



BlockSAFU

ADVANCE MANUAL SMART CONTRACT AUDIT



ApeGuild

Project: APEGUILD

Website: apeguild.games



BlockSAFU Score:

82

Contract Address:

0x86F3b68acA3B217e937f435EdA2e7044efFF20de

Disclaimer: BlockSAFU is not responsible for any financial losses.
Nothing in this contract audit is financial advice, please do your own reasearch.

DISCLAIMER

BlockSAFU has completed this report to provide a summary of the Smart Contract functions, and any security, dependency, or cybersecurity vulnerabilities. This is often a constrained report on our discoveries based on our investigation and understanding of the current programming versions as of this report's date. To understand the full scope of our analysis, it is vital for you to at the date of this report. To understand the full scope of our analysis, you need to review the complete report. Although we have done our best in conducting our investigation and creating this report, it is vital to note that you should not depend on this report and cannot make any claim against BlockSAFU or its Subsidiaries and Team members on the premise of what has or has not been included in the report. Please remember to conduct your independent examinations before making any investment choices. We do not provide investment advice or in any way claim to determine if the project will be successful or not.

By perusing this report or any portion of it, you concur to the terms of this disclaimer. In the unlikely situation where you do not concur to the terms, you should immediately terminate reading this report, and erase and discard any duplicates of this report downloaded and/or printed by you. This report is given for data purposes as it were and on a non-reliance premise and does not constitute speculation counsel. No one should have any right to depend on the report or its substance, and BlockSAFU and its members (including holding companies, shareholders, backups, representatives, chiefs, officers, and other agents) BlockSAFU and its subsidiaries owe no obligation of care towards you or any other person, nor does BlockSAFU make any guarantee or representation to any individual on the precision or completeness of the report.

ABOUT THE AUDITOR:

BlockSAFU (BSAFU) is an Anti-Scam Token Utility that reviews Smart Contracts and Token information to Identify Rug Pull and Honey Pot scamming activity. BlockSAFU's Development Team consists of several Smart Contract creators, Auditors Developers, and Blockchain experts. BlockSAFU provides solutions, prevents, and hunts down scammers. BSAFU is a utility token with features Audit, KYC, Token Generators, and Bounty Scammers. It will enrich the crypto ecosystem.

OVERVIEW

BlockSAFU was commissioned by APEGUILD to complete a Smart Contract audit. The objective of the Audit is to achieve the following:

- Review the Project and experience and Development team
- Ensure that the Smart Contract functions are necessary and operate as intended.
- Identify any vulnerabilities in the Smart Contract code.

DISCLAIMER: This Audit is intended to inform about token Contract Risks, the result does not imply an endorsement or provide financial advice in any way, all investments are made at your own risk. (<https://blocksafu.com/>)




SMART CONTRACT REVIEW


Token Name	APEGUILD
Token Symbol	APEG
Token Decimal	18
Total Supply	100,000,000 APEG
Contract Address	0x86F3b68acA3B217e937f435EdA2e7044eFF20de
Deployer Address	0x4C7379D24B4b8444ca4cf052e555e34a2df83F0a
Owner Address	0x4C7379D24B4b8444ca4cf052e555e34a2df83F0a
Tax Fees Buy	0%
Tax Fees Sell	0%
Gas Used for Buy	<i>will be updated after the DEX listing</i>
Gas Used for Sell	<i>will be updated after the DEX listing</i>
Contract Created	Jul-20-2022 05:37:16 PM +UTC
Initial Liquidity	<i>will be updated after the DEX listing</i>
Liquidity Status	Locked
Unlocked Date	<i>will be updated after the DEX listing</i>
Verified CA	Yes
Compiler	v0.8.4+commit.c7e474f2
Optimization	Yes with 200 runs
Sol License	MIT License
Top 5 Holders	<i>will be updated after the DEX listing</i>
Other	default evmVersion

TAX

BUY	0%	SELL	0%
-----	----	------	----

TOP HOLDER

Rank	Address	Quantity	Percentage	Analytics
1	0x4c7379d24b4b8444ca4cf052e555e34a2cf83f0a	85,000,000	85.0000%	
2	 PinkSale: PinkLock V2	15,000,000	15.0000%	

[Download CSV Export 

Team Review

The ApeGuild team has a nice website, their website is professionally built and the Smart contract is well developed, their social media is growing with over 5 people in their telegram group (count in audit date).

