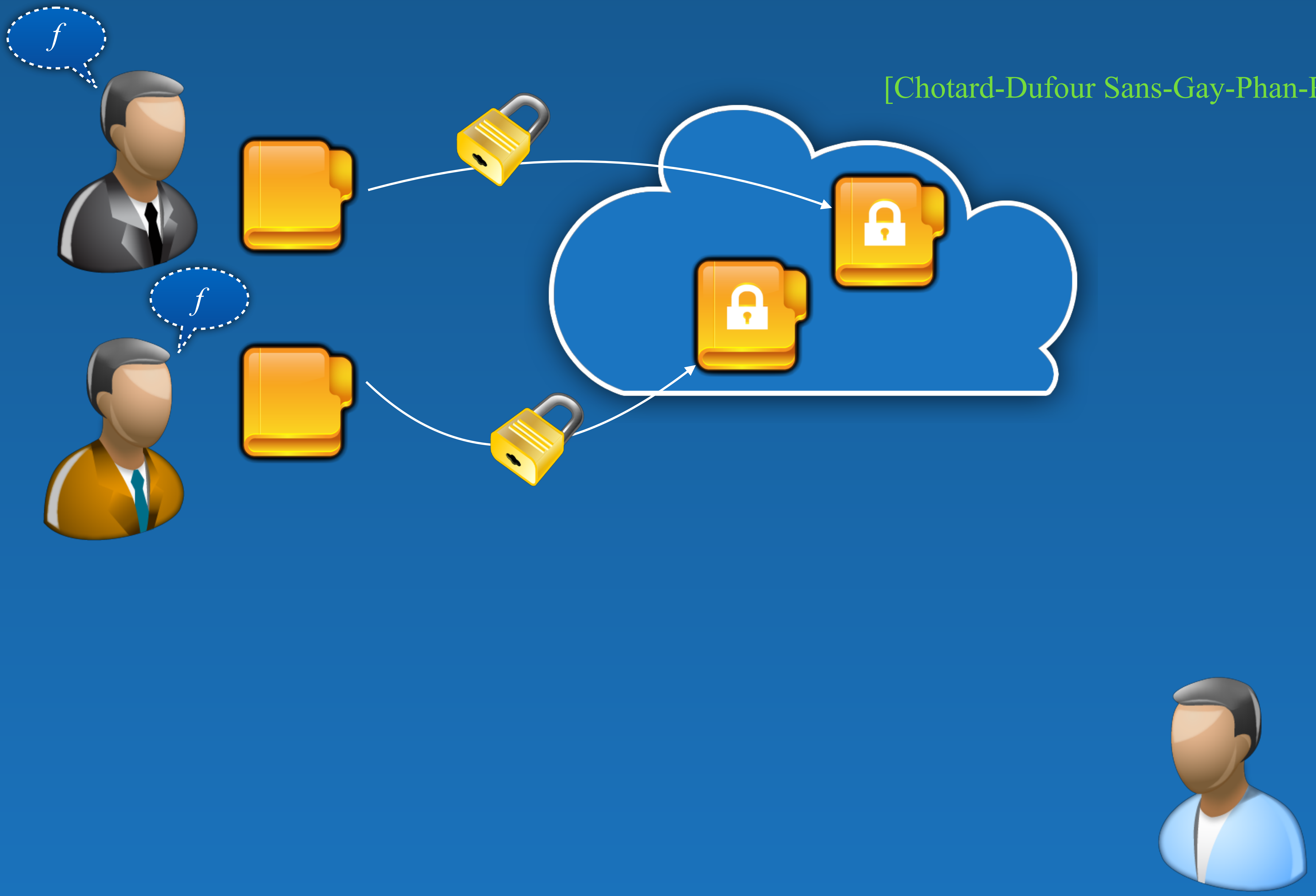
Decentralized MCFE

[Chotard-Dufour Sans-Gay-Phan-P. - Asiacrypt ’18 - EPrint 2017/989]



