

---

# **NEMS Linux Documentation Documentation**

***Release 1.6***

**Robbie Ferguson**

**Apr 22, 2024**

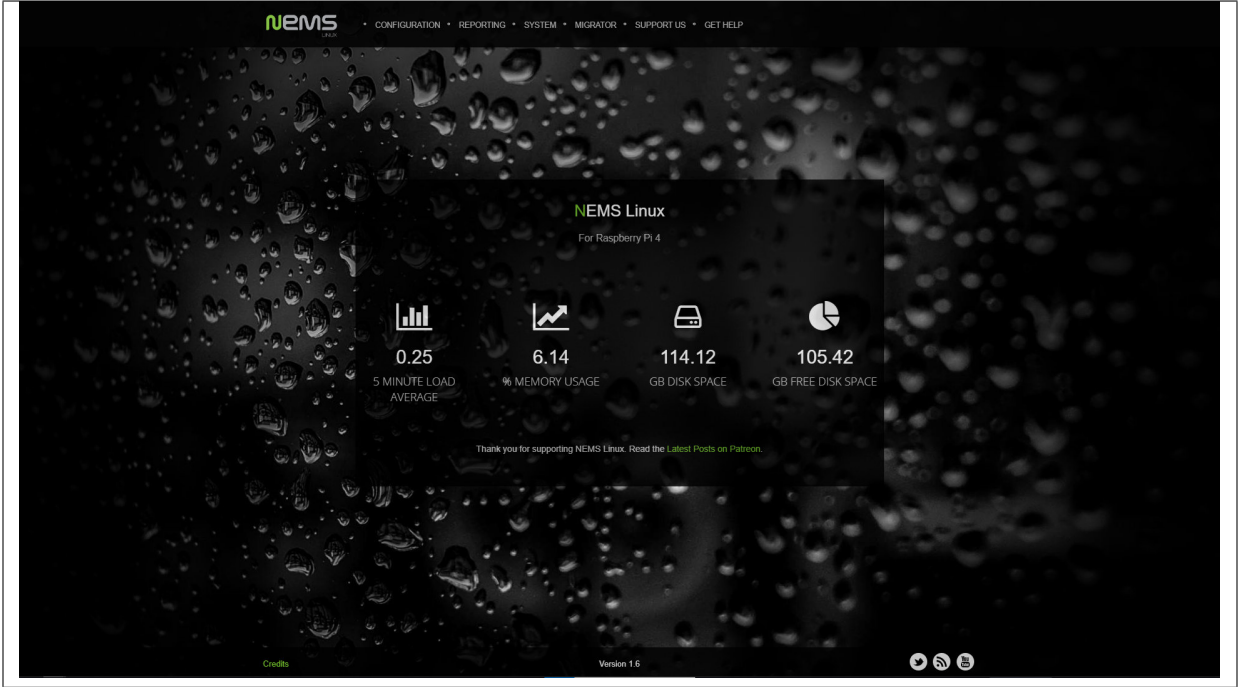




# GETTING STARTED GUIDE

<b>1</b>	<b>Preface</b>	<b>3</b>
<b>2</b>	<b>NEMS Linux Getting Started Guide</b>	<b>5</b>
2.1	Introduction	5
2.2	Table of Contents	5
2.3	Conclusion	30
2.4	What's Next	30
<b>3</b>	<b>Documentation</b>	<b>31</b>
3.1	NEMS Administrator User	31
3.2	Reboot or Power Off NEMS Server	32
3.3	Included Check Commands	33
3.4	NEMS Linux Tips: Backup Your NEMS Configuration Automatically	98
3.5	NEMS Navigation Menu	99
3.6	Features: Port 9590	102
3.7	Connect to NEMS Server Over SSH	102
3.8	Frequently Asked Questions	102
3.9	NEMS Migrator Local Backup	103
3.10	Available Platforms	104
3.11	nems-tools: GPIO Extender	114
3.12	nems-tools: NEMS Extender OS	115
3.13	nems-tools: NEMS Hero	116
3.14	nems-tools: Warning Light	116
3.15	NEMS Linux Notifications	118
3.16	NEMS Commands and Special Features	127
3.17	Network Configuration	127
3.18	Using Gmail SMTP For Nagios Notification Sending	131
3.19	Using MS Outlook.com SMTP For Nagios Notifications	132
3.20	NEMS Server Overview	132
3.21	NEMS System Settings Tool (SST)	134
3.22	NConf	134
3.23	Adagios	134
3.24	NEMS Mobile UI	135
3.25	NEMS TV Dashboard	135
3.26	NEMS Tactical Overview	136
3.27	Monitorix	136
3.28	Cockpit	137
3.29	Monit Service Monitor	137
3.30	NEMS Cloud Services	138
3.31	Monitoring Microsoft Windows Hosts with NEMS Linux	143

3.32	Monitoring Linux Hosts with NEMS Linux . . . . .	146
3.33	Upgrade NEMS Linux to Newer Version . . . . .	147
3.34	NEMS Migrator: Upgrade NEMS 1.0 or nagiospi to latest NEMS . . . . .	148
3.35	SNMP . . . . .	149
3.36	Multi Router Traffic Grapher (MRTG) . . . . .	154
3.37	Copying OS Icons to NEMS . . . . .	156
3.38	nems-api . . . . .	158
3.39	NEMS Accessories . . . . .	169
3.40	NEMS Linux Changelogs . . . . .	178
3.41	NEMSeOS Changelogs . . . . .	214
3.42	NEMS Mesh: The Self-Hosted NEMS Cloud . . . . .	215
3.43	NEMS Linux To Do List . . . . .	215
3.44	Known Issues . . . . .	217
3.45	How To Report Issues . . . . .	218
3.46	NEMS SSL and Self-Signed Certificates . . . . .	219
3.47	Why support may ask for your backup.nems file . . . . .	220
3.48	How NEMS Linux Uses ZRAM To Maximize Memory . . . . .	221
3.49	NEMS Licensing . . . . .	222
3.50	NEMS Branding . . . . .	222
3.51	NEMS Anonymous Stats . . . . .	224
3.52	NEMS Linux Source Code . . . . .	225
3.53	NEMS Linux Release MD5 Checksums . . . . .	228
3.54	NEMS Linux Vendor Branding . . . . .	235
3.55	Resolving DNS Hostnames on LAN . . . . .	237
3.56	Documentation Contributors . . . . .	237
3.57	NEMS SaaS . . . . .	238
<b>4</b>	<b>Important Links</b>	<b>243</b>
<b>5</b>	<b>Contribute to Documentation</b>	<b>245</b>
<b>6</b>	<b>Support</b>	<b>247</b>
<b>7</b>	<b>Credits</b>	<b>249</b>
<b>8</b>	<b>Patrons</b>	<b>251</b>
<b>9</b>	<b>License</b>	<b>253</b>
	<b>Index</b>	<b>255</b>





**PREFACE**

The NEMS Linux documentation is a work in progress, and we are constantly updating it. If you find any problems or inconsistencies, please either provide a pull request on [GitHub](#), file an issue on [GitHub](#), or contact us on [Discord](#).



## NEMS LINUX GETTING STARTED GUIDE

### 2.1 Introduction

The NEMS Linux Getting Started Guide is an introductory course in understanding and using NEMS Linux. From the principles that drive the project, to understanding how the various components work together.

By following this easy to understand guide, you will gain the fundamental knowledge and skills required to use NEMS Linux effectively.

This guide is a primer and not comprehensive or even overly technical. It is intended as an introduction to NEMS Linux for new users, or those who wish to better understand how to use NEMS Linux at an entry level.

### 2.2 Table of Contents

#### 2.2.1 Introduction to NEMS Linux

##### NEMS Linux

NEMS */nmz/* stands for “Nagios Enterprise Monitoring Server” and is a Linux distribution built as an appliance. It simplifies initial deployment and configuration of an enterprise monitoring server powered by Nagios Core and adds many complex features, integrating several open source projects to create a well rounded, fully functional appliance that is scalable for any sized network. NEMS is not just a “pre-installed Nagios Core distro,” but rather is an entire Linux distribution built around providing a powerful and feature-rich Nagios-based appliance.

##### How NEMS Linux Began

Nagios® Core™—which I’ll simply refer to as “Nagios” throughout this article—is a free, open source server application which monitors hosts and services that you specify, alerting you when things go bad and when they get better. I’ve been using Nagios for many years. If I had to hazard a guess as to when I started using it, I’d say it was around 2006.

My wife and I ran a small computer service company out of our home in those days, and in order to keep track of my customer sites and be as proactive as I could be, I had a beast of a computer in the garage keeping check on my customers’ hard drive space, backup state, CPU load, and antivirus definition updates, as well as various other services.

I remember setting up that old Nagios server. The process was onerous, and all the configuration was done through the Linux terminal by opening config files in vi. One malformed syntax and Nagios would fail to start. I made it work, and if anyone has ever doubted my nerdiness, they are clearly mistaken. Ah, who am I kidding? Nobody has ever doubted my nerdiness. As the years went on and my support business customer base continued to grow, I began repurposing old hardware, installing an independent Nagios server at each client site. This worked very well.

I received my first Raspberry Pi in 2014. After it sat on the shelf for a year, I began to consider possible ways I could put it to use. I realized that the power consumption, rack space, and noise of these old Nagios servers was an incredible waste of resources. I was convinced that a single-board computer could make an excellent Nagios server, and began tinkering.

Why reinvent the wheel? Ryan Siegel's NagiosPi image was out-of-the-box ready and gaining popularity. I started using it, but quickly became dismayed by the state of the distro, which appeared to be aging and was not being updated at a pace suitable for business use. It was a wonderful starting point, but felt in some important ways like an incomplete product. I began working on my own rebuild of NagiosPi, calling it NEMS; short for "Nagios Enterprise Monitoring Server".



Fig. 1: This customer Nagios server was replaced with the first ever NEMS server in 2016.

I'm a coder in my professional life. I develop server-side applications, mainly for web. Beyond building on a more current software base, the first thing I'd set out to do was build a responsive browser-based UI for NEMS, bringing all the components of NagiosPi together into a single interface. This later became the NEMS Dashboard, also known by its GitHub repository name, "nems-www."

Upon releasing NEMS to the public through my blog, Siegel himself said, "I'd love to upgrade NagiosPi, but i don't have [the] ability to make a GUI that can beat that of NEMS. I strongly feel that it has always been a necessary addition to NagiosPi and NEMS was able to deliver what is essentially an updated and improved version of NagiosPi. No reason not to start using NEMS for the time being. Nice work Robbie!" I didn't stop there, and in the wonderful spirit of community, Siegel even pitched in during the development of NEMS 1.2 in early 2017, bringing many additional Windows checks to NEMS.

With a new major release of NEMS every six months and rolling updates in between, NEMS is a Debian-based distro with Nagios Core at its heart. Having upgraded and maintained NEMS with a current software base, this also means things like an up-to-date version of PHP, current SSL certificates, and a suite of customized software optimized to work on a modern server. For example, NConf (a very useful tool for configuring Nagios) stopped development many years ago, so it only worked on PHP 5.3 or less. Therefore, I forked it and reworked the code to support PHP 7.0+. Of course, I made some other improvements along the way and now maintain an active fork of the product that is many commits ahead.

NEMS Linux, as it is now called (I had to find a dot-com, after all) takes the most modern network asset monitoring and



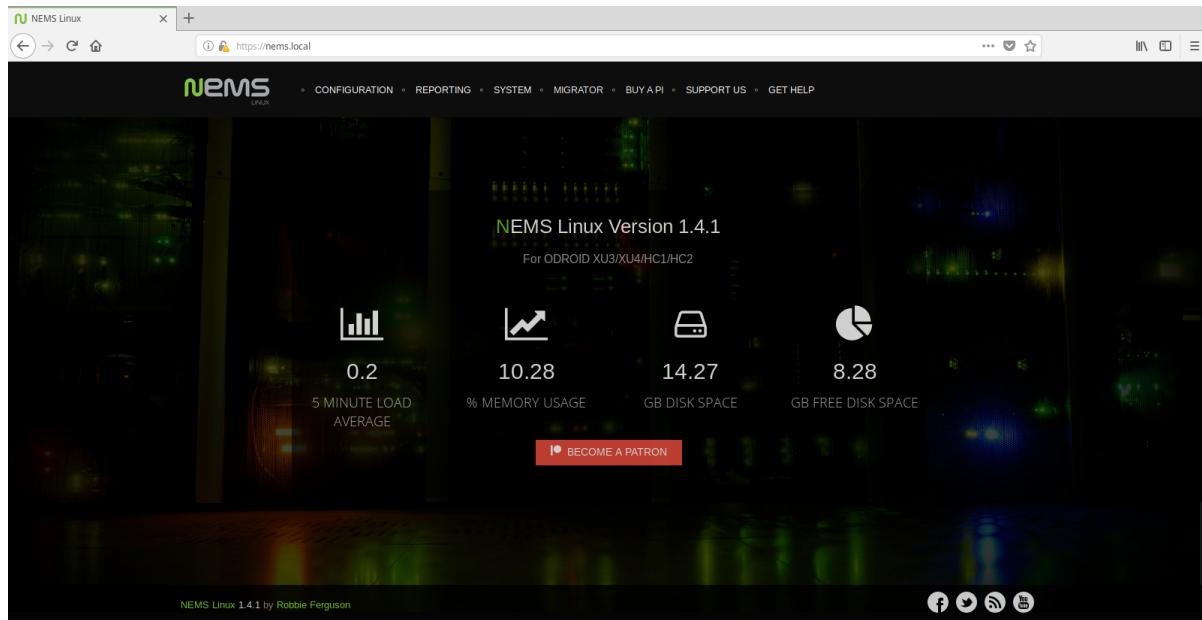


Fig. 2: An early screenshot of NEMS Linux’s Dashboard, circa September 2018.

does away with the old Nagios scripting requirement. The scripts are still there, it’s just that you (the user) don’t ever have to see them or touch them. The whole thing is controlled, configured, and monitored through your web browser, with email, Telegram, or Pushover notifications all operational out of the box. It also has a JSON API, a TV display for your server room, and more.

NEMS Linux has evolved to be what I feel is the best out-of-the-box Nagios experience available. As a Nagios user myself, this is the Nagios server I have longed for. As NEMS has continued to grow, I set out to find a more powerful platform than the Raspberry Pi. Keep in mind that in September 2017 the Raspberry Pi was much less powerful than it is today, with far less RAM. So at that time, I began a quest to port NEMS Linux to the ODROID-XU4. After nearly a year of development, NEMS Linux became available on ODROID boards, later followed by ports to many other brand SBCs.

## NEMS Features

I’ve already touched on the obvious interface and UX improvements that NEMS Linux brings to the Nagios experience. Those are perhaps the key points as to what makes NEMS stand out, but it’s important to understand that NEMS Linux is far more than just Debian with Nagios installed. Let’s look at a small selection of additional features.

## NEMS Migrator

When focusing on building a distro for single board computers (SBC), I took very seriously the fact that SD cards can and likely will fail, and data can be lost. I wanted to create a way for users to be able to easily backup and restore their configuration. Out of that desire, Migrator was born.

Migrator allows you to backup your entire NEMS configuration (hosts, services, checks, system settings, etc.) via a samba share, https download, or even an optional offsite backup service. The backups can be encrypted, and only you know the decryption key. Should your device fail, you can write the image to a new SD card, restore your Migrator backup, and be up and running in under five minutes with all your settings intact. Migrator also makes it easy to transition from one platform to another. For example, having setup a NEMS Linux server on a Raspberry Pi 3, you can easily move to an ODROID-XU4 or even a Virtual Machine.

Another advantage that Migrator brings to the table is a simpler upgrade path: as new major releases of NEMS Linux are introduced, you can easily write the new NEMS image, import your backup, and be on the latest version of NEMS in just minutes with your configuration intact.

### Check-In Notifications

That's all fine and good, but what happens if your NEMS Server does fail? How would you know? With these concerns in mind, early adopters of NEMS Linux were apt to setting up multiple NEMS Servers: NEMS Servers to monitor NEMS Servers. I wanted to alleviate that need, so I created NEMS Check-In. This optional service (disabled by default) will check in with the NEMS Cloud Services server every few minutes. Should your NEMS Server fail to check in, you'll get an email warning you that your NEMS Server stopped responding.

### UI-Based Configuration with NEMS Configurator

NEMS Configurator (NConf) is what makes browser-based Nagios configuration possible. This customized version of the old NConf configuration tool brings a sophisticated front-end to the modern architecture of NEMS. Your entire Nagios configuration is done through this interface: from adding hosts to configuring your service checks. It's all done through an intuitive browser-based system.

With NEMS NConf, you will never have to look at a Nagios cfg file again!

### NEMS System Settings Tool (SST)

Speaking of doing away with Nagios config files, several Nagios configuration options have been moved to a tool called NEMS System Settings Tool, also referred to as NEMS SST. Items such as your SMTP server settings, domain user credentials, and other defaults are part of this interface.

So now that you know a little about what NEMS is and how it came about, let's dive in!

## 2.2.2 Install NEMS Linux

### NEMS Linux for SOHO/Hobbyist

#### Install NEMS Linux on a Raspberry Pi

##### What You Need

- Raspberry Pi 3 Model B or higher model RPi
- 16 GB or higher fast MicroSD card or USB Flash Drive
- A reliable, high-quality power supply for your RPi, preferably connected to a UPS
- An Ethernet cable

## Video Demonstration

### Instructions

#### Install NEMS Linux to MicroSD

- Download the latest version of NEMS Linux from <https://nemslinux.com/>
- Install the Raspberry Pi Imager from <https://www.raspberrypi.com/software/>
- Click “Choose OS”
- Click “Choose Custom” at the bottom of the list
- Browse to your downloaded copy of NEMS Linux
- Click “Choose Storage”
- Insert your MicroSD card or USB Flash Drive
- Carefully select your MicroSD card or USB Flash Drive
- Click “Write”

#### First Boot

- Connect the MicroSD card or USB Flash Drive containing NEMS Linux to your Raspberry Pi
- Connect the gigabit Ethernet port of the Raspberry Pi to your network using an Ethernet cable
- Power on the Raspberry Pi to boot your NEMS Server
- Wait approximately 5 minutes to perform first-boot operations

During the first-boot operation, the filesystem will be automatically resized to the capacity of your storage and the NEMS Server will reboot.

After the first-boot operations have completed, visit <https://nems.local/> in your web browser. If name resolution doesn't work, try the IP address of your NEMS device instead, which you can find in your router's DHCP leases table, or on a TV connected to your NEMS Server's HDMI port.

#### Install NEMS Linux on an Indiedroid Nova

##### What You Need

- Indiedroid Nova SBC
- 16 GB or higher fast MicroSD card
- A reliable, high-quality power supply for your Nova, preferably connected to a UPS
- An Ethernet cable

```
NEMS Linux 1.4.1
Platform:      ODR0ID XU3/XU4/HC1/HC2
Hostname:      nems.local
IP Address:    10.0.0.120
CPU Usage:     0.911303%
Disk Usage:    42%
Active Sessions: 1 user
Internet Status: Online

To login, use SSH or press CTRL-ALT-F2

For help, visit: docs.nemslinux.com
```

Fig. 3: NEMS Server details as shown on a connected TV circa September 2018.

## Video Demonstration

While this video was created for the Raspberry Pi version, it is exactly the same process for Indiedroid Nova (just use the correct download for the Indiedroid Nova).

## Instructions

### Install NEMS Linux to MicroSD

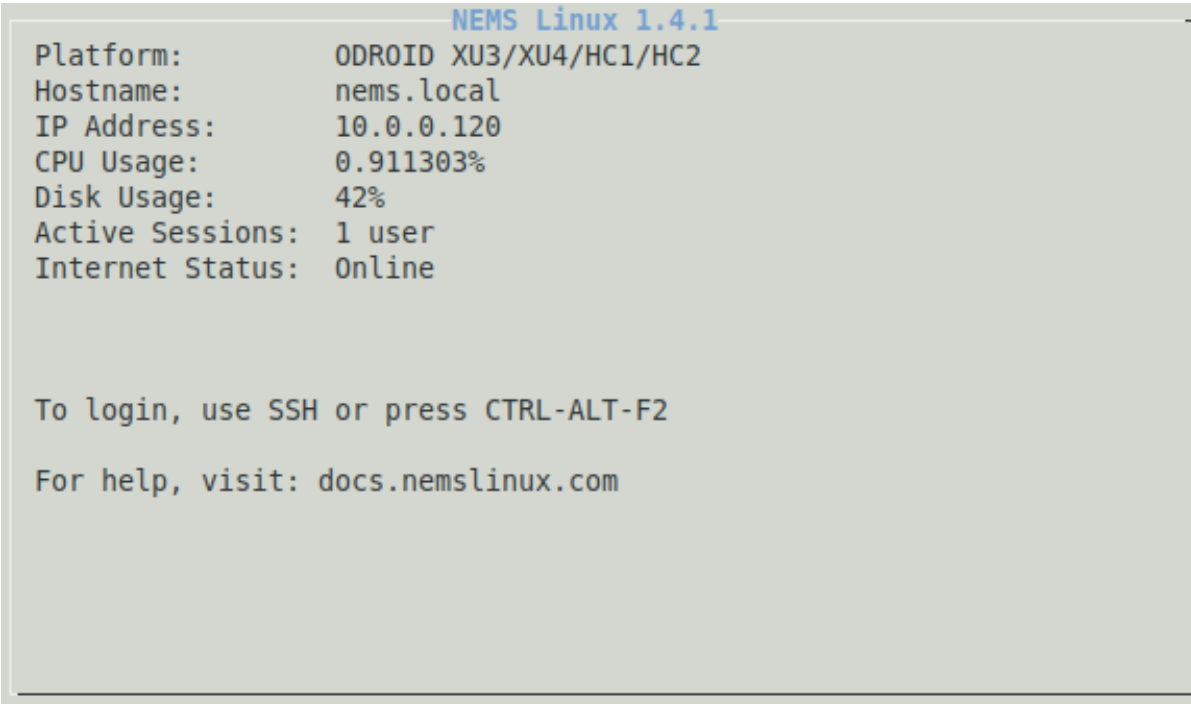
- Download the latest version of NEMS Linux from <https://nemslinux.com/>
- Install the Raspberry Pi Imager from <https://www.raspberrypi.com/software/>
- Click “Choose OS”
- Click “Choose Custom” at the bottom of the list
- Browse to your downloaded copy of NEMS Linux
- Click “Choose Storage”
- Insert your MicroSD card or USB Flash Drive
- Carefully select your MicroSD card or USB Flash Drive
- Click “Write”

## First Boot

- Connect the MicroSD card or USB Flash Drive containing NEMS Linux to your Indiedroid Nova
- Connect the gigabit Ethernet port of the Indiedroid Nova to your network using an Ethernet cable
- Power on the Indiedroid Nova to boot your NEMS Server
- Wait approximately 5 minutes to perform first-boot operations

During the first-boot operation, the filesystem will be automatically resized to the capacity of your storage and the NEMS Server will reboot.

After the first-boot operations have completed, visit <https://nems.local/> in your web browser. If name resolution doesn't work, try the IP address of your NEMS device instead, which you can find in your router's DHCP leases table, or on a TV connected to your NEMS Server's HDMI port.

A screenshot of a terminal window displaying system information for NEMS Linux 1.4.1. The text is as follows:

```
NEMS Linux 1.4.1
Platform:      ODROID XU3/XU4/HC1/HC2
Hostname:      nems.local
IP Address:    10.0.0.120
CPU Usage:     0.911303%
Disk Usage:    42%
Active Sessions: 1 user
Internet Status: Online

To login, use SSH or press CTRL-ALT-F2

For help, visit: docs.nemslinux.com
```

Fig. 4: NEMS Server details as shown on a connected TV circa September 2018.

## Install to eMMC

Now that you are booted into your NEMS Server via the MicroSD card, you may choose to utilize the Indiedroid Nova's built-in eMMC storage (more reliable, faster).

- Open the browser-based terminal in Cockpit or SSH to your NEMS Server.
- Type `sudo nems-install` and follow the prompts.

### 2.2.3 Initialization

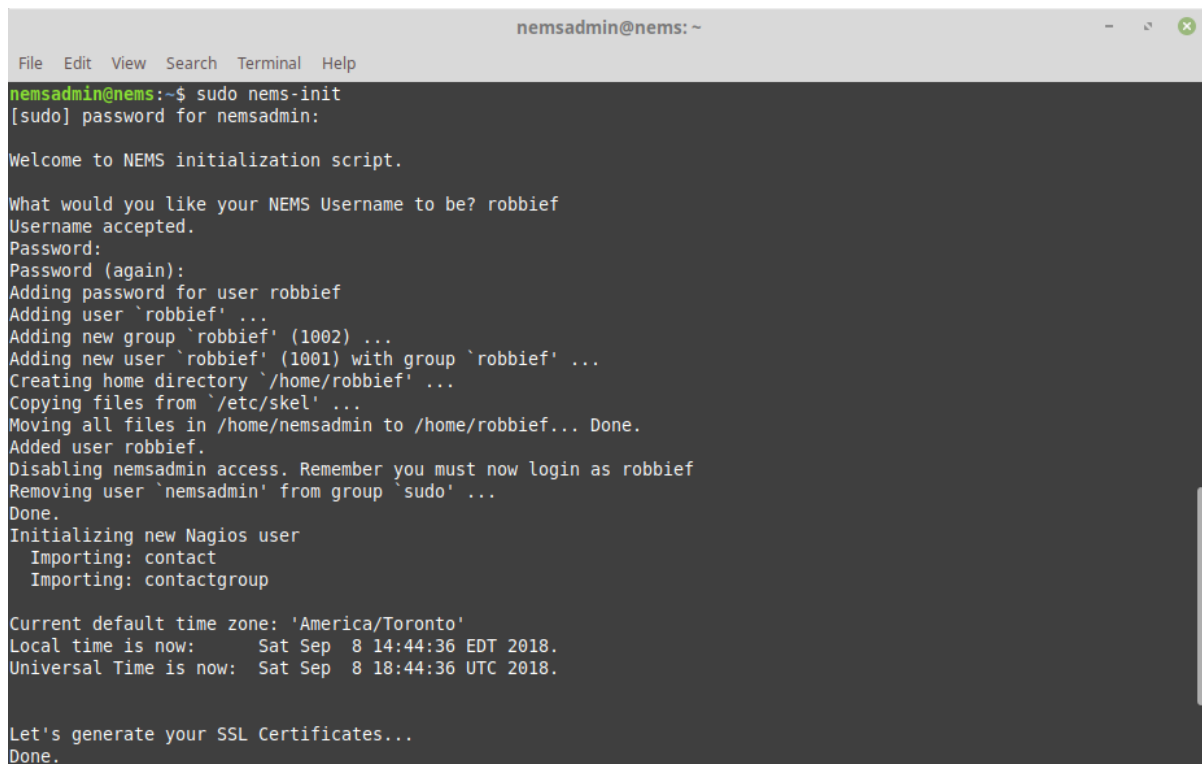
Generally speaking, the only time you'll really have to touch the Linux terminal on a NEMS server is during the [initialization procedure](#). This task works magic in automatically configuring your entire server in just a few seconds. It generates self-signed certificates so every NEMS Linux user has a unique certificate, allows you to configure your timezone, creates your Nagios admin user, your Linux account, and so on. To initialize your NEMS Linux server, connect to your server over SSH on the default Port 22 using the following credentials:

```
Username: nemsadmin
Password: nemsadmin
```

Once connected, type:

```
sudo nems-init
```

You'll be asked to enter the password again. Follow the prompts. All the complicated stuff is made easy.

A screenshot of a terminal window titled 'nemsadmin@nems: ~'. The terminal shows the execution of 'sudo nems-init'. The output includes a welcome message, prompts for a new NEMS username ('robbief') and password, and a series of system configuration steps: adding the user, adding a new group, adding the user to the group, creating a home directory, copying files from '/etc/skel', moving files from '/home/nemsadmin' to '/home/robbief', and disabling nemsadmin access. It also shows Nagios user initialization, current default time zone ('America/Toronto'), and local/universal time. Finally, it generates SSL certificates.

```
nemsadmin@nems:~$ sudo nems-init
[sudo] password for nemsadmin:

Welcome to NEMS initialization script.

What would you like your NEMS Username to be? robbief
Username accepted.
Password:
Password (again):
Adding password for user robbief
Adding user `robbief' ...
Adding new group `robbief' (1002) ...
Adding new user `robbief' (1001) with group `robbief' ...
Creating home directory `/home/robbief' ...
Copying files from `/etc/skel' ...
Moving all files in /home/nemsadmin to /home/robbief... Done.
Added user robbief.
Disabling nemsadmin access. Remember you must now login as robbief
Removing user `nemsadmin' from group `sudo' ...
Done.
Initializing new Nagios user
  Importing: contact
  Importing: contactgroup

Current default time zone: 'America/Toronto'
Local time is now:      Sat Sep  8 14:44:36 EDT 2018.
Universal Time is now:  Sat Sep  8 18:44:36 UTC 2018.

Let's generate your SSL Certificates...
Done.
```

Fig. 5: NEMS Initilization screen.

Congratulations! Your NEMS Linux server is now online and ready to monitor your network assets.

## 2.2.4 Connecting to Your NEMS Server

Now that your NEMS Linux server is up and running, you can access its entire feature set via your web browser. Simply point to `https://nems.local/` which should work in most networks out of the box, but if you need to, you can alternatively use the IP address of your NEMS server, which you can find in your router's DHCP leases table, or by connecting a TV directly to your NEMS Server.

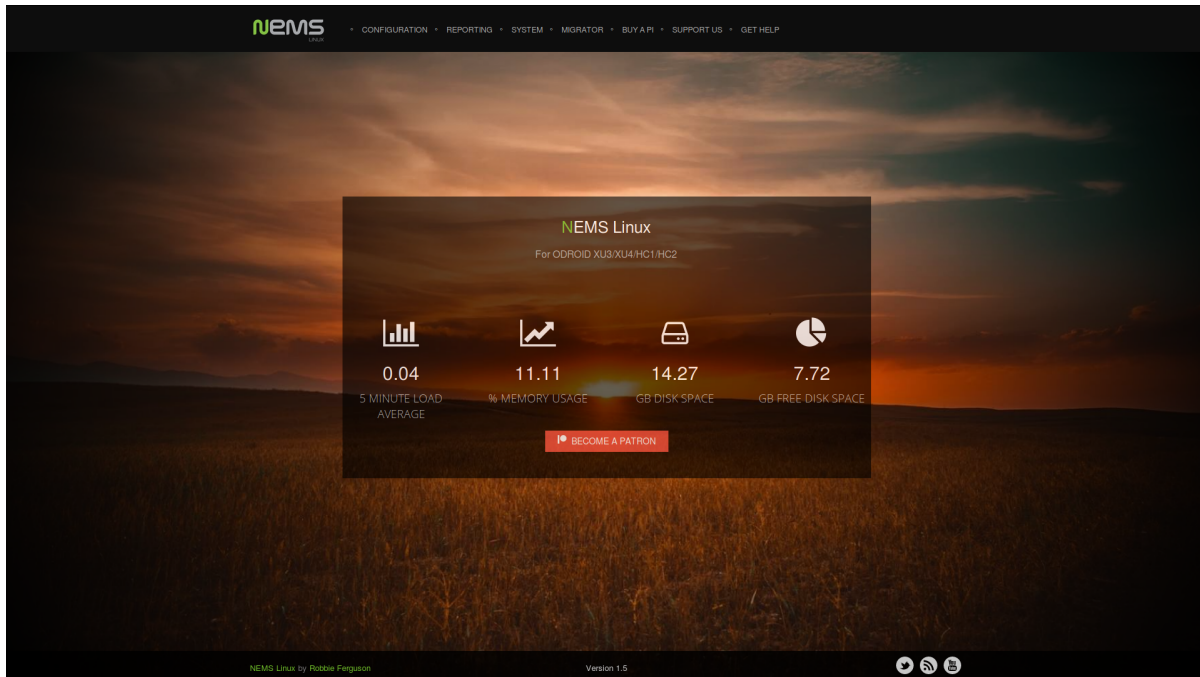


Fig. 6: NEMS Dashboard in NEMS Linux 1.5, with User-Uploaded Background

## 2.2.5 Configuring SMTP for NEMS Email Notifications

### Introduction

The first thing you'll want to do on your new NEMS Linux server is configure your SMTP settings. This will allow your NEMS server to email you if a problem is detected.

Access the NEMS System Settings Tool (SST) from the Configuration menu of your NEMS dashboard. This browser-based tool eliminates the need to use the traditional Nagios *resource.cfg* file to configure your email and other settings.

One of the nice things about NEMS Linux is that I really don't have to go into detail about how to do this. It is so intuitive that it does not require explanation. So I'll just provide a screenshot:

---

**Tip:** If you're using Gmail as your SMTP provider, be sure to review [the additional steps required](#).

---

**NEMS** UNL

• CONFIGURATION • REPORTING • SYSTEM • MIGRATOR • SUPPORT US • GET HELP

## NEMS System Settings Tool

General NEMS Migrator Backup NEMS Cloud Services **Notifications** TV Dashboard Hardware Optional Services

### SMTP Email Configuration [?](#)

SMTP Server Address  
For example: smtp.gmail.com

SMTP Server Port  
For example: 25

SMTP Secure Authentication  
Use TLS Secure Authentication

"From" Sender Email Address  
Email address

SMTP Authentication Username (Typically an email address)  
Username

SMTP Password  
Password

**Test SMTP Settings**

#### Telegram Account Info [?](#)

Bot API Token  
XXXXXXXX:YYYYYYYYYYYYYYYYYY

Telegram Chat ID  
gXXXXXXXX

#### Pushover Account Info [?](#)

API Key

User Key

#### Webhook Notifications

Send notifications to a Webhook.

Webhook URL

**Save All Settings (All Pages)**

Fig. 7: Configuring your SMTP server in NEMS is as simple as configuring a mail client.



## Test Email Notification Settings

---

**Note:** If you're using NEMS Linux 1.5.x or lower, the button does not exist in NEMS SST. Instead, open a terminal on your NEMS Server and type: `sudo nems-mailtest <recipient_email>`

---

NEMS Linux features an easy-to-use tool which will load your SMTP settings as entered in NEMS SST and send you a test email. It will also show you the results of the attempt, which may be helpful if you experience an error.

Once you've added your SMTP server settings in NEMS SST, simply press the *Test SMTP Settings* button.

## Save Your Settings

Once you are confident your SMTP settings are correctly entered, click **Save All Settings (All Pages)**.

## 2.2.6 Monitoring a Linux Computer on Your Network

Let's jump right into our first exercise as we learn to configure NEMS Linux to monitor a local Linux server's uptime. I will demonstrate how some of the features of NEMS Configurator (NConf) are interconnected, and will prepare you for adding IP-based hosts to your NEMS server.

A "Host" in NEMS Linux is any device you wish to monitor. This can be a computer, or a thermostat; it can be a router or a printer. The options are truly endless, and while NEMS Linux is free to download and use, there are no software-based limitations on how many hosts you can have set up. A single NEMS Server can easily handle hundreds of hosts.

### Adding a Host

Adding a host for monitoring within NEMS is done through the NEMS Configurator (NConf) user interface. You'll find this tool on the Configuration menu of your NEMS Dashboard. Within NConf, click the "Add" link next to "Hosts" on the left navigation. This will present you with the Add Host screen.

As illustrated, enter the hostname, a friendly alias for your own reference, and the IP Address of the host. As a side note, you'll want to make sure your hosts have static IP addresses so they don't change. Personally, I prefer to add DHCP reservations to my router rather than manually assigning the IP on the device. This keeps things simple and make it easier to ensure devices on my LAN always receive the same IP address, and that I don't accidentally assign the same IP to multiple devices.

Next, in the OS drop-down on the same screen, select your host's operating system. Note that if you don't see an appropriate type, you may also add operating systems under "Additional Items" in the left navigation menu. However, for our example we'll be adding our Linux server. "linux- server" is an out-of-the-box preset, so we will choose that.

### The linux-server Host Preset

A "Host Preset" allows you to add checks that are always used for this type of host. To help us understand what this is actually doing, let us digress for a moment and take a look under the hood. You can see what checks are going to be automatically applied via the selected Host Preset by pressing the "Show" link next to "Host Presets" on the left navigation menu.

Realizing that the linux-server Host Preset initiates the *check\_host\_alive* check command, we can review what that actually does by clicking the "Show" link next to "Misccommands."

It's running *check\_ping*—a Nagios check command to simply ping the IP address we provided in our host's configuration. The nice thing is, you didn't even have to script that (refer to this statement from the Getting Started Guide's

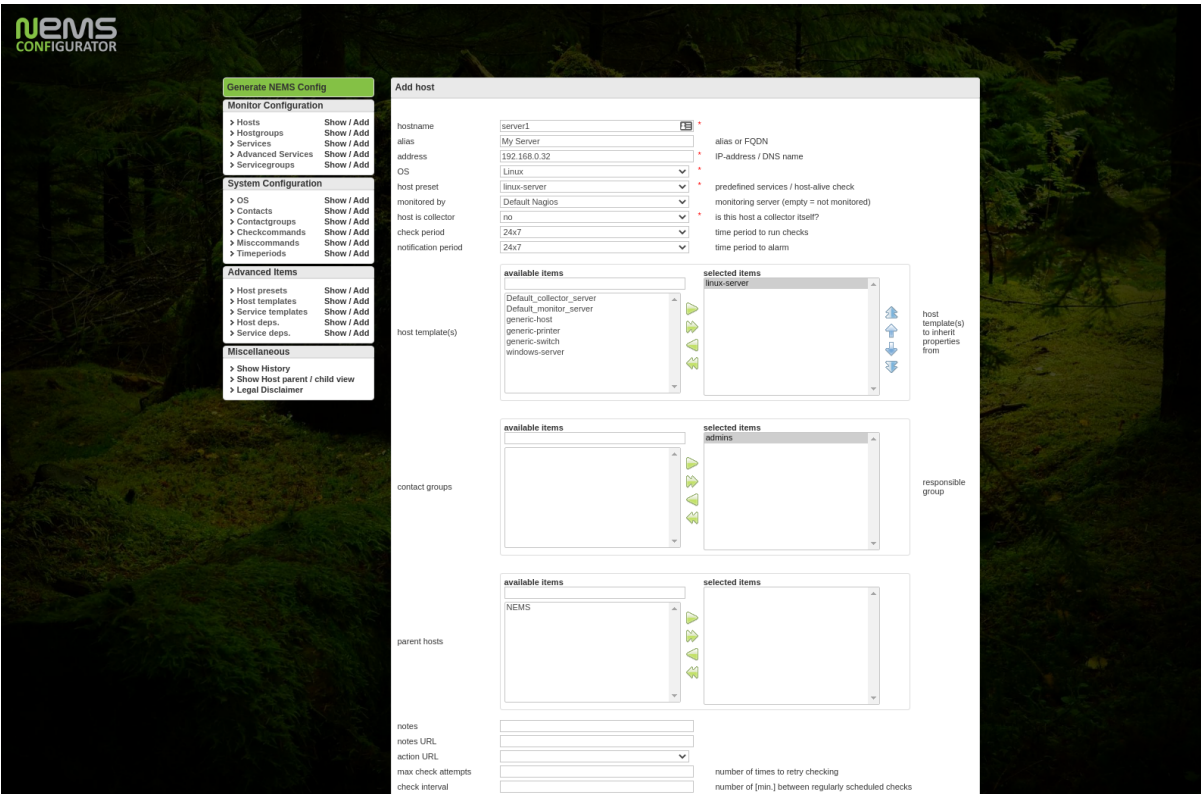


Fig. 8: Add a host to NEMS Linux using the NEMS Configurator

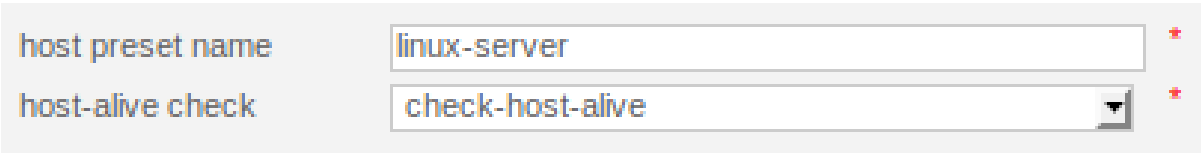


Fig. 9: linux-server Host Preset

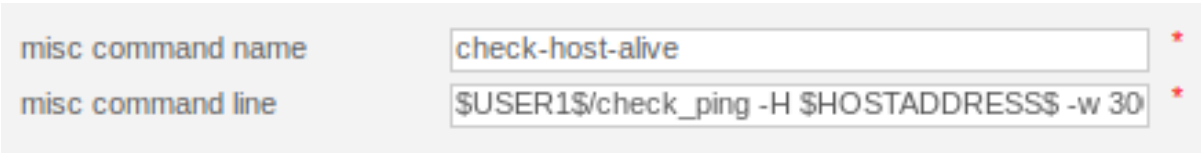


Fig. 10: check\_host\_alive check command details

**Introduction:** “[NEMS Linux] does away with the old Nagios scripting requirement”). I wanted to show you how it works, but as you’re just getting started with NEMS Linux, you will just select *linux-servers* and carry on, knowing that this will initiate a ping on that host (based on this example).

