Converging Privacy Computing, Verifiable Computation, and Layer-2 Blockchain Protocols: A Future-Oriented Architecture for Trustworthy Smart Contracts

Luca Benedetti[1], Sofia Marino[2], Matteo Ricci[3], *
[1] Department of Computer Science, University of Salerno, Fisciano, Italy, 84084
[2] Department of Economics and Management, University of Bari Aldo Moro, Bari, Italy, 70121
[3] Department of Engineering and Architecture, University of Trieste, Trieste, Italy, 34127
*Email: matteo.ricci@units.it (Corresponding Author)

Abstract

Smart contracts are increasingly expected to support private, auditable, and low-cost digital agreements in finance, supply chains, healthcare data exchange, public services, and machine-to-machine coordination. Yet the same properties that make blockchain execution attractive also create a difficult design tension: fully public on-chain computation offers auditability but exposes business logic and data, while heavy cryptographic privacy can protect confidentiality at a cost that limits practical adoption. Inspired by recent formal analyses of layer-2 optimistic rollups and replicated off-chain computation, this article develops a future-oriented architecture that converges privacy computing, verifiable computation, and layer-2 blockchain protocols for trustworthy smart contracts. Rather than proposing another single protocol, the study provides an architectural framework, risk taxonomy, and data-oriented evaluation model for systems that combine off-chain execution, selective disclosure, dispute resolution, commitment structures, and settlement-layer accountability. The proposed architecture emphasizes privacy-by-design, verification-by-default, and settlement-by-exception. It also addresses free-riding, copy behavior, no-action behavior, collusion risk, data leakage through dispute evidence, and the operational cost of adding privacy after deployment. A small comparative data analysis illustrates how privacy-preserving verification changes the distribution of gas cost, transaction frequency, and governance burden across accept and challenge scenarios. The article contributes to the literature by shifting attention from isolated smart contract privacy mechanisms toward integrated, future-proof protocol architecture. It concludes with a research agenda for reactive contracts, modular verification, AI-assisted anomaly detection, cross-rollup interoperability, and compliance-aware smart contract governance.

Keywords: privacy computing; verifiable computation; layer-2 blockchain; optimistic rollup; smart contracts; trustworthy architecture; dispute resolution; protocol governance

Article History:

Received: April 18, 2025
Revised: June 20, 2025
Accepted: August 16, 2025
Available Online: September 30, 2025

I. INTRODUCTION

Blockchains made it possible to execute agreements through programs rather than institutional intermediaries. In their most visible form, smart contracts are public, deterministic, and verifiable. These properties are valuable because the parties do not need to trust a single database operator or a private platform owner. However, the same transparency that creates public auditability can become a barrier to adoption when contract logic, intermediate states, commercial terms, bids, payments, or identity-linked data should not be exposed to every observer on a public chain. The problem is no longer whether smart contracts can run; the more important question is whether they can run with privacy, verifiability, scalability, and governance assurance at the same time (Kou et al.,2025； Fiore et al.,2014).

Layer-2 blockchain protocols emerged as a practical response to the cost and congestion limitations of layer-1 blockchains. Instead of forcing every node to repeat every instruction, layer-2 systems move most computation off chain and use the base chain mainly for settlement, dispute resolution, and final accountability. Optimistic rollups, validity rollups, state channels, and other layer-2 designs differ in their verification logic, but they share an architectural ambition: preserve the security anchoring of a public blockchain while avoiding the cost of fully on-chain execution. This ambition is especially important for smart contracts that require complex computation, repeated transactions, or confidential business data (Atzei et al.,2017； Walfish et al.,2015).

The research direction motivating this article concerns formal security and privacy models for layer-2 smart contracts, with particular attention to optimistic rollups, off-chain managers, dispute resolution, and free-riding risks. Its core implication is that

a layer-2 design may produce the correct output yet still fail to detect participants who accept results without independently computing them (Lu,2025； Setty et al.,2012).

This article builds on that research direction while taking a broader architectural path. It does not reproduce a formal protocol proof or present a narrow platform-specific model. Instead, it develops a future-oriented architecture that converges privacy computing, verifiable computation, and layer-2 settlement into a practical model for trustworthy smart contracts (Nikolic et al.,2018； Thaler,2013).

The phrase trustworthy smart contracts is used here in a broad but operational sense. A trustworthy smart contract architecture should ensure that authorized participants can execute private business logic, that public observers receive only the information needed for verification or settlement, that dishonest participants are discouraged and detected, that dispute evidence does not unnecessarily leak computation traces, and that the system remains deployable under realistic cost constraints. Trustworthiness therefore combines technical assurance with economic incentives and operational governance (Wu et al.,2025； Parno et al.,2013).

The paper makes three contributions. First, it develops an integrated architecture that places privacy computing, verifiable computation, and layer-2 settlement into a single design model. Second, it offers a risk taxonomy that links technical attacks with economic incentives and governance failures. Third, it provides a data-oriented analysis of representative cost and transaction patterns to show why privacy-by-design may be more practical than retrofitting privacy into an existing verifiable computation protocol. The article is conceptual and analytical rather than purely theoretical, and its target contribution is a future-oriented research framework for Crossroads of Future Technologies (Tsankov et al.,2018； Bonawitz et al.,2017).

II. BACKGROUND AND CONCEPTUAL FOUNDATION

A. Privacy Computing for Smart Contracts

Privacy computing refers to a family of techniques that allow computation, verification, or data exchange while limiting unnecessary exposure of sensitive information. In blockchain environments, privacy computing may include commitments, secure multiparty computation, homomorphic encryption, zero-knowledge proof systems, trusted execution, data minimization, selective disclosure, and hybrid approaches that combine cryptographic and institutional safeguards. The key issue is not simply hiding data; it is enabling useful computation without making the entire computation trace public (Lu et al.,2024； Acar et al.,2018).

For smart contracts, privacy has at least three layers. Transactional privacy protects senders, receivers, amounts, and inputs. State privacy protects the values that determine the contract state. Transition privacy protects the logic and intermediate execution steps through which a state changes. Many public blockchain designs are strong in integrity but weak in privacy because state transitions are deliberately visible. Layer-2 protocols offer a different design space: they can keep most computation off chain while allowing public verification only when required (Tikhomirov et al.,2018； Gentry,2009).

The architectural challenge is that privacy mechanisms often introduce computational overhead. Zero-knowledge proof systems may provide strong privacy and succinct verification, but proof generation can be complex, and application developers may require specialized expertise. Fully homomorphic encryption and generic obfuscation are powerful in principle but remain costly for broad smart contract workloads. A future-oriented architecture therefore needs a modular privacy layer that selects mechanisms according to the sensitivity, frequency, value, and dispute probability of each contract workflow (Xu et al.,2024； Dwork,2006).

B. Verifiable Computation and Replicated Execution

Verifiable computation allows a client or public verifier to gain assurance that a computation was performed correctly without repeating all of it. In blockchain systems, this assurance can be achieved through proof systems, replicated off-chain computation, challenge-response games, fraud proofs, or combinations of these mechanisms. Replicated execution is especially relevant for optimistic protocols because several managers or validators may execute the same task off chain, compare results, and rely on a public referee contract only when disagreement occurs (Durieux et al.,2020； Kairouz et al.,2021).

The main advantage of replicated off-chain computation is practical simplicity. Managers can execute ordinary programs, and the settlement layer is used mainly to check commitments, resolve disputes, and penalize misconduct. The main weakness is behavioral: if a manager can receive a reward without doing the work, the system may become correct only because someone else acted honestly. Over time, such free riding can weaken security, reduce incentives for diligent verification, and invite collusion. This is precisely why result confirmation, unpredictable challenge selection, and accountable acceptance become central design elements (Chen et al.,2024).

