Informações de Debug do contrato `SpyErcBridge`

Esses são os argumentos que foram passados para o método sym_exec da classe laserEVM:

Python

Código - Observações

- Locais marcados com `# f`:
  Indicam que, durante a execução do contrato analisado, a condição **não foi satisfeita** e, portanto, **não entrou** na condicional associada.

- Locais marcados com `# v`:
  Indicam que a condição **foi satisfeita** e, durante a execução, o contrato **entrou** na condicional associada.

Essas marcações ajudam a identificar o comportamento do contrato em diferentes cenários de execução.

Python

```
def sym_exec(
        self,
        world_state: WorldState = None,
        target_address: int = None,
        creation_code: str = None,
        contract_name: str = None,
    ) -> None:
        """Starts symbolic execution
        There are two modes of execution.
        Either we analyze a preconfigured configuration, in which case the
world_state and target_address variables
        must be supplied.
        Or we execute the creation code of a contract, in which case the
creation code and desired name of that
        contract should be provided.
```

```
        :param world_state The world state configuration from which to
perform analysis
        :param target_address The address of the contract account in the
world state which analysis should target
        :param creation_code The creation code to create the target contract
in the symbolic environment
        :param contract_name The name that the created account should be
associated with
        """
        pre_configuration_mode = target_address is not None
        scratch_mode = creation_code is not None and contract_name is not
None
        if pre_configuration_mode == scratch_mode: # f
            raise ValueError("Symbolic execution started with invalid
parameters")

        log.debug("Starting LASER execution")
        for hook in self._start_sym_exec_hooks:
            hook()

        time_handler.start_execution(self.execution_timeout)
        self.time = datetime.now()

        if pre_configuration_mode: # f
            self.open_states = [world_state]
            log.info("Starting message call transaction to
{}".format(target_address))

self.execute_transactions(symbol_factory.BitVecVal(target_address, 256))
        elif scratch_mode: # v
            log.info("Starting contract creation transaction")

            created_account = execute_contract_creation( # v
                self, creation_code, contract_name, world_state=world_state
            )
            log.info( # v
                "Finished contract creation, found {} open states".format(
                    len(self.open_states)
                )
            )

            if len(self.open_states) == 0: # f
                log.warning(
                    "No contract was created during the execution of
contract creation "
                    "Increase the resources for creation execution
(--max-depth or --create-timeout) "
```

```
                    "Check whether the bytecode is indeed the creation code,
otherwise use the --bin-runtime flag"
                )

            self.execute_transactions(created_account.address) # v

        log.info("Finished symbolic execution") # v
        if self.requires_statespace: # f
            log.info(
                "%d nodes, %d edges, %d total states",
                len(self.nodes),
                len(self.edges),
                self.total_states,
            )

        for hook in self._stop_sym_exec_hooks:
            hook()
```