Delegatecall to Untrusted Callee

Description

There exists a special variant of a message call, named `delegatecall` which is identical to a message call apart from the fact that the code at the target address is executed in the context of the calling contract and `msg.sender` and `msg.value` do not change their values. This allows a smart contract to dynamically load code from a different address at runtime. Storage, current address and balance still refer to the calling contract.

Calling into untrusted contracts is very dangerous, as the code at the target address can change any storage values of the caller and has full control over the caller's balance.

Remediation

Use delegatecall with caution and make sure to never call into untrusted contracts. If the target address is derived from user input, ensure to check it against a whitelist of trusted contracts.

Example:

Code:

```
pragma solidity ^0.4.24;

contract Proxy {

  address owner;

  constructor() public {
    owner = msg.sender;
  }

  function forward(address callee, bytes _data) public {
    require(callee.delegatecall(_data));
  }

}
```

Explanation:

The delegate call function delegates the rights of the caller contract to the callee contract (the target contract that is believed to perform the same activity) so that the callee contract may access the EOA address and value to complete its operation.

The fallback function exploits the delegate function to change the contract's data.

In the above example, any untrusted callee user may delegate without authorization to the target contract, which raises security problems.