Trustworthy architecture should not assume that every participant has a stable utility function or that rational incentives

explain all misconduct. A validator may delay publication, copy a result, remain silent, collude with another participant, or leak private computation details even when immediate economic payoff is unclear. Formal models are useful because they define what a protocol promises under explicit assumptions. Architectural models are also needed because deployment environments introduce software, governance, and interoperability constraints that are not fully captured by proof (Kalra et al.,2018).

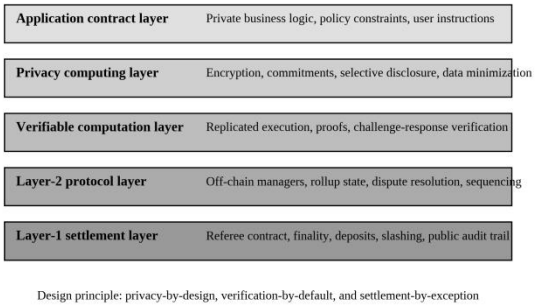

Fig 1. Layered architecture for converged privacy computing, verifiable computation, and layer-2 smart contract protocols.

Fig 1 summarizes the proposed architecture without presenting it as a linear pipeline. The layers are stacked because trustworthy smart contract execution depends on co-design rather than one-way processing. The application contract layer defines business logic and policy constraints. The privacy computing layer decides what should be hidden, committed, encrypted, or selectively disclosed. The verifiable computation layer supplies evidence that off-chain execution was correct. The layer-2 protocol layer coordinates managers, sequencing, dispute windows, and off-chain state. The layer-1 settlement layer provides public finality, deposits, slashing, and auditability. The central design principle is that the public chain should settle and verify exceptions, not expose every routine computational step (Lu et al.,2023).

This architecture differs from conventional smart contract engineering in two ways. First, it treats privacy as a first-order requirement rather than a secondary feature. Second, it treats verification as an operational process with multiple evidence levels rather than a single yes-or-no proof. Some contracts may need only commitments and optimistic challenge windows. Others may require zero-knowledge proofs, threshold signatures, or multiparty audit trails. The architecture is therefore not a replacement for existing rollup protocols; it is a design map for selecting and integrating mechanisms (Rodler et al.,2019).

TABLE I. DESIGN REQUIREMENTS FOR FUTURE-ORIENTED TRUSTWORTHY SMART CONTRACTS

Requirement	Architectural Meaning	Representative Design Response
Outsider privacy	Limit leakage to entities outside the authorized execution group	Commitments, encrypted state, minimal on-chain assertions
Execution integrity	Ensure off-chain computation matches the agreed contract logic	Replicated computation, fraud proofs, validity proofs, trace commitments
Free-rider resistance	Prevent parties from accepting rewards without doing the work	Result confirmation, unpredictable spot checks, proof of participation
Dispute confidentiality	Resolve disagreement without revealing the full computation trace	Merkle commitments, selective opening, private challenge design
Cost scalability	Avoid making layer-1 validators repeat routine work	Settlement-by-exception and batched finalization
Governance accountability	Make roles, incentives, and failure procedures auditable	Deposits, slashing, policy logs, compliance metadata

III. RESEARCH GAP AND PROBLEM DEFINITION

Table I shows that future trustworthy smart contracts cannot be evaluated by security proofs alone. Security proofs answer whether a protocol satisfies a formal definition under specified assumptions. Deployment decisions require additional questions: Who chooses the managers? What evidence is published when a dispute occurs? How are deposits calculated? What happens when the computation is reactive rather than fixed at the start? How is privacy maintained when the chain itself becomes an archive of dispute evidence? These questions are not peripheral; they determine whether a privacy-preserving layer-2 protocol can become usable infrastructure (Ye et al.,2022).

The first research gap concerns the separation between privacy computing and verifiable computation. Many privacy systems are designed around confidentiality, while many verification systems are designed around correctness. In smart contracts, the two cannot be separated. A dispute mechanism that proves correctness by revealing too many intermediate states may technically resolve a challenge while undermining the contract's privacy purpose. Conversely, a privacy mechanism that hides too much may prevent the referee from detecting misconduct. Future architecture must balance these requirements at the level of protocol

evidence (Hildenbrandt et al.,2018).

The second research gap concerns incentive-compatible verification. Optimistic protocols frequently depend on at least one honest or economically motivated participant to challenge false claims. However, if verification work is costly and acceptance can be passive, rational participants may wait for others to do the work. This behavior is subtle because the system may still produce correct outputs in many runs. The danger is not necessarily immediate incorrectness but the erosion of the replicated computation assumption that makes the protocol trustworthy (Lu,2022).

The third research gap concerns the cost of retrofitting privacy. A protocol originally designed for correctness may use public intermediate responses, public challenges, or referee-generated queries that are efficient for verification but damaging for privacy. Adding privacy later can require extra commitments, additional transactions, encrypted evidence, or redesigned dispute logic. This creates a broader principle for future smart contract engineering: privacy should be considered at the time the verification pathway is designed, not after the protocol becomes operational (Grishchenko et al.,2018).

The fourth gap concerns evaluation. Smart contract research often compares gas cost, throughput, proof size, or finality time. These are important, but they do not fully measure trustworthy architecture. A better evaluation framework should also measure leakage surface, misconduct detectability, dispute confidentiality, role complexity, manager storage burden, upgradeability, and compliance readiness. The article therefore defines a multidimensional evaluation model rather than relying on a single performance indicator (Zheng et al.,2022).

IV. PROPOSED ARCHITECTURE

A. Architectural Logic

The proposed architecture is built on three operating principles: privacy-by-design, verification-by-default, and settlement-by-exception. Privacy-by-design means that the protocol identifies sensitive objects before execution begins, including inputs, outputs, intermediate states, contract logic, role identities, and dispute evidence. Verification-by-default means that every acceptance decision should be accountable, even when no dispute occurs. Settlement-by-exception means that the layer-1 chain is used when a public commitment, finality event, or dispute check is necessary, not as a general-purpose computation engine (Torres et al.,2018).

At the start of a contract session, the client defines the contract package, participant set, privacy policy, reward rule, deposit condition, and verification mode. The contract package is not necessarily posted on chain. Instead, a commitment to the package and initial state is posted, while authorized managers receive the executable logic through a protected channel. The managers compute off chain and publish only the minimal assertion required by the protocol: normally a state commitment, output commitment, execution length, and metadata needed for the dispute window (Xu et al.,2021).

Acceptance is treated as an accountable act. A manager that accepts an assertion may need to provide lightweight confirmation that it has independently derived a checkpoint, selected state hash, or trace commitment. The purpose is not to reveal the computation but to discourage passive acceptance. When a dispute occurs, the architecture uses selective opening: only the disputed segment, proof path, or single-step evidence necessary for resolution should be disclosed. The referee contract validates the evidence and applies penalties according to the recommitted rules (Schneidewind et al.,2020).

This design differs from naive replicated computation. In naive replicated computation, the system may rely on matching results and disputes after disagreement. In the proposed architecture, the acceptance path itself is instrumented to provide evidence of work, while the challenge path is instrumented to prevent unnecessary privacy loss. This double instrumentation is essential for the convergence of privacy and verification (Zhang et al.,2021).

B. Core Components

The privacy policy engine classifies contract data into public, private-to-managers, private-to-subsets, and never-disclosed categories. Public data may include the existence of a contract session, deposits, and final commitments. Private-to-managers data may include computation logic and inputs. Private-to-subsets data may include role-specific information, such as bids or supplier terms. Never-disclosed data may include secrets that should remain hidden even during disputes, requiring proof-based or encrypted verification (Mavridou et al.,2019).

The trace commitment service records a compact representation of execution without storing the full trace on chain. Merkle trees, vector commitments, or polynomial commitments may be used depending on the workload. In a simple optimistic design, managers store the trace off chain and publish roots or checkpoints. During a dispute, only selected evidence is opened. This supports the architectural goal of dispute confidentiality (Lu et al.,2020).

