

Cybersecurity Graph Analytics

Cookbook

HPE Superdome Flex

Contents

[Overview 5](#_Toc31722494)

[HPE Superdome Flex PoC System Architecture 6](#_Toc31722495)

[HPE Superdome Flex Health Check 6](#_Toc31722496)

[Install RHEL 7.6 9](#_Toc31722497)

[Network configuration 9](#_Toc31722498)

[PoC tools deployment and configuration 10](#_Toc31722499)

[Automated tools deployment and configuration 10](#_Toc31722500)

[Manual tools deployment and configuration 12](#_Toc31722501)

[System Hardware Performance Validation 15](#_Toc31722502)

[Performance Metrics Collector: CPU, Disk Read , Disk Write , Memory , Network Testing 15](#_Toc31722503)

[Install and configure Graph Analytics Server Software (xGT) package from Trovares 17](#_Toc31722504)

[Implement Cybersecurity Graph Analytics for 3TB Dataset (Optional) 19](#_Toc31722505)

[Move 3TB LANL Dataset from Remote Source to Superdome Flex NVMe Data Drives 19](#_Toc31722506)

[Develop Lateral Movement Threat Detection using Trovares xGT 21](#_Toc31722507)

[Measure Query Performance against Lateral Movement Threat Detection 21](#_Toc31722508)

[Implement Cybersecurity Graph Analytics for 10TB Dataset 22](#_Toc31722509)

[Ingest 10TB Dataset from Remote Sources to Superdome Flex NVMe Data Drives 22](#_Toc31722510)

[Appendix A – Sample Cybersecurity Threat Detection Notebook 23](#_Toc31722511)

[Appendix B – Pre-Requisite Setup Cookbook Script 29](#_Toc31722512)

# 

# Overview

This cookbook provides end-to-end implementation details on deploying Trovares xGT Cybersecurity Graph Analytics PoC infrastructure at Customer location using HPE Superdome Flex server. Trovares is a Seattle-based software company creating a new generation of property graph analytics tools and this POC is targeted towards the cybersecurity use case – Detecting Lateral Movement Cyber Threat using Trovares graph analytics functionality.

Deployment assumptions:

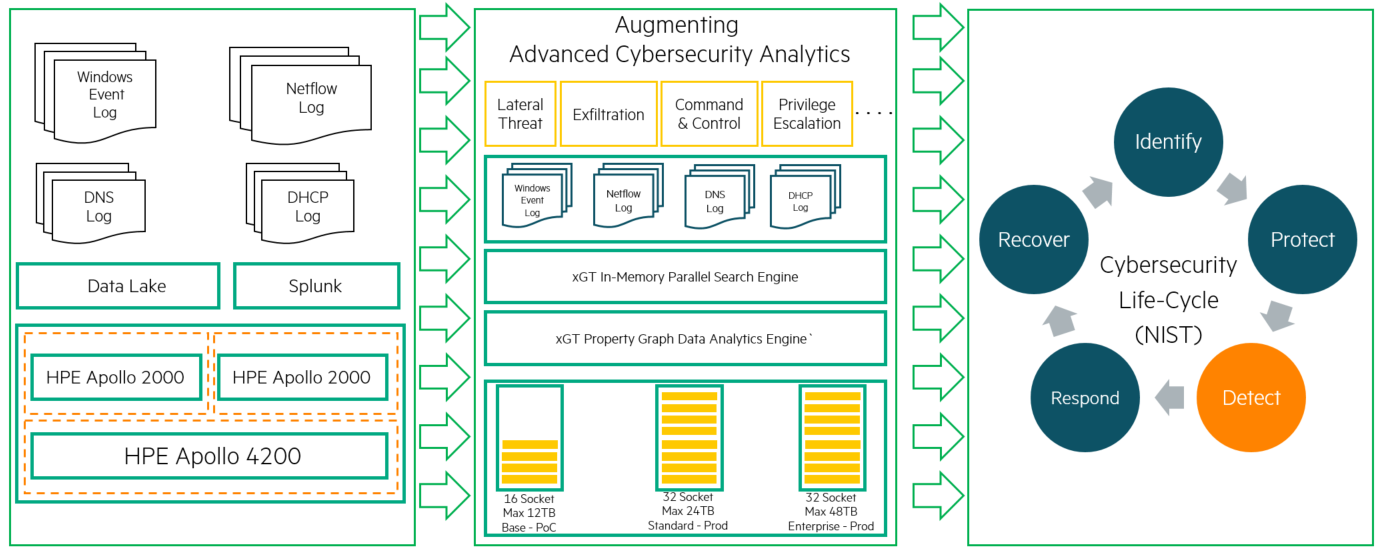
* HPE Superdome Flex server will be pre-built at the HPE factory with a single partition with 4 chassis configuration.
* Pointnext engineer is expected to do the following at the Customer location:
* Deploy the hardware and assign/change IP addresses of the RMC.
* Verify the partition configuration as mentioned in this document.
* Deploy RHEL 7.6, tools and configure the system.
* This cookbook should be considered as complement to Superdome Flex Installation and Linux deployment guides

Target audience: The target audience for this Cookbook is HPE Pointnext deployment / implementation engineer and Trovares engineer.

Document purpose: The purpose of this document is to enable field personal to deploy the proof of concept system at customer location and perform validation.

# HPE Superdome Flex PoC System Architecture

Following is the architecture that is considered for current PoC Cookbook.



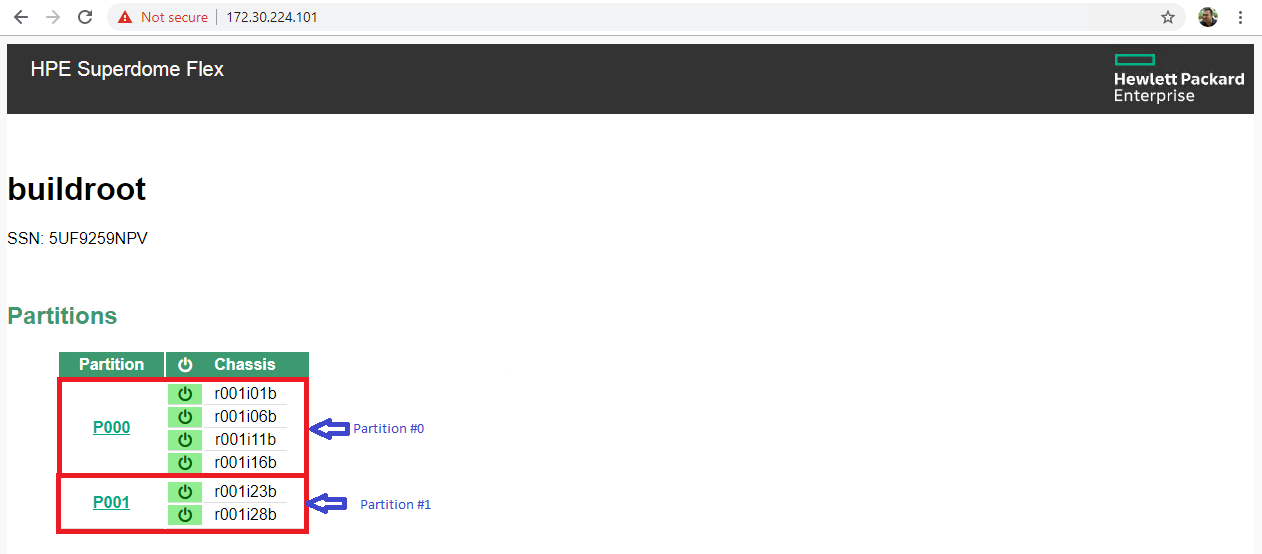
Current cookbook will be deployment with below configuration referred as PoC Environment #2.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | PoC Environment #1 ( Partition #0 ) | | | | PoC Environment #2 ( Partition #1 ) | |
| Chassis No | 1 | 2 | 3 | 4 | 5 | 6 |
| BMC | r001i01b | r001i06b | r001i11b | r001i16b | r001i23b | r001i28b |
| IP Address | 172.30.224.102 | 172.30.224.103 | 172.30.224.104 | 172.30.224.105 | 172.30.224.106 | 172.30.224.107 |
| Credentials | - | - | - | - | - | - |
| Cores | 112 | 112 | 112 | 112 | 112 | 112 |
| Memory(GB) | 3072 | 3072 | 3072 | 3072 | 3072 | 3072 |
| Storage(OS+App) |  |  |  |  |  |  |
| Storage(Data) | /dev/nvme0n1 : 1600.3 GB /dev/nvme1n1 : 1600.3 GB /dev/nvme2n1 : 1600.3 GB /dev/nvme3n1 : 1600.3 GB /dev/nvme4n1 : 1600.3 GB /dev/nvme5n1 : 1600.3 GB /dev/nvme6n1 : 1600.3 GB /dev/nvme7n1 : 1600.3 GB | /dev/nvme8n1 : 1600.3 GB /dev/nvme9n1 : 1600.3 GB /dev/nvme10n1 : 1600.3 GB /dev/nvme11n1 : 1600.3 GB /dev/nvme12n1 : 1600.3 GB /dev/nvme13n1 : 1600.3 GB /dev/nvme14n1 : 1600.3 GB /dev/nvme15n1 : 1600.3 GB | /dev/nvme16n1 : 1600.3 GB /dev/nvme17n1 : 1600.3 GB /dev/nvme18n1 : 1600.3 GB /dev/nvme19n1 : 1600.3 GB /dev/nvme20n1 : 1600.3 GB /dev/nvme21n1 : 1600.3 GB /dev/nvme22n1 : 1600.3 GB /dev/nvme23n1 : 1600.3 GB | /dev/nvme24n1 : 1600.3 GB /dev/nvme25n1 : 1600.3 GB /dev/nvme26n1 : 1600.3 GB /dev/nvme27n1 : 1600.3 GB /dev/nvme28n1 : 1600.3 GB /dev/nvme29n1 : 1600.3 GB /dev/nvme30n1 : 1600.3 GB /dev/nvme31n1 : 1600.3 GB | /dev/nvme0n1 : 1600.3 GB /dev/nvme1n1 : 1600.3 GB /dev/nvme2n1 : 1600.3 GB /dev/nvme3n1: 1600.3 GB /dev/nvme4n1: 1600.3 GB /dev/nvme5n1: 1600.3 GB /dev/nvme6n1: 1600.3 GB /dev/nvme7n1: 1600.3 GB | /dev/nvme8n1: 1600.3 GB /dev/nvme9n1: 1600.3 GB /dev/nvme10n1: 1600.3 GB /dev/nvme11n1: 1600.3 GB /dev/nvme12n1: 1600.3 GB /dev/nvme13n1: 1600.3 GB /dev/nvme14n1: 1600.3 GB |

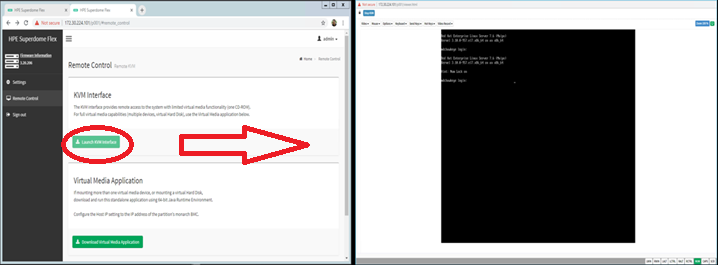
# HPE Superdome Flex Health Check

After the physical deployment of the hardware, perform the following validations of the environment.

