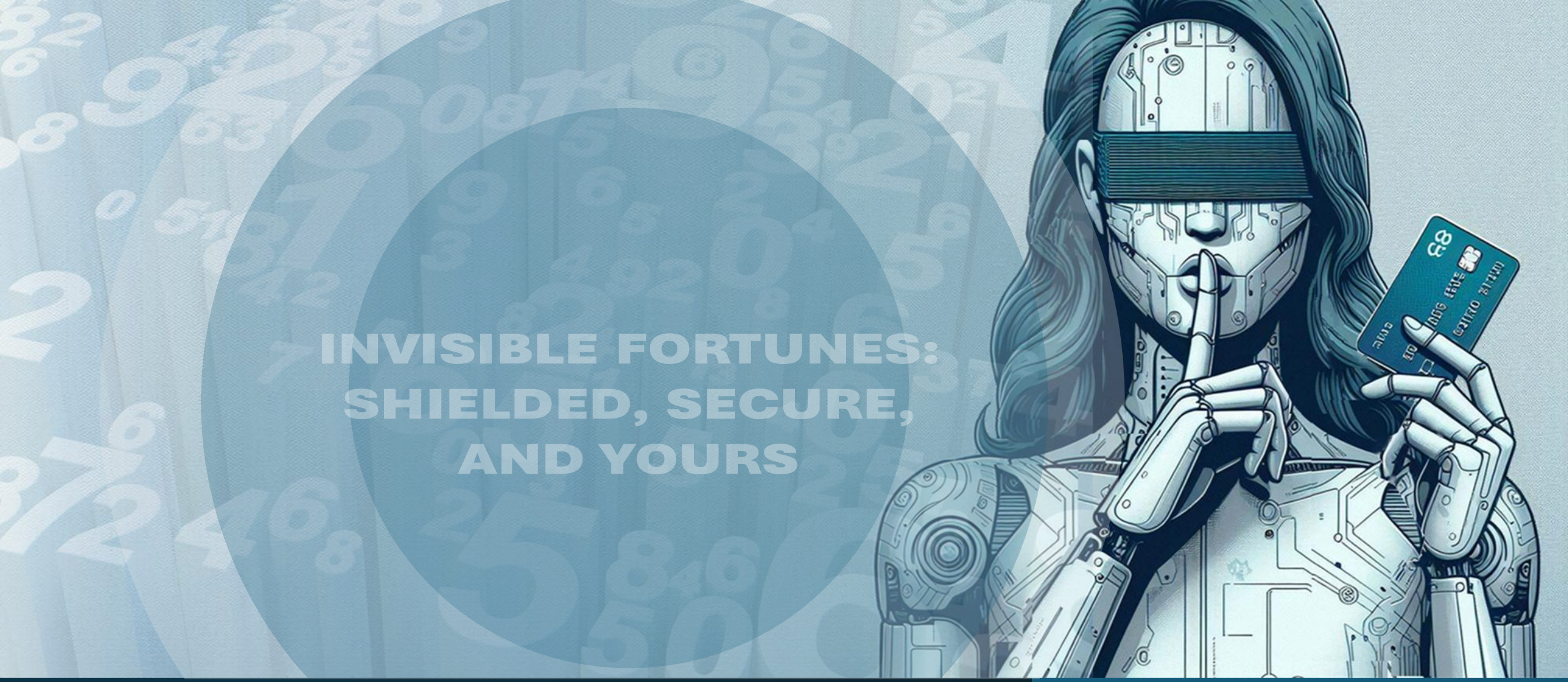


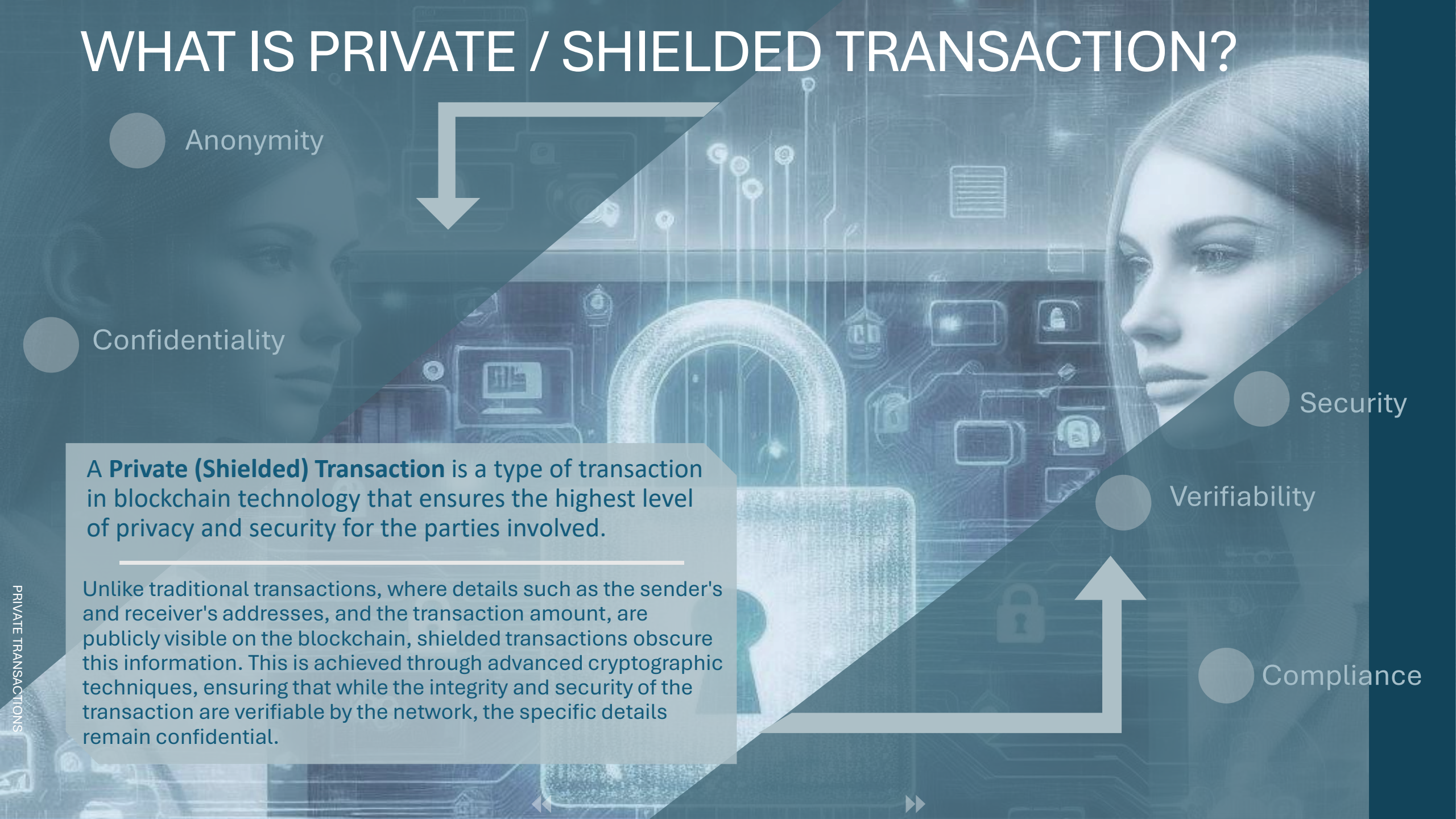
PRIVATE TRANSACTIONS

Securing Transactions: Definitions, Architecture, and Use Cases

REALM

2025-02-15

WHAT IS PRIVATE / SHIELDED TRANSACTION?



- Anonymity
- Confidentiality
- Security
- Verifiability
- Compliance

A **Private (Shielded) Transaction** is a type of transaction in blockchain technology that ensures the highest level of privacy and security for the parties involved.

Unlike traditional transactions, where details such as the sender's and receiver's addresses, and the transaction amount, are publicly visible on the blockchain, shielded transactions obscure this information. This is achieved through advanced cryptographic techniques, ensuring that while the integrity and security of the transaction are verifiable by the network, the specific details remain confidential.

WHAT IS THE PROBLEM?

Blockchain technology and traditional banking systems both serve as mechanisms to facilitate financial (and not financial) transactions, but they operate on fundamentally different principles regarding privacy, transparency, and trust. The issue of balancing privacy with transparency becomes particularly acute in the context of blockchain systems due to their inherent design and operation mechanisms.

BLOCKCHAIN

In the blockchain paradigm, transactions are processed by nodes through consensus and stored on a distributed ledger. All nodes have access to addresses, transaction attributes (amounts, etc.)

Transparency: One of the foundational features of blockchain technology is its transparency. Transactions on a public blockchain are visible to all participants, ensuring a high level of transparency and enabling trust among users.

Privacy Concerns: This transparency, however, raises significant privacy concerns. In a blockchain, the details of a transaction, including the sender's and receiver's addresses and the transaction amount, are publicly visible. This level of openness can compromise the privacy of individuals and organizations, making sensitive financial information accessible to anyone who knows how to look for it.

Need for Privacy Solutions: The actual challenge for blockchain systems is to integrate privacy-preserving mechanisms, such as shielded transactions, that protect user data while maintaining the integrity and security of the network.

BANKS AND FINANCE INSTITUTIONS

Bank transactions using financial messaging systems like SWIFT, SEPA, or Fedwire are centralized and rely on intermediaries for processing. These transactions are communicated and settled through a network of banks, with the sending and receiving banks coordinating to verify and complete the transaction. The process is regulated and not public, offering privacy but lacking the transparency and immutability of blockchain transactions.

Centralized Privacy Control: Traditional banking systems operate on a centralized model where the bank has control over all transaction data. Privacy is maintained through legal and regulatory frameworks, and customers' financial information is not publicly accessible.

