ETM555
DESIGN of INFORMATION SYSTEMS

ERC721 Token Based Supply Chain Traceability System

Instructor: Can Ozturan

Fall 2020/2021

Sevgican VAROL
Gorkem ATES
Deniz MEMIS

GitHub

Entities

- On truffle develop console we have the following three entities:
  - **Owner** (contract deployer)
  - **Consumer1** (simulating a consumer/intermediary)
  - **Consumer2** (simulating another consumer/intermediary)

```
truffle(develop)> owner
'0x5D801E748D9931eb0eaE907ACf85157223a91E64'
truffle(develop)> consumer1
'0x872Cf60361a652e45313f0fBFaEbF252a946317B'
truffle(develop)> consumer2
'0x53767ceC42fdB5C498aA2912Cb2dc3D559096A5e'
truffle(develop)>
```

StateVerification Contract Interactions

- **StateVerification** contract is deployed and interacted as **'stateAuthority'** object
- Created entities are approved by 'stateAuthority' using **'insertVerifiedEntity**(address _addr)' function as follows:
  - This method has **isOwner**() modifier, allowing only the contract owner to add entities to the whitelist.

- **Owner** is whitelisted:

```
truffle(develop)> stateAuthority.insertVerifiedEntity(owner);
{
  tx: '0x73d371821206e8c2d8f3d893f379f250792177b5d6e57ccc44030624f2cd70a6',
  receipt: {
    transactionHash: '0x73d371821206e8c2d8f3d893f379f250792177b5d6e57ccc44030624f2cd70a6',
    transactionIndex: 0,
    blockHash: '0x6034a6f09f943a6ea0536281a777f0280363547a4791437445a33d085ecc4d59',
    blockNumber: 10,
    from: '0x5d801e748d9931eb0eae907acf85157223a91e64',
    to: '0x07ad67c207640d5663c82a9fdc0bca0495529fae',
    gasUsed: 24271,
    cumulativeGasUsed: 24271,
    contractAddress: null,
    logs: [],
    status: true,
    logsBloom: '0x00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000',
    rawLogs: []
  },
  logs: []
}
```

- **Consumer1** is whitelisted:

```
truffle(develop)> stateAuthority.insertVerifiedEntity(consumer1);
{
  tx: '0x86747b3cbe528e3383ef0b4df40dce0faf9f4fd1db153492a551541db7a3c8f1',
  receipt: {
    transactionHash: '0x86747b3cbe528e3383ef0b4df40dce0faf9f4fd1db153492a551541db7a3c8f1',
    transactionIndex: 0,
    blockHash: '0x2dd99a267ef7d9971e8255adb57a48cf373af83a38295992bf48b3b04e51bc6d',
    blockNumber: 11,
    from: '0x5d801e748d9931eb0eae907acf85157223a91e64',
    to: '0x07ad67c207640d5663c82a9fdc0bca0495529fae',
    gasUsed: 24271,
    cumulativeGasUsed: 24271,
    contractAddress: null,
    logs: [],
    status: true,
    logsBloom: '0x00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000',
    rawLogs: []
  },
  logs: []
}
```

- **Consumer2** is whitelisted:

```
truffle(develop)> stateAuthority.insertVerifiedEntity(consumer2);
{
  tx: '0x078278aa53b079fb0fcfcec0f9f664a57444d75850af682253c0b99b14a2c9f3',
  receipt: {
    transactionHash: '0x078278aa53b079fb0fcfcec0f9f664a57444d75850af682253c0b99b14a2c9f3',
    transactionIndex: 0,
    blockHash: '0x621baddbf05d1e43ec71477c1b08cbdf1ce037db312dd5ace24ee96f36fbf068',
    blockNumber: 12,
    from: '0x5d801e748d9931eb0eae907acf85157223a91e64',
    to: '0x07ad67c207640d5663c82a9fdc0bca0495529fae',
    gasUsed: 24271,
    cumulativeGasUsed: 24271,
    contractAddress: null,
    logs: [],
    status: true,
    logsBloom: '0x00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000',
    rawLogs: []
  },
  logs: []
}
```

- Aforementioned entities are verified by **‘stateAuthority’** using ‘**verify**(address _addr)’ function as follows:

```
truffle(develop)> stateAuthority.verify(owner);
true
truffle(develop)> stateAuthority.verify(consumer1);
true
truffle(develop)> stateAuthority.verify(consumer2);
true
truffle(develop)>
```

