zk-ABAC
Un sistema di controllo degli accessi con garanzie di privacy basato su zero-knowledge

Anno Accademico 2019/2020

Relatori:

Prof. Laura Ricci

Dott. Damiano di Francesco Maesa

Candidato:

Gianluca Boschi

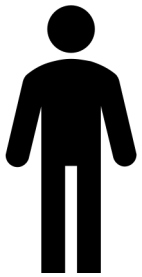
Access Control

Tecnica per decidere se un **Soggetto** che richiede di eseguire una **Azione** su una **Risorsa** in certo **Contesto** ha effettivamente il diritto di eseguirla

Access Control

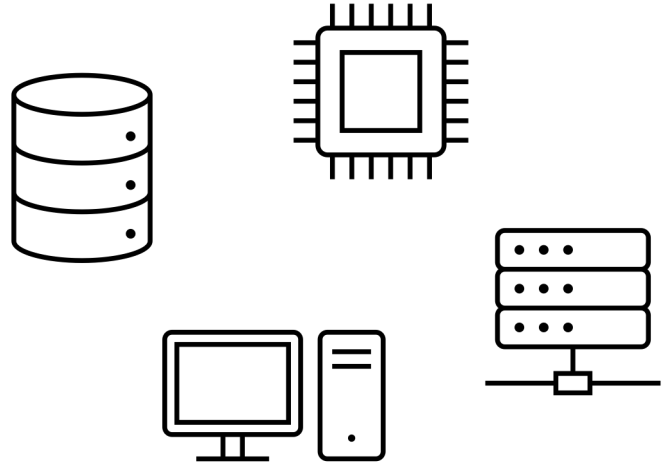
Access Control

Tecnica per decidere se un **Soggetto** che richiede di eseguire una **Azione** su una **Risorsa** in certo **Contesto** ha effettivamente il diritto di eseguirla



Soggetto

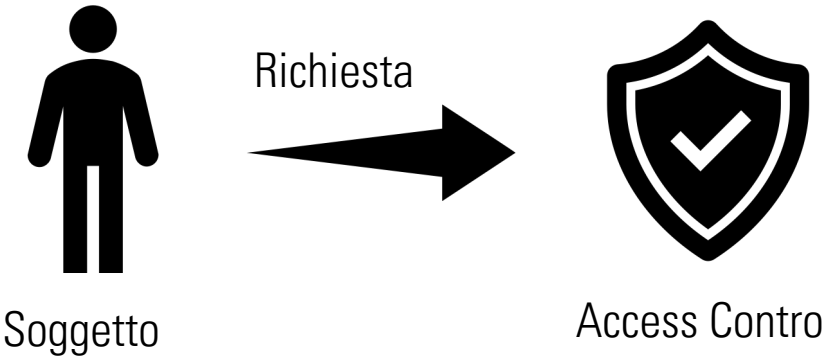
Access Control

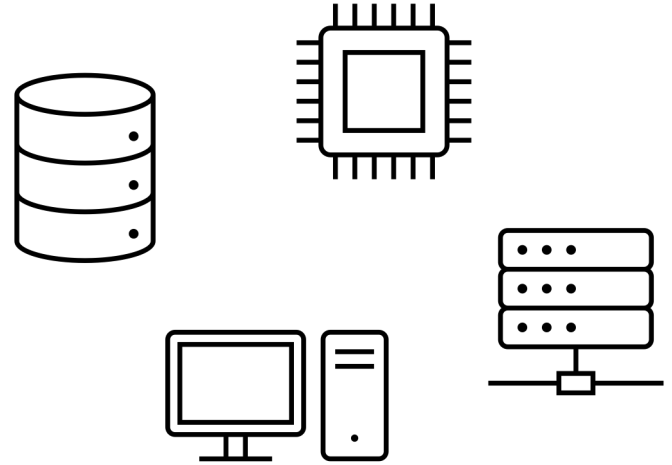


Risorse

Access Control

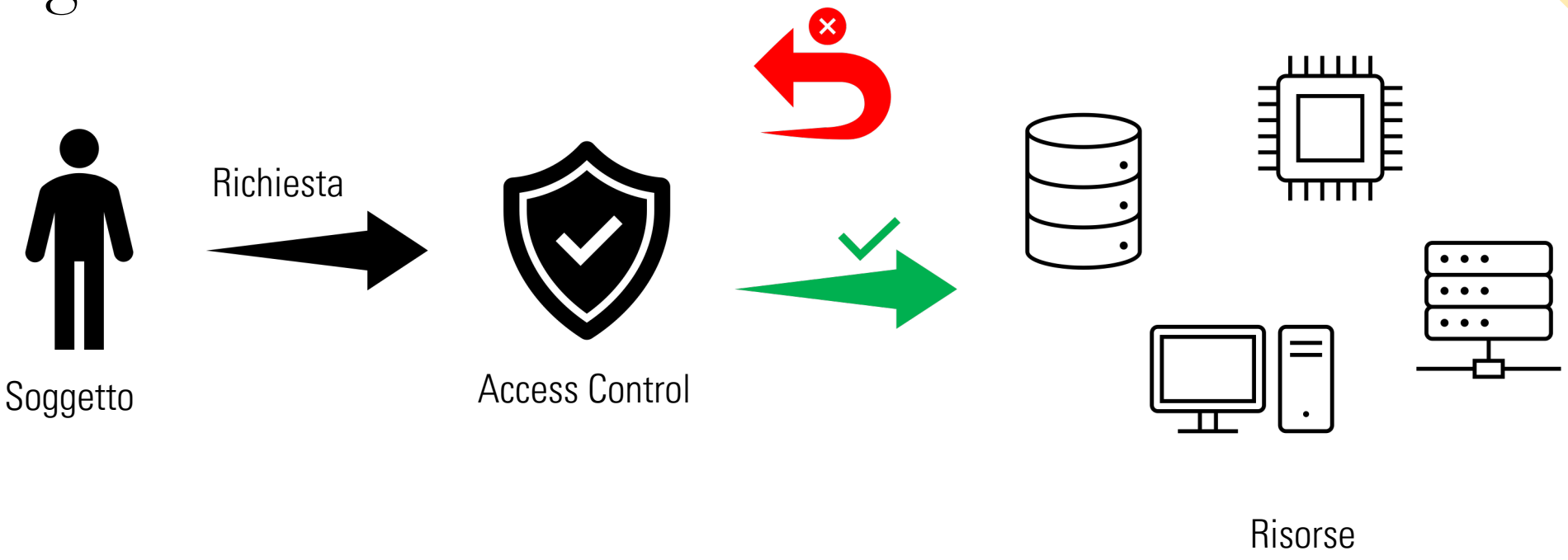
Tecnica per decidere se un **Soggetto** che richiede di eseguire una **Azione** su una **Risorsa** in certo **Contesto** ha effettivamente il diritto di eseguirla





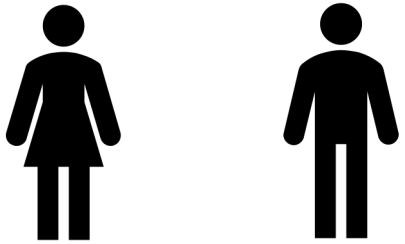
Access Control

Tecnica per decidere se un **Soggetto** che richiede di eseguire una **Azione** su una **Risorsa** in certo **Contesto** ha effettivamente il diritto di eseguirla



Attributi

Definiscono le **caratteristiche** di Soggetti, Risorse, Ambienti



Soggetti

- Ruolo aziendale
- Età
- Stipendio
- Salario

Risorse

- Proprietario
- Data di creazione
- Tipologia
- Dimensioni

Access Control ABAC

Def. "È un modello di access control in cui la richiesta di un soggetto di eseguire una operazione su una risorsa viene accettata o respinta sulla base degli attributi assegnati al soggetto, all'oggetto, all'ambiente, e alle politiche di access control che sono specificate sulla base di tali attributi"

Guide to Attribute Based Access Control (ABAC) Definition and Considerations.
NIST Special Publication 800-162

NIST National Institute of Standards and Technology U.S. Department of Commerce

Il documento può essere letto dal Soggetto se l'attributo **"DIPARTIMENTO"** del Soggetto ha valore **"INFORMATICA"**

La base: Access Control ABAC su blockchain

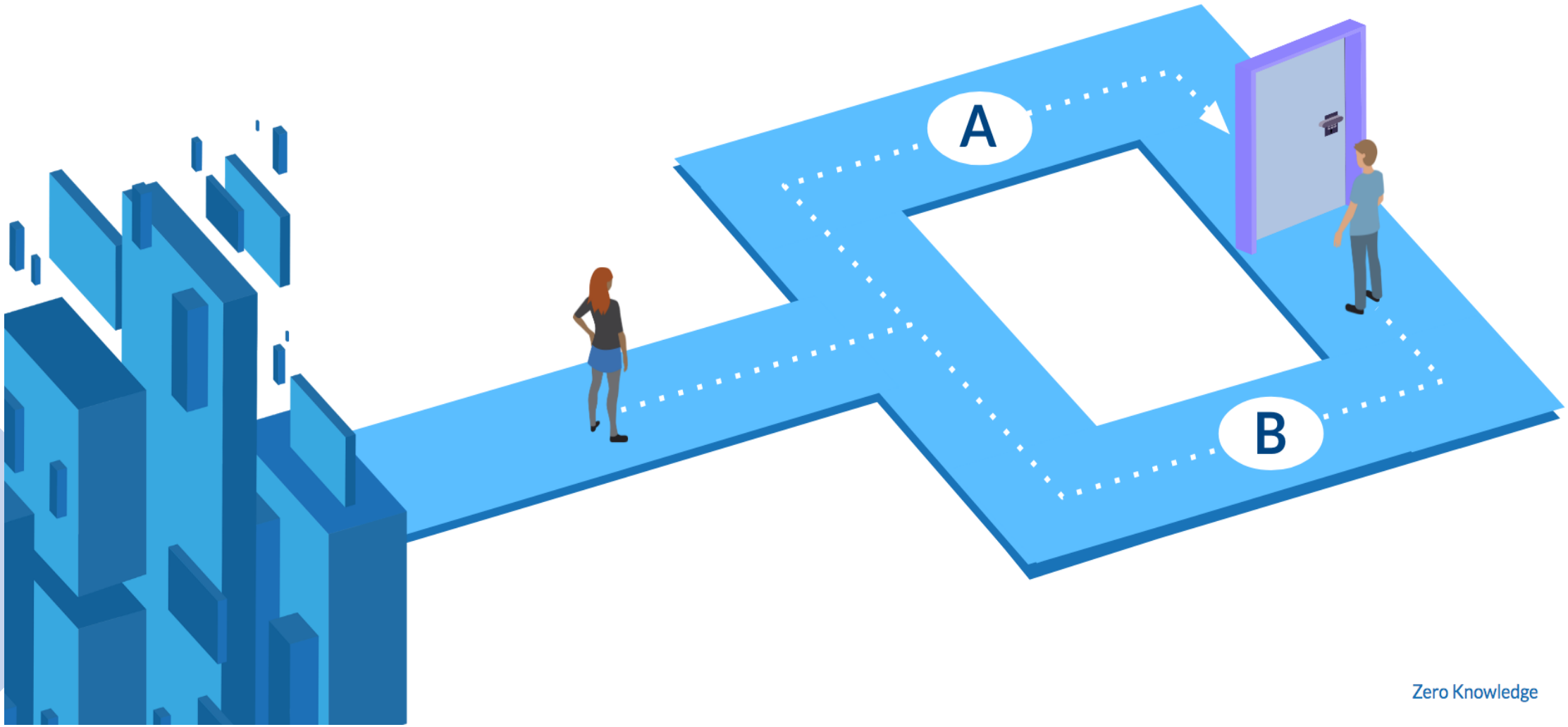
Il progetto parte da un Access Control ABAC implementato su blockchain e basato su **standard XACML,** sviluppato in **[1]**

Problema nell'utilizzo degli **attributi privati** su blockchain

Utilizzo di tecniche **Zero-Knowledge**

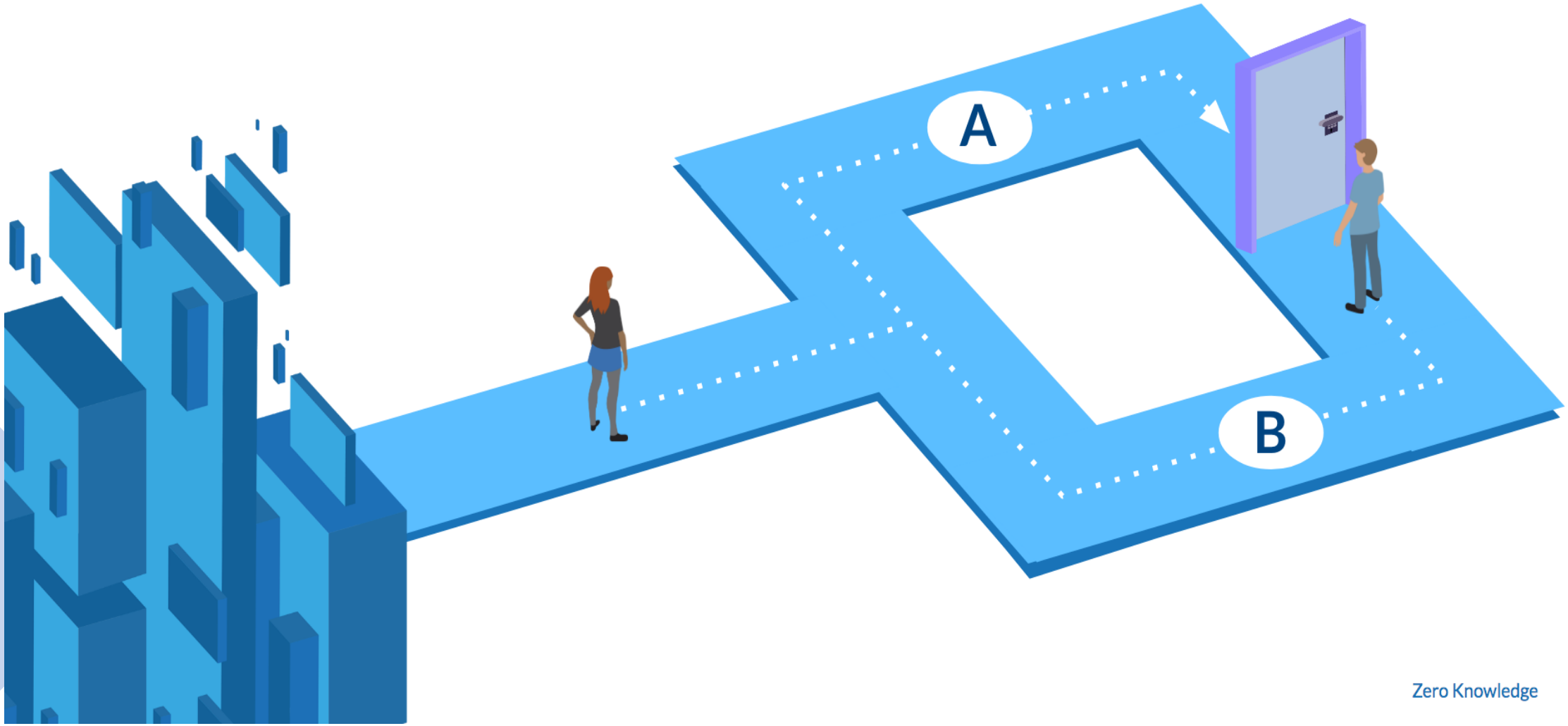
[1] *Damiano Di Francesco Maesa, Paolo Mori e Laura Ricci. ≪A blockchain based approach for the definition of auditable Access Control systems≫. In: Computers & Security 84 (2019), pp. 93–119.*