Transparency Issues: While banking systems offer more privacy compared to public blockchains, they lack the same level of transparency. The centralized control of information can lead to issues of trust, where customers have limited visibility into the bank's operations and the movement of their own funds.

DATA PRIVACY IN VARIOUS SECTORS

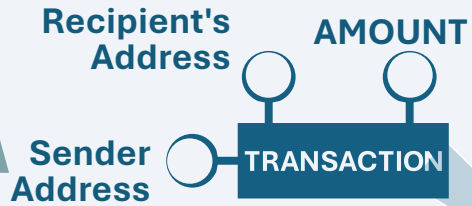
The issue of balancing privacy with transparency is not limited to financial transactions. It extends to any sector where sensitive information is handled, including healthcare, government records, and personal data on the internet.

PRIVACY FOR DECENTRALIZED PROCESSING

When transactions are processed in the network, nodes check the inputs and outputs of the transaction (sender and receiver addresses), the sufficiency of the amounts, and "vote" to include the transaction in a new block (consensus mechanism), which, once formed, becomes part of a common database – a distributed ledger. To make such a decision, all nodes participating in the consensus have access to the basic parameters of the transaction

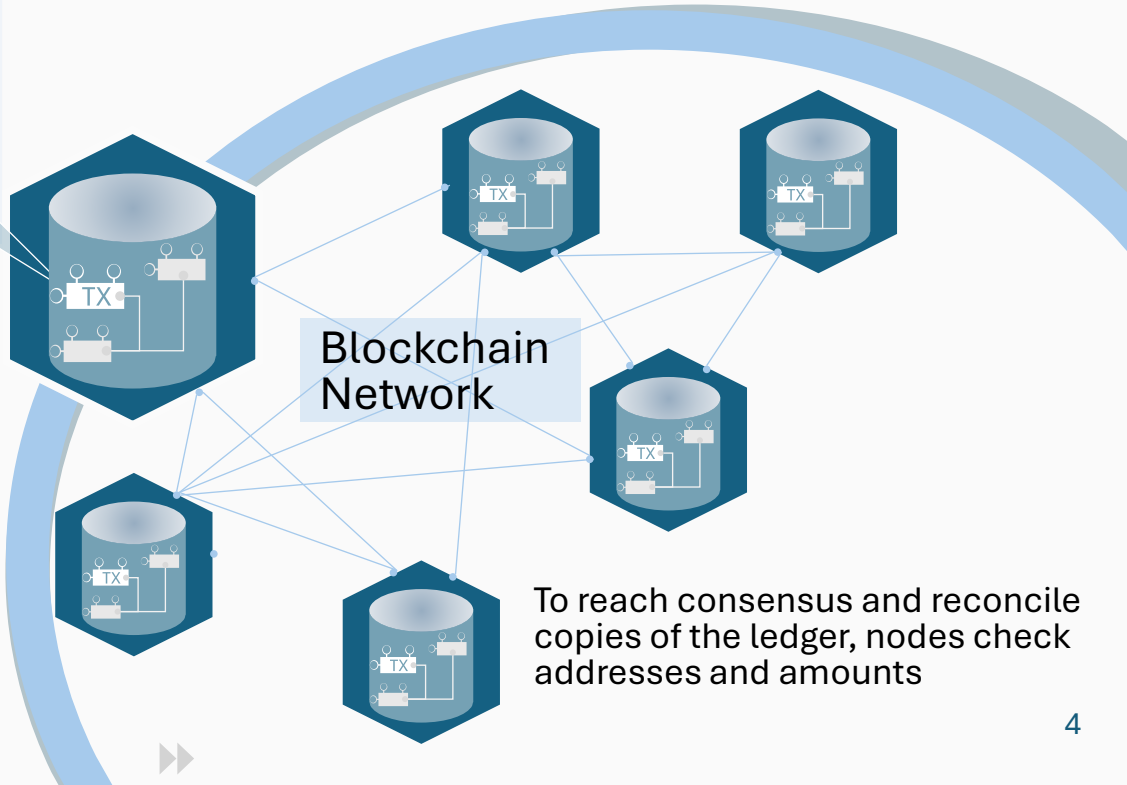
Hiding Addresses: Concealing the addresses of transaction participants protects their identities, making it difficult for third parties to trace transactions back to individuals or entities.

Hiding Amounts: Concealing transaction amounts helps in safeguarding financial details from public scrutiny, preventing potential targeting based on transaction values.



Implications of Third-Party Access:

- If third parties can access addresses, it could lead to privacy breaches, revealing the transaction history and balances of users
- Access to transaction amounts without context may not directly compromise identities but could reveal patterns or the magnitude of transactions, potentially raising questions about the source and use of funds.



To reach consensus and reconcile copies of the ledger, nodes check addresses and amounts

TECHNOLOGY CHALLENGE

Blockchain technology offers a revolutionary approach to digital transactions with its features of decentralization, transparency, and immutability. However, its adoption by financial institutions faces significant challenges due to the inherent lack of confidentiality, which conflicts with banking secrecy laws and privacy regulations that govern the financial industry.

Transparency vs. Banking Secrecy

Inherent Transparency: Blockchain's design ensures that all transactions are publicly recorded on the distributed ledger. This transparency, while beneficial for trust and security, stands in contrast to the principle of banking secrecy, which mandates that customer financial information must be kept confidential and disclosed only under strictly regulated conditions.

Banking Secrecy Laws: Financial institutions are bound by laws and regulations that require them to protect the confidentiality of their clients' information. Laws such as the Bank Secrecy Act (BSA) in the United States, and similar regulations globally, impose strict penalties for the unauthorized disclosure of customer financial information.

Privacy Regulations

Global Privacy Laws: The adoption of stringent privacy laws like the General Data Protection Regulation (GDPR) in the European Union, and others around the world, further complicates the integration of blockchain technology into financial systems. These laws grant individuals significant rights over their personal data, including the right to privacy, data minimization, and the right to be forgotten, which are challenging to reconcile with the permanent and transparent nature of blockchain.

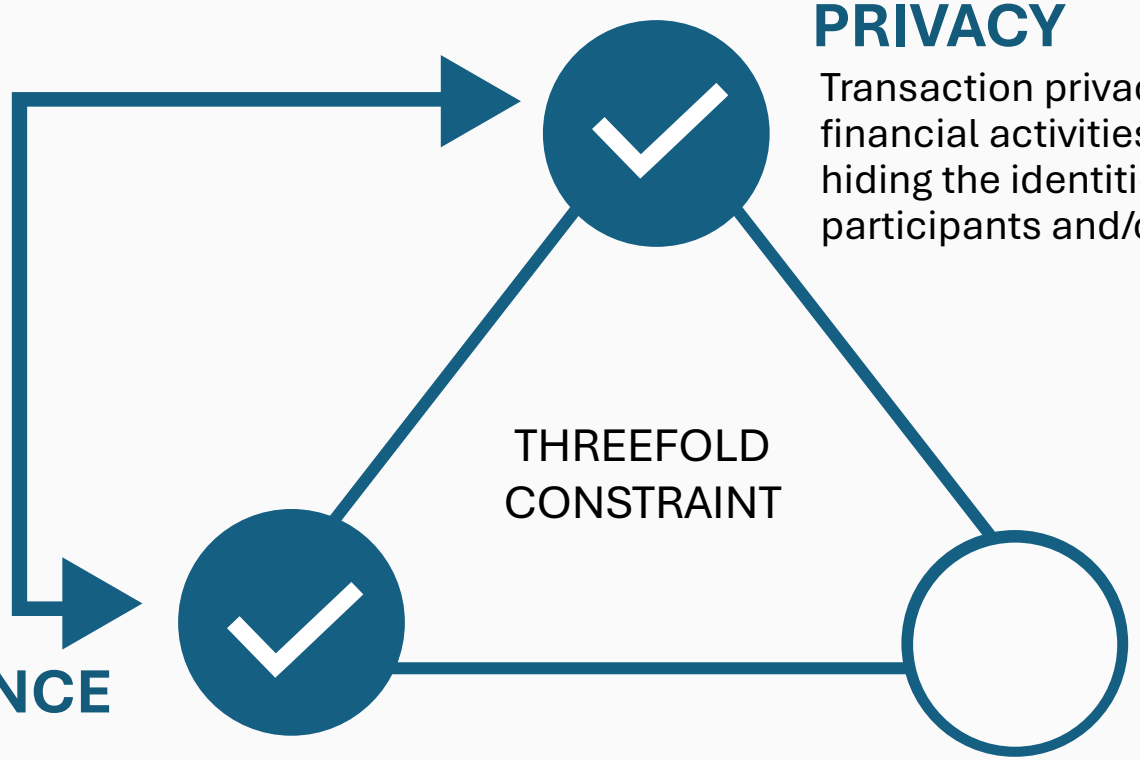
Challenge of Anonymity: While blockchain can offer pseudonymity using addresses that do not directly reveal the identity of users, the possibility of linking addresses to real-world identities through transaction pattern analysis and other means poses a risk to privacy. This risk is exacerbated in a financial context where transactions can reveal sensitive information about individuals and entities.

The challenge of integrating blockchain technology into financial institutions due to confidentiality concerns is significant but not insurmountable. By leveraging advanced cryptographic techniques and seeking regulatory clarity and accommodation, it is possible to bridge the gap between the transformative potential of blockchain and the critical requirements of banking secrecy and privacy laws. This delicate balance is essential for the future of blockchain in the financial sector, promising enhanced security, efficiency, and transparency in financial transactions.

