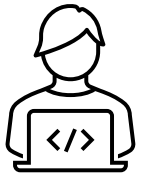
# Making the mainframe great and simple again with the Zowe Command Line Interface (aka CLI). #2997

Gene Johnston
eugene.Johnston@Broadcom.com
Priciple Software Engineer

Joe Winchester
winchest@uk.ibm.com
https://www.linkedin.com/in/joewinchester/
Open Mainframe Project Ambassador

# WHY ZOWE CLI ?

# Help context switch Integrate in z/OS tooling Skills onramp



z/OS

```
  Menu  Utilities  Compilers  Options  Status  Help
------------------------------------------------------------------------------
                ISPF Primary Option Menu for MV3B
Option ===> 3.4
                                                                More:      +
STANDARD OPTIONS                          HURSLEY EXTENSIONS

0  Settings     Terminal & user parms     DB Database   DB2 Managers
1  View         Display source data       DM Data Mgmt  Data Management
2  Edit         Create/change source      U  Utils      Utilities
3  Utilities    Perform utility funcs     SP SysProg    System Programming
4  Foreground   Interactive processing     I  IPCS       Dump Processing
5  Batch        Submit job processing     O  OMVS       UNIX System Services
6  Command      Enter TSO/wkstn cmds      R  RACF       Data Security Dialog
7  Dialog Test  Perform dialog testing    S  SDSF       Job/Output Display
8  LM Facility  Library admin funcs       LU Utils      Local Utilities
9  IBM Products Program dev products
10 SCLM         SW Config Lib Manager     X  EXIT       Exit from ISPF
11 Workplace    ISPF Workplace
12 z/OS System  z/OS sysprog appls
13 z/OS User    z/OS user appls

 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F10=Actions   F12=Cancel
 ONLINE-TLS 1.2                                        4,17
```

Change TSO picture to be a 3.4 animated gif
AND
a TSO listds picture
Trip to the web to get the help for the command

# WHO IS ZOWE ?

# Integrated help
# PC tools integration
# Faster onboarding


Add in picture about money and MIPS
Add in graphics for Jenkins, Tekton, JIRA
Github actions, q8s, Helm

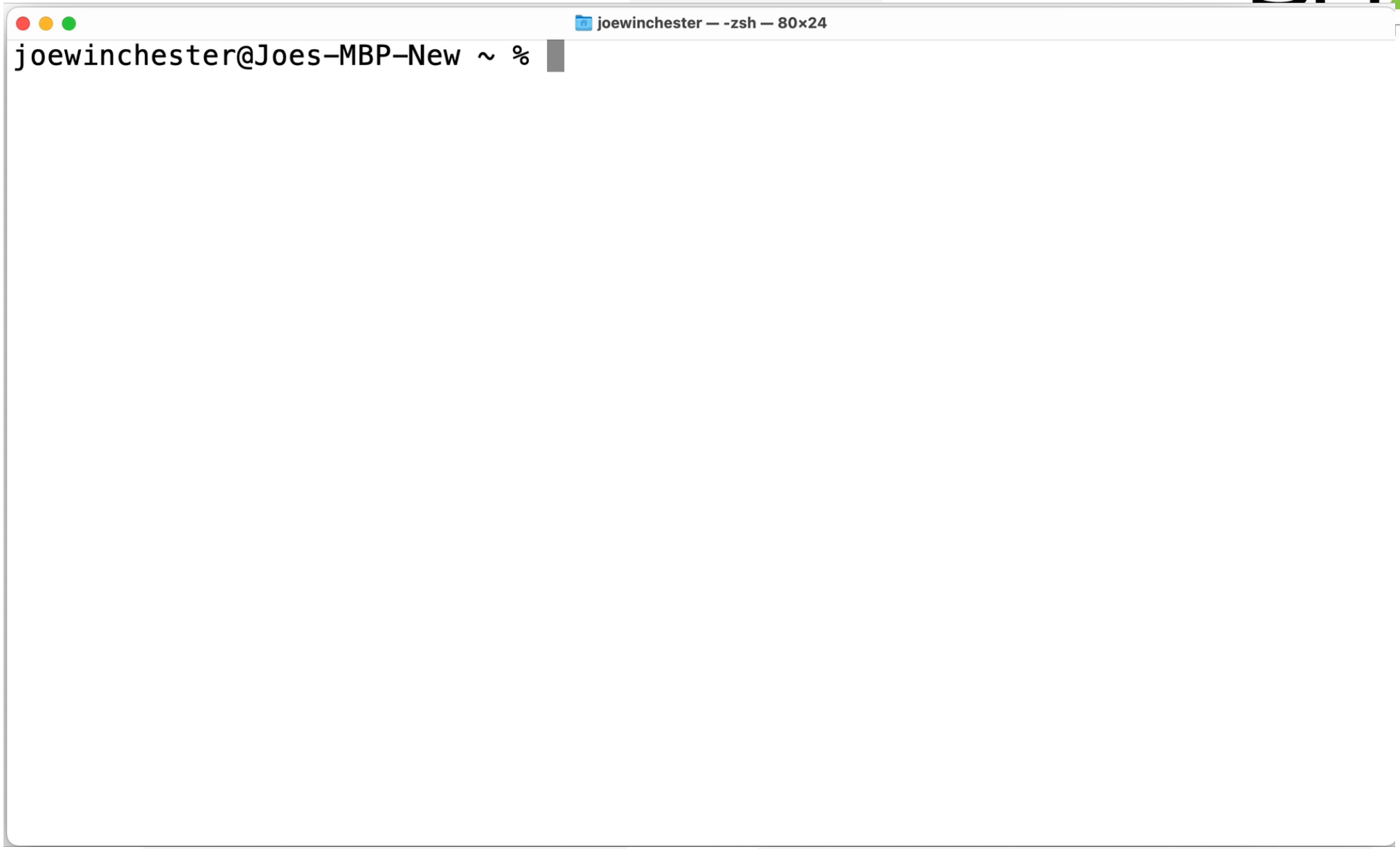
Built in help – show openssl, git, docker
Built in examples


**Zoë Alleyne Washburne**

**Biographical information**

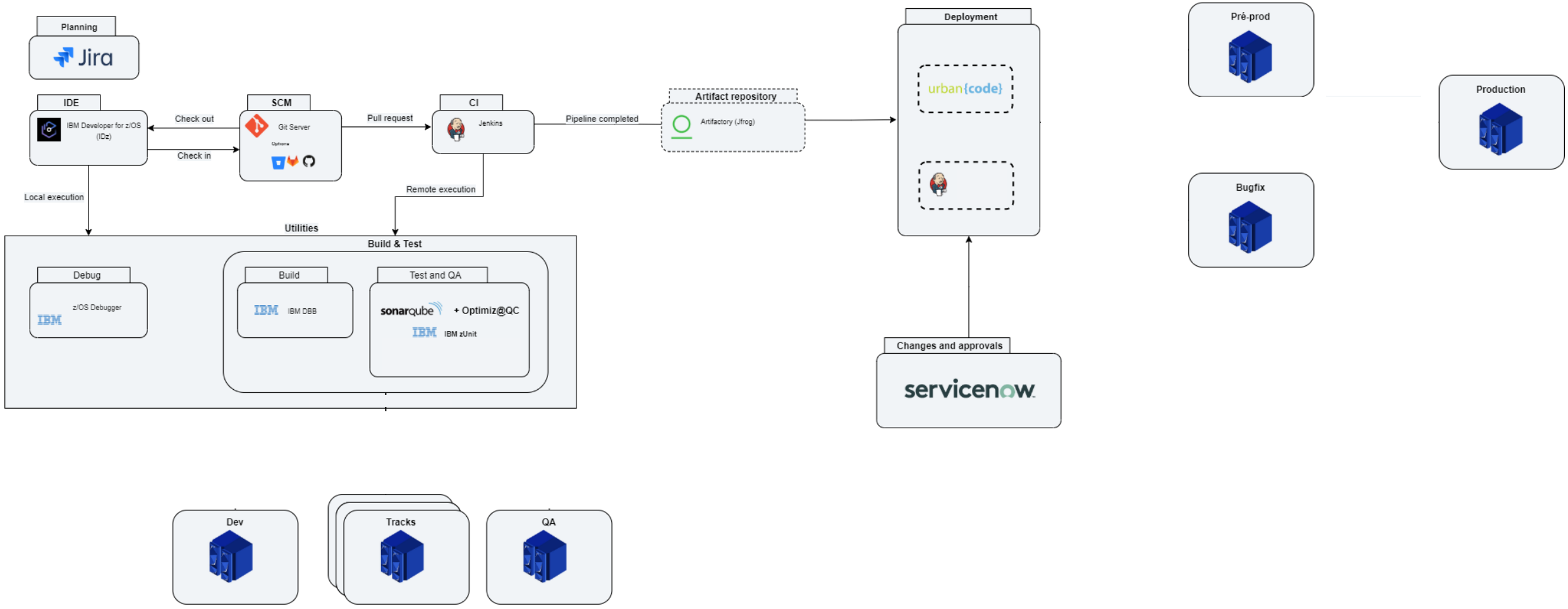
| | |
|---|---|
| Born | February 15, 2484 |
| Homeworld | None, born Vesselside |
| Gender | Female |

# CLI in action

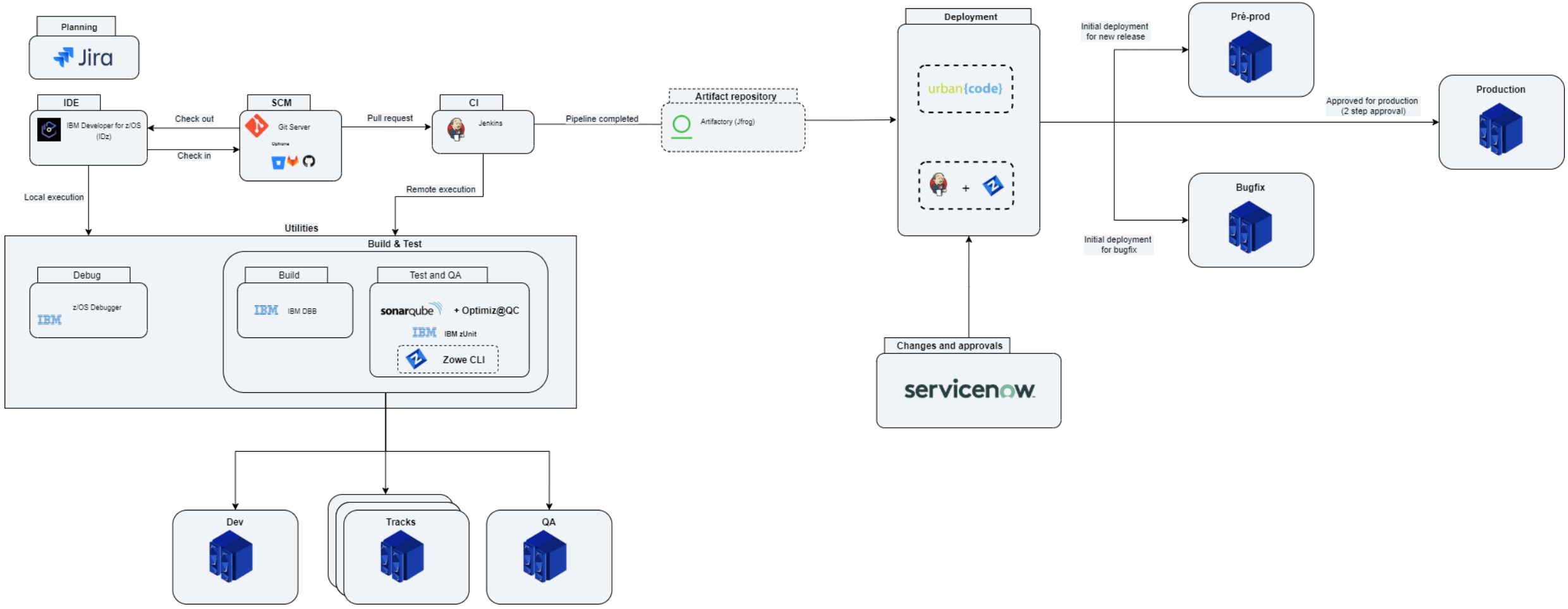


```
DEMO_PDS="STEVENH.DEMO.JCL"
ZOSMF_PROFILE=3bsh
# Check and see if pds already exists
MATCHES=`zowe zos-files list data-set "$DEMO_PDS" --zosmf-p $ZOSMF_PROFILE --response-format-json | jq -r '.data.apiResponse.returnedRows'`
if [ $MATCHES -gt 0 ]; then
    echo "Data set $DEMO_PDS already exists, deleting"
    zowe zos-files delete data-set -f "$DEMO_PDS" --zosmf-p $ZOSMF_PROFILE
fi

zowe zos-files create data-set-classic $DEMO_PDS --zosmf-p $ZOSMF_PROFILE
zowe zos-files upload stdin-to-data-set "$DEMO_PDS(INPUT)" <<< $1 --zosmf-p $ZOSMF_PROFILE
zowe zos-files upload stdin-to-data-set --zosmf-p $ZOSMF_PROFILE "$DEMO_PDS(COPY)" <<EOF
//COPY JOB 123456, 'TSTRADM',NOTIFY='TSTRADM',
//         CLASS=A,MSGCLASS=H,MSGLEVEL=(1,1)
//STEP1    EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=A
//SYSIN    DD DUMMY
//SYSUT1   DD DISP=SHR,DSN=$DEMO_PDS(INPUT)
//SYSUT2   DD DISP=SHR,DSN=$DEMO_PDS(OUTPUT)
//STEP2    EXEC PGM=AOPBATCH,PARM='sleep 5'
EOF
JOBID=`zowe jobs submit data-set "$DEMO_PDS(copy)" --zosmf-p $ZOSMF_PROFILE --response-format-json | jq -r '.data.jobid'`
echo "JOBID is $JOBID"

i="0"
while [ $i -lt 5 ]
do
    sleep 1s
    STATUS=`zowe jobs view job-status-by-jobid $JOBID --response-format-json --zosmf-p $ZOSMF_PROFILE | jq -r '.data.status'`
    if [ "$STATUS" = "OUTPUT" ]; then
        echo "Job $JOBID has now completed"
        i=5
    else
        echo "Waiting for job output to complete. Current status is $STATUS"
    fi
    i=$[$i+1]
done
```

