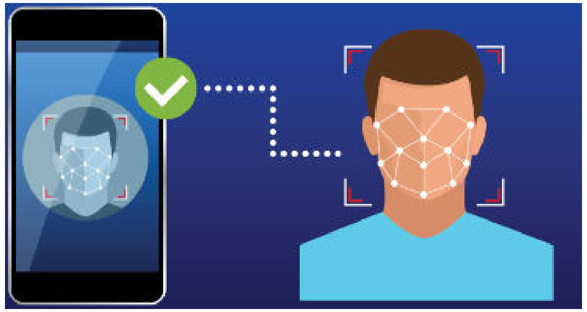
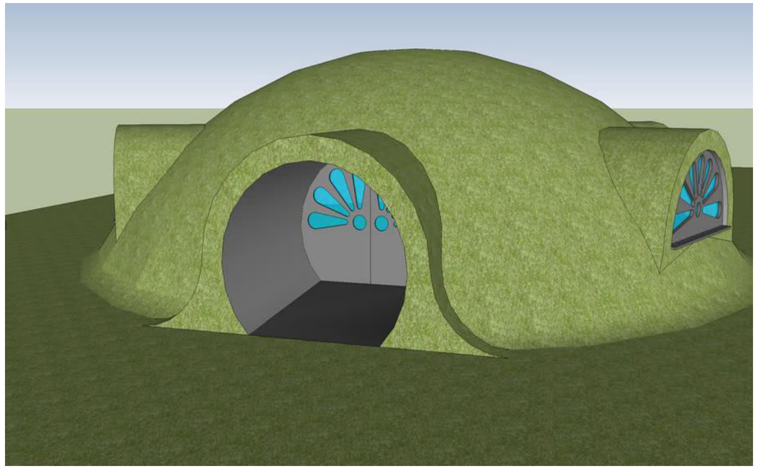
I. Introduction

Goal: Access Control in a company

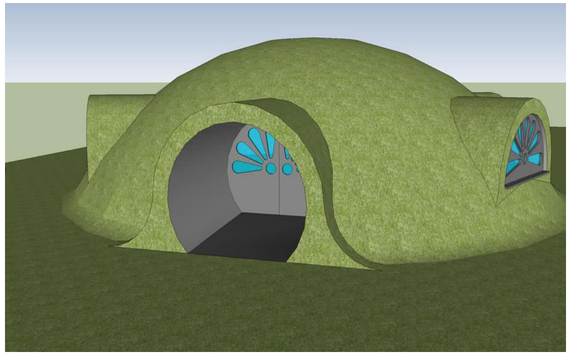




Properties

- Authentication:

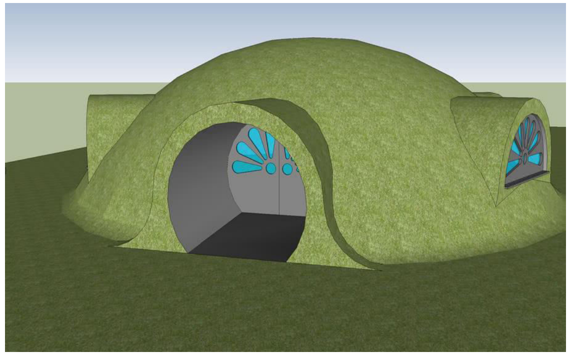
Is recognized as an employee of the company



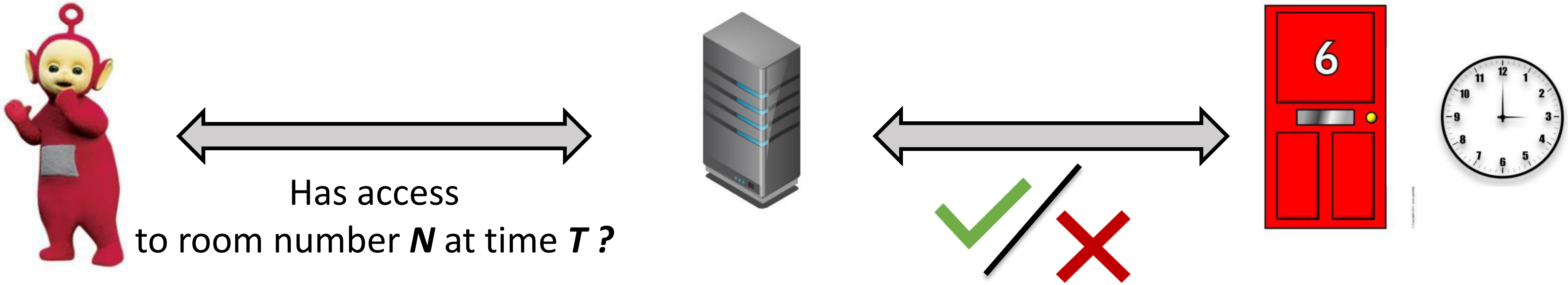
Properties

- Authentication:

Is recognized as an employee of the company



- Authorization:

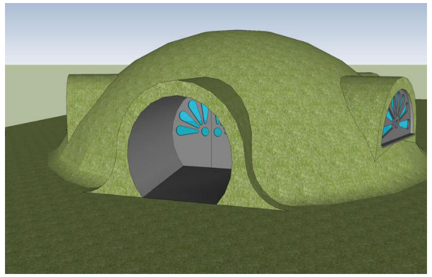


Has access to room number ***N*** at time ***T ?***

Properties

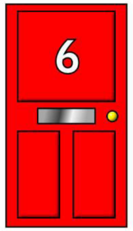
- Authentication:

Is recognized as an employee of the company



- Authorization:

Has access to room number ***N*** at time ***T***



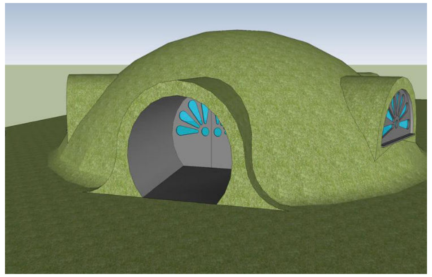
- Anonymity:

The identity of Po is not revealed to the server

Properties

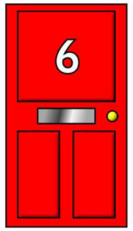
- Authentication:

Is recognized as an employee of the company



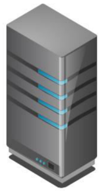
- Authorization:

Has access to room number ***N*** at time ***T***



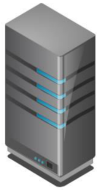
- Anonymity:

The identity of Po is not revealed to the server



- Non-traceability:

The server cannot know if it is the same person



Outline

II. Simplified protocol

III. Properties

IV. Adversarial Model

V. Primitives

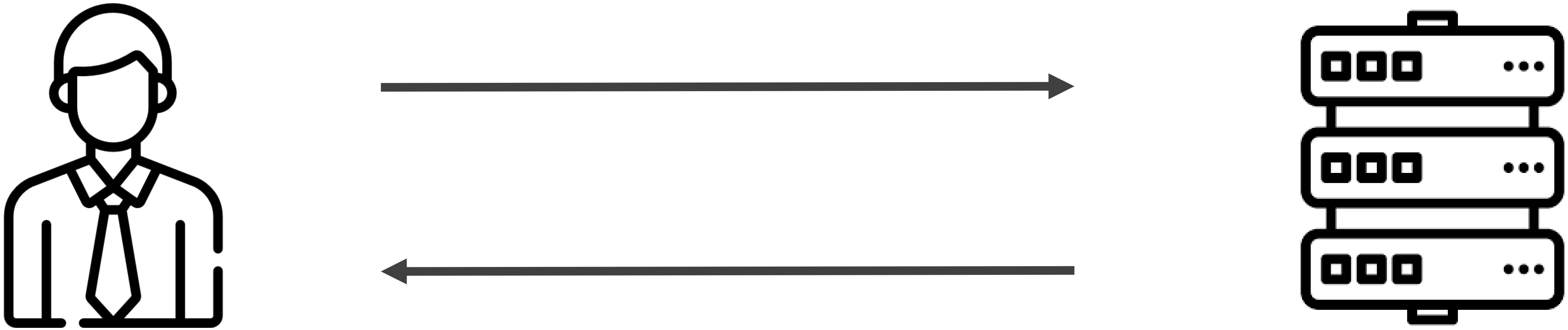
VI. Protocol

VII. Advanced properties

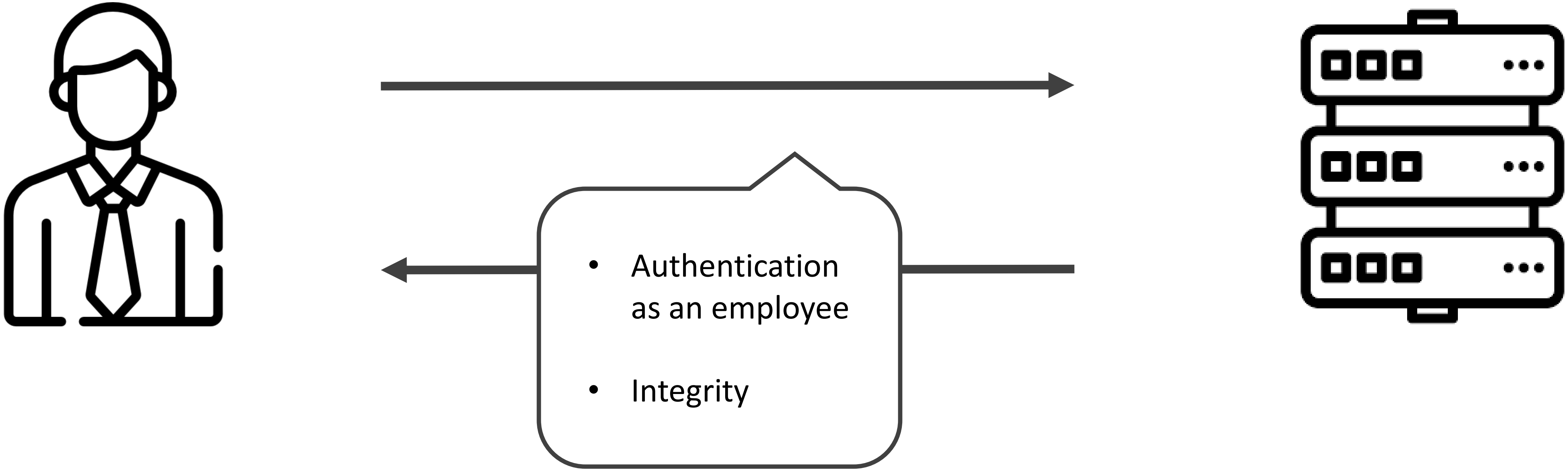
VIII. Discussion

II. Simplified Protocol

Simplified Protocol

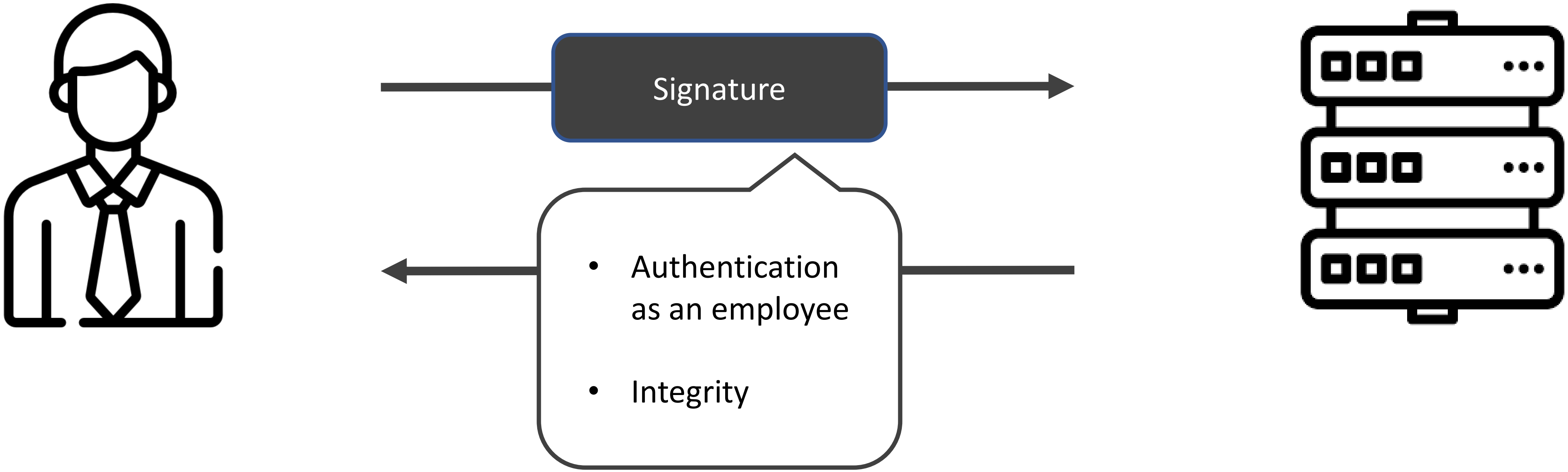


Simplified Protocol



- Authentication as an employee
- Integrity

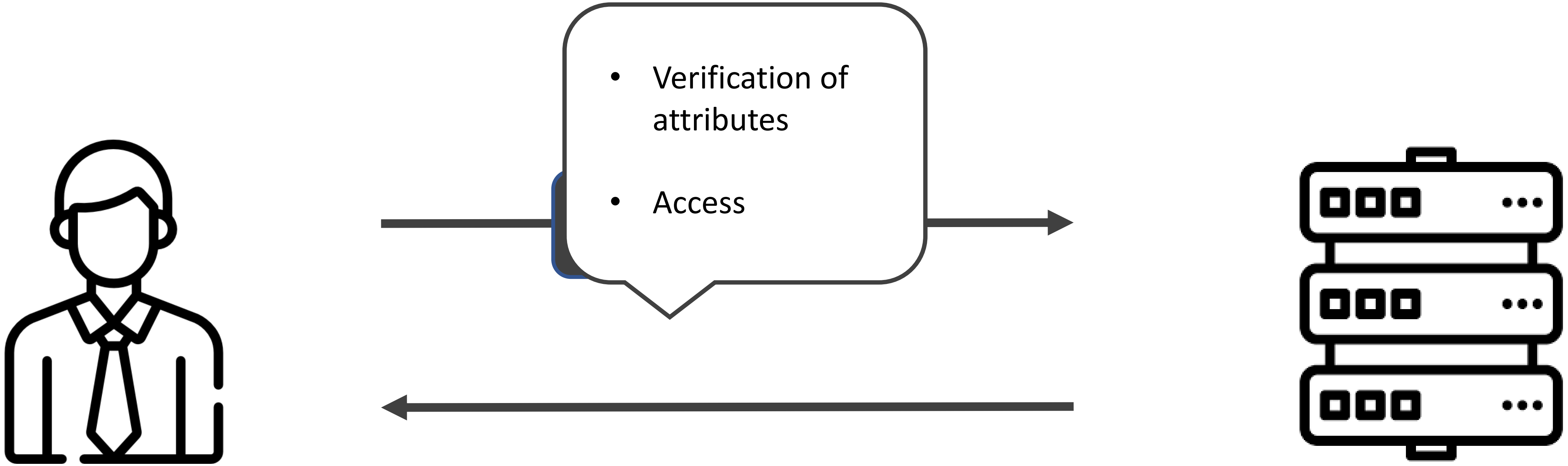
Simplified Protocol



Signature

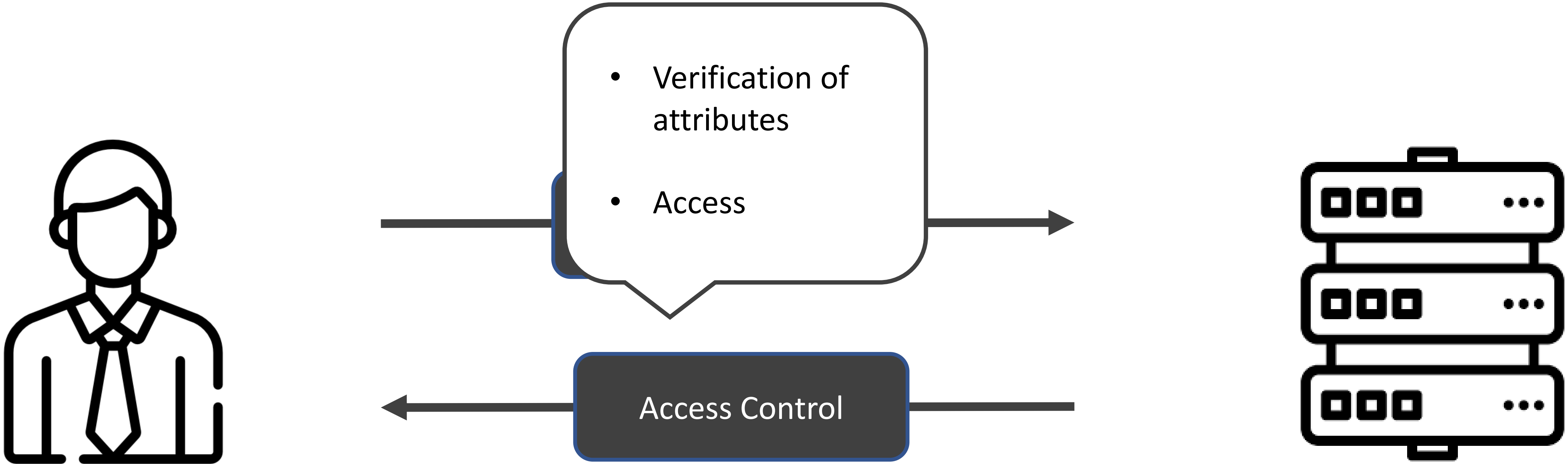
- Authentication as an employee
- Integrity