The verification coordinator manages acceptance, challenge, and penalty procedures. In manager-based systems, it should prevent non-action behavior by requiring explicit acceptance or accountable silence rules. If a party remains silent, the protocol must distinguish network failure, strategic non-response, and legitimate abstention. Such distinctions are difficult but ignoring

them creates incentives for no-action attacks. The coordinator therefore needs deadlines, deposits, recoverable failure procedures, and audit logs (Chen et al.,2020).

The governance and compliance layer records policy-relevant metadata. This does not mean that private data is disclosed. Instead, metadata may document which privacy mode was used, which parties were authorized, which evidence type resolved a dispute, and whether penalties were applied. For regulated environments, this layer can support auditability without forcing full transparency of business data (Lu et al.,2019).

TABLE II. COMPONENTS OF THE PROPOSED ARCHITECTURE

Component	Main Function	Risk Controlled	Operational Output
Privacy policy engine	Classify data and disclosure rules	Unnecessary public leakage	Privacy mode and disclosure map
Trace commitment service	Committee to execution states off chain	Evidence manipulation and full-trace exposure	State roots, checkpoints, proof paths
Verification coordinator	Manages acceptance and challenge behavior	Free-riding, no-action, strategic delay	Acceptance proofs, challenge logs, slashing events
Settlement contract	Anchors commitments and final decisions	Unenforceable off-chain promises	Finality, deposits, penalties, public audit record
Governance layer	Records role and compliance metadata	Ambiguous accountability	Audit Metadata and governance reports
Interoperability interface	Connects rollups, wallets, and external systems	Fragmented execution and vendor lock-in	Standard APIs and cross-rollup evidence format

V. RISK TAXONOMY FOR CONVERGED LAYER-2 PRIVACY SYSTEMS

Future-oriented architecture needs a risk taxonomy because layer-2 smart contract failures are not all of the same kind. Some failures are cryptographic, such as invalid commitments or broken proof assumptions. Some are economic, such as reward rules that make lazy behavior attractive. Some are operational, such as manager downtime, software incompatibility, or dispute window misconfiguration. Others are governance failures, such as unclear responsibility for selecting managers or updating contracts. Treating all risks as technical attacks leads to incomplete architecture (Bartoletti et al.,2018).

The first risk of family is computation integrity risk. It includes false assertions, incorrect state transitions, incomplete execution, malicious challenge generation, and invalid proof submission. In optimistic designs, integrity risk is controlled through disagreement detection, challenge-response procedures, and penalties. In proof-based designs, it is controlled through proof verification. In hybrid designs, a low-cost optimistic path may be used for routine execution, while stronger proofs are reserved for high-value or high-risk transactions (Lu,2019).

The second risk family is participation integrity risk. It includes free-riding, copy attacks, no-action behavior, and silent acceptance without computation. These behaviors can be difficult to detect because the final output may still be correct if another manager performed the work. The architecture therefore requires proof of participation or random checkpoint confirmation. This is a key distinction between producing a correct result once and maintaining a trustworthy verification ecosystem over time (Destefanis et al.,2018).

The third risk family is privacy leakage risk. It includes public exposure of inputs, outputs, intermediate states, computation logic, manager identities, and dispute evidence. A poorly designed dispute mechanism may leak more information during one challenge than the routine protocol leaks across many successful executions. Leakage should therefore be evaluated not only in normal operation but also in adversarial and exceptional operation (Lu,2018).

The fourth risk family is governance and compliance risk. Even when a protocol is technically sound, real users need to understand who is responsible for manager selection, contract upgrades, key management, emergency pauses, and data retention. Institutional users may require evidence of compliance with data protection, anti-fraud, auditing, and financial reporting rules. Layer-2 smart contract architecture must therefore include governance hooks rather than leaving these questions to application developers (Yli-Huumo et al.,2016).

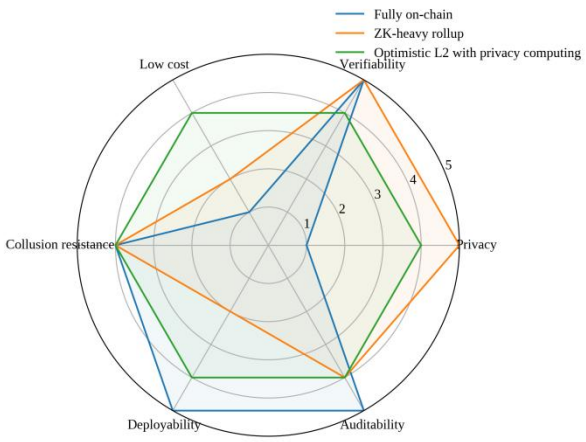

Fig 2. Comparative design trade-offs among three smart contract execution models.

Fig 2 compares three broad execution models across six design dimensions. Fully on-chain execution performs well in auditability and deploy ability but poorly in privacy and cost. Zero-knowledge-heavy rollups perform strongly in privacy and verifiability but face challenges in developer complexity and operational cost. Optimistic layer-2 systems with privacy computing occupy a middle position: they may not maximize every dimension, but they can provide a balanced architecture for applications where cost, privacy, and verifiability must all remain acceptable (Lu,2017).

The figure should not be interpreted as a universal ranking. The best design depends on the application. A confidential auction may require stronger privacy than a loyalty points program. A high-value derivatives contract may justify expensive proof generation, while a supply chain quality claim may prefer optimistic verification with selective dispute evidence. The point is that future architecture should support design choice rather than force every smart contract into a single verification model (Casino et al.,2019).

TABLE III. RISK TAXONOMY AND CONTROL MECHANISMS

Risk Type	Typical Manifestation	Control Mechanism	Architecture Implication
Computation integrity	False assertion or invalid state transition	Trace commitments, fraud proof, proof verification	Verification layer must be modular
Participation integrity	Copy attack, no-action behavior, passive acceptance	Checkpoint confirmation, deposits, accountable acceptance	Acceptance cannot be treated as costless silence
Privacy leakage	Intermediate states exposed during disputes	Selective opening, encrypted evidence, minimum disclosure	Dispute path must be privacy-aware
Collusion	Multiple managers coordinate around false behavior	Any-trust design, randomized checks, role diversity	Manager selection and incentive rules matter
Operational failure	Timeouts, network delays, software incompatibility	Grace periods, recovery rules, redundant interfaces	Governance must handle non-malicious failure
Compliance failure	Insufficient audit metadata or unclear responsibility	Policy logs, role records, disclosure classification	Technical protocols need institutional interfaces

VI. DATA ANALYSIS AND EVALUATION MODEL

To make the architectural argument more concrete, this section develops a compact data analysis based on representative protocol evaluation dimensions. The purpose is not to benchmark a production rollup. Rather, it is to illustrate how trustworthy architecture should be evaluated across cost, transaction count, verification depth, and privacy exposure. The motivating PDF manuscript provides a useful reference point because it compares accept and challenge scenarios for Arbitrum0, Arbitron, and a privacy-preserving SC-RDoC design, including gas cost and transaction count for a matrix multiplication benchmark (Saberi et al.,2019).

The main lesson from such a comparison is that privacy-aware verification changes where cost appears. Privacy-by-design architecture may add a small amount of routine cost to the acceptance path because accepting managers must provide evidence that they perform relevant work. However, it may reduce challenge-path cost or avoid the much larger cost of retrofitting privacy into a protocol that originally allowed the referee contract to see intermediate computation results. The difference is architectural, not merely numerical (Kshetri,2018).

In evaluation terms, four metrics are especially useful. The first is normal-path cost: gas and transactions when all managers agree. The second is dispute-path cost: gas, transactions, rounds, and revealed evidence when there is disagreement. The third is leakage index: the amount and sensitivity of information disclosed in normal and dispute paths. The fourth is participation

assurance: the probability or confidence that accepting parties performed the required computation. A system with low gas cost but low participation assurance may look efficient while becoming fragile over repeated use (Li et al.,2020).

