# get_contract_code_hash

get_contract_code_hash helper function

Contract code hash helper function

To retrieve a Secret Network smart contract's code hash from within a Secret contract call, you can useStargate queries to directly query the on-chain data from inside the contract.

This helper function is particularly interesting if you desire to make complicated contract structures that involve submessages or just cross-contract messages to different Secret Network contracts. With this code snippet, you do not need to supply thecode_hash of each contract that you are going to call in submessage or normal messages. It is sufficient to know the contract address of the other contract, the code snipped will fetch the latest on chaincode_hash for you.

Be aware that contracts can be upgraded on Secret Network! Since this code snippet always fetch thecode_hash directly from the chain without any extra check (which was implictly done by manually supplying thecode_hash ), you need to be careful about silently (perhaps maliciously) upgraded contracts which potentially releveal confidential information. The Secret Networkteam has designed a helper function,get_contract_code_hash , exactly for this purpose.

See an example usage of get_contract_code_hashhere . Simply add the Anybuf package and the"stargate" feature for cosmwasm-std to yourcargo.toml :

```

Copy [dependencies] cosmwasm-std={ package="secret-cosmwasm-std", version="1.1.10", features=["stargate"]} anybuf={version="0.5.0"}

```

And then add the function to your contract:

```

Copy fnget_contract_code_hash(deps:DepsMut, contract_address:String)->StdResult{ letcode_hash_query:cosmwasm_std::QueryRequest=cosmwasm_std::QueryRequest::Stargate{ path:"/secret.compute.v1beta1.Query/CodeHashByContractAddress".into(), data:Binary(Anybuf::new() .append_string(1, contract_address) .into_vec()) };

letraw=to_vec(&code_hash_query).map_err(|serialize_err|{ StdError::generic_err(format!("Serializing QueryRequest: {}", serialize_err)) })?;

letcode_hash=matchdeps.querier.raw_query(&raw) { SystemResult::Err(system_err)=>Err(StdError::generic_err(format!( "Querier system error: {}", system_err ))), SystemResult::Ok(ContractResult::Err(contract_err))=>Err(StdError::generic_err(format!( "Querier contract error: {}", contract_err ))), SystemResult::Ok(ContractResult::Ok(value))=>Ok(value) }?;

// Remove the "\n@" if it exists at the start of the code_hash letmutcode_hash_str=String::from_utf8(code_hash.to_vec()).map_err(|err|{ StdError::generic_err(format!("Invalid UTF-8 sequence: {}", err)) })?;

ifcode_hash_str.starts_with("\n@") { code_hash_str=code_hash_str.trim_start_matches("\n@").to_string(); }

Ok(code_hash_str) }

```