## Monitored By Default Nagios

Next, we need to change “Monitored By” to the only option available: *Default Nagios*. That is the preconfigured Nagios Core instance running on your NEMS Linux server. If you later wish to connect multiple NEMS Server together for a [mesh-style monitoring setup](#), this is where you’d select the NEMS Server to use. However, this is just the Getting Started guide. Let’s keep things simple and stick to the default.

## Host Templates are not Host Presets

A *Host Template* differs from a *Host Preset* in that it tells NEMS how we want our Host Preset to be performed: the monitoring schedule, the alert thresholds, and so-on. Based on the included *linux-server* Host Template, our *linux-server* Host Preset will check if the host is alive by pinging it every 10 minutes, and will send notifications during working hours if there is a problem. These defaults can always be changed by editing the Host Template. Of course, you can create your own presets and templates as you learn to use the system, though I recommend starting with the samples until you have a few hosts working.

### Quick Overview:

- A *Host Preset* tells NEMS Linux which check command to run on a particular group of hosts. This is usually something extremely simplistic, such as a ping to tell if it is up or down.
- A *Host Template* allows you to setup check commands, schedules and notifications for a group of hosts, saving you having to redundantly enter the same check commands for multiple hosts of the same type. By using a Host Template, you don’t have to fill in all this data on your Host Configuration. The Host Template will be applied.
- A Host Preset is required. A Host Template is optional.

## Setting Our Host Template

In the Host Templates section of our Add Host screen, we’ll highlight *linux-server* and press the right arrow icon to move it to the “Selected Items” list.

The only other item we must add to our host is who to contact if it is having problems. If we don’t specify this, no notifications will ever be received. By default, there is only one option: *Admins*. Highlight Admins and press the green arrow icon to move it to the Selected Items list.

Because we are using the Host Template, we do not need to specify our check or notification intervals: they are specified within the Host Template. If you were not using a Host Template, you’d need to specify those values here. Because we are using a Host Template which carries these values, we can just save the new host by pressing **Submit**.

On the next screen, you will be given the opportunity to add more service checks to this host, but for the sake of our example and because we are using a Host Preset and Host Template, we can skip this part.

---

**Tip:** In some cases, you may want your host checks to occur at different intervals than are specified within the Host Template. For example, you may wish your mission critical server to be pinged every minute rather than every 10 minutes. In these cases, rather than editing the Host Template (and thereby impacting all hosts which use that template) you can specify unique values on the Host Configuration screen, which will override the Host Template values only for this host.

---

## Generate NEMS Config: Make Your Changes Live

To make your changes live and begin monitoring your new host [Generate your NEMS Config](#).

If everything checks out, press **Deploy** and your new host will instantly be activated in Nagios.

### 2.2.7 Generate NEMS Config: Make Your Changes Live

In NEMS Configurator (NConf), you configure all your check commands and notification settings. But until you generate the config, the changes you make are not actually enabled on your running NEMS Server.

To make your changes live, press the *Generate NEMS Config* link on the left navigation. You should see 0 errors. If you do see errors, review where you went wrong, make the necessary changes, and try again. NConf is very good at showing you where the error is so you can go back and fix it rather quickly.

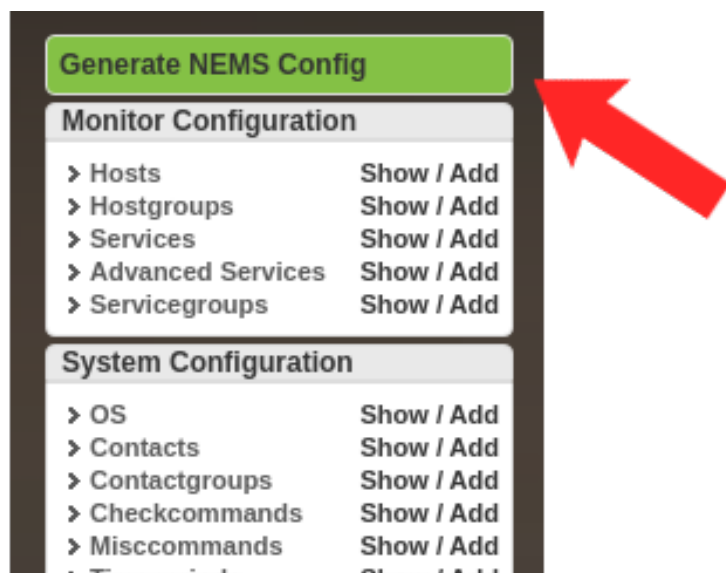


Fig. 11: Generate NEMS Config with the NEMS Configurator

If everything checks out, press *Deploy Config to NEMS Server* and your changes will instantly be activated with your NEMS Server's implementation of Nagios.

### 2.2.8 Monitoring Your Assets

#### Introducing Adagios

Now that we've configured our first host, let's see how to check its status. There are several ways to keep tabs on your assets with NEMS Linux. For the Nagios purists, Nagios Core is included on the Reporting menu. But we've included something much more modern: Adagios. This, too, is found on the Reporting menu. Adagios offers the same overall functionality of Nagios Core's front-end but replaces it with a modern, responsive bootstrap web interface.

To check the status of our hosts, simply click "Hosts" on the left navigation.

You'll see the host we [added previously](#) (server1) is showing with the status of UP. This means the ping replied. There is no Service Status, since we did not add any extra service monitors. To see an example of what is possible, expand the NEMS host (which is included on your NEMS Linux server) by clicking the triangle next to its name.

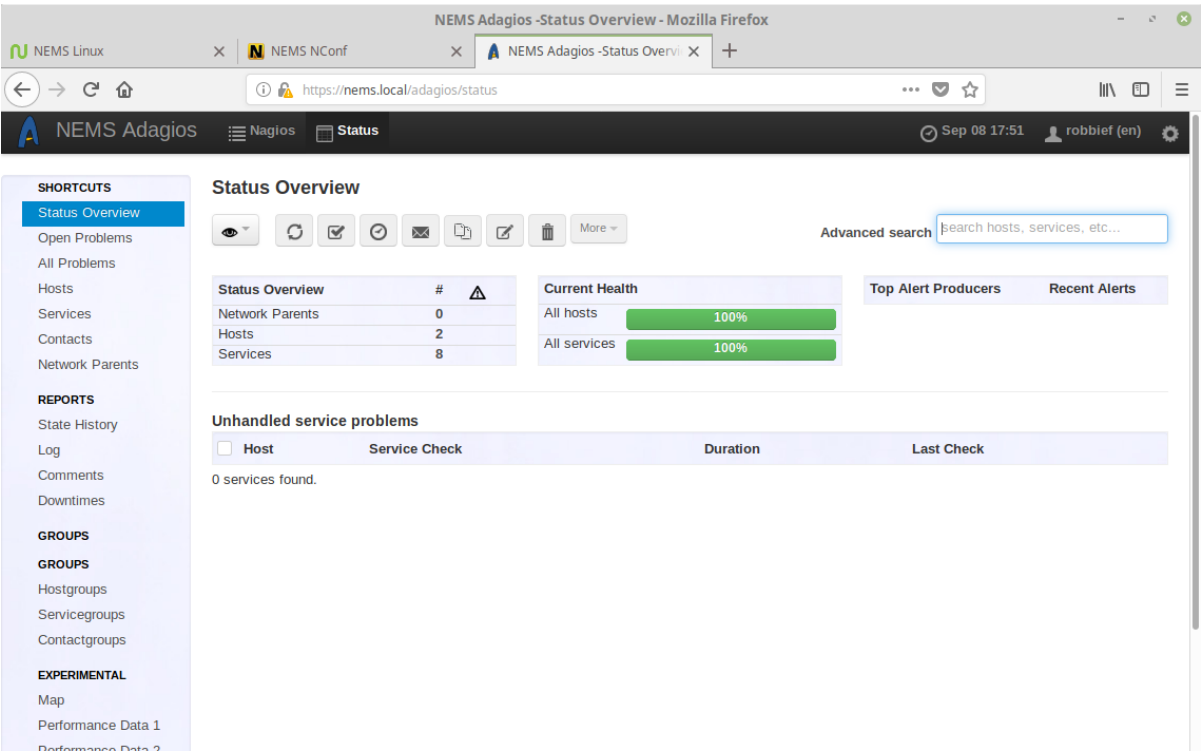


Fig. 12: Adagios responsive interface

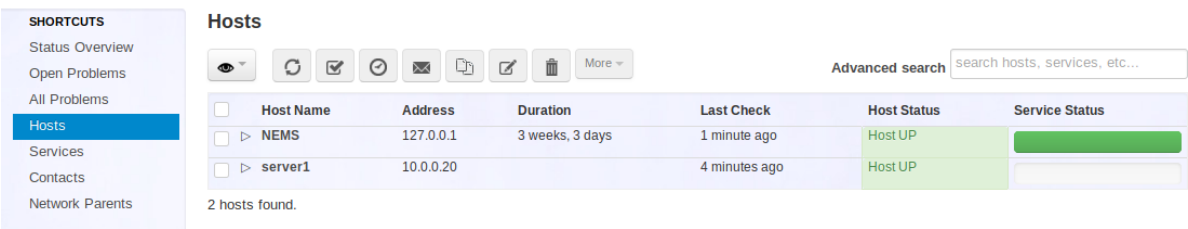


Fig. 13: Adagios hosts view.

## Hosts

More ▾

<input type="checkbox"/>	Host Name	Address	Duration
<input checked="" type="checkbox"/>	▼ NEMS	127.0.0.1	3 weeks, 3 days
<div> <div>Total Processes - PROCS OK: 80 processes with STATE = RSZDT</div> </div>			
<div> <div>Swap Usage - SWAP OK - 100% free (2047 MB out of 2047 MB)</div> </div>			
<div> <div>SSH - SSH OK - OpenSSH_7.4p1 Debian-10+deb9u3 (protocol 2.0)</div> </div>			
<div> <div>Root Partition - DISK OK - free space: / 8482 MB (58.06% inode=90%):</div> </div>			
<div> <div>PING - PING OK - Packet loss = 0%, RTA = 0.05 ms</div> </div>			
<div> <div>HTTP - HTTP OK: HTTP/1.1 200 OK - 14850 bytes in 1.104 second response time</div> </div>			
<div> <div>Current Users - USERS OK - 1 users currently logged in</div> </div>			
<div> <div>Current Load - OK - load average: 0.07, 0.08, 0.09</div> </div>			
<input type="checkbox"/>	▶ server1	10.0.0.20	

Fig. 14: Expanded view of Host reveals configured service checks.

## Problem Acknowledgement

In the case of a problem, you can open the host or service experiencing the problem to see more information, and even acknowledge the issue so you don't continue receiving notifications.

## Other Ways to Monitor Your Assets

I would also like to encourage you to test [NEMS Mobile UI](#), [NEMS TV Dashboard](#) and [NEMS Tactical Overview](#), all of which are also found on the Reporting menu of the NEMS Dashboard. The first is meant to offer you a complete mobile interface for monitoring your assets, and the latter two allow you to set up a TV display in your server room that shows a real-time tactical overview of your NEMS host and service checks.

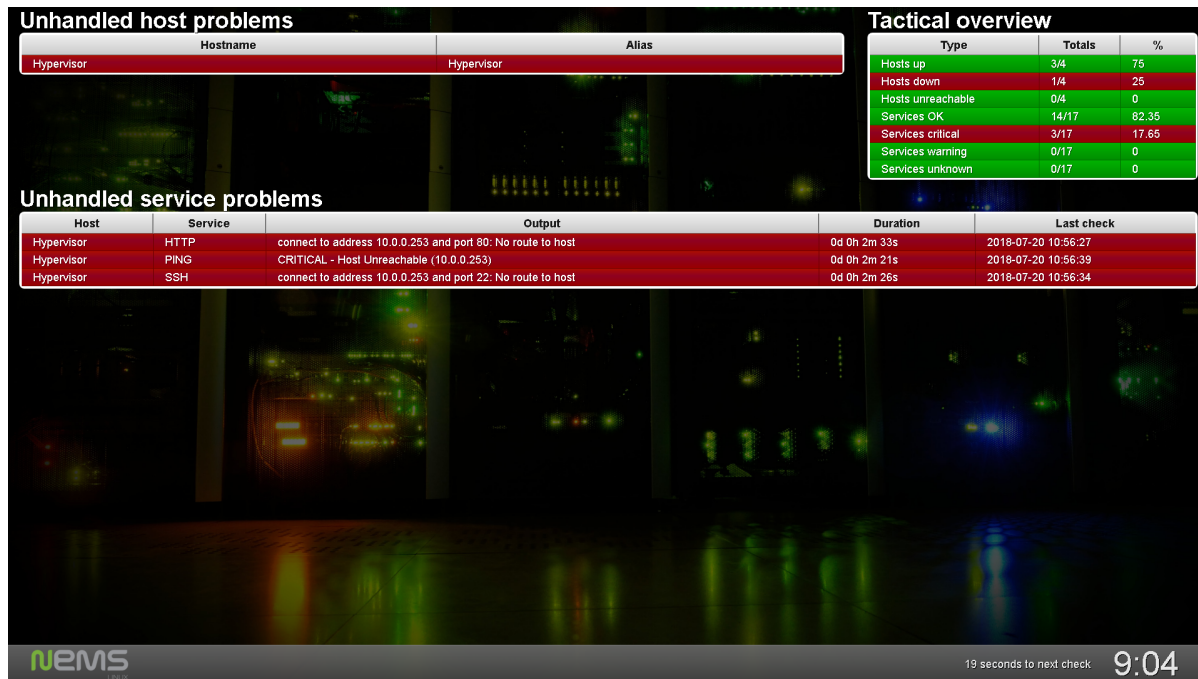


Fig. 15: NEMS TV Dashboard circa NEMS Linux 1.4.1.

## 2.2.9 Understanding Notification Definitions

Before you setup your first check commands, you'll need to understand what the single-character notification options mean. Failure to correctly set notification settings could result in no notifications being generated, or an excess of notifications being sent.

Refer back to this list as needed, as you'll need to understand what commands such as *w,u,c,r,f* mean.

### Host Notification Settings

When you see *d,u,r,f,s,n*, these are the definitions:

- **d** Notify if host is down,
- **u** Notify if host is unreachable (E.G. Internet down),
- **r** Notify upon recovery,
- **f** Notify if the host is flapping (up/down/up/down),
- **s** Notify if a scheduled service downtime begins or ends,
- **n** Never notify.

---

#### Exercise

The NEMS Server sample host uses the *linux-server* Host Template. So rather than having to set the notification options within the host itself, we set these within the template. Can you determine from the above list what the default notification settings are for *linux-server* devices by looking at the host template? You'll find the *linux-server* Host Template in Configuration -> NEMS Configurator. Press the *Show* button next to *Host templates* on the menu, followed by the edit pencil next to *linux-server*.

---

### Service Notification Settings

When you see *w,u,c,r,f,n*, these are the definitions:

- **w** Notify if in warning state,
- **u** Notify if in unknown state,
- **c** Notify if in critical state,
- **r** Notify if recovered from a previously bad state,
- **f** Notify if the service is flapping (up/down/up/down),
- **n** Never notify.



## 2.2.10 Beginner Exercises

Having made it through the Getting Started Guide, it's time to put your newly learned skills to the test. The following exercises will not only help you to review what you've learned thus far, but will also expand on your knowledge of how to use NEMS Linux by providing practical use cases, complete with step-by-step instruction.

### Monitor Your Web Site with `check_http`

#### Exercise Introduction

Your web site is the face of your business. If it ever goes down for any reason, or becomes sluggish, it's important to be proactive in remedying the situation. What's worse than having a customer contact you to let you know your web site is down? Realizing it might have been down for a week and the customers during that time didn't let you know: They just went elsewhere.

Having your web site become sluggish or unresponsive can also damage your organic SEO standings.

In this exercise, we'll learn how to setup NEMS Linux to notify you if your web site fails to respond for more than 10 minutes. With this skill, you will be able to proactively monitor your own, your customers' or any http/https web site for uptime or slow response time.

If your site goes offline, or becomes unresponsive or sluggish, NEMS Linux can send you an alert by a number of methods including email, Telegram or Pushover. This capability makes NEMS Linux a fantastic tool for web designers and hosts who want to ensure their customer sites are always up so the customer doesn't notice any downtime. If your site is hosted over SSL, NEMS can even notify you if your certificate has expired – or is about to expire. There are so many options since NEMS Linux has been built to *monitor everything*.

For this exercise, we'll use the built-in `check_http` command. For my example, I'll use `https://nemslinux.com/` – I would suggest you do the same for the sake of the lesson, and then try changing the Host to your own domain once you understand how everything is connected.

It may appear onerous as you glance over the following steps, but keep in mind once you create your config, you can reuse it for as many web site hosts as you like by simply assigning your host to the `web_site_ssl` host group, which you'll learn to create below.

Configuration is done in NEMS Configurator (NConf). Open that tool via the Configuration menu on NEMS Dashboard.

#### Steps

1. The `check-host-alive` command, found in `misccommands` in NConf, is used to check hosts to determine if they are up or down. My web site, `nemslinux.com`, will only respond on IPv4. However, the default `check-host-alive` command will attempt to use IPv6. Rather than editing the sample command, let's add a new one based upon it, but this one will only use IPv4. That way, we can still use the old command when we need IPv6 for a different host.
  1. Show the `misccommands` list.
  2. Edit `check-host-alive`
  3. Highlight and copy the entire command line to your clipboard.
  4. Click Add next to `misccommands` to add a new command.
  5. Name your new command `check-host-alive-ipv4`
  6. Paste the command line from your clipboard.

7. At the very end of the command line, simply add a space, followed by -4 to tell it to use IPv4 for this check.
8. Save the new command.

**Modify misccommand**

misc command name  \*

misc command line  \*

Fig. 16: Create New misccommand to check-host-alive Using IPv4

2. Our commands are ready for us, so now it's time to setup our Host Preset. We want to create one for IPv4 Web Sites. That way, we can reuse the preset for every IPv4-based web site we want to monitor with NEMS Linux.
  1. Add a new host preset.
  2. Name your preset Web Site IPv4
  3. Set the host alive check to the new command you created in Step 2: *check-host-alive-ipv4*
  4. Save your host preset.

**Modify host-preset**

host preset name  \*

host-alive check  \*

host preset commands

available items

- check\_ad
- check\_dhcp
- check\_exchange
- check\_ftp
- check\_hpjd
- check\_http
- check\_iis
- check\_imap
- check\_local\_disk

selected items

auto-create services for a new host based on these checkcommands

Fig. 17: New Host Preset for IPv4 Web Sites

3. So far, everything we've done can be reused for any web site whose hostname resolves to an IPv4 address. From here forward however, we'll be setting up our host group specifically for a secure (SSL) web site.
  1. Add a new hostgroup.
  2. Call this `web_site_ssl`
  3. For the alias, enter `Web Site (SSL)`

4. Leave everything else as is and save your new hostgroup.

Fig. 18: New hostgroup for web\_site\_ssl

4. Why would we create a new hostgroup if it has no settings beyond a name? Well, this is where the magic happens. We now have a check command, a check host alive command, a host preset and a hostgroup. Now, we can link them all together, starting with an Advanced Service. Remember, the idea here is that everything we do can be assigned to as many hosts as we like. No having to redo all this for the next web site.
  1. Click Add next to Advanced Services.
  2. Name your service: *Web Site (SSL)*
  3. Give it a description: *Uptime of SSL Web Site*
  4. Set the check command to *check\_http*
  5. Set the check period and notification period to *24x7*
  6. In *assign advanced-service to hostgroup*, highlight the hostgroup we created (*web\_site\_ssl*) and press the green arrow to add it to the selected items list.
  7. Under *contact groups* be sure to add *admins* as well. Otherwise, you won't receive notifications.
  8. Set your notifications as follows:
    - max check attempts: 10
    - check interval: 1
    - retry interval: 5
    - first notification delay: 10
    - notification interval: 30
    - notification options: w,u,c,r,f
  9. Finally, set your service parameters to: *-S -\sni*
  10. Save your advanced service.

---

**Tip:** The *-S* tells *check\_http* that this site is using SSL, and the *-\sni* enables SNI (Server Name Indication) since I use CloudFlare for SSL on nemslinux.com, and therefore my resolving IP address is associated with more than one domain name. For your site, if you have any trouble, try removing SNI by simply omitting *-\sni*. For the full documentation surrounding the *check\_http* command, visit [the check\\_http documentation](#).

---

5. Finally, let's add our web site host. From now on, this is the only step you have to take to add more sites to your NEMS Linux server.

**Modify advanced-service**

advanced service name:  \*

service description:  \*

check command:  \*

check period:  time period to run checks

notification period:  time period to alarm

assign advanced-service to host

available items:

- https://nemslinux.com
- Hypervisor
- Minetest Game Server
- NEMS
- Percy
- Shiraz
- wirecast
- Zimbra

selected items:

assign advanced-service to hostgroup

available items:

- linux-servers
- network-printers
- primary\_windows
- secondary\_windows
- switches
- windows-servers

selected items:

web\_site\_ssl

contact groups

available items:

selected items:

admins

responsible group

notes

notes URL

action URL

max check attempts:

check interval:  number of [min.] between regularly scheduled checks

retry interval:  number of [min.] to wait before scheduling a re-check

first notification delay:  number of [min.] to wait before sending the first notification

notification interval:  number of [min.] to wait before re-notifying a contact

notification options:  possible values: w,u,c,r,f,s,[n]

active checking:

passive checking:

notification enabled:

check freshness:

freshness threshold:

params for check command

service parameters

ARG1:

Submit Reset

Fig. 19: Creating an Advanced Service to Check SSL Web Sites

1. Add a new host.
2. Set the following:
  - hostname: `https://nemslinux.com`
  - alias: *NEMS Web Site*
  - address: *nemslinux.com*
  - OS: *Linux*
  - host preset: *Web Site IPv4* (See what we did there?)
  - monitored by: *Default Nagios*
  - host is collector: *no*
  - check period: *24x7*
  - notification period: *24x7*
  - max check attempts: *10*
  - check interval: *1*
  - retry interval: *5*
  - first notification delay: *10*
  - notification interval: *30*
  - notification options: *d,u,r,f*
  - assign host to hostgroup (are you ready for this?): *web\_site\_ssl*

Creating a Host to Monitor IPv4 SSL Web Site

3. Save the host.
6. Generate your config.

## Conclusion

If you followed the steps correctly and my web site is up, Adagios should report all is well.

To test what would happen if it were to start failing, change the hostname in the Host to `nemslinux.com1` (which obviously will not respond), and then generate your config again.

Once you feel ready, change the Host to your own web site. If your site is SSL, you should only need to change the hostname, alias and address of the host. If it's not SSL, repeat Step 3 above, but this time create a new hostgroup called `web_site_no_ssl`, and then repeat Step 4, but creating a new Advanced Service called Web Site (Non-SSL), assign it (4.5) to Web Site (Non-SSL) and leave off the SSL parameters in 4.8.

Modify host

hostname

https://nemslinux.com

\*

alias

NEMS Web Site

address

nemslinux.com

\*

OS

Linux

\*

host preset

Web Site IPv4

\*

monitored by

Default Nagios

\*

host is collector

no

\*

check period

24x7

\*

notification period

24x7

\*

alias or FQDN

IP-address / DNS name

predefined services / host-alive check

monitoring server (empty = not monitored)

is this host a collector itself?

time period to run checks

time period to alarm

max check attempts

10

number of times to retry checking

check interval

1

number of [min.] between regularly scheduled checks

retry interval

5

number of [min.] to wait before scheduling a re-check

first notification delay

10

number of [min.] to wait before sending the first notification

notification interval

30

number of [min.] to wait before re-notifying a contact

notification options

d,u,r,f

possible values: d,u,r,f,s,[n]

active checking

do active checking of hosts

passive checking

do passive checking of hosts

notification enabled

send notifications for hosts

check freshness

check age of last check results

freshness threshold

age threshold in [sec.]

assign host to hostgroup

available items

linux-servers

network-printers

primary\_windows

secondary\_windows

switches

windows-servers

selected items

web\_site\_ssl

Submit

Reset

https://nemslinux.com

nemslinux.com

21 hours, 38 minutes

0 minutes ago

Host UP

Check Uptime of SSL Web Site - HTTP OK: HTTP/1.1 200 OK - 22203 bytes in 0.264 second response time

Fig. 20: NEMS Adagios Shows nemslinux.com is UP

## Monitor A Non-Standard Port with `check_tcp`

### Exercise Introduction

In this exercise you will learn to monitor the state of any TCP/UDP port on a network connected device, and notify yourself if it stops responding. Some quick examples are to tell if your local blockchain node has stopped responding on port 8333, Apache2 has stopped responding on port 443, or even monitor the state of openssh-server running on port 22. The options are limitless.

NEMS Linux includes a dummy port listener running on port 9590. The port listener is cleverly called `9590`, and does nothing other than reply that it is up. This can be used to simulate a port on another device. While you can create a check for any port on any host, let's start with using this dummy port just to become familiar with how NEMS Linux works.

**Our goal:** To setup a service monitor on the NEMS host which warns us if port 9590 ever goes offline.

### Steps

1. On the left menu of NConf, you'll see "Services". Click "Add".
2. Set the Service Name to: `9590`
3. Leave Service Enabled set to: *Yes*
4. Set the Check Command to: `check_tcp`
5. Set Assigned to Host to: *NEMS* (this host comes pre-installed)
6. Leave Check Period set to: `24x7`
7. Set Notification Period to: `24x7`
8. Leave Service Templates as is, none selected.
9. Under Contact Groups highlight the 'admins' group and press the arrow pointed right to move it to Selected Items.
10. Leave Notes, Notes URL and Action URL blank.
11. Set Max Check Attempts to: `30`
12. Set Check Interval to: `1`
13. Set Retry Interval to: `1`
14. Set First Notification Delay to: `5`
15. Set Notification Interval to: `15`
16. Set Notification Options to: `w,u,c,r,f,s`
17. Leave Active Checking, Passive Checking, Notification Enabled, Check Freshness and Freshness Threshold blank.
18. Leave Assign Service to servicegroup as is, none selected.
19. Set Params for check command to the port number: `9590`
20. Press **Submit**
21. Generate your config.

### Testing Your Check Command

Once the new config is running, try failing the service by opening *Monit Service Manager* under *System* on the NEMS Dashboard. Click on the Process named 9590, and then click “Stop service”. You’ll notice in around 1 minute the status of 9590 will show as a problem in all status views (E.G., NEMS TV Dashboard, NEMS Adagios, Nagios Core), and after roughly 5 minutes you will receive a notification (assuming your notifications settings are configured).

Once you have received a notification, visit NEMS Adagios to Acknowledge the outage, and observe how that impacts your host and service state.

Then, return to Monit, open the 9590 Process, and click “Enable Monitoring”. This will re-load 9590 and you’ll soon see it change to a “Recovered” state.

### Real World Use

Once complete, try setting up a new service to monitor a real host on your network. Simply change the name of the service to something appropriate, the host in step 5 (you already know how to add new hosts if you don’t already have it configured), and the port number in step 19.

## 2.3 Conclusion

Having gone through each page of the Getting Started Guide, you now have a good foundation and are ready to use NEMS Linux in a production environment.

## 2.4 What’s Next

- [Become a Patron](#) for additional perks and to support this project
- [Join the Discord](#) to chat with other NEMS Linux users
- Keep reading the documentation below to learn more about how this feature rich appliance works
- [Visit the Community Forum](#) to bounce ideas around
- Tell everyone you know about NEMS Linux, blog about it, screencast on your YouTube channel, talk about it on your podcast. Spread the word.



## 3.1 NEMS Administrator User

### 3.1.1 Default Username and Password

The default username on a fresh installation of NEMS Linux is `nemsadmin`. This is used to sign-in to your NEMS Server (over SSH or keyboard connected to your NEMS Server) only until you run `nems-init`.

**Username:** `nemsadmin`

**Password:** `nemsadmin`

### 3.1.2 Post-Initialization

Once your `nems-init` process is complete, you will need to instead sign-in as your newly created account (which you specify during `nems-init`).

When you initialize NEMS, you will provide a password for the NEMS web interfaces. This username/password will be what you use to access NEMS features (eg., `nCONF`, Nagios Core, NagVis, Check\_MK, Samba, Webmin, SSH, etc.).

**Warning:** Do not use the traditional `passwd` command to change your NEMS administrator's password. Doing so will leave your system in a broken state (which can be fixed by running the correct command). Instead, use `sudo nems-passwd`

### 3.1.3 Multiple User Environments

At present, NEMS Linux is single-tenant. This means that if you wish to give one of your staff access to your NEMS Server, you would need to give them your NEMS Administrator credentials. This will change as NEMS Linux evolves. Please consider supporting the project on [Patreon](#) to help fund the development of NEMS Linux.

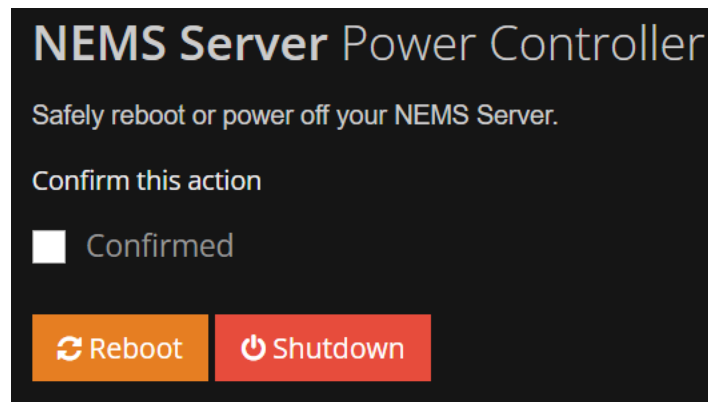
### 3.1.4 Browser Reports Compromized Password

The default `nemsadmin` password is a generic, single-use password that allows you to initialize your NEMS Administrator account / set your own password out-of-the-box. It is not left enabled on your NEMS Server, nor is your NEMS Server ever available to the outside world (unless you specifically do so, which is `_not_` recommended). Use best practices, and never setup your NEMS Server to be publicly accessible.

## 3.2 Reboot or Power Off NEMS Server

### 3.2.1 NEMS Power Controller

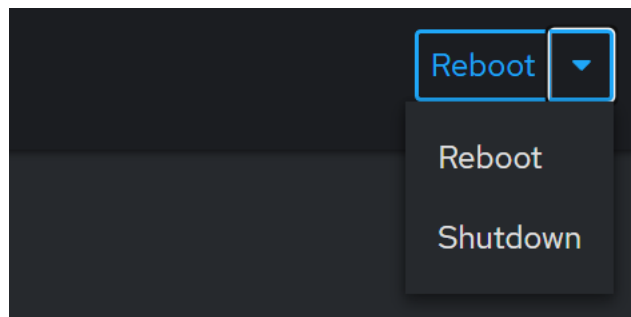
NEMS Linux (since version 1.7) includes NEMS Power Controller. From within NEMS Power Controller, you can safely reboot or power off your NEMS Server. This is the preferred method on modern NEMS Servers since it waits for NEMS tasks (such as updates) to complete before shutting down.



You'll find NEMS Power Controller on the "System" menu.

### 3.2.2 Cockpit

On legacy NEMS Servers, login to Cockpit with Administrator access enabled and look for the power options in the top right.



You'll find Cockpit on the "System" menu.

## 3.3 Included Check Commands

While not an exhaustive list of all available check commands in NEMS Linux, here are some of the available check commands and their corresponding documentation:

- `check_1wire_temp` - Monitor temperature using a 1-wire device such as the DS18S20.
- `check_apc` - Check APC UPS.
- `check_dhtxx` - Use a DHTxx (DHT11 / DHT22 / AM2302) Arduino sensor to report on the room temperature and humidity.
- `check_esxi_hardware` - Monitor the hardware of ESX and ESXi servers.
- `check_fortigate` - Check Fortinet FortiGate appliances.
- `check_http` - Check the status of an HTTP/HTTPS server on a remote host.
- `check_ilo2_health` - Check hardware health of HP Proliant Servers by querying the HPE Integrated Lights-Out (iLO) 2/3/4/5 Management Controller.
- `check_internet_speed` - Check the speed of your internet connection.
- `check_iperf` / `check_iperf3` - Monitor the speed between network links using the *iperf3* command.
- `check_mikrotik_switch` - Monitor stats for some MikroTik routers, including thermal sensors, packet loss, uptime, and so-on.
- `check_minecraft` - Monitor Minecraft server state.
- `check_mssql_mem` - Monitor MS SQL Server memory usage.
- `check_ncpa` - Monitor Windows, Mac and Linux hosts. NCPA is written in Python and is able to run on almost any Operating System.
- `check_nems_osb` - Determine whether NEMS Migrator Offsite Backup was successful.
- `check_nems_php_agent` - Monitor a Linux web server that has PHP using your custom NEMS PHP Agent.
- `check_netscaler` - Monitor your Netscaler ADC device or cluster.
- `check_nrpe` - Monitor your hosts at a deeper level. Things like CPU usage, free disk space, free RAM, and so-on.
- `check_ping` - Ping by hostname or IP address with warn/crit thresholds for response time and packet loss.
- `check_pve` - Monitor Proxmox Virtual Environment nodes.
- `check_qnap` - Monitor various sensors on a QNAP NAS.
- `check_synology` - Monitor various sensors on a Synology NAS.
- `check_temper` - Use a TEMPer USB temperature sensor to detect and report the room temperature.
- `check_sbc_temperature` - Check your NEMS SBC temperature with `perfd` and warn/crit thresholds.
- `check_tcp` - Check response of a specific TCP connection.
- `check_win_users` - Check the count of users on a Windows server based on a query.
- `check_win_*` (`check_wmi_plus`) - Check resources such as disk or CPU usage on Windows machines using Windows Management Instrumentation (WMI). (See [this](#) and [that](#))
- `custom_check_mem` - Monitor the percentage of RAM free on either the local NEMS server or a remote system via NRPE.
- `check_tasmota` - SONOFF / Tasmota IoT device monitoring.
- `check_truepool` - Check the status of your Chia farm on the Truepool.io pool.

- `check_ibmi*` - IBM i monitoring using Nagios i.

### 3.3.1 Check Command: `check_1wire_temp`

1-wire devices are a cheap and simple way to monitor temperature. Each 1-wire device has a unique serial number that allows multiple devices to communicate on one bus.

`check_1wire_temp` is a Nagios plugin that is used to monitor temperature using a 1-wire device such as the DS18S20. `check_1wire_temp` is included in NEMS NConf and can be added to any compatible 1-wire temperature device connected to your NEMS Linux server.

#### CLI Example

```
check_1wire_temp -v -h -l low_temp_warn -L low_temp_crit -w high_temp_warn -H high_temp_crit
```

#### Requirements

Requires NEMS Linux 1.6+.

#### Source

From <https://exchange.nagios.org/directory/Plugins/Hardware/Environmental/1-2DWire-Temperature-Check/details>

### 3.3.2 Check Command: `check_apc`

APC UPS's use SNMPv1. You can configure SNMP for your UPS from its web interface or within your UPS's configuration software client.

`check_apc` provides performance data and range monitoring for a wide series of data and a large number of APC UPS's, including Symmetra and SmartUPS models.

This check command requires NEMS Linux 1.6+.

#### `check_apc` Available Command Arguments

- `id` - Return the UPS model name (e.g. 'APC Smart-UPS 600') and interlan info about Firmware, CPU S/N and manufacturing date
- `bat_status` - Return the status of the UPS batteries
- `bat_capacity` - Return the remaining battery capacity expressed in percent of full capacity
- `bat_temp` - Return the current internal UPS temperature expressed in Celsius
- `bat_run_remaining` - Return the UPS battery run time remaining before battery exhaustion  
**Note:** thresholds must be expressed in minutes
- `bat_replace` - Return whether the UPS batteries need replacing
- `bat_num_batt` - Return the number of external battery packs connected to the UPS
- `bat_num_bad_batt` - Return the number of external battery packs connected to the UPS that are defective
- `bat_act_volt` - Return the actual battery bus voltage in Volts

**Note:** thresholds must be expressed in range as nearest values. ex:

normal=220, warning=215:225, critical=210:230

Additionally, the checks will look for Nominal Voltage (as returned by the UPS), and exit as CRITICAL if Actual Voltage is LOWER or Equal

- **power\_modules** - Return the status of the Power Modules
- **in\_phase** - Return the current AC input phase
- **in\_volt** - Return the current utility line voltage in VAC  
**Note:** thresholds must be expressed in range as nearest values. ex:  
normal=220, warning=215:225, critical=210:230
- **in\_freq** - Return the current input frequency to the UPS system in Hz  
**Note:** thresholds must be expressed in range as nearest values. ex:  
normal=50, warning=45:55, critical=40:60
- **out\_status** - Return the current state of the UPS
- **out\_phase** - Return the current output phase
- **out\_volt** - Return the output voltage of the UPS system in VAC  
**Note:** thresholds must be expressed in range as nearest values. ex:  
normal=220, warning=215:225, critical=210:230
- **out\_freq** - Return the current output frequency of the UPS system in Hz  
**Note:** thresholds must be expressed in range as nearest values. ex:  
normal=50, warning=45:55, critical=40:60
- **out\_load** - Return the current UPS load expressed in percent of rated capacity
- **out\_current** - Return the current in amperes drawn by the load on the UPS
- **comm\_status** - Return the status of agent's communication with UPS.

If no command is supplied, the script returns OKAY with the UPS model information.

### 3.3.3 DHT Sensors

#### Introduction

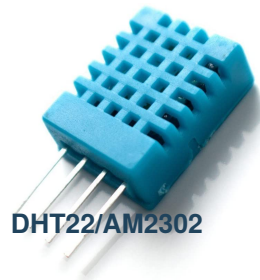
DHT sensors are an ultra-affordable Arduino sensor, compatible with various single board computers such as the Raspberry Pi. DHT sensors contain a capacitive humidity sensor and a thermistor, plus a basic chip to convert the analog data to digital and provide a digital signal with the temperature and humidity to any compatible NEMS Linux server, such as Raspberry Pi, ODROID or Pine64 NEMS Servers.