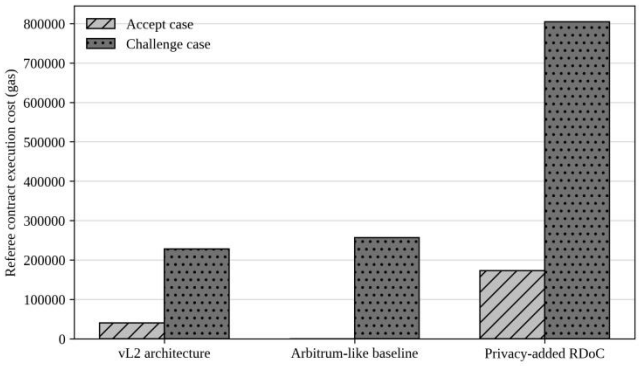

Fig 3. Illustrative on-chain verification cost patterns in accept and challenge cases.

Fig 3 uses representative gas values from a matrix multiplication evaluation to show why architectural placement of privacy matters. The privacy-added RDoC design has substantially higher accept and challenge costs because privacy protection is layered onto a verification design that was not originally optimized for confidential dispute handling. The vL2 architecture, by contrast, adds work in the acceptance case but can remain much lower than the privacy-added alternative in the challenge case. This supports the broader design principle that privacy should be embedded into the verification pathway early (Queiroz et al.,2020).

The acceptance case deserves special attention. Some baseline systems appear inexpensive because only one party submits a result, and no comparison is required on chain. But if acceptance is entirely passive, the protocol may be unable to tell whether another party computed the result or simply waited. A small acceptance cost can be justified if it materially improves participation assurance. Future evaluations should therefore report both gas cost and work-confirmation strength (Azzi et al.,2019).

The challenge case is equally important because disputes are rare in optimistic operation but critical for security credibility. The cost of a dispute includes not only gas and transactions but also privacy loss, delay, user experience, and governance burden. A dispute mechanism that is cheap but reveals sensitive computation may be unsuitable for private enterprise use. Conversely, a highly private challenge mechanism that is too expensive may discourage legitimate challenges and weaken integrity (Mendling et al.,2018).

TABLE IV. EVALUATION DIMENSIONS FOR TRUSTWORTHY LAYER-2 SMART CONTRACTS

Dimension	Metric Example	Why It Matters
Normal-path efficiency	Gas, transaction count, confirmation delay	Determines routine usability and cost predictability
Dispute-path robustness	Rounds, gas, evidence size, finality delay	Determines credibility when disagreement occurs
Privacy exposure	Public fields, opened states, leaked metadata	Determines whether confidentiality survives execution and dispute
Participation assurance	Evidence of independent computation or checkpoint knowledge	Controls free-riding and passive acceptance
Governance clarity	Role logs, upgrade procedures, emergency rules	Supports institutional adoption and accountability
Interoperability	Compatibility of evidence and state formats	Supports future cross-rollup and multi-chain deployment

VII. SCALABILITY, DEPLOYMENT, AND FUTURE APPLICATION SCENARIOS

The scalability promise of layer-2 architecture comes from moving routine computation off chain while retaining layer-1 settlement for finality and dispute resolution. However, scalability should not be reduced to transaction throughput. A privacy-preserving layer-2 system also has to scale manager storage, trace commitment generation, proof handling, dispute windows, monitoring, and governance operations. If managers must store long execution traces, storage requirements may become a hidden cost. If dispute procedures require many rounds, finality may become uncertain for complex contracts (Christidis et al.,2016).

The proposed architecture therefore recommends adaptive granularity. Some computations can be committed at the opcode or instruction level, which supports precise disputes but increases trace length. Others can be committed at block, function, or

semantic checkpoint level, which reduces trace size but may make dispute resolution less precise. The appropriate granularity should depend on contract value, privacy sensitivity, expected dispute probability, and computational complexity. This is a design decision that belongs to architecture, not a low-level implementation detail (Novo,2018).

Deployment complexity is another major issue. A real layer-2 smart contract system includes on-chain contracts, off-chain managers, virtual machine components, wallets, sequencers, monitoring services, and governance tools. Modifying the dispute mechanism may require changes across several components. Therefore, future architectures should define stable interfaces between the privacy layer, verification layer, and settlement layer. Without stable interfaces, each privacy or verification upgrade becomes a full system redesign (Ouaddah et al.,2017).

Application scenarios are broad. In decentralized finance, private trading strategies, collateral conditions, and liquidation logic could be verified without full public disclosure. In supply chains, smart contracts could verify supplier performance, quality claims, or carbon data while protecting commercially sensitive process information. In healthcare data exchange, patient data access agreements could be enforced with auditable commitments and privacy-preserving computation. In public procurement, bids and evaluation procedures could be committed, verified, and selectively disclosed after award decisions. These scenarios differ, but they share the need for privacy, correctness, and accountable settlement (Zyskind et al.,2015).

The most demanding future scenario is reactive smart contracts. Non-reactive computation begins with inputs that are available at the start, making trace commitments and replicated execution more straightforward. Reactive contracts receive events over time, interact with external systems, and may change behavior based on asynchronous inputs. For such contracts, the architecture must verify not only computation but also event ordering, input consistency, oracle reliability, and state continuity. This is a major research frontier for converged privacy and verification (Dorri et al.,2017).

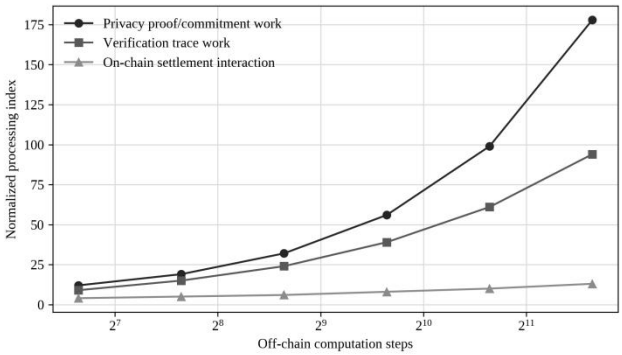

Fig 4. Simulated scaling pattern of privacy, verification, and settlement workload components.

Fig 4 presents a normalized simulation to illustrate how workload components may scale as off-chain computation steps increase. Privacy proof and commitment work may grow faster than settlement interaction because commitments and local evidence are generated off chain, while the settlement layer is used only for summarized state and exceptions. Verification trace work grows with computation complexity but can be moderated by checkpoint granularity. The most important implication is that layer-1 cost alone can understate the total operational cost of privacy-preserving verification (Reyna et al.,2018).

The simulation also suggests why architectural modularity matters. If privacy computation becomes the dominant workload, the system may need specialized proof services or hardware acceleration. If verification trace work dominates, the system may need better checkpointing and storage compression. If settlement interaction dominates, batching and dispute window design become more important. Future-proof architecture should be able to adjust these modules without rewriting the entire smart contract stack (Makhdoom et al.,2019).

VIII. DISCUSSION: TOWARD TRUSTWORTHY SMART CONTRACT INFRASTRUCTURE

The convergence of privacy computing, verifiable computation, and layer-2 protocols points toward a broader transformation in blockchain infrastructure. Early smart contract systems were designed around radical transparency. Later systems introduced scalability through off-chain execution. The next phase will require selective transparency: enough public evidence to make execution accountable, but not so much exposure that private contracts become commercially or legally unusable. Selective transparency is an architectural property, not a single cryptographic primitive (Sharma et al.,2019).

This perspective also changes how protocol success should be understood. A protocol that produces correct output in a single experiment may still be weak if it cannot identify non-performing participants. A protocol that preserves privacy in normal operation may still be weak if dispute resolution exposes sensitive traces. A protocol that has low gas cost may still be weak if governance responsibilities are unclear. Trustworthy smart contracts require a portfolio of assurances: correctness, privacy, incentive alignment, operational resilience, and institutional accountability (Dinh et al.,2018).

