Chap3#3: Privacy Preservation in Machine Learning #3

February 27, 2023

भारतीय प्रौद्योगिकी
संस्थान जम्मू
INDIAN INSTITUTE OF
TECHNOLOGY JAMMU

Devesh C Jinwala,
Professor, SVNIT and Adjunct Prof., CSE, IIT Jammu

Department of Computer Science and Engineering,
Sardar Vallabhhai National Institute of Technology, SURAT

Chap 2: ML Applications in Security: Topics to study

- Privacy Preservation, What is Privacy? Data Privacy. Machine Learning in Privacy Preservation: Four Main stakes to Privacy preservation in ML. Two principle approaches: (a) Augmenting the ML techniques with the conventional approaches in the domain of privacy preservation to achieve privacy viz. Homomorphic Encryption(HE Algorithms and the associated mathematics), Secret Multi-party Computations, Zero Knowledge Proofs, Perturbation techniques (e.g. differential privacy), Anonymization techniques (e.g.)k-Anonymity, l-Diversity) (b) ML-specific approaches like Federated Learning OR Ensemble Learning. Ethical issues and Law for data / process privacy : GDPR, Alexa, other relevant applications [6 hours]

Reviewing the theme of ML Paradigms for Privacy Preservation

Four Main stakes to Privacy preservation

There are four main stakes to privacy preservation in general:

- Privacy of the input data, input queries , web search queries
- Privacy of the computations
- Privacy of the output data, web search query results
- Data Privacy General Regulations, Data protection strategies, processes and principles

Four Main stakes to Privacy preservation

There are four main stakes to privacy preservation in general:

- Privacy of the input data, input queries , web search queries
- Privacy of the computations
- Privacy of the output data, web search query results
- Data Privacy General Regulations, Data protection strategies, processes and principles

We examine one of these viz. Privacy of Computations in greater detail shortly hereafter seeing main stakes to Privacy preservation in ML

Four Main stakes to Privacy preservation in ML

There are four main stakes to privacy preservation in general:

- Privacy of the input data
  - the assurance that other parties, including the model developer, will not be able to see a user's input data
- Privacy of the output data
  - the assurance that the output of a model is only accessible to the client whose data is being inferred upon.
- Privacy of the model
  - rhe assurance that a hostile party will not be able to steal the model
- Data privacy in training
  - the assurance that a malicious party will not reverse-engineer the training data - although gathering information about training data and model is more difficult than that for the data.

Privacy Preserving Machine Learning: How to achieve?

The goal of privacy-preserving machine learning is

- to bridge the gap between privacy while receiving the benefits of machine learning.

Privacy Preserving Machine Learning: How to achieve?

The goal of privacy-preserving machine learning is

- to bridge the gap between privacy while receiving the benefits of machine learning.
- is a critical facilitator for the protection of acquired data and adhering to data privacy laws.

Privacy Preserving Machine Learning: How to achieve?

The goal of privacy-preserving machine learning is

- to bridge the gap between privacy while receiving the benefits of machine learning.
- is a critical facilitator for the protection of acquired data and adhering to data privacy laws.

Privacy Preserving Machine Learning: How to achieve?

The goal of privacy-preserving machine learning is

- to bridge the gap between privacy while receiving the benefits of machine learning.
- is a critical facilitator for the protection of acquired data and adhering to data privacy laws.

Privacy-preservation in ML

- is achieved by augmenting conventional ML with different strategies that protect data privacy, that include....

Privacy Preserving Machine Learning: How to achieve?

The goal of privacy-preserving machine learning is

- to bridge the gap between privacy while receiving the benefits of machine learning.
- is a critical facilitator for the protection of acquired data and adhering to data privacy laws.

Privacy-preservation in ML

- is achieved by augmenting conventional ML with different strategies that protect data privacy, that include....
  - cryptographic approaches like

Privacy Preserving Machine Learning: How to achieve?

The goal of privacy-preserving machine learning is

- to bridge the gap between privacy while receiving the benefits of machine learning.
- is a critical facilitator for the protection of acquired data and adhering to data privacy laws.

Privacy-preservation in ML

- is achieved by augmenting conventional ML with different strategies that protect data privacy, that include....
  - cryptographic approaches like
    - homomorphic encryption

Privacy Preserving Machine Learning: How to achieve?

The goal of privacy-preserving machine learning is

- to bridge the gap between privacy while receiving the benefits of machine learning.
- is a critical facilitator for the protection of acquired data and adhering to data privacy laws.

Privacy-preservation in ML

- is achieved by augmenting conventional ML with different strategies that protect data privacy, that include....
  - cryptographic approaches like
    - homomorphic encryption
    - secure multi-party computing,

Privacy Preserving Machine Learning: How to achieve?

The goal of privacy-preserving machine learning is

- to bridge the gap between privacy while receiving the benefits of machine learning.
- is a critical facilitator for the protection of acquired data and adhering to data privacy laws.

Privacy-preservation in ML

- is achieved by augmenting conventional ML with different strategies that protect data privacy, that include....
  - cryptographic approaches like
    - homomorphic encryption
    - secure multi-party computing,
    - Zero knowledge proofs

Privacy Preserving Machine Learning: How to achieve?

The goal of privacy-preserving machine learning is

- to bridge the gap between privacy while receiving the benefits of machine learning.
- is a critical facilitator for the protection of acquired data and adhering to data privacy laws.

Privacy-preservation in ML

- is achieved by augmenting conventional ML with different strategies that protect data privacy, that include....
  - cryptographic approaches like
    - homomorphic encryption
    - secure multi-party computing,
    - Zero knowledge proofs
  - perturbation techniques like differential privacy

Privacy Preserving Machine Learning: How to achieve?

The goal of privacy-preserving machine learning is

- to bridge the gap between privacy while receiving the benefits of machine learning.
- is a critical facilitator for the protection of acquired data and adhering to data privacy laws.

Privacy-preservation in ML

- is achieved by augmenting conventional ML with different strategies that protect data privacy, that include....
  - cryptographic approaches like
    - homomorphic encryption
    - secure multi-party computing,
    - Zero knowledge proofs
  - perturbation techniques like differential privacy
  - anonymization techniques like k-Anonymity and l-Diversity

Privacy Preserving Machine Learning: How to achieve?

The goal of privacy-preserving machine learning is

- to bridge the gap between privacy while receiving the benefits of machine learning.
- is a critical facilitator for the protection of acquired data and adhering to data privacy laws.

Privacy-preservation in ML

- is achieved by augmenting conventional ML with different strategies that protect data privacy, that include....
  - cryptographic approaches like
    - homomorphic encryption
    - secure multi-party computing,
    - Zero knowledge proofs
  - perturbation techniques like differential privacy
  - anonymization techniques like k-Anonymity and l-Diversity
  - ML-specific approaches like Federated Learning OR Ensemble Learning - the Privacy-Preserving Techniques - modifying the conventional ML training methods to keep user data private.

Augmenting ML for Privacy Preservation: Zero Knowledge Proofs

Zero Knowledge Proofs

Zero Knowledge Proof

- is a way of proving the validity of a statement without revealing the statement itself.

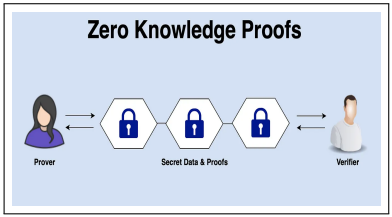


Figure: Eliminate the need to reveal information shared

Src: https://towardsdatascience.com/what-are-zero-knowledge-proofs-7ef6aab955fc

Zero Knowledge Proofs

Zero Knowledge Proof

- is a way of proving the validity of a statement without revealing the statement itself.
- based on a protocol between two parties viz. a prover and a verifier.

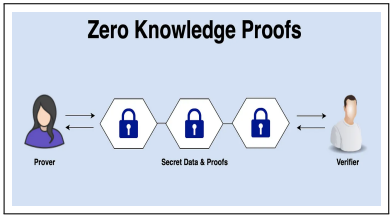


Figure: Eliminate the need to reveal information shared

Src: https://towardsdatascience.com/what-are-zero-knowledge-proofs-7ef6aab955fc

Zero Knowledge Proofs

Zero Knowledge Proof

- is a way of proving the validity of a statement without revealing the statement itself.
- based on a protocol between two parties viz. a prover and a verifier.
  - the *prover* is the party trying to prove a claim,

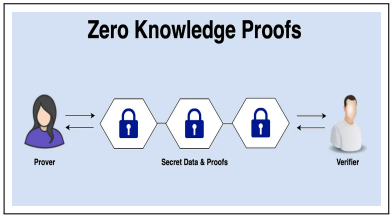


Figure: Eliminate the need to reveal information shared

Src: https://towardsdatascience.com/what-are-zero-knowledge-proofs-7ef6aab955fc

Zero Knowledge Proofs

Zero Knowledge Proof

- is a way of proving the validity of a statement without revealing the statement itself.
- based on a protocol between two parties viz. a prover and a verifier.
  - the *prover* is the party trying to prove a claim,
  - while the *verifier* is responsible for validating the claim, i.e. to prove the knowledge of some private information.

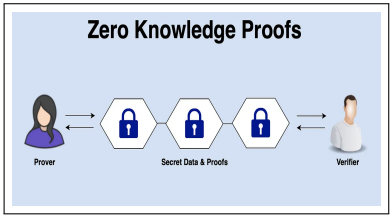


Figure: Eliminate the need to reveal information shared

Src: https://towardsdatascience.com/what-are-zero-knowledge-proofs-7ef6aab955fc

Zero Knowledge Proofs

Zero Knowledge Proof

- is a way of proving the validity of a statement without revealing the statement itself.
- based on a protocol between two parties viz. a prover and a verifier.
  - the *prover* is the party trying to prove a claim,
  - while the *verifier* is responsible for validating the claim, i.e. to prove the knowledge of some private information.
- For instance, consider the case where a user (i.e. a prover) has to prove his identity to a host (a verifier)

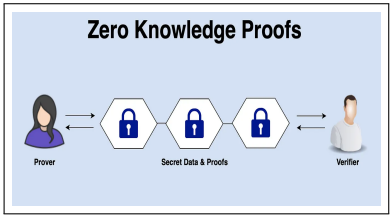


Figure: Eliminate the need to reveal information shared

Src: https://towardsdatascience.com/what-are-zero-knowledge-proofs-7ef6aab955fc

Zero Knowledge Proofs

Zero Knowledge Proof

- is a way of proving the validity of a statement without revealing the statement itself.
- based on a protocol between two parties viz. a prover and a verifier.
  - the *prover* is the party trying to prove a claim,
  - while the *verifier* is responsible for validating the claim, i.e. to prove the knowledge of some private information.
- For instance, consider the case where a user (i.e. a prover) has to prove his identity to a host (a verifier)
  - by logging in with his/her account and private password, but.....

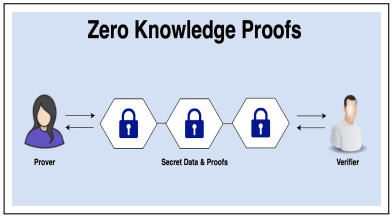


Figure: Eliminate the need to reveal information shared

Src: https://towardsdatascience.com/what-are-zero-knowledge-proofs-7ef6aab955fc

Zero Knowledge Proofs

Zero Knowledge Proof

- is a way of proving the validity of a statement without revealing the statement itself.
- based on a protocol between two parties viz. a prover and a verifier.
  - the *prover* is the party trying to prove a claim,
  - while the *verifier* is responsible for validating the claim, i.e. to prove the knowledge of some private information.
- For instance, consider the case where a user (i.e. a prover) has to prove his identity to a host (a verifier)
  - by logging in with his/her account and private password, but.....
  - but, very vitally without revealing the password to the verifier.

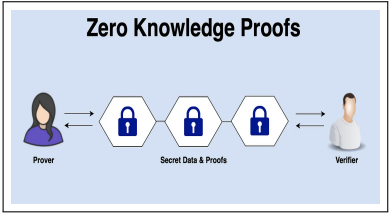


Figure: Eliminate the need to reveal information shared

Src: https://towardsdatascience.com/what-are-zero-knowledge-proofs-7ef6aab955fc

ZKP Research Evolution Timeline

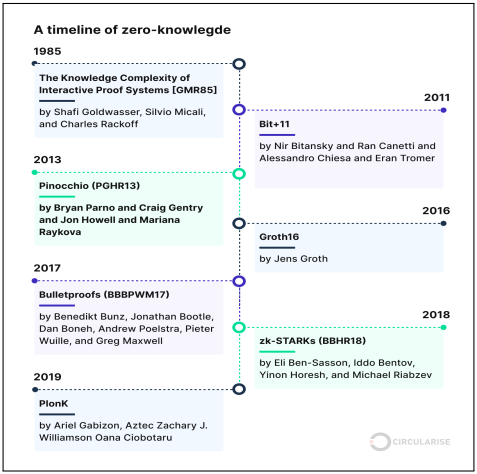
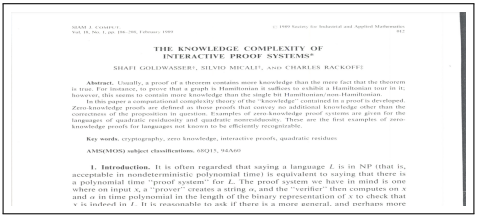


Figure: A timeline of the development of zero-knowledge proofs

Figure Src: https://www.circularise.com/blogs/zero-knowledge-proofs-explained-in-3-examples

ZKPs: First Definition



SIAM J. COMPUT.
Vol. 18, No. 1, pp. 186–208, February 1989

© 1989 Society for Industrial and Applied Mathematics
012

THE KNOWLEDGE COMPLEXITY OF INTERACTIVE PROOF SYSTEMS*

SHAFI GOLDWASSER†, SILVIO MICALI‡, AND CHARLES RACKOFF‡

Abstract. Usually, a proof of a theorem contains more knowledge than the mere fact that the theorem is true. For instance, to prove that a graph is Hamiltonian it suffices to exhibit a Hamiltonian tour in it; however, this seems to contain more knowledge than the single bit Hamiltonian/non-Hamiltonian.

In this paper a computational complexity theory of the "knowledge" contained in a proof is developed. Zero-knowledge proofs are defined as those proofs that convey no additional knowledge other than the correctness of the proposition in question. Examples of zero-knowledge proof systems are given for the languages of quadratic residuosity and quadratic nonresiduosity. These are the first examples of zero-knowledge proofs for languages not known to be efficiently recognizable.

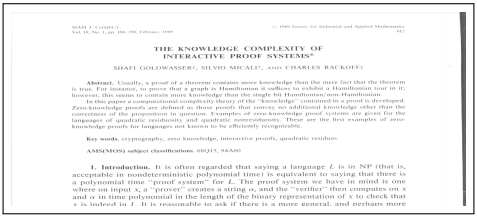
Key words. cryptography, zero knowledge, interactive proofs, quadratic residues

AMS(MOS) subject classifications. 68Q15, 94A60

1. Introduction. It is often regarded that saying a language L is in NP (that is, acceptable in nondeterministic polynomial time) is equivalent to saying that there is a polynomial time "proof system" for L. The proof system we have in mind is one where on input x, a "prover" creates a string σ, and the "verifier" then computes on x and σ in time polynomial in the length of the binary representation of x to check that x is indeed in L. It is reasonable to ask if there is a more general, and perhaps more

Figure: Golwasser, Micali, Rackoff 1985 Paper

ZKPs: First Definition



SIAM J. COMPUT.
Vol. 18, No. 1, pp. 186–208, February 1989

© 1989 Society for Industrial and Applied Mathematics
002

THE KNOWLEDGE COMPLEXITY OF INTERACTIVE PROOF SYSTEMS*

SHAFI GOLDWASSER†, SILVIO MICALI‡, AND CHARLES RACKOFF‡

Abstract. Usually, a proof of a theorem contains more knowledge than the mere fact that the theorem is true. For instance, to prove that a graph is Hamiltonian it suffices to exhibit a Hamiltonian tour in it; however, this seems to contain more knowledge than the single bit Hamiltonian/non-Hamiltonian.

In this paper a computational complexity theory of the "knowledge" contained in a proof is developed. Zero-knowledge proofs are defined as those proofs that convey no additional knowledge other than the correctness of the proposition in question. Examples of zero-knowledge proof systems are given for the languages of quadratic residuosity and quadratic nonresiduosity. These are the first examples of zero-knowledge proofs for languages not known to be efficiently recognizable.