DHTxx support requires NEMS Linux 1.6+.

## Compatible DHT Sensors

### DHT11

[Buy on Amazon.com](#)



The DHT11 sensor is reasonably accurate, but has a fairly narrow range, making it only suitable for indoor use where it is known the room temperature and humidity will not fall outside the available thresholds.

- 20-80% humidity readings with 5% accuracy
- 0-50°C temperature readings  $\pm 2^\circ\text{C}$  accuracy

[Buy on Amazon.com](#)

The DHT22 (or AM2302 for the wired version) is more accurate, and has a significantly wider range, making it suitable for measuring more extreme temperatures and humidity. This can be useful if you wish to use NEMS Linux to alert you should your pipes be at risk of freezing, or your freezer compressor failing.

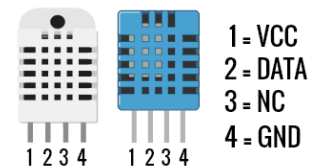
- 0-100% humidity readings with 2-5% accuracy
- -40 to 80°C temperature readings  $\pm 0.5^\circ\text{C}$  accuracy

### Connection



The pinout for the DHT11, DHT22 and AM2302 are the same, so no matter which DHT sensor you choose, it is an identical process to making it work with your NEMS Server.

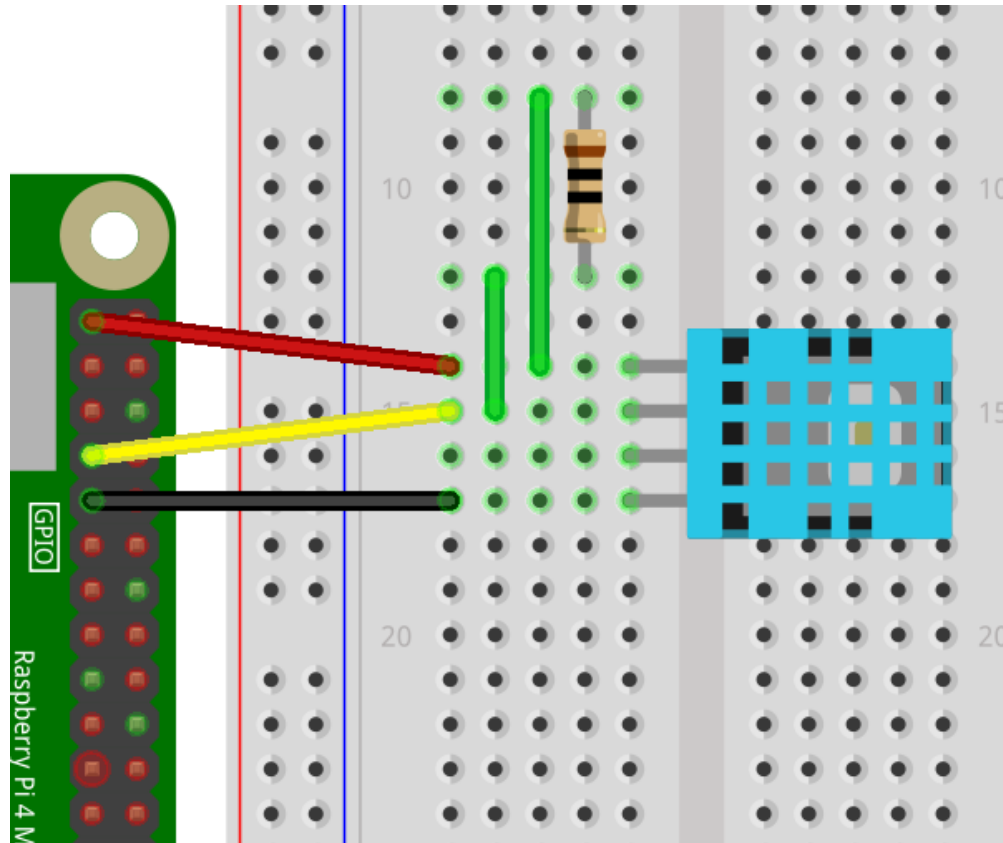
Place a 10k resistor between Pin 1 and Pin 2, which will ensure reliable data output from the sensor when it switches from input to output. **If you don't have a resistor handy, don't worry about it.** It'll improve reliability to have it, but won't hurt anything if you don't.



## Raspberry Pi

### DHT11/DHT22/AM2302

- Raspberry Pi Pin 1 (3.3V) to DHTxx Pin 1.
- Raspberry Pi Pin 7 (GPIO4, GPCLK0, T-Cobbler P04) to DHTxx Pin 2.
- DHTxx Pin 3 does not get connected.
- Raspberry Pi GND to DHTxx Pin 4.
- 10k resistor from Pin 1 to Pin 2 of the DHTxx sensor.



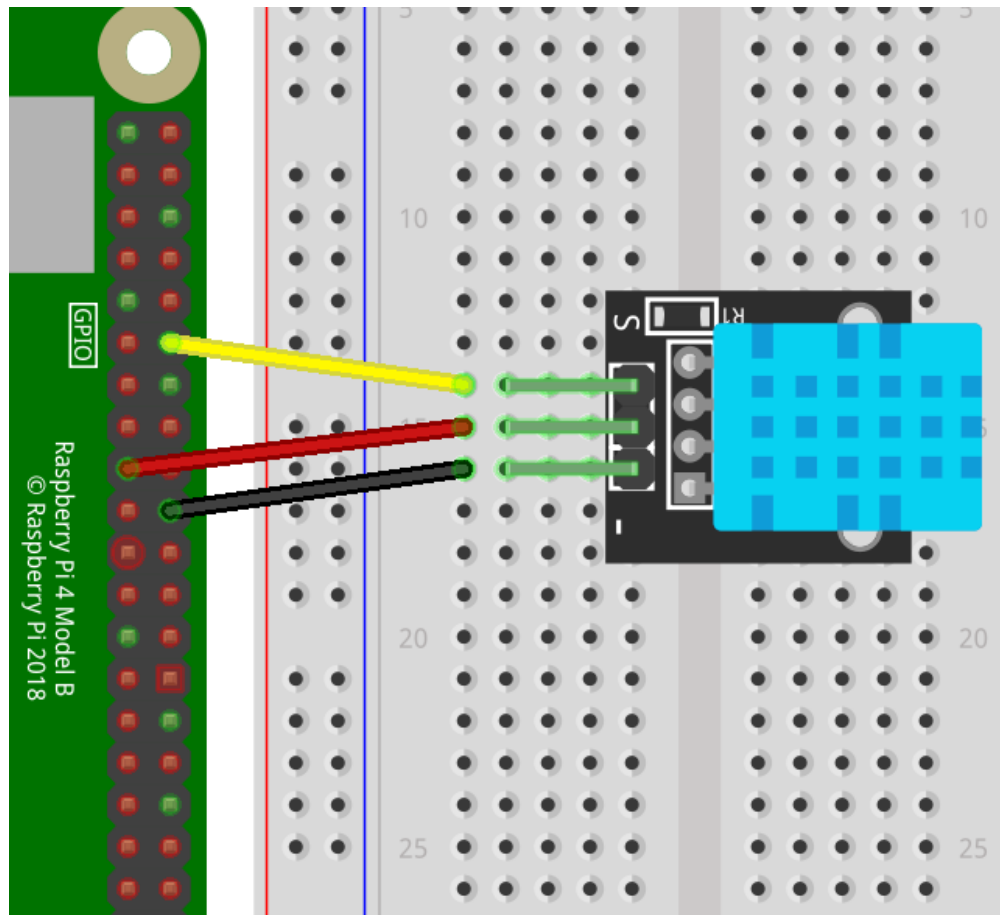
### KY-015 (DHT11 Sensor on 3-Pin PCB)

#### Check your pin labels first.

If you see **S** on one side and **-** on the other, your board is a KY-015. If, your board shows **+ OUT -**, it is not a KY-015, but is a DHT11 PCB. If that's the case, wiring this way may damage your sensor, so please make sure you are indeed using a KY-015.

- GPIO Pin 18 to Sensor Pin S.
- GPIO 3.3V to Sensor Middle Pin.
- GPIO GND to Sensor Pin -.

Since the KY-015 contains a built-in 1k resistor, adding one to your circuit is not necessary.





## ODROID-XU4

Note that the ODROID-XU4 provides 1.8V or 5V. As there is no 3.3V and the minimum voltage for this sensor is 3V, you will need to use the 5V pin.

### CLI Usage

*./dhtxx [VERSION] [GPIO PIN]*

#### Valid Versions:

- 11 - DHT11
- 22 - DHT22/AM2302

#### Valid Pins

- 4 - Older sensors, usually without PCB, Default
- 18 - Newer sensors (expecially KY-015), often with PCB

### JSON Output

NEMS Linux can output JSON temperature and humidity data from the DHT11 and DHT22/AM2302 device.

```
root@nems: /# /usr/local/share/nems/nems-scripts/dhtxx 11 18
{"dht": "11", "c": 28, "f": 82.4, "h": 43}
```

### Check Commands

NEMS Linux includes Robbie Ferguson's *check\_dhtxx* to monitor the temperature and humidity data provided by a DHT Sensor. You can specify both minimum and maximum values for the check command.

*check\_dhtxx* automatically detects the version of DHTxx sensor you are using.

Both the temperature and humidity sensors are supported, and check commands are included in NEMS NConf.

- *check\_dhtxx\_temp*
- *check\_dhtxx\_hum*

Both a low and high threshold is set in NEMS NConf, allowing the DHT sensor to enter a warning or critical state whether the temperature/humidity are either too high or too low.

### Calibration

As of NEMS Linux 1.6, both the thermal sensor and humidity sensor can be calibrated within NEMS SST to ensure the highest level of accuracy.

### 3.3.4 Check Command: *check\_esxi\_hardware*

*check\_esxi\_hardware* queries the CIM (Common Information Model) server running on the ESXi server to retrieve the current status of all discovered hardware. This check command comes from [claudiokuenzler.com](http://claudiokuenzler.com).

### Frequently Asked Questions

**Q:** The plugin is executed fine but then it hangs on a certain CIM element on a DELL server. Verbose output:

```
18:21:56 Connection to \https://esxi-001
18:21:56 Check classe OMC_SMASHFirmwareIdentity
18:21:57 Element Name = System BIOS
18:21:57 VersionString = 1.3.6
18:21:57 Check classe CIM_Chassis
18:21:57 Element Name = Chassis
18:21:57 Manufacturer = Dell Inc.
18:21:57 SerialNumber = xxxxx
```

(continues on next page)

(continued from previous page)

```
18:21:57 Model = PowerEdge R710
18:21:57 Element Op Status = 0
18:21:57 Check classe CIM_Card
18:21:58 Element Name = unknown
18:21:58 Element Op Status = 0
18:21:58 Check classe CIM_ComputerSystem
CRITICAL: Execution time too long!
```

**A:** According to user feedback and lots of tests, such problems are related to the Dell OMSA Offline Bundle. Especially OMSA version 6.5 made problems on ESXi 5.x servers.

**Q:** I get the following error message from the plugin:

```
(0, 'Socket error: [Errno 111] Connection refused')
```

**A:** Make sure the Monitoring Server is able to access tcp port 5989 (cim) on the ESX(i) server. Alternatively you can also set a different port with the -C parameter if you have a special DNAT or port forwarding in place.

**Q:** How do I use the -i parameter to ignore certain alarms?

**A:** As written in the documentation, the -i parameter awaits a comma separated list of elements to ignore. The “tricky” part is to find the correct element names (they can be pretty long sometimes). Run the plugin in verbose mode to have a list of all CIM elements. Here’s an example how to ignore several elements:

```
./check_esxi_hardware.py -H myesxi -U root -P mypass -V dell -i "IPMI
SEL","Power Supply 2 Status 0: Failure status","System Board 1 Riser
Config Err 0: Config Error"
```

**Q:** I have the following warning showing up but my server shows all sensors green:

```
WARNING : System Board 1 Riser Config Err 0: Connected - Server: Dell
Inc. PowerEdge R620 s/n: xxxxxxxx System BIOS: 1.1.2 2012-03-08
```

**A:** It seems that all Dell PowerEdge x620 servers are affected, it looks like a BMC firmware bug to me. A workaround for this bugged is in place since version 20121027. Please check [this post](/blog/299/check_esxi_hardware_dell-pe-620-workaround-riser-config-err-connected) for detailed information.

**Q:** The plugin returns the following output:

```
Authentication Error! - Server:
```

**A:** There are several answers to that:

1. Make sure you are either using the ESXi root user or that you create a user which is member of the root group. See [this post](/blog/114/check-esxi-wbem-esxi-4.1-user-authorization) for a short description how to do that.
2. The password you are using has some special characters like a question mark and you need to quote them.
3. The password you are using has a Dollar sign (\$) which you need to single-quote.

Generally, always put quotes around your password as this assures the content is handled as string.

—  
**Q:** Can the plugin also monitor other stuff like VMFS disk usage or cpu/memory usage?

**A:** No. The plugin makes use of the CIM (Common Infrastructure Model) API . The so-called CIM elements cover hardware only.

—  
**Q:** Some hardware is not being monitored by the plugin.

**A:** The plugin can only monitor the hardware which is “shown” by the server via the CIM API. If the hardware vendor does not include a certain hardware element into the CIM elements, then this piece of hardware can not be monitored. In all the years I’ve only seen this on no-name machines (and SUN) though.

—  
**Q:** The plugin is so slow that a timeout occurs.

**A:** In such cases always verify how the behavior is on your vSphere client in the Hardware tab. Click on the “Update” link and then “Refresh”. Are they fast or do they also take a long time to update?

In ESXi 5.0 Update 1, a bug was causing slow hardware discovery/checks. See [this article](#) for more information.

—  
**Q:** The plugin suddenly times out, but it was working fine before. The plugin returns the following output:

```
UNKNOWN: (0, 'Socket error: [Errno 110] Connection timed out')
```

**A:** In rare cases it is possible, that the sfcdb-watchdog service, running on the ESXi server, isn’t working correctly anymore. Follow [VMware KB entry 1013080](#) and restart the service by logging into the ESXi server by ssh and launch the following command:

```
/etc/init.d/sfcdb-watchdog restart
```

If this still doesn’t resolve your issue, a manual restart of the “CIM Server” could help. This option is found under the “Configuration” tab → “Security Profile”. Click on “Service ... Properties”.

—  
**Q:** After an update of the pywbem package the plugin doesnt work anymore. The following output is shown in verbose mode:

```
Unknown CIM Error: (0, 'SSL error: certificate verify failed')
```

**A:** This was seen in SLES 11 SP3 after an update of the package python-pywbem from 0.7-6.13 to 0.7-6.22. After reverting to the older version, the plugin worked again.

Update September 9th 2014: This error will be fixed in a future release of check\_esxi\_hardware.py, but it depends on the release of the new pywbem upstream version. See [https://github.com/Napsty/check\\_esxi\\_hardware/issues/7](https://github.com/Napsty/check_esxi_hardware/issues/7).

**Note:** Update June 26th 2015: This issue was fixed in version 20150626.

**Q:** On an IBM server with the ESXi image from IBM the following error appears but works fine with the regular image vom VMware:

```
Traceback (most recent call last):

  File "./check_esxi_hardware.py", line 625, in verboseoutput("Element
Name = "+elementName)

TypeError: cannot concatenate 'str' and 'NoneType' objects
```

**A:** The CIM definition coming from the IBM image seems to be lacking some information. Version 20150119 fixes this issue.

**Q:** I updated my Ubuntu 14.04 and pywbem package 0.7.0-4ubuntu1~14.04.1 was installed. Since then I get the following error when the plugin is run:

```

Traceback (most recent call last):

  File "/usr/local/bin/check_esxi_hardware.py", line 619, in
<module>instance_list = wbemclient.EnumerateInstances(classe)

  File "/usr/lib/pymodules/python2.7/pywbem/cim_operations.py", line 421,
in EnumerateInstances

  \**params)

  File "/usr/lib/pymodules/python2.7/pywbem/cim_operations.py", line 183,
in imethodcall

no_verification = self.no_verification)

  File "/usr/lib/pymodules/python2.7/pywbem/cim_http.py", line 268, in
wbem_request

h.endheaders()

File "/usr/lib/python2.7/httplib.py", line 969, in endheaders

self._send_output(message_body)

File "/usr/lib/python2.7/httplib.py", line 829, in _send_output

self.send(msg)

File "/usr/lib/pymodules/python2.7/pywbem/cim_http.py", line 115, in
send

self.connect()

```

(continues on next page)

(continued from previous page)

```
File "/usr/lib/pymodules/python2.7/pywbem/cim_http.py", line 167, in
connect
```

```
except ( Err.SSLError, SSL.SSLError, SSL.SSLTimeoutError
```

```
AttributeError: 'module' object has no attribute 'SSLTimeoutError'
```

**A:** It seems that Ubuntu did the same as SUSE, RedHat and Centos in the past: The pywbem was patched without changing the upstream version number. This goes into the same direction as issue #7 ([https://github.com/Napsty/check\\_esxi\\_hardware/issues/7](https://github.com/Napsty/check_esxi_hardware/issues/7)). A temporary fix is to manually install the older pywbem package like this:

```
aptitude install python-pywbem=0.7.0-4
```

**Note:** Update June 26th 2015: This issue was fixed in version 20150626.

**Q:** I use python3 but the plugin throws an error:

```
File "./check_esxi_hardware.py3", line 440
```

```
print "%s %s" % (time.strftime("%Y%m%d %H:%M:%S"), message)
```

```
^
```

```
SyntaxError: invalid syntax
```

**A:** An issue was opened on github ([https://github.com/Napsty/check\\_esxi\\_hardware/issues/13](https://github.com/Napsty/check_esxi_hardware/issues/13)) to address this compatibility issue.

**Note:** Update: This issue was fixed in version 20181001.

**Q:** I sometimes get the following error on an ESXi host:

```
CRITICAL: (0, 'Socket error: [Errno 8] \_ssl.c:510: EOF occurred in
violation of protocol')
```

**A:** After a lot of debugging and testing with a plugin user we came to the conclusion, that this problem arises from the ESXi host, not the plugin. A tcpdump revealed, that the ESXi host sent a TCP Reset packet rather than starting to submit data. A reboot of the affected ESXi host resolved the problem.

**Note:** Update October 17th, 2019: Such situations can (sometimes) also be confirmed in the vSphere Client UI using the Monitor → Hardware Health window. A click on the “REFRESH” button results in an error in the recent tasks list:

“A general system error occurred: Server closed connection after 0 response bytes read; <SSL....

Recent Tasks		Alarms
Task Name	Target	Status
Refresh hardware IPMI System Event Log	[REDACTED]	<div></div> A general system error occurred: Server closed connection after 0 response bytes read; <SSL(<io_obj p:0x00007fb1205ef600, h:129, <TCP '10.162.214.31 : 45358'>, <TCP '10.251.8.87 : 443'>>)>
Refresh hardware IPMI System Event Log	[REDACTED]	<div></div> A general system error occurred: Server closed connection after 0 response bytes read; <SSL(<io_obj p:0x00007fb168295fe0, h:5, <TCP '10.162.214.31 : 45202'>, <TCP '10.251.8.87 : 443'>>)>

**Q:** I have several ESXi hosts behind the same IP (NAT). How can I use the `check_esxi_hardware`?

**A:** Since version 20160531 it is possible to manually define the CIM port (which defaults to 5989). So if you set up port forwarding (DNAT) you can now monitor all ESXi servers behind the same NAT-address. The parameter you want in this case is `”-C”` (or `-cimport`).

**Q:** Is the plugin compatible with ESXi 6.x?

**A:** Yes. Please note that starting with ESXi 6.5 you might have to enable the CIM/WBEM services first, as they are disabled by default. Refer to <https://kb.vmware.com/s/article/2148910>.

pcscod	PC/SC Smart Card Daemon	Stopped	Base system	None
sfcbd-watchdog	CIM Server	Stopped	Base system	CIMHttpServer, CIMHttpsServer
snmpd	SNMP Server	Stopped	Base system	snmp

**Q:** I can’t execute the plugin and get the following error message. Permissions are correct however (e.g. 755).

```
execvpe(/usr/lib64/nagios/plugins/check_esxi_hardware.py) failed:
Permission denied
```

**A:** This error comes from SELinux. You need to write an allow rule for it.

**Q:** The plugin reports the following problem with memory, but no memory hardware issues can be found on the server:

```
CRITICAL : Memory - Server: HP ProLiant DL380p Gen8 s/n....
```

**A:** It is possible that an alert needs to be cleared in the servers IPMI log first. To do that, you need to login into your ESXi server with SSH and run the following commands:

```
localcli hardware ipmi sel clear

/sbin/services.sh restart
```

This might affect other CIM entries as well. So it’s a wise idea to clear the IPMI system event log (sel) first before investigating further.

**Q:** Certain hardware elements show incorrect health/operational states, e.g. “Cooling Unit 1 Fans”:

```
20190205 00:26:26

Element Name = Cooling Unit 1 Fans

20190205 00:26:26

Element HealthState = 1020190205 00:26:26

Global exit set to WARNING
```

**A:** Certain server models might show false hardware alarms when these particular hardware elements were disabled in BIOS, are idle or have disabled sensors. From the [HP FAQ](#):

---

**Note:** PR 2157501: You might see false hardware health alarms due to disabled or idle Intelligent Platform Management Interface (IPMI) sensors. Disabled IPMI sensors, or sensors that do not report any data, might generate false hardware health alarms.

---

In this case it makes sense to ignore these elements using the `-i` parameter.

—

**Q:** The `check_esxi_hardware` plugin is not working (anymore) since ESXi 6.7 U2/U3 on DELL servers.

**A:** The issue seems to be the “OpenManage” VIB. This can be verified by checking the list of installed VIB’s on an ESXi server:

```
esxcli software vib list
```

After uninstalling the OpenManage VIB, the plugin works again. According to DELL, ESXi 6.7 U2 is [not yet officially supported](#) (as of July 2019) by OpenManage:

---

**Note:** OpenManage Integration for VMware vCenter v4.3.1 (Initial 4.3 Download) (4.3.1 Release Notes) (4.3 Manuals) Does not add official 6.7 U2 support (support for 6.7 U2 will come in the fall with the next major release)

---

See also official [VMware KB 74696](#) entry for this.

Update October 15th 2019: OMSA 9.3.1 fixes this issue.

### 3.3.5 Check Fortinet FortiGate Appliances (`check_fortigate`)

NEMS Linux makes it easy and affordable to monitor your Fortinet Fortigate devices and clusters via SNMP, complete with perf data.

`check_fortigate` requires NEMS Linux 1.7 or higher.

---

#### Veriage Disclaimer

Please note that Fortinet’s use of the terms “master” and “slave” to distinguish between primary and secondary devices is an industry-standard terminology in the context of networking hardware. While NEMS Linux respects the terminology used by Fortinet for its products, it’s essential to recognize that these terms can carry historical connotations and may not reflect the values of equity and inclusivity that we strive for in modern society. NEMS Linux is committed to supporting equity, diversity, and inclusion in all aspects of its development and operation.

---

#### Configuring Fortigate

From the Fortigate web interface:

1. Select Network -> Interface -> Local interface
2. Administrative Access: Enable SNMP
3. Select Config -> SNMP
4. Enable SNMP, fill your details
5. SNMP v1/v2c: Create new
6. Configure for your needs, Traps are not required for this plugin!



## Available checkcommands

### Check Fortigate Cluster

Check the status of a Fortigate cluster, providing information about the active/passive state and any warnings or critical alerts.

**Check Command:** *check\_fortigate\_cluster*

**Parameters:** Community Name (default: public)

### Check Fortigate CPU

Monitor the CPU usage of a Fortigate device, ensuring that it remains within acceptable thresholds to maintain optimal network performance.

**Check Command:** *check\_fortigate\_cpu*

**Parameters:** Community Name (default: public)

### Check Fortigate RAM

Monitor the RAM usage of a Fortigate device, ensuring that it remains within acceptable thresholds to prevent network performance degradation.

**Check Command:** *check\_fortigate\_mem*

**Parameters:** Community Name (default: public)

### Check Fortigate Network

Monitor the network traffic on a Fortigate device, ensuring that it remains within acceptable thresholds to prevent congestion and bottlenecks.

**Check Command:** *check\_fortigate\_net*

**Parameters:** Community Name (default: public), Warning Bytes (default: 500000), Critical Bytes (default: 1000000)

### Check Fortigate Sessions

Monitor the session count on a Fortigate device, ensuring that it remains within acceptable thresholds to prevent resource exhaustion.

**Check Command:** *check\_fortigate\_ses*

**Parameters:** Community Name (default: public), Warning Sessions (default: 4500), Critical Sessions (default: 6000)

### Check Fortigate Slave CPU

Monitors the CPU usage of a slave unit in a Fortigate cluster, ensuring that it remains within acceptable thresholds for optimal performance.

**Check Command:** *check\_fortigate\_slave\_cpu*

**Parameters:** Community Name (default: public)

### Check Fortigate Slave RAM

Monitor the RAM usage of a slave unit in a Fortigate cluster, ensuring that it remains within acceptable thresholds to prevent performance degradation.

**Check Command:** *check\_fortigate\_slave\_mem*

**Parameters:** Community Name (default: public)

### Check Fortigate Slave Network

Monitor the network traffic on a slave unit in a Fortigate cluster, ensuring that it remains within acceptable thresholds to prevent congestion and bottlenecks.

**Check Command:** *check\_fortigate\_slave\_net*

**Parameters:** Community Name (default: public), Warning Bytes (default: 500000), Critical Bytes (default: 1000000)

### Check Fortigate Slave Sessions

This command monitors the session count on a slave unit in a Fortigate cluster, ensuring that it remains within acceptable levels to prevent resource exhaustion.

**Check Command:** *check\_fortigate\_slave\_ses*

**Parameters:** Community Name (default: public), Warning Sessions (default: 4500), Critical Sessions (default: 6000)

### Check Fortigate VPN

Monitor the status of VPN connections on a Fortigate device, ensuring that they are operational and secure. This check supports both IPSec and SSL/OpenVPN connections.

**Check Command:** *check\_fortigate\_vpn*

**Parameters:** Community Name (default: public), VPN Mode: ipsec, ssl, both (default: both)

## Check Fortigate Access Points

Check the status of FortiAPs (WTPs) on a Fortigate device, ensuring that they are operational and properly configured.

**Check Command:** *check\_fortigate\_wtp*

**Parameters:** Community Name (default: public)

## CLI Usage

```
check_fortigate.pl -H -C -T [-w|-c|-S|-s|-R|-M|-V|-?]
```

## Options

- H, --host** STRING or IPADDRESS  
Check interface on the indicated host.
  - P, --port** INTEGER  
Port of indicated host, defaults to 161.
  - v, --version** STRING  
SNMP Version, defaults to SNMP v2, v1-v3 supported.
  - T, --type** STRING  
CPU, MEM, Ses, VPN, Cluster, Firmware, HW, etc.
  - S, --serial** STRING  
Primary serial number.
  - s, --slave**  
Get values of slave.
  - w, --warning** INTEGER  
Warning threshold, applies to cpu, mem, session, firmware.
  - c, --critical** INTEGER  
Critical threshold, applies to cpu, mem, session, firmware.
  - R, --reset**  
Resets ip file (cluster only).
  - M, --mode** STRING  
Output-Mode: 0 => just print, 1 => print and show failed tunnel, 2 => critical.
  - V, --vpnmode** STRING  
VPN-Mode: both => IPSec & SSL/OpenVPN, ipsec => IPSec only, ssl => SSL/OpenVPN only.
- SNMP v1/v2c only:
- C, --community** STRING  
Community-String for SNMP, only at SNMP v1/v2c, defaults to public.
- SNMP v3 only:
- U, --username** STRING  
Username.

- A, --authpassword** STRING  
Auth password.
- a, --authprotocol** STRING  
Auth algorithm, defaults to sha.
- X, --privpassword** STRING  
Private password.
- x, --privprotocol** STRING  
Private algorithm, defaults to aes.
- ?, --help**  
Returns full help text.

## Dependencies

These dependencies are preinstalled on NEMS Linux:

- Net::SNMP
- List::Compare
- Getopt::Long
- Pod::Usage
- Switch

## CLI Examples

To use SNMPv3 just replace `-C public` with `-v 3 -U username -A this_is_auth_string -a sha -x aes128 -X this_is_priv_string`.

Cluster:

```
$ check_fortigate.pl -H 192.168.123.100 -C public -T cluster

OK: Fortinet 300C (Master: FGSERIALMASTER, Slave: FGSERIALSLAVE): HA (Active/Passive) is active
- Warning if unknown node appears
- Critical if single node
- Optional: Critical, if preferred master (-S Serial) is not master
```

CPU:

```
$ check_fortigate.pl -H 192.168.123.100 -C public -T cpu

OK: Fortinet 300C (Master: FGSERIALMASTER) CPU is okay: 1%|'cpu'=1%;80;90
```

CPU-Slave:

```
$ check_fortigate.pl -H 192.168.123.100 -C public -T cpu -s

OK: Fortinet 300C (Master: FGSERIALMASTER) slave_CPU is okay: 5%|'slave_cpu'=5%;80;90
- Defaults: 80%/90%
```

Memory:

```
$ check_fortigate.pl -H 192.168.123.100 -C public -T mem  
OK: Fortinet 300C (Master: FGSERIALMASTER) Memory is okay: 29%|'memory'=29%;80;90
```

Memory-Slave:

```
$ check_fortigate.pl -H 192.168.123.100 -C public -T mem  
OK: Fortinet 300C (Master: FGSERIALMASTER) slave_M
```

## Source

From <https://github.com/riskersen/Monitoring/tree/master/fortigate>

### 3.3.6 Check Command: check\_http

Check the status of a HTTP/HTTPS server running on a domain or IP address on a remote host.

The included default check\_https command uses the IP address of your host, as configured in NEMS Configurator.

## check\_http Argument Syntax

<b>-H</b>	<b>-\-hostname</b>	host name of the server where HTTP (or HTTPS) daemon is running
<b>-I</b>	<b>-\-IP-address</b>	ip address of the HTTP (or HTTPS) server
<b>-p</b>	<b>-\-port</b>	Port number where HTTP server runs. Default is 80
<b>-4</b>	<b>-\-use-ipv4</b>	This will use IPv4 connection
<b>-6</b>	<b>-\-use-ipv6</b>	This will use IPv6 connection
<b>-S</b>	<b>-\-ssl</b> or <b>-\-sni</b>	This will use HTTPS using default 443 port Enable SSL/TLS hostname extension support (SNI)
<b>-C</b>	<b>-\-certificate</b>	Minimum number of days a SSL certiface must be valid.
<b>-e</b>	<b>-\-expect</b>	Expected response string. Default is HTTP/1
<b>-s</b>	<b>-\-string</b>	Expected content string.
<b>-u</b>	<b>-\-url</b>	to check
<b>-P</b>	<b>-\-post</b>	encoded http POST data
<b>-N</b>	<b>-\-no-body</b>	Do not wait for whole document body to download. Stop once the headers are downloaded.
<b>-M</b>	<b>-\-max-age</b>	Check whether a document is older than x seconds. Use 5 for 5 seconds, 5m for 5 minutes, 5h for 5 hours, 5d for 5 days.
<b>-T</b>	<b>-\-content-type</b>	Indicate content type in header for POST request
<b>-l</b>	<b>-\-linespan</b>	Regular expression can span to new line (Use this with -r or -R option)
<b>-r</b>	<b>-\-regex</b> or <b>-\-ereg</b>	Use this regular expression to search for string in the HTTP page
<b>-R</b>	<b>-\-eregi</b>	Same as above, but with ignore case.
<b>-a</b>	<b>-\-authorization</b>	If the site uses basic authentication send uid, pwd in the format uid:pwd
<b>-A</b>	<b>-\-useragent</b>	Pass the specified string as “User Agent” in HTTP header.
<b>-k</b>	<b>-\-header</b>	Add additional tags that should be sent in the HTTP header.
<b>-L</b>	<b>-\-link</b>	The output is wrapped as link
<b>-f</b>	<b>-\-onredirect</b>	When a is redirected, use this to either follow the , or send ok, warning, or critical notification
<b>-m</b>	<b>-\-pagesize</b>	Specify the minimum and maximum page size expected in bytes. Format is minimum:maximum
<b>-m</b>	<b>-\-pagesize</b>	Specify the minimum and maximum page size expected in bytes. Format is minimum:maximum
<b>-w</b>	<b>-\-warning</b>	Response time in seconds for warning state
<b>-c</b>	<b>-\-critical</b>	Response time in seconds for critical state
<b>-t</b>	<b>-\-timeout</b>	Number of seconds to wait before connection times out. Default is 10 seconds

## NEMS Configurator Service Parameter Examples

Check if a host is responding on the default http port (ie., 80):

```
[blank]
```

Check if a host is responding on the default https port (ie., 443):

```
-S
```

Check if a host is responding on an alternate https port (ie., 8080):

```
-S -p 8080
```

Check the state of the hosts SSL certificate and treat as a problem if it expires in 30 days or less:

```
-C 30
```

## Troubleshooting

### I receive a message “CRITICAL - Cannot make SSL connection. SSL alert number 40” when trying to check a secure web site

Are you using Cloudflare or another SSL redirecting system where your certificate might be shared with other host-names? Try adding `-sni` to your service parameters to enable Server Name Indication (SNI), which allows the server to safely host multiple TLS Certificates for multiple sites, all under a single IP address.

Test openssl’s response by running this command from your NEMS server:

```
openssl s_client -connect YOURDOMAIN.COM:443 -debug
```

### I receive a message “CRITICAL - Socket timeout after 10 seconds” on NEMS TV Dashboard, Adagios and so-on

This means the particular board you’re using to run NEMS is a bit slow for the task. By default, `check_http` will timeout after 10 seconds. But what happens if your board takes 11? It generates the error “CRITICAL - Socket timeout after 10 seconds”.

To remedy this, yes, you could move to a faster board. But I’d suggest you could also add these two things to your service check ARGS as per the syntax above:

`-N` - only download the headers: this will result in a smaller transaction, which in turn takes less time.

`-T 30` - increase the timeout to 30 seconds.

So your ARGS would become:

```
-N -T 30
```

## 3.3.7 Check Command: `check_ilo2_health`

Check hardware health of HP Proliant Servers by querying the HPE Integrated Lights-Out (iLO) 2/3/4/5 Management Controller.

No need for `snmp` or installation of software.

Checks if all sensors are ok, returns warning on high temperatures and fan failures and critical on overall health failure.

**Note:** The plugin shows only temperature sensors by default. Faulty hardware components are only listed if iLO returns an error state.

The plugin makes use of the HP Lights-Out XML scripting interface.

### Arguments [Optional]

- `-e`: plugin ignores “syntax error” messages in the XML output. This may help for older firmwares.
- `-n`: output without temperature listing.
- `-d`: add PerfParse compatible temperature output.
- `-v`: print out the full XML output from the BMC.
- `-3`: support for iLO3/4
- `-a`: check fan redundancy (only some models)

- -c: check drive bays (only some models)
- -o: check power redundancy (only some models)
- -b: temperature output with location
- -l: parse iLO eventlog
- -b: show temperature with location
- -x: ignore battery missing
- -i: ignore NIC Link Down status (iLO4).
- -g: display additional infos like firmware version and servername (may need increased timeout!)
- -f: read input from file instead from iLO, possible to feed -v output to it
- --sslopts: Defaults to 'SSL\_verify\_mode => SSL\_VERIFY\_NONE'. Use 'SSL\_verify\_mode => SSL\_VERIFY\_NONE, SSL\_version => "TLSv1"' to avoid TLS Downgrade bug.

### Source

- [https://exchange.nagios.org/directory/Plugins/Hardware/Server-Hardware/HP-%28Compaq%29/check\\_ilo2\\_health/details](https://exchange.nagios.org/directory/Plugins/Hardware/Server-Hardware/HP-%28Compaq%29/check_ilo2_health/details)

### 3.3.8 Check Command: check\_internet\_speed

Tests the speed of your Internet connection.

#### Demo Data

Out of the box, NEMS Linux 1.5+ will check the Internet connectivity speed using [Cloudflare's Speed Test](#) service. By default, warning notifications will be generated if your upload or download speed falls below 10 Mb/s and Critical warnings if either falls below 7 Mb/s. These are just sample thresholds which can (and should) be modified to suit your connection speed by modifying the service in NEMS NConf.

#### Data Usage Warning

**Every time the speedtest runs, anywhere from 100-400 MB of data is transferred** (depending on your connection speed). Since the sample service is set to check every 30 minutes while in a good state, that could be nearly 5GB of data per day. If your Internet is slow, the sample service will check (retry) every 5 minutes, increasing the bandwidth usage significantly.

While in most cases this is fine, you *must* modify your thresholds to suit your connection, and modify your service schedule if you have limited or pay-per-use bandwidth.



## Logging

A cache log is saved at `/var/log/nems/speedtest.log` whenever the script is run. In NEMS Linux 1.6+, this cache is used by NEMS TV Dashboard to display current Internet speed.

The format of this multi-line cache file is as follows:

```
State of service
Ping time
Ping measurement
Download speed
Download speed measurement
Upload speed
Upload speed measurement
```

## Troubleshooting

### Service check timed out after [number] seconds

If you are receiving a service check timeout and are certain you do indeed have Internet connectivity, the most likely culprit is that your NEMS Server is unable to process the speedtest within 120 seconds. A Raspberry Pi 4 with a reasonably fast MicroSD card should be able to perform a speedtest in under 40 seconds. Please upgrade your NEMS Server to an officially-supported platform. Power users may increase the value of `service_check_timeout` in `/usr/local/nagios/etc/nagios.cfg` however this may result in cascading checks, which will bring a low-powered SBC to its knees. You are much safer to upgrade to a board that meets your requirements.

### 3.3.9 Check Command: `check_iperf` / `check_iperf3`

Monitor the speed between network links using the Linux `iperf` or `iperf3` command respectively.

#### Arguments

**Critical** A number representing the low unit result to treat service as critical.

**Warning** A number representing the low unit result to treat service as warning.

**Speed** [Optional] Set the test speed (E.g., 40M or 1G).

#### Source

- [https://exchange.nagios.org/directory/Plugins/Network-Connections%2C-Stats-and-Bandwidth/check\\_iperf/details](https://exchange.nagios.org/directory/Plugins/Network-Connections%2C-Stats-and-Bandwidth/check_iperf/details)
- [https://exchange.nagios.org/directory/Plugins/Network-Connections,-Stats-and-Bandwidth/check\\_iperf3/details](https://exchange.nagios.org/directory/Plugins/Network-Connections,-Stats-and-Bandwidth/check_iperf3/details)

### 3.3.10 Check Command: check\_mikrotik

#### Commands

#### check\_mikrotik\_voltage

Warning Voltage Critical Voltage

#### check\_mikrotik\_cpu

Warning CPU % Critical CPU %

#### check\_mikrotik\_temp

Warning Temperature Critical Temperature

#### check\_mikrotik\_port\_sum

check\_mikrotik\_port\_sum -ARG1 -ARG2

- **ARG1: Test Name**

- cpu
- activeFan
- voltage
- temperature
- processorTemperature
- current
- powerConsumption
- psu1State
- psu2State
- disk
- diskTotal
- diskUsed
- mem
- memTotal
- memUsed
- portName
- portOperState
- portAdminState
- portMtu
- portMacAddress

- portRxDiscards
- portTxDiscards
- portTxErrors
- portRxEErrors
- portRxPackets
- portTxPackets
- portTxBytes
- portRxBytes

- **ARG2: Switch Ports**

- Can be single, multiple and ranges possible, for example: 1, 5-10, 22-24

### check\_mikrotik\_port\_info

Same as check\_mikrotik\_port\_sum but output info without any thresholds.