OFFICIAL WEBSITE AND SOCIAL MEDIA

Website: <https://apeguild.games/>

Telegram Group: <https://t.me/apeguild>

Twitter: <https://twitter.com/apeguildbsc>

MANUAL CODE REVIEW

● Minor-risk

1 minor-risk code issue found

Could be fixed, and will not bring problems.

1. The return value of an external transfer/transferFrom return value is checked.
Recommendation: use SafeERC20, or ensure that the transfer/transferFrom return value is checked

```
function transferFrom(  
    address sender,  
    address recipient,  
    uint256 amount  
) external returns (bool);
```

● Medium-risk

0 medium-risk code issues found

Should be fixed, could bring problems.

● High-Risk

0 high-risk code issues found

Must be fixed, and will bring problem.

● Critical-Risk

0 critical-risk code issues found

Must be fixed, and will bring problem.

EXTRA NOTES SMART CONTRACT

IERC20

```
interface IERC20 {
    /**
     * @dev Returns the number of tokens in existence.
     */
    function totalSupply() external view returns (uint256);
    ...
    function balanceOf(address account) external view returns (uint256);
    ...
    function transfer(address recipient, uint256 amount) external returns (bool);
    ...
    function allowance(address owner, address spender) external view returns (uint256);
    ...
    function approve(address spender, uint256 amount) external returns (bool);
    ...
    function transferFrom(
        address sender,
        address recipient,
        uint256 amount
    ) external returns (bool);

    /**
     * @dev Emitted when `value` tokens are moved from one account (`from`) to
     * another (`to`).
     *
     * Note that `value` may be zero.
     */
    event Transfer(address indexed from, address indexed to, uint256 value);
    ...
}
```

IERC20 Normal Base Template

2. SafeMath Contract

```
library SafeMath {
    ...
    function add(uint256 a, uint256 b) internal pure returns
(uint256) {
        uint256 c = a + b;
        require(c >= a, "SafeMath: addition overflow");
        return c;
    }
    ...
    function sub(uint256 a, uint256 b, string memory errorMessage)
internal pure returns (uint256) {
        require(b <= a, errorMessage);
        uint256 c = a - b;

        return c;
    }
    /**
     * @dev Returns the multiplication of two unsigned integers,
reverting on
     * overflow.
     *
     * Counterpart to Solidity's `*` operator.
     *
     * Requirements:
     *
     * - Multiplication cannot overflow.
     */
    ...
    function mod(
        uint256 a,
        uint256 b,
        string memory errorMessage
    ) internal pure returns (uint256) {
        unchecked {
            require(b > 0, errorMessage);
            return a % b;
        }
    }
}
```

