

Research Article

Fault Management in MVDC Grids Using Blockchain and Smart Contracts: A Decentralization Approach

Ramanarayana Nandigam^{1,2*} , Ganesan Sivarajan³ , Veera Kumari Maddipati² ,
Subramanian Srikrishna⁴ 

¹Department of Electrical and Electronics Engineering, Annamalai University, Tamil Nadu, India

²Department of Electrical & Electronics Engineering, Sir C.R. Reddy College of Engineering, Eluru, India.

³Department of EEE, Government College of Engineering, Salem, Tamil Nadu, India.

⁴Department of Electrical Engineering, Annamalai University, Tamil Nadu, India.

*ramanarayana1@gmail.com

Abstract

Medium-voltage direct-current (MVDC) networks facilitate the integration of renewable generation, energy storage, and power-electronic converters, but their protection requires rapid local fault response and reliable coordination among distributed components. This paper proposes a proof-of-concept decentralized fault-management architecture that separates local anomaly detection from permissioned-blockchain-based coordination and auditing governed by a smart-contract state machine. Beyond an initial paired 50-run threshold study, a fixed-seed Monte Carlo stress test involving 2,000 events evaluates threshold uncertainty, temporary voltage sags, high-impedance-fault surrogates, sensor errors, reported-data manipulation, and 5% message loss. At thresholds of 0.6, 0.7, and 0.8 pu, the measured sensitivities were 61.73%, 77.05%, and 91.36%, respectively, while the corresponding false-positive rates were 0%, 0%, and 5.49%. Under a common logical benchmark, the blockchain-quorum coordination path achieved 91.85% accuracy and 89.59% sensitivity, with a modeled coordination latency of 1.362 ± 0.185 s. These latency results indicate that blockchain-based coordination is unsuitable as a replacement for primary high-speed DC protection. Additional quorum-delivery and message-complexity models quantify the effects of validator count and packet loss. Overall, the results demonstrate the architectural feasibility of blockchain-assisted MVDC coordination, including traceable state sequencing and reproducible logical evaluation, within the assumptions of the proposed models. Electromagnetic-transient (EMT) simulation, converter and breaker modeling, network emulation, and hardware-in-the-loop (HIL) testing are required before field deployment.

Keywords: MVDC Networks; Decentralized Fault Detection; Blockchain-Based Power Systems; Smart Contract Automation; Consensus Mechanisms.

INTRODUCTION

Medium voltage direct current (MVDC) distribution systems are growing in importance due to their ability to integrate renewable energy sources, battery energy storage systems, power-electronic converters with lower number of conversion stages, lower conversion losses, no reactive power flow in the DC part, and easy connection of asynchronous energy sources. The properties have driven the adoption of MVDC in ships, offshore wind collection, urban distribution, industrial systems, and data centres [1-4].

Despite all these benefits, MVDC protection is still a challenge. Fast and topology dependent fault transients are caused by DC-link capacitor discharge, low cable impedance, converter current limiting and missing natural current zero crossing. Protection, then, must include local detection and selective breaker action/distance action in addition to communication-assisted protection, as these physical requirements have been recognized and addressed in published methods [1-4].

Local relays, line protection, converter blocking, distributed agents, and supervisory monitoring and coordination with SCADA are used to provide MVDC protection [1-5]. This study examines if identified validators can come to a consensus on the event record and command sequence following the local trip decision, while not putting ledger finality on the trip path. The study in [6] examined decentralized power control and a tiered communication infrastructure for autonomous distribution networks, providing background for separating local decisions from wider coordination.

Each identified validator maintains a copy of an ordered event record. The framework uses a permissioned ledger rather than the proof-of-work network described by Nakamoto [7]. Signatures, state guards, and replicated records allow an operator to reconstruct who approved an action and when.

After a local detector has detected a candidate event, the smart contract verifies the signed record against the coordination rules [8-10].

Accordingly, this work evaluates a blockchain-assisted MVDC coordination layer implemented in MATLAB App Designer and Ganache. The local path produces the anomaly flag, while the ledger validates, orders, and records coordination events. This separation is essential because the modeled 1-2 s blockchain workflow is unsuitable for primary MVDC fault interruption.

Background and Literature Review

To investigate the impact on system performance and transaction finality, which are crucial for real-time protection applications, Agung and Handayani [11] investigated different Blockchain consensus algorithms. The study in [12] suggested the Blockchain as a secure and decentralized cyber infrastructure for modern power systems. Authors in [13] reviewed the use of Blockchain in power networks, concluding that permissionless Blockchains with heavy consensus protocols were more appropriate for systems with a secure operation, whereas permissioned Blockchains with lightweight consensus protocols would be more suitable for fault management due to the low communication delay

required for a protection system. Likewise, [14] investigated the Blockchain applications in energy trading, microgrid control, and integration with EVs, and noted that fault detection and protection in electric grids using the Blockchain have been studied only in a limited way.

In terms of fault management in Blockchain, authors in [15] proposed an IoT-based Blockchain framework for fault traceability, which is able to trace the real source of fault transparently. Likewise, [16] presented a cyber-protection system for a medium-voltage AC distribution network using the Blockchain and proved that the distributed ledger can meet the communication speed demanded for the protection of distribution network. But both of these studies were based on AC networks and studied the specific protection problems associated with MVDC networks are not taken into account – namely converter blocking, the very short time to rise of DC faults, and the coordination of solid-state circuit breakers.

State-of-the-art MVDC protection approaches include communication-assisted directional overcurrent protection and fault isolation [1], traveling-wave pilot protection [2], frequency-based current differential protection [3], and distance protection [4]. Multi-agent protection schemes have also been demonstrated in DC microgrids [5]. These approaches primarily address physical fault detection, fault discrimination, selectivity, and isolation, providing an appropriate technical benchmark for the present study. In contrast, the proposed ledger layer is complementary to conventional protection: it supports authenticated event recording, coordination-state verification, and traceability without introducing a new relay principle or claiming faster physical fault clearance.

Authors in [17] studied blockchain mechanisms for trust and resistance to record counterfeiting. In [18] reviewed privacy, encryption, and smart-contract mechanisms for mutual trust in smart-grid transactions.

Related blockchain studies address demand response, energy trading, governance, and cyber-security [12, 14, 19-29]. They demonstrate programmable multi-party coordination, but their timing and objectives differ from fault interruption. Therefore, the present comparison separates blockchain literature from primary MVDC protection studies [1-5] and avoids transferring energy-trading claims directly to protection performance. Authors in [30] reviewed blockchain-based transactive energy and proposed a cyber-physical framework, while [31] examined blockchain and machine-learning integration for peer-to-peer energy trading. The study in [32] surveyed blockchain-enabled trading between electric vehicles and smart grids, and [33] applied blockchain with multi-agent deep reinforcement learning to demand response. Authors in [34] combined blockchain and public-key infrastructure for secure vehicle-to-vehicle energy trading. These studies provide context for distributed coordination and authenticated exchanges; they do not establish MVDC fault-clearing performance.

Structured SOTA Comparison

Table 1 compares representative studies by application, validation role, fault-management function, smart-contract role, and limitation. Most blockchain studies

address trading, security, or traceability; fewer combine local detection, state-based isolation, recovery, timing analysis, and threshold sensitivity in an MVDC proof of concept.

Table 1. Structured Comparison of Representative State-of-the-Art Studies

Study	Grid / application	Ledger or validation role	Fault-management role	Smart-contract role	Main limitation relative to this study
[11]	General smart-grid review	Survey of blockchain and consensus	No implemented MVDC fault workflow	Discussed broadly	No MVDC model or quantitative fault validation
[16]	Medium-voltage AC feeder	Distributed-ledger cyber grid guard	Faulted-phase detection	Not used for full trip-reclose sequence	AC-focused; no MVDC isolation or recovery model
[15]	Power-grid fault traceability	IoT and blockchain event traceability	Post-event fault tracing	Limited protection automation	No protection-grade MVDC timing or breaker sequence
[8]	Smart-grid security	Decentralized authentication and access control	Secures control commands	Used for security enforcement	No MVDC fault dynamics or threshold study
[35]	Smart-grid cyber security	AI with blockchain integrity	Anomaly recognition	Not focused on MVDC protection states	No MVDC isolation and recovery evaluation
[12, 13]	General energy-system architecture	Blockchain design and cyber infrastructure	Conceptual support	General programmable automation	No application-level MVDC fault experiment
Proposed framework	Ten-node MVDC logical testbed	Permissioned ledger with authenticated, ordered event validation	Local detection, adjacency-based isolation, and supervised recovery	Finite-state protection workflow	Proof-of-concept only; no EMT or HIL validation

Gap-to-Solution Mapping

Table 2 maps the identified research gaps to framework elements, simulation evidence, and remaining limitations.

Table 2. Gap-to-Solution Mapping for the Proposed MVDC Fault-Management Framework

Identified gap	Proposed response	Evidence in this manuscript	Remaining limitation
Limited MVDC-specific blockchain protection studies	Combine local MVDC detection, permissioned validation, and smart-contract state control	Ten-node logical model and implemented event workflow	No EMT-level converter and cable dynamics
Weak timing analysis between protection and blockchain	Separate the end-to-end delay into detection, communication, consensus, contract, and isolation terms	Analytical latency equation and reported simulation timing	No real-time network or breaker benchmark
Smart contracts mainly used for trading or access control	Encode MONITORING, FAULT_DETECTED, ISOLATED, and RECOVERED states	State-transition logic and ledger sequence	No formal verification of the contract
Limited threshold sensitivity evidence	Test 0.6, 0.7, and 0.8 pu over 50 runs each	Detection, missed-fault, false-alarm, and 95% interval results	Larger noisy and high-impedance datasets are required
Weak reproducibility reporting	Provide core topology, threshold, fault, and software assumptions	Reproducibility table and metric definitions	Public code and fixed-seed package remain future work

Emerging Themes: Security, Intelligence, and Automation

If the fault management system malfunctions and fails to detect true trip indications, or issues false protection signals, the resulting damage may be worse than the damage caused by the actual fault. In order to enhance the security of power system protection, [8] presented a smart contract-based approach to prevent unauthorized command injection into the power system control systems, which can be applied in protecting the control signal of MVDC. Authors in [35] used a machine learning approach for anomaly detection and Blockchain technology to differentiate between real and misguided data manipulation. The study in [36] proposed a federated learning integrated with Blockchain for providing secure and privacy-preserving protection to the DERs that is applicable to multi-terminal MVDC systems where different operators may not be willing to share raw measurement data. The study in [37] presented a federated Blockchain architecture for intrusion detection in power grids, whereas [38] analyzed the techniques of Blockchain and IoT in cybersecurity that can be adopted to sensor-rich MVDC networks. Authors in [39] also explored anomaly detection techniques in Blockchain systems to further enhance system

security. In a related cyber-physical application, [40] integrated blockchain, IoT communication, and smart contracts for autonomous vehicles. This offers an example of software and communication integration, although its navigation application does not validate electrical protection timing.

Blockchain has been applied to distributed access control [41] and to preserving data privacy through distributive frameworks [42]. It has also been used to detect fake articles in public media through proof of trustworthiness [43] and for food supply chain traceability with micro-investment [44]. Beyond blockchain, recent work on adaptive federated intrusion detection for energy-constrained 6G IoT cyber-physical systems addresses the security of edge-centric environments [45].

The ledger functions evaluated in this document are explicitly restricted to submitting events and receiving authentication, performing quorum-controlled state transitions, logging events chronologically, and maintaining auditability. These functions can enhance coordination evidence but this research does not suggest that the use of ledgers equates to improved electrical reliability, fault selectivity, or clearing speed.

In the current proof-of-concept, blockchain is utilized to confirm and put related events in an ordered list, and the initial anomaly decision is made locally. Smart contracts then facilitate a deterministic coordination of the simulated isolation and recovery states. The separation does not rely on any assumptions about how blockchain consensus will be able to disable a fast-developing DC fault—it only asks if distributed validation and tamper-evident logging can be used to augment local MVDC protection. The architecture has been built in MATLAB App Designer and integrated with a simulated logical test environment using the Ganache platform.

Objectives and Scope of the Present Work

The main objectives of this paper are:

- To design a permissioned blockchain coordination network with identified validators, deterministic state transitions, quantified quorum-delivery probability, and explicit separation from the trusted local trip path.
- To validate the proposed framework by simulating representative MVDC fault scenarios, and compare the simulation results with a conventional centralized protection system, using a MATLAB App Designer application integrated with Ganache Blockchain platform.

The workflow continuously monitors the logical MVDC network, creates candidate events at the affected node, validates signed event summaries through a permissioned quorum, applies guarded smart-contract transitions, and presents the resulting status and audit trail in real time.

Novelty Statement and Research Hypotheses

The novelty is not in the new physical relay principle, or the new consensus protocol, or the complete electromagnetic-transient protection model. It is the MVDC-specific local threshold detection, authenticated permissioned-ledger validation, finite-state smart-

contract action, auditable event sequencing, latency decomposition and threshold sensitivity analysis in one reproducible proof-of-concept workflow.

- H1: Under the same logical simulation hypotheses, under the same processing hypothesis, the measurement processing time will be lower at the node level, compared to an architecture that sends all measurements to a central decision server.
- H2: With seven identified validators, a five-vote quorum, and independent 5% message loss, at least 99% of candidate events will obtain a quorum in one modeled round; accepted transitions will satisfy the contract guard; and mean modeled coordination latency will remain below 1.5 s. Any failed condition rejects H2 for this test configuration.
- H3: The proportion of faults detected by the selected voltage threshold will be significantly affected by the sensitivity data set containing 50 runs of paired faults, and the false alarm count will be unaffected for the 50 runs of no-fault data set tested.

H1 and H3 are assessed with software timing and paired classification outcomes. H2 uses the stated quorum, state-guard, and modeled latency criteria. Its assessment remains conditional on the logical loss and timing assumptions; it does not establish behavior under correlated outages or a field implementation.

Open Challenges and Gaps Addressed

Based on literature search, the following gaps in the existing knowledge are identified:

- Gap 1: Protection strategies for existing Blockchain and smart contract-based power systems are primarily designed for AC power systems. The unique protection requirements of MVDC systems, such as converter blocking, solid state circuit breaker coordination and capacitor discharge protection during DC fault [13, 14] have not been addressed in studies like [16] and [15].
- Gap 2: This gap can be addressed by several points. Firstly, the previous study [11] and [13] have already highlighted that protection applications do not require such large communication delays as energy trading applications, but they do not give a quantitative estimation of the latency of the Blockchain consensus with respect to the strict fault-clearing time requirements of MVDC systems. Further, the effect of fault events from multiple distributed converter stations on the performance of Blockchain consensus has not been studied [11-13].
- Gap 3: Most smart contracts currently in existence are applied to specific applications like energy trading, access control, and transaction management [9, 10, 17]. They offer no protection specific functions, including fail safe operating states, converter protection, circuit breaker tripping/reclosing sequences, and fault detection, which are critical for reliable MVDC fault management [8-10].

The contribution is an integration that can be inspected and rerun: a local MVDC decision is linked to an authenticated event; validators apply a stated quorum and a

guarded state machine; timing is separated into local and coordination stages; and the study measures threshold response, independent message loss, and validator scaling. It does not introduce an electrical relay, a consensus protocol, or an EMT model [1-5].

