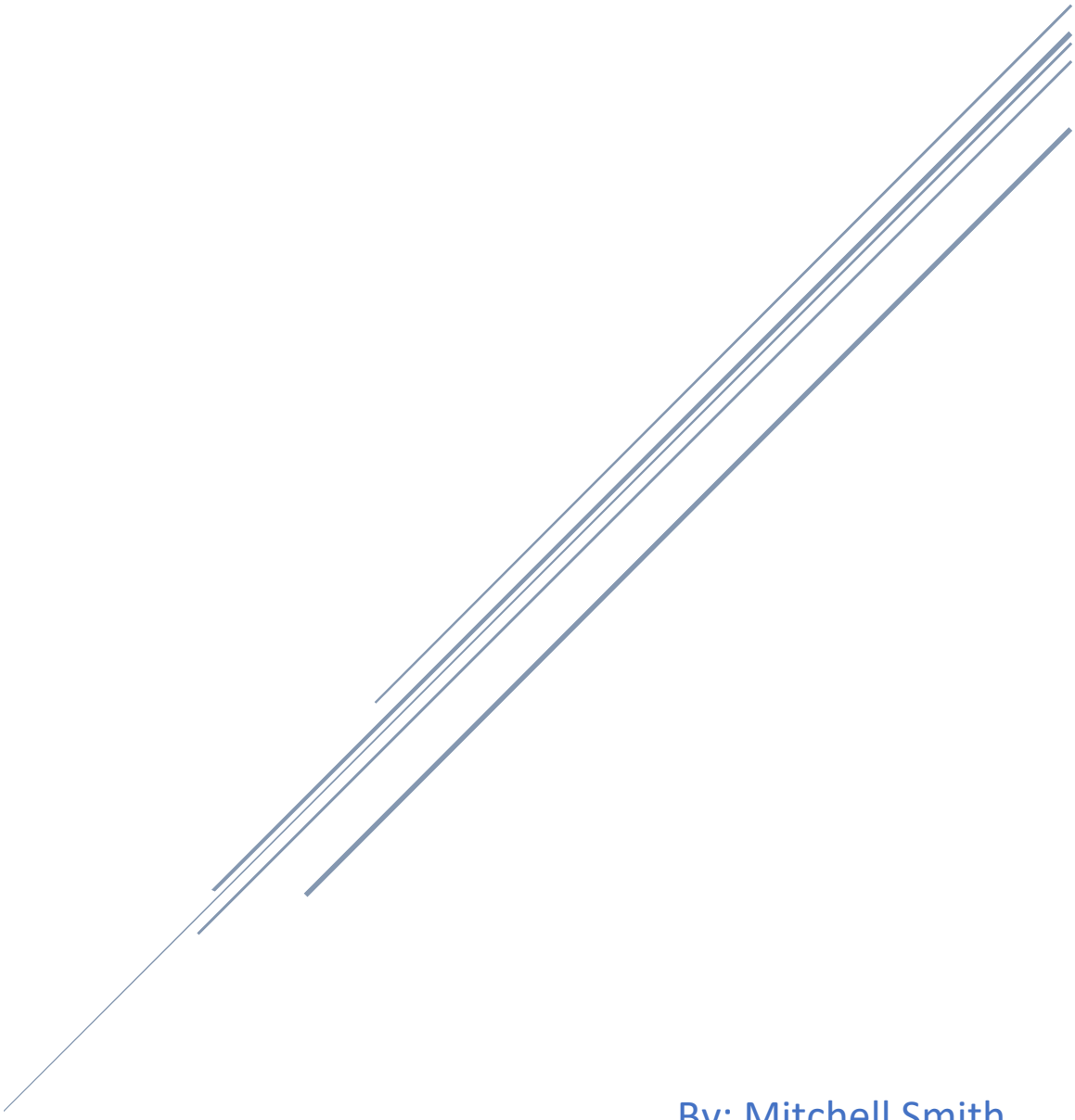
ASSIGNMENT #5

INET3700 – Network OS and Scripting



By: Mitchell Smith
W0440925

Table of Contents

Introduction

In this document, I am going to demonstrate my understanding of network OS and scripting. I am going to achieve this by providing detailed explanations and labelled screenshots for the required tasks that are Change Management, Backups, and Nagios.

Task 1: Change Management

Assignment Questions and Required Screenshots Provided Below.

Question 1. Why do I need to follow Formal Change Management?

Change Management is crucial for IT professionals to follow Because it is a specific method that is put in place for IT professionals to mitigate potential errors to services or systems when modifying / adjusting their elements. Following this method is crucial for IT professionals because it requires them to meet professional regulatory standards in terms of IT practices by mitigating errors that could potentially occur and failure in doing so can significantly impact companies financially, as well as impact their reputation and productivity level.

Question 2. What do I need to know about Change Management?

It is important to know that the Change Management process is based on the premise of protecting companies in multiple aspects when performing tasks within its systems / services that could potentially harm them without proper risk assessment being conducted. The three levels of risk assessment when abiding by the method of Change Management are as follows:

1. **Standard Changes:** These are changes to services / systems that are conducted often and carry a low-impact level in terms of a company's systems / services becoming affected by the modifications being made. These changes abide by a documented procedure and are already pre-approved due to the consistency in which they are conducted.
2. **Normal Changes:** These are higher-level changes in comparison to standard changes and are not consistently conducted to systems / services, but they are not emergency changes that are being made to the system. Due to these changes being higher-level tasks to perform, they require a risk assessment to be performed on them and approved before they can be carried out by an IT professional.
3. **Emergency Changes:** These changes occur when significant errors suddenly occur within systems / services of a company that must be addressed immediately in order to restore their functionality and prevent the company from potentially being impacted by these errors. Due to the threat of impact to the company, risk assessment for actions being taken to address the errors must be carried out quickly.

It is also important to understand the various roles that each IT professional plays in the change management process. These roles are as follows:

> **Change manager/coordinator:** The position that manages every level of the change management process. They are responsible for obtaining change request forms, performing risk assessments, and approving or denying changes.

Change authorities/approvers: This position works with the Change Manager to come to a decision regarding the approval or denial of change requests.

Business stakeholders: This position offers extra input regarding the approval or denial of system changes that are being proposed.

Engineers/developers: This position normally submits the change requests and implements the changes if they are approved.

Service desk agents: This job position identifies potential problems that could occur within the system from the changes that are being made through their communication with end-users.

Operations managers: This job position performs risk assessments on potential changes that are proposed to be made to the system.

Customer relationship managers: This job position communicates with customers and identifies their point of view and needs for the purpose of relaying that information to the stakeholders of the company.

Information security officers and network engineers: This job position identifies potential security threats from system changes that are being proposed.

Question 3. What documents would be included in formal Change Management?

The following documentation that are commonly included in the Change Management Process are as follows:

1. **Change Request Form:** This form documents all the information about the change request and why these changes are being proposed. This form is usually accompanied by supporting documents that provide specific proof and validates the information contained in the change request form.
2. **Change Request Plan:** This document outlines how the change that is being requested will be implemented.
3. **Risk Assessment Plan:** This document provides a risk assessment on the change that is being proposed.
4. **Communication Plan:** This document outlines which parties will be informed throughout the process of the change being implemented and when they will be informed.
5. **Manager approval/Denial Documentation:** This document contains the decision regarding if the change request has been approved or denied.
6. **Progress Updates:** This document will be given in the form of weekly reports and will provide updates as to the progression of the change being implemented.
7. **Change Review:** This document will contain the results of the change being implemented and will evaluate the positive and negative aspects of the change.

