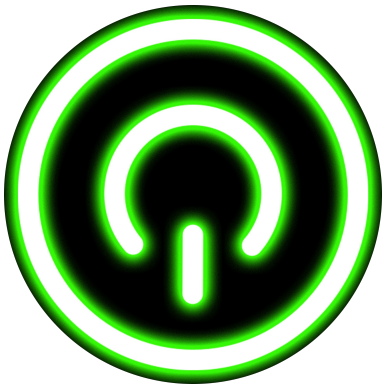
THE NEX NETWORK: A POST-QUANTUM, ZK-ML INTEGRATED ARCHITECTURE FOR A MASSIVELY SCALABLE AND SECURE WEB4 DLT V0.0.2

Jonathan Catalano Macpherson-Gray
0n@thenex.world
© Copyright 2026 by Jonathan Catalano Macpherson-Gray. All rights reserved.



Legal Disclaimer This paper provides an architectural overview of the second major release of the The NEX network, codenamed the "Experience Network" or "Experience". Nothing in this White Paper is an offer to sell, or the solicitation of an offer to buy, any tokens. NEX is publishing this White Paper solely to receive feedback and comments from the public. This White Paper outlines current plans, which could change at its discretion, and the success of which will depend on many factors outside NEX's control, including market-based factors, and factors within the data and cryptocurrency industries, among others. Any statements about future events are based solely on NEX's analysis of the issues described in this White Paper. That analysis may prove to be incorrect.

Abstract

This paper proposes a new distributed ledger technology (DLT) architecture, inspired by the current theoretical definitions of Web4, that aims to address many of the security issues facing DAG-based network design as well as the scalability problems of PoH and transitive learning-based networks. In this v0.0.2 iteration, we introduce Post-Quantum Cryptography (PQC) and Zero-Knowledge Machine Learning (ZK-ML) to solve the existential threats of quantum computing and AI consensus determinism.

This network is based upon Proof of Experience (PoE)—a proof for establishing a network recollection through deep learning and other AI models to improve on network security. PoE leverages the encoding of trustless passage of time into a ledger and combines it with DAG technology. By utilizing ZK-SNARKs, heavy AI inferences are proven cryptographically, reducing messaging overhead in a Byzantine Fault Tolerant replicated state machine and resulting in finality times of fractions of a second.

This paper proposes three more algorithms that leverage the experience tracking properties of the PoE ledger system: a Proof of Experienced Stake (PoES) algorithm that improves recovery from partitions of any size; an efficient real-time

Network Intelligence Control System (NICS) utilizing Federated Learning and Graph Neural Networks (GNNs); and a Dynamic Sub-Sharding (DSS) partition system. Finally, we introduce the Autonomous Threat Remediation Ledger (ATRL), establishing NEX as the premier infrastructure for decentralized cybersecurity agents like NixGuard.

1 Introduction

"Some people can read War and Peace and come away thinking it's a simple adventure story." [1] We are currently in the midst of a technological renaissance and tipping point that will bring distributed ledger technology and AI as a core and indispensable part of every human's everyday life. It is imperative that we continue to constantly learn and innovate on the technology so that the networks used by the builders of tomorrow are as fast, scalable, and cryptographically secure against future quantum threats as possible. Imagine if you can a world where the networks themselves could learn, become more robust, and autonomously defend themselves alongside the developers who work on them.

1.1 NEX Goals and Principles

The NEX is a high-performance, scalable, customizable, and secure distributed ledger platform. It targets three broad use cases:

1. **Autonomous Security & GRC:** Serving as an immutable, decentralized audit trail for autonomous cybersecurity agents (e.g., NixGuard) to execute and record threat remediations and compliance state transitions.
2. **Smart Assets:** Creating arbitrarily complex digital assets with personalized regulations, agreements, and zero-knowledge privacy stipulations.
3. **Web4 Dapps:** Building and launching highly scalable decentralized applications backed by deterministic AI consensus.

By construction, NEX possesses the following properties:

- **Scalable via ZK-ML:** The design of NEX is geared towards massive scalability, durability, and efficiency. The core consensus engine supports billions of devices. By utilizing Zero-Knowledge Machine Learning, heavy AI computations are performed by Leader nodes, while Verifier nodes simply validate lightweight cryptographic proofs, maintaining massive throughput.
- **Secure & Quantum-Resilient:** NEX is engineered to be secure and resilient. Traditional consensus protocols collapse completely when faced with an adversary of size $f + 1$. In contrast, DAGs provide a substantial guarantee of security that degrades gracefully. Furthermore, NEX utilizes NIST-standardized Lattice-based cryptography to ensure the network remains secure against Shor's algorithm and future quantum adversaries.
- **Decentralized & Unbiased:** The NEX network is committed to unparalleled decentralization. We utilize Federated Learning to train the network's internal AI models, ensuring data privacy and preventing model bias, making it a level playing field for miners, developers, and users alike.
- **Governable and Democratic:** $XP is designed to be a highly accessible platform that allows anyone to join its network and actively participate in validation and governance.
- **Interoperable and Flexible:** NEX aims to provide comprehensive and adaptable infrastructure for a variety of distributed ledgers and assets, accommodating a wide range of DLTs and supporting multiple deployment scenarios, including private ZK-subnets.

2 Outline

The remainder of this article is organized as follows. The Quantum Threat and NEX Mitigation is described in Section 3. Overall system design is described in Section 4. In-depth description of Proof of Experience via ZK-ML is described in Section 5. In-depth description of the proposed Network Intelligence Control System is described in Section 6. In-depth description of the proposed Dynamic Sub-Sharding algorithm is described in Section 7. Proof of Experienced Stake consensus is described in Section 8. System Architecture and performance limitations are analyzed in Sections 9, 10, and 11. The NixGuard Cybersecurity Integration is detailed in Section 12.

3 The Quantum Threat & NEX Mitigation

The advent of Cryptographically Relevant Quantum Computers (CRQCs) poses an existential threat to legacy Web3 networks. Algorithms relying on integer factorization (RSA) and elliptic curve discrete logarithms (ECDSA) will be trivially broken by Shor's algorithm.

3.1 Lattice-Based Cryptography

To future-proof the NEX Network, v0.0.2 transitions away from ECDSA. Transaction signing and state channel verifications are secured using Post-Quantum Cryptography (PQC), specifically Lattice-based signature schemes (such as CRYSTALS-Dilithium). These algorithms rely on the hardness of the Shortest Vector Problem (SVP), which currently has no known efficient quantum or classical algorithm.

3.2 Quantum-Secure Proof of History

The Proof of History (PoH) hash chain utilizes quantum-resistant hashing algorithms (e.g., SHA-3). Because Grover's algorithm only provides a quadratic speedup against symmetric cryptography, doubling the hash output size ensures the historical timeline cannot be rewritten by a quantum adversary.