Scientific Advancement of the Integrated Architecture

The local detector, validators, and contract have distinct duties. Detection takes place before any ledger exchange and remains available if that exchange fails. Validators then approve and order the event, and the contract restricts later isolation and recovery records to permitted transitions. The paper examines these boundaries through local timing, quorum delivery, state guards, and the resulting event history.

The scientific value is therefore not a faster relay characteristic. It is a reproducible method for measuring the cyber-physical trade-off introduced when independently administered MVDC agents must agree on a common event record. The same event set exposes both the benefit controlled, auditable coordination and the cost quadratic message growth and second-scale finality in the present Ganache implementation. This integration boundary is the contribution that is absent from the primary-protection studies in [1-5].

ARCHITECTURE OF THE PROPOSED DECENTRALIZED FAULT MANAGEMENT FRAMEWORK

This section presents the proposed blockchain-assisted decentralized fault-management framework for MVDC distribution networks. Four functional layers separate physical measurements, local anomaly detection, permissioned validation, and supervised isolation/recovery coordination. A trusted local relay or converter controller remains authorized to initiate immediate protection independently of blockchain availability.

System Overview and Functional Decomposition

Each logical network node has a monitoring agent that observes local voltage and current. When the local rule identifies an anomaly, the agent signs a compact event summary and submits it to identified validators. After quorum approval, the smart-contract state machine records and coordinates the simulated isolation or recovery action. Raw high-rate waveforms remain off-chain; only event summaries and state changes are committed.

The proposed system is comprised of four functional layers, which enables secure and reliable fault management.

Physical Grid Layer

This layer is a MVDC distribution network, composed of 10 nodes which are interconnected in a random symmetric topology. Every node communicates with its adjacent nodes via specific communication links, and also monitors the local bus voltage in per-unit (pu) with a nominal voltage of 1.0 pu. Adjacency matrix establishes both the electrical connection and the communication path, so that if a faulty part is isolated, the healthy parts are able to communicate with each other and share information.

Sensing and Event Generation Layer

The voltage and current sensors are installed at each node to constantly track the working conditions. When the system is operating normally, the measured values are within the range of operation. If a fault condition is detected (e.g. bus voltage less than 0.7 pu), the local monitoring unit detects the fault and stores the time of fault detection, as well as creating a fault report. The report contains node ID, voltage, current, fault type, time stamp and a cryptographic signature for authenticity of the data.

Blockchain Consensus and Smart Contract Layer

The fault report created is sent to a permissioned Blockchain that is known to all participating nodes. The transaction is verified by a lightweight consensus mechanism that filters out duplicated or outdated or unauthorized transactions. After successful validation, a smart contract automatically executes the protection logic. It detects the fault, identifies the section of the network impacted, sends circuit breaker commands, updates the status of the system, and permanently records all fault details in the Blockchain ledger.

Supervisory Monitoring and Recovery Layer

A GUI developed with MATLAB App Designer is used to monitor the whole system in real time. There are four panels on the interface: Network Topology, Fault Status, Blockchain Ledger, and System Logs. These panels provide insight into the status of every node, active fault locations, validated Blockchain transactions as well as time-stamped system events. After the fault is cleared, the isolated section can be restored automatically after a set period of time or by the operator. The smart contract checks that the fault has been cleared and then issues simulated reclose commands to the necessary circuit breaker(s) to resume normal operation before reconnecting the network.

Network Topology and Node Interactions

The MVDC distribution network is represented by an undirected graph ($G = (V, E)$), where the vertices are the converter stations or load buses, and the edges represent the electrical connections between the two. Such connections are represented by a symmetric adjacency matrix. In addition to representing the network topology, the adjacency matrix also controls how fault information is exchanged between neighbouring nodes. If a fault is detected in a node, those node's adjacent nodes only are involved in the fault isolation process. This way, the fault is contained in the affected part of the network, while healthy parts remain unaffected and continue to function.

Formal Network, Detection, Validation, and Latency Model

Table 3 defines the ten-node network, local detector, validation guard, and latency model in equations (1) until (8). Equation (1) specifies the graph and node count, equation (2) defines the undirected adjacency matrix. Equation (3) gives only a simplified resistive-fault interpretation: physical voltage in volts divided by resistance in ohms gives current in amperes. The App Designer model imposes voltage depressions and does not solve converter or cable transients.

Table 3. Mathematical Representation of Fault Management Logic

Expression		Meaning
$G = (\mathcal{V}, \mathcal{E}), \quad \mathcal{V} = 10$	(1)	The graph uses distinct node and edge sets; the model contains 10 converter/load nodes.
$A = [a_{ij}], \quad a_{ij} = a_{ji} \in \{0, 1\}, \quad a_{ii} = 0$	(2)	The symmetric adjacency matrix defines neighbouring nodes and the isolation boundary.
$I_{f,i} \approx \frac{V_i}{R_{f,i}}$	(3)	Under the simplified local-bus resistive-fault approximation, the fault-current component is approximately the local bus voltage divided by the fault resistance; source, line, converter, and transient effects are not represented by this relation.
$F_i(k) = \begin{cases} 1, & V_i(k) < \theta, \\ 0, & \text{otherwise,} \end{cases}$	(4)	The local detector flags node i when its per-unit voltage falls below the selected threshold.
$B_i = \{j \in \mathcal{V}: a_{ij} = 1\}$	(5)	The isolation-boundary set contains the nodes directly connected to faulted node i ; the associated switching devices define the physical isolation boundary.
$\mathcal{A}_i(k) = \text{Auth}_i \wedge \text{Fresh}_i \wedge \text{Unique}_i$	(6)	The indicator equals 1 only when authentication, freshness, uniqueness, and the quorum condition are all satisfied; otherwise, it equals 0.
$C_i(k) = 1\{\mathcal{A}_i(k) \wedge \sum_{j=1}^n v_{ij}(k) \geq q\}$	(7)	
$T_{\text{total}} = T_{\text{det}} + T_{\text{comm}} + T_{\text{cons}} + T_{\text{contract}} + T_{\text{iso}}$	(8)	End-to-end response time is the sum of local detection, communication, consensus, contract execution, and isolation time.
MONITORING $\rightarrow$ FAULT_DETECTED $\rightarrow$ ISOLATED $\rightarrow$ RECOVERED		The finite-state sequence prevents isolation or recovery actions from being executed out of order.

In equations (1) until (5), G is the graph; V and E are its node and edge sets; $|V|$ is the number of nodes; $A = \{a_{ij}\}$ is the adjacency matrix; and $a_{ij} = 1$ denotes a connection between nodes i and j . The diagonal $a_{ii} = 0$ excludes self-links. $I_{f,i}$ is the approximate fault current, $R_{f,i}$ is fault resistance, and V_i in equation (3) is physical bus voltage. In equation (4), $V_i(k)$ is per-unit voltage at sample k , θ is the selected threshold, and $F_i(k)$ is 1 for a threshold violation and 0 otherwise. Equation (5) defines B_i as the set of neighbours of node i used by the isolation rule.

In equations (6) and (7), $\mathcal{A}_i(k)$ is the event-admissibility flag; Auth_i , Fresh_i , and Unique_i are Boolean checks for an authorized source, a current timestamp, and absence of duplication. The symbol $\wedge$ denotes logical AND, and $1\{\cdot\}$ is the indicator of a true condition. $C_i(k)$ is the approval flag, $v_{ij}(k) \in \{0, 1\}$ is validator j 's vote for node i , n is the validator count, and q is the minimum approval count. At least q votes are required. In equation (8), T_{total} is the sum of detection T_{det} , communication T_{comm} , consensus T_{cons} , contract execution T_{contract} , and simulated isolation T_{iso} delays. This sum describes coordination response rather than a physical breaker-clearing time.

Analytical Fault-Propagation, Stability, Detection, and Consensus Models

Equation (9) defines a normalized graph-propagation envelope for the vector x_k of per-unit voltage deviations from nominal at discrete sample k :

$$x(k+1) = \alpha A_n x(k) + B_f u_f(k) + w(k), \quad 0 \leq \alpha < 1 \quad (9)$$

In Eq. (9), α is a dimensionless propagation factor, A_n is a row-normalized nonnegative adjacency matrix, $u_f(k)$ is the injected fault-disturbance surrogate, B_f selects its affected node, and w_k represents bounded measurement and load disturbances. The subscript n in A_n denotes normalization; it does not denote the validator count. The spectral radius $\rho(A_n)$, defined as the largest eigenvalue magnitude, is at most 1. With $0 \leq \alpha < 1$, $\rho(\alpha A_n) < 1$; consequently, the unforced envelope is asymptotically stable and bounded inputs produce bounded deviations. This is a graph-level result rather than a converter-control or cable-transient stability proof.

Equation (10) defines detection and false-alarm probabilities for a threshold θ :

$$P_D(\theta) = \Pr(V_{\min} < \theta \mid F = 1), \quad P_{FA}(\theta) = \Pr(V_{\min} < \theta \mid F = 0) \quad (10)$$

In Eq. (10), P_D is detection probability, P_{FA} is false-alarm probability, $\Pr$ denotes probability, and $V_{\min}$ is the minimum node voltage for an event. F is the event's ground-truth class: $F = 1$ denotes a fault and $F = 0$ a non-fault. This ground-truth label is distinct from the detector output $F_i(k)$ in Eq. (4). Sample estimates are the detected-fault and false-alarm proportions. Wilson intervals quantify binomial uncertainty, and the standard deviation across twenty non-overlapping batches of 100 events quantifies variability. The original 22-fault paired sample is assessed using paired risk differences and Cochran's Q test.

Equation (11) gives the probability of delivering at least q votes from n validators under independent message-loss probability p_l :

$$P_Q(n, q, p_l) = \sum_{j=q}^n \binom{n}{j} (1 - p_l)^j p_l^{n-j} \quad (11)$$

In equation (11), P_Q is the probability of obtaining a quorum in one round, j is the number of delivered votes, and the binomial coefficient counts the corresponding subsets of validators. The factor $1 - p_l$ is the delivery probability. Equation (12) defines the BFT-style message upper bound $M(n)$ and its communication-service delay T_{msg} :

$$M(n) = (2n + 1)(n - 1), \quad T_{msg} = 1000M(n)/C_m \quad (12)$$

In equation (12), C_m is message-service capacity in messages/s, and T_{msg} is in milliseconds. The scaling model uses $C_m = 2,000$ messages/s. Its message count is an analytical bound and does not imply that the implemented Ganache workflow is Byzantine-fault-tolerant. For the stages in Eq. (8), the mean total delay is the sum of the stage means; under independent stage delays, the total variance is the sum of the stage variances.

Consensus-Latency Scaling and the Protection-Coordination Constraint

Equations (11) and (12) expose the trade-off between quorum delivery and communication cost. More validators can improve delivery probability for the selected quorum, but increase traffic and service delay. Under independent retries of failed rounds, the expected number of attempts is $1/P_Q$. Validator diversity must therefore be assessed together with message traffic and retry delay.

The local-trip and ledger deadlines differ. T_{local} denotes the local relay decision delay, whereas T_{coord} denotes completion of ledger coordination. The former supports primary action; the latter supports notification, coordination, supervised recovery, and recording. The reported 1–2 s coordination interval exceeds the several-millisecond DC interruption requirement discussed in [1]. Small validator zones, bounded event messages, gateway timing limits, and an independent local trip are practical requirements.

Coordination-Correctness Properties

Proposition 1 (guarded-transition safety): A candidate event cannot enter `FAULT_CONFIRMED` through the normal validation path without at least q authorized approving votes. Proof: Equation (7) requires the admissibility checks and the vote sum to satisfy the quorum before approval. Without approval, the event is rejected or, on timeout, enters `FAIL_SAFE_ISOLATION`. The property follows from inspection of the transition guards summarized in Table 5.

Non-blocking local protection (Proposition 2): The modeled local trip output is not suppressed by a ledger outage. This output is generated by the detector before it submits a transaction and no contract transition returns a blocking instruction to the relay. In this software model, an outage will not affect the local decision as it will only affect the coordinated recording or subsequent actions. This is a state flow property, NOT an electrical selectivity or converter stability.

Equation (13) defines the reported detection, missed-fault, and false-alarm rates, expressed as percentages:

$$\begin{aligned} R_D &= 100 \frac{N_{\text{detected}}}{N_{\text{actual}}} \\ R_M &= 100 \frac{N_{\text{missed}}}{N_{\text{actual}}} \\ R_{FA} &= 100 \frac{N_{\text{false}}}{N_{\text{nonfault}}} \end{aligned} \quad (13)$$

In equation (13), N_{actual} is the number of true fault events, N_{detected} and N_{missed} are detected and missed faults, N_{nonfault} is the number of true non-fault events, and N_{false} is the number of false alarms. Thus, R_D and R_M use the fault denominator, while R_{FA} uses the non-fault denominator. For the paired study these denominators are 22 and 28, respectively.

The topology panel displays each modeled node and its status: green for healthy, red for faulted, and amber for isolated. It helps the operator identify the selected event and the links opened by the logical isolation rule.

Blockchain Ledger Design and Transaction Structure

Table 4 summarizes the transaction and validation record. Only authorized grid devices submit or validate these records. Each transaction includes the following information:

- A transaction hash that is unique.
- The name of the reporting node.
- The UTC timestamp (milliseconds).
- (iv) The type of the event (e.g., fault detected, fault isolation, recovery initiated or no-fault confirmation).
- Voltage that is measured at the reporting node.
- Digital signature of the reporting agent to provide authenticity and integrity of the data.

The participating validators are identified and authorized. The implemented round-robin workflow is used for functional testing in Ganache and is not claimed to provide Byzantine fault tolerance. To support deployment-oriented analysis, this study additionally evaluates a two-phase, BFT-style upper bound on message complexity, quorum-success probability under independent message loss, and a fail-safe rule that preserves autonomous local protection when ledger agreement is delayed or unavailable.

Table 4. Blockchain Transaction and Validation Record Structure

Field	Content and purpose
Transaction hash	Unique identifier for the submitted event
Originating node	Bus or monitoring agent that generated the event
UTC timestamp	Millisecond-resolution event ordering
Event class	Normal, suspected fault, physical fault, FDIA, isolation or recovery
Measured values	Actual/reported voltage and selected current information
Validator result	Individual votes, quorum status and consensus outcome
Signature/status	Authentication indicator and smart-contract state

Smart-Contract Logic for Fault Handling

The protection contract is represented as an extended finite-state machine with the states MONITORING, SUSPECTED_ANOMALY, VALIDATION, FAULT_CONFIRMED, FDIA_DETECTED, ISOLATED, RECOVERY_CHECK, RECOVERED and FAIL_SAFE_ISOLATION. This is because a transition is only accepted if the measurement condition, validator condition and current-state condition are all true. An accepted transition generates a ledger event with a time stamp.

A suspected event is created when a voltage-threshold violation, abnormal deviation or a trend mismatch is detected, and validated if it is seen again. The values of actual and reported are compared and the signed event is put on the validator set.

VALIDATION -> FAULT_CONFIRMED or FDIA_DETECTED -> ISOLATED: Either a physical fault or reported-data mismatch are detected by the quorum confirmation. The

contract fetches the neighbouring nodes from the node adjacency matrix and sends a simulated breaker open command on the selected isolation boundary.

ISOLATED -> RECOVERY_CHECK -> RECOVERED: Reconnection can only take place when the measured voltage is above the recovery threshold and the recovery event is approved by the validators. If the check fails, the system goes back to ISOLATED.

