SMART CONTRACT AUDIT

CryptoKitties 3D

PROJECT:
CRYPTOKITTIES

SUMMARY

There is some code that goes agains common and best practices which is reduce code comprehensibility for the community. Interface TokenProtect should be renamed to ITokenProtection and renounceOwnership uses uncommon code for instance. It is recommended to use well-known community code practices to increase readability of the code.

METHODOLOGY

Main tests list:

- Best code practices
- ERC20/BEP20 compliance (if applicable)
- Logical bugs
- General Denial Of Service(DOS)
- Locked ether
- Private data leaks
- Using components with known vulns
- Weak PRNG
- Unsed vars
- Uncheked call return method
- Code with no effects
- Function visibility
- Use of deprecated functions
- Authorization issues
- Re-entrancy
- Arithmetic Over/Under Flows
- Hidden Malicious Code
- External Contract Referencing
- Short Address/ Parameter Attack
- Race Conditions / Front Running
- Uninitialized Storage Pointers
- Floating Points and Precision
- Signatures Replay
- Pool Asset Security (backdoors in the underlying ERC-20)

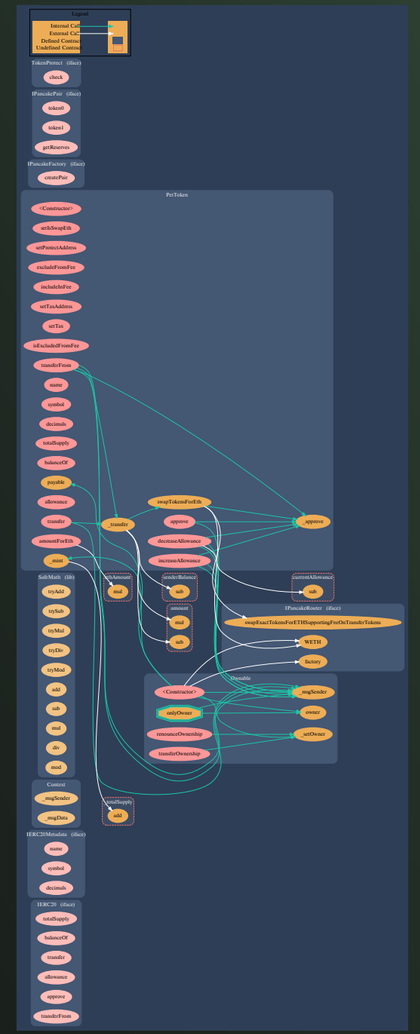
STRUCTURE OF CONTRACT
PETTOKEN.SOL

Usage of SafeMath library is not necessary and can be removed due to solidity 0.8.0.

Contract methods analysis

function setIsSwapEth(bool _is_swap)
Function can be declared external. Function should emit an event

function setProtectAddress(address _address)
Function can be declared external. Function should emit an event



Pic. 1.1.
PetToken.sol

function excludeFromFee(address account)
Function can be declared external. Function should emit an event

function includeInFee(address account)
Function can be declared external. Function should emit an event

function setTaxAddress(address _taxAddress)
Function can be declared external. Function should emit an event

function setTax(uint256 _taxFee)
Function can be declared external. Function should emit an event

function isExcludedFromFee(address account) public view returns (bool)
Function can be declared external

function amountForEth(uint256 ethAmount) public view returns(uint256 tokenAmount)
function can be reduced to getAmountsOut. function can be declared external

function name() public view virtual override returns (string memory)
Vulnerabilities not detected

function symbol() public view virtual override returns (string memory)
Vulnerabilities not detected

function decimals() public view virtual override returns (uint8)
Vulnerabilities not detected

function totalSupply() public view virtual override returns (uint256)
Vulnerabilities not detected

function balanceOf(address account) public view virtual override returns (uint256)
Vulnerabilities not detected

function transfer(address recipient, uint256 amount) public virtual override returns (bool)
Vulnerabilities not detected

function allowance(address owner, address spender)
public view virtual override returns (uint256)
Vulnerabilities not detected

