Authenticating Computation on Groups: New Homomorphic Primitives and Applications

Dario Catalano
Università di Catania

Antonio Marcedone
Cornell University

Orazio Puglisi
Università di Catania

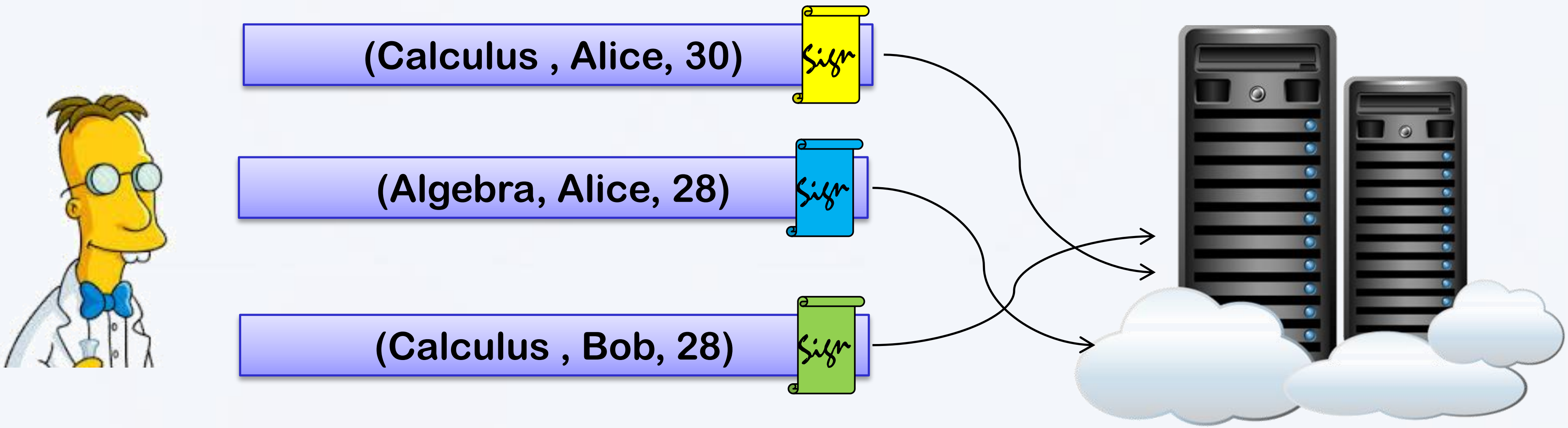
Outline

- Delegating computation on authenticated data
  - Motivating example
  - Linearly Homomorphic Signature
  - Authenticated Encryption

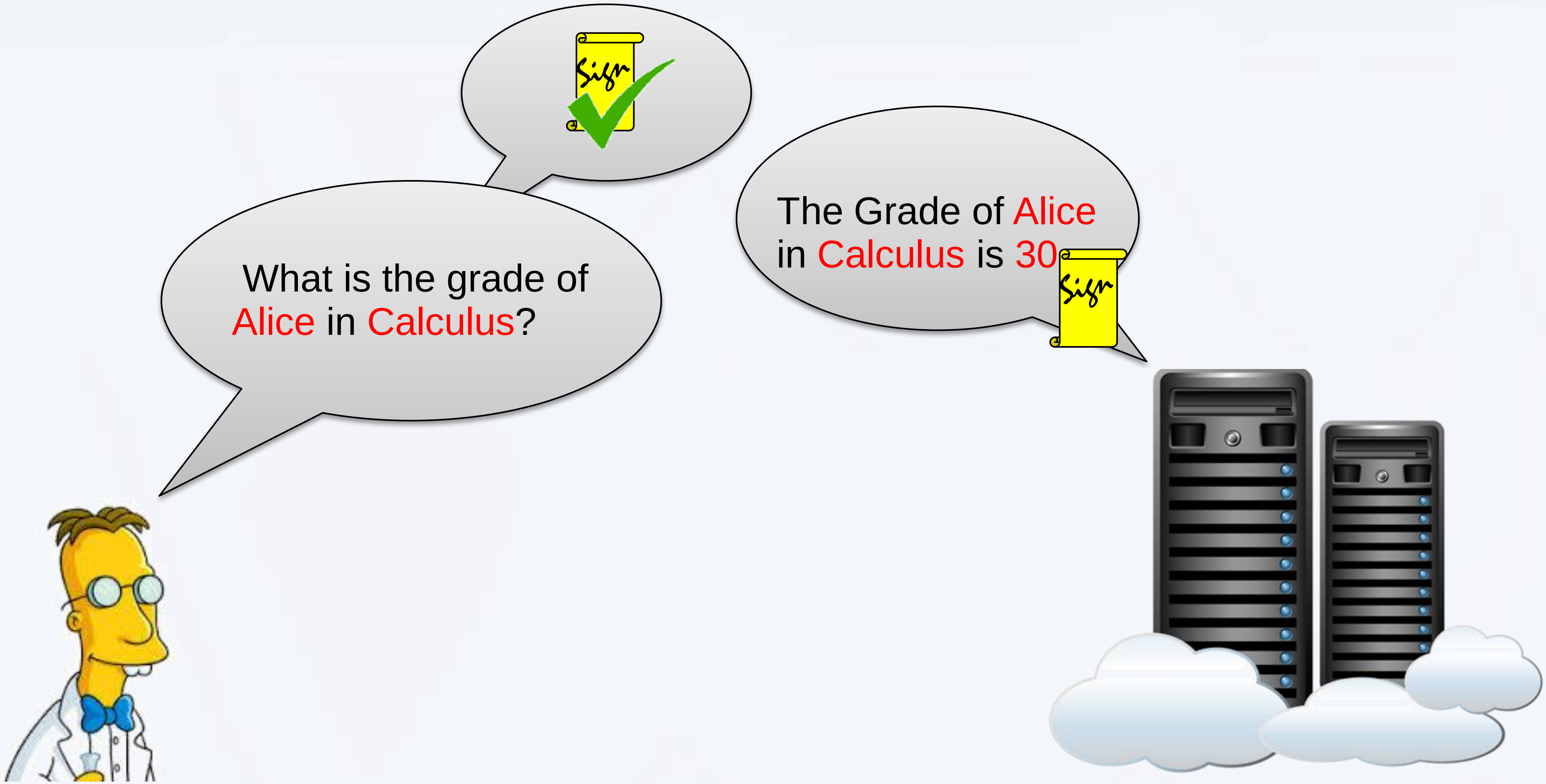
- Linearly Homomorphic Authenticated Encryption with Public Verifiability (LAEPuV)
  - Definition and security
  - Generic Construction outline and Instantiation