Task 2: Backup

Assignment Questions and Required Screenshots Provided Below.

Step 1. Create a viable Backup plan for your Database Server, justify your decisions as to why and when you backup.

The backup plan that I would implement for my database server would use various backup methods to achieve the goal of regular backups of all data in the event that a system / data recovery process would ever have to be conducted.

Full Backups: I would begin by conducting a full backup process for my database server once every two weeks. I would do this for the purpose of my system having a consistently updated state in the event that a system failure was to occur, which would allow me to restore it to a recent state by utilizing other backup procedures along with it. I would replace the old version of the full backup with the most recent version every two weeks.

Differential Backups: I would then conduct differential backups every 3-4 days to ensure that my server data is constantly being updated in the event that a data recovery process must occur. I would do this because if data was constantly being updated in my server, I would want the most recent versions of that data. I would delete this data once the next full backup process is conducted and continue with this backup schedule.

Transaction Log Backups: I would also conduct transaction log backups every 3-4 days to ensure that in the event of a system / data recovery process needing to be conducted, I have the most recent copies of my transactions that were committed, which would majorly benefit me in terms of returning my server to its original state along with the other backup procedures. I would delete this data once the next full backup process is conducted and continue with this backup schedule.

Copy-Only Backups: I would then conduct copy-only backup processes once a week to create and store a recent copy of all the data in my system without interfering with the other backup procedures. I would delete this data once the next full backup process is conducted and continue with this backup schedule.

File Group Backups: I would lastly conduct File group Backups once a week to store the most important elements of my database and keep a copy that is consistently up to date. I would delete this data once the next full backup process is conducted and continue with this backup schedule.

Step 2. Describe the difference between the backup types that are listed in the assignment instructions.

Full Backup – A full backup is the process of creating an identical copy of a systems full state at the time of the backup by copying all the information that is stored in a system and storing it in a location that is designated by a user. This backup serves the purpose of restoring your system entirely in the extreme circumstance that a system failure occurs, and a system recovery procedure must be conducted.

Differential backups – A differential backup is the process of creating a copy of only the data in a system that has been modified / updated since the last full backup of the system was conducted and storing them in a location that is designated by a user. This type of backup serves the purpose of recovering the most recent data in a system and combining it with the most recent full backup of a system in the event that a system failure occurs, and a system / data recovery process must be conducted.

File backups – a File Backup is the process of creating independent copies of files from a system and storing them in a location that is designated by a user in the event that a system / data recovery process must be conducted.

File group backups – A file group backup is the process of creating a grouped copy of files that are all related to one another from a system and storing them in a location that is designated by a user in the event that a system / data recovery process must be conducted.

Partial backups – A partial backup is the process of creating an identical copy of a certain set of file groups that are chosen by the user (or only the primary file group in some instances) and storing them in a location that is designated by a user. This process serves the purpose of restoring crucial data to a system in the event that a data recovery process must be conducted.

Copy-Only backups – Copy-only backups are a backup process that copies the data from a system in a secluded manner and prevents the backup itself from interfering with / affecting the systems backup and restore processes, as well as prevents it from modifying the system itself in certain aspects.

Mirror backups – A Mirror Backup is the process of a user choosing certain folders and files that they want to backup and then creating identical copies of those files and folders. The identical copies are stored in the backup destination that is selected by the user and they are stored as the separate folders and files that the user selected them as rather than all being stored in a compressed container file, which happens when a user utilizes other backup types.

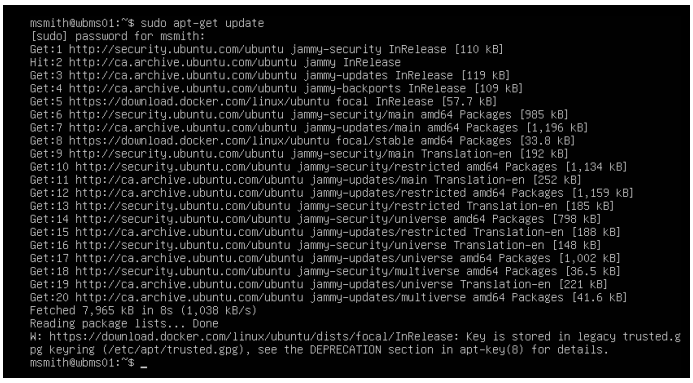
Transaction log backups – A transaction log backup is a backup process where every transaction log that has not yet been backed up is copied and stored in a destination that is designated by the user that is commencing the backup process. This process is used for restoration of all transactions that were committed in the system up until the time of the backup in the event that a user must conduct a system / data recovery process.

Task 3: Implement Nagios

Assignment Questions and Required Screenshots Provided Below.

Step 1. Implement Nagios into your server and configure it so that it can monitor your Web Server and Database. Follow the Nagios documentation and cite where necessary.

Screenshots of the steps taken to install the prerequisite packages:



```
msmith@wbms01:~$ sudo apt-get install -y autoconf gcc libc6 make wget unzip apache2 php libapache2-m
od-php7.4 libgd-dev_
```

```
msmith@wbms01:~$ sudo apt-get install openssl libssl-dev
```

Screenshot navigating to the "tmp" directory:

```
msmith@wbms01:~$ cd /tmp
```

Screenshot of downloading the Nagio source:

```
msmith@wbms01:/tmp$ wget -O nagioscore.tar.gz https://github.com/NagiosEnterprises/nagioscore/archiv
e/nagios-4.4.14.tar.gz
```

Screenshot of extracting data from Nagio source archive file:

```
msmith@wbms01:/tmp$ tar xzf nagioscore.tar.gz
```

Screenshot of navigating to the Nagio source file in the "tmp" directory:

```
msmith@wbms01:/tmp$ cd /tmp/nagioscore-nagios-4.4.14/
```

Screenshot of command used to configure Nagios:

```
msmith@wbms01:/tmp/nagioscore-nagios-4.4.14$ sudo ./configure --with-httpd-conf=/etc/apache2/sites-e
nabled
```

Screenshot of command used to construct Nagios:

```
msmith@wbms01:/tmp/nagioscore-nagios-4.4.14$ sudo make all_
```

Screenshot of command used to create the Nagios user and group:

```
msmith@wbms01:/tmp/nagioscore-nagios-4.4.14$ sudo make install-groups-users
```

Screenshot of command used to add "www-data" user to the Nagios group:

```
msmith@wbms01:/tmp/nagioscore-nagios-4.4.14$ sudo usermod -a -G nagios www-data_
```

Screenshot of command used to install binary files, CGIs, and HTML files:

```
msmith@wbms01:/tmp/nagioscore-nagios-4.4.14$ sudo make install
```

Screenshot of command used to install service or daemon files and configure them to start on boot:

```
msmith@wbms01:/tmp/nagioscore-nagios-4.4.14$ sudo make install-daemoninit_
```

Screenshot of command used to install and configure external command file:

```
msmith@wbms01:/tmp/nagioscore-nagios-4.4.14$ sudo make install-commandmode_
```