1. Perform Web RMC login using web browser for validation of shipped configuration. RMC Web Console: https://172.30.224.101/ (user : administrator, password : Password!234)



1. Login to Partition P001 and perform console login to host using “Launch KVM Interface” option



1. Perform shell login to RMC to gather chassis details.

NOTE:

To restart any particular partition, issue following command (if needed)

RMC:r001i21c cli> power off npar pnum=1 force

RMC:r001i21c cli> power on npar pnum=1

1. Get detailed view of all the partitions, chassis and devices status

RMC: r001i21c cli> show chassis info

1. List summarized status of each partition and each chassis

RMC:r001i21c cli> show chassis list

BMCs: 6

BMC Rack UPos Par Power Health CPUs CPU Cores DIMMs Memory (GiB) IO Cards Base-IO

Num Num State Status OK/In OK/In OK/In RAM/PMem OK/In

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

r001i01b 001 01 p0 On OK 4/4 112/112 24/24 3071/0 9/9 Active

r001i06b 001 06 p0 On OK 4/4 112/112 24/24 3072/0 9/9 Present

r001i11b 001 11 p0 On OK 4/4 112/112 24/24 3072/0 9/9 Present

r001i16b 001 16 p0 On OK 4/4 112/112 24/24 3072/0 9/9 Present

r001i23b 001 23 p1 On OK 4/4 112/112 24/24 3071/0 9/9 Active

r001i28b 001 28 p1 On OK 4/4 112/112 24/24 3072/0 10/10 Present

\* OK/In = OK/Installed

RMC:r001i21c cli>

1. Monitor health of each partition including no. of partitions, Cores/CPUs per partitions, Hyper-Threading Status

RMC:r001i21c cli> show npar verbose

This system is nPartition capable

Partitions: 2

Partition 0:

Run State : OS Boot

Health Status : OK

Chassis OK/In : 4/4

CPUs OK/In : 16/16

Cores OK/In : 448/448

DIMMs OK/In : 96/96

IO Cards OK/In : 36/36

Hyper-Threading : On

RAS : On

Boot Chassis : r001i01b

Boot Slots : 3,5

Secure Boot : Off

Secure Boot Next : Off

Volatile Memory : 12287 GiB

Persistent Memory : 0 GiB

Partition 1:

Run State : OS Boot

Health Status : OK

Chassis OK/In : 2/2

CPUs OK/In : 8/8

Cores OK/In : 224/224

DIMMs OK/In : 48/48

IO Cards OK/In : 19/19

Hyper-Threading : On

RAS : On

Boot Chassis : r001i23b

Boot Slots : 3,5

Secure Boot : Off

Secure Boot Next : Off

Volatile Memory : 6143 GiB

Persistent Memory : 0 GiB

\* OK/In = OK/Installed

# Install RHEL 7.6

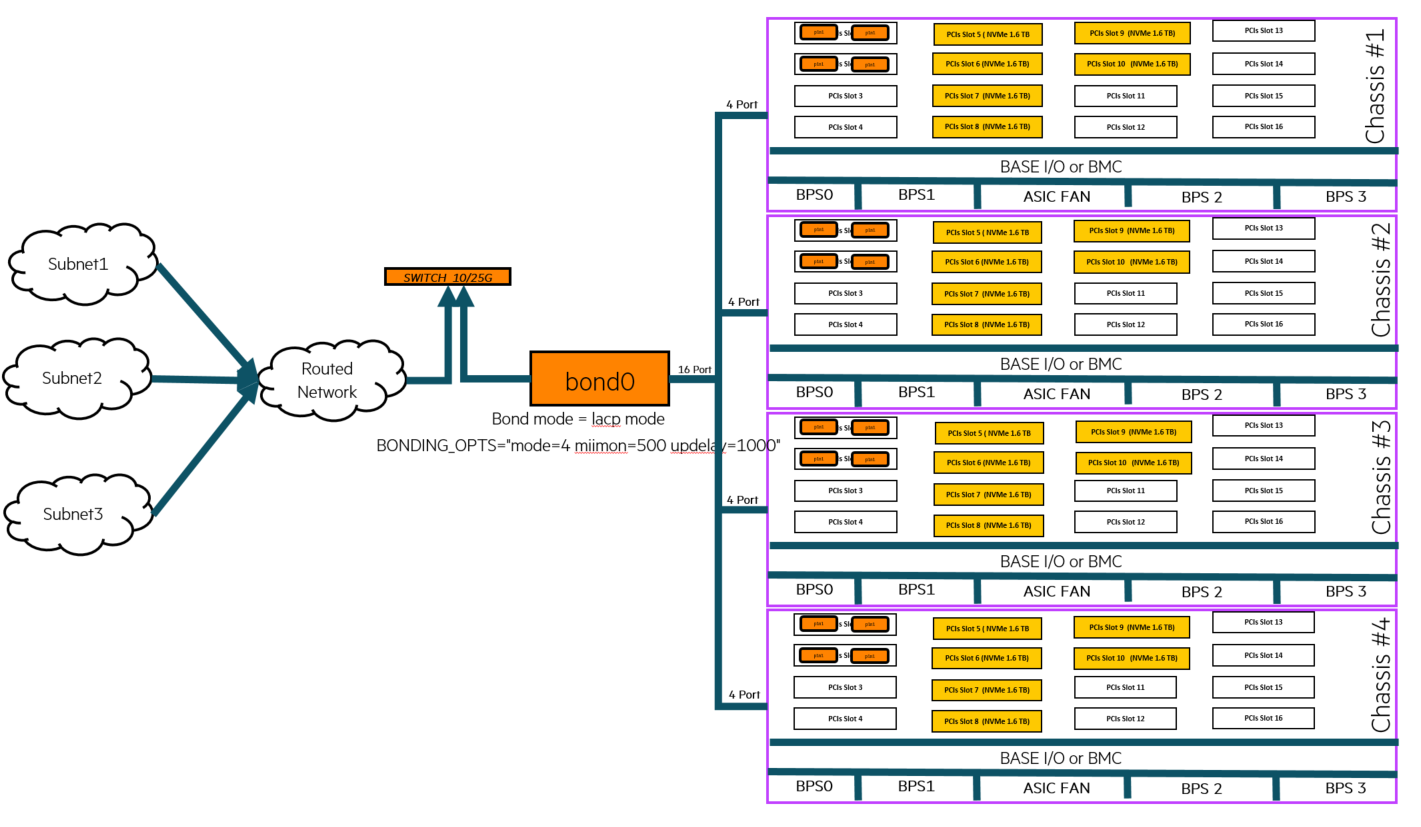
Follow the OS installation steps described in [Redhat Installation Guide for Superdome Flex](https://support.hpe.com/hpesc/public/docDisplay?docId=a00058577en_us) and install OS with below mentioned partition size. Allocate two separate disks for Trovares and Reservoir Lab bits and tools. Below are the recommended method for installing RHEL OS.

1. Configure O/S SSDs with RAID 1 then proceed with installation
2. Preferred RHEL installation type is Infrastructure Installation
3. VNCServer and VNCClient utilities should be installed and configured
4. Static IP Configuration option should be preferred for current demo

Table 1. OS filesystem – recommended size

|  |  |  |
| --- | --- | --- |
| Size | Mount Point | File System |
| 500MB | /boot/efi | **VFAT** |
| 2G | /boot | XFS |
| 300GB | / | XFS |
| 200GB | /tmp | XFS |
| 400GB | /var | XFS |
| 32GB | Swap | SWAP |
| 500 GB | /home | XFS |
| 1.8 TB [ Separate disk ] | /trovares\_POC | XFS |
| 1.8 TB [ Separate disk ] | /ReservoirLab\_POC | XFS |

# Network configuration

Manually create a teaming interface using 2 \* 25/10G network card interface connected to same network switch where customer’s staging server is connected to. Follow the standard RHEL network configuration procedure and use “round-robin” mode. Staging server is where customer data is located in the form of \*.csv files. These files have to be moved to Trovares POC system for cybersecurity analysis. 

NOTE:

If customer staging system is in different subnet (routed network) than the Trovares POC system, the data (\*.csv files) movement could take longer time. In HPE’s test environment, data movement using rsync without encryption gave ~300 MB/s speed over 10 GbE network (target connected to same L2 switch). With this speed, 10TB data movement could take around ~ 10 hours from staging server to Trovares POC system.

# PoC tools deployment and configuration

Tools deployment and configuration of the PoC setup can be performed either using automation tool or manually. Automated deployment tool does not require internet access for package deployment as the shipped SDFlex\_Trovares\_RL\_POC.tgz contains required repos as well, except RHEL OS & extension repo. Manual deployment requires internet access for python, jupyter, xGT python package deployment.

## Automated tools deployment and configuration

Follow the below steps to prepare the trovers POC system using automation tool. Approximate script execution time is around 20 minutes.

1. Download sdflex\_trovares\_POC.tgz file from below FTP location:

ftp://<TBD>/SDFlex\_Trovares\_RL\_POC.tgz

1. Unzip it using below command as root user. It will get unzipped at “/trovares\_POC”

# tar xvz --directory=/ -f SDFlex\_Trovares\_RL\_POC.tgz

1. Below are the files and directories located under “/trovares\_POC”

# ls /trovares\_POC

hpefsw configure\_rsync\_at\_customer\_staging\_server.sh jupyter python rsync\_mover.sh system\_prereq.sh xgt

Table 2. Description of scripts and repo packaged in SDFlex\_Trovares\_RL\_POC.tgz

|  |  |
| --- | --- |
| Component | Description |
| /trovares\_POC/hpefsw | Contains HPE Foundation Software ISO image.  hpe-foundation-2.2-cd1-media-rhel76-x86\_64.iso |
| /trovares\_POC/configure\_rsync\_at\_customer\_staging\_server.sh | What: This script installs (if required) and configures rsync software at customer server where CSV files have been staged for analysis.  Where to run: This script should be only run on customer server where CSV files have been staged. Do not run it on trovares POC system.  NOTE: If rsync is not installed, it will try to install using RHEL yum repository configured in the system. Optionally rsync rpm can be installed manually and then this script can be run. |
| /trovares\_POC/rsync\_data\_mover.sh | What: This script installs (if required) and configures rsync software at trovares POC system. This script moves only \*.csv file from customer staging server to trovares POC system.  Where to run: This script should be only run on the trovares POC system.  NOTE: If rsync is not installed, it will try to install using RHEL yum repository configured in the system. Optionally rsync rpm can be installed manually and then this script can be run. |
| /trovares\_POC/system\_prereq.sh | What: This script installs (if required) and configures various softwares and tunes the kernel & NVMe disks. Please refer to step 4 for more details on this script.  Where to run: This script should be only run on the trovares POC system. |
| /trovares\_POC/jupyter | Jupyter package pip repository |
| /trovares\_POC/xgt | xGT python package pip repository |
| /trovares\_POC/python | Python source code |

1. Setup yum repository for RHEL OS 7.6
   1. Mount the RHEL ISO image and create yum repository.
   2. # mkdir /rhelmnt
   3. # mount –o loop <RHEL76.ISO> /rhelmnt
   4. # vi /etc/yum.repos.d/rhel76.repo