Key words. cryptography, zero knowledge, interactive proofs, quadratic residues

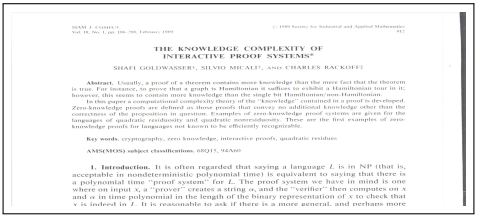
AMS(MOS) subject classifications. 68Q15, 94A60

1. Introduction. It is often regarded that saying a language L is in NP (that is, acceptable in nondeterministic polynomial time) is equivalent to saying that there is a polynomial time "proof system" for L. The proof system we have in mind is one where on input x, a "prover" creates a string or, and the "verifier" then computes on x and α in time polynomial in the length of the binary representation of x to check that x is indeed in L. It is reasonable to ask if there is a more general, and perhaps more

Figure: Golwasser, Micali, Rackoff 1985 Paper

- Appeared first in this paper, where the ZKPs are formally defined as follows.....

ZKPs: First Definition



SIAM J. COMPUT.
Vol. 18, No. 1, pp. 186–208, February 1989

© 1989 Society for Industrial and Applied Mathematics
002

THE KNOWLEDGE COMPLEXITY OF INTERACTIVE PROOF SYSTEMS*

SHAFI GOLDWASSER†, SILVIO MICALI‡, AND CHARLES RACKOFF‡

Abstract. Usually, a proof of a theorem contains more knowledge than the mere fact that the theorem is true. For instance, to prove that a graph is Hamiltonian it suffices to exhibit a Hamiltonian tour in it; however, this seems to contain more knowledge than the single bit Hamiltonian/non-Hamiltonian.

In this paper a computational complexity theory of the "knowledge" contained in a proof is developed. Zero-knowledge proofs are defined as those proofs that convey no additional knowledge other than the correctness of the proposition in question. Examples of zero-knowledge proof systems are given for the languages of quadratic residuosity and quadratic nonresiduosity. These are the first examples of zero-knowledge proofs for languages not known to be efficiently recognizable.

Key words. cryptography, zero knowledge, interactive proofs, quadratic residues

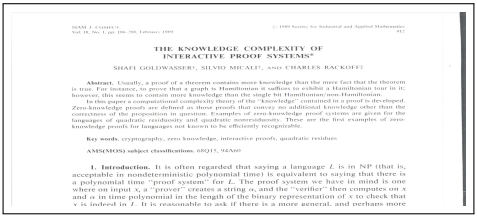
AMS(MOS) subject classifications. 68Q15, 94A60

1. Introduction. It is often regarded that saying a language L is in NP (that is, acceptable in nondeterministic polynomial time) is equivalent to saying that there is a polynomial time "proof system" for L. The proof system we have in mind is one where on input x, a "prover" creates a string or, and the "verifier" then computes on x and α in time polynomial in the length of the binary representation of x to check that x is indeed in L. It is reasonable to ask if there is a more general, and perhaps more

Figure: Golwasser, Micali, Rackoff 1985 Paper

- Appeared first in this paper, where the ZKPs are formally defined as follows.....

ZKPs: First Definition



SIAM J. COMPUT.
Vol. 18, No. 1, pp. 186–208, February 1989

© 1989 Society for Industrial and Applied Mathematics
002

THE KNOWLEDGE COMPLEXITY OF INTERACTIVE PROOF SYSTEMS*

SHAFI GOLDWASSER†, SILVIO MICALI‡, AND CHARLES RACKOFF‡

Abstract. Usually, a proof of a theorem contains more knowledge than the mere fact that the theorem is true. For instance, to prove that a graph is Hamiltonian it suffices to exhibit a Hamiltonian tour in it; however, this seems to contain more knowledge than the single bit Hamiltonian/non-Hamiltonian.

In this paper a computational complexity theory of the "knowledge" contained in a proof is developed. Zero-knowledge proofs are defined as those proofs that convey no additional knowledge other than the correctness of the proposition in question. Examples of zero-knowledge proof systems are given for the languages of quadratic residuosity and quadratic nonresiduosity. These are the first examples of zero-knowledge proofs for languages not known to be efficiently recognizable.

Key words. cryptography, zero knowledge, interactive proofs, quadratic residues

AMS(MOS) subject classifications. 68Q15, 94A60

1. Introduction. It is often regarded that saying a language L is in NP (that is, acceptable in nondeterministic polynomial time) is equivalent to saying that there is a polynomial time "proof system" for L. The proof system we have in mind is one where on input x, a "prover" creates a string or, and the "verifier" then computes on x and σ in time polynomial in the length of the binary representation of x to check that x is indeed in L. It is reasonable to ask if there is a more general, and perhaps more

Figure: Golwasser, Micali, Rackoff 1985 Paper

- Appeared first in this paper, where the ZKPs are formally defined as follows.....

First Definition of ZKP

A zero-knowledge protocol is a method by which one party (the prover) can prove to another party (the verifier) that something is true, without revealing any information apart from the fact that this specific statement is true.

S Goldwasser, S Micali, and C Rackoff. The knowledge complexity of interactive proof-systems. In Proc. of the 17th Annual ACM symposium on Theory of computing (STOC '85). 1985, ACM, NY, USA, 291–304.

Why do we need zero-knowledge proofs?

- There are many situations where we preserve the privacy and yet achieve the desired functionality. As for example.....

Figure: Proving the claim without revealing the knowledge

Fig Src: https://medium.com/@kotsbtechcdac/introduction-to-zero-knowledge-proof-the-protocol-of-next-generation-blockchain-305b2fc7f8e5

Why do we need zero-knowledge proofs?

- There are many situations where we preserve the privacy and yet achieve the desired functionality. As for example.....
  - how one might prove a claim (e.g., "I am a citizen of X country") to another party (e.g., a service provider).

Figure: Proving the claim without revealing the knowledge

Fig Src: https://medium.com/@kotsbtechcdac/introduction-to-zero-knowledge-proof-the-protocol-of-next-generation-blockchain-305b2fc7f8e5

Why do we need zero-knowledge proofs?

- There are many situations where we preserve the privacy and yet achieve the desired functionality. As for example.....
  - how one might prove a claim (e.g., "I am a citizen of X country") to another party (e.g., a service provider).
  - one may need to provide *evidence* to back up the claim, such as a *national passport or driver's license* OR some other PII

Figure: Proving the claim without revealing the knowledge

Fig Src: https://medium.com/@kotsbtechcdac/introduction-to-zero-knowledge-proof-the-protocol-of-next-generation-blockchain-305b2fc7f8e5

Why do we need zero-knowledge proofs?

- There are many situations where we preserve the privacy and yet achieve the desired functionality. As for example.....
  - how one might prove a claim (e.g., "I am a citizen of X country") to another party (e.g., a service provider).
  - one may need to provide *evidence* to back up the claim, such as a *national passport or driver's license* OR some other PII
  - results in the lack of privacy - the PIIs shared with third-party services is stored in central databases, which are vulnerable to hacks.

Figure: Proving the claim without revealing the knowledge

Fig Src: https://medium.com/@kotsbtechcdac/introduction-to-zero-knowledge-proof-the-protocol-of-next-generation-blockchain-305b2fc7f8e5

Why do we need zero-knowledge proofs?

- There are many situations where we preserve the privacy and yet achieve the desired functionality. As for example.....
  - how one might prove a claim (e.g., "I am a citizen of X country") to another party (e.g., a service provider).
  - one may need to provide *evidence* to back up the claim, such as a *national passport or driver's license* OR some other PII
  - results in the lack of privacy - the PIIs shared with third-party services is stored in central databases, which are vulnerable to hacks.
  - another example: one may send his/her public key p_k to a verifier V and to claim that he/she knows the secret key s_k corresponding to p_k.

Figure: Proving the claim without revealing the knowledge

Fig Src: https://medium.com/@kotsbtechcdac/introduction-to-zero-knowledge-proof-the-protocol-of-next-generation-blockchain-305b2fc7f8e5

Why do we need zero-knowledge proofs?

- There are many situations where we preserve the privacy and yet achieve the desired functionality. As for example.....
  - how one might prove a claim (e.g., "I am a citizen of X country") to another party (e.g., a service provider).
  - one may need to provide *evidence* to back up the claim, such as a *national passport or driver's license* OR some other PII
  - results in the lack of privacy - the PIIs shared with third-party services is stored in central databases, which are vulnerable to hacks.
  - another example: one may send his/her public key p_k to a verifier V and to claim that he/she knows the secret key s_k corresponding to p_k.
- Identity theft becomes a critical issue, here

Figure: Proving the claim without revealing the knowledge

Fig Src: https://medium.com/@kotsbtechcdac/introduction-to-zero-knowledge-proof-the-protocol-of-next-generation-blockchain-305b2fc7f8e5

Why do we need zero-knowledge proofs?...

- Many situations today, where one needs to provide all kinds of data to an online service to be authenticated - i.e. share a lot of private information.

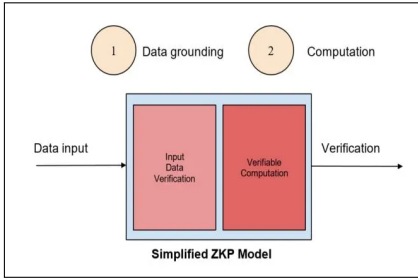


Figure: Eliminate the need to reveal information shared

Fig Src: https://www.newrealityblog.com/2019/02/20/an-introduction-to-zero-knowledge-proofs-and-three-ways-it-will-influence-our-world-part-1/

Why do we need zero-knowledge proofs?...

- Many situations today, where one needs to provide all kinds of data to an online service to be authenticated - i.e. share a lot of private information.
- the authentication in IoT devices may be between machine to machine,

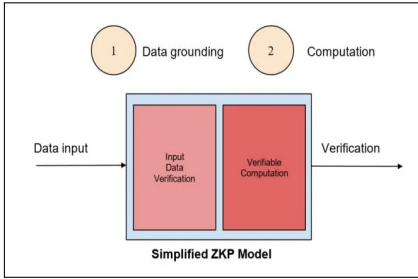


Figure: Eliminate the need to reveal information shared

Fig Src: https://www.newrealityblog.com/2019/02/20/an-introduction-to-zero-knowledge-proofs-and-three-ways-it-will-influence-our-world-part-1/

Why do we need zero-knowledge proofs?...

- Many situations today, where one needs to provide all kinds of data to an online service to be authenticated - i.e. share a lot of private information.
- the authentication in IoT devices may be between machine to machine,
- whenever a person or machine exchanges data, it is exposed to a data breach.

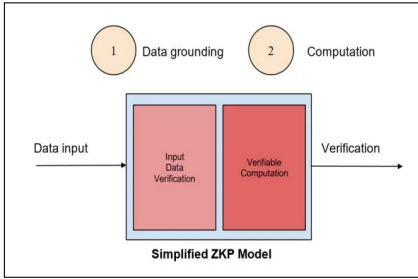


Figure: Eliminate the need to reveal information shared

Fig Src: https://www.newrealityblog.com/2019/02/20/an-introduction-to-zero-knowledge-proofs-and-three-ways-it-will-influence-our-world-part-1/

Why do we need zero-knowledge proofs?...

- Many situations today, where one needs to provide all kinds of data to an online service to be authenticated - i.e. share a lot of private information.
- the authentication in IoT devices may be between machine to machine,
- whenever a person or machine exchanges data, it is exposed to a data breach.
- Here, ZKPs allow the creation of a proof that a given computer program and set of inputs lead to the computed set of output values.

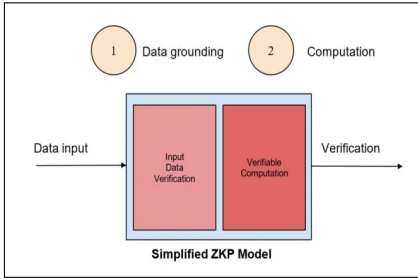


Figure: Eliminate the need to reveal information shared

Fig Src: https://www.newrealityblog.com/2019/02/20/an-introduction-to-zero-knowledge-proofs-and-three-ways-it-will-influence-our-world-part-1/

Why do we need zero-knowledge proofs?...

- Many situations today, where one needs to provide all kinds of data to an online service to be authenticated - i.e. share a lot of private information.
- the authentication in IoT devices may be between machine to machine,
- whenever a person or machine exchanges data, it is exposed to a data breach.
- Here, ZKPs allow the creation of a proof that a given computer program and set of inputs lead to the computed set of output values.
- As the prover (i.e. the computer program, here) needs access to the input, values cannot be hidden (obviously) from the prover.

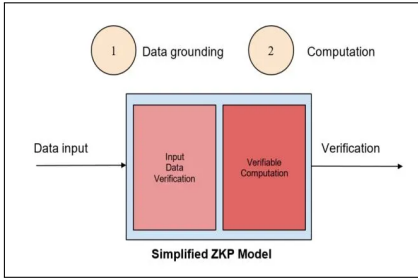


Figure: Eliminate the need to reveal information shared

Fig Src: https://www.newrealityblog.com/2019/02/20/an-introduction-to-zero-knowledge-proofs-and-three-ways-it-will-influence-our-world-part-1/

Why do we need zero-knowledge proofs?...

- Many situations today, where one needs to provide all kinds of data to an online service to be authenticated - i.e. share a lot of private information.
- the authentication in IoT devices may be between machine to machine,
- whenever a person or machine exchanges data, it is exposed to a data breach.
- Here, ZKPs allow the creation of a proof that a given computer program and set of inputs lead to the computed set of output values.
- As the prover (i.e. the computer program, here) needs access to the input, values cannot be hidden (obviously) from the prover.
- The solution for these use-cases always has two parts, as shown in the figure, here.

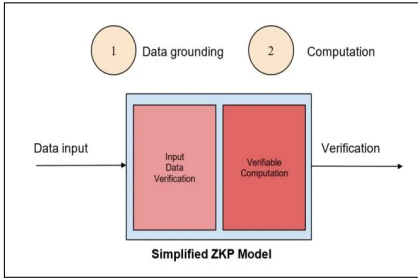


Figure: Eliminate the need to reveal information shared

Fig Src: https://www.newrealityblog.com/2019/02/20/an-introduction-to-zero-knowledge-proofs-and-three-ways-it-will-influence-our-world-part-1/

Why do we need zero-knowledge proofs?...

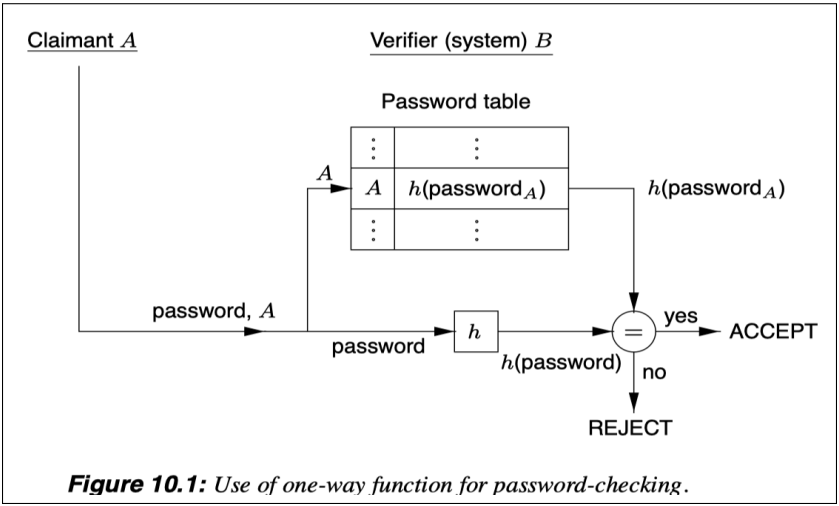


Figure 10.1: *Use of one-way function for password-checking.*

Figure: As above

Fig Src: Menezes, vanOorschot, Vanstone, Handbook of Applied Cryptography

ZKP: Formal Statement

The formal defintion of ZKP can be given as follows:

- Consider the problem parametrized by a binary relation $R \subset 0,1 * X0,1*$

ZKP: Formal Statement

The formal defintion of ZKP can be given as follows:

- Consider the problem parametrized by a binary relation $R \subset 0,1 * X0,1*$
- For a pair $(x, w)\ \epsilon\ R$, we call x an instance, and w the witness.

ZKP: Formal Statement

The formal defintion of ZKP can be given as follows:

- Consider the problem parametrized by a binary relation $R \subset 0,1 * X0,1*$
- For a pair $(x, w)\ \epsilon\ R$, we call x an instance, and w the witness.
- The prover knows $(x, w)\ \epsilon\ R$ and sends x to the verifier, claiming to know w such that $(x, w)\ \epsilon\ R$.

ZKP: Formal Statement

The formal defintion of ZKP can be given as follows:

- Consider the problem parametrized by a binary relation $R \subset 0,1 * X0,1*$
- For a pair $(x, w)\ \epsilon\ R$, we call x an instance, and w the witness.
- The prover knows $(x, w)\ \epsilon\ R$ and sends x to the verifier, claiming to know w such that $(x, w)\ \epsilon\ R$.
  - e.g. in the example of public-private key pairs provenance, $x = p_k$ and $w = s_k$, and R would be pairs (p_k, s_k) for which s_k is the secret key corresponding to p_k.

