The following was asked on the Enigma Telegram:

“Hey Guy, I have read the white paper, your original Enigma paper from 2015 and the documentation. I have 2 questions which I couldn’t fully understand.

First, is Enigma writing its own Turing complete language? If so, does it relay on the known logical gates like most of the current languages or is it using a ‘new’ set of gates and logic which are more compatible with homomorphic encryption?

Second, as per my understanding, the private contracts will be executed by nodes which run an isolated VM (virtual machine). These will be equivalent to master nodes.

Assuming this is correct, I assume this is due to enigma running in an adversarial environment. Would it be possible in the future to execute private contracts outside of the docker container? For example, on an IoT device.” - Eliahu

Guy’s response:

“We’re working at a lower layer (the interpreter), so you’d be able to use existing languages. Originally for my thesis, I wrote an MPC interpreter that worked with a restricted version of Python. Now we’re focusing more on WASM.

Basic VM ops are further compiled down to MPC protocols. For example, comparing two integers is done quite differently, although it’s a single instruction.

[In regards to the possibility of running a private contract outside the docker container:] Nothing prevents it. But there is one public network.”

Eliahu’s reply:

“I assumed the actual development of the contract would be abstracted.

Will this interpreter use old fashion methods with classic boolean logic or will it be using a new kind of logic?

I’ve seen some MPC implementations and papers which build on top of a logic specific to homomorphic encryption and have to write the whole logic from the ground up, is this the case here?

Could you elaborate please on why does the contract need to run on an isolated VM?

If I understand correctly, the data will be encrypted trough out its life span.

Are the nodes put in place in order to prevent an adversarial from breaking the encryption by making sure he is ‘well invested’? My understanding from the white paper and the architecture of the testnet was that the code is executed in a docker container acting as a TEE, this is the ‘core.’”

Guy’s response:

“Right, these are separate concerns. A lot has changed since my research days. Today we’re working on two implementations - a TEE version that is out, with a dockerized version to help you get started quickly (but it’s by no means a requirement).

The MPC network is coming out in the next iteration, which will not really require a TEE (well it might still need it at first for correctness, but not privacy).”