StrategyCurveYCRV.sol

TLDR: 2 plausible issues

Issue: Skip controller fees

Severity : Low

It's possible to skip any fees paid to the `controller` in `withdraw` and `harvest_want_deposit` by just going to the functions `withdrawAll` and `deposit` respectively. This may be an intentional design, at least with respect to `harvest_want_deposit` path and `deposit`.

However, it is less clear to me with `withdrawAll` and `withdraw`.
Q: Do we need a fee with `withdrawAll`? Since controller is performing the action too.

Issue: `Controller`, `Governance` and `Strategist` can be set to `0x0`

Severity : Low/Medium

There's currently no validation for setting these addresses with the setter functions. This may be an intentional design to burn the addresses, although I highly doubt that is true for `Controller` given the necessity of it for many major methods.

General

- pragma is not fixed. Consider ^0.6.2 -> 0.6.2
- double-check if const addresses are actually correct as there is more than a handful
  - [x] want -> Curve.fi yCrv token (https://etherscan.io/address/0xdf5e0e81dff6faf3a7e52ba697820c5e32d806a8#code)
  - [x] pool -> Curve.fi yCrv Liquidity Gauge (https://etherscan.io/address/0xfa712ee4788c042e2b7bb55e6cb8ec569c4530c1#code)
  - [x] mintr -> Curve Token Minter (https://etherscan.io/address/0xd061d61a4d941c39e5453435b6345dc261c2fce0#code)
  - [x] crv -> Curve ChaDAO Token (https://etherscan.io/address/0xd533a949740bb3306d119cc777fa900ba034cd52#code)
  - [x] uni -> Uniswap V2: Router 2 (https://etherscan.io/address/0x7a250d5630B4cF539739dF2C5dAcb4c659F2488D#code)
  - [x] weth -> WETH9 (https://etherscan.io/address/0xc02aaa39b223fe8d0a0e5c4f27ead9083c756cc2#code)

- [x] dai -> Dai Stablecoin (https://etherscan.io/address/0x6b175474e89094c44da98b954eedeac495271d0f#code)
- [x] ydai -> yearn: yDAI token (https://etherscan.io/address/0x16de59092dae5ccf4a1e6439d611fd0653f0bd01#code)
- [x] curve -> Curve.fi y Swap (https://etherscan.io/address/0x45F783CCE6B7FF23B2ab2D70e416cdb7D6055f51#code)

`constructor`

- sets `governance` and `strategist` to `msg.sender`, `controller` to parameter
- lgtm

`deposit`

- moves `want` token from contract to `pool` through doing two step `safeApprove` before calling `pool.deposit` of entire balanc
- lgtm

`withdraw(IERC20)`

- `controller` only function
- function overloading, another func called `withdraw(uint)`
- withdraws any non `want`, `crv`, `ydai`, `dai` ERC20 to `controller`

`withdraw(uint)`

- `controller` only function
- function overloading, another func called `withdraw(IERC20)`
- if `balance` of `want` of contract currently on hand is less than `_amount`, attempts to withdraw the difference from `pool` through `_withdrawSome`.
- **relies on `Gauge(pool).withdraw` to throw if `balance - _amount` is insufficient on `pool`. Usually this would be an issue as interface is not a guarantee of implementation, but since `pool` address is already known, OK**
- fee is set at 0.5%
- fee transferred to some `Controller.rewards()` address.
- remainder is tossed to `Controller.vaults(want)`
- lgtm

`withdrawAll`

- `controller` only function
- pretty much the same as `withdraw` except no fees?

`harvest`

- `strategist` and `governance` only
- looks like it could be expensive...
- mints `mintr` into `pool`
- if `crv` balance > 0, starts uniswap order for `crv -> weth -> dai`, for max 30 min. some major slipping may happen here, but i think that's understood with uniswap
- if `dai` balance > 0, moves `dai` to `ydai` vaults
- if `ydai` balance > 0, moves `ydai` to `curve` liquidity pool
- if `want/yCrv` balance > 0, takes 0.05% for `Controller.rewards()` and sends the rest to `deposit()`.

`_withdrawSome`

- lgtm, possible unit issues with how it works, but double checked with const addresses

`balanceOfWant`, `balanceOfPool`, `balanceOf`

- can be set to `external` visibility since unused by contract

`setGovernance`

- no validation, could go to `0x0`, intentional design?

`setController`

- no validation, could go to `0x0`, intentional design?

`setStrategist`

- no validation, could go to `0x0`, intentional design?