[rhel76]

name=rhel76

baseurl=file:///rhelmnt

enabled=1

gpgcheck=0

1. Execute the system\_prereq.sh script in the below mentioned sequence on the trovares POC system.

|  |  |  |
| --- | --- | --- |
| Step | Command syntax | Description and pre-requisite before running the command |
| Step-1 | ./system\_prereq.sh -f | Installs HPE Foundation Software for HPE Superdome Flex. Before running this command, make sure you have configured RHEL OS yum repo as described in previous step. Before running this command, define the following variables inside the script as shown below:  TROVARES\_POC\_HOME="/trovares\_POC" NOTE: Directory where SDFlex\_Trovares\_RL\_POC.tgz was extracted.  HPEFS\_MNT\_DIR="/hpefswmount" |
| Step-2 | ./system\_prereq.sh -m | * Creates filesystem on specified NVME disks. Before running this command, define the following variable inside the script as shown below.   NVME\_DISK\_LIST=(/dev/nvme0n1 /dev/nvme1n1 /dev/nvme2n1)   * Create mount points for all the NVMe disks as specified in the below variable (located inside the script). Update following variables, if required.   NVME\_MNT\_DIR="/nvme\_data"  NVME\_DEVICE\_NAME\_PATTERN="/dev/nvme0n\*"   * Updates /etc/fstab * Finally mount all the filesystems. For example: /dev/nvme0n1 on /nvme\_data0, /dev/nvme1n1 on /nvme\_data1 and so on. |
| Step-3 | ./system\_prereq.sh -d | Tunes the following parameter for all the NVMe disks mentioned in the script variable NVME\_DISK\_LIST.   * read\_ahead\_kb * nr\_requests * nomerges * rq\_affinity |
| Step-4 | ./system\_prereq.sh -s | Performs the following operations. You may need to reboot the server after this server in order to activate the configuration changes.   * Sets “permissive” to SELINUX * Setup limits.conf * Setup /etc/systemctl.conf * Disables the firewall * Sets the maximum frequency for the CPUs |
| Step-5 | ./system\_prereq.sh -n | Configure NTP client. Before running this command, configure below variable in this script pointing to your organization NTP server.  NTPSVR="server 0.pool.ntp.org" |
| Step-6 | ./system\_prereq.sh -j | Installs Java from RHEL OS Repo and configures PATH variable in the ~/.bash\_profile |
| Step-7 | ./system\_prereq.sh -p | Installs python package (Python-3.7.4.tar.xz) from local PIP repo @ /trovares\_POC/python and configures virtual env. |
| Step-8 | ./system\_prereq.sh -x | Installs xGT python package from local PIP repo @ /trovares\_POC/xgt |
| Step-9 | ./system\_prereq.sh -z | Installs Jupyter notebook server from local PIP repo @ /trovares\_POC/jupyter and configures it |
| Any time | ./system\_prereq.sh -r | Generate SAR reports for the following compute resources.   * CPU * Memory * Swap * Load average * Disk I/O Transfer * Process/Context Switching * Network interface * Sockets   Before running this command, configure below variable pointing to directory in which SAR reports will be saved. Below is the default location and script will automatically create the directory, if it does not exist already.  REPORT\_DIR="/hpeSarReport"  NETWORK\_INTERFACE=ens33 |

## Manual tools deployment and configuration

1. Configure YUM repositories for HPE Foundation Software, RHEL OS, EPEL (optional for PoC) and RHEL-Supp (optional for PoC). Following are example YUM repositories configuration. Download HPE Foundation Software from [here](https://support.hpe.com/hpsc/swd/public/detail?swItemId=MTX_2f5a467fecfd4d0eb1fd0b49a2). Highlighted line-items would need to be altered as per the proposed deployment environment.

[root@mdchawkeye yum.repos.d]# cat \*.repo

[EPEL7]

name=EPEL7

baseurl=http://172.30.1.89:8094/EPEL7

enabled=1

gpgcheck=0

[HPEFS]

name=HPEFS

baseurl=http://172.30.224.99/HPEFS/

enabled=1

gpgcheck=0

[RHEL76]

name=RHEL76

baseurl=http://172.30.224.99/RHEL76/

enabled=1

gpgcheck=0

[RHEL76-Supp]

name=RHEL76-Supp

baseurl=http://172.30.1.89:8094/RHEL76-Supp

enabled=1

gpgcheck=0

[root@mdchawkeye yum.repos.d]# yum repolist

Loaded plugins: langpacks, product-id, search-disabled-repos, subscription-manager

This system is not registered with an entitlement server. You can use subscription-manager to register.

repo id repo name status

EPEL7 EPEL7 14,613

HPEFS HPEFS 33

RHEL76 RHEL76 5,152

RHEL76-Supp RHEL76-Supp 18

repolist: 19,816

1. Configure and validate hostname

Set hostname to a unique name

[root@mdchawkeye ~]# hostnamectl set-hostname mdchawkeye.localdomain

Verify the hostname

[root@mdchawkeye ~]# hostname -f

Edit /etc/hosts with the IP addresses and fully qualified domian name

[root@mdchawkeye ~]# cat /etc/hosts

127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4

::1 localhost localhost.localdomain localhost6 localhost6.localdomain6

172.30.224.100 mdchulk mdchulk.localdomain

172.30.224.99 mdchawkeye mdchawkeye.localdomain

Edit /etc/sysconfig/network with the FQDN of this host only

[root@mdchawkeye ~]# vi /etc/sysconfig/network

HOSTNAME=mdchawkeye.localdomain

Run uname -a and check that the hostname matches the output of the hostname command

[root@mdchawkeye ~]# uname –a

1. Configure /etc/resolv.conf as per customer setup.
2. Install HPE Foundation Software using yum repo already configured. Check memlog service is running once the HPEFS is installed.

[root@mdchawkeye ~]# yum groupinstall "HPE Foundation Software"

[root@mdchawkeye ~]# systemctl status memlog

[root@mdchawkeye ~]# memlogd -s

[root@mdchawkeye ~]# grep MEMLOG /var/log/messages

1. Configure CPU frequency scaling

Get the available CPU frequencies:

max\_freq=$(for i in $(cat /sys/devices/system/cpu/cpu0/cpufreq/scaling\_available\_frequencies); do echo $i; done | sort -nr | head -1)

Set the CPU Max Frequency:

for cpu\_max\_freq in /sys/devices/system/cpu/cpu\*/cpufreq/scaling\_max\_freq; do echo $max\_freq > $cpu\_max\_freq; done

1. Disable firewall

# systemctl disable firewalld

# systemctl stop firewalld

# systectl status firewalld

1. Configure limits.conf

# vi /etc/security/limits.conf

\* soft nproc 65536

\* hard nproc 65536

\* soft nofile 262144

\* hard nofile 262144

1. Configure /etc/sysctl.conf

# vi /etc/sysctl.conf

kernel.sem = 250 32000 100 128

kernel.shmall = 2097152

kernel.shmmax = 2147483648

fs.file-max = 65536

kernel.shmmni = 4096

fs.aio-max-nr = 1048576

net.ipv4.ip\_local\_port\_range = 1024 65000

net.core.rmem\_default = 4194304

net.core.wmem\_default = 2621444

net.core.rmem\_max=4194304

net.core.wmem\_max=262144

net.ipv4.tcp\_rmem = 1048576 1048576 4194304

net.ipv4.tcp\_wmem = 1048576 1048576 1048576

net.ipv6.conf.default.disable\_ipv6 = 1

net.ipv6.conf.all.disable\_ipv6 = 1

kernel.nmi\_watchdog=0

vm.swappiness=1

#sysctl –p

1. Configure SELINUX to permissive.

# vi /etc/selinux/config file

SELINUX=permissive

1. Disable tuned

# tuned-adm off

# tuned-adm list ( check there are no active profiles)

# systemctl stop tuned

# systemctl disable tuned

1. Disable transparent huge page. To disable transparent hugepages on reboot, configure /etc/rc.d/rc.local as shown below and reboot.

# vi /etc/rc.d/rc.local

echo never > /sys/kernel/mm/transparent\_hugepage/enabled

echo never > /sys/kernel/mm/transparent\_hugepage/defrag

# chmod +x /etc/rc.d/rc.local

# reboot

1. Create filesystem on NVMe disks. Make sure there are no partition existing in the disk(s). If there is any partition, remove it first and then create XFS filesystem on the disk. Repeat every NVMe disk that have to be deployed for PoC purpose.

# fdisk –l /dev/nvme0n1

# mkfs.xfs /dev/nvme0n1

1. Create mount point for the NVMe disks. It is recommended to mention NVMe device number in the mount point name. It helps to manage disks mountpoint better. For example:

# mkdir /nvme\_data0 [ for NVMe disk /dev/nvme0n1 ]

# mkdir /nvme\_data1 [ for NVMe disk /dev/nvme1n1 ]

1. Update /etc/fstab with NVMe disk mount points. For example,

# echo “/dev/nvme0n1 /nvme\_data1 xfs noatime,nodiratime 0 0” >> /etc/fstab

1. Tune readahead to 8K for all the NVMe disk drives used for PoC. For example:

# blockdev --setra 8192 "/dev/nvme0n1"

1. Tune following parameters for all the NVMe disk drives used for PoC. For example:

# echo 4096 > /sys/block/nvme0n1/queue/read\_ahead\_kb

# echo 32 > /sys/block/nvme0n1/queue/nr\_requests

# echo 2 > /sys/block/nvme0n1/queue/nomerges

# echo 2 > /sys/block/nvme0n1/queue/rq\_affinity

1. Configure NTP client as per customer setup.
2. Install Java using RHEL OS repo

# yum install -y java-1.8.0-openjdk\*

1. Install python and virtualenv

# yum install gcc openssl-devel bzip2-devel libffi-devel sqlite-devel zlib-devel -y

# cd /tmp

# wget <https://www.python.org/ftp/python/3.7.4/Python-3.7.4.tar.xz>

# cd /root

# cp /tmp/Python-3.7.4.tar.xz .

# tar xvf Python-3.7.4.tar.xz

# cd Python-3.7.4

# ./configure --enable-optimizations

# make altinstall

# python3.7 -m pip install /trovares\_POC/virtualenv/virtualenv-16.7.9-py2\*

# virtualenv -p python3.7 Python\_3\_7

# source /root/Python\_3\_7/bin/activate

(Python\_3\_7) [root@localhost]# deactivate [[To come out of virtualenv environment]]

1. Install xGT python packages under virtualenv environment.

# source /root/Python\_3\_7/bin/activate

(Python\_3\_7) [root@localhost]# pip install grpcio

(Python\_3\_7) [root@localhost]# pip install protobuf

(Python\_3\_7) [root@localhost]# pip install xgt

1. Install Jupyter notebook package under virtualenv environment.

# source /root/Python\_3\_7/bin/activate

(Python\_3\_7) [root@localhost]# python3.7 -m pip install jupyter

(Python\_3\_7) [root@localhost]# jupyter notebook --ip <host\_ip\_address> --allow-root

# System Hardware Performance Validation

## Performance Metrics Collector: CPU, Disk Read , Disk Write , Memory , Network Testing

1. Install SAR Utility

yum install sysstat

1. Generate SAR Report

export LC\_TIME="POSIX"

sample\_interval = 1

number\_of\_samples = 10

# CPU

sar -u $sample\_interval $number\_of\_samples | grep -v -E "CPU|Average|^$" > data/cpu.dat &

# CPU DETAILED CSV FORMAT

sar -P ALL $sample\_interval $number\_of\_samples | grep -v -E "Linux|Average"| awk '{if ($0 ~ /[0-9]/) { print $1","$2","$3","$4","$5","$6","$7","$8","$9; } }' >data/CPU.csv &

# RAM

sar -r $sample\_interval $number\_of\_samples | grep -v -E "[a-zA-Z]|^$" > data/ram.dat &

#RAM DETAILED CSV FORMAT

sar -r $sample\_interval $number\_of\_samples | grep -v -E "Linux|Average"|\

awk '{if ($0 ~ /[0-9]/) { print $1","$2","$3","$4","$5","$6","$7","$8","$9","$10","$11","$12; } }'>data/RAM.csv &

