SafeSwap Design

Original discussion at [Design] SafeSwap Module - Modules - Safe Community Forum

Summary

Allow users to signal an intent to trade tokens in their Safe via a SwapRequest, and take advantage of MEV Searchers to execute it.

Background

A non crypto-native user's first interaction with Web3 will often be trying to trade an ERC20 token. However, the UX in making a trade is unsatisfactory for those who have limited DeFi experience. These users are overwhelmed with the complex decision-making process that goes in to trying to perform an optimal trade on a decentralized exchange (DEX).

This process is can be enough to give the user a sense of choice paralysis, where they choose to simply not interact with Web3 at all. SafeSwap attempts to remove the overburdening barriers-to-entry that these users face when attempting to execute trades.

Goals

- Provide a simple, accessible, and secure UX for swapping tokens directly from their Safe.
- Reduce the amount of Web3 & DeFi knowledge required to swap tokens and participate in the trading ecosystem.

Design

After the SafeSwap module is attached to a Safe, users can signal the intent and ability for someone else to transfer the assets that are held within their Safe in order to make the desired swap via a SwapRequest creation:

struct SwapRequest { bool cancelled; bool executed; address fromToken; address toToken; uint256 fromAmount; uint256 toAmount; uint256 deadline; }

With this, a user effectively says:

"I have X $TKA in my wallet, and want Y $TKB instead."

This intent can then be used by MEV Searchers to incorporate into their trading strategy, who will actually to do the transfer of the desired tokens when it meets their needs.

For MEV arbitrage, typical strategies involve multiple swaps across a range of DEXs and token pairs. Example 1 and Example 2 do 9 token swaps

in a single transaction:

DEX Swap → DEX Swap → … → DEX Swap → DEX Swap

By incorporating active SwapRequests from multiple Safes into their strategy, there will be even more opportunities for profitable transactions by using this new avaliable liquidity:

DEX Swap → SwapRequest Swap → … → DEX Swap → SwapRequest Swap → DEX Swap

To help signal this intent to MEV Searchers, an event gets emitted which they can incorporate as part of their strategy. They can also iterate over an array of SwapRequests (ensuring to filter cancelled

and executed

swaps appropriately).

If an MEV Searcher can use an active SwapRequest, they can executeSwapRequest() on the module, which will do the appropriate transfers.

Implementation

A quick proof-of-concept: GitHub - mattstam/safeswap: A proof-of-concept module that allows token swaps directly from your Safe wallet.

User Experience

The main is to massively improve the UX of swapping ERC20 tokens by reducing the level of knowledge and decisions required to make an optimal trade.

To see how SafeSwap achieves this, consider this scenario for a new Web3 user:

“I was airdropped 10 UNI tokens, now I want to exchange them for WETH”

Without SafeSwap:

1. Choose CEX or DEX
2. If CEX: incur additional fees, trust a third party to handle your assets
3. If DEX:
4. Choose an appropriate DEX that has this pair available (Uniswap, Balancer, CoWSwap, …)
5. If multiple pairs exist on different DEXs, weigh pro/cons of each
6. If concerned with front-running:
7. Learn how submit Flashbots bundles for private TX
8. Learn how submit Flashbots bundles for private TX
9. Choose an appropriate DEX that has this pair available (Uniswap, Balancer, CoWSwap, …)
10. If multiple pairs exist on different DEXs, weigh pro/cons of each
11. If concerned with front-running:
12. Learn how submit Flashbots bundles for private TX
13. Learn how submit Flashbots bundles for private TX
14. If CEX: incur additional fees, trust a third party to handle your assets
15. If DEX:
16. Choose an appropriate DEX that has this pair available (Uniswap, Balancer, CoWSwap, …)
17. If multiple pairs exist on different DEXs, weigh pro/cons of each
18. If concerned with front-running:
19. Learn how submit Flashbots bundles for private TX
20. Learn how submit Flashbots bundles for private TX
21. Choose an appropriate DEX that has this pair available (Uniswap, Balancer, CoWSwap, …)
22. If multiple pairs exist on different DEXs, weigh pro/cons of each
23. If concerned with front-running:
24. Learn how submit Flashbots bundles for private TX
25. Learn how submit Flashbots bundles for private TX
26. Calculate slippage and additional fees to use for the UNI swap
27. Submit swap

With SafeSwap:

1. add SwapSwap Module (if not already added)
2. Calculate the appropriate about of WETH for the UNI
3. Submit swap

All of these steps can have frontend incorporated to make the UX seemless. (0) adding modules already has support, and (1) can look up current trade ratios to suggest an appropriate price.

