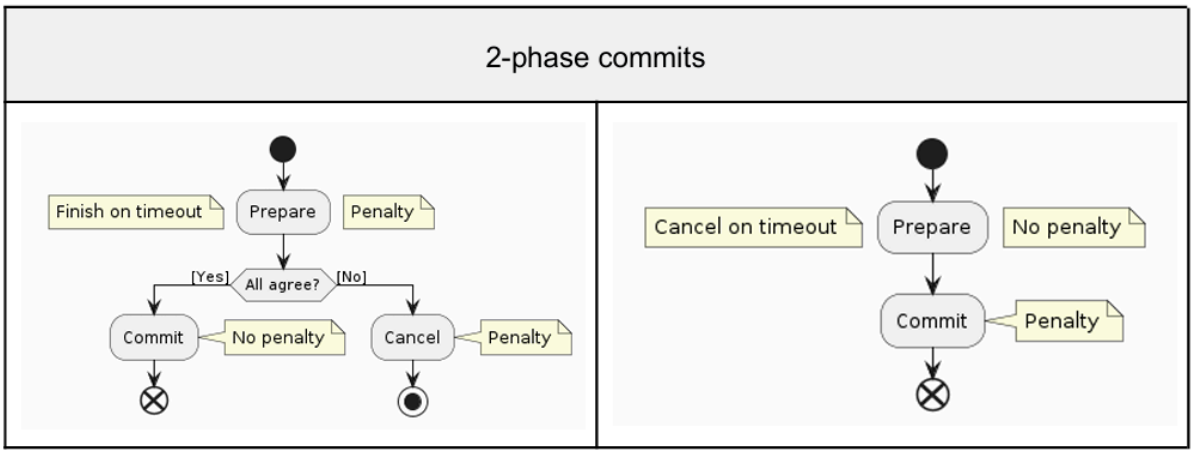
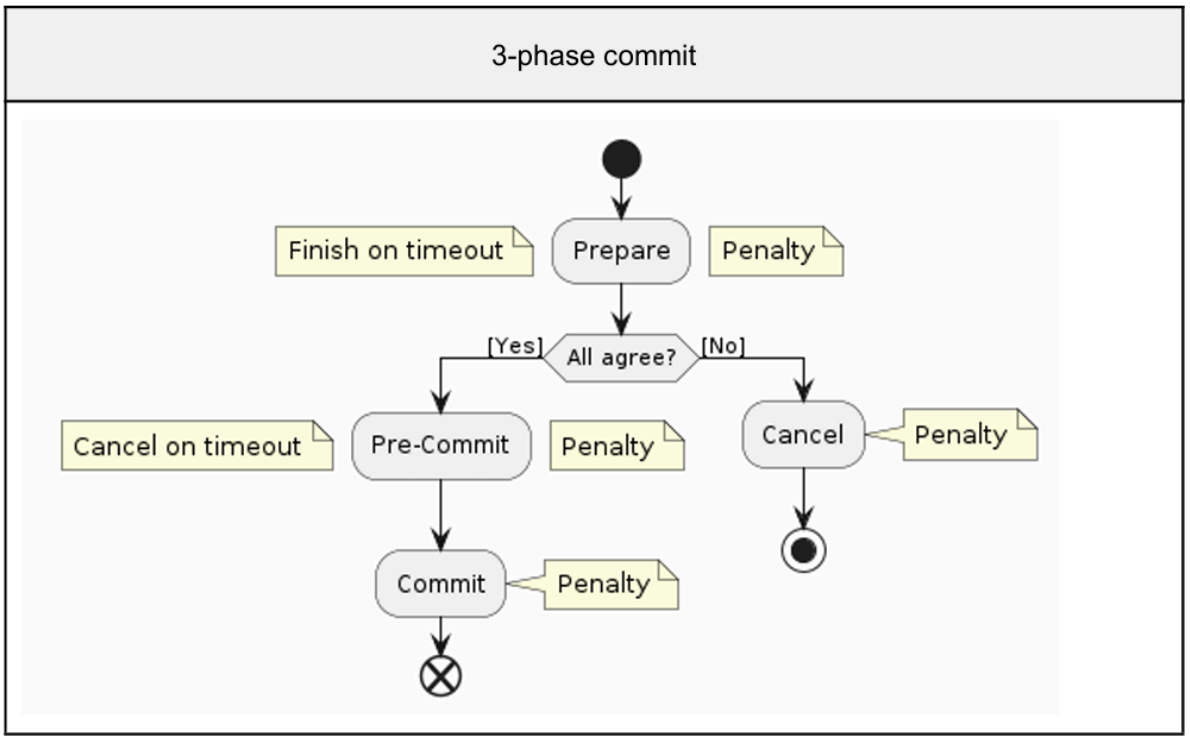
3-phase commit for multihop payments

The two 2-phase commits that are possible for multi-hop payments (cancel-on-timeout or finish-on-timeout) will always have a penalty on one of the phases but never on the other, and either 2-phase commit will have on opposite phases. To avoid a non-attacker getting stuck with the whole payment as the penalty, the penalty can be done in "chunks ". But when the penalty is done in chunks, the combined time until the payment has fully timed out tends to increase, and this makes the phase with no penalty vulnerable to Denial of Service (DoS) attacks. The solution is to combine both 2-phase commits, and use the one with the penalty on the first phase as the first phase, and the one with the penalty on the second phase as the second phase. This requires an intermediary phase that shifts from finish-on-timeout to cancel-on-timeout, resulting in a 3-phase commit. This is the logical way to do multi-hop payments.





"Cancel" in the "finish-on-timeout" 2-phase commit or the 3-phase commit is issued by the sender (not by intermediaries or recipient) and has to be authenticated by the sender (with a hash lock) to avoid an intermediary lying about it (otherwise intermediary could receive the full payment via the finish-on-timeout). This is analogous to how "Commit" is from the recipient (not from intermediaries or sender) and has to be authenticated as well, to avoid an intermediary lying about it (otherwise intermediary could receive the payment).

The 3-phase commit deters DoS attacks in all scenarios except when the person paying the penalty and the person receiving it are the same person. This scenario is easily deterred by adding a fee on top of the payment, paid out in proportion to how long payment was stuck. This is separate from the 3-phase commit (and note the 2-phase commits also have this problem, but, they rely on short timeout which means this problem never becomes a problem).

UML for 2-phase commit images

```
@startuml
start
:Prepare;
note right: No penalty
note left: Cancel on timeout
:Commit;
note right: Penalty
end
@enduml
```

```
@startuml
start
:Prepare;
note right: Penalty
note left: Finish on timeout
if (All agree?) then ([Yes])
:Commit;
note right: No penalty
end
else ([No])
:Cancel;
note right: Penalty
stop
endif
@enduml
```

UML for 3-phase commit image

```
@startuml
start
:Prepare;
note left: Finish on timeout
note right: Penalty
if (All agree?) then ([Yes])
:Pre-Commit;
note left: Cancel on timeout
note right: Penalty
:Commit;
note right: Penalty
end
else ([No])
:Cancel;
note right: Penalty
stop
endif
@enduml
```