3.3 Signature Compression in the DAG

A known trade-off of PQC is larger signature sizes, which traditionally bloat blockchain state. NEX mitigates this by leveraging its DAG architecture and Zero-Knowledge Rollups. Multiple PQC signatures within a sub-shard are aggregated into a single ZK-SNARK proof before being committed to the overarching DAG, preserving the network's theoretical limit of 710,000 TPS.

4 Network Design

In an effort to combine the efficiency of PoE with Dynamic Sub-Sharding, an AI control system composed of a variety of ML models can be used to efficiently control the network. The network intelligence control system (NICS) will learn from previous consensus block states to inform future consensus as well as assist with management and control of the network's complex dynamic node sharding system.

At any given point in time a system node is designated as Leader to generate a Proof of Experience sequence, which is a PoH sequence combined with output variables of AI models tracking the system in real-time. To solve the issue of AI determinism across different hardware, the Leader generates a ZK-SNARK proof of the AI inference. This provides the network system global read consistency, a verifiable passage of time, and a deterministic recollection of all network states. The Leader sequences user messages and orders them such that they can be efficiently processed by other nodes in the system.

Verifiers execute the same transactions on their copies of the state, but instead of re-running the heavy AI models, they simply verify the Leader's ZK-SNARK proof in milliseconds. The published confirmations serve as votes for the consensus algorithm.

In terms of CAP theorem, Consistency is almost always picked over Availability in an event of a Partition. In case of a large partition, this paper proposes a mechanism to recover control of the network from a partition of any size. [2]

4.1 Mechanism and Properties

The Experience* protocols function through repeated sampling of the network. Each node inquires a randomly selected group of neighbors, which is of a dynamic size, and changes its suggestion if the majority supports a different value.

- **Permissionless, Open to Churn, and Robust:** Experience* protocols have the advantage of providing strong security even when there are discrepancies in the network views of different nodes.
- **Scalable and Decentralized:** Unlike proof-of-stake protocols that delegate validation to subcommittees, Experience protocols enable every node to validate and participate in the system, providing the highest level of decentralization and security.
- **Adaptive:** Unlike other voting-based systems, Experience* protocols achieve higher performance when the adversary is small, and yet highly resilient under large attacks.



Figure 1: PoE transaction flow with ZK-ML Proof Generation

- **Asynchronously Safe:** Unlike traditional longest-chain protocols, Experience* protocols do not rely on synchronicity to prevent double-spends even when the network experiences partitions.
- **Low Latency:** Experience* protocols attain finality in a matter of seconds. Furthermore, Experience* protocols boast high throughput and can handle thousands of transactions per second while maintaining full decentralization. The performance figures for Experience* are taken from a real network running on 2000 nodes globally, using low-end machines.

5 Proof of Experience (PoE) via ZK-ML

5.1 Description

PoE is a unique consensus algorithm that utilizes historical data from the network and its participants to select bookkeeping nodes. It builds upon the concept of Proof of History (PoH), which uses cryptographic methods to verify the passage of time while also incorporating machine learning models and historical data to make informed decisions.

Historically, utilizing AI in consensus faced the ”Determinism Problem”—different GPUs calculate floating-point math slightly differently, causing network forks. PoE solves this by wrapping all AI inferences in Zero-Knowledge

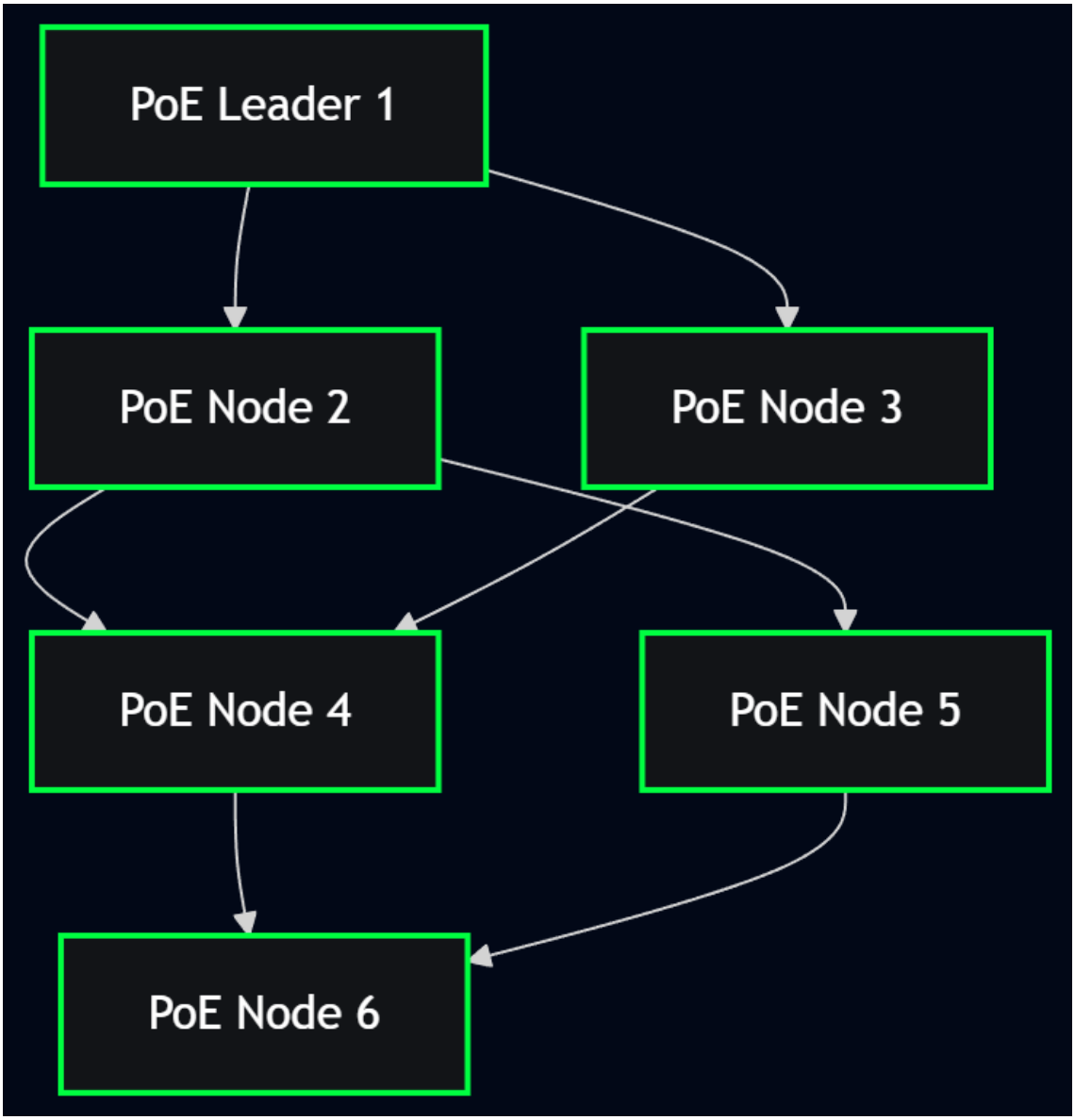


Figure 2: NEX network DAGxPoE visual representation

Machine Learning (ZK-ML) proofs. The network deterministically verifies the cryptographic proof rather than the raw floating-point math.