- Other results

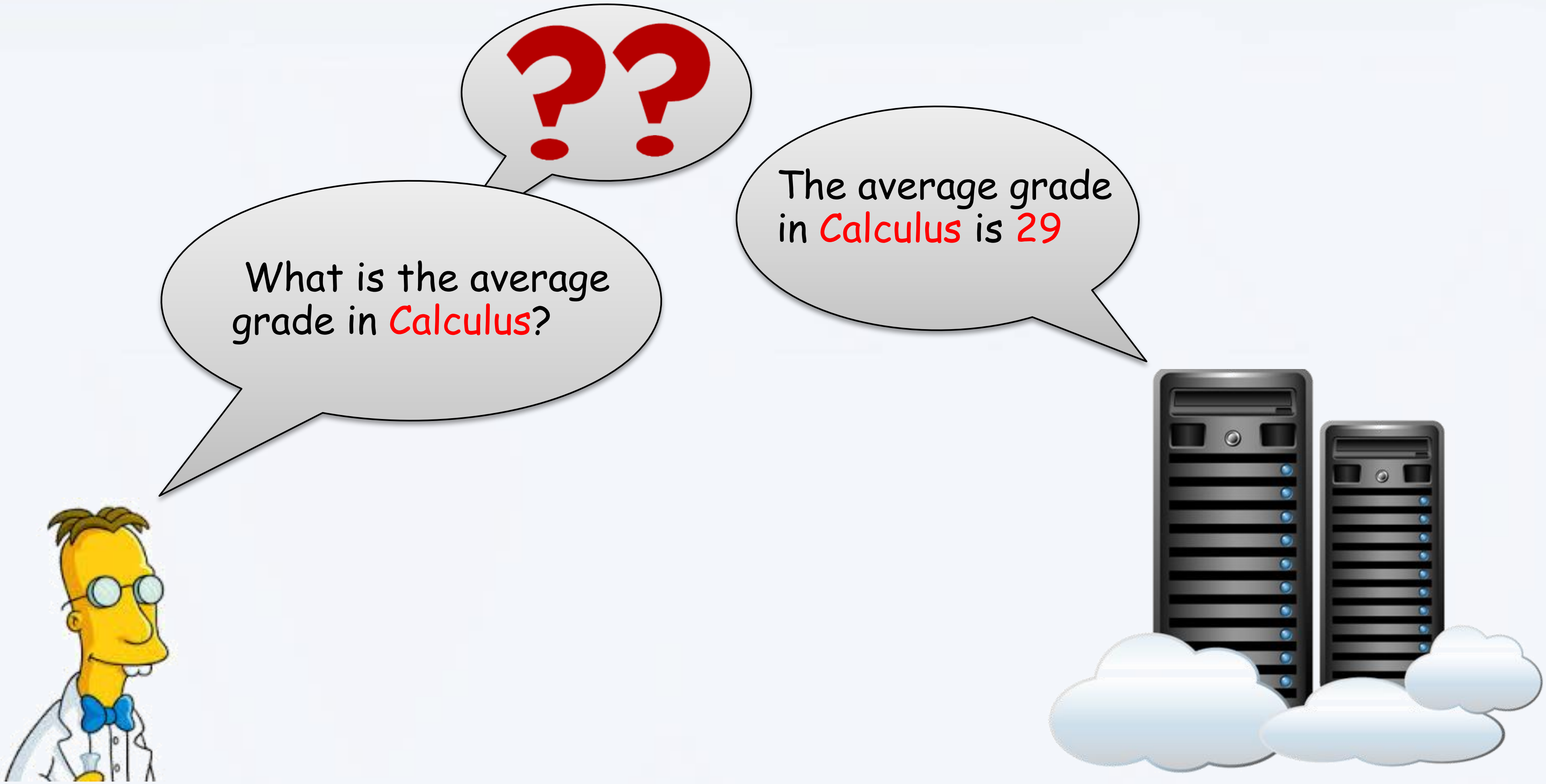
Delegating Computation on Authenticated Data



