## Prevention & Patching

Generally, preventing timing vulnerabilities is not easy since we must consider differences in processing time and what kind of information these differences might reveal to an attacker. In our case, we must implement the permission check before the computation of file meta-information. Thus, the function can return early if the user has insufficient permissions, and the web server can send an early response. Thus, there is no significant timing difference if the user provided a valid or invalid path.

We could implement this by adding a user argument to the get\_file\_details function and returning early in case of insufficient permissions:

Code: python

# return fileowner, filesize (recursively), and number of subfiles (recursively)  
def get\_file\_details(path, user):  
 try:  
 if not os.path.exists(path):  
 return '', 0, 0  
  
 # permission check  
 path = Path(path)  
 owner = path.owner()  
 if (user != 'root') and (user != owner):  
 return '', 0, 0  
  
 # number of subfiles  
 filecount = 0  
 for root\_dir, cur\_dir, files in os.walk(path):  
 filecount += len(files)  
  
 # file size  
 filesize = sum(f.stat().st\_size for f in path.glob('\*\*/\*') if f.is\_file())  
  
 return owner, filesize, filecount  
  
 except:  
 return '', 0, 0