5.2 PoE AI Models

In PoE, each block's transaction data is used to train a variety of unsupervised and supervised models.

5.2.1 Reinforcement Learning (RL)

Reinforcement learning models can be used to optimize the time for block creation in a PoE consensus mechanism. The model receives as input the historical stake of each validator over a certain time period, the current stake, and the current state of the network. Based on this input, the model outputs a weight for each validator. The Leader node generates a ZK-proof of this RL weight distribution, ensuring all Verifiers agree on the exact validator weights without re-running the RL agent.

5.2.2 Federated K-Means Clustering

K-means clustering is used to group validators with similar levels of experience or stake. To preserve privacy across subnets, NEX utilizes Federated K-Means. Validators train the clustering model locally and only share encrypted gradients. The algorithm groups validators with similar levels of experience together and assigns them to a cluster, allowing the network to adjust the weight of each validator's vote based on their cluster membership.

5.2.3 Transformer-Based Anomaly Detection

Upgrading from traditional Naive Bayes, NEX utilizes Transformer-based anomaly detection to identify validators with malicious behavior. By treating network traffic logs as a sequence of events, the Transformer model learns patterns in how validators make decisions (e.g., time taken, types of transactions processed). Validators with similar, honest behavior are clustered into sub-shards to increase the speed and efficiency of the consensus process.

5.2.4 Time Series Models

Time series models, such as ARIMA or LSTM networks, are used to analyze time-dependent data in PoE consensus. We can use time series models to analyze the historical transaction data of each validator and identify any patterns or trends in their transaction processing speed. If a validator's predicted transaction processing speed falls below a certain threshold, it could be flagged as a potential issue and subjected to additional scrutiny or penalties.

5.3 Verification

5.3.1 Parallel processing

The network hash sequence verification can be completed faster by further leveraging the asynchronous capabilities of the sharded network. By using multiple processing units to process different parts of the network hash sequence verification simultaneously, the time taken to complete the verification can be reduced.

If the expected time to produce the sequence is going to be:

$$\frac{Total\ number\ of\ hashes}{Hashes\ per\ second\ for\ 1\ core} \tag{1}$$

If we want to verify the sequence n times to maintain security standards, the expected time to verify that the sequence is correct is going to be:

$$\frac{\frac{Total\ number\ of\ hashes}{Hashes\ per\ second\ for\ 1\ core}}{(Number\ of\ cores\ available\ to\ verify)(Number\ of\ nodes\ in\ cluster\ delegated\ to\ verify)} \tag{2}$$

They can divide the hash sequence into 4000 slices and each node can verify a different slice. Each node can verify the integrity of the hashes in its slice, starting from the first hash in its slice to the last hash in the last slice.

5.3.2 Use of optimized algorithms & Caching

By using optimized algorithms for the hash function and caching the results of previous verifications, the time taken to complete the verification can be reduced by avoiding redundant computations.

5.3.3 Hardware acceleration

By using specialized hardware, such as GPUs or FPGAs, to perform the hash function or consensus method, the time taken to complete the verification can be significantly reduced. To scale vertically, the network supports the upgrade of hardware components seamlessly through virtualization technology.

5.4 Horizontal Scaling

It is possible to synchronize multiple shards running multiple Proof of Experience generators by first mixing the sequence state from each generator to each other generator, and then mixing each final mixed sequence state together into 1 final network sequence.

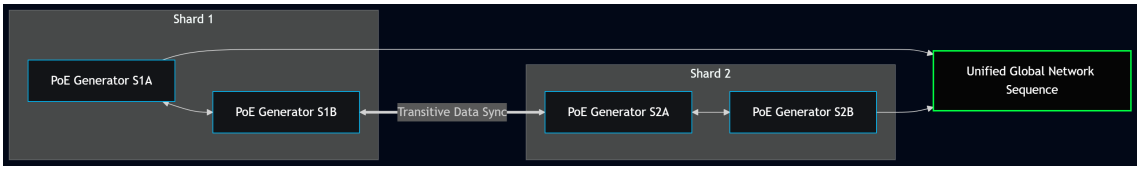


Figure 3: Transitive learning in a sharded network

Given shards S1 and S2 generators S1A & S1B on shard 1, and S2A & S2B on shard 2, S1A & S1B share data packets transitively. Only a subset of the generators in a shard will share data amongst other shards, thus reducing the load on the average generator in each shard.

5.5 Attacks

5.5.1 Reversal

Generating a reverse order would require an attacker to start the malicious sequence after the second event. This delay should allow any non-malicious peer-to-peer nodes to communicate about the original order.

5.5.2 Speed

Having multiple generators may make deployment more resistant to attacks. One generator could be high bandwidth, and receive many events to mix into its sequence, another generator could be high speed low bandwidth that periodically mixes with the high bandwidth generator. The high speed sequence would create a secondary sequence of data that an attacker would have to reverse.

5.5.3 Long Range Attacks

Long range attacks involve acquiring old discarded client Private Keys, and generating a falsified ledger [3]. Proof of History provides some protection against long range attacks. A malicious user that gains access to old private keys would have to recreate a historical record that takes as much time as the original one they are trying to forge. This would require access to a faster processor than the network is currently using, otherwise the attacker would never catch up in history length.

6 Network Intelligence Control System (NICS)

6.1 Description

The network intelligence control system or NICS is a complex amalgamation of a number of AI models and machine learning supported systems. In v0.0.2, NICS is upgraded to utilize Federated Learning and Graph Neural Networks (GNNs), ensuring privacy and state-of-the-art topological analysis.

6.2 Federated Learning & Privacy

A core challenge of network-level AI is data privacy. Nodes in private ZK-subnetworks cannot share their transaction data with the global network to train the NICS anomaly detection models. NEX solves this using **Federated Learning**. Instead of sending raw data to a central server, each node trains a local instance of the NICS model using its own localized data. The nodes then generate a cryptographic commitment of their model's weight updates and share only these encrypted gradients with the network. The global NICS model aggregates these updates, becoming smarter without exposing underlying user data.

6.3 Inputs

An AI model that aimed to track data from a network like this would typically take in the following inputs:

- **Network structure:** Understanding the structure of the network such as the number of nodes, the number of edges, and their relationships. This information helps the AI model to understand the network's topology and how data is flowing through the network.
- **Node characteristics:** Information about individual nodes such as their processing capacity, storage capacity, and communication capabilities. This information helps the AI model to understand the node's potential and limitations, and to optimize data flow through the network.
- **Node utilization:** Information about the current utilization of each node in terms of processing and storage capacity. This information helps the AI model to understand the current workload of each node and make decisions about how to allocate additional workload.
- **Network traffic:** Information about the data flow between nodes, including the volume and type of data. This information helps the AI model to understand the network's data patterns and optimize data flow.
- **Performance metrics:** Information about the performance of the network, such as latency, throughput, and error rate. This information helps the AI model to understand the network's performance and identify bottlenecks.
- **Historical data:** Historical data about the network, including previous network configurations, node utilization, and performance metrics. This information helps the AI model to make predictions and inform future decisions.