function approve(address spender, uint256 amount)
public virtual override returns (bool)
Vulnerabilities not detected

function increaseAllowance(address spender, uint256
addedValue) public virtual returns (bool)
Vulnerabilities not detected

function decreaseAllowance(address spender, uint256
subtractedValue) public virtual returns (bool)
Vulnerabilities not detected

function swapTokensForEth(uint256
tokenAmount,address _taxAddress)
Vulnerabilities not detected

function amountForEth(uint256 ethAmount) public view
returns(uint256 tokenAmount)
Vulnerabilities not detected

function _mint(address account, uint256 amount)
Vulnerabilities not detected

STRUCTURE OF CONTRACT
PETTOKENPROTECT.SOL

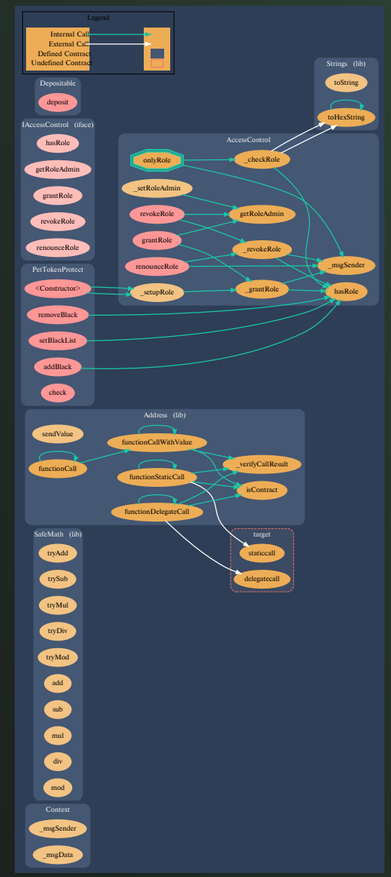
Contract methods analysis

function addBlack(address _userAddress)
Function can be declared external.
Recommended to use onlyRole modifier for access control instead of checking in require()

function removeBlack(address _userAddress)
Function can be declared external.
Recommended to use onlyRole modifier for access control instead of checking in require()

function setBlackList(address[] memory _addressList,bool _bool)
Function can be declared external.
Recommended to use onlyRole modifier for access control instead of checking in require()

function check(address sender, address receiver, uint256 amount)
Function can be declared external.

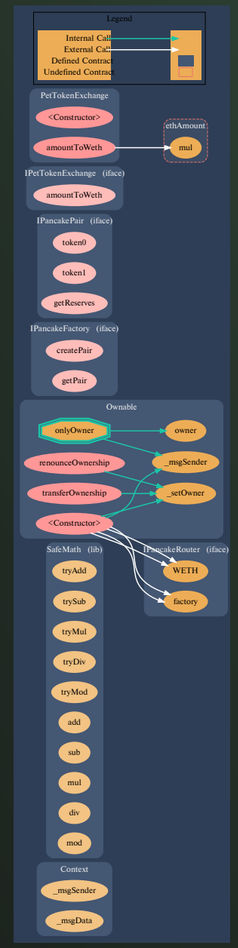


Pic. 1.2.
PetTokenProtect.sol

STRUCTURE OF CONTRACT
PETTOKENEXCHANGE.SOL

Contract methods analysis

function amountToWeth(uint256 ethAmount) public view returns(uint256 tokenAmount)
Function should be declared external. Function can be reduced to getAmountsOut. Function should be renamed to amountForEth, otherwise it’s logic has to be changed.



Pic. 1.3.
PetTokenExchange.sol

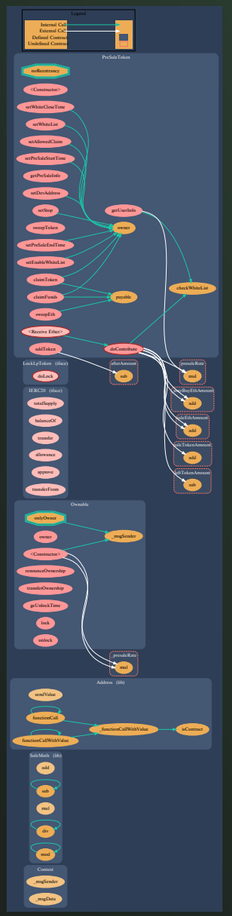
STRUCTURE OF CONTRACT
PRESALETOKEN.SOL

Usage of SafeMath library is not necessary and can be removed due to solidity 0.8.0.