THE PRIVACY-COMPLIANCE-PERFORMANCE LEMMA

In the evolving landscape of digital transactions, balancing privacy, regulatory compliance, and system performance presents a complex challenge. As we strive to protect user data and ensure transaction confidentiality, we must also navigate the regulatory frameworks designed to prevent fraud and money laundering, all while maintaining high system efficiency and speed.

DGT's solution for private transactions (as opposed to regular transactions) primarily addresses PRIVACY and COMPLIANCE, accepting some delays in the execution of special private transactions



PRIVACY

Transaction privacy involves concealing the details of financial activities from unauthorized parties, including hiding the identities (addresses) of transaction participants and/or the amounts being transferred.

COMPLIANCE

- Compliance with regulations like KYC (Know Your Customer) and AML (Anti-Money Laundering) is crucial for preventing fraud, terrorism financing, and other illicit activities.
- Implementing robust compliance measures often requires some level of access to transaction details, potentially compromising privacy.

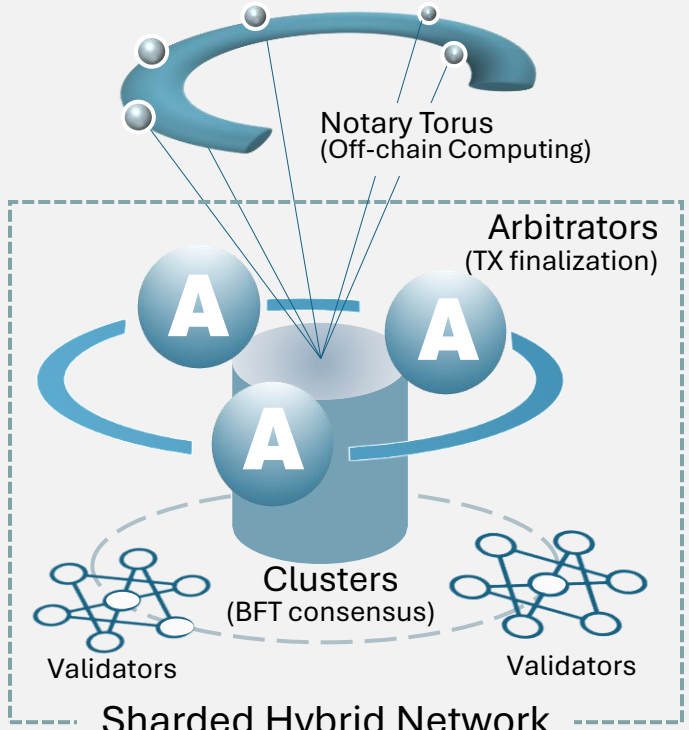
PERFORMANCE

- Advanced privacy-enhancing technologies (e.g., zk-SNARKs, Homomorphic Encryption) and compliance mechanisms can introduce computational complexity, leading to increased transaction processing times and reduced system throughput.
- Finding the optimal trade-off between implementing strong privacy features and maintaining high-performance transaction processing.

LEVERAGING DGT'S TECHNOLOGICAL ADVANCEMENTS

DGT's innovative architecture, which includes a hybrid network structure, two-level consensus mechanism, an off-chain computation layer, and a flexible transaction processing system, paves the way for creating specialized channels and addressing systems. These features collectively facilitate a more efficient, secure, and versatile blockchain ecosystem.

Special nodes (Notaries) anchored in the regular DGT (on-chain) network, but designed to store and process confidential information, form an additional layer of security: off-chain computing



Hybrid Network Architecture: DGT operates on a hybrid network model that seamlessly combines public and private blockchain functionalities. This unique structure enables DGT to offer the transparency and security of public blockchains while also providing the privacy and efficiency of private blockchains, making it ideal for a wide range of applications.

Two-Level Consensus Mechanism: The DGT blockchain utilizes a novel two-level consensus mechanism that divides the consensus process into two distinct layers. The first layer employs a faster, more scalable consensus among clusters (e.g., using Byzantine Fault Tolerance algorithms), while the second layer integrates a Proof of Stake (PoS) consensus among selected nodes. This dual approach ensures high transaction throughput and enhanced security.

Off-Chain Computation Layer (Notary Torus): DGT introduces the Notary Torus, an off-chain computation layer, to facilitate complex computations without burdening the main blockchain. This layer allows for the creation of special channels for private transactions, data processing, and smart contract execution, significantly reducing the load on the blockchain and improving transaction speeds.

Pluggable Transaction Processors (Transaction Families): The platform supports pluggable transaction processors, enabling the development and integration of custom transaction families. This modularity allows for the easy expansion of the blockchain's capabilities and the creation of tailored solutions for specific use cases.

Flexible Addressing System: DGT features a sophisticated and flexible addressing system that supports a wide range of address types, including hierarchical and alias-based addresses. This system allows for more intuitive and efficient management of digital assets, smart contracts, and user identities, facilitating a user-friendly blockchain experience.

TECHNOLOGIES BEHIND SHIELDED TRANSACTIONS

Private technologies use a wide range of cryptographic techniques that represent separate classes of solutions. The most mature solutions use several technological approaches together, which collectively enhance privacy, security, and scalability, enabling the creation of confidential and efficient digital transactions.

Zero-Knowledge Proofs (ZKP). ZKPs are a cryptographic method that enables one party (the prover) to prove to another party (the verifier) that a certain statement is true, without conveying any information apart from the truth of the statement itself. This technology underpins the privacy features in many blockchain applications by allowing the verification of transactions without revealing the transaction's details. Examples of protocols include: zk-SNARKs, zk-STARKs, Bulletproofs

Homomorphic Encryption (HE). HE is a form of encryption that allows computation on ciphertexts, producing an encrypted result that, when decrypted, reveals the outcome of operations as if they had been performed on the plaintext. This property is instrumental in processing sensitive data while preserving privacy and security. In the context of blockchain, HE can be used for privacy-preserving smart contracts, where computations can be performed on encrypted inputs, ensuring data confidentiality while still allowing for verifiable outcomes.

Pedersen Commitments (PC). Pedersen Commitments enable a user to commit to a selected value while keeping it hidden, with the ability to reveal and verify the committed value later without compromising its security. This technique is crucial for creating confidential transactions in blockchain systems. PC-technology is widely used in blockchain protocols to ensure the integrity of transaction amounts while preserving their privacy, as seen in Zcash and other cryptocurrencies focusing on enhanced privacy features.

Trusted Execution Environments (TEE). TEEs are secure areas within processors that provide a level of assurance regarding the integrity and confidentiality of the code and data inside. They ensure that the code executed within the TEE cannot be tampered with and that the data cannot be accessed by unauthorized processes. In blockchain, TEEs can secure sensitive transaction data and smart contract execution, enabling confidential computations and storage that complement the privacy guarantees of shielded transactions.

Secure Multi-Party Computation (SMPC). SMPC is a cryptographic technique that enables parties to jointly compute a function over their inputs while keeping those inputs private. It ensures that participants can collaborate on computations without revealing their private data to each other. Protocols like SPDZ (Secure Protocol for Distributed Zero-Knowledge) and Garbled Circuits are examples of SMPC being used to facilitate privacy-preserving computations, allowing for collaborative data analysis or financial transactions without compromising the privacy of the parties involved.

ZKP: REVOLUTIONIZING PRIVACY IN COMPUTING

Zero-Knowledge Proofs (ZKP) are a cryptographic method that allows one party (the prover) to prove to another (the verifier) that a certain statement is true, without revealing any information beyond the validity of the statement itself.

The concept of ZKP was first introduced in the 1980s by Shafi Goldwasser, Silvio Micali, and Charles Rackoff. Marked a paradigm shift in cryptography, enabling secure verification of information without disclosure.

Implication: ZKP enables secure authentication, privacy-preserving protocols, and confidential transactions

ROLES

Prover: The entity that wants to prove the knowledge of a secret without revealing it.

Verifier: The entity that verifies the truth of the prover's claim without learning anything about the secret itself.

TYPES

Interactive ZKP: Requires a back-and-forth communication between the prover and the verifier.

Non-Interactive ZKP (NIZK): The prover can make a claim with a single message, without any further interaction with the verifier.

APPLICATIONS

For Range: Proving a number lies within a certain range without revealing the number itself

For Membership: Demonstrating membership in a set without disclosing the element.

Mathematical Foundations:

Interactive proof systems are a model in which the verifier and the verifier exchange messages to verify the truth of a statement Knowledge complexity is a measure of how much information is revealed to the demonstrator in the process of proof

Zero complexity of knowledge is a property where the demonstrator does not reveal any information other than the fact that the statement is true

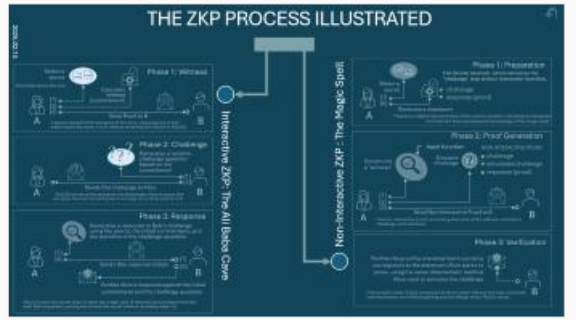
Complex problems are problems that are difficult to solve but easy to verify.

Cryptographic primitives are basic algorithms or protocols that are used to build more complex cryptographic systems.

Polynomial calculations are calculations that can be performed in polynomial time relative to the size of the input data.

Algebraic geometry is a branch of mathematics that studies geometric objects given by algebraic equations.

Code theory is a branch of mathematics that studies the ways in which information is encoded, transmitted, and recovered using corrective codes. For example,



EXPLORING BASIC ZKP ALGORITHMS AND CONCEPTS

Among the myriad of ZKP algorithms, Pedersen Commitment and Bulletproofs are particularly noteworthy for their contributions to the field. Understanding their underlying principles, such as group theory and the Schnorr protocol, is key to appreciating their potential and limitations.

