Troubleshooting Nagios Checks
IN719 Systems Administration

Introduction

After completing the last lab, your Nagios server is monitoring your database server. This includes checking the status of your MariaDB service. The thing is, those checks are failing. Since this is a new check, we have to consider three possibilities.

1. The MariaDB service is not functioning properly;
2. The Nagios check is not configured properly;
3. Both.

We will have to investigate to get to the bottom of this. Spoiler alert: You could just skip to the last section and and get the issue sorted, but that's not the point. Our real goal here is to walk through how Nagios performs checks so that we can troubleshoot other issues in the future.

1 Check the MariaDB service

It's a good idea when troubleshooting to start with the most obvious thing. Log onto your db server and make sure that the MariaDB service is running. If it's not, make a note to check your Puppet configuration on your management server. However, even when the database server is running you will find that the Nagios check still fails. The next thing to do is to examine our Nagios configuration for performing this service check.

2 Review our configuration file

If you have been following the labs to this point, then the check for the MariaDB service is defined in `/etc/nagios3/conf.d/ppt_se`
Check that file for two things in particular:

1. Find the hostgroup to which the check applies;
2. See what the `check_command` is.

Once you know what the hostgroup is, follow that thread to be sure that the database host is properly defined and that it is a member of the right hostgroup.

Next let's dig into what that check command does:

3 Test the check command manually

We see that our check command is probably `check_mysql`. That command is defined in a file in `/etc/nagios-plugins/config`. Use `grep` on the files in that directory to see which one defines `check_mysql`. Note that this definition shows that it executes a command `/usr/lib/nagios/plugins/check_mysql`, passing a host address that is automatically interpolated by Nagios and an optional second argument. Let's see what happens when we run the command ourselves. Simply enter the command

```
/usr/lib/nagios/plugins/check_mysql -H group00db.foo.org.nz
```

Note the output and compare it to what we see on the Nagios web interface. If they match, then Nagios is correctly reporting the results of this check. This is good, because that means we understand what is happening.

Now we need to see why that check fails.

4 Investigate MariaDB

With a little help from the web we can learn that the MariaDB service doesn't allow any connections from remote hosts by default. So we need to see how modify our database server configuration to allow connections from our Nagios server. Check the documentation to see how to do that and modify your Puppet module to apply the new configuration.

That's not enough, however, because the `check_mysql` command we are using tries to connect without a username or password. That's not going to work. But the same config file that defines `check_mysql` defines another check,

```
/usr/lib/nagios/plugins/check_mysql -H '$HOSTADDRESS$' -u '$ARG1$' -p '$ARG2$' '$ARG3$'
```

We just need to modify our service configuration that we set up in the Nagios Puppet module. The host address variable is populated automatically, but we need to pass in `ARG1` and `ARG2`. We also need to set up a username and password in MariaDB. Do that first.

Here's a little trick, for security, we can create a database user that can only perform `SELECT` queries on a nonexistent database. That's enough for our Nagios check to succeed. To do this, log onto your database server and open a MariaDB shell, then run the following SQL statement:

```
GRANT ALL PRIVILEGES ON nagiosdb.* to 'nagios'@'group00mgmt.foo.org.nz' IDENTIFIED BY 'mypasswd';
```

/var/lib/

Before you modify the nagios service configuration in your Puppet module, verify that the check will pass by running `/var/lib/nagios/plugins/check_mysql`, passing in the right values for the `-H`, `-u`, and `-p` options.

5 Modify your check command

If everything is working to this point, then we expect that changing the check command in our Nagios service configuration in the Puppet module will finally sort things out. We know that the command should be `check_mysql_cmdlinecred`, but how do we pass in the username and password arguments? Like this:

```
check_command => 'check_mysql_cmdlinecred!nagios!mypasswd',
```

assuming the our username is `nagios` and our password is `mypasswd`.

Modify your Puppet module accordingly, apply it to your management service, and verify that your Nagios check is passing.