What is front-running and MEV?

Front-running is when an attacker sees a pending transaction in the public mempool and submits their own with a higher fee to execute first, profiting from the victim's intended action. MEV (Maximal Extractable Value) is the broader term for value that can be extracted by reordering, inserting, or censoring transactions within a block, by validators or specialized bots. Common forms include front-running, back-running, and sandwich attacks around a victim trade. The defenses are design-level: commit-reveal schemes, slippage limits, and private transaction routing.

HOW IT WORKS

01 How it works and example

Common MEV/front-running patterns:

- Front-run: a bot sees a large buy that will move a price and buys just before it, then sells into the victim's buy.
- Sandwich attack: the bot places a buy before and a sell after the victim's trade, profiting from the price impact the victim causes and worsening the victim's execution.
- Back-run: act immediately after a known state change (for example, arbitrage right after a big swap).
- Liquidation/oracle races: race others to a profitable on-chain event.

Documented for defensive context.

> INTENTIONS ARE PUBLIC
>
> *On a public mempool, your transaction is visible and reorderable before it executes. Anything profitable to do "around" your transaction can and will be done by bots, so designs must not assume private intent.*

HOW TO DEFEND

- Use commit-reveal schemes so the intended action is hidden until it is committed, removing the information advantage.
- Enforce slippage limits and deadlines on trades so a sandwich cannot push execution to a bad price.
- Use private transaction routing / MEV-protection relays that keep transactions out of the public mempool.
- Design order-insensitive logic where possible, and batch or auction-based mechanisms that neutralize ordering.
- Audit for ordering dependence, identify where being first or last changes the outcome.

SOURCES

[1] OWASP Smart Contract Top 10

[2] Ethereum.org: Smart contract security

[3] SWC Registry: Smart Contract Weakness Classification