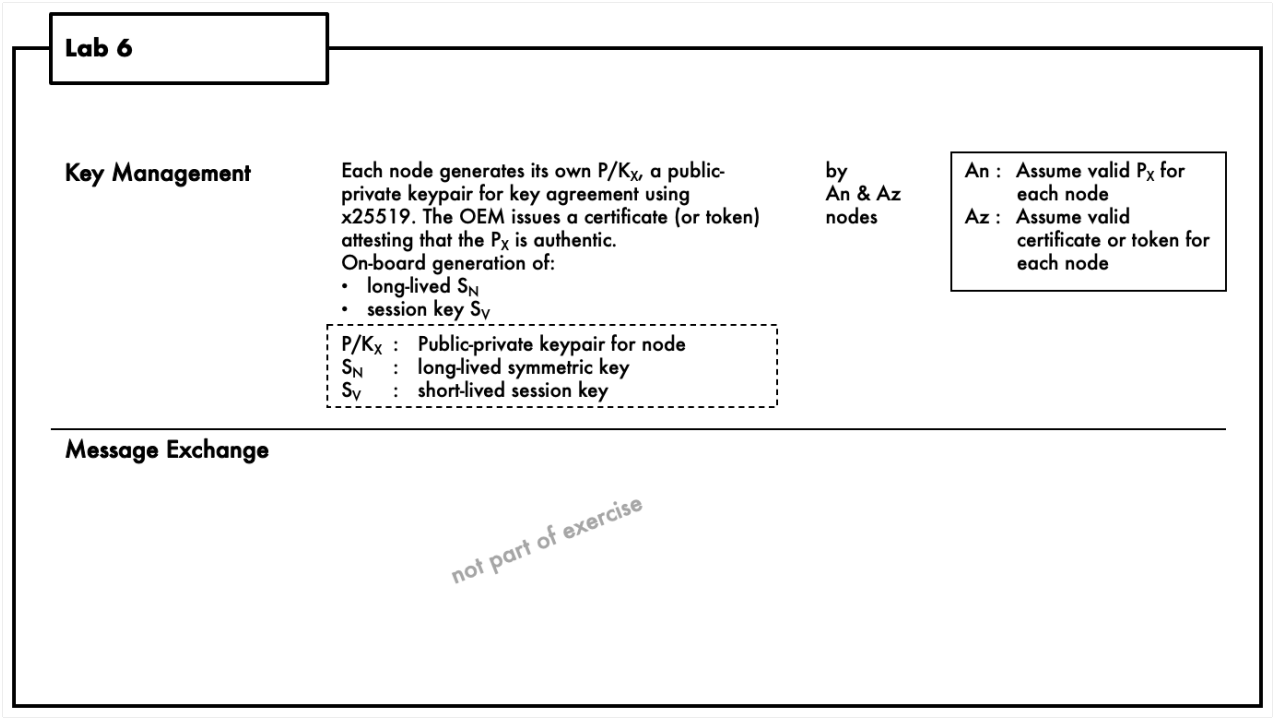

Scenario : The team needs solutions to the "hard part of security: key management".

The product engineers would like a simple PoC showing how we can create a symmetric key without having to inject it into they system.

They also would like to see how session keys could be created on each power cycle.

Exercise 1 : Use x25519 to create a shared secret and SHA256 to create a key, call this key S_N (symmetric network key). Each node should print to the console the AES-128 key that it derived.

Exercise 2 : Hardcode an S_N value and have the nodes exchange entropy that is used to create a session key, S_V.

```
Sn = bytes.fromhex("00000000 11111111 22222222 33333333")
```

Exercise 3 : Add two more nodes, C and D. Hardcode an S_N value into all four nodes. Demonstrate how these four nodes can arrive at a common shared S_V.

Hints/Notes:

- Use CAN FD to move the public keys between nodes
- For exercises 1 & 2 your network has two nodes: {A, B}.
- For exercise 3 your network hash four nodes: {A, B, C, D}.
- For exercise 2 the nodes can exchange 8 bytes; for exercise 3 they can exchange 4 bytes, each
- There is *no* need to send messages from the 1Hz generators in this lab.