==================================================================================

Processing Video on AWS-EC2 for AI/ML model.

==================================================================================

I started a meeting and demonstrated a python script that pulls up a video file, starts to, and modifies (blurring faces) while it's running. The challenge I’m facing here is rendering delay of video and wanting to process video files in a smooth manner.

Initially, I was running my computer vision project on my EC2 and passing video to the project for detection. I was using x-11 forwarding via ssh to the server.

The script takes a delay (~6-7 seconds) when running on ec2 instances which I have tried on different types of ec2 instances (including G3 instance optimized for graphics-intensive applications), compared to when it launched on my desktop and works slower compared to desktop.

My ec2 instance is c5.large running on Ubuntu 18.04, and I have video forwarded from the instance to your desktop over ssh (ssh -X), and it shows 3 times less FPS (~200) compared to Desktop (~600)

At first, I tried to check on CPU/Memory utilization by running a script in a screen session on the ec2 instance: Following are the steps to check

1. started screen session:

# screen -S test

2. Started script:

# python3 <script> --video ./file.mp4

3. Detached session:

# <CTRL>+A <CTRL>+D

4. run the top command with the load shown by vcpu:

# top # then pressed 1.

5. checked on memory also with the free tool:

# free -m

I didn't notice high CPU or per vcpu or utilization in general (3% max by python script), or memory utilization ("free -m" command confirmed the same for memory). But then I noticed that the popped-up video got terminated when detached from the screen because the duration of the video supplied is over. And I rerun the "top" and free tool in the second ssh session to the instance in parallel, while running the script in the first ssh session, and noticed the same low load.

Then we listed screen sessions:

# screen -ls

Re-attached to existing:

# screen -R test

Terminated it

<CTRL>+C

Then, also killed all screen sessions:

# killall screen.

Then I have verified load on EBS volume vol-05d4d7c3a744ee566 and didn't find limits hit, which would raise concern.

At this point I decided to try to run script on ec2 instance in graphical environment without forwarding of X11 from the instance to your desktop.

For this following need to be done:

#1. Create AMI[2] of ec2 instance , or EBS snapshot[3] of volume , as a backup measure before doing changes.

2. Install XRDP[4]: https://www.e2enetworks.com/help/knowledge-base/how-to-install-remote-desktop-xrdp-on-ubuntu-18-04/

Note: bellow are the steps I used to install XRDP with MATE in my test environment on Ubuntu 18.04:

# sudo apt update

# sudo apt-get install xrdp

# sudo apt-get install mate-core mate-desktop-environment mate-notification-daemon

# sudo sed -i.bak '/fi/a #xrdp multiple users configuration \n mate-session \n' /etc/xrdp/startwm.sh

# sudo adduser xrdp ssl-cert

# sudo ufw status

Note: If ufw status not shown as "inactive" issue "sudo ufw allow 3389/tcp" to allow RDP.

# sudo /etc/init.d/xrdp restart

Note if ubuntu user has no password, set it as bellow, or use another system user with password.

# sudo passwd ubuntu

After that RDP should also be allowed in Inbound rules of SecurityGroup [5]: https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/authorizing-access-to-an-instance.html#add-rule-authorize-access

Then connect to instance with RDP and run python script to check on performance.

===

[1] Amazon EC2 Instance Types: https://aws.amazon.com/ec2/instance-types/

[2] Create an AMI from an Amazon EC2 Instance - https://docs.aws.amazon.com/toolkit-for-visual-studio/latest/user-guide/tkv-create-ami-from-instance.html

[3] Create Amazon EBS snapshots - Create a snapshot - https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-creating-snapshot.html#ebs-create-snapshot

[4] How to Install Remote Desktop (Xrdp) on Ubuntu 18.04 https://www.e2enetworks.com/help/knowledge-base/how-to-install-remote-desktop-xrdp-on-ubuntu-18-04/

[5] Authorize inbound traffic for your Windows instances - Add a rule for inbound RDP traffic to a Windows instance -

https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/authorizing-access-to-an-instance.html#add-rule-authorize-access

===

After I have XRDP installed on instance, I connected to instance with RDP, started an application, and verified that screen with video starts right away without delay, unlike when application

screen was forwarded with 3-4 seconds delay with X11 forwarded over ssh.

I noticed that application still operates slow comparing to my desktop, and think to change instance type from c5.large to g3s.xlarge which is optimized for graphics-intensive applications[1],

and then to install and connect to instance with NiceDCV[2].