Standard Safemath contract

3. ApeGuild Contract

```
contract StandardToken is IERC20, Ownable, BaseToken {
    ...
    string private _name;
    string private _symbol;
    uint8 private _decimals;
    uint256 private _totalSupply;

    constructor(
        string memory name_,
        string memory symbol_,
        uint8 decimals_,
        uint256 totalSupply_,
        address serviceFeeReceiver_,
        uint256 serviceFee_
    ) payable {
        _name = name_;
        _symbol = symbol_;
        _decimals = decimals_;
        _mint(owner(), totalSupply_);

        emit TokenCreated(owner(), address(this),
Token Type.standard, VERSION);

        payable(serviceFeeReceiver_).transfer(serviceFee_);
    }

    /**
     * @dev Returns the name of the token.
     */
    function name() public view virtual returns (string memory) {
        return _name;
    }

    /**
     * @dev Returns the symbol of the token, usually a shorter
version of the
     * name.
     */
    function symbol() public view virtual returns (string memory) {
        return _symbol;
    }
}
```

```

/**
 * @dev Returns the number of decimals used to get its user
representation.
 * For example, if `decimals` equals `2`, a balance of `505`
tokens should
 * be displayed to a user as `5,05` ( $505 / 10^{** 2}$ ).
 *
 * Tokens usually opt for a value of 18, imitating the
relationship between
 * Ether and Wei. This is the value {ERC20} uses, unless
{_setupDecimals} is
 * called.
 *
 * NOTE: This information is only used for _display_ purposes:
it in
 * no way affects any of the arithmetic of the contract,
including
 * {IERC20-balanceOf} and {IERC20-transfer}.
 */
function decimals() public view virtual returns (uint8) {
    return _decimals;
}

/**
 * @dev See {IERC20-totalSupply}.
 */
function totalSupply() public view virtual override returns
(uint256) {
    return _totalSupply;
}

/**
 * @dev See {IERC20-balanceOf}.
 */
function balanceOf(address account)
    public
    view
    virtual
    override
    returns (uint256)
{

```

```

        return _balances[account];
    }

    /**
     * @dev See {IERC20-transfer}.
     *
     * Requirements:
     *
     * - `recipient` cannot be the zero address.
     * - the caller must have a balance of at least `amount`.
     */
    function transfer(address recipient, uint256 amount)
        public
        virtual
        override
        returns (bool)
    {
        _transfer(_msgSender(), recipient, amount);
        return true;
    }

    /**
     * @dev See {IERC20-allowance}.
     */
    function allowance(address owner, address spender)
        public
        view
        virtual
        override
        returns (uint256)
    {
        return _allowances[owner][spender];
    }

    /**
     * @dev See {IERC20-approve}.
     *
     * Requirements:
     *
     * - `spender` cannot be the zero address.
     */
    function approve(address spender, uint256 amount)

```

```

    public
    virtual
    override
    returns (bool)
{
    _approve(_msgSender(), spender, amount);
    return true;
}

/**
 * @dev See {IERC20-transferFrom}.
 *
 * Emits an {Approval} event indicating the updated allowance.
This is not
 * required by the EIP. See the note at the beginning of
{ERC20}.
 *
 * Requirements:
 *
 * - `sender` and `recipient` cannot be the zero address.
 * - `sender` must have a balance of at least `amount`.
 * - the caller must have allowance for ``sender``'s tokens of
at least
 * `amount`.
 */
function transferFrom(
    address sender,
    address recipient,
    uint256 amount
) public virtual override returns (bool) {
    _transfer(sender, recipient, amount);
    _approve(
        sender,
        _msgSender(),
        _allowances[sender][_msgSender()].sub(
            amount,
            "ERC20: transfer amount exceeds allowance"
        )
    );
    return true;
}

```

```

/**
 * @dev Atomically increases the allowance granted to `spender`
by the caller.
 *
 * This is an alternative to {approve} that can be used as a
mitigation for
 * problems described in {IERC20-approve}.
 *
 * Emits an {Approval} event indicating the updated allowance.
 *
 * Requirements:
 *
 * - `spender` cannot be the zero address.
 */
function increaseAllowance(address spender, uint256 addedValue)
    public
    virtual
    returns (bool)
{
    _approve(
        _msgSender(),
        spender,
        _allowances[_msgSender()][spender].add(addedValue)
    );
    return true;
}

/**
 * @dev Atomically decreases the allowance granted to `spender`
by the caller.
 *
 * This is an alternative to {approve} that can be used as a
mitigation for
 * problems described in {IERC20-approve}.
 *
 * Emits an {Approval} event indicating the updated allowance.
 *
 * Requirements:
 *
 * - `spender` cannot be the zero address.
 * - `spender` must have allowance for the caller of at least
 * `subtractedValue`.

