This was inspired by the recent front-end attack against BadgerDAO. Though this post focuses solely on securing Metamask and other browser-extension based wallets I'm hopeful that - with some discussion and smart ideas - it can be expanded to cover other types of wallets too.

For front-end attacks, the root issue at play is that usersdo not really know whether what they are signing is legit. How the information is presented in the Metamask popup doesn't really matter since in daily use most users just click through as quickly as they can anyway. Plus, having to double check the information one is seeing would just make for terrible UX.

I propose a simple solution whereby Metamask double-checks the information on behalf of the user. Broadly speaking:

1. Dapp makes a transaction signing request, which goes through to Metamask.
2. Metamask obtains the current browser tab URL. I'm assuming it's unlikely the user is able to switch to a different tab before the request reaches Metamask

.

1. Metamask uses keccak256(URL domain)

as a key into an on-chain (chain being whatever chain the tx is for) lookup table which lists the expected contract addresses for that domain.

1. Thus, Metamask is able to confirm whether the address that is being approved as a token spender is the expected one, or whether the contract being called in the tx is the expected one.

Regarding point (3), the on-chain lookup table would ideally be deployed at the same address on every chain. Multi-chain dapps would need to ensure their lookup data is kept up-to-date on all their supported chains. And perhaps there is some synergy to work with existing lookup table such as ENS, haven't yet through this through fully.

Clearly this would be an opt-in system - if Metamask is unable to find any on-chain registry data for the domain then it would inform the user of this whilst still letting the user proceed with the signature.

A dapp author would ideally create an entry for their domain in the lookup table prior to launching their dapp front-end. Only their account would be able to make updates to that domain's list (and obviously, a change of ownership would be possible at any time). Eventually the existing dev tooling would integrate this as an optional deployment step.

The benefits of this solution are two-fold:

- Prevent front-end attacks in the use-cases where this can be applied
- Enabling automatic verification without user input, thereby not disrupting the existing UX

The effectiveness of this solution relies on 3 assumptions which I think are quite reasonable and realistic:

- Metamask hasn't been compromised
- The browser hasn't been compromised
- The dapp developer is able to own the lookup table mapping for their dapp domain

There are obviously issues:

- It only works for Metamask and other browser-extension wallets
- It relies on proprietary browser APIs which can change
- It proposes extra work for Dapp developers
- It requires a bit more engineering though to cover use-cases which involve Dapps that deploy contracts on behalf of users and which then need to send txs involving those newly deployed contracts.

The solution is by no means complete and there are probably pitfalls I haven't thought of yet.