Linux Automation - CROND Email Notifications

In this tutorial, I will be demonstrating how to send email notifications for completed cron jobs, from an AWS EC2, using Amazon SES (Simple Email Service).

Prerequisites

- an AWS Free Tier account
- Amazon SES (Simple Email Service) configured
- AWS Ubuntu 20 EC2 instance with Postfix configured for SES
- AWS RHEL 8 EC2 instance with Postfix configured for SES
- internet access

If you do not have an AWS account, you can access my **AWS Create Free Tier Account** tutorial here.

After creating an AWS account, I have a number of tutorials that can be completed, in the order specified below, to be ready to begin this tutorial.

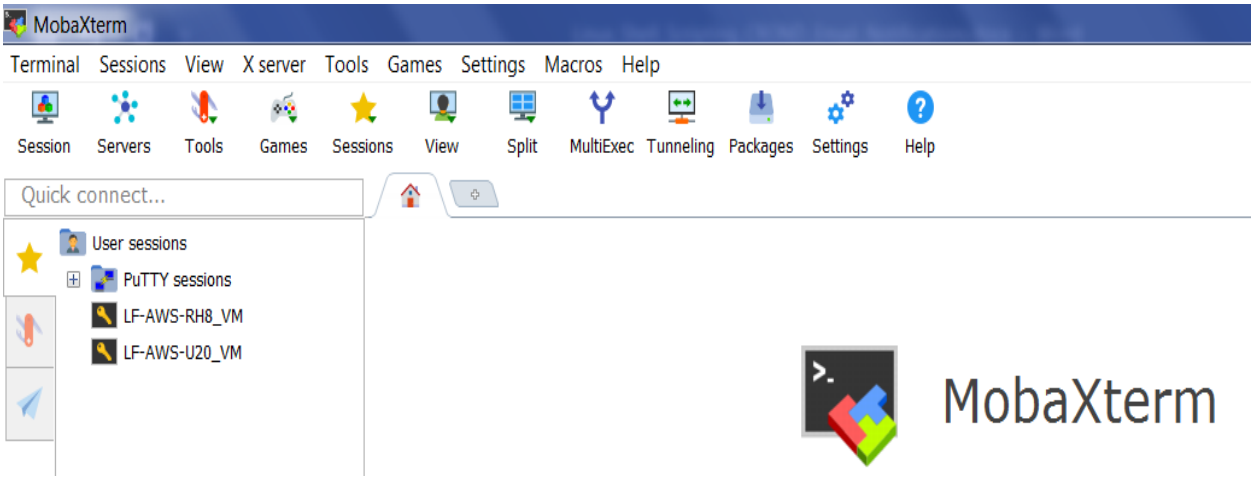
Ubuntu 20 EC2	RHEL 8 EC2
Create AWS Ubuntu 20 EC2 Instance	Create AWS RHEL 8 EC2 Instance
Amazon SES Configuration	Amazon SES Configuration
AWS Ubuntu 20 EC2 Postfix Install	AWS RHEL 8 EC2 Postfix Install

Finally, for the second, and third, cron jobs, I will be using the shell scripts created during my shell scripting tutorials, Check Local EC2 Services & Check Remote EC2 Services.

