USS

USS: Let’s Dive Deeper

cd circle && cd .. && cd - && ./shake_it_all_about.sh

12 steps 60 minutes

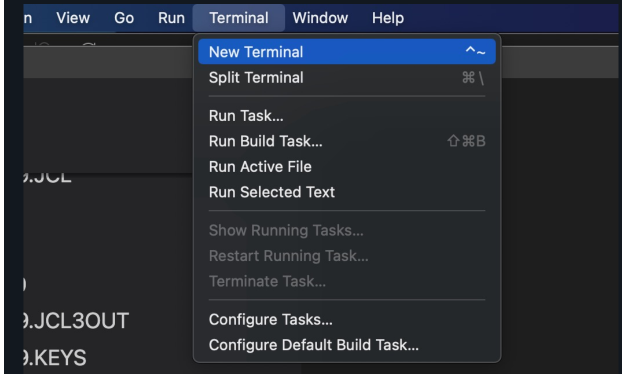
THE CHALLENGE

UNIX System Services (USS) is a POSIX-compliant implementation of a UNIX environment within z/OS that allows for a UNIX-like experience while still using the same system APIs as the z/OS we've been using so far.

All that will make more sense as you make your way through the shell, run a few scripts, and fall hopelessly in love with tab completion.

BEFORE YOU BEGIN

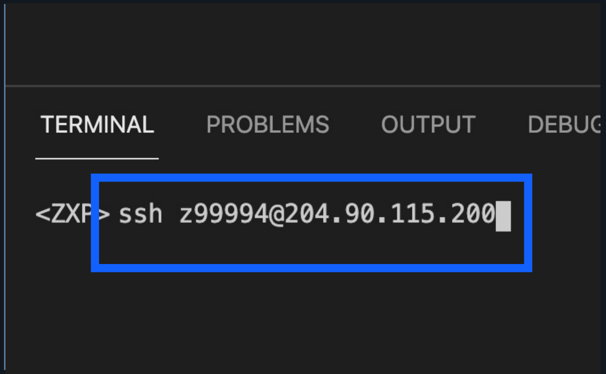
We're starting with a brand-new facet of z/OS, so you don't really need to know much else, but a working knowledge of how data sets and members will certainly help put things into perspective here as you learn about USS.



1. FIND THE TERMINAL

Look for the Terminal section in the lower portion of your VS Code window. If it’s not there, try using the Terminal menu and selection “New Terminal”. This is a text-based method of interacting with your own personal system.

If you have an older version of Windows or MacOS, you may need to download a separate SSH client, such as PuTTY. For guidance on this, consult the support forums.

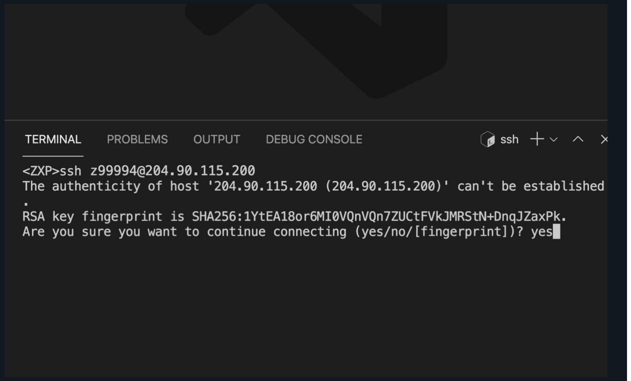


```
<ZXP> ssh z99994@204.90.115.200
```

2. CONNECT THROUGH SSH

Log into the z/OS system with the following command: **ssh zxxxxx@204.90.115.200** (replacing zxxxxx with your own userid. Put another way, this says “Use the ssh command to connect me (using this userid) to this system (at this IP address)”