## Sample Services

These need to be converted for NEMS NConf.

### Check Voltage

This will check each host that is listed in the MikroTik Switches group. It will issue a warning if the voltage is below 23V or above 26V and a critical error if it is below 22V or above 27V

```
define service {
    use                generic-service
    hostgroup_name     MikroTik Switches
    service_description MikroTik Voltage
    check_command      check_mikrotik_voltage!23:26!22:27
}
```

### Check TX Errors

This test adds up all tx-errors on ports 1-25 (all ports on a CRS125-24G-1S).

```
define service {
    use                generic-service
    hostgroup_name     MikroTik Switches
    service_description MikroTik TX Errors
    check_command      check_mikrotik_port_sum!portTxErrors!1-25!10!50
}
```

### Port Names

This test returns the port names of ports 1, 3, 4, 5 and 25

```
define service {
    use                generic-service
    hostgroup_name     MikroTik Switches
```

(continues on next page)

(continued from previous page)

```

    service_description      MikroTik Port Names
    check_command            check_mikrotik_port_info!portName!1,3-5,25
}

```

Source: [https://github.com/bemworld/check\\_mikrotik\\_switch](https://github.com/bemworld/check_mikrotik_switch)

### 3.3.11 Check Command: `check_minecraft`












`check_minecraft` provides the current state of the Minecraft server (Online, Down or Full), and also provides the following performance data:

- *Online Players* provides the current count of online players, plus the server maximum. A warning threshold is automatically set when the server reaches 75% of its set limits, and a critical threshold is met when 90% of the allowed players are connected.
- *Minecraft Version* informs you of the version of the running Minecraft server.
- *Response Time* tells you, in milliseconds, how long the Minecraft server is taking to respond.

**Service**  Patron Survival Server **on host**  Pinecraft



**General**   **Information**   **History**   **Custom Variables**   **Debug**

Status	 ok for 16 minutes																																						
Host Name	 <b>Pinecraft</b>																																						
Host Address	192.168.0.70																																						
Host Alias	Cat5 Pinecraft Server																																						
Service Name	Patron Survival Server																																						
Check output	online - 31/100 players																																						
Performance Metrics	<table> <tr> <th></th><th>label</th><th>value</th><th>warn</th><th>crit</th><th>unit</th><th>min</th><th>max</th></tr> <tr> <td></td><td>Online Players</td><td>1</td><td>75</td><td>90</td><td></td><td>0</td><td>100</td></tr> <tr> <td></td><td>Minecraft Version</td><td>1.20.2</td><td></td><td></td><td></td><td></td><td></td></tr> <tr> <td></td><td>Response Time</td><td>0.017</td><td></td><td></td><td>s</td><td></td><td></td></tr> </table>								label	value	warn	crit	unit	min	max		Online Players	1	75	90		0	100		Minecraft Version	1.20.2							Response Time	0.017			s		
	label	value	warn	crit	unit	min	max																																
	Online Players	1	75	90		0	100																																
	Minecraft Version	1.20.2																																					
	Response Time	0.017			s																																		

`check_minecraft` will first try to connect to your Minecraft server using a modern ( $\geq 1.7$ ) connection. If it fails to connect, it will then try using a legacy ( $\leq 1.6$ ) connection. If it fails then, the server is considered offline. This method allows NEMS Linux to check any version of Minecraft without your having to specify the version manually and remembering to update your check command every time you upgrade Minecraft.

## Get Your Own Minecraft Server

Need a Minecraft server? Create one for free and host it on a Raspberry Pi! Download [Pinecraft Installer](#) today!

*check\_minecraft* is compatible with all versions of Pinecraft Installer-powered Minecraft servers (as well as most other Minecraft, Paper, Spigot, Forge, etc. servers).

### Command Line

```
./check_minecraft -H [host_address] -P [game_port]
```

### 3.3.12 Check MS SQL Memory Usage (*check\_mssql\_mem*)

Monitor your MS SQL 2008, 2012 or 2019 server memory usage.

*check\_mssql\_mem* requires NEMS Linux 1.7 or higher.

### Configuring MS SQL Server

Install NSClient++ w/ NRPE and enable it on your MS SQL server, ensuring “allow\_arguments” and “allow\_nasty\_meta\_chars” options are enabled.

Enable SQL Server Perfmon Counters.

#### Parameters:

- *Port* MS SQL port. [Default 5666]
- *Check* Determines which check to run: - **db\_memory\_calc** - Calculates SQL Memory in use [Default] - **db\_pages\_used** - Pages in use in MB - **db\_pages\_total** - Total pages in MB
- *Warning %* - Specify max percentage for a WARNING state.
- *Critical %* - Specify max percentage for a CRITICAL state.

### Source

Originally from <https://github.com/grune/Nagios/blob/master/Nagios/nagios-mssql-memory.py> with significant changes for NEMS Linux in order to upgrade to Python 3 and enable Server 2019 support.

### 3.3.13 Check Command: *check\_ncpa*

Check Command version 1.2.3 added 1.6.

This plugin works with NCPA 2+ and is backwards compatible with NCPA 1.

I’ve included a couple sample check commands in NEMS Linux 1.6 to get you started:



*check\_ncpa\_mem* Check the memory usage of a system.








*check\_ncpa\_processes* Check number of running processes on a system.

See <https://www.nagios.org/ncpa/help/2.0/active.html> and <https://www.nagios.org/ncpa/>

### 3.3.14 Check Command: check\_nems\_osb

Monitor the status of your NEMS Migrator Off-Site Backup. If your backup fails, this check will notify you.

**Service**  NEMS Migrator Off-Site Backup **on host**  NEMS








More ▾

[General](#)
[Information](#)
[History](#)
[Custom Variables](#)
[Debug](#)



Status	 ok for 1 hour, 34 minutes
Host Name	 <b>NEMS</b>
Host Address	127.0.0.1
Host Alias	Nagios Enterprise Monitoring Server
Service Name	NEMS Migrator Off-Site Backup
Check output	OK - OSB Success (Tue Dec 8, 2020)
Performance Metrics	No Perfdata available.

Fig. 1: check\_nems\_osb output in NEMS Adagios (circa NEMS Linux 1.6)

If you do not have a NEMS Cloud Services account, `check_nems_osb` will return an *UNKNOWN* state.

Additionally, even if the last backup was successful, `check_nems_osb` will go into a *WARNING* state if the backup has not run for a couple days.

The included sample will not notify you if it is in this state (only *WARNING*, *CRITICAL* and *RECOVERED* will notify, unless you change the notification options of the included sample service).

`check_nems_osb` does not have any arguments. By adding it to your NEMS host, it will check and notify based on the state of your backups. There are no thresholds required since a failed backup is *CRITICAL*, and a successful backup is *OK*.

### 3.3.15 NEMS PHP Server Agent

Monitor your Linux-based PHP-enabled web server with the NEMS PHP Server Agent.

#### Why Use the NEMS PHP Server Agent

The NEMS PHP Server Agent is designed specifically to collect system data from a PHP server. Disk space and usage, memory size and usage, system load, etc. Unlike NRPE, the agent reports very specific data rather than running remote commands on your server. This makes it easier to use on a public server where firewall rules might be too complex for novice sysadmins to make NRPE a safe option. The NEMS PHP Server Agent is designed to be safe out of the box, and incredibly easy to deploy: Just upload it to a public folder on your web server and point your check commands on your NEMS Server to the URL.

#### Requirements

A Linux-based web server with PHP 5.2+ which can be reached by your NEMS Linux server (public web-based, or LAN).

I have tested successfully on Debian 10+ with PHP 7.3+ as well as Debian 7 with PHP 5.2. I have not tested on any non-Debian system, so if you do, please let me know if it works or not and I will add it here.

#### Usage

##### Obtaining your PHP Agent

In NEMS SST, download your custom nems-agent.php file. Upload this file to a web-accessible folder on your web server and add the NEMS PHP Agent check command in NEMS NConf for any checks you would like to perform.

---

**Tip:** You'll notice in the check commands below, the actual agent filename is entered within the URL. This is intentional, and allows you to add obscurity if desired by naming your NEMS PHP Server Agent anything you like (as long as it has a .php extension and can be seen by your NEMS Server).

---

#### Local and Remote Monitoring

Set the URL parameter to a web-accessible URL (such as `https://example.com/nems-agent.php`) or use a LAN server URL (such as `http://192.168.0.55/nems-agent.php`).

#### Check Command

`check_nems_php_agent` is part of NEMS Linux 1.6+.

### Check Command Arguments

- **Warn Threshold / Critical Threshold** - Set your thresholds. Can be a positive floating point number or integer.
- **URL** - The full URL to your *nems-agent.php* on the remote server. File can be renamed as desired, but provided URL must resolve to the agent on the remote server.
- **Check**
  - **mem** Percent Memory Usage
  - **disk** /path Percent Disk Usage of /path. /path is optional. Defaults to /. If target is not mounted, will trigger CRITICAL state, so be sure /path is your mountpoint. This way, you can have one check for / and another for /mnt/backup and if the backup drive dismounts, it will turn CRITICAL.
  - **net** Network Usage Mb/s
  - **netrx** Network Usage Mb/s Download Only
  - **nettx** Network Usage Mb/s Upload Only
  - **load** 15 Minute System Load Average

---

**Note:** All network checks require *ifstat* be installed on the remote server. This can easily be installed with apt or yum, depending on your distro.

---

### CLI Examples

WARN if 15 minute system load average exceeds 3, CRIT if over 9:

```
./check_nems_php_agent 3 9 https://example.com/nems-agent.php load
```

WARN if /mnt/backup disk usage is over 80%, CRIT if over 90%. Will also be CRIT if the disk is unmounted from /mnt/backup on the destination server.

```
./check_nems_php_agent 80 90 https://example.com/nems-agent.php disk /mnt/backup
```

WARN if either up or down network usage exceed 1 Mb/s, CRIT if over 2 Mb/s:

```
./check_nems_php_agent 1 2 https://example.com/nems-agent.php net
```

### Data Security

All data is AES-128-ECB encrypted server-side using an encryption/decryption key you provide in NEMS System Settings Tool.



## Under The Hood

The agent outputs the following JSON string (Sample data from early release):

```
{
  "ver": {
    "nems": "1.6",
    "nemsagent": "1.1",
    "data": "pICGwq5UL308yNEYdPrQh\\/  

    ↪8PGCjsXQUx9mh9VIQloFJ\\K8BsB5AT9L2ixwlsiDAJGjWR1RnhsrCFHVnKD9p3cmRxbQf\\/  

    ↪knW6F+EkDS3Cnkr1XLWSPJ6p+gfZjIq16NSREvfaaPJZEY93mBrgSFars+C8advgKL+0jz2a55ItGk0BY6AKv0MuFXfxzwd3i7485  

    ↪/wfPmUtkEtU841FVmcfGLxcEIooGzG9vjH8q7urs2RetcBVpVhj5Z+T+v8qa9oQ7Pi1tbf2\\/  

    ↪IhF+eLE9cSkmMlmbFbJ70hjJaY2gssiwb9tZ6g0dX+WA8+ujTzmCzBdNJ09HabaLVzXTqR4cGyFM3mXYQ1+SdDSdmeZ\\  

    ↪/vw\\sG4oSfxxKzhxmOpCM5qBw==",
    "auth":  

    ↪"312433c28349f63c4f387953ff337046e794bea0f9b9ebfcb08e90046ded9c76"}
}
```

That is essentially what a user would see if they were to open the agent in their browser, and is what is downloaded to your NEMS Server when the check commands run.

Your NEMS Server knows your decryption key used by the agent to encrypt the data. When decrypted by your NEMS Server, the data looks something like this:

```

Array
(
    [ver] => Array
        (
            [nems] => 1.6
            [nemsagent] => 1.1
        )
    [data] => Array
        (
            [cpu] => Array
                (
                    [usage] => 0
                    [model] => Intel(R) Xeon(R) CPU E5-2676 v3 @ 2.40GHz
                    [loadaverage] => Array
                        (
                            [1] => 0
                            [5] => 0
                            [15] => 0
                        )
                )
            [mem] => Array
                (
                    [percent] => 23.5
                    [total] => 0.472
                    [free] => 0.032
                    [used] => 0.44
                )
            [storage] => Array
                (
                    [.] => Array
                        (
                            [path] => /var/www/html
                            [free] => 6.11
                            [total] => 7.69
                            [used] => 1.58
                            [percent] => 0
                        )
                )
        )
)

```

(continues on next page)

(continued from previous page)

```

    )
    [/] => Array
    (
        [free] => 6.11
        [total] => 7.69
        [used] => 1.58
        [percent] => 0
    )
    [/var] => Array
    (
        [free] => 6.11
        [total] => 7.69
        [used] => 1.58
        [percent] => 0
    )
    )
    [network] => Array
    (
        [rx] => 0.01
        [tx] => 0.01
    )
    )
    [auth] => 312433c28349f63c4f387953ff337046e794bea0f9b9ebfcb08e90046ded9c76
    )

```

The “auth” hash is a cryptographically-safe hash of your encrypted passphrase, and is what your NEMS Server uses to ensure the NEMS Server passphrase matches that of your NEMS PHP Server Agent. In this way, a third party cannot find a nems-agent.php running on your server and access your data from their NEMS Server. They will receive an error that the auth key does not match. Similarly, it means you can deploy your NEMS PHP Server Agent on as many PHP servers as you like, and even use multiple NEMS Servers to monitor it (as long as you key in the same passphrase on each NEMS Server).

This data output above is used by your NEMS Server’s *check\_nems\_php\_agent* check commands.

### 3.3.16 Check NetScaler ADC (check\_netscaler)

NEMS Linux monitors Citrix NetScaler Application Delivery Controllers.

---

**Note:** *check\_netscaler* requires NEMS Linux 1.7 or higher.

---

#### NEMS NConf Parameters

- **Check Name** - See the table below for the available commands.
- **Object Name** - The object name to check.
- **Username** - The username to use to connect. This should be a limited account.
- **Password** - The password of that limited user.
- **Warning %** - The percentage to trigger a WARNING state.
- **Critical %** - The percentage to trigger a CRITICAL state.

## Commands

- `state` - check the current service state of vservers (e.g. lb, vpn, gslb), services, and service groups and servers
- `matches`, `matches_not` - check if a string matches the API response or not
- `above`, `below` - check if a value is above/below a threshold (e.g. traffic limits, concurrent connections)
- `sslcert` - check the lifetime for installed SSL certificates
- `nsconfig` - check for configuration changes which are not saved to disk
- `license` - check the expiry date of a locally installed license file
- `hastatus` - check the high availability status of an appliance
- `staserver` - check if configured STA (secure ticket authority) servers are available
- `servicegroup` - check the state of a service group and its members
- `hwinfo` - just print information about the Netscaler itself
- `interfaces` - check the state of all interfaces and add performance data for each interface
- `perfdata` - gather performance data from all sorts of API endpoints
- `ntp` - check the NTP synchronization status
- `debug` - debug command, print all data for an endpoint

## Source

From [https://github.com/slauger/check\\_netscaler](https://github.com/slauger/check_netscaler)

### 3.3.17 Check Command: `check_nrpe`

#### Configuration

#### Firewall Ports

If you have a firewall between your NEMS Server and your host, you must open incoming TCP connections on ports `5666` and `12489` on your host side.

Since `check_nrpe` can pose a security risk, please **do not** open these ports to the world. Rather, ensure your NEMS Server is the only outside IP allowed to connect to these ports.

#### Host-Side Service

The NRPE service must be installed before `check_nrpe` can be used to monitor the host.

See:

- [NRPE For Linux](#)
- [NRPE For Windows Clients](#)

### Important Note for Users of NEMS 1.5 and Under

In NEMS Linux 1.5 and lower, an older version of NRPE was used. This can be upgraded by running `sudo nems-upgrade` however you must also make a few minor changes in NEMS NConf as follows:

- Edit `check_nrpe` in `checkcommands`, changing the command line to: `$USER1$/check_nrpe -H $HOSTADDRESS$ -c $ARG1$` (essentially, just removing the quotation marks).
- Edit `/ Disk Space` and `/var Disk Space` in `Advanced Services`, changing the ARGS to: `check_disk -a '-w 80 -c 90 -p / -u GB'` and `check_disk -a '-w 80 -c 90 -p /var -u GB'` respectively.

### Sample Args for check\_nrpe

In order to use these commands, NRPE must be installed on the client system using the NEMS Linux installation procedure found here: [NRPE For Linux](#)

I suggest you always put NRPE in the service titles you create in order to prevent accidentally assigning a local service to a host who uses NRPE. So instead of calling the advanced service “Check Disk Space /” I’d call it “Check Disk Space / via NRPE”.

### Check CPU Temperature of Remote System

- **Description:** Detect temperature of remote system CPU.
- **Client Requirements:** Must have `lm-sensors` installed and working.
- **\$ARG1\$ Syntax:** `check_cpu_temp -a “WARN CRIT”`

#### \$ARG1\$ Examples:

Warn if CPU is hotter than 40°C and Critical if over 50°C:

```
check_cpu_temp -a “40 50”
```

Warn if CPU is hotter than 35°C and Critical if over 47°C:

```
check_cpu_temp -a “35 47”
```

### Check Disk Usage of Remote System

- **Description:** Determine disk usage by percentage.
- **Client Requirements:** None.
- **\$ARG1\$ Syntax:** `check_disk -a “-w WARN -c CRIT -p PATH”`

#### \$ARG1\$ Examples:

Warn if the disk at `/` is 80% full and Critical if over 90% full and report in gigabytes:

```
check_disk -a '-w 80 -c 90 -p / -u GB'
```

Warn if the disk mounted on `/var` is 80% full and Critical if over 90% full and report in gigabytes:

```
check_disk -a '-w 80 -c 90 -p /var -u GB'
```

Warn if the disk mounted on `/mnt/backup` is 50% full and Critical if over 70% full and report in gigabytes:

```
check_disk -a '-w 50 -c 70 -p /mnt/backup -u GB'
```

### 3.3.18 Check Command: check\_ping

#### Service Parameters

check\_ping requires two parameters: Warn Critical

Each is a group of one integer (response time) followed by a comma and a percentage (packet loss).

In this example, we'll warn when the round trip average is 10ms or higher, or the packet loss is 2% or more. We'll go critical if the round trip average is 20ms or higher, or the packet loss is 5% or more.

```
10,2% 20,5%
```

### 3.3.19 Check Proxmox Virtual Environment (check\_pve)

Monitor your Proxmox VE nodes with individual service checks for hypervisor CPU load, root filesystem usage on the PVE node, Proxmox VE version, memory usage and swap usage).

Service 🟡 Proxmox on host 🟢 Proxmox Hypervisor

🔄

✅

🕒

✉

📄

✎

🗑

More ▾

General

Information

History

Custom Variables

Debug

Status🟡 warning for 3 minutes (not acknowledged)

Host Name🟢 Proxmox Hypervisor

Host Address10.0.0.14

Host AliasProxmox Hypervisor

Service NameProxmox

Check output

WARNING: Your Proxmox version (8.0.3) is not the latest (8.1)

Performance Metrics

	label	value	warn	crit	unit	min	max
🟢	version	8.0.3					
🟢	latest	8.1					

### Usage

---

**Note:** *check\_pve* requires NEMS Linux 1.7 or higher.

---

### Check Command Arguments

- **ip:** IP address of the Proxmox server [Required]
- **port:** Port number Proxmox is accessible on (default: 8006)
- **node:** The name of the node you wish to check [Required]
- **username:** Username of user with PVEAuditor permission set [Required]
- **password:** Password for that user [Required]
- **realm:** Authentication realm (Eg., pve or pam) [Required]
- **check:** Specify the check to perform (load, rootfs, version, memory, swap) [Required]
- **warn:** Warning threshold [int] percentage (default: 80)
- **crit:** Critical threshold [int] percentage (default: 95)

### Configuration

- On Proxmox VE - Create a new user for NEMS Linux to use for the API. - Assign this new user **PVEAudit** permissions.
- In NEMS NConf: - Create a host entry for your Proxmox VE server. - Add the *check\_pve* service to that host, setting the arguments appropriately for your environment. - Generate your NEMS config.

**Warning:** Never use your Proxmox VE root user or any user with more than **PVEAudit** permissions for monitoring.

### CLI Example

```
./check_pve ip=10.0.0.5 port=8006 node=myserver username=auditor password=Str0ngP4ssw0rd_↵  
↵realm=pve check=load warn=80 crit=95
```

### Output

#### Performance Data

*check\_pve* outputs PerfData for further inspection, historical record or aggregation.

All *check\_pve* checks log their regular response (E.g., *rootfs* provides PerfData for the percentage of root filesystem usage).

The following check provide additional Performance Data:

load

- Current CPU load in percentage
- The low and high CPU load in percentage over the last 15 minutes
- Number of available threads
- Load average (1m, 5m, 15m)

#### version

- Currently running version of Proxmox
- Latest version of Proxmox available

#### swap

- Swap usage in percentage
- Swap space available
- Swap space usage

### Cache

*check\_pve* connects to your Proxmox VE server to generate a ticket to access the API with. This ticket is cached on your NEMS Server and re-used for the lifetime of the ticket. Tickets are automatically invalidated by PVE every 2 hours, so NEMS will automatically refresh your cached ticket every 90 minutes. Make sure you are careful to input the correct PVEAuditor username and password on every instance of *check\_pve* in your environment, otherwise you may experience odd authentication issues when the cache is recreated by an incorrectly-entered username/password combination.

We also cache the Proxmox version API JSON response to your NEMS Server for 6 hours, ensuring your server always knows the latest version, but without overtaxing the API.

Caches are stored in */tmp/pve\_\*.cache* which means if a miscreant obtained physical access to your NEMS Server they have up to 2 hours before your PVE API ticket expires. It is therefore imperative that you never use your *root* user or any user who has more than PVEAudit permissions to monitor a PVE server.

### 3.3.20 Check Command: *check\_qnap*

- *status* checks if QNAP is online
- *diskused* disk usage in percentage
- *cpu* system CPU load in percent
- *cputemp* system CPU temperature in C
- *freeram* free RAM
- *temp* system temperature in C
- *hdtemp* Harddrive temperature in C (all hard drives)
- *hdNtemp* Nth harddrive temperature in C (ex. *hd1temp*)
- *volstatus* Status of all volumes
- *volNstatus* Nth volume status
- *powerstatus* Power status
- *fans* Fans speed (RPM)

- `systemuptime` System uptime
- `sysinfo` System info display (model, number of hard drives, volumes, name and firmware)

Source: `check_qnap3.sh` file ([https://github.com/cloudcentral/nagios-plugins-check\\_qnap/blob/master/check\\_qnap3.sh](https://github.com/cloudcentral/nagios-plugins-check_qnap/blob/master/check_qnap3.sh))

### 3.3.21 Check Commands: `check_synology`

Synology NAS monitoring is included in NEMS Linux 1.6 or later.



#### Description

This check uses SNMP to check the state of a Synology NAS.

You will see any problems displayed on your NEMS Tactical Overview, NEMS Mobile UI, NEMS TV Dashboard, NEMS Adagios, or whichever tool you prefer for monitoring your network state with NEMS Linux. You'll receive an email, SMS, Telegram, Webhook or other notification (whichever you have configured on your NEMS Server) if a drive fails. Your NEMS Warning Light will even turn red if there is a problem with your Synology NAS.

Assigning your Synology NAS host to the Synology Host Group in NEMS Configurator adds the following sample Advanced Services:

- Synology Disks
- Synology Fans
- Synology Power
- Synology RAID
- Synology System
- Synology UPS
- Synology Version

By default, the *admins* contact group will be notified.

The sample Advanced Services provided in NEMS Linux assume the community name to use is *public*.

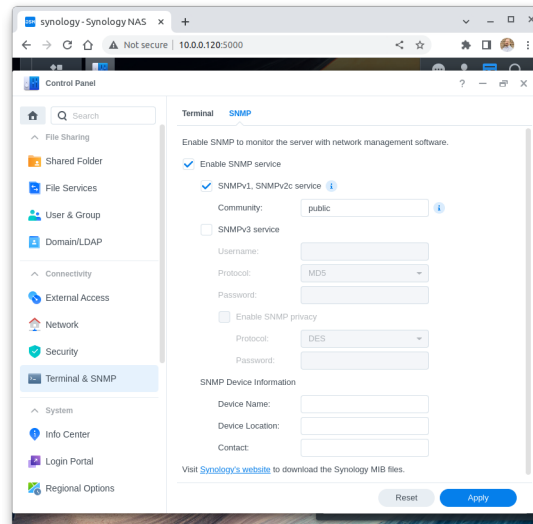
#### Basic Usage

##### In Synology DSM

- Visit Control Panel -> Terminal & SNMP -> SNMP
- Check *Enable SNMP Service*
- Check *SNMPv1, SNMPv2c service*
- Ensure your Community name is set to *public*
- You do not need to enable *SNMPv3 service* nor *SNMP Privacy* since we are using the *public* community.



- Press **Apply**



## In NEMS Configurator

- On the left menu, click **Add** next to **Hosts**
- Enter a friendly hostname for you to recognize the device. *Synology* or *Synology NAS* are good examples.
- Under Address enter the local IP address of your Synology device.
- Leave OS set to Linux.
- Set monitored by to **Default Nagios**
- Set max check attempts to a reasonable number of attempts, such as 10.
- Set assign host to hostgroup to **Synology** (Highlight it and press the single green arrow to add it to the right column).
- Click **submit**.

This next step is currently required the first time you setup your Synology NAS. In a future release of NEMS Linux, we will include new defaults to remove this requirement, but for now, you must set your defaults:

- Click **Show** next to **Advanced Services**.
- Click the edit pencil icon next to **Synology Disks**.
- Set max check attempts to a reasonable number of attempts, such as 10.
- Press **Submit**.
- Repeat these steps for each of the Synology Advanced Services until all have a default **max check attempts**

At this point, you may click **Generate NEMS Config** to activate your new Synology checks.

## Reporting



Fig. 2: NEMS Adagios displaying an overview of all Synology Advanced Services in the OK state.

Congratulations! NEMS Linux is now proactively monitoring your Synology NAS device.

## Compatibility

Tested with the following versions of Synology DiskStation Manager:

- DSM 6.2.2-24922 Update 4
- DSM 7.0-41890
- DSM 7.1.1-42962 Update 4

This isn't to say you must run one of these versions. Rather, these are simply the versions we have tested with successfully. It can be assumed that the Synology checks built-in to NEMS Linux will also work with other point releases of DSM.

## Source

`check_synology` uses `check_snmp_synology`.

### 3.3.22 TEMPer Thermal Sensor

The TEMPer is an affordable (starting at under \$20) USB digital thermometer that accurately senses temperatures from -55 to +125 degrees Celsius / -67 to +257 degrees Fahrenheit. This is an ideal addition to the server room to generate alerts should temperatures fall outside a safe threshold.

TEMPer devices work on all NEMS Linux hardware platforms. It will also work on the NEMS Linux virtual appliance, however you must connect the hardware to the virtual machine, which is beyond the scope of this documentation (please refer to the documentation for your hypervisor).

`check_temper` supports Critical and Warning states for both low (cold) and high (hot) temperatures.



Fig. 3: TEMPer USB Thermal Sensor

**Note:** For optimum accuracy, it is recommended to plug your TEMPer device into a short USB extension cord so the thermal data isn't impacted by the heat pulled from the USB port of your NEMS Server.

## Setting up check\_temper

Simply add `check_temper_temp` or `check_temper_hum` as a service to your NEMS host, having connected the TEMPer device to your NEMS Server's USB port. You may specify your temperature thresholds in either degrees Celsius or Fahrenheit. NEMS will automatically determine which you are using.

Sample check commands are already included in NEMS Linux 1.5.2+ in the demo config. If you're deploying a new NEMS Server, simply plug in your TEMPer device and these checks will automatically detect it and begin registering data.

## Buy TEMPer Thermal Sensor

- [Amazon.com](#)
- [Amazon.co.uk](#)
- [From the Manufacturer](#)

## Supported Devices

Support was originally provided by [urwen](#). In NEMS Linux 1.6, support was initially moved to the more up-to-date repository from [ccwienk](#) (adds several of the newer TEMPer devices) but then forked to [NEMSLinux](#) to add support for TEMPerGold\_V3.3 firmware (and others in the future).

NEMS Linux includes support for TEMPer temperature and humidity sensor data. This table also shows which have internal or external sensors. NEMS Linux will always opt for external sensor data, if present.

Table 1: List of Supported TEMPer Devices

Product	ID	Firmware	Temp	Hum	Int	Ext
TEMPer	0c45:7401	TEMPerF1.4	✓		✓	
TEMPer	413d:2107	TEMPerGold_V3.1	✓		✓	
TEMPer	1a86:e025	TEMPerGold_V3.3	✓		✓	
TEMPer	1a86:e025	TEMPerGold_V3.4	✓		✓	
TEMPerHUM	413d:2107	TEMPerX_V3.1	✓	✓	✓	
TEMPerHUM	1a86:e025	TEMPerHUM_3.9	✓	✓	✓	
TEMPer2	413d:2107	TEMPerX_V3.3	✓		✓	✓
TEMPer2	413d:2107	TEMPer2_V3.7	✓		✓	✓
TEMPer2	1a86:e025	TEMPer2_V3.9	✓		✓	✓
TEMPer2	1a86:e025	TEMPer2_M12_V1.3	✓		✓	✓
TEMPer1F	413d:2107	TEMPerX_V3.3	✓			✓
TEMPerX232	1a86:5523	TEMPerX232_V2.0	✓	✓	✓	✓
TEMPer1V1.1	0c45:7401	TEMPer1F1.1Per1F	✓	✓		✓

**Temp:** Device contains thermal sensor. **Hum:** Device contains humidity sensor. **Int:** Device uses an internal sensor built into the device. **Ext:** Device supports use of an external sensor probe.

## Terminal Output

To receive the JSON output, type: *nems-info temper*

Sample output from a TEMPer2 with an external sensor attached:

```
{"0":{"vendorid":16701,"productid":8455,"devices":["hidraw0","hidraw1"],"firmware":  
→"TEMPerX_V3.3","internal temperature":30.12,"external temperature":21.68},"sensors":{  
→"thermal":1,"temp_location":"external","humidity":0,"hum_location":"not_present"},  
→"output":{"temperature":21.68,"humidity":0}}}
```

To receive the JSON output, type *nems-info temper*

TEMPer devices seem to have an issue on low-powered systems (such as Raspberry Pi) where due to the low power to the USB port, temper.py will respond with errors such as:

- Cannot read firmware identifier from device
- Unknown firmware ld\_V3.1 TEMPerGold\_V3.1: b'80800f874e200000'

To remedy this, *nems-info* silently loops through the output multiple times until a good thermal reading is obtained. The errors are hidden and only the clean JSON output is generated. This all happens very quickly and transparently, resulting in good output every time, with no errors.

## Adding to your NEMS NConf

TEMPer check commands are already pre-configured in the NEMS Linux 1.5.2+ demo data. If you have removed them, or are using an older version of NEMS Linux, you can add the checks yourself.

*check\_temper\_temp* (temperature check) and *check\_temper\_hum* (humidity check) allow you to add TEMPer devices to your NEMS Server. The check has 4 thresholds: Critical Low, Warning Low, Warning High, Critical High. The number you enter may be in *either* degrees Celsius or Fahrenheit in the case of *check\_temper\_temp*. The system will automatically detect which you are using. The OK temperature will be any temperature that falls between Warning Low and Warning High. This way, you can receive alerts from your NEMS Server should the room temperature be either too cold or too hot. For *check\_temper\_hum*, the thresholds are in percent (just enter the number, not the percent sign).

## Humidity Sensor Support

Sample command line for humidity sensor:

```
/usr/lib/nagios/plugins/check_temper 20 35 65 80 hum
```

## Check Commands

As of NEMS Linux 1.5.2, both the temperature and humidity sensors are supported, and check commands are included in NEMS NConf.

- *check\_temper\_temp*
- *check\_temper\_hum*

## Output

Temperature and relative humidity data are checked against both low and high thresholds. If the temperature or relative humidity are either too low or too high, the state of the check command will be affected. The checks are only in an OK state when the current sensor reading is neither too low, nor too high.

NEMS Linux includes perfd data output: `Temperature[low]`, `Temperature[high]`, `Humidity[low]`, `Humidity[high]` depending which check is performed. Since perfd data aggregation tools do not generally employ a direct means of understanding that a low threshold exists in addition to a high threshold, please be mindful that extra care may be required when designing your perfd data assimilation. E.g., a low warning temperature of 15°C will show critical if the room temperature is 20°C since perfd data does not usually work in the negative direction. NEMS Linux itself is not impacted by this as the state of the check command is correctly reported, however third-party tools which aggregate the perfd data will need extra care.

## Calibration

As of NEMS Linux 1.6, both the thermal sensor and humidity sensor can be calibrated within NEMS SST to ensure the highest level of accuracy.

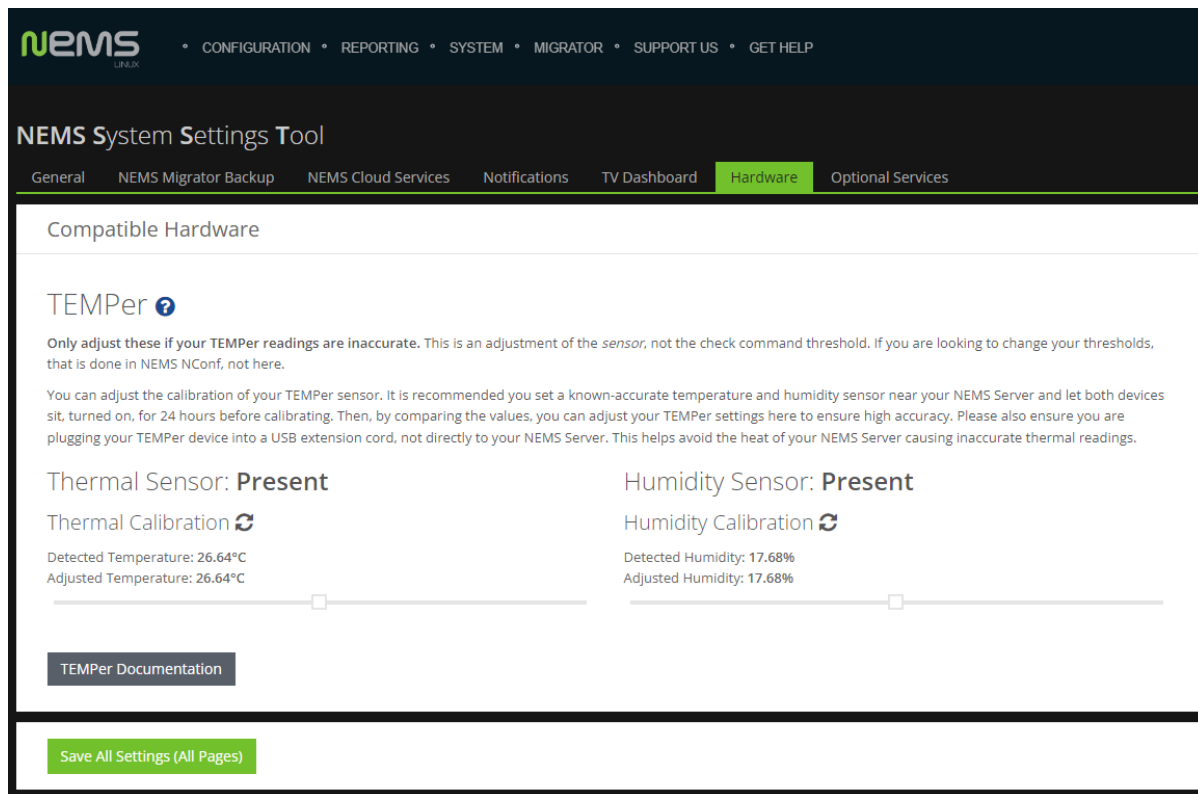


Fig. 4: TEMPer Sensor Calibration in NEMS SST

## External vs. Internal Sensors

If your TEMPer device supports an external sensor, this will be used if connected. If the external sensor is disconnected, the internal sensor will be selected automatically.

### 3.3.23 Check Command: `check_sbc_temperature`

**Requires:** NEMS 1.5+

`check_rpi_temperature` has been renamed `check_sbc_temperature` because it is not at all Raspberry Pi specific: Any system that has sysfs logged to `/sys/class/thermal/thermal_zone0/temp` will work.

Also tested on the NEMS Linux ODROID XU4 build.

This check command requires two arguments: Warning Temperature °C and Critical Temperature °C.

#### Default Thresholds

NEMS Linux comes with `check_sbc_temperature` pre-configured on the NEMS localhost. Here are the default thresholds:

- **ODROID XU4** 76°C Warn / 82°C Crit

### 3.3.24 Check Command: `check_tcp`

#### Service Parameters

NEMS Linux requires just one argument for the `check_tcp` check command: the port number to check.

#### Quick Use

1. Open NEMS NConf.
2. Click Services→Add.
3. Give it a friendly name.
4. Choose the Check Command “`check_tcp`”.
5. Set the notification period to 24/7.
6. Set your intervals and notification settings as normal.
7. Set ARG1 to the port number. Eg., 8080
8. Hit submit.

Then go [generate your config](#) and if you set your intervals correctly, all should be a-okay. If not, expand the error message to see where you went wrong.

## Exercise

NEMS Linux includes a dummy port listener running on port 9590. The port listener is cleverly called **9590**, and does nothing other than reply that it is up. This can be used to simulate a port on another device.

Let's setup a service monitor on the NEMS host to warn us if port 9590 ever goes offline.

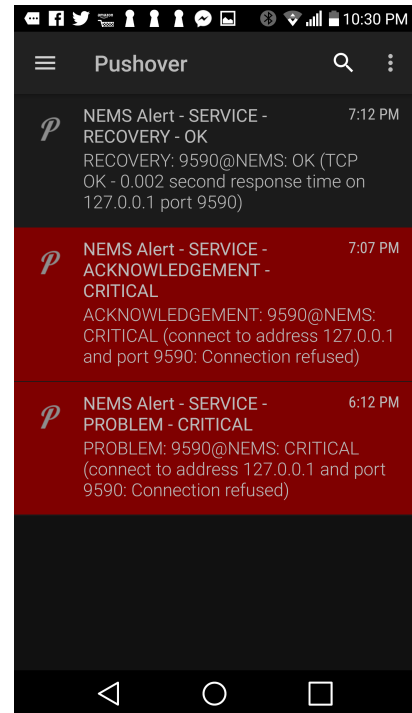
1. On the left menu of NConf, you'll see "Services". Click "Add".
2. Set the *Service Name* to: 9590
3. Leave *Service Enabled* set to: Yes
4. Set the *Check Command* to: check\_tcp
5. Set *Assigned to Host* to: NEMS (this host comes pre-installed)
6. Leave *Check Period* set to: 24×7
7. Set *Notification Period* to: 24/7
8. Leave *Service Templates* as is, none selected.
9. Under *Contact Groups* highlight the 'admins' group and press the arrow pointed right to move it to *Selected Items*.
10. Leave *Notes*, *Notes* and *Action* blank.
11. Set *Max Check Attempts* to: 5
12. Set *Check Interval* to: 30
13. Set *Retry Interval* to: 5
14. Set *First Notification Delay* to: 60
15. Set *Notification Interval* to: 90
16. Set *Notification Options* to: w,u,c,r,f,s
17. Leave *Active Checking*, *Passive Checking*, *Notification Enabled*, *Check Freshness* and *Freshness Threshold* blank.
18. Leave *Assign Service to servicegroup* as is, none selected.
19. Set *Params for check command* to the port number: 9590
20. Press *Submit*
21. [Generate Nagios Config](#)

Once the new config is running, try failing the service by opening [Monit Service Manager](#), click on the Process named **9590**, and then click "Stop service". You'll notice within 30 minutes the status of 9590 will show as a problem in all status views (Eg., NEMS TV Dashboard, NEMS Adagios, Nagios Core), and after 60 minutes you will receive a notification (assuming your notifications settings are configured).