Simplified Protocol



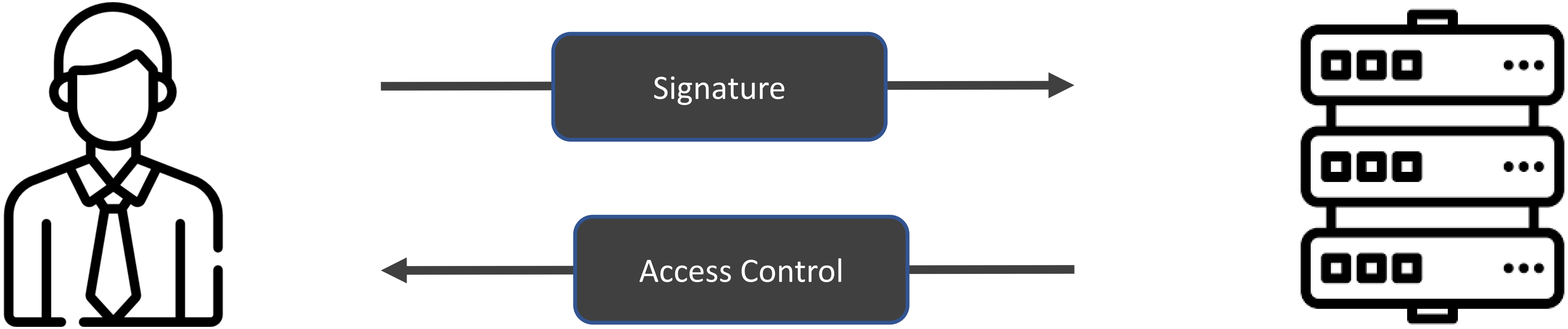
- Verification of attributes
- Access

Simplified Protocol



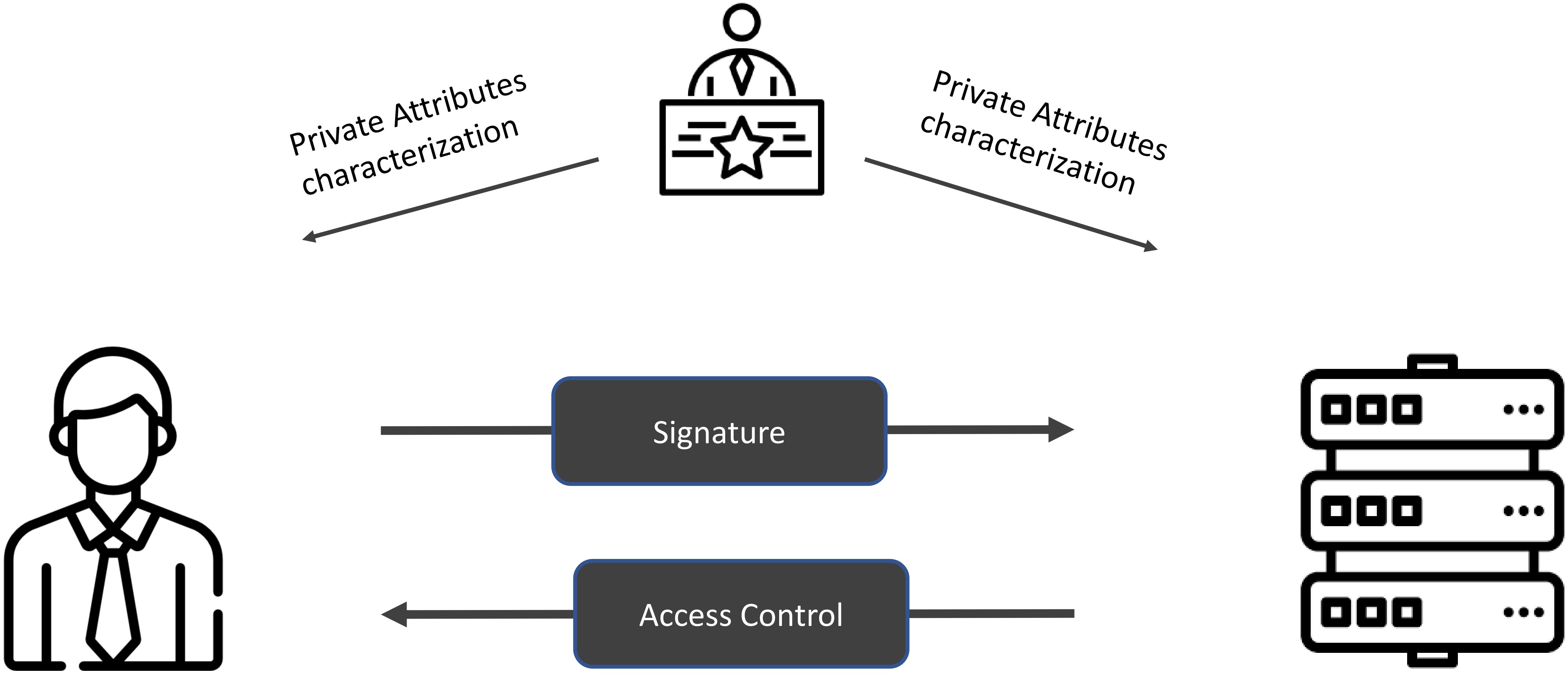
- Verification of attributes
- Access

Simplified Protocol



Signature

Access Control



Private Attributes characterization

Private Attributes characterization

Signature

Access Control

What are these ?

Signature

Access Control

Signature

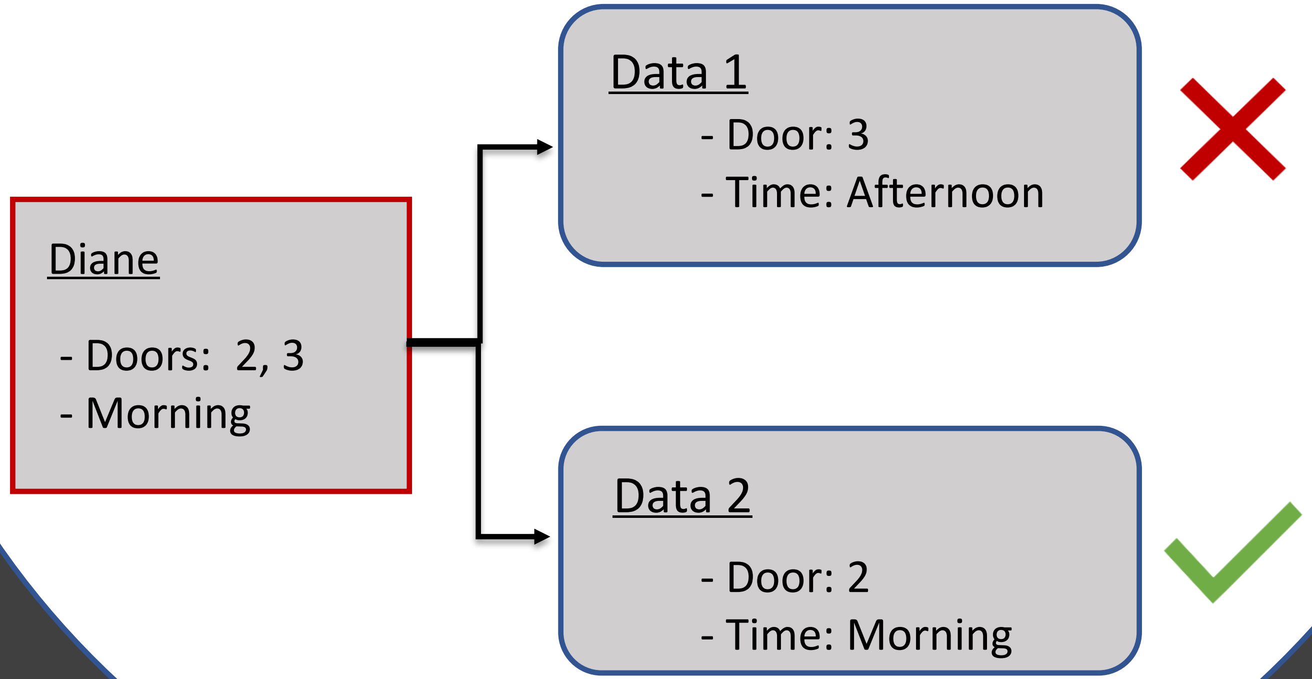
Access Control

- Authentication: Authentify the signer
- Unforgeability: Cannot forge a signature without secret information
- Integrity: Ensure the authenticity of a message
- Non-Repudiation: The Signer cannot question his signature

Signature

▶ Access Control (ABE)

- A user must be able to open a door it has access too
- A user cannot enter a door it has not access too



<u>Diane</u>

- Doors: 2, 3
- Morning

<u>Data 1</u>

- Door: 3
- Time: Afternoon

✗

<u>Data 2</u>

- Door: 2
- Time: Morning

✓

Properties Recap

✓ Authentication as an employee

✓ Access Control through Attributes

✗ Anonymity

✗ Non-Traceability

✗ Execution Authentication

✓ Authentication as an employee

✓ Access Control through Attributes

✗ An

✗ Non-Traceability

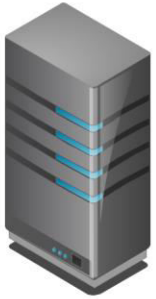
✗ Execution Authentication

How to achieve the remaining properties

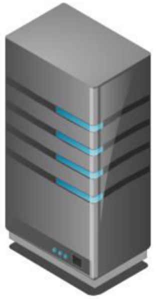
III. Verified Properties

Authenticity

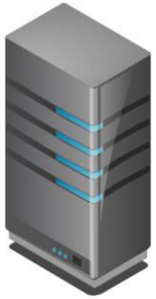
Po can prove to the server that he is an employee of the company



Authenticity
Po can prove to the server that he is an employee of the company



Integrity
Alteration of messages is detected by the server



Open the door number 10

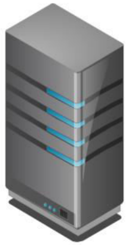
Open the door number 15 ✗

Attributes :
Door 1, 4, 7

Correctness

If Po has the right ***a*** then the server validates the requests linked to it

Open door 4



✓

Attributes :
Door 1, 4, 7

Correctness
If Po has the right ***a*** then the server validates the requests linked to it

Open door 4

✓

Attributes :
Door 1, 4, 7

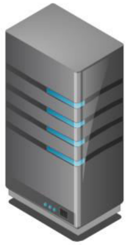
Soundness
If Po does not have the right ***a*** then the server will not validate a request that requires the right ***a***

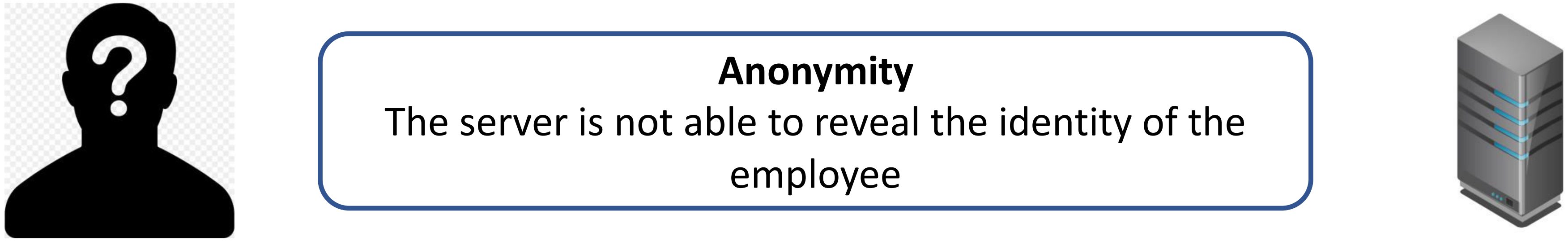
Open door 5

✗

Anonymity

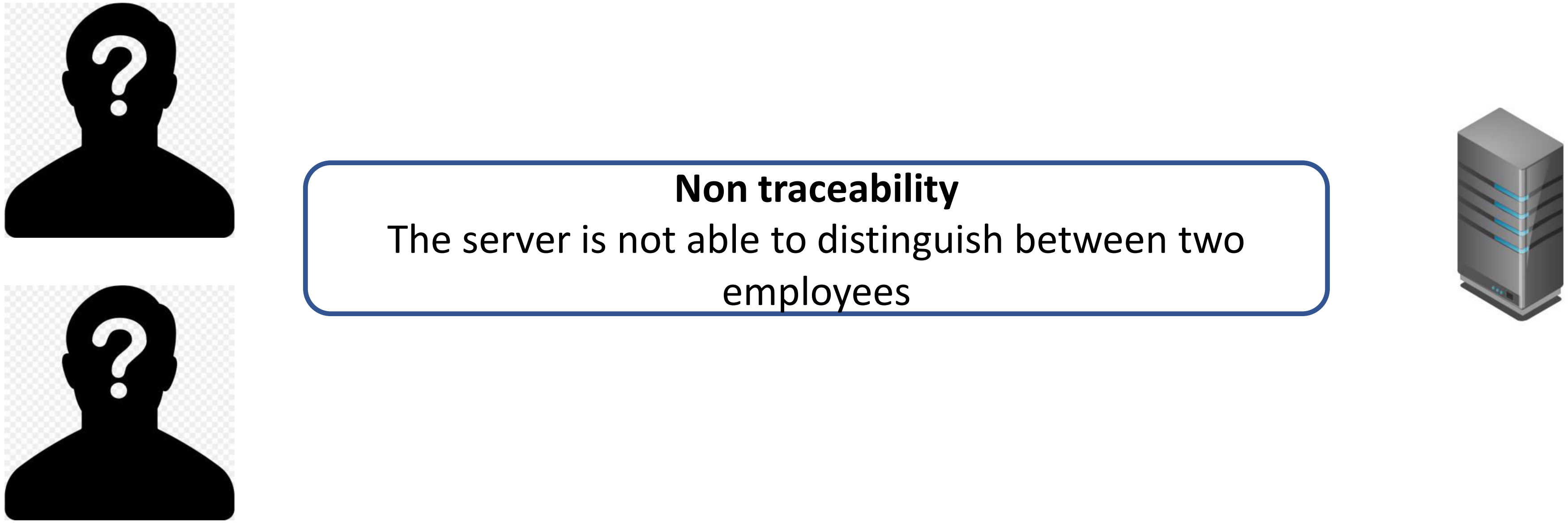
The server is not able to reveal the identity of the employee





Anonymity

The server is not able to reveal the identity of the employee

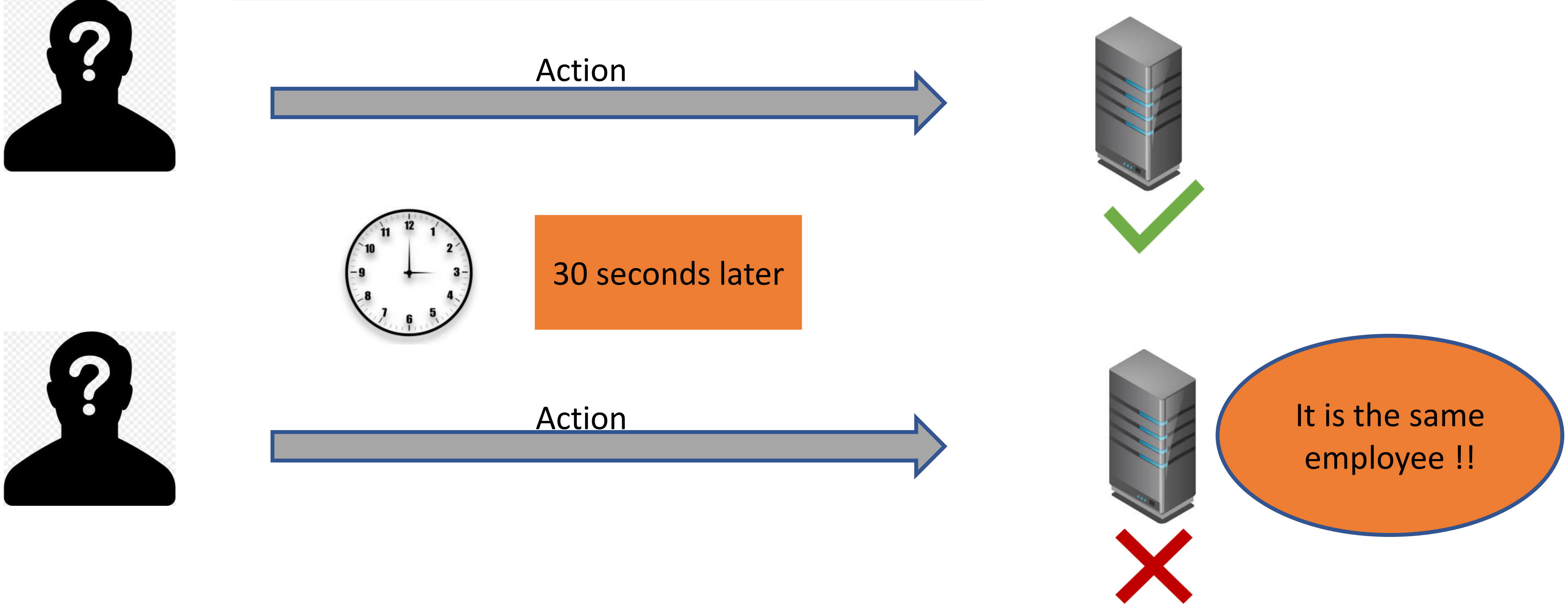


Non traceability

The server is not able to distinguish between two employees

Local traceability by the server

The server should not authorize an employee to do strictly more than one action in a small laps of time (Say one minute)



