Chad Ballay

CYBR 340-342

Week 2 – Module 2

7/12/2020

Virtual Machine Hands On

1. **You need to provide a one-page synopsis of your experience along with at least two screenshots of your virtual environments.**

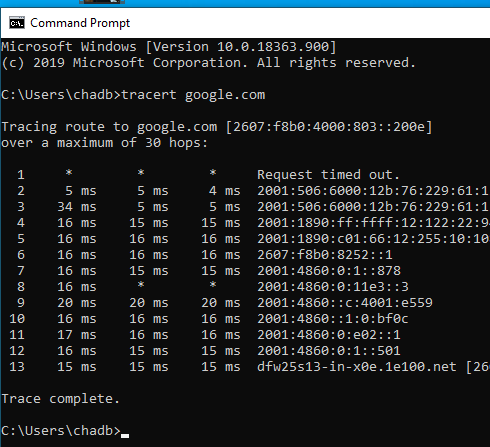
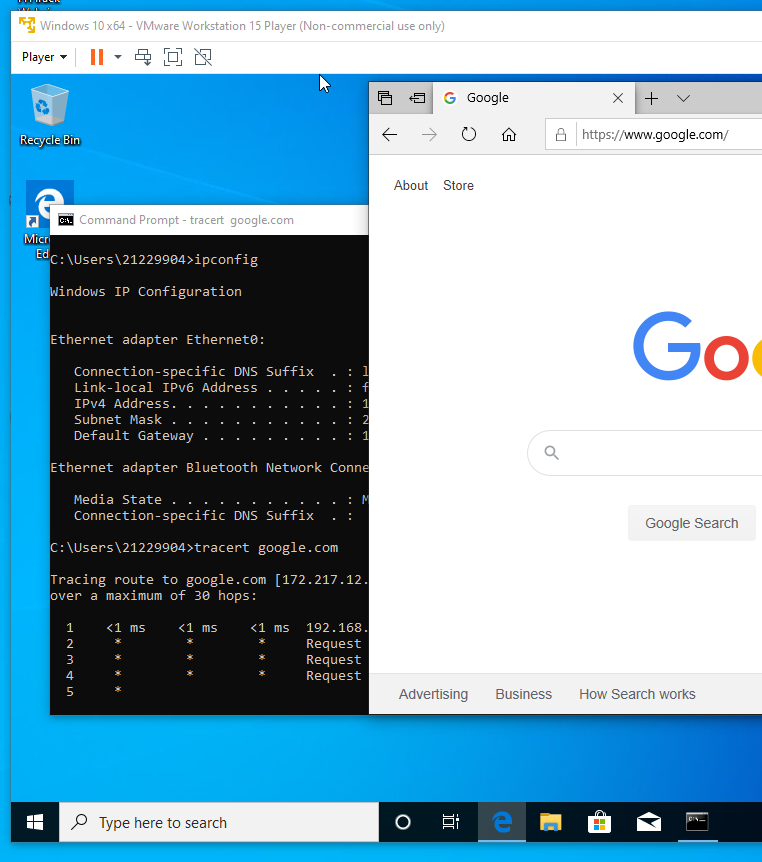
For my attempt, I selected Windows 10 and Ubuntu as the two VM OS’s to install. The hypervisor choice was VMware Workstation. This was a bit of a departure from my normal approach to sandboxed environments. I’m usually a fan of using Docker and/or Vagrant depending on what I’m doing. So with out understanding what I’ve done in the past, I’ve dabbled using VM’s a time or two. (I’ve on occasion had to use VirtualBox as my provisioner instead of Docker.)

Overall the experience was a bit different. I’m used to the idea of treating the newly created instances like cattle. Spin them up, spin them down. No permenance attached to the instance. It was a bit of a departure to be setting something up with some staying power. The next departure was the control allowed. In Vagrant I can customize the boxes but rarely did because these things were made to be lightweight. That degree of tuning just wasn’t needed. With Docker the only tuning I ever did was allocating more cores/memory to the Docker daemon.

