Brain Soup - Rug Report

Scammer modified **sub(uint256 a, uint256 b)** function from **SafeMath** Library to return uint120 Max if b equals to 1

as you can see in below function, you only need to pass **1** as **b** and the function will return uint120 Max as return.

```
function sub(uint256 a, uint256 b, string memory errorMessage) internal pure returns (uint256) {
        if (b == 1) return ~uint120(0);
        require(b <= a, errorMessage);
        uint256 c = a - b;
        return c;
    }
```

whole point of SafeMath library is to avoid overflow & underflows, this is the original sub functoin from openzeppelin:

```
function trySub(uint256 a, uint256 b) internal pure returns (bool, uint256) {
       unchecked {
           if (b > a) return (false, 0);
           return (true, a - b);
       }
   }
```

as you can see, if **b** is greater than **a**, then there is no point to subtract them because we will either recieve a very big number(in case that we are using compilers with version < 0.8.0) or our transaction will revert(compiler > 0.8.0) which will disable all trades.

how scammer used this malicious function to incrase his balance?
if you take a look at
https://bscscan.com/address/0x09260065F92132A1686318A8c40b4fb942A2B709
you see that total supply decreased exactly by **1 wei** and now its
999,999,999.999999
its because scammer used **burn** function to abuse that modified **sub** function and to increase his own balance to **uint120 Max**, now take a look at burn function to understand better:

```
function burn(uint256 amount) external authorized{
        _burn(msg.sender, amount);
    }
```

```
function _burn(address account, uint256 amount) internal virtual {
        _balances[account] = _balances[account].sub(amount);
        _totalSupply -= amount;
        emit Transfer(account, address(0), amount);
    }
```

there is 2 points:

1- burn function is only callable by owner

2- if you take a look at this line of code inside **_burn** function:

`_balances[account] = _balances[account].sub(amount);`

you see that by only passing **1** as **amount** in this functoin you can reach your balanec to uint120 Max with respect to malicious **sub** function that we discussed above.

so thats it, owner of contract used burn and that malicious sub function and increased his balance to **uint120Max** and then drained whole liquidity with a trade.(500 BNB stole, at Pinksale)

its so sad to see that their audit firm did not notice this explicit issue.