Action

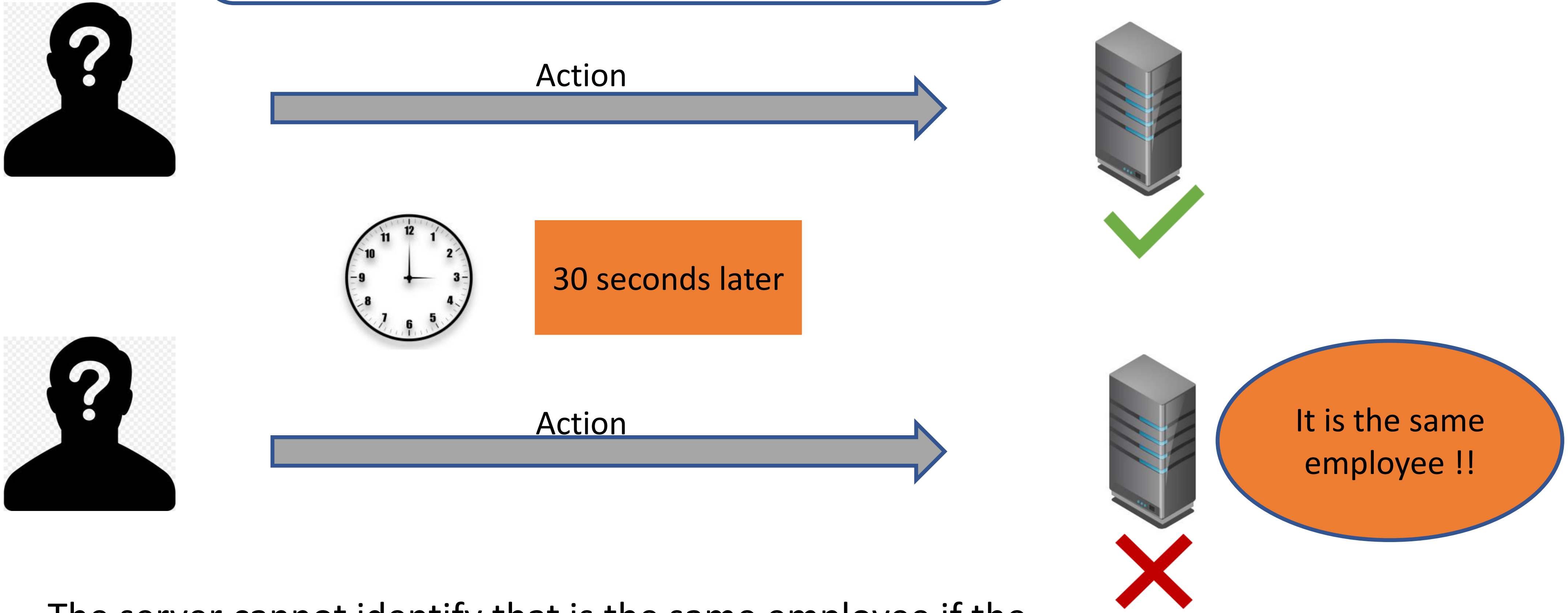
30 seconds later

Action

It is the same employee !!

Local traceability by the server

The server should not authorize an employee to do strictly more than one action in a small laps of time (Say one minute)



Action

✓

30 seconds later

Action

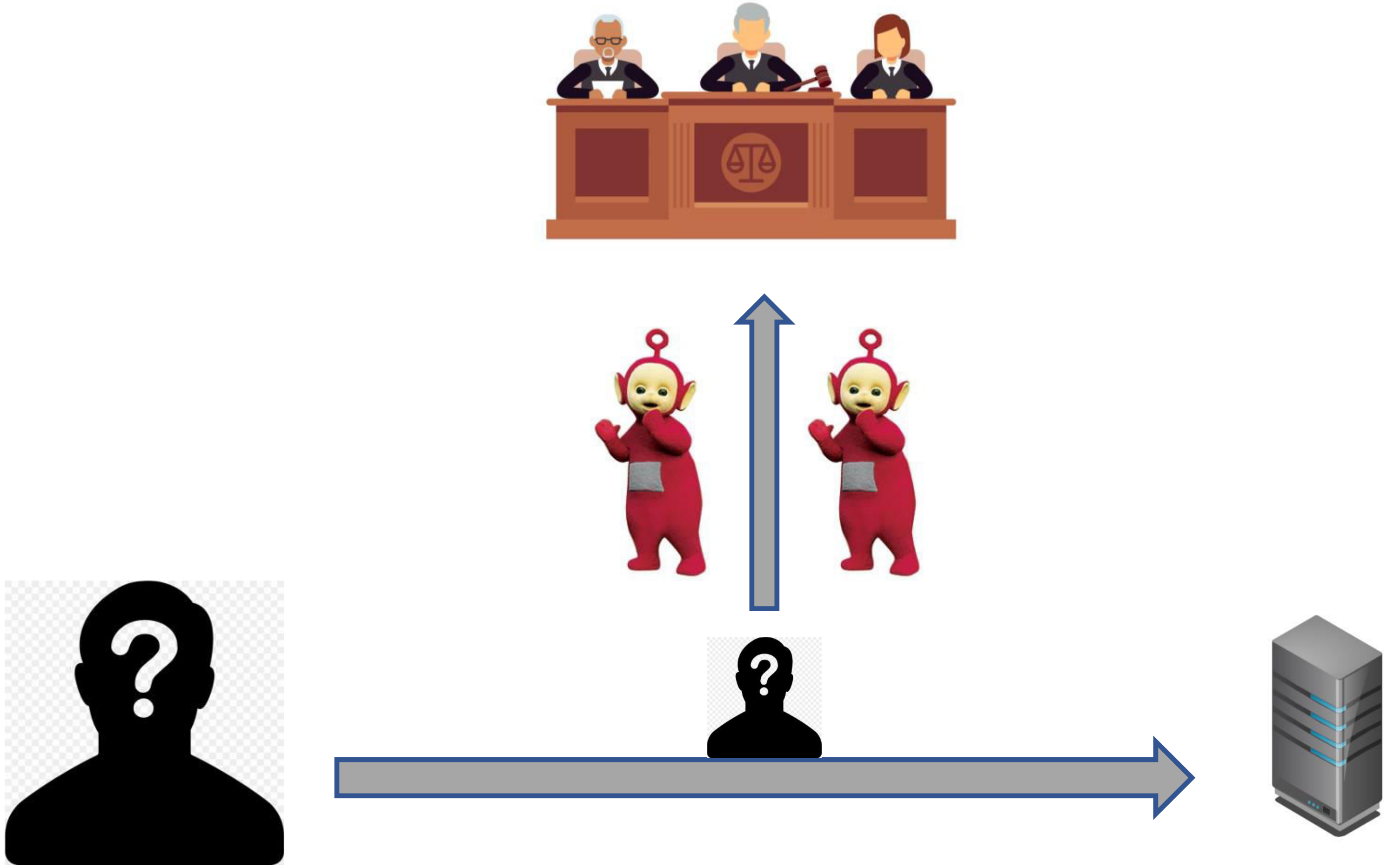
It is the same employee !!

✗

The server cannot identify that is the same employee if the time between the two actions is greater than **one minute !**

Identification by an authority

A predefined authority independent of the server can reveal the identity of an employee



IV. Environment

Protocol floating
in
its Environment

Adversary:

Adversary:

- Listen
- Build & send messages

Adversary:

- Listen
- Build & send messages

Corruption:

Adversary:

- Listen
- Build & send messages

Corruption:

- Collusion
- Static Corruption

Adversary:

- Listen
- Build & send messages

Semi-Honest Server:

Corruption:

- Collusion
- Static Corruption

Adversary:

- Listen
- Build & send messages

Corruption:

- Collusion
- Static Corruption

Semi-Honest Server:

- Honest: answers honestly to queries
- But curious: tries to learn information

V. Primitives

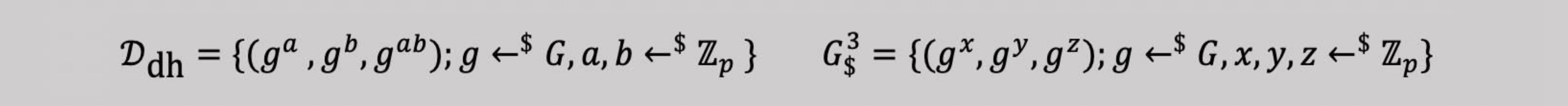
Warmup

Definition: Square Discrete Logarithm Assumption

In a group G of prime order p, it states that for any generator g, given $y = g^x$ and $z = g^{x^2}$, it is computationally hard to recover x.

Definition: Decisional Diffie-Hellman (DDH) Assumption

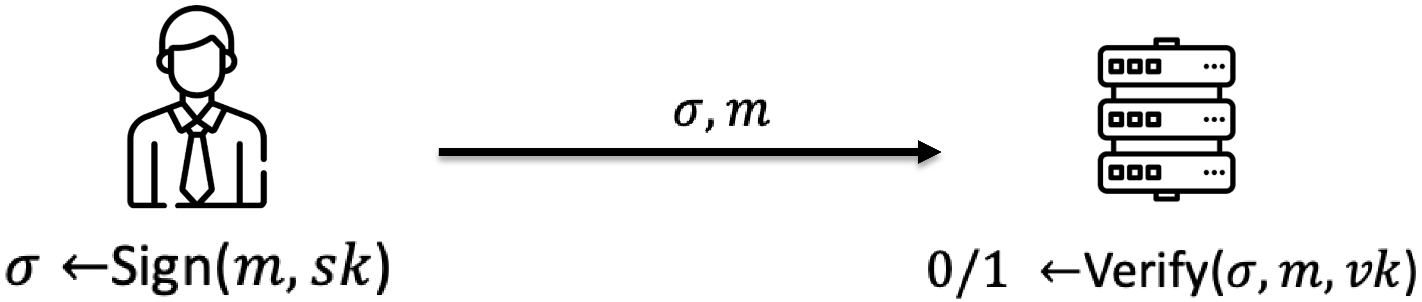
In a group G of prime order p, it states that the two following distributions are computationally indistinguishable:



$$\mathcal{D}_{\mathrm{dh}} = \{(g^a, g^b, g^{ab}); g \leftarrow^{\$} G, a, b \leftarrow^{\$} \mathbb{Z}_p\} \qquad G_{\$}^3 = \{(g^x, g^y, g^z); g \leftarrow^{\$} G, x, y, z \leftarrow^{\$} \mathbb{Z}_p\}$$

1. Signature Scheme

- Definition
  - Setup(1^λ) → $param$
  - Keygen($param$) → (sk, vk)
  - Sign(m, sk) → σ
  - Verify(σ, m, vk) → 1 if σ valid relative to vk, 0 otherwise.



σ, m

σ ←Sign(m, sk)

$0/1$ ←Verify(σ, m, vk)

Properties:

- Unforgeabelity: Cannot forge a signature without sk
- Integrity: Ensure the authenticity of a m
- Non-repudiation: The signer cannot question his signature

1. Signature Scheme

- Construction

  - Setup(1^λ): Generator g_2, Hash function $H: \{0,1\}^* \rightarrow G_1$.
  - Keygen(g, H): Pick $sk \xleftarrow{\$} \mathbb{Z}_p$ and compute v$k = g_2^{sk}$.
  - Sign(m, sk): Compute $h \leftarrow H(m) \in G_1$ and $\sigma \leftarrow h^{sk}$.
  - Verify(σ, m, vk): Compute $h \leftarrow H(m)$ and verifies that $e(\sigma, g) = e(vk, h)$.

Definition: Bilinear Pairing

Let q be a prime number. Let G_1, G_2 two additive cyclic group of order q, and let G_T another cyclic group of order q written multiplicatively. A pairing is a map $e: G_1 \times G_2 \rightarrow G_T$ which satisfies the following properties:

1. **(bilinearity)** $\boldsymbol{\forall\, a, b \in \mathbb{F}_q^*, u \in G_1, v \in G_2: e(u^a, v^b) = e(u, v)^{ab}}$
2. (non-degeneracy) $e(u, u) \neq 1$
3. (computability) e can be efficiently computed.

2. Anonymous Randomizable Signature

- Definition

Key Idea: Hide the identity of a user in an **Anonymous Ephemeral Identities**

- Setup(1^λ) → $param$
- Keygen($param$) → (sk, vk)
- GenTag($param$) → $(\tilde{\tau}, \tau)$
- Sign(m, sk, τ) → σ
- Verify(σ, τ, m, vk) → 1 if σ valid relative to vk and τ, 0 otherwise.
- RandSign($\sigma, \tau, m, vk, \alpha$) → σ' on m under the randomized tag τ' and the same key vk.

Additional Property:

- Anonymous: Cannot link a signature with the identity of the signer

2. Anonymous Randomizable Signature

- Construction: Warmup

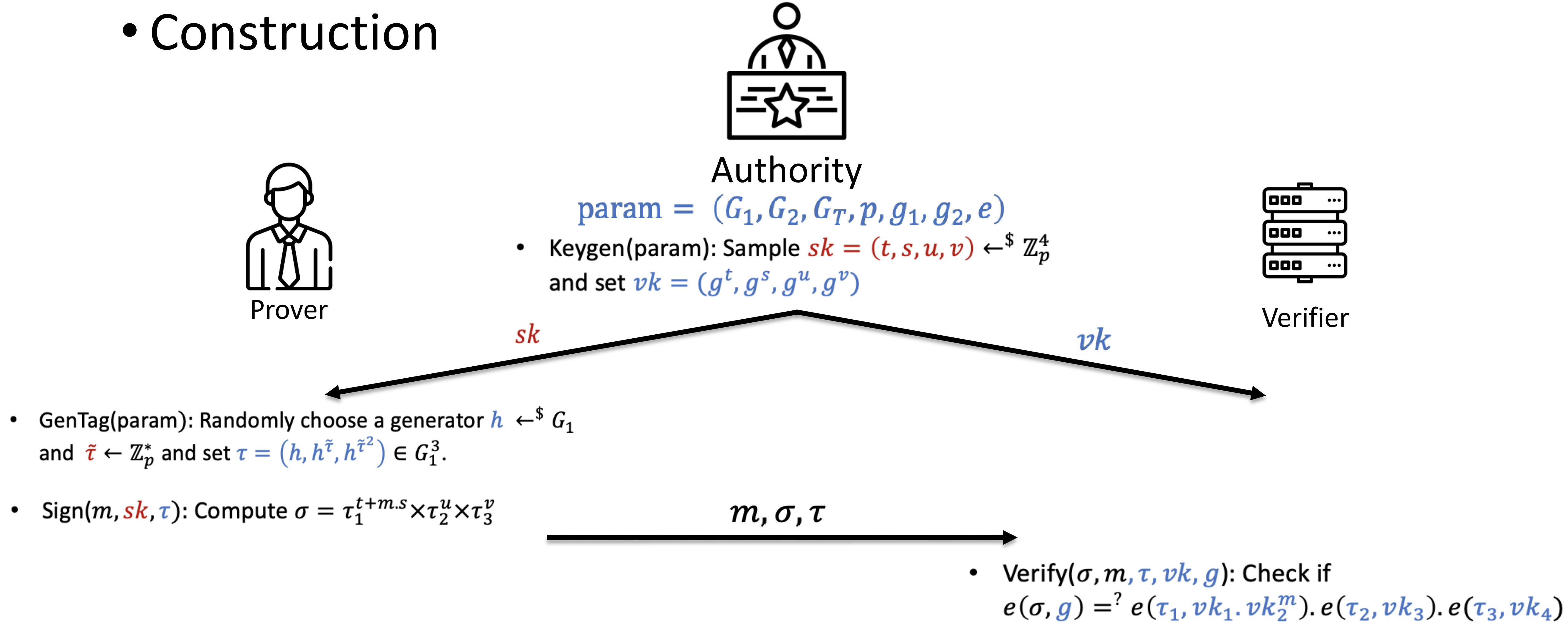
Definition: Decisional Square Diffie-Hellman (DSqDH) Assumption