Fail-safe behaviour: If a validation timeout, missing clearance confirmation or communication failure occurs the framework will remain in the logical state ISOLATED and raise an operator alert. This is a conservative simulation rule and not an independent hardware protection solution.

Figure 1 shows the four functional layers connecting the physical network, local detection, permissioned validation, and supervisory coordination.

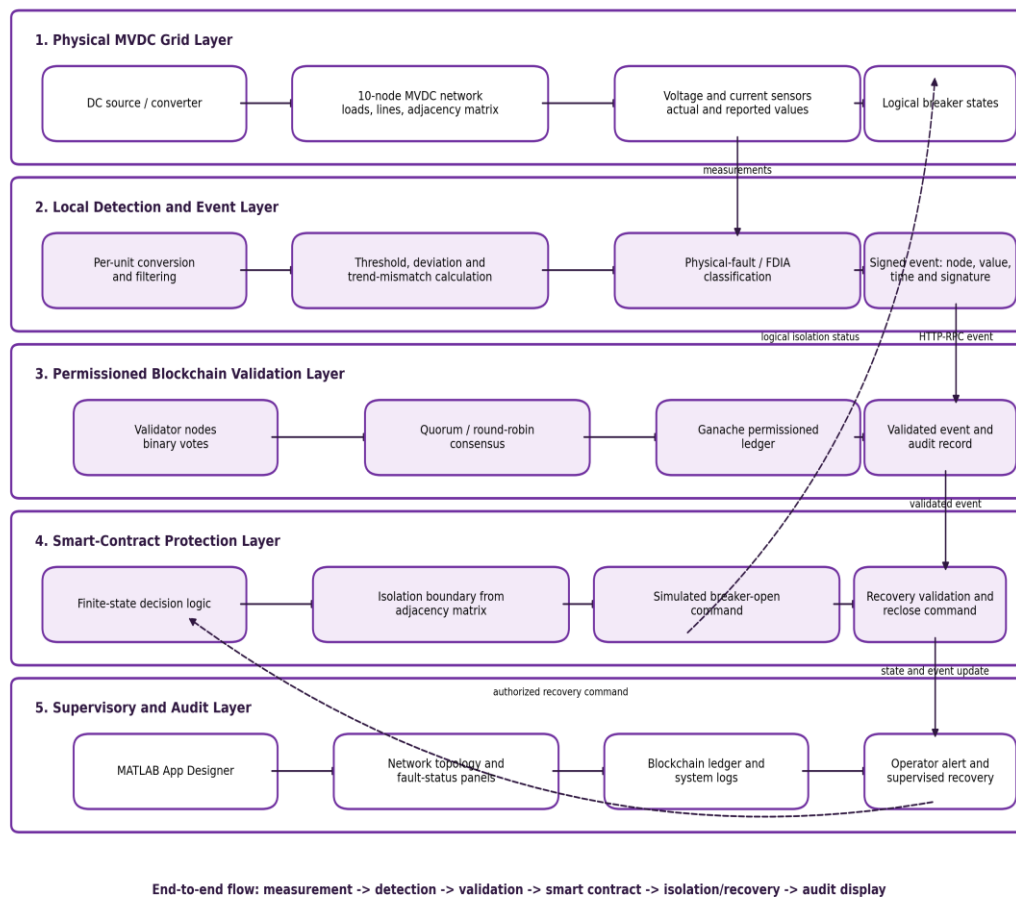


Figure 1. Layered Architecture of the Proposed Blockchain- and Smart-Contract-Enabled MVDC Fault-Management Framework

Voltage and current are processed locally, and candidate event summaries are sent to validators. After approval, the smart contract obtains the affected boundary from the adjacency matrix and updates the simulated switching state. Detection, isolation, recovery, and operator actions are recorded in the ledger and displayed in MATLAB.

Data Flow Model

The Level-1 data flow model in Figure 2 identifies the origin, transformation, storage and destination of each item used by the framework. The actual and reported voltage/current readings, as well as reference thresholds, validator votes and authorized operator commands are the principal inputs. Intermediate data include normalized measurements, deviation and trend features, the event class, a digital signature, consensus status and the isolation boundary. The outputs are simulated breaker commands, ledger transactions, topology updates, alerts and recovery records.

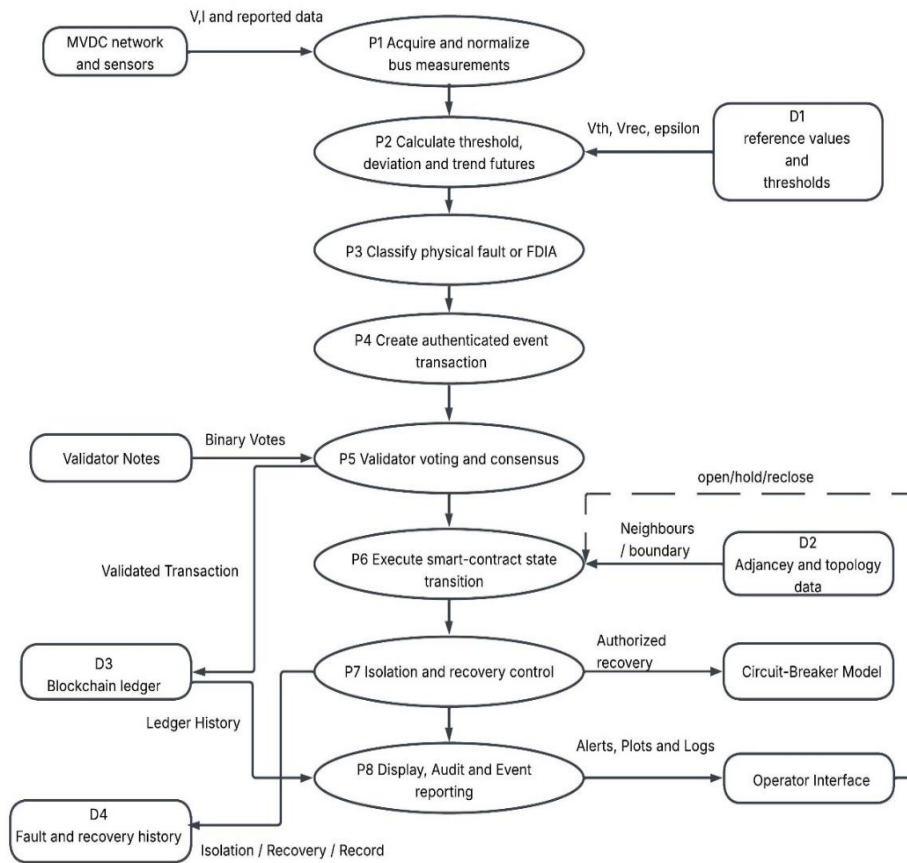


Figure 2. Level-1 Data Flow Diagram of Measurement Acquisition, Fault/FDIA Classification, Blockchain Validation, Isolation and Recovery

The decision rules use the local detector in equation (4), the approval guard in equation (7), and the latency decomposition in equation (8). The reported-data mismatch is defined in equation (14):

$$M_i(k) = |V_{i,actual}(k) - V_{i,reported}(k)|; \quad FDIA_i(k) = 1\{M_i(k) > \varepsilon\} \quad (14)$$

In equation (14), $M_i(k)$ is the absolute voltage mismatch in pu; $V_{i,actual}(k)$ and $V_{i,reported}(k)$ are the actual and reported voltage streams; ε is the mismatch tolerance; and $FDIA_i(k)$ is the binary reported-data anomaly flag. A flag identifies a discrepancy and does not by itself establish malicious intent.

Listing 1 uses $\varepsilon = 0.02$ pu for the illustrative core routine. The separate 2,000-event stress test uses $\varepsilon = 0.05$ pu, as stated with its results. These are distinct test settings.

Consensus approval requires at least q votes, consistent with equation (7). The alternative labels $Q_{\min}$ and $C(k)$ refer to the same minimum vote requirement and approval flag; q and $C_i(k)$ are used here for consistency.

In equation (8), $T_{\det}$ includes sensing and local decision processing, and T_{iso} denotes the simulated actuation stage. Thus $T_{\det} = T_{\text{sense}} + T_{\text{detect}}$, T_{cons} is the consensus stage, and T_{iso} corresponds to $T_{\text{actuation}}$ in the workflow. The reported 10–15 ms covers local sensing and detection only.

UML Sequence Model

Figure 3 shows the order of interactions between sensing, validators, the smart contract, and the ledger. Alternative paths handle normal measurements, actual/reported-data mismatch, rejected consensus, timeout, and recovery. The contract acts on validated events, and reclosure requires renewed measurements above the recovery threshold and validator approval.

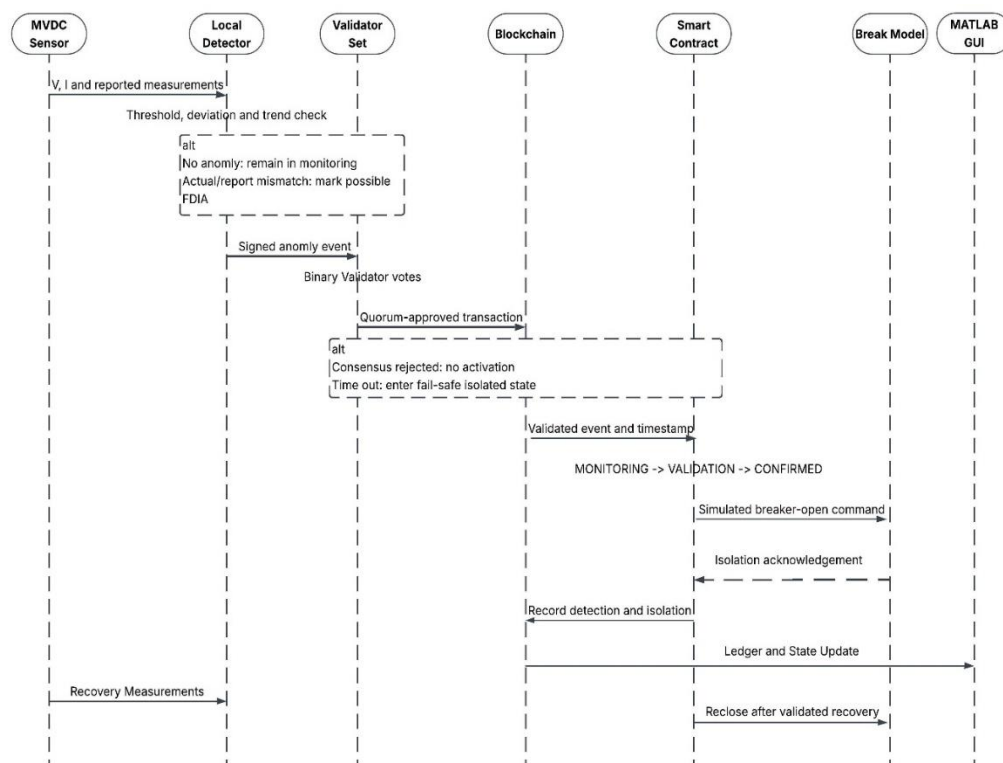


Figure 3. UML Sequence Diagram of Sensing, Distributed Validation, Smart-Contract Execution, Isolation, Ledger Recording and Recovery

UML State-Machine and Transition Conditions

Table 5 and Figure 4 describe the protection states, conditions for enabling and fail-safe responses. This Augmented Representation eliminates the ambiguity between fault

suspicion, distributed validation, confirmed physical fault, cyber-data mismatch, isolation and recovery.

Table 5. Smart-Contract States, Transition Conditions and Actions

Current state	Condition / event	Next state	Action
MONITORING	$V < V_{th}$, abnormal deviation or trend	SUSPECTED_ANOMALY	Create candidate event
SUSPECTED_ANOMALY	Signed event submitted	VALIDATION	Collect validator votes
VALIDATION	Quorum confirms physical fault	FAULT_CONFIRMED	Prepare isolation boundary
VALIDATION	Actual/report mismatch confirmed	FDIA_DETECTED	Reject compromised value and secure state
VALIDATION	Timeout or communication failure	FAIL_SAFE_ISOLATION	Retain conservative isolation
FAULT_CONFIRMED or FDIA_DETECTED	Isolation command issued	ISOLATED	Update topology and ledger
ISOLATED	Recovery measurement available	RECOVERY_CHECK	Evaluate V_{rec} and votes
RECOVERY_CHECK	Recovery approved	RECOVERED	Issue simulated reclose
RECOVERY_CHECK	Recovery not approved	ISOLATED	Hold isolation
RECOVERED	Network state restored	MONITORING	Resume acquisition

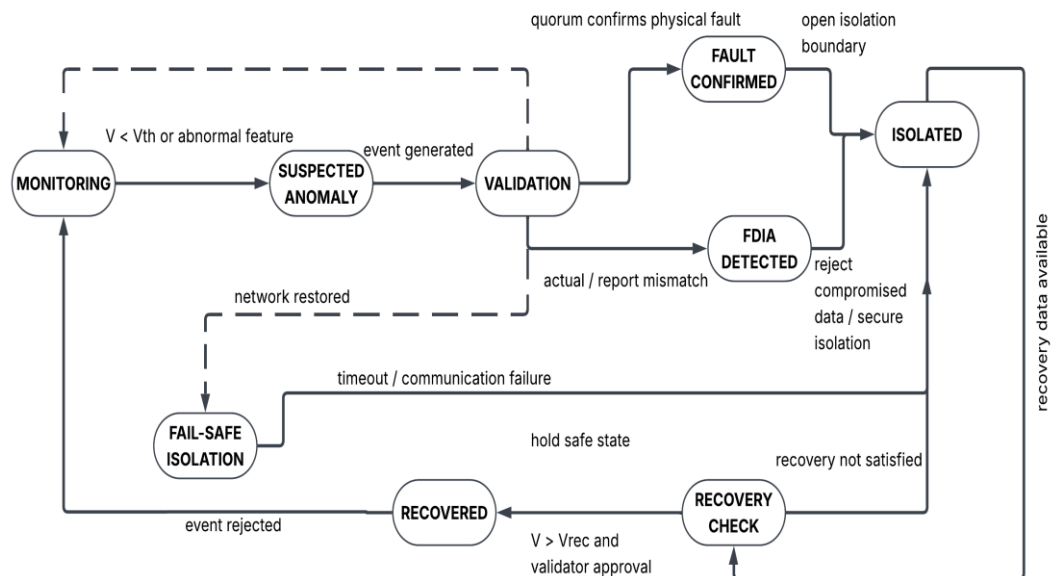


Figure 4. UML State-Machine Model of the Smart-Contract-Based Validation, Isolation and Recovery Process

METHODOLOGY: SIMULATION DESIGN, EXECUTION WORKFLOW, AND CRITICAL-PATH ANALYSIS

This section describes the simulation settings in Table 6, the operational algorithm, and the decision workflow implemented in MVDC_DFM_App.

Simulation Environment and Tool Selection

The framework runs in MATLAB App Designer with a local Ganache connection through HTTP-RPC. App Designer exercises the decision logic, data flow, transitions, and display; it does not model cable transients, converter switching or current limiting, breaker arcs, or physical reclosure. The companion script adds the fixed-seed 2,000-event logical stress test and analytical communication calculations.

Ganache emulates an Ethereum-compatible permissioned workspace with a nominal 1 s block interval. The measured 1-2 s workflow reported in the original implementation is treated as coordination latency, not clearing time. The new scalability model uses an explicit 1 s block contribution, a 65 ms fixed processing allowance, a 2,000-message/s service-rate assumption, and disclosed validator-message counts; these are sensitivity assumptions rather than field measurements.

