EchoPulse Protocol Structure

1. Symbol Set (Σ)

The EchoPulse Key Encapsulation Mechanism (KEM) operates on a finite set of 256 distinct, abstract symbols, denoted as $\Sigma=\{s0,s1,...,s255\}$. These symbols are non-algebraic in nature and do not possess inherent mathematical properties beyond their identity. Their function within the protocol is solely to define and trigger transitions between states within the state transition graph. Each symbol serves as a discrete input that, when processed in a given state, deterministically leads to a subsequent state as defined by the transition function.

2. State Transition Graph G(V,E)

The core of the EchoPulse KEM is a finite, directed graph $G(V,E)$, where V represents the set of states and E represents the set of directed edges connecting these states. The protocol defines a total of $|V|=1024$ distinct states. Each directed edge $e=(u,v)\in E$ originates from a state $u\in V$ and terminates at a state $v\in V$. Every edge in E is labeled with a unique symbol $\sigma\in\Sigma$, indicating that the transition from state u to state v occurs upon processing the symbol σ in state u.

The state transition is formally defined by the function $\delta:V\times\Sigma\rightarrow V$. For a given current state $u\in V$ and an input symbol $\sigma\in\Sigma$, $\delta(u,\sigma)=v$ specifies the next state $v\in V$. The construction of the graph $G(V,E)$ and the definition of the transition function δ can be achieved through various deterministic methods. These include pseudorandom generation seeded by a master secret or a deterministic algorithm based on a publicly known seed. The specific method of graph construction is a design parameter that influences the security properties of the KEM.

3. Encapsulation Process

The encapsulation process begins with the receiver possessing a public key that implicitly defines a specific public state $vpub\in V$. This public state is reached by applying a predetermined sequence of symbols, derived from a public seed, as transitions starting from a fixed initial state $v0\in V$.

To encapsulate a secret key, the sender generates a fresh, cryptographically secure random sequence of symbols $r=(\rho1,\rho2,...,\rho m)$, where each $\rho i\in\Sigma$. This random symbol sequence r constitutes the ciphertext C. The sender then applies the sequence of symbols in r as consecutive transitions starting from the receiver's public state $vpub$. The resulting final state after processing all symbols in r is denoted as $venc$:

$venc=\delta(\delta(...\delta(vpub,\rho1),\rho2)...,\rho m)$

The shared secret key K is derived by applying a cryptographic hash function H to the concatenation of the final state $venc$ (represented as a fixed-length encoding of the state) and the random symbol sequence r:

$K=H(venc||r)$

The encapsulated key K and the ciphertext $C=r$ are then transmitted to the receiver.

4. Decapsulation Process

The receiver possesses a private key SK, which corresponds to a specific private sequence of symbols. When applied as transitions starting from the same initial state $v0$ used during public key generation, this private symbol sequence deterministically leads to a private state $vpriv\in V$.

Upon receiving the ciphertext $C=r$, the receiver applies the sequence of symbols in r as consecutive transitions starting from their private state $vpriv$. The resulting final state after processing all symbols in r is denoted as $vdec$:

$vdec=\delta(\delta(...\delta(vpriv,\rho1),\rho2)...,\rho m)$

The receiver then derives a candidate shared secret key K' by applying the same cryptographic hash function H to the concatenation of the final state $vdec$ (represented in the same format as $venc$) and the received random symbol sequence r:

$K'=H(vdec||r)$

For correct decapsulation, the design of the state transition graph and the relationship between the public and private key sequences must ensure that $venc=vdec$ with a high probability, leading to $K'=K$.

5. Synchronization & Determinism

The adaptive mutation of the state transition graph $G(V,E)$ over time is a critical component of the EchoPulse protocol. This mutation is performed deterministically and is synchronized between the communicating parties using a shared secret salt and a session index. Both the sender and the receiver maintain a consistent view of the graph at any given session. The mutation function $\mu(G,salt,session\_index)\rightarrow G'$ defines how the graph evolves. This function takes the current graph G, a shared salt, and the current session index as input and outputs the mutated graph G'.

All state transitions within the EchoPulse protocol, including those involved in key generation, encapsulation, and decapsulation, must be entirely deterministic. Given a specific state and an input symbol, the resulting next state is uniquely defined by the transition function δ. This deterministic nature is essential for both parties to correctly derive the shared secret key.

6. Figures (optional)

[ASCII diagrams could be included here to illustrate the state transitions, but are omitted due to the text-based format requirement.]

Version 1.0 — Protocol Layer Specification — EchoPulse Initiative