The article's architecture has implications for developers. Smart contract developers should describe data sensitivity and verification requirements before choosing a deployment model. They should ask whether acceptance requires evidence of independent work, whether disputes reveal sensitive states, and whether the contract can tolerate delayed finality. They should also consider whether the application requires reactive behavior, cross-chain settlement, or compliance reporting. These choices should be made at design time, not after deployment (Androulaki et al.,2018).

The architecture has implications for researchers as well. Formal security models remain essential, but future research should connect them to empirical cost analysis and governance design. Game-theoretic models should be extended beyond two-party settings, but researchers should recognize that real adversaries may not behave as clean utility maximizers. Privacy models should include dispute-path leakage, metadata leakage, and long-term archival exposure. Evaluation studies should publish not only gas tables but also evidence formats, trace sizes, role assumptions, and implementation constraints (Gencer et al.,2018).

Finally, architecture has implications for regulators and institutional adopters. Public blockchains do not eliminate governance; they relocate it into code, incentives, and protocol roles. A privacy-preserving layer-2 contract may support auditability without exposing confidential data, but only if disclosure rules, evidence retention, and accountability mechanisms are explicit. Institutions adopting such systems should demand audit-ready metadata, role documentation, and clear upgrading procedures. Trustworthy infrastructure is not created by cryptography alone (Bonneau et al.,2015).

IX. GOVERNANCE BLUEPRINT AND IMPLEMENTATION PATHWAY

The proposed architecture becomes practical only when translated into an implementation pathway. A future-oriented smart contract project should begin with a privacy and verification assessment before code is deployed. This assessment should identify contract objects, private attributes, public commitments, authorized execution roles, and exceptional evidence that may be opened during disputes. In many blockchain projects, these decisions are made informally by developers after the protocol is already selected. That practice is risky because it pushes privacy and governance into late-stage patches. A more mature approach treats privacy classification, verification mode, and dispute procedure as part of the original system specification (Eyal et al.,2016).

The first implementation step is workload classification. Contracts should be grouped according to whether they are non-reactive, periodically reactive, or fully event driven. Non-reactive contracts are the easiest to support because inputs are known at the start and trace commitments can be built around a fixed computation. Periodically reactive contracts, such as supply chain milestone payments or recurring settlement agreements, require controlled event windows and repeated checkpointing. Fully event-driven contracts, such as automated market strategies or IoT-triggered insurance claims, require stronger event authenticity and ordering guarantees. Each category should receive a different privacy and verification profile (Kiayias et al.,2017).

The second implementation step is role design. The architecture assumes that users, managers, sequencers, challengers, auditors, and settlement contracts may have different visibility rights. A manager may see private computation logic but not user identity. An auditor may see compliance metadata but not underlying commercial terms. A public observer may see only commitments and final events. These distinctions are important because many real-world failures occur when all participants are implicitly treated as either fully trusted or fully public. Role-specific visibility makes trust assumptions explicit and supports safer system integration (Pass et al.,2017).

The third implementation step is incentive calibration. Deposits and rewards should be linked to the cost of computation, the value of the contract, the probability of dispute, and the damage caused by privacy leakage. A small deposit may be enough for low-value routine transactions but inadequate for high-value confidential settlements. Excessive deposits, however, can exclude smaller participants and reduce decentralization. The architecture therefore recommends adaptive deposits, where protocol parameters can be adjusted by contract class rather than fixed globally (Herlihy,2018).

The fourth implementation step is evidence of lifecycle management. Evidence should have a defined lifecycle: generated during execution, stored by authorized parties, committed on chain, selectively opened during disputes, and retained or deleted according to policy. If evidence is stored indefinitely without governance rules, privacy risk accumulates. If evidence is deleted too early, dispute resolution becomes impossible. A future trustworthy system needs retention rules that are auditable, technically enforceable, and aligned with legal requirements (Decker et al.,2013).

The fifth implementation step is independent monitoring. Even when a protocol has formal guarantees, operational monitoring remains necessary. Dashboards can report dispute frequency, average challenge duration, acceptance proof failure, manager silence, gas volatility, and unusual response timing. These indicators do not prove misconduct by themselves, but they provide early warnings that the incentive environment or implementation may be drifting away from the assumptions used in the protocol model (Daian et al.,2020).

TABLE VI. IMPLEMENTATION PATHWAY FOR TRUSTWORTHY SMART CONTRACT ARCHITECTURE

Step	Main Task	Expected Deliverable
Workload classification	Separate non-reactive, periodic, and event-	Contract execution profile

	driven contracts	
Role design	Define visibility and action rights for each participant	Role-permission matrix
Incentive calibration	Set deposits, rewards, and penalties by contract class	Economic parameter sheet
Evidence lifecycle	Specify generation, storage, opening, retention, and deletion rules	Evidence governance policy
Monitoring deployment	Track disputes, silence, response timing, and gas volatility	Operational assurance dashboard

X. FUTURE RESEARCH AGENDA

Four research directions deserve priority. The first is reactive privacy-preserving verification. As smart contracts become event-driven and connected to AI agents, IoT devices, and external oracles, protocols must verify the correctness of state transitions over time without leaking sensitive event streams. This will require new models for input consistency, event ordering, and adaptive dispute evidence (Qin et al.,2021).

The second direction is AI-assisted protocol monitoring. Machine learning models can analyze transaction timing, acceptance patterns, challenge behavior, deposit movement, and manager response histories to identify possible free-riding, collusion, or strategic delay. Such models should not replace formal verification, but they can complement protocol rules by detecting behavioral anomalies across many contract sessions (Schar,2021).

The third direction is cross-rollup interoperability. As layer-2 ecosystems multiply, smart contracts may need to move commitments, proofs, or dispute evidence across rollups. Standardized evidence formats and privacy-preserving bridges will become important. Without interoperability, each rollup becomes a silo, and trustworthy smart contract architecture remains fragmented (Chen et al.,2020b).

The fourth direction is compliance-aware privacy. Future systems should allow auditors or regulators to verify policy compliance without gaining unrestricted access to private data. This may involve selective disclosure credentials, zero-knowledge compliance proofs, privacy-preserving audit logs, and governance metadata that separates accountability from surveillance. The challenge is to design these mechanisms without undermining decentralization or creating new trusted monopolies (Victor et al.,2021).

TABLE V. FUTURE RESEARCH AGENDA

Research Direction	Key Question	Expected Contribution
Reactive verification	How can event-driven contracts be verified without leaking event streams?	Models for privacy-preserving state continuity
AI-assisted monitoring	How can behavioral data reveal free-riding or collusion?	Anomaly detection for protocol governance
Cross-rollup evidence	How can proofs and commitments move across layer-2 systems?	Interoperable trustworthy smart contract infrastructure
Compliance-aware privacy	How can auditors verify rules without seeing private data?	Selective disclosure and privacy-preserving auditability
Adaptive granularity	How should trace checkpoints be chosen for each workload?	Cost-sensitive dispute resolution
Human-centered governance	How should users understand roles, risks, and remedies?	Usable accountability for institutional adoption

XI. LIMITATIONS AND ETHICAL CONSIDERATIONS

The architecture proposed in this article has limitations. It is intentionally general and therefore does not replace protocol-specific formal analysis. A concrete implementation must still define adversary models, network assumptions, cryptographic primitives, timeout rules, and upgrade procedures in precise terms. The architecture should be understood as a bridge between formal protocol research and deployable smart contract infrastructure, not as a proof that any specific system is secure (Goldreich et al.,1991).

A second limitation is that the data analysis is illustrative. The gas values used to discuss accept and challenge behavior are useful for understanding trade-offs, but production systems may differ in virtual machine design, compiler optimization, transaction batching, proof format, storage pricing, and network congestion. Future empirical studies should evaluate multiple workloads, including financial settlement, supply chain claims, digital identity verification, and IoT-triggered contracts. They should also report off-chain computation and storage costs, not only on-chain gas (Micali,1994).