In a group G of prime order p, it states that the two following distributions are computationally indistinguishable:

$$\mathcal{D}_{\mathrm{sqdh}} = \{(g, g^{x}, g^{x^2}); g \leftarrow^{\$} G, x \leftarrow^{\$} \mathbb{Z}_p \} \qquad G_{\$}^{3} = \{(g, g^{x}, g^{y}); g \leftarrow^{\$} G, x, y \leftarrow^{\$} \mathbb{Z}_p\}$$

2. Anonymous Randomizable Signature

- Construction



$\text{param} = (G_1, G_2, G_T, p, g_1, g_2, e)$

- Keygen(param): Sample $sk = (t, s, u, v) \leftarrow^{\$} \mathbb{Z}_p^4$ and set $vk = (g^t, g^s, g^u, g^v)$

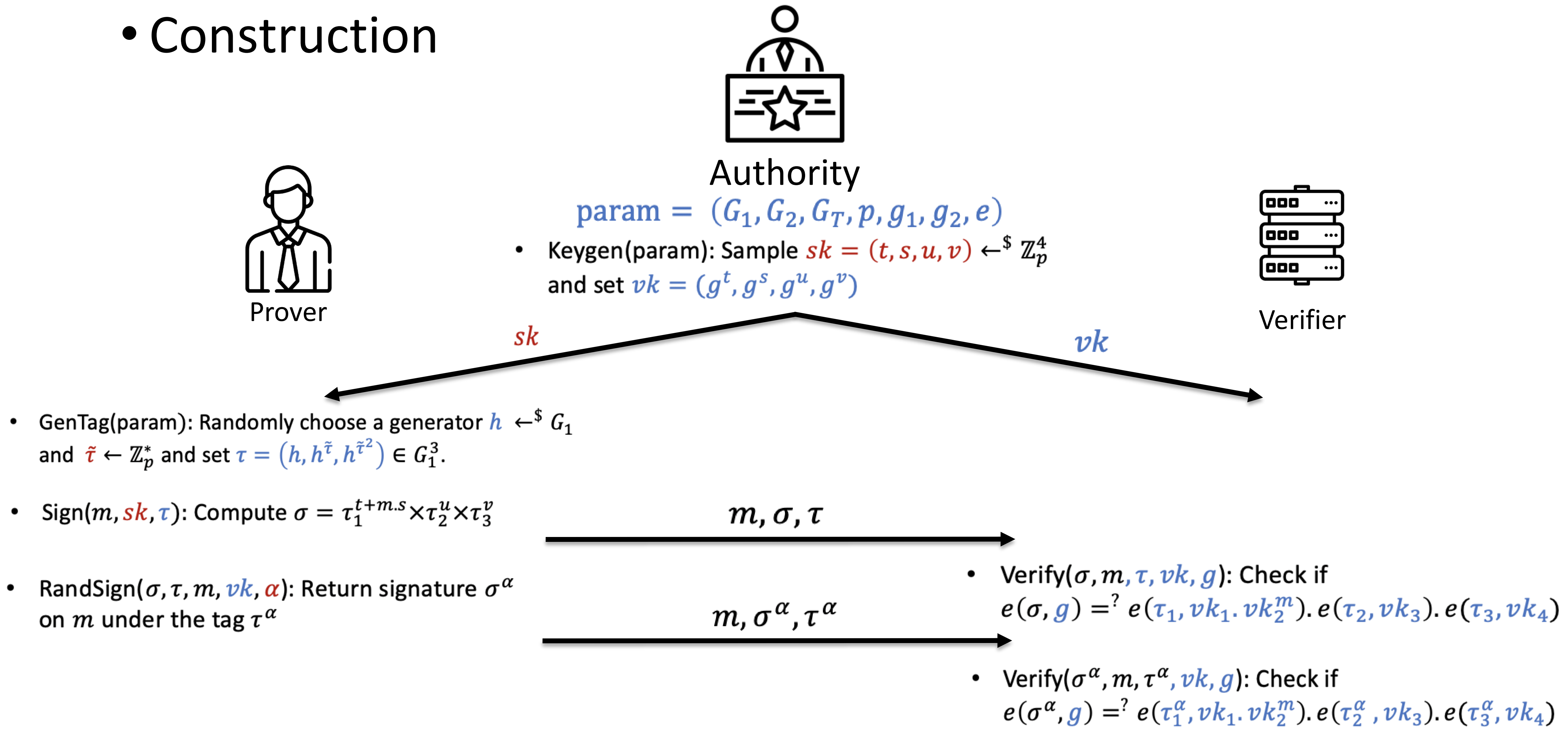
- GenTag(param): Randomly choose a generator $h \leftarrow^{\$} G_1$ and $\tilde{\tau} \leftarrow \mathbb{Z}_p^*$ and set $\tau = (h, h^{\tilde{\tau}}, h^{\tilde{\tau}^2}) \in G_1^3$.

- Sign(m, sk, τ): Compute $\sigma = \tau_1^{t+m.s} \times \tau_2^u \times \tau_3^v$

- Verify(σ, m, τ, vk, g): Check if $e(\sigma, g) =^? e(\tau_1, vk_1 . vk_2^m) . e(\tau_2, vk_3) . e(\tau_3, vk_4)$

2. Anonymous Randomizable Signature

- Construction



Authority

$\text{param} = (G_1, G_2, G_T, p, g_1, g_2, e)$

- Keygen(param): Sample $sk = (t, s, u, v) \xleftarrow{\$} \mathbb{Z}_p^4$ and set $vk = (g^t, g^s, g^u, g^v)$

Prover

Verifier

sk

vk

- GenTag(param): Randomly choose a generator $h \xleftarrow{\$} G_1$ and $\tilde{\tau} \leftarrow \mathbb{Z}_p^*$ and set $\tau = \left(h, h^{\tilde{\tau}}, h^{\tilde{\tau}^2}\right) \in G_1^3$.
- Sign(m, sk, τ): Compute $\sigma = \tau_1^{t+m.s} \times \tau_2^u \times \tau_3^v$
- RandSign($\sigma, \tau, m, vk, \alpha$): Return signature σ^α on m under the tag τ^α

m, σ, τ

$m, \sigma^\alpha, \tau^\alpha$

- Verify(σ, m, τ, vk, g): Check if $e(\sigma, g) =^? e(\tau_1, vk_1 . vk_2^m) . e(\tau_2, vk_3) . e(\tau_3, vk_4)$
- Verify($\sigma^\alpha, m, \tau^\alpha, vk, g$): Check if $e(\sigma^\alpha, g) =^? e(\tau_1^\alpha, vk_1 . vk_2^m) . e(\tau_2^\alpha , vk_3) . e(\tau_3^\alpha, vk_4)$

2. Anonymous Randomizable Signature

- Construction

$$\text{param} = (G_1, G_2, G_T, p, g_1, g_2, e)$$

- Sign(m, sk, τ): Compute $\sigma = \tau_1^{t+m.s} \times \tau_2^{u} \times \tau_3^{v}$

$$\xrightarrow{\quad m, \sigma, \tau \quad}$$

- Verify(σ, m, τ, vk, g): Check if $e(\sigma, g) =^? e(\tau_1, vk_1.vk_2^m).e(\tau_2, vk_3).e(\tau_3, vk_4)$

- RandSign($\sigma, \tau, m, vk, \alpha$): Return signature σ^α on m under the tag τ^α

$$\xrightarrow{\quad m, \sigma^\alpha, \tau^\alpha \quad}$$

- Verify($\sigma^\alpha, m, \tau^\alpha, vk, g$): Check if $e(\sigma^\alpha, g) =^? e(\tau_1^\alpha, vk_1.vk_2^m).e(\tau_2^\alpha, vk_3).e(\tau_3^\alpha, vk_4)$

<u>Main intuition</u>: $e(\sigma^\alpha, g) = e(\tau^\alpha, vk) = e(\tau, vk^\alpha) = e(\tau, g^{\overbrace{\alpha.sk}^{\text{Randomized key}}})$

$$\boxed{\text{(bilinearity) } \forall\, a, b \in \mathbb{F}_q^*, u \in G_1, v \in G_2 \colon e\left(u^a, v^b\right) = e(u, v)^{ab}}$$

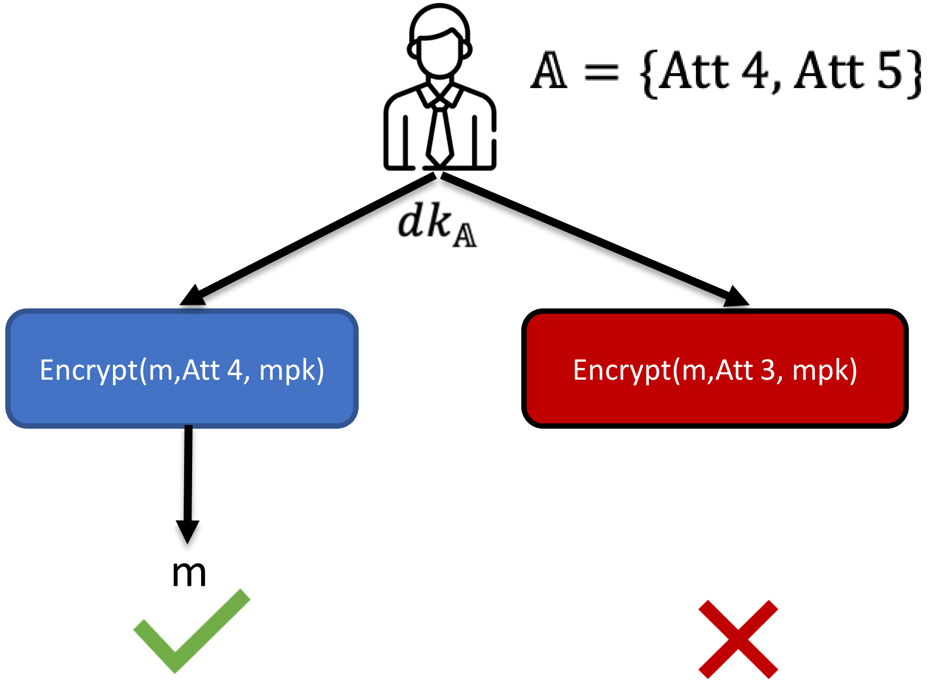
3. Attribute-based Encryption Scheme

- Definition

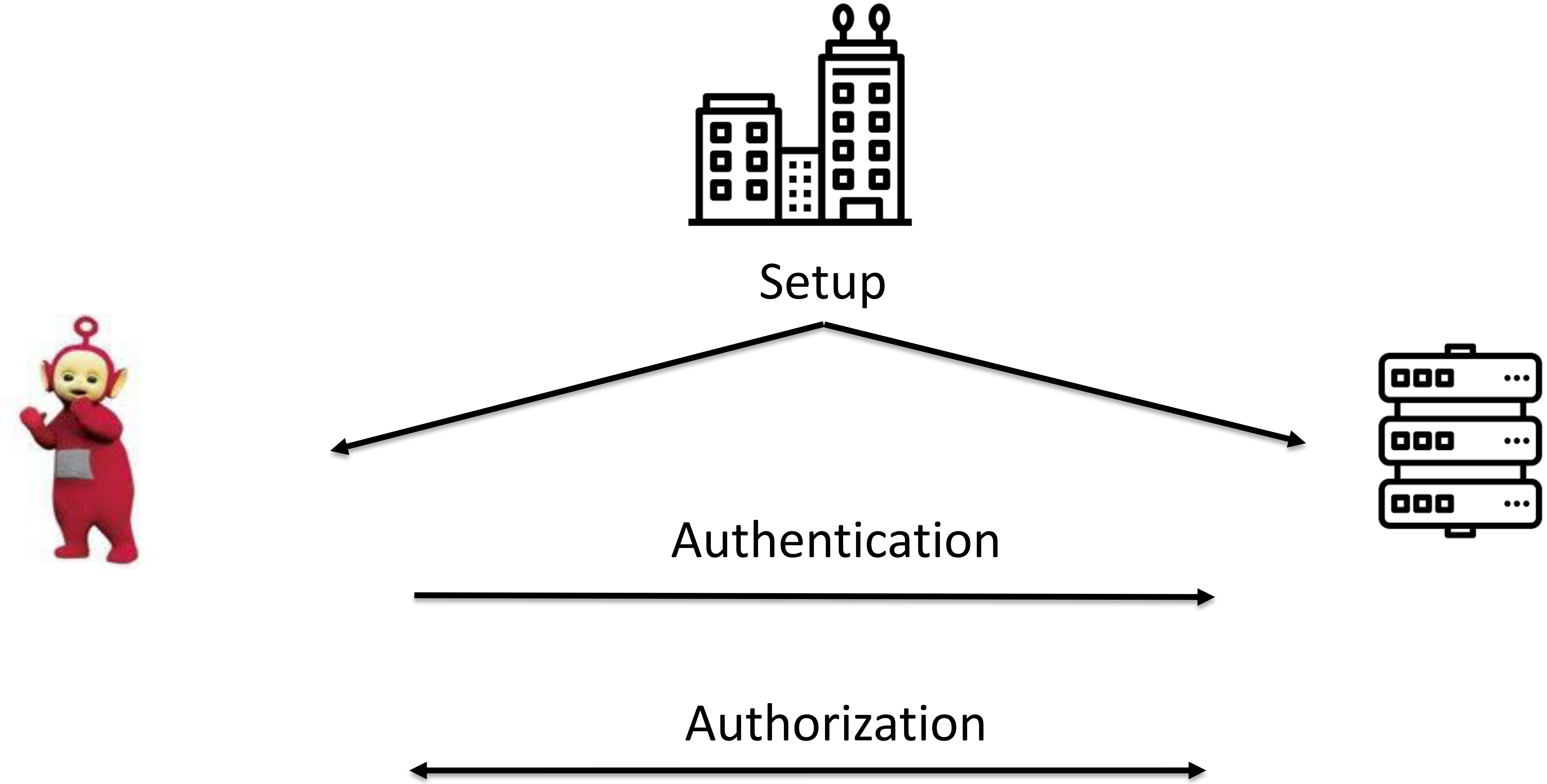
Definition: Access structure

Let $\mathcal{U}$ be a set of attributes. An access structure $\mathbb{A}$ is a collection of non-empty subset of $\mathcal{U}$.