```

```

    */
    function decreaseAllowance(address spender, uint256
subtractedValue)
        public
        virtual
        returns (bool)
    {
        _approve(
            _msgSender(),
            spender,
            _allowances[_msgSender()][spender].sub(
                subtractedValue,
                "ERC20: decreased allowance below zero"
            )
        );
        return true;
    }

    /**
     * @dev Moves tokens `amount` from `sender` to `recipient`.
     *
     * This is internal function is equivalent to {transfer}, and
can be used to
     * e.g. implement automatic token fees, slashing mechanisms,
etc.
     *
     * Emits a {Transfer} event.
     *
     * Requirements:
     *
     * - `sender` cannot be the zero address.
     * - `recipient` cannot be the zero address.
     * - `sender` must have a balance of at least `amount`.
     */
    function _transfer(
        address sender,
        address recipient,
        uint256 amount
    ) internal virtual {
        require(sender != address(0), "ERC20: transfer from the
zero address");
        require(recipient != address(0), "ERC20: transfer to the

```

```

zero address");

    _beforeTokenTransfer(sender, recipient, amount);

    _balances[sender] = _balances[sender].sub(
        amount,
        "ERC20: transfer amount exceeds balance"
    );
    _balances[recipient] = _balances[recipient].add(amount);
    emit Transfer(sender, recipient, amount);
}

/** @dev Creates `amount` tokens and assigns them to `account`,
increasing
    * the total supply.
    *
    * Emits a {Transfer} event with `from` set to the zero
address.
    *
    * Requirements:
    *
    * - `to` cannot be the zero address.
    */
function _mint(address account, uint256 amount) internal
virtual {
    require(account != address(0), "ERC20: mint to the zero
address");

    _beforeTokenTransfer(address(0), account, amount);

    _totalSupply = _totalSupply.add(amount);
    _balances[account] = _balances[account].add(amount);
    emit Transfer(address(0), account, amount);
}

/**
    * @dev Destroys `amount` tokens from `account`, reducing the
    * total supply.
    *
    * Emits a {Transfer} event with `to` set to the zero address.
    *
    * Requirements:

```



```

*
* - `account` cannot be the zero address.
* - `account` must have at least `amount` tokens.
*/
function _burn(address account, uint256 amount) internal
virtual {
    require(account != address(0), "ERC20: burn from the zero
address");

    _beforeTokenTransfer(account, address(0), amount);

    _balances[account] = _balances[account].sub(
        amount,
        "ERC20: burn amount exceeds balance"
    );
    _totalSupply = _totalSupply.sub(amount);
    emit Transfer(account, address(0), amount);
}

/**
 * @dev Sets `amount` as the allowance of `spender` over the
`owner` s tokens.
 *
 * This internal function is equivalent to `approve`, and can
be used to
 * e.g. set automatic allowances for certain subsystems, etc.
 *
 * Emits an {Approval} event.
 *
 * Requirements:
 *
 * - `owner` cannot be the zero address.
 * - `spender` cannot be the zero address.
 */
function _approve(
    address owner,
    address spender,
    uint256 amount
) internal virtual {
    require(owner != address(0), "ERC20: approve from the zero
address");
    require(spender != address(0), "ERC20: approve to the zero