Screenshot of command used to install Nagios configuration files:

```
msmith@wbms01:/tmp/nagioscore-nagios-4.4.14$ sudo make install-config_
```

Screenshot of command used to install Apache web server configuration files:

```
msmith@wbms01:/tmp/nagioscore-nagios-4.4.14$ sudo make install-webconf_
```

Screenshot of command used to configure / enable Apache "rewrite" module:

```
msmith@wbms01:/tmp/nagioscore-nagios-4.4.14$ sudo a2enmod rewrite_
```

Screenshot of command used to configure / enable Apache "cgi" module:

```
msmith@wbms01:/tmp/nagioscore-nagios-4.4.14$ sudo a2enmod cgi
```

Screenshot of command used to allow port 80 inbound traffic on local firewall to reach Nagios Core web interface:

```
msmith@wbms01:/tmp/nagioscore-nagios-4.4.14$ sudo ufw allow Apache
```

Screenshot of command used to reload firewall and save changes:

```
msmith@wbms01:/tmp/nagioscore-nagios-4.4.14$ sudo ufw reload_
```

Screenshot of command used to create an Apache user account:

```
msmith@wbms01:/tmp/nagioscore-nagios-4.4.14$ sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users n
agiosadmin
```

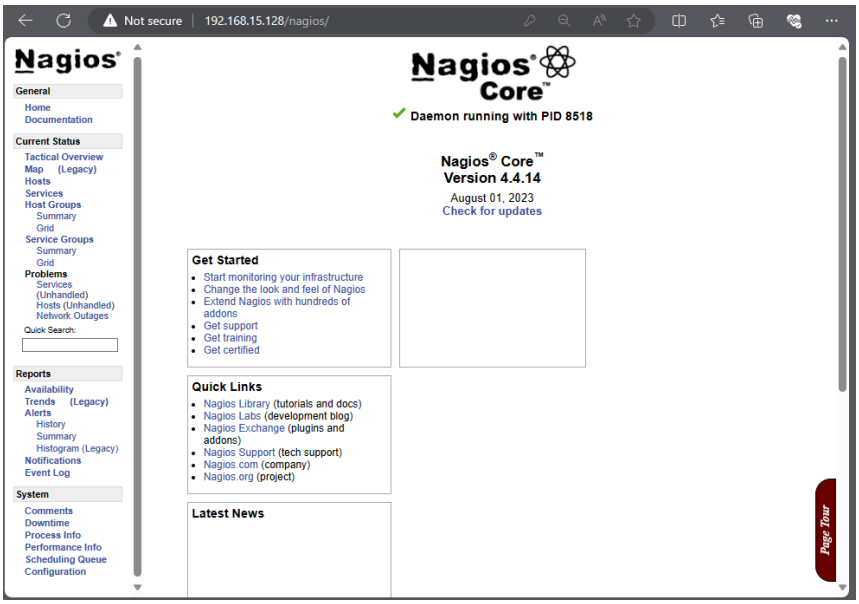
Screenshot of command used to restart Apache Web Server:

```
msmith@wbms01:/tmp/nagioscore-nagios-4.4.14$ sudo systemctl restart apache2.service_
```

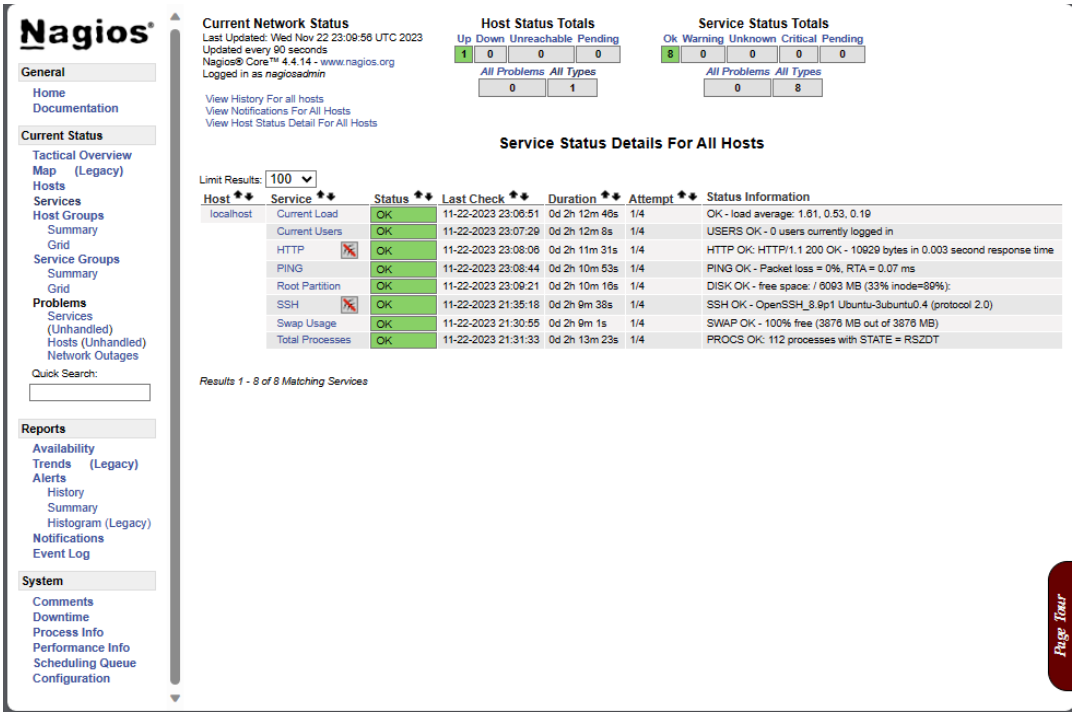
Screenshot of command used to start Nagios Core:

```
msmith@wbms01:/tmp/nagioscore-nagios-4.4.14$ sudo systemctl start nagios.service_
```

Screenshot proving Nagios Core is successfully running when using the URL "http://192.168.15.128/nagios" and logging into the web interface:



Screenshot of active services in Nagios Core:



Screenshot of command used to install Nagios plugins:

```
msmith@wbms01:/tmp/nagioscore-nagios-4.4.14$ sudo apt install monitoring-plugins
```

