Mini Homework 4

*Create load-balancer: HW4-balancer*

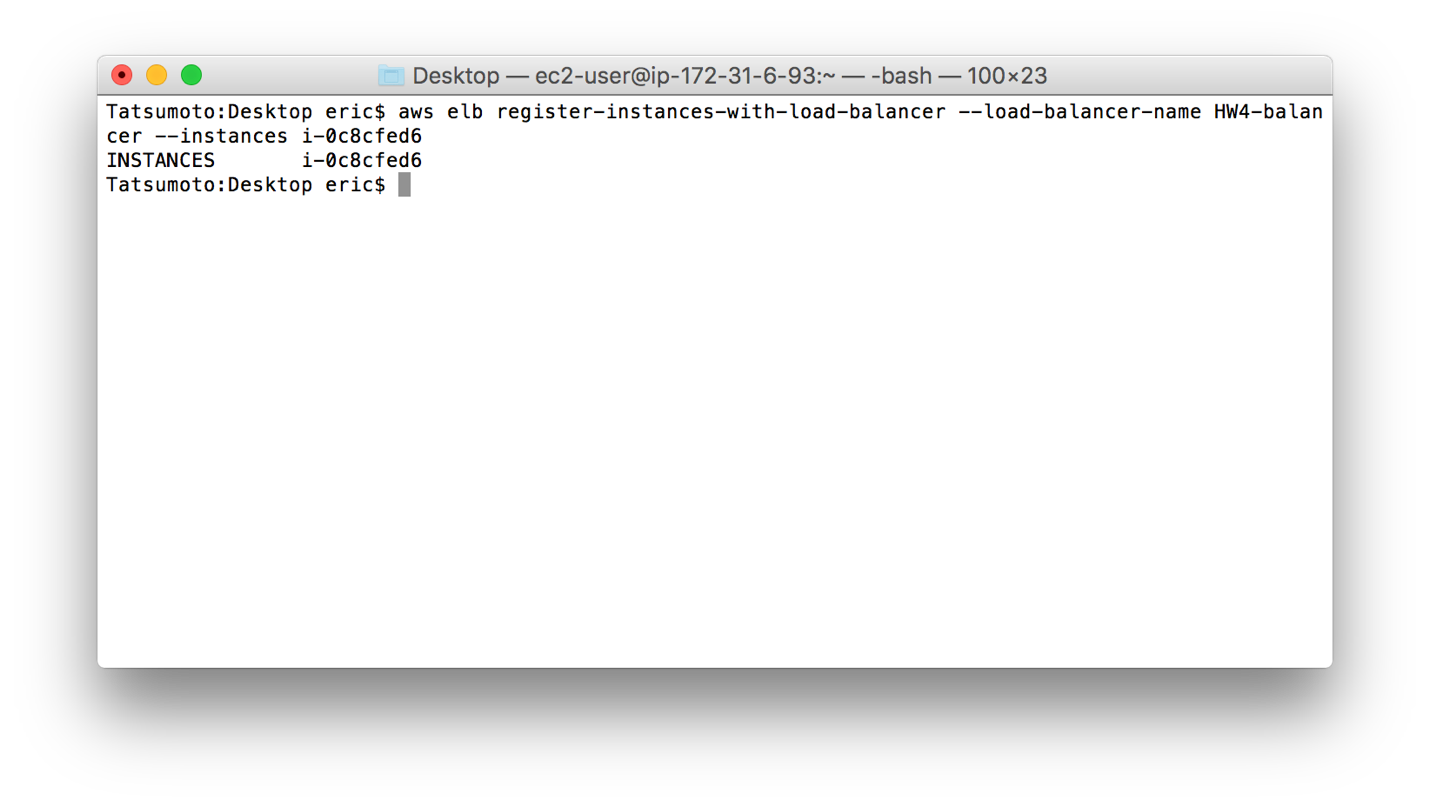
aws elb create-load-balancer --load-balancer-name HW4-balancer --listeners "Protocol=HTTP,LoadBalancerPort=80,InstanceProtocol=HTTP,InstancePort=80" "Protocol=HTTP,LoadBalancerPort=443,InstanceProtocol=HTTP,InstancePort=443" --availability-zones us-west-2c



load-balancer ID: HW4-balancer-1560852537.us-west-2.elb.amazonaws.com

*Register my instance with load-balancer*

aws elb register-instances-with-load-balancer --load-balancer-name HW4-balancer --instances i-0c8cfed6

