1 Creating Instances

Manually:
Choose Ubuntu as AMI
In additional settings, write bash script in the user data field as it will be run as root in the root directory when booting up. The script needed is

```
#!/bin/bash

#initializating instance
apt-get update
apt-get install tor -y
```

It is recommended to create a tag for the instance (e.g. iceball-client) especially in the case of needing to run commands on them afterward.
It is also recommended to create a security group first and assign it when creating instances. For security rules make sure ssh, webrtc (UDP), 443 ports are open.
Configure target capacity based on how many instances are needed.
Configure vCPU and Memory requirements as needed.
Leave everything else as default.

Launch Template
Use the same configuration of manually creating instances.
For creating tags using launch template, it is in Launch template name and description/template tags.
For user data, it is at the bottom of the page in Advanced details.

Figure 1: Location of user data when creating manually

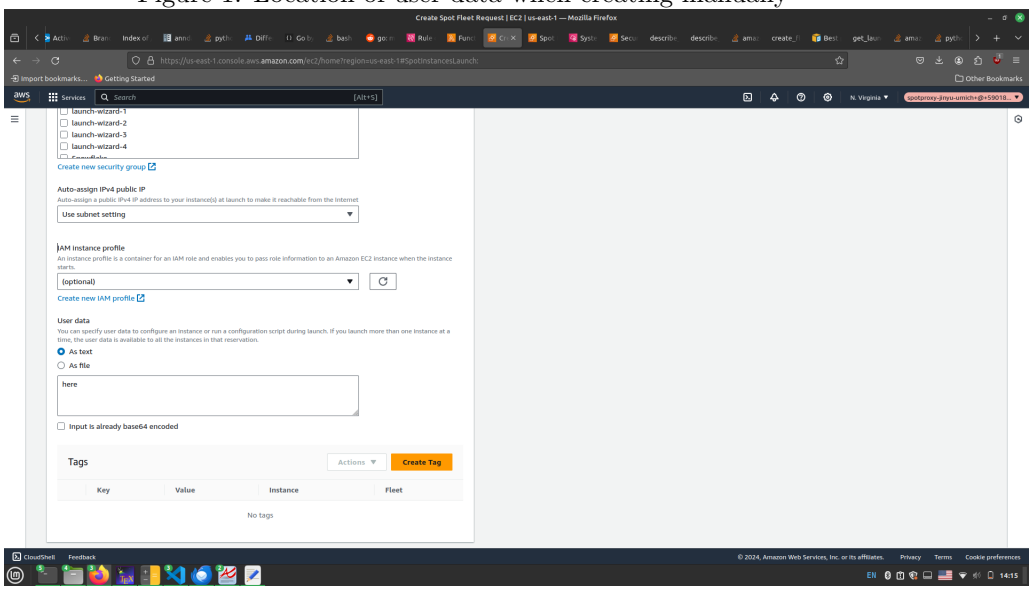
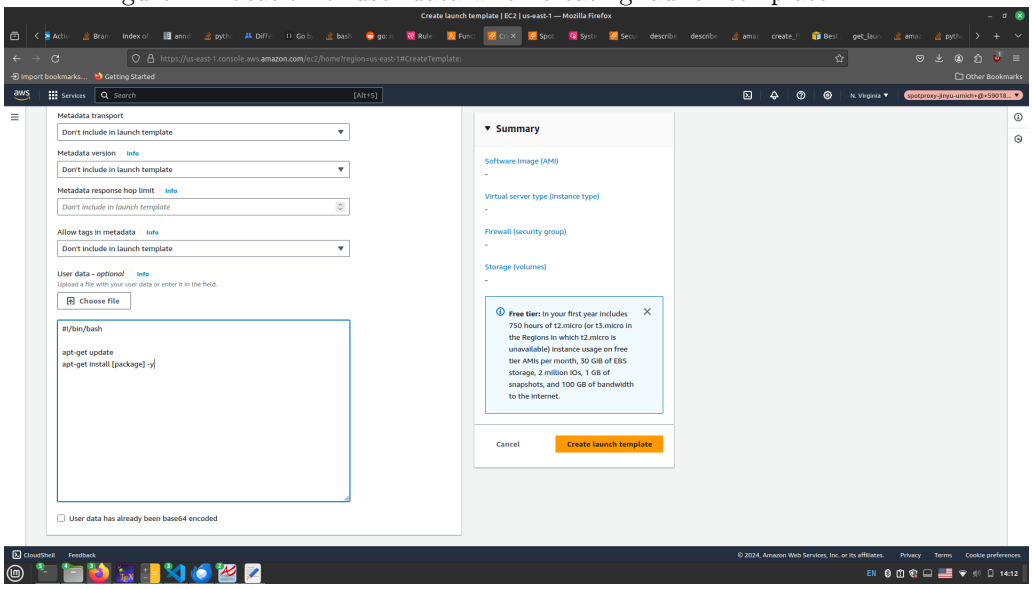


