DSM: Tutorial 7

Question 1

$n - k + t + 1 = 9 - 5 + 2 + 1 = 7$ locks.

It does not matter what kind of failure occurs: in order to detect a conflict, an RM must be able to both send and receive messages. All of crash, crash+link and failstop failures, and receive, send and general omissions prevent at least one of send and receive. Only one RM must report conflict successfully, so Byzantine failures that fail to report a conflict are equivalent to the other failure types. Byzantine failures that cause a *false positive* on conflict detection will not damage data integrity, but will slow down the system by preventing feasible locks from being obtained. Alternatively, write locks could be applied to even more RMs to make a majority decision on conflict.

Question 2

- 2PC has a fairly short protocol and minimal communication, so performance is not unreasonably affected.
- 2PC is abort-biased, so in the face of failures throughput will be poor.
- 2PC uses timeouts, so longer delays will occur between failure and the ability to act upon it.
- 2PC's speed is bottlenecked by the slowest or most faulty host, so it does not give performance transparency.
- Other types of transparency (location, replication, fault, etc.) are maintained, if the 2PC protocol is handled by some front-end.

Question 3

Uniform timed reliable broadcast could be used to send the result of votes in the 2PC method. If a server receives `C-COMMIT` then all others have received it too and it can commit; if it does not receive `C-COMMIT` before the timeout then no server will have, so it can abort.

This would require a lot more message-passing to implement.