

# Week7-8.15

## Day1 CIP23 EIP712

hex

EIP712

conflu

- message conf \x19Conflux Signed Message: \n \x19Ethereum Signed Message: \n
- EIP712domain CIP23domain
- CIP23domainchainId

## Day2 CIP23

- :  
encode(message :  $8^n$ ) = "\x19Conflux Signed Message:\n" || len(message) || message  
where len(message) is the non-zero-padded ascii-decimal encoding of the number of bytes in message.
- encode(domainSeparator :  $2^{56}$ , message : ) = "\x19\x01" || domainSeparator ||  
hashStruct(message) where domainSeparator and hashStruct(message) are defined below.

conflux-sdk

sdk

java-sdk

```
package conflux.web3j.crypto;

import org.junit.jupiter.api.Test;
import org.web3j.utils.Numeric;

import java.io.IOException;

import static org.junit.jupiter.api.Assertions.assertEquals;

public class SignDataTests {
```

```

//cfx_signTypedData_v4
@Test
public void testSignValidStructure() throws IOException {
    StructuredDataTests t = new StructuredDataTests();

    // TypedData
    String msg = t.getResource(
        "build/resources/test/"
        + "structured_data_json_files/ValidStructuredData.json");

    // msg
    StructuredDataEncoder dataEncoder = new StructuredDataEncoder(msg);
    // msghash
    org.web3j.crypto.Sign.SignatureData sign =
org.web3j.crypto.Sign.signMessage(dataEncoder.hashStructuredData(), SampleKeys.KEY_PAIR,
false);
    assertEquals(
        "0x371ef48d63082d3875fee13b392c5b6a7449aa638921cb9f3d419f5b6a817ba754d085965fb3a041c3b178d3ae3
798ea322ae74cb687dd699b5f6045c7fe47a91c",
        Numeric.toHexString(sign.getR()) + Numeric.toHexStringNoPrefix(sign.getS()) +
Numeric.toHexStringNoPrefix(sign.getV()));
    }

    //personal_sign
    @Test
    public void testSignAnyMessage() throws IOException {
        String message = "v0G9u7huK4mJb2K1";
        // msg msghash
        org.web3j.crypto.Sign.SignatureData sign =
Sign.signPrefixedMessage(message.getBytes(), SampleKeys.KEY_PAIR);
        assertEquals(
            "0xbb0ee8492623f2ef6ed461ea638f8b5060b191a1c8830c93d84245f3fb27e20a755e24ff60fe76482dd4377a0ae
f036937ef88537b2d0fdd834a54e76ecafadc1c",
            Numeric.toHexString(sign.getR()) + Numeric.toHexStringNoPrefix(sign.getS()) +
Numeric.toHexStringNoPrefix(sign.getV()));
        }
    }
}

```

# Day3 cip23

cip23

sdk

java-sdk

sdk

```
package conflux.web3j.crypto;

import org.junit.jupiter.api.Test;
import org.web3j.crypto.*;
import org.web3j.crypto.Sign;
import org.web3j.utils.Numeric;

import java.io.IOException;
import java.util.Arrays;

import static org.junit.jupiter.api.Assertions.assertEquals;

public class ECTest {

    private String getAddress() {
        return Numeric.prependHexPrefix(Keys.getAddress(getPubKey()));
    }

    private String getPubKey() {
        return SampleKeys.KEY_PAIR.getPublicKey().toString();
    }

    @Test
    public void testSignAndRecoverMessage() {
        String message = "v0G9u7huK4mJb2K1";

        byte[] msgHash = conflux.web3j.crypto.Sign.getConfluxMessageHash(message.getBytes());

        Sign.SignatureData sign =
conflux.web3j.crypto.Sign.signPrefixedMessage(message.getBytes(), SampleKeys.KEY_PAIR);
//        recover,
        String recoverAddress = conflux.web3j.crypto.Sign.recoverSignature(sign, msgHash,
```

```

getAddress());
    assertEquals(recoverAddress, getAddress());
}

// fluent                recover
@Test
public void testRecoverTyped() throws IOException {
    StructuredDataTests t = new StructuredDataTests();
    String msg = t.getResource(
        "build/resources/test/"
            + "structured_data_json_files/ValidStructuredData.json");
    StructuredDataEncoder dataEncoder = new StructuredDataEncoder(msg);

    //                fluent
    String signature =

"0x371ef48d63082d3875fee13b392c5b6a7449aa638921cb9f3d419f5b6a817ba754d085965fb3a041c3b178d3ae3
798ea322ae74cb687dd699b5f6045c7fe47a91c";

    //
    byte[] signatureBytes = Numeric.hexStringToByteArray(signature);
    byte v = signatureBytes[64];
    if (v < 27) {
        v += 27;
    }

    Sign.SignatureData sd =
        new Sign.SignatureData(
            v,
            (byte[]) Arrays.copyOfRange(signatureBytes, 0, 32),
            (byte[]) Arrays.copyOfRange(signatureBytes, 32, 64));
    // //getAddress()        fluent
    String recoverAddress = conflux.web3j.crypto.Sign.recoverSignature(sd,
dataEncoder.hashStructuredData(), getAddress());
    assertEquals(recoverAddress, getAddress());
}

//    testSignAndRecoverMessage()
@Test
public void testSignAndRecoverTyped() throws IOException {

