# 349.https://stackoverflow.com/questions/71474649/prevent-smart-contracts-to-mint-outside-of-the-website

**T:**Prevent Smart Contracts to mint outside of the website [closed]

**Q:**focused(/help/closed-questions)Want to improve this question? Update the question so it focuses on one problem only by editing this post.  
  
Closed 7 months ago.  
  
  
  
 Improve this question  
  
 (/posts/71474649/edit)I have this website which mints an avatar of the user's choice of pieces (background color, nose types, eyes types etc.).  
  
The contract that I am using right now works just fine for the website. However, it is able to mint from snowtrace as well. I want to limit my contract such that it should only allow minting from the website.  
  
I want to prevent any others to mint dummy stuff in my contract using contract interaction snowtrace. Contract should only allow minting from the website.

3 **Answer**

**A1:**I think it is possible using signatures hash, the mint function would require a signature from an address, that private key address would be held on website backend, it would sign a message to be validated on the mint function.  
  
As long as the private key is only accessed by the website backend, always the user would need to pass through that. I'm just not sure how you would prevent user to mint one more time if he/she uses a previous hash message already known, but I guess it could be solved as well.  
  
Some resources that could be helpful as well:https://solidity-by-example.org/signature/https://blog.chainsafe.io/how-to-verify-a-signed-message-in-solidity-6b3100277424  
  
update: you can prevent user to mint outside website even if he/she knows a previous hash message by adding a dynamic data as part of the required hash message, as per example, the msg.sender and its nonce. So, even if the user try to mint again with a previous message it won't be valid because the nonce is different and not valid anymore, once contract would require a hash message with the latest nonce of the msg.sender.

**C1:**yes I used this exact logic. Basically my backend has an admin wallet and every time a minting request comes, backend generates a message and sign for that message. Before I mint my NFT I check whether this message and signatures are valid. If they are I keep them in a list in the contract so that they can't be used again and again. Thanks!

**C2:**Cool, that is a possibility as well, the possible downside is that gas fees will be paid through owner address. Restricting with signatures hash the gas fee can still be paid by the user, he/she only need to pass the hashed message as argument (which is handled by the server). I edited the main answer to include a way to mint that could prevent user to mint directly on contract if he/she has minted before and know any hashed message signed by the server addresss.

**C3:**just be careful, since having that private key exposed online can be a vulnerability and never use the same contract owner wallet for that

**C4:**Yes, the private key is supposed to be private in a reliable server, it also is suppose to be possible to be changed by a function in the contract accessed only by the owner.

**A2:**Well, i don't think that's possible. You can't check that condition.Unless maybe you create another contract and restrict the minting only for that contract. You use that new contract to interact in your website and you don't verify it on Snowtrace so you cant interact with it directly from Snowtrace.

**C1:**Thank you for your answer. I used some kind of an admin wallet in my backend and made my contract minting function only owner mintable. Thus, only backend (a.k.a. admin/owner wallet) is able to mint. Any interaction (other than owner interactions) on Snowtrace returns an error sayin only owner can mint. Thank you!

**A3:**Not really possible. The smart contract can't know where that transaction is coming from. Your best bet is to only allow mints at a specific time.