- Another consumer/intermediary called Consumer3 is created as follows:

```
truffle(develop)> let consumer3 = accounts[3];
undefined
truffle(develop)> consumer3
'0x7bD6A22360Eec71B7c0645613EAb1239Ca1Ef0B2'
truffle(develop)>
```

- A consumer could be removed from the whitelist using ‘**removeVerifiedEntity**(address _address)’ function as follows:

```
truffle(develop)> stateAuthority.insertVerifiedEntity(consumer3);
{
  tx: '0xfd940a6245491f7fa0ed0eed0beea98d82ec13f0aa5222799e0e8b5a37685569',
  receipt: {
    transactionHash: '0xfd940a6245491f7fa0ed0eed0beea98d82ec13f0aa5222799e0e8b5a37685569',
    transactionIndex: 0,
    blockHash: '0x333b6d48a43a2c7f4bf2575c0e101acf28207d1bac73a69b6b1541f079e9e578',
    blockNumber: 13,
    from: '0x5d801e748d9931eb0eae907acf85157223a91e64',
    to: '0x07ad67c207640d5663c82a9fdc0bca0495529fae',
    gasUsed: 43471,
    cumulativeGasUsed: 43471,
    contractAddress: null,
    logs: [],
    status: true,
    logsBloom: '0x00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000',
    rawLogs: []
  },
  logs: []
}
truffle(develop)> stateAuthority.removeVerifiedEntity(consumer3);
{
  tx: '0xb8526bc04942e97738964db9022315cf0d5045a24df91a21c9376b85e7d82760',
  receipt: {
    transactionHash: '0xb8526bc04942e97738964db9022315cf0d5045a24df91a21c9376b85e7d82760',
    transactionIndex: 0,
    blockHash: '0x64db2b317de3a9518a4028419e43c3fb66f9ba1f9cab94e8b80dc478990db0d0',
    blockNumber: 14,
    from: '0x5d801e748d9931eb0eae907acf85157223a91e64',
    to: '0x07ad67c207640d5663c82a9fdc0bca0495529fae',
    gasUsed: 14258,
    cumulativeGasUsed: 14258,
    contractAddress: null,
    logs: [],
    status: true,
    logsBloom: '0x00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000',
    rawLogs: []
  },
  logs: []
}
truffle(develop)> stateAuthority.verify(consumer3);
false
```

ProductProvenance Contract Interactions

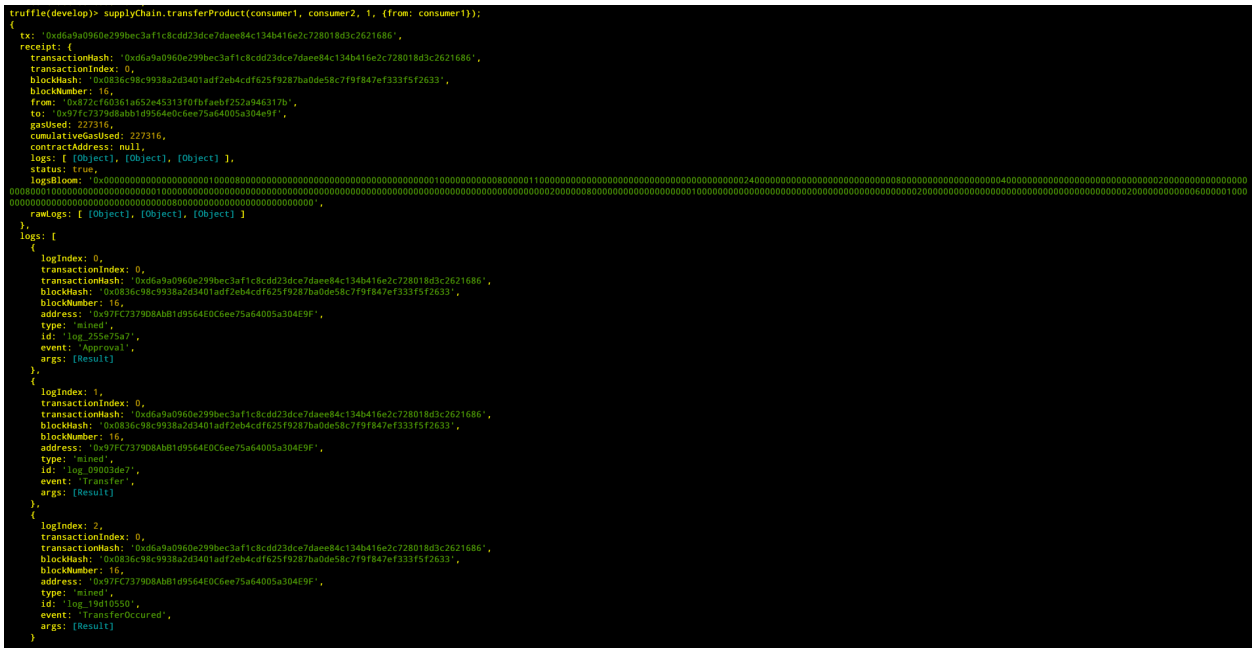
- **ProductProvenance** contract is deployed and can be interacted as '**supplyChain**'

