Offchain Labs Assessment

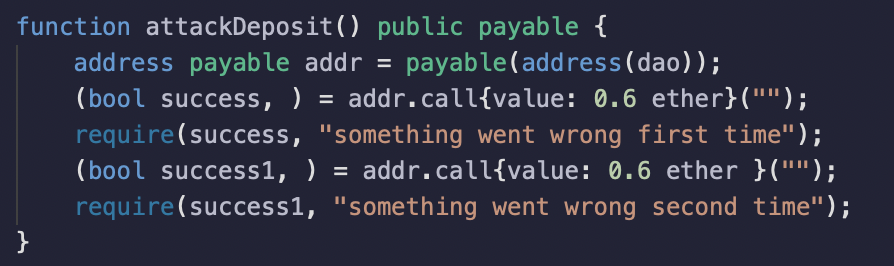
Explain how someone could deposit more than 1eth per block

I found 2 ways to do just that:

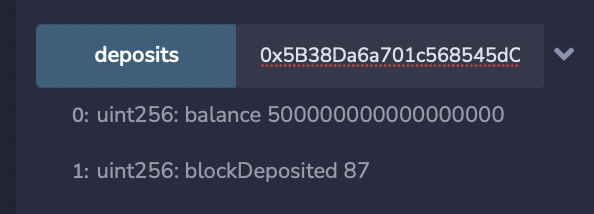
1. Use the selfdestruct function of an attack smart contract that will force send its balance to the DAOEscrowFarm contract.



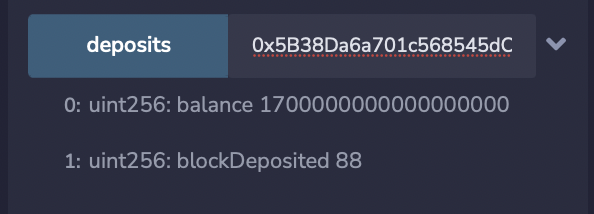
1. Have an address send some ether, let’s say 0.5 eth, to the DAOEscrowFarm contract, then have this address call a smart contract that will call twice within a block the receive() function of the DAOEscrowFarm with 0.6 eth as msg.value.



That way, for the first call maxDeposit will be equal to DEPOSIT\_LIMIT\_PER\_BLOCK since it’s the first call within that block, the “if(msg.value > maxDeposit)” conditions aren’t matched and that part of the code is skipped, then the prev.balance is updated to 1.1 (since 0.5 eth were deposited previously and 0.6 eth were deposited just now) and prev.blockDeposited is updated to the current block. The second call of the receive() function, since it’s the second deposit within the same block, the ternary operator will return true and maxDeposit will be equal to DEPOSIT\_LIMIT\_PER\_BLOCK - prev.balance with is 1 - 1.1 and will return an underflow, so the “if(msg.value > maxDeposit)” conditions aren’t matched and that part of the code is skipped and the balance is updated again.

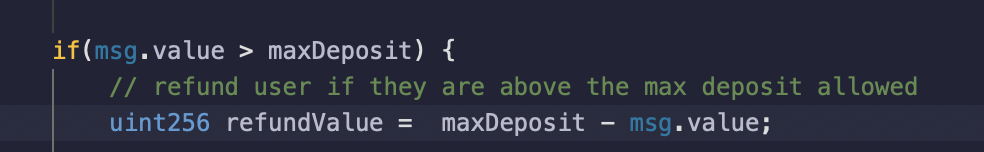


We have managed to transfer 1.2 eth within one block.



Find a reentrancy vulnerability and send us a sample contract that exploits it

The obvious reentrancy here is within the receive() function since the transfer for the refund happens before the balance is updated, which is reminiscent of the DAO hack, but there is something deeply wrong with the receive function.



This part is just wrong.

There is no value that would pass “msg.value > maxDeposit” and not underflow for “maxDeposit - msg.value”.

So refundValue will always be near the upper limit of a uint256 and the “(bool success,) = msg.sender.call{value: refundValue}("");” will never work since no contract has such a huge amount of ether to send.

The logical function would have “msg.value - maxDeposit”, which would work and actually refund a user of whatever was sent to the contract that is above 1eth per block.

I have to say that I’m really confused with those 2 lines of codes and I spent more time on it than I should have and would love an explanation as to what I didn’t understand about the assignment.

Thanks for reading if you made it this far.