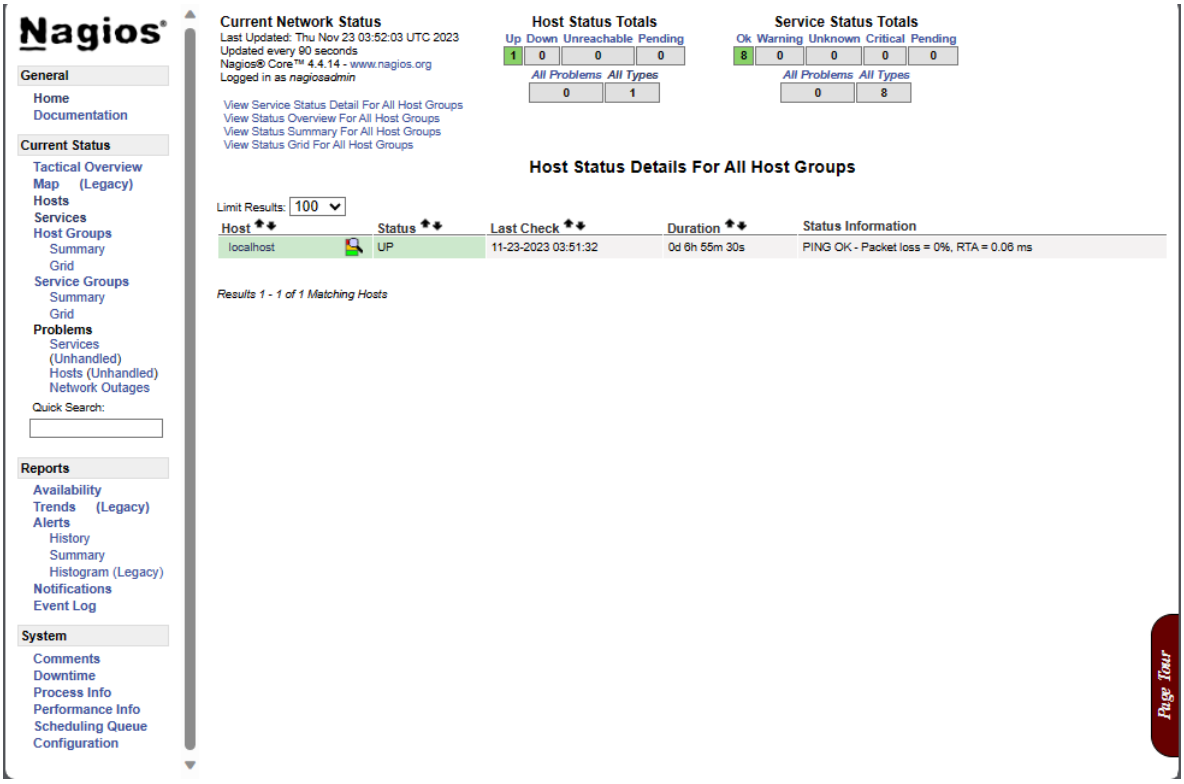
Screenshot of command used to create a symbolic link to the "/usr/local/nagios/libexec" directory:

```
msmith@wbms01:/tmp/nagioscore-nagios-4.4.14$ sudo ln -s /usr/lib/nagios/plugins/* /usr/local/nagios/
libexec
```

Screenshot proving Nagios plugins have been installed:

```
lrwxrwxrwx 1 root root 34 Nov 22 20:55 check_ntp -> /usr/lib/nagios/plugins/check_ntp
lrwxrwxrwx 1 root root 35 Nov 22 20:55 check_ntps -> /usr/lib/nagios/plugins/check_ntps
lrwxrwxrwx 1 root root 32 Nov 22 20:55 check_nt -> /usr/lib/nagios/plugins/check_nt
lrwxrwxrwx 1 root root 33 Nov 22 20:55 check_ntp -> /usr/lib/nagios/plugins/check_ntp
lrwxrwxrwx 1 root root 38 Nov 22 20:55 check_ntp_peer -> /usr/lib/nagios/plugins/check_ntp_peer
lrwxrwxrwx 1 root root 38 Nov 22 20:55 check_ntp_time -> /usr/lib/nagios/plugins/check_ntp_time
lrwxrwxrwx 1 root root 36 Nov 22 20:55 check_nwstat -> /usr/lib/nagios/plugins/check_nwstat
lrwxrwxrwx 1 root root 36 Nov 22 20:55 check_oracle -> /usr/lib/nagios/plugins/check_oracle
lrwxrwxrwx 1 root root 36 Nov 22 20:55 check_overcr -> /usr/lib/nagios/plugins/check_overcr
lrwxrwxrwx 1 root root 35 Nov 22 20:55 check_pgsql -> /usr/lib/nagios/plugins/check_pgsql
lrwxrwxrwx 1 root root 34 Nov 22 20:55 check_ping -> /usr/lib/nagios/plugins/check_ping
lrwxrwxrwx 1 root root 36 Nov 22 20:55 check_pollen -> /usr/lib/nagios/plugins/check_pollen
lrwxrwxrwx 1 root root 33 Nov 22 20:55 check_pop -> /usr/lib/nagios/plugins/check_pop
lrwxrwxrwx 1 root root 35 Nov 22 20:55 check_procs -> /usr/lib/nagios/plugins/check_procs
lrwxrwxrwx 1 root root 36 Nov 22 20:55 check_radius -> /usr/lib/nagios/plugins/check_radius
lrwxrwxrwx 1 root root 34 Nov 22 20:55 check_real -> /usr/lib/nagios/plugins/check_real
lrwxrwxrwx 1 root root 33 Nov 22 20:55 check_rpc -> /usr/lib/nagios/plugins/check_rpc
lrwxrwxrwx 1 root root 39 Nov 22 20:55 check_rta_multi -> /usr/lib/nagios/plugins/check_rta_multi
lrwxrwxrwx 1 root root 37 Nov 22 20:55 check_sensors -> /usr/lib/nagios/plugins/check_sensors
lrwxrwxrwx 1 root root 35 Nov 22 20:55 check_simap -> /usr/lib/nagios/plugins/check_simap
lrwxrwxrwx 1 root root 34 Nov 22 20:55 check_smtp -> /usr/lib/nagios/plugins/check_smtp
lrwxrwxrwx 1 root root 34 Nov 22 20:55 check_snmp -> /usr/lib/nagios/plugins/check_snmp
lrwxrwxrwx 1 root root 34 Nov 22 20:55 check_spop -> /usr/lib/nagios/plugins/check_spop
lrwxrwxrwx 1 root root 33 Nov 22 20:55 check_ssh -> /usr/lib/nagios/plugins/check_ssh
lrwxrwxrwx 1 root root 35 Nov 22 20:55 check_ssmtp -> /usr/lib/nagios/plugins/check_ssmtp
lrwxrwxrwx 1 root root 34 Nov 22 20:55 check_swap -> /usr/lib/nagios/plugins/check_swap
lrwxrwxrwx 1 root root 33 Nov 22 20:55 check_tcp -> /usr/lib/nagios/plugins/check_tcp
lrwxrwxrwx 1 root root 34 Nov 22 20:55 check_time -> /usr/lib/nagios/plugins/check_time
lrwxrwxrwx 1 root root 33 Nov 22 20:55 check_udp -> /usr/lib/nagios/plugins/check_udp
lrwxrwxrwx 1 root root 33 Nov 22 20:55 check_ups -> /usr/lib/nagios/plugins/check_ups
lrwxrwxrwx 1 root root 35 Nov 22 20:55 check_users -> /usr/lib/nagios/plugins/check_users
lrwxrwxrwx 1 root root 34 Nov 22 20:55 check_wave -> /usr/lib/nagios/plugins/check_wave
lrwxrwxrwx 1 root root 30 Nov 22 20:55 negate -> /usr/lib/nagios/plugins/negate
lrwxrwxrwx 1 root root 30 Nov 22 20:55 urlize -> /usr/lib/nagios/plugins/urlize
lrwxrwxrwx 1 root root 32 Nov 22 20:55 utils.pm -> /usr/lib/nagios/plugins/utils.pm
lrwxrwxrwx 1 root root 32 Nov 22 20:55 utils.sh -> /usr/lib/nagios/plugins/utils.sh
msmith@wbms01:/tmp/nagioscore-nagios-4.4.14$ _
```