# Distributed DevOps pipeline



# Distributed DevOps pipeline integrating z/OS



THE LINUX FOUNDATION

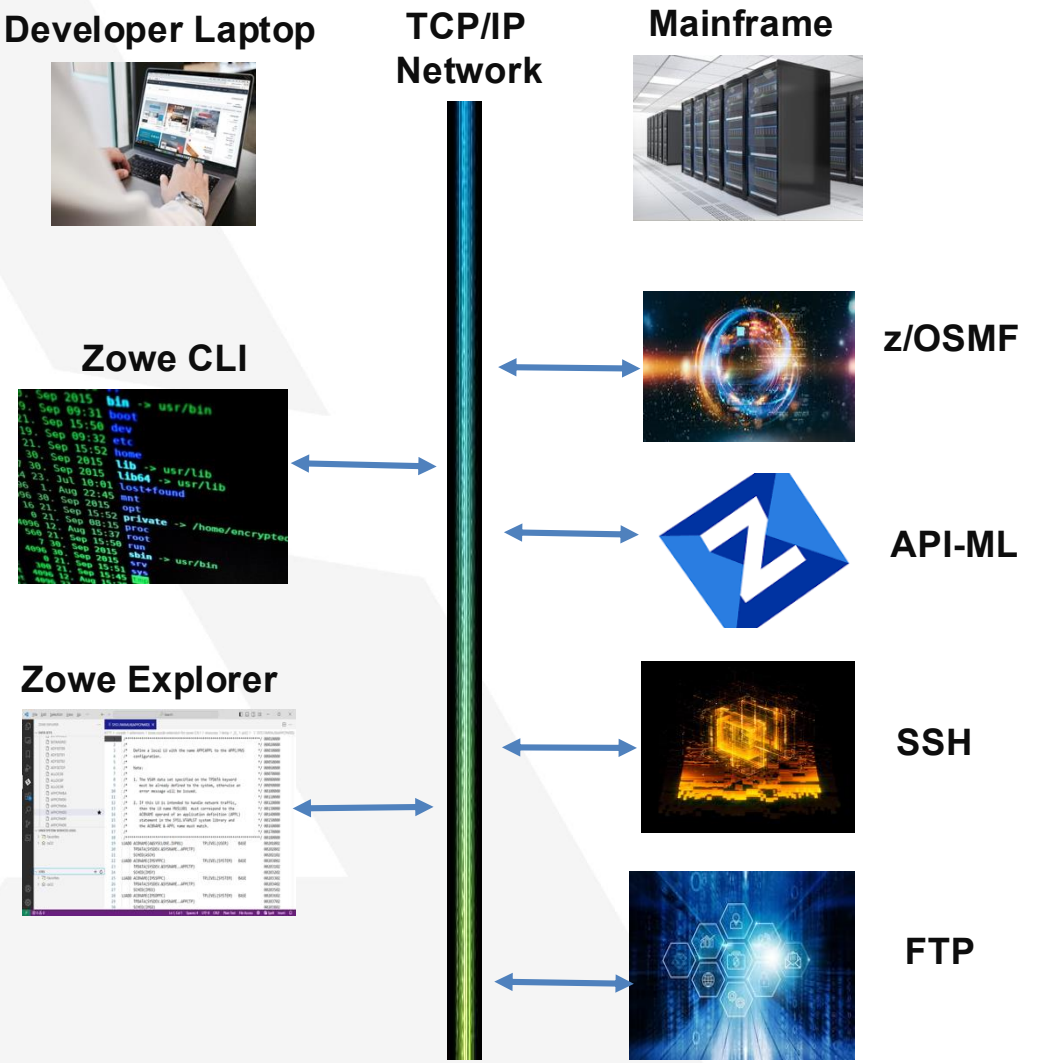
SHARE
EDUCATE • NETWORK • INFLUENCE

The Linux Foundation and the projects we support form the most ambitious and successful investment in the creation of shared technology

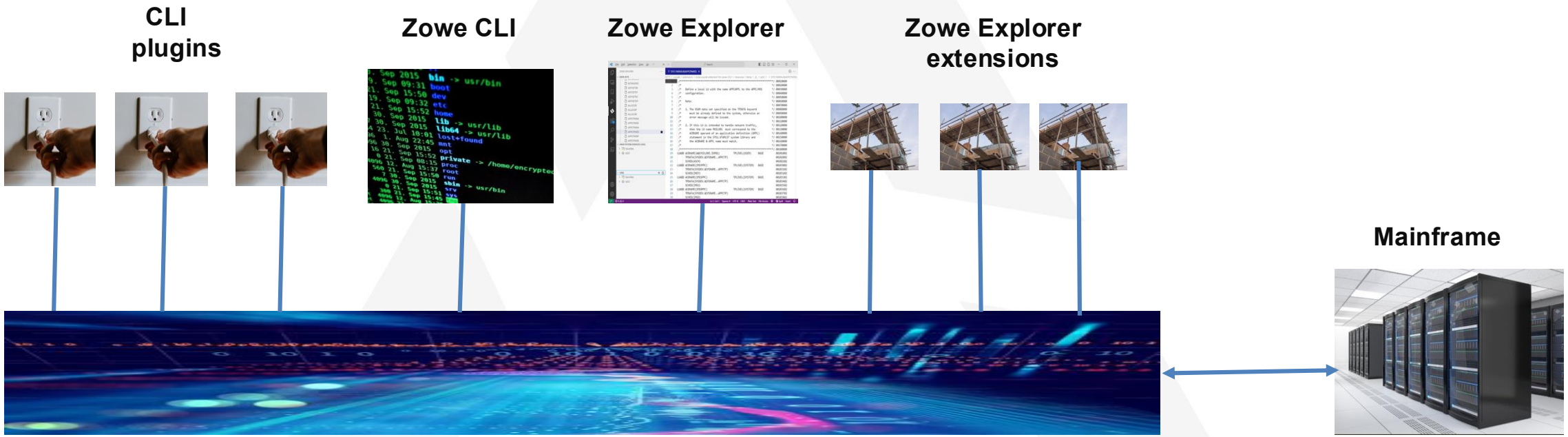
Jenkins
HELM
kubernetes
OPEN MAINFRAME PROJECT
node JS
Real-Time LINUX
GraphQL Foundation
OPEN MAINFRAME PROJECT Zowe
OPENAPI INITIATIVE
MOCHA

# Zowe clients within the mainframe ecosystem

- Install client apps on your laptop
  - Runs on Windows, Linux, & Mac
- Capabilities
  - Manage data sets
  - Submit & manage jobs
  - Issue commands to TSO and USS
  - Use CLI for automation and pipelines
  - ZE provides mainframe IDE experience
- Plugins and extensions expand these capabilities



# Client-side APIs are the foundation of our apps

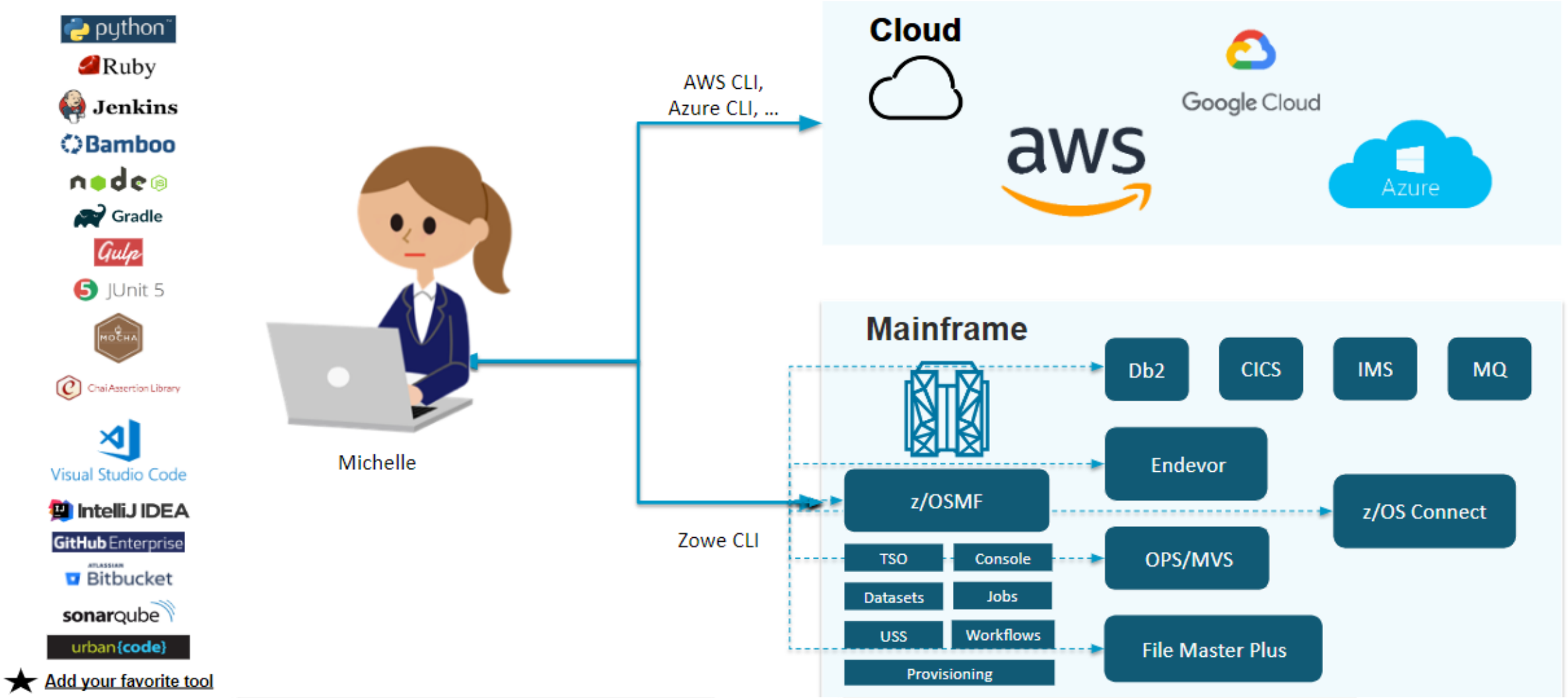


# Continuous Integration pipelines

Pipelines use scripted actions to create your app

- Compile source code
- Deploy libraries to test area
- Create test data sets
- Run jobs to test new logic
- Regression test exiting logic
- Trigger the procedures that bundle the app for release
- Sanity-test the bundle

# Include mainframe apps in CI pipelines



# WHAT'S NEW

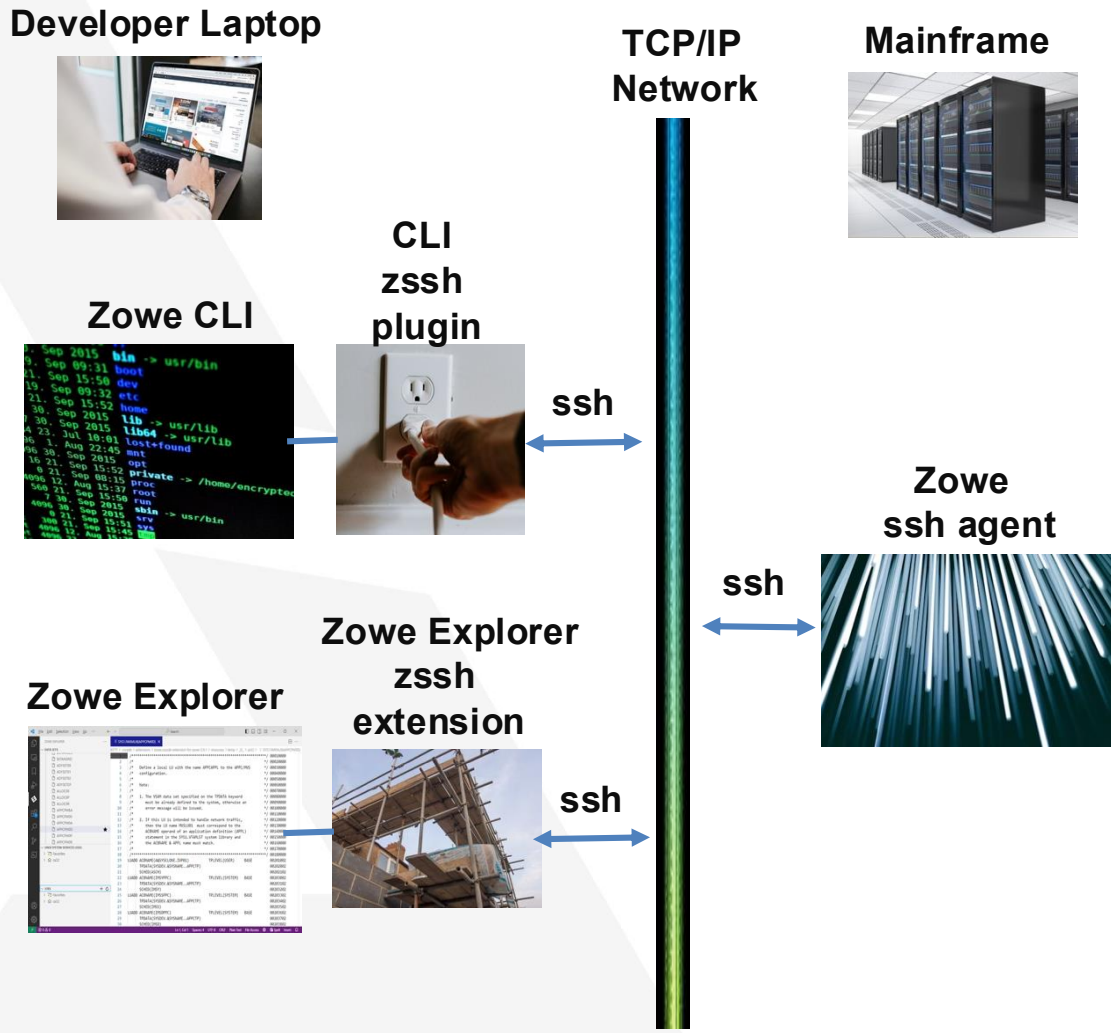
# What's new and what's cooking in our lab

- Recently delivered
  - Search for text in data-sets or PDS members
  - Download PDS members matching a pattern
  - Copy data set without having to pre-allocate target
  - Copy members from a source PDS to existing PDS
  - New option to prompt before overwriting data set
    - --safe-replace
  - New options to control timing of job download
    - --wait-for-active & --wait-for-output
- Work in progress
  - Change your mainframe password from Zowe clients
  - Allow user to control the authentication used for any profile

# SSH AGENT

# Zowe self-deploying “ssh agent” project

- Proof of concept application
  - Provides remote access to z/OS services
- Zowe client app deploys the agent
  - No mainframe installation or started task
  - The agent is provided with the client app
  - Developers upload the agent themselves
  - Useful for developers working on ZD&T and ZVDT systems
- Requires SSH on your mainframe
  - Typically available at most sites
- Requesting consumer validation
  - Basic capabilities to be expanded
  - Pre-release version on a public website soon



# Zowe ssh agent technology

- C++ / Metal-C mainframe binary code
  - Some embedded assembler
  - Communicates with underlying z/OS services
- GO language mainframe binary code
  - Handles the communications between SSH and the C++ code
- TypeScript in the Zowe clients
  - Communicates to SSH
  - Uploads the ssh agent to the mainframe
  - Sends requests to the ssh agent
  - Processes responses from the ssh agent