Zero Knowledge



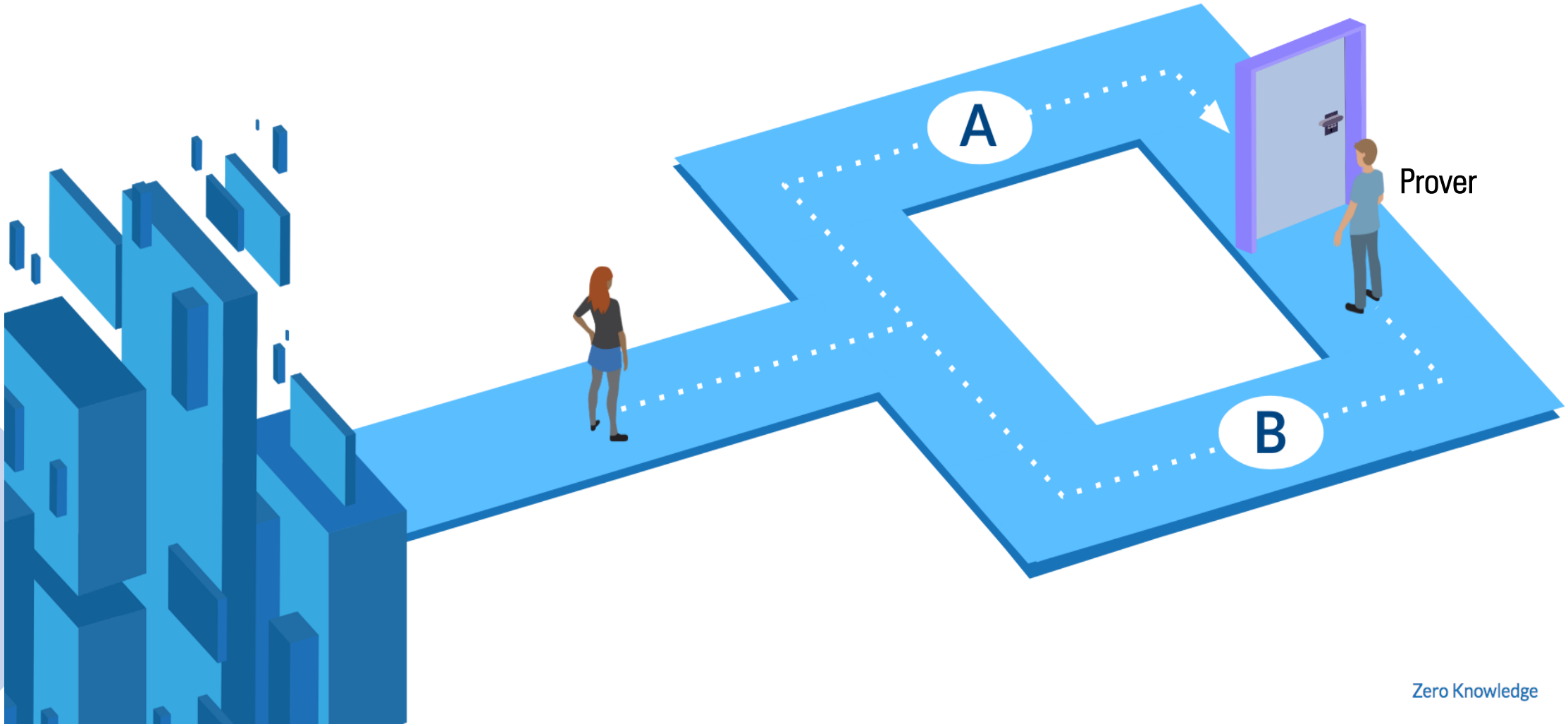
Zero Knowledge

*Tecnica **crittografica** per lo scambio di dati*



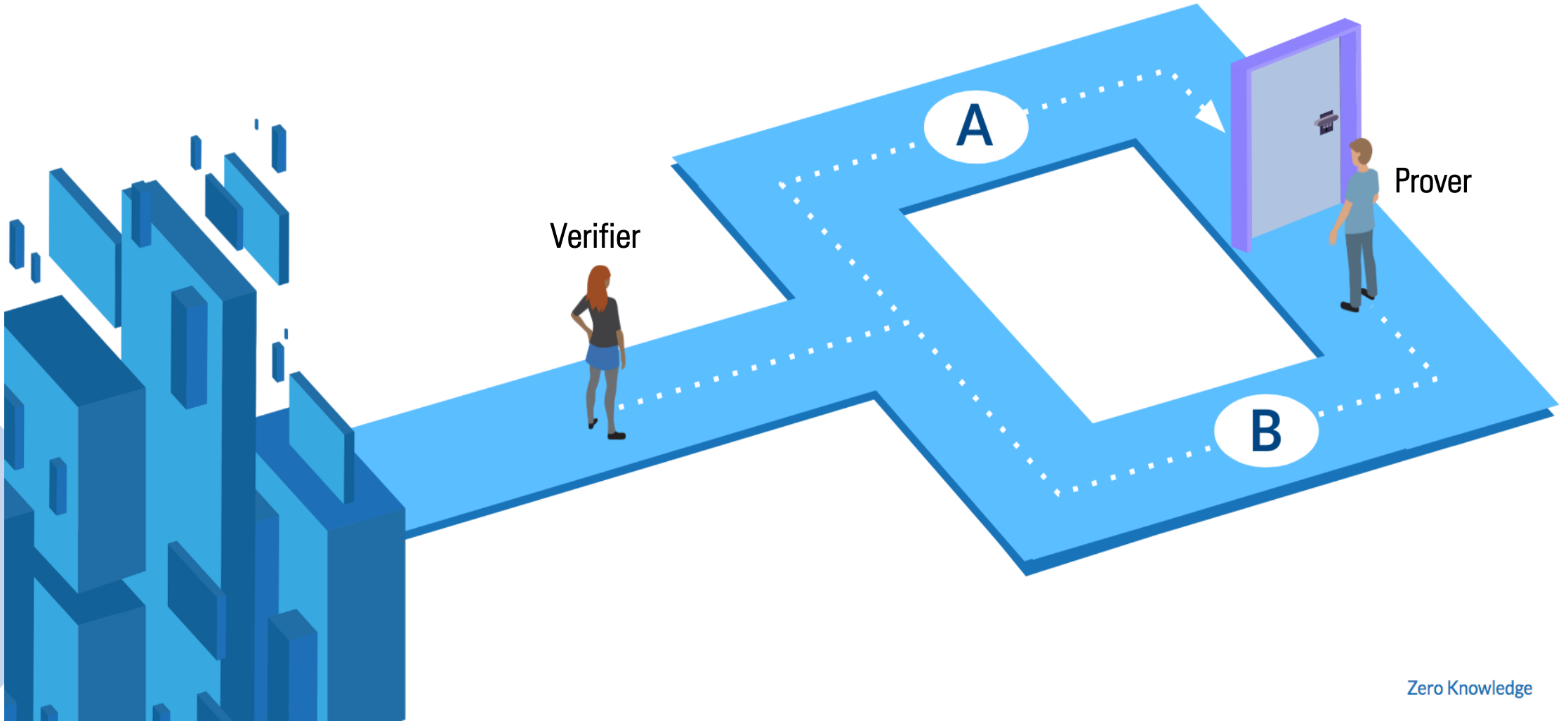
Zero Knowledge

*Tecnica **crittografica** per lo scambio di dati*



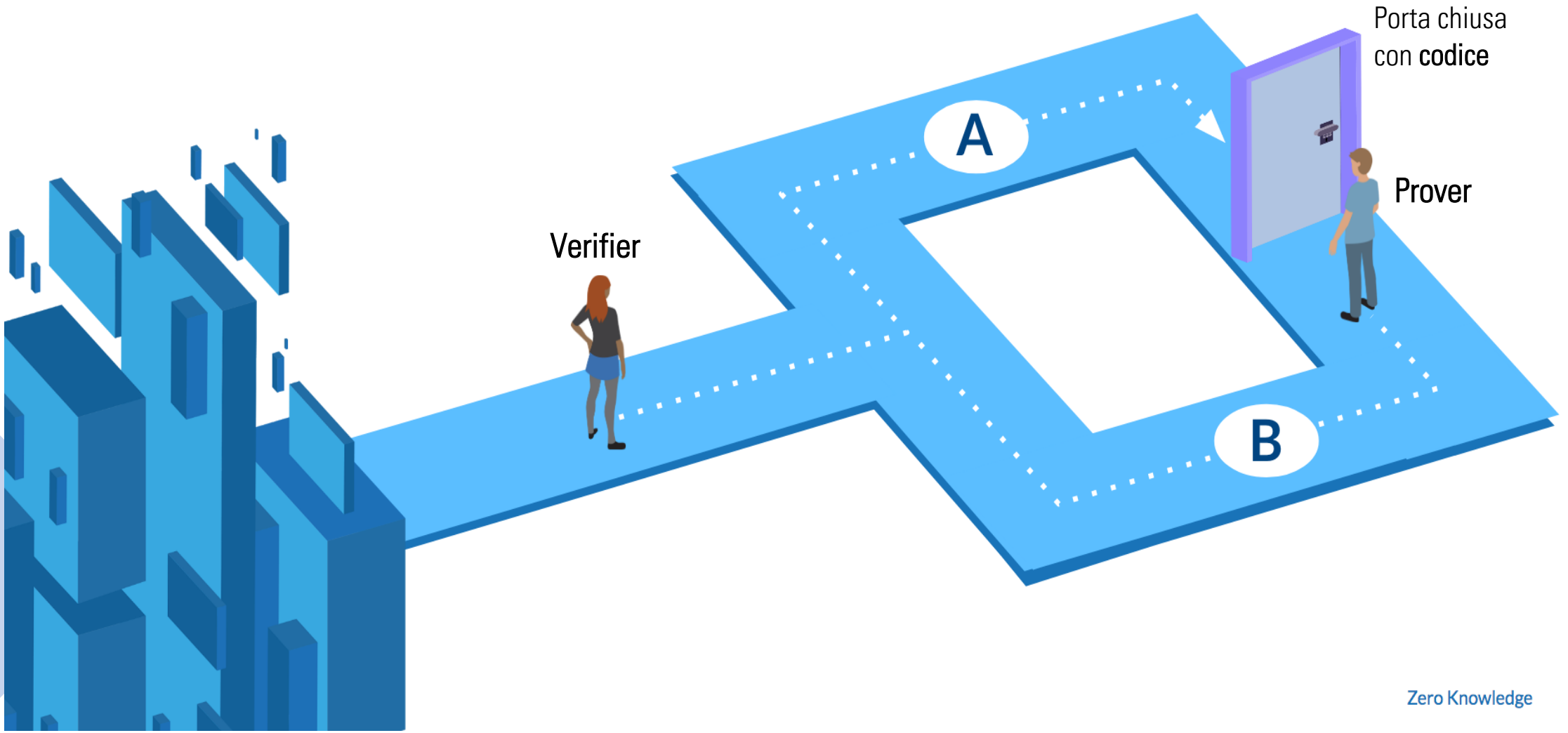
Zero Knowledge

*Tecnica **crittografica** per lo scambio di dati*



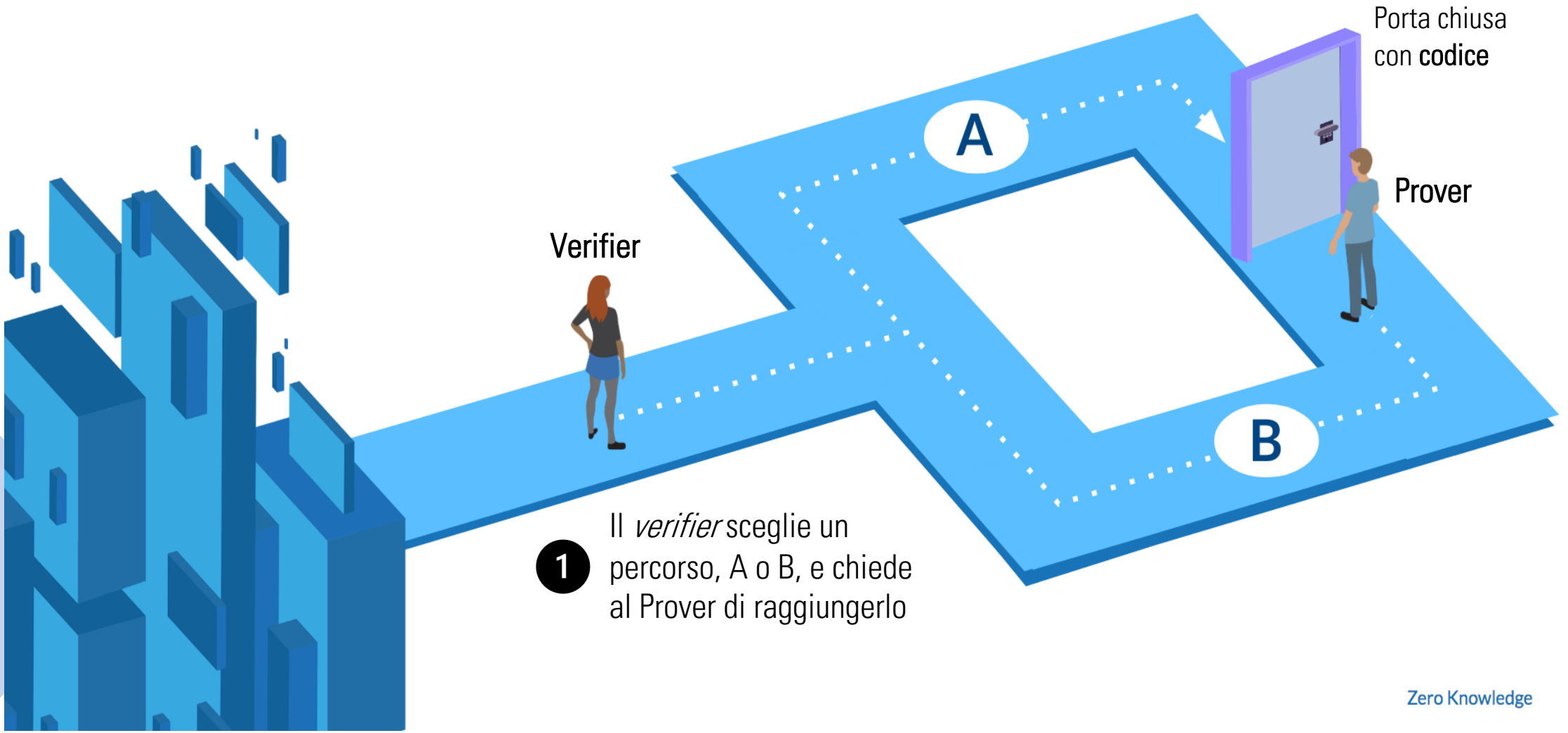
Zero Knowledge

*Tecnica **crittografica** per lo scambio di dati*



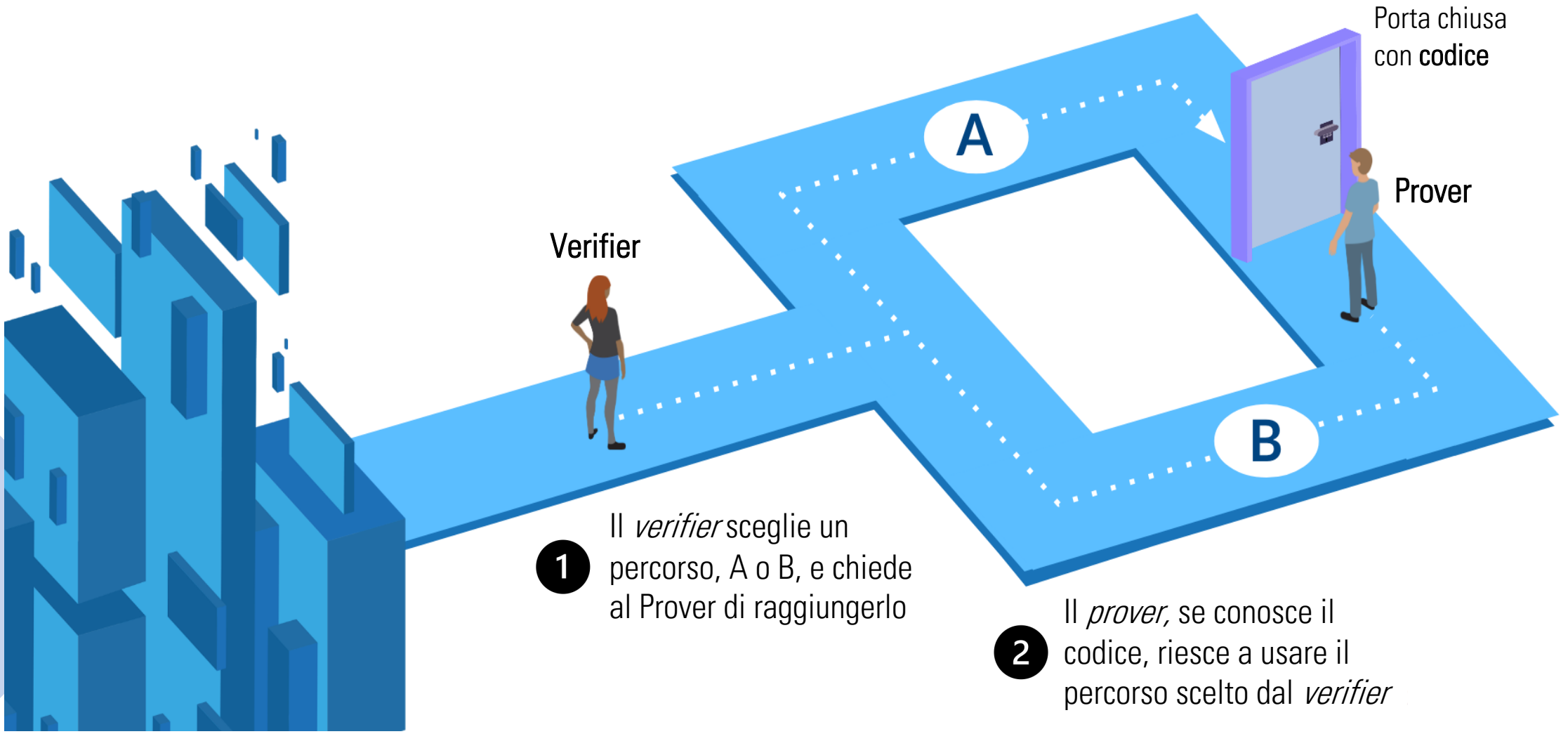
Zero Knowledge

*Tecnica **crittografica** per lo scambio di dati*



Zero Knowledge

*Tecnica **crittografica** per lo scambio di dati*

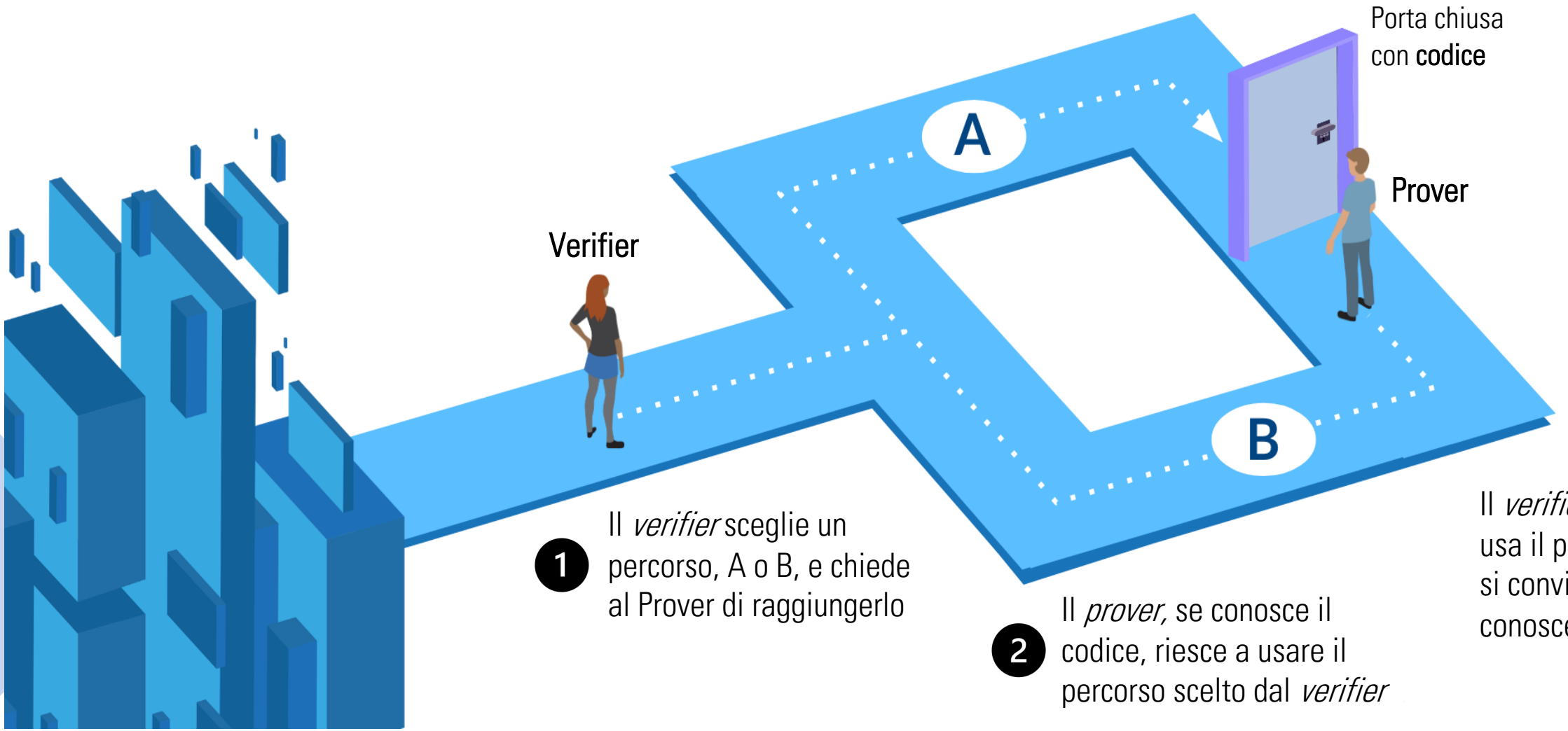


1. Il *verifier* sceglie un percorso, A o B, e chiede al Prover di raggiungerlo

2. Il *prover,* se conosce il codice, riesce a usare il percorso scelto dal *verifier*

Zero Knowledge

*Tecnica **crittografica** per lo scambio di dati*



1. Il *verifier* sceglie un percorso, A o B, e chiede al Prover di raggiungerlo
2. Il *prover,* se conosce il codice, riesce a usare il percorso scelto dal *verifier*

Il *verifier,* se il *prover* usa il percorso giusto si convince che conosce il codice

ZoKrates

Tool che implementa le Zero-Knowledge

Permette di creare un verificatore in linguaggio Solidity

Utilizza schemi **zk-SNARKs**

ZoKrates

Tool che implementa le Zero-Knowledge

Permette di creare un verificatore in linguaggio Solidity

Utilizza schemi **zk-SNARKs**

