Journal 01

by Alexandru Cirlan

Overlook

For the project, we've chosen to continue the monitoring system using CheckMK. We continue this week by developing the virtual machines we need to test the system. Each teammate will create one of the following: a CPU hog, a RAM hog and disk storage hog. I've chosen to create a RAM hog.

Entry 01 | 2024-11-24

Creating a RAM hog

The only idea I have for this initially Is downloading google chrome, because it destroys RAM by itself. Though that probably wouldn't work, I decided to ask Mr. Jipity (ChatGPT). Here is the prompt I gave it:

> *Good evening, I would like to know if there is anyway, on a Debian system, to simulate RAM usage (and any other computer process if possible).*

It answered me with multiple methods, through the stress package or using python and bash. I decided to use the stress package, and it ended up working while also being relatively easy to use requiring only one command after installation. The best part about this is that you can decide however long the test needs to run, and we could probably simulate it with SysteMD on startup.

Set Up:

1. Install the stress package by running: sudo apt install stress
2. Run the command : stress –vm 2 –vm-bytes [QT]M –timeout [Time(seconds)]
   ex: stress –vm 2 –vm-bytes 1024M –timeout 60

This created 2 workers that would each eat up 1024 MB of RAM for 60 seconds.

Automating Ram Hogging

All I had to do was create a .sh file with the command up above (with a longer timer so I could see the difference) and link it to a .service file and then enable it using systemd.

Set Up:

1. Create .sh file wherever:
   ex: (in home directory) sudo nano [file-name].sh
   and add the commands in it.
2. Make sure to give it execution permission:
   ex: sudo chmod +x /path/to/file-name.sh
3. Create a .service file in the path: /etc/sytstemd/system/
   ex: (in the path stated above) sudo nano [file-name].sh

4. Add this inside the file and save:

```
[Unit]
Description=My Startup Script

[Service]
ExecStart=/path/to/file-name.sh
Restart=on-failure

[Install]
WantedBy=multi-user.target
```

5. Reload systemd: sudo systemctl daemon-reload
6. Enable the service by running: sudo systemctl enable [file-name].service

Running normally my virtual machine used 2.3GB of memory, but after adding 2GB worth of workers, it spiked up to 4GB.