ZKP: Formal Statement

The formal defintion of ZKP can be given as follows:

- Consider the problem parametrized by a binary relation $R \subset 0,1 * X0,1*$
- For a pair $(x, w) \epsilon R$, we call x an instance, and w the witness.
- The prover knows $(x, w) \epsilon R$ and sends x to the verifier, claiming to know w such that $(x, w) \epsilon R$.
  - e.g. in the example of public-private key pairs provenance, $x = p_k$ and $w = s_k$, and R would be pairs (p_k, s_k) for which s_k is the secret key corresponding to p_k.
  - The protocol P is required to substantiate the claim without leaking anything about w to V.

ZKP: Formal Statement

The formal defintion of ZKP can be given as follows:

- Consider the problem parametrized by a binary relation $R \subset 0,1 * X0,1*$
- For a pair $(x, w) \epsilon R$, we call x an instance, and w the witness.
- The prover knows $(x, w) \epsilon R$ and sends x to the verifier, claiming to know w such that $(x, w) \epsilon R$.
  - e.g. in the example of public-private key pairs provenance, $x = p_k$ and $w = s_k$, and R would be pairs (p_k, s_k) for which s_k is the secret key corresponding to p_k.
  - The protocol P is required to substantiate the claim without leaking anything about w to V.
- Efficient multiparty computation can be used to construct efficient zero-knowledge proofs of knowledge.

Zero Knowledge Proofs USecases and Applications

ZKP: A Toy usecase

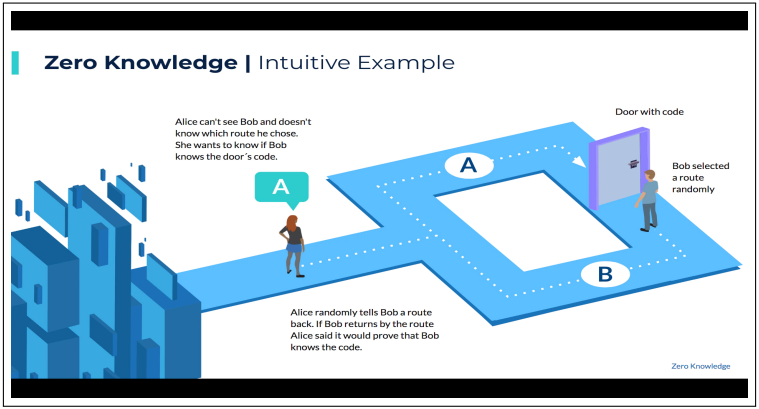


Figure: Alibaba Cave ZKP Usecase

• Bob (the 'prover') owns a code for a door inside cave and Alice (the 'verifier') wants to buy it, but first she wants to be sure that Bob is not lying. How can Bob show Alice that he has the code without revealing its contents?

Src: https://www.bbva.com/en/zero-knowledge-proof-how-to-maintain-privacy-in-a-data-based-world/

ZKP still relevant in 2020s ?

Why have ZKP been so talked off today, having been invented wayback in 1985 ?

- Blockchain technology and ZKP are tied to each other.

ZKP still relevant in 2020s ?

Why have ZKP been so talked off today, having been invented wayback in 1985 ?

- Blockchain technology and ZKP are tied to each other.
- Blockchain technology

ZKP still relevant in 2020s ?

Why have ZKP been so talked off today, having been invented wayback in 1985 ?

- Blockchain technology and ZKP are tied to each other.
- Blockchain technology
  - has revolutionized the traditional finance industry by bringing decentralization, transparency, and immutability to the forefront.

ZKP still relevant in 2020s ?

Why have ZKP been so talked off today, having been invented wayback in 1985 ?

- Blockchain technology and ZKP are tied to each other.
- Blockchain technology
  - has revolutionized the traditional finance industry by bringing decentralization, transparency, and immutability to the forefront.
  - uses distributed ledger technology - does not need the overseer in the form of a trusted central authority.

ZKP still relevant in 2020s ?

Why have ZKP been so talked off today, having been invented wayback in 1985 ?

- Blockchain technology and ZKP are tied to each other.
- Blockchain technology
  - has revolutionized the traditional finance industry by bringing decentralization, transparency, and immutability to the forefront.
  - uses distributed ledger technology - does not need the overseer in the form of a trusted central authority.
  - aims to enable users to act anonymously, perform transactions with high-end security, and give users control over their data and privacy back.

ZKP still relevant in 2020s ?

Why have ZKP been so talked off today, having been invented wayback in 1985 ?

- Blockchain technology and ZKP are tied to each other.
- Blockchain technology
  - has revolutionized the traditional finance industry by bringing decentralization, transparency, and immutability to the forefront.
  - uses distributed ledger technology - does not need the overseer in the form of a trusted central authority.
  - aims to enable users to act anonymously, perform transactions with high-end security, and give users control over their data and privacy back.
- However, there is privacy paradox in the blockchains

ZKP still relevant in 2020s ?

Why have ZKP been so talked off today, having been invented wayback in 1985 ?

- Blockchain technology and ZKP are tied to each other.
- Blockchain technology
  - has revolutionized the traditional finance industry by bringing decentralization, transparency, and immutability to the forefront.
  - uses distributed ledger technology - does not need the overseer in the form of a trusted central authority.
  - aims to enable users to act anonymously, perform transactions with high-end security, and give users control over their data and privacy back.
- However, there is privacy paradox in the blockchains
  - at present, the transactions on public blockchain networks recorded on the public ledger, have to be transparent.

ZKP still relevant in 2020s ?

Why have ZKP been so talked off today, having been invented wayback in 1985 ?

- Blockchain technology and ZKP are tied to each other.
- Blockchain technology
  - has revolutionized the traditional finance industry by bringing decentralization, transparency, and immutability to the forefront.
  - uses distributed ledger technology - does not need the overseer in the form of a trusted central authority.
  - aims to enable users to act anonymously, perform transactions with high-end security, and give users control over their data and privacy back.
- However, there is privacy paradox in the blockchains
  - at present, the transactions on public blockchain networks recorded on the public ledger, have to be transparent.
  - that is, all participants in public blockchain networks trust in the sanctity of the information

ZKP still relevant in 2020s ?

Why have ZKP been so talked off today, having been invented wayback in 1985 ?

- Blockchain technology and ZKP are tied to each other.
- Blockchain technology
  - has revolutionized the traditional finance industry by bringing decentralization, transparency, and immutability to the forefront.
  - uses distributed ledger technology - does not need the overseer in the form of a trusted central authority.
  - aims to enable users to act anonymously, perform transactions with high-end security, and give users control over their data and privacy back.
- However, there is privacy paradox in the blockchains
  - at present, the transactions on public blockchain networks recorded on the public ledger, have to be transparent.
  - that is, all participants in public blockchain networks trust in the sanctity of the information
  - This is so, because they can all see and analyze that information equally and in real time.

ZKP still relevant in 2020s ?

Why have ZKP been so talked off today, having been invented wayback in 1985 ?

- Blockchain technology and ZKP are tied to each other.
- Blockchain technology
  - has revolutionized the traditional finance industry by bringing decentralization, transparency, and immutability to the forefront.
  - uses distributed ledger technology - does not need the overseer in the form of a trusted central authority.
  - aims to enable users to act anonymously, perform transactions with high-end security, and give users control over their data and privacy back.
- However, there is privacy paradox in the blockchains
  - at present, the transactions on public blockchain networks recorded on the public ledger, have to be transparent.
  - that is, all participants in public blockchain networks trust in the sanctity of the information
  - This is so, because they can all see and analyze that information equally and in real time.
  - e.g. DApps. What are these ?

ZKP still relevant in 2020s ?

Why have ZKP been so talked off today, having been invented wayback in 1985 ?

- Blockchain technology and ZKP are tied to each other.
- Blockchain technology
  - has revolutionized the traditional finance industry by bringing decentralization, transparency, and immutability to the forefront.
  - uses distributed ledger technology - does not need the overseer in the form of a trusted central authority.
  - aims to enable users to act anonymously, perform transactions with high-end security, and give users control over their data and privacy back.
- However, there is privacy paradox in the blockchains
  - at present, the transactions on public blockchain networks recorded on the public ledger, have to be transparent.
  - that is, all participants in public blockchain networks trust in the sanctity of the information
  - This is so, because they can all see and analyze that information equally and in real time.
  - e.g. DApps. What are these ?

ZKP still relevant in 2020s ?

Why have ZKP been so talked off today, having been invented wayback in 1985 ?

- Blockchain technology and ZKP are tied to each other.
- Blockchain technology
  - has revolutionized the traditional finance industry by bringing decentralization, transparency, and immutability to the forefront.
  - uses distributed ledger technology - does not need the overseer in the form of a trusted central authority.
  - aims to enable users to act anonymously, perform transactions with high-end security, and give users control over their data and privacy back.
- However, there is privacy paradox in the blockchains
  - at present, the transactions on public blockchain networks recorded on the public ledger, have to be transparent.
  - that is, all participants in public blockchain networks trust in the sanctity of the information
  - This is so, because they can all see and analyze that information equally and in real time.
  - e.g. DApps. What are these ?

Src: A Must Read: https://www.dentons.com/en/insights/articles/2022/june/9/the-privacy-paradox-in-blockchain-best-practices-for-data-management-in-crypto

ZKP still relevant in 2020s ...?

What are decentralized applications (DApps) and why they face privacy concerns?

- DApps are built from software deployed on the blockchain (e.g., smart contracts) that are typically designed to execute business operations for companies.

ZKP still relevant in 2020s ...?

What are decentralized applications (DApps) and why they face privacy concerns?

- DApps are built from software deployed on the blockchain (e.g., smart contracts) that are typically designed to execute business operations for companies.
- The operations of the smart contracts/DApps - are often made publicly available to every node.

ZKP still relevant in 2020s ...?

What are decentralized applications (DApps) and why they face privacy concerns?

- DApps are built from software deployed on the blockchain (e.g., smart contracts) that are typically designed to execute business operations for companies.
- The operations of the smart contracts/DApps - are often made publicly available to every node.
- This is in the form bytecode, - that can be reverse engineered to reveal the same transactional information as metadata in peer-to-peer transactions

ZKP still relevant in 2020s ...?

What are decentralized applications (DApps) and why they face privacy concerns?

- DApps are built from software deployed on the blockchain (e.g., smart contracts) that are typically designed to execute business operations for companies.
- The operations of the smart contracts/DApps - are often made publicly available to every node.
- This is in the form bytecode, - that can be reverse engineered to reveal the same transactional information as metadata in peer-to-peer transactions
- But, what about the privacy concerns?

ZKP still relevant in 2020s ...?

What are decentralized applications (DApps) and why they face privacy concerns?

- DApps are built from software deployed on the blockchain (e.g., smart contracts) that are typically designed to execute business operations for companies.
- The operations of the smart contracts/DApps - are often made publicly available to every node.
- This is in the form bytecode, - that can be reverse engineered to reveal the same transactional information as metadata in peer-to-peer transactions
- But, what about the privacy concerns?
- Today, the inventors and developers look at ZKPs as a panacea to enhance privacy and security.

ZKP still relevant in 2020s ...?

What are decentralized applications (DApps) and why they face privacy concerns?

- DApps are built from software deployed on the blockchain (e.g., smart contracts) that are typically designed to execute business operations for companies.
- The operations of the smart contracts/DApps - are often made publicly available to every node.
- This is in the form bytecode, - that can be reverse engineered to reveal the same transactional information as metadata in peer-to-peer transactions
- But, what about the privacy concerns?
- Today, the inventors and developers look at ZKPs as a panacea to enhance privacy and security.

ZKP still relevant in 2020s ...?

What are decentralized applications (DApps) and why they face privacy concerns?

- DApps are built from software deployed on the blockchain (e.g., smart contracts) that are typically designed to execute business operations for companies.
- The operations of the smart contracts/DApps - are often made publicly available to every node.
- This is in the form bytecode, - that can be reverse engineered to reveal the same transactional information as metadata in peer-to-peer transactions
- But, what about the privacy concerns?
- Today, the inventors and developers look at ZKPs as a panacea to enhance privacy and security.

Src: A Must Read: https://www.dentons.com/en/insights/articles/2022/june/9/the-privacy-paradox-in-blockchain-best-practices-for-data-management-in-crypto

ZKP in Bitcoin and Ethereum: Public verification of transactions

- Enhancing privacy in cryptocurrencies: Cryptocurrencies Zcash, Monero (https://www.getmonero.org/, use ZKPs to provide anonymous transactions when sending and receiving money over its network.

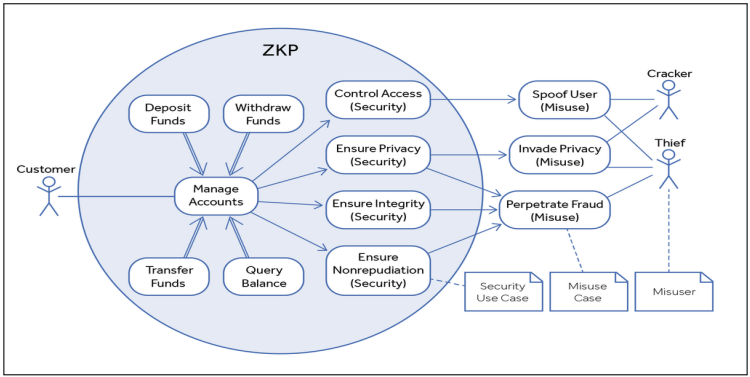


Figure: One of the ZKP Usecase

Src: https://www.privid.co/usecases

Other interesting usecases of ZKP

- Opaque pricing

Other interesting usecases of ZKP

- Opaque pricing
  - Two competitive business houses paying to buy the same material from the same supplier.

Other interesting usecases of ZKP

- Opaque pricing
  - Two competitive business houses paying to buy the same material from the same supplier.
  - Both wish to know whether both are paying the same price per kilogram of the material or not, without revealing own or knowing the other's price.

Other interesting usecases of ZKP

- Opaque pricing
  - Two competitive business houses paying to buy the same material from the same supplier.
  - Both wish to know whether both are paying the same price per kilogram of the material or not, without revealing own or knowing the other's price.
- Self-Sovereign Identity (SSI) systems

Other interesting usecases of ZKP

- Opaque pricing
  - Two competitive business houses paying to buy the same material from the same supplier.
  - Both wish to know whether both are paying the same price per kilogram of the material or not, without revealing own or knowing the other's price.
- Self-Sovereign Identity (SSI) systems
  - where users have the mechanisms to control, consent, and widely use their identities among different services

Other interesting usecases of ZKP

- Opaque pricing
  - Two competitive business houses paying to buy the same material from the same supplier.
  - Both wish to know whether both are paying the same price per kilogram of the material or not, without revealing own or knowing the other's price.
- Self-Sovereign Identity (SSI) systems
  - where users have the mechanisms to control, consent, and widely use their identities among different services

Other interesting usecases of ZKP

- Opaque pricing
  - Two competitive business houses paying to buy the same material from the same supplier.
  - Both wish to know whether both are paying the same price per kilogram of the material or not, without revealing own or knowing the other's price.
- Self-Sovereign Identity (SSI) systems
  - where users have the mechanisms to control, consent, and widely use their identities among different services

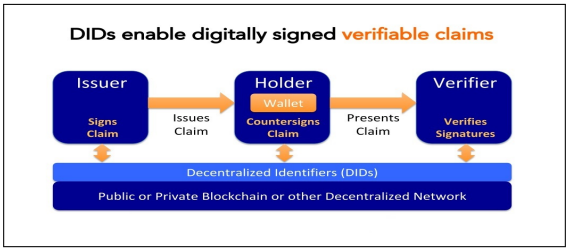


Figure: Decentralized identifiers enable digitally-signed verifiable claims

Figure Src: wiki

Other interesting usecases of ZKP...

- Blockchains for ensuring privacy and Verifiable computations
  - ZKPs are used in Blockchain and Cryptocurrencies for various purposes such as Anonymous payments, Verifiable computations, Reducing bribery and collusion in on-chain voting, Secure Transactions and so on.

Other interesting usecases of ZKP...

- Blockchains for ensuring privacy and Verifiable computations
  - ZKPs are used in Blockchain and Cryptocurrencies for various purposes such as Anonymous payments, Verifiable computations, Reducing bribery and collusion in on-chain voting, Secure Transactions and so on.
  - the transparency of public blockchains such as Bitcoin and Ethereum enable public verification of transactions.

Other interesting usecases of ZKP...