- Prove di **piccole dimensioni**
- Protocollo **non-interattivo**
- Creare prove errate è difficile
- Creare prove corrette senza conoscere il witness è impossibile

ZoKrates

Tool che implementa le Zero-Knowledge

Permette di creare un verificatore in linguaggio Solidity

Utilizza schemi **zk-SNARKs**

- Prove di **piccole dimensioni**
- Protocollo **non-interattivo**
- Creare prove errate è difficile
- Creare prove corrette senza conoscere il witness è impossibile

ZoKrates

Tool che implementa le Zero-Knowledge

Permette di creare un verificatore in linguaggio Solidity

Utilizza schemi **zk-SNARKs**

- Prove di **piccole dimensioni**
- Protocollo **non-interattivo**
- Creare prove errate è difficile
- Creare prove corrette senza conoscere il witness è impossibile

file.zok

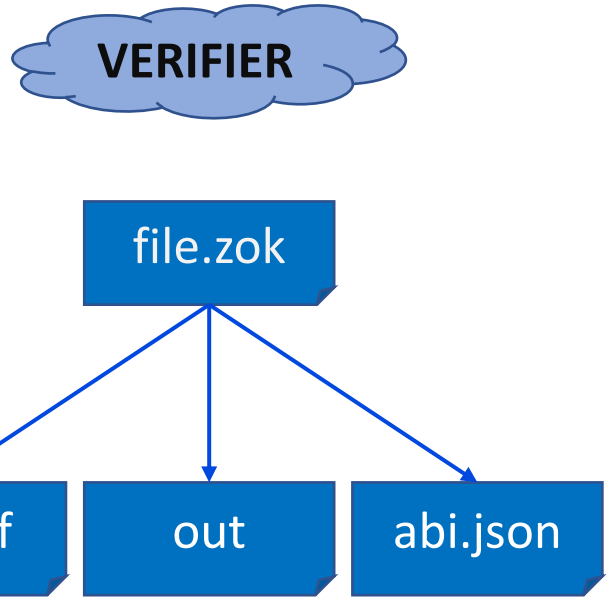
ZoKrates

Tool che implementa le Zero-Knowledge

Permette di creare un verificatore in linguaggio Solidity

Utilizza schemi **zk-SNARKs**

- Prove di **piccole dimensioni**
- Protocollo **non-interattivo**
- Creare prove errate è difficile
- Creare prove corrette senza conoscere il witness è impossibile



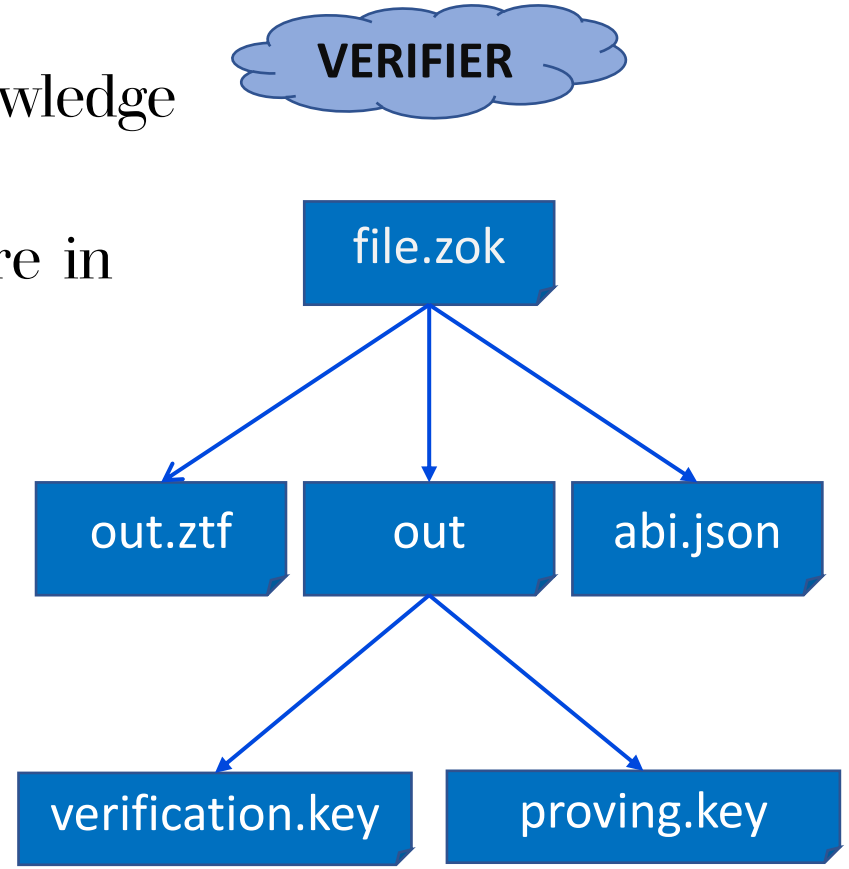
ZoKrates

Tool che implementa le Zero-Knowledge

Permette di creare un verificatore in linguaggio Solidity

Utilizza schemi **zk-SNARKs**

- Prove di **piccole dimensioni**
- Protocollo **non-interattivo**
- Creare prove errate è difficile
- Creare prove corrette senza conoscere il witness è impossibile