If you get a scary message saying ***"Remote Host Identification Has Changed"***, it's probably because you've connected to this system before, at a different IP address. Welcome back, returning contestant! Follow the instructions in the message to remove the older entry from your known_hosts file.



```
<ZXP>ssh z99994@204.90.115.200
The authenticity of host '204.90.115.200 (204.90.115.200)' can't be established
.
RSA key fingerprint is SHA256:1YtEA18or6MI0VQnVQn7ZUCtFVkJMRStN+DnqJZaxPk.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

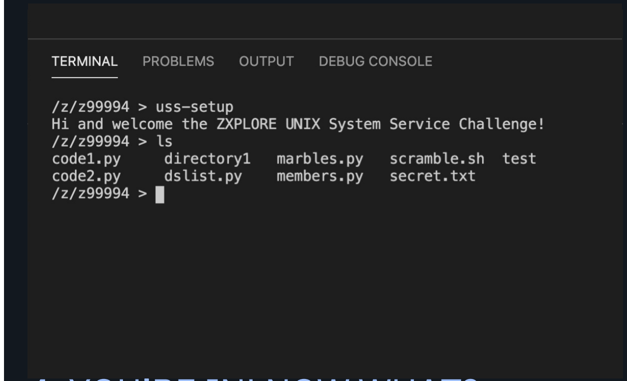
3. VERIFY ME, THANK YOU

You will be asked for your password, which is the same password you used to log into the z/OS system through VS Code. A few things:

1. ***You will not see any characters as you type your password!*** This is to make it so people looking over your shoulder can't steal your password, but the system can still see it.
2. You may be prompted to trust or accept a key from the remote system.
3. You may see “Authenticity can’t be established” (see screenshot above)

You can safely say **yes** to any of these prompts.

USS It’s a Bash!



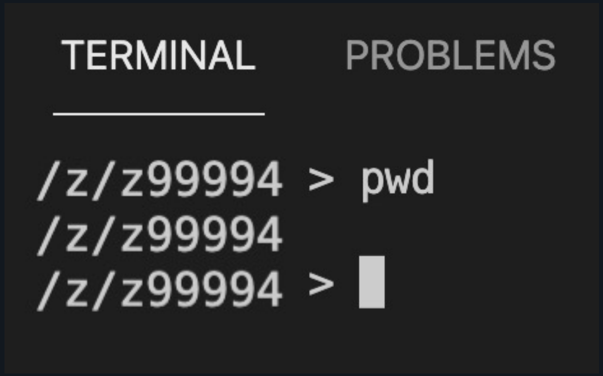
```
/z/z99994 > uss-setup
Hi and welcome the ZXPLORE UNIX System Service Challenge!
/z/z99994 > ls
code1.py     directory1   marbles.py   scramble.sh  test
code2.py     dslist.py    members.py   secret.txt
/z/z99994 > █
```

4. YOU’RE IN! NOW WHAT?

Now that you're logged into the USS environment with ssh, you can look around with the command ***ls***. This will show all of the files and folders (also known as directories in UNIX-land). If your default (“home”) directory is empty, enter the command ***uss-setup***, and then run the *ls* command again.

USS uses a hierarchical structure, where there are files and directories within other directories. You're probably used to this type of setup on your own computer, where you can keep files in folders, and put folders within folders and so on.

If you want to disconnect, just enter the command ***exit*** and you will be logged out. You can use **ssh** to get reconnected later.

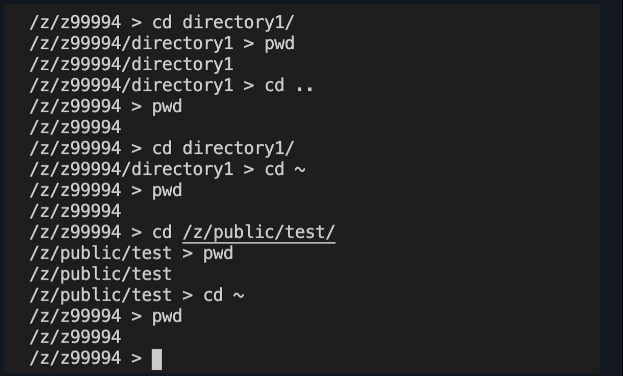


```
/z/z99994 > pwd
/z/z99994
/z/z99994 > █
```

5. GETTING ORIENTED

In addition to looking around with ***ls***, you may also want to know "Where am I?". The command prompt will usually show you where you are, but you can also type ***pwd*** to *print* the *working directory*.

This will come in handy shortly because we'll want to start using ***cd*** to move around the filesystem. Right now, you're in your home directory, which is where your USS files live. You can get back here any time by typing ***cd ~*** *(that's the tilde key, usually on the top left corner of the keyboard)*



