BUG REPORT : Payment Channel Signature Verification Does Not support most Web3 providers

Author : Philippe Castonguay

Date : 12.04.2018

Affects: GNTPaymentChannels.sol

Short Description : Clients like Geth add a prefix message when signing a hash and the current `isValidSig()` function does not account for this.

Description

Some Ethereum clients like **Geth** add a prefix message before allowing users to sign a message. According to issue #3731 (https://github.com/ethereum/go-ethereum/issues/3731):

> Geth prepends the string `\x19Ethereum Signed Message:\n<length of message>` to all data before signing it (https://github.com/ethereum/wiki/wiki/JSON-RPC#eth_sign). If you want to verify such a signature from Solidity, you'll have to prepend the same string in solidity before doing the ecrecovery.

The `isValidSig()` function in **GNTPaymentChannels.sol** would therefore always return false when the signature comes from these clients. **Anyone trying to sign messages via Geth, Metamask, Infura would see `isValidSig()` return false**. This means that none of the clients mentioned above can effectively interact with the **GNTPaymentChannels.sol** contract.

See https://ethereum.stackexchange.com/questions/20962/should-signed-text-messages-use-the-x19ethereum-signed-message-prefix?rq=1 for a survey of the problem.

Bug Impact

Impact: High

Most Ethereum clients will not be able to successfully use the payment channels on **GNTPaymentChannels.sol**.

Components:

- **GNTPaymentChannels.sol**

Reproduction:

Test file : https://github.com/PhABC/golem-contracts/blob/signature/test/Signature.test.js

Test contract file : https://github.com/PhABC/golem-contracts/blob/signature/contracts/GNTPaymentChannels.sol

Instructions:

```
npm install
npm test
```

Fix

Replace current `isValidSig()` function with the following ;

```
function isValidSig(
        bytes32 _ch,
        uint _value,
        uint8 _v,
        bytes32 _r,
        bytes32 _s)
        view returns (bool)
{
    //Hash to sign
    bytes32 hash = keccak256(_ch, _value);

    //No prefix when hash signed
    if ( channels[_ch].owner == ecrecover(hash, _v, _r, _s) ){
        return true;
    }

    //Prefix when hash signed
    bytes32 prefixedHash = keccak256("\x19Ethereum Signed Message:\n32", hash);

    if ( channels[_ch].owner == ecrecover(prefixedHash, _v, _r, _s) ){
        return true;
    }

    return false;
}
```

This should handle *both* prefixed and non-prefixed message signatures, allowing all clients to use payment channels easily. Note that I changed `sha3()` to `keccak256` since the former was deprecated.