Once you have received a notification, visit NEMS Adagios to Acknowledge the outage.

Then, return to Monit, open the 9590 Process, and click "Enable Monitoring". This will re-load 9590 and you'll soon see it change to a *Recovered* state.

Another fun experiment is to try changing your "First Notification Delay" on the NEMS:9590 service in NEMS NConf and disable it again.



### 3.3.25 Check Command: check\_win\_users

Check the count of users on a Windows server based on a query.

#### Valid Query Options

- AccountType
- Caption
- Description
- Disabled
- Domain
- FullName
- InstallDate
- LocalAccount
- Lockout
- Name
- PasswordChangeable
- PasswordExpires
- PasswordRequired
- SID
- SIDType
- Status

#### Example Queries

- List all users:

```
status like '%'
```

- Show users where the Status is OK:

```
status='OK'
```

- Show usernames that contains the string “EX”:

```
name like '%EX%'
```

- Show usernames that start with the string “Admin”:

```
name like 'Admin%'
```

- Show usernames that end with the string “test”:

```
name like '%test'
```

- Show users whose full name contains the String “Exchange”:



```
fullname like '%Exchange%'
```

- Show users that belong to the Domain called “TEST”:

```
Domain='TEST'
```

- Show users where a password is not required:

```
PasswordRequired!='True'
```

- Show users that belong to the Domain called “TEST” and that where a password is not required:

```
PasswordRequired!='True' AND Domain='TEST'
```

- Show users that do not belong to the Domain called “TEST”:

```
Domain!='TEST'
```

- Show users that have local accounts:

```
LocalAccount='True'
```

### 3.3.26 Monitor Windows Machines with Windows Management Instrumentation (WMI)

#### Introduction to WMI

WMI is a set of specifications included in Microsoft Windows which provides NEMS Linux with information about the status of Windows-based hosts.

Disabled by default, WMI requires some configuration before NEMS Linux can monitor the Windows host.

#### Windows Versions Tested and Verified to Work With NEMS 1.6 and WMI (3/17/2023)

- Windows 10 Pro Version 22H2
- Windows 11 Pro Version 22H2
- Windows Server 2016 Version 1607
- Windows Server 2019 Version 1809
- Windows Server 2022 Version 21H2

#### Setting up WMI

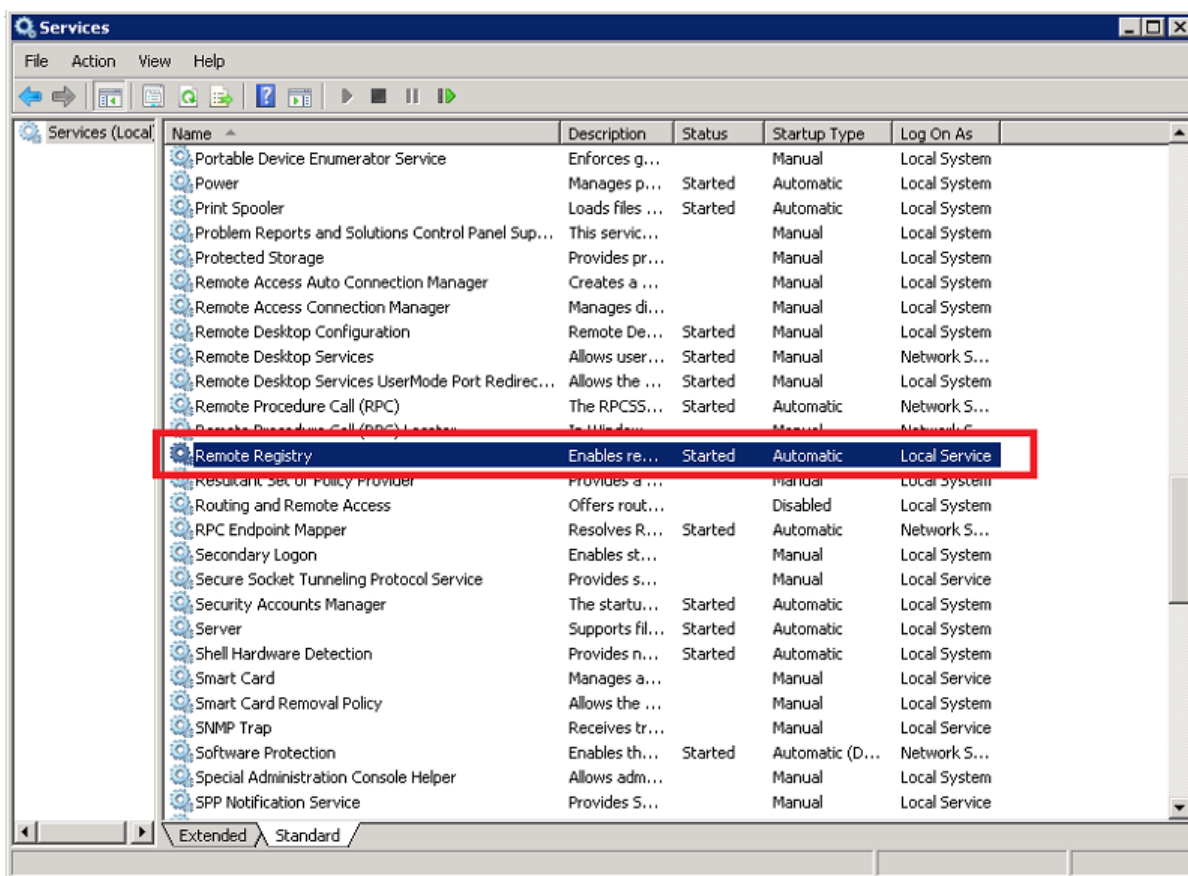
##### Configure WMI on the Windows end

In order for the agent to have access to query WMI to collect and database metrics, the agent must have permission to access both DCOM and WMI.

Verify that the Remote Registry, Server, and the Windows Management Instrumentation services are running.

- Click **Start**.
- Click **Run**.
- Type *services.msc* and press “Enter”. This will open the Services dialogue.

- Scroll down to the *Remote Registry service*. Verify that the service is started and is set to **Automatic**.



By default even if the Remote Registry service is started Windows 7 and later systems will deny remote access to the registry.

- Scroll down to the *Server* service. Verify that the service is started and set to **Automatic**.
- Scroll down to the *Windows Management Instrumentation* service. Verify that it too is started and set to **Automatic**.

---

**Note:** The best practice is to use a Local account on the monitored host as the agent user.

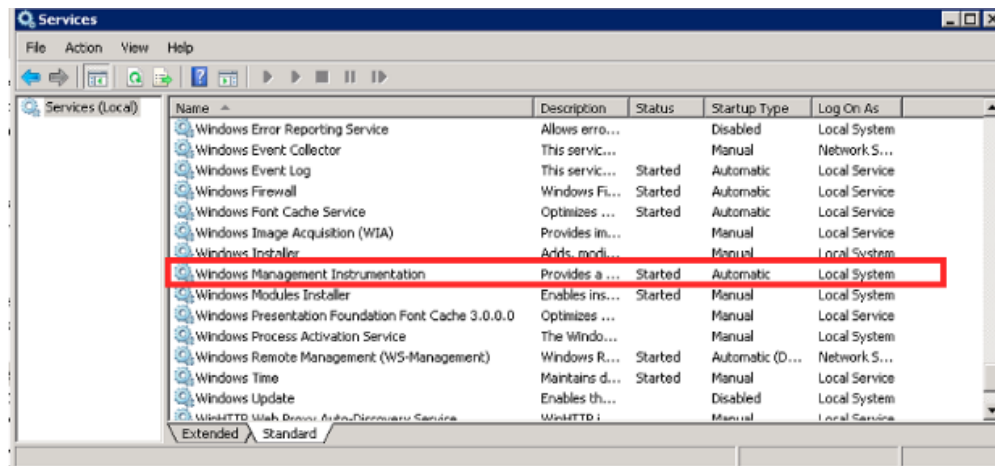
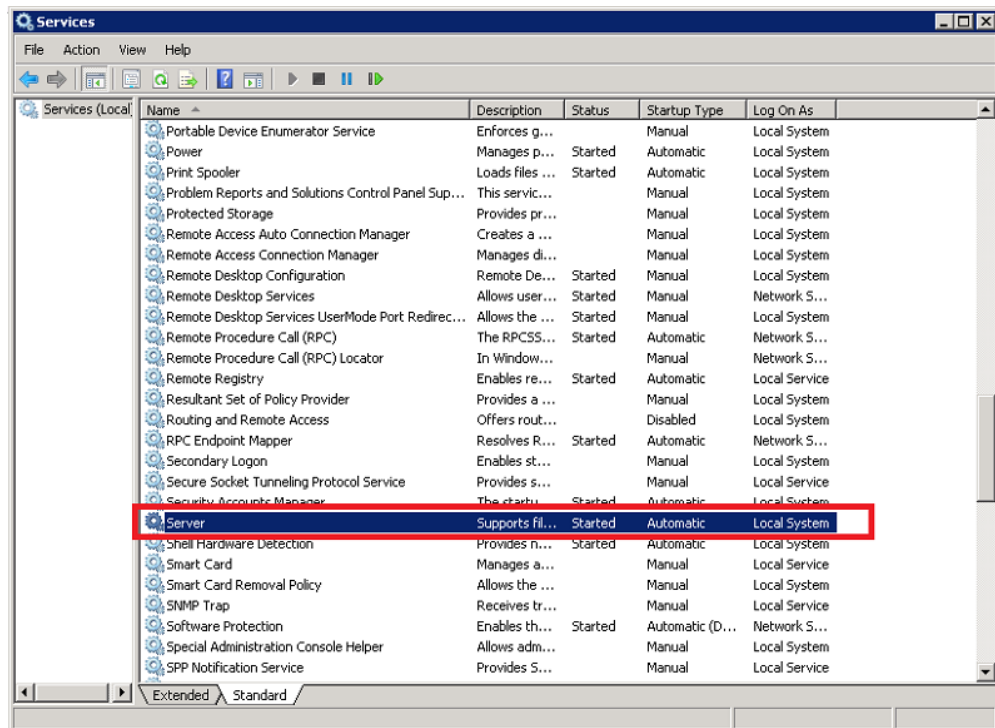
---

Where this is not possible, use these procedures to grant permissions for a remote user.

- All windows workstations must have a user with the same local user name and password.
- Local user account on the target computer must have explicit DCOM and WMI namespace access rights granted specifically for remote connections.
- User must also be a member of the *Performance Log Users* group
- Local security policies must be enabled for *Classic - local users authenticate as themselves*

#### Grant minimal WMI permissions to the remote user

This limits users other than those configured from remotely accessing WMI.

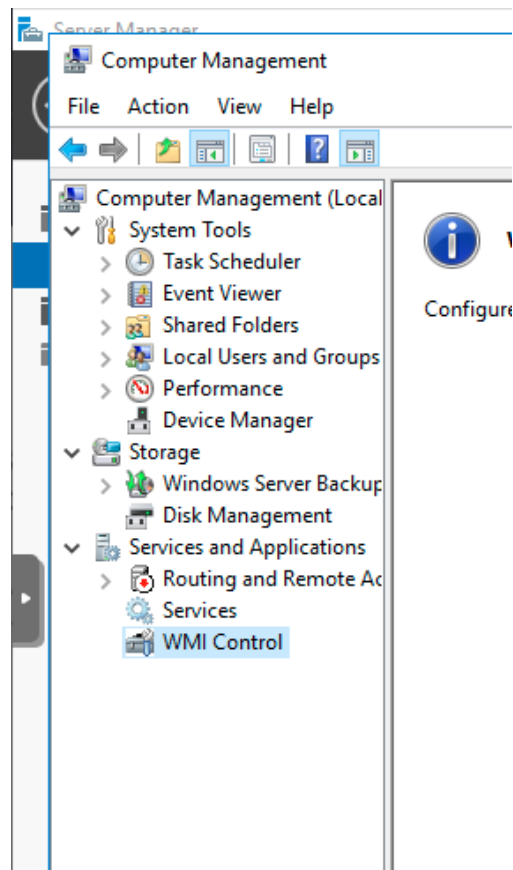


**Note:** In the following example, replace “remoteuser” with the username of the user created on your Windows hosts.

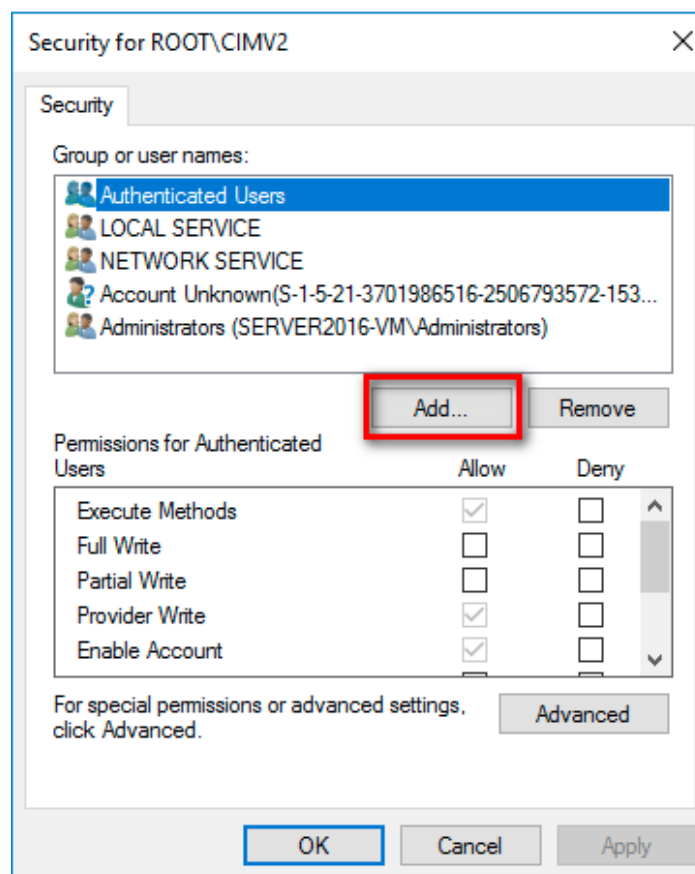
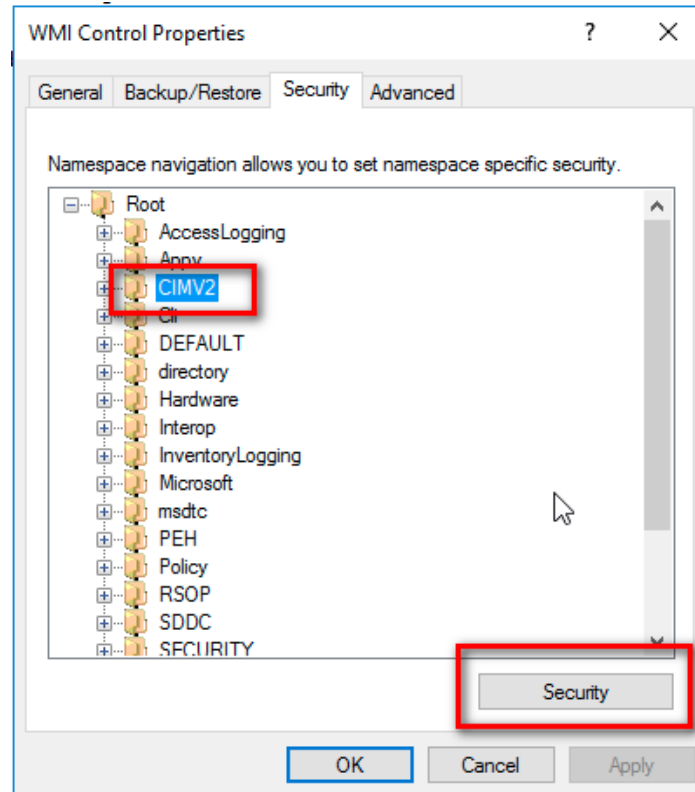
---

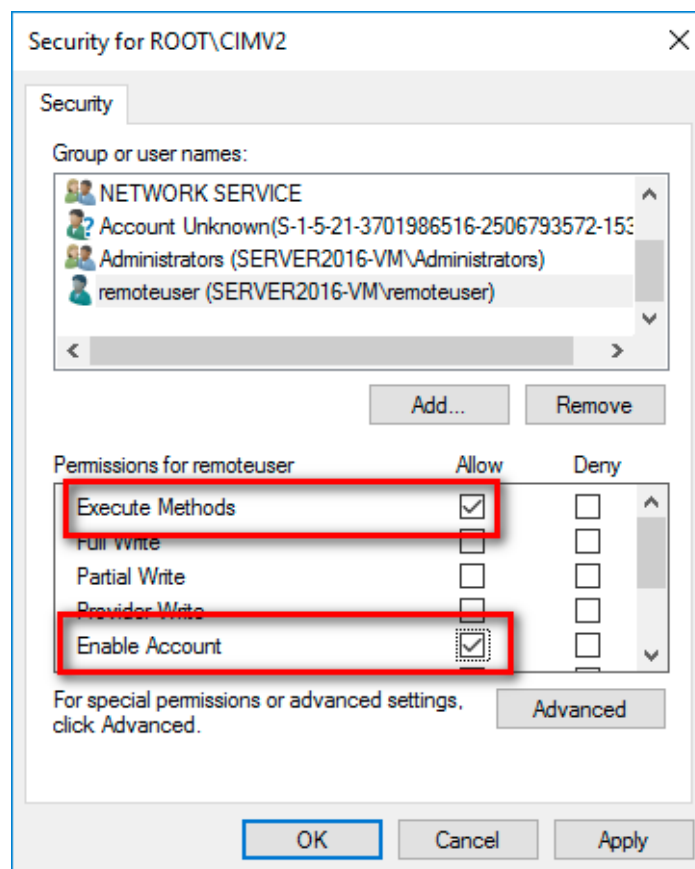
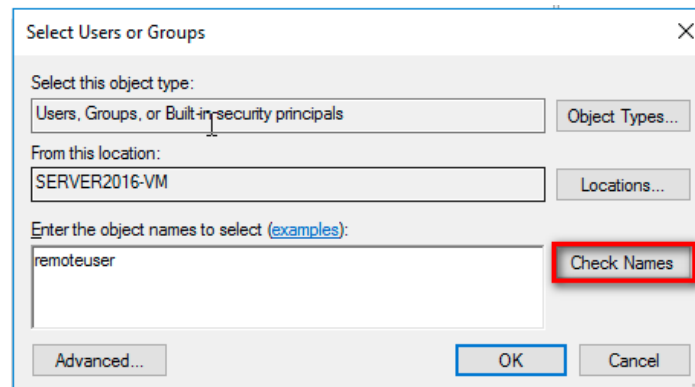
On the monitored host machine

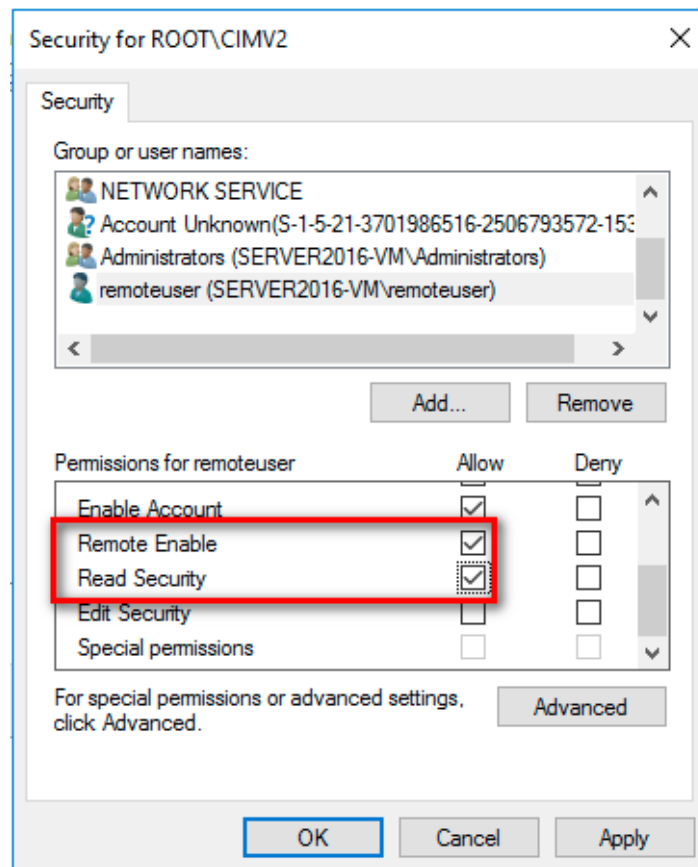
- Right-click on **This PC**
- Click **Manage**
- Navigate to *Services and Applications* → *WMI Control*



- Right-click **WMI Control** and click **Properties**
- In the WMI Control Properties dialog box, click the **Security** tab.
- Expand the Root node and select **CIMV2**, then click **Security**
- Select the user in the **Group or user names** box. If not listed select **Add**.
- Type in the user name and click **Check Names**
- Click **OK**
- Grant the required permissions to the remote user by enabling the following check boxes in the Allow column:
  1. Execute Methods
  2. Enable Account
  3. Remote Enable
  4. Read Security

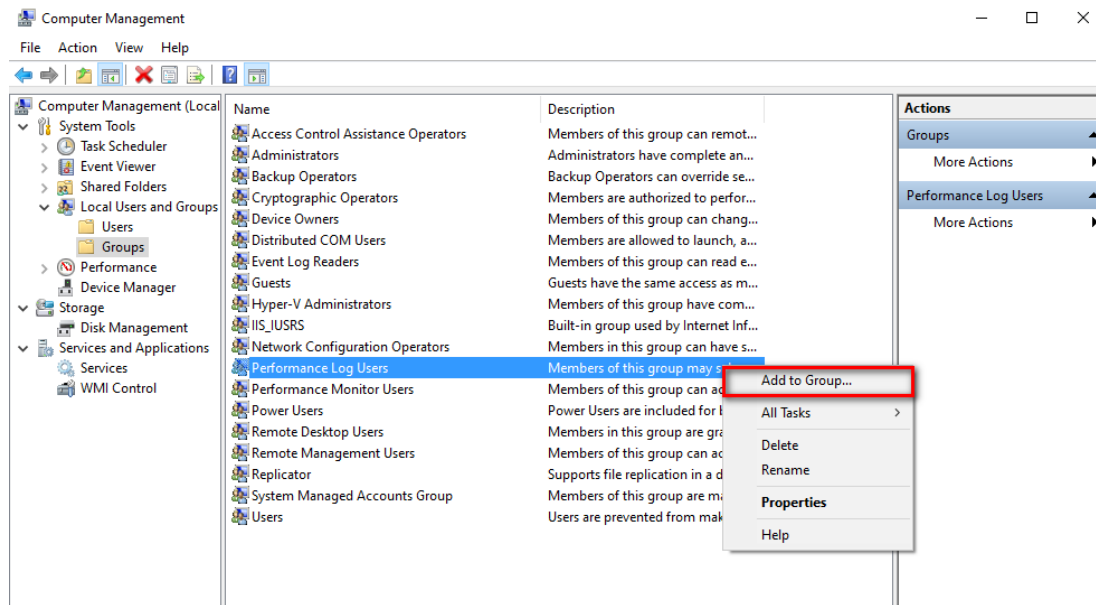






While still in Computer Management

- Expand **System Tools**
- Expand **Local Users and Groups**
- Click **Groups**
- Right click **Performance Log Users**
- Click **Add to Group**

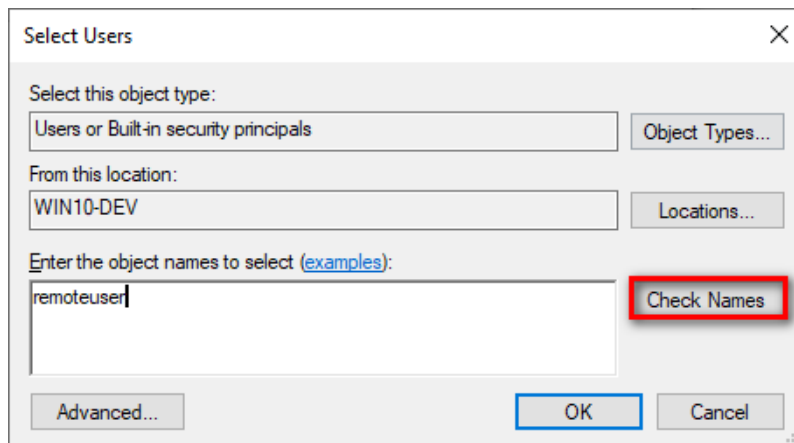
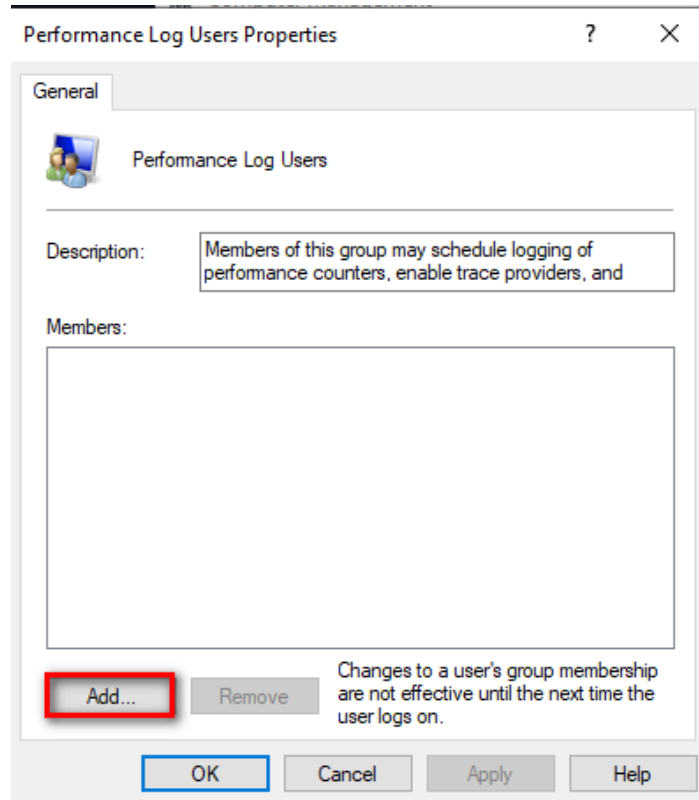


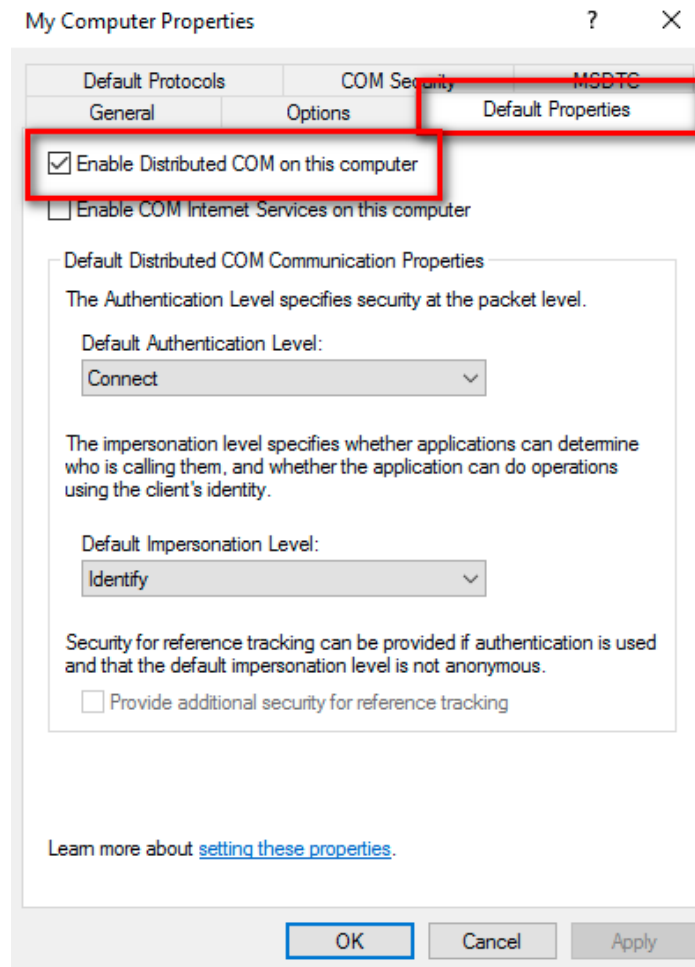
- Click **Add**
- Type in the username and click **Check Names**
- Click **OK**
- Click **Apply**
- Click **OK** to close dialog box
- Close **Computer Management** window

#### To grant DCOM permissions to a remote user

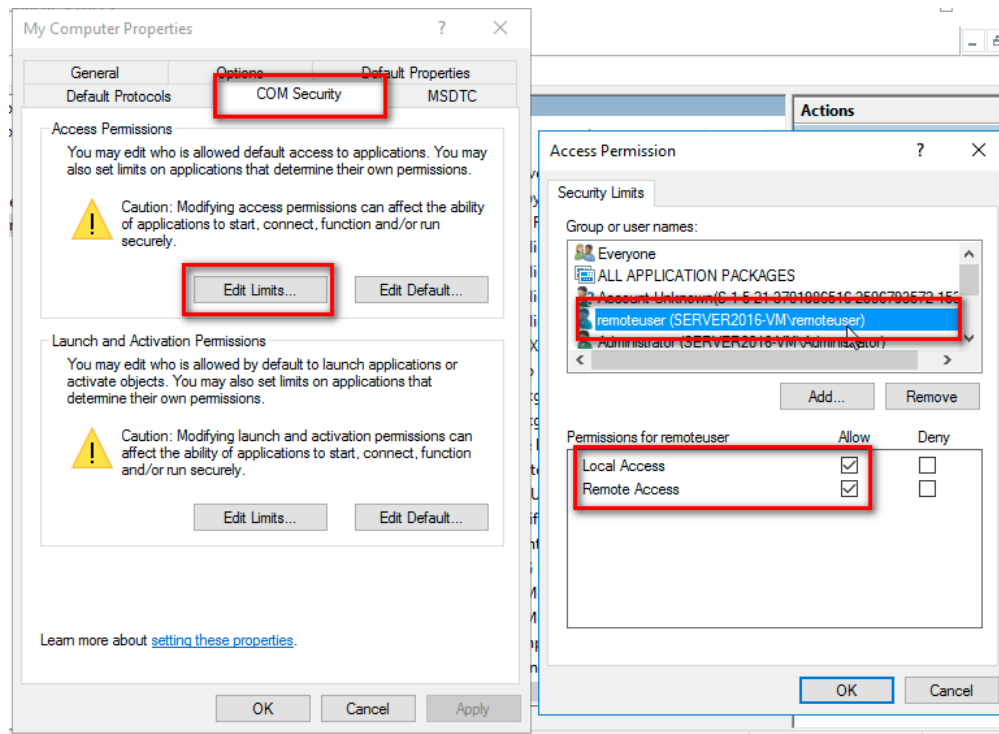
- On the monitored host machine, at the Windows Run prompt, type *DCOMCNFG* and press Enter.
- In the Component Services dialog box that opens, navigate to *Component Services* → *Computers* → *My Computer*.
- Right-click **My Computer** and click **Properties**.
- Select the **Default Properties** tab.
- To enable DCOM, select the **Enable Distributed COM on this computer** checkbox.
- Click **Apply**.
- In the *My Computer* Properties dialog box, click the **COM Security** tab.
- Under Access Permissions, click **Edit Limits**.
- In the Access Permission dialog box, add the user or group name necessary for the remote user.







- Ensure Local Access and Remote Access are checked and click **OK**



- In the Launch and Activation Permissions area, click **Edit Limits**.
- In the Launch and Activation Permission dialog box, add the user or group name necessary for the remote user.
- Grant the remote user all the permissions available in the Permissions for Administrators area by enabling all of the check boxes in the Allow column.
- Click **OK** and/or **Yes** to close the dialog boxes.

#### Enable Classic Security policies for Windows Systems that are not part of a domain.

- Open the Control panel, and go to *Administrative Tools* → *Local Security Policy*.
- The Local Security Settings window appears.
- Go to *Local Policies* → *Security Options*.
- Change the value of *Network access: Sharing and security model for local accounts*. to **Classic**.

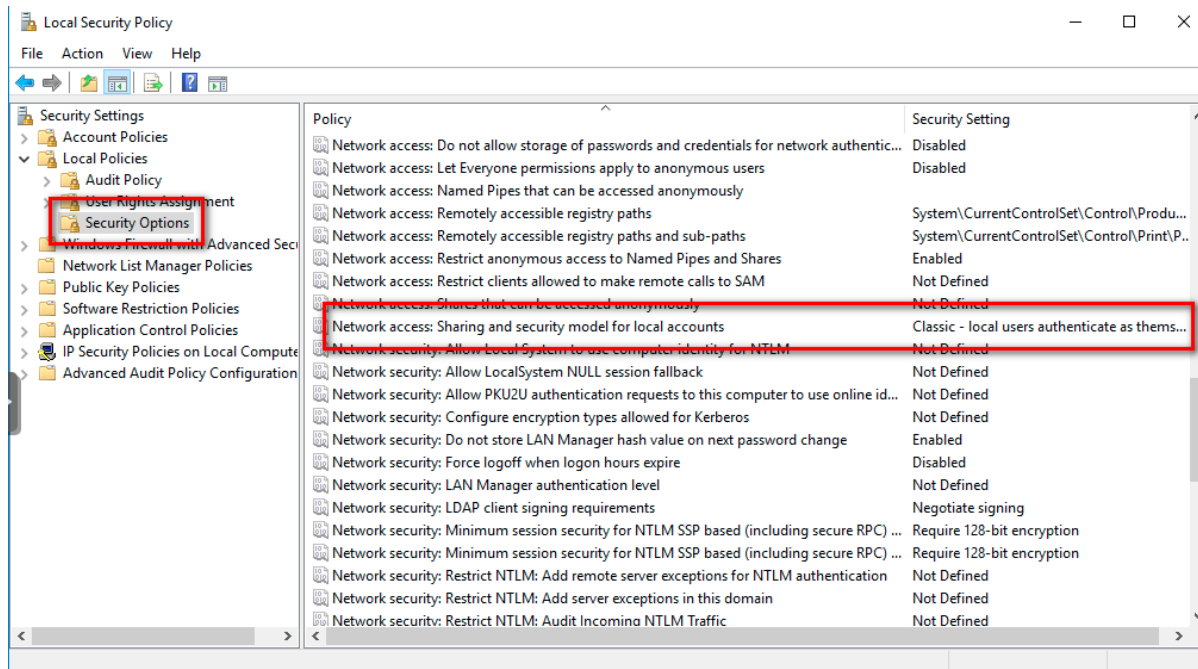
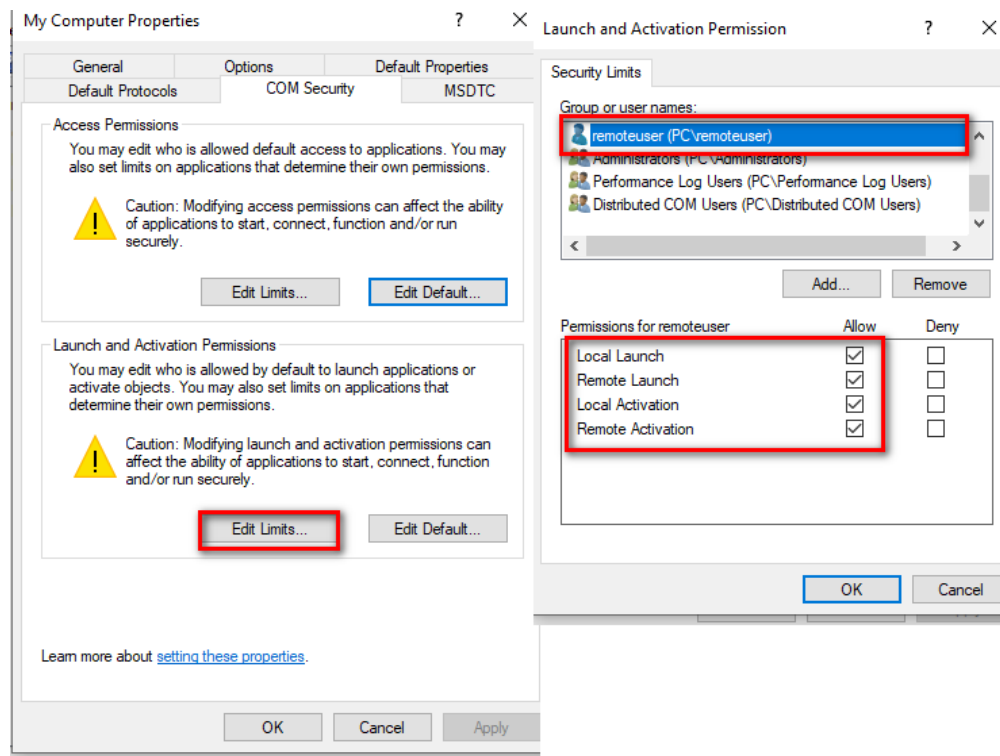
#### Open the Windows firewall for WMI traffic

Enter the following in an Administrator Command Prompt:

```
netsh advfirewall firewall set rule group="windows management
instrumentation (wmi)" new enable=yes
```

#### Add Your Windows User to NEMS SST

Enter the username and password of the user created on the Windows devices who was granted access to the WMI data.



NEMS System Settings Tool

NEMS • CONFIGURATION • REPORTING • SYSTEM • MIGRATOR • SUPPORT US • GET HELP

### This NEMS Server

NEMS Server Alias

NEMS Update

Background Image

Blur Background

#### Windows Domain Access

Administrator domain/username. If not on a domain, use username only.

