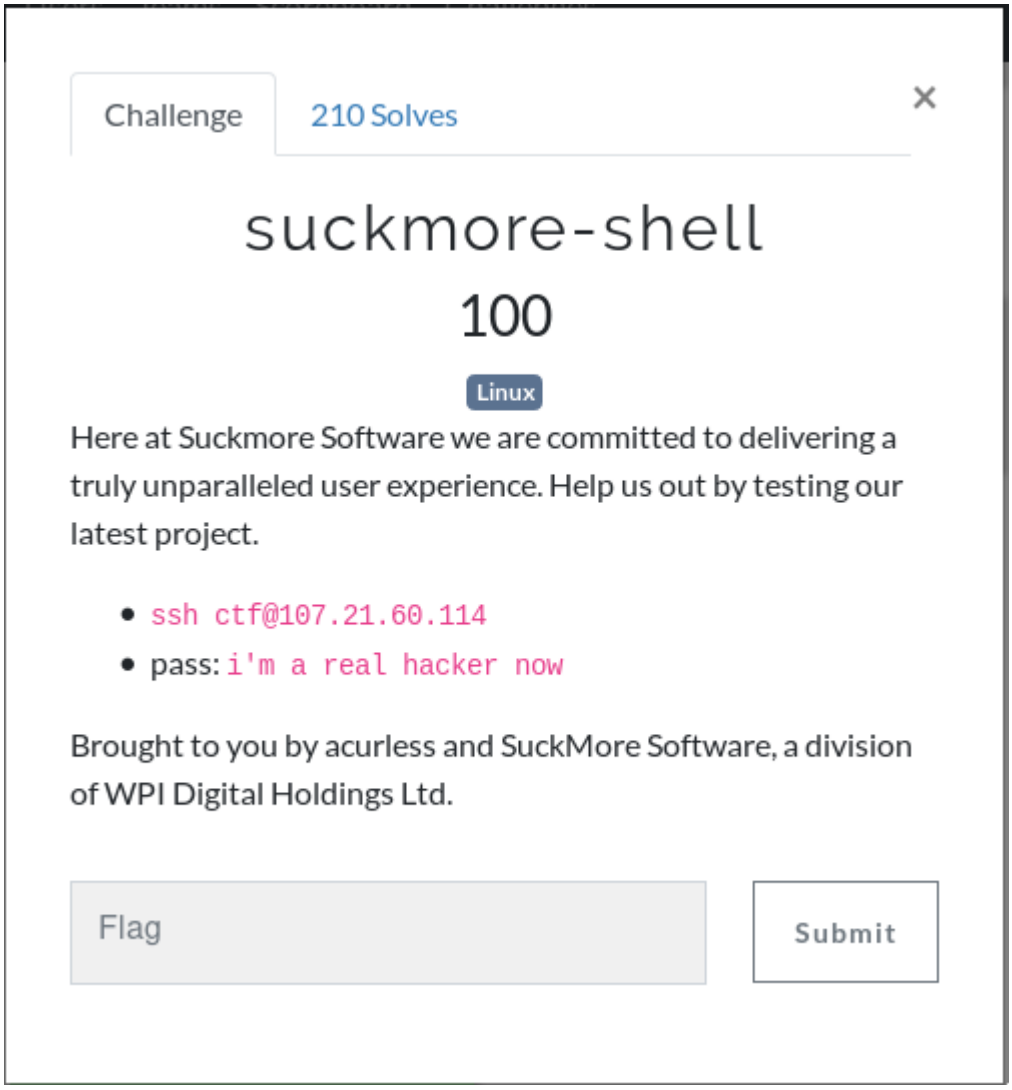
suckmore-shell

suckmore-shell

Author: Karol Baryła (aka Lorak_)

suckmore-shell was a task from WPI CTF 2019. Task description:



Here is whole process of solving task, explanation below:

```
ssh ctf@107.21.60.114ctf@107.21.60.114's password:
https://wiki.armiaprezesa.pl/books/wpictf2019/page/wpiSuckMORE shell v1.0.1. Note: for POSIX
support update to v1.1.0
suckmore>ls
suckmore>ls -a
sleep: invalid option -- 'a'
Try 'sleep --help' for more information.
suckmore>alias
alias bash='sh'
alias cat='sleep 1 && vim'
alias cd='cal'
alias cp='grep'
alias dnf=''
alias find='w'
alias less='echo "We are suckMORE, not suckless"'
alias ls='sleep 1'alias more='echo "SuckMORE shell, v1.0.1, (c) SuckMore Software, a division of
WPI Digital Holdings Ltd."'
alias nano='touch'
alias pwd='uname'
alias rm='mv /u/'
alias sh='echo "Why would you ever want to leave suckmore shell?"'
alias sl='ls'
alias vi='touch'
alias vim='touch'
alias which='echo "Not Found"'
suckmore>unalias -a
suckmore>ls
bash: /usr/bin/ls: Permission deniedsuckmore>/lib64/ld-linux-x86-64.so.2  /usr/bin/ls
/usr/bin/ls: error while loading shared libraries: /usr/bin/ls: cannot open shared object file:
Permission denied
suckmore>echo *bin boot dev etc home lib lib64 lost+found media mnt opt proc root run sbin srv
sys tmp usr var
suckmore>echo /root/*
/root/*
suckmore>echo /home/*
/home/ctf
suckmore>echo /home/ctf/*
```

```
/home/ctf/flag
suckmore>echo $(</home/ctf/flag)
WPI{bash_sucks0194342}
suckmore>Connection to 107.21.60.114 closed.
```

So, I was greeted by shell, probably bash. `ls` does nothing, `ls -a` gives us error from `sleep` command, so I know that `ls` is an alias for `sleep`. Next step is of course checking any other aliases for anything intresting, but all of them are just obastacles, so I deleted them with `unalias -a`, which makes the shell pretty much normal bash. Next step is to find and print flag, but it looks like executables don't have execute permission. No problem, lets execute them trough ld-linux (For those who don't know: it is something like an interpreter, but for binary files. Just as you can do `python ./scriptname.py` you can do `/lib64/ld-linux-x86-64.so.2 ./binaryname`. It allows us to bypass missing execution permissions). Unfortunately, it didn't work there, turns out those files didn't have read permissions too. It means that I needed to get flag only with bash builtins. Easy enough. `ls` can be replaced with `echo /path/*`, `cat` can be replaced with `echo $(< /path/to/file)`. That was enough to get the flag.

Revision #5
Created Mon, Apr 15, 2019 4:02 PM by Lorak_
Updated Thu, Jul 4, 2019 9:57 AM by Lorak_