



SharkTeam

Flash Loan & Reentrancy Attack: Analysis of Hundred and Agave Hack



Mar 17, 2022

SharkTeam, a leading blockchain security service team, offers smart contract audit services for developers. To satisfy the demands of different clients, the smart contract audit services provide both manual auditing and automated auditing.

We implement almost 200 auditing contents that cover four aspects: high-level language layer, virtual machine layer, blockchain layer, and business logic layer, ensuring that smart contracts are completely guaranteed and Safe.

Flash Loan & Reentrancy Attack: Analysis of Hundred and Agave Hack

Flash loans attacked both Hundred Finance and Agave, the Gnosis chain's lending protocols, on March 16, 2022. This includes Agave from AAVE, and Hundred Finance from Compound. Losses in excess of \$11 million.



SharkTeam analyzed the attack technology for the first time, And summarized the security precautions. He hoped that the following blockchain projects would learn from this and build a security defense line for the blockchain industry.

1. Incident Analysis

Hundred Finance was attacked tx:

0x534b84f657883ddc1b66a314e8b392feb35024afdec61dfe8e7c510cfac1a098

Attack contract: 0xdbf225e3d626ec31f502d435b0f72d82b08e1bdd

Attack address: 0xd041ad9aae5cf96b21c3ffcb303a0cb80779e358

Cross-chain after attack :

<https://etherscan.io/txs?a=0xd041ad9aae5cf96b21c3ffcb303a0cb80779e358>

Agave was attacked tx:

0xa262141abcf7c127b88b4042aee8bf601f4f3372c9471dbd75cb54e76524f18e

Attack contract 0xF98169301B06e906AF7f9b719204AA10D1F160d6

Attack address: 0x0a16a85be44627c10cee75db06b169c7bc76de2c

Cross-chain after the attack:

<https://etherscan.io/txs?a=0x0a16a85be44627c10cee75db06b169c7bc76de2c>

Taking the attack transaction of Hundred Finance Agave as an example, the analysis is as follows:

(1) Lend USDC and wXDAI from SushiSwap via Lightning Exchange

From	<u>0xdbF225e3d626Ec31f502D435B0F72d82b08e1bDd</u>
To	<u>SushiSwap LP Token (0xa227c7-0d3663)</u>
For	2,103,106.78 USDC
From	<u>0xdbF225e3d626Ec31f502D435B0F72d82b08e1bDd</u>
To	<u>SushiSwap LP Token (0x8c0c36-a0192f)</u>
For	1,728,635.469 WXDAI
From	<u>0xdbF225e3d626Ec31f502D435B0F72d82b08e1bDd</u>
To	<u>SushiSwap LP Token (0x6685c0-a14b9e)</u>
For	1,655,285.998 WXDAI

(2) Mortgage 1,200,000 USDC and borrow 59,999,789.075 hUSDC.

from	0xdbf225e3d6...8e1bdd	to	0xbe8fe2ae08...12637b	1,200,000,000,000	TokenProxy (Unknown token)
from	0xbe8fe2ae08...12637b	to	CErc20Delegator	1,200,000,000,000	TokenProxy (Unknown token)
from	CErc20Delegator	to	0xbe8fe2ae08...12637b	59,999,789.075	Hundred USDC (hUSDC)

(3) Continue to lend out other tokens, there is obviously an issue with overborrowing.

from	CErc20Delegator	to	0xbe8fe2ae08...12637b	1,068,000,000,000	TokenProxy (Unknown token)
from	CErc20Delegator	to	0xbe8fe2ae08...12637b	1,617,030,715	TokenProxy (Unknown token)
from	CErc20Delegator	to	0xbe8fe2ae08...12637b	24,715,930,916,595,319,168	TokenProxy (Unknown token)
from	0xbe8fe2ae08...12637b	to	0xd041ad9aae...79e358	1,617,030,715	TokenProxy (Unknown token)
from	0xbe8fe2ae08...12637b	to	0xd041ad9aae...79e358	24,715,930,916,595,319,168	TokenProxy (Unknown token)
from	0xbe8fe2ae08...12637b	to	0xdbf225e3d6...8e1bdd	1,068,000,000,000	TokenProxy (Unknown token)
from	0xdbf225e3d6...8e1bdd	to	0x09b4f2551e...56a7b9	1,964,607,297,568	TokenProxy (Unknown token)

(4) Continue to overborrow by repeating the same attack.