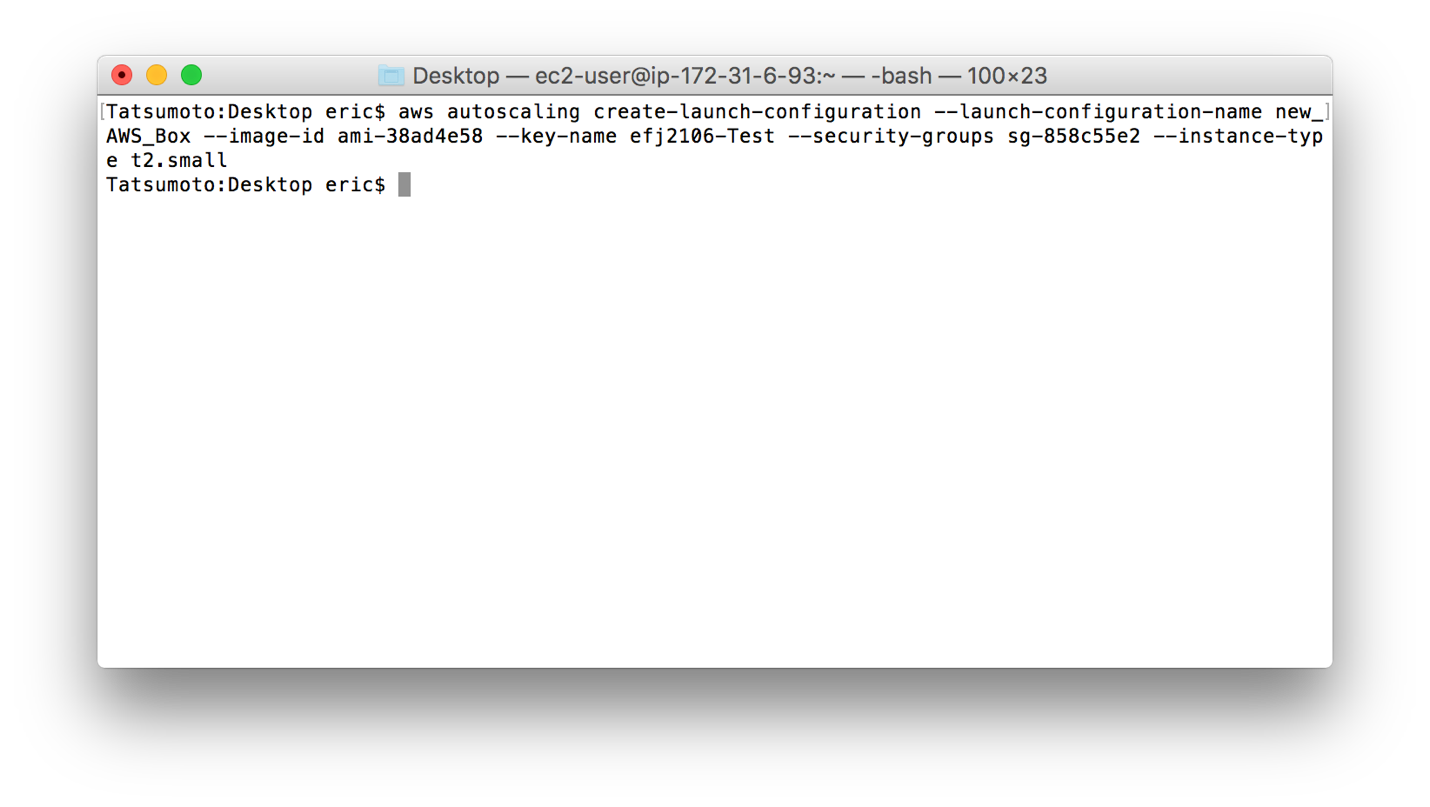


*Download AWS AutoScaling CLI Tools and Unzip*

\*\* I discovered you don’t actually need to download an run from local, these commands are available in aws cli under (aws autoscaling help)