# Comparison of ssh agent capabilities

https://github.com/zowe/zowe-native-proto/blob/main/doc/apis.md

## API Matrix

### Legend

- ✅ Supported
- 🚧 Partially supported
- ❌ Not supported
- ➖ Not applicable
- *italic* Target for MVP

### Data Sets

| Operation | z/OSMF | FTP | Backend | Middleware | SDK | CLI | VSCE |
|---|---|---|---|---|---|---|---|
| *List data sets* | ✅ | ✅ | 🚧 [1] | ✅ | ✅ | ✅ | ✅ |
| *List data set members* | ✅ | ✅ | 🚧 [1] | ✅ | ✅ | ✅ | ✅ |
| *Read data set/member* | ✅ | ✅ | 🚧 [2] | ✅ | ✅ | ✅ | ✅ |
| *Write data set/member* | ✅ | ✅ | 🚧 [2] | ✅ | ✅ | ❌ | ❌ |
| *Create data set* | ✅ | ✅ | ✅ | ❌ | ❌ | ❌ | ❌ |
| *Delete data set/member* | ✅ | ✅ | ✅ | ❌ | ❌ | ❌ | ❌ |
| Migrate data set | ✅ | ❌ | ❌ | ❌ | ❌ | ❌ | ❌ |
| Recall data set | 🚧 [3] | ❌ | ✅ | ✅ | ✅ | ✅ | ❌ |
| Delete migrated data set | ✅ | ❌ | ❌ | ❌ | ❌ | ❌ | ❌ |
| Rename data set | ✅ | ✅ | ❌ | ❌ | ❌ | ❌ | ❌ |
| Copy data set | ✅ | ❌ | ❌ | ❌ | ❌ | ❌ | ❌ |
| Invoke AMS (VSAM) | ✅ | ❌ | ❌ | ❌ | ❌ | ❌ | ➖ |
| Search data sets | 🚧 [4] | ❌ | ❌ | ❌ | ❌ | ❌ | ❌ |

1. Not all attributes are retrieved
2. Streaming is not supported for large files
3. Does not support some migration utilities like CA Disk
4. Limited options compared to ISPF `srchfor`

# Video: Using Zowe with the ssh agent

# Zowe Secure Shell (ssh)
# Self-deploying Agent

# What did we not have time to show you

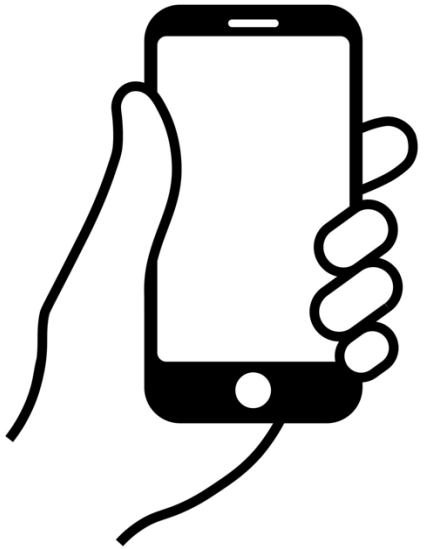
- CLI Deep Dive
  - zowe jobs submit ds “HLQ.JCL(MBR) –jcl-symbols (A=DS1 B=DS2) --wfo --vasc”
  - zowe tso ‘exec “HLQ.REXX(MBR)”’
  - zowe cics get resource CICSPRogram –criteria “PROGRAM=LG*”
- Scaling up
  - Nested and base profiles
  - Daemon. TSO Address Space start -> send -> stop
- Authentication and security
  - Secure Credentials Store
  - X509 client certificates
  - API Mediation Layer for tokenBased one time password (MFA) scenarios

# Your feedback is important!

**Submit a session evaluation** for each session you attend:

www.share.org/evaluation

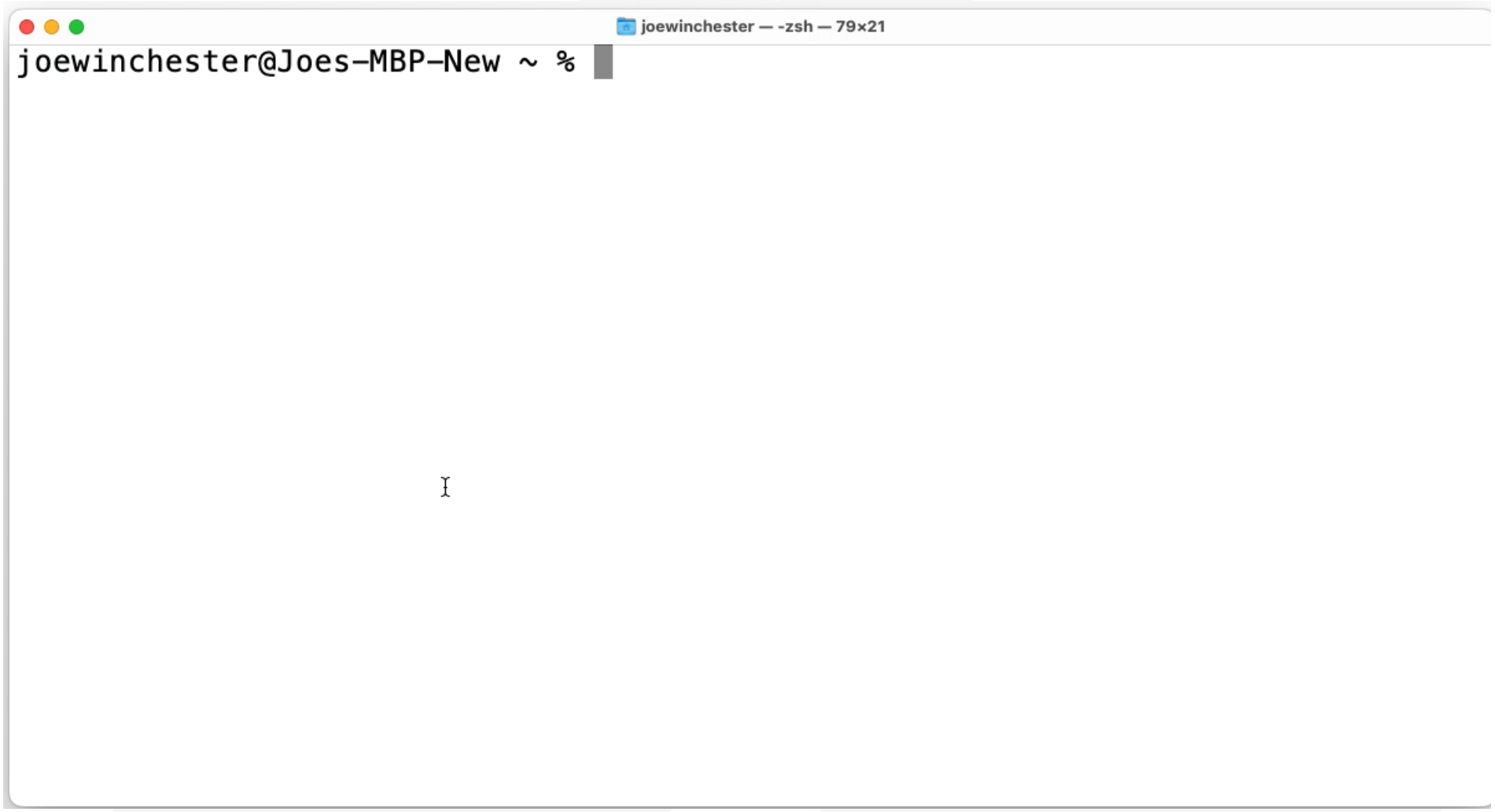
SHARE mobile app

# CLI COMMAND DIVE INTO EXAMPES

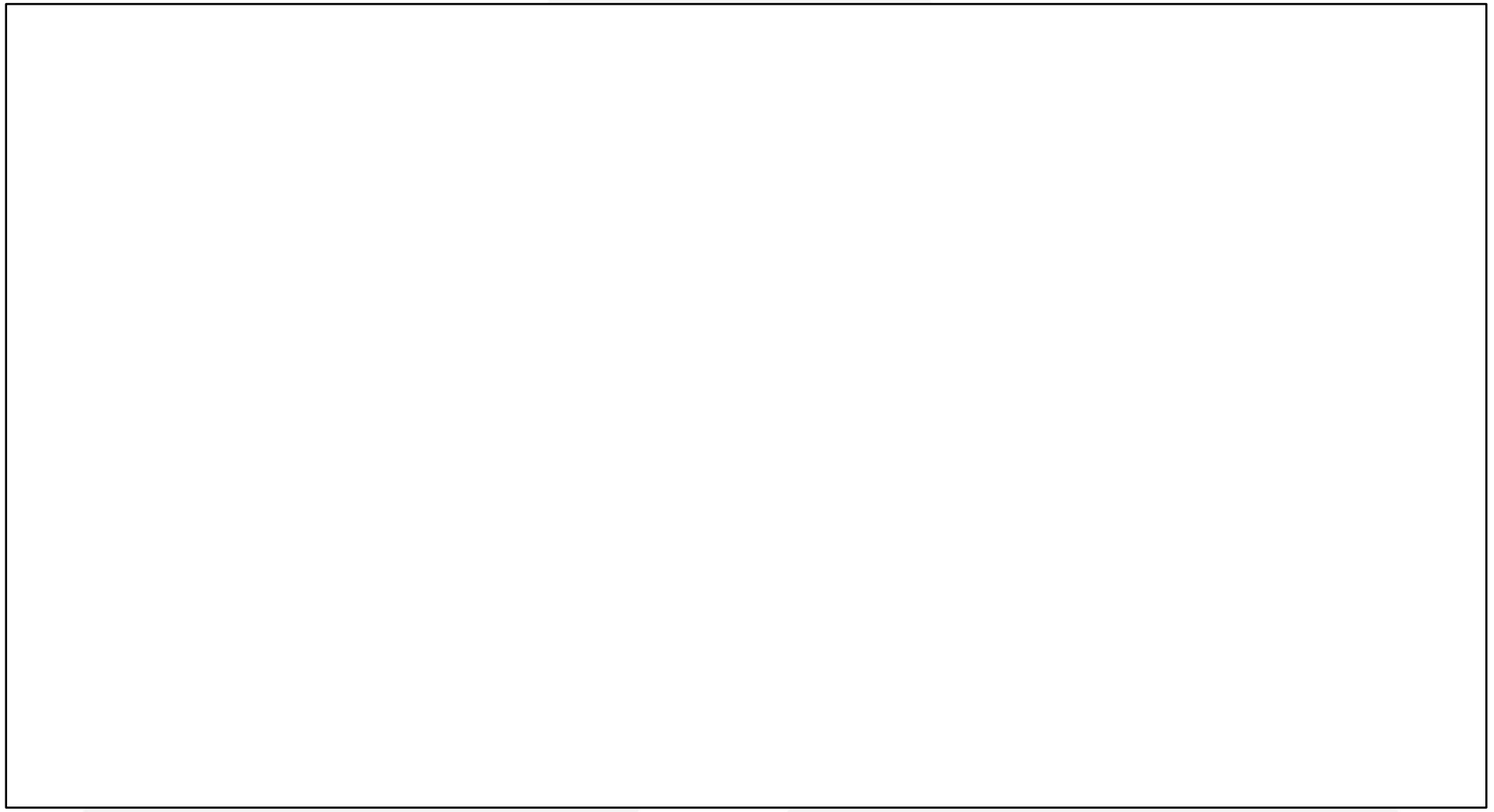
# CLI with Arguments

zowe jobs submit ds ”HLQ.JCL(MBR)” –jcl-symbols “symbol=value”



