几个问题:

1. 12台机器的配置是什么？什么系统,其中9台计算节点，多少的内存，多少的CPU；

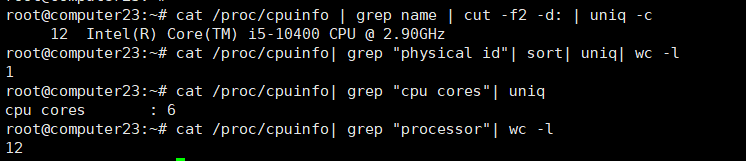
3台服务节点，1000Mbps的处理能力。

12台机器的配置都相同:

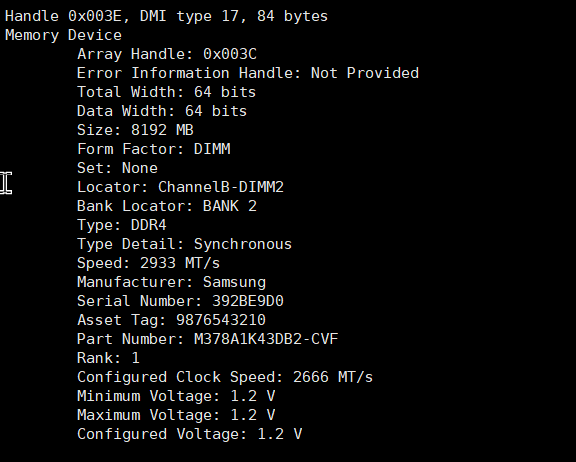
系统均是ubuntu 18.04.1,内核版本5.4.0



CPU都是intel i5-10400 CPU, 1个物理CPU,6个core,12个逻辑cpu



内存:每个机器两个相同的内存条,每个内存条8G,共16G,DDR4



1. 我们部署的虚拟机实例是哪里的，有哪些实例？Openstack的版本号是多少？

虚拟机用到的实例是我利用**virt-manager** 和 **virt-install,**参考以下这些资料,基于ubuntu18.04.01, 替换了里面的apt,pip源,并安装了一些iperf3等测试用到的软件做的qemu镜像

<https://www.cnblogs.com/dongzhanyi123/p/13301029.html>

<https://docs.openstack.org/image-guide/obtain-images.html>

<https://docs.openstack.org/image-guide/ubuntu-image.html>

然后利用glance的服务,将做好的镜像部署到系统中

openstack image create "ubuntu\_sdnlab" --file ./ubuntu\_1804\_bionic\_sdnlab.qcow2

在创建vm的过程中,通过修改vm的创建脚本,在vm中启动6个Iperf进程,对应端口25000-25005用于发包测试,并在对应租户的security group中的允许25000-25005端口收发的tcp,udp的包,用于iperf收发包测试,tcp的22端口用于远程ssh连接,并允许icmp用于ping测延时

所有的虚拟机用到的镜像都是这同一个,vm间的差别主要是部署方案中描述的vcpu和ram的区别

openstack使用最新的victoria版本(2020.12.9的包)

1. 我们的实验一共四个部署方案,分别是LLF,SJF,WRR,OUR
2. 8个租户，可能任意一个租户变为恶意租户发生攻击，对每一个系统的带宽影响？

具体的指标

1. 攻击了几个系统？(期望OUR攻击系统数小于其他三个,发包端的数目,别人的是2或3,我们的是1或2,每个租户称为恶意租户是攻击服务节点,数目)

表格: LLF, SJF, WRR, OUR (0,1,2,3)

租户0攻击系统数

租户1攻击系统数

租户2攻击系统数

…

note: 3+3+1, 3(单位个)

sys0被攻击的vm数, sys1被攻击的vm数, sys2被攻击的vm数,被攻击的sys数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 租户 | LLF | SJF | WRR | OUR |
| 0 | 3+3+1, 3 | 3+3+1, 3 | 1+3+3, 3 | 3+4+0, 2 |
| 1 | 1+3+2, 3 | 0+3+3, 2 | 3+2+1, 3 | 6+0+0, 1 |
| 2 | 3+2+2, 3 | 3+3+1, 3 | 3+1+3, 3 | 0+7+0, 1 |
| 3 | 2+2+3, 3 | 3+1+3, 3 | 2+2+3, 3 | 0+0+7, 1 |
| 4 | 3+2+2, 3 | 1+4+2, 3 | 3+1+3, 3 | 0+7+0, 1 |
| 5 | 0+3+2, 2 | 2+2+1, 3 | 1+1+3, 3 | 0+0+5, 1 |
| 6 | 1+2+3, 3 | 3+2+1, 3 | 2+3+1, 3 | 6+0+0, 1 |
| 7 | 2+3+2, 3 | 3+1+3, 3 | 2+3+2, 3 | 0+0+7, 1 |

1. 系统的可用带宽由多少缩减为多少？

note: 300,250,320(单位Mbps),记录第30s时候的带宽和,前20s攻击前,20-40s攻击,40-60s恢复,然后结束

1. 或者说对每一台虚拟机而言，造成了多少的性能的下降？
2. 我们的算法和对比算法受攻击后，FCT的变化，是否会有显著的区别；

note: fct受到攻击的的数据是在原来的fct\_common的数据上，在不同的租户攻击时，属于该租户部署vm的sys的所有带宽整体乘0.7跑出来的

1. 我们的算法和对比算法的服务节点失效，FCT的变化，是否有显著的区别；

note: fct实效的数据是在额外跑的一组fct\_common的数据上，对不同的sys，在30s的时候offline 60s得到的数据