# Swap

sar -S $sample\_interval $number\_of\_samples | grep -v -E "[a-zA-Z]|^$" > data/swap.dat &

# Swap DETAILED CSV FORMAT

sar -S $sample\_interval $number\_of\_samples | grep -v -E "Linux|Average"| awk '{if ($0 ~ /[0-9]/) { print $1","$2","$3","$4","$5","$6","$7; } }' > data/swap.csv &

# Load average and tasks

sar -q $sample\_interval $number\_of\_samples | grep -v -E "[a-zA-Z]|^$" > data/loadaverage.dat &

# Load average and tasks DETAILED CSV FORMAT

sar -q $sample\_interval $number\_of\_samples | grep -v -E "Linux|Average"| awk '{if ($0 ~ /[0-9]/) { print $1","$2","$3","$4","$5","$6","$7","$8; } }' > data/loadaverage.csv &

# IO transfer

sar -b $sample\_interval $number\_of\_samples | grep -v -E "[a-zA-Z]|^$" > data/iotransfer.dat &

# IO transfer DETAILED CSV FORMAT

sar -b $sample\_interval $number\_of\_samples | grep -v -E "Linux|Average"| awk '{if ($0 ~ /[0-9]/) { print $1","$2","$3","$4","$5","$6","$7; } }' > data/iotransfer.csv &

# Process/context switches

sar -w $sample\_interval $number\_of\_samples | grep -v -E "[a-zA-Z]|^$" > data/proc.dat &

# Process/context switches DETAILED CSV FORMAT

sar -w $sample\_interval $number\_of\_samples | grep -v -E "Linux|Average"| awk '{if ($0 ~ /[0-9]/) { print $1","$2","$3","$4; } }' > data/proc.csv &

# Network Interface

sar -n DEV $sample\_interval $number\_of\_samples | grep $network\_interface | grep -v "Average" > data/netinterface.dat &

# Network Interface DETAILED CSV FORMAT

sar -n DEV $sample\_interval $number\_of\_samples | grep $network\_interface | grep -v "Linux|Average"| awk '{if ($0 ~ /[0-9]/) \

{ print $1","$2","$3","$4","$5","$6","$7","$8","$9","$10; } }' > data/netinterface.csv &

# Sockets

sar -n SOCK $sample\_interval $number\_of\_samples | grep -v -E "[a-zA-Z]|^$" > data/sockets.dat &

# Sockets DETAILED CSV FORMAT

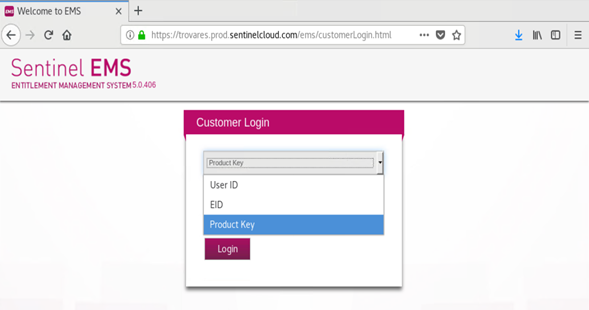
sar -n SOCK $sample\_interval $number\_of\_samples | grep -v -E "Linux|Average"| awk '{if ($0 ~ /[0-9]/) { print $1","$2","$3","$4","$5","$6","$7","$8; } }' > data/sockets.csv &

# Install and configure Graph Analytics Server Software (xGT) package from Trovares

This requires internet access at Customer location to download xGT package from Trovares public website. xGT server will be run in a hardware locked license. Follow below steps to get license entitlement from Trovares:

1. Retrieve entitlement grant from Trovares.
2. Login to entitlement management link:

<http://trovares.prod.sentinelcloud.com:80/ems/customerLogin.html>



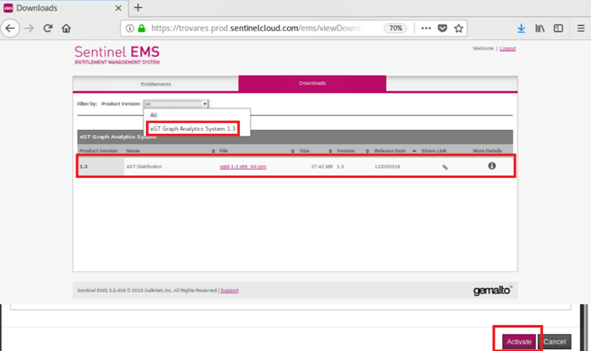
1. Use the assigned Entitlement Id/Product Key for license activation

Entitlement Id: 5d65fb56-9c55-46e7-a924-cc656679f46d (Example)

Product Key: 401b5242-c679-48ae-92d0-9d735f70f80e (Example)

1. Download and install xgtd-1-3.x86\_64 from the screen shown below:

Navigate to downloads tab and download xgtd-1.3 RPM.



1. Install the xgtd-1.3 rpm on the PoC system and setup path variables.

# rpm –ivh xgtd-1-3.x86\_64

1. Setup following environment variables for xgtd in ~/.bash\_profile

# vi ~/.bash\_profile

export PATH=$PATH:/opt/xgtd/bin

export PATH=$PATH:/opt/xgtd/lib

export PATH=$PATH:/opt/xgtd/util

export LSHOST="NO-NET"

export LSERVRC="/opt/xgtd/bin/lservrc"

export LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:/usr/local/lib:/opt/xgtd/lib

1. Run this executable /opt/xgtd/bin/echoid64 on target node to generate “RMS Client Info” values.

# /opt/xgtd/bin/echoid64

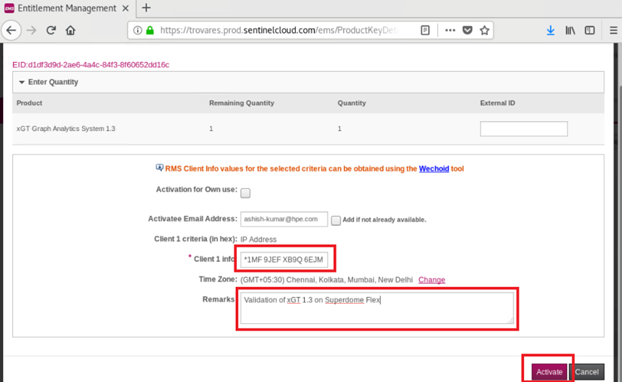
Sentinel RMS Development Kit 9.4.0.0023 Host Locking Code Information Utility

Copyright (C) 2018 SafeNet, Inc.

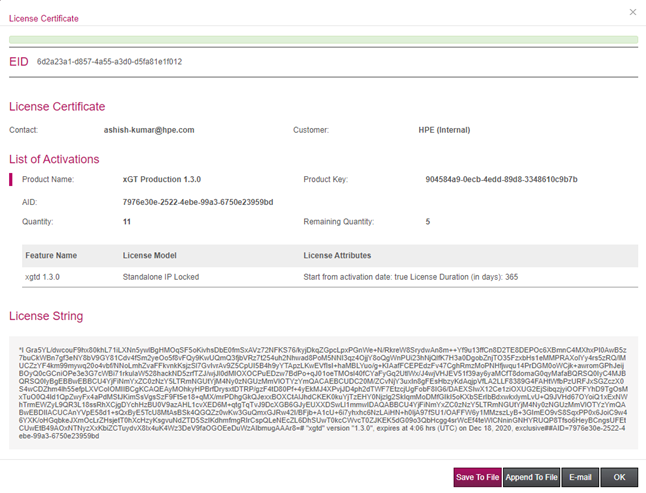
Locking Code 1 : 2-\*1MF 9JEF XB9Q 6EJM

Locking Code 1 (Old Style) : 2-2B9AB

1. Enter RMS Client Info in RMS Login screen and activate.



1. License certificate gets generated successfully. Save the license for future use.



# Implement Cybersecurity Graph Analytics for 3TB Dataset (Optional)

This step is intended only for functional validation of xGT setup and should not be considered as customer PoC implementation use-case.

We'll be using the LANL Unified Host and Network Dataset, a set of netflow and host event data collected on an internal Los Alamos National Lab network.

Our goal will be to turn about 3TB of CSV files into a single connected graph. Cybersecurity Analyst then provides the criteria for specific cyber threat patterns. Subgraph Isomorphism or pattern matching is performed on large memory graphs and sub-graphs are queried to detect attack infected hosts.

The Unified Host and Network Dataset is a subset of network and computer (host) events collected from the Los Alamos National Laboratory enterprise network over the course of approximately 90 days. The host event logs originated from most enterprise computers running the Microsoft Windows operating system on Los Alamos National Laboratory's (LANL) enterprise network. The network event data originated from many of the internal enterprise routers within the LANL enterprise network.

## Move 3TB LANL Dataset from Remote Source to Superdome Flex NVMe Data Drives

This dataset is made available by [Los Alamos National Laboratory (LANL)](https://www.lanl.gov/). LANL has placed this data into the *public domain*. The original LANL data can be found here:

1) [Netflow](https://s3-us-gov-west-1.amazonaws.com/unified-host-network-dataset/2017/netflow/index.html) – Network Event Data. The data is provided in CSV format, one record per line. The network events represent bi-directional events where possible. The following table contains a description of each field:

|  |  |
| --- | --- |
| Field Name | Description |
| Time | The start time of the event in epoch time format |
| Duration | The duration of the event in seconds |
| SrcDevice | The device that likely initiated the event |
| DstDevice | The receiving device |
| Protocol | The protocol number |
| SrcPort | The port used by the SrcDevice |
| DstPort | The port used by the DstDevice |
| SrcPacket | The number of packets the SrcDevice sent during the event |
| DstPacket | The number of packets the DstDevice sent during the event |
| SrcBytes | The number of bytes the SrcDevice sent during the event |
| DstBytes | The number of bytes the DstDevice sent during the event |

2) [Windows Logging Service](https://s3-us-gov-west-1.amazonaws.com/unified-host-network-dataset/2017/wls/index.html) - The host event data set is a subset of host event logs collected from all computers running the Microsoft Windows operating system on LANL’s enterprise network. The following table contains a description of each field:

|  |  |
| --- | --- |
| Field Name | Description |
| Time | The epoch time of the event in seconds |
| EventID | Four digit integer corresponding to the event id of the record |
| LogHost | The hostname of the computer that the event was recorded on. In the case of directed authentication events, the LogHost will correspond to the computer that the authentication event is terminating at (destination computer). |
| LogonType | Integer corresponding to the type of logon |
| LogonTypeDescription | Description of the LogonType |
| UserName | The user account initiating the event. If the user ends in $, then it corresponds to a computer account for the specified computer |
| DomainName | Domain name of UserName |
| LogonID | A semi-unique (unique between current sessions and LogHost) number that identifies the logon session just initiated. Any events logged subsequently during this logon session should report the same Logon ID through to the logoff event. |
| SubjectUserName | For authentication mapping events, the user account specified by this field is mapping to the user account in UserName |
| SubjectDomainName | Domain name of SubjectUserName |
| SubjectLogonID | See LogonID |
| Status | Status of the authentication request. “0x0” means success otherwise failure |
| Source | For authentication events, this will correspond to the the computer where the authentication originated (source computer), if it is a local logon event then this will be the same as the LogHost |
| ServiceName | The account name of the computer or service the user is requesting the ticket for |
| Destination | This is the server the mapped credential is accessing. This may indicate the local computer when starting another process with new account credentials on a local computer |
| AuthenticationPackage | The type of authentication occurring including Negotiate, Kerberos, NTLM plus a few more |
| FailureReason | The reason for a failed logon |
| ProcessName | The process executable name, for authentication events this is the process that processed the authentication event. ProcessNames may include the file type extensions (i.e exe) |
| ProcessID | A semi-unique (unique between currently running processes AND LogHost) value that identifies the process. Process ID allows you to correlate other events logged in association with the same process through to the process end |
| ParentProcessName | The process executable that started the new process. ParentProcessNames often do not have file extensions like ProcessName but can be compared by removing file extensions from the name |
| ParentProcessID | Identifies the exact process that started the new process. Look for a preceding event 4688 with a ProcessID that matches this ParentProcessID |