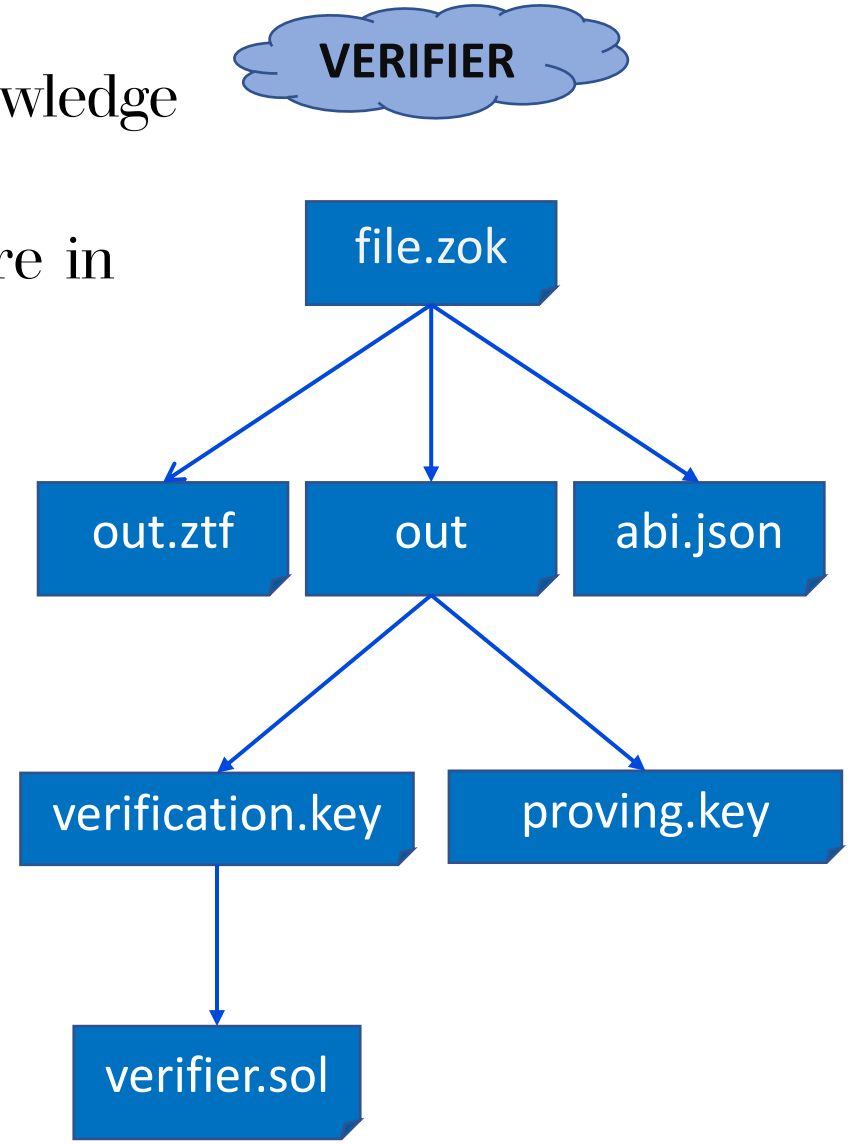
ZoKrates

Tool che implementa le Zero-Knowledge

Permette di creare un verificatore in linguaggio Solidity

Utilizza schemi **zk-SNARKs**

- Prove di **piccole dimensioni**
- Protocollo **non-interattivo**
- Creare prove errate è difficile
- Creare prove corrette senza conoscere il witness è impossibile



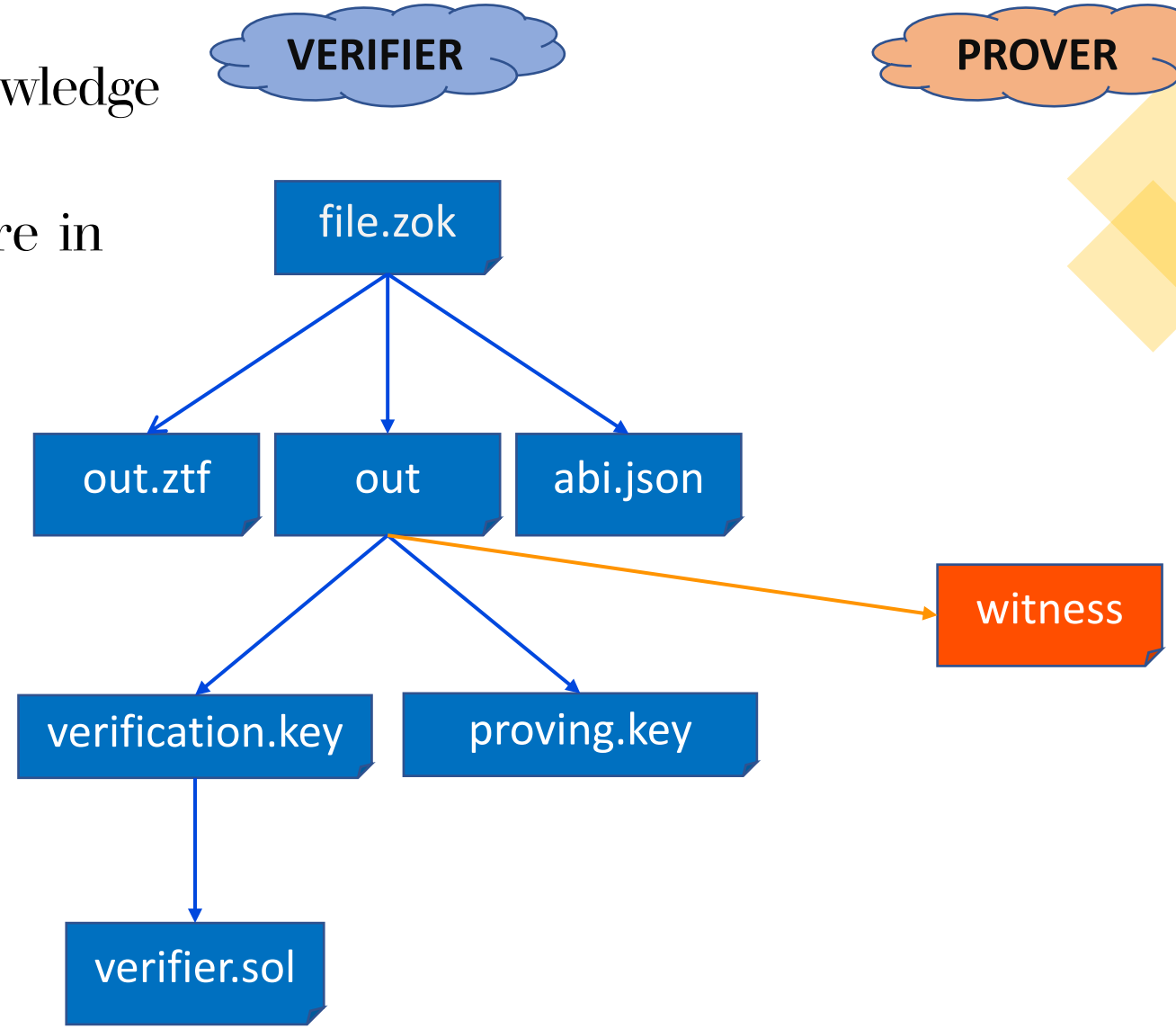
ZoKrates

Tool che implementa le Zero-Knowledge

Permette di creare un verificatore in linguaggio Solidity

Utilizza schemi **zk-SNARKs**

- Prove di **piccole dimensioni**
- Protocollo **non-interattivo**
- Creare prove errate è difficile
- Creare prove corrette senza conoscere il witness è impossibile



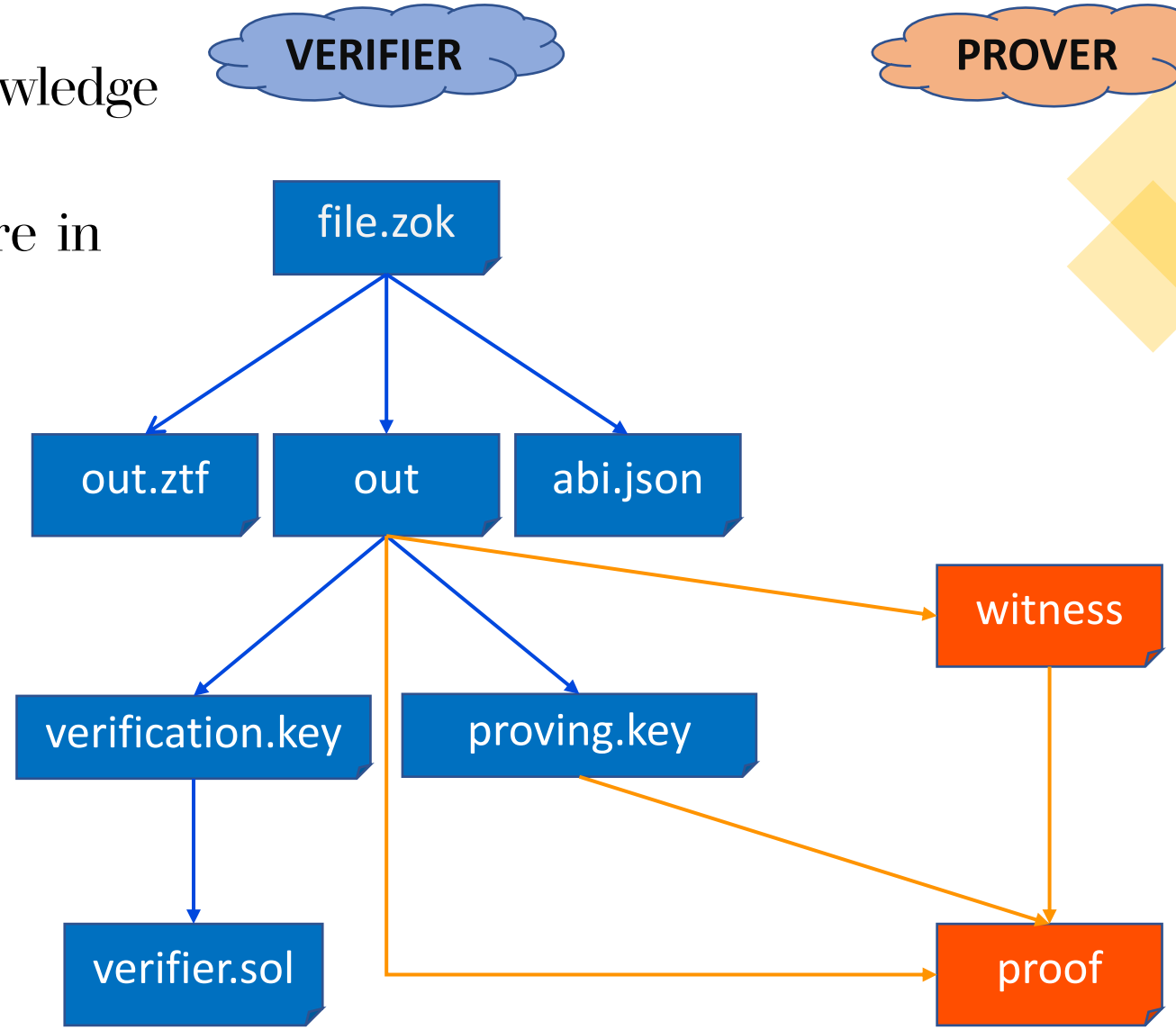
ZoKrates

Tool che implementa le Zero-Knowledge

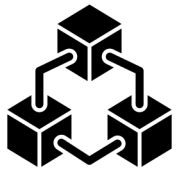
Permette di creare un verificatore in linguaggio Solidity

Utilizza schemi **zk-SNARKs**

- Prove di **piccole dimensioni**
- Protocollo **non-interattivo**
- Creare prove errate è difficile
- Creare prove corrette senza conoscere il witness è impossibile

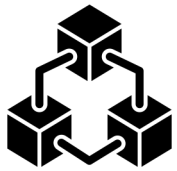


Design del sistema

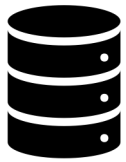


I VALORI DEGLI ATTRIBUTI PRIVATI NON POSSONO ESSERE MEMORIZZATI SU BLOCKCHAIN

Design del sistema

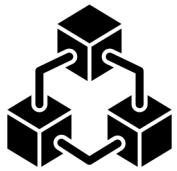


I VALORI DEGLI ATTRIBUTI PRIVATI NON POSSONO ESSERE MEMORIZZATI SU BLOCKCHAIN

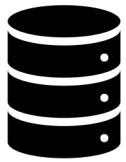


USARE UN NODO OFF-CHAIN PER MANTENERE GLI ATTRIBUTI PRIVATI, CHE POSSA ANCHE GENERARE LE PROOF ZOKRATES

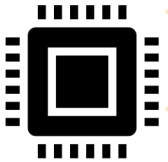
Design del sistema



I VALORI DEGLI ATTRIBUTI PRIVATI NON POSSONO ESSERE MEMORIZZATI SU BLOCKCHAIN

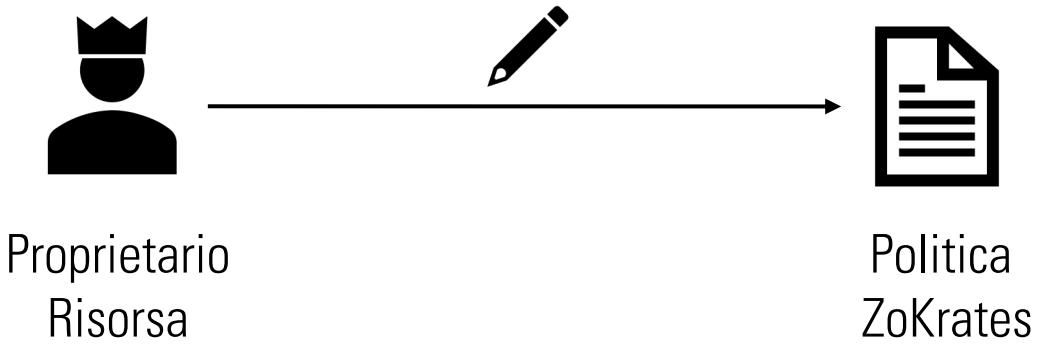


USARE UN NODO OFF-CHAIN PER MANTENERE GLI ATTRIBUTI PRIVATI, CHE POSSA ANCHE GENERARE LE PROOF ZOKRATES



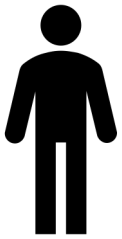
I GESTORI DEGLI ATTRIBUTI SONO LE COMPONENTI DEL SISTEMA CHE PIÙ SI PRESTANO A RICOPRIRE IL RUOLO DI NODI OFF-CHAIN

Protocollo ad alto livello per zk-ABAC: *SETUP*



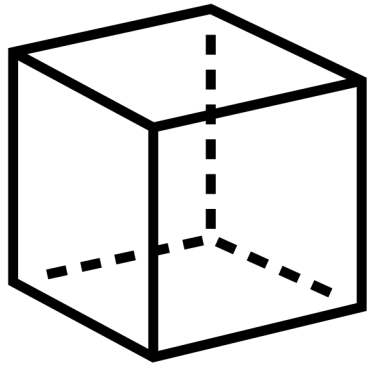
Protocollo ad alto livello per zk-ABAC: *SETUP*