- A whitelisted consumer (i.e Consumer1, Consumer2) is allowed to mint a token/product.
    - This operation is done using **mintProduct**() function.
    - This function uses the built-in **_mint**() function, and replicates its default behaviour.
    - Minted token is transferred to message sender from the contract owner (**Owner**)
- **Consumer1** minted a token:

- **TokenID** starts from zero and incremented at each minting operation. Hence one can trace product provenance by using **traceTokenHistory**(uint256 tokenId) function as follows:

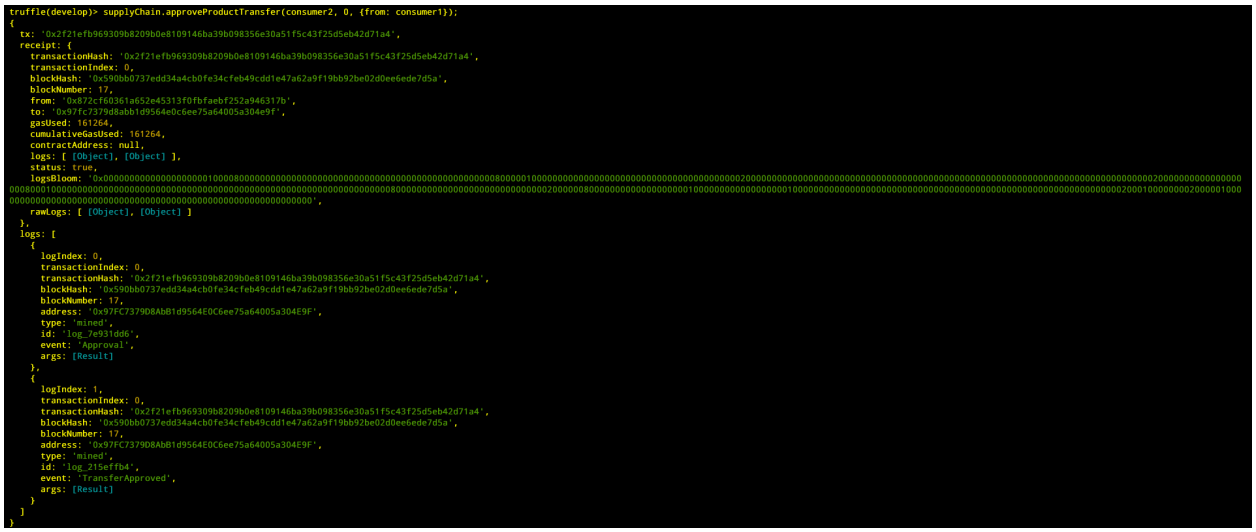
- To transfer a token/product,
    - **The token owner** can directly transfer using **transferProduct**(address _addressFrom, address _addressTo, uint256 tokenId) function
    - Any other consumer should be approved by **the token owner** to transfer the token on behalf of the owner using **approveProductTransfer**(address _addressTo, uint256 tokenId) function.

- **Consumer1 (as token owner)** can directly transfer the token to **Consumer2** as follows:



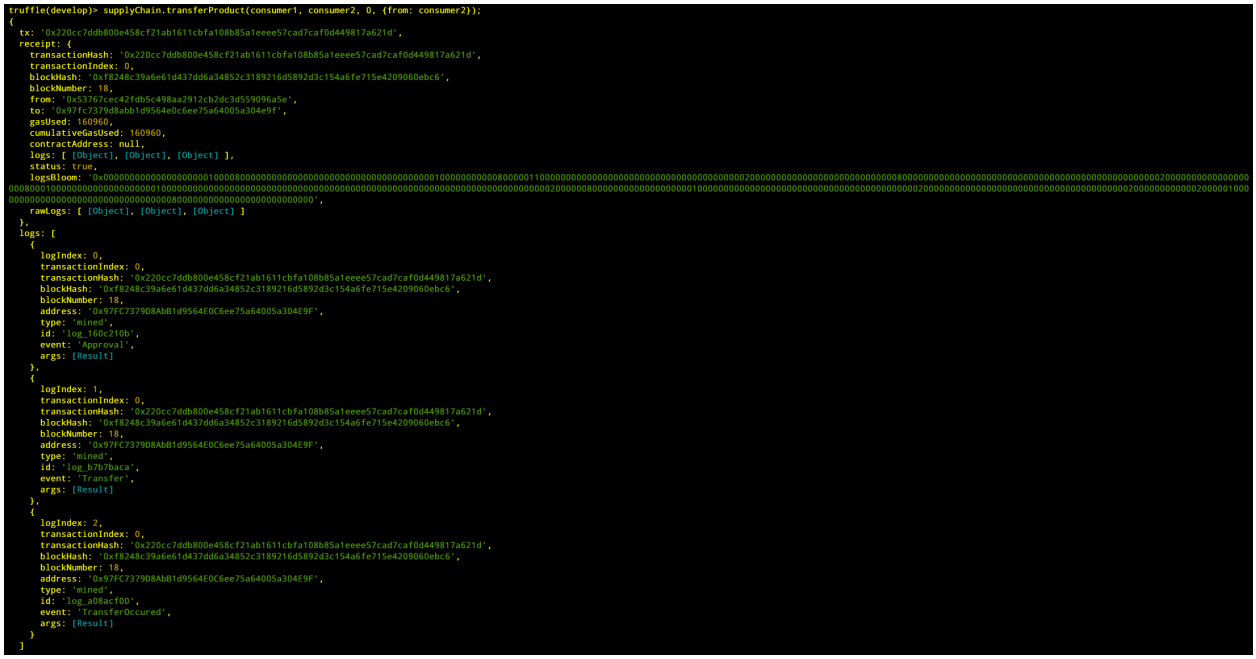
- Let's trace the token history post-transfer, notice that **transfer is occured**:

- **Consumer2** can transfer the token of **Consumer1** to himself, but first **Consumer1** should approve **Consumer2** for a specific token to be transferred:



- Let's trace the token history post-transfer-approval, notice that the **transfer is approved**:

- **Consumer2** could now complete the transfer, since it is approved by **Consumer1**:



- Let's trace the token history post-transfer-approval, notice that the **transfer is occured**:

- Let's try to make a transfer to a non-whitelisted entity:
    - The receiver entity of this function is **NOT** whitelisted by the state authority!

```
truffle(develop)> supplyChain.transferProduct(consumer2, consumer3, 0, {from: consumer2});
Uncaught:
Error: Returned error: VM Exception while processing transaction: revert The receiver entity of this functi
on is NOT whitelisted by the state authority! -- Reason given: The receiver entity of this function is NOT
whitelisted by the state authority!.
```

- Let's try to mint a token as a non-whitelisted entity:
    - The receiver entity of this function is **NOT** whitelisted by the state authority!

```
truffle(develop)> supplyChain.mintProduct({from: consumer3});
Uncaught:
Error: Returned error: VM Exception while processing transaction: revert The receiver entity of this functi
on is NOT whitelisted by the state authority! -- Reason given: The receiver entity of this function is NOT
whitelisted by the state authority!.
```

- Let's try to whitelist an entity as a consumer/intermediary:
    - This function is **restricted** to the contract owner!

```
truffle(develop)> stateAuthority.insertVerifiedEntity(consumer3, {from: consumer4});
Uncaught:
Error: Returned error: VM Exception while processing transaction: revert This function is restricted to the
 contract owner -- Reason given: This function is restricted to the contract owner.
```

- Let's try to remove a whitelisted address from state registry as a **white-listed** consumer/intermediary:
    - This function is **restricted** to the contract owner!

```
truffle(develop)> stateAuthority.removeVerifiedEntity(consumer2, {from: consumer1});
Uncaught:
Error: Returned error: VM Exception while processing transaction: revert This function is restricted to the
 contract owner -- Reason given: This function is restricted to the contract owner.
```

- Let's try to trace a token/product history as a non white-listed entity:
    - The caller entity of this function is **NOT** whitelisted by the state authority!

```
truffle(develop)> supplyChain.traceTokenHistory(0, {from: consumer4});
Uncaught:
Error: Returned error: VM Exception while processing transaction: revert The caller entity of this function
 is NOT whitelisted by the state authority!
```