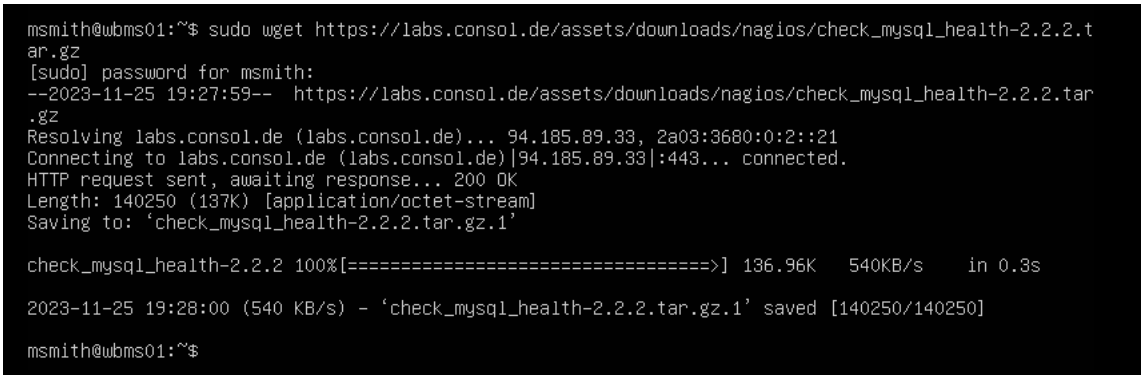
Screenshot proving Nagios is monitoring the web server:



Screenshot of creating users that can be monitored from local host and any source, granting the users all privileges, and flushing privileges to apply them to the users:

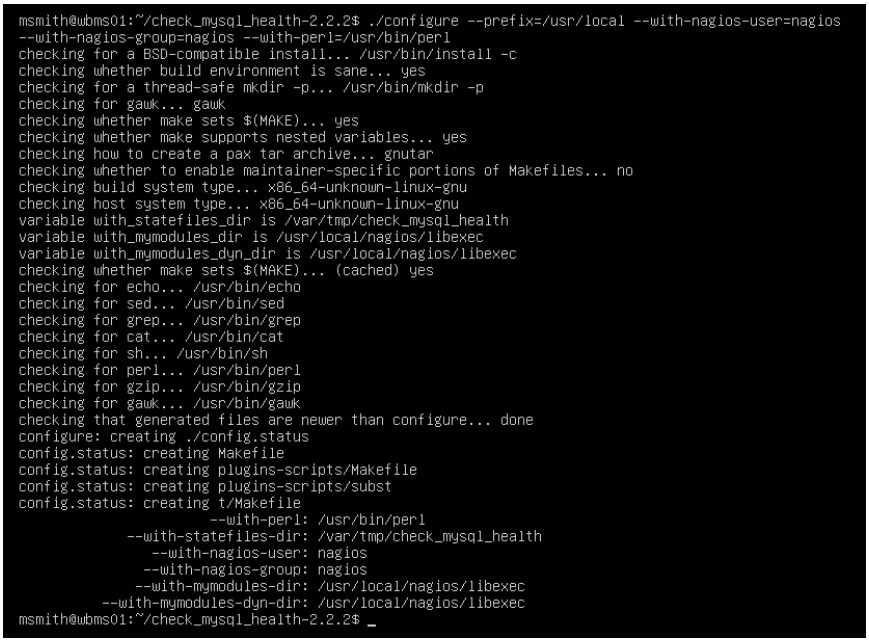
```
mysql> CREATE USER 'nagios'@'localhost' IDENTIFIED BY 'Student@2023';
Query OK, 0 rows affected (0.01 sec)

mysql> GRANT ALL PRIVILEGES ON *.* TO 'nagios'@'localhost';
Query OK, 0 rows affected (0.01 sec)

mysql> CREATE USER 'nagios'@'%' IDENTIFIED BY 'Student@2023';
Query OK, 0 rows affected (0.01 sec)

mysql> GRANT ALL PRIVILEGES ON *.* TO 'nagios'@'%';
Query OK, 0 rows affected (0.01 sec)

mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.01 sec)

mysql>
```

Screenshot of downloading the "check_mysql_health-2.2.2.tar.gz" package:

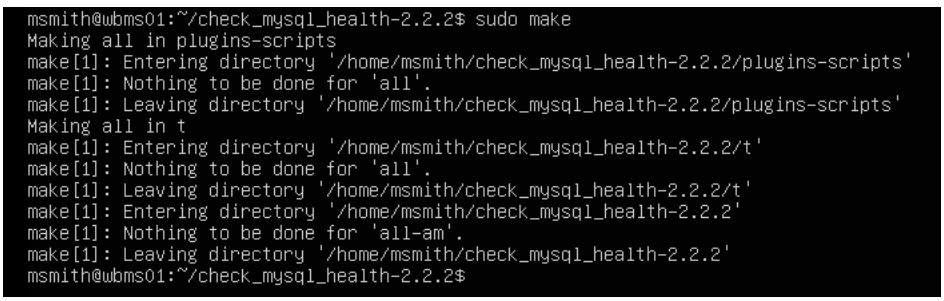


Screenshot of command used to extract "check_mysql_health-2.2.2.tar.gz" package:

Screenshot of navigating to the "check_mysql_health-2.2.2" directory and configuring the package:



Screenshot of output when using the "sudo make" command:



Screenshot of output when using the "sudo make install" command:

```
msmith@wbms01:~/check_mysql_health-2.2.2$ sudo make install
Making install in plugins-scripts
make[1]: Entering directory '/home/msmith/check_mysql_health-2.2.2/plugins-scripts'
make[2]: Entering directory '/home/msmith/check_mysql_health-2.2.2/plugins-scripts'
 /usr/bin/mkdir -p '/usr/local/libexec'
 /usr/bin/install -c check_mysql_health '/usr/local/libexec'
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/home/msmith/check_mysql_health-2.2.2/plugins-scripts'
make[1]: Leaving directory '/home/msmith/check_mysql_health-2.2.2/plugins-scripts'
Making install in t
make[1]: Entering directory '/home/msmith/check_mysql_health-2.2.2/t'
make[2]: Entering directory '/home/msmith/check_mysql_health-2.2.2/t'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/home/msmith/check_mysql_health-2.2.2/t'
make[1]: Leaving directory '/home/msmith/check_mysql_health-2.2.2/t'
make[1]: Entering directory '/home/msmith/check_mysql_health-2.2.2'
make[2]: Entering directory '/home/msmith/check_mysql_health-2.2.2'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/home/msmith/check_mysql_health-2.2.2'
make[1]: Leaving directory '/home/msmith/check_mysql_health-2.2.2'
msmith@wbms01:~/check_mysql_health-2.2.2$
```

Screenshot of command used to install DBD module:

```
msmith@wbms01:~/check_mysql_health-2.2.2/plugins-scripts$ sudo apt-get install -y libdbd-mysql-perl
```

Task 4: Change Management Log

Completed Change Management Log provided below.

Change Management Log		
Activity Number	**Action Taken**	**Description of Action**
A001	System packages updates / upgraded.	Following command was used to update system packages: ***"sudo apt update"*** Following command was used to upgrade system packages: ***"sudo apt upgrade"***
A002	/etc/apache2/apache2.conf file edited.	/etc/apache2/apache2.conf file edited to incorporate the additional text to the end of the file: ***"#Disable Trace HTTP Requests – MS TraceEnable off"*** This task was performed using the following command: ***"sudo nano /etc/apache2/apache2.conf"***.

