IDC Desktops and SlicerOnDemand

Background

There has been a background project for a few years to explore on-demand virtual desktops in the cloud[1] to facilitate the use of traditional GPU-enabled workstation applications like 3D Slicer on virtual machines. These VMs are both "close" to the data in a networking sense to provide fast access, but also can be configured with hardware to meet the demands of a given visualization or processing task. Machines with up to 4 TB of RAM or up to 640 GB of GPU memory are currently available but require cloud computing expertise to make use of. Our goal has been to streamline the process so that users can more easily get access to these resources.

So far we have succeeded in creating GCP bootable images that come with a full linux desktop pre-installed that exposes a remote desktop server. We have shown that we can get from button press to a fully provisioned and running application in just under 90 seconds and demonstrated that the remote desktop performance is suitable for tasks such as volume rendering and interactive segmentation. In addition, these machines support installation of GPU accelerated machine learning tools such as PyTorch and MONAI.

We would like to make this service available so that scientists who are used to working in desktop environments can more easily transition to cloud computing and make use of IDC[2] resources.

Current Status

The current bootable images are based on Ubuntu 20.04 with customized systemd scripts to bring up X11, the mwm window manager, x11vnc, and noVNC. Slicer is also started automatically. The boot images are generated by scripts[3], so exact behavior can be configured and images created in about 10 minutes. We also created a prototype of a one-click launcher[4] that uses the gcloud installation on a local computer to launch a VM with this image and then open a browser tab to the remote desktop when the instance has booted. The browser is pointed to a localhost port which has been tunneled through gcloud ssh so that access to the machine is strongly controlled via standard Google security.

In practice, the VM is very usable as a workstation that can run a variety of analysis tools and the user can either keep it running or delete it depending on their needs.

---

[1] https://projectweek.na-mic.org/PW35_2021_Virtual/Projects/SlicerOnDemand/
[2] https://portal.imaging.datacommons.cancer.gov/
[3] https://github.com/pieper/SlicerMachines
[4] https://github.com/pieper/SlicerOnDemand

Desired State

There are some improvements we'd like to see in this process that fall into two broad categories related to launching and managing the VMs and then the user experience with the VM desktop itself.

Launching and Managing

It would be much easier for users if users didn't need to install the gcloud SDK on their local machine. Even for PhD-level computer scientists this is a barrier due to the many steps involved and it would be more friendly to provide this functionality via a web app. Most of the the gcloud functions in the current prototype (launching VM, monitor status, etc) are also available to authenticated users via the JavaScript API available in the browser, so it should be possible with a statically hosted web app to provide a console to launch, connect/reconnect, stop, start, reboot, and terminate one or more VMs. The exception is the ssh tunnel, since the web app cannot control local ports. One alternative would be to install and configure Chrome Remote Desktop on the VM to allow secured access. Another alternative would be to use the API to configure GCP's Identity Aware Proxy (IAP) such that only the user can access the noVNC endpoint on the VM. A complexity of the IAP approach is that it can only be used with https endpoints so we would need to automate the creation of trusted certificates. Also in previous experiments IAP was just noticeably slower than gcloud ssh for remote desktops. Chrome Remote Desktop is comparable to the noVNC + gcloud ssh method in terms of performance and security.

We should also consider how this environment can help scientists make use of IDC data. One idea is to pre-populate a disk image with the contents of an IDC cohort or to make an organized directory hierarchy of symlinks to DICOM files in the IDC buckets exposed via gcsfuse.

The idea of the web app for managing these VMs would be to give the users the functionality they need without the complexity of the GCP console. Research and design work will be required to determine the right trade-offs between flexibility and complexity in terms of CPU/GPU types, memory and disk sizes, etc. Probably a few "standard" options like small, medium, and large could handle a lot of use cases.

VM Desktop User Experience

The current prototype exposes a workable, but minimalist desktop environment with the Motif Window Manager, mwm, a decades-old tool that lacks modern conveniences and appearance. The reason for this is that in order to boot directly to the application with vnc enabled and other window managers weren't working correctly when started with systemd. These issues are no doubt solvable by a sysadmin or advanced user with deep familiarity with the technologies involved (Xorg, gdm, Unity, etc).

A related task is to determine a set of useful utilities to be pre-installed on the boot images to facilitate cancer research with IDC data. These could include 3D Slicer with extensions like SlicerRadiomics and SlicerJupyter pre-installed, but also possibly Chrome/Firefox, R Studio, gcsfuse, or bioinformatics tools.

Further Considerations

The current prototype is focused on GCP, but it would be desirable to stick to a technology stack that can also be applied in AWS and Azure, and also possibly deployed on-premises. Also we have focused on Linux desktops, but some users might prefer Windows, which is also available as a cloud desktop option. In fact, AWS offers App Streaming and Microsoft is said to be releasing Windows 365 cloud hosted desktops so we should investigate the extent to which those technologies might address similar issues as the work proposed here.