Table 6. Configuration, Topology, Platform and Deployment Settings Used in the Evaluation

Item	Configuration used in the revised evaluation	Purpose / limitation
Software platform	MATLAB R2026a App Designer	Logical algorithm, interface and output generation
Blockchain environment	Local Ganache Ethereum-compatible workspace; nominal 1 s block interval	Controlled transaction testing; not protection-grade finality
Network topology	10 nodes; undirected symmetric adjacency matrix; at least two neighbours per node	Logical connectivity and isolation-boundary selection
Measurements	Per-unit voltage; current and power variables retained by the application	Threshold, deviation and event construction
Threshold study	0.6, 0.7 and 0.8 pu on the same 50 runs	Paired sensitivity and uncertainty assessment
Fault injection	Synthetic bus-voltage reduction to 0.2-0.6 pu in selected runs	Proof-of-concept fault emulation; no EMT waveform model
Communication	MATLAB-to-Ganache HTTP-RPC event exchange	Logical transaction path; no detailed network simulator
Consensus	Known validator set with round-robin/quorum approval	Simple deterministic validation; scalability not established
Outputs	Topology, fault table, voltage plots, ledger and time-stamped logs	Reproducible visual and tabular evidence

The application interface is organised into four panels as in Table 7.

Table 7. Application Interface Components

Panel	Function	Key Components
Network Topology	Displays the MVDC grid graph with real-time node colouring	Graph plot, adjacency matrix, node labels
Fault Status	Shows active and historical fault records	Fault table, status lamp, alert label
Blockchain Ledger	Lists all validated transactions chronologically	Transaction table with hash, node, time, status
System Logs	Records operator actions and system state changes	Time stamped log entries, start/stop controls

Reproducibility, Core Script and Operational Algorithm

The paired 50-run study retains `rng(2026,'twister')`. The accompanying analysis script instead starts a specified linear congruential generator at 2026 to reproduce its uniform stream across MATLAB and independent implementations. The script declares the distributions, thresholds, validator sets, message-loss assumptions, and timing ranges before generating cases. Table 8 distinguishes the paired-study reproducibility settings from the extended stress-test parameters in Table 9.

Table 8. Reproducibility Parameters and Disclosed Assumptions

Reproducibility item	Revised specification
Random seed	2026 using the MATLAB twister generator
Number of runs	50 total: 22 fault-containing and 28 no-fault runs
Threshold reuse	Identical run dataset evaluated at 0.6, 0.7 and 0.8 pu
Primary outputs	Detection flag, fault node, FDIA flag, validator result, isolation boundary, recovery state and ledger event
Code availability	Core logic reproduced in Listing 1; full App Designer project and raw output are available from the corresponding author on reasonable request
Unmodelled elements	EMT transients, converter controls, breaker dynamics and packet-level communication

Stress-Test Parameters

Listing 1 provides the illustrative core MATLAB routine, and Algorithm 1 summarizes the operational sequence. V_{act} and V_{rep} are actual and reported per-unit voltage vectors; V_{th} and V_{rec} are detection and recovery thresholds; A is the adjacency matrix; and $votes$ contains binary validator approvals. EPS is the mismatch tolerance, and $\lceil$ rounds upward. The routine sets $q = \lceil 2n/3 \rceil$, where $n = \text{numel}(votes)$. A recovery check does not itself authorize a physical reclose. This separation ensures that blockchain-based coordination supports supervisory decision-making without replacing independent local protection and the safety interlocks required for physical switching.

Table 9. Disclosed Parameters for the Monte Carlo and Communication Stress Test

Parameter	Value	Purpose / boundary
Events	2,000	Reduces binomial uncertainty; logical cases only
Fault / non-fault share	60% / 40%	Disclosed simulation design, not field prevalence
High-impedance surrogate	30% of faults; Vmin = 0.65-0.87 pu	Tests weak voltage depression
Temporary sag	10% of non-faults; Vmin = 0.72-0.84 pu	Creates a realistic threshold false-positive challenge
Sensor error	Uniform +/-0.015 pu	Bounded measurement uncertainty
Reported-data manipulation	12%; concealment +0.12 to +0.22 pu	Targeted FDIA stress condition
Validators / quorum	7 / 5	Two-thirds-style approval threshold
Independent message loss	5% per validator message	Communication robustness sensitivity
Latency assumptions	Disclosed uniform stage ranges	Model-based, not wall-clock measurements

Listing 1 - Reference MATLAB script for fault/FDIA classification, quorum validation and state update.

```

function out = mvdc_dfm_step(Vact,Vrep,Vth,Vrec,A,votes)
% Core logical decision path used by the proof-of-concept
EPS = 0.02;
faultMask = Vact < Vth;
mismatch = abs(Vact - Vrep) > EPS;
[~,faultNode] = min(Vact);
physicalFault = any(faultMask) && ~mismatch(faultNode);
fdiaEvent = mismatch(faultNode);
if fdiaEvent
    eventClass = "FDIA";
else
    eventClass = "FAULT";
end
quorum = sum(votes) >= ceil(2*numel(votes)/3);
out.state = "MONITORING";
if quorum && (physicalFault || fdiaEvent)
    out.state = "ISOLATED";
    out.boundary = find(A(faultNode,:) ~= 0);
    out.event = struct('node',faultNode,'V',Vact(faultNode), ...
        'class',eventClass, ...
        'time',datetime('now','TimeZone','UTC'));
elseif ~quorum
    out.state = "VALIDATION_REJECTED";
end
end
if out.state == "ISOLATED" && Vact(faultNode) > Vrec
    out.state = "RECOVERY_CHECK"; % reclose only after renewed approval
end
end

```

Algorithm 1 - Local fault classification, validator approval, and recovery-state update in the MVDC coordination model.

1. Set the network links, fault thresholds, validators, ledger state, and random seed.

2. Read actual and reported voltage and current; express the measurements in per-unit form.
3. Calculate the node thresholds, deviations, and trend discrepancies.
4. Label the event normal, suspected physical fault, or possible reported-data manipulation.
5. Sign an event with the node ID, measured value, event class, and UTC timestamp.
6. Request validator votes and check whether the required quorum is reached.
7. Reject a candidate without quorum; on timeout, retain the independent fail-safe protection output and log the incomplete coordination.
8. If the event has been approved, run the smart-contract transition and pull back the isolation boundary from the adjacency matrix.
9. Change the simulated breaker state, topology, fault table, ledger and system logs.
10. Re-evaluate the affected node; re-close as long as $V > V_{rec}$ and the recovery event has re-validator approval.

Execution Workflow: Step-By-Step Description

The MVDC_DFM_App proceeds through a set of 10 steps, divided into 3 phases: initialization, fault simulation and reaction, post-fault handling.

Phase I – Initialization (Steps 1-4)

Step 1: Application Start-Up

The system status indicator turns green and the status label will show "System Online" when the application is run. The application starts up the core data structures such as the grid adjacency matrix, the Blockchain ledger, and a voltage vector for each node in the network. This step is for preparing the system for fault monitoring and simulation.

Step-2: Constructing the network topology

The MVDC network is represented by a random symmetric adjacency matrix with 10 nodes. The matrix is structured to ensure that the building blocks are connected to at least 2 neighbouring building blocks, thus allowing uninterrupted communication in the matrix and avoiding isolated sections of the building blocks. The resulting network topology will appear on the Network Topology panel, with nodes being shown as green circular markers, and Electrical Connections as blue lines.

Step 3: Here Blockchain ledger is set up. It is about Initializing the Blockchain ledger.

Initially the Blockchain ledger is created with a genesis block and a collection of sample transaction records with timestamps and labels for the "Confirmed" field. These are initial records to prove the Blockchain interface is working properly, and to represent a template for subsequent failure transactions.

Step 4: In this step, the voltage display is initialized.

The per-unit voltage of all the 10 network nodes is plotted using a grouped bar chart. The vertical axis is from 0-1.5 pu with 1 pu (nominal voltage) and 0.7 pu (fault detection

threshold) marked. All nodes are set to 1.0 pu at the beginning of the simulation, meaning that the MVDC network is in a healthy state.

Phase II – Fault Simulation and Response (Steps 5-6)

Step 5: Fault Simulation

When the operator presses the Start button, the simulation starts. Event history is maintained by storing fault records in a previous simulation, if any, in the Previous Output table. The application then performs a Simulate MVDC Fault Management function. In the process, a node in the network is taken at random, and a specified fault probability is used to decide if a fault occurs. When a fault does occur, the voltage at the chosen node is decreased between 0.2 pu and 0.6 pu, which is a DC bus fault.

Step 6: Fault Detection & Decision Logic

A binary decision process is used to evaluate the results of the simulation.

Fault Detected (Yes Branch):

When the bus voltage is measured below the fault detection threshold, a fault will be detected. The alarm message will read "⚠️Fault Detected at Node X" and the status indicator will be red. A new record is recorded in the fault table that includes the node number which has detected the fault, the detection time, and the isolation status (Isolated). The smart contract then transitions into its "FAULT_DETECTED" and eventually "ISOLATED" states, and the associated fault history is safely saved in the Blockchain ledger.

No fault detected (No Branch):

When there is no fault detected: alert message reads "No Fault Detected" and the status indicator is green. The fault table is cleared to show that conditions are normal for the system and a "No Fault – System Normal" transaction is recorded on the Blockchain ledger. The smart contract is still in the MONITORING state.

Post Fault Handling (Steps 7-10)

Step 7: System Update

The voltage chart will show the voltage of the modelled nodes at the end of a case: the node affected will have lower voltages and unaffected nodes will have voltages close to 1.0 pu. The application also creates a ledger transaction for the fault and/or creates a no-fault ledger transaction with a generated hash and timestamp.

Step 8: System Pause

If the simulation is stopped due to a click on the Stop button, then simulation is stopped. The system status indicator will turn to red, and the status label will read "System Paused". The fault is cleared and all node voltages are reset to their nominal value (1.0 pu). This enables the operator to check the Blockchain ledger and fault history without causing further events.

Step 9: User Interface Initialization

The create Components method is used to create the interface of the application on start-up. It generates the four primary panels and initializes all of the graphical

components such as tables, plots, labels, status indicators, and control buttons, according to the MATLAB App Designer style.

Step 10: Supporting Functions

There are multiple utility functions that are used when the app is running. A helper function is used to make it easy to perform conditional operations and make the code more readable. The Simulate MVDC Fault Management function generates the fault, applies the voltage changes to the faulted node and prepares the simulation results for fault evaluation and processing in the Blockchain in Step 6.

This Critical-Path Flowchart contains Switch Logic

The main decision-making process of the MVDC_DFM_App is shown in the following flowchart. It demonstrates the order of the different operations in the system, from its start-up until detecting, validating, isolating and recovering from faults in the Blockchain. The flowchart is depicted as a figure to illustrate the working of the proposed framework

Critical-Path Flowchart with Switch Logic

Figure 5 presents the complete decision-critical execution path and explicitly identifies the blockchain validation and smart-contract stages that were not visible in the earlier simplified workflow.

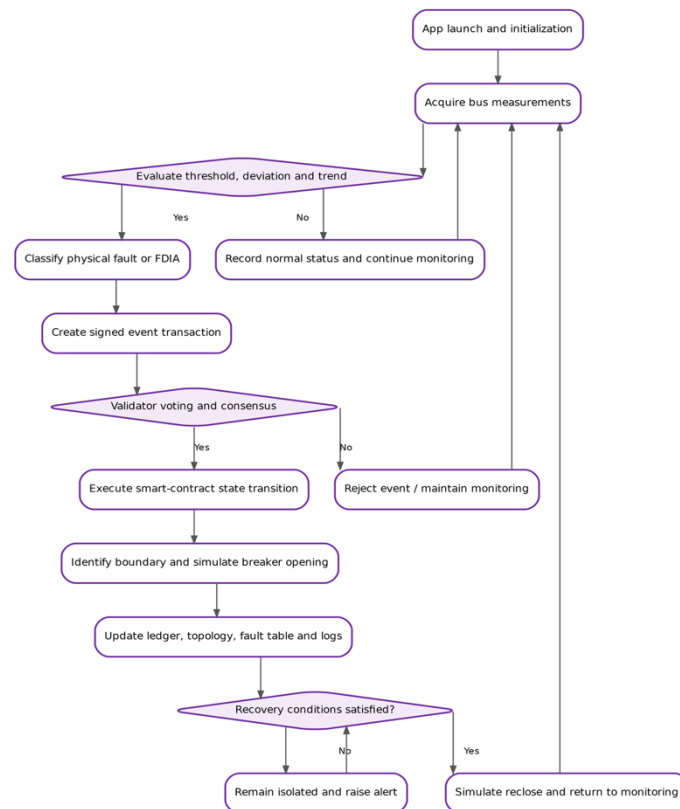


Figure 5. MATLAB App Designer Simulation Execution Flow Including Validation, Isolation, Ledger Update and Recovery.

Critical Analysis of The Decision Logic

The important aspect of the proposed framework is the binary fault detection decision in step 6. The accuracy and response time decide that whether the system can provide efficient fault protection or just record the fault event. The main design considerations are discussed below.

Threshold Selection:

In this implementation the fault detection threshold is set as 0.7 pu. This threshold establishes the trade-off between fault sensitivity and accuracy of the fault detection. A lower threshold will lead to fewer false alarms, but to a slower response time to faults; a higher threshold will lead to a faster response time in detecting faults, but to a greater number of false alarms. For practical MVDC systems additional adaptive threshold or intelligent detection can help achieve accurate fault detection.

Fault Simulation Method:

The fault cases use a probabilistic logical model that is useful for exercising the decision workflow across operating conditions. It does not reproduce how line impedances, converters, current limits, and adjacent buses shape an MVDC transient. Those effects require an electrical network model in a subsequent study.

Blockchain Consensus Latency:

Ganache makes the event and contract workflow observable, but its transaction interval is much longer than an MVDC primary clearing interval. Deployment would require a separately validated protection path; a faster coordination protocol alone cannot establish breaker-level timing.

Fault Isolation Strategy:

With the current implementation, all connections to the faulted node are disconnected when the fault is isolated. This is easy and effective, but could be overkill due to isolating healthy parts of the network. An advanced protection scheme would be more precise in detecting the fault and isolate only the faulty network portion which would increase the reliability and minimize service interruptions.

System Recovery:

When the bus voltage is restored to its normal operating range, the proposed framework will restore the isolated network. However, in a real power system, additional checks such as checking the status of the equipment and system integrity are required prior to reconnection of the isolated portion. The proposed system is also reliable and safe, as the smart contract can be extended to run several validations before restoring.

SIMULATIONS AND PERFORMANCE EVALUATION IN MULTIPLE DIMENSIONS

The results of the simulation carried out using the MVDC_DFM_App are presented in this section, and the performance of the proposed decentralized fault management

framework is analyzed. There are four key metrics used to measure the framework. Moreover, a comparison is made with a traditional central protection system to show the advantages of the proposed method.

Local Fault Detection Latency

In the configured App Designer cases, the rule raised a flag 10-15 ms after a simulated voltage sample crossed the setting. This software interval covers sampling and threshold evaluation. It does not include validation or converter blocking or breaker action, cable transients, or complete fault clearing. Then the application creates a signed summary for the coordination phase, which contains the node ID, voltage and time stamp.