- $\text{Setup}(1^{\lambda}) \rightarrow (mpk, msk)$
- $\text{Keygen}(\mathbb{A}, msk, mpk) \rightarrow dk_{\mathbb{A}}$
- $\text{Encrypt}(m, \gamma, mpk) \rightarrow ct_{\gamma}$
- $\text{Decrypt}(ct_{\gamma}, dk_{\mathbb{A}}, mpk) \rightarrow m$ if $\gamma \in \mathbb{A}$



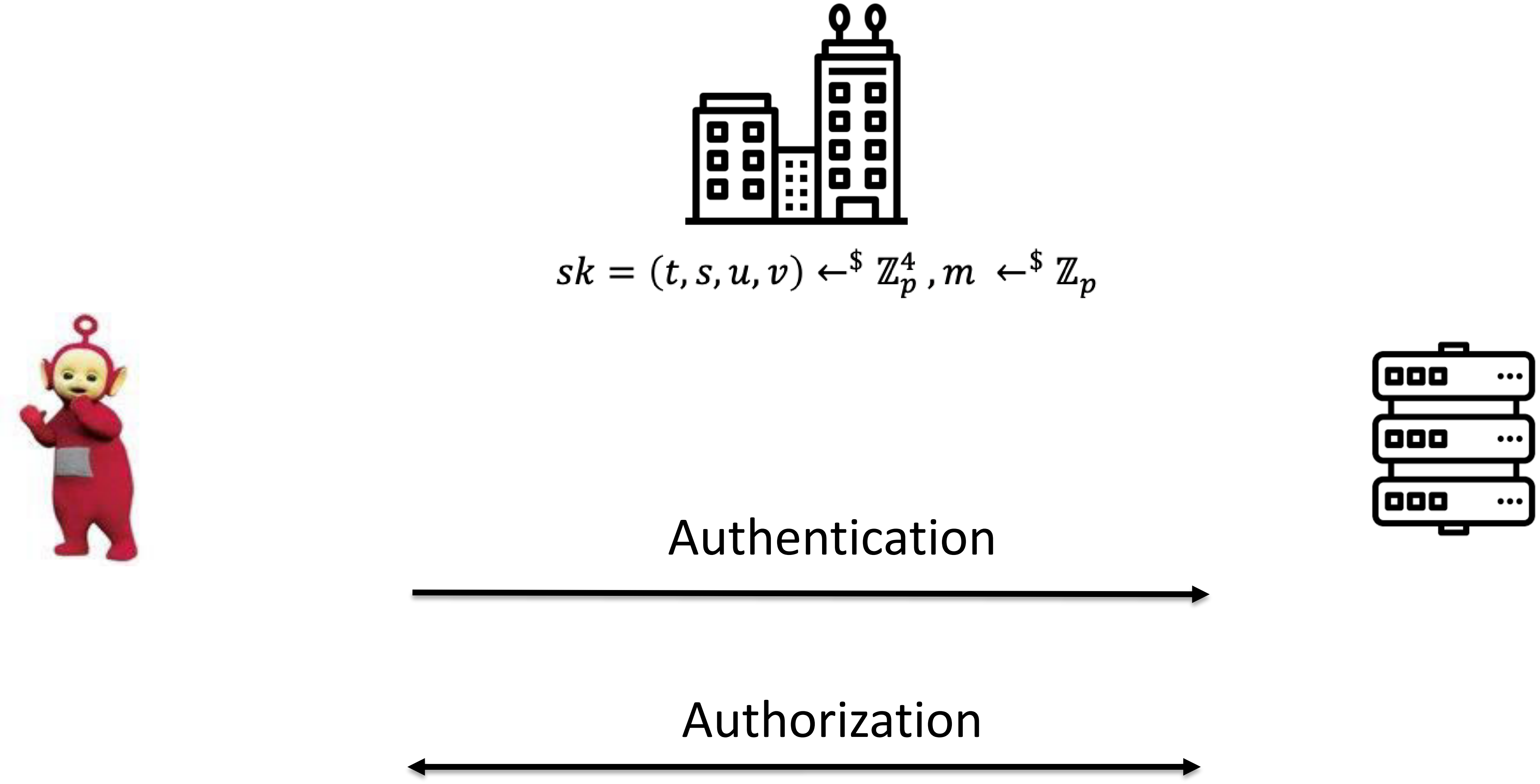
VI. Protocol



Setup

Authentication

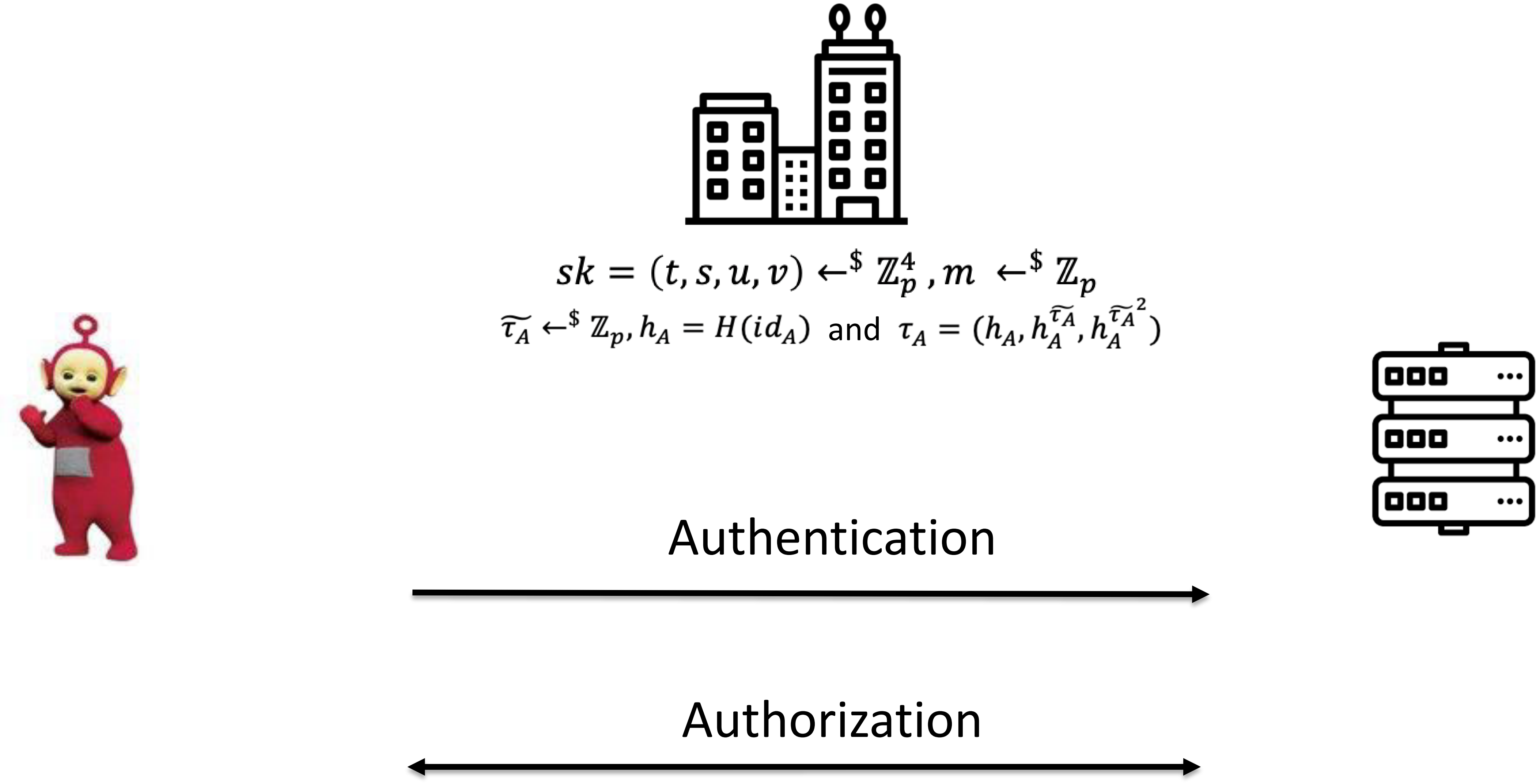
Authorization



$$sk = (t, s, u, v) \xleftarrow{\$} \mathbb{Z}_p^4 , m \xleftarrow{\$} \mathbb{Z}_p$$

Authentication

Authorization



$$sk = (t, s, u, v) \leftarrow^{\$} \mathbb{Z}_p^4, m \leftarrow^{\$} \mathbb{Z}_p$$

$\widetilde{\tau_A} \leftarrow^{\$} \mathbb{Z}_p, h_A = H(id_A)$ and $\tau_A = (h_A, h_A^{\widetilde{\tau_A}}, h_A^{\widetilde{\tau_A}^2})$

Authentication

Authorization

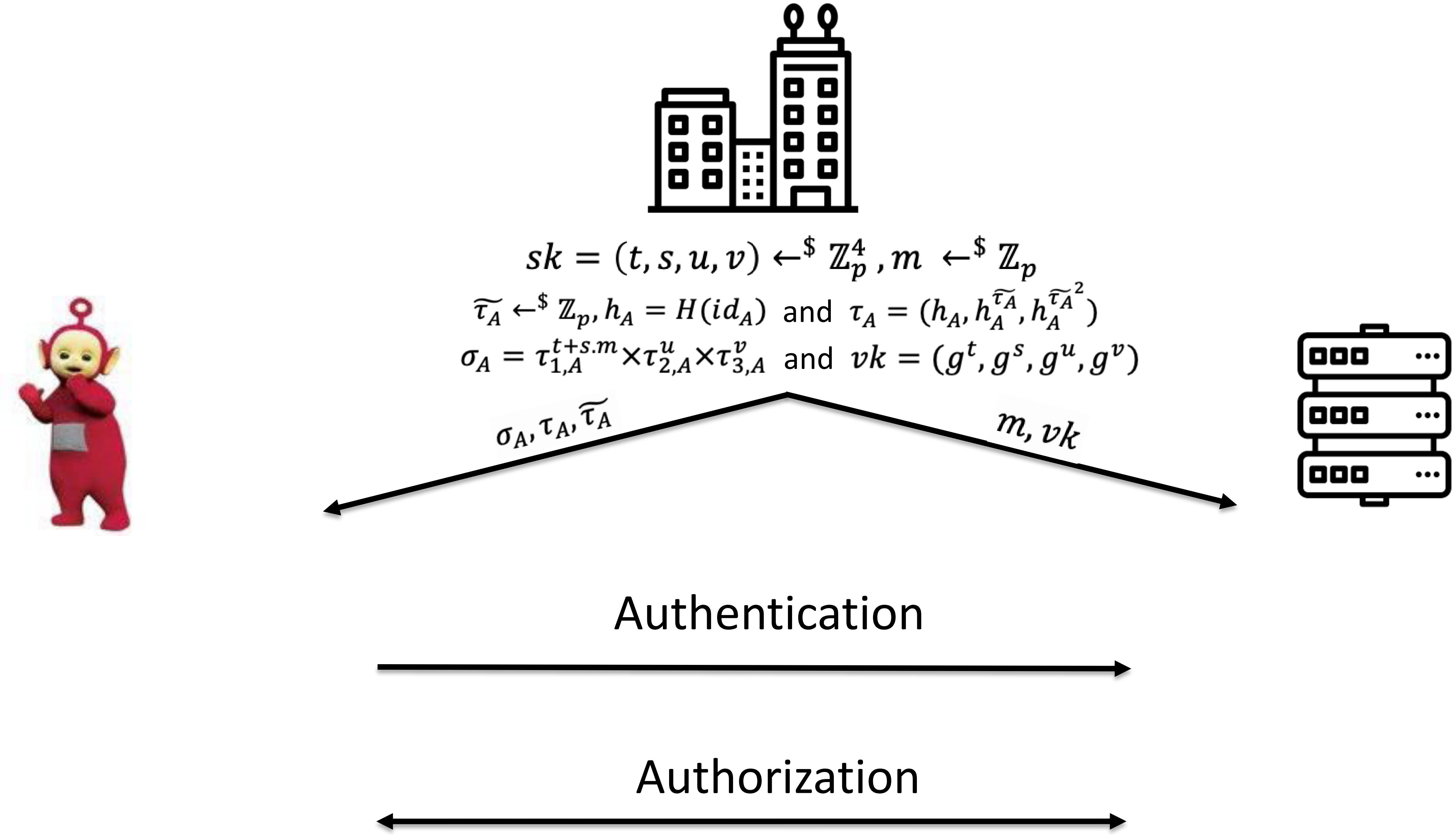
$$sk = (t, s, u, v) \leftarrow^{\$} \mathbb{Z}_p^4, m \leftarrow^{\$} \mathbb{Z}_p$$

$\widetilde{\tau_A} \leftarrow^{\$} \mathbb{Z}_p, h_A = H(id_A)$ and $\tau_A = (h_A, h_A^{\widetilde{\tau_A}}, h_A^{\widetilde{\tau_A}^2})$

$\sigma_A = \tau_{1,A}^{t+s.m} \times \tau_{2,A}^{u} \times \tau_{3,A}^{v}$ and $vk = (g^t, g^s, g^u, g^v)$

Authentication

Authorization



$sk = (t, s, u, v) \xleftarrow{\$} \mathbb{Z}_p^4, m \xleftarrow{\$} \mathbb{Z}_p$

$\widetilde{\tau_A} \xleftarrow{\$} \mathbb{Z}_p, h_A = H(id_A)$ and $\tau_A = (h_A, h_A^{\widetilde{\tau_A}}, h_A^{\widetilde{\tau_A}^2})$

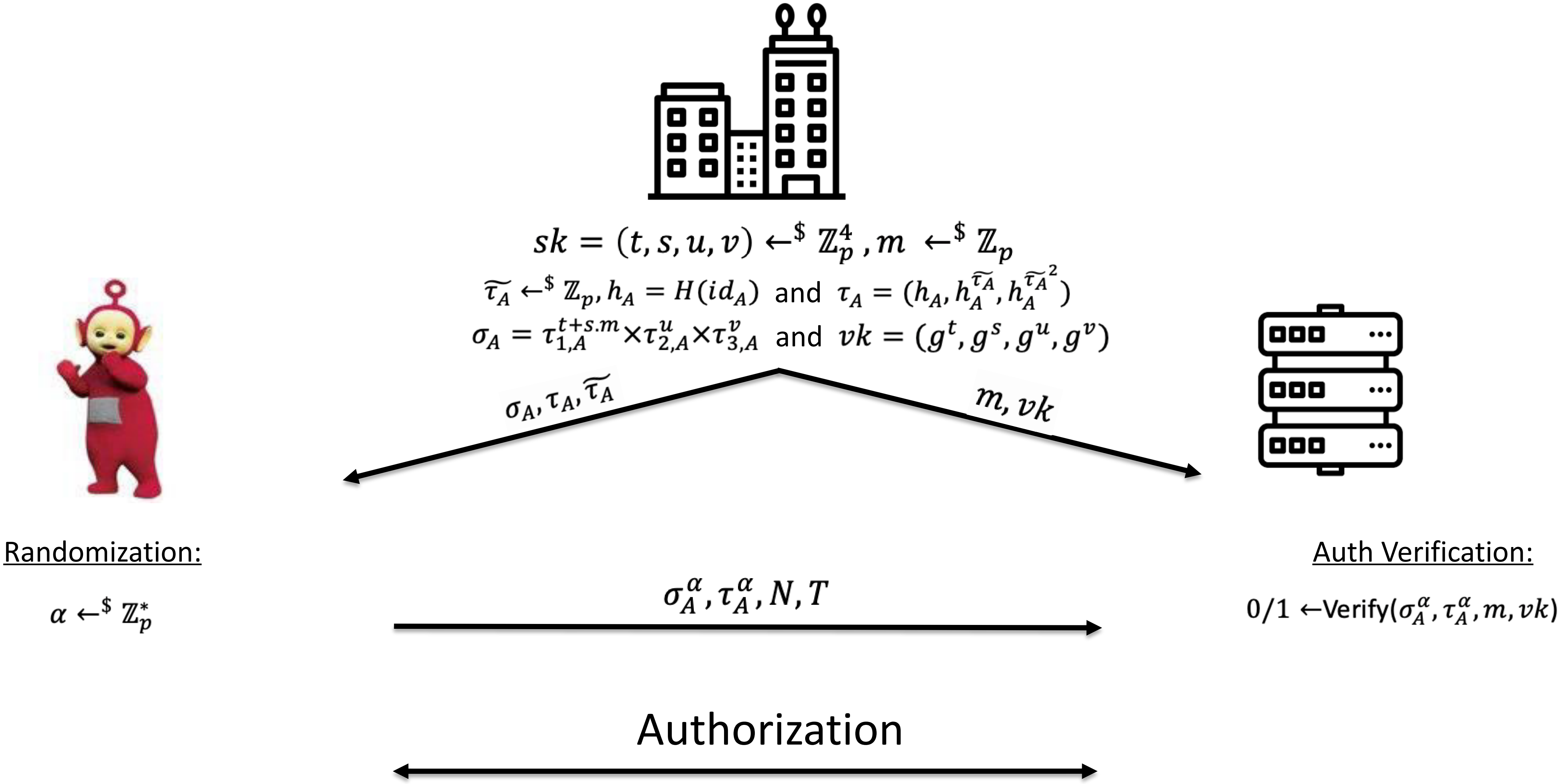
$\sigma_A = \tau_{1,A}^{t+s.m} \times \tau_{2,A}^{u} \times \tau_{3,A}^{v}$ and $vk = (g^t, g^s, g^u, g^v)$

$\sigma_A, \tau_A, \widetilde{\tau_A}$

m, vk

Authentication

Authorization



$sk = (t, s, u, v) \leftarrow^{\$} \mathbb{Z}_p^4 , m \leftarrow^{\$} \mathbb{Z}_p$

$\widetilde{\tau_A} \leftarrow^{\$} \mathbb{Z}_p, h_A = H(id_A)$ and $\tau_A = (h_A, h_A^{\widetilde{\tau_A}}, h_A^{\widetilde{\tau_A}^2})$

$\sigma_A = \tau_{1,A}^{t+s.m} \times \tau_{2,A}^{u} \times \tau_{3,A}^{v}$ and $vk = (g^t, g^s, g^u, g^v)$

$\sigma_A, \tau_A, \widetilde{\tau_A}$

m, vk

Randomization:

