Example: Patching actor state

In this example, we show how you can write an upgrade migration which patches theActorState of an existing actor stored on chain.

More specifically, in this example we want to patch the state of thechainmetadata actor which was deployed at genesis. This actor is used to store blockhashes of the previous blocks on chain. Internally, this actor has the following state:

```

Copy // in fendermint/actors/chainmetadata/src/shared.rs pubstructState{ // the AMT root cid of blockhashes pubblockhashes:Cid, // the maximum size of blockhashes before removing the oldest epoch publookback_len:u64, }

```

At genesis, this actor was deployed withlookback_len of 256. In this migration, we want to change thelookback_len to 512 to extend the lookback history.

Inside this migration function, we need to retrieve the actor state associated with thechainmetadata actor, update itslookback_len to 512, save the new state to the block store and then update the actor state in the state tree.

Our migration function is defined as follows:

```

Copy pubfnpatch_actor_state_func(state:&mutFvmExecState)->anyhow::Result<()> { letstate_tree=state.state_tree_mut();

// get the ActorState from the state tree // letactor_state=matchstate_tree.get_actor(CHAINMETADATA_ACTOR_ID)?{ Some(actor)=>actor, None=>{ returnErr(anyhow!("chainmetadata actor not found")); } }; println!( "chainmetadata code_cid: {:?}, state_cid: {:?}", actor_state.code, actor_state.state );

// retrieve the chainmetadata actor state from the blockstore // letmutchainmetadata_state:State=matchstate_tree.store().get_cbor(&actor_state.state)?{ Some(v)=>v, None=>returnErr(anyhow!("chain metadata actor state not found")), }; println!( "chainmetadata lookback length: {}", chainmetadata_state.lookback_len );

// lets patch the state, here we increase the lookback_len from the default (256) to 512 // chainmetadata_state.lookback_len=512;

// store the updated state back to the blockstore and get the new state cid // letnew_state_cid=state_tree .store() .put_cbor(&chainmetadata_state, Code::Blake2b256) .map_err(|e|anyhow!("failed to put chain metadata actor state: {}", e))?; println!("new chainmetadata state_cid: {:?}", new_state_cid);

// next we update the actor state in the state tree // state_tree.set_actor( CHAINMETADATA_ACTOR_ID, ActorState{ code:actor_state.code, state:new_state_cid, sequence:actor_state.sequence, balance:actor_state.balance, delegated_address:actor_state.delegated_address, }, );

Ok(()) }

```

Once we have finished writing ourUpgrade migration, we can add it to theUpgradeScheduler :

```

Copy letmutscheduler=UpgradeScheduler::new(); letupgrade=Upgrade::new(chain_name, block_height, app_version, patch_actor_state_func); scheduler.add(upgrade);

// when initializing the FvmMessageInterpreter, specify the upgrade schedule letinterpreter=FvmMessageInterpreter:::new( ... scheduler, );

```

Previous Upgrading a subnet Next Example: Upgrading Wasm actor Last updated9 days ago On this page Was this helpful? Edit on GitHub