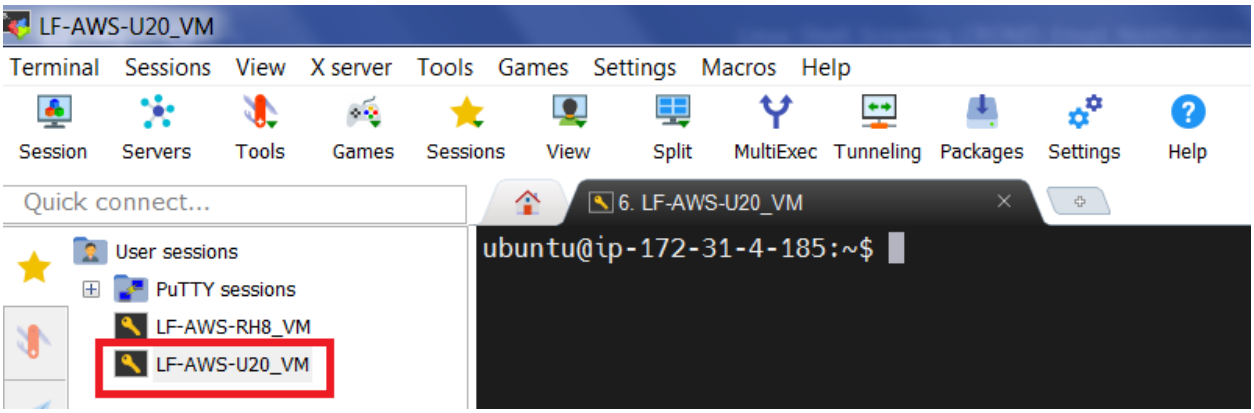
Steps to complete tutorial:

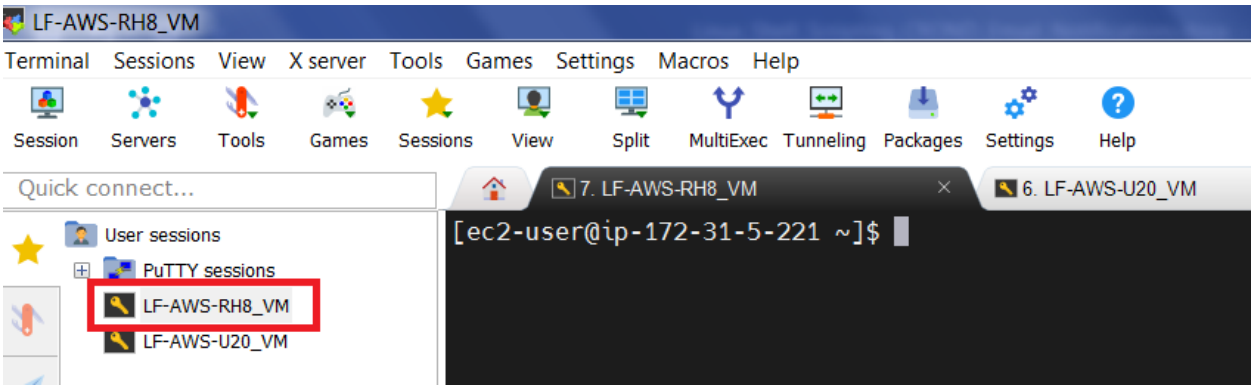
- Cron Job #1 - Send Simple Message
- Cron Job #2 - Check Local EC2 Service
- Cron Job #3 - Check Remote EC2 Service

To begin, I will connect to both of my running EC2 instances (Ubuntu 20 & RHEL 8) via SSH. I am using MobaXterm Portable as my SSH client. I find it to be a great tool and easy to use.



After the sessions opened, I executed the **clear** command to clear each of the terminals.





I will be creating cron jobs on each of my EC2 instances (Ubuntu 20 & RHEL 8). The first cron job will send a simple message while the second, and third, cron jobs will execute a script and send the results of execution. After each of the cron jobs complete, a notification will be sent to the designated recipient email address (Amazon SES verified identity).

Cron Job #1 - Send Simple Message

First, I will add a simple cron job to the root user's crontab file on each of my EC2 instances. Then, I will verify notification transmission via Amazon SES. The job will execute every 2 minutes.