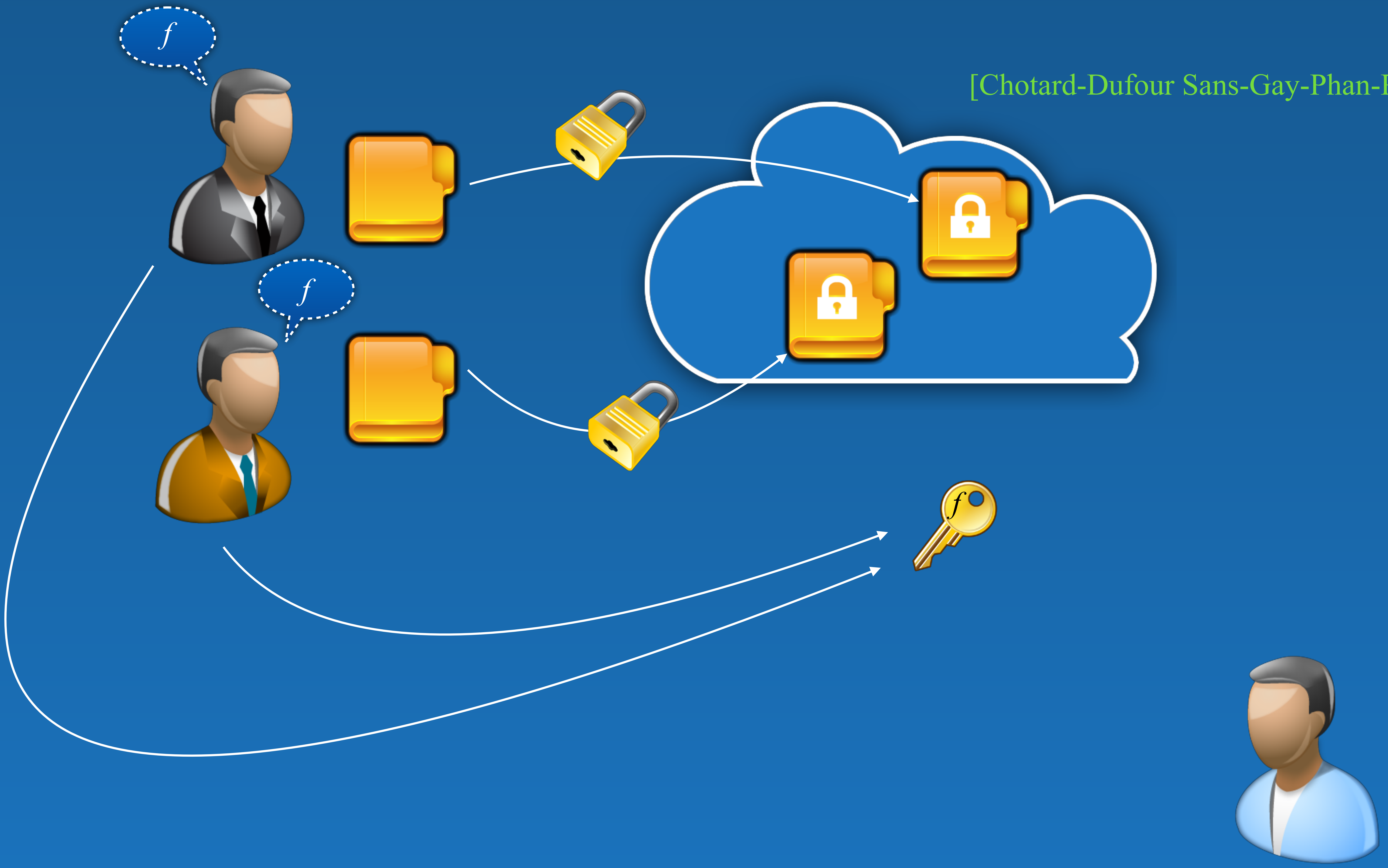
Decentralized MCFE

[Chotard-Dufour Sans-Gay-Phan-P. - Asiacrypt '18 - EPrint 2017/989]