- Blockchains for ensuring privacy and Verifiable computations
  - ZKPs are used in Blockchain and Cryptocurrencies for various purposes such as Anonymous payments, Verifiable computations, Reducing bribery and collusion in on-chain voting, Secure Transactions and so on.
  - the transparency of public blockchains such as Bitcoin and Ethereum enable public verification of transactions.
  - However, it also implies little privacy and can lead to deanonymization of users.

Other interesting usecases of ZKP...

- Blockchains for ensuring privacy and Verifiable computations
  - ZKPs are used in Blockchain and Cryptocurrencies for various purposes such as Anonymous payments, Verifiable computations, Reducing bribery and collusion in on-chain voting, Secure Transactions and so on.
  - the transparency of public blockchains such as Bitcoin and Ethereum enable public verification of transactions.
  - However, it also implies little privacy and can lead to deanonymization of users.
  - ZKPs can introduce more privacy to public blockchains.

Other interesting usecases of ZKP...

- Blockchains for ensuring privacy and Verifiable computations
  - ZKPs are used in Blockchain and Cryptocurrencies for various purposes such as Anonymous payments, Verifiable computations, Reducing bribery and collusion in on-chain voting, Secure Transactions and so on.
  - the transparency of public blockchains such as Bitcoin and Ethereum enable public verification of transactions.
  - However, it also implies little privacy and can lead to deanonymization of users.
  - ZKPs can introduce more privacy to public blockchains.
  - Three such example projects are cryptocurrency Zcash, the Ethereum blockchain iand ZK-Rollup.

Other interesting usecases of ZKP...

- Blockchains for ensuring privacy and Verifiable computations
  - the cryptocurrency Zcash is based on Zero-Knowledge Succinct Non-Interactive Argument of Knowledge (zk-SNARK),

Other interesting usecases of ZKP...

- Blockchains for ensuring privacy and Verifiable computations
  - the cryptocurrency Zcash is based on Zero-Knowledge Succinct Non-Interactive Argument of Knowledge (zk-SNARK),
  - it is a type of zero-knowledge cryptographic method for

Other interesting usecases of ZKP...

- Blockchains for ensuring privacy and Verifiable computations
  - the cryptocurrency Zcash is based on Zero-Knowledge Succinct Non-Interactive Argument of Knowledge (zk-SNARK),
  - it is a type of zero-knowledge cryptographic method for
    - transactions to be private and fully encrypted on the blockchain, while still being validated using the network's consensus rules.

Other interesting usecases of ZKP...

- Blockchains for ensuring privacy and Verifiable computations
  - the cryptocurrency Zcash is based on Zero-Knowledge Succinct Non-Interactive Argument of Knowledge (zk-SNARK),
  - it is a type of zero-knowledge cryptographic method for
    - transactions to be private and fully encrypted on the blockchain, while still being validated using the network's consensus rules.
    - can show that the sender has the amount of funds they want to transfer without making that information public.

Other interesting usecases of ZKP...

- Blockchains for ensuring privacy and Verifiable computations
  - the cryptocurrency Zcash is based on Zero-Knowledge Succinct Non-Interactive Argument of Knowledge (zk-SNARK),
  - it is a type of zero-knowledge cryptographic method for
    - transactions to be private and fully encrypted on the blockchain, while still being validated using the network's consensus rules.
    - can show that the sender has the amount of funds they want to transfer without making that information public.
  - the Ethereum blockchain is based on Zero-Knowledge Scalable Transparent Argument of Knowledge (zk-STARK)

Other interesting usecases of ZKP...

- Blockchains for ensuring privacy and Verifiable computations
  - the cryptocurrency Zcash is based on Zero-Knowledge Succinct Non-Interactive Argument of Knowledge (zk-SNARK),
  - it is a type of zero-knowledge cryptographic method for
    - transactions to be private and fully encrypted on the blockchain, while still being validated using the network's consensus rules.
    - can show that the sender has the amount of funds they want to transfer without making that information public.
  - the Ethereum blockchain is based on Zero-Knowledge Scalable Transparent Argument of Knowledge (zk-STARK)
    - zk-STARK provides privacy and scalability - for a similar purpose as zk-SNARK

Other interesting usecases of ZKP...: zk_SNARK

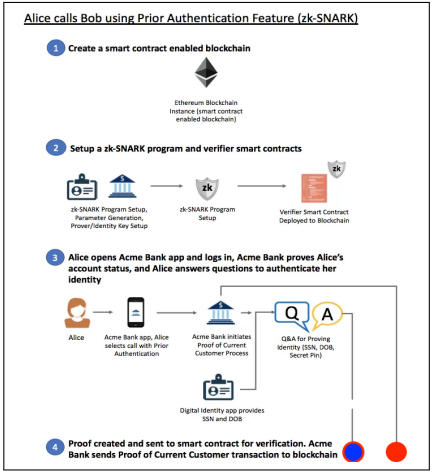


Figure: Alice calls Bob using prior authentication based on ZK-snark

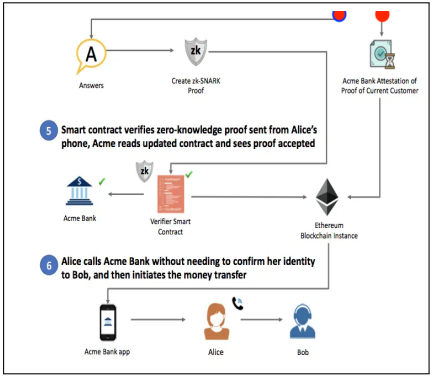


Figure: Alice calls Bob using prior authentication based on ZK-snark

Figure Src: https://medium.com/coinmonks/zk-snarks-a-realistic-zero-knowledge-example-and-deep-dive-c5e6eaa7131c

Other interesting usecases of ZKP...: zk_SNARK...

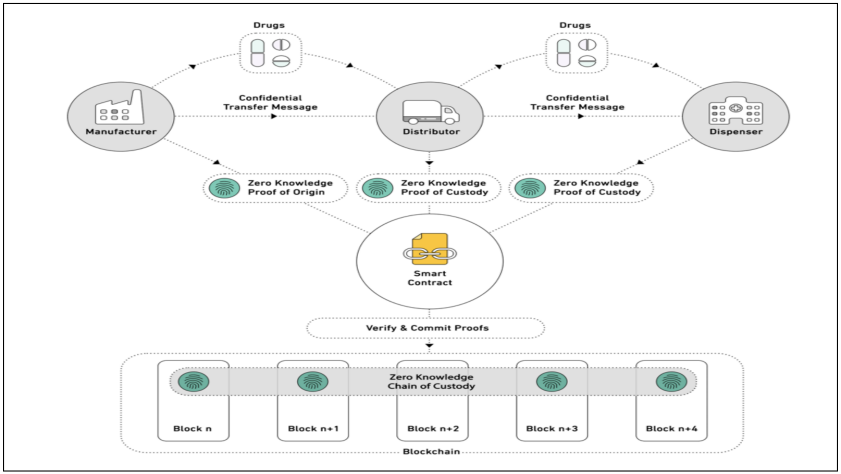


Figure: ZK-Snark for pharmaceutical industry

Figure Src: https://fisher.wharton.upenn.edu/wp-content/uploads/2020/09/Thesis_Terrence-Jo.pdf

Other interesting usecases of ZKP...

Zero-knowledge rollups (ZK-rollups)

- bundle (or 'roll up') transactions in Ethereum into batches that are executed off-chain.

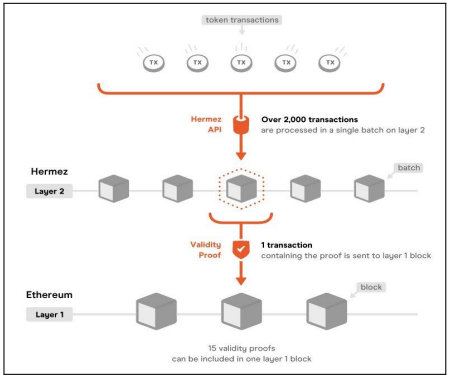


Figure: Hermez Crypto Ecosystem: A popular ZK-rollup project

Figure Src: https://blog.pantherprotocol.io/zk-rollup-projects-inner-workings-importance-analysis/

Other interesting usecases of ZKP...

Zero-knowledge rollups (ZK-rollups)

- bundle (or 'roll up') transactions in Ethereum into batches that are executed off-chain.
- off-chain computation reduces the amount of data that has to be posted to the blockchain.

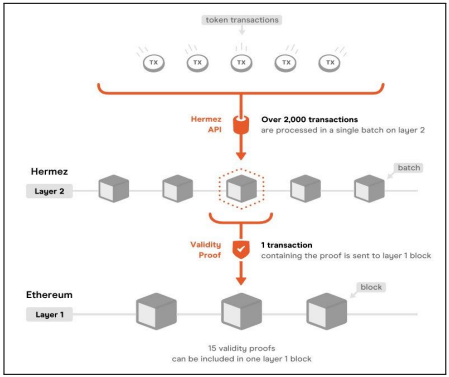


Figure: Hermez Crypto Ecosystem: A popular ZK-rollup project

Figure Src: https://blog.pantherprotocol.io/zk-rollup-projects-inner-workings-importance-analysis/

Other interesting usecases of ZKP...

Zero-knowledge rollups (ZK-rollups)

- bundle (or 'roll up') transactions in Ethereum into batches that are executed off-chain.
- off-chain computation reduces the amount of data that has to be posted to the blockchain.
- operators submit a summary of the changes required to represent all the transactions in a batch rather than sending each transaction individually.

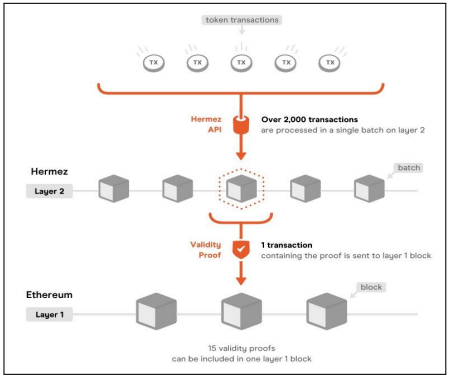


Figure: Hermez Crypto Ecosystem: A popular ZK-rollup project

Figure Src: https://blog.pantherprotocol.io/zk-rollup-projects-inner-workings-importance-analysis/

Other interesting usecases of ZKP...

Zero-knowledge rollups (ZK-rollups)

- bundle (or 'roll up') transactions in Ethereum into batches that are executed off-chain.
- off-chain computation reduces the amount of data that has to be posted to the blockchain.
- operators submit a summary of the changes required to represent all the transactions in a batch rather than sending each transaction individually.
- also produce validity proofs to prove the correctnes of their changes.

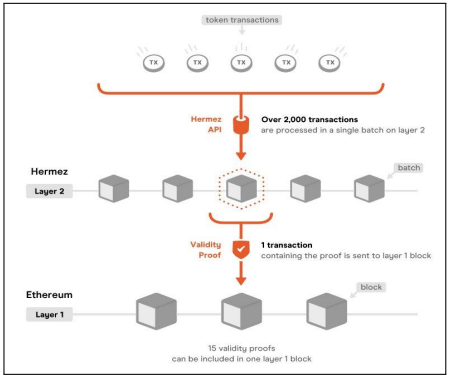


Figure: Hermez Crypto Ecosystem: A popular ZK-rollup project

Figure Src: https://blog.pantherprotocol.io/zk-rollup-projects-inner-workings-importance-analysis/

Other interesting usecases of ZKP...

Zero-knowledge rollups (ZK-rollups)

- bundle (or 'roll up') transactions in Ethereum into batches that are executed off-chain.
- off-chain computation reduces the amount of data that has to be posted to the blockchain.
- operators submit a summary of the changes required to represent all the transactions in a batch rather than sending each transaction individually.
- also produce validity proofs to prove the correctnes of their changes.
- the validity proof - based on ZKP -demonstrates with cryptographic certainty that the proposed changes to Ethereum's state are truly the end-result of executing all the transactions in the batch.

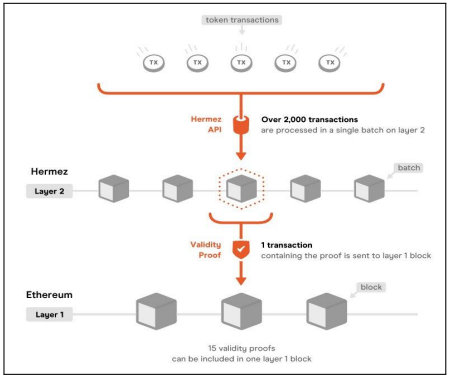


Figure: Hermez Crypto Ecosystem: A popular ZK-rollup project

Figure Src: https://blog.pantherprotocol.io/zk-rollup-projects-inner-workings-importance-analysis/

Other interesting usecases of ZKP...

- Finance

Other interesting usecases of ZKP...

- Finance
  - ING - the banking company uses ZKPs that allow customers to prove that their secret number lies in a known range.

Other interesting usecases of ZKP...

- Finance
  - ING - the banking company uses ZKPs that allow customers to prove that their secret number lies in a known range.
  - For example, a mortgage applicant can prove that their income is in the admissible range without revealing their exact salary.

Other interesting usecases of ZKP...

- Finance
  - ING - the banking company uses ZKPs that allow customers to prove that their secret number lies in a known range.
  - For example, a mortgage applicant can prove that their income is in the admissible range without revealing their exact salary.
- Online voting

Other interesting usecases of ZKP...

- Finance
  - ING - the banking company uses ZKPs that allow customers to prove that their secret number lies in a known range.
  - For example, a mortgage applicant can prove that their income is in the admissible range without revealing their exact salary.
- Online voting
  - ZKPs can allow voters to vote anonymously and to verify that their vote was included in the final tally.

Other interesting usecases of ZKP...

- Finance
  - ING - the banking company uses ZKPs that allow customers to prove that their secret number lies in a known range.
  - For example, a mortgage applicant can prove that their income is in the admissible range without revealing their exact salary.
- Online voting
  - ZKPs can allow voters to vote anonymously and to verify that their vote was included in the final tally.
- Authentication

Other interesting usecases of ZKP...

- Finance
  - ING - the banking company uses ZKPs that allow customers to prove that their secret number lies in a known range.
  - For example, a mortgage applicant can prove that their income is in the admissible range without revealing their exact salary.
- Online voting
  - ZKPs can allow voters to vote anonymously and to verify that their vote was included in the final tally.
- Authentication
  - ZKPs can be used to authenticate users without exchanging secret information such as passwords.

Other interesting usecases of ZKP...

- Finance
  - ING - the banking company uses ZKPs that allow customers to prove that their secret number lies in a known range.
  - For example, a mortgage applicant can prove that their income is in the admissible range without revealing their exact salary.
- Online voting
  - ZKPs can allow voters to vote anonymously and to verify that their vote was included in the final tally.
- Authentication
  - ZKPs can be used to authenticate users without exchanging secret information such as passwords.
- Machine Learning

Other interesting usecases of ZKP...

- Finance
  - ING - the banking company uses ZKPs that allow customers to prove that their secret number lies in a known range.
  - For example, a mortgage applicant can prove that their income is in the admissible range without revealing their exact salary.
- Online voting
  - ZKPs can allow voters to vote anonymously and to verify that their vote was included in the final tally.
- Authentication
  - ZKPs can be used to authenticate users without exchanging secret information such as passwords.
- Machine Learning
  - ZKPs can allow the owner of a machine learning algorithm to convince others about the results of the model without revealing any information about the ML model itself.

Other interesting usecases of ZKP...

- Proof of membership in Social Networks

Other interesting usecases of ZKP...

- Proof of membership in Social Networks
  - A person Alice meets unknown an other person Bobon a social network - one who is not known to Alice - but claims to be also a member of the group Alice is part of.

Other interesting usecases of ZKP...

- Proof of membership in Social Networks
  - A person Alice meets unknown an other person Bobon a social network - one who is not known to Alice - but claims to be also a member of the group Alice is part of.
  - How can Alice know if she can trust Bob?

Other interesting usecases of ZKP...

- Proof of membership in Social Networks
  - A person Alice meets unknown an other person Bobon a social network - one who is not known to Alice - but claims to be also a member of the group Alice is part of.
  - How can Alice know if she can trust Bob?
- Critical Applications like Nuclear disarmament:

Other interesting usecases of ZKP...

- Proof of membership in Social Networks
  - A person Alice meets unknown an other person Bobon a social network - one who is not known to Alice - but claims to be also a member of the group Alice is part of.
  - How can Alice know if she can trust Bob?
- Critical Applications like Nuclear disarmament:
  - In 2016, Princeton Plasma Physics Laboratory and Princeton University demonstrated technique that would allow inspectors to confirm if an object is nuclear weapon or not without recording, sharing, or revealing internal workings.

Other interesting usecases of ZKP...