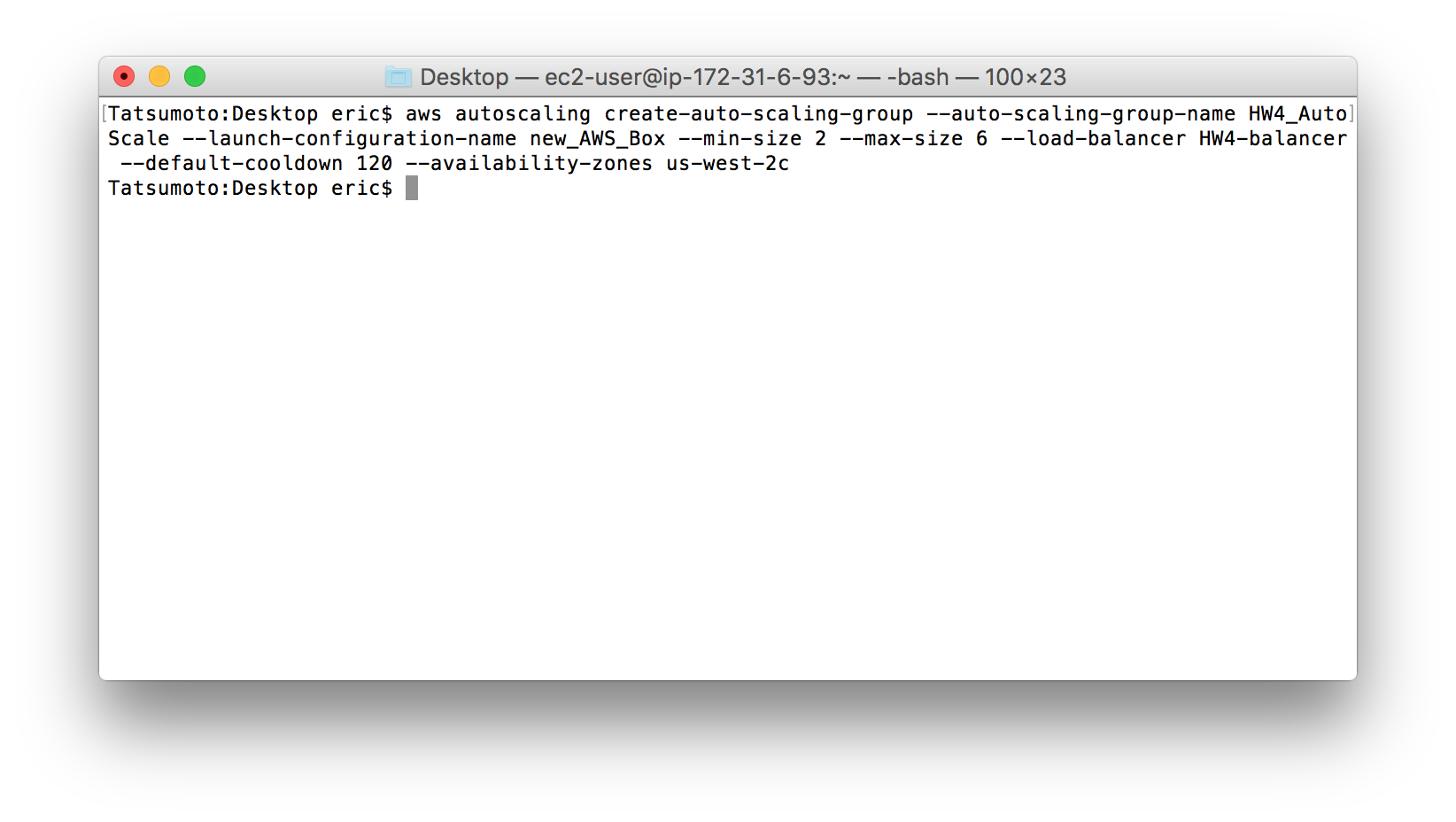
*Create Autoscale Configuration*

aws autoscaling create-launch-configuration --launch-configuration-name new\_AWS\_Box --image-id ami-38ad4e58 --key-name efj2106-Test --security-groups sg-858c55e2 --instance-type t2.small



