# 621.https://stackoverflow.com/questions/69171562/using-ipfs-to-mint-an-nft-the-safe-way

**T:**Using IPFS to mint an NFT the safe way

**Q:**I'm quite new to solidity and NFTs. In many tutorials, including the official IPFS tutorial, i see something like this:  
  
 function mintToken(address owner, string memory metadataURI) public returns (uint256) { \_tokenIds.increment(); uint256 id = \_tokenIds.current(); \_safeMint(owner, id); \_setTokenURI(id, metadataURI); return id; }  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
So if I understand it correctly anyone could call the mint function. Means everyone can put ANY url as the second parameter, right? So let's say somebody would abuse that an place any other image in the mint function, I would have this stored in my smart contract and would never be able to get rid of it? Am I right and is there a safe way? Didn't find something about that.

2 **Answer**

**A1:**Your assumption is correct - anyone can execute this function and pass any string parameter that would be stored in your contract.  
  
You can limit who can execute the function, for example using the ownable pattern. Here's an article by OpenZeppelin covering the topic - link.  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
It's a simple authorization scheme validating whether the sender is one particular address. If it is (the authorized address), the function continues to execute. If it isn't, the execution reverts.  
  
modifier onlyOwner { // only allow requests from the `0x123` address // requests from other addresses revert require(msg.sender == address(0x123), 'Not authorized'); \_;}function mintToken(address owner, string memory metadataURI) public onlyOwner // added the modifier returns (uint256){ // ...}  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]

**C1:**but this only works if the owner would be the only one who mints?

**C2:**@ccc1da70-5e63-4879-a4cb-adb604 That's right. You can modify the authorization to use multiple allowed addresses. Or you can add a condition that only allows a certain domain... It all depends on your needs.

**A2:**I ended up not passing the ID to the mint function but instead passing a seed to the constructor to hash the tokenId. Using the same seed to hash my uploaded files to IPFS.