A003	/etc/apache2/apache2.conf file edited.	/etc/apache2/apache2.conf file edited to incorporate the additional text: ***"#Hide Server Tokens and Signatures – MS*** ***ServerTokens Prod*** ***ServerSignature Off*** ***<IfModule mod_headers.c>*** ***Header unset Server*** ***Header unset X-powered-By*** ***</IfModule>"*** using the command: ***"sudo nano /etc/apache2/apache2.conf".***
A004	"apachegroup" group created.	"apachegroup" group created using the following command: ***"sudo groupadd apachegroup"***
A005	"Apacheuser" user created.	"apachegroup" user created using the following command: ***"sudo useradd –d /var/www/ -g apachegroup –s /sbin/nologin apacheuser"*** Password for "apacheuser" created using the following command: ***"Sudo passwd apacheuser"*** Password was set to ***"Student@2023"*** as per assignment requirements.
A006	/etc/apache2/envvars file edited.	/etc/apache2/envvars file edited for the purpose of locating the following lines of text: ***"Export APACHE_RUN_USER"*** ***"Export APACHE_RUN_GROUP"*** This was done using the following command: ***"sudo nano /etc/apache2/envvars"*** After locating these lines of text in the text editor, the text was altered to display the following text: ***"Export APACHE_RUN_USER=apacheuser"*** ***"Export APACHE_RUN_GROUP=apachegroup"***

A007	Applied text editor changes.	After making various changes to the files detailed in the change management log, I used the following command to apply the changes: ***"sudo systemctl restart apache2"***
A008	"boss" username created for MySQL.	The following command was used to create the "boss" user: ***"CREATE USER 'boss'@'localhost' IDENTIFIED BY 'Student@2023';"***
A009	All privileges granted for "boss" user.	The following command was used to grant all privileges to "boss" user: **"GRANT ALL PRIVILEGES ON *.* TO 'boss'@'localhost' WITH GRANT OPTION;"**
A010	Created "inet" database in MySQL.	The following command was used to create the "inet" database in MySQL: **"CREATE DATABASE inet;"**
A011	Created a "members" table in the "inet" database in MySQL and created columns.	The following command was used to create the "members" table and the columns in the "inet" database: **"CREATE TABLE members (user_id INT(11) AUTO_INCREMENT, username VARCHAR(50), password VARCHAR(50), PRIMARY KEY (user_id));**
A012	Inserted data entries into columns of "members" table in "inet" database.	The following commands were used to implement the data entried into the columns of the "members" table in the "inet" database: **"INSERT INTO members (user_id, username, password) VALUES (1, 'Bruce', MD5('student')), (2, 'Charlotte', MD5('student'));**
A013	Created a bash script to create databases.	Bash script was created to create two databases, the "staging" database and the "production" database. The "staging" database consisted of the "task" table with the following columns and data: **CREATE DATABASE IF NOT EXISTS staging; USE staging; CREATE TABLE IF NOT EXISTS tasks (task_id INT AUTO_INCREMENT PRIMARY KEY, title VARCHAR(255) NOT NULL, start_date DATE, due_date DATE, status TINYINT NOT NULL, priority TINYINT NOT NULL, description TEXT**

		); **INSERT INTO tasks (title, start_date, due_date, status, priority, description)** **VALUES** **('task1', '2020-07-01', '2020-07-31', 1, 1, 'this is the first task'),** **('task2', '2020-08-01', '2020-08-31', 2, 2, 'this is the second task'),** **('task3', '2020-09-01', '2020-09-30', 1, 1, 'this is the third task'),** **('task4', '2020-10-01', '2020-10-31', 1, 1, 'this is fourth task');** The "production" database consisted of the "completed" table with the following columns and data: **CREATE DATABASE IF NOT EXISTS production;** **USE production;** **CREATE TABLE IF NOT EXISTS completed (task_id INT AUTO_INCREMENT PRIMARY KEY, task_name VARCHAR(255) NOT NULL,** **finished_date DATE, status TEXT, description TEXT** **);** **USE production;** **INSERT INTO completed (task_name, finished_date, status, description) VALUES** **('task1', '2020-07-31', 'done', 'task one finished'),** **('task2', '2020-08-31', 'completed', 'task two finished'),** **('task3', '2020-09-30', 'done', 'task three finished'),** **('task4', '2020-10-31', 'done', 'task four finished');**
A014	Package lists updated.	The following command was used to update the package lists: **"sudo apt-get update"**
A015	ACL installed onto ubuntu server.	The following command was used to install ACL onto ubuntu server: **"sudo apt-get install acl"**
A016	Created the "Data" directory.	The following command was used to create the "Data" directory: **"mkdir Data"**
A017	Assigned "rw" permissions for my "msmith" user for Data directory.	The following command was used to assign "rw" permissions to my "msmith" user for my "Data" directory: **"setfacl -m u:msmith:rw Data"**

A018	Created "ACLpermissionstest" file in "Data" directory.	The following command was used to Create the "ACLpermissionstest" file in "Data" directory: **"~/Data$ touch ACLpermissionstest"**
A019	Backed up MySQL Databases to a backup file called "restore_file.sql"	The following commands were used to back up the MySQL Databases to a backup file called "restore_file.sql": **"mysqldump -u boss -p inet > restore_file.sql"** **"mysqldump -u boss -p mysql > restore_file.sql"** **"mysqldump -u boss -p production > restore_file.sql"** **"mysqldump -u boss -p staging > restore_file.sql"** **"mysqldump -u boss -p sys > restore_file.sql"**
A020	Set ownership / permissions for the "msmith" user for the "Data" directory.	The following commands were used to set ownership / permissions for the "msmith" user for the "Data" directory: **"Sudo chown -R msmith:msmith /home/msmith/Data"** **"sudo chmod 750 /home/msmith/Data"**
A021	MySQL data copied to "Data" directory.	The following command was used to copy the MySQL data to the "Data" directory: **"sudor sync -av /var/lib/mysql/ /home/msmith/Data"**
A022	Edited MySQL Configuration file.	The following lines of code were implemented into the MySQL configuration file: **"Socket=/home/msmith/Data/mysqld.sock"** **"datadir=/home/msmith/Data"**
A023	Edited MySQL Client Configuration file.	The following lines of code were implemented into the MySQL client configuration file: **"[client]"** **"Socket=/home/msmith/Data/mysql.sock"**
A024	Edited "/etc/rsyslog.conf" text file to provide UDP and TCP transport reception on port 514	The following lines of code were added to the "/etc/rsyslog.conf" text file to provide UDP and TCP transport reception on port 514: **"$ModLoad imudp"** **"$UDPServerRun 514"** **"$ModLoad imtcp"** **"$Input TCPServerRun 514"**
A025	Enabled UFW.	The following command was used to enable UFW: **"sudo ufw enable"**
A026	Configured TCP/UDP ports to receive incoming traffic	The following commands were used to configure TCP/UDP ports to receive incoming traffic: **"sudo ufw allow 514/udp"** **"sudo ufw allow 514/tcp"**

