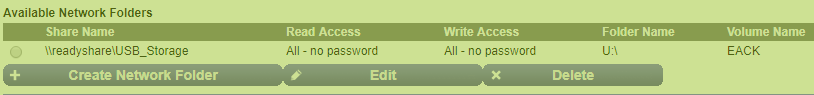
Information disclosure

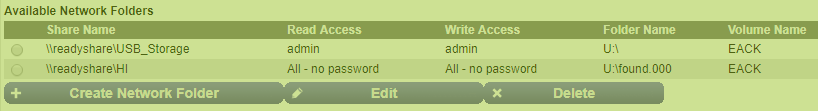
There is no control over access rights when accessing USB shared files. It is possible to bypass password authentication and obtain arbitrary directory files.

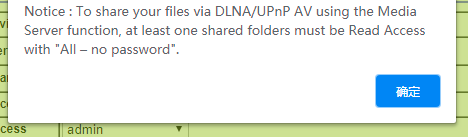
The vulnerability scenario is as follows:  
When the user inserts the USB, a shared root directory "\\readyshare\USB\_Storage" will be generated in the control page. The default password is blank.



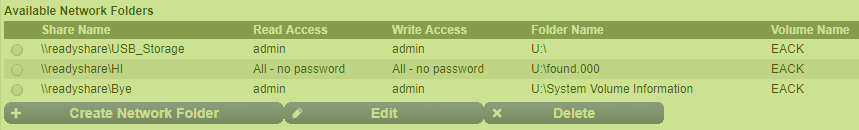
Let's assume that the user has set a password for it, and then the user creates another (or more) shared folder.

Named "\\readyshare\HI", and according to the rules for creating folders, you can only select the existing folder from the USB directory as the new shared directory. The access permissions of this shared directory are set to be readable by any user (ps: To share your files via DLNA/UPnP AV using the Media Server function, at least one shared folders must be Read Access with “All - no password”.)

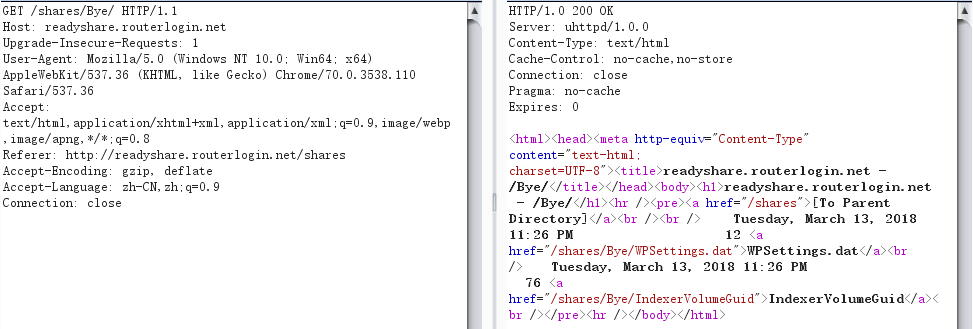




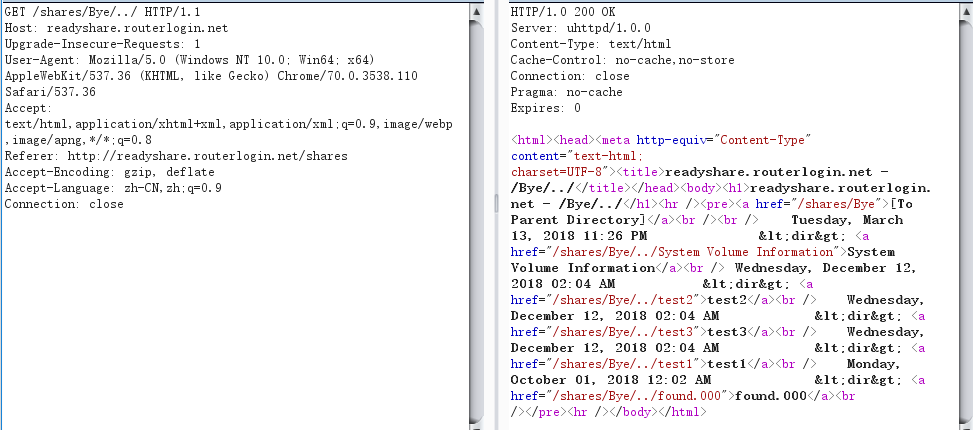
Because the two shared directories are the relationship between the parent and the child, in this case, it is too simple to read the file across the directory (just "../"), so I create another shared directory with the password set, named " \\readyshare\Bye", Then delete the root directory of the USB, Set to not share



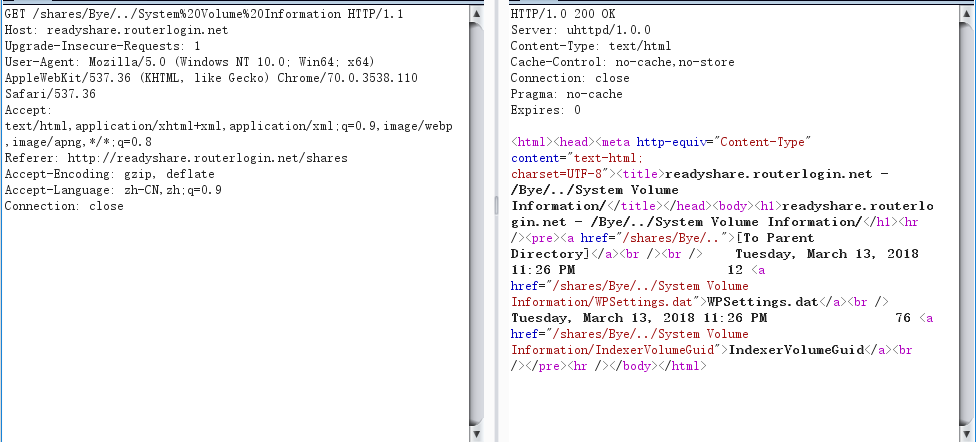
Now we are starting to verify the vulnerability:

We access the unencrypted shared directory "\\readyshare\HI" in the system's default http form.

Then change the url of the access to "GET /shares/Bye/../", you can access the root directory of the USB, although it is set to not share



Next, we can go to the "\\readyshare\Bye" directory parallel to the "\\readyshare\HI" directory, even though we don't know its encrypted password.



The above situation is because the program does not filter the '../', so that the attacker can obtain the user's encrypted file after connecting to the user wifi.