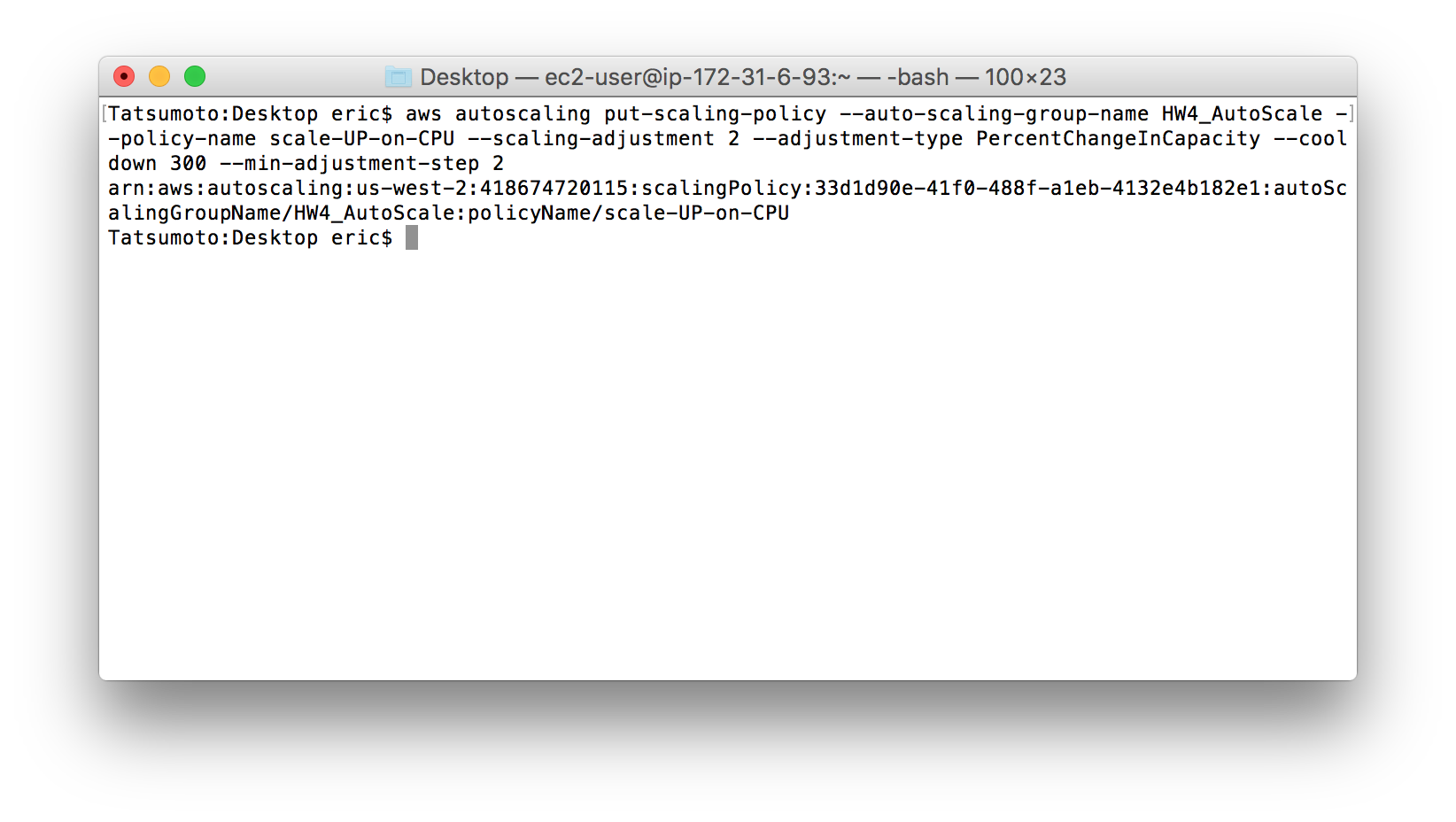
*Create Autoscale Group*

aws autoscaling create-auto-scaling-group --auto-scaling-group-name HW4\_AutoScale --launch-configuration-name new\_AWS\_Box --min-size 2 --max-size 6 --load-balancer HW4-balancer --default-cooldown 120 --availability-zones us-west-2c