Delegating Computation on Authenticated Data



Delegating Computation on Authenticated Data



Linearly Homomorphic Signatures

[Desmedt93], [JMSW02], [BFKW09]

$(M_1, \mathrm{Sign}(sk, M_1))$
$(M_2, \mathrm{Sign}(sk, M_2))$
.....
$(M_n, \mathrm{Sign}(sk, M_n))$

COMBINE(PK,f,...)

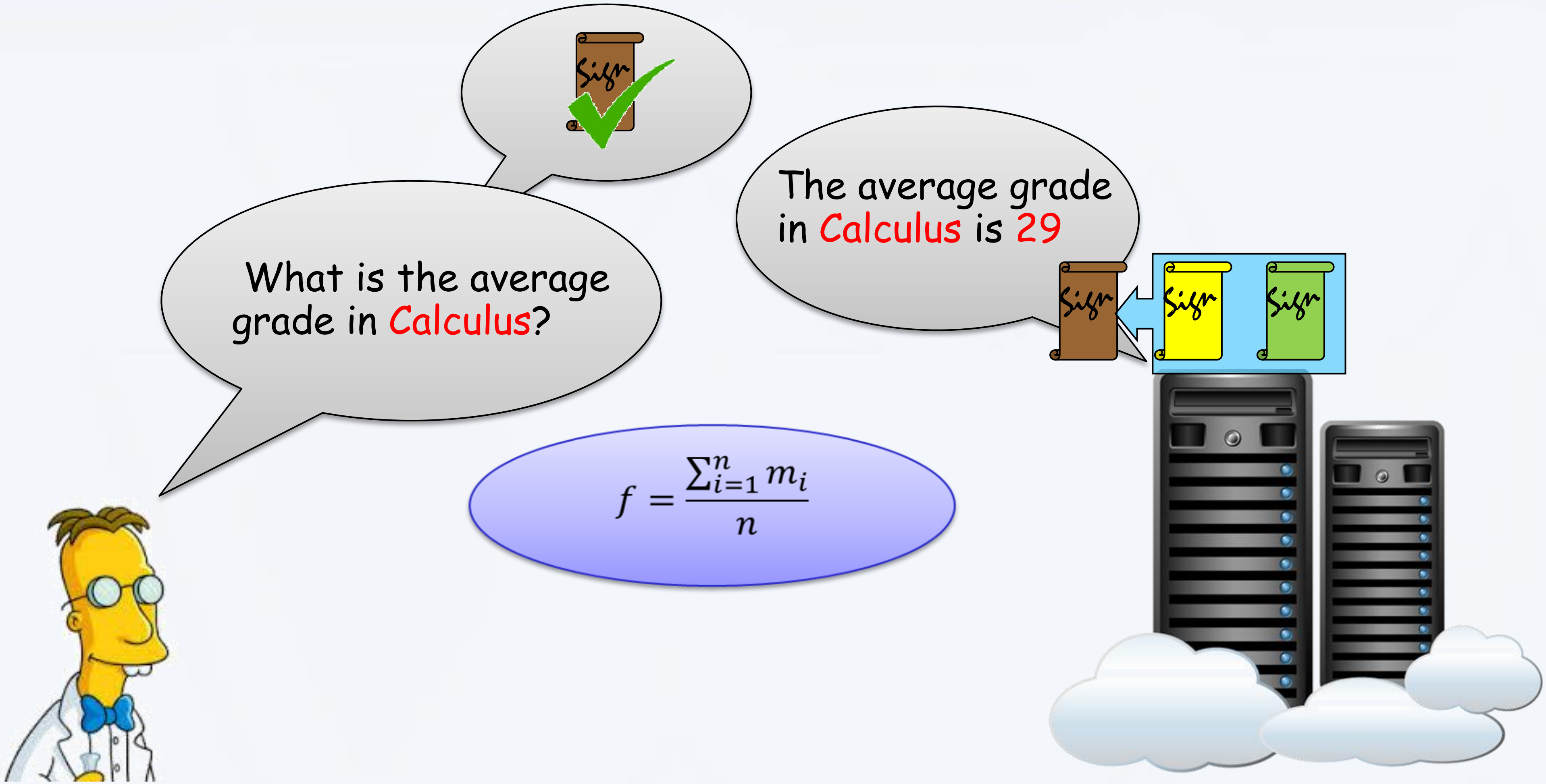
$(f(M_1,\ldots,M_n),$
$\mathrm{Sign}(sk, f(M_1,\ldots,M_n)))$