Each input parameter is crucial to understanding the network, optimizing data flow, and ensuring optimal performance.

6.4 NICS AI Models

There is a wide range of AI models that could be used to process information from a network, depending on the specific goals of the model. Some common models include:

6.4.1 Graph Neural Networks (GNNs) for Topology

Because the NEX Network utilizes a Directed Acyclic Graph (DAG) rather than a linear blockchain, traditional time-series models are insufficient for structural analysis. GNNs are deployed to analyze the spatial relationships between nodes, transactions, and shards. By understanding the network's topology, the GNN can predict congestion bottlenecks before they occur and dynamically reroute transaction gossip to underutilized sub-shards.

6.4.2 Time-series forecasting models

These models could be used to track metrics such as network traffic or resource utilization over time, with the goal of predicting future trends. Examples of these models include ARIMA and LSTM. For example, NICS could use a seasonal autoregressive integrated moving average (ARIMA) model to forecast the network's traffic volume and bandwidth usage over time. The NICS could collect data on the network's historical traffic volume and bandwidth usage, as well as other metrics like latency and packet loss, and use this data to train the seasonal ARIMA model. The model could then be used to generate forecasts of these metrics for future time periods, allowing the NICS to anticipate and proactively address any potential issues before they occur.

6.4.3 Transformer-Based Anomaly Detection

Upgrading from traditional Isolation Forests, NICS employs Transformer-based anomaly detection to identify unusual or suspicious activity within the network. By treating network traffic logs as a sequence of events, the Transformer

model can identify malicious behavioral patterns with high precision. Once anomalies have been detected, the NICS can take action to mitigate the impact of the anomalous behavior. For example, it could flag the behavior for further investigation, or it could trigger automatic responses such as routing traffic through alternative paths or even blocking certain nodes from the network.

6.4.4 Cluster analysis models

These models could be used to group similar nodes in the network based on shared characteristics, with the goal of understanding the underlying structure of the network. Examples of these models include K-Means and Hierarchical Clustering. For example, if a cluster of nodes is consuming significantly more resources or generating a larger number of transactions than other clusters, this may be a sign of malicious activity or a vulnerability in the network. The NICS can use this information to trigger an alert and take action to prevent further damage to the network, such as isolating the cluster of nodes and launching a deeper investigation into the potential threat.

6.5 Outputs & ZK-Verification

Some possible outputs that could be tracked by these models include:

- **Predictive metrics:** These metrics could include forecasts for network traffic, resource utilization, and other important metrics over time.
- **Anomaly scores:** These scores could indicate the likelihood that a given node or network behavior is unusual, and help identify potential security risks.
- **Centrality measures:** These measures could indicate the importance of a given node within the network, based on factors such as connectivity, in-degree, and out-degree.
- **Cluster labels:** These labels could indicate which nodes belong to which groups within the network, helping to identify key communities or subgroups.

Because these outputs directly influence consensus, the inference must be trustless. The Leader node executing the NICS model generates a ZK-SNARK proof of the inference, allowing the entire network to verify the AI's decision deterministically in milliseconds.

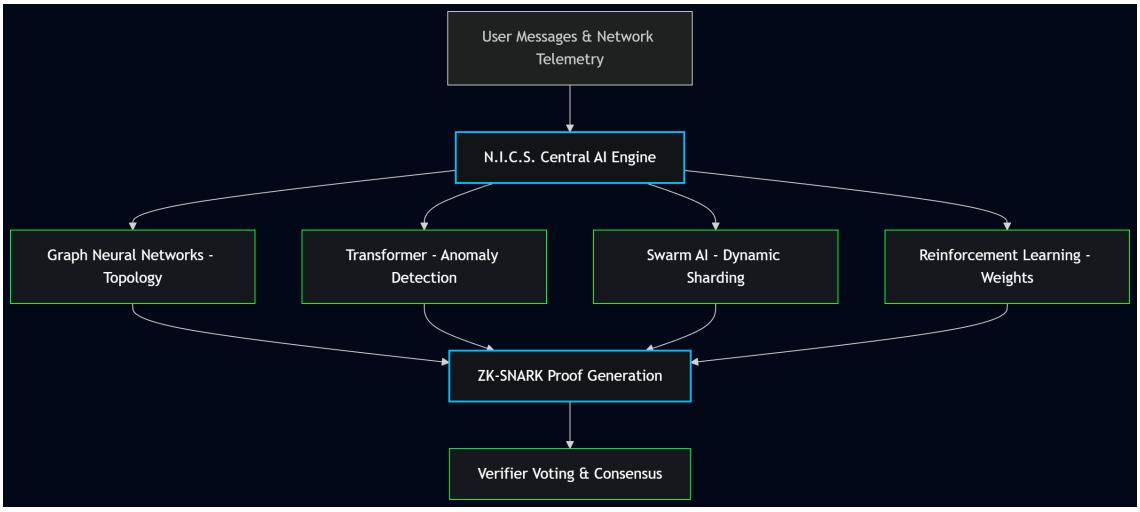


Figure 4: N.I.C.S. visual representation with ZK-ML integration

6.6 Verification Models

AI models can improve the verification process in various ways, such as:



6.6.1 Fraud Detection

AI models can be used to detect anomalies and suspicious behavior within the network, which can be indicative of fraudulent activity. These models can be trained on large amounts of data to identify patterns in the data that may not be apparent to humans.

6.6.2 Optimization of Verification Processes

AI models can be used to optimize the verification process by finding the most efficient way to perform verifications. This can include identifying which nodes to target, which data to verify, and how often to perform verifications. For example, the NICS could use NLP models to analyze documentation and other types of data related to network operations and maintenance, such as software patches, system logs, and user manuals. The models could help to identify potential vulnerabilities, provide insights into system performance and usage, and inform decisions around resource allocation and network optimization.

6.6.3 Automation

AI models can be used to automate the verification process, reducing the amount of manual effort required. This can increase the speed and accuracy of the verification process, as well as reduce the potential for human error.

6.7 Horizontal Scaling

To scale horizontally, the system can be designed to be distributed across multiple nodes, with each node responsible for processing a portion of the data and running the AI models. The nodes can communicate with each other to share information and collaborate on the decision-making process. One approach to scaling the system horizontally is to use containerization and container orchestration technologies, such as Docker and Kubernetes. Another approach is to use cloud-based services to provide the necessary computing resources for the system.