*Create AutoScaling Policy (UP) - POLICY*

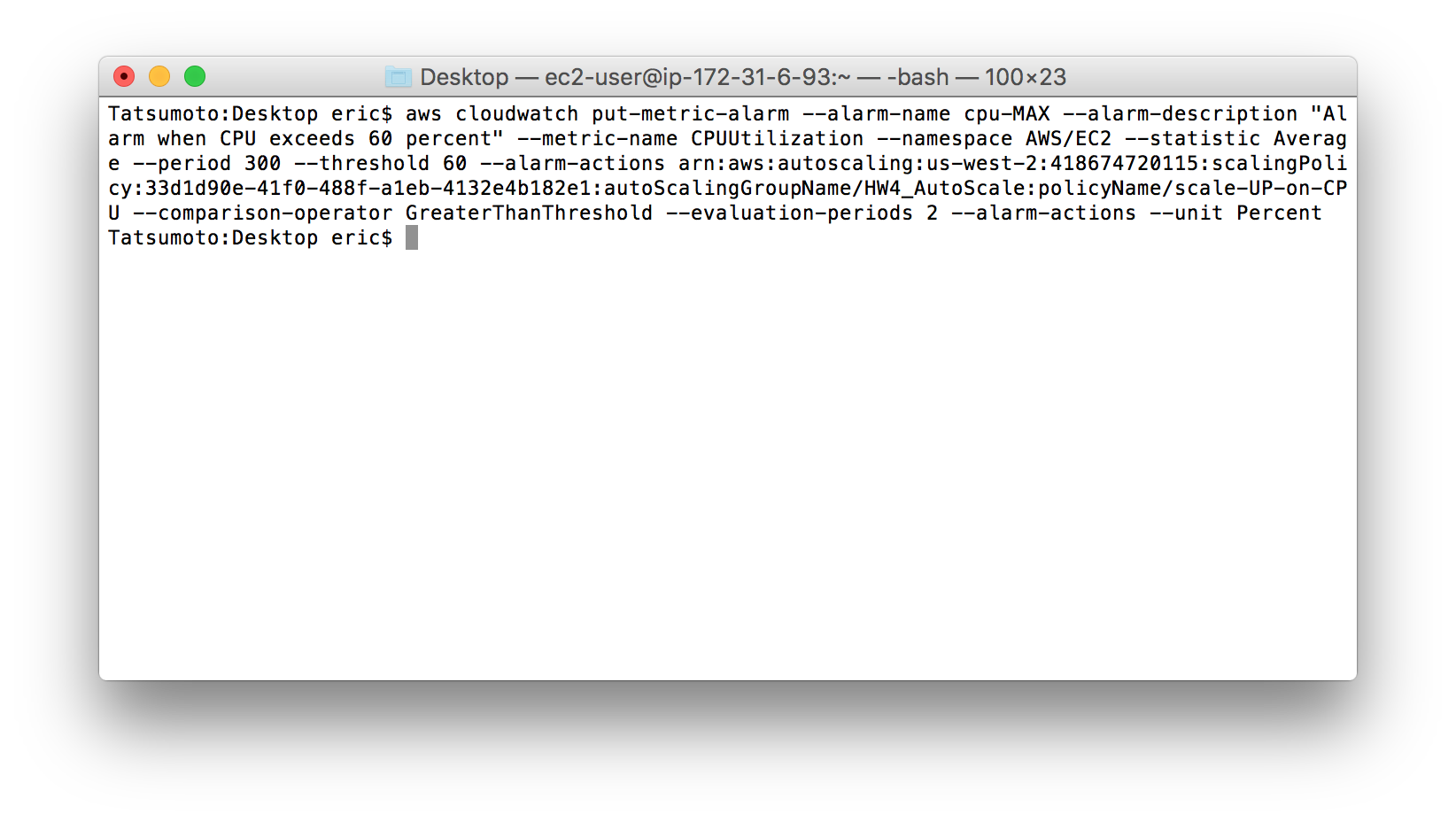
aws autoscaling put-scaling-policy --auto-scaling-group-name HW4\_AutoScale --policy-name scale-UP-on-CPU --scaling-adjustment 2 --adjustment-type PercentChangeInCapacity --cooldown 300 --min-adjustment-step 2