- Verification is done w.r.t. a function f:

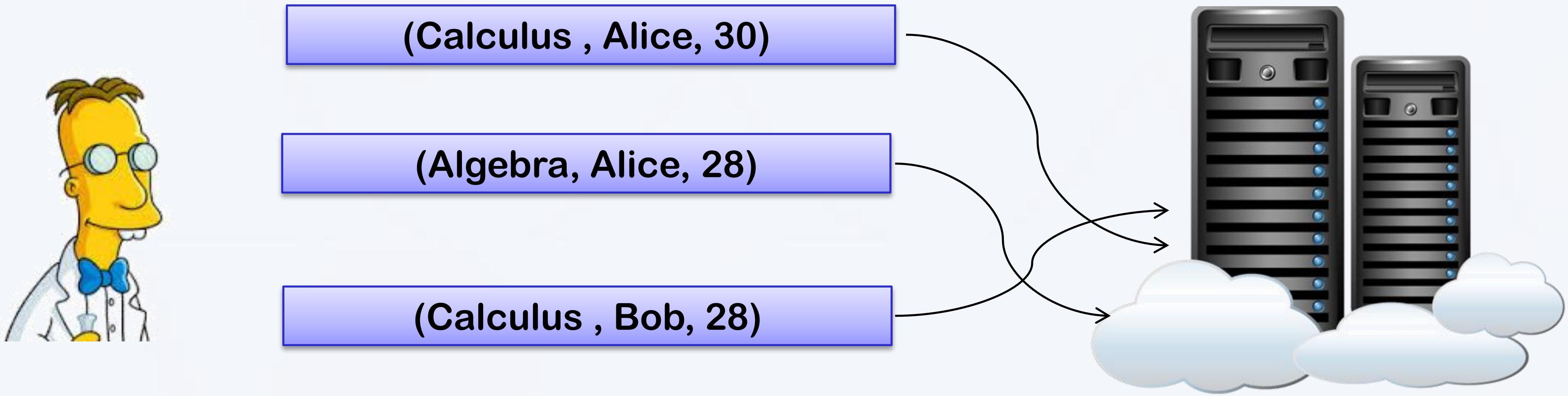
  Ver(pk, f, M, σ)

- The signatures must be succinct (indipendent on the number of messages).

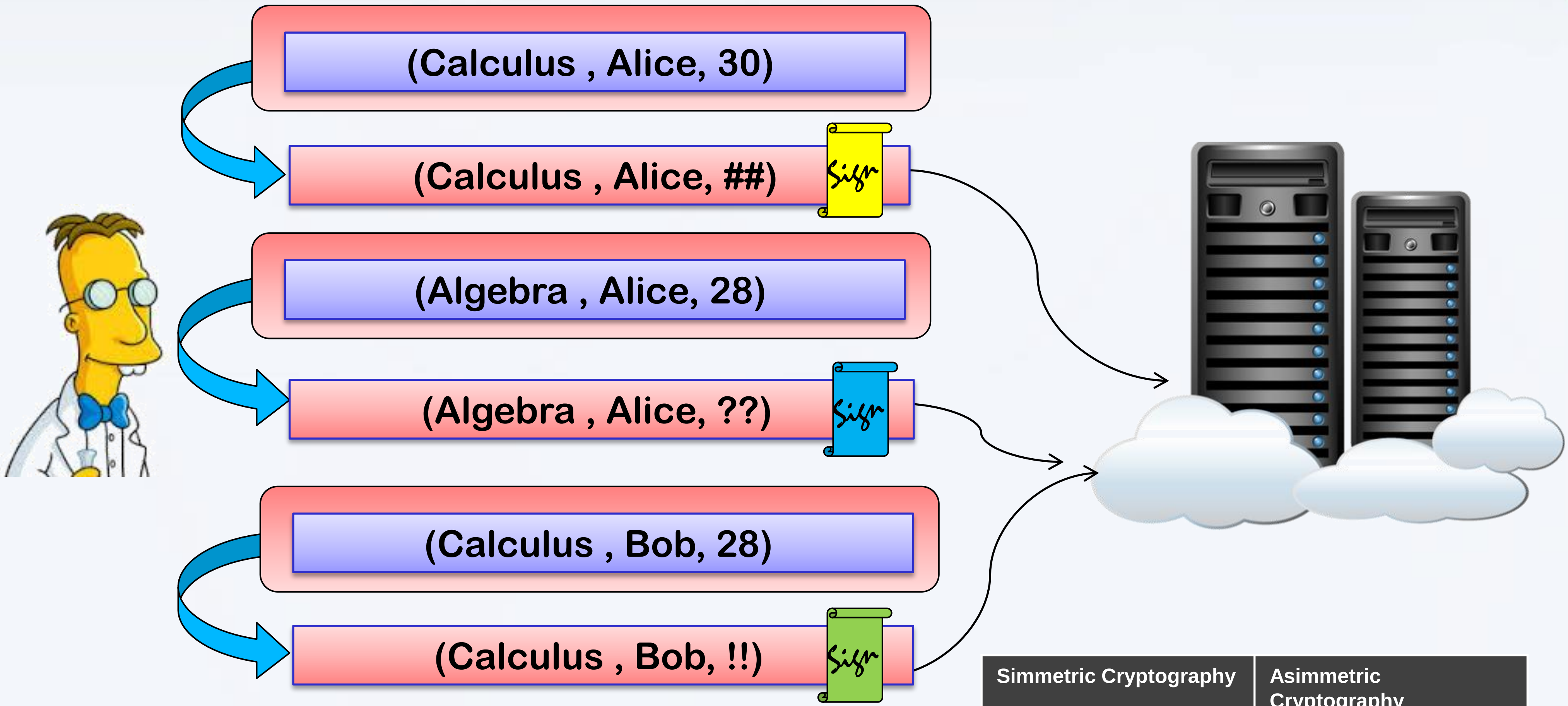
Delegating Computation on Authenticated Data