A027	Installed libuser package.	The following command was used to install the libuser package: ***"sudo apt install libuser"***
A028	Installed members package	The following command was used to install the members package: ***"sudo apt install members"***
A029	Created Directory to back up MySQL Databases	Created a directory called "MySQLBackupDirectory" to back up MySQL databases. The following command was used to create this directory: ***"Mkdir MySQLBackupDirectory"***
A030	Created a bash script to back up databases and gave it executable permissions.	Created a bash script called "backup.sh" to back up databases. The following commands were used to create this script and give it executable permissions: ***"sudo nano backup.sh"*** ***"sudo chmod +x backup.sh"***
A031	Modified cron job scheduler utility to run "backup.sh"script every Friday at 10:00PM	Modified cron job scheduler utility to run "backup.sh"script every Friday at 10:00PM using the ***"crontab -e"*** command. The following line of code was implemented to execute this: ***"0 22 * * 5 /home/msmith/MySQLBackupDirectory/backup.sh"***
A032	Created Directory for grandfather, father, and son backups of system data.	Created a directory called "DataBackupDirectory" to store grandfather, father, and son backups of system data. The following command was used to create this directory: ***"Mkdir DataBackupDirectory"***
A033	Created a bash script to store grandfather, father, and son backups of system data and gave it executable permissions.	Created a bash script called "backup.sh" to store grandfather, father, and son backups of system data. The following commands were used to create this script and give it executable permissions: ***"sudo nano GFSDataBackupScript.sh"*** ***"sudo chmod +x GFSDataBackupScript.sh"***
A034	Modified cron job scheduler utility to run "GFSDataBackupScript.sh"s cript every morning at 2:00AM	Modified cron job scheduler utility to run "GFSDataBackupScript.sh"script every morning at 2:00AM using the ***"crontab -e"*** command. The following line of code was implemented to execute this: ***"0 2 * * * /home/msmith/ GFSDataBackupScript.sh"***
A035	Created a directory to save MySQL status files every time the system reboots	Created a directory called "MySQLStatusDirectory" to save MySQL status files every time the system reboots. The following command was used to create this:

		"mkdir MySQLStatusDirectory"
A036	Created a bash script to save the MySQL status to a file and gave it executable permissions.	Created a bash script called "MySQLStatusReport.sh" to save the MySQL status to a file. The following commands were used to create this script and give it executable permissions: ***"sudo nano MySQLStatusReport.sh"*** ***"sudo chmod +x MySQLStatusReport.sh"***
A037	Modified cron job scheduler utility to run "MySQLStatusReport.sh"script every time the system reboots	Modified cron job scheduler utility to run "MySQLStatusReport.sh"script every time the system reboots using the ***"crontab -e"*** command. The following line of code was implemented to execute this: ***"@reboot /home/msmith/ MySQLStatusReport.sh"***
A038	Installed sysbench to create benchmark reports	Installed sysbench to create benchmark reports. The following command was used to install it: ***"sudo apt install sysbench"***
A039	Created directory to store benchmark report files.	Created a directory to store CPU and Memory benchmark report files. The following command was used to create it: ***"mkdir BenchmarkReportDirectory"***
A040	Created a bash script to create CPU and Memory benchmark reports and save it to the created directory and gave it executable permissions.	Created a bash script called "BenchmarkReport.sh" to create CPU and Memory benchmark reports and save it to the "BenchmarkReportDirectory" directory. The following commands were used to create this script and give it executable permissions: ***"sudo nano BenchmarkReport.sh"*** ***"sudo chmod +x BenchmarkReport.sh"***
A041	Modified cron job scheduler utility to run "BenchmarkReport.sh"script every Monday at 12:01AM.	Modified cron job scheduler utility to run "BenchmarkReport.sh"script every Monday at 12:01AM. The following command was implemented to execute this: ***"1 0 * * 1 /home/msmith/BenchmarkReport.sh"***
A042	Updated package lists before Docker installation	Updated package lists before Docker installation. The following command was used to perform this: ***"sudo apt update"***
A043	Installed prerequisite packages that allow apt to use packages over HTTPS	Installed prerequisite packages that allow apt to use packages over HTTPS. The following command was used to do this: ***"Sudo apt install apt-transport-https ca-certificates curl software-properties-common"***
A044	added the GPG key for the implementation of the official Docker repository	added the GPG key for the implementation of the official Docker repository. The following command was used to do this:

		"curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add –"
A045	added Docker repository to APT sources.	added Docker repository to APT sources. The following command was used to do this: ***"sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu focal stable"***
A046	Installed Docker	Installed Docker onto the Ubuntu server. The following command was used to do this: ***"Sudo apt install docker-ce"***
A047	Prerequisite packages installed for Nagios Core Installation	The following commands were used to install prerequisite packages for the installation of Nagios Core: ***"sudo apt-get update"*** ***"sudo apt-get install -y autoconf gcc libc6 make wget unzip apache2 php libapache2-mod-php7.4 libgd-dev"*** ***"sudo apt-get install openssl libssl-dev"***
A048	downloaded Nagio source:	The following command was used to download the Nagio source: ***"cd /tmp"*** ***"wget -O nagioscore.tar.gz https://github.com/NagiosEnterprises/nagioscore/archive/nagios-4.4.14.tar.gz"*** ***"tar xzf nagioscore.tar.gz"***
A049	Configured and constructed Nagios.	The following commands were used to configure and construct Nagios: ***"cd /tmp/nagioscore-nagios-4.4.14/"*** ***"sudo ./configure --with-httpd-conf=/etc/apache2/sites-enabled"*** ***"sudo make all"***
A050	Created a Nagios user and group.	The following commands were used to create a Nagios user and group: ***"sudo make install-groups-users"*** ***"sudo usermod -a -G nagios www-data"***
A051	Installed binary files, HTML files, and CGI's for Nagios installation.	The following commands were used to Install binary files, HTML files, and CGI's for the Nagios installation: ***"sudo make install"***

A052	Installed service / daemon files and configured them to start on boot.	The following commands were used to Install service / daemon files and configure them to start on boot: ***"sudo make install-daemoninit"***
A053	Installed / configured external command file.	The following commands were used to Install / configure the external command file: ***"sudo make install-commandmode"***
A054	Installed configuration files.	The following commands were used to install the configuration files: ***"sudo make install-config"***
A055	Installed Apache configuration files.	The following commands were used to install the Apache configuration files: ***"sudo make install-webconf"*** ***"sudo a2enmod rewrite"*** ***"sudo a2enmod cgi"***
A056	Configured firewall for Nagios Core web interface.	The following commands were used to configure the firewall for the Nagios Core web interface: ***"sudo ufw allow Apache"*** ***"sudo ufw reload"***
A057	Created Apache user account for Nagios Core.	The following commands were used to create an Apache user for the Nagios Core: ***"sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin"***
A058	Installed Nagios plugins.	The following commands were used to install the Nagios plugins: ***"apt install monitoring-plugins"***
A059	Created a symbolic link for Nagios plugins.	The following commands were used to create a symbolic link to the "/usr/local/nagios/libexec" directory for the Nagios plugins: ***"ln -s /usr/lib/nagios/plugins/* /usr/local/nagios/libexec"***
A060	Created MySQL Nagios user to be monitored from local host and granted all privileges to user.	The following commands were used to create a MySQL Nagios user that can be monitored from the local host and grant all privileges to the user: ***"CREATE USER 'nagios'@'localhost' IDENTIFIED BY 'Student@2023';"*** ***"GRANT ALL PRIVILEGES ON *.* TO 'nagios'@'localhost';"***