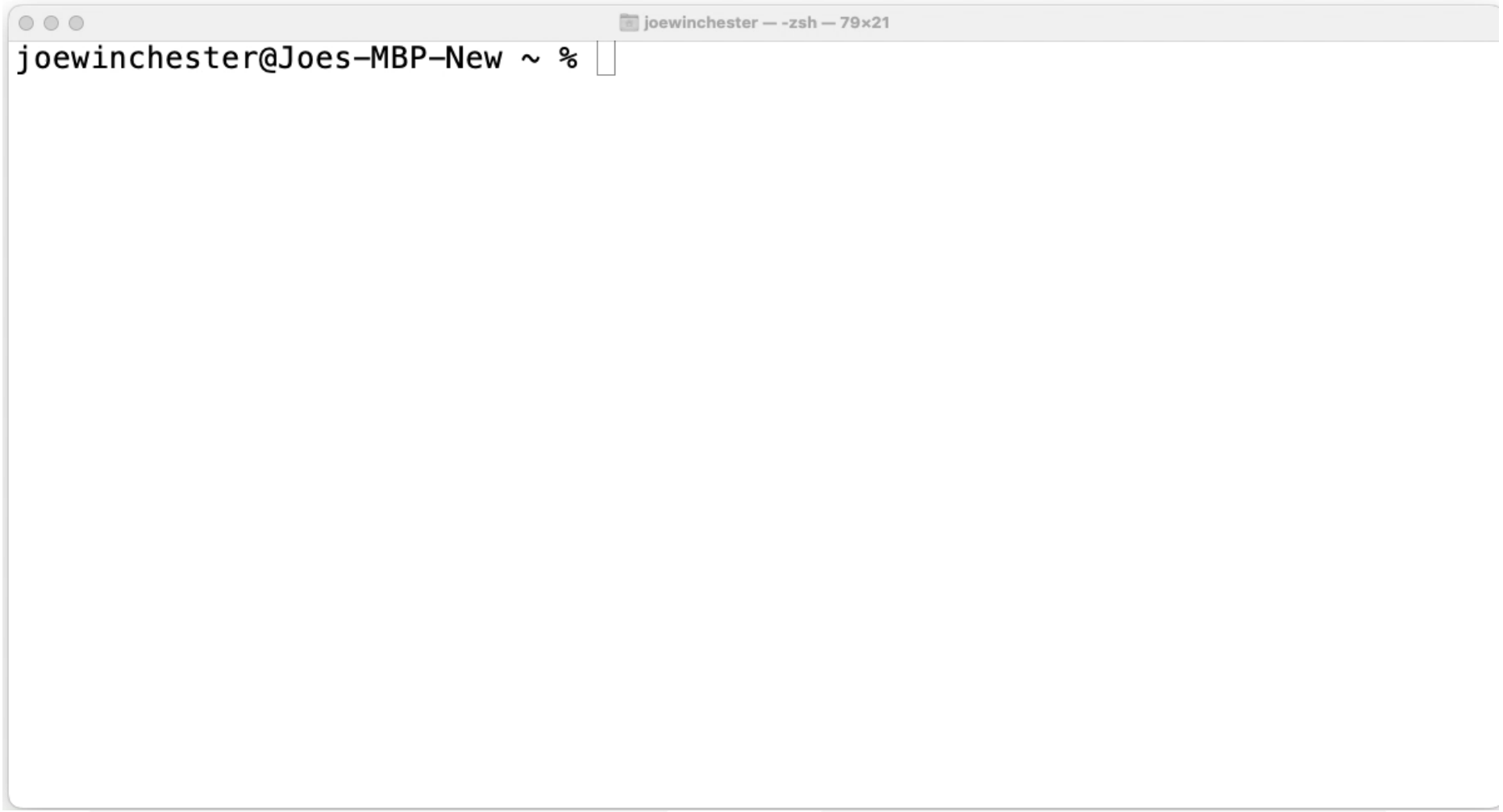
# Zowe TSO = CLI Unlimited

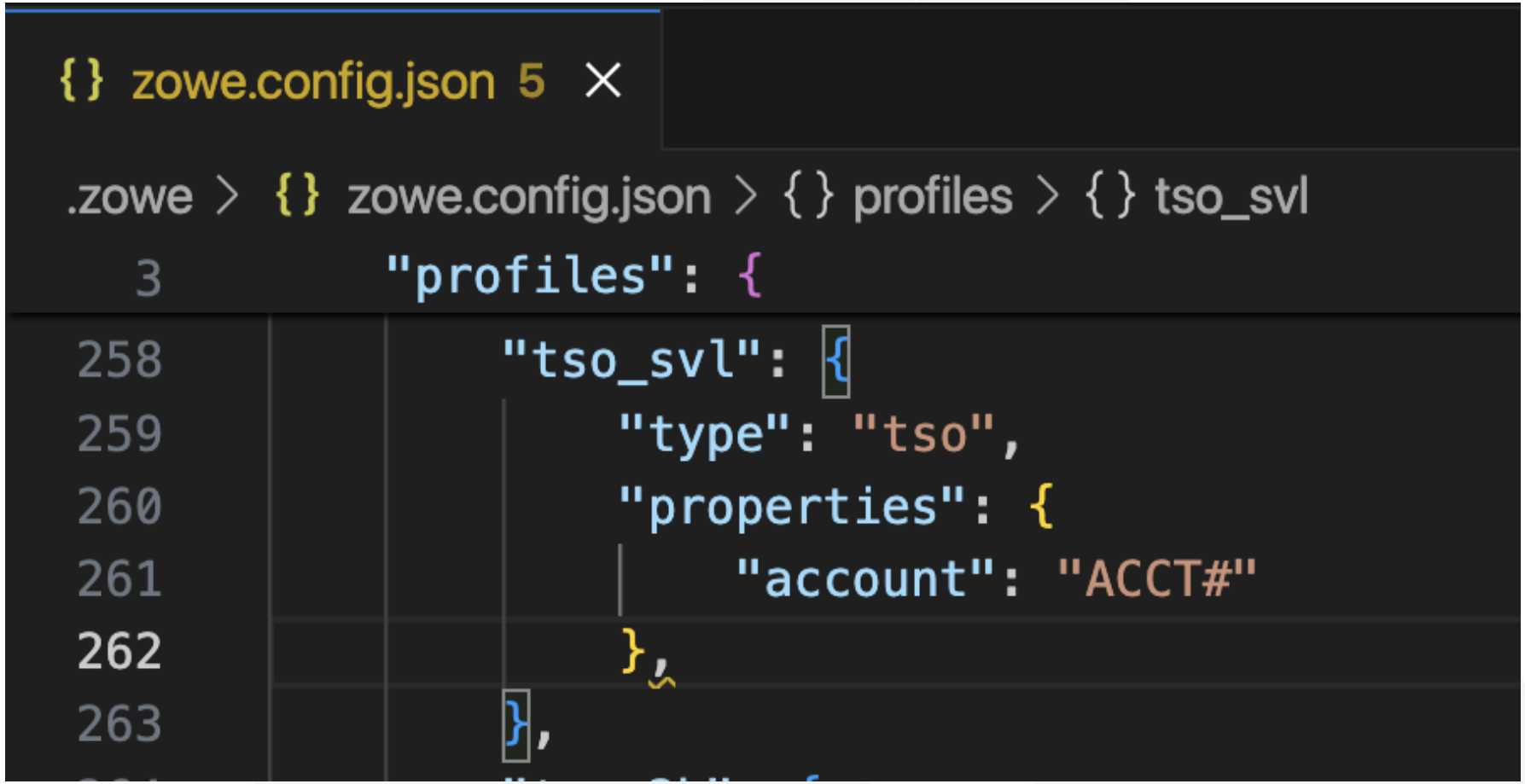
```
zowe tso issue command "TSO COMMAND"
```



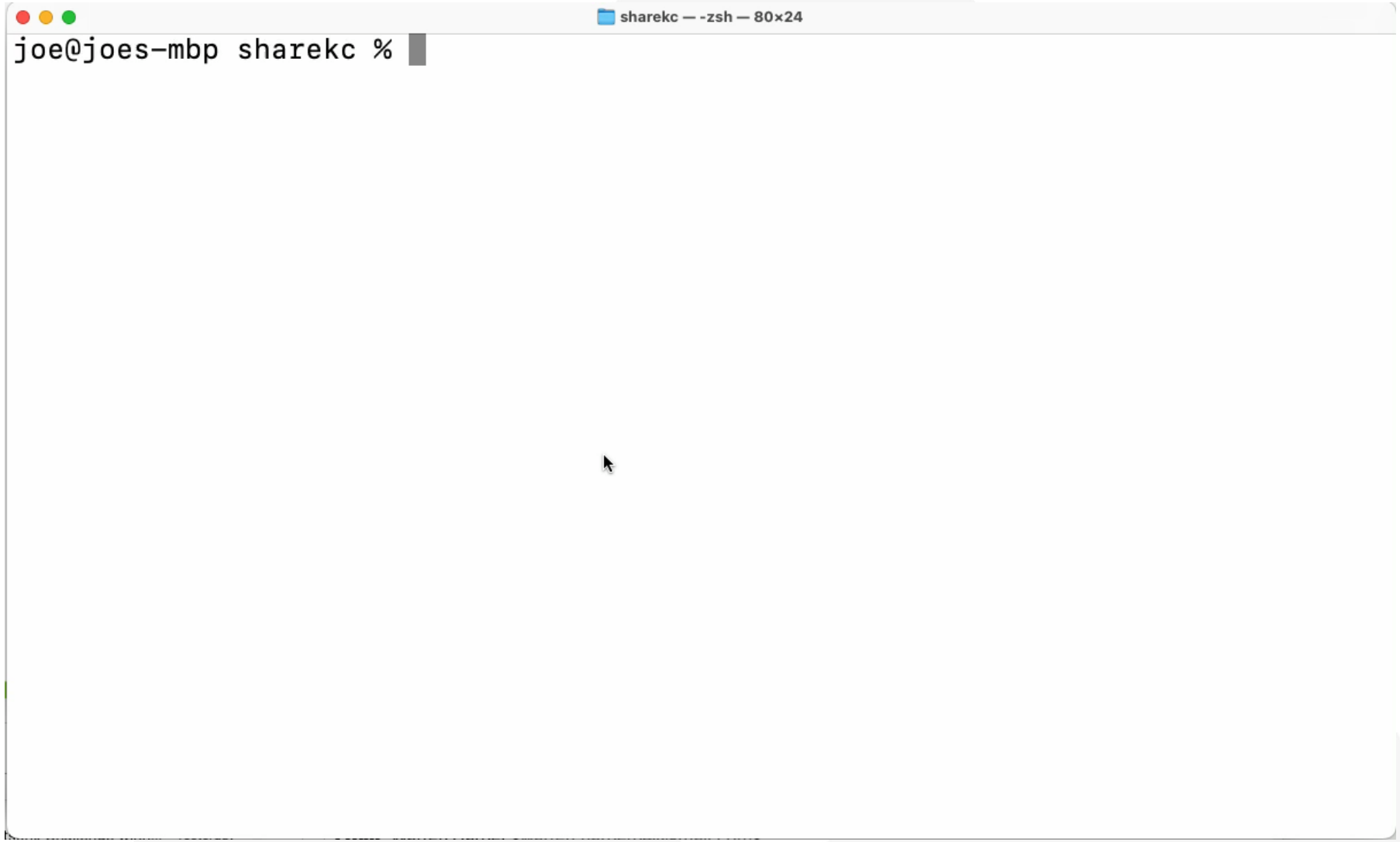
# REXX with Arguments through TSO

```
zowe tso issue command ”exec ‘HLQ.REXX(MBR)’ ‘ARG1’ ‘ARG2’”
```





# SSH Profiles



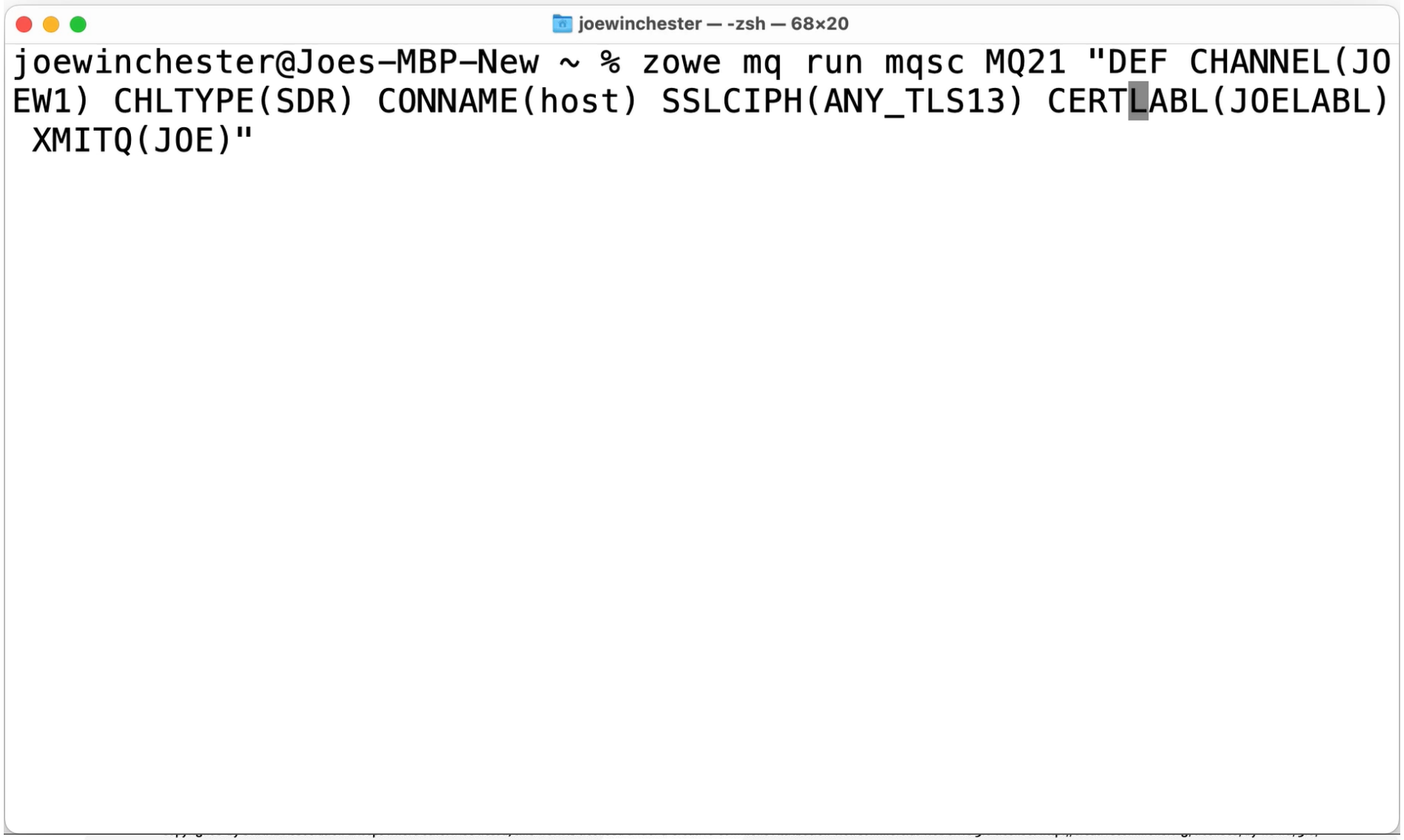
{} zowe.config.json 4 ×

.zowe > {} zowe.config.json > {} profiles > {} ssh_3b > {} properties

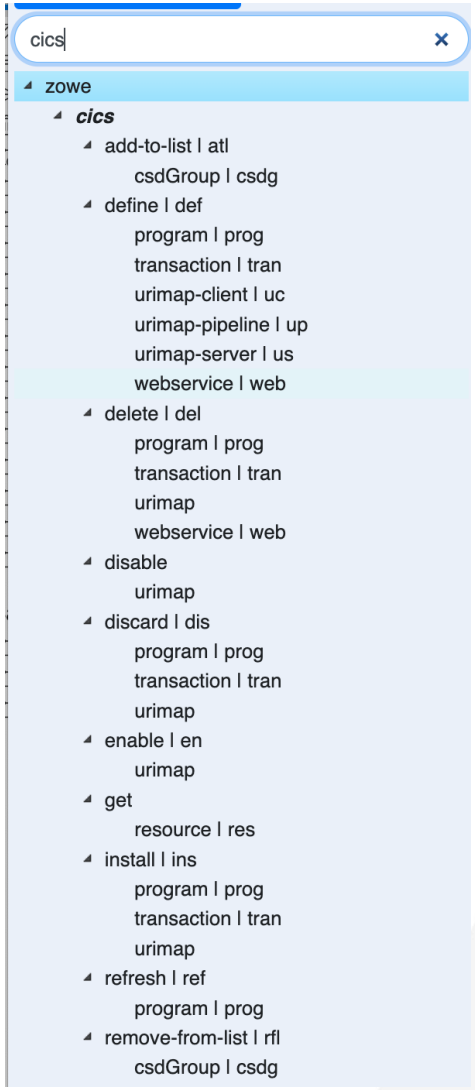
```
  3        "profiles": {
238            "ssh_svl": {
239                "type": "ssh",
240                "properties": {
241                    "host": "zowehost",
242                    "port": 22
243                },
244                "secure": [
245                    "user",
246                    "password"
247                ]
248            },
```