6.8 Consistency

In a blockchain network, consistency refers to the agreement between nodes in the network on the state of the blockchain. AI models could potentially improve the consistency of the network by detecting and preventing malicious behavior, improving consensus algorithms, enhancing data validation, monitoring network behavior, and improving network efficiency.

6.9 Overhead

NICS can improve the overhead of a network by using machine learning algorithms to optimize various network processes such as resource allocation, load balancing, and data management. By automating these processes, AI models could reduce the overhead and complexity of network management, freeing up resources for other important tasks.

6.10 Attacks

NICS can improve the defensive capabilities of a network by detecting and preventing malicious behavior in real-time. For example, AI models can be trained on network data to identify patterns of behavior associated with attacks, such as Denial-of-Service (DoS) or Sybil attacks.

6.10.1 Reversal

In theory, NICS can improve the ability of a network to deal with reversal attacks by providing enhanced security measures. For example, AI models can be trained on historical network data to identify potential reversal attacks and to take preventative measures before they occur. By analyzing network traffic, AI models could detect abnormal behavior and respond with countermeasures such as rate limiting or temporary blocking of the attacker's IP address.

6.10.2 Long Range Attacks

NICS can potentially improve the network's defense against long-range attacks by analyzing network activity and detecting anomalies that could indicate an attack. For example, an AI model could monitor the frequency and volume

of transactions in the network, compare them to historical data, and flag any significant deviations as potential signs of an attack.

7 Dynamic Subsharding

7.1 Description

Dynamic sub-sharding is a method of breaking down a large network into smaller, more manageable parts, or shards, that can be processed in parallel. In a blockchain network, dynamic sub-sharding can be used to improve the scalability, efficiency, and security of the network by distributing the workload, reducing the overhead, and reducing the risk of centralization.

In dynamic sub-sharding, the network is divided into multiple sub-networks or shards, each of which can have its own consensus mechanism, data structure, and security mechanism. The process of dividing the network into shards can be done dynamically, based on the current state of the network, the current workload, the number of nodes, and other factors. The sharding process is managed by the NICS mechanism, which determines the size and number of shards, assigns nodes to shards, and reassigns nodes as needed.

In v0.0.2, DSS integrates seamlessly with Zero-Knowledge Rollups (ZK-Rollups). When a sub-shard processes a batch of transactions, it does not need to broadcast the entire payload to the global network. Instead, the sub-shard generates a succinct ZK proof of the state transition and anchors it to the overarching DAG. This compresses massive amounts of data, offsetting the byte-size increase introduced by Post-Quantum Cryptography signatures.

The main advantage of dynamic sub-sharding is improved scalability, as the network can handle a larger number of transactions and nodes without sacrificing performance. This is achieved by distributing the workload across multiple shards, reducing the overhead, and reducing the risk of centralization. In addition, dynamic sub-sharding can improve the security of the network by reducing the attack surface and making it harder for malicious actors to compromise the network.

Dynamic sub-sharding is a complex process that requires advanced AI models and algorithms to manage the sharding process, ensure the consistency of the network, and prevent attacks. These models must be able to detect anomalies, predict network behavior, and make real-time decisions based on the current state of the network.

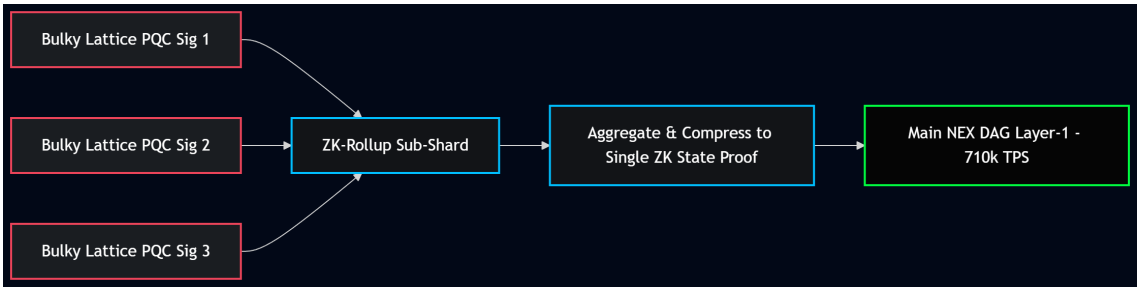


Figure 5: Dynamic Sub-Sharding & Post-Quantum Signature Compression

7.2 DSS AI Models

7.2.1 Swarm AI

Swarm AI models could potentially be useful in controlling the dynamic sub-sharding system in our proposed network design. Swarm AI models are based on swarm intelligence, which is a decentralized approach to problem solving that relies on collaboration between simple agents. This approach aligns well with the principles of sub-sharding, which involves dividing the network into smaller, more manageable pieces.

In the context of dynamic sub-sharding, swarm AI models could be used to help manage and optimize the process of dividing the network, as well as to ensure that sub-shards are functioning optimally and making the best use of resources. For example, swarm AI algorithms could be used to determine which transactions should be assigned to which sub-shards, to monitor the performance of sub-shards in real time, and to make adjustments to the sub-sharding process as needed to ensure the network is operating efficiently.

Particle Swarm Optimization (PSO) - This model uses a swarm of particles to search for optimal solutions. It could be useful in finding the best partitioning of the network into sub-shards.

Ant Colony Optimization (ACO) - This model uses the behavior of ants to solve optimization problems. It could be useful in finding the best routes for data and transactions to traverse the network.

Artificial Bee Colony (ABC) - This model uses the behavior of bees to solve optimization problems. It could be useful in finding the most efficient way to distribute work across the network and ensure the network remains balanced.

Artificial Fish Swarm Algorithm (AFSA) - This model uses the behavior of fish to solve optimization problems. It could be useful in finding the most efficient way to organize the network and respond to changes in load and demand.

8 Proof of Experienced Stake Consensus

8.1 Description

Proof of Experienced Stake (PoES) combines the traditional Proof of Stake (PoS) mechanism, which takes into account the amount of stake held by a validator, with a historical stake component. By considering both the current stake and the historical stake, PoES aims to provide an additional layer of security and to prevent malicious actors from accumulating a large stake over time and then potentially disrupting the network.

8.2 Terminology

bonds: In Proof of Work, the cost of hardware and electricity is considered a capital expense. The miner invests in these resources and dedicates them to a specific branch within the blockchain. In contrast, a bond in a blockchain represents a validator's commitment to use their coins as collateral while validating transactions.

slashing: This solution aims to address the nothing-at-stake problem in Proof of Stake systems [4]. If a validator publishes proof of voting for a different branch, their bond can be destroyed, which serves as an economic disincentive to prevent validators from confirming multiple branches.

super majority: A "super majority" refers to a vote by two-thirds of the validators based on the weight of their bonds. This indicates that the network has achieved consensus, and at least one-third of the network would need to vote maliciously for the branch to be considered invalid. This implies that the cost of an attack would be one-third of the market cap of the coin.