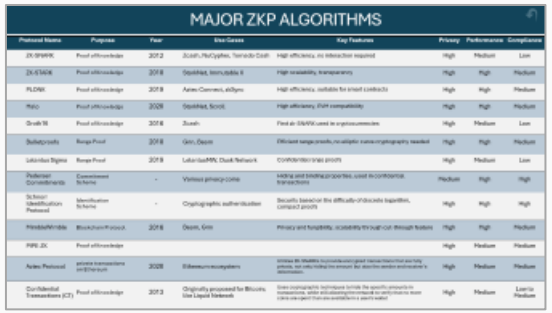
Group Theory in Cryptography: Cryptographic protocols, including ZKPs, often rely on mathematical structures known as groups. A group is a set equipped with a single binary operation that combines any two elements to form a third element, satisfying certain conditions (closure, associativity, identity element, and invertibility).

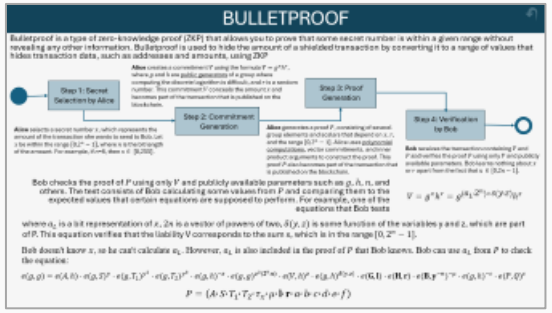
Group Generator: Within a group, a generator is an element from which every other element of the group can be derived through repeated application of the group operation. In cryptographic terms, group generators are foundational for constructing cyclic groups used in protocols like Pedersen Commitment and Bulletproofs.

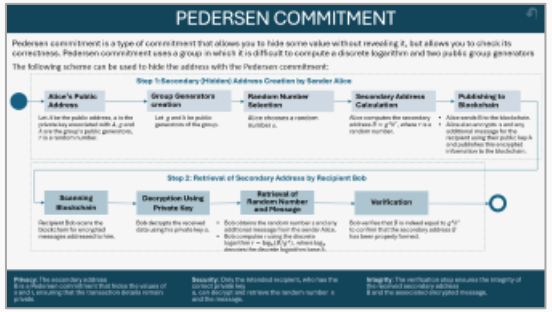
Public group generators are numbers that are members of the group in which the calculations are performed. They are known to all participants in the protocol and can be selected from standard parameters or randomly generated.

Proof as a Set of Group Elements: In the context of ZKP, a proof often consists of a set of group elements that demonstrate the validity of a statement without revealing any additional information. These elements are carefully chosen to maintain the zero-knowledge property.

Schnorr Protocol: The Schnorr protocol is a simple, yet powerful identification scheme based on the discrete logarithm problem. It is a fundamental component of many ZKP systems, providing a way to prove knowledge of a secret (like a private key) corresponding to a public value without revealing the secret itself.







HARNESSING HOMOMORPHIC ENCRYPTION

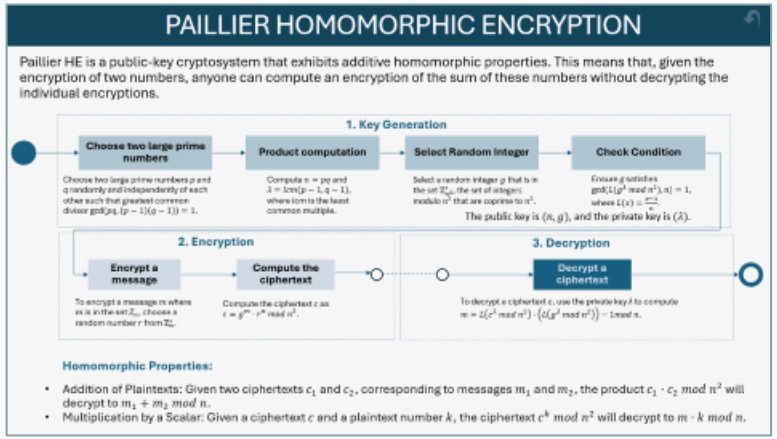
Homomorphic Encryption (HE) is a form of encryption that allows for computations to be performed on encrypted data without needing to decrypt it first. This groundbreaking technology enables the secure processing of sensitive information while fully preserving privacy, opening new avenues for regulatory compliance and secure data analytics.

Homomorphic Encryption allows data to be encrypted in such a way that specific types of operations performed on the ciphertext yield the same result as if the operations had been performed on the plaintext. If Enc represents the encryption function, $\oplus$ and $\otimes$ represent homomorphic addition and multiplication, then for any plain texts x and y:

$$Enc(x) \oplus Enc(y) = Enc(x + y)$$
$$Enc(x) \otimes Enc(y) = Enc(x \times y)$$

This property ensures that the structure of the data is preserved through the encryption and computation process, enabling meaningful operations on encrypted data.

Compliance with Data in Crypto Vaults: HE can store sensitive data in encrypted form, known as crypto vaults, ensuring that the data remains confidential and tamper-proof. Financial institutions can perform encrypted searches and calculations on client data to comply with anti-money laundering regulations without exposing individual client details. A prover can perform computations on encrypted data and generate a proof of computation without accessing the plaintext.



To verify the correctness of computation c on encrypted inputs $Enc(x)$ and $Enc(y)$, one can check:

> If $c = Enc(x) \oplus Enc(y)$, then upon decryption, *Dec*(*c*) should equal *x*+*y*.

DGT-ZK PROTOCOL

To implement shielded transactions, the DGT-ZK protocol is used, which uses the existing features of the DGT architecture with an emphasis on privacy and compliance. Such secure transactions are implemented by a separate transaction family (a dedicated processor) with the participation of an off-chain layer (Notaries) and coexist with regular open transactions that have high performance.

APPROACH: The DGT-ZK protocol advances the **MimbleWimble framework** by incorporating an explicit off-chain layer of Notaries and auxiliary Homomorphic Encryption (HE).

COMPLIANCE: This integration ensures KYC/AML compliance is upheld without sacrificing the privacy of transactions.

PARALLEL ADDRESSING AND ANONYMIZATION: Shielded transactions operate within an auxiliary address space that links to DGT's main transactional family, DEC-Processing. Pedersen Commitments anonymize these addresses, while anchored notaries on the mainnet maintain communication between them. The dual approach of Account-based and UTXO models is harmoniously unified.

PRIVACY OF AMOUNTS: Transaction amounts are concealed using Bulletproofs, further strengthening transaction privacy.

OFF-CHAIN CRYPTO VAULT: Encrypted transaction data is stored in a secure off-chain vault, inaccessible to notaries. Multi-Party Computing (MPC) and Private Set Intersection mechanisms allow for the identification of transactions related to blacklisted or sanctioned entities.

TRUSTED EXECUTED ENVIRONMENT: Enhanced with TEE and Selective Disclosure options, plus ML algorithms for anomaly detection.

CANCELLATION WINDOW MECHANISM: A "Cancellation Window" provides a controlled delay for transactions to ensure regulatory compliance and user protection. This window allows for additional verification and the opportunity to halt transactions deemed high-risk.

PROTOCOL PARTICIPANTS:

- **Users**: Owners of the transactions.
- **Notaries**: Nodes responsible for recording and verifying transactions.
- **Validator Nodes**: Confirm the validity of transactions.
- **Authority**: Ensures KYC/AML requirements are met.
- **Third-Party Network Participants**: Other stakeholders in the blockchain network.

SECURITY ASSUMPTIONS: The off-chain notary layer is expected to maintain high security, with potential for additional consensus mechanisms to reduce reliance on notaries (currently synchronized by RAFT).

Step 1: Initialization

Users generate key pairs and create public and hidden addresses for transactions.
Notaries record the relationship between public and hidden addresses by storing information in a secure vault.

Step 2: Prepare the Transaction

Users initiate a transaction by applying AI to analyze the context of the transaction and create vector representations of the data using FastText or similar libraries.
Transaction data is encrypted using HE to ensure confidentiality.

Step 3: Transaction Holding and

The Analysis transaction is placed in the hold status (cancellation window), allowing time for analysis.
Notaries, together with validator nodes, use SMPC to analyze encrypted transaction data for fraud without disclosing the original information.
If necessary, the regulator can request additional information using the Selective Disclosure and TEE mechanisms.

Step 4: Transaction Verification and Confirmation

After analysis, validator nodes confirm the validity of the transaction, and it is executed if no signs of fraud are found.
If suspicious activity is detected, the transaction may be canceled or frozen for further investigation.

Step 5: Final Processing

Users are notified of the status of the transaction.
The regulator and notaries update transaction records to comply with KYC/AML requirements, taking action in accordance with the law if necessary.

THE DGT-ZK PROTOCOL WORKFLOW

Sender

Generates a pair of public-private keys (PKA, SK_A).

Pedersen Commitments for transaction anonymity: $Addr_{secondary} = g^s h^r$, where s and r are secret numbers, and g, h are public group generators.

Keys Generation

Creates a secondary address

Chooses a transaction amount x

Encrypts transaction details

Encrypts transaction details using **Homomorphic Encryption (HE)** to allow computations on encrypted data: $Enc(x, PK_B)$.

Submits the transaction

including $Addr_{secondary}$, Bulletproof, and encrypted data, to the blockchain.

Recipient

Keys Generation

Generates his pair of public-private keys (PK_B, SK_B).

Decrypts the transaction details using his private key SK_B

Decrypts the transaction

Computes the discrete logarithm

Computes the discrete logarithm to retrieve s and verifies the transaction: $r = \log_h (Addr_{secondary}/g^s)$.