Authenticated Encryption

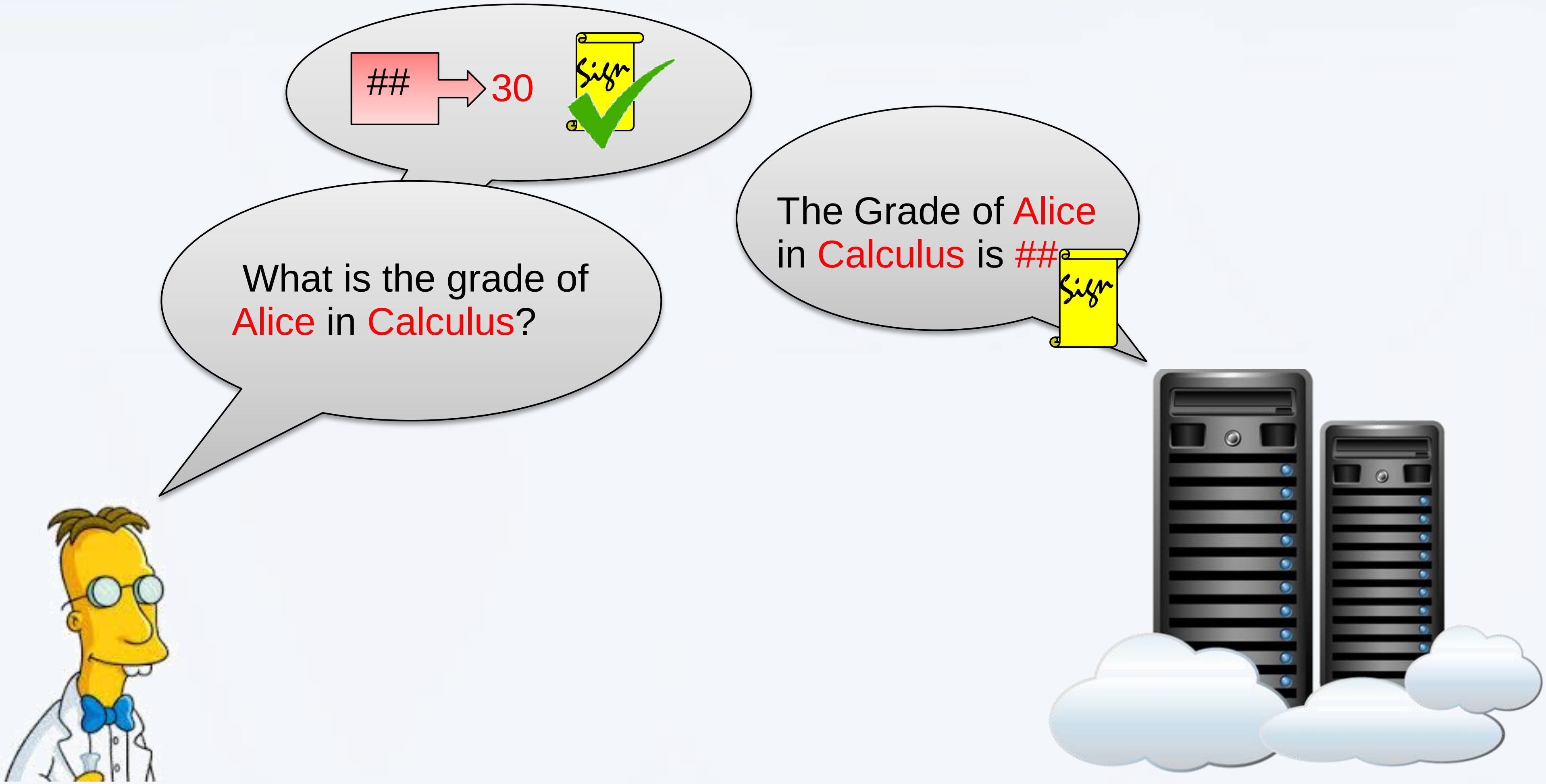


Authenticated Encryption