Fetch 90 Days LANL Dataset from public portal.

mkdir –p /nvme\_data1/data\_1v

for i in {1..90}

do

wget "https://datasets.trovares.com/LANL/xgt/wls\_day-"$i"\_1v.csv" -O "/nvme\_data1/data\_1v/wls\_day-"$i"\_2v.csv"

done;

mkdir –p /nvme\_data1/data\_2v

for i in {1..90}

do

wget "https://datasets.trovares.com/LANL/xgt/wls\_day-"$i"\_2v.csv" -O "/nvme\_data1/data\_2v/wls\_day-"$i"\_2v.csv"

done;

mkdir –p /nvme\_data1/data\_nf

for i in {2..90}

do

wget "[https://datasets.trovares.com/LANL/xgt/nf\_day-“$i”.csv](https://datasets.trovares.com/LANL/xgt/nf_day-)” -O "/nvme\_data1/data\_nf/nf\_day-"$i".csv"

done;

## Develop Lateral Movement Threat Detection using Trovares xGT

Step #1: Navigate to directory where Working LANL Dataset is available and start the xGT server from that location

[root@mdchawkeye]# cd /nvme\_data1

[root@mdchawkeye nvme\_data1]# xgtd

[n0 w3100322112 0.001] Starting xGT version: 1.3.0

[n0 w3100322112 0.039] Available memory for xGT data: 6390723641344

[n0 w3100322112 0.039] TBB worker threads: 448

[n0 w3100322112 0.040] Pinning threads: false

Step #2: Start iPython Notebook, build and load Cybersecurity Graph Analytics Notebook

(Appendix-A)

(Python\_3\_7) [root@mdchawkeye eventgen]# jupyter notebook --ip 172.30.224.99 --allow-root

## Measure Query Performance against Lateral Movement Threat Detection

Step #1: Load and execute Lateral Movement Cyber-Threat Detection Notebook (Refer Appendix-3)

Step #2: Measure following parameters

High Level Query Performance Results:

|  |  |
| --- | --- |
| Summary | Count(xGT 1.3) |
| Python Version | 3.7.4 |
| Devices (vertices) | 933,714 |
| Netflow (edges) | 17,882,795,024 |
| Host event 1-vertex (edges) | 1,468,936,024 |
| Host event 2-vertex (edges) AuthEvents | 4,022,436,222 |
| Total (edges) | 23,374,167,270 |
| RDPflow (edges) | 757,436 |
| Total Memory Used (GB) | 3028 |
| Total Memory Available (TB) | 5,862 |
| % Memory Used | 52 |
| Total Data Volume | 1.4TB |
| Total Cores | 224 |
| CPU Utilization | >90% |

Data Load Performance Results:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Function | Devices  (Vertices) | Netflow  (Edges) | Host-Event  (Edges) (1 Vertex) | AuthEvent  (Edges) (2 Vertex) | Total Edges | CPU Time  (User) | CPU Time  (System) | CPU Time  (Total) | Wall Time |
| Load-1(90 Days) - xGT 1.3 | 13,491 | 0 | 1,468,936,024 | 0 | 1,468,936,024 | 238 ms | 350 ms | 589 ms | 3m 57s |
| Load-2(90 Days) - xGT 1.3 | 18,925 | 0 | 1,468,936,024 | 4,022,436,222 | 5,491,372,246 | 382 ms | 412 ms | 794 ms | 4m 49s |
| Load-3(90 Days) - xGT 1.3 | 933,714 | 17,882,795,024 | 1,468,936,024 | 4,022,436,222 | 23,374,167,270 | 896 ms | 907 ms | 1.8 s | 14m 22s |

Cyber Threat Query Execution Performance Results:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Function | Query Type | Netflow Edges | CPU Time(User) | CPU Time(System) | CPU Time(Total) | Wall Time |
| Query-1 (xGT 1.3) | Forward RDP Edges | 2,176 | 202 ms | 90.7 ms | 293 ms | 1m 6s |
| Query-2 (xGT1.3) | Reverse RDP Edges | 755,260 | 145 ms | 76.6 ms | 222 ms | 1m 10s |
| Query-3 (xGT1.3) | RDP Hack | 57,057 | 230 ms | 125 ms | 355 ms | 7m 1s |
| Query-3 (xGT1.3) | RDP Hack(Optimized) | 57,057 | 61.1ms | 8.99 ms | 70.1 ms | 3.8s |
| Query-3 (xGT1.3) | Hijack Events | 11,715,150 | 83.5 ms | 43.2 ms | 127 ms | 41 s |
| Query-3 (xGT1.3) | PrivEsc Events | 8,695,220 | 81.4 ms | 21.5 ms | 103 ms | 25.6 s |
|  |  |  |  |  |  |  |

# Implement Cybersecurity Graph Analytics for 10TB Dataset

Above 3TB Data Implementation can be considered for 10TB dataset implementation. Consider following assumptions while implementing against 10TB

1. 10TB Dataset needs to be provided by customer.
2. For current scope of PoC, assumption is that data would be made available in csv format
3. PoC data would be made available in a remote file-system
4. There is a minimum 10 GbE network connection between remote system and target Superdome Flex system

## Ingest 10TB Dataset from Remote Sources to Superdome Flex NVMe Data Drives

Below are the steps to move the \*.csv files from Customer staging server to Trovares POC system.

1. Copy the script “**/trovares\_POC/configure\_rsync\_at\_customer\_staging\_server.sh**” to Customer system where \*.CSV files are staged.

# scp $TROVARES\_POC\_HOME/configure\_rsync\_at\_customer\_staging\_server.sh [root@stageserver.abc.com:/tmp](mailto:root@stageserver.abc.com:/tmp)

1. Go to customer staging server (For example: stagingserver.abc.com), configure below mentioned variables and execute the script:

* MODULE: This is rsync parameter and need to be same as configured in $TROVARES\_POC\_HOME/rsync\_data\_mover.sh. Usually we create one module for every \*.CSV file location (directory).
* DEST\_USER: Specify the user name who would be the owner of \*.CSV files at the target (Trovares POC) system.
* DATA\_DIR: \*.CSV file locations

# chmod u+x /tmp/configure\_rsync\_at\_customer\_staging\_server.sh

# /tmp/configure\_rsync\_at\_customer\_staging\_server.sh

1. On the Trovares POC system, run the following command to move the data from stageserver.abc.com. Before running this script, configure following variables in the script:

* SOURCE\_IP: Customer staging server IP address.
* DEST\_DIR: Directories to which \*.CSV files should be copied to.
* MODULE: This is rsync parameter and need to be same as configured in /tmp/configure\_rsync\_at\_customer\_staging\_server.sh. Usually we create one module for every \*.CSV file location (directory).

# $TROVARES\_POC\_HOME/rsync\_data\_mover.sh

# Appendix A – Sample Cybersecurity Threat Detection Notebook

import xgt

import os

import pandas

from platform import python\_version

print (python\_version())

if os.environ.get('https\_proxy'):

del os.environ['https\_proxy']

if os.environ.get('http\_proxy'):

del os.environ['http\_proxy']

try:

devices = conn.get\_vertex\_frame('Devices')

except xgt.XgtNameError:

devices = conn.create\_vertex\_frame(

name='Devices',

schema=[['device', xgt.TEXT]],

key='device')

devices

try:

netflow = conn.get\_edge\_frame('Netflow')

except xgt.XgtNameError:

netflow = conn.create\_edge\_frame(

name='Netflow',

schema=[['epoch\_time', xgt.INT],

['duration', xgt.INT],

['src\_device', xgt.TEXT],

['dst\_device', xgt.TEXT],

['protocol', xgt.INT],

['src\_port', xgt.INT],

['dst\_port', xgt.INT],

['src\_packets', xgt.INT],

['dst\_packets', xgt.INT],

['src\_bytes', xgt.INT],

['dst\_bytes', xgt.INT]],

source=devices,

target=devices,

source\_key='src\_device',

target\_key='dst\_device')

netflow

try:

host\_events = conn.get\_edge\_frame('HostEvents')

except xgt.XgtNameError:

host\_events = conn.create\_edge\_frame(

name='HostEvents',

schema=[['epoch\_time', xgt.INT],

['event\_id', xgt.INT],

['log\_host', xgt.TEXT],

['user\_name', xgt.TEXT],

['domain\_name', xgt.TEXT],

['logon\_id', xgt.INT],

['process\_name', xgt.TEXT],

['process\_id', xgt.INT],

['parent\_process\_name', xgt.TEXT],

['parent\_process\_id', xgt.INT]],

source=devices,

target=devices,

source\_key='log\_host',

target\_key='log\_host')

host\_events

try:

auth\_events = conn.get\_edge\_frame('AuthEvents')

except xgt.XgtNameError:

auth\_events = conn.create\_edge\_frame(

name='AuthEvents',

schema = [['epoch\_time',xgt.INT],

['event\_id',xgt.INT],

['log\_host',xgt.TEXT],

['logon\_type',xgt.INT],

['logon\_type\_description',xgt.TEXT],

['user\_name',xgt.TEXT],

['domain\_name',xgt.TEXT],

['logon\_id',xgt.INT],

['subject\_user\_name',xgt.TEXT],

['subject\_domain\_name',xgt.TEXT],

['subject\_logon\_id',xgt.TEXT],

['status',xgt.TEXT],

['src',xgt.TEXT],

['service\_name',xgt.TEXT],

['destination',xgt.TEXT],

['authentication\_package',xgt.TEXT],

['failure\_reason',xgt.TEXT],

['process\_name',xgt.TEXT],

['process\_id',xgt.INT],

['parent\_process\_name',xgt.TEXT],

['parent\_process\_id',xgt.INT]],

source = 'Devices',

target = 'Devices',

source\_key = 'src',

target\_key = 'destination')

auth\_events

# Utility to print the sizes of data currently in xGT

def print\_data\_summary():

print('Devices (vertices): {:,}'.format(devices.num\_vertices))

print('Netflow (edges): {:,}'.format(netflow.num\_edges))

print('Host events (edges): {:,}'.format(host\_events.num\_edges))

print('Authentication events (edges): {:,}'.format(auth\_events.num\_edges))

print('Total (edges): {:,}'.format(

netflow.num\_edges + host\_events.num\_edges + auth\_events.num\_edges))

print\_data\_summary()

%%time

# Load the HostEvents event data:

if host\_events.num\_edges == 0:

# urls = ["https://datasets.trovares.com/LANL/xgt/wls\_day-85\_1v.csv"]

urls = ["xgtd://nvme\_data1/data\_1v/wls\_day-{:02d}\_1v.csv".format(\_) for \_ in range(2,91)]

# urls = ["xgtd://data\_1v/wls\_day-11\_1v.csv"]

host\_events.load(urls)

print\_data\_summary()

%%time

# Load the AuthEvents event data:

if auth\_events.num\_edges == 0:

# urls = ["https://datasets.trovares.com/LANL/xgt/wls\_day-85\_2v.csv"]

urls = ["xgtd://nvme\_data9/data\_2v/wls\_day-{:02d}\_2v.csv".format(\_) for \_ in range(2,91)]