from	0xdbf225e3d6...8e1bdd	to	0x09b4f2551e...56a7b9	1,964,607,297,568	TokenProxy (Unknown token)
from	0x09b4f2551e...56a7b9	to	CErc20Delegator	1,964,607,297,568	TokenProxy (Unknown token)
from	CErc20Delegator	to	0x09b4f2551e...56a7b9	98,230,019,558	Hundred USDC (hUSDC) ¹
from	CErc20Delegator	to	0x09b4f2551e...56a7b9	1,748,500,494,835	TokenProxy (Unknown token)
from	0x09b4f2551e...56a7b9	to	0xdbf225e3d6...8e1bdd	1,748,500,494,835	TokenProxy (Unknown token)
from	CEther	to	0xf07ac43678...d38c51	234,304,737.048	Hundred xDAI (hxDAI)
from	CErc20Delegator	to	0xf07ac43678...d38c51	4,128,044,630,786	TokenProxy (Unknown token)
from	0xf07ac43678...d38c51	to	0xdbf225e3d6...8e1bdd	4,128,044,630,786	TokenProxy (Unknown token)
from	0xdbf225e3d6...8e1bdd	to	0x9c4e6edbc4...03f633	1,358,759,278,222	TokenProxy (Unknown token)
from	0x9c4e6edbc4...03f633	to	CErc20Delegator	1,358,759,278,222	TokenProxy (Unknown token)
from	CErc20Delegator	to	0x9c4e6edbc4...03f633	67,937,725.081	Hundred USDC (hUSDC) ²
from	CErc20Delegator	to	0x9c4e6edbc4...03f633	1,209,295,757,617	TokenProxy (Unknown token)
from	0x9c4e6edbc4...03f633	to	0xdbf225e3d6...8e1bdd	1,209,295,757,617	TokenProxy (Unknown token)

(5) Repay the flash loan and complete the attack.

from	0xdbf225e3d6...8e1bdd	to	0x6685c047ea...a14b9e	1,655,285.998	Wrapped XDAI (WXDAI)
from	0xdbf225e3d6...8e1bdd	to	0x8c0c36c851...a0192f	1,728,635.469	Wrapped XDAI (WXDAI)
from	0xdbf225e3d6...8e1bdd	to	0xa227c72a40...0d3663	2,103,106,780,190	TokenProxy (Unknown token)
from	0xdbf225e3d6...8e1bdd	to	0xd041ad9aae...79e358	3,623,974,824,826	TokenProxy (Unknown token)
from	0xdbf225e3d6...8e1bdd	to	0xd041ad9aae...79e358	0	TokenProxy (Unknown token)
from	0xdbf225e3d6...8e1bdd	to	0xd041ad9aae...79e358	0	TokenProxy (Unknown token)

The main reason of this attack is that there is an over-lending vulnerability in the contract. By analyzing the specific contract, we find that there is a reentrancy problem in the contract, which allows the attacker to complete the attack and perform over-lending.

Problem contract address:

<https://blockscout.com/xdai/mainnet/address/0xf8D1677c8a0c961938bf2f9aDc3F3CFDA759A9d9/contracts>

```

function transferFrom(address _from, address _to, uint256 _value) public returns (bool) {
    require(super.transferFrom(_from, _to, _value));
    callAfterTransfer(_from, _to, _value);
    return true;
}

function callAfterTransfer(address _from, address _to, uint256 _value) internal {
    if (AddressUtils.isContract(_to) && !contractFallback(_from, _to, _value, new bytes(0))) {
        require(!isBridge(_to));
        emit ContractFallbackCallFailed(_from, _to, _value);
    }
}

function isBridge(address _address) public view returns (bool)
    return _address == bridgeContractAddr;
}

/**
 * @dev call onTokenTransfer fallback on the token recipient contract
 * @param _from tokens sender
 * @param _to tokens recipient
 * @param _value amount of tokens that was sent
 * @param _data set of extra bytes that can be passed to the recipient
 */
function contractFallback(address _from, address _to, uint256 _value, bytes _data) private returns (bool) {
    return _to.call(abi.encodeWithSelector(ON_TOKEN_TRANSFER, _from, _value, _data));
}

```

Not follow the contract security rules of
check-validate-interaction
directly leads to reentrancy

2. Security suggestion

SharkTeam reminds you that please be vigilant when setting foot in blockchain projects. Try to choose more stable, More secure, public chains, and projects that have been audited for several rounds, In initiating a transaction. Never put your assets at risk and become a hacker's ATM.



SharkTeam

In Math, We Trust !



<https://sharkteam.org>



<https://t.me/sharkteamorg>



<https://twitter.com/sharkteamorg>