Ethically, privacy-preserving smart contracts creates a dual-use challenge. The same mechanisms that protect legitimate business confidentiality and personal data can also hide abusive, fraudulent, or illegal activity. The appropriate response is not to

abandon privacy but to design accountable privacy. Selective disclosure, auditable governance metadata, role-based visibility, and legally controlled evidence opening can preserve confidentiality while still supporting investigation and remedy when serious misconduct occurs (Groth,2016).

There is also a fairness issue. If privacy-preserving verification requires expensive hardware, specialized cryptographic expertise, or large deposits, smaller firms and individual users may be excluded from trustworthy smart contract ecosystems. Future architecture should therefore include lightweight modes, open standards, and transparent parameter choices. Trustworthy infrastructure should not become a privilege available only to large institutions with dedicated protocol teams (Kate et al.,2010).

XII. CONCLUSION

This article has proposed a future-oriented architecture for trustworthy smart contracts by converging privacy computing, verifiable computation, and layer-2 blockchain protocols. The starting point is a central tension in public blockchain systems: the need for public verification conflicts with the need for private computation. Layer-2 systems reduce costs by moving computation off chain, but privacy and verification remain difficult to reconcile when dispute resolution, reward rules, and evidence disclosure are not designed together (Bootle et al.,2016).

The proposed architecture responds to this challenge through privacy-by-design, verification-by-default, and settlement-by-exception. It emphasizes accountable acceptance, selective dispute evidence, trace commitments, modular privacy mechanisms, and governance metadata. The risk taxonomy shows that trustworthy smart contracts must address computation integrity, participation integrity, privacy leakage, collusion, operational failure, and compliance risk simultaneously. The data-oriented analysis further illustrates that privacy-by-design can be more efficient than adding privacy to a protocol after its verification logic has already been fixed (Bunz et al.,2018).

The broader implication is that the future of smart contracts will not be defined by a single best protocol. It will be defined by architectures that allow different privacy and verification mechanisms to be combined according to application needs. Finance, supply chains, healthcare, public procurement, and machine-to-machine systems will require different trade-offs, but all will require credible assurance that private off-chain execution remains correct, accountable, and governable. Crossroads of Future Technologies is an appropriate venue for this discussion because the problem sits exactly at the intersection of cryptography, blockchain engineering, data governance, and future digital infrastructure (Ben-Sasson et al.,2019).

ACKNOWLEDGEMENT

Author contributions: All authors contributed to conceptualization, framework design, analysis, writing, and revision. Funding: No external funder applicable. Declarations: The authors declare no conflict of interest. Data availability: The comparative numerical values used in the illustrative analysis are reproduced and reinterpreted from publicly described protocol-evaluation scenarios in the source manuscript; the simulated scaling index in Fig 4 is generated for conceptual illustration (Gennaro et al.,2010).

REFERENCES

Kou, G., & Lu, Y. (2025). FinTech: A literature review of emerging financial technologies and applications. Financial Innovation, 11(1), 1-34. https://doi.org/10.1186/s40854-024-00668-6

Atzei, N., Bartoletti, M., & Cimoli, T. (2017). A survey of attacks on Ethereum smart contracts (SoK). In Principles of Security and Trust (pp. 164-186). Springer. https://doi.org/10.1007/978-3-662-54455-6_8

Lu, Y. (2025). The current status and developing trends of Industry 4.0: A review. Information Systems Frontiers, 27(1), 215-234. https://doi.org/10.1007/s10796-021-10221-w

Nikolic, I., Kolluri, A., Sergey, I., Saxena, P., & Hobor, A. (2018). Finding the greedy, prodigal, and suicidal contracts at scale. Proceedings of the 34th Annual Computer Security Applications Conference, 653-663. https://doi.org/10.1145/3274694.3274743

Wu, H. P., Liu, Z., Dong, H. Y., Lu, Y., & Xu, L. D. (2025). Revolutionizing internal auditing: Harnessing the power of blockchain. Enterprise Information Systems, 19(1-2). https://doi.org/10.1080/17517575.2024.2448003

Tsankov, P., Dan, A., Drachsler-Cohen, D., Gervais, A., Bünzli, F., & Vechev, M. (2018). Securify: Practical security analysis of smart contracts. Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, 67-82. https://doi.org/10.1145/3243734.3243780

Lu, W., Lu, Y., Li, J., Sigov, A., Ratkin, L., & Ivanov, L. A. (2024). Quantum machine learning: Classifications, challenges, and solutions. Journal of Industrial Information Integration, 42, 100736. https://doi.org/10.1016/j.jii.2024.100736

Tikhomirov, S., Voskresenskaya, E., Ivanitskiy, I., Takhaviev, R., Marchenko, E., & Alexandrov, Y. (2018). SmartCheck: Static analysis of Ethereum smart contracts. 2018 IEEE/ACM 1st International Workshop on Emerging Trends in Software Engineering for Blockchain, 9-16. https://doi.org/10.1109/WETSEB.2018.00009

Xu, R., Zhu, J., Yang, L., Lu, Y., & Xu, L. D. (2024). Decentralized finance (DeFi): A paradigm shift in the FinTech. Enterprise Information Systems, 18(9). https://doi.org/10.1080/17517575.2024.2397630

Durieux, T., Ferreira, J. F., Abreu, R., & Cruz, P. (2020). Empirical review of automated analysis tools on 47,587 Ethereum smart contracts. Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering, 530-541. https://doi.org/10.1145/3377811.3380364

Chen, Y., Lu, Y., Bulysheva, L., & Kataev, M. Y. (2024). Applications of blockchain in Industry 4.0: A review. Information Systems Frontiers, 26(5), 1715-1729. https://doi.org/10.1007/s10796-022-10248-7

Kalra, S., Goel, S., Dhawan, M., & Sharma, S. (2018). ZEUS: Analyzing safety of smart contracts. Proceedings of the Network and Distributed System Security Symposium. https://doi.org/10.14722/ndss.2018.23082

Lu, Y., Sigov, A. S., Ratkin, L., Ivanov, L. A., & Zuo, M. (2023). Quantum computing and industrial information integration: A review. Journal of Industrial Information Integration, 35, 100511. https://doi.org/10.1016/j.jii.2023.100511