#urls = ["xgtd://data\_2v/wls\_day-11\_2v.csv"]

auth\_events.load(urls)

print\_data\_summary()

%%time

# Load the netflow data:

if netflow.num\_edges == 0:

#urls = ["https://datasets.trovares.com/LANL/xgt/nf\_day-85.csv"]

urls = ["xgtd://nvme\_data1/data\_nf/nf\_day-{:02d}.csv".format(\_) for \_ in range(2,91)]

# urls = ["xgtd://nvme\_data1/data\_nf/nf\_day-11.csv"]

netflow.load(urls)

print\_data\_summary()

# Utility function to launch queries and show job number:

# The job number may be useful if a long-running job needs

# to be canceled.

def run\_query(query, table\_name = "answers", drop\_answer\_table=True, show\_query=False):

if drop\_answer\_table:

conn.drop\_frame(table\_name)

if query[-1] != '\n':

query += '\n'

query += 'INTO {}'.format(table\_name)

if show\_query:

print("Query:\n" + query)

job = conn.schedule\_job(query)

print("Launched job {}".format(job.id))

conn.wait\_for\_job(job)

table = conn.get\_table\_frame(table\_name)

return table

# Generate a new edge frame for holding only the RDP edges

import time

query\_start\_time = time.time()

conn.drop\_frame('RDPFlow')

rdp\_flow = conn.create\_edge\_frame(

name='RDPFlow',

schema=netflow.schema,

source=devices,

target=devices,

source\_key='src\_device',

target\_key='dst\_device')

rdp\_flow

%%time

#Extract forward RDP edges

q = """

MATCH (v0)-[edge:Netflow]->(v1)

WHERE edge.dst\_port=3389

CREATE (v0)-[e:RDPFlow {epoch\_time : edge.epoch\_time,

duration : edge.duration, protocol : edge.protocol,

src\_port : edge.src\_port, dst\_port : edge.dst\_port,

src\_packets : edge.src\_packets, dst\_packets : edge.dst\_packets,

src\_bytes : edge.src\_bytes, dst\_bytes : edge.dst\_bytes}]->(v1)

RETURN count(\*)

"""

data = run\_query(q)

print('Number of answers: {:,}'.format(data.get\_data()[0][0]))

%%time

#Extract reverse RDP edges

q = """

MATCH (v0)-[edge:Netflow]->(v1)

WHERE edge.src\_port=3389

CREATE (v1)-[e:RDPFlow {epoch\_time : edge.epoch\_time,

duration : edge.duration, protocol : edge.protocol,

src\_port : edge.dst\_port, dst\_port : edge.src\_port,

src\_packets : edge.dst\_packets, dst\_packets : edge.src\_packets,

src\_bytes : edge.dst\_bytes, dst\_bytes : edge.src\_bytes}]->(v0)

RETURN count(\*)

"""

data = run\_query(q)

print('Number of answers: {:,}'.format(data.get\_data()[0][0]))

data=None

if rdp\_flow.num\_edges == 0:

print("RDPFlow is empty")

elif rdp\_flow.num\_edges <= 1000:

data = rdp\_flow.get\_data\_pandas()

else:

data = 'RDPflow (edges): {:,}'.format(rdp\_flow.num\_edges)

data

# Utility to print the data sizes currently in xGT

def print\_netflow\_data\_summary():

print\_data\_summary()

print('RDPFlow (edges): {:,}'.format(rdp\_flow.num\_edges))

print\_netflow\_data\_summary()

%%time

#Lateral Movement Query

time\_threshold\_between\_step = 3600 # one hour

time\_threshold\_hijack = 180 # three minutes

time\_threshold\_one\_step = 480 # eight minutes

q = """

MATCH (A)-[rdp1:RDPFlow]->(B)-[rdp2:RDPFlow]->(C),

(A)-[hijack1:HostEvents]->(A)-[privEsc1:HostEvents]->(A),

(B)-[hijack2:HostEvents]->(B)-[privEsc2:HostEvents]->(B)

WHERE A <> B AND B <> C AND A <> C

AND privEsc1.event\_id = 4688

AND (privEsc1.process\_name = "Proc336322.exe" OR privEsc1.process\_name = "Proc695356.exe")

AND hijack1.event\_id = 4688 AND hijack1.process\_name = "Proc249569.exe"

AND privEsc2.event\_id = 4688

AND (privEsc2.process\_name = "Proc336322.exe" OR privEsc2.process\_name = "Proc695356.exe")

AND hijack2.event\_id = 4688 AND hijack2.process\_name = "Proc249569.exe"

// Check time constraints on the overall pattern

AND rdp1.epoch\_time <= rdp2.epoch\_time

AND rdp2.epoch\_time - rdp1.epoch\_time < {0}

// Check time constraints on step from A to B

AND privEsc1.epoch\_time <= hijack1.epoch\_time

AND hijack1.epoch\_time <= rdp1.epoch\_time

AND rdp1.epoch\_time - hijack1.epoch\_time < {1}

AND rdp1.epoch\_time - privEsc1.epoch\_time < {2}

// Check time constraints on step from B to C

AND privEsc2.epoch\_time <= hijack2.epoch\_time

AND hijack2.epoch\_time <= rdp2.epoch\_time

AND rdp2.epoch\_time - hijack2.epoch\_time < {1}

AND rdp2.epoch\_time - privEsc2.epoch\_time < {2}

RETURN rdp1.src\_device, rdp1.dst\_device, rdp1.epoch\_time, rdp2.dst\_device, rdp2.epoch\_time

""".format(time\_threshold\_between\_step, time\_threshold\_hijack, time\_threshold\_one\_step)

answer\_table = run\_query(q)

print('Number of answers: {:,}'.format(answer\_table.num\_rows))

%%time

# Build HijackEvents table

import time

start\_optimized\_query\_time = time.time()

conn.drop\_frame('HijackEvents')

hijack\_events = conn.create\_edge\_frame(

name ='HijackEvents',

schema = [['epoch\_time', xgt.INT],

['src\_host', xgt.TEXT],

['dst\_host', xgt.TEXT]],

source = devices,

target = devices,

source\_key = 'src\_host',

target\_key = 'dst\_host')

query = """

MATCH (v0)-[edge:HostEvents]->(v0)

WHERE edge.process\_name = "Proc249569.exe"

AND edge.event\_id = 4688

CREATE (v0)-[e:HijackEvents { epoch\_time : edge.epoch\_time }]->(v0)

RETURN count(\*)

"""

run\_query(query)

print('HijackEvents (edges): {:,}'.format(hijack\_events.num\_edges))

%%time

# Build a PrivEscEvents table

conn.drop\_frame('PrivEscEvents')

priv\_esc\_events = conn.create\_edge\_frame(

name ='PrivEscEvents',

schema = [['epoch\_time', xgt.INT],

['src\_host', xgt.TEXT],

['dst\_host', xgt.TEXT]],

source = devices,

target = devices,

source\_key = 'src\_host',

target\_key = 'dst\_host')

query = """

MATCH (v0)-[edge:HostEvents]->(v0)

WHERE edge.process\_name = "Proc336322.exe" OR

edge.process\_name = "Proc695356.exe"

AND edge.event\_id = 4688

CREATE (v0)-[e:PrivEscEvents { epoch\_time : edge.epoch\_time }]->(v0)

RETURN count(\*)

"""

run\_query(query)

print('PrivEscEvents (edges): {:,}'.format(priv\_esc\_events.num\_edges))

%%time

# Now run the lateral movement query using these new index tables

q = """

MATCH (A)-[rdp1:RDPFlow]->(B)-[rdp2:RDPFlow]->(C),

(A)-[hijack1:HijackEvents]->(A)-[priv\_esc1:PrivEscEvents]->(A),

(B)-[hijack2:HijackEvents]->(B)-[priv\_esc2:PrivEscEvents]->(B)

WHERE A <> B AND B <> C AND A <> C

// Check time constraints on the overall pattern

AND rdp1.epoch\_time <= rdp2.epoch\_time

AND rdp2.epoch\_time - rdp1.epoch\_time < {0}

// Check time constraints on step from A to B

AND priv\_esc1.epoch\_time <= hijack1.epoch\_time

AND hijack1.epoch\_time <= rdp1.epoch\_time

AND rdp1.epoch\_time - hijack1.epoch\_time < {1}

AND rdp1.epoch\_time - priv\_esc1.epoch\_time < {2}

// Check time constraints on step from B to C

AND priv\_esc2.epoch\_time <= hijack2.epoch\_time

AND hijack2.epoch\_time <= rdp2.epoch\_time

AND rdp2.epoch\_time - hijack2.epoch\_time < {1}

AND rdp2.epoch\_time - priv\_esc2.epoch\_time < {2}

RETURN rdp1.src\_device, rdp1.dst\_device, rdp1.epoch\_time, rdp2.dst\_device, rdp2.epoch\_time

""".format(time\_threshold\_between\_step, time\_threshold\_hijack, time\_threshold\_one\_step)

start\_timer = time.time()

answer\_table = run\_query(q)

end\_timer = time.time()

query\_end\_time = time.time()

print('Number of answers: {:,}'.format(answer\_table.num\_rows))

print("Time for query: {:,.2f}".format(end\_timer - start\_timer))

print("Overall time for query: {:,.2f}".format(query\_end\_time - query\_start\_time))

# Appendix B – Pre-Requisite Setup Cookbook Script

#!/bin/bash

#

# xGT pre-requisites setup

#

# Please update below variables before running this script in your setup.

#

# Specify the directory name where you have extracted the SDFlex\_Trovares\_RL\_POC.tgz file

TROVARES\_POC\_HOME="/root/trovares\_POC"

#

# Mention a mount point name for NVMe disks and it has to be in absolute path (full path)

# NVMe disks will be mounted on /nvme\_data0, /nvme\_data1, /nvme\_data2 and so on

NVME\_MNT\_DIR="/nvme\_data"

NVME\_DEVICE\_NAME\_PATTERN="/dev/nvme0n\*"

# Update your organization NTP Server IP/hostname here

NTPSVR="server 0.pool.ntp.org"

# Update NVMe disks list here. Enter complete path. For example: /dev/nvme0n1

NVME\_DISK\_LIST=(/dev/nvme0n1 /dev/nvme0n2 /dev/nvme0n3)

# HPE FOUNDATION SOFTWARE mount point

HPEFS\_MNT\_DIR="/hpefswmount"

# Below parameters are used by SAR function to collect network statistics

NETWORK\_INTERFACE=ens33

REPORT\_DIR="/hpeSarReport"

function usage {

echo ""

echo 'Usage:'

echo "------"

echo '# ./setup\_prereq.sh -[Option]'

echo ""

echo "Options:"

echo "-f: [Step 1] Installs HPE Foundation Software for HPE Superdome Flex, which is located at"

echo " $TROVARES\_POC\_HOME/hpefsw.(Requires RHEL OS yum repo configured & Foundation ISO shipped"

echo " along with this script)"

echo "-m: [Step 2] Create filesystem on below NVME disks (defined inside the script),"

echo " `echo "Currently selected NVMe Disks: "${NVME\_DISK\_LIST[@]}`,"

echo " create mount points, update /etc/fstab, and mount them at "$NVME\_MNT\_DIR"0, "$NVME\_MNT\_DIR"1"

echo " and so on. "

echo "-d: [Step 3] Tune NVMe disks:`echo ${NVME\_DISK\_LIST[@]}`"

echo "-s: [Step 4] Sets SELINUX to Permissive, setup limits.conf, tunes kernel params, disables Firewall"

echo " sets maximum CPU frequency, disables HugePage."