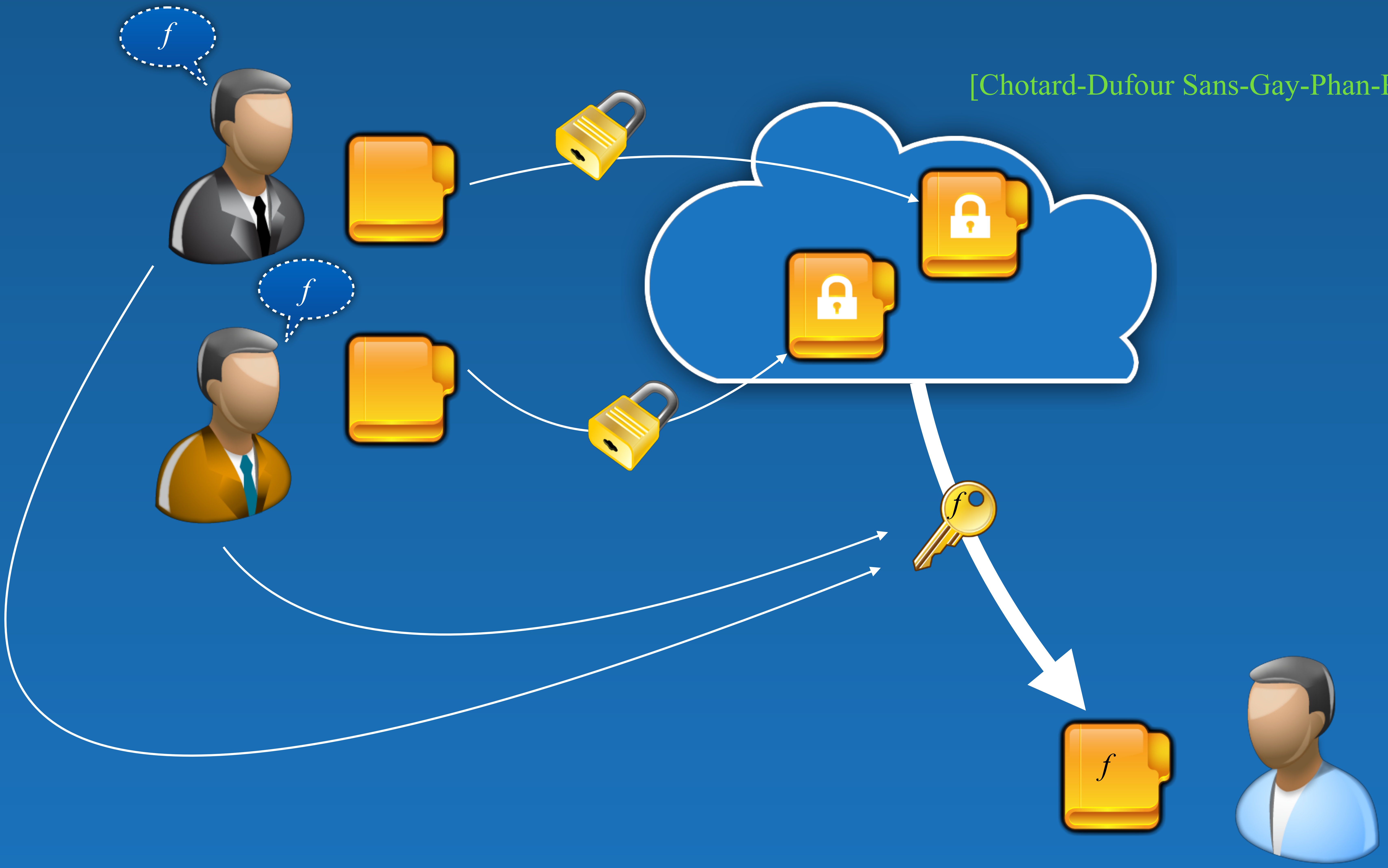
Decentralized MCFE

[Chotard-Dufour Sans-Gay-Phan-P. - Asiacrypt ’18 - EPrint 2017/989]



