I am facing following error . But can’t understand why? I was trying to execute my own smart contract.

unable to resolve type URL /cosmwasm.wasm.v1.MsgExecuteContract: tx parse error"

full response:

{“tx_response”: {“code”: 2, “codespace”: “sdk”, “data”: “”, “events”:

, “gas_used”: “0”, “gas_wanted”: “0”, “height”: “0”, “info”: “”, “logs”:

, “raw_log”: “unable to resolve type URL /cosmwasm.wasm.v1.MsgExecuteContract: tx parse error”, “timestamp”: “”, “tx”: null, “txhash”: “D88F0E00460E352E0BEE780F221552F42D29992E5A7973488567D0E75AE8AAC7”}}

full Code snippet:

```
async function t() {

const mnemonic =

‘nice same resist skirt cross tone object grass remember then pluck fall’;

const chainId = ‘pulsar-2’;

const cosmos = new Cosmos(lcdUrl, chainId);

cosmos.setPath(“m/44’/529’/0’/0/0”);

cosmos.setBech32MainPrefix(‘secret’);

const address = cosmos.getAddress(mnemonic);

console.log('address: ', address);

const privKey = cosmos.getECPairPriv(mnemonic);

const pubKeyAny = cosmos.getPubKeyAny(privKey);

console.log(‘priv key:’, privKey);

console.log('public key: ', pubKeyAny);

await cosmos.getAccounts(address).then(data => {

console.log('data: ', data);

const CONTRACT_ADDRESS = 'secret14snjfvwxdc4xyl3t4kd5a8u56y5eu9rpzr9hwt';

const CONTRACT_CODE_HASH = 'e31293e76ff207af810dbe93dab6f9428864d392c5341acbc50a5c0e2cbe26b2';

const VIEWING_KEY = 'my viewing key'; const v_key_msg = { set_viewing_key: { key: VIEWING_KEY, }, }; let cw20Contract = 'juno10rktvmllvgctcmhl5vv8kl3mdksukyqf2tdveh8drpn0sppugwwqjzz30z'; //let transferBytes = new Buffer(v_key_msg, 'utf8'); const msgExecuteContract = new message.cosmwasm.wasm.v1.MsgExecuteContract({ sender: address, contract: CONTRACT_ADDRESS, msg: v_key_msg, funds: [{denom: 'uscrt', amount: String(3000000)}], });

const msgExecuteContractAny = new message.google.protobuf.Any({ type_url: '/cosmwasm.wasm.v1.MsgExecuteContract', value: message.cosmwasm.wasm.v1.MsgExecuteContract.encode( msgExecuteContract, ).finish(), });

const txBody = new message.cosmos.tx.v1beta1.TxBody({ messages: [msgExecuteContractAny], memo: '', }); console.log('txBody: ', txBody);

const signerInfo = new message.cosmos.tx.v1beta1.SignerInfo({ public_key: pubKeyAny, mode_info: { single: { mode: message.cosmos.tx.signing.v1beta1.SignMode.SIGN_MODE_DIRECT, }, }, sequence: data.account.sequence, });

const feeValue = new message.cosmos.tx.v1beta1.Fee({ amount: [{denom: 'uscrt', amount: String(25000)}], gas_limit: 200000, });

const authInfo = new message.cosmos.tx.v1beta1.AuthInfo({ signer_infos: [signerInfo], fee: feeValue, });

const signedTxBytes = cosmos.sign( txBody, authInfo, data.account.account_number, privKey, ); cosmos.broadcast(signedTxBytes).then(response => console.log(response));

});
```

}