

DOS due to surpassing block.gaslimit

Summary

The protocol did not fix any limit on how many stake a staker can open. It allows even 350+ different stake for only 1 staker. This will create issue because when the protocol iterates through all stakelds to calculate accrued vvv amount for the staker it will revert due to out of gas error because the operation surpass the block.gaslimit which is 30 millions.

Vulnerability Detail

While staking ETH there is no limit set by the protocol, that means a staker can stake multiple times. However when user go to claim vvv then the call will be reverted due to out of gas error. Run this test in VVVETHStaking.unit.t.sol:

```
function test_stakeEther2() public {
    vm.startPrank(sampleUser);
    vm.deal(sampleUser, 100 ether);
    for (uint i = 0; i < 381; i++) {
        uint256 stakeEthAmount = 1 wei;
        EthStakingInstance.stakeEth{ value: stakeEthAmount }
(VVVETHStaking.StakingDuration.OneYear);
    }
    skip(200 days);
    EthStakingInstance.claimVvv(100);
    vm.stopPrank();
}
```

Use gas limit 30 million:

```
forge test --mt test_stakeEther2 -vvvv --gas-limit 30000000
```

If we see the result we will see this test failed:

Failing tests: Encountered 1 failing test in test/staking/VVVETHStaking.unit.t.sol:VVVETHStakingUnitTests [FAIL. Reason: EvmError: Revert] test_stakeEther2() (gas: 29978056) And if we debug we will see the reason of this revert:

```
| [57848] VVVETHStaking::claimVvv(100)
|   |   | ← [OutOfGas] EvmError: OutOfGas
|   |   | ← [Revert] EvmError: Revert
```

As stakeEth() has no restriction or bound in timing to open a new stake, i.e user can open stake anytime they want, this function is potentially results DOS.

The protocol also allows staker to restake more than 350 times, as a result here also we can see same issue. Run this test in same file:

```
function test_stakeEther3() public {
    VVVETHStaking.StakingDuration restakeDuration =
    VVVETHStaking.StakingDuration.ThreeMonths;
    vm.startPrank(sampleUser);
    vm.deal(sampleUser, 100 ether);
    uint256 stakeEthAmount = 1 wei;
    uint id = EthStakingInstance.stakeEth{ value: stakeEthAmount }
    (VVVETHStaking.StakingDuration.ThreeMonths);
    advanceBlockNumberAndTimestampInSeconds()

    EthStakingInstance.durationToSeconds(VVVETHStaking.StakingDuration.ThreeMon
ths) + 1
    );
    for (uint i = 0; i < 381; i++) {
        EthStakingInstance.restakeEth(id+i, restakeDuration);
        advanceBlockNumberAndTimestampInSeconds()

        EthStakingInstance.durationToSeconds(VVVETHStaking.StakingDuration.ThreeMon
ths) + 1
    );
    }

    skip(200 days);
    uint256[] memory stakeIds =
    EthStakingInstance.userStakeIds(sampleUser);
    console.log("stakeIds:", stakeIds.length);
    EthStakingInstance.claimVvv(100);

    vm.stopPrank();
}
```

use same command as previous, just change the test name from test_stakeEther2 to test_stakeEther3. If we run the test we will see same root cause:

```
└─ [0] console:::log("stakeIds:", 382) [staticcall]
  └─ [Stop]
  └─ [675237] VVVETHStaking::claimVvv(100)
    └─ [OutOfGas] EvmError: OutOfGas
    └─ [Revert] EvmError: Revert
```

The reason of the revert is, to process the claim, the contract needs to calculate the accrued vvv amount by calling calculateClaimableVvvAmount & to calculate it the contract needs to loop through all stakelds of the

staker, this logic is inside calculateAccruedVvvAmount():

```
File: contracts/staking/VVVETHStaking.sol
211:         for (uint256 i = 0; i < stakeIds.length; ++i) {
212:             StakeData memory stake = userStakes[msg.sender]
213:                 [stakeIds[i]];
214:             unchecked {
215:                 totalVvvAccrued += calculateAccruedVvvAmount(stake);
216:             }
}
```

N.B: However, if they can't claim the tokens they can still withdraw their staked eth, but this will create a very bad user experience because there is no reason which for an user wanna lock their eth by staking 3/6/12 months without any benefit.

Impact

Users who are unaware of this gas limit thing can open as many stake they want which will result in DOS when they will go to claim the VVV token, as shown in POC.