8.3 Bonding

When a bonding transaction is made, a certain amount of coins is transferred to a bonding account that belongs to the user. These coins cannot be used or spent and must stay in the account until the user decides to remove them. The user can only remove coins that have timed out. A super majority vote by the current stakeholders must confirm the sequence before the bonds become valid.

8.4 Voting

The Proof of Experience generator is expected to publish a state signature at regular intervals. Bonded identities must confirm this signature by publishing their own signed version. A vote is cast with a simple "yes" and there is no option to vote "no." If a super majority of bonded identities have voted within a specified time frame, the branch will be deemed valid.

8.5 Unbonding

If a certain number of votes (N) are missing, the coins associated with those votes are considered stale and cannot be used for voting anymore. To remove them, the user can initiate an unbonding transaction. N is calculated dynamically based on the proportion of stale votes to active ones, which means that N increases as the number of stale votes grows. This feature helps the larger branch to recover more quickly than the smaller one in case of a significant network partition.

8.6 Elections

When a failure of the PoE generator is detected, a new election is held. The validator with the highest voting power or public key address (if there's a tie) becomes the new generator, with their historical success rate taken into account. Super majority of confirmations are required for the new sequence. If the new leader fails before there are enough confirmations, the next highest validator is selected, and a new set of confirmations is required. Vote switching requires a higher PoE sequence counter and the inclusion of the votes to switch to avoid being slashable. Once a PoE generator is established, a Secondary can be elected to take over transaction processing. The platform is designed so that the Secondary becomes Primary in case of exception or on a predefined schedule, with lower rank generators being promoted.

8.7 Election Triggers

8.7.1 Forked Proof of Experience Generators

PoE generators are created with an identity that signs the generated sequence, and a fork can only happen if the PoE generator's identity has been compromised. A fork is identified when two distinct historical records are published on the same PoE identity.

8.7.2 Runtime Exceptions

If a hardware failure, bug, or intentional error occurs in the PoE generator, it could generate an invalid state and publish a signature that doesn't match the validators' results. Validators will share the correct signature through gossip, triggering new elections. Validators who accept an invalid state will have their bonds slashed.

8.7.3 Network Timeouts

An election would is triggered in the event of a network timeout.

8.8 Slashing

If a validator votes for two separate sequences, it will result in slashing of the bonded coins and adding them to the mining pool as a proof of malicious vote. However, a vote that includes a previous vote on a contending sequence is not eligible as proof of malicious voting. In such cases, the currently cast vote on the contending sequence will be removed instead of slashing the bonds. Additionally, if a vote is cast for an invalid hash generated by the PoE generator, the validator's bonds will be slashed. It is expected that the generator may randomly generate an invalid state, triggering a fallback to Secondary.

8.9 Secondary Elections

Proposals to elect Secondary and lower ranked Proof of Experience generators can be made and approved. These proposals include a timeout, and if a super majority of the vote approves the motion before the timeout, the Secondary is elected and will take over duties as planned. The Primary can initiate a smooth handover to the Secondary by adding a message to the generated sequence signaling the handover, or by creating an invalid state that prompts the network to fallback to the Secondary. In case the Primary fails, a Secondary that has been elected will be considered the first option during an election.

8.10 Availability & Self-DDoS Mitigation

In our network design, we face a trade-off between Consistency and Availability in the event of partitions. We prioritize Availability but also have a measure of time to ensure Consistency with reasonable timeouts. Our Proof of Experienced Stake verifiers stake a certain amount of coins to vote for transactions and their stake is locked up as a transaction in the PoE stream. Their votes, also entered as transactions, allow us to determine how much time has passed between each vote and how long each verifier has been unavailable during a partition.

A critical vulnerability in legacy networks is the incentive for nodes to intentionally go offline (self-DDoS) to avoid voting on contentious forks, thereby avoiding slashing penalties while their stake "ages." PoES mathematically eliminates this vector. "Experience" is not a measure of time; it is a measure of *active, correct participation*. If a validator goes offline (intentionally or otherwise), their Experience multiplier decays rapidly. To handle partitions, we dynamically unstake unavailable verifiers within reasonable human timeframes. The speed of this process is dependent on

the number of verifiers available, with faster unstaking for more available verifiers and slower unstaking for fewer available verifiers.

8.11 Recovery

In the system we propose, the ledger can be fully recovered from any failure, which means that any person can create a valid fork by adding newly generated hashes and transactions to any random point in the ledger. If there are no verifiers available for this fork, achieving 2/3 super majority consensus will take a very long time, and a significant number of hashes must be added to the ledger for full recovery with no available validators. Only after unstaking all the unavailable validators will new bonds be able to validate the ledger.

NICS could potentially improve recovery by automating the process of identifying and recovering from failures. For example, AI models could be trained to detect patterns in the ledger data that indicate a failure has occurred and automatically initiate recovery procedures. Additionally, AI could be used to optimize the recovery process by identifying the most efficient way to add new hashes and transactions to the ledger to achieve consensus.

8.12 Finality

In our network design, the PoE system enables verifiers to observe past events with a reasonable degree of certainty. The PoE generator produces a stream of messages that requires all verifiers to submit their signatures of the state within a specified time limit, which is currently set at 500ms and can be adjusted based on network conditions. Each verification is entered into the stream, allowing everyone in the network to validate that all verifiers submitted their votes within the required timeout without directly observing the voting process.

8.13 Attacks

8.13.1 Tragedy of Commons

In our network design, the PoES verifiers confirm the state hash produced by the PoE generator. However, there is a risk that verifiers may not perform any work and simply approve every generated state hash. To prevent this, the PoE generator must randomly inject an invalid hash at intervals, and any voters for this hash should be penalized. Once the invalid hash is detected, the network should immediately promote the secondary elected PoE generator. Verifiers are required to respond within a short timeout, which should be set low enough to discourage malicious behavior and prevent an observer from getting their votes into the stream quickly.

8.13.2 Collusion with the PoE Generator

A verifier who colludes with the PoE generator would have advance knowledge of when the invalid hash will be generated and could choose not to vote for it. However, this situation is similar to the scenario where the PoE identity has a larger verifier stake. Regardless, the PoE generator still bears the responsibility of producing the state hash. NICS could potentially be used to monitor the behavior of verifiers and detect any patterns or anomalies that may indicate collusion with the PoE generator. For example, machine learning algorithms could be trained on historical data to identify unusual voting behavior or sudden changes in voting patterns that could be indicative of collusion.

8.13.3 Censorship

In case one-third of bond holders refuse to validate new bond sequences, censorship or denial of service may occur. The protocol can prevent this attack by dynamically adjusting the rate at which bonds become stale. In the event of a denial of service, the larger partition will fork and censor the Byzantine bond holders, while the larger network recovers as Byzantine bonds become stale. The algorithm involves electing a new Leader who censors the Byzantine bond holders, generating a sequence with the Proof of Experience generator until enough Byzantine bonds become stale, and dynamically adjusting the rate at which bonds become stale based on the percentage of active bonds. The smaller Byzantine partition would have to wait much longer to recover a super majority, while the majority fork recovers faster. Once a super majority is established, slashing could permanently punish the Byzantine bond holders.