- Proof of membership in Social Networks
  - A person Alice meets unknown an other person Bobon a social network - one who is not known to Alice - but claims to be also a member of the group Alice is part of.
  - How can Alice know if she can trust Bob?
- Critical Applications like Nuclear disarmament:
  - In 2016, Princeton Plasma Physics Laboratory and Princeton University demonstrated technique that would allow inspectors to confirm if an object is nuclear weapon or not without recording, sharing, or revealing internal workings.
- Authentication systems

Other interesting usecases of ZKP...

- Proof of membership in Social Networks
  - A person Alice meets unknown an other person Bobon a social network - one who is not known to Alice - but claims to be also a member of the group Alice is part of.
  - How can Alice know if she can trust Bob?
- Critical Applications like Nuclear disarmament:
  - In 2016, Princeton Plasma Physics Laboratory and Princeton University demonstrated technique that would allow inspectors to confirm if an object is nuclear weapon or not without recording, sharing, or revealing internal workings.
- Authentication systems
  - the usecase seen earlier about verification of a user password without revealing it to system.

Zero Knowledge Proofs Requirements and Constituents

ZKP Requirements

- ZK protocols rely on algorithms that take some data as input and return 'true' or 'false' as output.
- A zero-knowledge protocol must satisfy the following criteria:
  - Completeness:

ZKP Requirements

- ZK protocols rely on algorithms that take some data as input and return 'true' or 'false' as output.
- A zero-knowledge protocol must satisfy the following criteria:
  - Completeness:
    - If the input is valid, the zero-knowledge protocol must always returns true.

ZKP Requirements

- ZK protocols rely on algorithms that take some data as input and return 'true' or 'false' as output.
- A zero-knowledge protocol must satisfy the following criteria:
  - Completeness:
    - If the input is valid, the zero-knowledge protocol must always returns true.
    - Hence, if the underlying statement is true, and the prover and verifier act honestly, the proof can be accepted.

ZKP Requirements

- ZK protocols rely on algorithms that take some data as input and return 'true' or 'false' as output.
- A zero-knowledge protocol must satisfy the following criteria:
  - Completeness:
    - If the input is valid, the zero-knowledge protocol must always returns true.
    - Hence, if the underlying statement is true, and the prover and verifier act honestly, the proof can be accepted.
  - Soundness:

ZKP Requirements

- ZK protocols rely on algorithms that take some data as input and return 'true' or 'false' as output.
- A zero-knowledge protocol must satisfy the following criteria:
  - Completeness:
    - If the input is valid, the zero-knowledge protocol must always returns true.
    - Hence, if the underlying statement is true, and the prover and verifier act honestly, the proof can be accepted.
  - Soundness:
    - If the input is invalid, it is theoretically impossible to fool the zero-knowledge protocol to return 'true'.

ZKP Requirements

- ZK protocols rely on algorithms that take some data as input and return 'true' or 'false' as output.
- A zero-knowledge protocol must satisfy the following criteria:
  - Completeness:
    - If the input is valid, the zero-knowledge protocol must always returns true.
    - Hence, if the underlying statement is true, and the prover and verifier act honestly, the proof can be accepted.
  - Soundness:
    - If the input is invalid, it is theoretically impossible to fool the zero-knowledge protocol to return 'true'.
    - Hence, a lying prover cannot trick an honest verifier into believing an invalid statement is valid - except with a tiny margin of probability.

ZKP Requirements

- ZK protocols rely on algorithms that take some data as input and return 'true' or 'false' as output.
- A zero-knowledge protocol must satisfy the following criteria:
  - Completeness:
    - If the input is valid, the zero-knowledge protocol must always returns true.
    - Hence, if the underlying statement is true, and the prover and verifier act honestly, the proof can be accepted.
  - Soundness:
    - If the input is invalid, it is theoretically impossible to fool the zero-knowledge protocol to return 'true'.
    - Hence, a lying prover cannot trick an honest verifier into believing an invalid statement is valid - except with a tiny margin of probability.
  - Zero-knowledge

ZKP Requirements

- ZK protocols rely on algorithms that take some data as input and return 'true' or 'false' as output.
- A zero-knowledge protocol must satisfy the following criteria:
  - Completeness:
    - If the input is valid, the zero-knowledge protocol must always returns true.
    - Hence, if the underlying statement is true, and the prover and verifier act honestly, the proof can be accepted.
  - Soundness:
    - If the input is invalid, it is theoretically impossible to fool the zero-knowledge protocol to return 'true'.
    - Hence, a lying prover cannot trick an honest verifier into believing an invalid statement is valid - except with a tiny margin of probability.
  - Zero-knowledge
    - The verifier learns nothing about a statement beyond its validity or falsity (i.e. they have "zero knowledge" of the statement).

ZKP Requirements

- ZK protocols rely on algorithms that take some data as input and return 'true' or 'false' as output.
- A zero-knowledge protocol must satisfy the following criteria:
  - Completeness:
    - If the input is valid, the zero-knowledge protocol must always returns true.
    - Hence, if the underlying statement is true, and the prover and verifier act honestly, the proof can be accepted.
  - Soundness:
    - If the input is invalid, it is theoretically impossible to fool the zero-knowledge protocol to return 'true'.
    - Hence, a lying prover cannot trick an honest verifier into believing an invalid statement is valid - except with a tiny margin of probability.
  - Zero-knowledge
    - The verifier learns nothing about a statement beyond its validity or falsity (i.e. they have "zero knowledge" of the statement).
    - This requirement also prevents the verifier from deriving the original input (the statement's contents) from the proof.

Basic Constituents of a ZKP

In basic form, a zero-knowledge proof is made up of three elements: witness, challenge, and response.

- Witness:

Basic Constituents of a ZKP

In basic form, a zero-knowledge proof is made up of three elements: witness, challenge, and response.

- Witness:
  - With a zero-knowledge proof, the prover wants to prove knowledge of some hidden information.

Basic Constituents of a ZKP

In basic form, a zero-knowledge proof is made up of three elements: witness, challenge, and response.

- Witness:
  - With a zero-knowledge proof, the prover wants to prove knowledge of some hidden information.
  - the secret information is the witness to the proof...

Basic Constituents of a ZKP

In basic form, a zero-knowledge proof is made up of three elements: witness, challenge, and response.

- Witness:
  - With a zero-knowledge proof, the prover wants to prove knowledge of some hidden information.
  - the secret information is the witness to the proof...
  - the prover's assumed knowledge of the witness establishes a set of questions that can only be answered by a party with knowledge of the information.

Basic Constituents of a ZKP

In basic form, a zero-knowledge proof is made up of three elements: witness, challenge, and response.

- Witness:
  - With a zero-knowledge proof, the prover wants to prove knowledge of some hidden information.
  - the secret information is the witness to the proof...
  - the prover's assumed knowledge of the witness establishes a set of questions that can only be answered by a party with knowledge of the information.
  - Thus, the prover starts the proving process by randomly choosing a question, calculating the answer, and sending it to the verifier.

Basic Constituents of a ZKP

In basic form, a zero-knowledge proof is made up of three elements: witness, challenge, and response.

- Witness:
  - With a zero-knowledge proof, the prover wants to prove knowledge of some hidden information.
  - the secret information is the witness to the proof...
  - the prover's assumed knowledge of the witness establishes a set of questions that can only be answered by a party with knowledge of the information.
  - Thus, the prover starts the proving process by randomly choosing a question, calculating the answer, and sending it to the verifier.
- Challenge

continued....

Basic Constituents of a ZKP

In basic form, a zero-knowledge proof is made up of three elements: witness, challenge, and response.

- Witness:
  - With a zero-knowledge proof, the prover wants to prove knowledge of some hidden information.
  - the secret information is the witness to the proof...
  - the prover's assumed knowledge of the witness establishes a set of questions that can only be answered by a party with knowledge of the information.
  - Thus, the prover starts the proving process by randomly choosing a question, calculating the answer, and sending it to the verifier.
- Challenge
  - The verifier randomly picks another question from the set and asks the prover to answer it

continued....

Basic Constituents of a ZKP...

In basic form, a zero-knowledge proof is made up of three elements: witness, challenge, and response.

- Response

Basic Constituents of a ZKP...

In basic form, a zero-knowledge proof is made up of three elements: witness, challenge, and response.

- Response
  - The prover accepts the question, calculates the answer, and returns the answer to the verifier.

Basic Constituents of a ZKP...

In basic form, a zero-knowledge proof is made up of three elements: witness, challenge, and response.

- Response
  - The prover accepts the question, calculates the answer, and returns the answer to the verifier.
  - The prover's response allows the verifier to check if the former really has access to the witness.

Basic Constituents of a ZKP...

In basic form, a zero-knowledge proof is made up of three elements: witness, challenge, and response.

- Response
  - The prover accepts the question, calculates the answer, and returns the answer to the verifier.
  - The prover's response allows the verifier to check if the former really has access to the witness.
  - To ensure the prover isn't guessing blindly, the verifier picks more questions to ask.

Basic Constituents of a ZKP...

In basic form, a zero-knowledge proof is made up of three elements: witness, challenge, and response.

- Response
  - The prover accepts the question, calculates the answer, and returns the answer to the verifier.
  - The prover's response allows the verifier to check if the former really has access to the witness.
  - To ensure the prover isn't guessing blindly, the verifier picks more questions to ask.
  - By repeating this interaction many times, the possibility of the prover faking knowledge of the witness drops significantly until the verifier is satisfied.

Basic Constituents of a ZKP...

In basic form, a zero-knowledge proof is made up of three elements: witness, challenge, and response.

- Response
  - The prover accepts the question, calculates the answer, and returns the answer to the verifier.
  - The prover's response allows the verifier to check if the former really has access to the witness.
  - To ensure the prover isn't guessing blindly, the verifier picks more questions to ask.
  - By repeating this interaction many times, the possibility of the prover faking knowledge of the witness drops significantly until the verifier is satisfied.

Basic Constituents of a ZKP...

In basic form, a zero-knowledge proof is made up of three elements: witness, challenge, and response.

- Response
  - The prover accepts the question, calculates the answer, and returns the answer to the verifier.
  - The prover's response allows the verifier to check if the former really has access to the witness.
  - To ensure the prover isn't guessing blindly, the verifier picks more questions to ask.
  - By repeating this interaction many times, the possibility of the prover faking knowledge of the witness drops significantly until the verifier is satisfied.

• The components described are those of the early zero-knowledge protocols - ones that used interactive zero-knowledge proof - ....

Types of ZKPs

Typically the ZKP protocols take one of the two forms:

- Interactive ZKP

Types of ZKPs

Typically the ZKP protocols take one of the two forms:

- Interactive ZKP
  - the two parties viz. prover and the reverifier need to communicate in rounds with information flowing in both directions throughout the protocol, to conclude the proof.

Types of ZKPs

Typically the ZKP protocols take one of the two forms:

- Interactive ZKP
  - the two parties viz. prover and the reverifier need to communicate in rounds with information flowing in both directions throughout the protocol, to conclude the proof.
- Non-Interactive ZKP

Types of ZKPs

Typically the ZKP protocols take one of the two forms:

- Interactive ZKP
  - the two parties viz. prover and the reverifier need to communicate in rounds with information flowing in both directions throughout the protocol, to conclude the proof.
- Non-Interactive ZKP
  - here, the prover only needs to run a program taking (f, x, y) as input and outputs the proof p_i.

Types of ZKPs

Typically the ZKP protocols take one of the two forms:

- Interactive ZKP
  - the two parties viz. prover and the reverifier need to communicate in rounds with information flowing in both directions throughout the protocol, to conclude the proof.
- Non-Interactive ZKP
  - here, the prover only needs to run a program taking (f, x, y) as input and outputs the proof p_i.
  - any verifier with (f, y, p_i) can verify it and be convinced that the prover indeed knows x without seeing it.

Types of ZKPs

Typically the ZKP protocols take one of the two forms:

- Interactive ZKP
  - the two parties viz. prover and the reverifier need to communicate in rounds with information flowing in both directions throughout the protocol, to conclude the proof.
- Non-Interactive ZKP
  - here, the prover only needs to run a program taking (f, x, y) as input and outputs the proof p_i.
  - any verifier with (f, y, p_i) can verify it and be convinced that the prover indeed knows x without seeing it.
  - the workflow of an NIZK proof resembles a digital signature:

Interactive Zero Knowledge Proofs

Types of ZKP: Interactive ZKPs

- Interactive zero-knowledge proofs:

Types of ZKP: Interactive ZKPs

- Interactive zero-knowledge proofs:
  - here, the prover and the verifier interact several times in order yield the conclusion.

Types of ZKP: Interactive ZKPs

- Interactive zero-knowledge proofs:
  - here, the prover and the verifier interact several times in order yield the conclusion.
  - the verifier challenges the prover who provides replies to these challenges

Types of ZKP: Interactive ZKPs

- Interactive zero-knowledge proofs:
  - here, the prover and the verifier interact several times in order yield the conclusion.
  - the verifier challenges the prover who provides replies to these challenges
  - the interaction continues till the verifier is convinced about the possession of the knowledge by the prover.

Types of ZKP: Interactive ZKPs

- Interactive zero-knowledge proofs:
  - here, the prover and the verifier interact several times in order yield the conclusion.
  - the verifier challenges the prover who provides replies to these challenges
  - the interaction continues till the verifier is convinced about the possession of the knowledge by the prover.
- suffer from the limitations viz.

Types of ZKP: Interactive ZKPs

- Interactive zero-knowledge proofs:
  - here, the prover and the verifier interact several times in order yield the conclusion.
  - the verifier challenges the prover who provides replies to these challenges
  - the interaction continues till the verifier is convinced about the possession of the knowledge by the prover.
- suffer from the limitations viz.
  - Limited transferability - In order to prove same proof again to another verifier, entire process needs to be repeated and

Types of ZKP: Interactive ZKPs

- Interactive zero-knowledge proofs:
  - here, the prover and the verifier interact several times in order yield the conclusion.
  - the verifier challenges the prover who provides replies to these challenges
  - the interaction continues till the verifier is convinced about the possession of the knowledge by the prover.
- suffer from the limitations viz.
  - Limited transferability - In order to prove same proof again to another verifier, entire process needs to be repeated and
  - Lack of scalability – require both verifier and prover to be online at same time - not scalable.

Interactive ZKP: Basic Operation

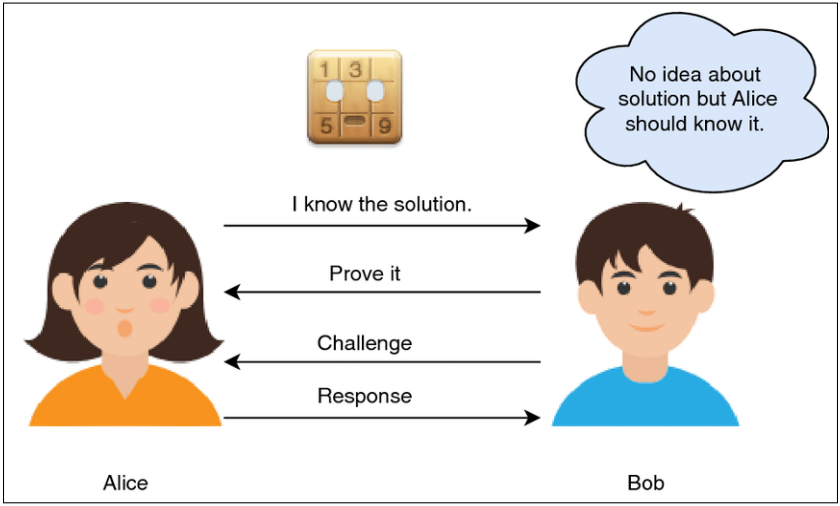


Figure: Interactive ZKP

Src: https://medium.com/@kotsbtechcdac/introduction-to-zero-knowledge-proof-the-protocol-of-next-generation-blockchain-305b2fc7f8e5

Interactive ZKP: A Usecase for Data Exchange in an Auction

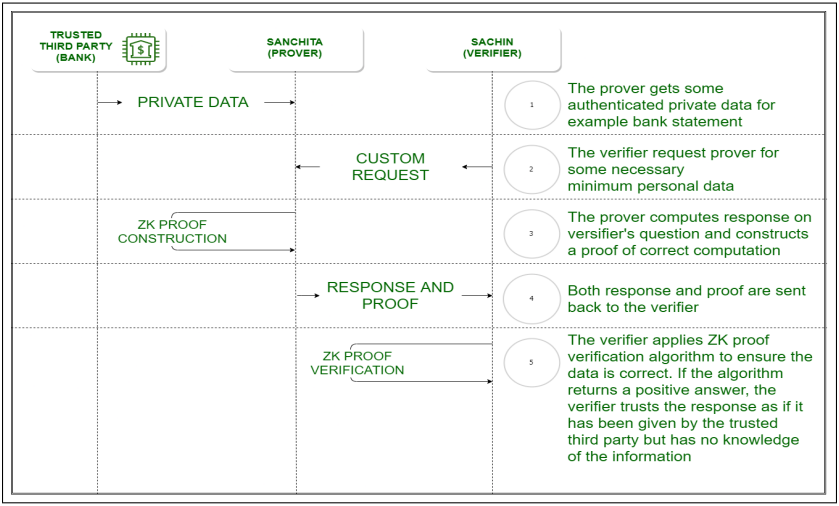


Figure: Interactive ZKP

Src: https://medium.com/@kotsbtechcdac/introduction-to-zero-knowledge-proof-the-protocol-of-next-generation-blockchain-305b2fc7f8e5

Interactive ZKP: A Usecase for Data Exchange in an Auction

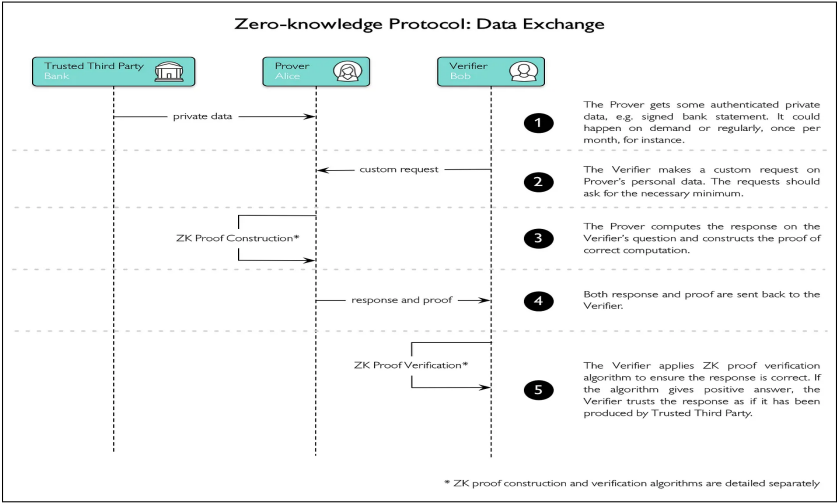


Figure: Interactive ZKP

Src: https://medium.com/@kotsbtechcdac/introduction-to-zero-knowledge-proof-the-protocol-of-next-generation-blockchain-305b2fc7f8e5

Interactive Zero Knowledge Proofs Protocols

Discrete Logarithm & Discrete Logarithm Problem

- In mathematics, we already know for given real numbers a and b, the logarithm log_b^a is a number x such that $b^x = a$.

