This is a question about whether it is possible, given some level of access to other computations on an enclave if you are running something on that enclave, to convert that into full access to any data for any computations on the network.

The attack, in general, works as follows.

Malicious actor M designs a computation that gives them indefinite access to cache memory for a given enclave (assume, for the moment, that such a computation exists given the ability to run arbitrary enigma-acceptable code on the enclave). M then puts this computation out onto the market for a low enough price that nobody honest would be willing to take that computation (eliminating the cost risks of a long-running comp). Compatriot/collaborator C running a node performs the computation as it is broadcast. Compatriot C then takes every computation broadcast to the network (since the computation is broadcast to every node, and the lottery is only checked on result submission as seen in https://enigma.co/protocol/SubsystemArchitecture.html ) and runs them, thus funneling all their info to M.

Pieces necessary for this attack to function:

Memory unsafety: Either caches must be imperfectly cleaned, or tasks must be performable simultaneously

Task direction unsafety: Tasks must be broadcast to every node on the network, rather than only being broadcast to the node winning the worker selection pseudo-random number generation lottery (this design also increases the incentive to break SGX in general, since now a broken SGX machine has access to all computations not just some fraction)

Task side effect unsafety: It must be impossible to detect computations that read information from other computations being run on the same enclave.

Do y'all agree with this overall assessment? Does anybody have any information about attacks run from inside the enclave on other inside-enclave computations?