After the instance's type was changed I performed following:

1. Followed prerequisites for Linux NICE DCV servers[3]: https://docs.aws.amazon.com/dcv/latest/adminguide/setting-up-installing-linux-prereq.html

2. Verified that Nvidia driver is already installed[4]: https://docs.aws.amazon.com/dcv/latest/adminguide/setting-up-installing-linux-server.html

# nvidia-smi -q | head

3. Installed NICE DCV Server on Linux[5]: https://docs.aws.amazon.com/dcv/latest/adminguide/setting-up-installing-linux-server.html

4. Performed Post-Installation checks[6]: https://docs.aws.amazon.com/dcv/latest/adminguide/setting-up-installing-linux-checks.html

5. started and enabled NICE DCV Server[7]: https://docs.aws.amazon.com/dcv/latest/adminguide/manage-start.html#manage-start-linux

# sudo systemctl start dcvserver

# sudo systemctl enable dcvserver

6. started NICE DCV console session[8]: https://docs.aws.amazon.com/dcv/latest/adminguide/managing-sessions-start.html#managing-sessions-start-manual

# sudo dcv create-session --owner=ubuntu --type=console my-session

# dcv list-sessions

===

[1] https://aws.amazon.com/ec2/instance-types/

[2] What Is NICE DCV? - https://docs.aws.amazon.com/dcv/latest/adminguide/what-is-dcv.html

[3] Prerequisites for Linux NICE DCV servers - https://docs.aws.amazon.com/dcv/latest/adminguide/setting-up-installing-linux-prereq.html

[4] Install NVIDIA drivers on Linux instances - Installation options - https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/install-nvidia-driver.html#nvidia-installation-options

[5] Install the NICE DCV Server on Linux - https://docs.aws.amazon.com/dcv/latest/adminguide/setting-up-installing-linux-server.html

[6] Post-Installation checks - https://docs.aws.amazon.com/dcv/latest/adminguide/setting-up-installing-linux-checks.html

[7] Starting the NICE DCV Server - Starting the NICE DCV Server on Linux - https://docs.aws.amazon.com/dcv/latest/adminguide/manage-start.html#manage-start-linux

[8] Starting NICE DCV sessions - Manually starting console and virtual sessions - https://docs.aws.amazon.com/dcv/latest/adminguide/managing-sessions-start.html#managing-sessions-start-manual

[9] Optimize GPU settings - https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/optimize\_gpu.html

===

when I connected to g3s.xlarge ec2 instance with NICE DCV, and get the same experience for application's performance as on my desktop pc. I also noticed little network

lag which can be reduced, when it(instance) should be launched in AWS Region closer to your Zonal location instead of us-east-2 (Ohio) to decrease network latency: https://aws.amazon.com/about-aws/global-infrastructure/regions\_az/

Bellow I have combined steps you followed during 2 last Chime sessions, to install and connect to NICE DCV server on g3s.xlarge instance i-02a8f54afbe2574a3:

1. Followed prerequisites for Linux NICE DCV servers on for Ubuntu 18.x [1]: https://docs.aws.amazon.com/dcv/latest/adminguide/setting-up-installing-linux-prereq.html

NOTE: step to "Install GPU drivers for graphics instances" was skipped because drivers were preinstalled on AMI, which was verified with following command: "nvidia-smi -q | head"

# sudo apt update

# sudo apt install ubuntu-desktop

# sudo apt install lightdm #select lightdm not gdm3 when prompted.

# sudo apt upgrade

# sudo reboot

# sudo systemctl get-default

# sudo systemctl set-default graphical.target

# sudo systemctl isolate graphical.target

# ps aux | grep X | grep -v grep

# sudo apt install mesa-utils

# sudo DISPLAY=:0 XAUTHORITY=$(ps aux | grep "X.\*\-auth" | grep -v grep | sed -n 's/.\*-auth \([^ ]\+\).\*/\1/p') glxinfo | grep -i "opengl.\*version"

# sudo nvidia-xconfig --preserve-busid --enable-all-gpus

# sudo systemctl isolate multi-user.target

# sudo systemctl isolate graphical.target

# sudo DISPLAY=:0 XAUTHORITY=$(ps aux | grep "X.\*\-auth" | grep -v grep | sed -n 's/.\*-auth \([^ ]\+\).\*/\1/p') glxinfo | grep -i "opengl.\*version"

2. Installed NICE DCV Server on Linux for Ubuntu 18.04[2]: https://docs.aws.amazon.com/dcv/latest/adminguide/setting-up-installing-linux-server.html

NOTE: all optional steps were skipped.