$ sudo crontab -e

on Ubuntu 20 EC2
 # m h dom mon dow (m==minute, h==hour, dom==day of month, mon==month, dow==day of week)
***/2 * * * * echo 'This email was sent by the root user from Ubuntu 20 EC2 via Amazon SES'**

on RHEL 8 EC2
 # m h dom mon dow (m==minute, h==hour, dom==day of month, mon==month, dow==day of week)
***/2 * * * * echo 'This email was sent by the root user from RHEL 8 EC2 via Amazon SES'**

I will check the system logs to ensure that the cron job is running on my Ubuntu 20 EC2.

```
$ cat /var/log/syslog | grep CRON | tail -n 4
```

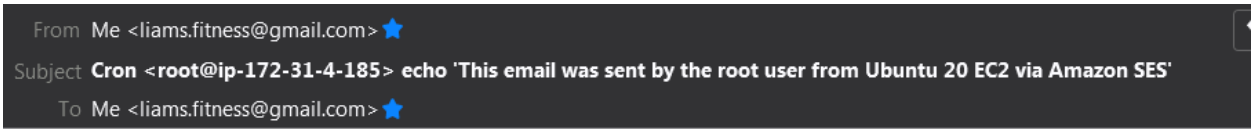
```
ubuntu@ip-172-31-4-185:~$
ubuntu@ip-172-31-4-185:~$ cat /var/log/syslog | grep CRON | tail -n 4
Sep  7 11:10:01 ip-172-31-4-185 CRON[181694]: (root) CMD (echo 'This email was sent by the root user from Ubuntu 20 EC2
n SES')
Sep  7 11:12:01 ip-172-31-4-185 CRON[181710]: (root) CMD (echo 'This email was sent by the root user from Ubuntu 20 EC2
n SES')
Sep  7 11:14:01 ip-172-31-4-185 CRON[181783]: (root) CMD (echo 'This email was sent by the root user from Ubuntu 20 EC2
n SES')
Sep  7 11:16:01 ip-172-31-4-185 CRON[181812]: (root) CMD (echo 'This email was sent by the root user from Ubuntu 20 EC2
n SES')
```

I will check the cron logs to ensure that the cron job is running on my RHEL 8 EC2.

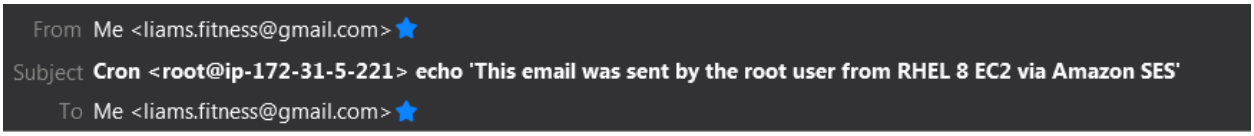
`$ tail -n 5 /var/log/cron`

```
[ec2-user@ip-172-31-5-221 ~]$
[ec2-user@ip-172-31-5-221 ~]$ tail -n 5 /var/log/cron
Sep  7 11:08:20 ip-172-31-5-221 crontab[99491]: (root) REPLACE (root)
Sep  7 11:08:20 ip-172-31-5-221 crontab[99491]: (root) END EDIT (root)
Sep  7 11:09:01 ip-172-31-5-221 crond[1064]: (root) RELOAD (/var/spool/cron/root)
Sep  7 11:10:01 ip-172-31-5-221 CROND[99520]: (root) CMD (echo 'This email was sent by the root user from RHEL 8 EC2
ES')
Sep  7 11:12:01 ip-172-31-5-221 CROND[99559]: (root) CMD (echo 'This email was sent by the root user from RHEL 8 EC2
ES')
```

I will now confirm that the cron notification emails were sent using my email client.



This email was sent by the root user from Ubuntu 20 EC2 via Amazon SES



This email was sent by the root user from RHEL 8 EC2 via Amazon SES

Both cron job notifications were successfully sent and received. Now would be a good time to disable each cron job to prevent unnecessary email notifications. Do this by commenting out, or deleting, each job via **sudo crontab -e** on each EC2 instance.

Cron Job #2 - Check Local EC2 Service

Please note, the SSH (Ubuntu 20 EC2)/SSHD (RHEL 8 EC2) service must be running on an EC2 instance for it to be accessible. Should the SSH/SSHD service be down on an EC2 instance, the AWS console and EC2

dashboard would need to be accessed manually to restart the instance so that the service would become available.