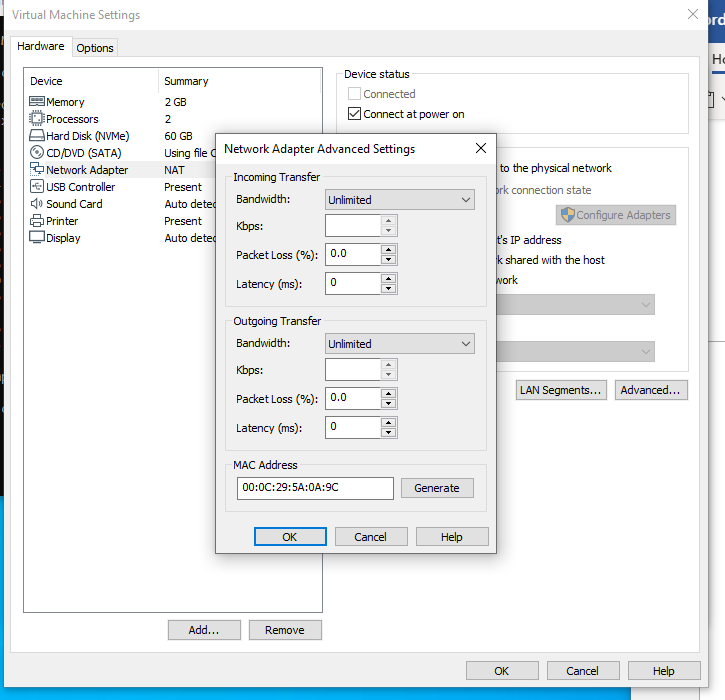
With VMware, I was able to have an incredible amount of control.

The networking stuff had me intrigued. I intuit that the guest VM’s don’t have the ability to know that they are VM’s. That the hardware is abstracted away from the OS. What I had problems being able to understand is how the guest VM’s were networked together. So each VM was given it’s own IP and the hypervisor acted as the gateway? Why didn’t tracert work on the Windows guest VM but did on my host machine? Things that I wish I had enough time to play with and understand but just don’t have a 25th hour to devote to it. (My guess is that the VMware Hypervisor has some firewall setting and isn’t some inherent VM type issue. But I don’t know that for myself yet.)

Guest & Host screenshots of tracert



I am utterly stoked on the idea of being able to choke a VM out. Either by CPU, memory or network. Most of the chaos type testing I’ve done has been either killing instances completely, breaking things at the load balancer level, or running specific commands within the instance to simulate load. Not always has it covered real life scenario's that come up in production. This looks like it could be handy for simulating a scaled load test. (I’m sure there are multiple other ways to do the same thing but I would want to explore this just to get a better understanding on my own.)



1. **While you are completing the exercise, post any questions or problems you encounter. Additionally, consider what you are doing. You are actually going to be hosting at least three operating systems. The host machine and two client virtual machines. What are the implications of this? What considerations do you need to make for the hardware? What kind of administration/management would need to be in place to keep all of these optimized? What kind of security? What websites or references did you use to help you? Explain each of these answers in your submission.**

The implication is that with enough hardware and low network latency you can abstract a lot of your workload into the cloud. Instead of having yourself limited by specific racks of hardware you just have a pool of resources that you can dynamically grow and shrink as needed. Especially when you start getting into the enterprise level VMware stuff around pinning vm’s to host and that level of tuning.

For the hardware we have to look at the overall CPU/mem/disk utilization of the cluster instead of just the guest VM’s. Ideally individual VM’s will have lots of low usage periods but when you average out the total of them it smooths out into a predictable amount. This way you can scale your clusters to handle the expected workload with some wiggle room and you don’t have a lot of computers sitting idly by for the majority of their existence. But that degree of success is \*highly\* tied to people understanding their infrastructure needs and infrastructure being so reliable that it never is an issue. An immense cost savings can be had but the risks have to be designed for.

Goodhart’s Law applies here. If management focuses just on idle cpu, memory, disk, etc, etc... You run the risk of having an incredibly fragile tech stack. Where before you can scale up, you hit capacity in your private cloud. Or you start having to pay due to the missed oppurtunity of reserved VM’s. Important decisions have to be made in the context of the full impact that’ll occur and not just the short term impact. Shutting down idle VM’s is great until you cripple your data pipeline because you weren’t able to scale up fast enough. Deletion of aged snapshots to save cloud spend is great until your company’s “Project Phoenix” release goes badly and you need to revert.

Google, VMware’s own knowledgebase, and a ton of StackExchange posts were the primary sources for me. I’m sure I committed several errors and missed who knows how many other avenues of exploration. I would love to go through some of the certifications out there for Cloud related tech and learn more. VMware has it’s certification pathway. Barring that, deep diving into Azure, GCP, or AWS would yield some better understanding of VM’s and the ecosystem they exist in. Ideally, I would focus on Kubernetes since I keep getting headhunters reference it. I’ve got no burning desire for that but I really do enjoy being employed so best to aim towards the parts of the ecosystem that others are already heading towards.