Protocollo ad alto livello per zk-ABAC: *ACCESSO*

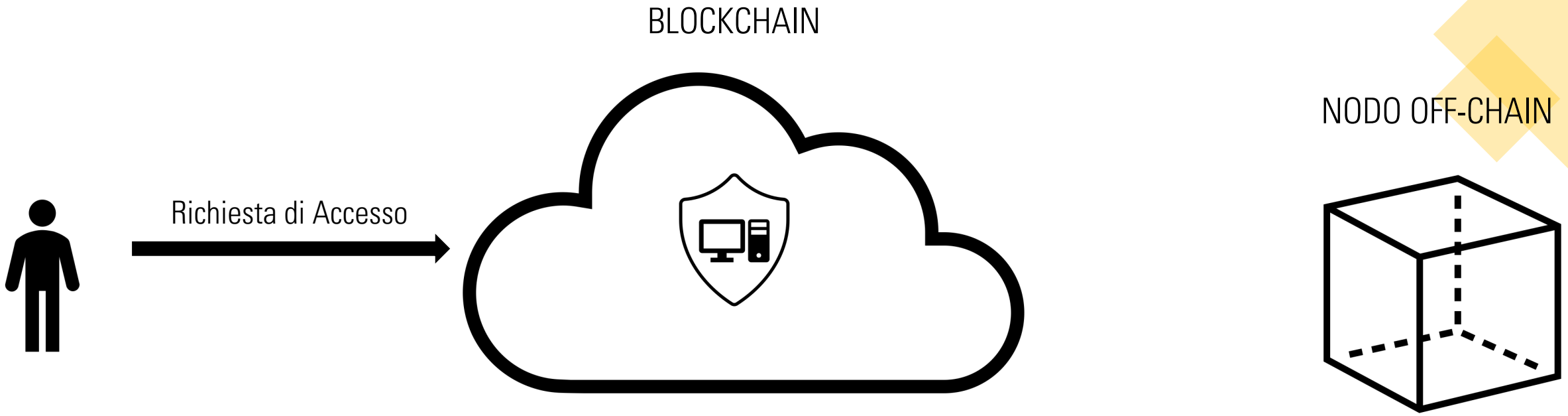


BLOCKCHAIN

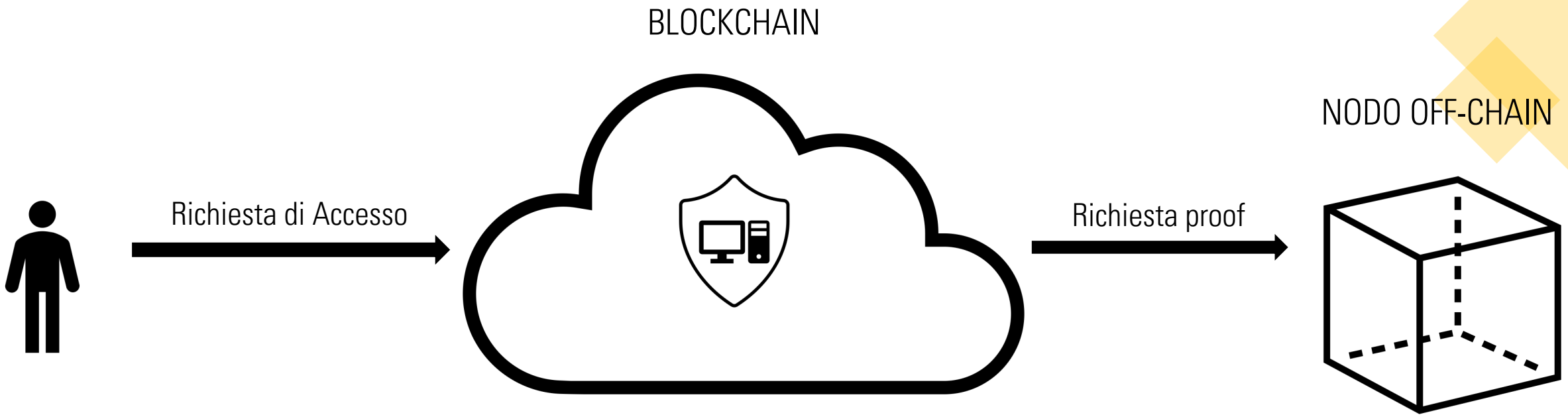
NODO OFF-CHAIN



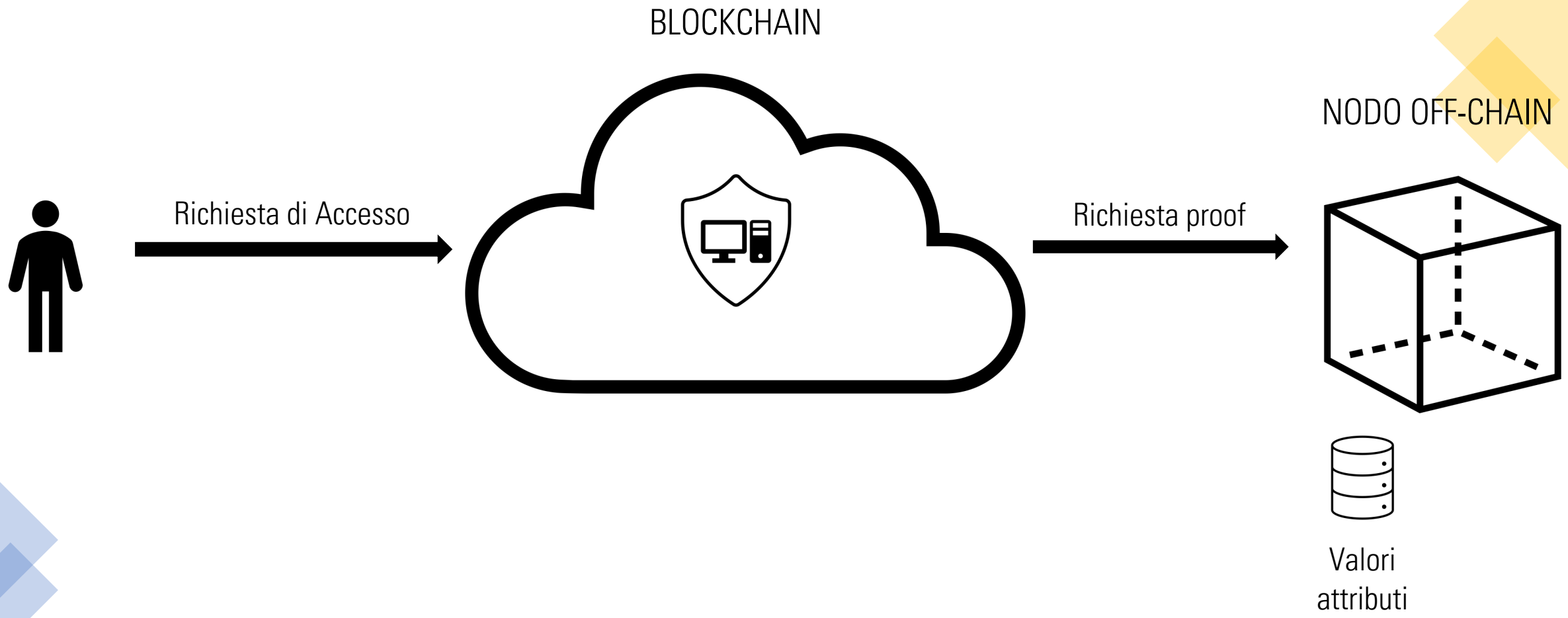
Protocollo ad alto livello per zk-ABAC: *ACCESSO*



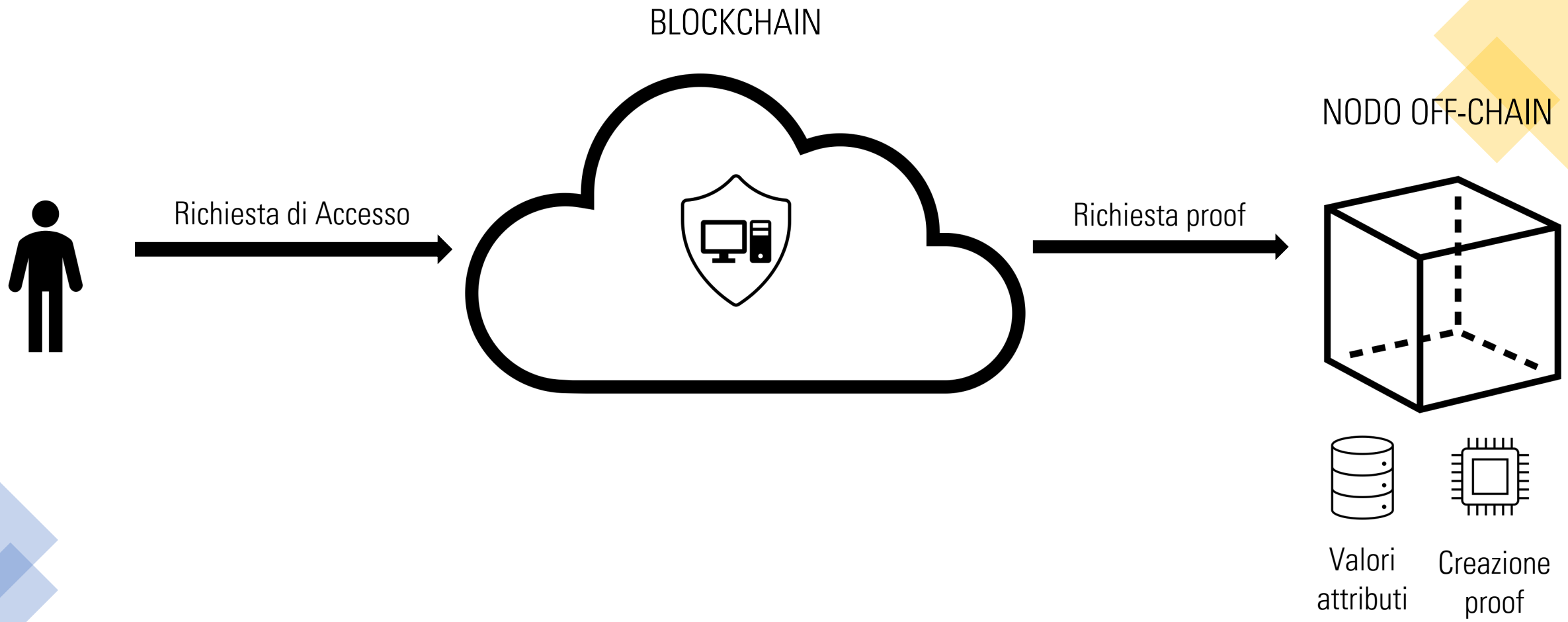
Protocollo ad alto livello per zk-ABAC: *ACCESSO*



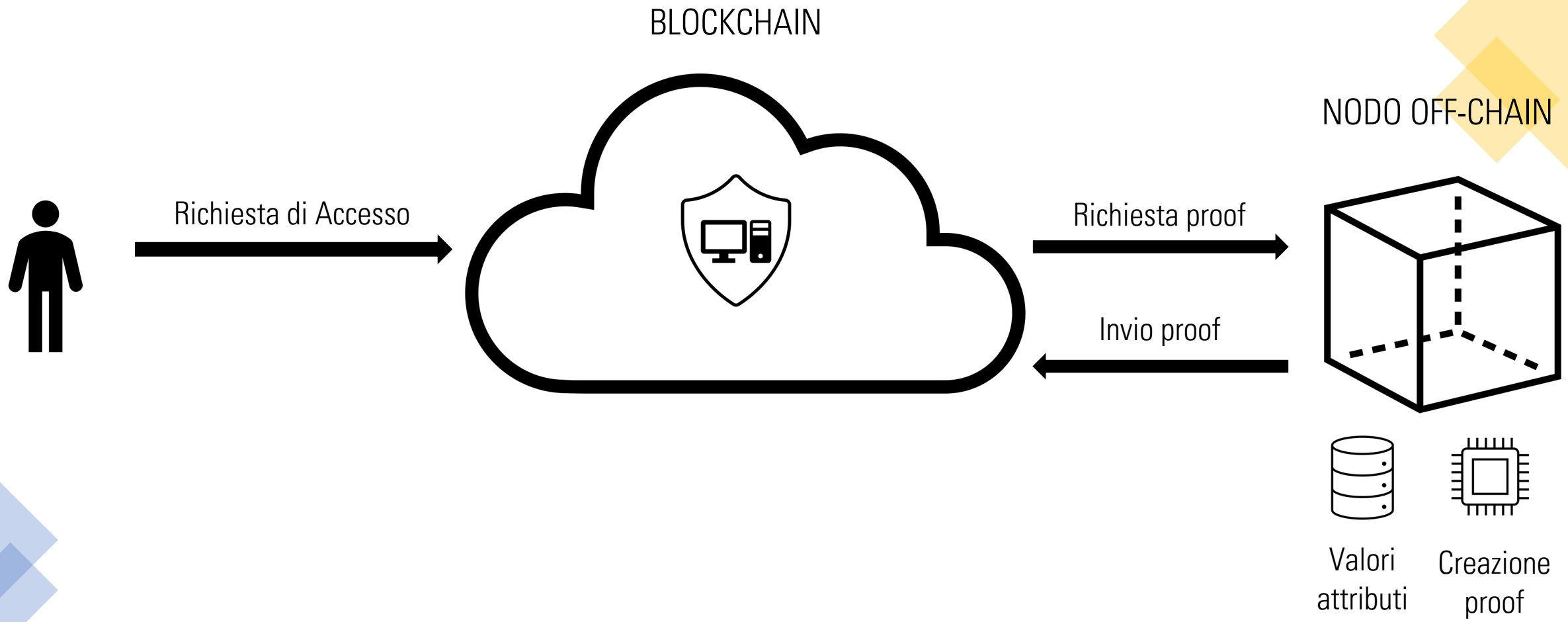
Protocollo ad alto livello per zk-ABAC: *ACCESSO*



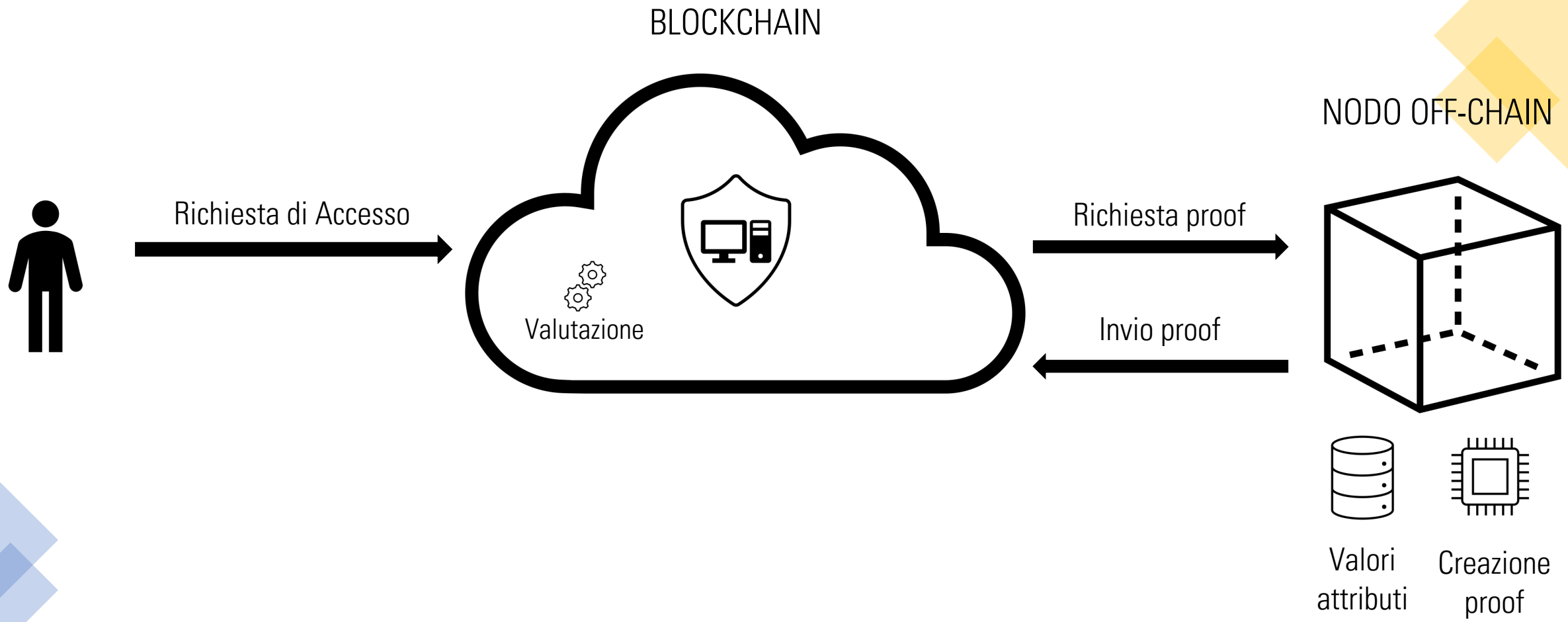
Protocollo ad alto livello per zk-ABAC: *ACCESSO*