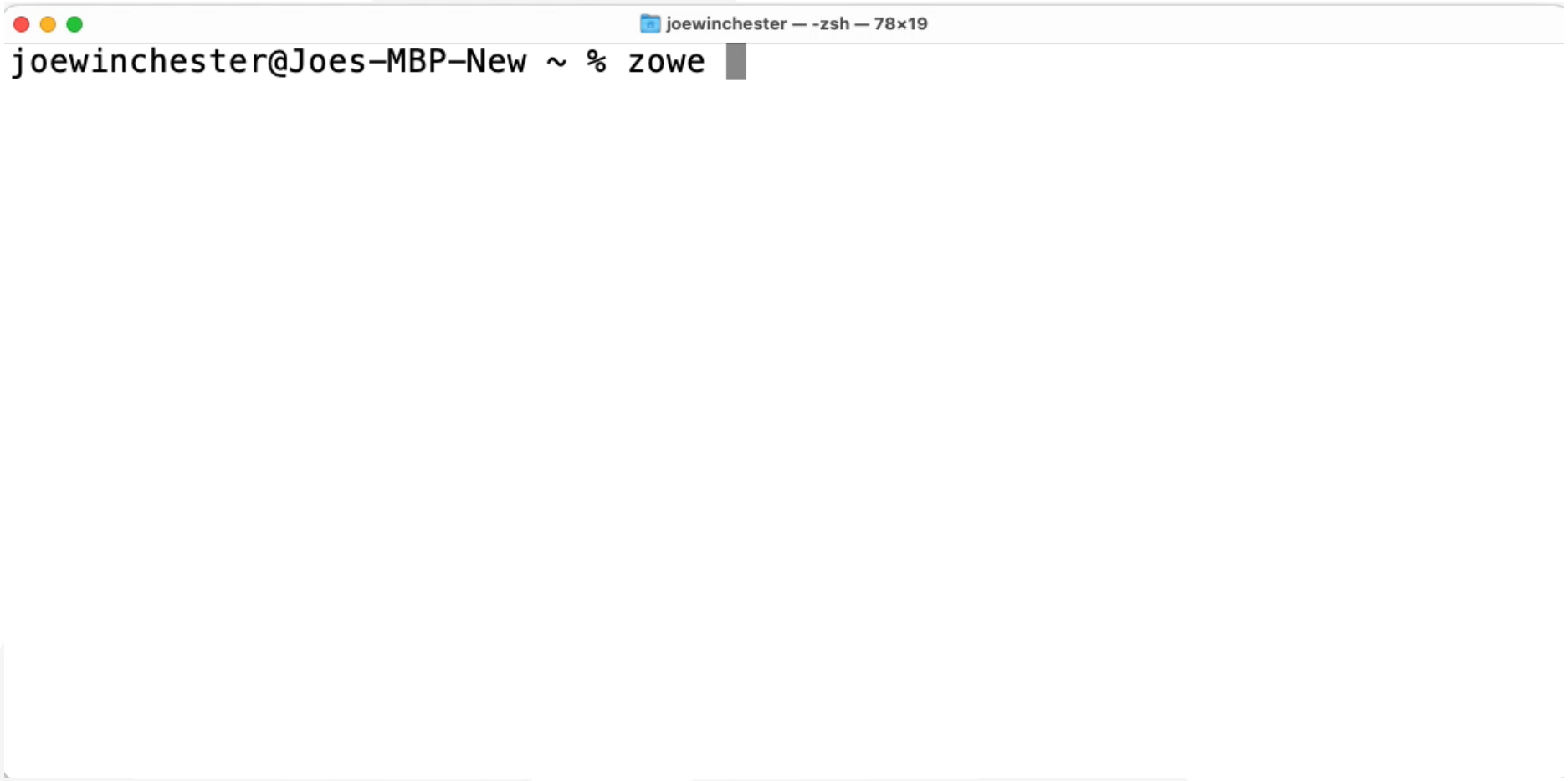
# MQ Channel Commands

zowe mq run mqsc MQ21 ”<MQSC_CMD>”



# CICS me baby one more time




- db2
  - call
    - procedure | proc | sp
  - execute
    - sql
  - export
    - table

**Examples**

- Execute a dummy SQL query:
  - `zowe db2 execute sql --query "SELECT 'Hello World' FROM SYSIBM.SYSDUMMY1"` Copy
- Retrieve the employees table and total number of rows:
  - `zowe db2 execute sql -q "SELECT * FROM SAMPLE.EMP; SELECT COUNT(*) AS TOTAL FROM SAMPLE.EMP"` Copy
- Execute a file with SQL statements:
  - `zowe db2 execute sql --file backup_sample_database.sql` Copy

# NESTED PROFILES

```
{
  "$schema": "./zowe.schema.json",
  "profiles": {
    "tvt_zosmf": {
      "type": "zosmf",
      "properties": {
        "host": "zowehost",
        "port": 443,
        "rejectUnauthorized": false,
        "protocol": "https"
      },
      "secure": [
        "user",
        "password"
      ]
    },
    "tvt_ssh": {
      "type": "ssh",
      "properties": {
        "host": "zowehost",
        "port": 22,
      },
      "secure": [
        "user",
        "password"
      ]
    },
  },
  "defaults": {
    "zosmf": "tvt_zosmf",
    "ssh": "tvt_ssh",
  },
}
```

# Profile information duplication problem

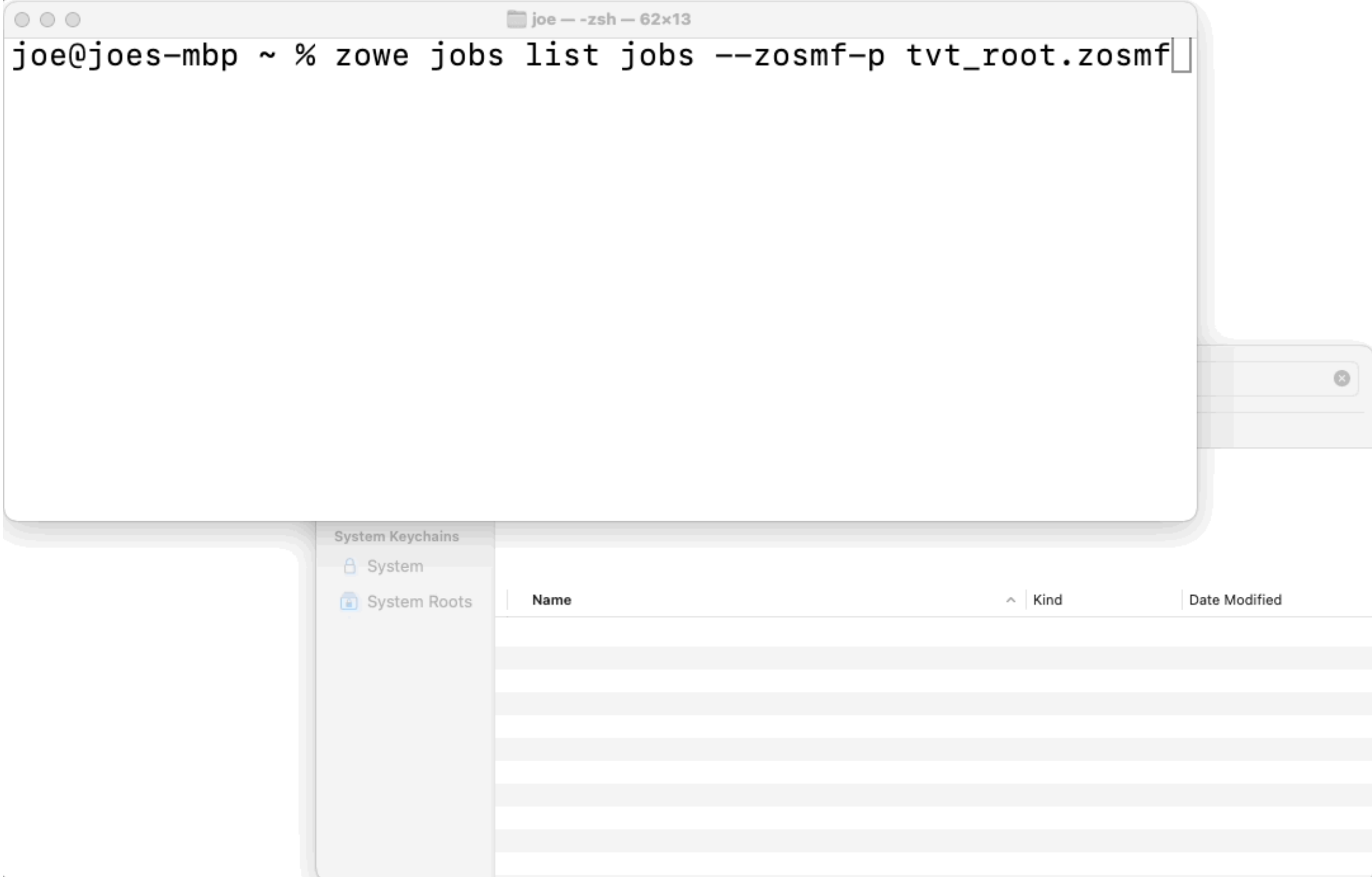
User ID and password are defined twice

```
{
    "$schema": "./zowe.schema.json",
    "profiles": {
        "tvt_base": {
            "type": "base",
            "properties": {
                "host": "zowehost",
                "rejectUnauthorized": false
            },
            "secure": [
                "user",
                "password"
            ]
        },
        "tvt_zosmf": {
            "type": "zosmf",
            "baseProfile": "tvt_base",
            "properties": {
                "protocol": "https",
                "port": 443
            }
        },
        "tvt_ssh": {
            "type": "ssh",
            "baseProfile": "tvt_base",
            "properties": {
                "port": 22
            }
        }
    },
    "defaults": {
        "zosmf": "tvt_zosmf",
        "ssh": "tvt_ssh",
        "base": "tvt_base"
    },
    "autoStore": true
}
```

# Base profile is parent of child profiles

Common shared attributes defined in base and shared across children, e,g

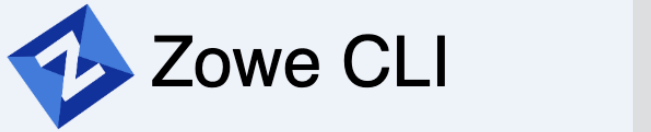
- host
- user
- password
- …


joe — -zsh — 62×13
joe@joes-mbp ~ % zowe jobs list jobs --zosmf-p tvt_root.zosmf
System Keychains
System
System Roots
Name
Kind
Date Modified

```
{
  "$schema": "./zowe.schema.json",
  "profiles": {
   "tvt_root": {
     "properties": {
       "host": "zowehost"
     },
     "secure": [
       "user",
       "password"
     ],
     "profiles": {
       "zosmf": {
         "type": "zosmf",
         "properties": {
           "protocol": "https",
           "rejectUnauthorized": false,
           "port": 443
         }
       },
       "ssh": {
         "type": "ssh",
         "properties": {
           "port": 22
         }
       }
     }
   }
}
```

# Profiles can have child profiles

User ID and password are defined in the parent that has two child profiles




Tree View | Flat View

config

- zowe
  - *config*
    - auto-init
    - convert-profiles | convert
    - edit
    - import
    - init
    - list | ls
    - profiles
    - report-env | re
    - schema
    - secure
    - set
    - update-schemas | us

# zowe › config › secure

prompt for secure configuration properties

### Usage

zowe config secure [options]

### Options

- `--global-config` | `--gc` *(boolean)*
  - Secure properties in global config.

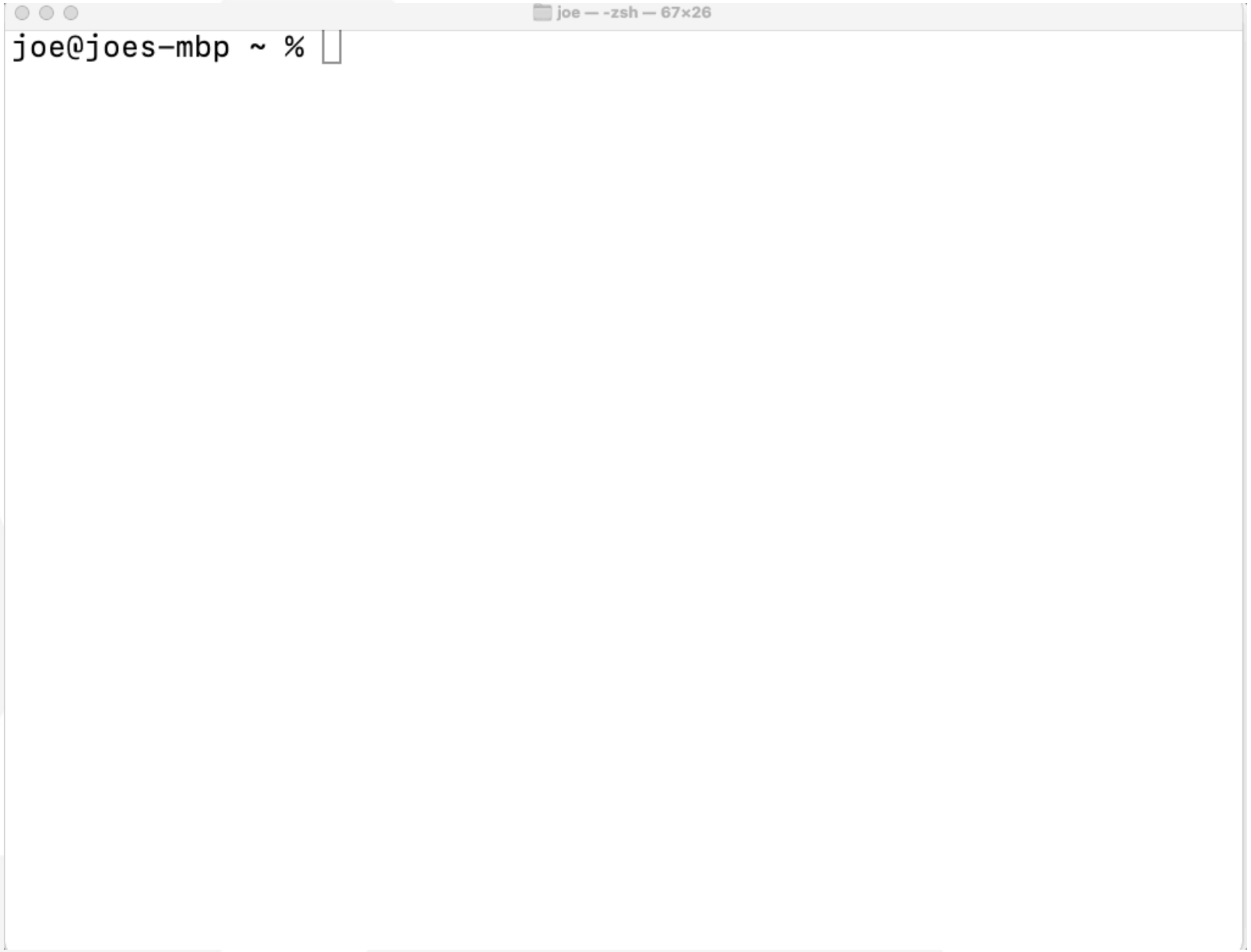
    Default value: false
- `--user-config` | `--uc` *(boolean)*
  - Secure properties in user config.