```
/z/z99994 > cd directory1/
/z/z99994/directory1 > pwd
/z/z99994/directory1
/z/z99994/directory1 > cd ..
/z/z99994 > pwd
/z/z99994
/z/z99994 > cd directory1/
/z/z99994/directory1 > cd ~
/z/z99994 > pwd
/z/z99994
/z/z99994 > cd /z/public/test/
/z/public/test > pwd
/z/public/test
/z/public/test > cd ~
/z/z99994 > pwd
/z/z99994
/z/z99994 > █
```

6. CHANGE DIRECTORY WITH CD

To navigate into another directory, type ***cd***, followed by the name of the directory. For example, we can type *cd directory1* and we'll go into directory1, assuming that's a directory we can see with the *ls* command. Try it out, then type *pwd*.

You should see that *pwd* now shows your path as /z/zxxxxx/directory1.

To go back to your home directory, we need to go back one level. We can go back one level by typing ***cd ..*** (two dots), or we can use the tilde shortcut to go to our home directory from anywhere (***cd ~***)

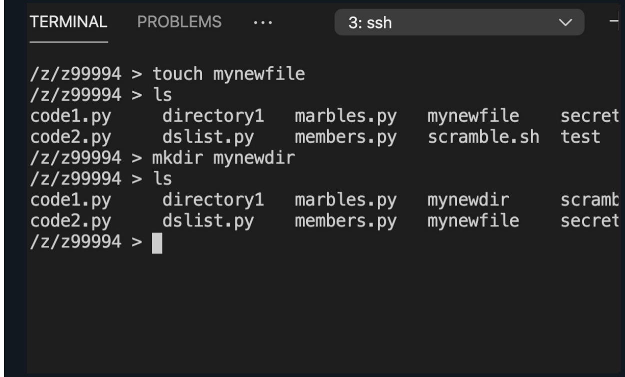
So far, we've been moving back and forth one step at a time. We can also change directories by specifying the full path we want to go to. For example, ***cd /z/public/test*** will take us directly to that new location. Then we can use ***cd ~*** to go back home after we've gone there and looked around a little. (Again, if your home directory is empty, run the ***uss-setup*** command)

MAKE LIFE EASIER. USE TAB COMPLETION AND UP/DOWN ARROWS

Notice how the word “directory” gets annoying to type out after a while? Try this neat trick... the next time you have to type out a long command or filename, type out the first few letters, then hit **Tab.**

Hitting it once will auto-complete as much of the command or name as it can and hitting it twice will show you all of the possible completions. So, if you have a **directory1** and a **directory2,** you can type out **cd dire** followed by hitting the Tab key, and it will auto-complete the “**directory”** part. Hit it one more time and it will tell you “there’s directory1 and directory 2”. Give it a try, and you’ll be amazed at how much faster you can get from the command prompt.

One more trick, if you want to use a command you typed not that long ago, you can recall recent commands by hitting the **Up arrow** on your keyboard. Then just hit **Enter** to use it. Pretty sweet, right?



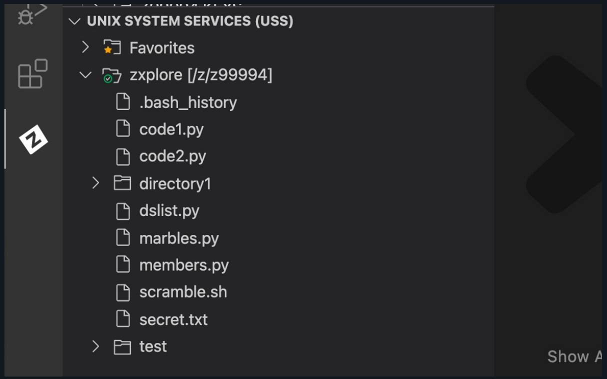
```
/z/z99994 > touch mynewfile
/z/z99994 > ls
code1.py     directory1   marbles.py   mynewfile    secret
code2.py     dslist.py    members.py   scramble.sh  test
/z/z99994 > mkdir mynewdir
/z/z99994 > ls
code1.py     directory1   marbles.py   mynewdir     scramb
code2.py     dslist.py    members.py   mynewfile    secret
/z/z99994 > █
```

7. FRESH FILES AND FOLDERS

The ***touch*** command is commonly used to update the "last modified" timestamp of a file but can also be used to create an empty file. Enter the command ***touch mynewfile*** and follow that up with a ***ls*** to see a brand-new file.

You can create brand new directories with the ***mkdir*** command. For example, try ***mkdir mynewdir*** and you should see a shiny new directory afterwards.

Feel like cleaning up afterwards? Check out the grey box underneath this step right down there.

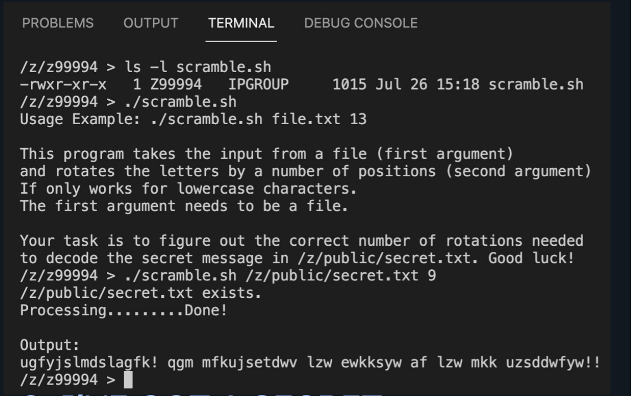


8. INVITE ZOWE TO THE PARTY

You can view your USS files and directories through Zowe. Just click the magnifying glass (🔎) next to your profile in USS and enter the full path of your home directory.

Your home directory is /z/zxxxxx

Make sure you use all lowercase letter, and that you use YOUR userid. Not zxxxxx or z99994. Don't worry, we won't keep saying this every time… just a few *more* times.