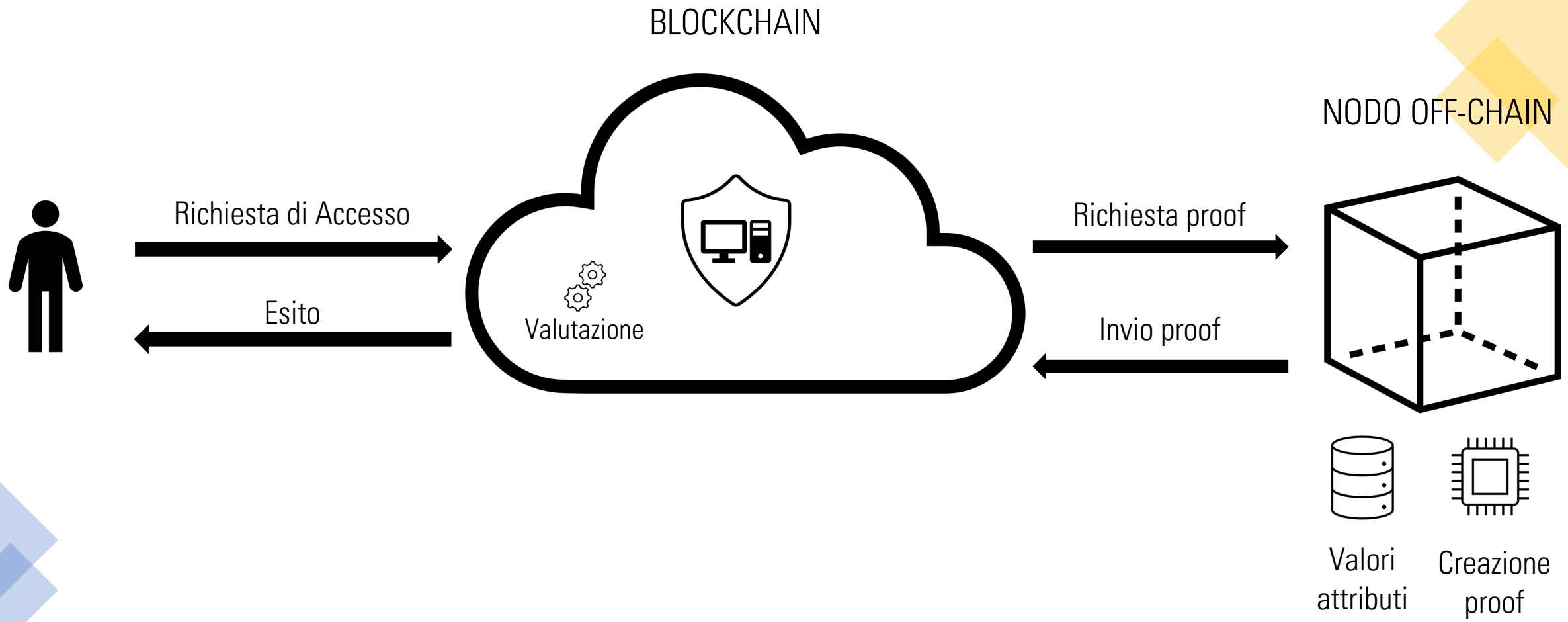
Protocollo ad alto livello per zk-ABAC: *ACCESSO*



Protocollo ad alto livello per zk-ABAC: *ACCESSO*

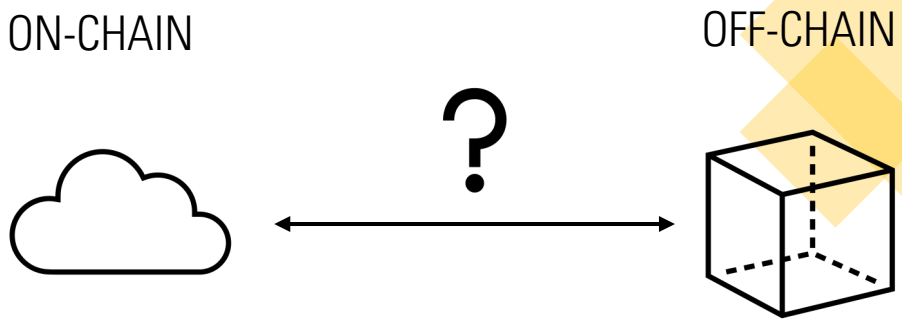


Protocollo ad alto livello per zk-ABAC: *ACCESSO*



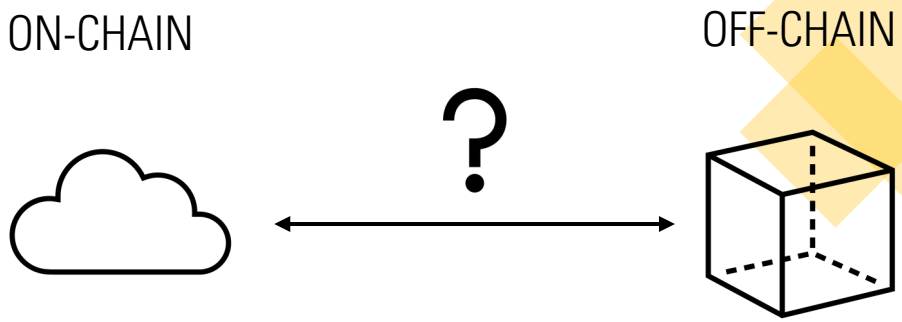
Integrazione ZoKrates – blockchain: sfide

- Memorizzazione file ZoKrates off-chain
- Creazione delle prove off-chain
- Interazione tra moduli off-chain – on-chain
- **Suddivisione del flusso di lavoro in più transazioni**



Integrazione ZoKrates – blockchain: sfide

- Memorizzazione file ZoKrates off-chain
- Creazione delle prove off-chain
- Interazione tra moduli off-chain – on-chain