To prevent this, I will create a cron job on my Ubuntu 20 EC2 that will ensure the SSH service is running locally. I will use my shell script, **check_service.sh**, to check whether the SSH service is running, or not. If it isn't, it will automatically be restarted.

I want the cron job to run every 5 minutes and have the output generated sent as an email notification each time the job completes.

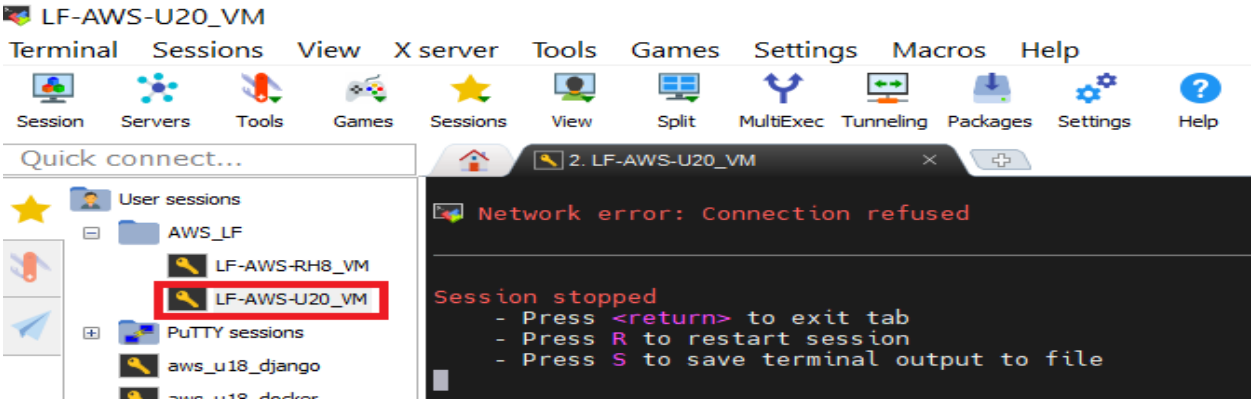
$ sudo crontab -e

```
# Ubuntu 20 EC2
# m h dom mon dow (m==minute, h==hour, dom==day of month, mon==month, dow==day of week)
*/5 * * * * /home/ubuntu/scripts/check_service.sh ssh
```

Now that the cron job is running, I will manually stop the service on my Ubuntu EC2 instance.

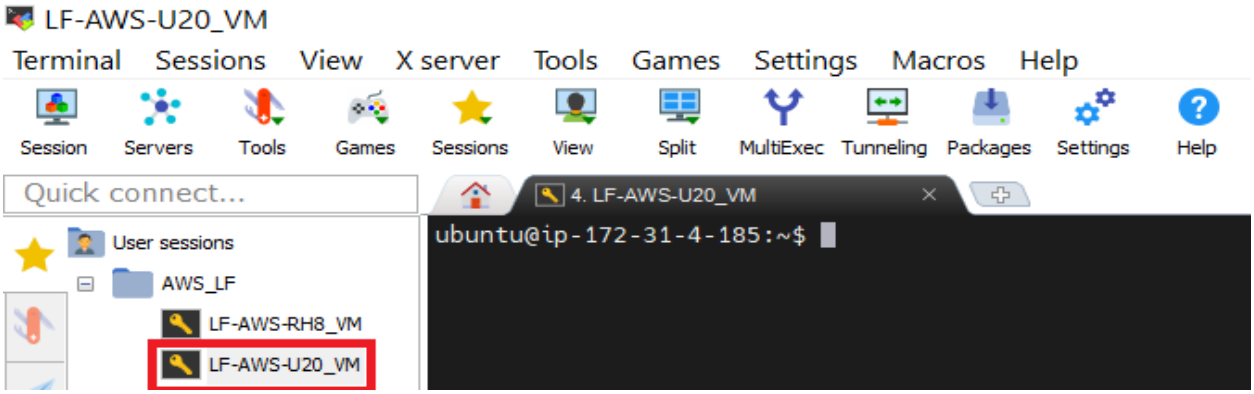
```
# Ubuntu 20 EC2
$ sudo systemctl stop ssh
```

I will now close my open Ubuntu 20 EC2 SSH session and attempt to reopen it.



