# 585.https://stackoverflow.com/questions/69580545/transaction-complete-but-nft-not-transferred

**T:**Transaction complete, but NFT not transferred

**Q:**I created a smart contract, minted an nft, and now trying to transfer it.My problem is that transaction completes fine- I can see it in etherscan etc, but nft is not transferred. What are the possible root causes?  
  
async function transferNFT() { const nonce = await web3.eth.getTransactionCount(PUBLIC\_KEY, 'latest'); //get latest nonce //the transaction const tx = { 'from': sender, 'to': receiver, 'nonce': nonce, 'gas': 500000, 'data': myContract.methods.transferFrom(sender,receiver, 1).encodeABI() } const signPromise = web3.eth.accounts.signTransaction(tx, PRIVATE\_KEY) signPromise .then((signedTx) => { web3.eth.sendSignedTransaction( signedTx.rawTransaction, function (err, hash) { if (!err) { console.log( "The hash of your transaction is: ", hash, "\nCheck Alchemy's Mempool to view the status of your transaction!" ) } else { console.log( "Something went wrong when submitting your transaction:", err ) } } ) }) .catch((err) => { console.log(" Promise failed:", err) })}transferNFT()  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
And the contract. I actually don't call the transfer function as I was expecting to use the openzeppelin transferFrom function. But if I use the transfer function from the contract - the result is the same:  
  
 ● transaction is executed  
  
  
 ● NFT is not transferred.  
 contract MyNFTv2 is ERC721URIStorage, Ownable {  
 using Counters for Counters.Counter;  
 Counters.Counter private \_tokenIds;  
 event NftBought(address \_seller, address \_buyer, uint256 \_price);  
 event Transfer(address indexed \_from, address indexed \_to, uint256 indexed \_tokenId);  
  
 mapping (uint256 => uint256) public tokenIdToPrice;  
 mapping(uint256 => address) internal idToOwner;  
  
  
 constructor() public ERC721("MyNFTv2", "NFT") {}  
  
 function mintNFT(address recipient, string memory tokenURI)  
 public onlyOwner  
 returns (uint256)  
 {  
 \_tokenIds.increment();  
  
 uint256 newItemId = \_tokenIds.current();  
 \_mint(recipient, newItemId);  
 \_setTokenURI(newItemId, tokenURI);  
  
 return newItemId;  
 }  
  
 function allowBuy(uint256 \_tokenId, uint256 \_price) external {  
 require(msg.sender == ownerOf(\_tokenId), 'Not owner of this token');  
 require(\_price > 0, 'Price zero');  
 tokenIdToPrice[\_tokenId] = \_price;  
 }  
  
 function disallowBuy(uint256 \_tokenId) external {  
 require(msg.sender == ownerOf(\_tokenId), 'Not owner of this token');  
 tokenIdToPrice[\_tokenId] = 0;  
 }  
  
 function buy(uint256 \_tokenId) external payable {  
 uint256 price = tokenIdToPrice[\_tokenId];  
 require(price > 0, 'This token is not for sale');  
 require(msg.value == price, 'Incorrect value');  
  
 address seller = ownerOf(\_tokenId);  
 \_transfer(seller, msg.sender, \_tokenId);  
 tokenIdToPrice[\_tokenId] = 0; // not for sale anymore  
 payable(seller).transfer(msg.value); // send the ETH to the seller  
  
 emit NftBought(seller, msg.sender, msg.value);  
 }  
  
 function transfer(address \_to, uint256 \_tokenId) public {  
 require(msg.sender == idToOwner[\_tokenId]);  
 idToOwner[\_tokenId] = \_to;  
 emit Transfer(msg.sender, \_to, \_tokenId);  
 }  
  
}

**C1:**It depends on the transferFrom() contract function. Can you share it, including its dependencies (or link to the full contract source code)?

**C2:**I have shared the contract code. But I thought it will use the default openzeppeline transferFrom function?

**C3:**Ok I wasn't aware that you're using the OpenZeppelin (and not custom) implementation. Does it fail with a specific error message? Or just the generic has been reverted? Can you link one of the failed transfers on a blockchain explorer?

**C4:**Its on ropsten testnet: ropsten.etherscan.io/tx/… It says transaction successful, but NFT transfer doesn't happen...

1 **Answer**

**A1:**const tx = { 'from': sender, 'to': receiver,  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
You're sending the transaction to the token receiver, which is a regular address that doesn't hold any smart contract.  
  
So when you pass them the data field, there's no contract to process it and the transaction just passes through ignoring the data field.  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
Solution: Set the to field of the transaction to the NFT collection contract. Then it will be able to process the tranferFrom() function and its arguments passed in the data field. Note that the receiver is already passed in the 2nd argument of the function.  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]