    Default value: false
- `--prune` | `-p` *(boolean)*
  - Delete properties stored in the vault for team config files that do not exist.

    Default value: false

`zowe config secure`

lists and lets you edit properties



# zowe › config › set

create or update a configuration property

### Usage

zowe config set <property> [value] [options]

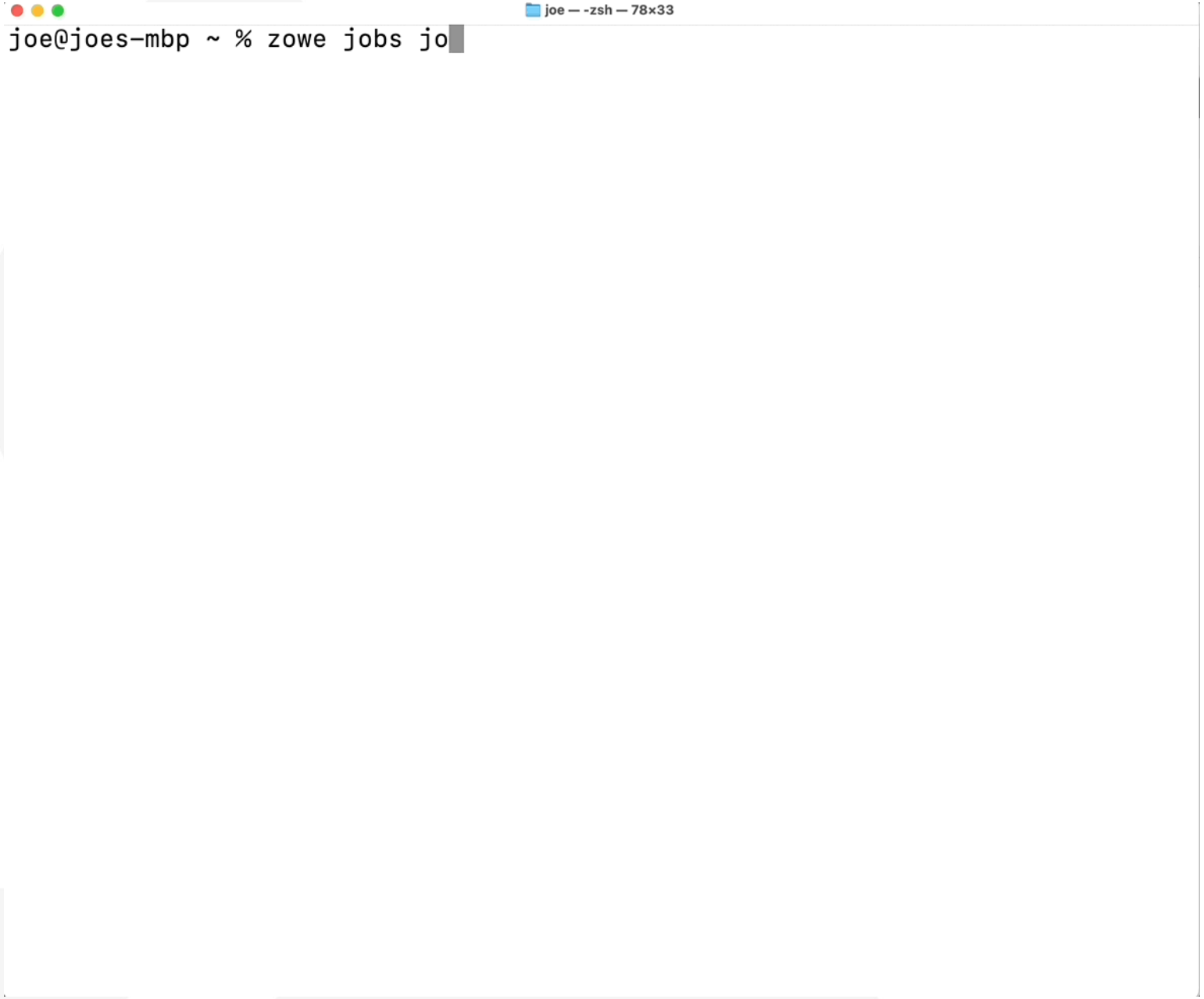
### Positional Arguments

- `property` *(string)*
  - The property to set. You may specify a path using dot notation (e.g. profiles.host1.profiles.service1.properties.setting)
- `value` *(string)*
  - The property value to set. The value may be JSON. Use '--json' to indicate.

- Set the property in global config:
  - `zowe config set "profiles.host1.profiles.service1.properties.setting" "value" --global-config` Copy
- Set the property in user config:
  - `zowe config set "profiles.host1.profiles.service2.properties.setting" "value" --user-config` Copy
- Set property value to JSON:
  - `zowe config set "profiles.host1.profiles.service3.properties.setting" "{"property":"value"}" --json` Copy
- Store the property value:
  - `zowe config set "profiles.host1.profiles.service1.properties.setting" "value" --secure` Copy

```
zowe config set
“key” “value”
```

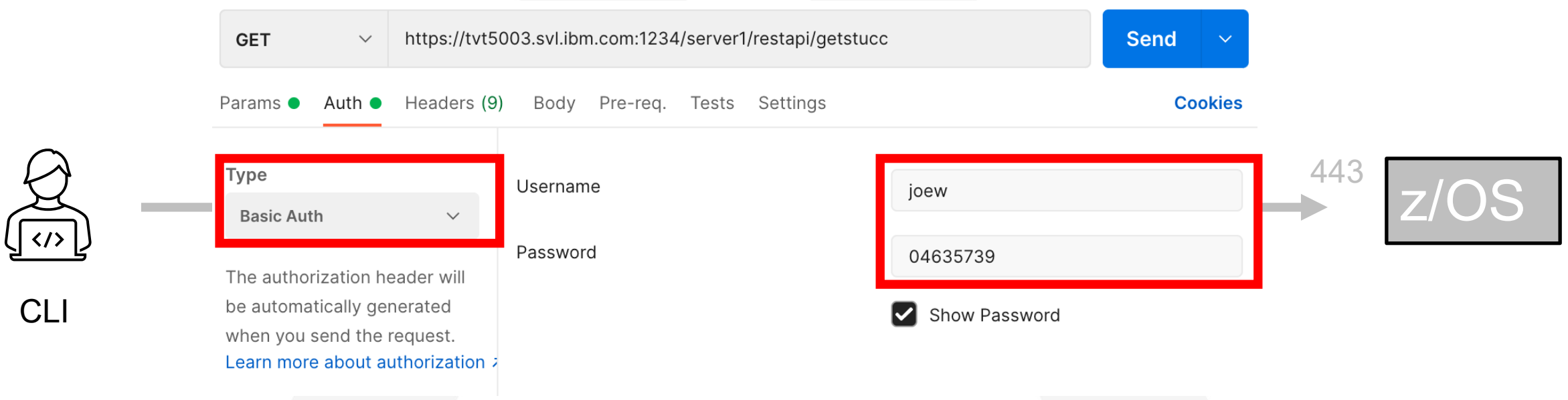
lets you fine grain edit properties

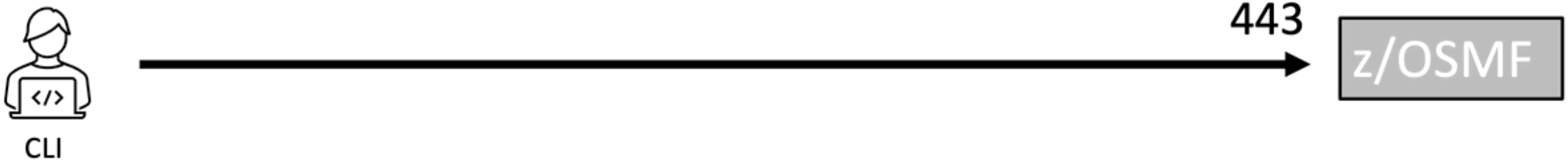


# CLI AUTHENTICATION

# How does z/OS know who I am ?

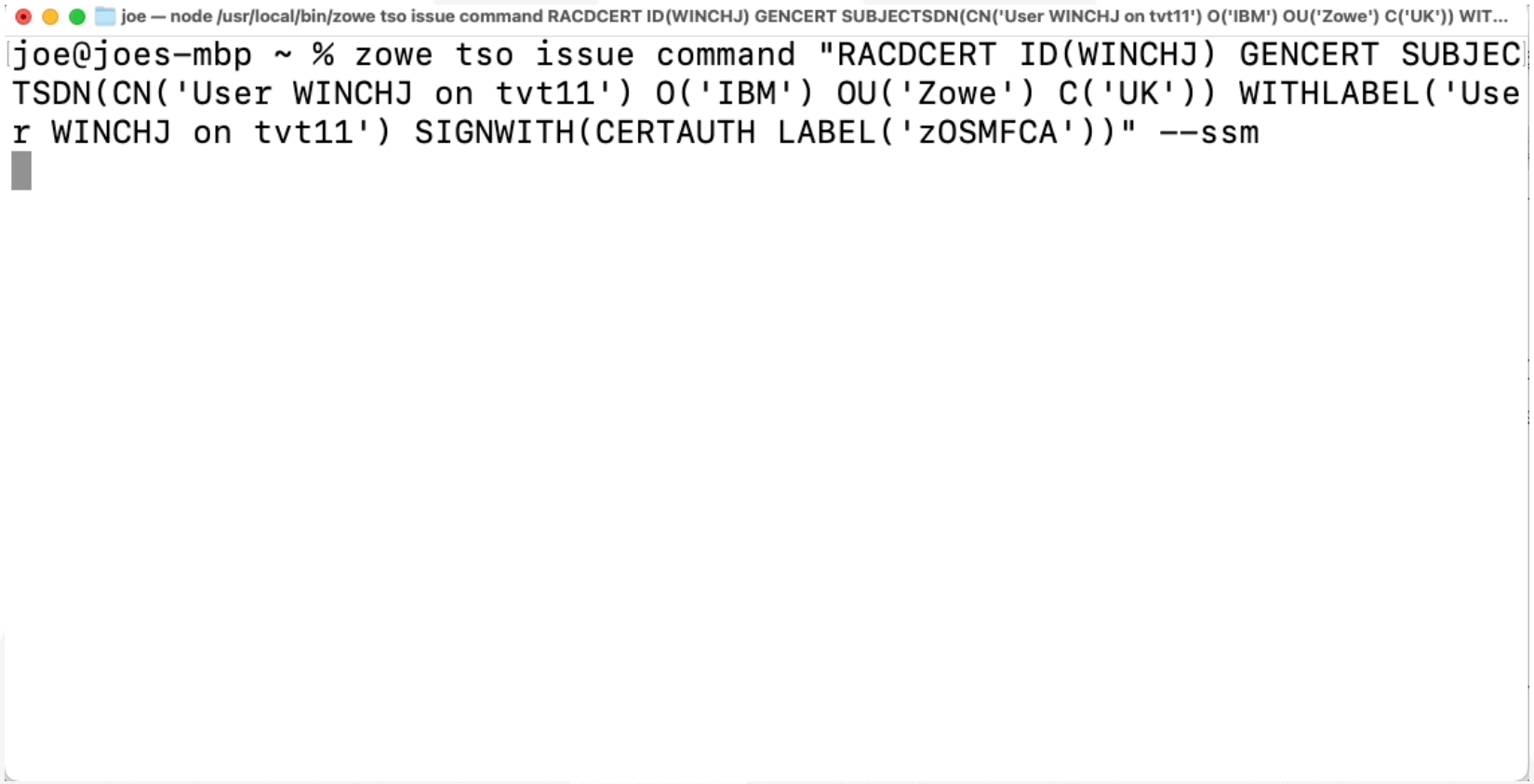
# TSO user ID and password




443
z/OSMF
CLI

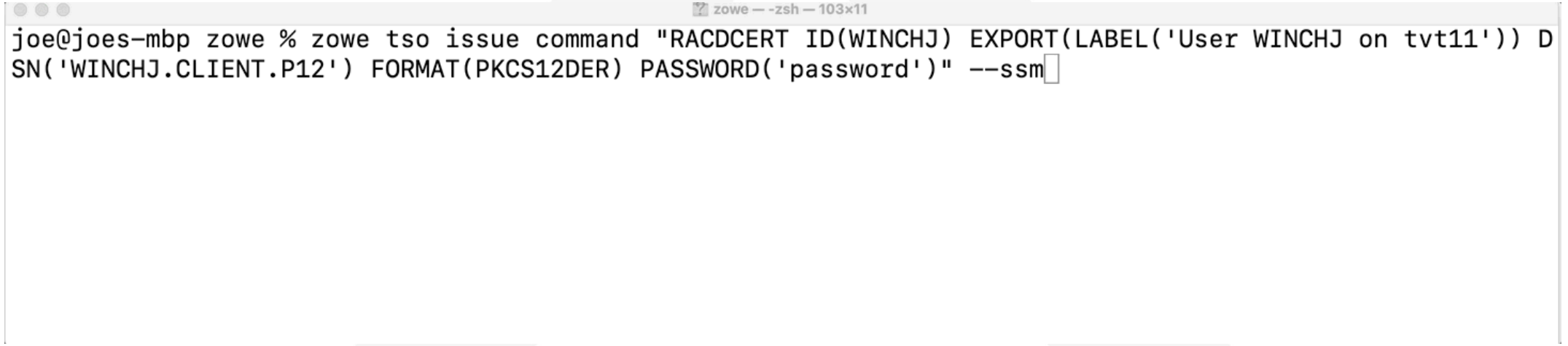
# Step 1: Create a z/OS personal certificate

RACDCERT ID(WINCHJ) GENCERT SUBJECTSDN(CN('User WINCHJ on tvt11') O('IBM') OU('Zowe') C('UK')) WITHLABEL('User WINCHJ on tvt11') SIGNWITH(CERTAUTH LABEL('**zOSMFCA**')



# Step 2: Export the cert to a sequential DS

RACDCERT ID(WINCHJ) EXPORT(LABEL('User WINCHJ on tvt11')) DSN('WINCHJ.CLIENT.P12') FORMAT(**PKCS12DER**) PASSWORD('password')




