Lab 5

Key Management

A shared key exists between nodes 1 & 2.
Each node has its own P/K and other's P.

by
OEM

An : nodes with S_{12}
Az : all messages

S_{12} : symmetric key shared by two nodes
P/K_i : public/private keypair for node i.

Message Exchange

`([]64x100){F32|32 I32}`

F :
I :
C :
An : access to S_{12}
Az : all messages
Av :

Choose Your Scenario In this lab you get to choose your scenario. You have two weeks to work on this. Please plan to present a demo to the instructor in week 2.

Scenario 1 -- Stream : Say it turns out that we can't afford to use byte 8 for security overhead, but we still need a general purpose security protocol. You are allowed to introduce some new messages that can be used to wrap a stream of messages.

The stream is point-to-point, i.e., there are just two nodes participating in the secure stream. Assume that both nodes have established a shared key S_{12}, have their own P_x / K_x, and know the other nodes P_x.

Exercise 1 : [DEFEND] Consider how to defend the stream, 10 messages at a time, by protecting the stream, as opposed to individual messages. You are given some leeway to create and define new messages to convey security overhead, e.g., a message that marks the boundaries of each section of the protected stream.

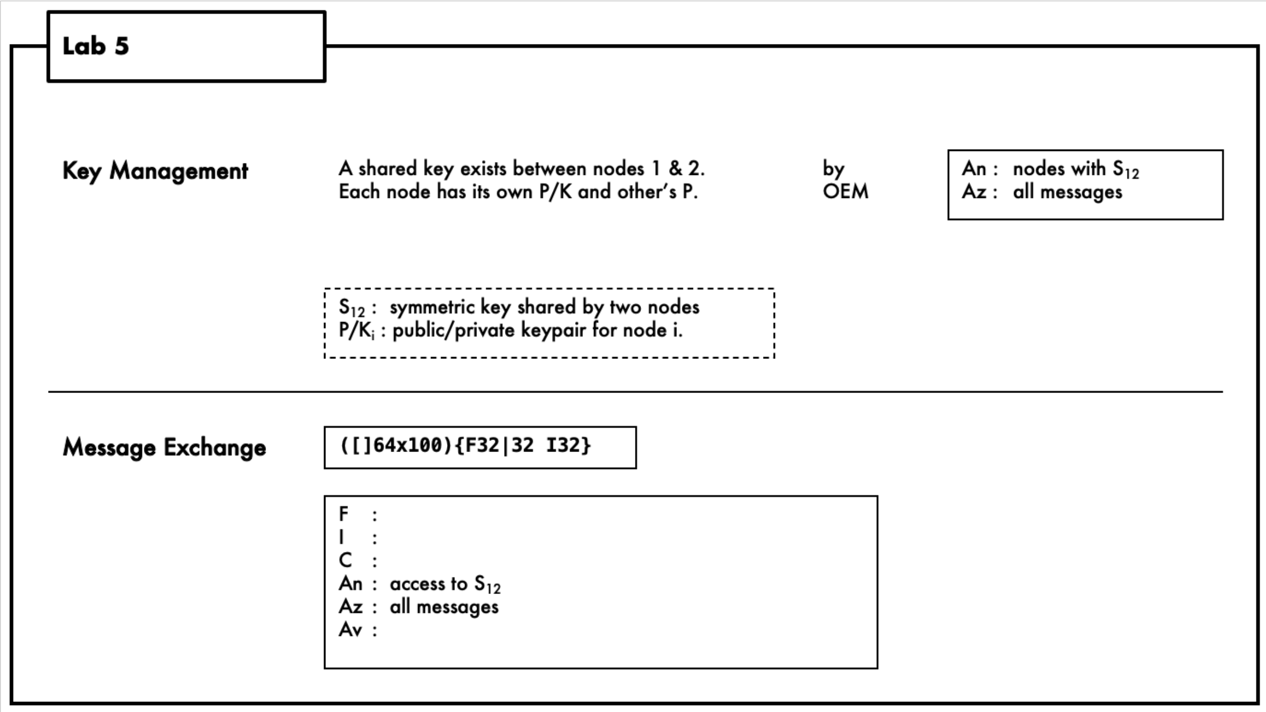
- You can assume Classic CAN or CAN FD.
- You can use the message exchange format shown above, or change it.
- You can use any symmetric and/or asymmetric techniques you find useful.
- You can use any hashing scheme you find useful.

While this method dramatically reduces the overhead, it comes with the (possibly huge) downside of using data before you find out that it has been tampered with.

Also, without some careful consideration, it could suffer from fragility in cases of lost or out-of-order messages, e.g., a message gets "lost" at a receiver (say due to a burst of messages that the microcontroller can't keep up with).
Discuss how to minimize these issues.

Exercise 2 : Create a PoC.

- Show how a dropped (or out-of-order) message is handled.
- Show the detection of tampered with messages.



Scenario 2 -- In Place : Say it turns out that we can't afford to use byte 8 for security overhead, but we are willing to create a non-general purpose protocol. You know that each message has at least 2 bytes that aren't being used.

There are any number of nodes participating in the protocol. Assume that all nodes have established a shared key S_{12}, have their own P_x / K_x, and know the other nodes P_x.

Exercise 1 : [DEFEND] Consider how to defend the messages with message exchange format as shown above.

- You can assume Classic CAN or CAN FD.
- You can use the message exchange format shown above, or change it.
- You can use any symmetric and/or asymmetric techniques you find useful.
- You can use any hashing scheme you find useful.

What is nice about this approach is that you confirm a message is valid *before* using the data.

However, without some careful consideration, it could suffer from fragility in cases of lost freshness synchronization.
Discuss how to minimize these issues.

Exercise 2 : Create a PoC.

- Show how a dropped (or out-of-order) message is handled.
- Show the detection of tampered with messages.