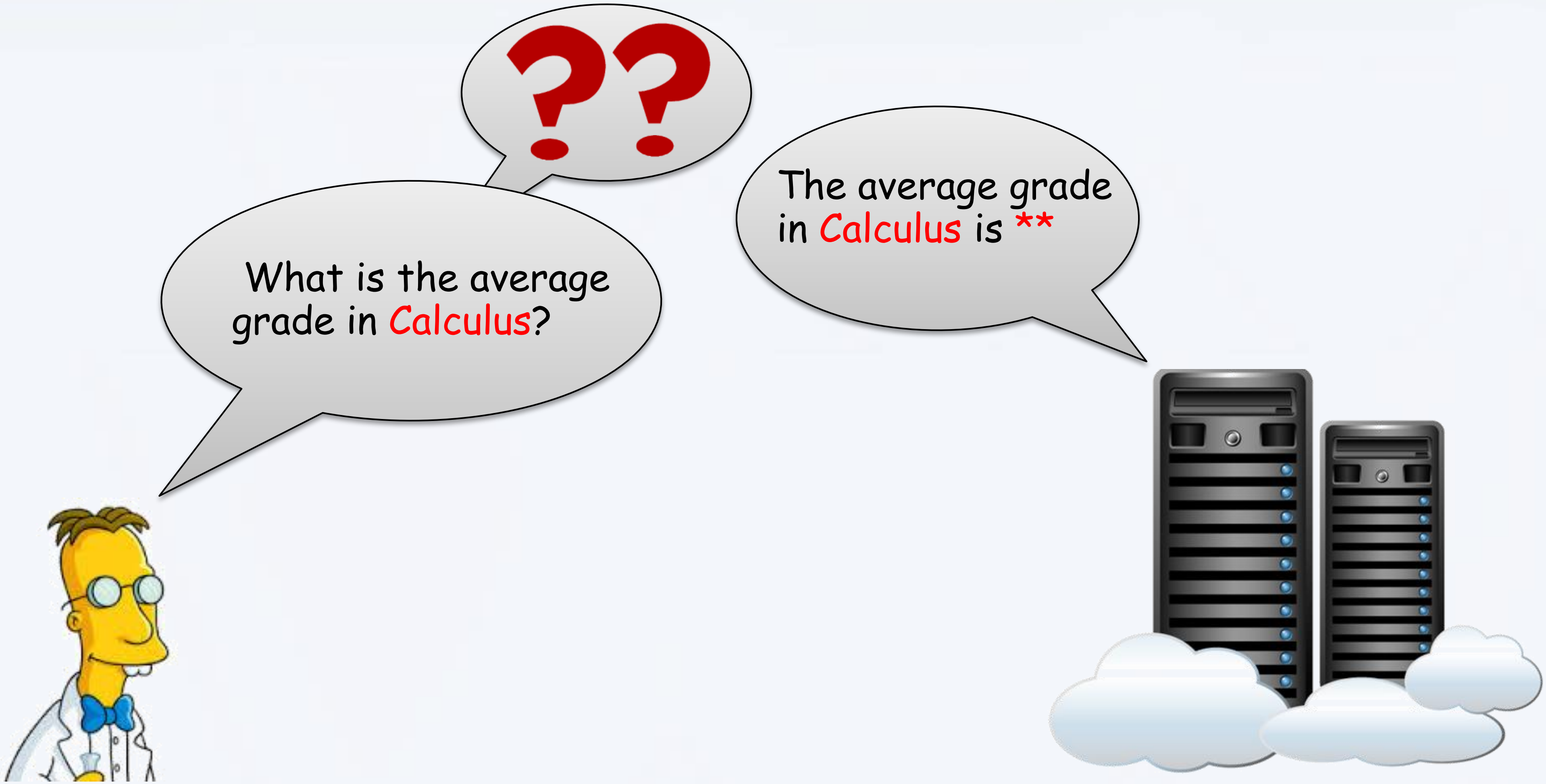


Simmetric Cryptography	Asimmetric Cryptography
TLS – SSH – IpSec [BelNam00]	[An01]