Validators

Check the validity of the Bulletproof

to ensure the transaction amount is within the allowed range.

Verify $Addr_{secondary}$

conforms to the DGT DEC-Processing family using the public keys and commitments.

Anchor TX

Notaries

Anchor the secondary address

Anchor the secondary address $Addr_{secondary}$ and encrypted data on the main-net

Utilize the **Cancellation Window** mechanism to hold the transaction temporarily, allowing time for additional checks or regulatory compliance.

Utilize the Cancellation Window

ML/KYC/AML Procedures

Optionally uses Selective Disclosure and TEE for secure and compliant data access.

Unfreeze TX

Regulator

Requests data disclosure for compliance checks

Utilize Multi-Party Computing (MPC)

Utilizes Multi-Party Computing (MPC) to verify transactions against blacklists or sanctions lists without decrypting the transaction data.

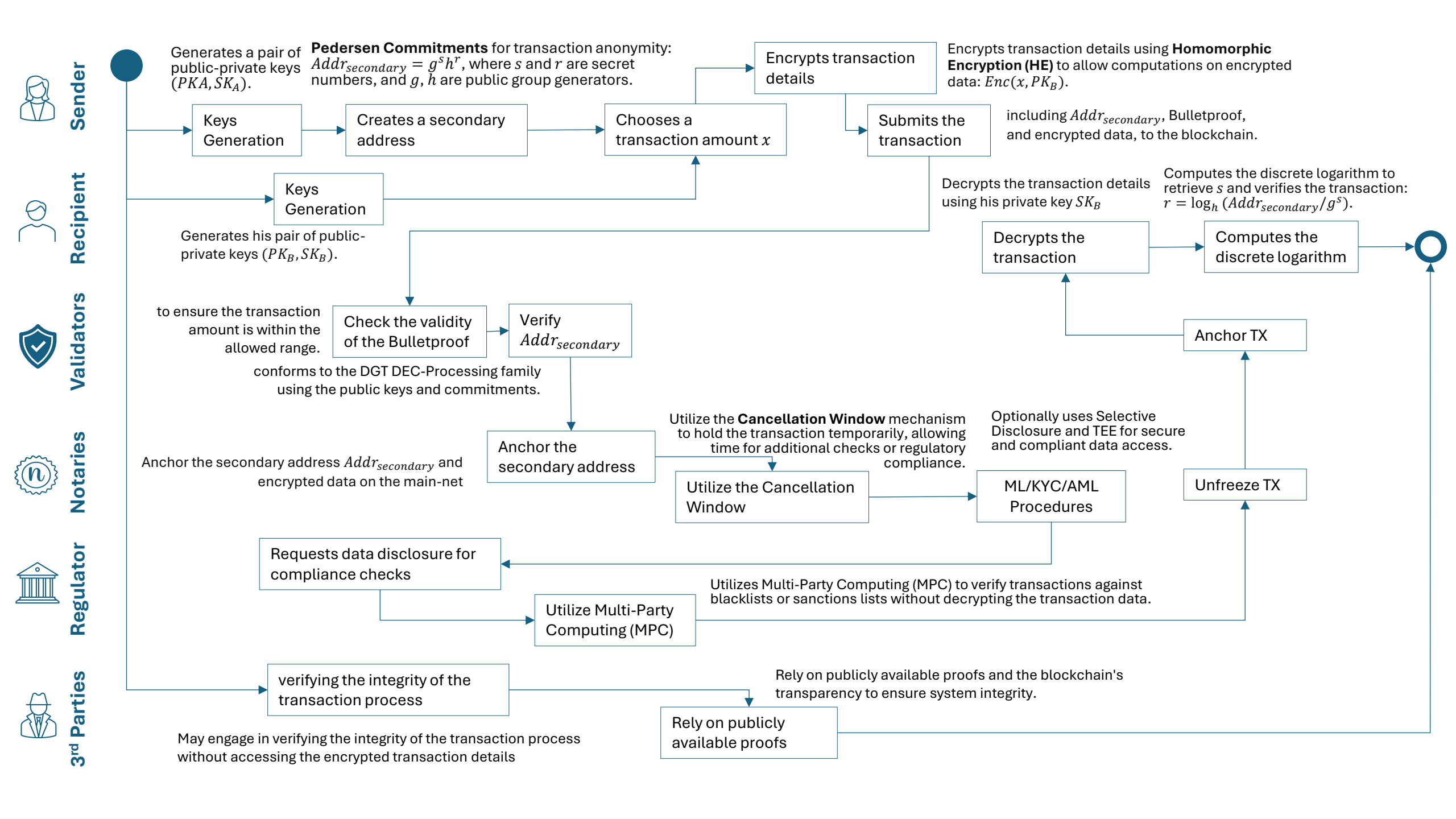
3rd Parties

verifying the integrity of the transaction process

May engage in verifying the integrity of the transaction process without accessing the encrypted transaction details

Rely on publicly available proofs

Rely on publicly available proofs and the blockchain's transparency to ensure system integrity.


Sender
Recipient
Validators
Notaries
Regulator
3rd Parties
Generates a pair of public-private keys (PKA, SK_A).
Pedersen Commitments for transaction anonymity: $Addr_{secondary} = g^s h^r$, where s and r are secret numbers, and g, h are public group generators.
Keys Generation
Creates a secondary address
Chooses a transaction amount x
Encrypts transaction details
Encrypts transaction details using Homomorphic Encryption (HE) to allow computations on encrypted data: $Enc(x, PK_B)$.
Submits the transaction
including $Addr_{secondary}$, Bulletproof, and encrypted data, to the blockchain.
Keys Generation
Generates his pair of public-private keys (PK_B, SK_B).
Decrypts the transaction details using his private key SK_B
Computes the discrete logarithm to retrieve s and verifies the transaction: $r = \log_h (Addr_{secondary}/g^s)$.
Decrypts the transaction
Computes the discrete logarithm
to ensure the transaction amount is within the allowed range.
Check the validity of the Bulletproof
Verify $Addr_{secondary}$
conforms to the DGT DEC-Processing family using the public keys and commitments.
Anchor TX
Anchor the secondary address
Anchor the secondary address $Addr_{secondary}$ and encrypted data on the main-net
Utilize the Cancellation Window mechanism to hold the transaction temporarily, allowing time for additional checks or regulatory compliance.
Utilize the Cancellation Window
ML/KYC/AML Procedures
Optionally uses Selective Disclosure and TEE for secure and compliant data access.
Unfreeze TX
Requests data disclosure for compliance checks
Utilize Multi-Party Computing (MPC)
Utilizes Multi-Party Computing (MPC) to verify transactions against blacklists or sanctions lists without decrypting the transaction data.
verifying the integrity of the transaction process
Rely on publicly available proofs and the blockchain's transparency to ensure system integrity.
Rely on publicly available proofs
May engage in verifying the integrity of the transaction process without accessing the encrypted transaction details

SHIELDED TRANSACTIONS FOR ENHANCED PRIVACY

By enabling transactions where details are encrypted, yet verifiable, shielded transactions are not just a blockchain novelty but a practical necessity for large-scale, institutional, and healthcare transactions, as well as sensitive inter-organizational exchanges outside the blockchain realm..

Shielded transactions are mainly used in blockchain and cryptocurrency applications, such as Zcash, Monero, and Manta Network12. In an era where digital privacy is paramount, shielded transactions stand out as a cornerstone for securing sensitive financial and personal data:

High-Value Blockchain Transactions:

Large financial transfers within the blockchain, such as real estate purchases or corporate acquisitions, where disclosing the transaction amount could affect market dynamics or privacy.

Blockchain to Institution Transactions:

Transfers of funds from blockchain entities to traditional financial institutions or universities for purposes like investment, funding, or tuition payments.

Healthcare Provider Transactions:

Secure transfer of patient data and payments between healthcare providers and insurers using blockchain to ensure data integrity and privacy.

Fintech Inter-organizational Exchanges

Secure sharing of financial data and transactions between fintech organizations, leveraging encrypted channels for collaboration without compromising data security.