Discrete Logarithm & Discrete Logarithm Problem

- In mathematics, we already know for given real numbers a and b, the logarithm log_b^a is a number x such that $b^x = a$.
  - a=8, b=2, x=3, we have log_2^8=x=3, $2^3 = 8$.

Discrete Logarithm & Discrete Logarithm Problem

- In mathematics, we already know for given real numbers a and b, the logarithm log_b^a is a number x such that $b^x = a$.
  - a=8, b=2, x=3, we have log_2^8=x=3, $2^3 = 8$.
- Analogously, in any group $\mathbb{G}$, powers b^k can be defined for all integers k, and the discrete logarithm log_b^a is an integer k such that $b^k = a$ .....OR

Discrete Logarithm & Discrete Logarithm Problem

- In mathematics, we already know for given real numbers a and b, the logarithm log_b^a is a number x such that $b^x = a$.
  - a=8, b=2, x=3, we have log_2^8=x=3, $2^3 = 8$.
- Analogously, in any group $\mathbb{G}$, powers b^k can be defined for all integers k, and the discrete logarithm log_b^a is an integer k such that $b^k = a$ .....OR
- given three integers a, b and m, finding an integer k such that $a^k \equiv b \pmod{m}$ where a and m are relatively prime. If it is not possible for any k to satisfy this relation, print -1.

```
Input: 2 3 5
Output: 3
Explanation:
a = 2, b = 3, m = 5
The value which satisfies the above equation
is 3, because
=> 2^3 = 2 * 2 * 2 = 8
=> 2^3 (mod 5) = 8 (mod 5)
=> 3
which is equal to b i.e., 3.

Input: 3 7 11
Output: -1
```

Figure: Discrete Logarithm

Discrete Logarithm Problem

- Given a cyclic group $\mathbb{G}$ of order m, a group element h, a generator g of the group, (i.e. $h^g \bmod \mathbb{G} \equiv 1 \bmod \mathbb{G}$) the problem is to find an integer k such that $g^k \equiv h \pmod{\ } \mathbb{G}$

Discrete Logarithm Problem

- Given a cyclic group $\mathbb{G}$ of order m, a group element h, a generator g of the group, (i.e. h^g mod $\mathbb{G} \equiv 1$ mod $\mathbb{G}$) the problem is to find an integer k such that $g^k \equiv h$ (mod) $\mathbb{G}$
- this is the same as saying that given a group $\mathbb{G}$, a generator g of the group $\mathbb{G}$ and an element h of $\mathbb{G}$, to find the discrete logarithm to the base g of h in the group $\mathbb{G}$ i.e. to find log_g^h mod $\mathbb{G}$

Discrete Logarithm Problem

- Given a cyclic group $\mathbb{G}$ of order m, a group element h, a generator g of the group, (i.e. h^g mod $\mathbb{G} \equiv 1$ mod $\mathbb{G}$) the problem is to find an integer k such that $g^k \equiv h$ (mod) $\mathbb{G}$
- this is the same as saying that given a group $\mathbb{G}$, a generator g of the group $\mathbb{G}$ and an element h of $\mathbb{G}$, to find the discrete logarithm to the base g of h in the group $\mathbb{G}$ i.e. to find log_g^h mod $\mathbb{G}$
- e.g. consider the following parameters: $\mathbb{Z}_5$, generator $g = 2$, element h=1, then the discrete logarithm of 1 is 4 because 2^4 mod $5 \equiv 1$ mod 5.

Why Schnorr's protocol ?

- A well-known principle for designing robust public key protocols is as follows: Do not assume that a message you receive has a particular form (such as g^r for known r) unless you can check this

Why Schnorr's protocol ?

- A well-known principle for designing robust public key protocols is as follows: Do not assume that a message you receive has a particular form (such as g^r for known r) unless you can check this
- This is the sixth of the eight principles defined by Ross Anderson and Roger Needham at InRedCrypto '95.

Why Schnorr's protocol ?

- A well-known principle for designing robust public key protocols is as follows: Do not assume that a message you receive has a particular form (such as g^r for known r) unless you can check this
- This is the sixth of the eight principles defined by Ross Anderson and Roger Needham at InRedCrypto '95.
- Hence, it is also known as the "sixth principle".

Why Schnorr's protocol ?

- A well-known principle for designing robust public key protocols is as follows: Do not assume that a message you receive has a particular form (such as g^r for known r) unless you can check this
- This is the sixth of the eight principles defined by Ross Anderson and Roger Needham at InRedCrypto '95.
- Hence, it is also known as the "sixth principle".
- one technique that allows one to prove the knowledge of a discrete logarithm (e.g., r for g^r) without revealing its value. is the Schnorr Identification scheme.

Why Schnorr's protocol ?

- A well-known principle for designing robust public key protocols is as follows: Do not assume that a message you receive has a particular form (such as g^r for known r) unless you can check this
- This is the sixth of the eight principles defined by Ross Anderson and Roger Needham at InRedCrypto '95.
- Hence, it is also known as the "sixth principle".
- one technique that allows one to prove the knowledge of a discrete logarithm (e.g., r for g^r) without revealing its value. is the Schnorr Identification scheme.
- The Schnorr identification scheme runs interactively between Alice(prover) and Bob(verifier).

Why Schnorr's protocol ?

- A well-known principle for designing robust public key protocols is as follows: Do not assume that a message you receive has a particular form (such as g^r for known r) unless you can check this
- This is the sixth of the eight principles defined by Ross Anderson and Roger Needham at InRedCrypto '95.
- Hence, it is also known as the "sixth principle".
- one technique that allows one to prove the knowledge of a discrete logarithm (e.g., r for g^r) without revealing its value. is the Schnorr Identification scheme.
- The Schnorr identification scheme runs interactively between Alice(prover) and Bob(verifier).
  - In the setup of the scheme, Alice publishes her public key $A = g^a \ mod \ p$, where a is the private key chosen uniformly at random from $[0, q-1]$

Why Schnorr's protocol ?

- A well-known principle for designing robust public key protocols is as follows: Do not assume that a message you receive has a particular form (such as g^r for known r) unless you can check this
- This is the sixth of the eight principles defined by Ross Anderson and Roger Needham at InRedCrypto '95.
- Hence, it is also known as the "sixth principle".
- one technique that allows one to prove the knowledge of a discrete logarithm (e.g., r for g^r) without revealing its value. is the Schnorr Identification scheme.
- The Schnorr identification scheme runs interactively between Alice(prover) and Bob(verifier).
  - In the setup of the scheme, Alice publishes her public key $A = g^a \ mod \ p$, where a is the private key chosen uniformly at random from $[0, q-1]$
- The Schnorr's protocol is such an identification scheme to validate the public key, using interactive ZKP.

Why Schnorr's protocol ?

- A well-known principle for designing robust public key protocols is as follows: Do not assume that a message you receive has a particular form (such as g^r for known r) unless you can check this
- This is the sixth of the eight principles defined by Ross Anderson and Roger Needham at InRedCrypto '95.
- Hence, it is also known as the "sixth principle".
- one technique that allows one to prove the knowledge of a discrete logarithm (e.g., r for g^r) without revealing its value. is the Schnorr Identification scheme.
- The Schnorr identification scheme runs interactively between Alice(prover) and Bob(verifier).
  - In the setup of the scheme, Alice publishes her public key $A = g^a \ mod \ p$, where a is the private key chosen uniformly at random from $[0, q-1]$
- The Schnorr's protocol is such an identification scheme to validate the public key, using interactive ZKP.
- Now let us look at the Schnorr's protocol.....

Objective: With it, a Prover i.e. Alice can convince the Verifier i.e. Bob that she knows the discrete logarithm x of some value $h = g^x$ without revealing x

Public input: cyclic group $\mathbb{G}$ of prime order q, a generator g of $\mathbb{G}$ and an $h \in \mathbb{G}$
Private input: Prover knows secret $x \in {\mathbb{Z}^*}_q$ such that $h = g^x$

- Alice chooses a number r uniformly at random from ${\mathbb{Z}^*}_q$ i.e. from $[0, q-1]$ and computes $u = g^r \bmod q$.

Interactive ZKP: Schnorr's Protocol: Proof of Knowledge of Discrete Log

Objective: With it, a Prover i.e. Alice can convince the Verifier i.e. Bob that she knows the discrete logarithm x of some value $h = g^x$ without revealing x

Public input: cyclic group $\mathbb{G}$ of prime order q, a generator g of $\mathbb{G}$ and an $h \in \mathbb{G}$
Private input: Prover knows secret $x \in {\mathbb{Z}^*}_q$ such that $h = g^x$

- Alice chooses a number r uniformly at random from ${\mathbb{Z}^*}_q$ i.e. from $[0, q-1]$ and computes $u = g^r \bmod q$.
- Alice sends u to Bob.

Interactive ZKP: Schnorr's Protocol: Proof of Knowledge of Discrete Log

Objective: With it, a Prover i.e. Alice can convince the Verifier i.e. Bob that she knows the discrete logarithm x of some value $h = g^x$ without revealing x

Public input: cyclic group $\mathbb{G}$ of prime order q, a generator g of $\mathbb{G}$ and an $h \in \mathbb{G}$
Private input: Prover knows secret $x \in \mathbb{Z}^*_q$ such that $h = g^x$

- Alice chooses a number r uniformly at random from $\mathbb{Z}^*_q$ i.e. from $[0, q-1]$ and computes $u = g^r \bmod q$.
- Alice sends u to Bob.
- Bob chooses a challenge c uniformly at random from $[0, 2^t - 1]$, where t is the bit length of the challenge (say, $t = 160$).

Interactive ZKP: Schnorr's Protocol: Proof of Knowledge of Discrete Log

Objective: With it, a Prover i.e. Alice can convince the Verifier i.e. Bob that she knows the discrete logarithm x of some value $h = g^x$ without revealing x

Public input: cyclic group $\mathbb{G}$ of prime order q, a generator g of $\mathbb{G}$ and an $h \in \mathbb{G}$
Private input: Prover knows secret $x \in \mathbb{Z}^*_q$ such that $h = g^x$

- Alice chooses a number r uniformly at random from $\mathbb{Z}^*_q$ i.e. from $[0, q-1]$ and computes $u = g^r \bmod q$.
- Alice sends u to Bob.
- Bob chooses a challenge c uniformly at random from $[0, 2^t - 1]$, where t is the bit length of the challenge (say, $t = 160$).
- Bob sends c to Alice.

Interactive ZKP: Schnorr's Protocol: Proof of Knowledge of Discrete Log

Objective: With it, a Prover i.e. Alice can convince the Verifier i.e. Bob that she knows the discrete logarithm x of some value $h = g^x$ without revealing x

Public input: cyclic group $\mathbb{G}$ of prime order q, a generator g of $\mathbb{G}$ and an $h \in \mathbb{G}$
Private input: Prover knows secret $x \in \mathbb{Z}^*_q$ such that $h = g^x$

- Alice chooses a number r uniformly at random from $\mathbb{Z}^*_q$ i.e. from $[0, q-1]$ and computes $u = g^r \bmod q$.
- Alice sends u to Bob.
- Bob chooses a challenge c uniformly at random from $[0, 2^t - 1]$, where t is the bit length of the challenge (say, $t = 160$).
- Bob sends c to Alice.
- Alice computes $z = r + x * c \bmod q$

Interactive ZKP: Schnorr's Protocol: Proof of Knowledge of Discrete Log

Objective: With it, a Prover i.e. Alice can convince the Verifier i.e. Bob that she knows the discrete logarithm x of some value $h = g^x$ without revealing x

Public input: cyclic group $\mathbb{G}$ of prime order q, a generator g of $\mathbb{G}$ and an $h \epsilon \mathbb{G}$
Private input: Prover knows secret $x \epsilon \mathbb{Z}^*_q$ such that $h = g^x$

- Alice chooses a number r uniformly at random from $\mathbb{Z}^*_q$ i.e. from $[0, q-1]$ and computes $u = g^r \bmod q$.
- Alice sends u to Bob.
- Bob chooses a challenge c uniformly at random from $[0, 2^t - 1]$, where t is the bit length of the challenge (say, $t = 160$).
- Bob sends c to Alice.
- Alice computes $z = r + x * c \bmod q$
- Alice sends it to Bob. item Bob computes g^z mod q and checks whether $g^z \stackrel{?}{=} u.h^c \bmod q$.

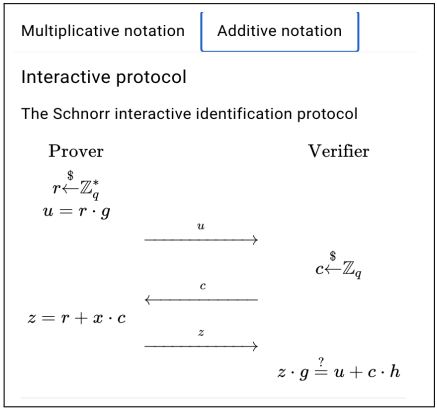


Figure: Interactive ZKP: Schnorr's Protocol: Additive

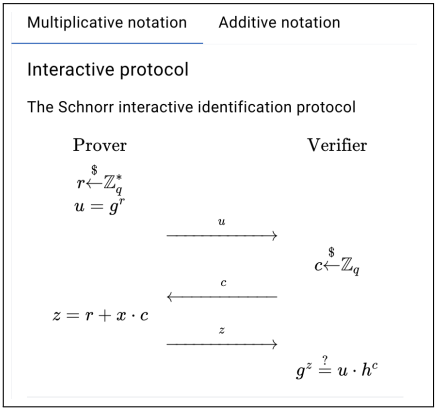


Figure: Interactive ZKP: Schnorr's Protocol: Multiplicative

Src: https://www.zkdocs.com/docs/zkdocs/zero-knowledge-protocols/schnorr/

Interactive ZKP: Schnorr's Protocol: Proof of Knowledge of Discrete Log

- Suppose, a prover wants to prove it knows the discrete logarithm x of some group element $h = g^x \in \mathbb{G}$, where $\mathbb{G}$ is a group of prime order q.