You will notice that I cannot connect to my Ubuntu 20 EC2 instance.

I will wait the 5 minutes and then try connecting again.

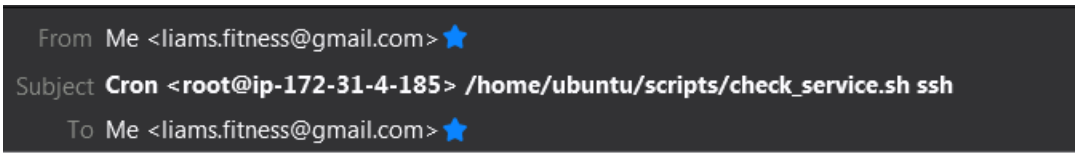


I will check the system logs to ensure that the cron job is running on my Ubuntu 20 EC2.

```
$ cat /var/log/syslog | grep CRON | tail -n 4
```

```
ubuntu@ip-172-31-4-185:~$
ubuntu@ip-172-31-4-185:~$ cat /var/log/syslog | grep CRON | tail -n 4
Sep  7 11:20:01 ip-172-31-4-185 CRON[181843]: (root) CMD (echo 'This email was sent by the root user t
n SES')
Sep  7 11:22:01 ip-172-31-4-185 CRON[181855]: (root) CMD (echo 'This email was sent by the root user t
n SES')
Sep  7 11:25:01 ip-172-31-4-185 CRON[181878]: (root) CMD (/home/ubuntu/scripts/check_service.sh ssh )
Sep  7 11:30:01 ip-172-31-4-185 CRON[181913]: (root) CMD (/home/ubuntu/scripts/check_service.sh ssh )
ubuntu@ip-172-31-4-185:~$
```

Finally, I will now confirm that the cron notification email was sent using my email client.



ssh is a service name
ssh is dead
ssh was restarted

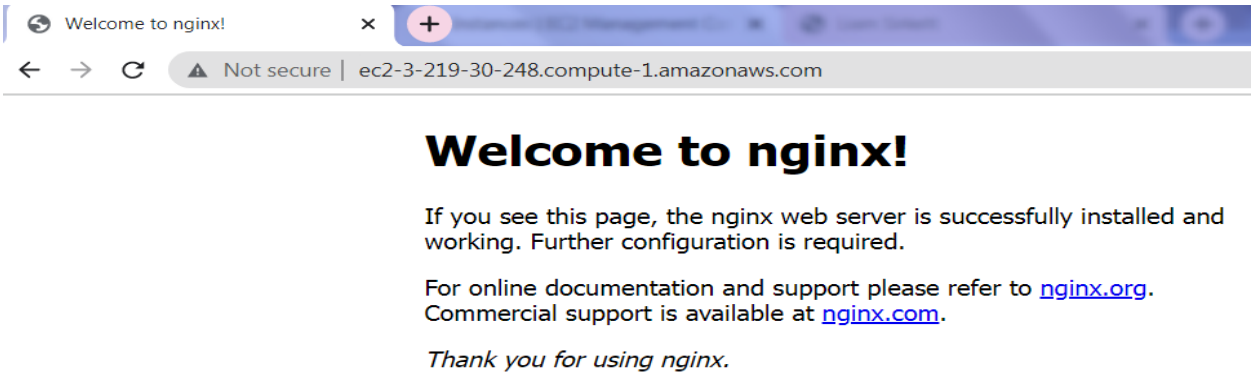
The cron job notification was successfully sent and received. This cron job ensures that I will always be able to access my Ubuntu 20 EC2 via SSH. I will leave it in place but discard the output generated. This will prevent cron job email notifications from being sent out each time the job completes.

```
# m h dom mon dow (m==minute, h==hour, dom==day of month, mon==month, dow==day of week)
*/5 * * * * /home/ubuntu/scripts/check_service.sh ssh &> /dev/null
```

Cron Job #3 - Check Remote EC2 Service

Now, on my RHEL 8 EC2, I will create a cron job that will ensure the NGINX service is running remotely on my Ubuntu 20 EC2. I will use my shell script, **check_remote_service.sh**, to check whether the NGINX service is running remotely. If it isn't, it will automatically be restarted.

