# Message call with hardcoded gas amount

## **Description**

The transfer() and send() functions forward a fixed amount of 2300 gas. Historically, it has often been recommended to use these functions for value transfers to guard against reentrancy attacks. However, the gas cost of EVM instructions may change significantly during hard forks which may break already deployed contract systems that make fixed assumptions about gas costs. For example. <u>EIP 1884</u> broke several existing smart contracts due to a cost increase of the SLOAD instruction.

### Remediation

Avoid the use of transfer() and send() and do not otherwise specify a fixed amount of gas when performing calls. Use .call.value(...)("") instead. Use the checks-effects-interactions pattern and/or reentrancy locks to prevent reentrancy attacks.

## **Example:**

#### Code:

```
pragma solidity 0.6.4;

interface ICallable {
    function callMe() external;
}

contract HardcodedNotGood {
    address payable _callable = 0xaAaAaAaaAaAaAaAaAaAaAaAaaAaaAaaaaa;
    ICallable callable = ICallable(_callable);

constructor() public payable {
    function doTransfer(uint256 amount) public {
        _callable.transfer(amount);
    }

function doSend(uint256 amount) public {
        _callable.send(amount);
}

function callLowLevel() public {
        _callable.call.value(0).gas(10000)("");
}

function callwithArgs() public {
        callable.callMe{gas: 10000}();
}
```

#### Explanation:

Transaction Gas value is hardcoded at Line 24.