Administrator Password

#### IPMI Credentials

IPMI Username

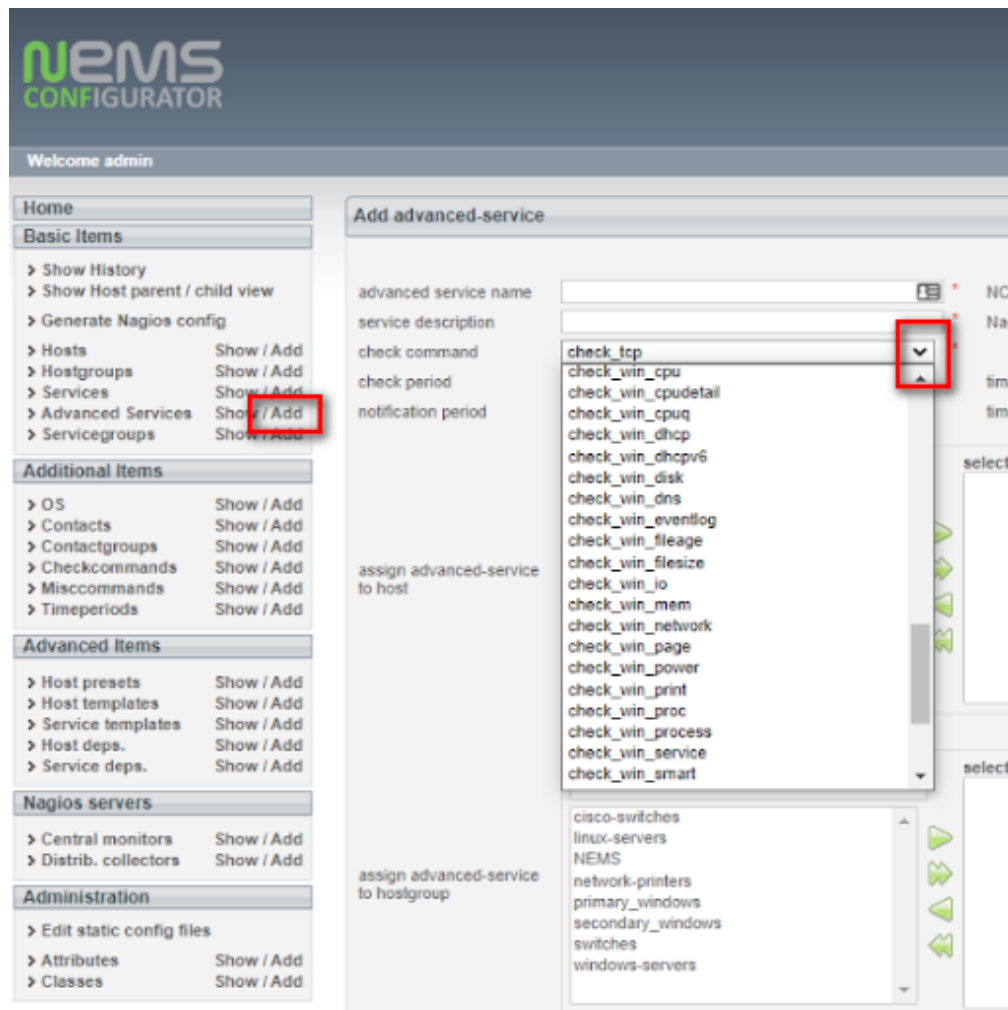
IPMI Password

Save All Settings (All Pages)

## Basic Configuration of Windows Devices In NEMS Linux Using WMI Check Commands

### Adding check\_win\_xxxx Commands in Advanced Services

1. In NEMS NConf click the *Add* button next to *Advanced Services*. Then click the drop-down arrow in the *check command* select list, and scroll down to the *check\_win\_xxx* commands to choose the command you wish to add.



2. Configure the required fields and be sure to assign the Advanced Service to your Windows host. Then click *Submit*. You will see your new command in the list of available Advanced Services.

Repeat Steps 1 and 2 above as needed to add any further *check\_win\_xxx* services you require.

When complete these commands will now be available in the *Advanced Services* list.

Configure these Advanced Services as required to meet your needs and assign them to one or multiple Windows devices.

**NEMS CONFIGURATOR**

Welcome admin

**Home**

**Basic Items**

- › Show History
- › Show Host parent / child view
- › Generate Nagios config
- › Hosts Show / Add
- › Hostgroups Show / Add
- › Services Show / Add
- › Advanced Services Show / Add
- › Servicegroups Show / Add

**Additional Items**

- › OS Show / Add
- › Contacts Show / Add
- › Contactgroups Show / Add
- › Checkcommands Show / Add
- › Misccommands Show / Add
- › Timeperiods Show / Add

**Advanced Items**

- › Host presets Show / Add
- › Host templates Show / Add
- › Service templates Show / Add
- › Host deps. Show / Add
- › Service deps. Show / Add

**Nagios servers**

- › Central monitors Show / Add
- › Distrib. collectors Show / Add

**Administration**

- › Edit static config files
- › Attributes Show / Add
- › Classes Show / Add

**Show: advanced-service**

Searchfilter

**Overview** Entries 1 - 25 of 30 25 50 100 all

advanced service name	[ actions ]
/ Disk Space	
/var Disk Space	
C:\ Drive Space	
Check CPU Temperature	
Check Windows CPU	
Check Windows Disk Usage	
CPU Load	
CPU Temperature	
Current Users	
Current Users_02	
Explorer	
HTTP	
HTTP_02	
Memory Usage	
Memory Usage NRPE	
NEMS Current Load	
NEMS Current Load_02	
NSClient++ Version	

### Special Thanks to Bill Marshall

This documentation would not be possible were it not for the effort of Bill, also known as UltimateBugHunter-NitPicker on our Discord server. Bill setup a test environment, tested, documented, and screen captured the entire setup process and submitted it for inclusion in the official docs. Thanks Bill!

### 3.3.27 Check Command: `custom_check_mem`

NEMS Linux includes the `custom_check_mem` checkcommand, as well as an Advanced Service called *Memory Usage NRPE* to allow checking of remote systems.

It is recommended to add *Memory Usage NRPE* to the *linux-servers* hostgroup.

For WMI, use `check_win_mem` instead.

### 3.3.28 Check Command: `check_tasmota`

Check IoT sensor states for sensors running the Tasmota Firmware, such as SONOFF IoT devices.

#### Included Check Commands

**Note:** I do not have one of these devices for testing in NEMS Linux, so have done my best to understand the command parameters. If you test your device, please report back to me (Robbie) with your feedback as to the accuracy of this documentation, or any changes that are required. If you encounter an issue you cannot resolve, please let me know.

NEMS Linux includes the two sample check commands provided by the plugin author:

- `check_tasmota_power` - Check the power state of a compatible IoT device. Warn/Crit should be a binary threshold: 0 or 1. - `check_tasmota_sensor` - Check the state of any compatible sensor. Must specify the device (E.G., ENERGY or AM2301) and the sensor (E.G., Humidity, Temperature, Voltage, Current, Total, Today, Yesterday).

#### Included Hostgroups

NEMS Linux includes hostgroups for three styles of IoT devices which use the Tasmoto firmware:

- Tasmota **ENERGY** WifiPlugs - Assign to the *tasmota-ENERGY* hostgroup to check the following: Ping, Power State, Voltage, Current, Total Consumption (All Time), Total Consumption (Today Only), Total Consumption (Yesterday Only).
- Tasmota **AM2301** WifiPlugs - Assign to the *tasmota-AM2301* hostgroup to check the following: Ping, Humidity, Temperature
- Tasmota **Switch** WifiPlugs - Assign to the *tasmota-switch* hostgroup to check the following: Ping, Power State



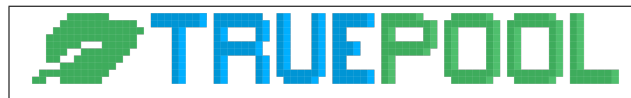
## Requirements

Requires NEMS Linux 1.6+.

## Source

From [https://github.com/dirkheitzmann/nagios-plugin-check\\_tasmota](https://github.com/dirkheitzmann/nagios-plugin-check_tasmota)

### 3.3.29 Check Command: `check_truepool`



Regrettably, TruePool shut down in 2022. However, leaving this here for now since this check command can be adapted and used with other Chia pool servers.

TruePool.io is a trusted Chia cryptocurrency farming (mining) pool. This NEMS check command allows you to check your TruePool.io farmer status to ensure your Chia farm is online and actively farming to the pool.

Learn more about what makes TruePool.io unique in [this video](#).

`check_truepool` expects just one argument (Launcher ID) and responds accordingly. You can create multiple `check_truepool` advanced services to check any number of Launcher IDs. For example, monitor your own and your family member's farm status.

Status	● ok for 3 hours, 17 minutes
Host Name	● NEMS
Host Address	127.0.0.1
Host Alias	Nagios Enterprise Monitoring Server
Service Name	TruePool Farm - Robbie
Check output	OK - Points: 91,020 (24 Hrs) / Share: 9.1250% / Diff: 183

Fig. 5: NEMS Adagios output of `check_truepool` command

This check command requires NEMS Linux 1.6+.

### Expected Responses

- OK - Farm is online and actively gaining points on Truepool.io
- CRITICAL - Either “Not found” (you provided an invalid Launcher ID) or “Farm Offline” (your farm has gained 0 points since the last check)

**Sample Output:** *OK - Points: 830 (24 hrs), 243 (this block) / Share: 0.0810% / Diff: 1*

---

#### [Cache]

If your `check_truepool` response includes `[Cache]` it means you are running the check command too frequently. Check your farm and determine approximately how long it takes you to solve a partial, then set your `check_truepool` to run with a cushion to allow variance. For example, if it takes you 60 seconds to solve a partial, you should not be running the check more frequently than every 3 minutes or so (if you need that level of monitoring). I'd say running every 10 minutes would be appropriate for most users.

---

### Configuration

Obtain your Chia launcher ID. See [this truepool.io knowledgebase article](#) for help with this.

#### NEMS Configurator Setup

- Add a new Advanced Service
- Give the service a name such as “TruePool (Robbie)”
- Give the service a description such as “Chia Farm - Robbie” - I identify my farmer since I monitor multiple farmers
- Set Check Period to 24/7
- Set Notification Period to Work Hours (I don't need to be awoken if my farm goes down)
- Assign the advanced service to host *NEMS*
- Set max check attempts: 5
- Set check interval: 10
- Set retry interval: 10
- Set first notification delay: 30
- Set notification interval: 120
- Set notification options: *w,u,c,r*
- Add your Launcher ID to the appropriate field
- Save, and generate your NEMS Config

### 3.3.30 Check Command: check\_ibmi\*

#### Check Commands

Table 2: IBM i Check Commands in NEMS Linux

Command Name in NEMS NConf	Description
check_ibmi_cpu	Check the CPU utilization of the entire IBM i host.
check_ibmi_job_cpu	Check the CPU usage of a specific job.
check_ibmi_overload_jobs	Retrieve the number of jobs that exceeds the expected CPU usage.
check_ibmi_temp_jobs	Retrieve top jobs that have the most temp storage usage. Number of jobs is user configurable.
check_ibmi_jobs	Check the number of active jobs.
check_ibmi_disk_config	Report the disk configuration of the host.
check_ibmi_disk_usage	Disk usage.
check_ibmi_asp_usage	Retrieve the ASP usage percentage of the entire system.
check_ibmi_disk	Disk utilization.
check_ibmi_messages	Check IBM i messages. <b>I do not know what the ‘ty’ argument represents. If you know, please message me or create a PR.</b>
check_ibmi_message	Check if a specific message (by Message ID) exists in a specific message queue. May be a comma-separated list of Message IDs.
check_ibmi_page_faults	Check for page faults.
check_ibmi_subjobs	Number of subsystem jobs.
check_ibmi_sql	Retrieve the longest running SQL.
check_ibmi_sql_custom	The user could leverage SQL services to create self-defined matrix. That’s what the manual says. Someone, please clarify.
check_ibmi_users	Report the number of users currently logged in (no thresholds or notifications).
check_ibmi_info	Report basic info about the IBM i host.
check_ibmi_integration	Check if the IBM i NEMS Linux integration daemon is running. If not, make sure you enabled it in NEMS SST (it is disabled by default).

#### Configuration

NEMS Linux includes the command `add_ibmi` to allow you to easily configure the credentials for your IBM i Host or SST. To run it, access your NEMS Linux terminal and type `sudo add_ibmi`

Once you have added the credentials, you may proceed to add the host to NEMS NConf as usual. Be sure to add the IBM\_i host template to your host configuration in NEMS NConf.

#### Environment

GROUP PTF - V7R1: SF99701 LEVEL 38 - V7R2: SF99702 LEVEL 16 - V7R3: SF99703 LEVEL 4

USER PROFILE - check-ibmi-long-run-sql needs \*ALLOBJ - check-ibmi-disk-config needs \*ALLOBJ,\*SERVICE,\*IOSYSCFG

PORT - as-central 8470 - as-database 8471 - as-dtaq 8472 - as-file 8473 - as-netprt 8474 - as-rmtcmd 8475 - as-signon 8476 - as-svrmap 449

## Requirements

Requires NEMS Linux 1.6+.

## Source

From <https://www.ibm.com/support/pages/setting-nagios-plug-ibm-i>

## 3.4 NEMS Linux Tips: Backup Your NEMS Configuration Automatically

### 3.4.1 Introduction

One of the worst things that can happen to your NEMS Linux deployment is having your SD card fail. So keeping a current backup of your NEMS configuration is a smart idea.

### 3.4.2 Backup Location

Your NEMS Migrator snapshots are always accessible at `https://NEMSIP/backup/backup.nems` or via Samba at `\\nems.local\backup\backup.nems` - accessing either will automatically generate and send a *backup.nems* file, which contains all the NEMS configuration settings, logs, data, etc. to allow an easy recovery by restoring to a new NEMS deployment.

Both locations are protected by your NEMS username and password, as created during `nems-init`.

The *backup.nems* file can also be used to migrate your NEMS Linux deployment from one architecture to another. For example, if your infrastructure outgrows a Raspberry Pi 3-based NEMS server, you can deploy on an ODROID XU4 and migrate your entire NEMS setup to the new deployment.

Knowing the location of your *backup.nems* file makes it easy to add a NEMS backup to your daily backup script for automated inclusion in your regular backup.

### 3.4.3 Backup Your backup.nems File Automatically

From another Linux server (eg., where your backups run) simply add this to your backup task:

#### Download Via wget Using Secure SSL (Recommended)

---

**Note:** Using SSL not only protects the content of your backup, but also protects your username and password from prying eyes

---

```
wget -O "/backup/backup.nems" https://NEMSIP/backup/ --user=YOURUSER --  
↪password=YOURPASSWORD --no-check-certificate
```

I included `--no-check-certificate` since NEMS is using a self-signed certificate. This will allow the communication to be encrypted, but not fail on the “invalid” self-signed cert.

Note the `-O "/backup/backup.nems"` is just an example of the OUTPUT folder. You’ll want to change this to suit your local system. Eg., `-O "/home/robbie/backups/backup.nems"` - the `-O` is telling it where to save the local copy.

## Download Via wget Without Encryption

**Warning:** This is only recommended for legacy NEMS versions that don't support SSL

```
wget -O "/backup/backup.nems" http://nems.local/backup/ --user=YOURUSER --  
↪password=YOURPASSWORD
```

In both of the above examples, replace `/backup/backup.nems` with where you want nems-migrator to output the download, and `YOURUSER` and `YOURPASSWORD` to those you set during `nems-init`.

Once you have a `backup.nems` file being backed up to a different system, I recommend you have your backup script run an *rdiff-backup* of your `/backup` folder (in this example) to allow for versioning.

## Windows-Based Backup

If you are on a Windows network and would like to include your `backup.nems` file in your nightly backup set, you may access it at `\\nems.local\backup\backup.nems` using your NEMS username and password as set in `nems-init`.

# 3.5 NEMS Navigation Menu

## 3.5.1 Configuration

### NEMS Server Overview

At-a-glance overview of some important information pertaining to your NEMS Server.

### NEMS System Settings Tool

The configuration tool used to setup variables used by Nagios traditionally stored in `resource.cfg`. An example would be your SMTP settings. This is also where you configure various optional components of NEMS Linux.

### NEMS Configurator (NConf)

NEMS' browser-based frontend to your Nagios host and service configs. This is where you'll setup all your hosts, services and check commands.

## 3.5.2 Reporting

### Modern

### Adagios

A modern, responsive web interface for your Nagios check statuses.

### NEMS Mobile UI

A mobile front-end for NEMS Linux for easy status updates via smartphone or tablet.

### NEMS TV Dashboard

Server room TV display which shows any problem hosts or services in realtime, hiding all data that is not currently relevant.

### NEMS Tactical Overview

A realtime tactical overview designed for server room TV display with sound and ability to see historical data.

### Legacy

#### Nagios Core

A themed version of Nagios Core's web interface.

#### NagVis

A graphical representation of your current check statuses.

#### Nagios Graphs

Historical graph data from NEMS perfdata (if configured).

## 3.5.3 System

This is where you can view information about your NEMS server itself (not the hosts it checks).

### Monitorix

Monitorix provides easy-to-read graphs of your NEMS Linux server's system resources.

### RPi-Monitor

View advanced stats on Raspberry Pi systems.

## Cockpit

System interface.

## monit Service Monitor

Your NEMS server keeps an eye on some of the crucial services it expects to be running. If they crash, *monit* will restart them. Using the web interface, you can monitor, restart or stop these services.

## 3.5.4 NEMS Migrator

Tool for easy backup and restore of your NEMS server configuration.

### Backup

Download your current NEMS Migrator backup.

### Restore

Restore a NEMS Migrator backup to your NEMS server.

### Off Site Backup

Optional service which allows you to securely backup and restore your NEMS Linux configuration over the Internet.

## 3.5.5 Buy a Pi

Support the NEMS Linux project by buying your hardware through this link.

## 3.5.6 Support Us

Another way to support NEMS Linux is to [become a patron on Patreon](#) or shop online using our partner links. Those are listed on this menu.

## 3.5.7 Get Help

Various links to helpful resources such as the online documentation, Discord server and community forum.

## 3.6 Features: Port 9590

### 3.6.1 Introduction

Port 9590 is a dummy listener on all NEMS 1.4.1+ servers.

This listener does nothing but reply to TCP requests when it is up. Its purpose is to provide a safe port to enable and disable while learning how to use the check command `check_tcp`.

### 3.6.2 Disable 9590

To disable 9590 temporarily, open its process in Monit Service Monitor and choose “Stop service”. It will however resume normal operation at next reboot.

To disable 9590 permanently, first stop it with Monit (otherwise Monit will re-run the service once it detects it is not running). Then, open an SSH connection to your NEMS server and type:

```
sudo systemctl disable 9590
```

To see how 9590 works, simply review [the source code](#).

## 3.7 Connect to NEMS Server Over SSH

The first time you connect to your NEMS server you will use the `nemsadmin` account.

After you have successfully [initialized NEMS](#), you will instead use the account you create during that process.

### 3.7.1 Linux, Microsoft Windows or macOS

- Open a terminal window (“command prompt” on Windows).
- Type: `ssh nemsadmin@nems.local`

## 3.8 Frequently Asked Questions

### 3.8.1 My browser warns, “Your connection is not secure”. Why?

NEMS Linux uses SSL (aka *https*) connections to secure your connection and the data you transmit and receive to and from your NEMS server. This is accomplished using what is called a *self-signed certificate*. By nature, self-signed certificates are considered “untrusted” by your browser because, simply put, anyone can make them. It does not mean your connection is not encrypted or secure, but rather it means your browser cannot determine who created the certificate, and therefore they cannot verify your security. If you visited a web site, say *google.com* and received a warning that your connection is not secure, you should immediately stop what you’re doing and not proceed. However, in the case of NEMS Linux, which is a local server on your network (not a “dot com” on the web), you can safely trust the self-signed certificates and add an exception to your browser.

See [this documentation](#) for more details.



### 3.8.2 How do I configure WiFi?

See the Wireless Network Interfaces section of the [Networking documentation](#).

### 3.8.3 How do I set a static IP address?

See the IP Address/DNS Settings section of the [Networking documentation](#).

### 3.8.4 Why does Cockpit have greyed-out features? I can't change anything!

As per [the Cockpit docs](#): make sure you also check the box “use my password for privileged tasks” while logging in. Otherwise your level of access will match the non-elevated user and all features which require root access will be greyed out.

### 3.8.5 Why does NEMS Linux “call home” so much?

If you're running a custom DNS server or monitoring DNS lookups, you'll see a lot of requests for nemslinux.com. If you've enabled NEMS Check-In, NEMS will do a DNS lookup for nemslinux.com every 5 minutes. That's 288 requests per day. Then, there is the *nems-info online* command, which pings nemslinux.com to simply determine if you have an Internet connection. This is required every time NEMS Linux runs a command that requires Internet connectivity. It checks for that connection first, and if successful, it resumes. Else, it does not go through with the command since no Internet connection is detected. This is probably where you're seeing the most lookups. Then, there are API requests, such as when it checks to see if you are authenticated with the NEMS Cloud Services server, which will occur at various times depending on what checks are occurring and what you are doing (E.G., if you open NEMS SST, it will immediately check). Another one is that you may have a sample check command which pings nemslinux.com to see if it is up or down. You can remove that check if it exists using NEMS NConf.

### 3.8.6 Why is NEMS using so much bandwidth?

NEMS Linux includes a small handful of sample check commands to get you started, and one of those is an Internet Speedtest which uses Ookla's Speedtest service to monitor and report your Internet speed. For more information, please read [Check Command: check\\_internet\\_speed](#).

## 3.9 NEMS Migrator Local Backup

### 3.9.1 Introduction

SD cards are prone to failure, and NEMS Linux has protections in place to ensure the longevity of your storage media. However, what happens if your card does fail?

NEMS Migrator provides a constant backup of your NEMS Server's configuration, which can be backed up locally or in [NEMS Cloud Services](#) (or both). Restoring your NEMS Migrator backup gets your configuration back online within minutes.

Combined with [NEMS CheckIn](#) to notify you in event of failure, NEMS Migrator is an excellent way to ensure your NEMS Server is protected against failure.

## 3.9.2 Local Backups

### Backup

Your NEMS Linux server automatically generates a backup of your configuration ready for you to download at any time.

Your NEMS Migrator backup (filename: *backup.nems*) contains your NEMS server configuration, allowing you to quickly re-deploy on a new NEMS server or re-flashed device without having to re-create your Nagios configuration manually.

A NEMS Migrator backup is proprietary to NEMS. It is not intended for user consumption, but rather can be used to restore NEMS Linux to a previous configuration, or quickly rebuild after an SD card failure (for example).

NEMS Migrator will check your backup status every few minutes. If the backup is older than 30 minutes, it will replace your backup.nems file with a current snapshot. Therefore you know that if you make a copy of your backup.nems file, it is no older than 30 minutes. If however you make changes to your configuration, those changes may not appear in your backup for up to 30 minutes.

To use NEMS Migrator to upgrade from an earlier version of NEMS Linux (or even NagiosPi), please read [Upgrade NEMS Linux to Newer Version](#).

To automate your NEMS Migrator backup, please see [Backup Your NEMS Configuration Automatically](#) for helpful resources and tips.

### Restore

Please see [nems-restore](#).

## 3.10 Available Platforms

NEMS Linux is available on many platforms, such as Single Board Computer, Virtual Appliance, Cloud Deployment or Docker.

Visit the [NEMS Linux web site](#) for a complete list of supported platforms.

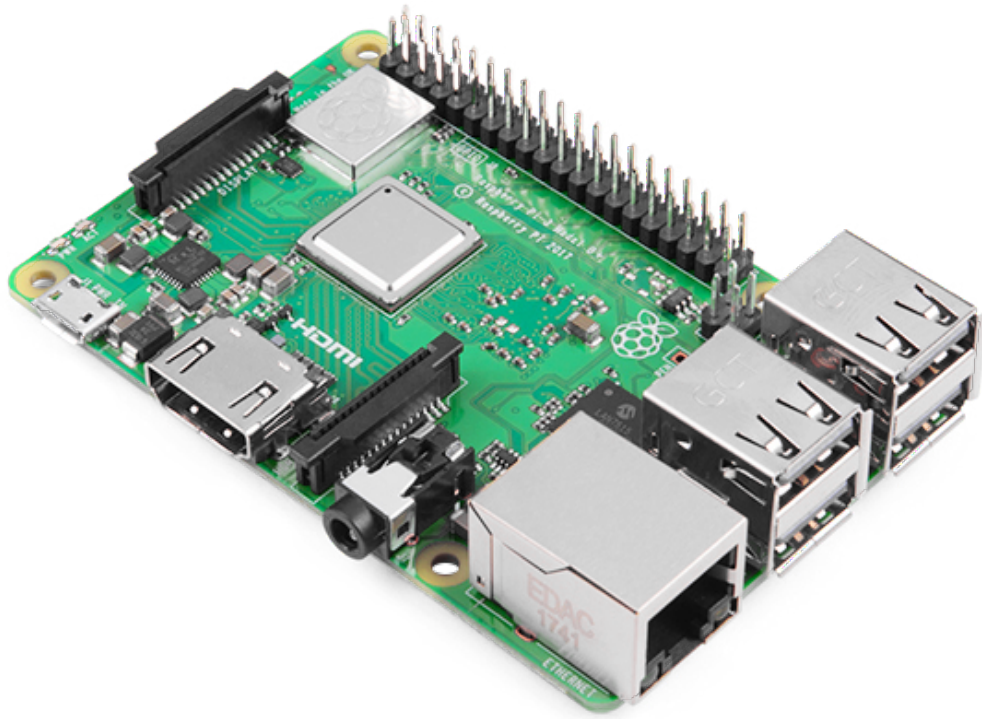
### 3.10.1 NEMS Linux on Single Board Computers

When choosing your hardware, general SBC comparisons are not necessarily relevant since you will be deploying a NEMS Linux server specifically. As an example, an ODROID-C2 vs Raspberry Pi 3 comparison will say Raspberry Pi 3 has better support for video drivers. Well, that won't matter to you; you're using NEMS Linux and we've pre-built the distro for you, and nothing about it requires good video driver support. So because of this, it is helpful to review [the NEMS Linux Stats page](#), and even [discuss it on our Discord Server](#) to make an educated decision.

Here are some general guidelines.

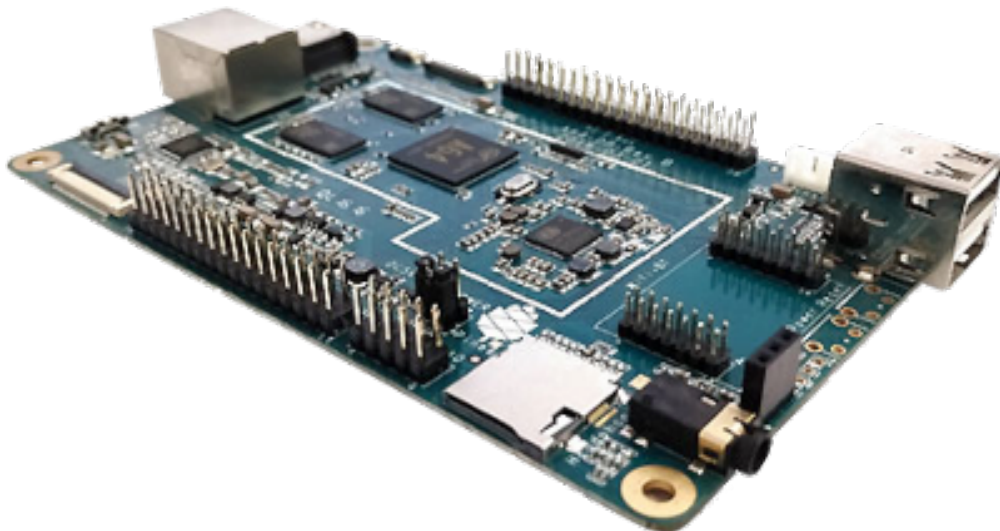
1. eMMC storage is better than an SD Card. This is a universal truth. SD Cards have a high failure rate whereas eMMC tends to operate with perceptively similar reliability and performance to a traditional SSD. NEMS Linux performs a lot of read/write operations, as you can imagine, so the more reliable your storage medium, the more reliable your NEMS Server.
2. More RAM means better performance. The minimum recommended RAM is 1GB, though 2GB or higher will greatly improve performance and reliability of your NEMS Server.

## Raspberry Pi

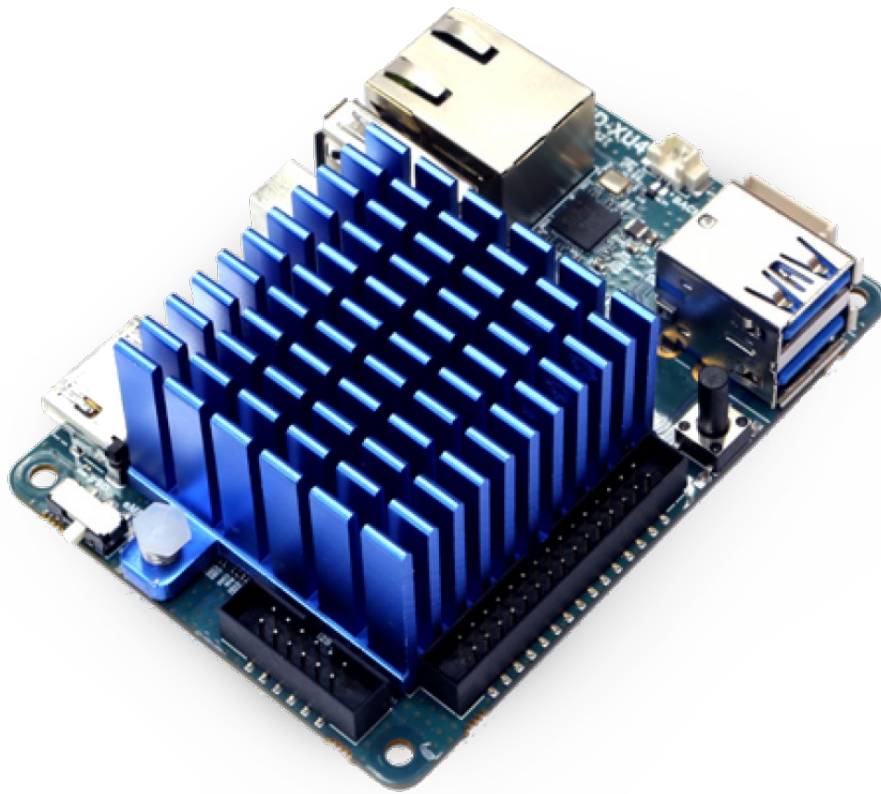


[BUY NOW](#)

## Pine64



## Hardkernel ODROID XU4



## FriendlyElec

## Orange Pi

## ASUS Tinker Board / S

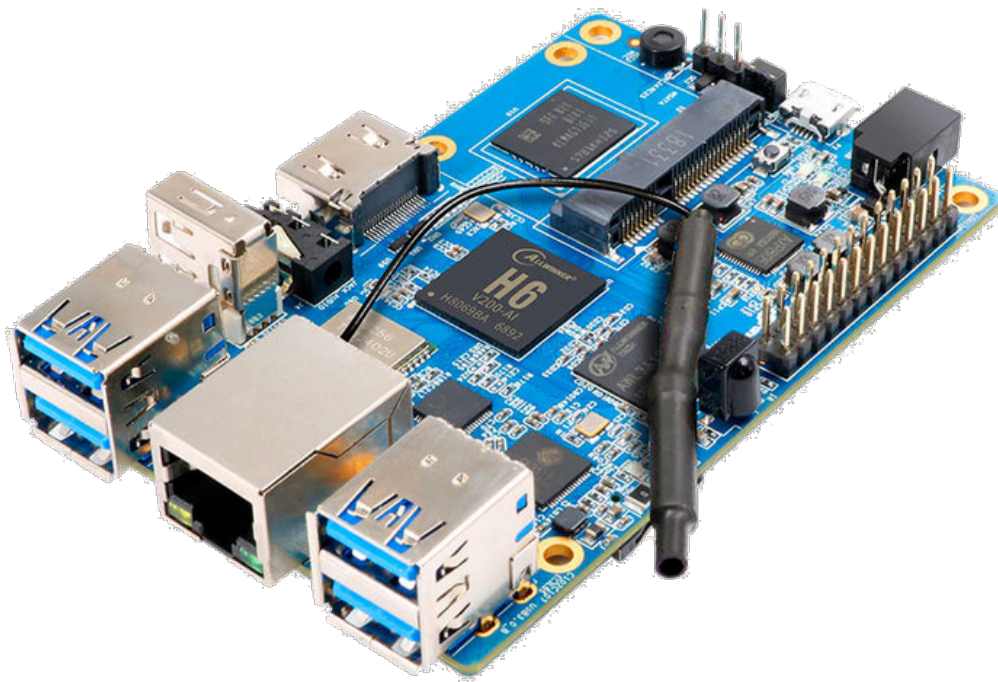
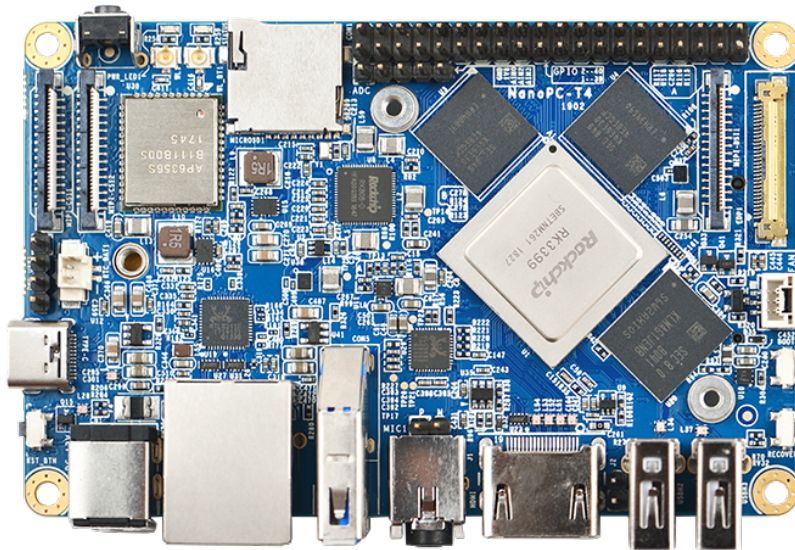
ASUS Tinker Board S must be switched to Maskrom boot mode in order to boot from SD card. The built-in eMMC is not big enough to run NEMS Linux from.

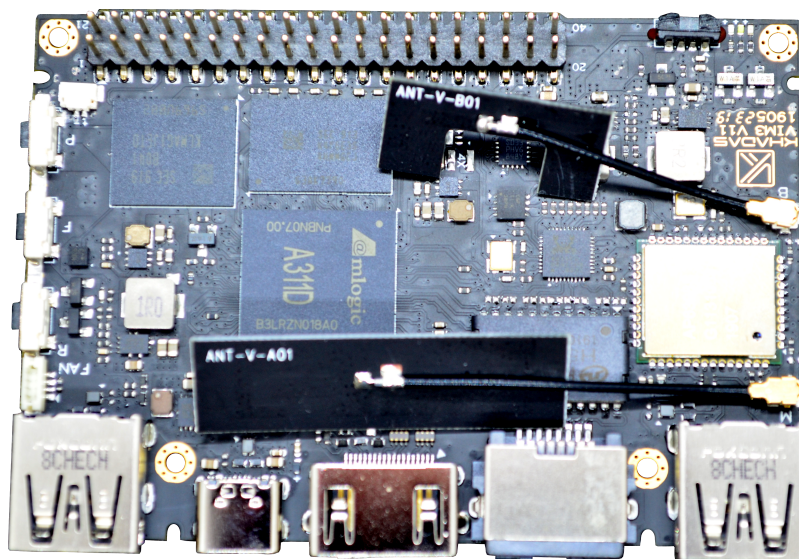
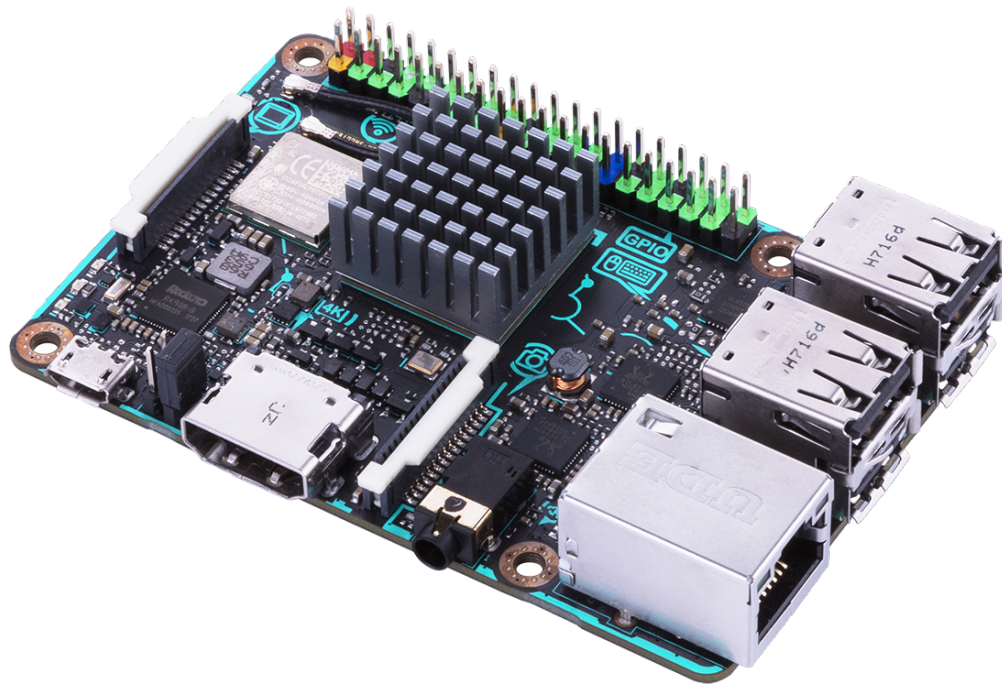
## Khadas VIM3

You can boot from SD or USB, then install NEMS Linux to the integrated eMMC storage by typing;

```
sudo nems-install
```

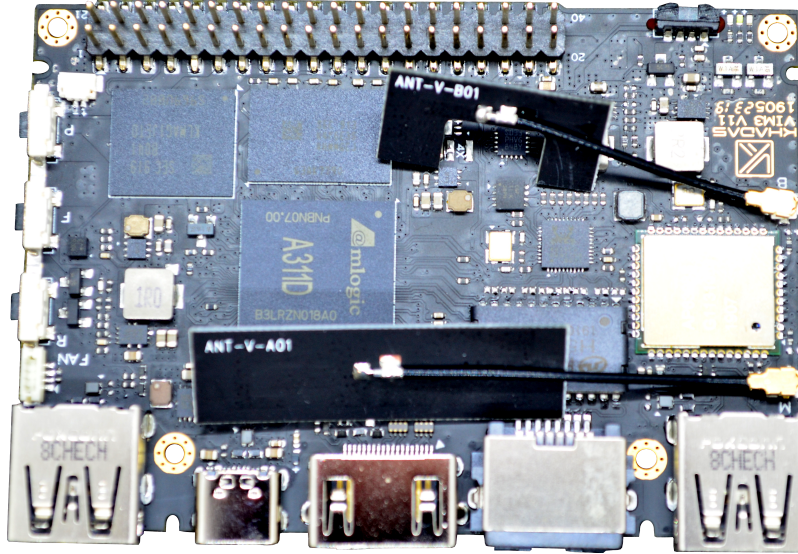






## NEMS Linux for Khadas Single Board Computers

### Khadas VIM3



If you are unable to boot NEMS Linux from your microSD card or USB flash drive on a Khadas VIM3 board, please be sure to use the Krescue tool first to perform a full erase of your eMMC. <https://dl.khadas.com/Firmware/Krescue/dump/VIM3.krescue.sd.img.gz>

You can boot from SD or USB, then install NEMS Linux to the integrated eMMC storage by typing;

```
sudo nems-install
```

### LED Indicators

The Khadas VIM3 contains two built-in LEDs which NEMS Warning Light utilizes by default.

Table 3: Warning Light on VIM3

LED	Meaning
White Flashing	Unknown State or System is Booting
White Solid	OK state
White Solid, Red Flashing	Warning State
Red Solid	Critical State



### 3.10.2 NEMS Linux Virtual Appliance

The NEMS Linux Virtual Appliance is only available to [Patrons](#).