Decentralized MCFE

[Chotard-Dufour Sans-Gay-Phan-P. - Asiacrypt '18 - EPrint 2017/989]



Decentralized MCFE

[Chotard-Dufour Sans-Gay-Phan-P. - Asiacrypt ’18 - EPrint 2017/989]

- **Setup**() → secret key sk_i and encryption key ek_i for each sender $\mathbf{S}_i$
- **Encrypt**(ek_i,$\boldsymbol{\lambda}$,x_i) → c_i = **E**(ek_i,$\boldsymbol{\lambda}$,x_i) for the label $\boldsymbol{\lambda}$
- **DKeyGen**($(sk_i)_i$,f) → dk_f
- **Decrypt**(dk_f,$\boldsymbol{\lambda}$,**C**) → $f(\boldsymbol{x})$ if **C** = (c_i = **E**(ek_i,$\boldsymbol{\lambda}$,x_i))$_i$

- **Encrypt/Decrypt** are non-interactive algorithms
- **Setup/DKeyGen** are interactive protocols between the senders
- **DKeyGen** should be a **one-round** protocol only

FE: IP with ElGamal

[Abdalla-Bourse-De Caro-P. - PKC '15 - EPrint 2015/017]

Parameters:	a group $\mathbb{G} = \langle g \rangle$ of prime order p
Secret key:	$\vec{s} = (s_j)_j$, for random scalars in $\mathbb{Z}_p$
Public key:	$\vec{h} = (h_j = g^{s_j})_j$
Encryption:	$c = g^r$ and $\vec{C} = (C_j = h_j^r \cdot g^{x_j})_j$
	$D = \vec{f} \cdot \vec{C} = \prod_j C_j^{f_j}$
	$= g^{r \sum_j f_j s_j} g^{\sum_j f_j x_j} = g^{r \cdot \vec{f} \cdot \vec{s}} g^{\vec{f} \cdot \vec{x}}$
Functional key:	$dk_f = \sum_j f_j s_j = \vec{f} \cdot \vec{s}$
Decryption:	$D = c^{dk_f} \cdot g^m \longrightarrow m = \log_g(\vec{f} \cdot \vec{C} / c^{dk_f}) = \vec{f} \cdot \vec{x}$

Because of the common r in the ciphertext, a unique sender must encrypt the full vector

MCFE: IP with ElGamal

[Chotard-Dufour Sans-Gay-Phan-P. - Asiacrypt '18 - EPrint 2017/989]

Parameters: $\mathbb{G} = \langle g \rangle$ of prime order p, hash function $\mathcal{H}$

Encryption/Secret key: $ek_i = sk_i = s_i$, for random scalar in $\mathbb{Z}_p$

Encryption: $C_i = \mathcal{H}(\lambda)^{s_i} \cdot g^{x_i}$

$$D = \vec{f} \cdot \vec{C} = \prod_i C_i^{f_i} = \mathcal{H}(\lambda)^{\sum_i f_i s_i} g^{\sum_i f_i x_i} = \mathcal{H}(\lambda)^{\vec{f}\cdot\vec{s}} g^{\vec{f}\cdot\vec{x}}$$

Functional key: $dk_f = \sum_i f_i s_i = \vec{f} \cdot \vec{s}$

Decryption: $D = \mathcal{H}(\lambda)^{dk_f} \cdot g^m \longrightarrow m = \log_g(\vec{f} \cdot \vec{C} / \mathcal{H}(\lambda)^{dk_f}) = \vec{f} \cdot \vec{x}$

Encryption can be performed by independent senders

DMCFE: IP with ElGamal

[Chotard-Dufour Sans-Gay-Phan-P. - Asiacrypt '18 - EPrint 2017/989]