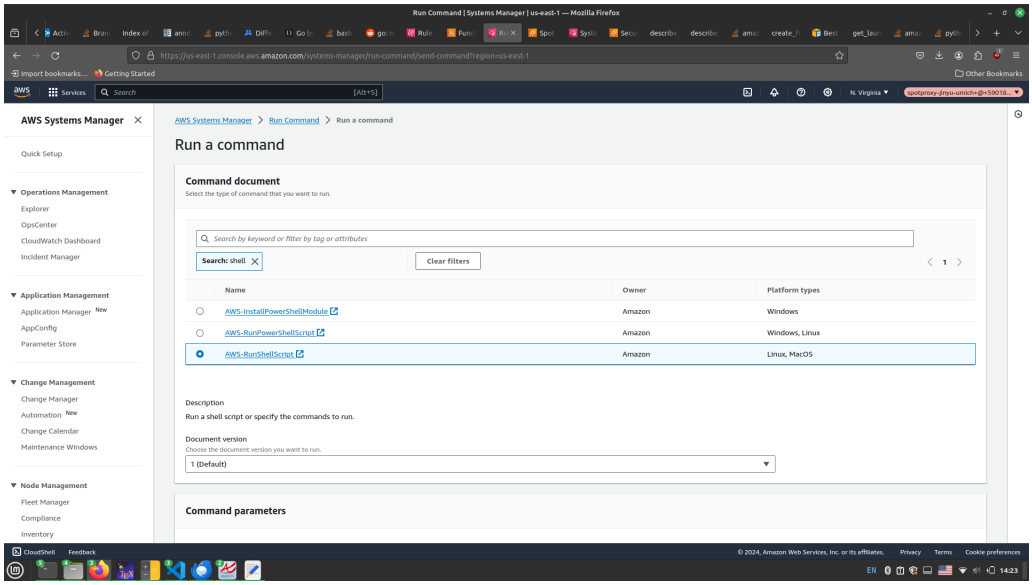
Figure 2: Location of user data when creating launch template



2 Running Commands on multiple Instances for Setting Up

This is useful for configuration after launch and run script not as root, or issuing commands to start the test procedure.

Use System Manager
Under Fleet Manager is all of the managed instances. There is a known issue that it might take up to 30 minutes for a newly created instance to show up. To speed up the process, try restarting all instances after creation.
To issue commands to all of them, use Run Command.



Put shell script in Command Parameters, and optional working directory (usually /home/ubuntu is the home directory).
Use instance tags to filter and select the instances
If there are more than 50 instances, change the Rate control targets limit to let all instances run the command at the same time.

Building Client

```
#!/bin/bash

#install golang
wget https://go.dev/dl/go1.21.3.linux-amd64.tar.gz
sudo rm -rf /usr/local/go
sudo tar -C /usr/local -xzf go1.21.3.linux-amd64.tar.gz
```

```
#Build
git clone https://github.com/unknown-cstdio/iceball.git
export PATH=$PATH:/usr/local/go/bin
export HOME=/home/ubuntu
export GOPATH=$HOME/go
export GOCACHE=$HOME/.cache/go-build
cd iceball/client
go build
```

Running Client Executable for Testing

```
#!/bin/bash

trap 'kill $(jobs -p)' EXIT

#Make sure Tor is stopped
sudo service tor stop
cd iceball/client

#This part is for logging
sudo rm client_log.log
sudo rm test.log
touch client_log.log
sudo chmod 777 client_log.log

#Runing the client
sudo tor -f torrc &
sleep 10

#Testing client by downloading large file
torify curl -O https://us.mirrors.cicku.me/ctan/systems/texlive
/Images/texlive2023.iso 2> test.log &

#How long it runs
sleep 300
```

Building Proxy

```
#!/bin/bash

#install golang
wget https://go.dev/dl/go1.21.3.linux-amd64.tar.gz
sudo rm -rf /usr/local/go
sudo tar -C /usr/local -xzf go1.21.3.linux-amd64.tar.gz

#Build
git clone https://github.com/unknown-cstdio/iceball.git
export PATH=$PATH:/usr/local/go/bin
```

```
export HOME=/home/ubuntu
export GOPATH=$HOME/go
export GOCACHE=$HOME/.cache/go-build
cd iceball/proxy
go build
```

Running Proxy

```
#!/bin/bash

cd iceball/proxy
proxy -broker [url to broker]
```

Building and Running the Broker There is only one broker needed so just configure it manually following the README in github repo. Make sure to modify the DNS of the domain so it points to the new ip.