Rapid Block Gossiping Mechanism

Background:
In a specialized blockchain architecture, there is a unique sequencer producing blocks every 0.2 seconds. These blocks are then batched for 30 seconds before they are written to a decentralized data storage, which is the ultimate source of truth for this system. Full nodes serve light clients requiring fast transaction responses, making real-time synchronization between nodes and the sequencer essential. Full nodes can sync with the data storage, but that would mean an unacceptable delay for the light clients.

Objective:
Design a basic mechanism that provides near-real-time synchronization of full nodes with the sequencer.

Key Focus Areas:

1. Efficiency: Minimal overhead on the sequencer's operations.
2. Fault Tolerance: Ensure synchronization even if some nodes disconnect.
3. Scalability: Accommodate an increasing number of full nodes.
4. Latency: Reduce the time for a block to reach most full nodes after creation.

Instructions:

1. **Block Propagation:** Briefly describe how blocks can be shared rapidly from the sequencer to full nodes.
2. **Handling Disconnections:** In a few points, outline how you'd manage node disconnections or failures.

3. **Security Considerations:** List potential threats and a brief description of how to mitigate them.
4. **Scalability:** Jot down your thoughts on accommodating more full nodes as the network grows.
5. **Node Resynchronization After Outage:** Consider a situation where the sequencer and a node (Node A) are at block height X. At the next moment, the sequencer advances to height X+1, but Node A experiences a network outage. When Node A's connection is restored, the sequencer is at height X+100. Given the time it takes for Node A to sync, it's still at height X+100 when the sequencer sends the block at height X+111. How does Node A catch up and synchronize with the sequencer after such an outage?
6. **Evaluation:** Suggest two or three metrics you'd track to gauge the success of your mechanism.

Submission:
Provide a concise report showcasing your design and thoughts on the instructions given.

Evaluation Criteria:

1. Practicality: Is the design feasible for real-world implementation?
2. Relevance: Does the solution address the main objectives?
3. Clarity: Are the ideas well-presented and easy to understand?