8.13.4 Long Range Attacks

PoE provides a natural defense against long-range attacks by making it difficult for attackers to recover the ledger from any point in the past. They would need to overtake the valid ledger in time by outpacing the speed of the PoE generator. The consensus protocol serves as a second layer of defense by requiring any attack to take longer than the time it takes



to unstake all the valid validators. This creates an availability gap in the ledger's history. When comparing two ledgers of the same height, the one with the smallest maximum partition can be considered valid based on objective criteria.

8.13.5 ASIC Attacks

There are two opportunities for ASIC attacks - during partition and cheating timeouts in Finality. The rate at which bonds are unstaked during partitions is non-linear, making ASIC attacks inefficient for networks with large partitions. During Finality, ASIC attacks can be carried out by Byzantine validators who collaborate with a PoE generator to inject their votes. The PoE generator can then use its faster ASIC to generate 500ms worth of hashes in less time, allowing for network communication between the generator and collaborating nodes. However, if the PoE generator is also Byzantine, it may communicate the exact counter when the failure is expected to be inserted. NICS could potentially be used to protect the network from ASIC attacks by using machine learning algorithms to detect patterns of behavior that are indicative of ASIC attacks.

8.13.6 DDOS Attacks

AI can be used to protect the network from DDoS attacks by implementing anomaly detection algorithms that can detect and respond to unusual network traffic patterns. These algorithms can be trained using techniques such as semi-supervised or self-supervised learning, even with limited labeled data. Additionally, model fairness indicators in TensorFlow can be utilized to evaluate the effectiveness of the anomaly detection models and ensure that they do not introduce bias or discrimination. By continuously monitoring and analyzing network traffic using these AI-powered tools, the network can proactively defend against DDoS attacks and ensure its availability to legitimate users. [5]

9 System Architecture - Dynamic PoE Cluster Shards

9.1 Components

9.1.1 Leader, Proof of Experience Generator

The Proof of Experience generator is elected as the designated Leader. Its role is to receive transactions from users and produce a unique global order for them through a Proof of Experience sequence. Upon completion of processing each batch of transactions, the Leader generates a signature that represents the final state of the network, alongside a ZK-SNARK proof of any NICS AI inferences, which is then signed with the Leader's post-quantum identity and propagated to the overarching DAG data structure.

9.1.2 State

The theoretical state includes various data structures to represent the current state of the network. For example, there might be a table or database indexed by users' addresses, which contains information such as their account balance and transaction history. Each cell in this table might contain the user's full address and the memory required for this computation, totaling 32 bytes per cell. Additionally, there might be a Proof of Experienced Stake bonds table, which tracks the bonds staked by validators in the network throughout their history, totaling 64 bytes. These data structures are constantly updated as new transactions are added and new validators join or leave.

9.1.3 Verifier, State Replication

To replicate the state of the network, the verifier nodes synchronize with the other nodes on the network and download a copy of the distributed ledger (DL) or DAG data structure. Once the verifier nodes have a copy of the DAG, they verify the validity of transactions and blocks by executing the consensus algorithms and verifying the Leader's ZK-SNARK proofs. This ensures that the verifier nodes have an accurate and up-to-date copy of the state without needing to re-run heavy AI computations.

9.1.4 Validators

These nodes are utilizing Verifiers' bandwidth. They are virtual nodes that can operate on the same machines as the Verifiers or the Leader, or on separate machines that are configured specifically for the consensus algorithm of this network.

9.2 Network Limits

The network is limited by the number of verifiers that can validate each shard, which is directly related to the computational power and memory available to the verifiers and the leader nodes. As the number of shards increases, the number of verifiers required to validate them also increases. The Leader in charge of generating the Proof of Experience sequence is expected to optimize the sequence's efficiency by ordering incoming user packets in the most efficient way possible, minimizing faults and maximizing prefetching.

The network is also limited by the available bandwidth for communication between nodes. On a 1gbps network connection, the maximum number of transactions possible is 1 gigabit per second divided by the packet size of 176 bytes, which allows for approximately 710k transactions per second (with some loss expected due to Ethernet framing). While Post-Quantum Cryptography (PQC) signatures are inherently larger than legacy ECDSA signatures, NEX maintains this 176-byte average packet size by utilizing ZK-Rollup compression at the sub-shard level, aggregating thousands of PQC signatures into a single, succinct ZK-proof before broadcasting to the global network. To increase availability, Reed-Solomon codes can be used to code the output and stripe it to the available downstream Verifiers.

9.3 Computational Limits

The computational limits of our network design are determined by the amount of computational power and memory available to the verifiers and the leader nodes. Historically, GPU-based ECDSA verification servers demonstrated experimental results of 900k operations per second. However, by transitioning to Lattice-based PQC (e.g., CRYSTALS-Dilithium) and ZK-SNARKs, the computational bottleneck shifts from elliptic curve cryptography to polynomial multiplication and zero-knowledge proof verification. Because ZK-proof verification is highly parallelizable and computationally lightweight, the theoretical performance ceiling scales linearly with the number of cores available on the verifier nodes.

9.4 Memory Limits

In theory, a basic implementation of the state that uses a 50% full hashtable with 32-byte entries for each account could accommodate 10 billion accounts within 640GB. The system can handle 1.1 $\times 10^7$ writes or reads per second during steady state random access. Based on 2 reads and 2 writes per transaction, memory throughput is capable of processing 2.75 million transactions per second. These measurements were obtained using an Amazon Web Services 1TB $\times$ 1.16 $\times$ large instance.

10 System Architecture - Overarching DAG structure

10.1 Integration with the PoE clusters

A network design combining PoE and DAG technology uses PoE as a method for securing and verifying the historical order of transactions, and DAG for organizing and structuring these transactions into a directed graph data structure.

The PoE generator generates a hash chain of the transactions that are informed by the NICS. The hashes generated are verified via ZK-SNARKs and anchored onto the DAG, allowing for a tamper-evident historical record and artificial system memory of all transactions. The directed graph structure of the DAG represents the relationships between transactions, providing a compact and efficient representation of the network's state.

10.2 Subnetworks & Zero-Knowledge Privacy

A subnetwork, or subnet, is a dynamic set of validators working together to achieve consensus on the state of a set of distributed ledgers. Each DL is validated by one subnet, and a subnet can validate arbitrarily many distributed ledgers. The Experience platform supports the creation and operation of arbitrarily many subnets.