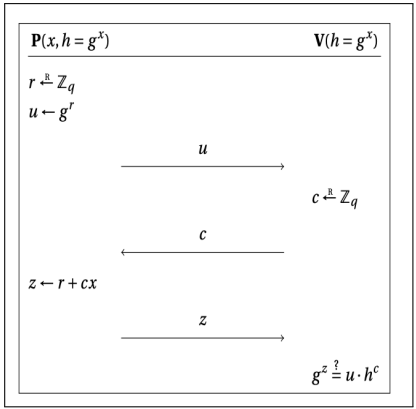


Figure: Interactive ZKP: Schnorr's Protocol

Src: https://www.researchgate.net/figure/Interactive-zero-knowledge-proof_fig3_354455915

Interactive ZKP: Schnorr's Protocol: Proof of Knowledge of Discrete Log

- Suppose, a prover wants to prove it knows the discrete logarithm x of some group element $h = g^x \in \mathbb{G}$, where $\mathbb{G}$ is a group of prime order q.
- Here, $\mathbb{R} = \{ (x, h) \in \mathbb{Z}_{\shortparallel} \times \mathbb{G}: g^x = h \}$, where the group $\mathbb{G}$ and the generator g are public parameters.

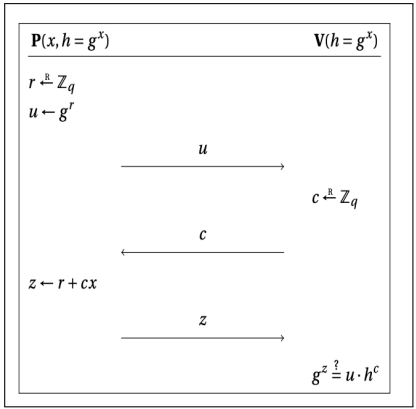


Figure: Interactive ZKP: Schnorr's Protocol

Src: https://www.researchgate.net/figure/Interactive-zero-knowledge-proof_fig3_354455915

Interactive ZKP: Schnorr's Protocol: Proof of Knowledge of Discrete Log

- Suppose, a prover wants to prove it knows the discrete logarithm x of some group element $h = g^x \in \mathbb{G}$, where $\mathbb{G}$ is a group of prime order q.
- Here, $\mathbb{R} = \{ (x, h) \in \mathbb{Z}_{||} \times \mathbb{G}: g^x = h \}$, where the group $\mathbb{G}$ and the generator g are public parameters.
- then the protocol followed is as shown..

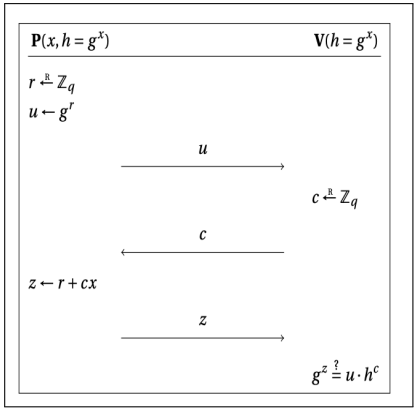


Figure: Interactive ZKP: Schnorr's Protocol

Src: https://www.researchgate.net/figure/Interactive-zero-knowledge-proof_fig3_354455915

Interactive ZKP: Goldwasser Micali Rackoff QR ZKP

Objective: Assuming that QR $= \{(x, y|\ y$ is a quadratic residue mod $x)\}$ and QNR $= \{(x, y|\ y$ is a quadratic nonresidue mod $x)\}$, the goal for the verifier Bob is to verify x is a quadratic residue modulo a number or not.

Shared values: x, the number Alice is trying to prove to Bob that it is quadratic residue and n the modulus being used Private input: As in the code below

- Alice selects a random number v and computes $y = v^2\ mod\ n$

Interactive ZKP: Goldwasser Micali Rackoff QR ZKP

Objective: Assuming that QR $= \{(x, y|\ y$ is a quadratic residue mod $x)\}$ and QNR $= \{(x, y|\ y$ is a quadratic nonresidue mod $x)\}$, the goal for the verifier Bob is to verify x is a quadratic residue modulo a number or not.

Shared values: x, the number Alice is trying to prove to Bob that it is quadratic residue and n the modulus being used Private input: As in the code below

- Alice selects a random number v and computes $y = v^2\ mod\ n$
- Alice sends y to Bob.

Interactive ZKP: Goldwasser Micali Rackoff QR ZKP

Objective: Assuming that QR $= \{(x, y|\ y$ is a quadratic residue mod $x)\}$ and QNR $= \{(x, y|\ y$ is a quadratic nonresidue mod $x)\}$, the goal for the verifier Bob is to verify x is a quadratic residue modulo a number or not.

Shared values: x, the number Alice is trying to prove to Bob that it is quadratic residue and n the modulus being used Private input: As in the code below

- Alice selects a random number v and computes $y = v^2\ mod\ n$
- Alice sends y to Bob.
- Bob chooses a challenge $b\epsilon\ \{0, 1\}$

Interactive ZKP: Goldwasser Micali Rackoff QR ZKP

Objective: Assuming that QR $= \{(x, y|\ y$ is a quadratic residue mod $x)\}$ and QNR $= \{(x, y|\ y$ is a quadratic nonresidue mod $x)\}$, the goal for the verifier Bob is to verify x is a quadratic residue modulo a number or not.

Shared values: x, the number Alice is trying to prove to Bob that it is quadratic residue and n the modulus being used Private input: As in the code below

- Alice selects a random number v and computes $y = v^2\ mod\ n$
- Alice sends y to Bob.
- Bob chooses a challenge $b\epsilon\ \{0, 1\}$
- Bob sends b to Alice.

Interactive ZKP: Goldwasser Micali Rackoff QR ZKP

Objective: Assuming that QR $= \{(x, y|\ y$ is a quadratic residue mod $x)\}$ and QNR $= \{(x, y|\ y$ is a quadratic nonresidue mod $x)\}$, the goal for the verifier Bob is to verify x is a quadratic residue modulo a number or not.

Shared values: x, the number Alice is trying to prove to Bob that it is quadratic residue and n the modulus being used Private input: As in the code below

- Alice selects a random number v and computes $y = v^2\ mod\ n$
- Alice sends y to Bob.
- Bob chooses a challenge $b\epsilon\ \{0, 1\}$
- Bob sends b to Alice.
- Alice selects a number u such that $x = u^2\ mod\ n$

Interactive ZKP: Goldwasser Micali Rackoff QR ZKP

Objective: Assuming that QR $= \{(x, y|\ y$ is a quadratic residue mod $x)\}$ and QNR $= \{(x, y|\ y$ is a quadratic nonresidue mod $x)\}$, the goal for the verifier Bob is to verify x is a quadratic residue modulo a number or not.

Shared values: x, the number Alice is trying to prove to Bob that it is quadratic residue and n the modulus being used Private input: As in the code below

- Alice selects a random number v and computes $y = v^2\ mod\ n$
- Alice sends y to Bob.
- Bob chooses a challenge $b\epsilon\ \{0, 1\}$
- Bob sends b to Alice.
- Alice selects a number u such that $x = u^2\ mod\ n$
- Alice sends to Bob $z = u^b v\ mod\ n$

Interactive ZKP: Goldwasser Micali Rackoff QR ZKP

Objective: Assuming that QR $= \{(x, y|\ y$ is a quadratic residue mod $x)\}$ and QNR $= \{(x, y|\ y$ is a quadratic nonresidue mod $x)\}$, the goal for the verifier Bob is to verify x is a quadratic residue modulo a number or not.

Shared values: x, the number Alice is trying to prove to Bob that it is quadratic residue and n the modulus being used Private input: As in the code below

- Alice selects a random number v and computes $y = v^2\ mod\ n$
- Alice sends y to Bob.
- Bob chooses a challenge $b\epsilon\ \{0, 1\}$
- Bob sends b to Alice.
- Alice selects a number u such that $x = u^2\ mod\ n$
- Alice sends to Bob $z = u^b v\ mod\ n$
- Bob checks to see if $z^2 \equiv xy\ mod\ n$

Interactive ZKP: Goldwasser Micali Rackoff QR ZKP

Objective: Assuming that QR $= \{(x, y|\ y$ is a quadratic residue mod $x)\}$ and QNR $= \{(x, y|\ y$ is a quadratic nonresidue mod $x)\}$, the goal for the verifier Bob is to verify x is a quadratic residue modulo a number or not.

Shared values: x, the number Alice is trying to prove to Bob that it is quadratic residue and n the modulus being used Private input: As in the code below

- Alice selects a random number v and computes $y = v^2\ mod\ n$
- Alice sends y to Bob.
- Bob chooses a challenge $b\epsilon\ \{0, 1\}$
- Bob sends b to Alice.
- Alice selects a number u such that $x = u^2\ mod\ n$
- Alice sends to Bob $z = u^b v\ mod\ n$
- Bob checks to see if $z^2 \equiv xy\ mod\ n$
- How does this equivalence convince Bob that the claim of Alice that x is QR of u is indeed true ?

Interactive ZKP: Fiat-Shamir Simple Identification Protocol

- Objective: Alice to identify herself to verifier Bob by proving knowledge of a secret s (associated with Alice through authentic public data), without revealing any information about InReds not known or computable by B prior to execution of the protocol.

Interactive ZKP: Fiat-Shamir Simple Identification Protocol

- Objective: Alice to identify herself to verifier Bob by proving knowledge of a secret s (associated with Alice through authentic public data), without revealing any information about InReds not known or computable by B prior to execution of the protocol.
- The security relies on the Quadratic Residuosity problem again, i.e. the difficulty of extracting square roots modulo large composite integers n of unknown factorization, which is equivalent to that of factoring n.

Interactive ZKP: Fiat-Shamir Simple Identification Protocol

- Objective: Alice to identify herself to verifier Bob by proving knowledge of a secret s (associated with Alice through authentic public data), without revealing any information about InReds not known or computable by B prior to execution of the protocol.
- The security relies on the Quadratic Residuosity problem again, i.e. the difficulty of extracting square roots modulo large composite integers n of unknown factorization, which is equivalent to that of factoring n.
- then the protocol execution is as shown further....

Interactive ZKP: Fiat-Shamir Simple Identification Protocol

Objective: Alice proves knowledge of s to Bob in t executions of a 3-pass protocol.
One-time setup :

- A trusted center T selects and publishes an RSA-like modulus $n/;= /; pq$. The primes p and q are kept secret.

Interactive ZKP: Fiat-Shamir Simple Identification Protocol

Objective: Alice proves knowledge of s to Bob in t executions of a 3-pass protocol.
One-time setup :

- A trusted center T selects and publishes an RSA-like modulus $n/;= /; pq$. The primes p and q are kept secret.
- The prover Alice selects a secret s coprime to n, $1 \leq s \leq (n - 1)$,

Interactive ZKP: Fiat-Shamir Simple Identification Protocol

Objective: Alice proves knowledge of s to Bob in t executions of a 3-pass protocol.
One-time setup :

- A trusted center T selects and publishes an RSA-like modulus $n/;= /; pq$. The primes p and q are kept secret.
- The prover Alice selects a secret s coprime to n, $1 \leq s \leq (n - 1)$,
- Alice computes $v = s^2 \text{mod } n$

Interactive ZKP: Fiat-Shamir Simple Identification Protocol

Objective: Alice proves knowledge of s to Bob in t executions of a 3-pass protocol.
One-time setup :

- A trusted center T selects and publishes an RSA-like modulus $n/;= /; pq$. The primes p and q are kept secret.
- The prover Alice selects a secret s coprime to n, $1 \leq s \leq (n - 1)$,
- Alice computes $v = s^2 \text{mod } n$
- Alice registers v with T as her public key.

Interactive ZKP: Fiat-Shamir Simple Identification Protocol

Objective: Alice proves knowledge of s to Bob in t executions of a 3-pass protocol.
One-time setup :

- A trusted center T selects and publishes an RSA-like modulus $n/;= /; pq$. The primes p and q are kept secret.
- The prover Alice selects a secret s coprime to n, $1 \leq s \leq (n - 1)$,
- Alice computes $v = s^2 \text{mod } n$
- Alice registers v with T as her public key.

Interactive ZKP: Fiat-Shamir Simple Identification Protocol

Objective: Alice proves knowledge of s to Bob in t executions of a 3-pass protocol.
One-time setup :

- A trusted center T selects and publishes an RSA-like modulus $n/;= /; pq$. The primes p and q are kept secret.
- The prover Alice selects a secret s coprime to n, $1 \leq s \leq (n - 1)$,
- Alice computes $v = s^2 \text{mod } n$
- Alice registers v with T as her public key.

Protocol messages. Each of t rounds has three messages as follows:

- $A \rightarrow B$: $x = r^2 mod\ n$ .....(1)

Interactive ZKP: Fiat-Shamir Simple Identification Protocol

Objective: Alice proves knowledge of s to Bob in t executions of a 3-pass protocol.
One-time setup :

- A trusted center T selects and publishes an RSA-like modulus $n/;= /; pq$. The primes p and q are kept secret.
- The prover Alice selects a secret s coprime to n, $1 \leq$ s $\leq$ (n - 1),
- Alice computes $v = s^2 \text{mod } n$
- Alice registers v with T as her public key.

Protocol messages. Each of t rounds has three messages as follows:

- $A \rightarrow B$: $x = r^2 mod\ n$ .....(1)
- $B \rightarrow A$: e $\epsilon \{0, 1\}$ .....(2)

Interactive ZKP: Fiat-Shamir Simple Identification Protocol

Objective: Alice proves knowledge of s to Bob in t executions of a 3-pass protocol.
One-time setup :

- A trusted center T selects and publishes an RSA-like modulus $n/;= /; pq$. The primes p and q are kept secret.
- The prover Alice selects a secret s coprime to n, $1 \leq s \leq (n - 1)$,
- Alice computes $v = s^2 \text{mod } n$
- Alice registers v with T as her public key.

Protocol messages. Each of t rounds has three messages as follows:

- $A \rightarrow B$: $x = r^2 mod\ n$ .....(1)
- $B \rightarrow A$: e $\epsilon \{0, 1\}$ .....(2)
- $A \rightarrow B$: $y = r * s^e mod\ n$ .....(3)

Interactive ZKP: Fiat-Shamir Simple Identification Protocol

Objective: Alice proves knowledge of s to Bob in t executions of a 3-pass protocol.
One-time setup :

- A trusted center T selects and publishes an RSA-like modulus $n/;= /; pq$. The primes p and q are kept secret.
- The prover Alice selects a secret s coprime to n, $1 \leq s \leq (n - 1)$,
- Alice computes $v = s^2 \text{mod } n$
- Alice registers v with T as her public key.

Protocol messages. Each of t rounds has three messages as follows:

- $A \rightarrow B$: $x = r^2 mod\ n$ .....(1)
- $B \rightarrow A$: e $\epsilon \{0, 1\}$ .....(2)
- $A \rightarrow B$: $y = r * s^e mod\ n$ .....(3)

Interactive ZKP: Fiat-Shamir Simple Identification Protocol

Objective: Alice proves knowledge of s to Bob in t executions of a 3-pass protocol.
One-time setup :

- A trusted center T selects and publishes an RSA-like modulus $n/;= /; pq$. The primes p and q are kept secret.
- The prover Alice selects a secret s coprime to n, $1 \leq$ s $\leq$ (n - 1),
- Alice computes $v = s^2 \text{mod } n$
- Alice registers v with T as her public key.

Protocol messages. Each of t rounds has three messages as follows:

- $A \rightarrow B$: $x = r^2 mod\ n$ .....(1)
- $B \rightarrow A$: e $\epsilon \{0, 1\}$ .....(2)
- $A \rightarrow B$: $y = r * s^e mod\ n$ .....(3)

Then the protocol actions follow as shown further.

Interactive ZKP: Fiat-Shamir Simple Identification Protocol

Objective: Alice proves knowledge of s to Bob in 1 executions of a 3-pass protocol.
Protocol actions:

- A chooses a random (commitment) r, $1 \leq r \leq (n-1)$

Interactive ZKP: Fiat-Shamir Simple Identification Protocol

Objective: Alice proves knowledge of s to Bob in 1 executions of a 3-pass protocol.
Protocol actions:

- A chooses a random (commitment) r, $1 \leq r \leq (n-1)$
- A sends (the witness) $x = r^2 \ mod \ n$ to B.

Interactive ZKP: Fiat-Shamir Simple Identification Protocol

Objective: Alice proves knowledge of s to Bob in 1 executions of a 3-pass protocol.
Protocol actions:

- A chooses a random (commitment) r, $1 \leq r \leq (n-1)$
- A sends (the witness) $x = r^2 \ mod \ n$ to B.
- B randomly selects a (challenge) bit $e = 0$ or $e = 1$

Interactive ZKP: Fiat-Shamir Simple Identification Protocol

Objective: Alice proves knowledge of s to Bob in 1 executions of a 3-pass protocol.
Protocol actions:

- A chooses a random (commitment) r, $1 \leq r \leq (n-1)$
- A sends (the witness) $x = r^2 \ mod \ n$ to B.
- B randomly selects a (challenge) bit $e = 0$ or $e = 1$
- B sends e to A.