Before creating the cron job, I will confirm that the default **nginx** website is accessible from my Ubuntu 20 EC2 instance. I will use the **Public IPv4 DNS** of my Ubuntu 20 EC2 instance to confirm this.



Now, I will manually stop the **nginx** service on my Ubuntu 20 EC2 instance.

```
# Ubuntu 20 EC2
$ sudo systemctl stop nginx
```

I will now confirm that the default **nginx** website is not accessible from my Ubuntu 20 EC2 instance.

I want the cron job to run every 5 minutes and have the output generated sent as an email notification each time the job completes.

```
$ sudo crontab -e
```

```
# RHEL 8 EC2, check NGINX service on Ubuntu 20 EC2
# m h dom mon dow (m==minute, h==hour, dom==day of month, mon==month, dow==day of week)
*/5 * * * * /home/ec2-user/scripts/check_remote_service.sh nginx
```

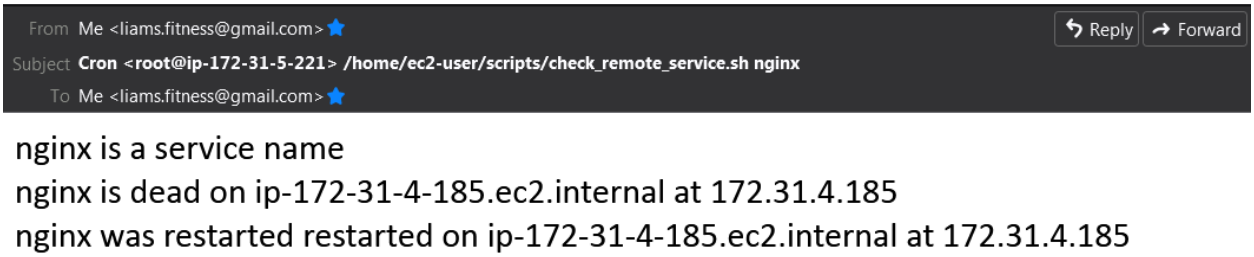
After cron job creation, I will wait five minutes and then try accessing the default **nginx** website again.

I will check the cron logs to ensure that the cron job ran on my RHEL 8 EC2.

```
$ tail -n 5 /var/log/cron
```

```
[ec2-user@ip-172-31-5-221 ~]$
[ec2-user@ip-172-31-5-221 ~]$ tail -n 5 /var/log/cron
Sep  7 11:41:41 ip-172-31-5-221 crontab[99840]: (root) BEGIN EDIT (root)
Sep  7 11:42:10 ip-172-31-5-221 crontab[99840]: (root) REPLACE (root)
Sep  7 11:42:10 ip-172-31-5-221 crontab[99840]: (root) END EDIT (root)
Sep  7 11:43:01 ip-172-31-5-221 crond[1064]: (root) RELOAD (/var/spool/cron/root)
Sep  7 11:45:01 ip-172-31-5-221 CROND[99862]: (root) CMD (/home/ec2-user/scripts/check_remote_service.sh nginx
[ec2-user@ip-172-31-5-221 ~]$
```

Finally, I will now confirm that the cron notification email was sent using my email client.



The cron job notification was successfully sent and received. I will leave this cron job running to ensure that I will always be able to access my websites hosted on my Ubuntu 20 EC2. Like the previous cron job I will discard the output generated to prevent email notifications.

I hope you have enjoyed completing this tutorial and found it helpful.

If you're interested, my main tutorials page is accessible **here**.

Back to Top