EchoPulse Key Derivation Model

1. Derivation Principle

The EchoPulse Key Encapsulation Mechanism (KEM) derives the shared secret key K through a process involving the cryptographic hashing of a final state $v \in V$ of the state transition graph $G(V,E)$ and the sequence of abstract symbols $r \in \Sigma m$ that led to this state during the encapsulation or decapsulation procedures. This derivation principle is intentionally designed to avoid reliance on algebraic structures or operations, thereby mitigating potential vulnerabilities associated with algebraic leakage that might be exploited by quantum algorithms. Furthermore, the entropy of the derived key is primarily driven by the path traversed within the state graph, influenced by the randomness of the symbol sequence r and the structural complexity of $G(V,E)$.

2. Hash-Based Key Generation

The shared secret key K is generated by applying a cryptographically secure hash function H to the concatenation of the final state v and the symbol sequence r:

$$K=H(v||r)$$

Here, H is a robust cryptographic hash function such as SHA-3, chosen for its strong security properties. The final state v, which is an element of the vertex set V (where $|V|=1024$), is encoded as a fixed-length binary representation of its index (e.g., a 10-bit integer). The symbol sequence $r=(\rho 1,\rho 2,\ldots,\rho m)$ is a fixed-size array of m symbols, where each $\rho i \in \Sigma$ and $|\Sigma|=256$ (each symbol can be represented by 1 byte). The concatenation operation $||$ denotes the sequential joining of the binary encoding of v and the byte sequence of r before being input to the hash function H.

3. Collision and Preimage Resistance

The selection of the cryptographic hash function H is critical for the security of the key derivation process. H must exhibit strong preimage resistance, ensuring that it is computationally infeasible to find an input (a state v and a symbol sequence r) that hashes to a specific target key K. Additionally, H must possess strong collision resistance, making it computationally infeasible to find two distinct pairs $(v1,r1)$ and $(v2,r2)$ that hash to the same key.

While collisions in the symbolic paths within the graph $G(V,E)$ (i.e., different symbol sequences leading to the same state) might theoretically occur, the cryptographic hash function H applied to the final state and the specific random symbol sequence r significantly reduces the security implications of such path collisions at the key derivation level. Furthermore, the dynamic mutation of the graph $G(V,E)$ across sessions, as described in the "EchoPulse State Graph Design" specification, further mitigates the risk of an attacker exploiting any potential structural weaknesses that might lead to predictable path collisions over extended periods.

4. Salt, Nonce, and Session Parameters

To enhance the randomness and security of the key derivation process, a shared secret salt can be incorporated into the initial graph construction and potentially as an additional input to the hash function H. A nonce, generated by the sender during encapsulation, could also be included in the hashed input to provide per-session uniqueness and prevent key reuse even if the final state and the symbol sequence r were to repeat.

The session index plays a crucial role in synchronizing the mutation of the state transition graph between the communicating parties. This synchronization ensures that both sender and receiver are operating on the same graph structure for each key exchange, preventing replay attacks that might exploit a static graph. The session index itself is not directly used in the key derivation hash but is essential for establishing the context (the specific version of the graph) in which the final state v is reached.

5. Path Entropy Model

The security of the derived key is intrinsically linked to the entropy of the path traversed within the state transition graph. Assuming a relatively uniform distribution of transitions, the approximate entropy in bits achieved after traversing a path of length l symbols, where each state has an average out-degree d, can be estimated as:

$$\text{Entropy} \approx l \times \log 2(d)$$

Given the graph parameters specified in the "EchoPulse State Graph Design" (out-degree between 16 and 32), a path length m of sufficient size (e.g., $m \geq 25$) can be chosen to achieve a target security level of approximately 128 bits. For instance, with an average out-degree of 24 ($\log 2(24) \approx 4.58$), a path length of around 28 symbols would yield approximately $28 \times 4.58 \approx 128$ bits of entropy. The length m of the random symbol sequence r (which dictates the path length during encapsulation and decapsulation) is a crucial parameter that needs to be chosen to meet the desired security target.

6. Replay Resistance & Robustness

The inherent uniqueness of the randomly generated symbol sequence r for each encapsulation process, combined with the session-based mutation of the underlying state transition graph, significantly reduces the risk of replay attacks. Even if an attacker were to intercept a ciphertext $C=r$, its utility would be limited to the specific session for which it was generated due to the subsequent graph mutations.

The deterministic nature of the state transitions and the key derivation process ensures robustness, particularly in low-power embedded systems where maintaining precise timing or complex state management can be challenging. Since the key derivation is solely based on the final state reached through deterministic transitions and the received symbol sequence, both parties will arrive at the same shared secret key K (assuming correct transmission and synchronized graph state) without requiring complex synchronization protocols beyond the session index.

7. Conclusion

The EchoPulse key derivation model, based on hashing the final state reached through symbolic transitions and the random symbol sequence used to reach it, offers a lightweight and pattern-resistant approach to generating shared secrets. By avoiding algebraic operations and relying on the entropy introduced by the path traversal and the cryptographic hash function, it presents a potential avenue for secure key exchange in post-quantum environments, particularly for resource-constrained devices. The integration of session-based graph mutation