```

```

StructuredDataTests t = new StructuredDataTests();
String msg = t.getResource(
    "build/resources/test/"
        + "structured_data_json_files/ValidStructuredData.json");
StructuredDataEncoder dataEncoder = new StructuredDataEncoder(msg);
Sign.SignatureData sign = Sign.signMessage(dataEncoder.hashStructuredData(),
SampleKeys.KEY_PAIR, false);

    String recoverAddress = conflux.web3j.crypto.Sign.recoverSignature(sign,
dataEncoder.hashStructuredData(), getAddress());
    assertEquals(recoverAddress, getAddress());
}
}

```

recover()

```

public static String recoverSignature(SignatureData sd, byte[] data, String address) {
    String addressRecovered = null;

    // Iterate for each possible key to recover
    for (int i = 0; i < 4; i++) {
        BigInteger publicKey =
            org.web3j.crypto.Sign.recoverFromSignature(
                (byte) i,
                new ECDSASignature(
                    new BigInteger(1, sd.getR()), new BigInteger(1,
sd.getS()),
                    data);

        if (publicKey != null) {
            addressRecovered =
Numeric.prependHexPrefix(Keys.getAddress(publicKey.toString()));

            if (addressRecovered.equals(address)) {
                break;
            }
        }
    }

    return addressRecovered;
}

```

```
}
```

[https://github.com/web3j/web3j/blob/7dea3d99c5bdbfcc03aaeaa8575fb0c9a9a771ab/crypto/src/main/java/org/web3j/crypto/Sign.java#:~:text=public%20static%20BigInteger%20recoverFromSignature\(int%20recId%2C%20ECDSASignature%20sig%2C%20byte%5B%5D%20message\)%20%7B](https://github.com/web3j/web3j/blob/7dea3d99c5bdbfcc03aaeaa8575fb0c9a9a771ab/crypto/src/main/java/org/web3j/crypto/Sign.java#:~:text=public%20static%20BigInteger%20recoverFromSignature(int%20recId%2C%20ECDSASignature%20sig%2C%20byte%5B%5D%20message)%20%7B)

<https://wiki.conflux123.xyz/link/60#bkmrk-%E5%90%88%E7%BA%A6%E9%AA%8C%E7%AD%BE>

# Day4 fluent

java-sdk test sdk-provider sdk-provider fluent cfx eth fluent

personal\_sign console

```
conflux
  .request({
    method: 'personal_sign',
    params: [
      'v0G9u7huK4mJb2K1', ' <your_address>' ]})
```

fluent



```

> conflux
  .request({
    method: 'personal_sign',
    params: [
      'v0G9u7huK4mJb2K1', 'cfxtest:aajb342mw5kzad6pjkdz0wxx0tr54nfupbu6yaj49' ])
< Promise {<pending>} 1
  ▶ [[Prototype]]: Promise
  [[PromiseState]]: "fulfilled"
  [[PromiseResult]]: "0x7e4216720b40f8d7f2cda70433d4a94f3926635517cd5691c778e38eae87236759ddf7c53ec55c5463f8e966ebd5a32c2dbe1061e95afcb64ef3a8187badcb00"

```

cfx\_signTypedData\_v4

console

```

conflux
  .request({
    method: 'cfx_signTypedData_v4',
    params: [
      '<your_address>',
      {
        "types": {
          "CIP23Domain": [
            {
              "name": "name",
              "type": "string"
            },
            {
              "name": "version",
              "type": "string"
            },
            {
              "name": "chainId",
              "type": "uint256"
            },
            {
              "name": "verifyingContract",
              "type": "address"
            }
          ],
          "Person": [
            {
              "name": "name",
              "type": "string"
            }
          ]
        }
      }
    ]
  })