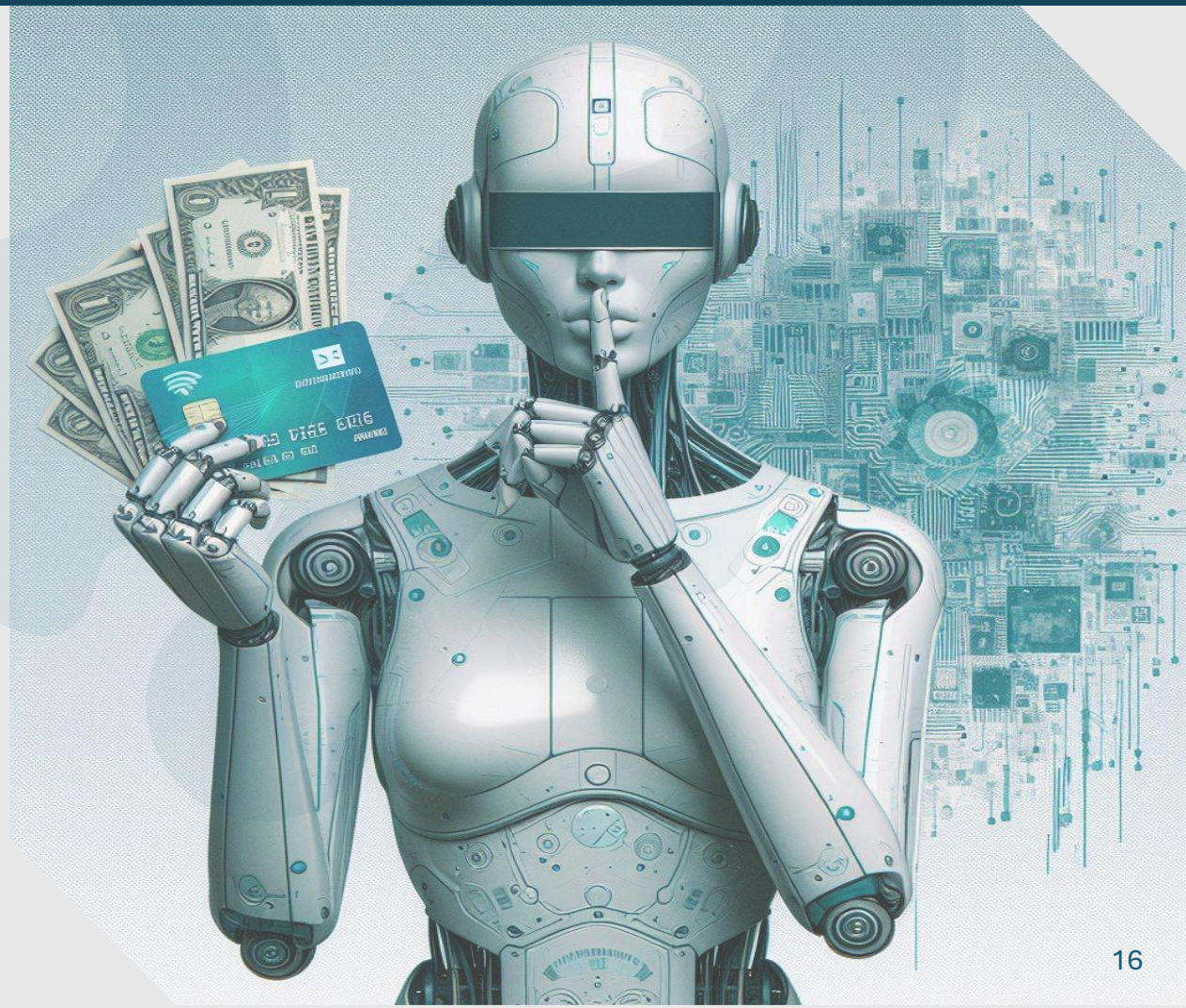
There are also some potential use cases for shielded transactions outside of blockchain, such as:

Digital identity: Shielded transactions can help users protect their personal information and credentials from identity theft and fraud. For example, a user can prove their age, citizenship, or credit score without revealing their name, address, or social security number.

Voting: Shielded transactions can enable secure and anonymous voting systems, where voters can cast their ballots without revealing their identity or preferences. For example, a voter can prove they are eligible to vote without disclosing their political affiliation or candidate choice.

Healthcare: Shielded transactions can improve the privacy and security of medical records and data, where patients and providers can share sensitive information without compromising confidentiality. For example, a patient can prove they have a certain condition or prescription without exposing their medical history or personal details.

THANK YOU



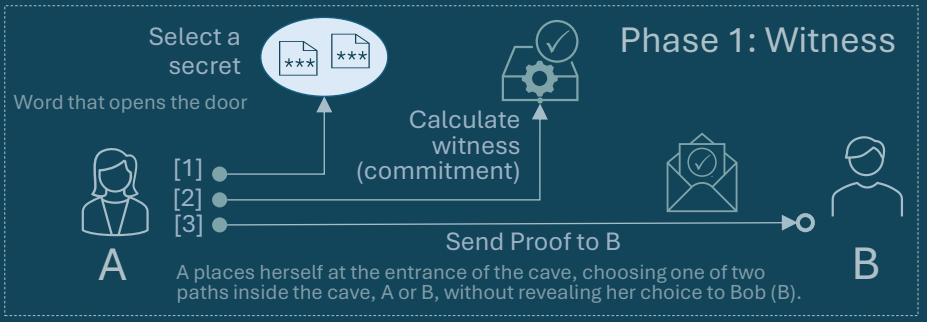
APPENDIX

THE ZKP PROCESS ILLUSTRATED

2025-02-15

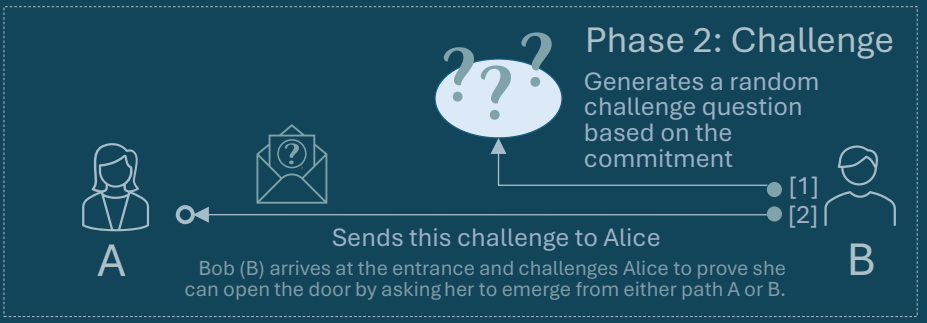
Interactive ZKP: The Ali Baba Cave

Phase 1: Witness



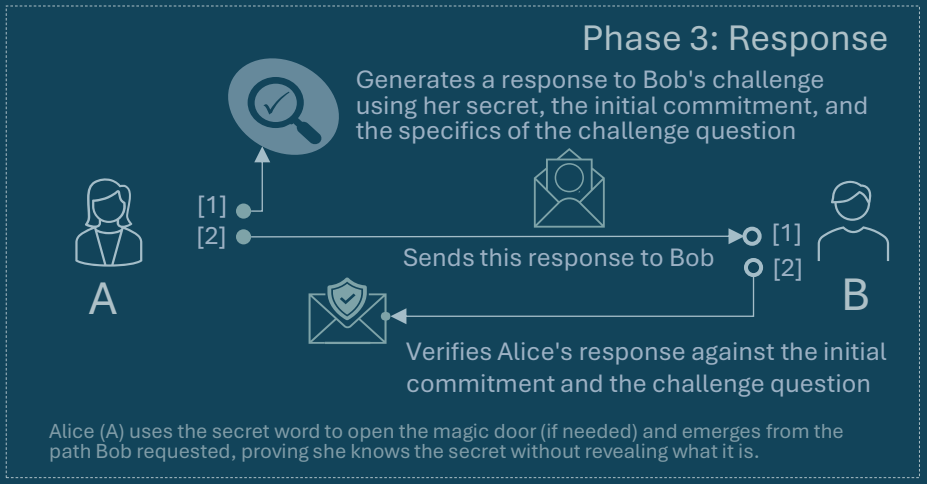
A places herself at the entrance of the cave, choosing one of two paths inside the cave, A or B, without revealing her choice to Bob (B).

Phase 2: Challenge



Bob (B) arrives at the entrance and challenges Alice to prove she can open the door by asking her to emerge from either path A or B.

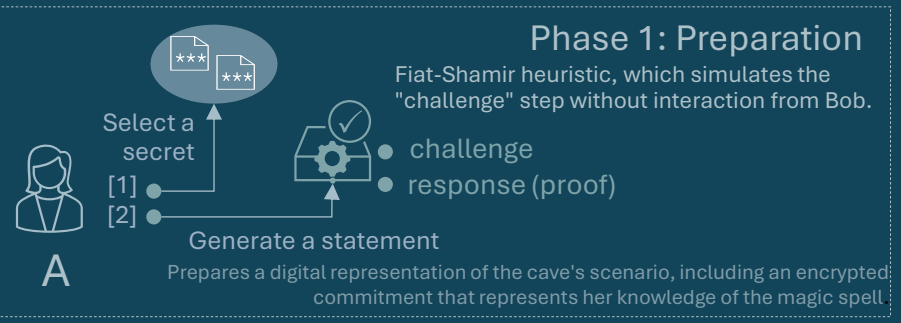
Phase 3: Response



Alice (A) uses the secret word to open the magic door (if needed) and emerges from the path Bob requested, proving she knows the secret without revealing what it is.

Non-Interactive ZKP : The Magic Spell

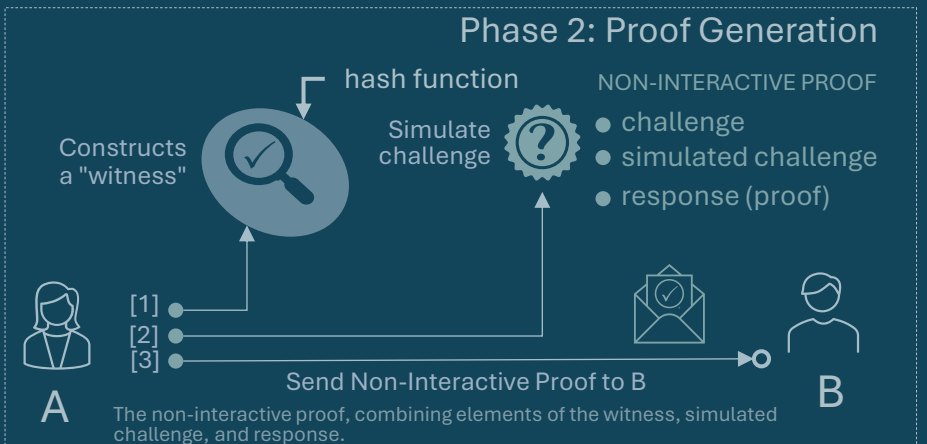
Phase 1: Preparation



Fiat-Shamir heuristic, which simulates the "challenge" step without interaction from Bob.