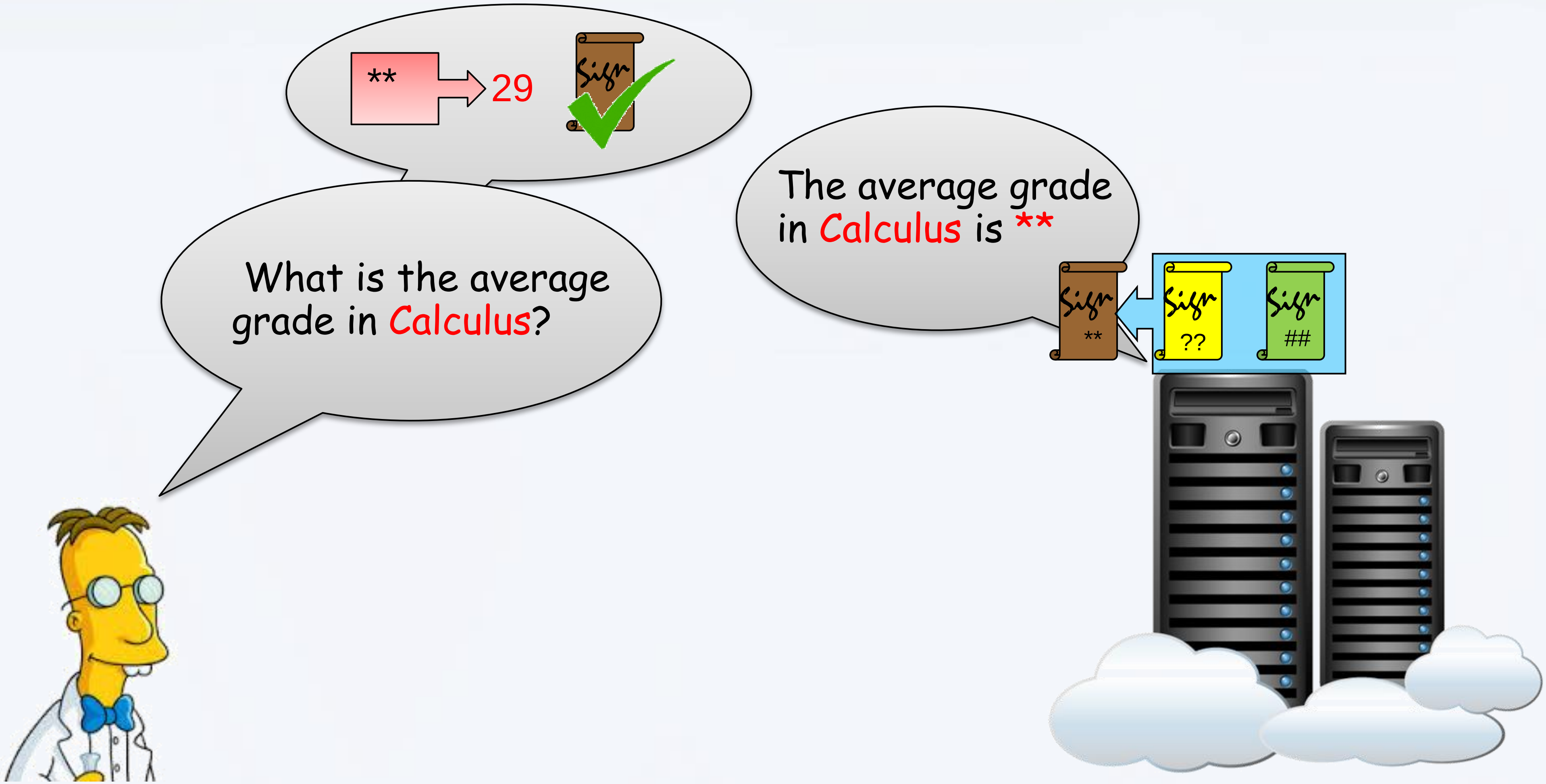
Authenticated Encryption



Authenticated Encryption



Delegating Computation on Authenticated Data with privacy



Linearly Homomorphic Authenticated Encryption with Public Verifiability

Inspired by [JY14]

- AE-KeyGen($1^\wedge$, k)→(sk, vk)
- AE-Encrypt (sk, FID, i, M) → C
- AE-Verify(vk, FID, C, f)→{0,1}
- AE-Decrypt (sk, FID, C, f) → M or ⊥
- AE-Eval(vk, f, FID, $\{C_i\}_{i=1,...,k}$) →C

Public Verifiability

Security: LH-IND-CCA for privacy, LH-Uf-CMA for integrity

LAEPuV - General Construction

M message space, additive group

R randomness space, multiplicative group

C ciphertext space, multiplicative group

LAEPuV - General Construction

M message space, additive group

R randomness space, multiplicative group

C ciphertext space, multiplicative group

T IND-CPA secure

Public Key Encryption Scheme

$$Enc_{pk}(M_1,R_1)*Enc_{pk}(M_2,R_2)=Enc_{pk}(M_1+M_2,R_1*R_2)$$

LAEPuV - General Construction

M message space, additive group

R randomness space, multiplicative group

C ciphertext space, multiplicative group

T IND-CPA secure

Public Key Encryption Scheme

$\mathrm{Enc}_{pk}(M_1,R_1)*\mathrm{Enc}_{pk}(M_2,R_2)=\mathrm{Enc}_{pk}(M_1+M_2,R_1*R_2)$