```

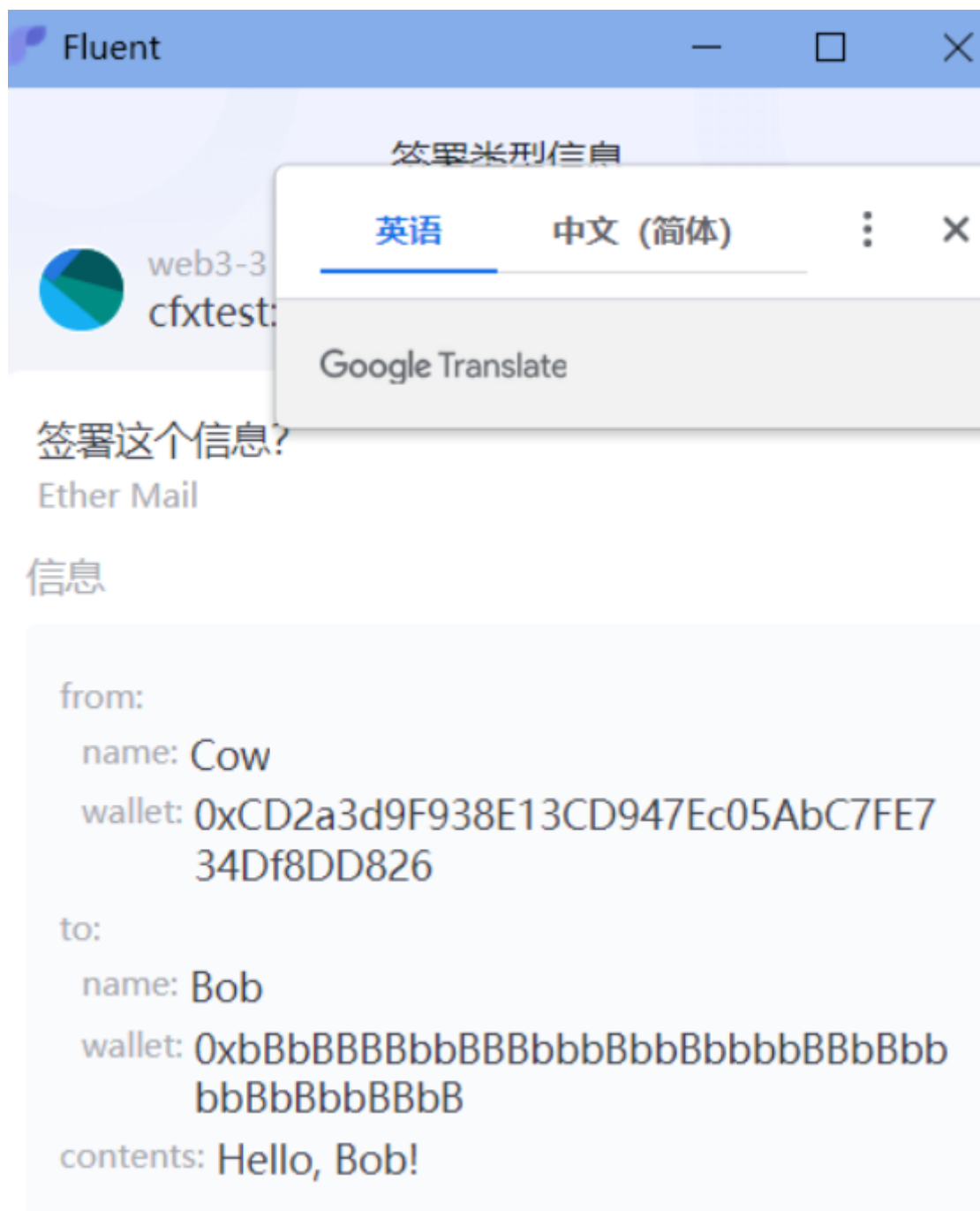
```

        "name": "wallet",
        "type": "address"
    }
],
"Mail": [
    {
        "name": "from",
        "type": "Person"
    },
    {
        "name": "to",
        "type": "Person"
    },
    {
        "name": "contents",
        "type": "string"
    }
]
},
"primaryType": "Mail",
"domain": {
    "name": "Ether Mail",
    "version": "1",
    "chainId": 1,
    "verifyingContract": "0xCcCCccccCCCCcCCCCcCcCccCcCCcCcccccccC"
},
"message": {
    "from": {
        "name": "Cow",
        "wallet": "0xCD2a3d9F938E13CD947Ec05AbC7FE734Df8DD826"
    },
    "to": {
        "name": "Bob",
        "wallet": "0xbBbBBBBbbBBBbbbBbbBbbbbBBbBbbbbBbBbbBBbB"
    },
    "contents": "Hello, Bob!"
}
}
}
})

```



message



# Day5 cip23 sdk

java-sdk js-sdk [cip23](#) [cip23js-sdk](#) [cip23](#)

go-sdk python-sdk cip23

Revision #12

Created 15 August 2022 01:28:34 by Xianqi