# Step 3: Download the certificate to your PC

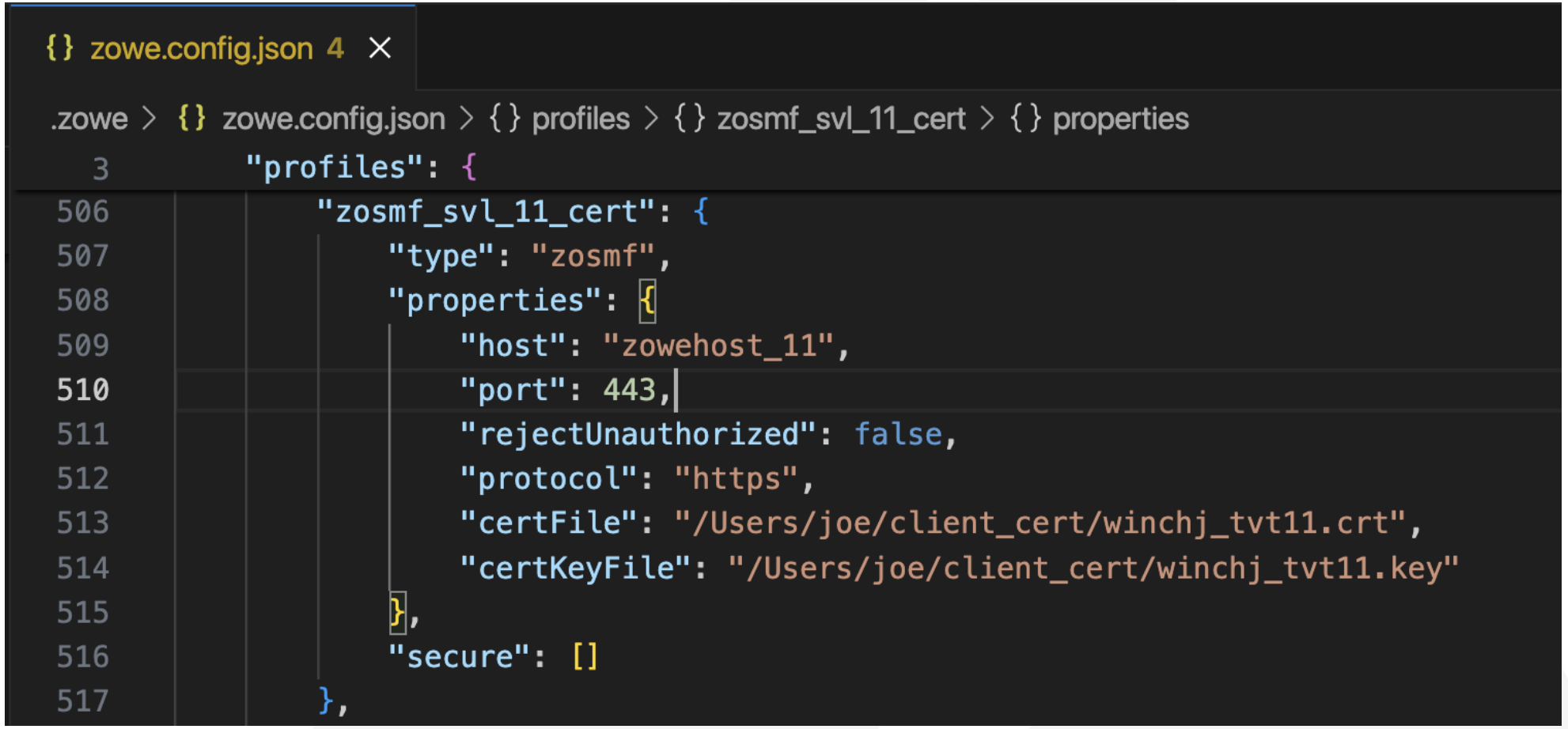
zowe files download ds "WINCHJ.CLIENT.P12" -b -f winchj_tvt11.12

# Step 3: Crack the cert apart into its private .key and public .crt

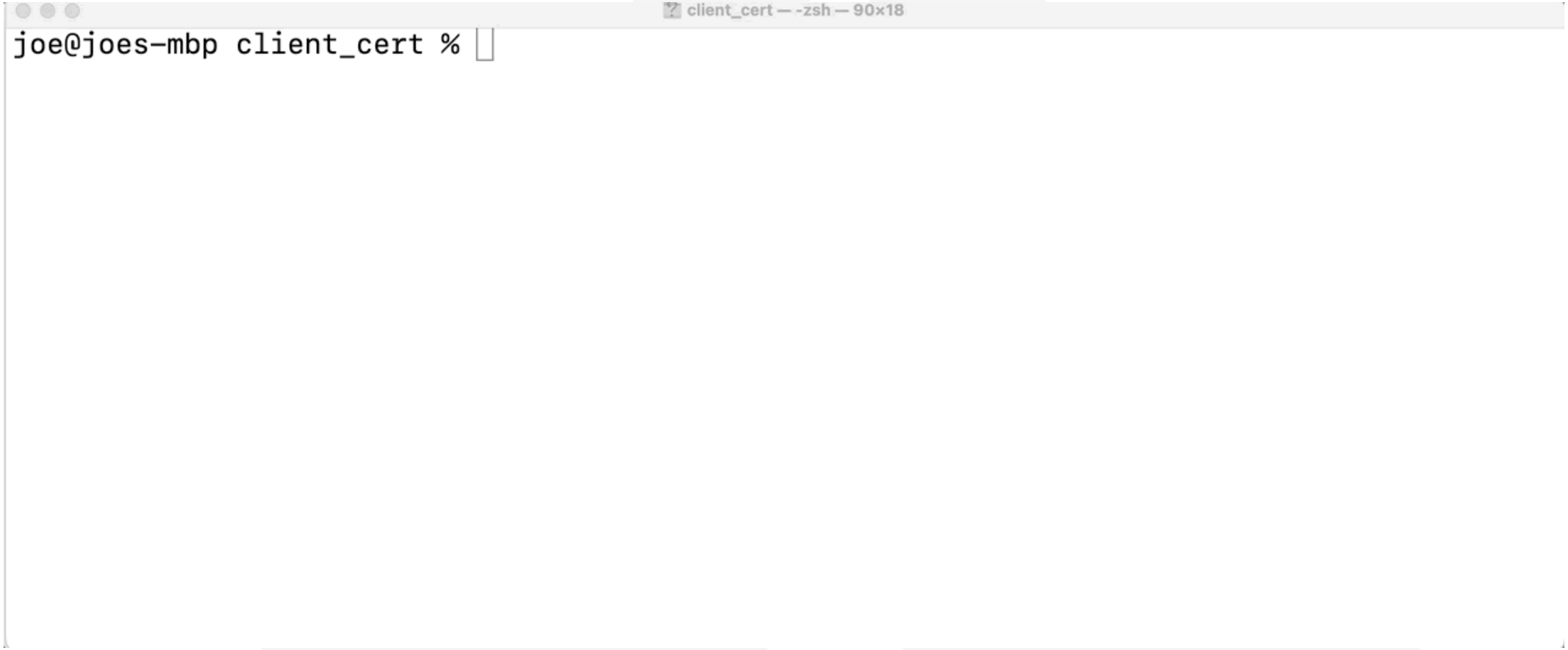
openssl pkcs12 -in winchj_tvt11.p12 -nocerts -**nodes** -out winchj_tvt11.key


openssl pkcs12 -in winchj_tvt11.p12 -clcerts -nokeys -out winchj_tvt11.crt

# Point to .crt and .key and remove user and password

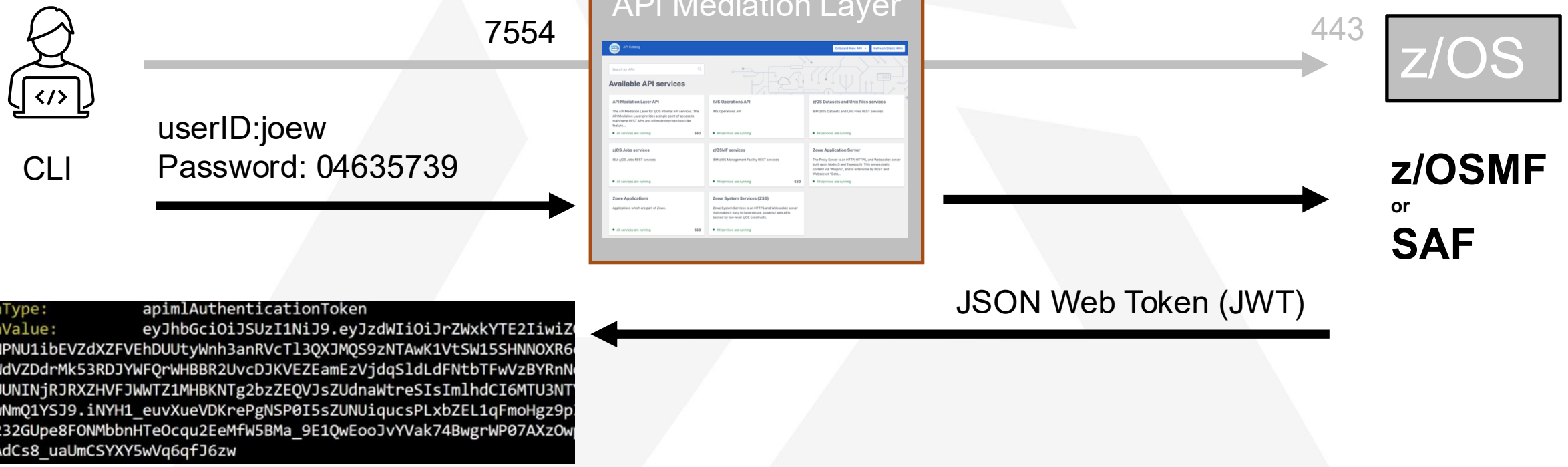


# `--show-inputs-only` is your CLI friend

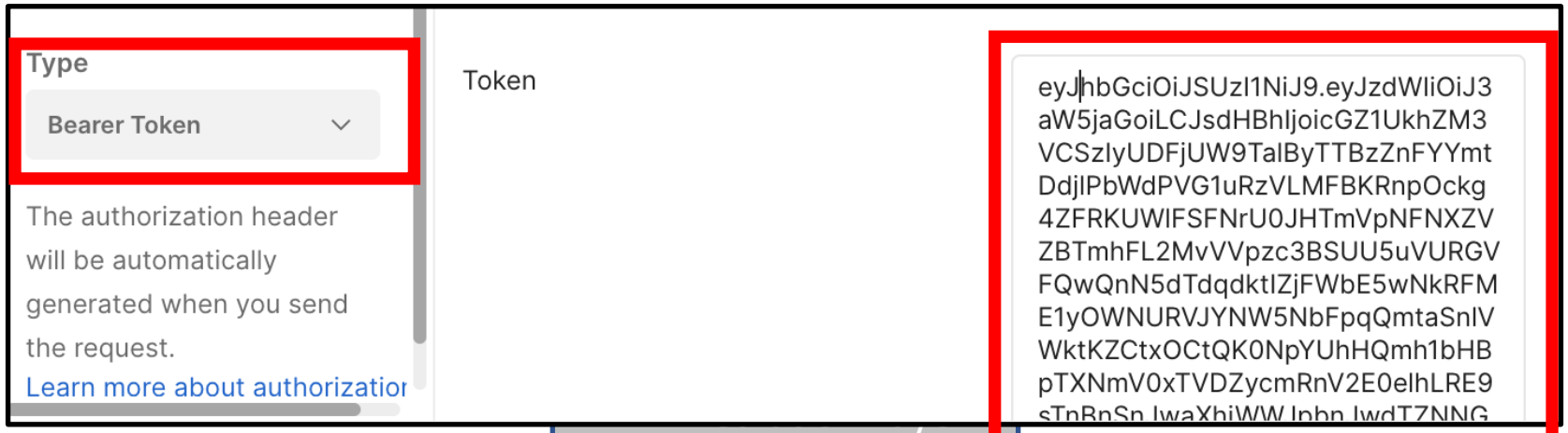


# APIML AND BASE PROFILES

# Exchange password for a token

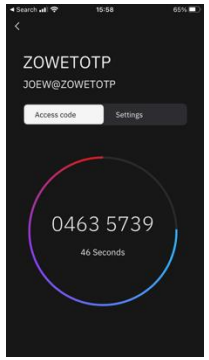


# Use Token on Subsequent requests

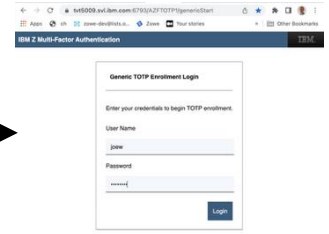


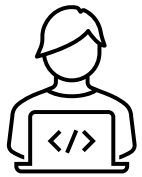

```
    tokenType:          apimlAuthenticationToken
    tokenValue:         eyJhbGciOiJSUzI1NiJ9.eyJzdWIiOiJrZWxkYTE2IiwiZ
RESjk2VzNPNU1ibEVZdXZFVEhDUUtyWnh3anRVcTl3QXJMQS9zNTAwK1VtSW15SHNNOXR6
nh6Vmd5UWdVZDdrMk53RDJYWFQrWHBBR2UvcDJKVEZEamEzVjdqSldLdFNtbTFwVzBYRnN
dDJpK0FTUUNINjRJRXZHVFJWWTZ1MHBKNTg2bzZEQVJsZUdnaWtreSIsImlhdCI6MTU3NT
iMmJlMWIwNmQ1YSJ9.iNYH1_euvXueVDKrePgNSP0I5sZUNUiqucsPLxbZEL1qFmoHgz9p
4HlhnPdB232GUpe8FONMbbnHTeOcqu2EeMfW5BMa_9E1QwEooJvYVak74BwgrWP07AXzOw
f7MGq2bSAdCs8_uaUmCSYXY5wVq6qfJ6zw
```