If the ledger is not available, the relay decision is still local. The reported software interval is not a comparison of hardware to a SCADA or other protection device as the other devices were not tested on the same hardware. Later in the paper, a validator analysis quantifies the impact of adding participants to messages and finality under the assumptions of the model.

One case is shown in Figure 6 in the App Designer interface. The selected node is under the voltage level, the status is changed and an alert is raised in the application. Other modeled nodes remain within their assigned ranges. The image verifies how the case is displayed and labelled; it does not test electrical fault-location accuracy.

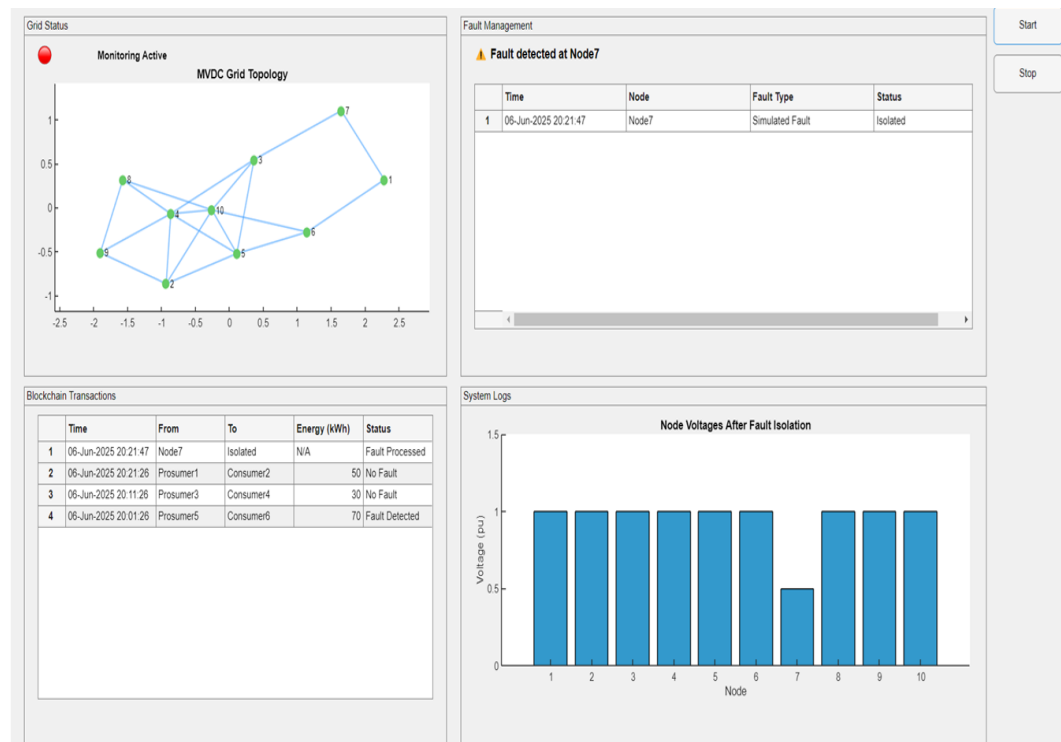


Figure 6. App Designer Display After One Injected Logical Fault Event

Fault Isolation and Containment using Smart Contracts

In the configured Ganache environment, the smart-contract workflow simulated the isolation-state transition in about 1-2 seconds after the local detection and validator

approval. This interval is for the logical transaction processing, consensus, and application update, not a measured solid state breaker opening time. The result illustrates the ability of the architecture to coordinate and record the isolation; however, the blockchain path is too time-consuming to be a primary protection path for MVDC faults in the microsecond scale.

Figure 7 shows the App Designer interface after logical fault processing and the recovery-state update. The topology, transaction table, and system log provide the resulting event and state history.

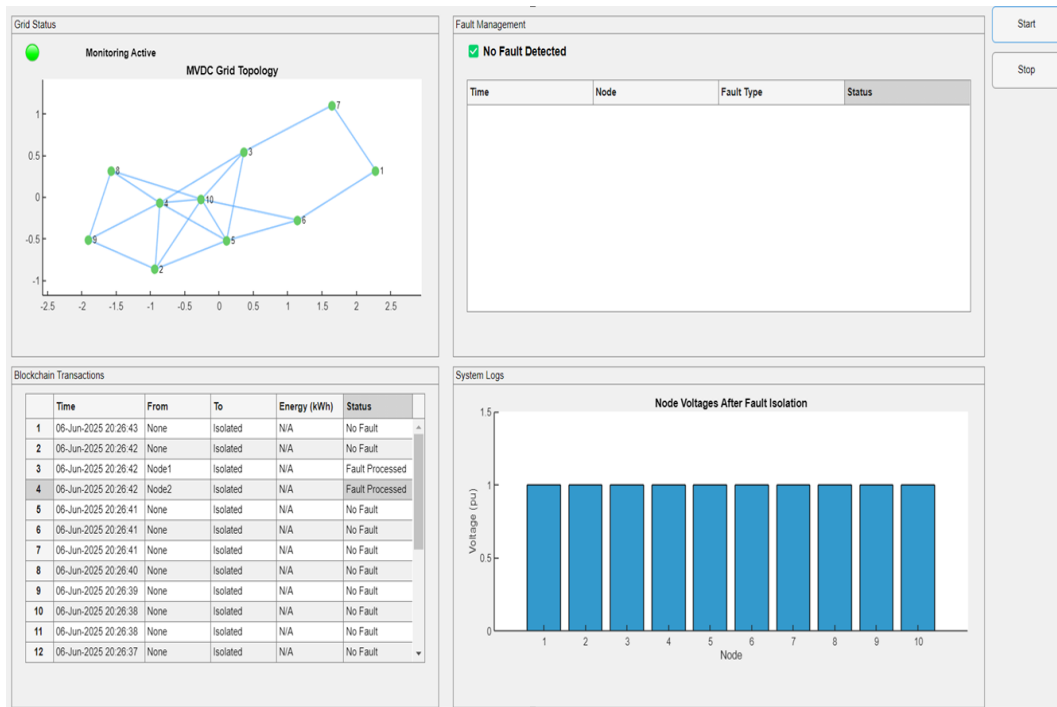


Figure 7. Enhanced-Resolution MATLAB App Designer Interface After Logical Fault Processing and Recovery-State Update.

Threshold Sensitivity Analysis Procedure

The original sensitivity procedure applies 0.6, 0.7, and 0.8 pu thresholds to the same 50 runs, creating a paired comparison. The extended stress test independently generates 2,000 logical events: 60% faults, 30% high-impedance surrogates among faults, 10% temporary sags among non-faults, ± 0.015 pu sensor error, 12% reported-data manipulation, seven validators, a five-vote quorum, and 5% independent message loss. These distributions are deliberately transparent and assess algorithmic robustness; they are not estimates of field event prevalence.

Equation (15) gives the run-level threshold rule used in the paired sensitivity study; V_{th} is the same threshold denoted by θ in equations (4) and (10):

$$D = 1\{V_{min} < V_{th}\} \quad (15)$$

In equations (15), D is the binary run-level detection output and $V_{\min}$ is the minimum node voltage in that run. $D = 1$ indicates a threshold violation; otherwise, $D = 0$. This run-level output differs from the node-specific flag $F_i(k)$ in equation (4).

Automated Post Fault Recovery and System Restoration

The recovery logic brought the isolated node back on-line to the configured recovery mode in the defined time (less than 30 s) when the voltage exceeded the recovery threshold and the event was simulated and validated. This is a software state transition result, no physical insulation check or converter restart sequence or breaker reclose transient is modelled.

This benefit does not necessarily indicate that there is a general reduction in the recovery time, but that a known and measurable recovery sequence is delivered. In the field implementation, the equipment-health-checks and interlocks would be used along with synchronisation to the converter controls and operator policies prior to energisation.

Blockchain Integrity and Operational Transparency of Ledger

Fault, isolation, recovery and normal-confirmation events are recorded on the ledger. The messages per transaction contain event ID, node, timestamp, event type, measurement summary, validator outcome and signature. A later modification to the replicated data can be identified and the accepted sequence can be reassembled with the help of hash linked records. This is a property that is audited and not an electrical dependability result.

Authorized participants keep copies of the accepted records. The integrity claim observed is limited to authenticated submission, changes noticed to recorded data, and the order of transition of the contract. The availability through partitions, Byzantine behaviour and field reliability should be tested through separate fault-injection and protocol tests.

Comparison with Decentralized Non-Blockchain MVDC Protection

In Table 10, the functions of decentralized relays/agents and the ledger are compared. It is the responsibility of local protection to make the time critical decision. The table can then be used to coordinate and investigate if it has been approved by the validator and has the shared event history.

Table 10. Qualitative Comparison with Non-Blockchain Protection Architectures That are Decentralized

Criterion	Local decentralized relay / multi-agent protection	Proposed blockchain-assisted framework
Primary decision path	Local relay or peer controller	Local detector followed by distributed validation
Expected protection latency	Potentially very low; no ledger consensus	Higher coordination delay because transactions and votes are processed
Single coordinating server	Not necessarily required	Not required for ledger agreement

Event integrity	Depends on local logs and communication security	Shared authenticated and append-only transaction history
Cross-owner trust	Requires external governance or trusted coordinator	Consensus and common state provide explicit multi-party validation
Auditability	Distributed logs may differ between agents	Common time-ordered ledger record
Failure concern	Agent/communication failure and inconsistent state	Consensus bottleneck, validator failure and ledger growth
Best deployment role	Primary high-speed fault detection and trip	Secondary validation, coordination, audit and supervised recovery

Comparative Assessment Against Centralised Fault Management

Table 11 describes architectural differences from a SCADA-style supervisory arrangement. Marked intervals are not from the same hardware and communications test, but rather indicate a context for this implementation versus measurement of a improvement over SCADA.

Table 11. Centralized SCADA-based fault management for comparison; values are not from a controlled common testbed.

Performance dimension	Proposed proof-of-concept	Centralized SCADA context	Permitted interpretation
Local detection delay	10-15 ms local logic	Communication + central processing	Indicative only
Blockchain-assisted isolation	1-2 s state workflow	Application-dependent	Coordination feasibility only
Recovery-state update	<30 s configured	Operator/equipment dependent	No physical restoration test
Decision-server dependence	Validator quorum	Hierarchical server	Structural comparison
Fault-record integrity	Shared append-only log	Editable secured database	Structural comparison

In one application, the study was conducted with local flagging, validator decisions, and event recording. Each method can be measured in its classification result for the common logical cases, and the ledger stage has also a significant time cost. These cases are not used to compare the proposed workflow with actual physical MVDC relays, tested in different networks.

RESULTS

Threshold Sensitivity Analysis

The same 50 cases were re-run in the threshold study at 0.6, 0.7 and 0.8 pu. A total of 22 faults were injected and 28 normal cases were observed. We recorded a total of detections, misses, and false alarms, computed proportion and provided a Wilson 95% interval for each proportion. Table 12 reports the resulting counts and uncertainty intervals using the rates in equation (13).

Table 12. Fault Decisions in the Paired 50-Case Threshold Study

Thresh old	Runs	Actual faults	Detected faults	Missed faults	False alarms	Detecti on rate	95% CI for detection rate	False- alarm rate	95% CI for false- alarm rate
0.6 pu	50	22	6	16	0	27.27%	13.15– 48.15%	0 %	0–12.06%
0.7 pu	50	22	14	8	0	63.64%	42.95– 80.27%	0 %	0–12.06%
0.8 pu	50	22	21	1	0	95.45%	78.20– 99.19%	0 %	0–12.06%

A total of 22 faults and 28 normal observations were used for the three settings to compare them with the same voltage cases. Pairing of the cases allows the effect of changing the threshold to be separated from changes in the sampled cases. Uncertainty intervals can also be reported for the resulting counts even though the precision is limited by the small fault and normal groups.

One of the minimum node voltages for each run is shown in Figure 8, as well as lines at 0.6, 0.7, and 0.8 pu. If the minimum value of a run is below the line, the run is marked. The increase in line height introduces more moderate voltage depressions which is why this dataset has a higher sensitivity.

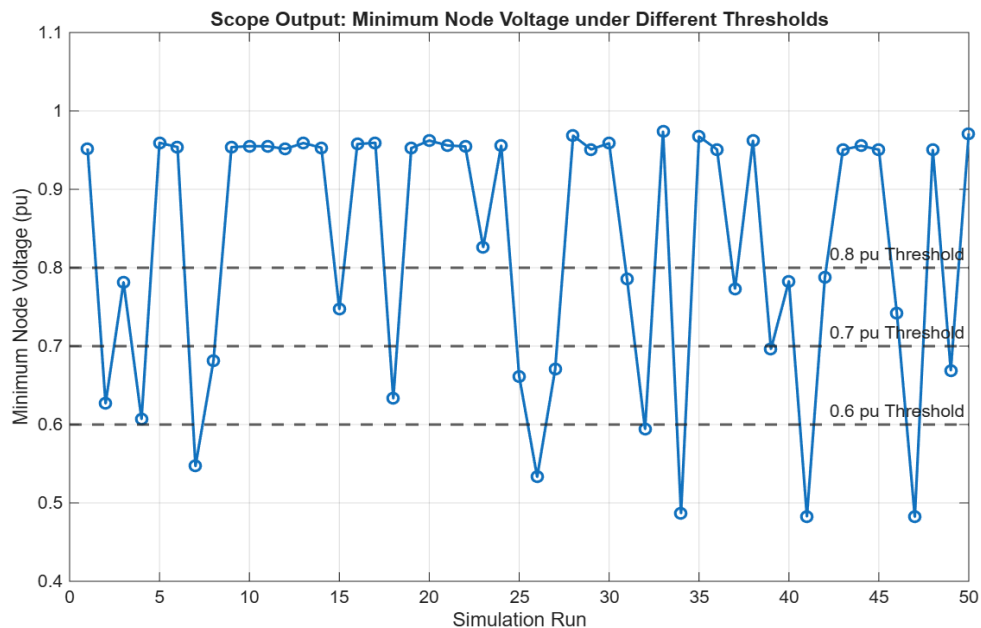
**Figure 8.** Minimum Node Voltage for Each of 50 Paired Cases Along with the Three Node Voltage Thresholds Selected

Figure 9 reports the flagging at each threshold for each case. The 0.6 pu rule will be triggered in the event of significant voltage loss, while the 0.7 pu and 0.8 pu rules will be activated for progressively less severe voltage loss. The vertical offsets are used to visually separate the three pulse traces and are not electrical.