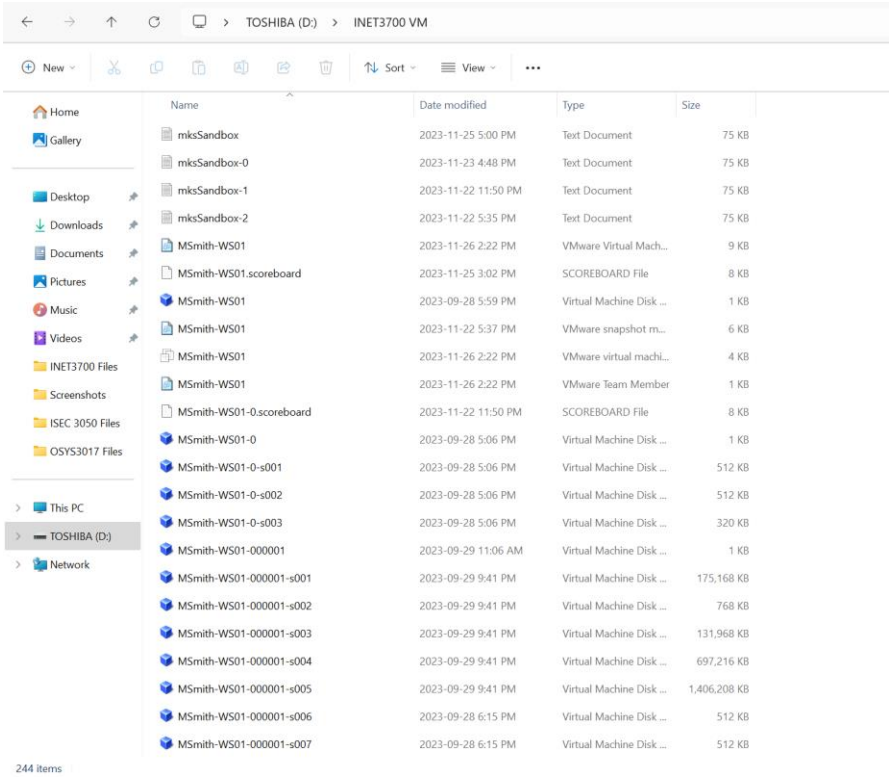
A061	Created Nagios user to be monitored from any source and granted all privileges to user.	The following commands were used to create a MySQL Nagios user that can be monitored from any source and grant all privileges to the user: ***"CREATE USER 'nagios'@'%' IDENTIFIED BY 'Student@2023';"*** ***"GRANT ALL PRIVILEGES ON *.* TO 'nagios'@'%';"***
A062	Downloaded the "check_mysql_health-2.2.2.tar.gz" plugin package.	The following commands were used to Download the "check_mysql_health-2.2.2.tar.gz" plugin package: ***"wget https://labs.consol.de/assets/downloads/nagios/check_mysql_health-2.2.2.tar.gz"***
A063	Extracted the "check_mysql_health-2.2.2.tar.gz" plugin package.	The following commands were used to extract the "check_mysql_health-2.2.2.tar.gz" plugin package: ***"tar -zxvf check_mysql_health-2.2.2.tar.gz"***
A064	Installed and configured MySQL Nagios plugin.	The following commands were used to Install and configure the MySQL Nagios plugin: ***"./configure --prefix=/usr/local/nagios --with-nagios-user=nagios --with-nagios-group=nagios --with-perl=/usr/bin/perl"*** ***"sudo make"*** ***"sudo make install"***
A065	Installed DBD Module for Nagios MySQL plugin.	The following commands were used to Install the DBD Module for the Nagios MySQL plugin: ***"sudo apt-get install -y libdbd-mysql-perl"***

Task 5: Gold Copy

Proof of Gold Copy backed up onto external drive provided below.

Screenshot of updated VM snapshot taken:

Screenshot of updated Gold Copy:



Conclusion

In this document, I have successfully demonstrated my understanding of network OS and scripting. I have achieved this by providing detailed explanations and labelled screenshots for the required tasks that are Change Management, Backups, and Nagios.

References

Website Names, Titles, Authors, Dates, and Links of Articles Used Provided Below.

Website Name: Tech Target

Title: What is a Full Backup?

Author: Rich Castagna

Date: December 2022

Link: [What is a Full Backup? Definition from TechTarget]

Website Name: Acronis

Title: What is the difference between incremental, differential, and full backup?

Author: N/A

Date: September 28th, 2023

Link: Incremental vs. Differential vs. Full Backup - A Comparison Guide (acronis.com)

Website Name: Nagios Support Knowledgebase

Title: Nagios Core - Installing Nagios Core From Source

Author: RSpielman

Date: August 1st, 2023

Link: Nagios Core - Installing Nagios Core From Source

Website Name: Kifarunix

Title: Install and Setup Nagios on Ubuntu 22.04

Author: Jay Decrame

Date: January 8th, 2023

Link: Install and Setup Nagios on Ubuntu 22.04 - kifarunix.com

Website Name: Backup4All

Title: What is a Mirror Backup?

Author: Lorant (Softland)

Date: October 14th, 2022

Link: What is a Mirror Backup? - Backup4all

Website Name: Data Numen

Title: What are Copy-Only Backups and How to Create Them with SQL Server Management Studio

Author: N/A

Date: March 16th, 2020

Link: What are Copy-Only Backups and How to Create Them with SQL Server Management Studio (datanumen.com)

Website Name: Micro Focus Open Text

Title: Microsoft SQL Server Database Backups

Author: N/A

Date: N/A

Link: Microsoft SQL Server Database Backups (microfocus.com)

Website Name: Microsoft

Title: Partial Backups (SQL Server)

Author: N/A

Date: February 28th, 2023

Link: Partial Backups (SQL Server) - SQL Server | Microsoft Learn

Website Name: Atlassian

Title: Change Management Roles and Responsibilities

Author: N/A

Date: N/A

Link: Change Management Roles and Responsibilities | Atlassian

Website Name: File Center

Title: The Foolproof Guide to Change Management Documentation

Author: Lucion Technologies

Date: June 20th, 2023

Link: The Foolproof Guide to Change Management Documentation (filecenter.com)

Website Name: BrightHub

Title: Important Documents Used in Change Management

Author: N/A

Date: N/A

Link: Important Documents Used in Change Management - BrightHub Project Management (brighthubpm.com)

Website Name:

Title: How to monitor MySQL using Nagios?

Author: Psychz - Ramesh

Date: October 12th, 2017

Link: How to monitor MySQL using Nagios? (psychz.net)

Website Name: Youtube

Title: Monitor MySQL Server with Nagios

Author: SysAdmGirl

Date: March 4th, 2017

Link: (109) Monitor MySQL Server with Nagios - YouTube

Website Name: ZoomAdmin

Title: How To Install "libdbd-mysql-perl" Package on Ubuntu

Author: N/A

Date: N/A

Link: How to install libdbd-mysql-perl ubuntu package on Ubuntu 20.04/Ubuntu 18.04/Ubuntu 19.04/Ubuntu 16.04 (zoomadmin.com)