```
/z/z99994 > ls -l scramble.sh
-rwxr-xr-x   1 Z99994   IPGROUP     1015 Jul 26 15:18 scramble.sh
/z/z99994 > ./scramble.sh
Usage Example: ./scramble.sh file.txt 13

This program takes the input from a file (first argument)
and rotates the letters by a number of positions (second argument)
If only works for lowercase characters.
The first argument needs to be a file.

Your task is to figure out the correct number of rotations needed
to decode the secret message in /z/public/secret.txt. Good luck!
/z/z99994 > ./scramble.sh /z/public/secret.txt 9
/z/public/secret.txt exists.
Processing.........Done!

Output:
ugfyjslmdslagfk! qgm mfkujsetdwv lzw ewkksyw af lzw mkk uzsddwfyw!!
/z/z99994 > █
```

9. I'VE GOT A SECRET

You've got a program called scramble.sh in your home directory. If you don't, look in the grey box in the bottom left corner for how to get it back.

You can tell this is an executable program because when you enter the command ***ls –l*** it shows up with an **x** in the fourth spot. This means that in addition to you being able to **R**ead and **W**rite it, you can also e**X**ecute it. There's a lot more to know about permission bits, but we'll save that for a more advanced USS lesson, perhaps.

For now, just know that you can run the program with ***./scramble.sh*** and the output of the program will tell you everything you need to know. Good luck.

Hint: The correct value for the number of rotations is somewhere between 1-26. Use your skills of deduction to try and figure out the value in as few tries as possible.

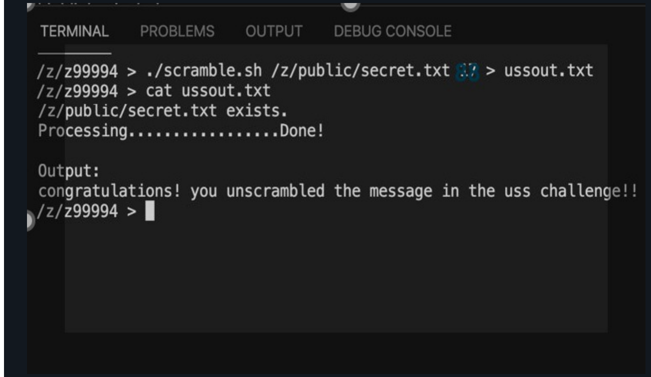
"I MADE A BUNCH OF FILES AND FOLDERS; HOW DO I GET RID OF THEM?"

To remove a file, type **rm** followed by the filename you want to delete. For example, **rm mynewfile.**

You can also use the command **rmdir** to erase an empty directory. For example, **rmdir mynewdir** will get rid of that directory you made in Step #7.

There are ways to specify non-empty directories, as well as ways you can delete lots of files all at once. If you have some experience in a shell, you probably know these commands. If not, it's probably best we don't show them to you this early. We'd hate to see you delete all of your hard work.

If you DO accidentally delete something, you can usually find another copy of it in /z/public. Use the **cp** command to copy it. Example: **cp /z/public/test ~/test** will give you a fresh copy of **test.**



```
/z/z99994 > ./scramble.sh /z/public/secret.txt [redacted] > ussout.txt
/z/z99994 > cat ussout.txt
/z/public/secret.txt exists.
Processing.................Done!