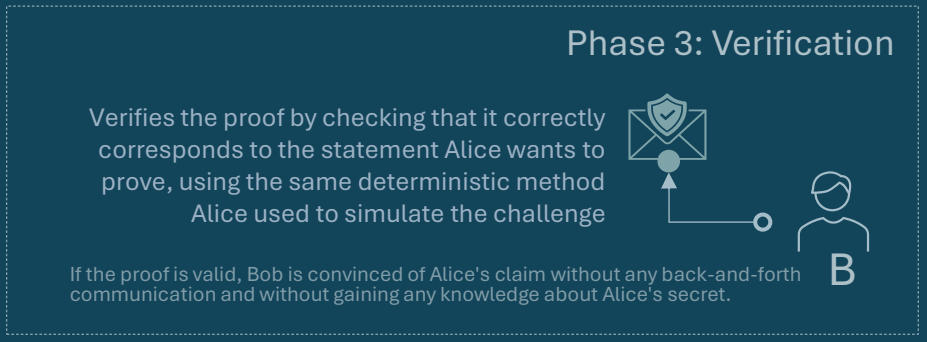
Prepares a digital representation of the cave's scenario, including an encrypted commitment that represents her knowledge of the magic spell

Phase 2: Proof Generation



The non-interactive proof, combining elements of the witness, simulated challenge, and response.

Phase 3: Verification



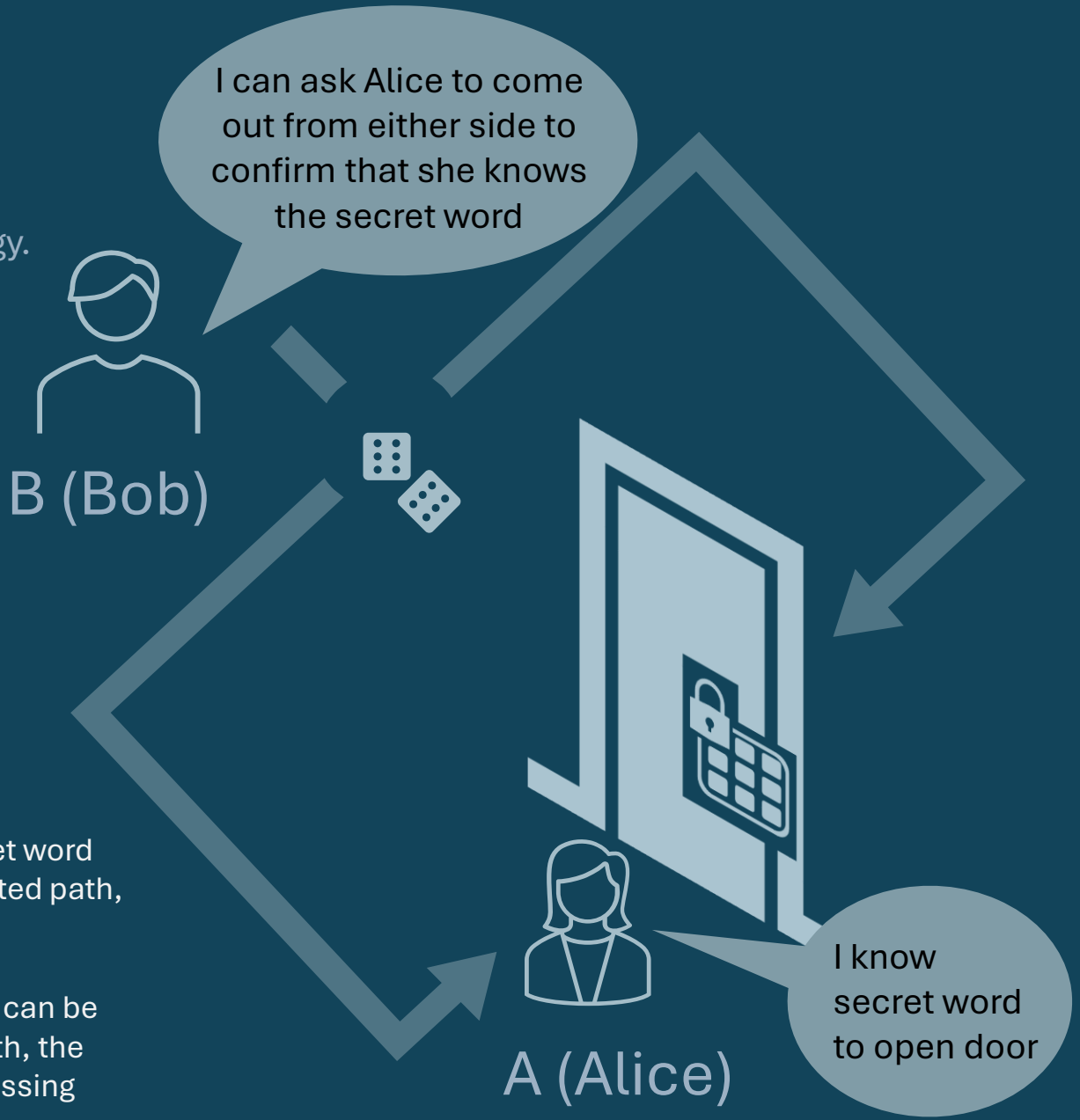
If the proof is valid, Bob is convinced of Alice's claim without any back-and-forth communication and without gaining any knowledge about Alice's secret.

THE ALI BABA CAVE

2025-02-15

The Ali Baba Cave story is a classic parable used to explain the concept of Zero-Knowledge Proofs (ZKP), a cryptographic method that allows one party (the prover) to prove to another party (the verifier) that a certain statement is true, without revealing any information beyond the truth of the statement itself. This story illustrates the essence of ZKP through a simple yet powerful analogy.

1. **Initial Positioning**: Alice enters the cave while Bob waits outside. Alice takes one of the two paths, A or B, but does not tell Bob which one she has chosen.
2. **Challenge**: After Alice has entered, Bob comes to the cave entrance and shouts which path, A or B, he wants Alice to emerge from.
3. **Response**: Alice uses the secret word to open the magic door if necessary and emerges from the requested path. If Alice truly knows the secret word, she can emerge from either path, regardless of her initial choice.



Zero-Knowledge: Throughout the process, Bob learns nothing about the secret word itself. All he knows is whether Alice can successfully emerge from the requested path, proving her claim of knowing the secret word.

Repeated Trials: To convince Bob beyond a reasonable doubt, this procedure can be repeated multiple times. If Alice consistently emerges from the requested path, the probability that she does not actually know the secret word and is merely guessing correctly each time becomes vanishingly small.

2025-02-15

THE MAGIC SPELL

Alice claims she knows a magic spell- secret (digital equivalent of the secret word) that can perform a certain encrypted operation.

PROOF GENERATION

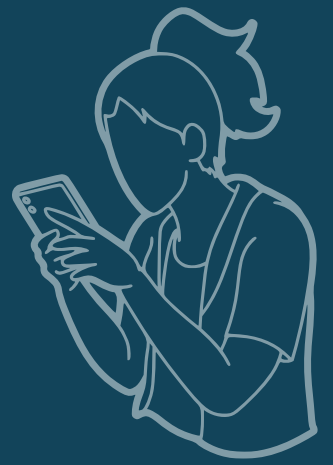
Alice prepares a digital representation of the cave's scenario, including an encrypted commitment that represents her knowledge of the magic spell.
Using the Fiat-Shamir heuristic, she simulates the challenge by hashing the commitment with some public randomness or nonce, creating a scenario where she "proves" she can cast the spell without revealing it.

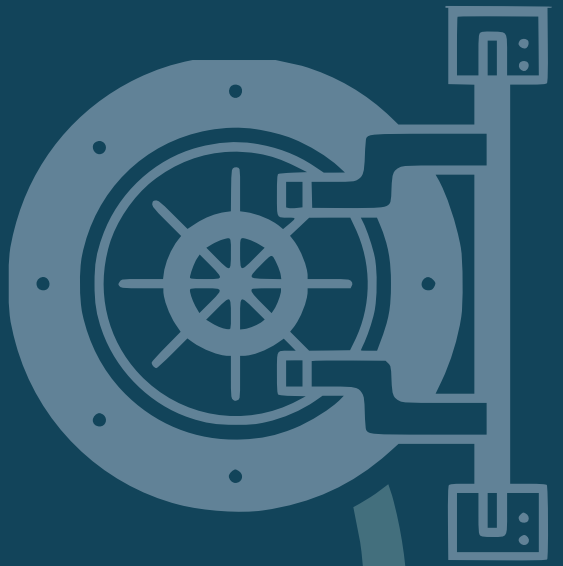
PROOF

A digital proof that combines Alice's initial commitment, the simulated challenge, and the response showing she can "cast the spell" to achieve an effect (like opening a door) without revealing the spell itself.

VERIFICATION

Bob receives Alice's proof and independently computes the hash to verify the challenge-response process. If Alice's submission satisfies the verification process, Bob is convinced Alice knows the spell without ever learning the spell itself.





Statement: Alice claims she knows a magic spell (digital equivalent of the secret word) that can perform a certain encrypted operation.

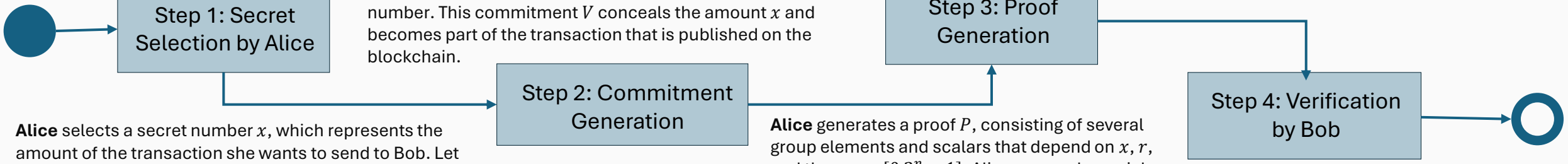
Secret: The magic spell itself.