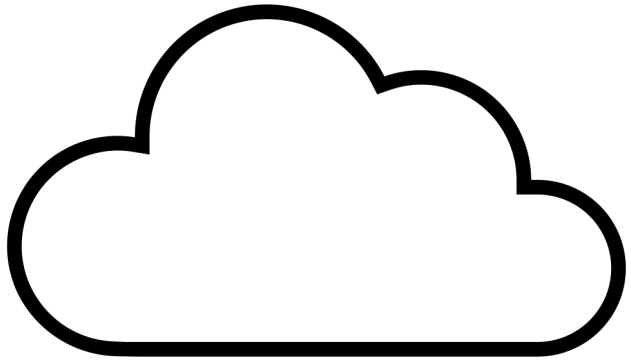


- **Suddivisione del flusso di lavoro in più transazioni**

Concorrenza delle transazioni

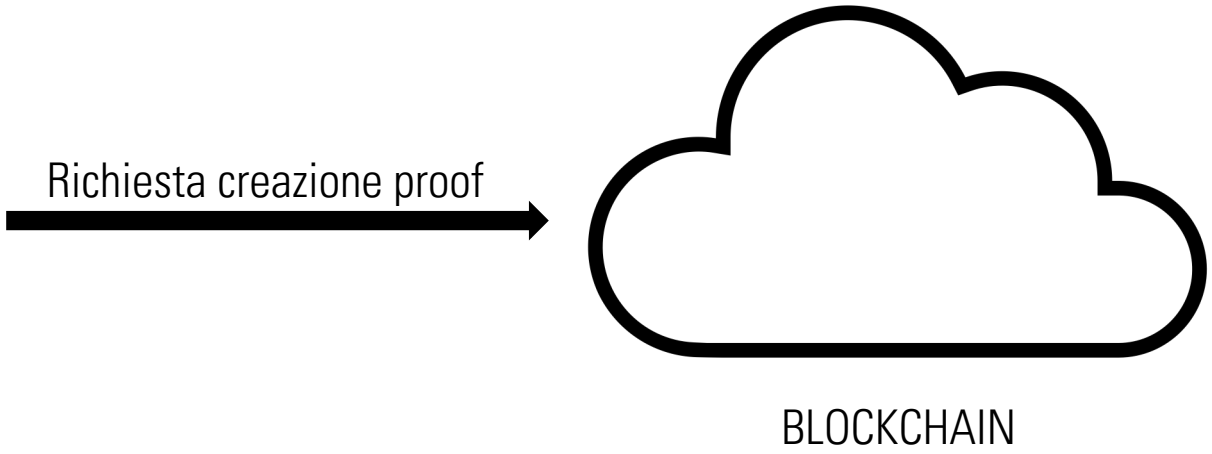
Associazione richiesta-proof

Funzionamento nodo off-chain

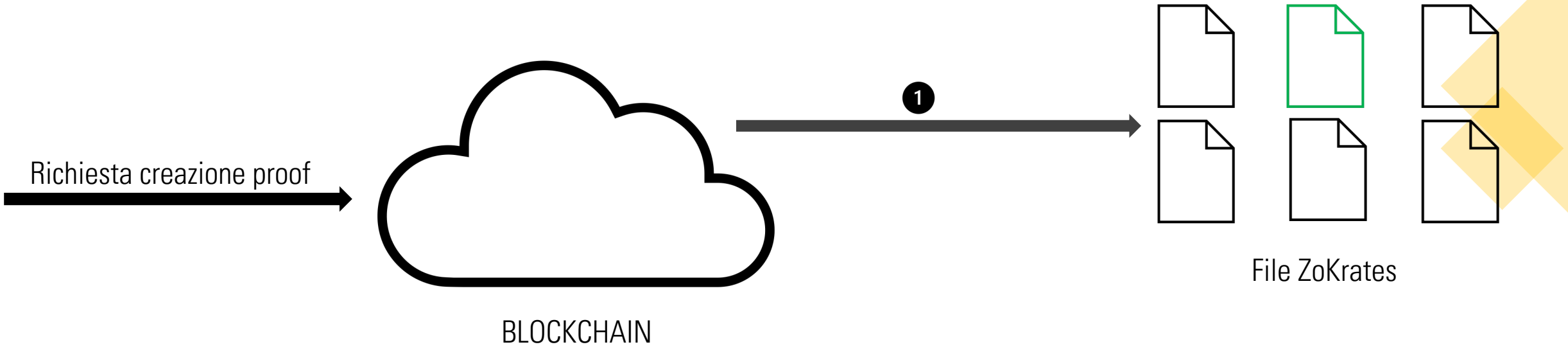


BLOCKCHAIN

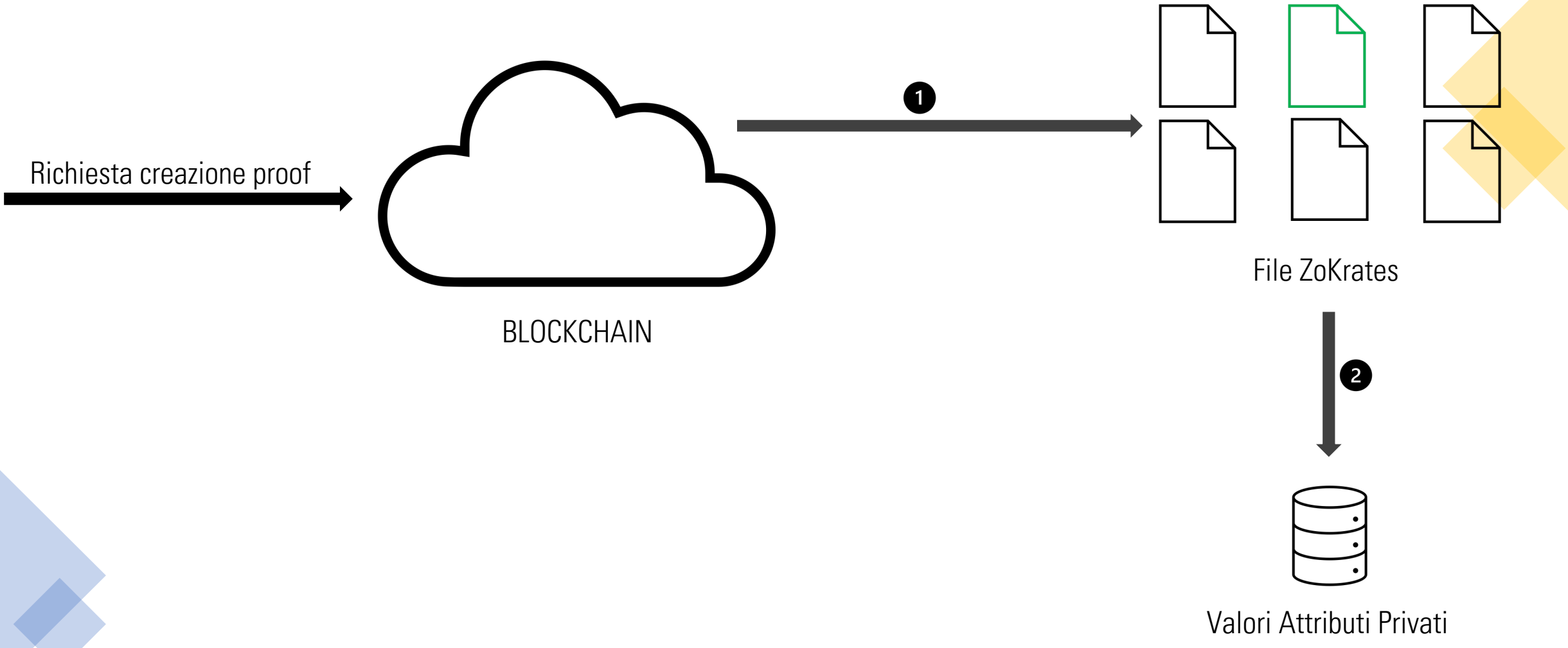
Funzionamento nodo off-chain



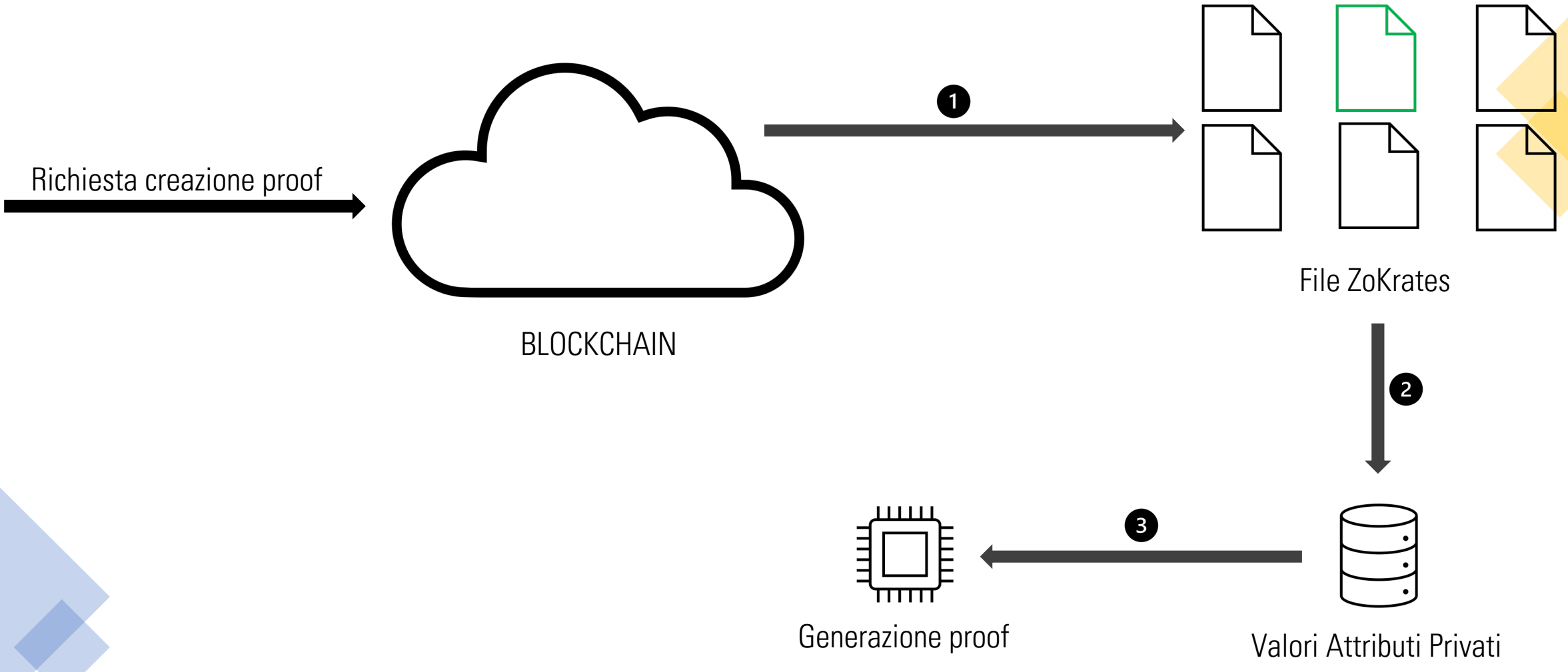
Funzionamento nodo off-chain



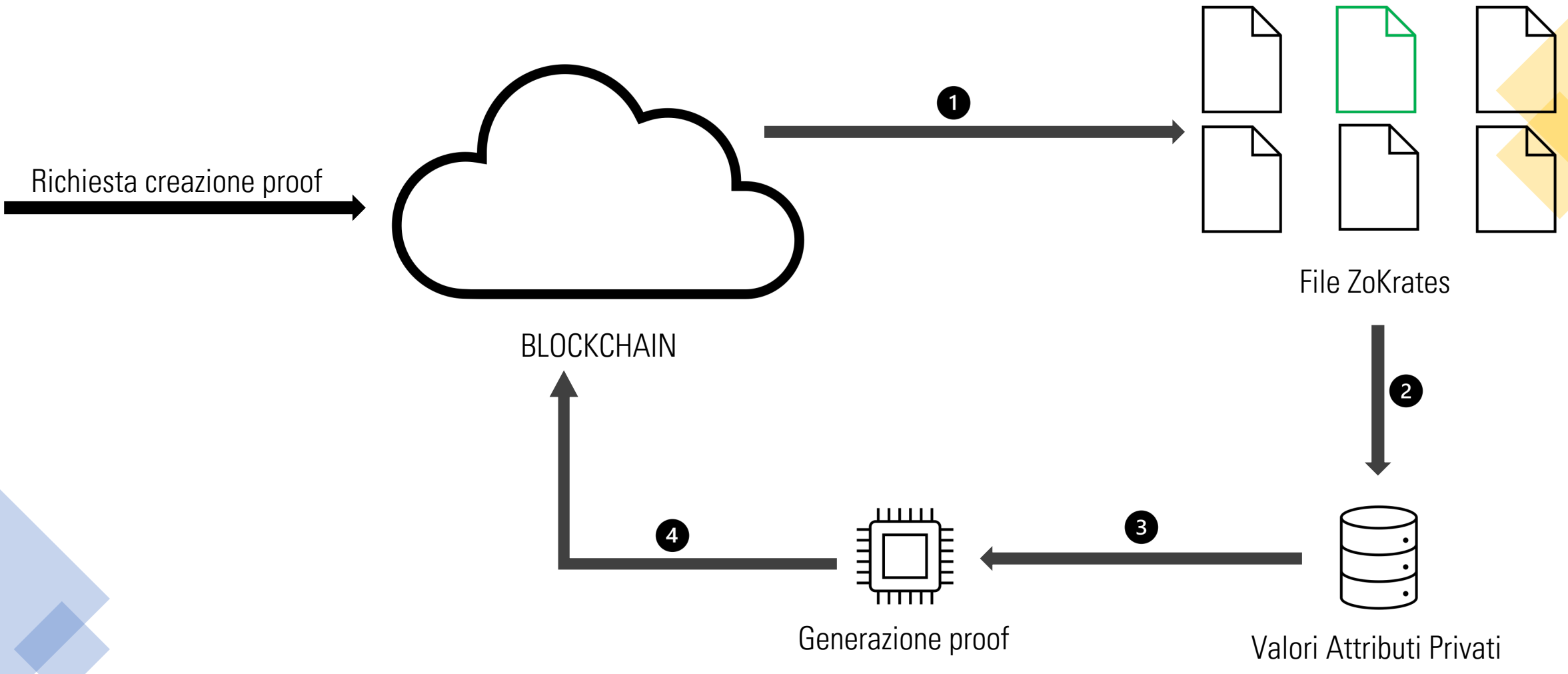
Funzionamento nodo off-chain



Funzionamento nodo off-chain



Funzionamento nodo off-chain



Implementazione di zk-ABAC

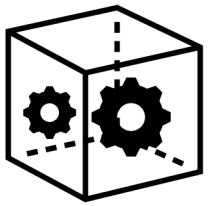
Transizione da modello a implementazione

Blockchain

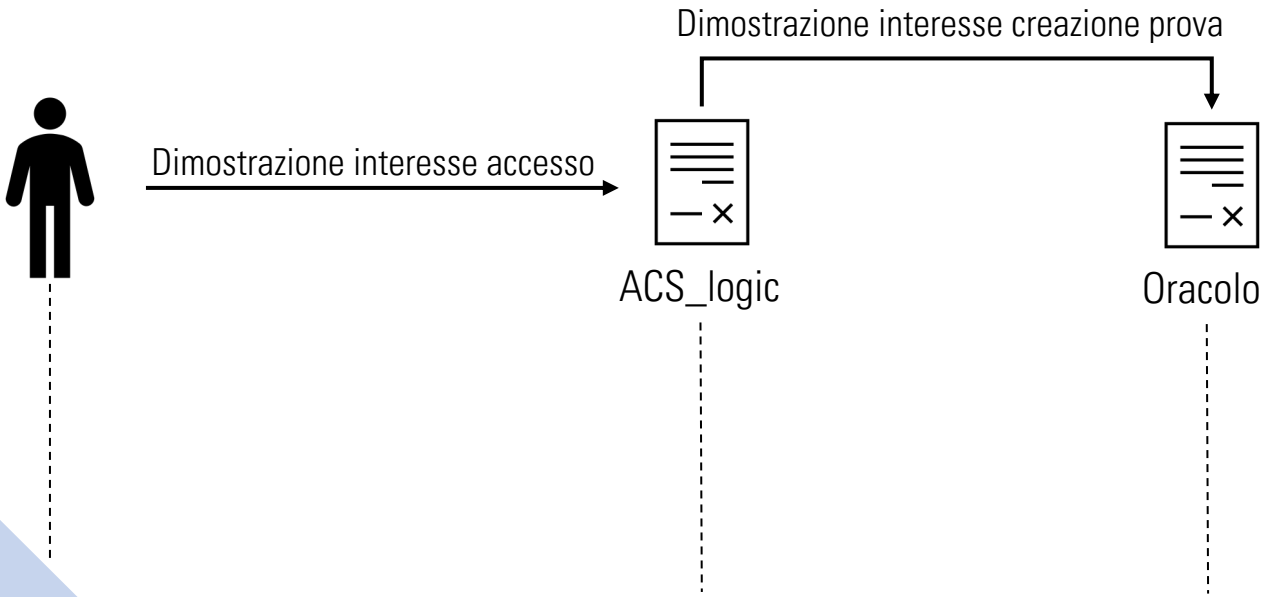
Logica del sistema di controllo → ACS_logic

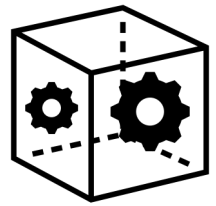
Interfaccia on-chain del nodo off-chain → Oracolo