# MFA Flow Alongside Token based SSO




Out of Band

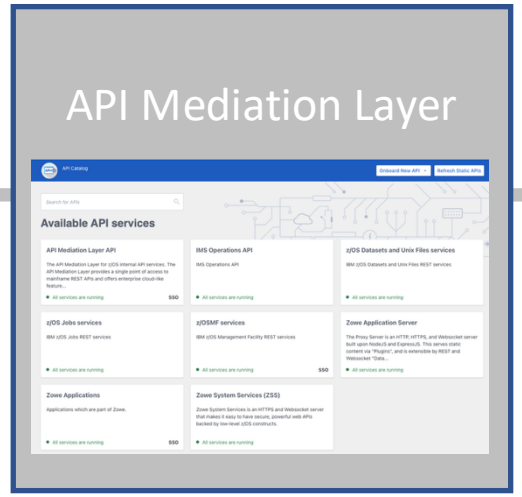


CLI

7554

userID:joew
Password: 04635739




443

z/OS

**z/OSMF**
or
**SAF**

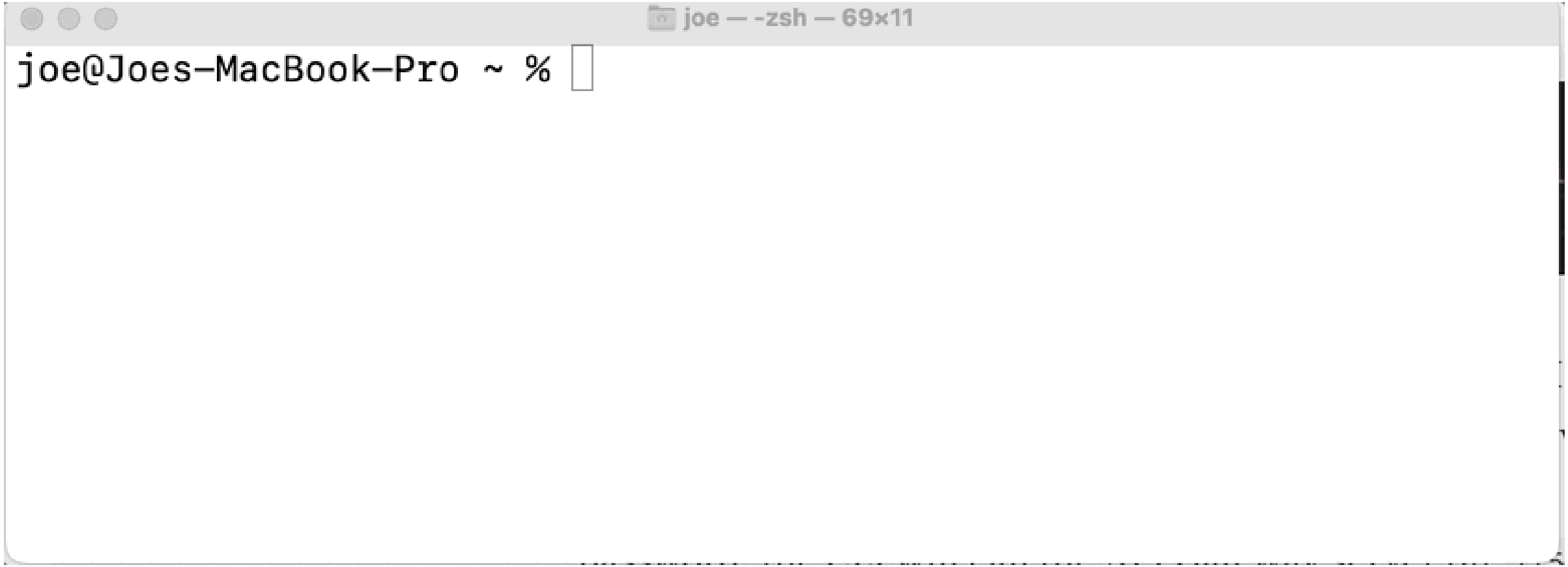
JSON Web Token (JWT)

```
tokenType:          apimlAuthenticationToken
tokenValue:         eyJhbGciOiJSUzI1NiJ9.eyJzdWIiOiJrZWxkYTE2IiwiZ
RESjk2VzNPNU1ibEVZdXZFVEhDUUtyWnh3anRVcTl3QXJMQS9zNTAwK1VtSW15SHNNOXR6
nh6Vmd5UWdVZDdrMk53RDJYWFQrWHBBR2UvcDJKVEZEamEzVjdqSldLdFNtbTFwVzBYRnN
dDJpK0FTUUNINjRJRXZHVFJWWTZ1MHBKNTg2bzZEQVJsZUdnaWtreSIsImlhdCI6MTU3NT
iMmJlMWIwNmQ1YSJ9.iNYH1_euvXueVDKrePgNSP0I5sZUNUiqucsPLxbZEL1qFmoHgz9p
4HlhnPdB232GUpe8FONMbbnHTeOcqu2EeMfW5BMa_9E1QwEooJvYVak74BwgrWP07AXzOw
f7MGq2bSAdCs8_uaUmCSYXY5wVq6qfJ6zw
```

# APIML endpoint is base profile

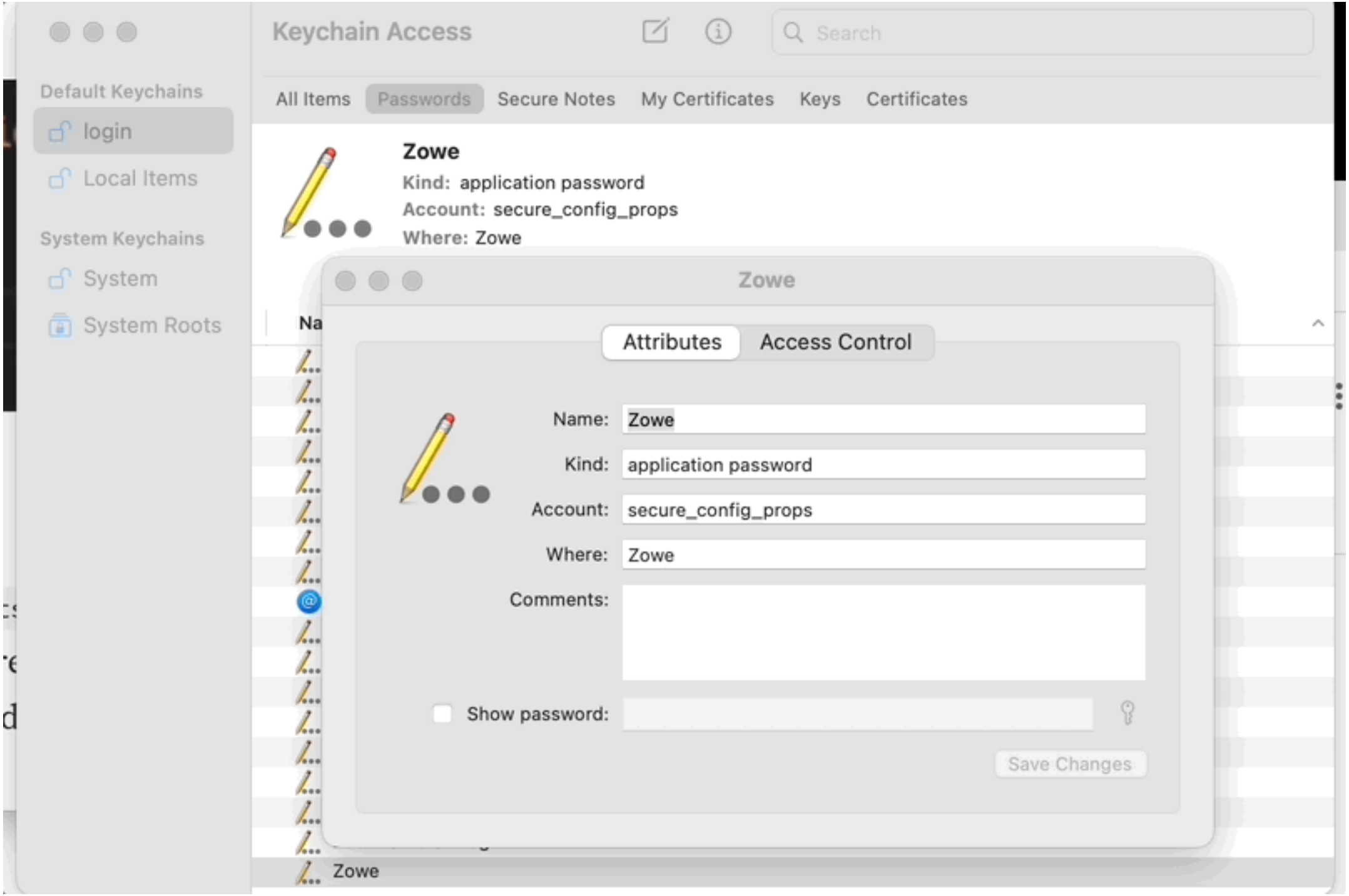
```
400         "zoweapiml": {
401             "type": "base",
402             "properties": {
403                 "host": "zowehost",
404                 "port": 27554,
405                 "rejectUnauthorized": true,
406                 "tokenType": "apimlAuthenticationToken"
407             },
408         }
409     },
```

# Zowe auth login apiml to get token



# tokenValue secured in SCS

```
400         "zoweapiml": {
401             "type": "base",
402             "properties": {
403                 "host": "zowehost",
404                 "port": 27554,
405                 "rejectUnauthorized": true,
406                 "tokenType": "apimlAuthenticationToken"
407             },
408             "secure": [
409                 "tokenValue"
410             ]
```


Keychain Access
Search
Default Keychains
login
Local Items
System Keychains
System
System Roots
All Items
Passwords
Secure Notes
My Certificates
Keys
Certificates
Zowe
Kind: application password
Account: secure_config_props
Where: Zowe
Zowe
Attributes
Access Control
Name: Zowe
Kind: application password
Account: secure_config_props
Where: Zowe
Comments:
Show password:
Save Changes
Zowe

# basePath is /dialing code for the southbound endpoint

```
393         "zosmf_svl": {
394             "type": "zosmf",
395             "properties": {
396                 "host": "tvt5003.svl.ibm.com",
397                 "port": 443,
398                 "rejectUnauthorized": false,
399                 "protocol": "https"
400             },
401             "secure": [
402                 "user",
403                 "password"
404             ]
405         },
```

```
394         "zosmf_via_apiml": {
395             "type": "zosmf",
396             "baseProfile": "zoweapiml",
397             "properties": {
398                 "basePath": "/ibmzosmf/api/v1"
399             }
400         },
```

https://zowehost:27554/ibmzosmf/api/v1/zosmf/restjobs/jobs?prefix=IZU*&owner=*

Type

Bearer ...

Token

The authorization header will be automatically generated when you send the request. Learn more about authorization ↗

eyJhbGciOiJSUzI1NiJ9.eyJzdWIiOiJ3aW5jaG
oiLCJpYXQiOjE2ODYzMDQ4NDMsImV4cCI6
MTY4NjMzMzY0MywiaXNzIjoiQVBJTUwiLCJ
qdGkiOiIzMTNlY2M3NS0zNDA4LTRmYjUtOG
QwYS04NDRkNzI3OGZhNjkiLCJsdHBhIjoiMV
hEbGdaanRSVUpRUmtiQVlkUlBwOGM2VlNG
N3VyVHBkUXV3NnNkZEgzektGaXFUNmk5V
2RWZ1BVVkNNWkRrOEliVVNBa2lwSXVmS05

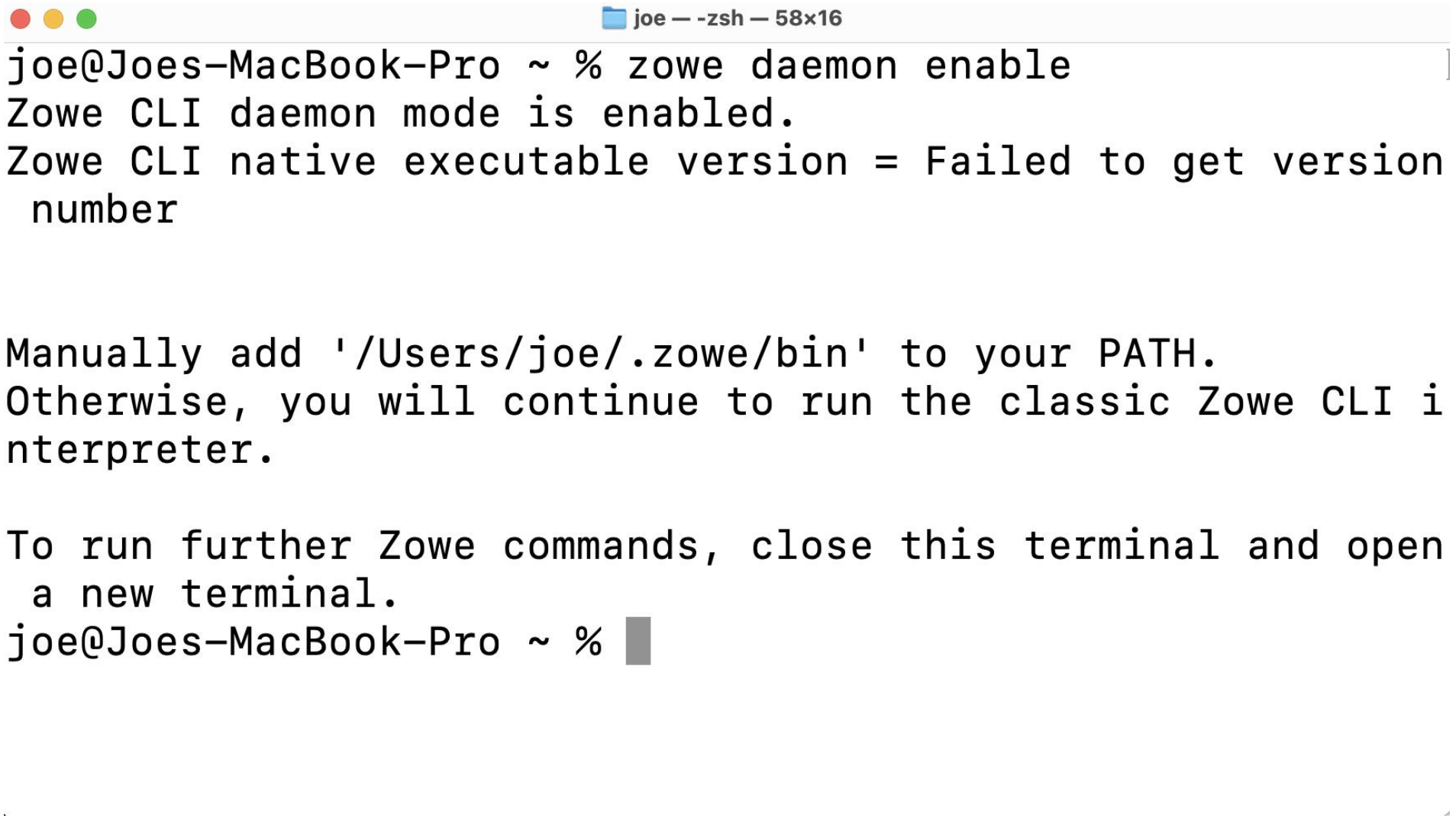
https://zowehost:443/zosmf/restjobs/jobs?prefix=IZU*&owner=*

Pretty Raw Preview Visualize JSON


```
[
    {
        "owner": "IZUSVR",
        "phase": 14,
        "subsystem": "JES2",
        "phase-name": "Job is actively executing",
        "job-correlator": "S0008622TVT5011.DD405CFB.......:",
        "type": "STC",
        "url": "https:/ XXXXXXXXXXXXXXX /zosmf/restjobs/jobs/S0008622TVT5011.DD405CFB.......%3A",
        "jobid": "STC08622",
        "class": "STC",
        "files-url": "https:/ XXXXXXXXXXXXXXX /zosmf/restjobs/jobs/S0008622TVT5011.DD405CFB.......
            %3A/files",
        "jobname": "IZUSVR1",
```

# SQUEEZING THE CLI

# Embrace your inner daemon



# Use Zowe TSO Start and Send for Stateful client server


DevOps



For zowe ssh issue command <CMD> consider sending a script and executing remotely to save latency

# Your feedback is important!

**Submit a session evaluation** for each session you attend:

www.share.org/evaluation

SHARE mobile app