Rodler, M., Li, W., Karame, G. O., & Davi, L. (2019). Sereum: Protecting existing smart contracts against re-entrancy attacks. Proceedings of the Network and Distributed System Security Symposium. https://doi.org/10.14722/ndss.2019.23413
Ye, Z., & Lu, Y. (2022). Quantum science: A review and current research trends. Journal of Management Analytics, 9(3), 383-402. https://doi.org/10.1080/23270012.2022.2089064
Hildenbrandt, E., Saxena, M., Zhu, X., Rodrigues, N., Daian, P., Guth, D., Moore, B., Park, D., Zhang, Y., Stefanescu, A., & Rosu, G. (2018). KEVM: A complete formal semantics of the Ethereum Virtual Machine. 2018 IEEE 31st Computer Security Foundations Symposium, 204-217. https://doi.org/10.1109/CSF.2018.00022
Lu, Y. (2022). Implementing blockchain in information systems: A review. Enterprise Information Systems, 16(12), 1876-1907. https://doi.org/10.1080/17517575.2021.2008513
Grishchenko, I., Maffei, M., & Schneidewind, C. (2018). A semantic framework for the security analysis of Ethereum smart contracts. Proceedings of the 7th ACM SIGPLAN International Conference on Certified Programs and Proofs, 243-269. https://doi.org/10.1145/3167084
Zheng, X. R., & Lu, Y. (2022). Blockchain technology: Recent research and future trend. Enterprise Information Systems, 16(12), 1939895. https://doi.org/10.1080/17517575.2021.1939895
Torres, C. F., Schütte, J., & State, R. (2018). Osiris: Hunting for integer bugs in Ethereum smart contracts. Proceedings of the 34th Annual Computer Security Applications Conference, 664-676. https://doi.org/10.1145/3274694.3274737
Xu, L. D., Lu, Y., & Li, L. (2021). Embedding blockchain technology into IoT for security: A survey. IEEE Internet of Things Journal, 8(13), 10452-10473. https://doi.org/10.1109/JIOT.2021.3060508
Schneidewind, C., Grishchenko, I., Scherer, M., & Maffei, M. (2020). eThor: Practical and provably sound static analysis of Ethereum smart contracts. Proceedings of the ACM Conference on Computer and Communications Security, 621-640. https://doi.org/10.1145/3372297.3417250
Zhang, C., & Lu, Y. (2021). Study on artificial intelligence: The state of the art and future prospects. Journal of Industrial Information Integration, 23, 100224. https://doi.org/10.1016/j.jii.2021.100224
Mavridou, A., & Laszka, A. (2018). Designing secure Ethereum smart contracts: A finite state machine based approach. Financial Cryptography and Data Security Workshops, 523-540. https://doi.org/10.1007/978-3-662-58820-8_35
Lu, Y., & Zheng, X. (2020). 6G: A survey on technologies, scenarios, challenges, and the related issues. Journal of Industrial Information Integration, 19, 100158. https://doi.org/10.1016/j.jii.2020.100158
Chen, T., Zhu, Y., Li, Z., Chen, J., Li, X., Luo, X., Lin, X., & Zhang, X. (2020). Understanding Ethereum via graph analysis. ACM Transactions on Internet Technology, 20(2), 1-32. https://doi.org/10.1145/3381036
Lu, Y., & Xu, L. D. (2019). Internet of Things (IoT) cybersecurity research: A review of current research topics. IEEE Internet of Things Journal, 6(2), 2103-2115. https://doi.org/10.1109/JIOT.2018.2869847
Bartoletti, M., & Pompianu, L. (2017). An empirical analysis of smart contracts: Platforms, applications, and design patterns. Financial Cryptography and Data Security Workshops, 494-509. https://doi.org/10.1007/978-3-319-70278-0_31
Lu, Y. (2019). The blockchain: State-of-the-art and research challenges. Journal of Industrial Information Integration, 15, 80-90. https://doi.org/10.1016/j.jii.2019.04.002
Destefanis, G., Marchesi, M., Ortu, M., Tonelli, R., Bracciali, A., & Hierons, R. (2018). Smart contracts vulnerabilities: A call for blockchain software engineering? 2018 International Workshop on Blockchain Oriented Software Engineering, 19-25. https://doi.org/10.1109/IWBOSE.2018.8327567
Lu, Y. (2018). Blockchain and the related issues: A review of current research topics. Journal of Management Analytics, 5(4), 231-255. https://doi.org/10.1080/23270012.2018.1516523
Yli-Huumo, J., Ko, D., Choi, S., Park, S., & Smolander, K. (2016). Where is current research on blockchain technology? A systematic review. PLOS ONE, 11(10), e0163477. https://doi.org/10.1371/journal.pone.0163477
Lu, Y. (2017). Industry 4.0: A survey on technologies, applications and open research issues. Journal of Industrial Information Integration, 6, 1-10. https://doi.org/10.1016/j.jii.2017.04.005
Casino, F., Dasaklis, T. K., & Patsakis, C. (2019). A systematic literature review of blockchain-based applications: Current status, classification and open issues. Telematics and Informatics, 36, 55-81. https://doi.org/10.1016/j.tele.2018.11.006
Saberi, S., Kouhizadeh, M., Sarkis, J., & Shen, L. (2019). Blockchain technology and its relationships to sustainable supply chain management. International Journal of Production Research, 57(7), 2117-2135. https://doi.org/10.1080/00207543.2018.1533261
Kshetri, N. (2018). Blockchain's roles in meeting key supply chain management objectives. International Journal of Information Management, 39, 80-89. https://doi.org/10.1016/j.ijinfomgt.2017.12.005
Li, J., Maiti, A., Springer, M., & Gray, T. (2020). Blockchain for supply chain quality management: Challenges and opportunities in context of open manufacturing and industrial Internet of Things. International Journal of Computer Integrated Manufacturing, 33(12), 1321-1355. https://doi.org/10.1080/0951192X.2020.1801705
Queiroz, M. M., Telles, R., & Bonilla, S. H. (2020). Blockchain and supply chain management integration: A systematic review of the literature. Supply Chain Management, 25(2), 241-254. https://doi.org/10.1108/SCM-03-2018-0143
Azzi, R., Chamoun, R. K., & Sokhn, M. (2019). The power of a blockchain-based supply chain. Computers & Industrial Engineering, 135, 582-592. https://doi.org/10.1016/j.cie.2019.06.042
Mendling, J., Weber, I., van der Aalst, W., vom Brocke, J., Cabanillas, C., Daniel, F., Debois, S., Di Ciccio, C., Dumas, M., Dustdar, S., Gal, A., Garcia-Banuelos, L., Governatori, G., Hull, R., La Rosa, M., Leopold, H., Leymann, F., Recker, J., Reichert, M., Reijers, H. A., Rinderle-Ma, S., Solti, A., Rosemann, M., Schulte, S., Singh, M. P., Slaats, T., Staples, M., Weber, B., Weidlich, M., Weske, M., Xu, X., & Zhu, L. (2018). Blockchains for business process management: Challenges and opportunities. ACM Transactions on Management Information Systems, 9(1), 1-16. https://doi.org/10.1145/3183367
Christidis, K., & Devetsikiotis, M. (2016). Blockchains and smart contracts for the Internet of Things. IEEE Access, 4, 2292-2303. https://doi.org/10.1109/ACCESS.2016.2566339
Novo, O. (2018). Blockchain meets IoT: An architecture for scalable access management in IoT. IEEE Internet of Things Journal, 5(2), 1184-1195. https://doi.org/10.1109/JIOT.2018.2812239
Ouaddah, A., Abou Elkalam, A., & Ait Ouahman, A. (2017). Towards a novel privacy-preserving access control model based on blockchain technology in IoT. Europe and MENA Cooperation Advances in Information and Communication Technologies, 523-533. https://doi.org/10.1007/978-3-319-46568-5_53
Zyskind, G., Nathan, O., & Pentland, A. (2015). Decentralizing privacy: Using blockchain to protect personal data. 2015 IEEE Security and Privacy Workshops, 180-184. https://doi.org/10.1109/SPW.2015.27
Dorri, A., Kanhere, S. S., & Jurdak, R. (2017). Blockchain in Internet of Things: Challenges and solutions. Computer, 50(6), 28-37. https://doi.org/10.1109/MC.2017.195
Reyna, A., Martin, C., Chen, J., Soler, E., & Diaz, M. (2018). On blockchain and its integration with IoT: Challenges and opportunities. Future Generation Computer Systems, 88, 173-190. https://doi.org/10.1016/j.future.2018.05.046
Makhdoom, I., Abolhasan, M., Abbas, H., & Ni, W. (2019). Blockchain's adoption in IoT: The challenges, and a way forward. Journal of Network and Computer Applications, 125, 251-279. https://doi.org/10.1016/j.jnca.2018.10.019
Sharma, P. K., Chen, M. Y., & Park, J. H. (2019). A software defined fog node based distributed blockchain cloud architecture for IoT. IEEE Access, 6, 115-124. https://doi.org/10.1109/ACCESS.2017.2757955

Dinh, T. T. A., Wang, J., Chen, G., Liu, R., Ooi, B. C., & Tan, K. L. (2018). BLOCKBENCH: A framework for analyzing private blockchains. Proceedings of the 2017 ACM International Conference on Management of Data, 1085-1100. https://doi.org/10.1145/3035918.3064033

Androulaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., De Caro, A., Enyeart, D., Ferris, C., Laventman, G., Manevich, Y., Muralidharan, S., Murthy, C., Nguyen, B., Sethi, M., Singh, G., Smith, K., Sorniotti, A., Stathakopoulou, C., Vukolic, M., Cocco, S. W., & Yellick, J. (2018). Hyperledger Fabric: A distributed operating system for permissioned blockchains. Proceedings of the Thirteenth EuroSys Conference, 1-15. https://doi.org/10.1145/3190508.3190538