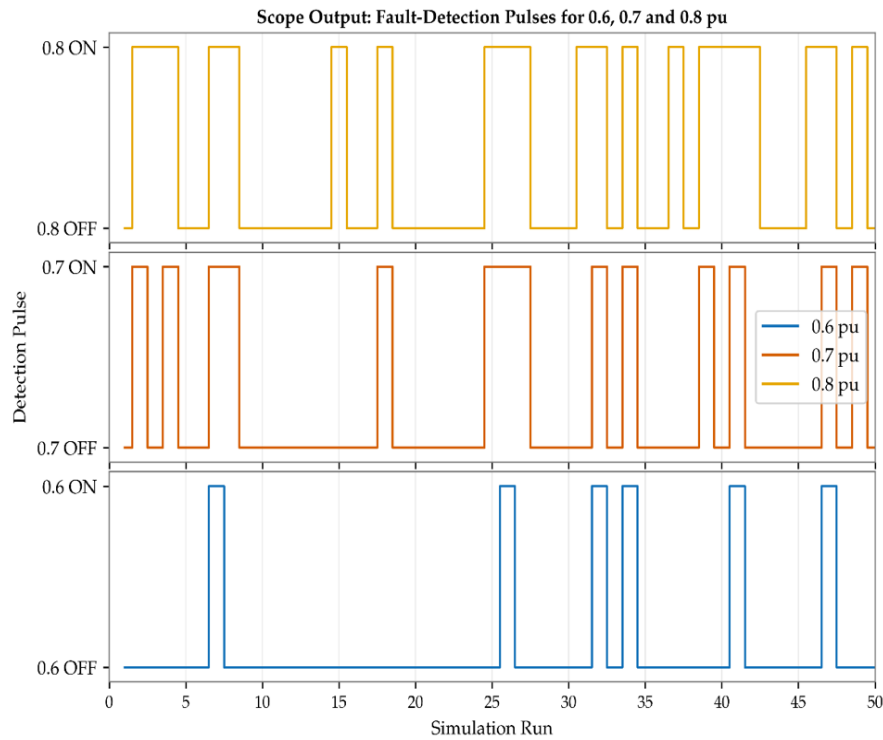


Figure 9. For Cases with Paired Threshold Settings, Outputs from Binary Detection are Provided

The fault counts are shown in Figure 10. Of 22 faults, the 0.6 pu setting detected 6 (27.27%), the 0.7 pu setting detected 14 (63.64%), and the 0.8 pu setting detected 21 (95.45%). They had 16, 8 and 1 missed faults, respectively. There were no flagged cases in the 28 normal cases. The 0-12.06% Wilson interval for the proportion of cases in the normal population illustrates why the observed proportion of zero should not be interpreted as an absolute zero risk.

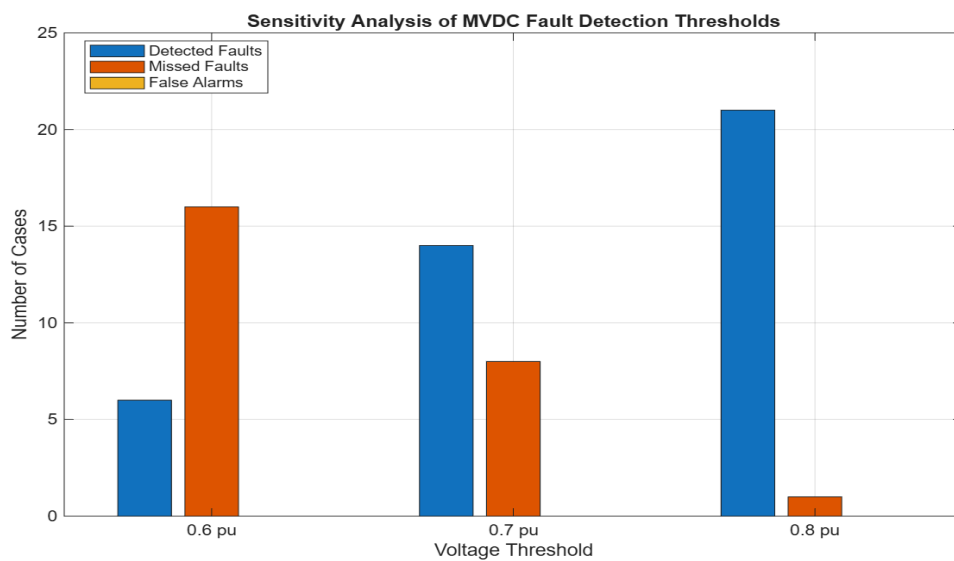


Figure 10. Three Setting Detected Fault, Missed Fault and False Alarm

Statistical Significance of Threshold Selection

Cochran's Q examines the difference between binary decisions, with the fault cases paired across settings. It yielded $Q = 22.53$, $df = 2$, $p < 0.001$. Sensitivity increased by 36.36 percentage points from 0.6 to 0.7 pu, 31.82 points from 0.7 to 0.8 pu and 68.18 points from 0.6 to 0.8 pu. These are the effects in the set of fixed synthetic cases. Here Q is Cochran's test statistic, df is its degrees of freedom, and p is the significance probability; Q is distinct from the quorum q .

The standardized index $Q/\{N(K - 1)\}$ is $22.53/\{22(3 - 1)\} = 0.512$, where $N = 22$ is the number of paired fault cases and $K = 3$ is the number of threshold settings. The separate 2,000-event stress test gives a sensitivity difference of 29.63 percentage points between 0.6 and 0.8 pu, a risk ratio of 1.48, and a descriptive detection-odds ratio of 6.56. The 0.8 pu false-positive rate is 5.49 percentage points higher. These comparisons remain conditional on the synthetic input distributions.

The 0.8 pu setting detected 21 of 22 faults in the paired study (Wilson 95% interval 78.20-99.19%). When it comes to sensitivity these three settings works best, but a higher sensitivity can also indicate temporary dips, fluctuating loads and noisy readings. If they were to choose a field setting, those cases, including high-impedance faults, would have to be tested on a typical system.

Hypothesis-Linked Experimental Observations

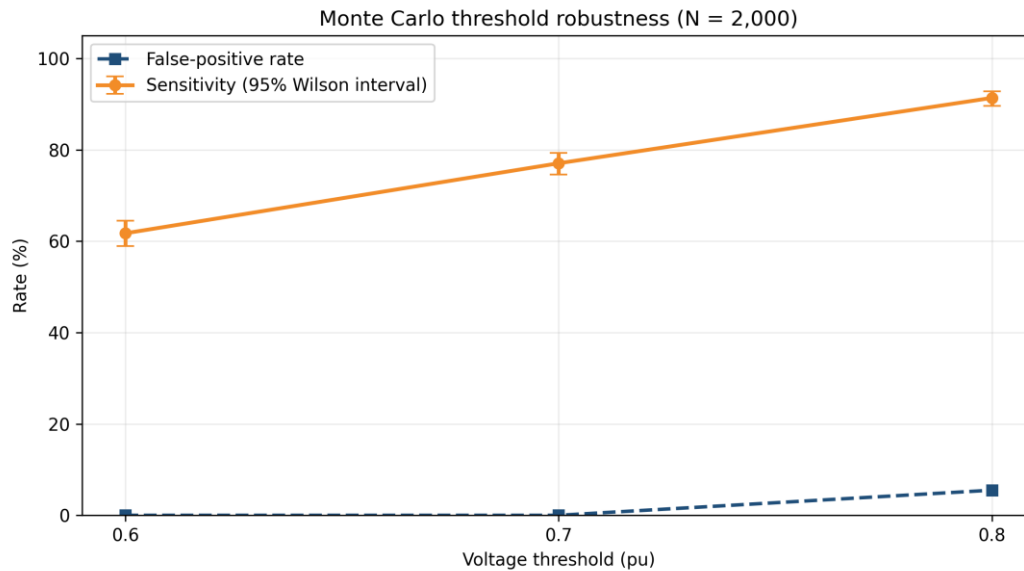
In the case of H1, for the software stage, local evaluation was performed within 10-15 ms, and the modeled ledger path was an average of 1.362 +/- 0.185 s, with no physical SCADA test run. In the case of H2, the calculated independent loss of 5% was 99.62%, the modeled mean coordination delay was 1.362 s, and the accepted events followed the state guards. Additionally, within this logical configuration its 3 specified criteria are satisfied, but not the case with security against collusion and availability during correlated loss. H3 is supported in the paired 50-case study ($Q = 22.53$, $df = 2$, $p < 0.001$; standardized index 0.512); in the separate stress test, sensitivity rose from 61.73% to 91.36% and the 0.8 pu false-positive rate reached 5.49%. All of these decisions do not assume field protection performance.

Monte Carlo Robustness and Uncertainty

The fixed-seed 2,000-event stress test contained 1,181 fault events and 819 non-fault events. Table 13 and Figure 11 show that sensitivity increased with threshold. At 0.8 pu, sensitivity reached 91.36% (95% Wilson interval 89.62-92.83%), but 45 temporary-sag cases were classified as faults, producing a 5.49% false-positive rate (95% interval 4.13-7.27%). Across twenty batches of 100 events, sensitivity standard deviations were 7.22, 7.01, and 3.38 percentage points at 0.6, 0.7, and 0.8 pu, respectively.

Table 13. Fixed-Seed 2,000-Event Monte Carlo Threshold Results

Threshold	TP / FN / FP / TN	Sensitivity (95% CI)	FPR (95% CI)	Batch sensitivity mean +/- SD
0.6 pu	729 / 452 / 0 / 819	61.73% (58.92-64.46)	0% (0-0.47)	61.71 +/- 7.22%
0.7 pu	910 / 271 / 0 / 819	77.05% (74.57-79.36)	0% (0-0.47)	77.01 +/- 7.01%
0.8 pu	1,079 / 102 / 45 / 774	91.36% (89.62-92.83)	5.49% (4.13-7.27)	91.28 +/- 3.38%

**Figure 11.** Monte Carlo Threshold Sensitivity and False-Positive Trade-Off with 95% Wilson Intervals

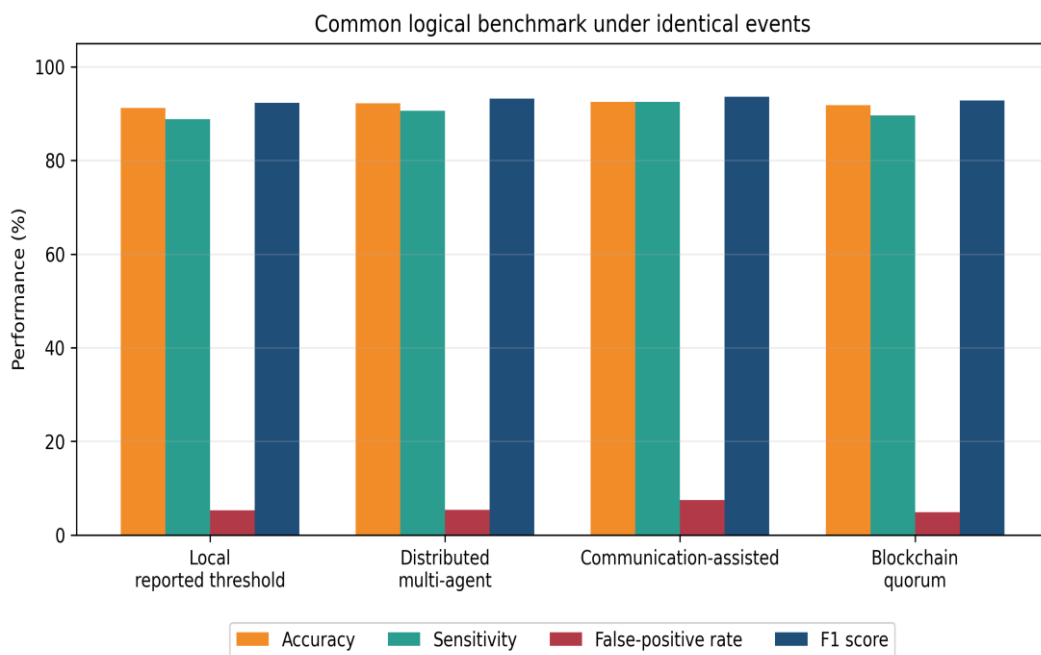
Common Logical Benchmark Against Non-Blockchain Architectures

Four simplified decision paths were executed on the same events: a local threshold acting on the reported value, a seven-agent majority rule, a three-message communication-assisted rule, and the proposed five-of-seven blockchain quorum. This is a controlled algorithmic benchmark, not a reimplement of the physical schemes in [1-5]. The communication-assisted proxy produced the highest sensitivity (92.46%) but also the highest false-positive rate (7.45%). The blockchain quorum reduced the false-positive rate to 4.88% at the cost of lower sensitivity and much higher coordination latency. Table 14 and Figure 12 report this common comparison. TP, FN, FP, and TN denote true positives, false negatives, false positives, and true negatives; accuracy is $(TP + TN)/(TP + FN + FP + TN)$, sensitivity is $TP/(TP + FN)$, and FPR is $FP/(FP + TN)$, each expressed as a percentage. $F1 = 2TP/(2TP + FP + FN)$ is likewise reported as a percentage. CI denotes confidence interval and SD standard deviation.

The reported-data mismatch rule in Eq. (14), with $\varepsilon = 0.05$ pu, detected all 235 deliberately manipulated reports without flagging an unmanipulated report in this bounded stress condition. This result is a consistency check created by a minimum 0.12 pu attack offset; it must not be generalized to stealthy or adaptive FDIA.

Table 14. Common Logical Benchmark under Identical Events, Sensor Uncertainty, Manipulation, and 5% Message Loss

Method	TP / FN / FP / TN	Accuracy	Sensitivity	FPR	F1
Local reported threshold	1,049 / 132 / 43 / 776	91.25%	88.82%	5.25%	92.30%
Distributed multi-agent	1,070 / 111 / 44 / 775	92.25%	90.60%	5.37%	93.25%
Communication-assisted	1,092 / 89 / 61 / 758	92.50%	92.46%	7.45%	93.57%
Blockchain quorum	1,058 / 123 / 40 / 779	91.85%	89.59%	4.88%	92.85%

**Figure 12.** Accuracy, Sensitivity, False-Positive Rate, and F1 Score in the Common Logical Benchmark

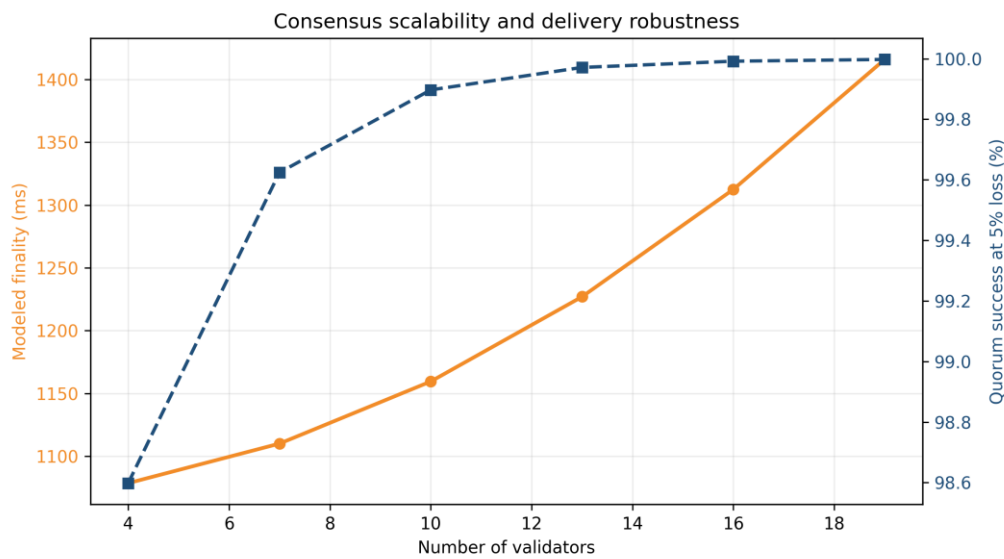
Latency Distribution, Message Loss, and Validator Scalability

The modeled mean $\pm$ standard-deviation latencies were 9.95 $\pm$ 1.15 ms for the local path, 16.94 $\pm$ 2.13 ms for the distributed-agent path, 20.46 $\pm$ 2.51 ms for the communication-assisted proxy, and 1,361.96 $\pm$ 185.11 ms for blockchain coordination. The corresponding 2.5th-97.5th percentile ranges were 8.09-11.90 ms, 12.96-20.86 ms, 15.72-25.32 ms, and 1,007.76-1,719.25 ms. These distributions are generated from the declared timing ranges and confirm the architectural separation of trip and coordination paths.