The NEMS Linux virtual appliance has 3 available releases:

- **NEMS OVA** (Open Virtual Appliance) can be easily deployed on virtualization hypervisors such as VMware ESXi, vSphere, Player or Workstation or Oracle VirtualBox. The OVA package contains the entire virtual appliance and is ready to import and boot.
- **NEMS VHD** (Virtual Hard Disk) can be used to deploy NEMS Linux on Microsoft Hyper-V.
- **NEMS QCOW2** (QEMU Copy-On-Write) can be used to deploy NEMS Linux on QEMU, KVM, Proxmox VE, and other hypervisors that support the QCOW2 format.

The underlying software in each release is identical. The individual releases are created in order to ease deployment across a variety of the most popular virtualization hypervisors.

#### Host Requirements

##### All Hypervisors

- VT-x/AMD-V capable CPU with feature enabled in BIOS/UEFI
- Minimum 6 GB free RAM
- 100 GB hard disk space

##### VMware ESXi

- Version 7.0 or higher. NEMS Linux uses Virtual Hardware Version 14.

To run NEMS Linux on a legacy version of ESXi, such as ESXi 6.7, you will need to extract the VMDK from the OVA file (rename the file `.tar` and open it with a tool such as `tar` or `7-zip`) and use `vmkfstools` to expand the disk to thick provisioned. Please consider upgrading your hypervisor for official support. See: <https://kb.vmware.com/s/article/1028943>

##### Guest Specifications

- 64-Bit
- 80 GB Virtual Hard Disk (Dynamic / Thin Provisioning where supported)
- 4 GB RAM

##### Global Deployment Notes

- **Network Bridge** - Before booting, you must configure your virtual Network Interface to use your actual LAN in Bridged mode.
- **Unique MAC Address** - While configuring your virtual Network Interface, you must generate a new MAC address for the virtual NIC. If your hypervisor does not offer a feature to automatically generate a MAC address you can visit [nemslinux.com/api/mac](https://nemslinux.com/api/mac) to generate one. Do not simply enter random numbers. **Record your virtual MAC address somewhere safe.** Do not change your MAC address after initializing NEMS. Doing so would result in your HWID changing, which will disassociate your Virtual Appliance with NEMS Cloud Services.



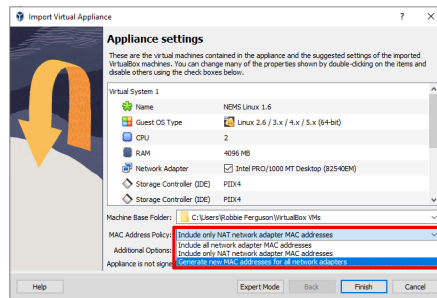
- **CPU Virtualization Features** - x86 Virtualization must be available and enabled on your physical CPU in order to boot the Virtual Appliance. This is found in your host machine's BIOS/UEFI settings and will be called VT-x (Intel) or AMD-V (AMD), or something similar such as "Virtualization Extensions".
- **Static RAM Mode** - Ensure RAM is not assigned as "dynamic" RAM. ESXi, for example, may remove all RAM from the appliance if set to dynamic, which will result in NEMS Linux not functioning correctly. RAM should be static.
- **Hyper-V** - Choose Generation 1 under Specify Generation when creating VM.
- Following initialization, you'll see a CPU Temperature check that reports an unknown state since your hypervisor doesn't report CPU Temperature data for virtual machines. You can remove this check in NEMS Configurator by going to *Advanced Services* [Show], Edit *CPU Temperature*, and remove *NEMS* from your assigned hosts. Save the change and generate your config.

## Installation Procedures

### VirtualBox

#### Minimum VirtualBox Version: 7.0

When importing the NEMS Linux OVA, you must set the MAC Address Policy to "Generate new MAC address for all network adapters".



### 3.10.3 NEMS Linux Amazon Machine Image (AMI)

The NEMS Linux Amazon Machine Image is available in the Amazon EC2 Community AMIs marketplace. Simply search for *NEMS Linux* when launching your instance.

#### Important Note

Your being here means you are an early adopter of NEMS Linux on Amazon Web Services. During this early testing phase, it is available through the community marketplace. However, once NEMS has been tried-and-true, it will be moving into the Amazon Marketplace. This means it will inevitably fall under Amazon's fee structure. For now, it's as free as Amazon allows me to make it.

### AMI IDs

The NEMS Linux AMI is found under *Community AMIs* on the following AWS Service Endpoints. If you wish to deploy on a different AWS Service Endpoint and are a current Patron supporting the project on Patreon, please let me know and I will copy the AMI to your preferred region. Since this costs me extra money to do, I only do it by request, and only for those who contribute to the project.

### NEMS 1.6 AMI Build 1

- North Virginia (us-east-1) ami-0e5001643e1929a55

### NEMS 1.5 AMI Build 1

This is a legacy version and should not be used.

- North Virginia (us-east-1) ami-03480e018178d1c75
- Ireland (eu-west-1) ami-07d0a43c2844ae01c

### Introduction

The NEMS Linux AMI leverages Amazon's T2 instance types, dramatically reducing the cost of running a NEMS Server in the Cloud by bursting to full core performance only when required. T2 instances are also available to use in the AWS Free Tier, which includes 750 hours of t2.micro instances each month for one year for new AWS customers.

The NEMS Linux AMI is an amd64 build.

### AWS Requirements

The NEMS Linux AMI requires the following:

- If monitoring 1-20 hosts: t2.micro or higher EC2 instance
- If monitoring more than 20 hosts: t2.medium or higher EC2 instance
- An elastic IP address
- Volume is 16 GB by default and may need to be increased in time

### Deployment Notes

- **Important:** Before booting, you must configure an elastic IP address for your NEMS Linux instance. Failure to do this will break several key features, including NEMS Cloud Services, NEMS CheckIn, and your daily backup.
- To access NEMS Linux remotely, you will need to configure your Security Group for the NEMS Linux instance to allow incoming connections on the NEMS Linux ports (See [Networking](#) for more info). It is recommended to make these accessible only from your trusted IP address(es).
- NEMS Linux allows you to use either [username/password combinations](#) or username/key pair combinations to login via SSH. As this could pose a security issue, please ensure only your own IP address has access to NEMS Linux ports (in your EC2 Security Group configuration for the instance).

## Re-Initializing

- If you run `nems-init` on a NEMS Server that has already been initialized, the server's public key and all trust relationships will be transferred to the new user prior to the old user being purged.

### 3.10.4 NEMS Linux Docker Container

---

**Note:** The NEMS Linux Docker Container is not yet publicly available. It is coming soon.

---

NEMS Linux for Docker **IS NOT YET AVAILABLE**.

#### Install NEMS Linux for Docker

##### Basic Installation

This command will launch a new Docker container called *nemslinux* using default settings:

```
docker run --hostname nems --mount
type=tmpfs,destination=/tmp,tmpfs-mode=1777 --mount
type=tmpfs,destination=/var/www/html/backup/snapshot,tmpfs-mode=1770
--restart=unless-stopped --stop-timeout 120 --name nemslinux -d
baldnerd/nemslinux:1.6_build1
```

#### Install NEMS Linux Docker Container on a Physical Network

Docker is unlike a standard deployment since by default (with a basic install) only the host computer will have access to it. That of course is not ideal for a NEMS Linux server if you wish to be able to administer it from multiple systems, view dashboards, or use a NEMS Warning Light.

While NEMS Linux will function fine on a Docker network (eg., 172.17.0.2), if you wish to have full access to your NEMS Server just as you would with a physical appliance, you will need to connect it to your physical network.

The two most common options for specifying a network is to use either DHCP or a Static IP Address:

##### Using DHCP

```
docker run --network=multi-host-network --hostname nems --mount
type=tmpfs,destination=/tmp,tmpfs-mode=1777 --mount
type=tmpfs,destination=/var/www/html/backup/snapshot,tmpfs-mode=1770
--restart=unless-stopped --stop-timeout 120 --name nemslinux -d
baldnerd/nemslinux:1.6_build1
```

## Using Static IP

Change the sample 10.0.0.105 IP address to suit your needs.

```
docker network connect --ip 10.0.0.105 multi-host-network run --hostname
nems --mount type=tmpfs,destination=/tmp,tmpfs-mode=1777 --mount
type=tmpfs,destination=/var/www/html/backup/snapshot,tmpfs-mode=1770
--restart=unless-stopped --stop-timeout 120 --name nemslinux -d
baldnerd/nemslinux:1.6_build1
```

Please see [Docker's Network Connections](#) documentation for more help.

## With USB Support

To connect a USB device such as [temper](#) to your Docker-based NEMS Server, first determine its /dev assignment on your host, and then run NEMS as follows, replacing ttyUSB0 with your actual USB device:

```
docker run --device=/dev/ttyUSB0 --hostname nems --mount
type=tmpfs,destination=/tmp,tmpfs-mode=1777 --mount
type=tmpfs,destination=/var/www/html/backup/snapshot,tmpfs-mode=1770
--restart=unless-stopped --stop-timeout 120 --name nemslinux -d
baldnerd/nemslinux:1.6_build1
```

## Initialize Your Docker-Based NEMS Server

Initializing a NEMS Server within a Docker Container is different than all other platforms.

On the Docker host, simply run:

```
docker exec -it nemslinux nems-init
```

## Access NEMS Linux CLI

Should you have need to access the NEMS Linux CLI, you may do so by launching *bash* in your container.

```
docker exec -it nemslinux bash
```

## 3.11 nems-tools: GPIO Extender

If your NEMS Linux server is not powered by a Raspberry Pi, or if it is not located where you'd like your NEMS Warning Light, plugging a NEMS Warning Light device into the GPIO isn't possible. That's where the NEMS Tools GPIO Extender comes in.

To use the NEMS Tools GPIO Extender, you'll need any Raspberry Pi to act as the receiver, but your NEMS Linux server can be any supported platform, including Virtual Appliance. A Pi Zero WH would do very nicely for the task. Since the GPIO Extender receiver is separate from your NEMS Linux server, you can add Raspberry Pi's GPIO to your ODROID-N2 NEMS Server, for example. Or install your NEMS Warning Light on a ceiling-mounted unit closer to the area where your technicians work, connected to the GPIO Extender server (your NEMS Linux server) over wifi. Or install your NEMS Warning Light on the other side of the world: Since the NEMS Tools GPIO Extender is IP-based, it can be anywhere as long as it has network access to your NEMS Linux server (TCP port 9595).

To top it off, you can use an *unlimited* number of NEMS Tools GPIO Extender receivers with a single NEMS Linux Server.

NEMS Linux 1.5+ has the NEMS Tools GPIO Extender server running on port 9595.

To connect to the NEMS Linux GPIO Extender, simply boot up a separate device running [NEMS Extender OS](#).

## 3.12 nems-tools: NEMS Extender OS

The NEMS Extender OS (NEMSeOS) turns any Raspberry Pi device into a NEMS GPIO Extender receiver, complete with NEMS Warning Light connectivity.

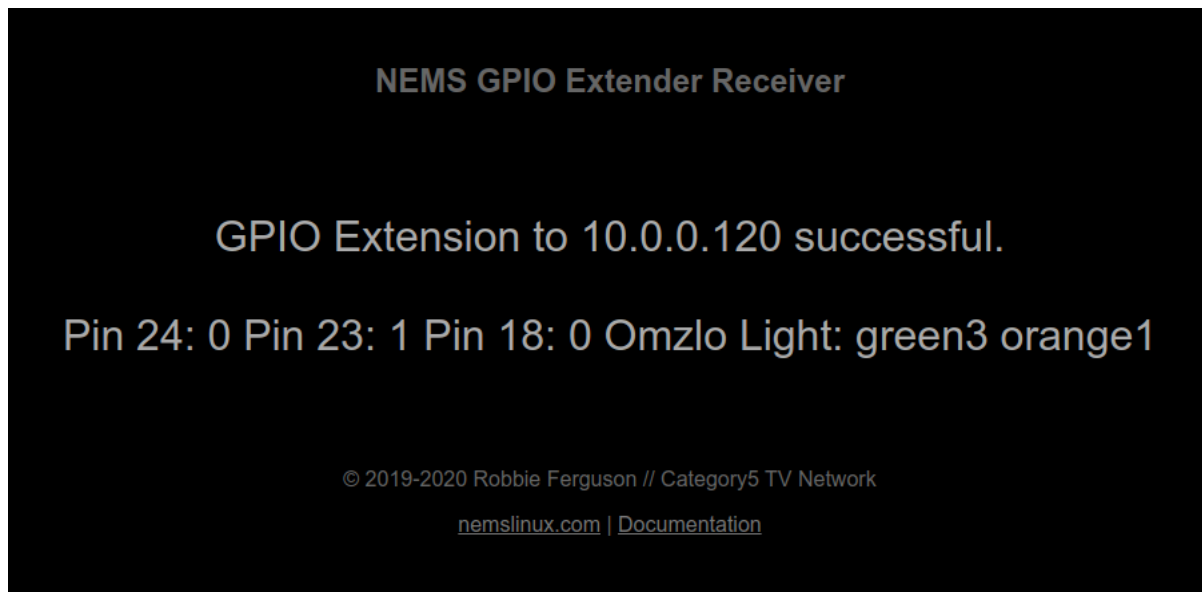


Fig. 6: Browser view of NEMS Extender OS

With NEMS Extender OS, your NEMS Server is the transmitter, and the Raspberry Pi running NEMS Extender OS is the receiver. In this way, you can run your NEMS Server on a different platform, such as virtual appliance, ODROID-XU4 or PINE64 RockPro64, but then connect a NEMS Warning Light (or other compatible GPIO devices) to a separate Raspberry Pi device.

A single NEMS Server can provide data to an unlimited number of NEMS GPIO Extender receivers. This means, for example, you can place a NEMS Warning Light in your server room, and one in your office. You can even place one at home if you're really that committed to uptime.

NEMS Extender OS will be released at the same time as NEMS Linux 1.6.

Following first boot, a file *nems-tools.conf* will be created in the /boot partition. If you need to change the configuration, simply shutdown NEMS Extender OS and plug the card into a computer to edit the file.

## 3.13 nems-tools: NEMS Hero

In the event of a catastrophic issue, such as a corrupt NEMS user or forgotten password, NEMS Linux technical support may use NEMS Hero to recover your NEMS Server, or to provide technical assistance.

NEMS Hero grants temporary root access to a NEMS Server which has been freshly rebooted, and only if the NEMS Server is accessible (via a LAN or WAN connection).

### 3.13.1 NEMS Hero Security

NEMS Hero uses an RSA key pair to establish an SSH trust relationship between a NEMS Server and the NEMS Linux technical support team. This trust relationship self destructs once a NEMS Server has been running for more than 15 minutes.

In order for a connection to be established, the following conditions must be met:

1. port forwarding must be setup on the network of the NEMS Server to allow a support technician remote SSH access to port 22 on the NEMS Server,
2. the NEMS Server must have been rebooted within the past 15 minutes,
3. the NEMS Server must not have a `/boot/no-hero` file in place,
4. the NEMS Linux technical support representative must have our private RSA key,
5. the NEMS Linux technical support representative must know our strong password to access the private RSA key.

---

**Tip:** If desired, this functionality can be disabled on a self-hosted NEMS Server by creating a file `/boot/no-hero` on your NEMS Server. Because the `/boot` partition can be viewed and modified by plugging your NEMS Server's storage into a Windows, macOS or Linux computer, you can easily delete that file later and reboot your NEMS Server to re-enable NEMS Hero.

---

### 3.13.2 System Requirements

NEMS Hero requires NEMS Linux 1.6 or higher.

## 3.14 nems-tools: Warning Light

*Warning Light* is a lightweight daemon running on NEMS Linux (1.4.1+). It is a different type of notification system exclusive to NEMS Linux.



### 3.14.1 NEMS Warning Light: DIY Hardware

If the state of your monitored hosts should change, *Warning Light* could be used to trigger a visual response by way of a tower signal lamp via GPIO, for example.

1. OK: Solid green light
2. WARNING: Solid orange light
3. UNKNOWN: Flashing orange light
4. CRITICAL: Short siren every 15 minutes, solid red light.

*Warning Light* uses [nems-api](#) to monitor the status of your NEMS Linux server.

*Warning Light* can be connected directly to a supported NEMS Linux server, or to a supplemental device running the [NEMS Extender OS](#) with network access, even through the Internet.

### 3.14.2 NEMS Warning Light: Pre-Built Hardware

Omzlo (the makers of the PiVoyager and PiWatcher pHATs) have created an i2c pHAT that integrates with NEMS Warning Light. Learn more on [the Omzlo NEMS Warning Light page](#).

### 3.14.3 System Requirements

*Warning Light* can be connected directly to the GPIO of your Raspberry Pi-based NEMS Linux server. If your NEMS server is a different platform or GPIO to the server is not convenient, you can connect your *Warning Light* to a supported device (such as a Raspberry Pi Zero) and install NEMS Extender OS, which will allow you to add an unlimited number of GPIO receivers to your NEMS server, regardless of platform.

### 3.14.4 Technical Info for Makers

#### GPIO Pin Assignments

On any Raspberry Pi NEMS Linux server or NEMS Tools GPIO Extender receiver, the pinout is as follows:

- OK state -> Pin 24
- UNKNOWN or WARN state -> Pin 23
- CRIT state -> Pin 18

See also, [NEMS Linux: Raspberry Pi GPIO Pinout](#).

**Please Note:** When the final version is released, it will most likely use the I2C protocol (unless I decide to create a USB version instead of GPIO-based). However, these GPIO pins will remain functional in order to provide an alternate means of connecting for tinkerers.

Pins are LOW (0) by default, and will turn HIGH (1) appropriately in event of a state change. For example, if your NEMS Linux server is in OK state, Pin 24 will be HIGH while pins 23 and 18 will be LOW.

The pinout should remain the same, though I may expand it to support a siren signal or passive alarm buzzer. I'd suggest using these pins to trip a 5V relay, powering a higher-voltage indicator such as a signal lamp.

Please share what you create. I'd love to see your designs to make NEMS Warning Light work in your environment.

### 3.14.5 NEMS Tools GPIO Extender

If your NEMS Linux server is not powered by a Raspberry Pi, or if it is not located where you'd like your NEMS Warning Light, plugging a NEMS Warning Light device into the GPIO isn't possible. That's where the NEMS Tools GPIO Extender comes in.

To use the NEMS Tools GPIO Extender, you'll need any Raspberry Pi to act as the receiver, but your NEMS Linux server can be any supported platform, including Virtual Appliance. A Pi Zero WH would do very nicely for the task. Since the GPIO Extender receiver is separate from your NEMS Linux server, you can add Raspberry Pi's GPIO to your ODROID-N2 NEMS Server, for example. Or install your NEMS Warning Light on a ceiling-mounted unit closer to the area where your technicians work, connected to the GPIO Extender server (your NEMS Linux server) over wifi. Or install your NEMS Warning Light on the other side of the world: Since the NEMS Tools GPIO Extender is IP-based, it can be anywhere as long as it has network access to your NEMS Linux server (TCP port 9595).

To top it off, you can use an *unlimited* number of NEMS Tools GPIO Extender receivers with a single NEMS Linux Server.

Your NEMS Linux server (1.5+) is already running the NEMS Tools GPIO Extender server on port 9595.

NEMS Tools GPIO Extender Receiver OS allows you to convert any Raspberry Pi device into a NEMS Tools GPIO Extender.

### 3.14.6 NEMS Warning Light: Webhook

NEMS Warning Light's webhook integration has been replaced. See [notify-by-webhook](#) for details.

### 3.14.7 NEMS Warning Light: Log Files

You can monitor what is happening with *Warning Light* by tailing the log file.

```
tail -f /var/log/nems/nems-tools/warninglight
```

You can view the current state of NEMS Warning Light:

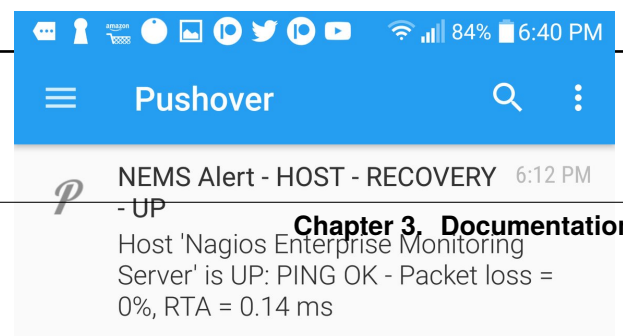
```
cat /var/log/nems/nems-tools/currentstate
```

## 3.15 NEMS Linux Notifications

### 3.15.1 Configuring Pushover for NEMS Notifications

**Warning:** `notify-service-by-pushover` and `notify-host-by-pushover` require NEMS 1.4+

1. Install the [Pushover](#) app on your iOS or Android Device.







2. Sign up for an account, which is free to try (perfect for setting up and testing) but when you decide to use it indefinitely, there is an in-app purchase of around \$5 to get the full license.
3. Login to their web site on your computer and create an app. This will give you the Key you need to add to NEMS SST, along with your USER ID (also provided when you login to the web site).
4. Copy and paste your key and User key into NEMS SST and click “Save”.
5. In NEMS NConf, add **notify-service-by-pushover** and **notify-host-by-pushover** to your contact(s).
6. Done!

### 3.15.2 Configuring Telegram for NEMS Notifications

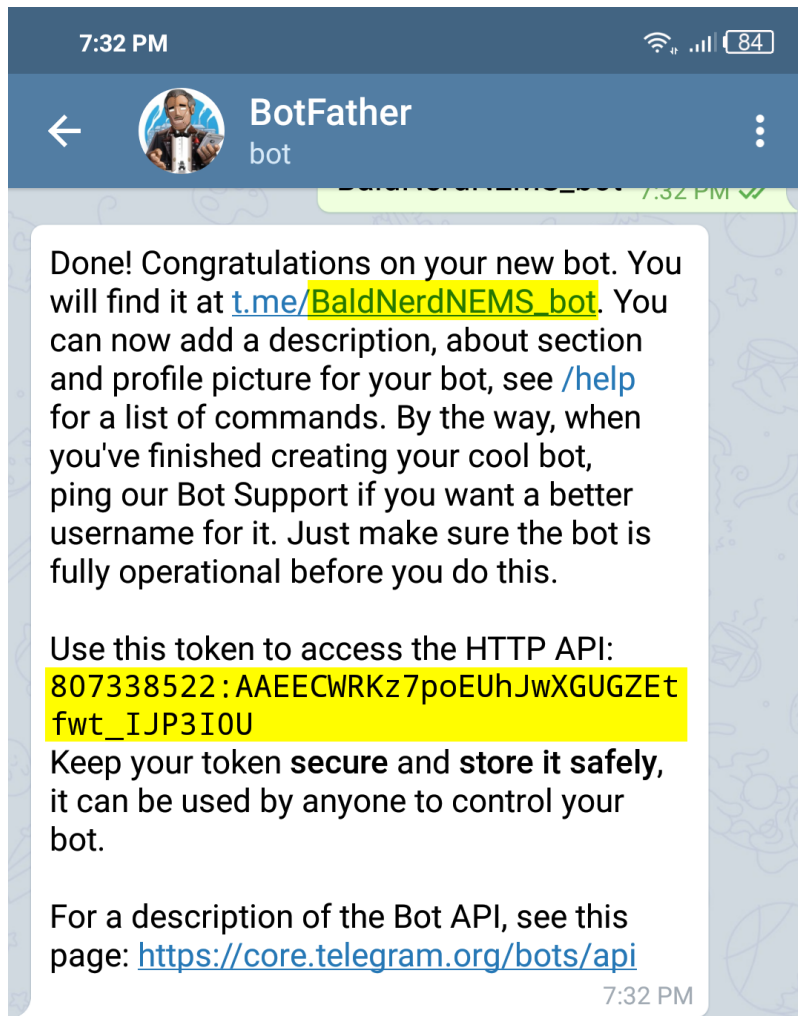
Telegram is a free chat service with an supporting Push notifications on Android and iOS.

NEMS Linux includes *notify\_host\_by\_telegram* and *notify\_service\_by\_telegram*.

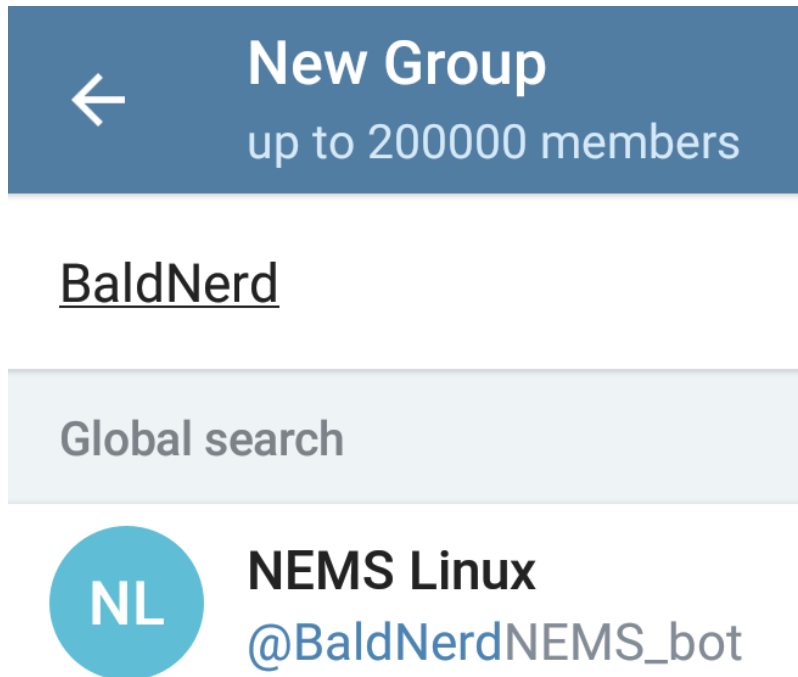
It’s fairly simple to setup, as long as you have your device (ie., smartphone) and a computer/laptop handy.


- 1) Install the free [Telegram app](#) on your device.
- 2) Run the app and follow the simple on-screen prompts to activate a Telegram account (it’s free).
- 3) Create a new message and search for *botfather*.
  - 4) Click on the BotFather bot result, and then click *Start* at the bottom of your app screen.
  - 5) Click **newbot** and then type a name for your bot. *NEMS Linux* is one idea.
  - 6) Choose a username for the bot, which must end in the word *bot*. For example, you could call it *mycool-bot*, *mycoolbot\_bot* or even *mycool\_bot*. I might consider *BaldNerdNEMS\_bot* as my bot username.
  - 7) If all went well, you will then receive an in-app notice that says “Done. Congratulations on your new bot.” followed by some information. In particular, you’ll need to remember your bot name (created in the step above) and the token.
- 8) Click *back*  followed by the hamburger menu  then choose *New Group*.





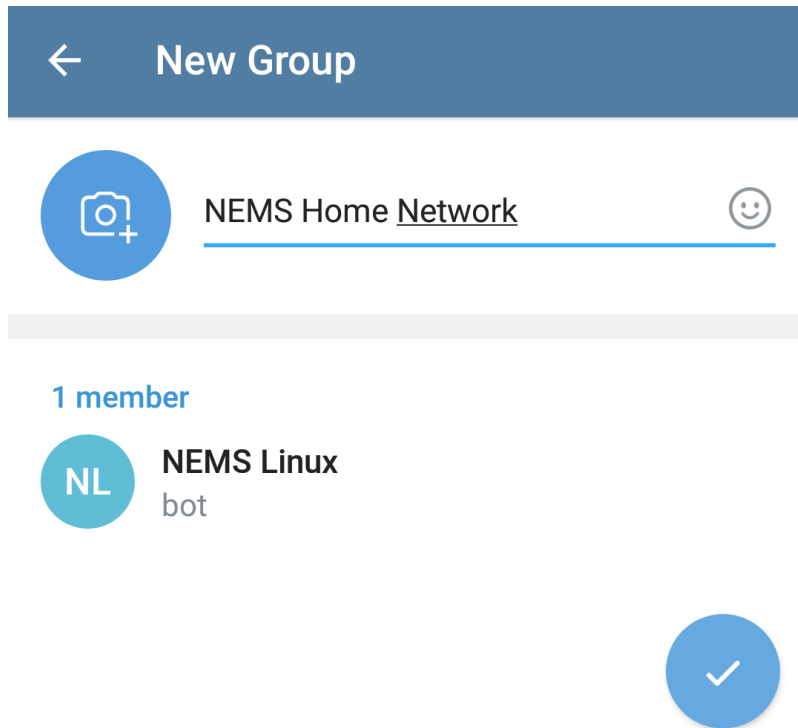
9) Search for the bot you just created just as if it was a person, and click it to add the bot to your new group.



Click your bot's name from the search results, followed by the right arrow  to proceed to the next screen.

- 10) Enter a name for your group and click the checkmark to save.
- 11) On a computer, open the <https://web.telegram.org/> and sign in with the phone number you used to activate your Telegram account.
- 12) Click on the group chat you added your bot to and look at the address bar. It will have a such as <https://web.telegram.org/#/im?p=gXXXXXXXXXX> - hold on to that info (Chat ID: **gXXXXXXXXXX**)
- 13) On your NEMS Server, open [NEMS SST](#) and add your bot Token and Chat ID to the Telegram Account Info section on the *Notifications* tab.
- 14) Finally, open [NEMS NConf](#) and modify your [Contacts](#) ([Contacts](#) → [Show](#) → [Modify](#)). Add `notify_host_by_telegram` and `notify_service_by_telegram` appropriately. Save, and generate your Nagios Config.

Thanks to [baggins](#) for contributing this feature and Vincenzo Di Iorio for assisting with this documentation.



### 3.15.3 Pushover vs. Telegram

#### Why Choose One Or The Other?

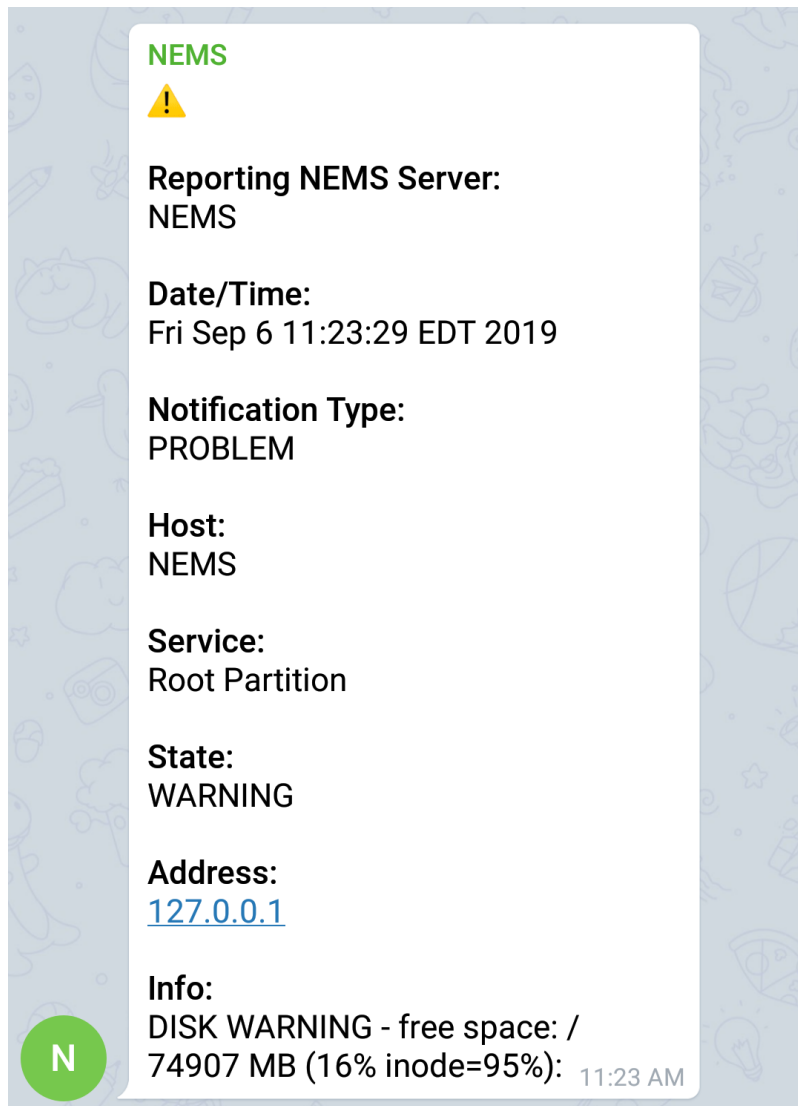
This is not a comprehensive comparison. But rather, this document serves to help you select which push app is right for your needs.

#### Pushover

See: [Notifications Using Pushover](#)

#### Pros

- Notifications feature colorization based on status, which helps critical alerts stand out.
- There is a free trial, allowing you to test it before committing to buying.
- You can set up delivery groups so your entire team is able to receive the notifications.

A notification card from NEMS. It has a green header with the word "NEMS" and a yellow warning triangle icon. The card contains several fields: "Reporting NEMS Server: NEMS", "Date/Time: Fri Sep 6 11:23:29 EDT 2019", "Notification Type: PROBLEM", "Host: NEMS", "Service: Root Partition", "State: WARNING", "Address: 127.0.0.1" (with a blue underline), and "Info: DISK WARNING - free space: / 74907 MB (16% inode=95%): 11:23 AM". A green circle with a white "N" is in the bottom left corner. The background of the card is white with a light blue border, and the overall background is a light blue pattern of various icons.

**NEMS**

⚠

**Reporting NEMS Server:**  
NEMS

**Date/Time:**  
Fri Sep 6 11:23:29 EDT 2019

**Notification Type:**  
PROBLEM

**Host:**  
NEMS

**Service:**  
Root Partition

**State:**  
WARNING

**Address:**  
[127.0.0.1](#)

**Info:**  
DISK WARNING - free space: / 74907 MB (16% inode=95%): 11:23 AM

### Cons

- It costs \$4.99 USD to buy the app. Multiply this by the number of people in your team who need to receive the notifications.
- Having paid for the app, you receive 7,500 “free” notifications per month. If you exceed this, you must purchase more to continue receiving notifications.

### Telegram

See: [Push Notifications Using Telegram](#)

### Pros

- Entirely Free. Your whole team can install the app and it will cost you nothing. There is no fee to purchase the app, and no subscription fees. There are not even any ads.
- Viewing notifications is not limited to the app: You can open your bot in your web browser on your computer to see alerts.
- You can add team members to the room so they also receive the notifications on their device.
- Rich, easy-to-read notifications. Notices include a relevant emoji (eg., exclamation mark in event of problem, checkmark if all is well) and clean layout.

### Cons

- It cannot make me a latte.

## 3.15.4 Notifications via SMS

Many mobile service providers offer the ability to receive SMS messages via an SMS Gateway. This means by sending an email to a special email address provided by your phone service company, you will receive the email as an SMS message.

Of course, this is ideal for NEMS Linux notifications.

## Finding Your SMS Gateway Email Address

Rather than maintain a massive list here (reinventing the wheel), please see [mfitzp's List of SMS Gateways CSV](#) file.

## How To Configure

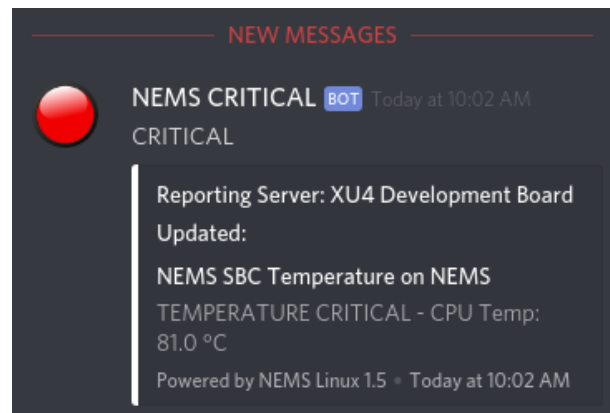
Simply set your contact email address in NEMS NConf to the appropriate SMS Gateway address.

### 3.15.5 Receiving Notifications via Webhook

Send notifications directly to your webhook. For example, have your NEMS Linux server post alerts directly to your company Discord channel.

To have your NEMS Server notify you via webhook, simply add your webhook to [NEMS SST](#) and then add notify-by-webhook to both the Host and Service notifications for your contact in NEMS Configurator.

Webhook functionality was first introduced in NEMS Linux 1.5, but that version was deprecated in favor of notify-by-webhook. Therefore, webhook notifications require NEMS Linux 1.7+. NEMS only sends webhooks. It does not receive them.



#### Currently Supported Webhooks:

- Discord
- Slack
- Microsoft Teams

If you would like support added for another webhook, please simply put it in as a feature request in the [Community Forum](#).

#### Test Your Webhook

```
sudo nems-webhooktest
```



## 3.16 NEMS Commands and Special Features

### 3.16.1 NEMS Commands

The following commands are available in the NEMS Linux terminal, which may be accessed either by a connected keyboard/monitor, or over SSH.

`nems-backup` - NEMS Migrator's backup component.

`nems-benchmark` - Benchmark your NEMS Linux server.

`nems-cert` - Generate or re-generate NEMS SSL Certificates.

`nems-info` - Displays info about your NEMS Linux server.

`nems-init` - Initialize your NEMS Linux server.

`nems-quickfix` - Quickly run all rolling updates and bring in new patches and fixes with one quick command.

`nems-restore` - NEMS Migrator's restore component.

`nems-update` - Update your NEMS Linux server within the current point release.

`nems-upgrade` - Upgrade your NEMS Linux server to the latest point release.

`nems-support` - Generate a special backup file to provide needed information to NEMS Linux support.

### 3.16.2 NEMS Special Features

`nems-migrator` - Backup or restore your entire NEMS server configuration.

## 3.17 Network Configuration

NEMS Linux requires an Internet connection to obtain updates and patches, to tap into [NEMS Cloud Services](#) features such as [NEMS CheckIn](#), [NEMS Migrator Off-Site Backup](#), and [NEMS Anonymous Stats](#), and to send notifications.

**You cannot use NEMS Linux without an Internet connection.**

---

### Security Notice

NEMS Linux is intended to be accessible only to your trusted admin team. Never open ports to the world. Rather, specifically allow your admin IP address(es) access to the needed resources, or use a VPN.

---

### 3.17.1 Firewall Ports

To allow you/others to access your NEMS Server from outside your LAN, you will require the following ports be opened to your NEMS Server:

- **SSH Access:** *22 TCP In*
- **NEMS Dashboard Web Interface / `nems-api`:** *80, 443 TCP In*
- **NEMS Tools GPIO Extender:** *9595 TCP In*
- **Monit Service Monitor Web Access:** *2812 TCP In*
- **Cockpit Admin Interface:** *9090 TCP In*

- **AVAHI / mDNS Name Resolution:** 548, 5353, 5354 TCP In/Out

Most standard network configurations allow servers to communicate with the outside world without any additional setup. However, if you have a very restrictive firewall configuration, you may need to open additional ports for your NEMS Server to be able to communicate with systems it is monitoring, as well as the NEMS Update servers.

- **NEMS Update:** 80, 443, 9418 TCP Out
- **NRPE Check Commands:** 5666, 12489 TCP Out
- **WMI Check Commands:** 135, 445, 1024-1034 Out

You must also ensure your NEMS Server is allowed to communicate with the following TLDs:

- \*.nemslinux.com
- \*.github.com

### 3.17.2 How to Configure Networking on NEMS Linux

NEMS Linux includes the [Cockpit](#) administrative front-end. With some exceptions (listed below) you will only use this interface to configure your networking on a NEMS Linux server.

Setting up your networking by any other means (E.G., a general Linux networking tutorial, modifying system files, or using third-party tools) may result in your NEMS Server's network stack being broken, and so is strongly discouraged. Should you ignore this warning and break your NEMS Server, re-flash NEMS Linux and import your [NEMS Migrator Backup](#) to recover.