# cd

# wget https://d1uj6qtbmh3dt5.cloudfront.net/NICE-GPG-KEY

# gpg --import NICE-GPG-KEY

# wget https://d1uj6qtbmh3dt5.cloudfront.net/2022.0/Servers/nice-dcv-2022.0-12627-ubuntu1804-x86\_64.tgz

# tar -xvzf nice-dcv-2022.0-12627-ubuntu1804-x86\_64.tgz && cd nice-dcv-2022.0-12627-ubuntu1804-x86\_64

# sudo apt install ./nice-dcv-server\_2022.0.12627-1\_amd64.ubuntu1804.deb

# sudo usermod -aG video dcv

3. Performed Post-Installation checks[3]: https://docs.aws.amazon.com/dcv/latest/adminguide/setting-up-installing-linux-checks.html

# sudo systemctl isolate multi-user.target

# sudo systemctl isolate graphical.target

# sudo DISPLAY=:0 XAUTHORITY=$(ps aux | grep "X.\*\-auth" | grep -v grep | sed -n 's/.\*-auth \([^ ]\+\).\*/\1/p') xhost | grep "SI:localuser:dcv$"

4. started and enabled NICE DCV Server[4]: https://docs.aws.amazon.com/dcv/latest/adminguide/manage-start.html#manage-start-linux

# sudo systemctl start dcvserver

# sudo systemctl enable dcvserver

5. started NICE DCV console session manually[5]: https://docs.aws.amazon.com/dcv/latest/adminguide/managing-sessions-start.html#managing-sessions-start-manual

# sudo dcv create-session --owner=ubuntu --type=console my-session

# dcv list-sessions

6. Enabled Automatic Console Session[6]: https://docs.aws.amazon.com/dcv/latest/adminguide/managing-sessions-start.html#managing-sessions-start-auto

# sudo nano /etc/dcv/dcv.conf

[session-management]

create-session = true

[session-management/automatic-console-session]

owner="ubuntu"

# sudo systemctl restart dcvserver

# dcv list-sessions

We discussed that when session created manually, the session id is provided ("my-session" in this case), and access to session from the client would have syntax:

server\_hostname\_or\_IP:port#session\_id

But with enabled automatic console session, the session id is skipped and connection can be done without it:

server\_hostname\_or\_IP:port

7. Added Inbound rules into Security Group of instance to allow connections to NICE DCV server on ports TCP/8443 and UDP/8443.

8. Optimized GPU settings for G3 instance to achieve the best performance on NVIDIA GPU[7]: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/optimize\_gpu.html

# sudo nvidia-persistenced

# sudo nvidia-smi --auto-boost-default=0

# sudo nvidia-smi -ac 2505,1177

We discussed that according to man for nvidia-smi[8], the "-ac" option specifies maximum <memory,graphics> clocks as a pair (2505,1177) that defines GPU's speed while running applications on a GPU:

https://code.tools/man/1/alt-nvidia-304-smi/#lbAY

9. Installed the latest NICE DCV Windows client on your desktop[9]: https://docs.aws.amazon.com/dcv/latest/userguide/client-windows.html#client-windows-install

10. Connected to NICE DCV session with Windows client[10]: https://docs.aws.amazon.com/dcv/latest/userguide/using-connecting-win.html

server\_hostname\_or\_IP:port

We also discussed that you are going to follow the steps above to test you application on new c5.large instance with NICE DCV server and configure it with dummy driver for non-GPU instances as described in according

section of prerequisites for Linux NICE DCV servers[11]: https://docs.aws.amazon.com/dcv/latest/adminguide/setting-up-installing-linux-prereq.html#linux-prereq-nongpu

We also scheduled Chime meeting for 05/24/2022 8:00 PST to cover this, and I'm going to send you a link for new Chime meeting in this case 5-10 minutes before scheduled time.

I'm looking forward into our next Chime meeting.