Table 15 and Figure 13 show the analytical validator-scaling results obtained from equations (11) and (12).

Table 15. Analytical BFT-Style Message and Finality Scaling at 2,000 Messages/S and 5% Independent Message Loss

Validators	Quorum	Messages/event	Modeled finality	Quorum success at 5% loss
4	3	27	1,078.5 ms	98.60%
7	5	90	1,110.0 ms	99.62%
10	7	189	1,159.5 ms	99.90%
13	9	324	1,227.0 ms	99.97%
16	11	495	1,312.5 ms	99.99%
19	13	702	1,416.0 ms	99.998%

**Figure 13.** Modeled Finality and Quorum-Delivery Probability Versus Validator Count

The modeled quorum probability is high when the independent message loss is 5%. With an increasing validator set from 4 to 19, the number of messages per event increases from 27 to 702. Such things as congestion, correlated failures, unavailable leader, malicious validators are not part of the probability calculation. For larger deployments, there would be a need for smaller zones, bounded payloads, event priority, and explicit timeout handling. The table's modeled finality includes a 1,000 ms block allowance, 65 ms fixed processing allowance, and T_{msg} from equation (12); it differs from the randomized stage-latency experiment.

Higher-Message-Loss Robustness

The loss sweep fixes each validator/quorum pair and evaluates Eq. (11) over independent loss rates from 0% to 30%. For seven validators and a five-vote quorum, calculated delivery falls from 99.62% at 5% loss to 92.62% at 15%, 85.20% at 20%, and 64.71% at 30% (Table 16 and Figure 14). This is an analytical measure of probability and not a network measure.

The model assumes a loss of communication but in reality, if the communication fails in the same way for many simultaneously, more may be lost.

Table 16. Analytical Quorum Delivery Probability for Increased Independent Message Loss

Loss	n=4, q=3	n=7, q=5	n=10, q=7	n=13, q=9	n=16, q=11	n=19, q=13
0%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
5%	98.60%	99.62%	99.90%	99.97%	99.99%	100.00%
10%	94.77%	97.43%	98.72%	99.35%	99.67%	99.83%
15%	89.05%	92.62%	95.00%	96.58%	97.65%	98.37%
20%	81.92%	85.20%	87.91%	90.09%	91.83%	93.24%
25%	73.83%	75.64%	77.59%	79.40%	81.03%	82.51%
30%	65.17%	64.71%	64.96%	65.43%	65.98%	66.55%

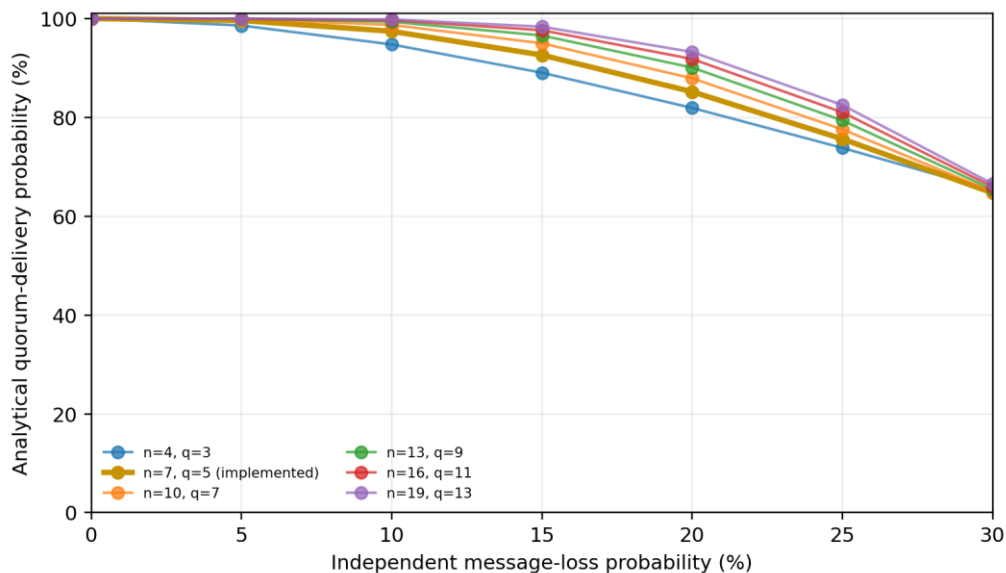


Figure 14. Analytical Quorum-Delivery Probability at Independent Message-Loss Rates From 0% To 30%.

Seven validators make it to 85.20% of single rounds at 20% independent loss. In an independent retry a typical number of attempts would be $1/0.8520 = 1.17$ attempts. The modeled success is 64.71% (or 1.55 attempts on average) when the loss rate is 30%. These expectations are based on independent rounds, and are subject to false optimism if devices are connected to the same failed switch or link.

Hypothesis-to-Evidence Summary

Table 17 maps each hypothesis to the quantitative evidence, its decision, and the scope of that decision.

Table 17. Explicit Mapping of Research Hypotheses to Quantitative Evidence and Limitations

Hypothesis	Quantitative evidence	Decision	Boundary
H1: local processing is faster than coordination paths	Local model: 9.95 +/- 1.15 ms; distributed: 16.94 +/- 2.13 ms; communication-assisted: 20.46 +/- 2.51 ms; blockchain coordination: 1.362 +/- 0.185 s	Supported in logical timing model	No hardware-matched SCADA benchmark
H2: quorum $\geq 99\%$ at 5% loss; guarded accepted transitions; mean delay < 1.5 s	Five-of-seven $P_q = 99.62\%$; accepted transitions pass guards; modeled mean = 1.362 s	Criteria met in the stated logical model	Independent loss; state-flow tests only; correlated failures and adversarial validators untested
H3: threshold materially changes detection	$Q = 22.53$, $p < 0.001$ in paired 50-run test; Monte Carlo sensitivity 61.73-91.36%; FPR rises to 5.49% at 0.8 pu	Supported within synthetic datasets	Threshold not universally optimal

DISCUSSION

The ledger produces a shared sequence of signed event summaries, votes, guarded commands, and recovery records. The tests establish how that sequence is formed in the model. They do not measure an improvement in fault interruption, electrical selectivity, or equipment reliability.

Blockchain penalties: Each event submitted must be communicated, validated, processed and replicated stored. The configured Ganache workflow thus adds a time delay that is significantly larger than the calculation of the local threshold. Ledger growth, signature verification, serialization and execution of transactions and contracts are also computation and communication expensive. The raw high-rate sensor samples should therefore not be recorded on the blockchain. Only compact authenticated event summaries and state changes should be committed (waveform data should be stored locally or off chain with ledger hash for integrity checking).

Limits of scalability: behaviour in the small validator set and in the present ten-node network does not apply to a feeder system of hundreds of devices. Event count and validator count can cause message exchange, quorum formation and ledger replication to become a bottleneck. Concurrent faults can also cause transaction bursts and competing isolation requests. For practical scaling, one would need to implement a hierarchical or

zonal validator group, event batching, off-chain data channels, bounded ledger retention and deterministic conflict-resolution mechanisms. Throughput, finality, storage growth and worst-case latency must all be considered when making these design decisions, not only average latency.

Consensus bottlenecks and resilience: The round-robin model is adapted for functional testing but not an actual Byzantine-fault-tolerant protocol, and is subject to the delay of the leader. A deployment-oriented design should specify the maximum number of faulty validators it will allow, network-partition behaviour, stale-event rejection and the time to finality when packets are dropped. If there is no agreement on the ledger, then protection should default to a safe local action. Thus, a trusted local relay/convertor controller should never be blocked from triggering a protection function by the blockchain layer.

Realistic IEC 61850 and RTDS deployment pathway: (1) map the local relay output to authenticated IEC 61850 GOOSE messages and map slower measurements/status to Sampled Values or MMS as appropriate; (2) preserve a hard-wired or protection-grade local trip independent of the blockchain gateway; (3) emulate delay, loss, duplication, reordering, congestion, time synchronization, and gateway failure in a packet-level network model; (4) connect the gateway and smart-contract client to an RTDS or OPAL-RT MVDC model containing converters, DC-link capacitance, cable frequency dependence, current limiting, protection IEDs, and breaker dynamics; (5) perform controller/relay HIL fault campaigns over fault type, location, resistance, operating point, noise, and communication impairment; and (6) verify selectivity, dependability, security, recovery interlocks, and worst-case end-to-end timing before any field trial.

Relation to the state of the art: The framework is also architectural, but it provides a full mapping from the measurements of the MVDC nodes to the distributed validation and to the protection-state records, it includes a DFD, a UML sequence, a state-machine, a reproducibility script and a paired threshold analysis and uncertainty reporting. This mix allows for both the integration assumptions and trade-offs to be auditable. It is not a new fault-detection property of the MVDC that is being introduced with the study, nor is there a better performance than with existing high-speed decentralized protection; it is simply more transparency and, above all, repeatability of coordination based on the blockchain.

Why Blockchain Coordination Does Not Replace MVDC Primary Protection

The electrical methods in [1-4] address rapid fault discrimination and interruption with line measurements, traveling waves, differential or distance elements, converter action, and breakers. Consensus cannot sample a waveform or interrupt DC current. Because the ledger interval is much longer than the primary protection interval, relays, converter blocking, and breaker interlocks retain their own authority.

A candidate event reaches the ledger only after the local decision. Validators can then authenticate its source, order it with other events, enforce state guards, and retain an audit history. If the gateway or validator network fails, the independent protection equipment must still operate at its local setting.

QUANTITATIVE COMPARISON WITH PUBLISHED MVDC PROTECTION

The present experiment evaluates a logical coordination layer while references [1-5] evaluate physical protection functions. Authors in [1] adopted the hardware-in-the-loop approach of OPAL-RT and referred to a need for several milliseconds of DC interruption. The study [2] used high-frequency, backward propagating waves for direction pilot protection, and a window of -1 to +2 ms was used as a signal-analysis window, which is not a ledger finality measurement. Authors in [3] reported tripping internal faults in less than 3 ms. The study in [4] examined distance relaying for bolted and resistive faults in an electrical model. Authors in [5] examined multi-agent DC microgrid protection under uncertainty. Table 18 keeps these different endpoints visible.

The common metrics are available only for the four methods applied to this paper's same 2,000 logical cases (Table 14). Relative to the local reported threshold, the blockchain-quorum path changed accuracy from 91.25% to 91.85% (+0.60 percentage points), sensitivity from 88.82% to 89.59% (+0.76 points from the unrounded counts), and false-positive rate from 5.25% to 4.88% (-0.37 points). Relative to the distributed multi-agent model, its accuracy was 0.40 points lower, sensitivity 1.02 points lower, and false-positive rate 0.49 points lower. Relative to the communication-assisted model, the changes were -0.65, -2.88, and -2.56 points, respectively. Its modeled 1,362 ms coordination time is 136.9 times the 9.95 ms local processing mean; this ratio compares different stages of the same logical workflow and is not a common physical trip-time benchmark.

Table 18. Published Protection Endpoints and Normalized Comparisons Within the Common Logical Benchmark

Study	Endpoint	Acc. / sens. / FPR (%)	Stage time	Scope
[1]	OPAL-RT HIL	No common cases	Several ms	Physical trip
[2]	Wave pilot	No common cases	-1 to +2 ms window	Window ≠ trip
[3]	Current differential	No common cases	<3 ms trip	Trip ≠ finality
[4]	DC distance	No common cases	Not matched	Different faults
[5]	DC microgrid agents	No common cases	Not matched	Different grid
Local threshold	2,000 logical cases	91.25 / 88.82 / 5.25%	9.95 ms; 1.00x	Case baseline
Multi-agent	2,000 logical cases	92.25 / 90.60 / 5.37%	16.94 ms; 1.70x	Δ +1.00 / +1.78 / +0.12 pp
Comm. assisted	2,000 logical cases	92.50 / 92.46 / 7.45%	20.46 ms; 2.06x	Δ +1.25 / +3.64 / +2.20 pp
Blockchain quorum	2,000 logical cases	91.85 / 89.59 / 4.88%	1,362 ms; 136.9x	Δ +0.60 / +0.76 / -0.37 pp

SUMMARY AND CONCLUSION

This study presented a logical proof-of-concept framework for medium-voltage direct-current (MVDC) protection coordination, linking locally generated fault flags to

authenticated event records and controlled coordination states. The framework traced the decision process from voltage measurement through validator voting to the recovery record. Its architecture and operational logic were documented using data-flow, sequence, and state diagrams, mathematical formulations, defined test cases, and a fixed-seed simulation script, enabling the implementation to be inspected and reproduced.

The principal findings demonstrate the capabilities and limitations of the implemented framework. First, the local software-rule response time was 10–15 ms, excluding consensus processing and physical switching. Second, the Ganache-based state-coordination workflow required approximately 1–2 s, with a mean execution time of 1.362 ± 0.185 s across 2,000 modeled events. This performance supports coordination and event logging rather than time-critical primary fault interruption. Third, following measurement and validator checks, the logical recovery routine returned to the MONITORING state within the specified limit of 30 s; however, physical breaker reclosure was not evaluated. Fourth, the preserved ledger maintained the accepted event sequence, while threshold, communication-loss, and scalability studies characterized the behavior and limitations of the modeled workflow.

The statistical and sensitivity analyses further quantified the trade-offs within the proposed approach. In the paired comparison, Cochran's (Q) was 22.53 ((df = 2), ($p < 0.001$)), with a reported standardized index of 0.512. In a separate 2,000-event stress test, sensitivity increased from 61.73% to 91.36% as the voltage threshold varied from 0.6 to 0.8 pu, while the false-positive rate reached 5.49% at 0.8 pu. For a five-of-seven validator quorum, the modeled event-delivery rate decreased from 99.62% under 5% message loss to 85.20% under 20% loss and 64.71% under 30% loss. These results quantify the performance trade-offs of the logical model under the specified simulation conditions and should not be interpreted as measurements of field protection performance.

Despite these findings, the proposed framework remains a logical proof of concept implemented using MATLAB App Designer and Ganache. It does not incorporate electromagnetic-transient (EMT) representations of capacitor discharge, converter current limiting, cable traveling waves, or circuit-breaker dynamics. Furthermore, packet-level network behavior, real-time execution on RTDS or OPAL-RT platforms, and hardware-in-the-loop (HIL) validation were not investigated. Although the reported effect sizes and state-of-the-art (SOTA) comparison strengthen the statistical evaluation and contextualize the coordination results, they do not replace physical uncertainty propagation, receiver operating characteristic (ROC) analysis and area-under-the-curve (AUC) assessment using validated signals, protection-selectivity testing, or hardware-based timing measurements.

Several modeling assumptions also warrant further investigation. The adjacency-based isolation rule may disconnect more network links than necessary to isolate a cable fault, while the probabilistic model simplifies the voltage distribution and message-loss process. Future research should therefore investigate correlated communication failures, network-topology changes, malicious validator collusion, simultaneous conflicting faults, clock synchronization errors, and realistic communication queues. Further validation should

integrate EMT-based converter and breaker models with communication-network emulation on RTDS or OPAL-RT platforms, followed by HIL experiments under representative operating and fault conditions. Formal verification of the smart-contract logic and testing with IEC 61850-based communication emulation would also strengthen the assessment of coordination reliability and implementation robustness.