Functional key: $dk_f = \sum_i f_i s_i = \vec{f} \cdot \vec{s} = \vec{1} \cdot \vec{X}$ where $\vec{X} = (X_i = f_i s_i)_i$

- The senders can encrypt $(X_i{=}f_i\, s_i)_i$ under another IP-MCFE and the label f
- The aggregator knows the functional key for $(1,\ldots,1)$
- From the ciphertext of $(X_i{=}f_i\, s_i)_i$, it can extract dk_f

- This would work with a perfect IP-MCFE: any plaintext can be decrypted 🙂
- Here, only small plaintexts can be decrypted: dk_f is large! 😞

DMCFE: IP with Pairings

[Chotard-Dufour Sans-Gay-Phan-P. - Asiacrypt '18 - EPrint 2017/989]

Bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$

- Two IP-MCFE: **E**$_1$ in $\mathbb{G}_1$ and **E**$_2$ in $\mathbb{G}_2$
- The senders encrypt the messages x_i with **E**$_1$
- The senders encrypt the functional key shares X_i with **E**$_2$
- The aggregator knows the functional key for $(1,\ldots,1)$ in **E**$_2$ $\to$ it gets $g_2^{dk_f}$
- From $g_2^{dk_f}$ and ciphertexts of x_i with **E**$_1$ in $\mathbb{G}_1$ $\to$ one gets $g_T^{\boldsymbol{f} \cdot \boldsymbol{x}}$

DMCFE: IP with Pairings

[Chotard-Dufour Sans-Gay-Phan-P. - Asiacrypt '18 - EPrint 2017/989]

Bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$

- Two IP-MCFE: $\mathbf{E}_1$ in $\mathbb{G}_1$ and $\mathbf{E}_2$ in $\mathbb{G}_2$
- The senders encrypt the messages x_i with $\mathbf{E}_1$
- The senders encrypt the functional key shares X_i with $\mathbf{E}_2$
- The aggregator knows the functional key for $(1,\ldots,1)$ in $\mathbf{E}_2 \to$ it gets g_2^{dkf}
- From g_2^{dkf} and ciphertexts of x_i with $\mathbf{E}_1$ in $\mathbb{G}_1 \to$ one gets $g_T^{\boldsymbol{f}.\boldsymbol{x}}$

The discrete logarithm is small: can be extracted!

DMCFE: IP with Pairings

[Chotard-Dufour Sans-Gay-Phan-P. - Asiacrypt ’18 - EPrint 2017/989]

Our Decentralised Multi-Client Functional Encryption:

- Security
  - with Adaptive Corruptions of the Clients/Senders
  - under the classical SXDH assumption
- Efficiency
  - **Setup**: generation of the functional key for $(1,\ldots,1)$
  - **DKeyGen** protocol: just one ciphertext sent by each sender

Machine Learning and Encrypted Data

- **Fully Homomorphic Encryption**
  - **Outsourced Machine Learning**
    - Build an encrypted model from encrypted data (for the owner of the data)
    - Then classify encrypted data using this encrypted model
  - **Outsourced Classification**: encrypted inputs and **encrypted output**
    - Appropriate choice of the model [Bourse-Minelli-Minihold-Paillier - Crypto ’18 - EPrint 2017/1114]
    - MNIST Dataset: 96% of accuracy in less than 2 seconds
      with Discretized Neural Networks with 100 neurons
- **Functional Encryption** [Dufour Sans-Gay-P. - EPrint 2018/206]
  - Classification on encrypted data: encrypted inputs but **result in clear**
    - Inner products lead to linear model, but possible extension to quadratic model
    - MNIST Dataset: 97% of accuracy in less than 5 seconds,
      with a Quadratic Model

Conclusion

- **Functional Encryption**
  - Ideal functionalities on encrypted data with result in clear
  - Authority-based functionality
  - Encrypted inputs from a unique sender
- **DMCFE**
  - Aggregation of multi-source encrypted inputs
  - Functionality under control of the senders

Encrypted inputs and result in clear