MAJOR ZKP ALGORITHMS

Protocol Name	Purpose	Year	Use Cases	Key Features	Privacy	Performance	Compliance
ZK-SNARK	Proof of Knowledge	2012	Zcash, NuCypher, Tornado Cash	High efficiency, no interaction required	High	Medium	Low
ZK-STARK	Proof of Knowledge	2018	StarkNet, Immutable X	High scalability, transparency	High	High	Medium
PLONK	Proof of Knowledge	2019	Aztec Connect, zkSync	High efficiency, suitable for smart contracts	High	High	Medium
Halo	Proof of Knowledge	2020	StarkNet, Scroll	High efficiency, EVM compatibility	High	High	Medium
Groth16	Proof of Knowledge	2016	Zcash	First zk-SNARK used in cryptocurrencies	High	Medium	Low
Bulletproofs	Range Proof	2018	Grin, Beam	Efficient range proofs, no elliptic curve cryptography needed	High	High	Medium
Lelantus Sigma	Range Proof	2019	LelantusMW, Dusk Network	Confidential range proofs	High	Medium	Low
Pedersen Commitments	Commitment Scheme	-	Various privacy coins	Hiding and binding properties, used in confidential transactions	Medium	High	High
Schnorr Identification Protocol	Identification Scheme	-	Cryptographic authentication	Security based on the difficulty of discrete logarithm, compact proofs	High	High	High
MimbleWimble	Blockchain Protocol	2016	Beam, Grin	Privacy and fungibility, scalability through cut-through feature	High	High	Medium
PIPE-ZK	Proof of Knowledge				High	Medium	Medium
Aztec Protocol	private transactions on Ethereum	2020	Ethereum ecosystem	Utilizes ZK-SNARKs to provide encrypted transactions that are fully private, not only hiding the amount but also the sender and receiver's information.	High	Medium	Medium
Confidential Transactions (CT)	Proof of Knowledge	2013	Originally proposed for Bitcoin; like Liquid Network	Uses cryptographic techniques to hide the specific amounts in transactions, while still allowing the network to verify that no more coins are spent than are available in a user's wallet	High	Medium	Low to Medium

BULLETPROOF

Bulletproof is a type of zero-knowledge proof (ZKP) that allows you to prove that some secret number is within a given range without revealing any other information. Bulletproof is used to hide the amount of a shielded transaction by converting it to a range of values that hides transaction data, such as addresses and amounts, using ZKP



Alice creates a commitment V using the formula $V = g^x h^r$, where g and h are public generators of a group where computing the discrete logarithm is difficult, and r is a random number. This commitment V conceals the amount x and becomes part of the transaction that is published on the blockchain.

Alice selects a secret number x, which represents the amount of the transaction she wants to send to Bob. Let x be within the range $[0,2^n - 1]$, where n is the bit length of the amount. For example, if n=8, then $x \in [0,255]$.

Alice generates a proof P, consisting of several group elements and scalars that depend on x, r, and the range $[0,2^n - 1]$. Alice uses polynomial computations, vector commitments, and inner product arguments to construct the proof. This proof P also becomes part of the transaction that is published on the blockchain.

Bob receives the transaction containing V and P and verifies the proof P using only V and publicly available parameters. Bob learns nothing about x or r apart from the fact that $x \in [0,2n - 1]$.

Bob checks the proof of P using only V and publicly available parameters such as g, h, n, and others. The test consists of Bob calculating some values from P and comparing them to the expected values that certain equations are supposed to perform. For example, one of the equations that Bob tests

$$V = g^x h^r = g^{\langle a_L, 2^n \rangle + \delta(y,z)} h^z$$

where a_L is a bit representation of x, $2n$ is a vector of powers of two, $\delta(y,z)$ is some function of the variables y and z, which are part of P. This equation verifies that the liability V corresponds to the sum x, which is in the range $[0, 2^n - 1]$.

Bob doesn't know x, so he can't calculate a_L. However, a_L is also included in the proof of P that Bob knows. Bob can use a_L from P to check the equation:

$$e(g,g) = e(A,h) \cdot e(g,S)^y \cdot e(g,T_1)^{y^2} \cdot e(g,T_2)^{y^3} \cdot e(g,h)^{-z} \cdot e(g,g)^{z^2 \langle \mathbf{2^n}, \mathbf{u} \rangle} \cdot e(V,h)^z \cdot e(g,h)^{\delta(y,z)} \cdot e(\mathbf{G},\mathbf{l}) \cdot e(\mathbf{H},\mathbf{r}) \cdot e(\mathbf{B},\mathbf{y}^{-n})^{-\mu} \cdot e(g,h)^{-c} \cdot e(P,Q)^c$$

$$P = (A, S, T_1, T_2, \tau_x, \mu, \mathbf{l}, \mathbf{r}, a, b, c, d, e, f)$$

PEDERSEN COMMITMENT

Pedersen commitment is a type of commitment that allows you to hide some value without revealing it, but allows you to check its correctness. Pedersen commitment uses a group in which it is difficult to compute a discrete logarithm and two public group generators

The following scheme can be used to hide the address with the Pedersen commitment:

Step 1:Secondary (Hidden) Address Creation by Sender Alice

Alice's Public Address

Let A be the public address, a is the private key associated with A, g and h are the group's public generators, r is a random number.

Group Generators creation

Let g and h be public generators of the group.

Random Number Selection

Alice chooses a random number s.

Secondary Address Calculation

Alice computes the secondary address $B = g^s h^r$, where r is a random number.

Publishing to Blockchain

- Alice sends B to the blockchain.
- Alice also encrypts s and any additional message for the recipient using their public key A and publishes this encrypted information to the blockchain.

Step 2: Retrieval of Secondary Address by Recipient Bob

Scanning Blockchain

Recipient Bob scans the blockchain for encrypted messages addressed to him.

Decryption Using Private Key

Bob decrypts the received data using his private key a.

Retrieval of Random Number and Message

- Bob obtains the random number s and any additional message from the sender Alice.
- Bob computes r using the discrete logarithm $r = \log_h(B/g^s)$, where $\log_h$ denotes the discrete logarithm base h.

Verification

Bob verifies that B is indeed equal to $g^s h^r$ to confirm that the secondary address B has been properly formed.

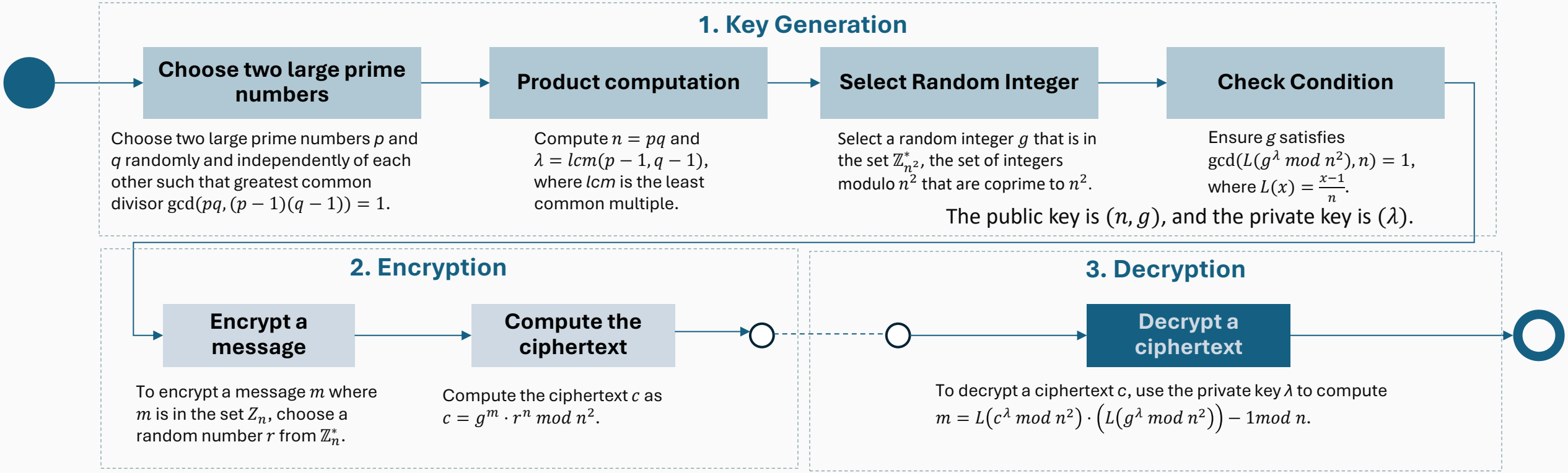
Privacy: The secondary address B is a Pedersen commitment that hides the values of s and r, ensuring that the transaction details remain private.

Security: Only the intended recipient, who has the correct private key a, can decrypt and retrieve the random number s and the message.

Integrity: The verification step ensures the integrity of the received secondary address B and the associated encrypted message.

PAILLIER HOMOMORPHIC ENCRYPTION

Paillier HE is a public-key cryptosystem that exhibits additive homomorphic properties. This means that, given the encryption of two numbers, anyone can compute an encryption of the sum of these numbers without decrypting the individual encryptions.



Homomorphic Properties:

- Addition of Plaintexts: Given two ciphertexts c_1 and c_2, corresponding to messages m_1 and m_2, the product $c_1\cdot c_2 \bmod n^2$ will decrypt to $m_1+m_2 \bmod n$.
- Multiplication by a Scalar: Given a ciphertext c and a plaintext number k, the ciphertext $c^k \bmod n^2$ will decrypt to $m\cdot k \bmod n$.