Overall, the study establishes a reproducible logical framework for examining validator-based MVDC protection coordination, event recording, and recovery-state management. Its principal contribution is the explicit linkage of local fault detection, distributed validation, ledger-recorded decisions, and logical recovery within a testable software workflow. Demonstrating its suitability for practical protection applications, however, will require real-time and hardware-based validation under representative electrical and communication conditions.

NOMENCLATURE

Symbol / Abbreviation	Meaning
MVDC; DFM; DLT	Medium-voltage direct current; decentralized fault management; distributed ledger technology
FDIA; BFT; DER	False data injection attack; Byzantine fault tolerance; distributed energy resources
EMT; HIL; RTDS	Electromagnetic transient; hardware-in-the-loop; real-time digital simulator
SCADA; GOOSE; IED	Supervisory control and data acquisition; Generic Object-Oriented Substation Event; intelligent electronic device
HTTP-RPC; GUI; DFD; UML	Hypertext Transfer Protocol-Remote Procedure Call; graphical user interface; data flow diagram; Unified Modeling Language
MMS; UTC; pu	Manufacturing Message Specification; Coordinated Universal Time; per unit
CI; SD; TP; FN; FP; TN	Confidence interval; standard deviation; true positive; false negative; false positive; true negative
G ; V ; E ; $ V $	Graph; node set; edge set; number of nodes
A ; a_{ij} ; i , j ; B_i	Adjacency matrix; adjacency entry; node/validator indices as specified locally; neighbour set
I_{fi} ; R_{fi} ; V_i	Approximate resistive-fault current (A); fault resistance (Ω); physical voltage (V) in Eq. (3)
$V_i(k)$; k ; θ ; V_{th} ; V_{rec}	Per-unit node voltage; sample index; detection threshold ($\theta = V_{th}$); recovery threshold
$F_i(k)$; F ; D ; V_{min}	Node detection flag; ground-truth class; run detection flag; minimum event/run voltage
$A_i(k)$; $Auth_i$; $Fresh_i$; $Unique_i$	Event-admissibility flag and authentication, freshness, uniqueness checks

$C_i(k); v_{ij}(k); n; q$	Approval flag; validator vote; validator count; minimum approvals
$1\{\cdot\}; \wedge; \text{Pr}; \varrho$	Indicator function; logical AND; probability; spectral radius
$T_{\text{total}}; T_{\text{det}}; T_{\text{comm}}; T_{\text{cons}}; T_{\text{contract}}; T_{\text{iso}}$	Total coordination response and detection, communication, consensus, contract, simulated-isolation stage delays
$T_{\text{local}}; T_{\text{coord}}; T_{\text{sense}}; T_{\text{detect}}; T_{\text{actuation}}$	Local decision; ledger completion; sensing; decision processing; simulated actuation delays
$x_k; A_n; \alpha; B_f; u_f(k); w_k$	Voltage-deviation vector; normalized adjacency; propagation factor; fault selector; injected disturbance; bounded disturbance
$P_D; P_{FA}; P_Q; p_l$	Detection probability; false-alarm probability; one-round quorum success; independent message-loss probability
$M(n); C_m; T_{\text{msg}}$	Messages per event; message-service capacity (messages/s); message-service delay (ms)
$M_i(k); \epsilon; FDIA_i(k)$	Voltage mismatch (pu); mismatch tolerance (pu); binary reported-data anomaly flag
$N_{\text{actual}}; N_{\text{detected}}; N_{\text{missed}}; N_{\text{nonfault}}; N_{\text{false}}$	Counts of true faults, detections, misses, true non-faults, and false alarms
$R_D; R_M; R_{FA}$	Detection, missed-fault, and false-alarm rates (%)
$Q; df; p; N; K$	Cochran statistic; degrees of freedom; significance probability; paired fault count; threshold-setting count
$\text{FPR}; F1; pp; \Delta$	False-positive rate; harmonic precision/recall score; percentage points; difference from the stated baseline

AUTHORS CONTRIBUTIONS

Conceptualization, R.N., and V.K.M.; Methodology, R.N. and G.S.; Validation, G.S., and S.S.; Formal Analysis, R.N.; Investigation, R.N., and S.S.; Resources, R.N.; Data Curation, R.N.; Writing – Original Draft Preparation, R.N.; Writing – Review & Editing, G.S., S.S., and V.K.M.; Visualization, R.N.; Supervision, G.S. and S.S.; Project Administration, S.S. All authors have read and agreed to the published version of the manuscript.

DATA AVAILABILITY STATEMENT

The datasets generated and analyzed during the current study are available from the corresponding author upon reasonable request.

CONFLICT OF INTERESTS

The authors have no conflicts of interest to declare that pertain to this research.

REFERENCES

1. Monadi, M., Koch-Ciobotaru, C., Luna, A., Candela, J.I., Rodriguez, P. Multi-terminal medium voltage DC grids fault location and isolation. *IET Generation Transmission & Distribution* **2016**, 10(14), 3517–2358.
2. Li, B., He, J., Li, Y., Li, B., Wen, W. High-speed directional pilot protection for MVDC distribution systems. *International Journal of Electrical Power & Energy Systems* **2020**, 121, 106141.
3. Li, M., Luo, Y., Jia, K., Bi, T., Yang, Q. Frequency-based current differential protection for VSC-MVDC distribution lines. *International Journal of Electrical Power & Energy Systems* **2019**, 117, 105626.
4. Allam, M.A., Saleh, K.A. A novel DC distance relay for MVDC microgrids. *IEEE Transactions on Smart Grid* **2021**, 13(2), 962–974.
5. Bamshad, A., Ghaffarzadeh, N. Bi-level multi-agent system (MAS) to protect DC microgrids (DCMGs) in free renewable sources and uncertainties penetration. *IET Generation Transmission & Distribution* **2023**, 17(18), 4188–4214.
6. Lo, C.H., Ansari, N. Decentralized controls and communications for autonomous distribution networks in smart grid. *IEEE Transactions on Smart Grid* **2012**, 4(1), 66–77.
7. Nakamoto, S. Bitcoin: A Peer-to-Peer electronic cash system. Available online <https://bitcoin.org/bitcoin.pdf> (Accessed on 05 January 2026).
8. Mohanta, B.K., Awad, A.I., Elsaka, T., Kheddar, H., Baraka, E. Smart-contract-based blockchain-enabled decentralized scheme for improving smart-grid security. *Internet of Things* **2025**, 34, 101811.
9. Honari, K., Rouhani, S., Falak, N. E., Liu, Y., Li, Y., Liang, H., et al. Smart contract Design in Distributed Energy Systems: A Systematic review. *Energies* **2023**, 16(12), 4797.
10. Santos, L., Gomes, A., Rupino, P. Energy trading using blockchain: Smart contracts functionalities – A systematic review. *Energy Strategy Reviews* **2025**, 61, 101825.
11. Agung, A.A.G., Handayani, R. Blockchain for smart grid. *Journal of King Saud University - Computer and Information Sciences* **2020**, 34(3), 666–975.
12. Dong, Z., Luo, F., Liang, G. Blockchain: a secure, decentralized, trusted cyber infrastructure solution for future energy systems. *Journal of Modern Power Systems and Clean Energy* **2018**, 6(5), 958–967.
13. Guo, Y., Wan, Z., Cheng, X. When blockchain meets smart grids: A comprehensive survey. *High-Confidence Computing* **2022**, 2(2), 100059.
14. Alladi, T., Chamola, V., Rodrigues, J.J.P.C., Kozlov, S.A. Blockchain in Smart Grids: A review on different use cases. *Sensors* **2019**, 19(22), 4862.
15. Ji, Q., Hu, C., Duan, Q., Huang, C., Zhao, X. Decentralized power grid fault traceability system based on internet of things and blockchain technology. *Frontiers in Energy Research* **2023**, 10, 1–15.
16. Hahn, G., Piescorovsky, E. C., Hink, R. B., Werth, A. Detection of faulted phases in a medium-voltage main feeder using the cyber grid guard system with distributed ledger technology. *International Journal of Electrical Power & Energy Systems* **2024**, 161, 110162.
17. Shamaseen, A., Qatawneh, M., Elshqeirat, B. Smart grid system based on blockchain technology for enhancing trust and preventing counterfeiting issues. *Energies* **2025**, 18 (13), 3523. <https://doi.org/10.3390/en18133523>.

18. Ma, Y., Yang, P., Guo, H., Jia, W. Survey on the Application of Blockchain in Enhancing Mutual Trust in smart grid Transactions. *Academic Journal of Science and Technology* **2024**, 12(1), 35–40.
19. Koukaras, P., Afentoulis, K.D., Gkaidatzis, P.A., Mystakidis, A., Ioannidis, D., Vagropoulos, S. I., et al. Integrating blockchain in smart Grids for Enhanced Demand Response: challenges, strategies, and future directions. *Energies* **2024**, 17(5), 1007.
20. Su, K., Yu, Y., Zhang, J. Blockchain-based smart grid power trading technology. *Journal of Engineering and Applied Science* **2024**, 71, 220.
21. Wang, S., Taha, A.F., Wang, J., Kvaternik, K., Hahn, A. Energy crowdsourcing and Peer-to-Peer energy trading in Blockchain-Enabled smart grids. *IEEE Transactions on Systems Man and Cybernetics Systems* **2019**, 49(8), 1612–1623.
22. Junaidi, N., Abdullah, M.P., Alharbi, B., Shaaban, M. Blockchain-based management of demand response in electric energy grids: A systematic review. *Energy Reports* **2023**, 9, 5075–5100.
23. Ullah, N., Al-Rahmi, W.M., Alblehai, F., Fernando, Y., Alharbi, Z.H., Zhanbayev, R., et al. Blockchain-powered grids: Paving the way for a sustainable and efficient future. *Heliyon* **2024**, 10(10), e31592.
24. Sun, H., Weingärtner, T. A Blockchain-Based architecture for energy trading to enhance power grid stability. *Electronics* **2025**, 14(23), 4629.
25. Moniruzzaman, M., Yassine, A., Benlamri, R. Blockchain and cooperative game theory for peer-to-peer energy trading in smart grids. *International Journal of Electrical Power & Energy Systems* **2023**, 151, 109111.
26. Saeed, N., Wen, F., Afzal, M. Z. Decentralized peer-to-peer energy trading in microgrids: Leveraging blockchain technology and smart contracts. *Energy Reports* **2024**, 12, 1753–1764.
27. Sun, C., Wu, Z., Zhu, Z., Jiang, B. Microgrid optimisation: blockchain-enabled peer-to-peer energy trading. *Cyber-Physical Systems* **2024**, 11(3), 325–46.
28. Din, J., Su, H. Blockchain-Enabled smart grids for optimized electrical billing and Peer-to-Peer energy trading. *Energies* **2024**, 17(22), 5744.
29. Lu, X., Guan, Z., Zhou, X., Du, X., Wu, L., Guizani, M. A Secure and Efficient Renewable Energy Trading Scheme Based on Blockchain in Smart Grid. *2019 IEEE 21st International Conference on High Performance Computing and Communications, IEEE 17th International Conference on Smart City, IEEE 5th International Conference on Data Science and Systems* **2019**, 15, 1839–1844.
30. Mazrae, A.K., Naderian, H., Baghaee, H.R., Sheikh-El-Eslami, M.K., Karimi, M. Transactive energy and peer-to-peer energy trading based on blockchain: A comprehensive review and a generalized cyber-physical framework. *Energy Strategy Reviews* **2025**, 62, 101949.
31. Fatima, S., Arshad, M.J. A comprehensive review of blockchain and machine learning integration for Peer-to-Peer energy trading in smart grids. *IEEE Access* **2025**, 13, 92756–92782.
32. Aoudia, M., Alaraj, M.B.M., Waraga, O.A., Mokhamed, T., Talib, M.A., Bettayeb, M., et al. Toward better blockchain-enabled energy trading between electric vehicles and smart grids in Internet of Things environments: a survey. *Frontiers in Energy Research* **2024**, 12, 1–16.
33. Singh, A.R., Kumar, R.S., Bajaj, M., Kumar, B.H., Blazek, V., Prokop, L. A blockchain-enabled multi-agent deep reinforcement learning framework for real-time demand response in renewable energy grids. *Energy Strategy Reviews* **2025**, 62, 101905.

34. Hossain, M.S., Rodine, C., Tsiropoulou, E.E. A blockchain and PKI-Based secure Vehicle-to-Vehicle Energy-Trading protocol. *Energies* **2024**, 17(17), 4245.
35. Ghadi, Y.Y., Mazhar, T., Shahzad, T., Jaghdam, I.H., Khan, S., Khan, M.A., et al. A hybrid AI-Blockchain security framework for smart grids. *Scientific Reports* **2025**, 15(1), 20882.
36. Babayomi, O. O., Igboanusi, I. S., Ahakonye, L. A. C., Kim, D. S. Integrated blockchain and federated learning for the cybersecurity of distributed energy resources. *International Journal of Electrical Power & Energy Systems* **2025**, 173, 111286.
37. Sundareswaran, N., Sasirekha, S. Federated Blockchain Model for Cyber intrusion analysis in smart grid networks. *Intelligent Automation & Soft Computing* **2023**, 36(2), 2129–2143.
38. Ahakonye, L.A.C., Nwakanma, C.I., Kim, D.S. Tides of Blockchain in IoT Cybersecurity. *Sensors* **2024**, 24(10), 3111.
39. Shevchuk, R., Martsenyuk, V., Adamyk, B., Benson, V., Melnyk, A. Anomaly Detection in Blockchain: A Systematic review of trends, challenges, and future directions. *Applied Sciences* **2025**, 15(15), 8330.
40. De Anda-Suárez, J., López-Ramírez, J.L., Jimenez-Mendoza, D., Benitez-Quintero, J.M., Avina-Bravo, E. G., Gutierrez-Hernandez, D. A., et al. An integrated blockchain framework for secure autonomous vehicle communication system. *Information* **2025**, 16(7), 557.
41. Mijwil, M.M., Aljanabi, M., Abotaleb, M., Shukur, B.S., Al Sailawi, A.S.A, Bala, I., Hiran, K.K., Doshi, R., and Dhoska, K. Exploring the Impact of Blockchain Revolution on the Healthcare Ecosystem: A Critical Review, *Mesopotamian Journal of CyberSecurity*, **2025**, 5(1), 6.
42. Swapna, D., Madhuri, A., Lakshmi, T.S., Praveen, S.P. An efficient distributive framework for preserving data privacy through block chain. *International Journal of Recent Technology and Engineering* **2019**, 8(2), 5236–5239.
43. Praveen, S., Nguyen, H., Swapna, D., Rao, K., Kumar, D. The efficient way to detect and stall fake articles in public media using the blockchain technique: proof of trustworthiness. *International Journal on Emerging Technologies* **2020**, 11(3), 158–163.
44. Swapna, D., Sri, U.K., Himaja, V.S.N., Varshita, T.N., Gayatri, V., Praveen, S.P. Crypto logistic network: food supply chain and micro investment using blockchain. In *2023 2nd International Conference on Automation, Computing and Renewable Systems (ICACRS)*, IEEE, **2023**, pp. 908–915.
45. Praveen, S.P., Sharma, K., Parashar, D., et al. Design of an iterative method for adaptive federated intrusion detection for energy-constrained edge-centric 6G IoT cyber-physical systems. *Scientific Reports* **2025**, 15, 41387.