In v0.0.2, subnets natively support **Zero-Knowledge (ZK) Privacy**. Enterprises can deploy private subnets where transaction data is completely obfuscated. The subnet only submits a ZK-proof of state validity to the Default Subnet (the main DAG). This allows for regulatory compliance (e.g., GDPR, HIPAA) where public verification is required, but data exposure is strictly prohibited.

The subnet model offers a number of advantages:

- Reduces network traffic and computational resources, as validators only join subnets they care about.



- Allows for the creation of private subnets validated only by a set of trusted validators.
- Allows for jurisdictional or compliance-based subnets (e.g., validators bound by real-world contracts).

10.3 Virtual Machines

A Virtual Machine (VM) serves as a blueprint for each distributed ledger instance. A VM defines the interface, state, and behavior of a distributed ledger, including the contents of a block, the state transition logic, the APIs exposed, and the data persisted to disk. NEX accommodates a wide range of VMs, allowing for easy migration of existing DLTs and support for multiple scripting languages.

10.4 Bootstrapping

The first step in participating in Experience is bootstrapping. The process occurs in three stages: connection to seed anchors, network and state discovery, and becoming a validator.

10.4.1 Seed Anchors

Crypto networks typically employ DNS seed nodes, known as seed anchors. In Experience, the majority of seed anchors must be correct for the network to function properly. Any node can become a seed anchor, and no set of seed anchors can control network entry since nodes can attach to any set of seed anchors to discover the latest network of peers.

10.4.2 Network and State Discovery

After connecting to the seed anchors, a node requests the latest set of state transitions, which we refer to as the accepted frontier (the set of vertices that are accepted but have no accepted children). The state transitions accepted by a majority of the seed anchors are defined as accepted, and the correct state is extracted by synchronizing with the sampled nodes. [6]

11 System Architecture - Network Intelligence Control System

11.1 Components

The system architecture of the NICS can be broken down into the following high-level components:

11.1.1 Data Collection & Analysis

Gathers data from a range of sources, including network devices, servers, and applications, and analyzes it to obtain insights into network performance, traffic patterns, usage trends, and security threats.

11.1.2 Network Monitoring

Continuously monitors the network and triggers notifications when problems are identified. It consists of tools like Network Performance Monitoring (NPM), Security Information and Event Management (SIEM), and User and Entity Behavior Analytics (UEBA).

11.1.3 Network Control

Implements the policies and rules that govern the network, ensuring optimal network performance and preventing security threats.

11.1.4 Network Orchestration

Automates network management tasks, such as configuration changes, policy updates, and provisioning/deprovisioning of network resources.

11.1.5 User Interface & Integration

Provides dashboards, reports, and analytics for network administrators, and enables integration with third-party tools and systems such as ERP, CRM, and SIEM systems.

12 Autonomous Security & The NixGuard Integration

While the NEX Network is a generalized Web4 infrastructure, its flagship implementation and primary architectural driver is decentralized cybersecurity.

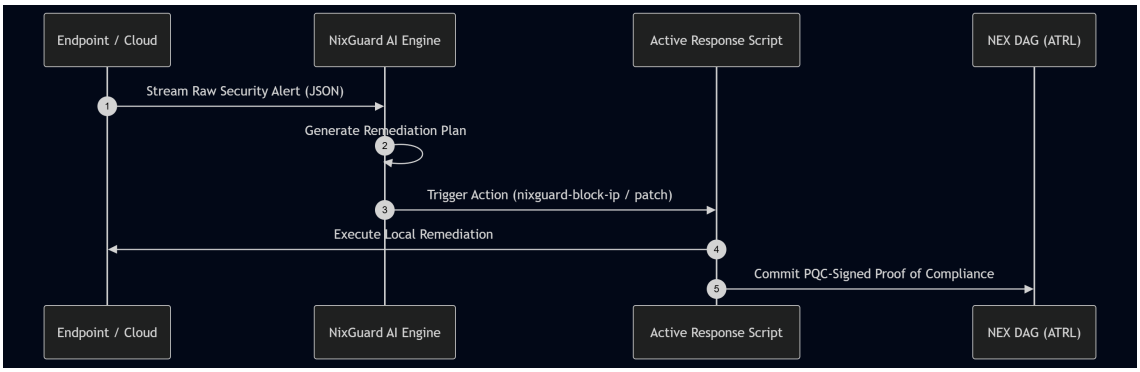


Figure 6: Autonomous Threat Remediation Ledger (ATRL) Sequence Flow

12.1 The Autonomous Threat Remediation Ledger (ATRL)

Modern enterprise security relies on autonomous AI agents (such as **NixGuard**) to detect, triage, and remediate threats in real-time across cloud, SaaS, and endpoint environments. However, AI-driven remediation introduces a critical compliance challenge: non-repudiation and auditability.

The NEX Network serves as the **Autonomous Threat Remediation Ledger (ATRL)**. When a NixGuard agent executes a remediation (e.g., isolating a compromised server or revoking a malicious OAuth token), the state transition is recorded immutably on the NEX DAG. This provides enterprises with cryptographic proof of compliance for SOC 2 and ISO 27001 audits.

12.2 Decentralized Threat Intelligence

By utilizing the NEX Network, NixGuard agents globally can share anonymized, ZK-proven threat telemetry. The NICS AI models aggregate this global threat intelligence via Federated Learning. If a NixGuard agent in one jurisdiction detects a novel zero-day attack, the cryptographic signature of that threat is instantly propagated through the NEX DAG, immunizing all other enterprise subnets globally in milliseconds.

13 Conclusion

In conclusion, the network we have designed using a combination of PoE, PoES, DSS, NICS, and other innovative protocols has several advantages over existing blockchain and distributed ledger technologies. Our network provides a more scalable, efficient, and secure platform for decentralized applications and services.

Crucially, by integrating Post-Quantum Cryptography (PQC) and Zero-Knowledge Machine Learning (ZK-ML) in v0.0.2, NEX solves the existential threats facing legacy Web3 networks: quantum vulnerability and AI consensus determinism. As the foundational layer for autonomous security agents like NixGuard, the NEX Network is positioned to be the premier infrastructure for the decentralized, democratized, and secure Web4 ecosystem. We hope that our work will inspire further research and development in this field, and ultimately contribute to a more open, decentralized, and democratized web for everyone.



Acknowledgements
The author thanks Dr. Harry Gaebler for his help and mentorship throughout the writing and publishing process.

References

[1] E. Cline, *Ready Player One*. New York City: Crown Publishing Group, 2011.

[2] A. Yakovenko, "Solana: A new architecture for a high performance blockchain," 2017.

[3] V. Buterin and V. Griffith, "Proof of stake faq: Long range attacks," Jan 2019.

[4] V. Buterin, "Slasher: A punitive proof-of-stake algorithm," Jan 2014.

[5] V. Basu, "An approach to detect ddos attack with a.i.," Oct 2020.

[6] K. Sekniqi, D. Laine, S. Buttolph, and E. Gün Sirer, "Avalanche platform whitepaper," Jun 2020.