Σ Linearly Homomorphic signature scheme for elements in M

LAEPuV - General Construction

M message space, additive group

R randomness space, multiplicative group

C ciphertext space, multiplicative group

T IND-CPA secure

Public Key Encryption Scheme

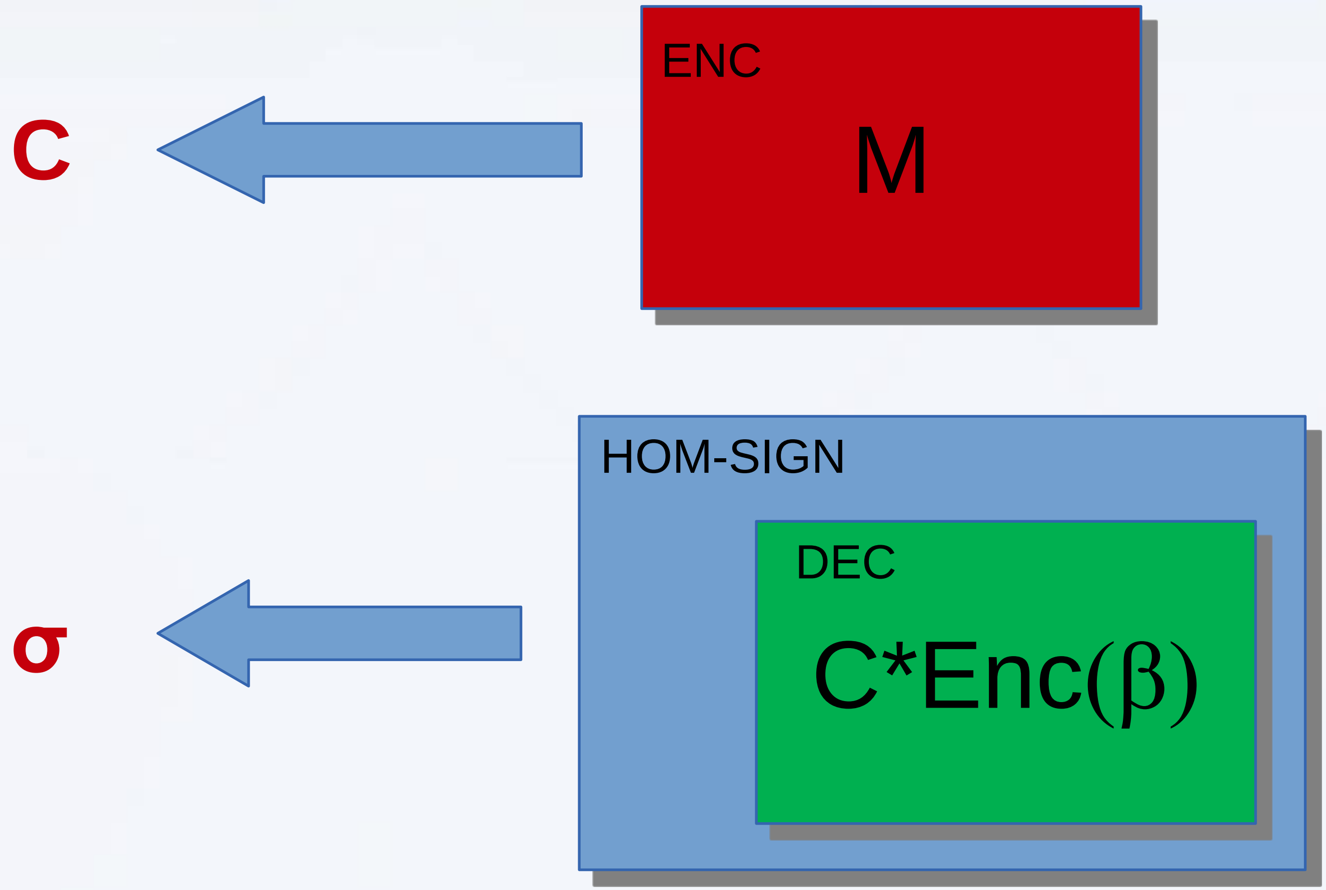
$Enc_{pk}(M_1,R_1)*Enc_{pk}(M_2,R_2)=Enc_{pk}(M_1+M_2,R_1*R_2)$

Σ Linearly Homomorphic signature scheme for elements in M

H Random Oracle

$H_K:\{0,1\}^*\rightarrow C$

LAEPuV – Encryption



LAEPuV – Eval

$f = (\alpha_1, \alpha_2, \dots, \alpha_k)$

c

$$\prod_{i=1}^{k} c_i^{\alpha_i}$$

$f(m_1 + \beta_1, \dots, m_k + \beta_k)$

σ

HOM-EVAL

f

$m_1 + \beta_1, \dots, m_k + \beta_k$

$\sigma_1, \dots, \sigma_k$

LAEPuV – Practical Instantiation

- Paillier's encryption guarantees the homomorphic property of **T**

$$(g^{m_1} r_1^N)(g^{m_2} r_2^N) = g^{m_1+m_2}(r_1 r_2)^N$$

- As concrete instantiation of the linearly homomorphic signature scheme one can use a simple variant of the (strong) RSA based scheme [CFW12]

Other results (in the paper)

- A Linearly homomorphic signature scheme to sign elements in (bilinear) groups

- This has nice applications in the context of On-line/Off-line signatures

Related works

- **Efficient Delegation of Computation over encrypted data**
  - [JY14], [FGP14]

Conclusion and Open problems

- ✔ Very efficient
- ✔ General construction
- ✔ Public Verifiability

Only linear functions ✗

Needs ROM ✗

Conclusion and Open problems

Interesting Open questions remain :

- How to extend to more general functionalities?
- How to avoid ROM?


Thank You