Gencer, A. E., Basu, S., Eyal, I., van Renesse, R., & Sirer, E. G. (2018). Decentralization in Bitcoin and Ethereum networks. Financial Cryptography and Data Security, 439-457. https://doi.org/10.1007/978-3-662-58387-6_24

Bonneau, J., Miller, A., Clark, J., Narayanan, A., Kroll, J. A., & Felten, E. W. (2015). SoK: Research perspectives and challenges for Bitcoin and cryptocurrencies. 2015 IEEE Symposium on Security and Privacy, 104-121. https://doi.org/10.1109/SP.2015.14

Eyal, I., Gencer, A. E., Sirer, E. G., & van Renesse, R. (2016). Bitcoin-NG: A scalable blockchain protocol. 13th USENIX Symposium on Networked Systems Design and Implementation, 45-59. https://doi.org/10.1145/3011540.3011541

Kiayias, A., Russell, A., David, B., & Oliynykov, R. (2017). Ouroboros: A provably secure proof-of-stake blockchain protocol. Advances in Cryptology - CRYPTO 2017, 357-388. https://doi.org/10.1007/978-3-319-63688-7_12

Pass, R., & Shi, E. (2017). Hybrid consensus: Efficient consensus in the permissionless model. 31st International Symposium on Distributed Computing, 39:1-39:16. https://doi.org/10.4230/LIPIcs.DISC.2017.39

Herlihy, M. (2018). Atomic cross-chain swaps. Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing, 245-254. https://doi.org/10.1145/3212734.3212736

Decker, C., & Wattenhofer, R. (2013). Information propagation in the Bitcoin network. 2013 IEEE Thirteenth International Conference on Peer-to-Peer Computing, 1-10. https://doi.org/10.1109/P2P.2013.6688704

Daian, P., Goldfeder, S., Kell, T., Li, Y., Zhao, X., Bentov, I., Breidenbach, L., & Juels, A. (2020). Flash boys 2.0: Frontrunning in decentralized exchanges, miner extractable value, and consensus instability. 2020 IEEE Symposium on Security and Privacy, 910-927. https://doi.org/10.1109/SP40000.2020.00040

Qin, K., Zhou, L., Afonin, Y., Lazzaretti, L., & Gervais, A. (2021). CeFi vs. DeFi: Comparing centralized to decentralized finance. 2021 IEEE Symposium on Security and Privacy Workshops, 1-8. https://doi.org/10.1109/SPW53761.2021.00001

Schär, F. (2021). Decentralized finance: On blockchain- and smart contract-based financial markets. Federal Reserve Bank of St. Louis Review, 103(2), 153-174. https://doi.org/10.20955/r.103.153-74

Chen, W., Zheng, Z., Cui, J., Ngai, E., Zheng, P., & Zhou, Y. (2020). Detecting Ponzi schemes on Ethereum: Towards healthier blockchain technology. World Wide Web, 23, 1409-1431. https://doi.org/10.1007/s11280-019-00756-7

Victor, F., & Lüders, B. K. (2021). Measuring Ethereum-based ERC20 token networks. Financial Cryptography and Data Security, 113-129. https://doi.org/10.1007/978-3-662-64331-0_7

Goldreich, O., Micali, S., & Wigderson, A. (1991). Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. Journal of the ACM, 38(3), 690-728. https://doi.org/10.1145/116825.116852

Micali, S. (1994). CS proofs. Proceedings of the 35th Annual Symposium on Foundations of Computer Science, 436-453. https://doi.org/10.1109/SFCS.1994.365746

Groth, J. (2016). On the size of pairing-based non-interactive arguments. Advances in Cryptology - EUROCRYPT 2016, 305-326. https://doi.org/10.1007/978-3-662-49896-5_11

Kate, A., Zaverucha, G. M., & Goldberg, I. (2010). Constant-size commitments to polynomials and their applications. Advances in Cryptology - ASIACRYPT 2010, 177-194. https://doi.org/10.1007/978-3-642-17373-8_11

Bootle, J., Cerulli, A., Chaidos, P., Groth, J., & Petit, C. (2016). Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. Advances in Cryptology - EUROCRYPT 2016, 327-357. https://doi.org/10.1007/978-3-662-49890-3_18

Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., & Maxwell, G. (2018). Bulletproofs: Short proofs for confidential transactions and more. 2018 IEEE Symposium on Security and Privacy, 315-334. https://doi.org/10.1109/SP.2018.00020

Ben-Sasson, E., Bentov, I., Horesh, Y., & Riabzev, M. (2019). Scalable, transparent, and post-quantum secure computational integrity. IACR Cryptology ePrint Archive; also in TCC 2018. https://doi.org/10.1007/978-3-030-17653-2_23

Gennaro, R., Gentry, C., & Parno, B. (2010). Non-interactive verifiable computing: Outsourcing computation to untrusted workers. Advances in Cryptology - CRYPTO 2010, 465-482. https://doi.org/10.1007/978-3-642-14623-7_25

Fiore, D., Gennaro, R., & Pastro, V. (2014). Efficiently verifiable computation on encrypted data. Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, 844-855. https://doi.org/10.1145/2660267.2660366

Walfish, M., & Blumberg, A. J. (2015). Verifying computations without reexecuting them. Communications of the ACM, 58(2), 74-84. https://doi.org/10.1145/2641562

Setty, S., McPherson, R., Blumberg, A. J., & Walfish, M. (2012). Making argument systems for outsourced computation practical. 19th Annual Network and Distributed System Security Symposium. https://doi.org/10.14722/ndss.2012.23106

Thaler, J. (2013). Time-optimal interactive proofs for circuit evaluation. Advances in Cryptology - CRYPTO 2013, 71-89. https://doi.org/10.1007/978-3-642-40041-4_5

Parno, B., Howell, J., Gentry, C., & Raykova, M. (2013). Pinocchio: Nearly practical verifiable computation. 2013 IEEE Symposium on Security and Privacy, 238-252. https://doi.org/10.1109/SP.2013.47

Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., Ramage, D., Segal, A., & Seth, K. (2017). Practical secure aggregation for privacy-preserving machine learning. Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, 1175-1191. https://doi.org/10.1145/3133956.3133982

Acar, A., Aksu, H., Uluagac, A. S., & Conti, M. (2018). A survey on homomorphic encryption schemes: Theory and implementation. ACM Computing Surveys, 51(4), 1-35. https://doi.org/10.1145/3214303

Gentry, C. (2009). Fully homomorphic encryption using ideal lattices. Proceedings of the 41st Annual ACM Symposium on Theory of Computing, 169-178. https://doi.org/10.1145/1536414.1536440

Dwork, C. (2006). Differential privacy. Automata, Languages and Programming, 1-12. https://doi.org/10.1007/11787006_1

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., D'Oliveira, R. G. L., Eichner, H., El Rouayheb, S., Evans, D., Gardner, J., Garrett, Z., Gascón, A., Ghazi, B., Gibbons, P. B., Gruteser, M., Harchaoui, Z., He, C., Huo, Z., Hutchinson, B., Hsu, J., Jaggi, M., Javidi, T., Konečný, J., Korolova, A., Koushanfar, F., Lepoint, T., Liu, Y., Mittal, P., Mohri, M., Nock, R., Özgür, A., Pagh, R., Qi, H., Ramage, D., Raskar, R., Song, D., Stich, S. U., Sun, Z., Suresh, A. T., Tramèr, F., Vepakomma, P., Wang, J., Xiong, L., Xu, Z., Yang, Q., Yu, H., & Zhao, S. (2021). Advances and open problems in federated learning. Foundations and Trends in Machine Learning, 14(1-2), 1-210. https://doi.org/10.1561/2200000083