echo "-n: [Step 5] Configure NTP Client (Update your NTP Server IP in the script before running this)."

echo "-j: [Step 6] Installs JAVA (Requires RHEL OS yum repo configured)."

echo "-p: [Step 7] Installs Python (Python-3.7.4.tar.xz Source file shipped along with this script)."

echo "-x: [Step 8] Installs xGT python packages."

echo "-z: [Step 9] Installs Jupyter."

echo "-r: Collect SAR Reports at $REPORT\_DIR"

echo " "

echo "############################################################################################"

echo "NOTE: It is recommended to use above options step by step. For example: "

echo "Execute first './setup\_prereq.sh -f' and then './setup\_prereq.sh -m', and so on"

echo "############################################################################################"

echo " "

}

if [ $# -eq 0 ]; then

echo "No arguments provided. Check the syntax:"

usage

exit 1

fi

function setupFSWrepo {

rpm -qa | grep "hpe-foundation"

if [ $? -eq 0 ]; then

echo "HPE Foundation Software is already installed !"

exit 0

else

echo "HPE Foundation Software is not yet installed !"

fi

ls $TROVARES\_POC\_HOME/hpefsw/hpe-foundation\*.iso >/dev/null 2>&1

if [ $? -ne 0 ]; then

echo "Trying to install now, but could not find HPE Foundation Software ISO at $TROVARES\_POC\_HOME/hpefsw/. Please perform the following steps:"

echo "1) Please download SDFlex\_Trovares\_RL\_POC.tgz file from ftp location mentioned in the Cookbook."

echo "2) Untar it using 'tar xvzf SDFlex\_Trovares\_RL\_POC.tgz' as root user."

echo "3) Update TROVARES\_POC\_HOME parameter in system\_prereq.sh"

echo "3) Finally re-run the script with '-f' option."

echo " "

exit 1

else

echo "Found `ls $TROVARES\_POC\_HOME/hpefsw/hpe-foundation\*.iso`"

echo "Creating directory $HPEFS\_MNT\_DIR, if it does not exist already"

mkdir $HPEFS\_MNT\_DIR >/dev/null 2>&1

echo "Mounting `ls $TROVARES\_POC\_HOME/hpefsw/hpe-foundation\*.iso` on $HPEFS\_MNT\_DIR"

mount -t iso9660 -ro loop `ls $TROVARES\_POC\_HOME/hpefsw/hpe-foundation\*.iso` $HPEFS\_MNT\_DIR

fi

echo "Creating HPE Foundation Software Repo.."

cat << EOF > /etc/yum.repos.d/hpefsw.repo

[hpefsw]

name=HPE Foundation Software - Trovares Setup

baseurl=file://`echo $HPEFS\_MNT\_DIR`/RPMS

enabled=1

gpgcheck=0

gpgkey=file://`echo $HPEFS\_MNT\_DIR`/RPM-GPG-KEY-hpe file://`echo $HPEFS\_MNT\_DIR`/RPM-GPG-KEY-sgi

EOF

# yum repolist

}

function installFSW {

echo 'Installing HPE Foundation Software: yum groupinstall "HPE Foundation Software" -y'

yum groupinstall "HPE Foundation Software" -y

if [ $? -ne 0 ]; then

echo ''

echo "##################################################################################"

echo 'WARNING: Installation failed'

echo 'NOTE: yum groupinstall "HPE Foundation Software" failed. Please check manually.'

echo 'NOTE: HPE Foundation software has dependency on many RPMs located in RHEL OS image.'

echo "Please configure yum repo for the RHEL OS and re-try."

echo "###################################################################################"

echo 'For example: Configure OS repo using ISO image as shown below and re-try.'

echo 'Create ISO mount point => # mkdir /rhelmnt'

echo 'Mount the RHEL ISO image => # mount -t iso9660 -ro loop /root/RHEL-7.6-20181010.0-Server-x86\_64-dvd1.iso /rhelmnt'

echo ''

echo 'Create repo file as shown below:'

echo '# cat /etc/yum.repos.d/rhelmnt.repo'

echo '[rheliso]'

echo 'name=RHEL OS Image'

echo 'baseurl=file:///rhelmnt'

echo 'enabled=1'

echo 'gpgcheck=0'

echo ''

exit 1

fi

systemctl status memlog

}

function createFilesytemOnNvmeDisk {

ls $NVME\_DEVICE\_NAME\_PATTERN >/dev/null 2>&1

if [ $? -ne 0 ]; then

echo "NVMe disks not found"

exit 1

fi

for disk in "${NVME\_DISK\_LIST[@]}"

do

fdisk -l /dev/nvme0n3 | grep label

partexist=`echo $?`

file -sL $disk | grep filesystem

if [ $? -eq 0 ]; then

echo "INFO: Filesystem already exists on $disk. Not creating filesystem !"

elif [ $partexist -eq 0 ]; then

echo "WARNING: Partition found on the $disk. Please remove partition from $disk manually and re-run this command !"

else

echo "INFO: Creating XFS filesystem on $disk .."

mkfs.xfs $disk

fi

done

}

function createMountPoints {

ls $NVME\_DEVICE\_NAME\_PATTERN >/dev/null 2>&1

if [ $? -ne 0 ]; then

echo "NVMe disks not found."

exit 1

fi

for disk in "${NVME\_DISK\_LIST[@]}"

do

# Finding NVMe disk number

dsknum=`echo $disk | cut -d"/" -f3 | cut -d"e" -f2 | cut -d"n" -f1`

mkdir /"$NVME\_MNT\_DIR"$dsknum

done

}

function mountNVMEdisks {

ls $NVME\_DEVICE\_NAME\_PATTERN >/dev/null 2>&1

if [ $? -ne 0 ]; then

echo "NVMe disks not found."

exit 1

fi

for disk in "${NVME\_DISK\_LIST[@]}"

do

echo "Adding entry in fstab..."

grep $disk /etc/fstab

if [ $? -eq 0 ]; then

echo "fstab entry already exists for $disk, skipping."

else

dsknum=`echo $disk | cut -d"/" -f3 | cut -d"e" -f2 | cut -d"n" -f1`

echo "# Below entry added by Trovares POC setup script" >> /etc/fstab

echo "$disk "$NVME\_MNT\_DIR"$dsknum xfs noatime,nodiratime 0 0" >> /etc/fstab

echo "Added fstab entry for $disk."

fi

done

mount -a

}

function setReadAhead {

ls $NVME\_DEVICE\_NAME\_PATTERN >/dev/null 2>&1

if [ $? -ne 0 ]; then

echo "This system does not have NVMe disks. Not setting tunables."

exit 0

fi

for i in "${NVME\_DISK\_LIST[@]}"

do

echo "Setting read ahead using blockdev on $i."

blockdev --setra 8192 $i;

done;

}

function setDiskParam {

ls $NVME\_DEVICE\_NAME\_PATTERN >/dev/null 2>&1

if [ $? -ne 0 ]; then

echo "This system does not have NVMe disks. Not setting tunables."

exit 0

fi

for f in "${NVME\_DISK\_LIST[@]}";

do

dsk=`echo $f | cut -d"/" -f3`

val=$(cat /sys/block/$dsk/queue/rotational)

if [[ "$val" -eq "0" ]]; then

echo "Disk /sys/block/$dsk configured as NVME\_SSD"

echo "Setting $f/queue/read\_ahead\_kb to 4096 for $f"

echo "echo 4096 > /sys/block/$dsk/queue/read\_ahead\_kb" >> /etc/rc.d/rc.local

echo "Setting $f/queue/nr\_requests to 32 for $f"

echo "echo 32 > /sys/block/$dsk/queue/nr\_requests" >> /etc/rc.d/rc.local

echo "Setting $f/queue/nomerges to 2 for $f"

echo "echo 2 > /sys/block/$dsk/queue/nomerges" >> /etc/rc.d/rc.local

echo "Setting $f/queue/rq\_affinity to 2 for $f"

echo "echo 2 > /sys/block/$dsk/queue/rq\_affinity" >> /etc/rc.d/rc.local

echo "========================================="

fi

done

}

function setSelinuxHugepageSystectl {

echo "######################################"

echo "### Setting SELINUX=permissive #######"

echo "######################################"

sed -i -e 's/^SELINUX=.\*/SELINUX=permissive/g' /etc/selinux/config

echo "Below change has been done in /etc/selinux/config"

grep ^SELINUX /etc/selinux/config

echo " "

echo "/etc/selinux/config file modified !"

echo "##########################################################################################"

echo "### Creating /etc/rc.d/rc.local and adding command to disable transparent\_hugepage #######"

echo "###########################################################################################"

cat << EOF >> /etc/rc.d/rc.local

echo never > /sys/kernel/mm/transparent\_hugepage/enabled

echo never > /sys/kernel/mm/transparent\_hugepage/defrag

EOF

echo "Setting up I/O Schedular for NVMe disks in /etc/rc.d/rc.local"

for i in "${NVME\_DISK\_LIST[@]}"

do

dsk=`echo $i | cut -d"/" -f3`

echo "echo 'mq-deadline' > /sys/block/$dsk/queue/scheduler" >> /etc/rc.d/rc.local

done;

echo " "

echo "Setting up maximum frequency for the CPU"

echo " "

ls -l /sys/devices/system/cpu/cpu0/cpufreq/scaling\_available\_frequencies >/dev/null 2>&1

if [ $? -ne 0 ]; then

echo "##################################################################################################################"

echo "WARNING: Unable to set CPU maximum frequency as /sys/devices/system/cpu/cpu0/cpufreq/scaling\_available\_frequencies"

echo "could not be found !"

echo "##################################################################################################################"

else

max\_freq=$(for c in $(cat /sys/devices/system/cpu/cpu0/cpufreq/scaling\_available\_frequencies); do echo $c; done | sort -nr | head -1)

for cpu\_max\_freq in /sys/devices/system/cpu/cpu\*/cpufreq/scaling\_max\_freq

do

echo "echo $max\_freq > $cpu\_max\_freq" >> /etc/rc.d/rc.local

done

fi

echo "Below is the modified content of /etc/rc.d/rc.local file"

cat /etc/rc.d/rc.local

chmod +x /etc/rc.d/rc.local

echo "######################################"

echo "### Updating sysctl.conf file #######"

echo "######################################"

echo ""

cp /etc/sysctl.conf /etc/sysctl.conf\_HPEOLD

echo "Existing /etc/sysctl.conf has been backed up as /etc/sysctl.conf\_HPEOLD. Creating a new /etc/sysctl.conf and applying it."

cat <<EOF > /etc/sysctl.conf

kernel.sem = 250 32000 100 128

kernel.shmall = 2097152

kernel.shmmax = 2147483648

fs.file-max = 65536

kernel.shmmni = 4096

fs.aio-max-nr = 1048576

net.ipv4.ip\_local\_port\_range = 1024 65000

net.core.rmem\_default = 4194304

net.core.wmem\_default = 2621444

net.core.rmem\_max=4194304

net.core.wmem\_max=262144

net.ipv4.tcp\_rmem = 1048576 1048576 4194304

net.ipv4.tcp\_wmem = 1048576 1048576 1048576

net.ipv6.conf.default.disable\_ipv6 = 1

net.ipv6.conf.all.disable\_ipv6 = 1

kernel.nmi\_watchdog=0

vm.swappiness=1

EOF

echo "Here is the current sysctl configuration"

sysctl -p

echo "You must reboot the server now to activate the configuration done now."