Benefits & Drawbacks

Pros:

- No necessary understanding of finance or DeFi protocols
- No interactions with external contracts
- Only ever interact with trusted, secure Safe + SafeSwap Module contracts, both of which Safe can provide a UI for
- Only ever interact with trusted, secure Safe + SafeSwap Module contracts, both of which Safe can provide a UI for
- Lower possible gas cost initial gas costs
- Storage write + event emit to signal trade
- Searcher pays gas for the actual transfers
- Storage write + event emit to signal trade
- Searcher pays gas for the actual transfers
- Zero slippage (you specify an exact tokenOut

)

- No trade fee*
- Limited gas fee*
- Expiration/cancellation flexibility (e.g. GTC)
- Set-and-forget experience
- No address whitelist management (for every DEX address)
- No potential for getting sandwich attacked

Cons:

- Needs x amount of users to adopt before searchers add these to their strategy
- Usually not executed as quickly as using a DEX directly

Challenges

The main barrier for getting this scheme to work at scale is getting enough MEV Searchers to include it in their arbitrage bot logic.

To overcome this, this design will take advantage of Safe's unique characteristics:

1. popularity
2. indexability

1. Popularity

For MEV Searchers to look for these opportunities, a sufficient number of users have to be utilizing the module. This is a classic two-sided marketplace problem, where the initial effort required is getting both sides sufficient usage. Consider the example of ride-sharing app in a new city:

- without drivers, riders never use the app
- without available riders, nobody becomes a driver

By using the popularity of Safe, along with the ease-of-use of attaching new Modules, it is possible for this scheme to gain widespread adoption.

The upside is, once both sides of the marketplace have reached sufficient capacity, this scheme runs itself with no necessary intervention. As more MEV Searchers include these swaps, the UX becomes better as SwapRequests will be fulfilled at a quicker rate.

2. Indexablity

This is a property that MEV Searchers need to be able to easily create a local cache of all possible swaps, which makes building a strategy viable utilizing SwapRequests.

This is similar to when a MEV Searcher needs to cache all known UniswapV2 pairs. So they look at UniswapV2Factory.

Safe utilizes a similar factory, and tracking down existing Safes is easy (which was useful for the SAFE airdrop). MEV searchers will already have experience

with this pattern, which should help them adapt.

Risks

If a sufficient number of SwapRequests is not achieved, then few MEV Searchers will include it in their arbitration bot logic, and users will be dissatisfied that their SwapRequest never gets executed.

To circumvent this, it may make sense to offer a SAFE token incentivization program that rewards users for executing creating SwapRequests or getting them executed. It may also make sense to choose to reward MEV Searchers for executing, but incentivizing just the Safe users should be enough.

Questions

Are there any similar protocols?

Closest comparison is with “meta-DEX” like CoWSwap, which also:

- utilizes existing DEX protocols
- abstracts away gas cost
- avoids MEV sandwich attacks

But SafeSwap is different in a few very important ways:

- doesn’t require additional

off-chain actors

- removes any “external” calls to protocols

Will this be as cost-effective as an equivalent, perfectly-timed DEX trade?

Generally, no. Since the MEV Searcher needs to make enough profit to pay for overhead, the actual DEX price at the time a SwapRequest is executed is always going to be at a better ratio

than what the SwapRequest is for.

But this gap will be minimal due to the competitiveness of MEV Searchers, and will only shrink as more MEV Searchers include this in their arbitration bot logic.

Future

SwapRequests are generalizable to all smart contract wallet implementations, and therefor should be made as an EIP to preserve compatibility and interoperability.