Output:
congratulations! you unscrambled the message in the uss challenge!!
/z/z99994 > 
```

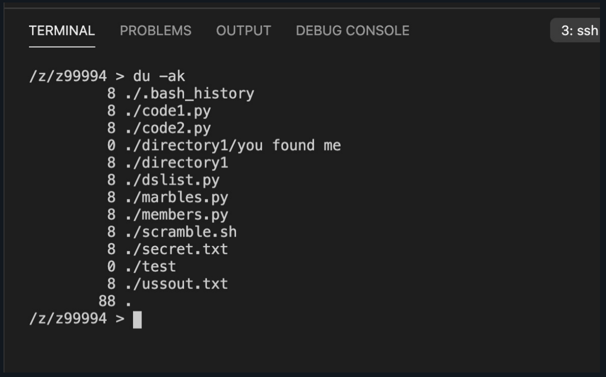
10. REDIRECT THE OUTPUT

Once you've cracked the code, let's put the output of the program into a file. This is super easy to do, using a redirection. Type or recall the command with the correct values, then append **> ussout.txt** to the end, so it will look like the screenshot above. You won't see output while it's working.

When you put > after a command, or anything that produces output, it says "Instead of putting output to the screen, stick it in this file instead". If the file doesn't already exist, it will create a new one for you, but be careful… it will also overwrite anything already in there. You can verify with the *cat* command.

You can also use >> to redirect the output to the bottom of the file, which is exactly what we'll do in Step #11.

You can check on what's in the file by opening it with VS Code in the USS view. Just keep in mind you may need to Right-Click and "Pull from Mainframe" to refresh it after writing out to it.



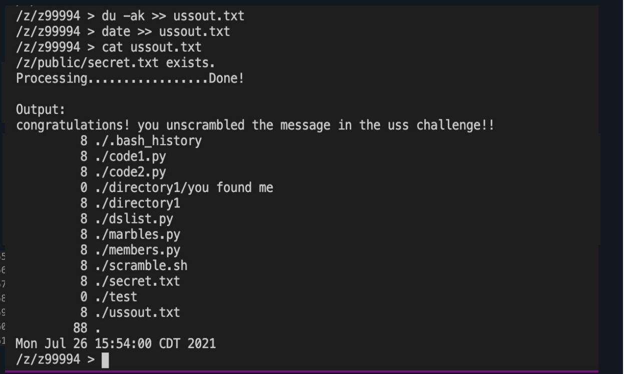
```
/z/z99994 > du -ak
         8 ./.bash_history
         8 ./code1.py
         8 ./code2.py
         0 ./directory1/you found me
         8 ./directory1
         8 ./dslist.py
         8 ./marbles.py
         8 ./members.py
         8 ./scramble.sh
         8 ./secret.txt
         0 ./test
         8 ./ussout.txt
        88 .
/z/z99994 > 
```

11. EXPLORING SPACE

Once you get the decoded output into your ussout.txt, append that file with the output of ***du –ak***

The du command outputs the **d**isk **u**sage of the directory you're in, plus all of the directories inside that directory (that's the a) and will give you the output in kilobytes. At this point, you should be using less than 50 total kilobytes.

When you do this correctly, your ussout.txt should have the secret decoded message followed by the output of your disk usage command.



```
/z/z99994 > du -ak >> ussout.txt
/z/z99994 > date >> ussout.txt
/z/z99994 > cat ussout.txt
/z/public/secret.txt exists.
Processing.................Done!

Output:
congratulations! you unscrambled the message in the uss challenge!!
         8 ./.bash_history
         8 ./code1.py
         8 ./code2.py
         0 ./directory1/you found me
         8 ./directory1
         8 ./dslist.py
         8 ./marbles.py
         8 ./members.py
         8 ./scramble.sh
         8 ./secret.txt
         0 ./test
         8 ./ussout.txt
        88 .
Mon Jul 26 15:54:00 CDT 2021
/z/z99994 > 
```

12. MAKE IT COUNT

Lastly, redirect the output of the ***date*** command into the same file. Same way as before, no tricks here.

Make sure your **ussout.txt** has the secret message, your disk usage output, plus the output of the date command. If it all looks good, then fire off CHKUSS from ZXP.PUBLIC.JCL and you're done!

NICE JOB! LET’S RECAP

You connected to USS through a secure shell (SSH) as well as through the Zowe plugin in VS Code. From there, you navigated directories, managed output, hacked some text, and learned all about arguments. It may. Not seem like much, but think of what you’ve done, and what you knew when you started.

If you’re into the UNIX way of doing things, you will probably really enjoy the LINUX-based challenge which is made available later on. If this wasn’t your cup of tea, at least now you’re done!

NEXT UP…

We’re going to keep working in the USS space for the next challenge, so make sure you keep those terminal commands and VS Code tabs handy.

We’ll be doing some very basic coding using Python. Don’t panic if you’ve never coded before! We’ll show you everything you need.