arn:aws:autoscaling:us-west-2:418674720115:scalingPolicy:33d1d90e-41f0-488f-a1eb-4132e4b182e1:autoScalingGroupName/HW4\_AutoScale:policyName/scale-UP-on-CPU

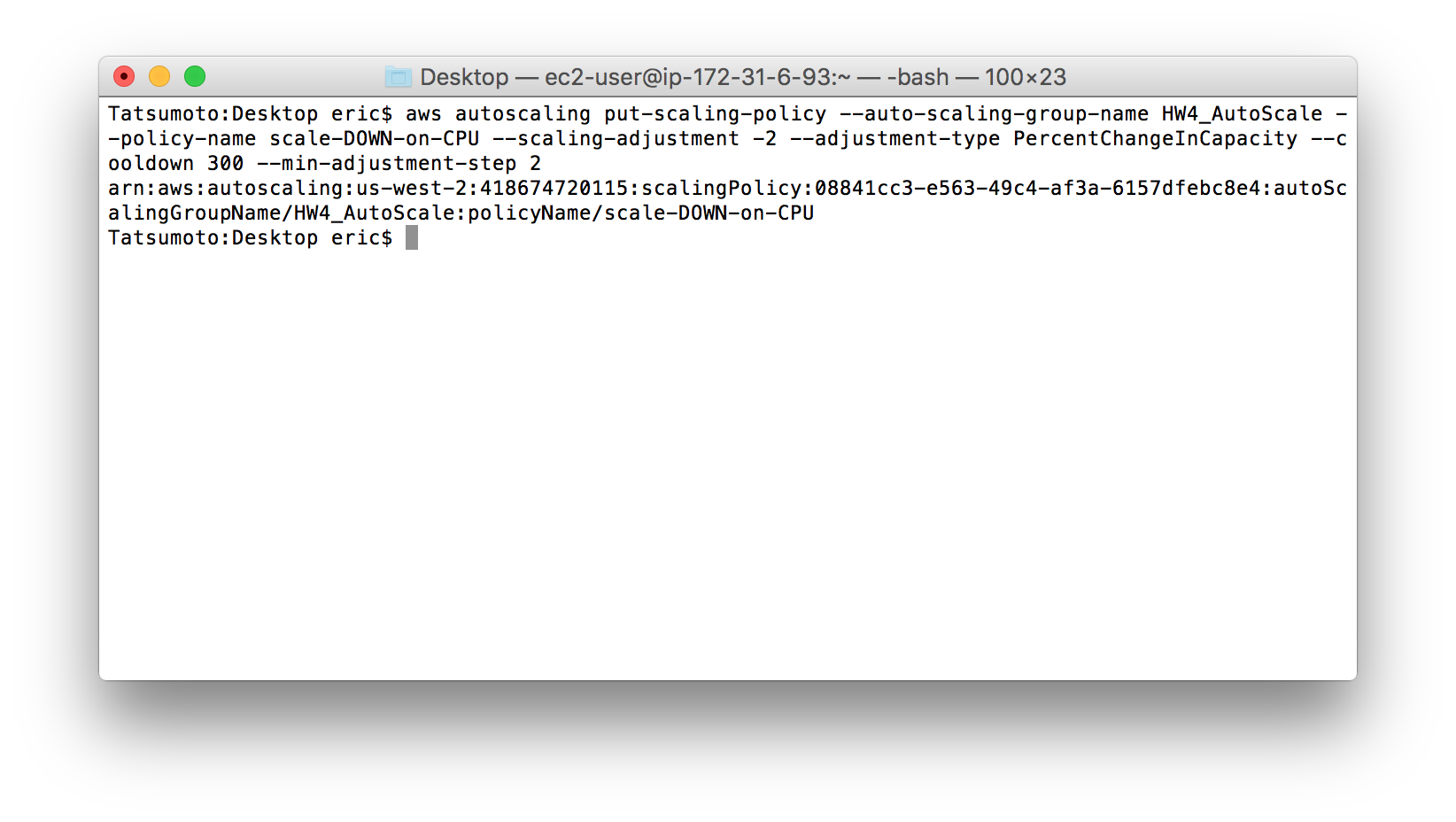
*Create Cloudwatch Alaram (UP) - POLICY ALARAMS*