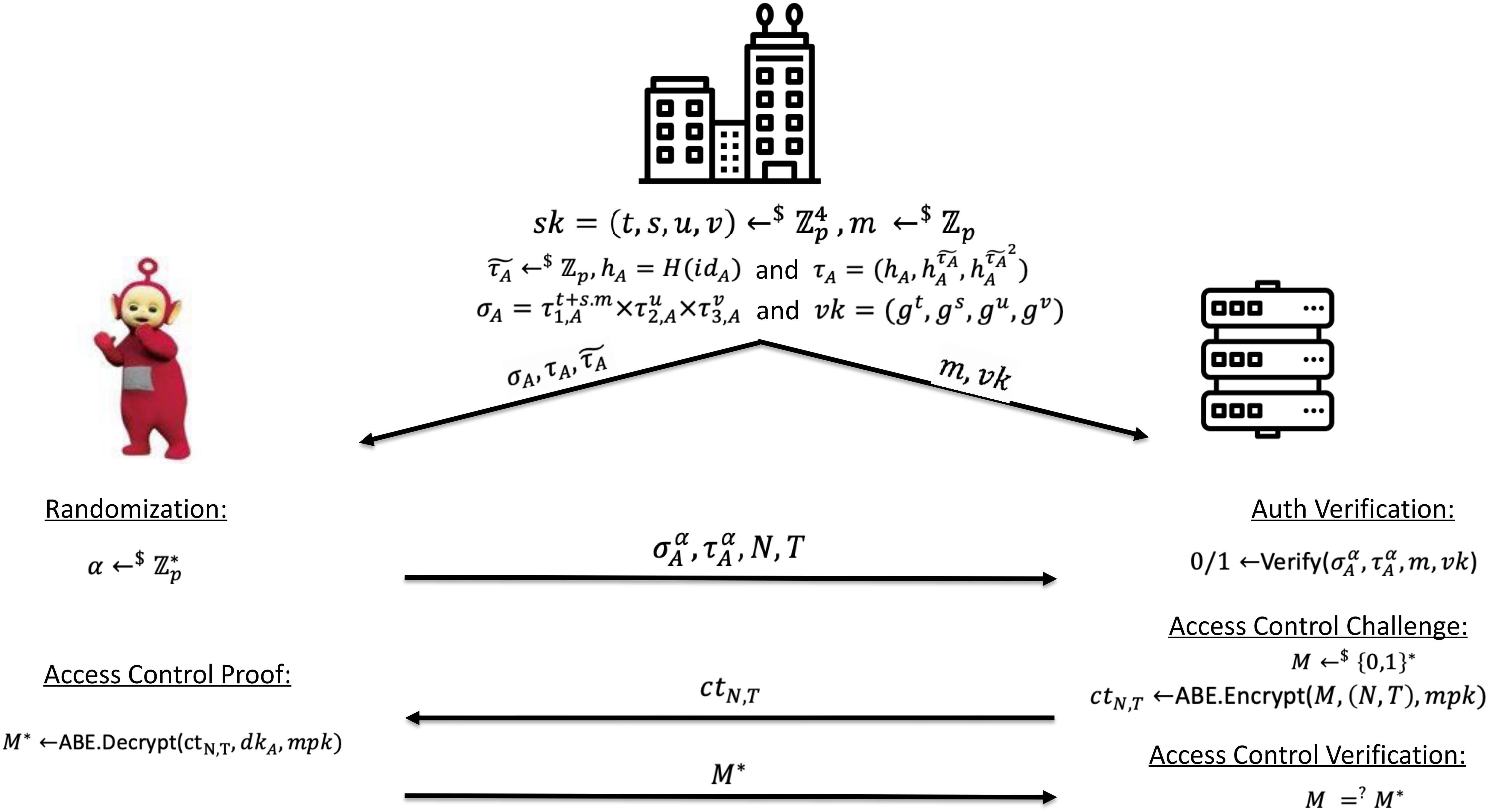
$\alpha \leftarrow^{\$} \mathbb{Z}_p^*$

$\sigma_A^{\alpha}, \tau_A^{\alpha}, N, T$

Auth Verification:

0/1 ←Verify($\sigma_A^{\alpha}, \tau_A^{\alpha}, m, vk$)

Authorization



$sk = (t, s, u, v) \leftarrow^{\$} \mathbb{Z}_p^4, m \leftarrow^{\$} \mathbb{Z}_p$

$\widetilde{\tau_A} \leftarrow^{\$} \mathbb{Z}_p, h_A = H(id_A)$ and $\tau_A = (h_A, h_A^{\widetilde{\tau_A}}, h_A^{\widetilde{\tau_A}^2})$

$\sigma_A = \tau_{1,A}^{t+s.m} \times \tau_{2,A}^{u} \times \tau_{3,A}^{v}$ and $vk = (g^t, g^s, g^u, g^v)$

$\sigma_A, \tau_A, \widetilde{\tau_A}$

m, vk

Randomization:

$\alpha \leftarrow^{\$} \mathbb{Z}_p^*$

$\sigma_A^\alpha, \tau_A^\alpha, N, T$

Auth Verification:

$0/1$ ←Verify($\sigma_A^\alpha, \tau_A^\alpha, m, vk$)

Access Control Challenge:

$M \leftarrow^{\$} \{0,1\}^*$

$ct_{N,T}$ ←ABE.Encrypt($M, (N, T), mpk$)

Access Control Proof:

$ct_{N,T}$

M^* ←ABE.Decrypt($\mathrm{ct}_{\mathrm{N,T}}, dk_A, mpk$)

M^*

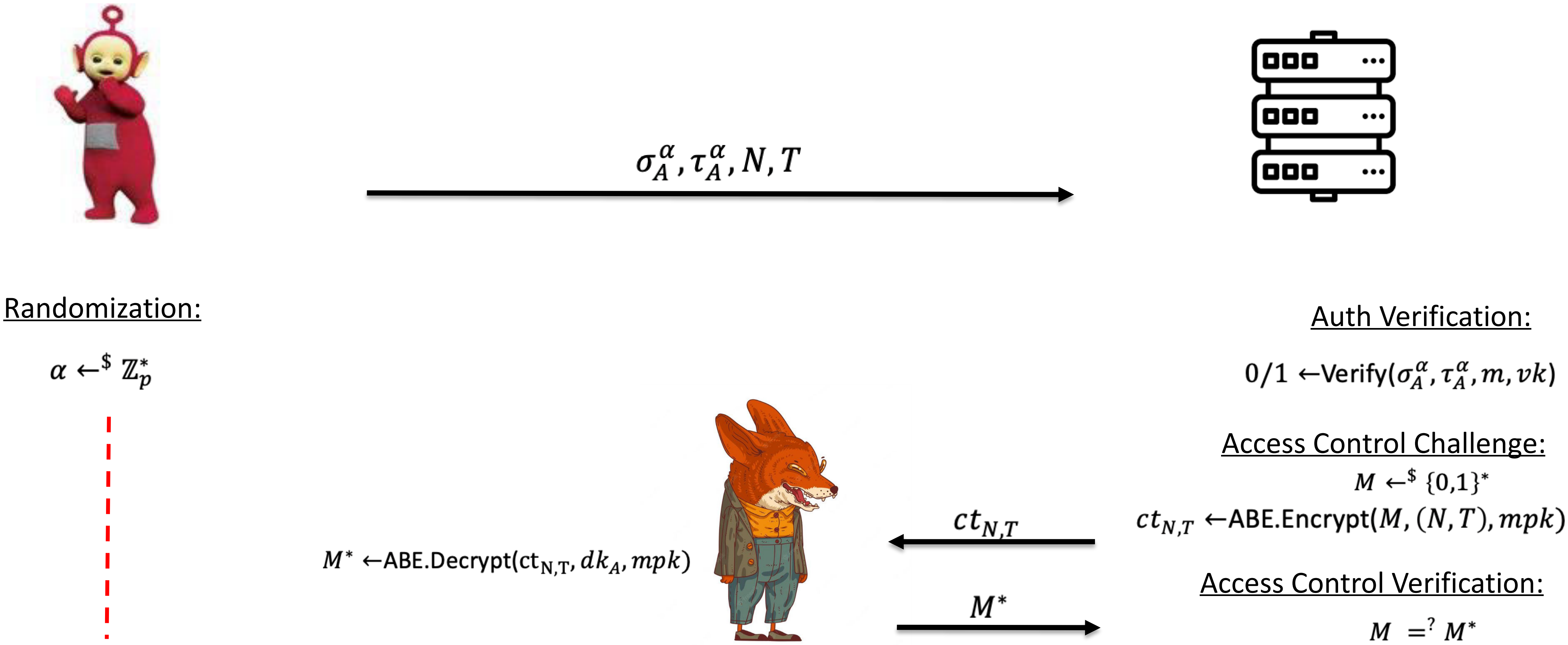
Access Control Verification:

$M =^? M^*$

Ephemeral Signing Key: a malicious employee appears!

As long as I have the capability to access a room queried by an employee, I can impersonate him and open the door !

A Malicious Employee attacks



$\sigma_A^\alpha, \tau_A^\alpha, N, T$

Randomization:

$\alpha \leftarrow^{\$} \mathbb{Z}_p^*$

Auth Verification:

$0/1 \leftarrow$Verify$(\sigma_A^\alpha, \tau_A^\alpha, m, vk)$

Access Control Challenge:

$M \leftarrow^{\$} \{0{,}1\}^*$

$ct_{N,T} \leftarrow$ABE.Encrypt$(M, (N, T), mpk)$

$ct_{N,T}$

$M^* \leftarrow$ABE.Decrypt$(\mathrm{ct}_{\mathrm{N,T}}, dk_A, mpk)$

M^*

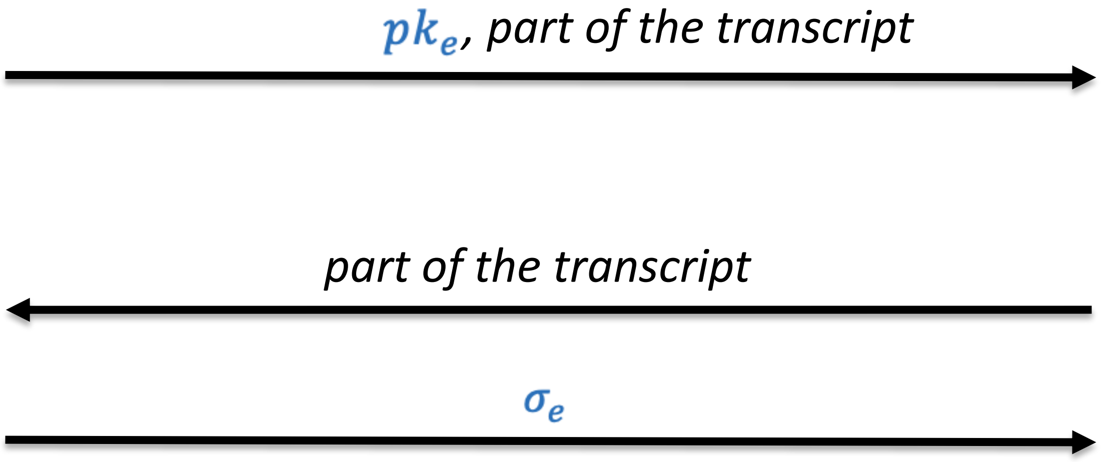
Access Control Verification:

$M \overset{?}{=} M^*$

Solution:
Ephemeral Signature

Randomization:

(pk_e, sk_e) ←Sign.Keygen()



pk_e, *part of the transcript*

Auth Verification:

$0/1$ ←Verify($\sigma_A^\alpha, \tau_A^\alpha, m, vk$)

Access Control Challenge:

$M \xleftarrow{\$} \{0,1\}^*$

$ct_{N,T}$ ←ABE.Encrypt($M, (N, T), mpk$)

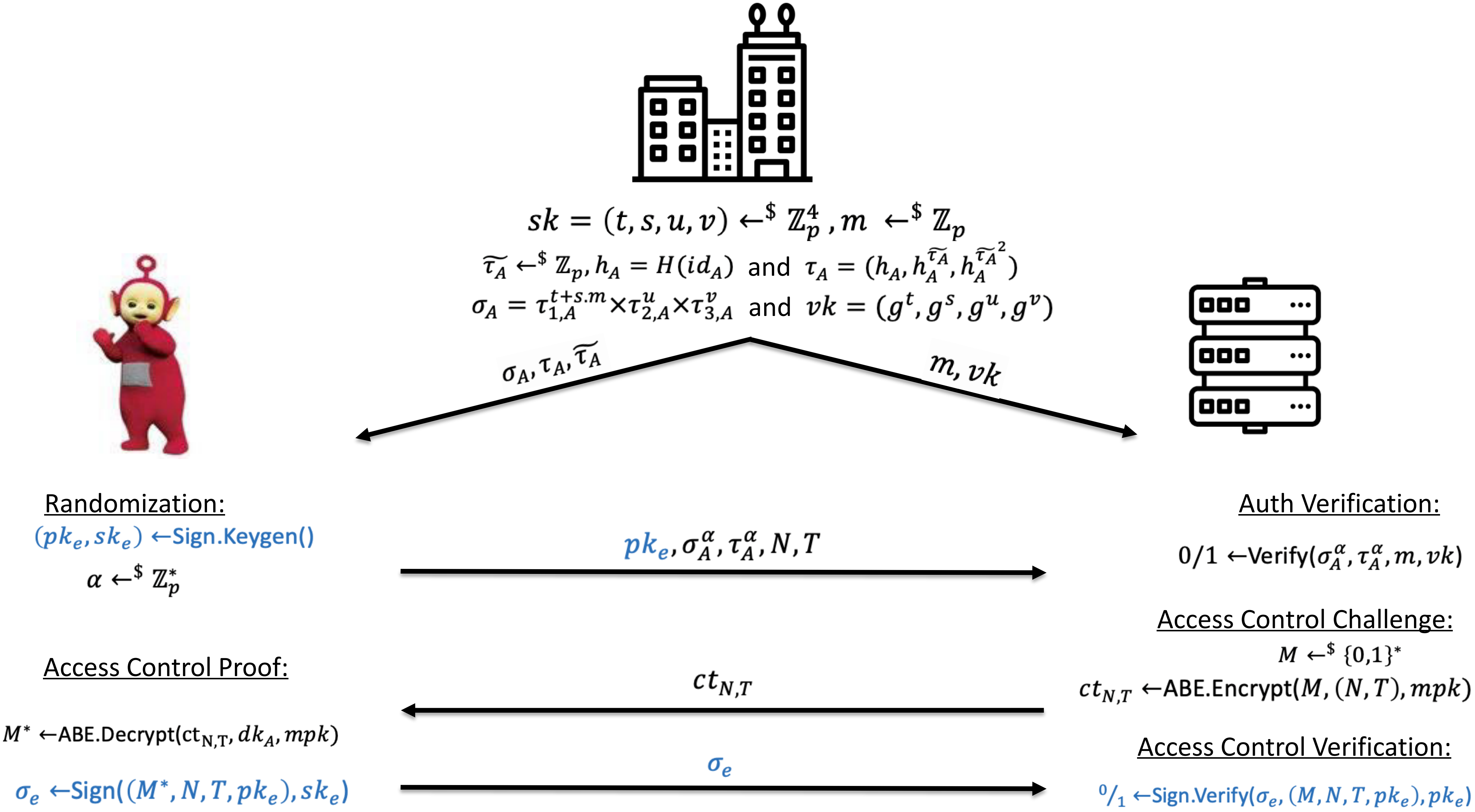
part of the transcript

σ_e ←Sign(transcript, sk_e)

σ_e

Access Control Verification:

$^0/_1$ ←Sign.Verify(σ_e, transcript, pk_e)



$sk = (t, s, u, v) \leftarrow^{\$} \mathbb{Z}_p^4 , m \leftarrow^{\$} \mathbb{Z}_p$

$\widetilde{\tau_A} \leftarrow^{\$} \mathbb{Z}_p, h_A = H(id_A)$ and $\tau_A = (h_A, h_A^{\widetilde{\tau_A}}, h_A^{\widetilde{\tau_A}^2})$

$\sigma_A = \tau_{1,A}^{t+s.m} \times \tau_{2,A}^{u} \times \tau_{3,A}^{v}$ and $vk = (g^t, g^s, g^u, g^v)$

$\sigma_A, \tau_A, \widetilde{\tau_A}$

m, vk

Randomization:

(pk_e, sk_e) ←Sign.Keygen()

$\alpha \leftarrow^{\$} \mathbb{Z}_p^*$

$pk_e, \sigma_A^{\alpha}, \tau_A^{\alpha}, N, T$

Auth Verification:

$0/1$ ←Verify($\sigma_A^{\alpha}, \tau_A^{\alpha}, m, vk$)

Access Control Challenge:

$M \leftarrow^{\$} \{0,1\}^*$

$ct_{N,T}$ ←ABE.Encrypt($M, (N, T), mpk$)

Access Control Proof:

$ct_{N,T}$

M^* ←ABE.Decrypt($\mathrm{ct}_{\mathrm{N,T}}, dk_A, mpk$)

Access Control Verification:

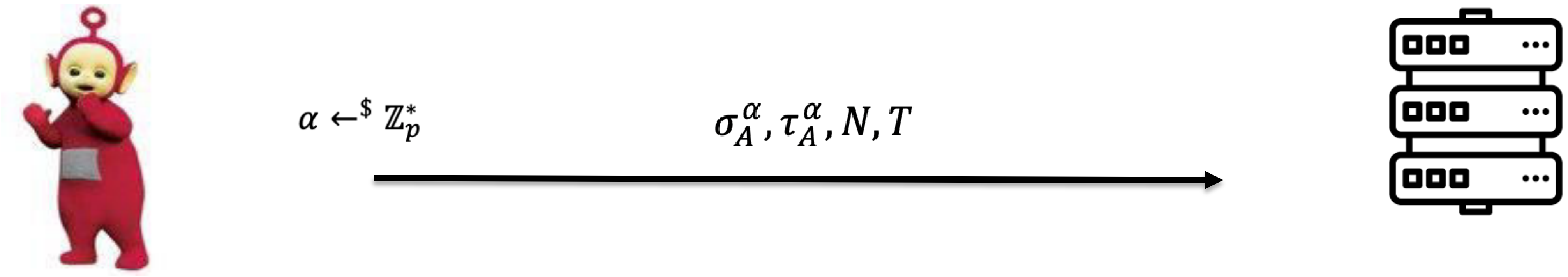
σ_e

σ_e ←Sign($(M^*, N, T, pk_e), sk_e$)

$^0/_1$ ←Sign.Verify($\sigma_e, (M, N, T, pk_e), pk_e$)

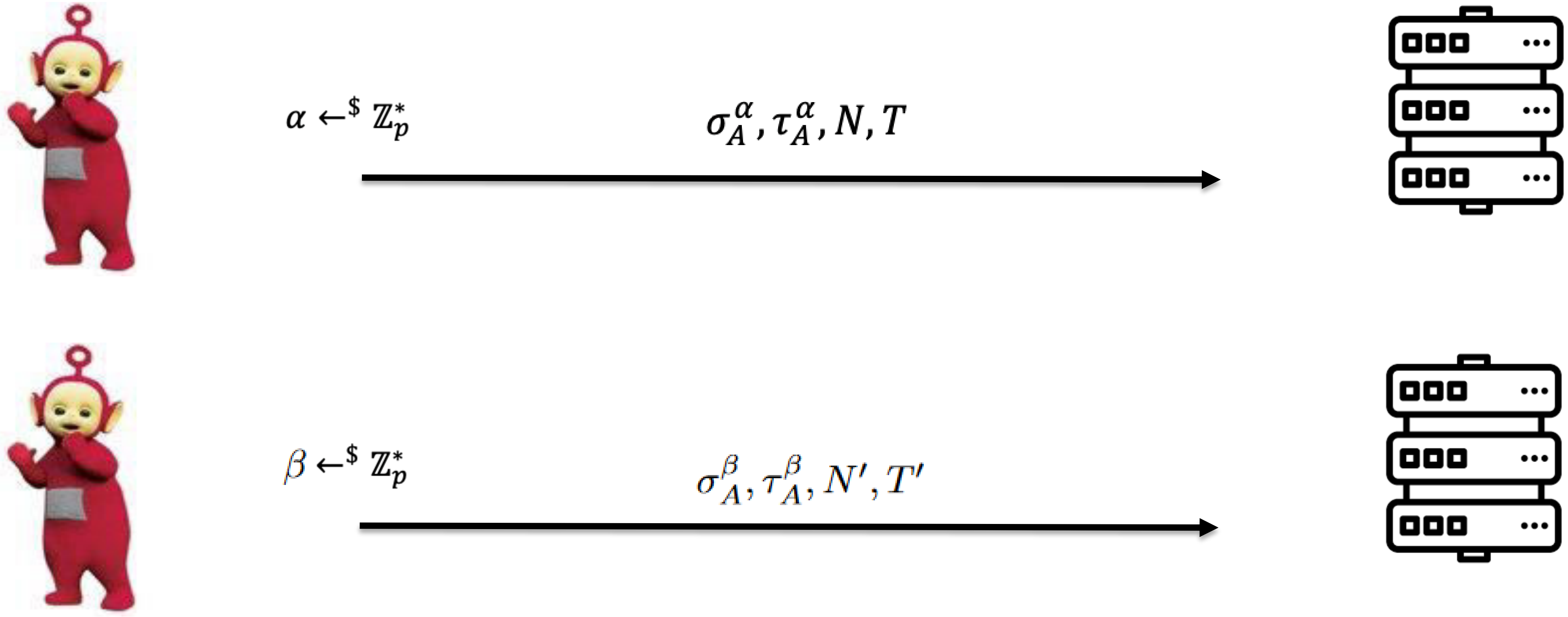
⚠ A Malicious Employee is back ⚠

Using the Signature Homomorphism the Malicious Employee records messages and then creates a valid signature.



$\alpha \leftarrow^{\$} \mathbb{Z}_p^*$

$\sigma_A^{\alpha}, \tau_A^{\alpha}, N, T$

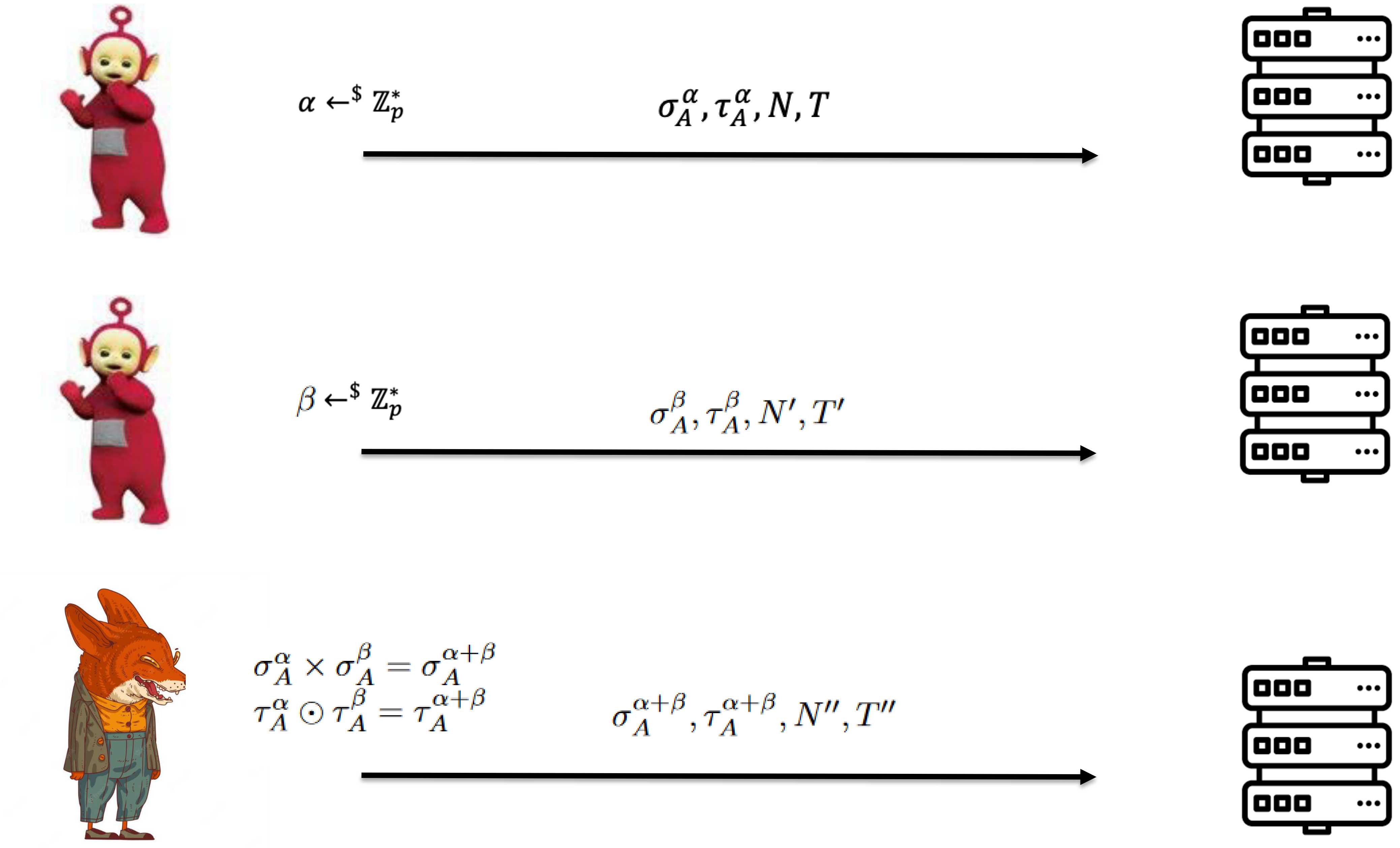


$\alpha \xleftarrow{\$} \mathbb{Z}_p^*$

$\sigma_A^{\alpha}, \tau_A^{\alpha}, N, T$

$\beta \xleftarrow{\$} \mathbb{Z}_p^*$

$\sigma_A^{\beta}, \tau_A^{\beta}, N', T'$



$\alpha \leftarrow^{\$} \mathbb{Z}_p^*$

$\sigma_A^{\alpha}, \tau_A^{\alpha}, N, T$

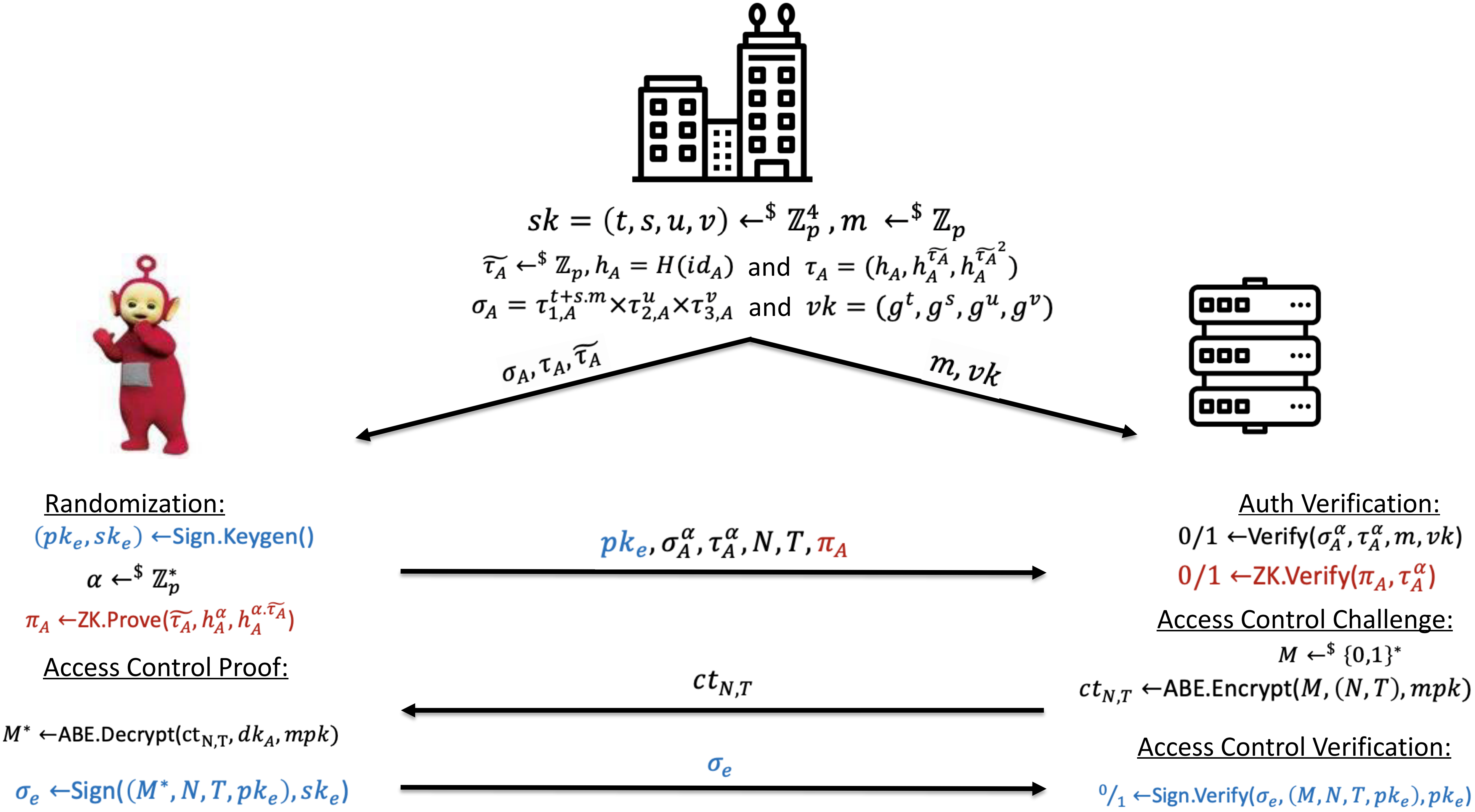
$\beta \leftarrow^{\$} \mathbb{Z}_p^*$

$\sigma_A^{\beta}, \tau_A^{\beta}, N', T'$

$\sigma_A^{\alpha} \times \sigma_A^{\beta} = \sigma_A^{\alpha+\beta}$

$\tau_A^{\alpha} \odot \tau_A^{\beta} = \tau_A^{\alpha+\beta}$

$\sigma_A^{\alpha+\beta}, \tau_A^{\alpha+\beta}, N'', T''$



$sk = (t, s, u, v) \xleftarrow{\$} \mathbb{Z}_p^4 , m \xleftarrow{\$} \mathbb{Z}_p$

$\widetilde{\tau_A} \xleftarrow{\$} \mathbb{Z}_p, h_A = H(id_A)$ and $\tau_A = (h_A, h_A^{\widetilde{\tau_A}}, h_A^{\widetilde{\tau_A}^2})$

$\sigma_A = \tau_{1,A}^{t+s.m} \times \tau_{2,A}^{u} \times \tau_{3,A}^{v}$ and $vk = (g^t, g^s, g^u, g^v)$

$\sigma_A, \tau_A, \widetilde{\tau_A}$

m, vk

Randomization:

(pk_e, sk_e) ←Sign.Keygen()

$\alpha \xleftarrow{\$} \mathbb{Z}_p^*$

π_A ←ZK.Prove($\widetilde{\tau_A}, h_A^{\alpha}, h_A^{\alpha.\widetilde{\tau_A}}$)

Access Control Proof:

M^* ←ABE.Decrypt($\mathrm{ct}_{\mathrm{N,T}}, dk_A, mpk$)

σ_e ←Sign($(M^*, N, T, pk_e), sk_e$)

$pk_e, \sigma_A^{\alpha}, \tau_A^{\alpha}, N, T, \pi_A$

$ct_{N,T}$

σ_e

Auth Verification:

0/1 ←Verify($\sigma_A^{\alpha}, \tau_A^{\alpha}, m, vk$)

0/1 ←ZK.Verify(π_A, τ_A^{α})

Access Control Challenge:

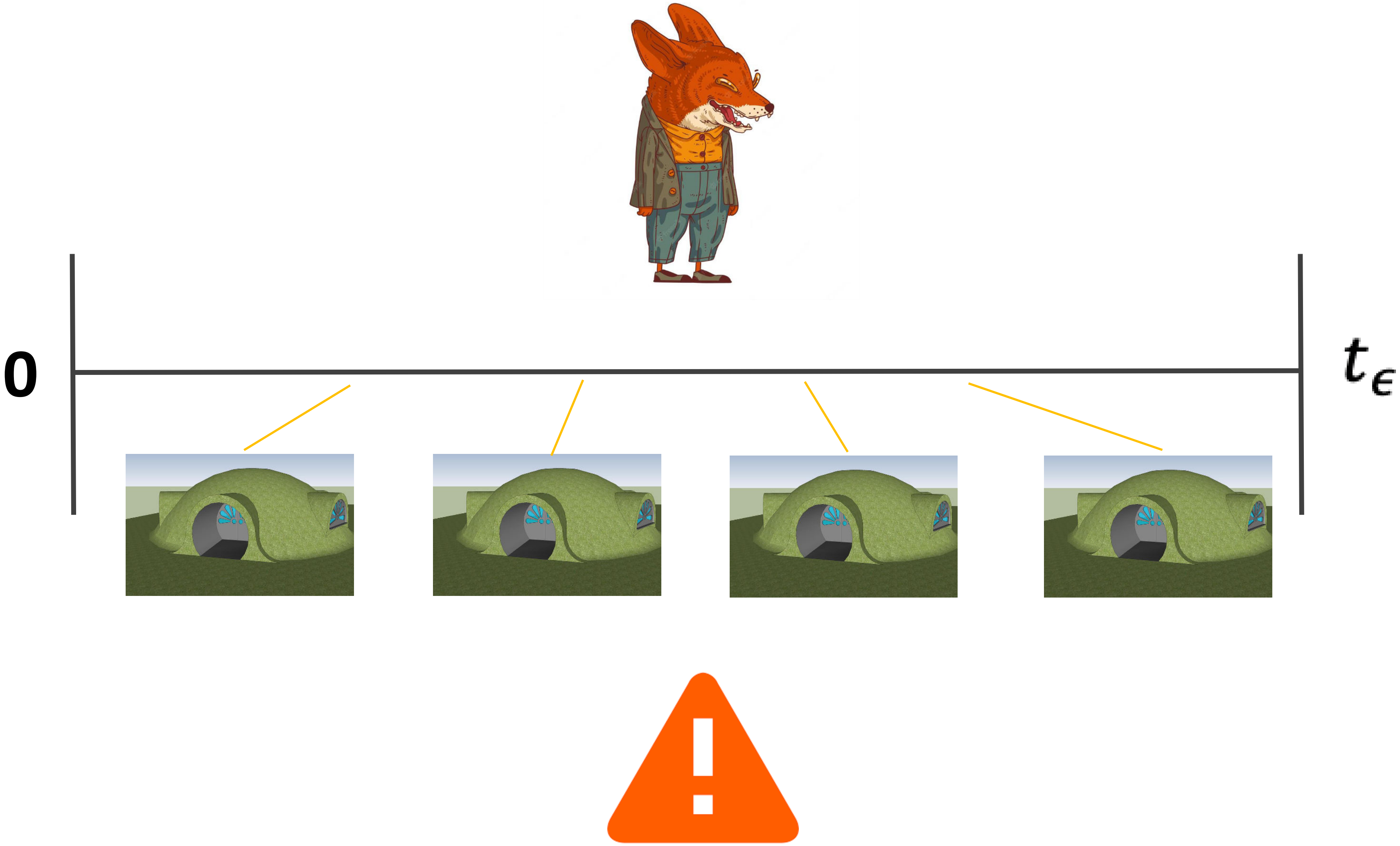
$M \xleftarrow{\$} \{0,1\}^*$

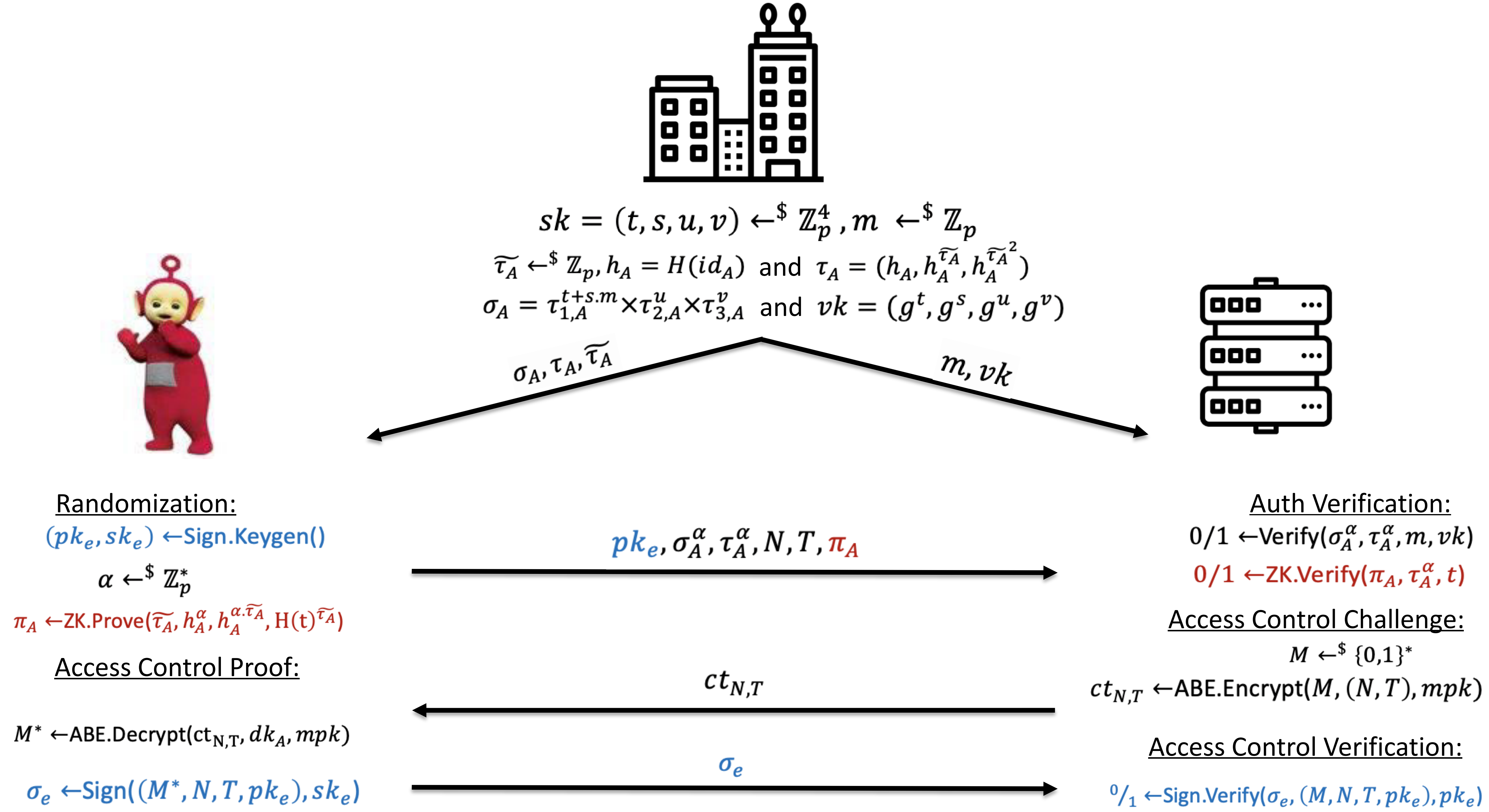
$ct_{N,T}$ ←ABE.Encrypt($M, (N, T), mpk$)

Access Control Verification:

$^0/_1$ ←Sign.Verify($\sigma_e, (M, N, T, pk_e), pk_e$)

Last Property: Enabling Partial Traceability





$sk = (t, s, u, v) \xleftarrow{\$} \mathbb{Z}_p^4 , m \xleftarrow{\$} \mathbb{Z}_p$

$\widetilde{\tau_A} \xleftarrow{\$} \mathbb{Z}_p, h_A = H(id_A)$ and $\tau_A = (h_A, h_A^{\widetilde{\tau_A}}, h_A^{\widetilde{\tau_A}^2})$

$\sigma_A = \tau_{1,A}^{t+s.m} \times \tau_{2,A}^{u} \times \tau_{3,A}^{v}$ and $vk = (g^t, g^s, g^u, g^v)$

$\sigma_A, \tau_A, \widetilde{\tau_A}$

m, vk

Randomization:

(pk_e, sk_e) ←Sign.Keygen()

$\alpha \xleftarrow{\$} \mathbb{Z}_p^*$

π_A ←ZK.Prove($\widetilde{\tau_A}, h_A^{\alpha}, h_A^{\alpha.\widetilde{\tau_A}}, \mathrm{H(t)}^{\widetilde{\tau_A}}$)

Access Control Proof:

M^* ←ABE.Decrypt($\mathrm{ct}_{\mathrm{N,T}}, dk_A, mpk$)

σ_e ←Sign($(M^*, N, T, pk_e), sk_e$)

$pk_e, \sigma_A^{\alpha}, \tau_A^{\alpha}, N, T, \pi_A$

$ct_{N,T}$

σ_e

Auth Verification:

0/1 ←Verify($\sigma_A^{\alpha}, \tau_A^{\alpha}, m, vk$)

0/1 ←ZK.Verify($\pi_A, \tau_A^{\alpha}, t$)

Access Control Challenge:

$M \xleftarrow{\$} \{0,1\}^*$

$ct_{N,T}$ ←ABE.Encrypt($M, (N, T), mpk$)

Access Control Verification:

${}^0/_1$ ←Sign.Verify($\sigma_e, (M, N, T, pk_e), pk_e$)

VII. Advanced Properties

Traceable-Anonymous Randomizable Signature

- Definition

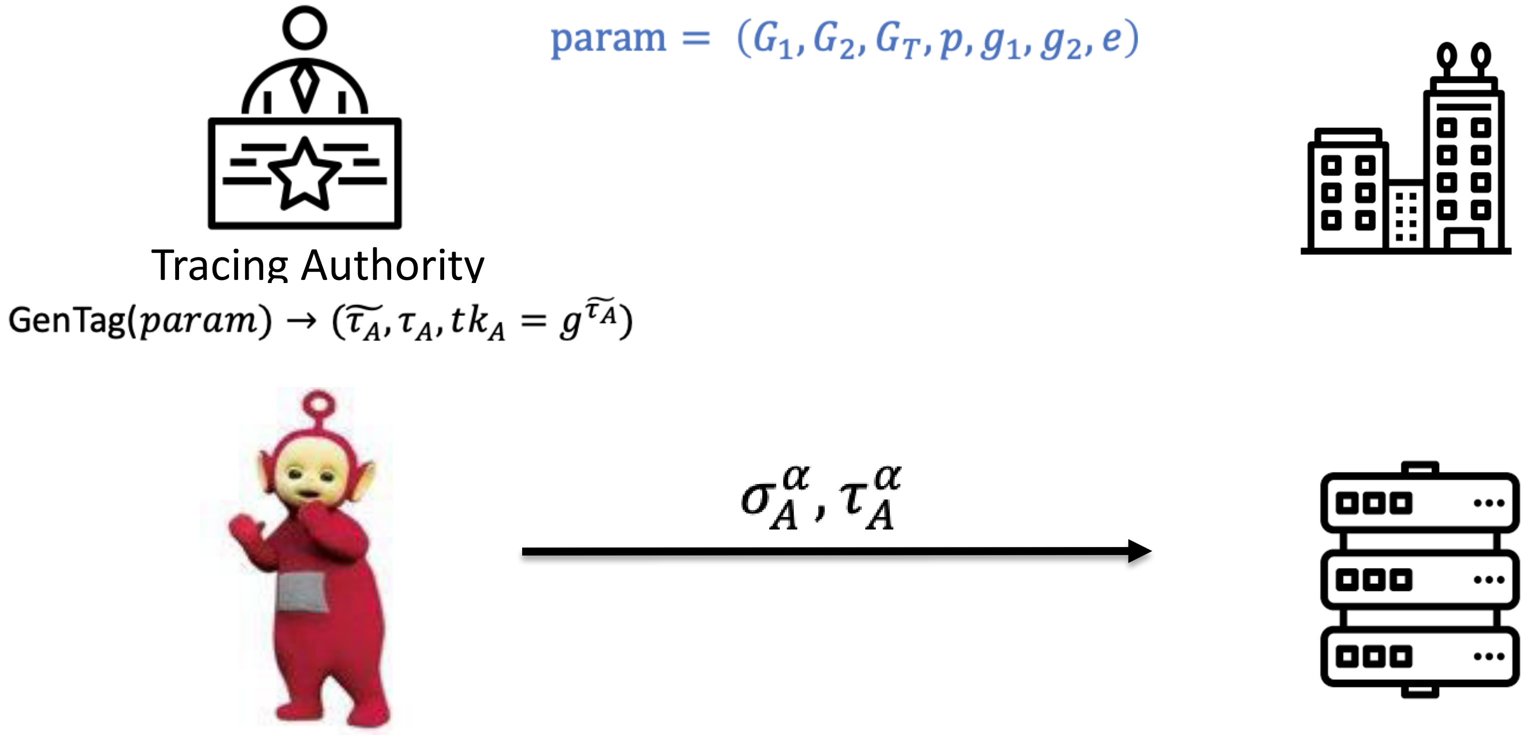
Main Idea: Anonymous but traceable tags
Tracing authority can revoke anonymity (traceability), and publish the identity of the guilty, without being able to accuse an innocent (non-frameability).

- Setup$(1^\lambda) \rightarrow param$
- Keygen$(param) \rightarrow (sk, vk)$
- GenTag$(param) \rightarrow (\tilde{\tau}, \tau, tk)$
- Sign$(m, sk, \tau) \rightarrow \sigma$
- Verify$(\sigma, \tau, m, vk) \rightarrow$ 1 if σ valid relative to vk and τ, 0 otherwise.
- RandSign$(\sigma, \tau, m, vk, \alpha) \rightarrow \sigma'$ on m under the randomized tag τ' and the same key vk.
- TraceId$(tk, \tau') \rightarrow \pi$ of whether, for tk associated to τ, $tk \sim \tau'$ or not
- JudgeId$(\tau, \tau', \pi) \rightarrow$ 1 if π is correct.

Traceable-Anonymous Randomizable Signature

- Definition

Main Idea: Anonymous but traceable tags
Tracing authority can revoke anonymity (traceability), and publish the identity of the guilty, without being able to accuse an innocent (non-frameability).



$\mathrm{param} = (G_1, G_2, G_T, p, g_1, g_2, e)$

Tracing Authority

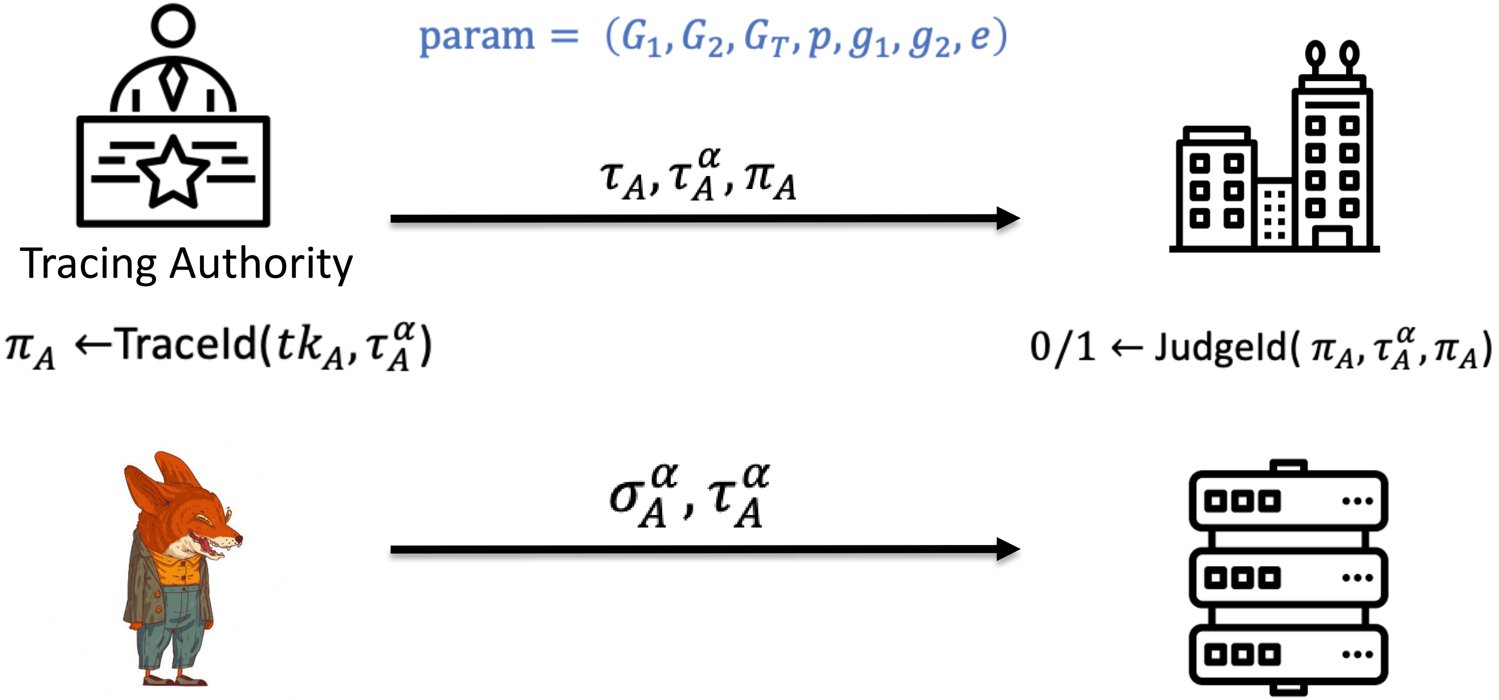
GenTag$(param) \rightarrow (\widetilde{\tau_A}, \tau_A, tk_A = g^{\widetilde{\tau_A}})$

$\sigma_A^{\alpha}, \tau_A^{\alpha}$

Traceable-Anonymous Randomizable Signature

- Definition

Main Idea: Anonymous but traceable tags
Tracing authority can revoke anonymity (traceability), and publish the identity of the guilty, without being able to accuse an innocent (non-frameability).



param = $(G_1, G_2, G_T, p, g_1, g_2, e)$

$\tau_A, \tau_A^\alpha, \pi_A$

Tracing Authority

$\pi_A \leftarrow \text{TraceId}(tk_A, \tau_A^\alpha)$

$0/1 \leftarrow \text{JudgeId}(\pi_A, \tau_A^\alpha, \pi_A)$

$\sigma_A^\alpha, \tau_A^\alpha$

VIII. Discussion

What does the protocol Accomplishes ?

- Anonymity towards the server.
- Local traceability but non global traceability towards the server.
- Traceability or identification (as wanted) towards the authority.

Limitations ?

- 1-collusion : If two adversaries that are employees collaborate, one can sign and the other get access.

- If one uses the CoverCrypt implementation for the ABE, two adversaries that are employees can create a third unknown key that give access to the union of their rights. Although one of them must sign.