Nodo off-chain → Macchina con software specifico

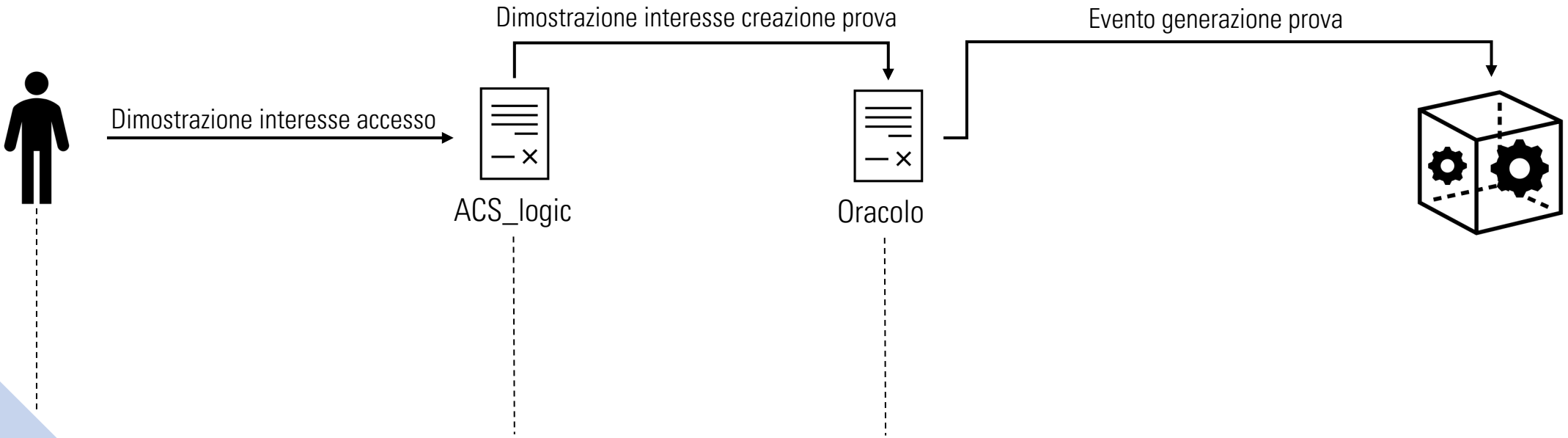


Implementazione su Ethereum

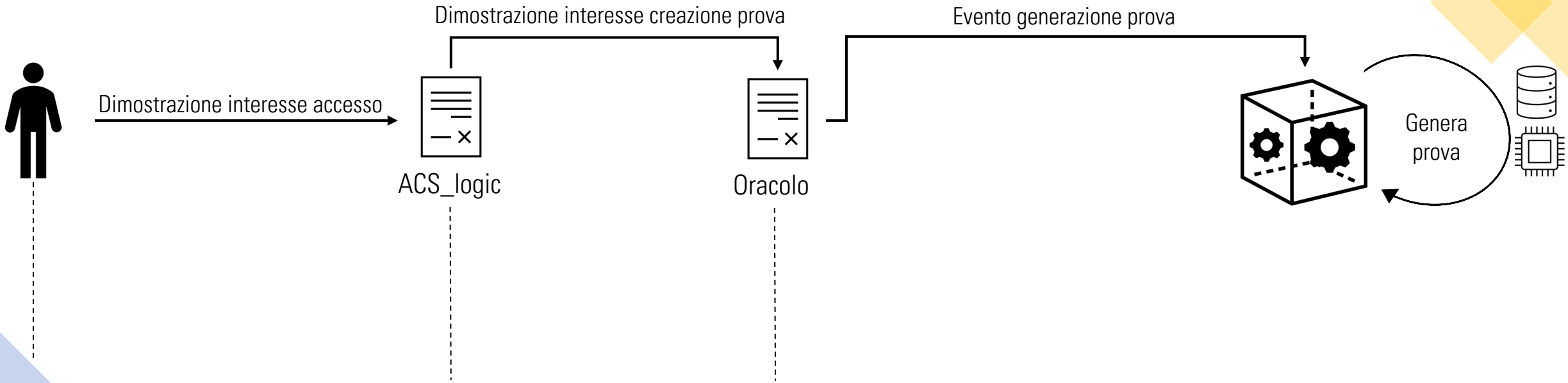




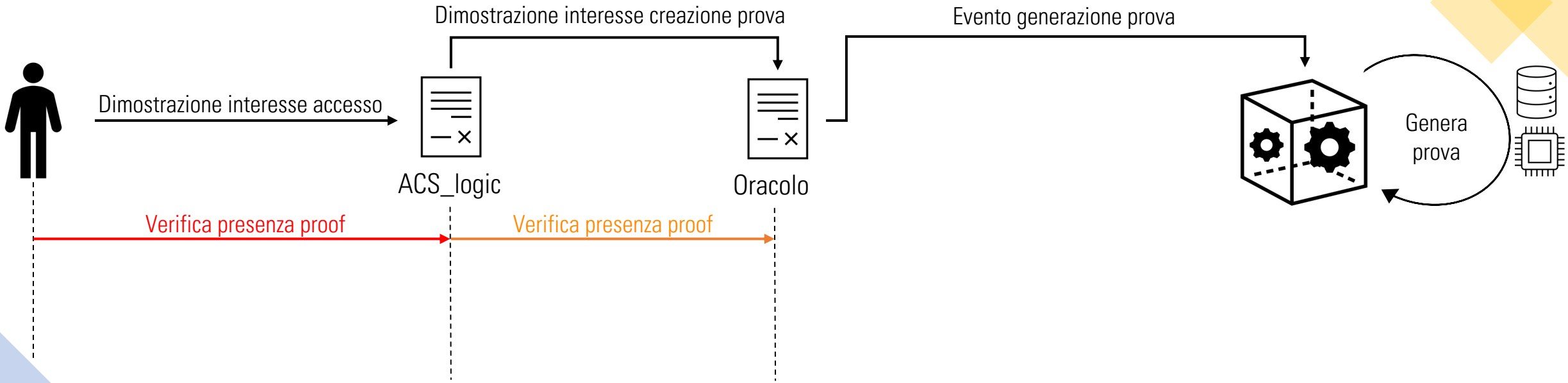
Implementazione su Ethereum



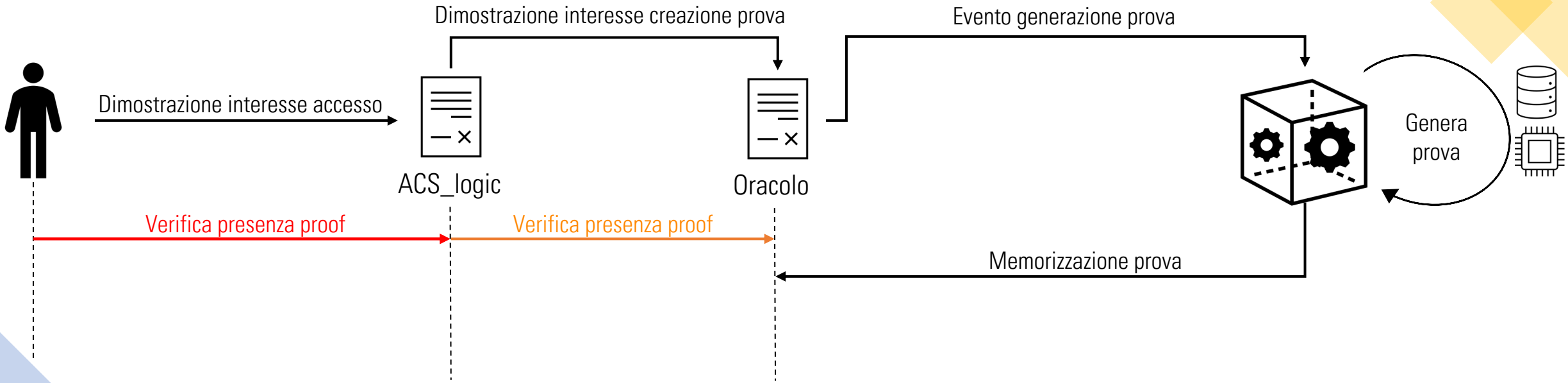
Implementazione su Ethereum



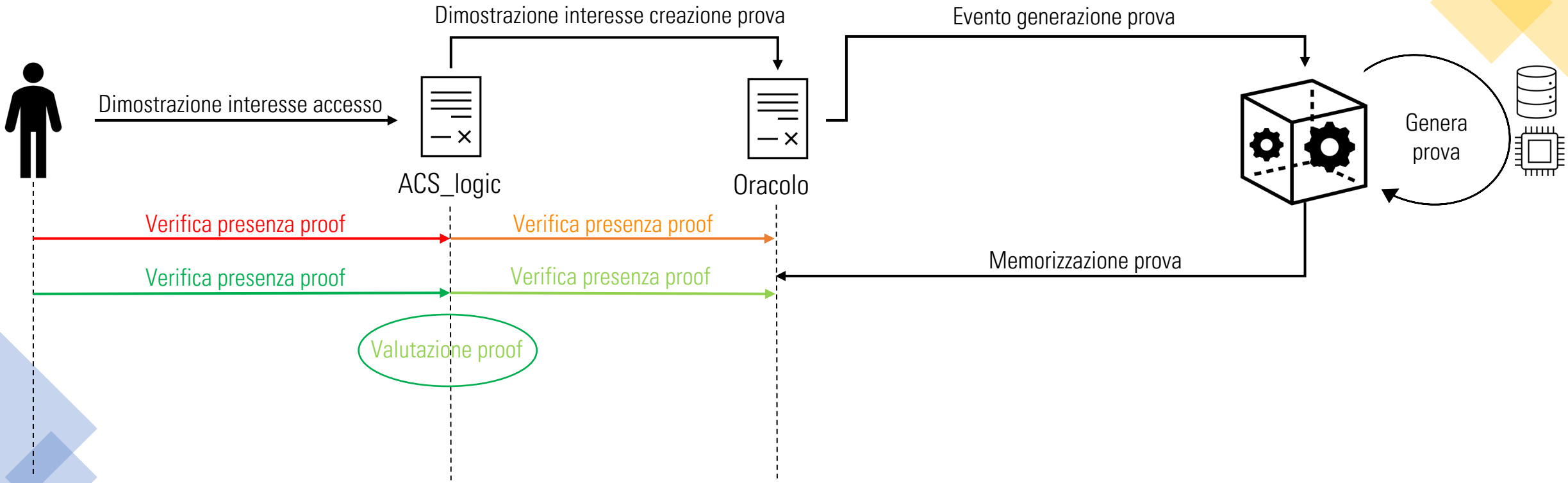
Implementazione su Ethereum



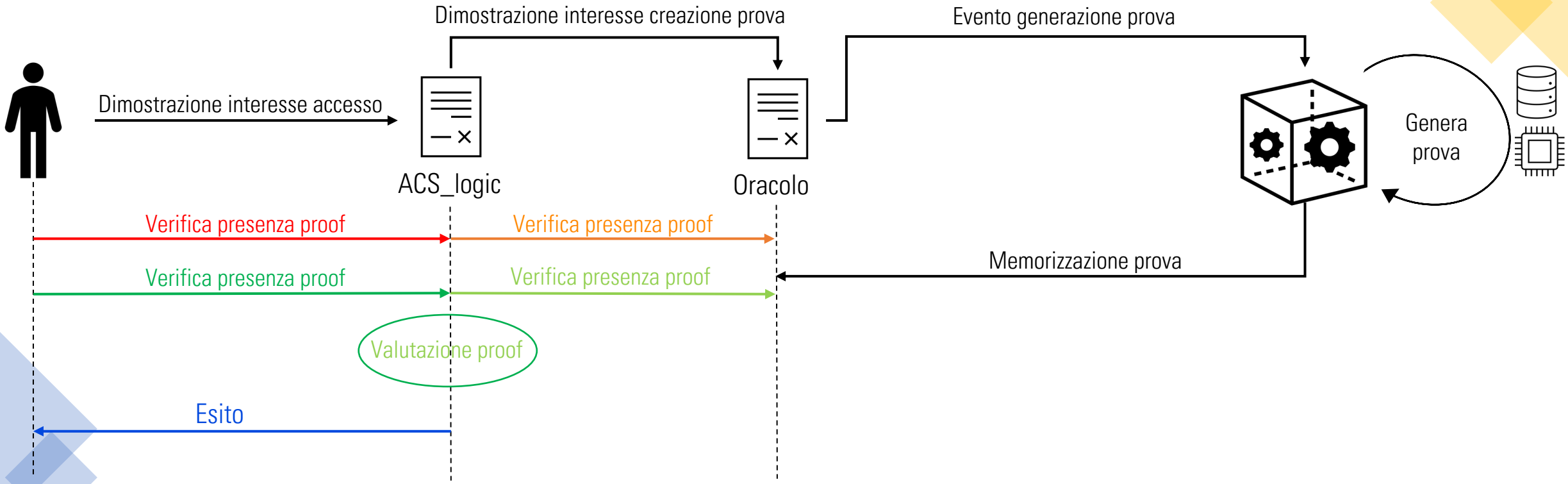
Implementazione su Ethereum



Implementazione su Ethereum

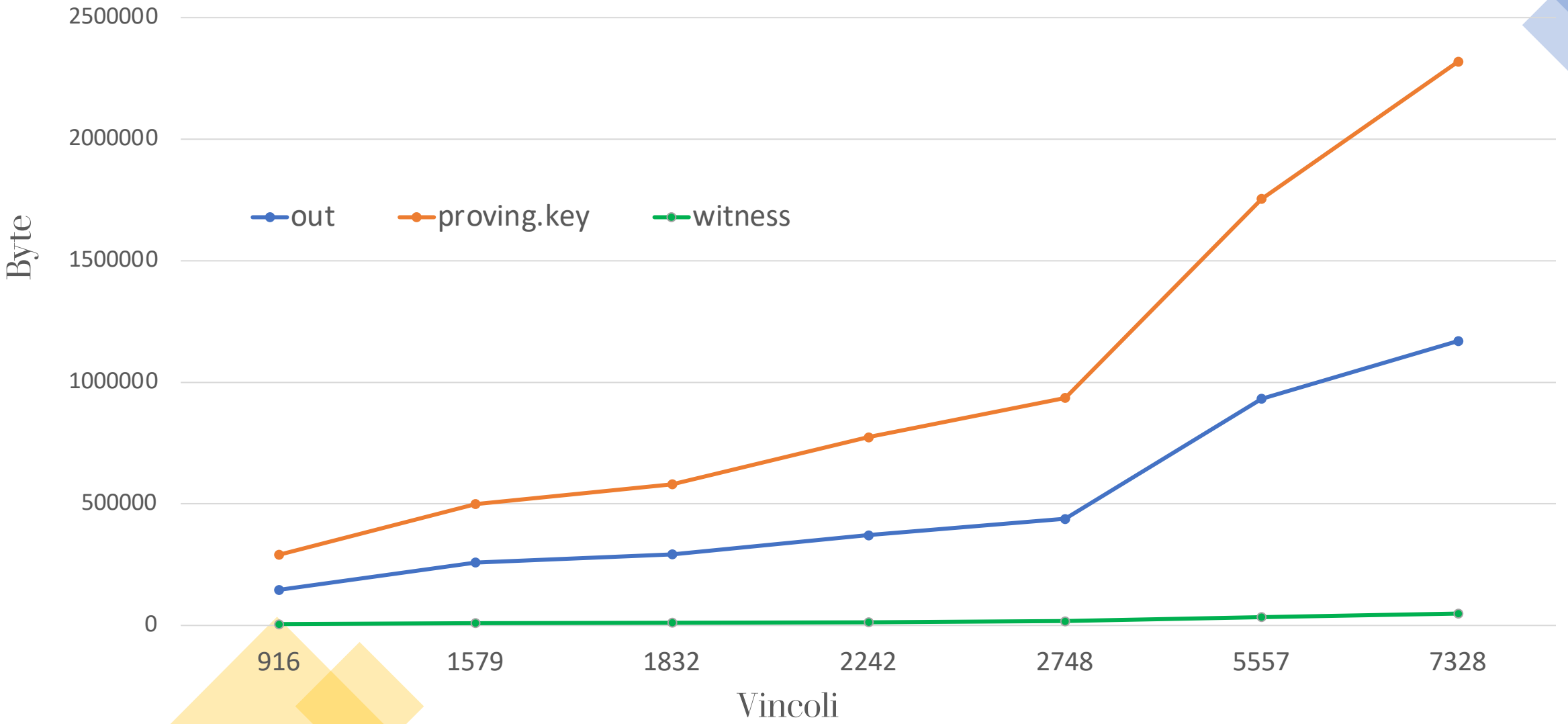


Implementazione su Ethereum

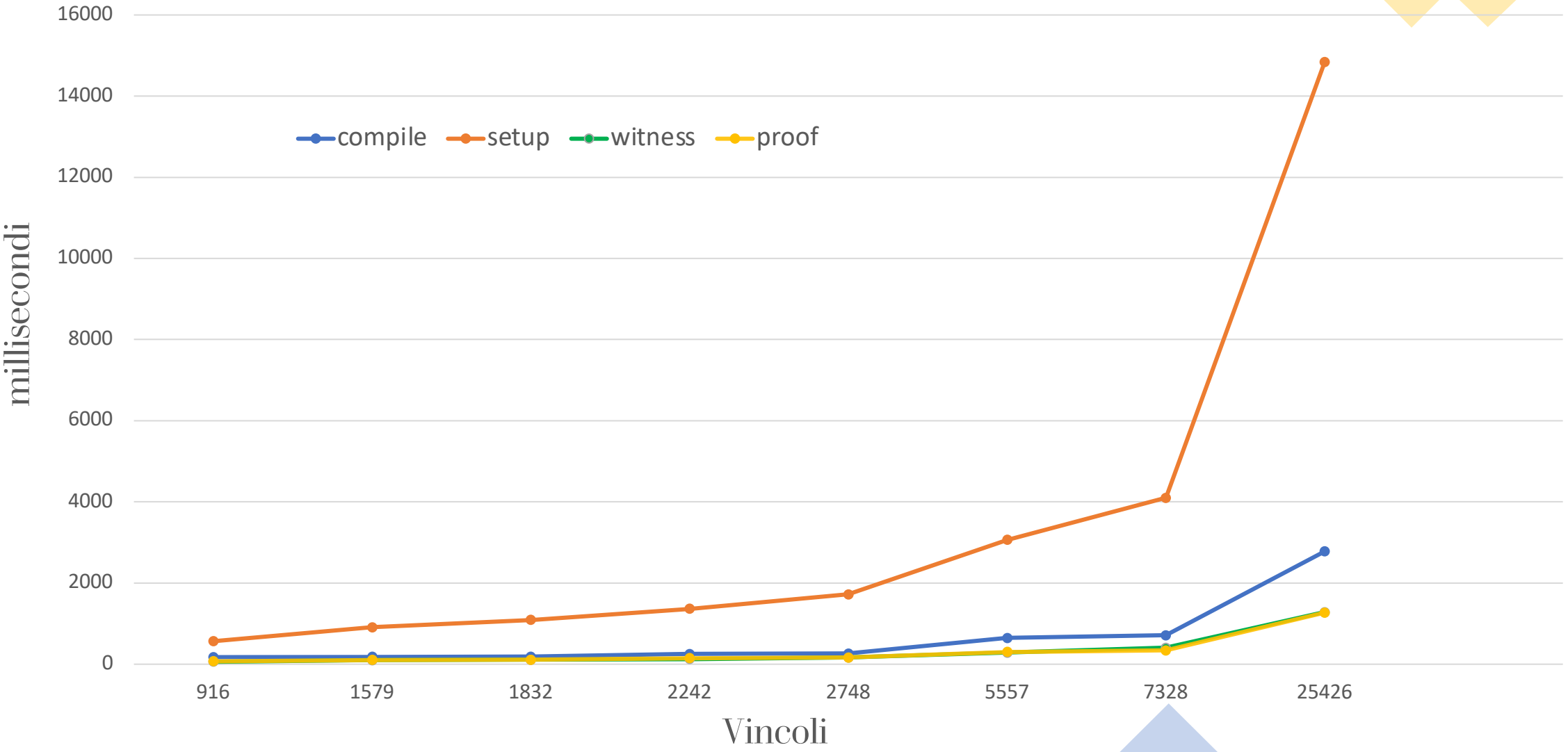


Valutazioni di ZoKrates e di zk-ABAC

Variazioni dimensioni file ZoKrates



Variazioni tempi operazioni ZoKrates



Valutazioni del sistema su Ethereum

Le transazioni su Ethereum hanno un costo in termini di **gas**

Le transazioni vengono raggruppate in **blocchi**

Il **Gas Limit** è il gas massimo disponibile all'interno di un certo blocco.

GAS LIMIT ATTUALE: 12.400.000

	Dimostrazioni Interesse di Accesso	Memorizzazione proof	Recupero proof e decisione
Per blocco	91	54	43
Al secondo	7	4	4

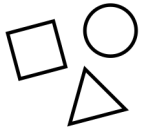
Conclusioni e considerazioni

- PoC di zk-ABAC mostra come sia possibile utilizzare attributi privati nei sistemi ABAC su blockchain
- Il tool ZoKrates non comporta problemi di integrazione e ha una buona scalabilità
- Ethereum è la scelta migliore per l'auditability e la trasparenza, ma non per le prestazioni e la scalabilità

Lavori futuri

Parser per la traduzione delle politiche XACML in ZoKrates

Scalabilità: utilizzo di blockchain permissioned per zk-ABAC



Interoperabilità: utilizzo di oracoli con politiche modulari

GRAZIE DELL' ATTENZIONE