```

```

address");

    _allowances[owner][spender] = amount;
    emit Approval(owner, spender, amount);
}

/**
 * @dev Sets {decimals} to a value other than the default one
of 18.
 *
 * WARNING: This function should only be called from the
constructor. Most
 * applications that interact with token contracts will not
expect
 * {decimals} to ever change, and may work incorrectly if it
does.
 */
function _setupDecimals(uint8 decimals_) internal virtual {
    _decimals = decimals_;
}

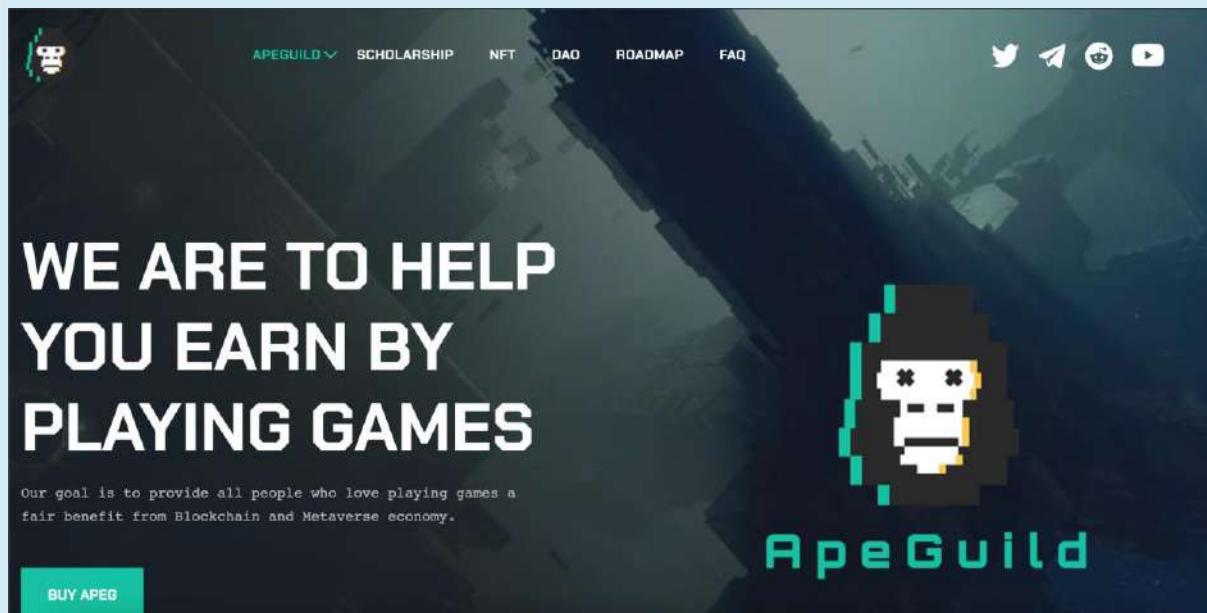
/**
 * @dev Hook that is called before any transfer of tokens. This
includes
 * minting and burning.
 *
 * Calling conditions:
 *
 * - when `from` and `to` are both non-zero, `amount` of
`from`'s tokens
 * will be transferred to `to`.
 * - when `from` is zero, `amount` tokens will be minted for
`to`.
 * - when `to` is zero, `amount` of `from`'s tokens will be
burned.
 * - `from` and `to` are never both zero.
 *
 * To Learn more about hooks, head to xref:ROOT:extending-
contracts.adoc#using-hooks[Using Hooks].
 */
function _beforeTokenTransfer(
    address from,

```

```
        address to,  
        uint256 amount  
    ) internal virtual {}  
}
```

ApeGuild Contract

WEBSITE REVIEW



- **Mobile Friendly**
- **Contains no code error**
- **SSL Secured (By Sectigo SSL)**

Web-Tech stack: Apache, Bootstrap, Sectigo, jQuery

Domain. finance (namecheap) - Tracked by whois

First Contentful Paint:	515ms
Fully Loaded Time	794ms
Performance	79%
Accessibility	87%
Best Practices	92%
SEO	80%

RUG-PULL REVIEW

Based on the available information analyzed by us, we come to the following conclusions:

- Locked Liquidity (Locked by pinksale)

(Will be updated after DEX listing)

- TOP 5 Holder.

(Will be updated after DEX listing)

- **The team is No KYC.**

(Will be updated after KYC)

HONEYPOT REVIEW

- Ability to sell.
- The owner is not able to pause the contract.
- The owner cannot change the fee

Note: Please check the disclaimer above and note, that the audit makes no statements or warranties on the business model, investment attractiveness, or code sustainability. The report is provided for the only contract mentioned in the report and does not include any other potential contracts deployed by the project owner.