The Mobile Shell: mosh

```
# Forward TCP port 8022 to port 22 on internal
# host0 for initial mosh-server startup
# Forward UDP port 60000/60001 to internal
# host0 after startup

mosh --ssh="ssh -p 8022" \
  --server=/usr/bin/mosh-server -p 60000 \
  username@public_ip_address
```

mosh host1

```
# Forward TCP port 8023 to port 22 on internal
# host1 for initial mosh-server startup
# Forward UDP port 60002/60003 to internal
# host1 after startup

mosh --ssh="ssh -p 8023" \
  --server=/usr/bin/mosh-server -p 60002 \
  username@public_ip_address
```

Inventory file

```
# Debian on Raspberry Pi
cm5
pi5

# FreeBSD on x86_64
freebsd

# Gentoo on i686 and x86_64
dell-i686
x86_64-x86-mhaswell-lenovo
```

Append to a file

```
- name: Edit a single line in a file
  ansible.builtin.lineinfile:
    path: /etc/hosts
    line: "10.254.254.254      testmachine"
    state: present
```

Edit a line

```
- name: Verify sshd port 22 is un-commented
  ansible.builtin.lineinfile:
    path: /etc/ssh/sshd_config
    line: 'Port 22'
    regexp: '^#Port 22'
    state: present
```

Insert a line after an existing line

```
- name: Add a port 8022 entry
  ansible.builtin.lineinfile:
    path: /etc/ssh/sshd_config
    line: 'Port 8022'
    insertafter: 'Port 22\n'
    state: present
```

Create a user

```
- name: Create lugtest user
  ansible.builtin.user:
    name: lugtest
    state: present
    create_home: false
```

Non-Shell Command

```
- name: Run a command without using a shell
  ansible.builtin.command:
    cmd: /usr/bin/uptime
  register: cmd_output

- name: Print command output
  ansible.builtin.debug:
    var: cmd_output.stdout
```

Full Shell Command

```
- name: Run a command using a shell
  ansible.builtin.shell:
    cmd: ps -aux | grep mosh | grep -v grep
  register: cmd_output

- name: Print command output
  ansible.builtin.debug:
    var: cmd_output.stdout
```

Install Software

```
- name: Print the OS
  ansible.builtin.debug:
    msg: "{{ ansible_facts['distribution'] }}"

- name: Install software
  ansible.builtin.package:
    name: nginx
    state: present
```

Run Services

```
- name: Enable and start a service
  ansible.builtin.service:
    name: nginx
    state: started
    enabled: true
```

Status Services

```
  - name: Status all services for facts
    ansible.builtin.service_facts:

  - name: Print nginx status
    ansible.builtin.debug:
      msg: "nginx state is
{{ ansible_facts.services['nginx.service'].state }}"
    when: ansible_facts.services['nginx.service'] is defined
```

Stop Service and Delete Software

```
- name: Disable and stop a service
  ansible.builtin.service:
    name: nginx
    state: stopped
    enabled: false

- name: Remove software
  ansible.builtin.package:
    name: nginx
    state: absent
```

Create Directory and File

```
- name: Ensure directory exists
  ansible.builtin.file:
    path: /etc/testdir
    state: directory
    mode: '0755'

- name: Create new file
  ansible.builtin.copy:
    dest: /etc/testdir/testfile
    content: "hello, world\n"
    mode: '0755'
    owner: root
    backup: true
```

Use Loops

```
- name: Run useful commands
  ansible.builtin.command:
    cmd: "{{ item }}"
  loop:
    - /bin/hostname
    - /usr/bin/uptime
    - /usr/bin/who
  register: cmd_result

- name: Print command results
  ansible.builtin.debug:
    msg: "{{ item.stdout_lines }}"
  loop: "{{ cmd_result.results }}"
```

Do block/rescue/always

```
- name: Do an Ansible block
  block:
    - name: Flip a coin
      ansible.builtin.set_fact:
        coin_toss: "{{ ['heads', 'tails'] | random(seed=inventory_hostname) }}"

    - name: Act on results of coin toss
      ansible.builtin.assert:
        that:
          - coin_toss == "heads"
        success_msg: "Success: coin toss was heads"
        fail_msg: "Failure: coin toss was tails"
```

Do block/rescue/always (cont)

```
rescue:
  - name: Run recovery actions
    ansible.builtin.debug:
      msg: "Running recovery because there was a failure"

always:
  - name: Always run this code
    ansible.builtin.debug:
      msg: "This code runs regardless of success"
```

pre_tasks / tasks / post_tasks

```
pre_tasks:
- name: Run a pre_task
  ansible.builtin.debug:
    msg: "Running a pre_task (shutdown database)"

tasks:
- name: Run a task
  ansible.builtin.debug:
    msg: "Running a task (backup database)"

post_tasks:
- name: Run a post_task
  ansible.builtin.debug:
    msg: "Running a post_task (start database)"
```

Create a YUM repository

```
 - name: Ensure directory exists
   ansible.builtin.file:
     path: /etc/yum.repos.d
     state: directory
     mode: '0755'

- name: Ensure myrepo YUM/DNF repository exists
   ansible.builtin.yum_repository:
     file: myrepo
     name: myrepo-internal
     description: Widget Company internal YUM/DNF repository
     baseurl: http://myrepo.example.com/yum/repository
     enabled: yes
     gpgcheck: no
     state: present
```