}

function setLimits {

echo "###################################################"

echo "### Updating /etc/security/limits.conf file #######"

echo "###################################################"

echo ""

cp /etc/security/limits.conf /etc/security/limits.conf\_HPEOLD

echo "Existing /etc/sysctl.conf has been backed up as /etc/security/limits.conf\_HPEOLD. Creating a new /etc/security/limits.conf and applying it."

cat << EOF > /etc/security/limits.conf

\* soft nproc 65536

\* hard nproc 65536

\* soft nofile 262144

\* hard nofile 262144

EOF

echo "Below is the current /etc/security/limits.conf."

cat /etc/security/limits.conf

}

function stopStartServices {

echo "###################################################"

echo "### Disabling firewall, tuned-adm services #######"

echo "###################################################"

echo ""

systemctl disable firewalld

systemctl stop firewalld

systemctl status firewalld

echo "Disabling tuned service. Ignore any error message. Error message may be reported if tuned was already disabled."

tuned-adm off

tuned-adm list

systemctl stop tuned

systemctl disable tuned

#echo "Starting rpcbind service"

#service rpcbind start

echo "#######################################################################"

echo "You must reboot the server now to activate the configuration done now."

echo "########################################################################"

}

function ntpClient {

echo "################################"

echo "### Setting up NTP Client#######"

echo "################################"

echo ""

rpm -qa | grep ntp-

if [ $? -eq 0 ]; then

echo "`rpm -qa | grep ntp-` is already installed. "

else

echo "Installing NTP using RHEL OS Repository"

yum -y install ntp

if [ $? -ne 0 ]; then

echo "ntp installation failed. Please check RHEL OS yum repository configured or not."

exit 1

fi

fi

echo "Creating /etc/ntp.conf with dummy NTP server."

cat << EOF > /etc/ntp.conf

driftfile /var/lib/ntp/drift

restrict default nomodify notrap nopeer noquery

restrict 127.0.0.1

restrict ::1

server `echo $NTPSVR`

includefile /etc/ntp/crypto/pw

keys /etc/ntp/keys

disable monitor

EOF

echo "Staring ntpd daemon..."

systemctl enable ntpd.service

systemctl start ntpd.service

#sleeping for 3 seconds for the daemon to start fully

sleep 3

echo "Checking ntpstat..."

ntpstat

echo "If above reports that ntp is unsynchronised, please check with server admin."

echo "NOTE: Below is the current configuration. Please replace $NTPSVR with"

echo " your NTP server name and restart NTP service"

cat /etc/ntp.conf

echo " "

echo "###########################################################################"

echo "NOTE: Manually update NTP Server information in /etc/ntp.conf."

echo "Ask your Server admin to provide NTP server IP or hostname for your setup"

echo "###########################################################################"

}

function installJava {

yum install -y java-1.8.0-openjdk\*

echo "##################################################################################################"

echo "Setting up JAVA path in $HOME/.bash\_profile:"

echo "This script assumes that java-1.8.0-openjdk-1.8.0.181-7.b13.el7.x86\_64 was installed in the system"

echo "and the home directory is /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.181-7.b13.el7.x86\_64."

echo "If you have installed different Java version, please update JAVA\_HOME in $HOME/.bash\_profile"

echo "##################################################################################################"

echo "export JAVA\_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.181-7.b13.el7.x86\_64" >> $HOME/.bash\_profile

echo "export PATH=$PATH:$JAVA\_HOME/bin" >> $HOME/.bash\_profile

echo "export JAVA\_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.181-7.b13.el7.x86\_64"

echo "export PATH=$PATH:$JAVA\_HOME/bin"

echo "Checking Java installation.."

java -version

if [ $? -ne 0 ]; then

echo "java -version command failed. Please check the path and try again."

echo 'Check this command "/usr/sbin/alternatives --config java" and choose the desired Java installation.'

exit 1

fi

}

function installPythonVirtualenv {

yum install gcc openssl-devel bzip2-devel libffi-devel sqlite-devel zlib-devel -y

cp $TROVARES\_POC\_HOME/python/Python-3.7.4.tar.xz /root

# wget https://www.python.org/ftp/python/3.7.4/Python-3.7.4.tar.xz

cd /root

tar xvf Python-3.7.4.tar.xz

cd Python-3.7.4

./configure --enable-optimizations

make altinstall

echo "Installing virtualenv from $TROVARES\_POC\_HOME/virtualenv"

python3.7 -m pip install $TROVARES\_POC\_HOME/virtualenv/virtualenv-16.7.9-py2\*

cd /root

virtualenv --extra-search-dir=$TROVARES\_POC\_HOME/virtualenv --never-download -p python3.7 Python\_3\_7

echo "#############################################################################"

echo 'Execute "source /root/Python\_3\_7/bin/activate" to activate virtual environment'

echo 'Execute "deactivate" to come out of virtual environment'

echo "#############################################################################"

echo "source /root/Python\_3\_7/bin/activate" >> $HOME/.bash\_profile

}

function installxGTRPMs {

source /root/Python\_3\_7/bin/activate

echo "Installing xGT python packages from $TROVARES\_POC\_HOME/xgt"

python3.7 -m pip install $TROVARES\_POC\_HOME/xgt/\*

}

function installJupyter {

source /root/Python\_3\_7/bin/activate

echo "Installing Jupyter.."

python3.7 -m pip install $TROVARES\_POC\_HOME/jupyter/\*

jupyter notebook --generate-config

ipaddr=`ip a | grep global | awk '{print $2}' | cut -d"/" -f1`

#sed -i -e "s/^#c.NotebookApp.ip.\*/c.NotebookApp.ip=$ipaddr/g" /root/.jupyter/jupyter\_notebook\_config.py

echo "#############################################################################"

echo "You may start Jupyter notebook server using below commands:"

echo "source /root/Python\_3\_7/bin/activate"

echo "jupyter notebook --ip $ipaddr --allow-root"

echo "#############################################################################"

}

function hpeSarReport {

mkdir $REPORT\_DIR >/dev/null 2>&1

if [ $? -ne 0 ]; then

echo "NOTE: $REPORT\_DIR already exists. Proceeding with report collection."

fi

dt=`date | sed -e "s/ /\_/g"`

mkdir $REPORT\_DIR/$dt

if [ $? -ne 0 ]; then

echo "mkdir $REPORT\_DIR/$dt failed. Please check directory permission and run again."

exit 1

fi

sample\_interval=1

number\_of\_samples=10

export LC\_TIME="POSIX"

echo "Storing SAR reports under $REPORT\_DIR/$dt"

echo "Collecting $number\_of\_samples samples at every $sample\_interval second(s) interval.."

# CPU

sar -u $sample\_interval $number\_of\_samples | grep -v -E "CPU|Average|^$" > $REPORT\_DIR/$dt/cpu.dat &

# CPU DETAILED CSV FORMAT

sar -P ALL $sample\_interval $number\_of\_samples | grep -v -E "Linux|Average"| awk '{if ($0 ~ /[0-9]/) { print $1","$2","$3","$4","$5","$6","$7","$8","$9; } }' >$REPORT\_DIR/$dt/CPU.csv &

# RAM

sar -r $sample\_interval $number\_of\_samples | grep -v -E "[a-zA-Z]|^$" > $REPORT\_DIR/$dt/ram.dat &

#RAM DETAILED CSV FORMAT

sar -r $sample\_interval $number\_of\_samples | grep -v -E "Linux|Average" | awk '{if ($0 ~ /[0-9]/) { print $1","$2","$3","$4","$5","$6","$7","$8","$9","$10","$11","$12; }}' >$REPORT\_DIR/$dt/RAM.csv &

# Swap

sar -S $sample\_interval $number\_of\_samples | grep -v -E "[a-zA-Z]|^$" > $REPORT\_DIR/$dt/swap.dat &

# Swap DETAILED CSV FORMAT

sar -S $sample\_interval $number\_of\_samples | grep -v -E "Linux|Average"| awk '{if ($0 ~ /[0-9]/) { print $1","$2","$3","$4","$5","$6","$7; } }' > $REPORT\_DIR/$dt/swap.csv &

# Load average and tasks

sar -q $sample\_interval $number\_of\_samples | grep -v -E "[a-zA-Z]|^$" > $REPORT\_DIR/$dt/loadaverage.dat &

# Load average and tasks DETAILED CSV FORMAT

sar -q $sample\_interval $number\_of\_samples | grep -v -E "Linux|Average"| awk '{if ($0 ~ /[0-9]/) { print $1","$2","$3","$4","$5","$6","$7","$8; } }' > $REPORT\_DIR/$dt/loadaverage.csv &

# IO transfer

sar -b $sample\_interval $number\_of\_samples | grep -v -E "[a-zA-Z]|^$" > $REPORT\_DIR/$dt/iotransfer.dat &

# IO transfer DETAILED CSV FORMAT

sar -b $sample\_interval $number\_of\_samples | grep -v -E "Linux|Average"| awk '{if ($0 ~ /[0-9]/) { print $1","$2","$3","$4","$5","$6","$7; } }' > $REPORT\_DIR/$dt/iotransfer.csv &

# Process/context switches

sar -w $sample\_interval $number\_of\_samples | grep -v -E "[a-zA-Z]|^$" > $REPORT\_DIR/$dt/proc.dat &

# Process/context switches DETAILED CSV FORMAT

sar -w $sample\_interval $number\_of\_samples | grep -v -E "Linux|Average"| awk '{if ($0 ~ /[0-9]/) { print $1","$2","$3","$4; } }' > $REPORT\_DIR/$dt/proc.csv &

# Network Interface

sar -n DEV $sample\_interval $number\_of\_samples | grep $NETWORK\_INTERFACE | grep -v "Average" > $REPORT\_DIR/$dt/netinterface.dat &

# Network Interface DETAILED CSV FORMAT

sar -n DEV $sample\_interval $number\_of\_samples | grep $NETWORK\_INTERFACE | grep -v -E "Linux|Average"| awk '{if ($0 ~ /[0-9]/) \

{ print $1","$2","$3","$4","$5","$6","$7","$8","$9","$10; } }' > $REPORT\_DIR/$dt/netinterface.csv &

# Sockets

sar -n SOCK $sample\_interval $number\_of\_samples | grep -v -E "[a-zA-Z]|^$" > $REPORT\_DIR/$dt/sockets.dat &

# Sockets DETAILED CSV FORMAT

sar -n SOCK $sample\_interval $number\_of\_samples | grep -v -E "Linux|Average"| awk '{if ($0 ~ /[0-9]/) { print $1","$2","$3","$4","$5","$6","$7","$8; } }' > $REPORT\_DIR/$dt/sockets.csv &

echo "Collecting reports in the background. Check the reports at $REPORT\_DIR/$dt."

echo "Sleeping for $(($sample\_interval \* $number\_of\_samples)) seconds"

sleep $(($sample\_interval \* $number\_of\_samples))

}

while getopts ":fmdsnjpxzr" o; do

case "${o}" in

f) #f=${OPTARG}

setupFSWrepo

installFSW

;;

m) #m=${OPTARG}

createFilesytemOnNvmeDisk

createMountPoints

mountNVMEdisks

;;

d) #d=${OPTARG}

setReadAhead

setDiskParam

;;

s) #s=${OPTARG}

setSelinuxHugepageSystectl

setLimits

stopStartServices

;;

n) #n=${OPTARG}

ntpClient

;;

j) #j=${OPTARG}

installJava

;;

p) #p=${OPTARG}

installPythonVirtualenv

;;

x)

#x=${OPTARG}

installxGTRPMs

;;

z) #z=${OPTARG}

installJupyter

;;

r) #r=${OPTARG}

hpeSarReport

;;

\*)

usage

;;

esac

done

shift $((OPTIND-1))