Interactive ZKP: Fiat-Shamir Simple Identification Protocol

Objective: Alice proves knowledge of s to Bob in 1 executions of a 3-pass protocol.
Protocol actions:

- A chooses a random (commitment) r, $1 \leq r \leq (n-1)$
- A sends (the witness) $x = r^2 \ mod \ n$ to B.
- B randomly selects a (challenge) bit $e = 0$ or $e = 1$
- B sends e to A.
- A computes y as follows: if (if e $= 0$), y $=$ r and (if e $= 1$) y $= r * s \ mod \ n$

Interactive ZKP: Fiat-Shamir Simple Identification Protocol

Objective: Alice proves knowledge of s to Bob in 1 executions of a 3-pass protocol.
Protocol actions:

- A chooses a random (commitment) r, $1 \leq r \leq (n-1)$
- A sends (the witness) $x = r^2 \ mod \ n$ to B.
- B randomly selects a (challenge) bit $e = 0$ or $e = 1$
- B sends e to A.
- A computes y as follows: if (if e = 0), y = r and (if e = 1) $y = r * s \ mod \ n$
- A sends the response y to B

Interactive ZKP: Fiat-Shamir Simple Identification Protocol

Objective: Alice proves knowledge of s to Bob in 1 executions of a 3-pass protocol.
Protocol actions:

- A chooses a random (commitment) r, $1 \leq r \leq (n-1)$
- A sends (the witness) $x = r^2 \ mod \ n$ to B.
- B randomly selects a (challenge) bit $e = 0$ or $e = 1$
- B sends e to A.
- A computes y as follows: if (if e = 0), y = r and (if e = 1) $y = r * s \ mod \ n$
- A sends the response y to B
- B rejects the proof if y = 0, and otherwise accepts upon verifying $y^2 \equiv x * v^e \ (mod \ n)$.

.

Interactive ZKP: Fiat-Shamir Simple Identification Protocol

Objective: Alice proves knowledge of s to Bob in 1 executions of a 3-pass protocol.
Protocol actions:

- A chooses a random (commitment) r, $1 \leq r \leq (n-1)$
- A sends (the witness) $x = r^2 \ mod \ n$ to B.
- B randomly selects a (challenge) bit $e = 0$ or $e = 1$
- B sends e to A.
- A computes y as follows: if (if e = 0), y = r and (if e = 1) $y = r * s \ mod \ n$
- A sends the response y to B
- B rejects the proof if y = 0, and otherwise accepts upon verifying $y^2 \equiv x * v^e \ (mod \ n)$.
  - This is so because, depending on e, $y^2 = x$ or $y^2 = x * v \bmod n$ since $v = s^2 \ mod \ n$. Note that checking for y = 0 precludes the case r = 0

.

- A trusted center T selects p=101 and q=103 (secret) and publishes an RSA-like modulus $n = 101 * 103 = 10403$.

Interactive ZKP: Fiat-Shamir Simple Identification Protocol Demo

- A trusted center T selects p=101 and q=103 (secret) and publishes an RSA-like modulus $n = 101 * 103 = 10403$.
- The prover Alice selects a secret s =23 coprime to n, $1 \leq s \leq (n - 1)$

Interactive ZKP: Fiat-Shamir Simple Identification Protocol Demo

- A trusted center T selects p=101 and q=103 (secret) and publishes an RSA-like modulus $n = 101 * 103 = 10403$.
- The prover Alice selects a secret s =23 coprime to n, $1 \leq s \leq (n - 1)$
- Alice computes $v = s^2 \bmod n$=$23^2 \bmod 10403$ and makes it public.

Interactive ZKP: Fiat-Shamir Simple Identification Protocol Demo

- A trusted center T selects p=101 and q=103 (secret) and publishes an RSA-like modulus $n = 101 * 103 = 10403$.
- The prover Alice selects a secret s =23 coprime to n, $1 \leq s \leq (n - 1)$
- Alice computes $v = s^2 \bmod n$=$23^2 \bmod 10403$ and makes it public.
- Assume that Alice i.e. A is the client and wishes to prove her identity to a bank i.e. B to check the funds in her bank account.

Interactive ZKP: Fiat-Shamir Simple Identification Protocol Demo

- A trusted center T selects p=101 and q=103 (secret) and publishes an RSA-like modulus $n = 101 * 103 = 10403$.
- The prover Alice selects a secret s =23 coprime to n, $1 \leq s \leq (n - 1)$
- Alice computes $v = s^2 \bmod n$=$23^2 \bmod 10403$ and makes it public.
- Assume that Alice i.e. A is the client and wishes to prove her identity to a bank i.e. B to check the funds in her bank account.
- Protocol actions as follows: iterated t times (sequentially and independently). B accepts the proof if all t rounds succeed.

Interactive ZKP: Fiat-Shamir Simple Identification Protocol Demo

- A trusted center T selects p=101 and q=103 (secret) and publishes an RSA-like modulus $n = 101 * 103 = 10403$.
- The prover Alice selects a secret s =23 coprime to n, $1 \leq s \leq (n - 1)$
- Alice computes $v = s^2 \bmod n$=$23^2 \bmod 10403$ and makes it public.
- Assume that Alice i.e. A is the client and wishes to prove her identity to a bank i.e. B to check the funds in her bank account.
- Protocol actions as follows: iterated t times (sequentially and independently). B accepts the proof if all t rounds succeed.

Interactive ZKP: Fiat-Shamir Simple Identification Protocol Demo

- A trusted center T selects p=101 and q=103 (secret) and publishes an RSA-like modulus $n = 101 * 103 = 10403$.
- The prover Alice selects a secret s =23 coprime to n, $1 \leq s \leq (n - 1)$
- Alice computes $v = s^2 \bmod n$=$23^2 \bmod 10403$ and makes it public.
- Assume that Alice i.e. A is the client and wishes to prove her identity to a bank i.e. B to check the funds in her bank account.
- Protocol actions as follows: iterated t times (sequentially and independently). B accepts the proof if all t rounds succeed.

p=101, q=103, n=10403	p=101, q=103, n=10403
secret s=23, r=21	secret s=23, r=21
$v=s^2 \bmod n = 23^2 \bmod 10403$	$v=s^2 \bmod n = 23^2 \bmod 10403$
$x = r^2 \bmod n$ i.e. $21^2 \bmod 10403$, sends x	r=21, $x = r^2 \bmod n$ i.e. $21^2 \bmod 10403$, sends x
e=0	e=1
Alice sends: y=r i.e. y=21	Alice sends $y=r*s \bmod n$ i.e. y=21*23 mod 10403
y != 0 so Bank does not reject the proof	y != 0 so Bank does not reject the proof
Bank accepts if : $y^2 \equiv x * v^e \bmod n$ i.e. $21^2 \equiv (21^2 \bmod 10403) * 1 \bmod 10403 \equiv 21^2 \bmod 10403$	Bank accepts if : $y^2 \equiv x * v^e \bmod n$ i.e. $y^2 \equiv x * v \bmod n$ i.e. $21^2*23^2 \bmod 10403 \equiv 21^2 \bmod 10403 * (23^2 \bmod 10403) \bmod 10403 \equiv 21^2*23^2 \bmod 10403$

Figure: Interactive ZKP: Fiat-Shamir's Protocol

Non-Interactive Zero Knowledge Proofs

Types of ZKP: Non-Interactive ZKPs

Non-Interactive zero-knowledge proofs:

- a general structure consists of just a single action between participants Prover and Verifier

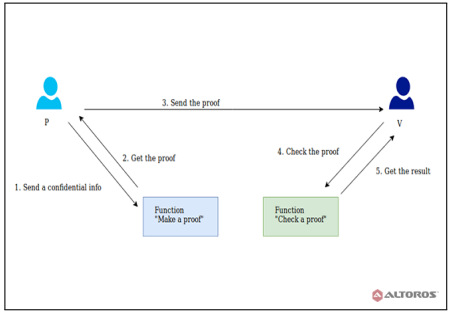


Figure: Non-Interactive ZKP

Figure Src: https://www.altoros.com/blog/zero-knowledge-proof-improving-privacy-for-a-blockchain/

Types of ZKP: Non-Interactive ZKPs

Non-Interactive zero-knowledge proofs:

- a general structure consists of just a single action between participants Prover and Verifier
- this single action is a witness.

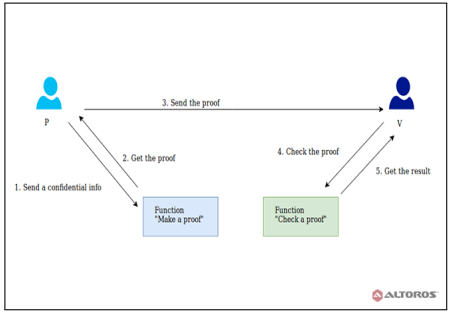


Figure: Non-Interactive ZKP

Figure Src: https://www.altoros.com/blog/zero-knowledge-proof-improving-privacy-for-a-blockchain/

Types of ZKP: Non-Interactive ZKPs

Non-Interactive zero-knowledge proofs:

- a general structure consists of just a single action between participants Prover and Verifier
- this single action is a witness.
- As shown, P passes the secret information as an argument to a special function—"make a proof"

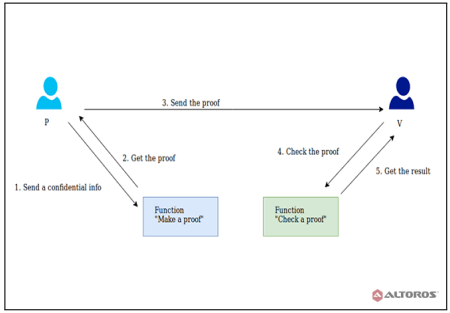


Figure: Non-Interactive ZKP

Figure Src: https://www.altoros.com/blog/zero-knowledge-proof-improving-privacy-for-a-blockchain/

Types of ZKP: Non-Interactive ZKPs

Non-Interactive zero-knowledge proofs:

- a general structure consists of just a single action between participants Prover and Verifier
- this single action is a witness.
- As shown, P passes the secret information as an argument to a special function—"make a proof"
- The output is some value of "proof."

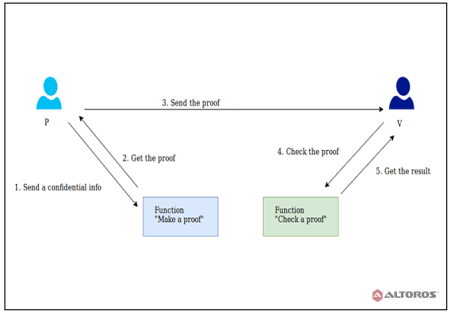


Figure: Non-Interactive ZKP

Figure Src: https://www.altoros.com/blog/zero-knowledge-proof-improving-privacy-for-a-blockchain/

Types of ZKP: Non-Interactive ZKPs

Non-Interactive zero-knowledge proofs:

- a general structure consists of just a single action between participants Prover and Verifier
- this single action is a witness.
- As shown, P passes the secret information as an argument to a special function—"make a proof"
- The output is some value of "proof."
- Prover then sends "proof" to Verifier.

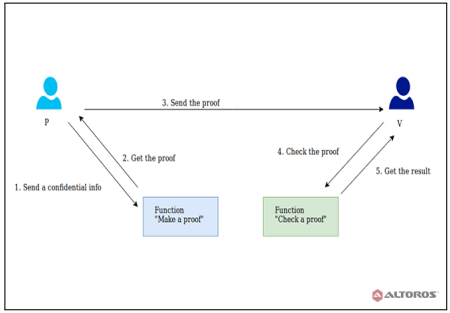


Figure: Non-Interactive ZKP

Figure Src: https://www.altoros.com/blog/zero-knowledge-proof-improving-privacy-for-a-blockchain/

Types of ZKP: Non-Interactive ZKPs

Non-Interactive zero-knowledge proofs:

- a general structure consists of just a single action between participants Prover and Verifier
- this single action is a witness.
- As shown, P passes the secret information as an argument to a special function—"make a proof"
- The output is some value of "proof."
- Prover then sends "proof" to Verifier.
- V then checks if P knows the secret information using the "proof" and another special function—"check a proof."

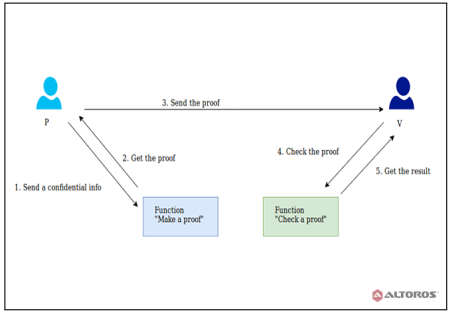


Figure: Non-Interactive ZKP

Figure Src: https://www.altoros.com/blog/zero-knowledge-proof-improving-privacy-for-a-blockchain/

Non-Interactive ZKPs: General Applications

Non-Interactive ZKPs: Three Key Applications

- Ethereum 2.0 and zk-rollups:

Non-Interactive ZKPs: Three Key Applications

- Ethereum 2.0 and zk-rollups:
  - designed to increase transaction speed and reduce fees by moving computation and state-storage off-chain.

Non-Interactive ZKPs: Three Key Applications

- Ethereum 2.0 and zk-rollups:
  - designed to increase transaction speed and reduce fees by moving computation and state-storage off-chain.
  - makes it possible to process thousands of transactions in a batch but only post minimal summary data to Mainnet

Non-Interactive ZKPs: Three Key Applications

- Ethereum 2.0 and zk-rollups:
  - designed to increase transaction speed and reduce fees by moving computation and state-storage off-chain.
  - makes it possible to process thousands of transactions in a batch but only post minimal summary data to Mainnet
- User authentication

Non-Interactive ZKPs: Three Key Applications

- Ethereum 2.0 and zk-rollups:
  - designed to increase transaction speed and reduce fees by moving computation and state-storage off-chain.
  - makes it possible to process thousands of transactions in a batch but only post minimal summary data to Mainnet
- User authentication
  - facilitates transmitting sensitive information like authentication information with better security by building a secure channel for users to use their information without revealing it

Non-Interactive ZKPs: Three Key Applications

- Ethereum 2.0 and zk-rollups:
  - designed to increase transaction speed and reduce fees by moving computation and state-storage off-chain.
  - makes it possible to process thousands of transactions in a batch but only post minimal summary data to Mainnet
- User authentication
  - facilitates transmitting sensitive information like authentication information with better security by building a secure channel for users to use their information without revealing it
  - e.g. if a two-factor authentication (2FA) system requires both Aadhar and PAN-Card number, a ZKP algorithm can take certain segments of the two and link them together
    item to determine the probability that the individual is who they claim to be, while keeping their main information hidden.

Non-Interactive ZKPs: Three Key Applications

- Ethereum 2.0 and zk-rollups:
  - designed to increase transaction speed and reduce fees by moving computation and state-storage off-chain.
  - makes it possible to process thousands of transactions in a batch but only post minimal summary data to Mainnet
- User authentication
  - facilitates transmitting sensitive information like authentication information with better security by building a secure channel for users to use their information without revealing it
  - e.g. if a two-factor authentication (2FA) system requires both Aadhar and PAN-Card number, a ZKP algorithm can take certain segments of the two and link them together
    item to determine the probability that the individual is who they claim to be, while keeping their main information hidden.
- Data storage protection

Non-Interactive ZKPs: Three Key Applications

- Ethereum 2.0 and zk-rollups:
  - designed to increase transaction speed and reduce fees by moving computation and state-storage off-chain.
  - makes it possible to process thousands of transactions in a batch but only post minimal summary data to Mainnet
- User authentication
  - facilitates transmitting sensitive information like authentication information with better security by building a secure channel for users to use their information without revealing it
  - e.g. if a two-factor authentication (2FA) system requires both Aadhar and PAN-Card number, a ZKP algorithm can take certain segments of the two and link them together
    item to determine the probability that the individual is who they claim to be, while keeping their main information hidden.
- Data storage protection
  - advanced encryption technology that ensures no one can access one's data as ZKP validates it without sharing it with a third party.

Non-Interactive ZKPs: Three Key Applications

- Ethereum 2.0 and zk-rollups:
  - designed to increase transaction speed and reduce fees by moving computation and state-storage off-chain.
  - makes it possible to process thousands of transactions in a batch but only post minimal summary data to Mainnet
- User authentication
  - facilitates transmitting sensitive information like authentication information with better security by building a secure channel for users to use their information without revealing it
  - e.g. if a two-factor authentication (2FA) system requires both Aadhar and PAN-Card number, a ZKP algorithm can take certain segments of the two and link them together
    item to determine the probability that the individual is who they claim to be, while keeping their main information hidden.
- Data storage protection
  - advanced encryption technology that ensures no one can access one's data as ZKP validates it without sharing it with a third party.
  - since zero-knowledge proof has the potential to encrypt data in portions, it enables providing access to a particular user while limiting access for others

Blank

Blank

Blank

Blank

Blank