aws cloudwatch put-metric-alarm --alarm-name cpu-MAX --alarm-description "Alarm when CPU exceeds 60 percent" --metric-name CPUUtilization --namespace AWS/EC2 --statistic Average --period 300 --threshold 60 --alarm-actions arn:aws:autoscaling:us-west-2:418674720115:scalingPolicy:33d1d90e-41f0-488f-a1eb-4132e4b182e1:autoScalingGroupName/HW4\_AutoScale:policyName/scale-UP-on-CPU --comparison-operator GreaterThanThreshold --evaluation-periods 2 --alarm-actions --unit Percent



*Create AutoScaling Policy (DOWN) - POLICY*

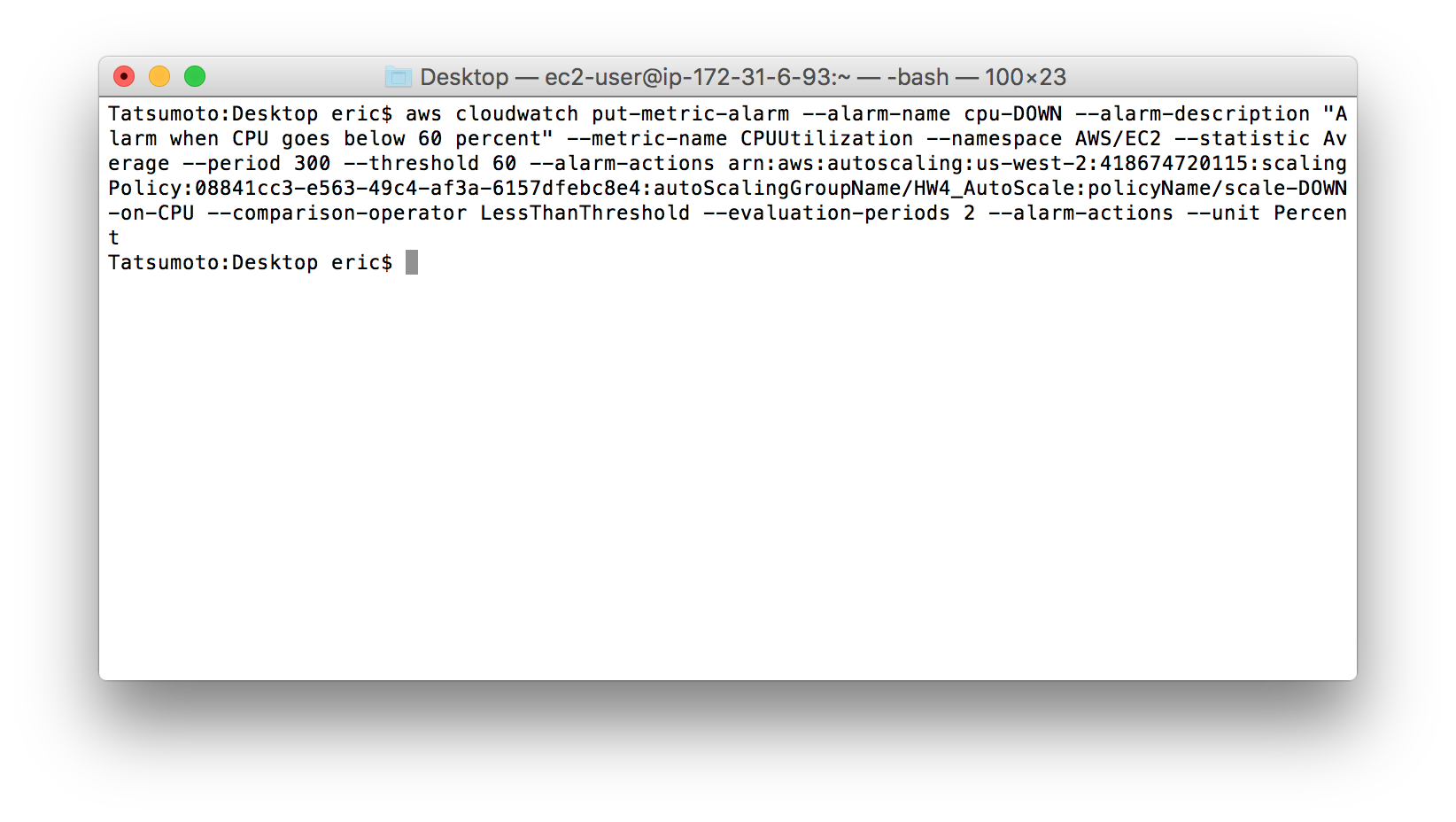
aws autoscaling put-scaling-policy --auto-scaling-group-name HW4\_AutoScale --policy-name scale-DOWN-on-CPU --scaling-adjustment -2 --adjustment-type PercentChangeInCapacity --cooldown 300 --min-adjustment-step 2



arn:aws:autoscaling:us-west-2:418674720115:scalingPolicy:08841cc3-e563-49c4-af3a-6157dfebc8e4:autoScalingGroupName/HW4\_AutoScale:policyName/scale-DOWN-on-CPU

*Create Cloudwatch Alaram (DOWN) - POLICY ALARAMS*

aws cloudwatch put-metric-alarm --alarm-name cpu-DOWN --alarm-description "Alarm when CPU goes below 60 percent" --metric-name CPUUtilization --namespace AWS/EC2 --statistic Average --period 300 --threshold 60 --alarm-actions arn:aws:autoscaling:us-west-2:418674720115:scalingPolicy:08841cc3-e563-49c4-af3a-6157dfebc8e4:autoScalingGroupName/HW4\_AutoScale:policyName/scale-DOWN-on-CPU --comparison-operator LessThanThreshold --evaluation-periods 2 --alarm-actions --unit Percent



2. Run a script inside the VM1 to generate load so that auto-scale adds another VM when CPU Util > 60%.

Originally I was going to install R and then run a program to hit the CPU threshold but the installation of R alone actually hit the 60% utilization. You can see the utilization on the right using HTOP and the extra instances that were spawned beneath it in the EC2 Console.