Contract methods analysis

function setPreSaleStartTime(uint256 _preSaleStartTime)
Function can be declared external. Function should emit an event

function setDevAddress(address _devAddress)
Function can be declared external. Function should emit an event



Pic. 1.4.
PreSaleToken.sol

function setStop(bool _is_stop)
Function can be declared external. Function should emit an event

function setPreSaleEndTime(uint256 _preSaleEndTime)
Function can be declared external. Function should emit an event

function getPreSaleInfo()
Function can be declared external. Function should emit an event

function getUserInfo(address _user_address)
public view returns(bool _is_white, uint256 _mine_tokens,uint256 _mine_pay_eth,uint256 _mine_left,bool _allowedClaim,uint _isHardCap,bool _userHasClaim)
Function can be declared external. Function should emit an event

PAYABLE

function doContribute()
We recommend to remove check for tx.origin==msg.sender, because if user is using multisig wallet, his tx will revert. Hardcap will be reached only in case if saleEthAmount == hardCap. This can cause some problems, because in case if left eth amount for sale is less than minimal contrubtion, hard cap will never be reached and users will have to wait until pre sale ends in order to claim there tokens. To fix this we recommend to extend a check at line 597 and make it: saleEthAmount + minContribution> hardCap. This will make possible to avoid problems with unreachable hard cap by skipping unpurchasable amount. Function should emit an event

eth in, can be called by anyone

PAYABLE

function claimFunds()
We recoomend to remove check for tx.origin==msg.sender, because if user is using multisig wallet, his tx will revert. Function can be declared external.

eth out in case presale failed, can be called by anyone

PAYABLE

function addToken()
Function can be declared external. Function should emit an event

tokens in, onlyOwner

PAYABLE

function sweepToken()
Function can be declared external. Function should emit an event

tokens out, onlyOwner

PAYABLE

function sweepEth()
Function can be declared external. Function should emit an event

eth out, onlyOwner

PAYABLE

function claimToken()
Function can be declared external. Function should emit an event

tokens out, can be called by anyone in case presale succeeded

function setEnableWhiteList(uint _enable)
Recommended to use onlyOwner modifier, instead on require. Function can be declared external. Function should emit an event

function setEnableWhiteList(uint _enable)
Recommended to use onlyOwner modifier, instead on require. Function can be declared external. Function should emit an event

function setWhiteCloseTime(uint256 _closetime)
Function can be declared external. Function should emit an event

function setWhiteList(address[] memory _addressList,bool _bool)
Function can be declared external. Function should emit an event

function checkWhiteList(address _address) public view returns (bool)
Vulnerabilities not detected

function setAllowedClaim(bool _bool)
Function can be declared external. Function should emit an event

VERIFICATION CHECK SUMS

Contract Name	Solc version	Optimization	Bytecode hash (SHA 256)
PetToken	**0.8.0**	**200**	-3322f95de6071f54611d0d-797b84ae28f40757fde685f-5c07d05cd20fa037eb0
PetTokenProtect	**0.8.0**	**200**	-6a2081893b-8f21ade8084b7c6f97c8f-c29165206211520ed31cf-793f960a8a76
PetTokenExchange	**0.8.0**	**200**	73d5cc3954e4f5e-b52013e3cb761f7f632d-d4c6c2d4302b-475c3e8c407e8d131
PreSaleToken	**0.8.4**	**200**	-6e1c80230008e609ca671b-01cad87fa6e88a9664d-3ca3c757cf001c0e43f3a5f

GET IN TOUCH

info@smartstate.tech
smartstate.tech