===

[1] Prerequisites for Linux NICE DCV servers - https://docs.aws.amazon.com/dcv/latest/adminguide/setting-up-installing-linux-prereq.html

[2] Install the NICE DCV Server on Linux - https://docs.aws.amazon.com/dcv/latest/adminguide/setting-up-installing-linux-server.html

[3] Post-Installation checks - https://docs.aws.amazon.com/dcv/latest/adminguide/setting-up-installing-linux-checks.html

[4] Starting the NICE DCV Server - Starting the NICE DCV Server on Linux - https://docs.aws.amazon.com/dcv/latest/adminguide/manage-start.html#manage-start-linux

[5] Starting NICE DCV sessions - Manually starting console and virtual sessions - https://docs.aws.amazon.com/dcv/latest/adminguide/managing-sessions-start.html#managing-sessions-start-manual

[6] Starting NICE DCV sessions - Enabling Automatic Console Sessions - https://docs.aws.amazon.com/dcv/latest/adminguide/managing-sessions-start.html#managing-sessions-start-auto

[7] Optimize GPU settings - https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/optimize\_gpu.html

[8] man nvidia-smi (1) https://code.tools/man/1/alt-nvidia-304-smi/#lbAY

[9] Windows client - Installable Windows client - https://docs.aws.amazon.com/dcv/latest/userguide/client-windows.html#client-windows-install

[10] Connecting to a NICE DCV session using the Windows client - https://docs.aws.amazon.com/dcv/latest/userguide/using-connecting-win.html

[11] Prerequisites for Linux NICE DCV servers - Install XDummy driver for non-GPU instances - https://docs.aws.amazon.com/dcv/latest/adminguide/setting-up-installing-linux-prereq.html#linux-prereq-nongpu

===

we walked through the steps of installation of NICE DCV on c5.large ec2 instance i-0a396227759cbe376 running Ubuntu 18.04. You completed almost the same steps as for NICE DCV installation on g3s.xlarge, with only following differences:

1. Due to c5.large has no GPU like g3s.xlarge, it requires installation and configuration of XDummy driver instead drivers for GPU, and the section of GPU drivers installation was skipped and corresponding section for

XDummy driver installation and configuration was followed according to documentation for NICE DCV prerequisites[1]: https://docs.aws.amazon.com/dcv/latest/adminguide/setting-up-installing-linux-prereq.html#linux-prereq-gpu

2. While installing NICE DCV Server for Ubuntu 18.04 by following steps in the documentation[2], the installation of NICE DCV server failed with following error:

# sudo apt install ./nice-dcv-server\_2022.0.12760-1\_amd64.ubuntu1804.deb

..

var/lib/dpkg/info/nice-dcv-server.postinst: 8: /var/lib/dpkg/info/nice-dcv-server.postinst: dpkg-architecture: not found

dpkg: error processing package nice-dcv-server (--configure):

installed nice-dcv-server package post-installation script subprocess returned error exit status 127

..

You have it addressed by installing "dpkg-dev" package containing required dpkg-architecture tool[3]: https://www.linuxquestions.org/questions/linux-desktop-74/ubuntu-dpkg-architecture-not-found-501592/

# sudo apt install dpkg-dev

And after that successfully installed nice-dcv-server deb package:

# sudo apt install ./nice-dcv-server\_2022.0.12760-1\_amd64.ubuntu1804.deb

3. You have created password for ubuntu user:

# sudo passwd ubuntu

4. At the first login with NICE DCV, the graphical session appeared in unresponsive black screen, and you have rebooted the instance to address it.

After that you installed python version 3.9 and dependencies required for your application, then downloaded your application on instance and started it. We noticed that performance of application was alike for g3s.xlarge,

and discussed that leverage of GPU is not required for your application.

We also discussed that should you run your application working with video streams, you may consider C5n ec2 instance to take advantage of improved network throughput and packet rate performance[4]:

https://aws.amazon.com/ec2/instance-types/

At this point we finished our Chime session. Please let me know should you have any questions or need assistance.

===

[1] Prerequisites for Linux NICE DCV servers - Install GPU drivers for graphics instances - https://docs.aws.amazon.com/dcv/latest/adminguide/setting-up-installing-linux-prereq.html#linux-prereq-gpu

[2] Install the NICE DCV Server on Linux - Install the NICE DCV Server - https://docs.aws.amazon.com/dcv/latest/adminguide/setting-up-installing-linux-server.html#linux-server-install

[3] Ubuntu dpkg-architecture not found https://www.linuxquestions.org/questions/linux-desktop-74/ubuntu-dpkg-architecture-not-found-501592/

[4] Amazon EC2 Instance Types https://aws.amazon.com/ec2/instance-types/

===