There are some common misconceptions/superficial arguments about TEE's like SGX and their intended use floating around so we thought we would compile some quickfire responses.

cowritten with @socrates1024

tl;dr

- Myth: "SGX is being deprecated! Trusted hardware is over.

" Actually Confidential Compute is booming.

The direction the trusted hardware industry is headed is going to be useful for web3. SGX is continuing, just on Xeons only. Which is great, to hell with BluRay players and other "Digital Restrictions Management." Alternatives like AMD SEV, and Intel TDX (which is kind of like SEV but built on SGX) are promising too. There's even opportunities for blockchain native trusted hardware as well, but that's a bigger question.

- Myth: TEEs are broken, so we should never use them. Having to trust Intel is a non-starter.

TEEs WILL

be broken again, but they're useful

anyway. The emerging design theme is to only use TEEs where nothing else (ZK, FHE, etc.) can work, to limit the damage… this includes all the "interesting" applications that have global private state. It IS

very difficult to work around this, and the cost is accounted for as an engineering tradeoff. The separation of roles, like Azure+Intel (Cloud + Hardware Vendor) is a compromise between decentralization and credibility. It's hard to exploit the SGX bug supply chain indefinitely, so it prevents bulk surveillance and "scope creep" abuse of data from mev infra startups… these are the real threats.

- Myth: Using a TEE is the easy way out, real cryptography deserves my brain power.

Some smart people have avoided looking at the problem at all, at least in part because "what's the point of working on cryptography if TEEs can just do it all?"

Well, it turns out that TEEs can't do everything, actually we still need a blockchain for proof of publication and related. And, regardless, to prepare for TEE failures, the full fleet of cryptography tools are needed TOGETHER with TEE for a good solution.

Additionally, the use cases being uniquely served by constructions leveraging TEEs merit attention even if the underlying tech doesn't overwhelm you with mathematical sexiness.

1) "SGX Is Failure Prone"

Yes, SGX is failure prone, but

- A failure-prone defense mechanism is better than none at all (and often additive to what you have already).

This is true in blockchains today in which a bunch of centralised actors like builders, relays and solvers are completely trusted actors. It would be great if the security of every system in the crypto industry could be mathematically proven with only the weakest of assumptions. Unfortunately, we do not currently have the tools to do this in a way that is practical for use cases which demand low latency and cost.

- Failure doesn't need to be catastrophic

. Acknowledging that a TEE failure is possible means that one can design a system that doesn't perform too badly when it does happen. One can do this by choosing the right use case (e.g. front-running protection vs. storing funds) for which failure is not the end of the world. One can also employ other techniques like ZK and MPC in ones system for "defense-in depth".

- Many exploits depend on physical access to TEEs, while one can require TEEs to be running in the cloud.

This isn't a beautiful abstract security proof nor is it a shining example of permissionlessness. Practically speaking, however, its much less likely that a cloud provider with reputation at stake will carry out an attack on their customers and requiring machines to be running in the cloud may be a reasonable concession or stepping-stone for certain use cases.

- Some of the exploits that we have seen in the recent past have been due to mistakes in implementation, not flaws in the TEEs.

Similarly, exploits have been exacerbated by inefficient protocols for dealing with disclosed vulnerabilities.

- TEEs are improving.

Just like ZK was too slow for most use cases in the past, TEEs are early in an arc of development. Many vulnerabilities in SGX stem form Intel's decision to build SGX on top of their existing CPU architecture. Future designs such as this proposal would not suffer from the same flaws. TEEs may never be mathematically perfect, but the frequency of failures need not be high.

2) SGX (TEE) Requires Trust In Intel (Manufacturer)

Intel does have the ability to create fake "spy SGX" chips (similar to EGXhere). However,

- If they did decide to cross that line, it seems unlikely that a corporation of such scale with a huge reputation at stake would choose to do this for some MEV profits

. A PRISM-like government backdoor is more likely to be used for legitimate issues of national security. If the value secured by SGX in the crypto industry did grow large enough to warrant such misbehaviour from Intel, we could consider designing and manufacturing our own TEEs in a more palatable way.

- Similar to the arguments from (1), the risk of Intel joining the dark side is negligible in comparison to the risk of current centralised actors misbehaving.

Perhaps for some apps this risk is still too great, but those applications should be built relying more heavily on other techniques.

- TEEs running in the cloud would need the cloud provider and Intel to collude,

making such an attack even less likely.

- The network can be made resillient to failures in single TEE architectures

by employing multiple TEEs from different manufacturers (e.g. sharing a secret between SGX and SEV or requiring one of multiple heterogenous nodes to be honest).

3) Intel Is Discontinuing SGX

- This is false. Yes, SGX on consumer hardware is being deprecated, but Intel's Xeon chips for enterprise and cloud use is continued.
- Even if this were true, there are other TEE manufacturers that provide ~alternatives such as SEV.

4) TEEs Are Too Slow For Latency-Sensitive Apps

It depends. TEEs are clearly much faster than ZK and simply running a computation in a TEE can be close to as good as fast as without it. Overhead comes in when additional techniques like ORAM and oblivious algorithms are required to bolster the privacy provided by the TEE. The need for such mitigations depends on the use case. For example, a DEX swapper only trying to hide the size or direction of their trade may not require any such techniques at all.

5) MPC/FHE Can Do Everything TEEs Can

Today, the answer is certainly no from a practical standpoint. It's not obvious whether this will ever be true.

- These techniques are currently very slow and resource intensive
- MPC and threshold FHE both have honest-majority assumptions which differ from that of TEEs
- TEEs can do some neat things like make HTTP calls