### 3.17.3 Docker / Amazon Web Services

Network setup for NEMS Linux on Docker and Amazon Web Services is administered from the host, not the appliance. In these environments, Cockpit is not used to configure the network, and you must familiarize yourself with the host platform's configuration options.

- **Docker:** You must assign the IP upon launching the container. See [Docker documentation](#).
- **Amazon Web Services:** You must assign an Elastic IP to your NEMS Linux instance. See [Amazon Web Services documentation](#).

### 3.17.4 Important Notes

1. When logging in to Cockpit, make sure you check the box "Reuse my password for privileged tasks" otherwise none of these options will be available to you. See [here](#).
2. You'll find Cockpit in the System menu of your NEMS Dashboard. For more details about Cockpit, [read the documentation](#).
3. Network configuration does not persist from board to board: if you configure NEMS on one device and then move its storage (E.G., SD card) to a different device, the new device will revert to the default setting of DHCP.

### 3.17.5 Network Interfaces

NEMS Linux will enable any connected Interface it detects on boot.

#### Ethernet Interface

Recommended. Simply plug in the cable.

#### Wireless Network Interface

Because of the very nature of NEMS, the possible reliability issues on WiFi, and the fact that a WiFi connection will likely result in false notifications, it is not recommended to run your NEMS server on a WiFi connection. However, this may be the only option in some environments, and so great effort has gone into ensuring it will work as well as possible.

Examples where NEMS Linux will be hindered by a WiFi connection:

- User has Gigabit Internet. WiFi connection is 54 Mb/sec. User wants a notification if the Internet ever drops below 200 Mb/sec, but this is impossible since the WiFi connection is *always* below 200 Mb/sec.
- While perhaps untrue for the average home user, an enterprise WiFi connection, if configured securely, should never have access to particular resources on the network. For example, at the Category5 TV studio, WiFi can access the Internet, but not the server. Therefore a NEMS Server in this environment would not be able to monitor items such as server disk space.
- User's WiFi occasionally drops connection, which could result in false notifications that the Internet is down, or cause [NEMS CheckIn](#) to lose contact with the NEMS Server, resulting in yet more false notifications.

**Warning: Raspberry Pi users:** Do not use *raspi-config* for your WiFi.

#### Enable the WiFi Radio in NEMS Linux

In order to be able to activate/deactivate your WiFi connection in Cockpit, you must first add the connection information to your NEMS Linux server using the terminal. If you do not, you'll receive an error "A 'wireless' setting is required if no AP path was given." when trying to activate WiFi.

Determine if your WiFi radio is enabled or disabled:

```
sudo nmcli radio wifi
```

If it is disabled, enable it:

```
sudo nmcli radio wifi on
```

Scan for available wireless networks:

```
sudo nmcli device wifi list
```

Determine which is the one you wish to connect to and enter the following command:

```
sudo nmcli device wifi connect [SSID] password [password]
```

Eg., if your network SSID is MyWiFi and your WiFi password is MyPass123, the command would look like this:

```
sudo nmcli device wifi connect MyWiFi password MyPass123
```

Now, you can enable or disable your wireless connection within Cockpit→Networking.

### 3.17.6 IP Address/DNS Settings

#### DHCP Assigned IP Address

By default, NEMS Linux will obtain its network settings from your DHCP server. For this reason, a quick and easy way to set a static IP on your NEMS server would be to add it as a DHCP reservation within your router/DHCP server. To find out what IP address your NEMS server resides on, either check your DHCP pool, or connect a TV to your NEMS server. You can also try accessing it at <https://nems.local> from another computer on the same network and then open [NEMS Server Overview](#) to see your IP.

#### Static IP Address

If your NEMS server is already initialized, it is recommended that you copy [your backup.nems](#) to a different system prior to setting a static IP address. This gives you an easy way to recover should you accidentally lock yourself out of your NEMS server by breaking the network configuration.

When setting a static IP address, it is important to make sure it is outside your DHCP pool. Otherwise, some routers/DHCP servers may assign the IP to a second device, causing all kinds of unforeseen issues.

#### Set a static IP Address in NEMS Linux

1. Open [Cockpit](#).
2. Login. Use the default credentials if you have not initialized NEMS, or your created credentials if you have. Check the box “Reuse my password for privileged tasks” while signing in.
3. Click “Networking”.
4. Click the network interface (E.G., eth0).
5. Ensure “Connect automatically” is checked.
6. Click the “Automatic (DHCP)” or IP address currently assigned to this NIC next to IPv4.
7. Ensure “Manual” is selected in the dropdown.
8. Add your new IP settings.
9. Make sure you click the + next to DNS settings and assign at least one DNS server. 8.8.8.8 (Google) or 1.1.1.1 (Cloudflare) will do.
10. Press “Apply” and wait for it to test the connection.
11. Click “Change the setting” after the test is complete.
12. You should now open your NEMS Dashboard at the new IP address. Within a few moments, the old one will stop working.

## 3.18 Using Gmail SMTP For Nagios Notification Sending

When using Gmail for SMTP, a few extra steps are required.

First, if you are a G Suite user (that is to say, you're using Gmail but have an @yourdomain.com email address on their service as opposed to an @gmail.com address), visit [this page](#) and configure the SMTP Relay Service like this:

### SMTP relay service

NEMS [Edit](#)

#### 1. Allowed senders

Only addresses in my domains ▼

#### 2. Authentication

☐ Only accept mail from the specified IP addresses

☒ Require SMTP Authentication

#### 3. Encryption

☒ Require TLS encryption

Of course, if you happen to have a static public IP address on your NEMS server, go ahead and enable IP filtering for even more security.

Then, visit [this page](#) and enable “Allow users to manage their access to less secure apps”.

Now that you've done that, **or if you are not a G Suite user**, you'll need to enable the setting for “less secure apps” here: <https://myaccount.google.com/lesssecureapps>

Please note that if you are using Google's two-factor authentication, then you will **not** be able to enable the “less secure apps” setting. Instead, you will need to create an “app specific password”. For more information and details on how to configure app specific passwords, please visit <https://support.google.com/accounts/answer/185833?hl=en>

## 3.19 Using MS Outlook.com SMTP For Nagios Notifications

**NEMS System Settings Tool**

General NEMS Migrator Backup NEMS Cloud Services **Notifications** TV Dashboard Optional Services

**SMTP Email Configuration** ⓘ

SMTP Server Address  
smtp-mail.outlook.com

SMTP Server Port  
587

SMTP Secure Authentication  
Use TLS Secure Authentication

"From" Sender Email Address  
<valid outlook account>@outlook.com

SMTP Authentication Username (Typically an email address)  
<valid outlook account>@outlook.com

SMTP Password  
App password created from outlook.com account

If you are a Microsoft Outlook.com email user configure settings as shown above:

**SMTP Server Address:** smtp-mail.outlook.com

**SMTP Server Port:** 587

**SMTP Secure Authentication:** Use TLS Secure Authentication

**"From" Sender Address:** Must be a valid Microsoft account address

**SMTP Authentication Username (Usually an email address):** Must be a valid Microsoft account address

**SMTP Password:** App password created from MS account page - Login to your account, click on your avatar, click My Account, click Security, click Advanced security options, under App Passwords click Create New App Password, copy and paste password into this field

Click Save All Settings

## 3.20 NEMS Server Overview

NEMS Server Overview, found in the Configuration menu of NEMS Dashboard, is an at-a-glance overview of some important information about your NEMS Server.

On this screen, you can obtain your unique NEMS Hardware ID (HWID), IP address, and other useful information about your running NEMS Server.

You may also review what anonymous data NEMS Linux shares with the NEMS API.

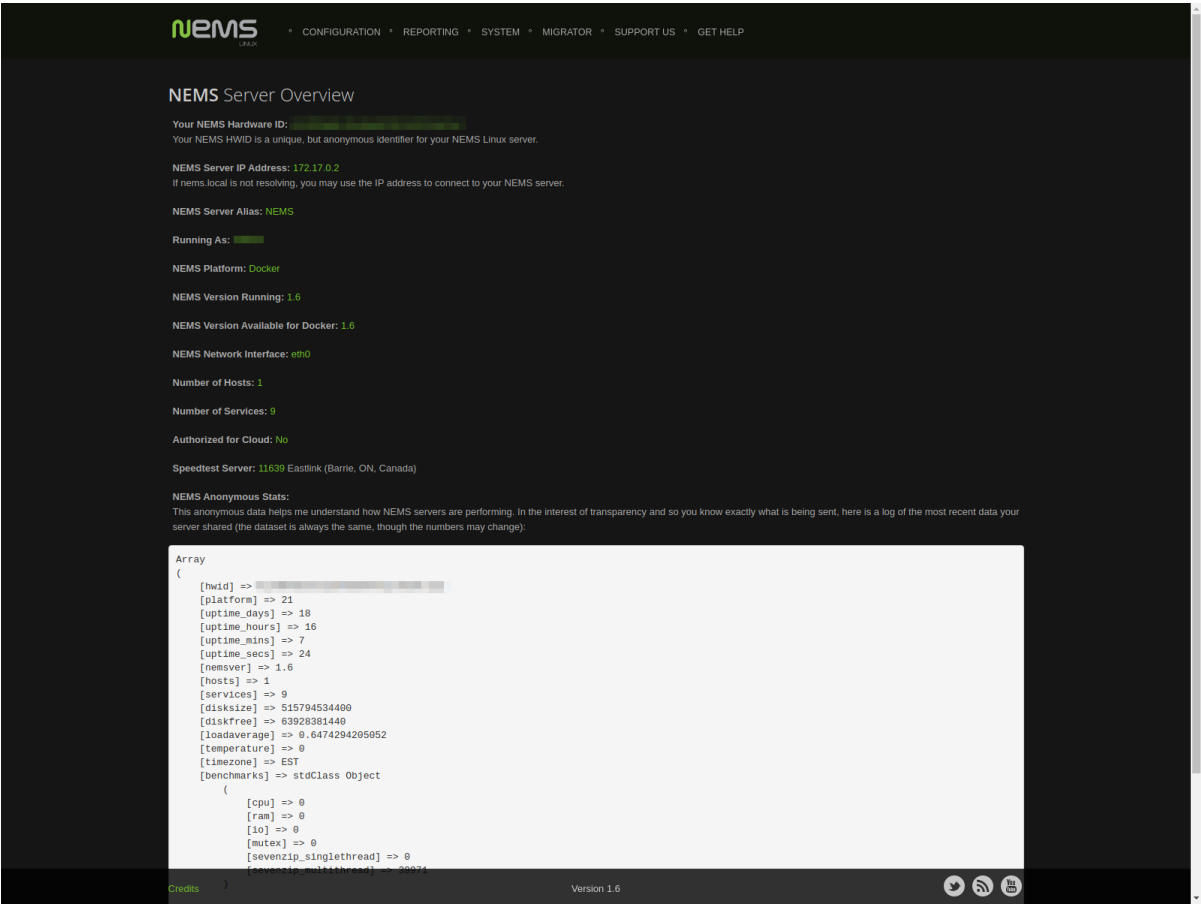


Fig. 7: NEMS Server Overview screen circa NEMS Linux 1.6.

## 3.21 NEMS System Settings Tool (SST)

---

**Note:** NEMS SST requires NEMS Linux 1.3+

---

Gone are the days of manually editing your Nagios *resource.cfg* file.

NEMS SST provides a web-based GUI to allow easy configuration of system settings such as SMTP server settings.

To access NEMS SST, click Configuration→NEMS System Settings Tool.

To login, use your NEMS username and password as configured during `nems-init`.

### 3.21.1 Custom Appearance

Beginning in NEMS Linux 1.5, NEMS SST features the ability to change the background on some NEMS screens.

**Background Selection** allows you to select from the following:

- *Daily Image* (default) - Load a new image every day.
- *Daily Color* - Load a new background base color every day, based on the color pallet of the daily image.
- *NEMS Legacy* - Classic server room image from NEMS Linux 1.4.
- *Custom Color* - Choose your own base color to use for the background.
- *Upload Image* - Upload your own preferred wallpaper image.

**Blur Background Selection** allows you to add a blur effect to background images:

- *Slight Blur* will add a subtle Gaussian blur to the background image.
- *Medium Blur* will add a more pronounced blur to the background image.
- *Heavy Blur* will blur the background image so heavily that only the color scheme of the image is recognizable.

### 3.21.2 TV Dashboard

You can turn it on and off again. (Placeholder for a ref in TV Dashboard page)

## 3.22 NConf

Topic content does not yet exist

## 3.23 Adagios

Topic content does not yet exist



## 3.24 NEMS Mobile UI

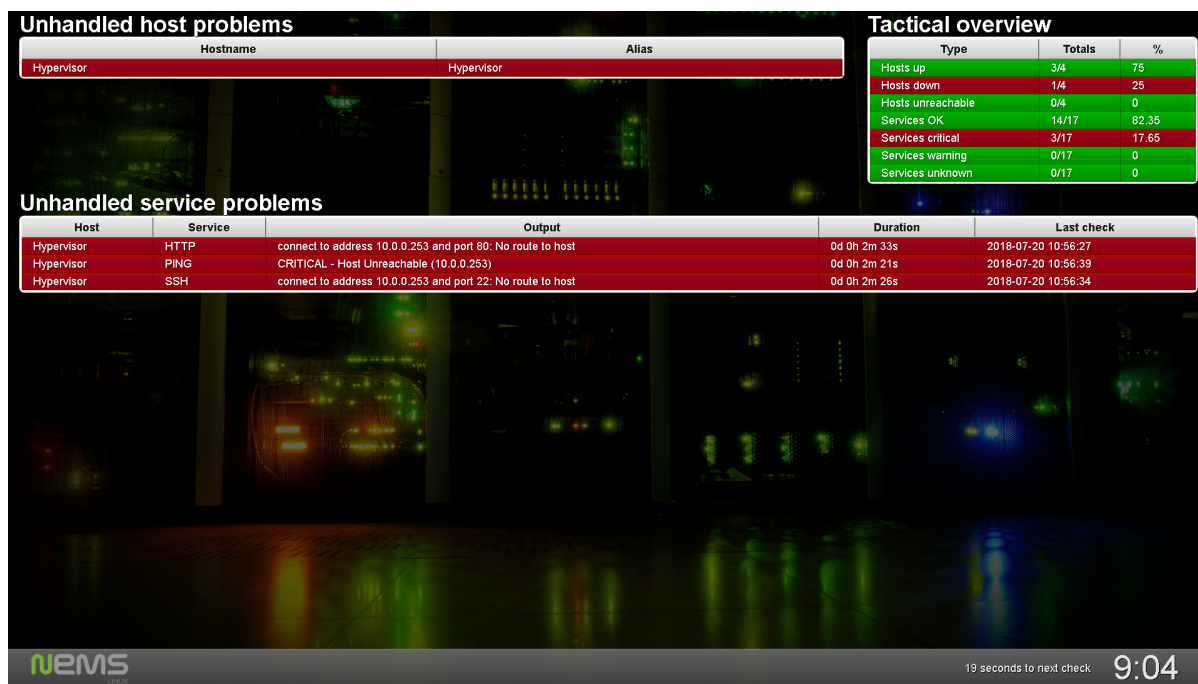
Topic content does not yet exist

## 3.25 NEMS TV Dashboard

### 3.25.1 Introduction

NEMS TV Dashboard provides a tactical display specifically designed for single-screen TV display where a keyboard/mouse are not used. Ideal as a status monitor in your server room, NEMS TV Dashboard will report all unacknowledged host and service alerts as they occur.

#### Screenshot



### 3.25.2 Usage

To use NEMS TV Dashboard, simply connect a computer (E.G., Raspberry Pi Zero) to your TV display and open <https://nems.local/tv> in the browser.

### 3.25.3 Fullscreen Mode

Ideally, make NEMS TV Dashboard fullscreen by pressing F11. Press it again to restore the window.

### 3.25.4 Security

For convenience and usability, NEMS TV Dashboard does not prompt for your NEMS username and password by default. If you prefer to keep the Dashboard private, please open [NEMS SST](#) and turn off “Allow TV Dashboard Without Password” - but make sure you have a keyboard connected to the computer connected to your TV (or something like VNC access) to login.

### 3.25.5 Under The Hood

NEMS TV Dashboard uses the Check\_MK Livestatus socket in the back-end, making it fast, efficient, and lightweight.

### 3.25.6 Requirements

- NEMS Linux 1.4.1+
- A display of some sort, connected to a computer/SBC with Web Browser and network connection with access to NEMS server.
- NEMS TV Dashboard requires a modern web browser to operate. If you experience issues, please test in a different browser.
- NEMS Linux 1.6+ to include Internet speedtest at all times.

### 3.25.7 Source Code

[nems-tv](#) on [Github](#) is based on *merlin-dashboard* by [fnordpojk](#). Licensed under [GPL v3](#).

## 3.26 NEMS Tactical Overview

Topic content does not yet exist

## 3.27 Monitorix

Topic content does not yet exist

## 3.28 Cockpit

NEMS Linux includes [Cockpit from Red Hat](#). In its NEMS implementation, Cockpit provides browser-based SSH access, some realtime performance graphs, and basic system administration tools such as the ability to reboot or safely shutdown your NEMS server.

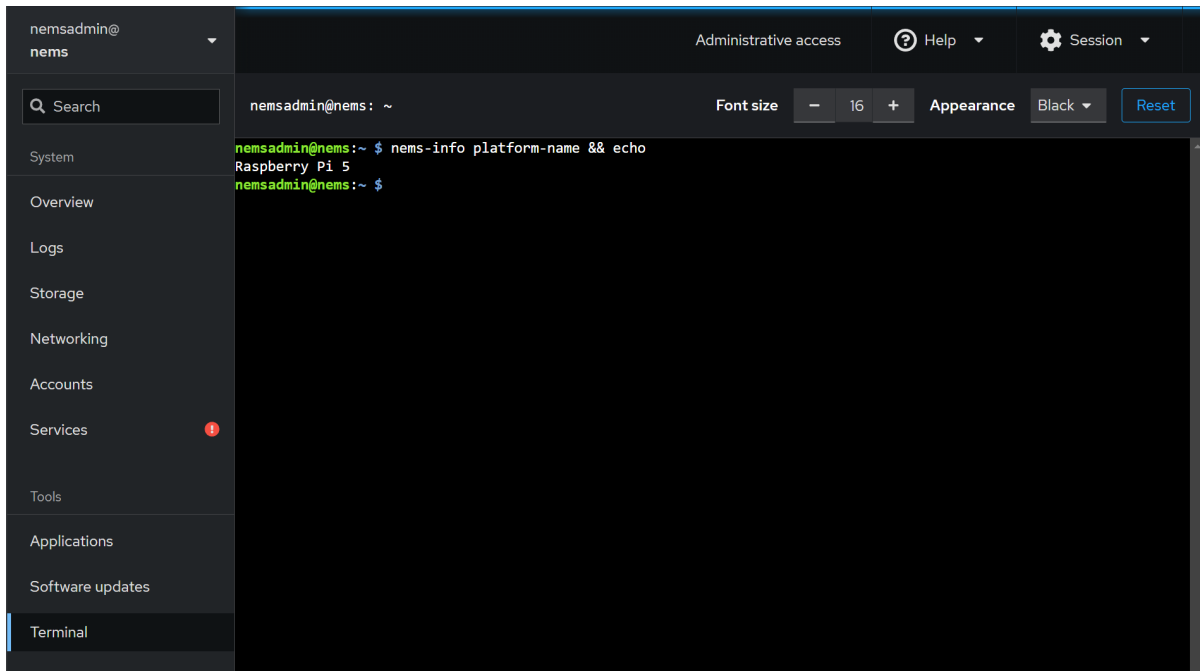


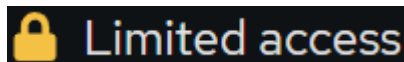
Fig. 8: Cockpit allows terminal access and other system-level tools within a browser session.

### 3.28.1 Login to Cockpit

You'll find Cockpit in the System menu on NEMS Dashboard. To login, enter your NEMS Linux username and password.

#### Seeing Disabled Features in Cockpit?

When you first login to Cockpit, you will be running in Limited access mode. Simply click the button at the top of the Cockpit window to enable Administrative access:



Otherwise your level of access will match the non-elevated user and all features which require root access will be greyed out.

## 3.29 Monit Service Monitor

Topic content does not yet exist

## 3.30 NEMS Cloud Services

Quick points

- *nems-info cloudatauth* will show if your NEMS Cloud Services account is active or not.
- When active, NEMS Cloud Services will receive the updated status of all your NEMS servers every 60 seconds.

### 3.30.1 Requirements

In order to use NEMS Cloud Services, you must be running NEMS Linux 1.5 or higher. You must also be a Patron of [NEMS Linux on Patreon](#) at a tier that includes NEMS Cloud Services.

### 3.30.2 Included Services

#### NEMS Migrator Off-Site Backup

##### Introduction

SD cards are prone to failure, and NEMS Linux has protections in place to ensure the longevity of your storage media. However, what happens if your card does fail?

NEMS Migrator provides a constant backup of your NEMS Server's configuration, which can be [backed up locally](#) or in NEMS Cloud Services (or both). Restoring your NEMS Migrator backup gets your configuration back online within minutes.

Combined with [NEMS CheckIn](#) to notify you in event of failure, NEMS Migrator is an excellent way to ensure your NEMS Server is protected against failure.

##### Off-Site Backup

NEMS Migrator Off-Site Backup (OSB) is a feature of NEMS Cloud Services. It encrypts your NEMS Migrator backup on your local NEMS server and then transmits it to our cloud server over a secure connection where it is retained securely for 90 days.

Your NEMS Migrator Off-Site Backup is encrypted using a password you create locally in NEMS SST. Because of this, only you can restore from this file. Your private password is never sent to our server. Your account is authenticated based on your NEMS HWID and your NEMS Cloud Services Key which is provided to you when you register.

Sign up on [Patreon](#) (observe the reward options) and then add your NEMS Cloud Services Key to NEMS-SST to activate the service.

Once activated, NEMS will automatically backup your configuration off-site every day at 4am.

## Logging

The backup results are stored in `/var/log/nems/nems-osb.log`

## Log Format

The log file is a single line of data sent by the server. Each variable is separated by two colons (:) and begin with variable 0.

- **0** - Plain text date/time of your NEMS server's when the backup began
- **1** - Response code after file sent
- **2** - Plain-text interpretation of 1 (ie., Success or Failed)
- **3** - Plain-text interpretation of 1 (ie., Reason for 2)

If (and only if) the backup was successful with a response code of 1, the following data will also be logged:

- **4** - The remote server's unix timestamp at time of off-site backup authentication
- **5** - The size of this backup
- **6** - Your total usage on the remote storage server, including this backup
- **7** - How many daily backups are currently retained for your account

## Backup Maintenance Schedule

Your NEMS Off-Site Backup is maintained automatically by the NEMS Cloud Services system. The backup schedule is as follows:

- **Past Month:** Every successful backup will be retained and available to you.
- **Past Year:** All backups beyond the past month will have the first successful backup per week retained and available to you. Additional backups will be purged. For example, if you had a successful backup every day, Monday's backup will be retained and available to you, but Tuesday-Sunday will be purged.
- **Beyond One Year:** Backups of this age are automatically purged and no longer available. If you require older backups, please ensure you keep a local copy of your `backup.nems`.

## NEMS CheckIn

### Introduction

NEMS CheckIn is a feature of NEMS Cloud Services designed to address concerns that if a NEMS Server was to go offline, the admin would stop receiving notifications and may not realize this for some time. In early versions of NEMS Linux, this led to some users setting up multiple NEMS Servers—NEMS Servers to monitor NEMS Servers.

NEMS CheckIn is the answer to this. If enabled, NEMS CheckIn will tell the NEMS Cloud Services API that your NEMS server is online. If your NEMS server goes offline (stops reporting in), the NEMS Cloud Services server will send you an email letting you know it lost contact with your NEMS server.

## Configuration

NEMS CheckIn can be enabled within NEMS System Settings Tool under NEMS Cloud Services. It is disabled by default.

### NEMS CheckIn Notifications

---

Receive an email if your NEMS server goes offline or crashes.

State

Enabled 

Email Address for CheckIn Alerts

CheckIn email address 

Email Address for CheckIn Alerts

When to Notify

After 2 Hours 

Fig. 9: Configuring NEMS CheckIn in NEMS SST is a breeze

Should the server not hear from your NEMS Linux server within a set time frame, an email will be sent to the email address you've specified in NEMS System Settings Tool, alerting you that your NEMS Server has failed to check in. It is a simple but highly effective solution.

## Notifications

A total of up to 5 notifications will be sent should your NEMS server fail to check in. Notices are sent at intervals of:

1. The interval you choose in NEMS SST (Default is after your NEMS server has been offline for 2 hours)
2. After 6 hours of downtime.
3. After 24 hours of downtime.
4. After 48 hours of downtime.
5. Following recovery.

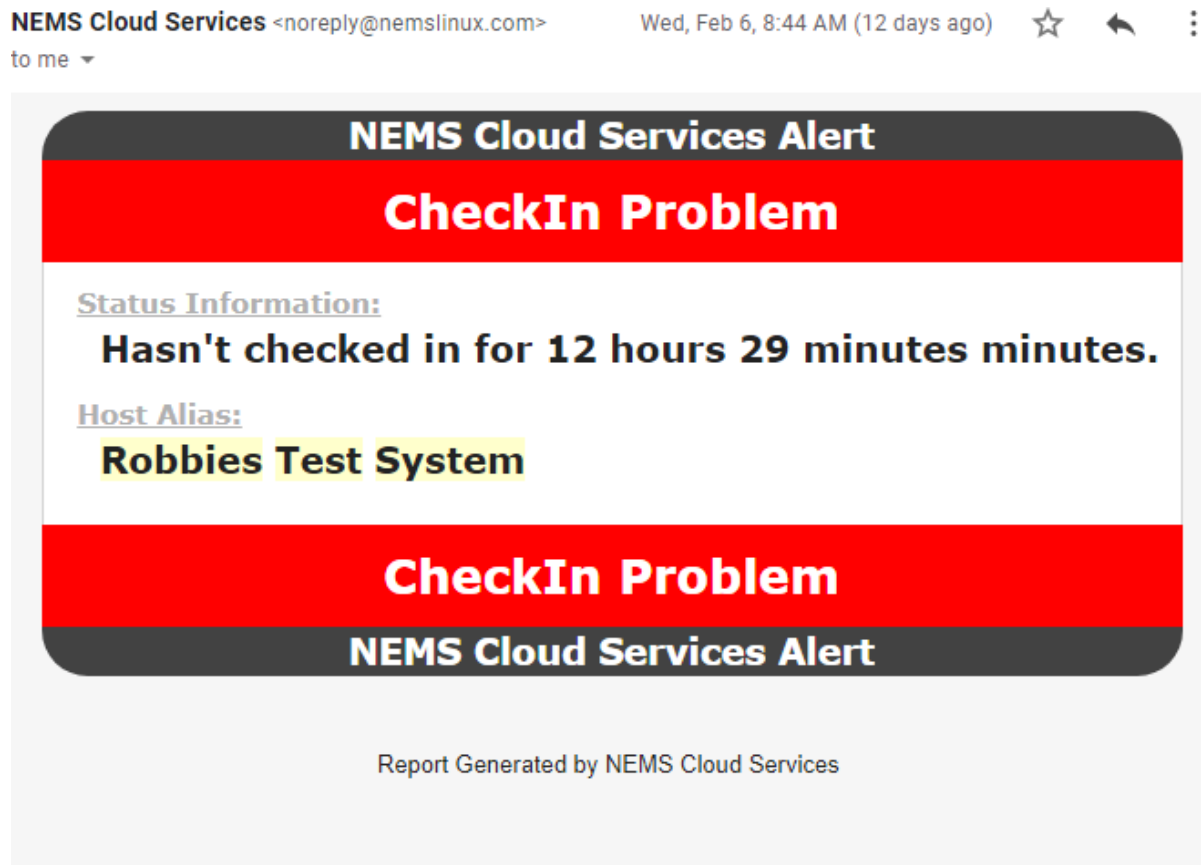


Fig. 10: A sample NEMS CheckIn notification

### Anti-Spam

All NEMS CheckIn notifications will come from *noreply@nemslinux.com* and will match our DKIM. However, aggressive anti-spam may move these important messages to your junk folder. Please be sure to proactively whitelist *noreply@nemslinux.com* so you always see your NEMS CheckIn notifications.

### System Requirements

NEMS CheckIn requires the following:

- An active NEMS Cloud Services account,
- NEMS Linux 1.5 or higher,
- A working recipient email account to send notifications to.

NEMS CheckIn does not require that you have a sending email service.

### Troubleshooting

#### State shows “Enabled [Not Functional]”

In order to use NEMS Cloud Services, you must have a valid account linked to your NEMS Server. Please sign in using the NEMS Cloud Services Key in order to use NEMS CheckIn.

### NEMS Cloud Services API

NEMS Cloud Services features some basic API functionality to improve the NEMS Linux experience. These are more like helper tools to make NEMS just a bit better.

- <https://nemslinux.com/api/mac> - Generate a Random MAC Address. Can be helpful when configuring a new NEMS Virtual Appliance.
- <https://nemslinux.com/api/ip> - IP Address detection. Can be used to determine the public IP address of your network.

```
curl https://nemslinux.com/api/ip
```

## 3.30.3 NEMS Cloud Services Dashboard

### Features

- View your NEMS Linux state information from anywhere.
- Open NEMS TV Dashboard for any of your NEMS Servers from anywhere.
- Connect multiple NEMS Servers to a single dashboard to be able to see the state of multiple distributed NEMS Servers on one screen.
- Extend NEMS Warning Light and NEMS GPIO Extender to support multiple NEMS Servers and be geographically located anywhere in the world.



## Security Points

- Cloud Dashboard password is set in NEMS SST
- Password is salted with a 256-byte key before the data from nems is encrypted
- data includes state information only (same info that appears on nems tv dashboard)
- multiple NEMS servers may be linked together to a single cloud dashboard
- since nobody knows your custom password, only you can see your data (it cannot be decrypted without your password)
- leaving dashboard open continually extends cookie, so it is suitable for server room (don't need to continually login)

## Frequently Asked Questions

**Why do I have to enter my NEMS HWID, NEMS Cloud Services Key and Decryption Password to login? Can't you make it so I can create a username and password and login that way?**

Your NEMS HWID and NEMS Cloud Services Key authenticates your session. Your Decryption Password on the other hand allows NEMS Cloud Services to decrypt your NEMS State information. For your safety, your decryption password is never stored on the server, and only you know it. I will not create a mechanism for your private decryption password to be associated with your account. You must enter it every time you login, since it is never stored within NEMS Cloud Services.

**Since I have to enter my decryption password only once per session, how can you say it is not stored?**

When you enter your encryption password, it is transmitted over your browser's SSL connection to the NEMS Cloud Services server. There, it is not stored in the session. Rather, your password is salted and hashed, and this hash is what is stored in the session. Once your session has expired, your hash is discarded.

## 3.31 Monitoring Microsoft Windows Hosts with NEMS Linux

### 3.31.1 Monitor Windows Machines with Ping

The simplest type of check in NEMS Linux is to ping a host. The reply is binary: Host is up when ping replies, host is down when ping does not reply.

**Using a ping check is an excellent starting point for new NEMS Linux users to become familiar with an easy-to-administer check command.**

To begin pinging a Windows host, simply add it to the windows-servers group. The host check alive command will use `check_ping` to verify it as online or offline.

Most software firewalls, including the Windows [Defender] Firewall, block ping (ICMP) requests by default. So NEMS will think your host is down unless you create a rule to allow ICMP echo requests and replies through your Windows firewall.

## Windows Defender Firewall

To enable ICMP response when using Windows Defender Firewall, simply run the Command Prompt as Administrator and type the following:

IPv4 Networks:

```
netsh advfirewall firewall add rule name="ICMP Allow incoming V4 echo request" protocol=icmpv4:8,any **dir**\ =in action=allow
```

IPv6 Networks:

```
netsh advfirewall firewall add rule name="ICMP Allow incoming V6 echo request" protocol=icmpv6:8,any **dir**\ =in action=allow
```

## Third Party Firewall

If you are using a different firewall (eg., ESET Endpoint Security) you will need to consult that software's documentation to allow ICMP echo responses.

Here are some things to look for:

- Many firewalls can exempt certain IP addresses or ranges from being blocked. This is often called a Trusted Zone, or whitelisted IP. You could add your NEMS Server as a trusted device. If you do this, make absolutely certain your NEMS Server is not accessible from the world (ie., do not port forward anything to NEMS Linux), and your NEMS user password is a strong one that only you know.
- Most firewalls allow exemption of certain protocols or services. In those cases, you can simply enable/allow ping replies. It may be called "ping", "incoming ping", "ICMP Echo Reply", or similar.

**Warning:** DO NOT SIMPLY DISABLE YOUR FIREWALL. Correctly establish a firewall rule within your firewall application.

### 3.31.2 Monitor Windows Machines with NRPE

---

#### Note

NRPE is deprecated. This is not the recommended way to monitor Windows hosts. Please opt for [WMI](#) instead. NRPE documentation is here for legacy versions of NEMS Linux (ie., 1.0-1.3.x).

---

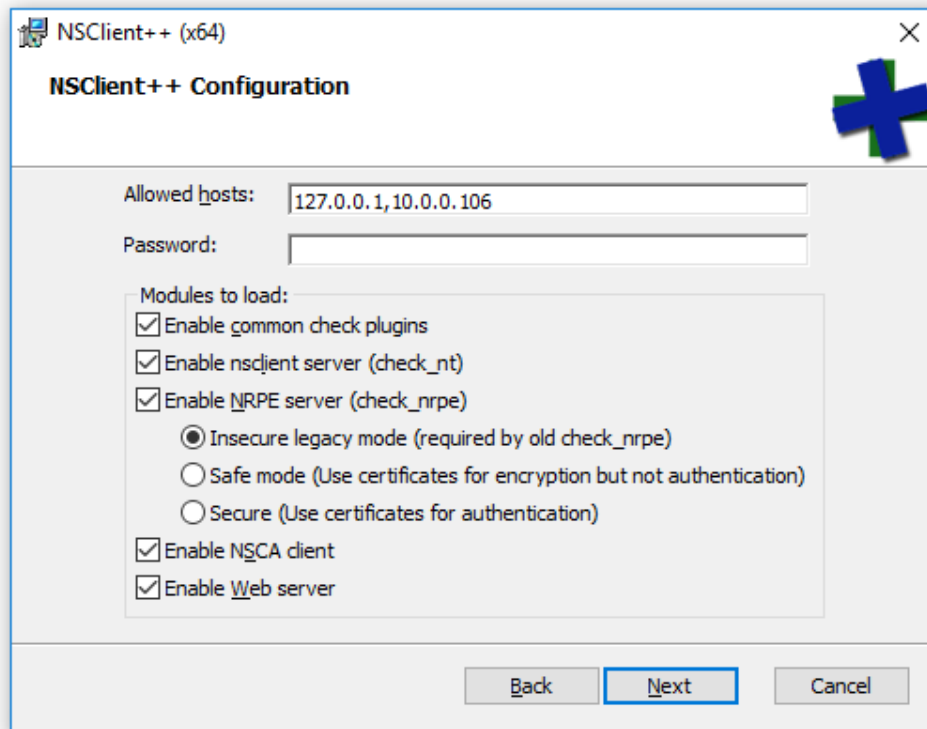
The Nagios Remote Plugin Executor (NRPE) allows your Nagios Enterprise Monitoring Server to communicate with the Linux machines on your server to determine things like free disk space, CPU load, and detect possible issues that a simple ping can't determine.

As of NEMS 1.2 NSClient++ is optional for monitoring of Windows computers (thanks to the addition of WMIC). If you'd like to use it, please follow the directions below, otherwise use the provided WMIC-based check commands.

**This is not supported on modern NEMS Linux.**

1. Grab the latest Windows client at <https://www.nsclient.org/download/>
2. Install the client with the following settings:
  - Select to install the "Generic mode" NSClient++.

- Choose “Complete installation” and if asked, choose to save config to ini file.
- Under “Allowed Hosts” it should read 127.0.0.1,*NEMSIP* (where NEMSIP is the IP address of your NEMS server)
- Clear the Password field for ease of deployment. NEMS sample scripts are setup to use NRPE without a password because I’m making the assumption that this is being deployed in a trusted LAN. If you do not blank the password here, you will have to edit all the scripts before NEMS will be able to communicate with this computer.
- Enable all modules and change the NRPE mode to Legacy.
- Screen should look a little something like this:



- Add your Windows host to NEMS.

---

**Tip: Important Firewall Note** If you have a software firewall running on your Windows machine, setup an exception for your NEMS server IP to gain access through ports 5666 and 12489.

---

And there we have it! Your NEMS Server can now check your Windows machine at a deeper level using `check_nrpe`.

## 3.32 Monitoring Linux Hosts with NEMS Linux

### 3.32.1 Monitor Linux Hosts with NRPE

**Warning:** NRPE is deprecated. Please use the more modern options provided within NEMS Linux (such as [NCPA](#)).


**Warning:** These instructions are for your Linux *client*. **NEVER** run these commands on your NEMS Server.

The Nagios Remote Plugin Executor (NRPE) allows your Nagios Enterprise Monitoring Server to communicate with the Linux machines on your server to determine things like free disk space, CPU load, and detect possible issues that a simple ping can't determine.

There are countless instructions online to download tar.gz files and install manually, or use a PPA to install via apt-get, but NEMS includes a helpful installer that will configure everything for you.

**Warning:** Do not install the NRPE plugin via software repositories as these are abandoned and lack some important functionality.

To install the needed NRPE client on Debian / Ubuntu / other Debian-based Linux operating systems (as root):

```
wget -O - https://raw.githubusercontent.com/Cat5TV/nems-admin/master/build/047-nrpe |  bash
```

**Please note:** Rudimentary support for RPM-based distros is included, but may not work. Please consider RPM support as experimental, and report your findings (especially your solutions or pull requests) so I can improve it.

Next, we just have to tell NRPE that it's allowed to communicate with our Nagios server. On the client system, open the file `/usr/local/nagios/etc/nrpe.cfg`

Find the line that reads: `allowed_hosts=127.0.0.1,::1`

Now there are a few ways we can allow our server. First (and most obvious) is to add its IP address like this:

```
allowed_hosts=127.0.0.1,::1,192.168.0.5
```

Where 192.168.0.5 is our Nagios/NEMS server.

Alternatively we can tell NRPE that it's allowed to communicate with any local system:

```
allowed_hosts=127.0.0.1,::1,192.168.0.0/24
```

Now, save the file and restart NRPE as follows:

```
systemctl restart nrpe
```

**Tip:** If you have a software firewall running on your Linux machine, setup an exception for your NEMS server IP to gain access through TCP ports 5666 and 12489.

And there we have it! Your NEMS Server can now check your Linux machine at a deeper level using [check\\_nrpe](#).

## 3.33 Upgrade NEMS Linux to Newer Version

### 3.33.1 Definition of Version Terms

#### Major Release

The whole number and first decimal place of any NEMS Linux version represents the major release. 1.7 is a major release. A major release requires reinstallation of NEMS Linux. The steps to migrate all your settings is found below.

#### Point Release

When we designate a version with an ‘x’, such as NEMS Linux 1.2.x, we mean anything within the 1.2 rolling release cycle. This means 1.2 (which can also be considered 1.2.0 if desired), 1.2.1, 1.2.2, and so-on. Point releases within the currently-running major release can be obtained with the `sudo nems-upgrade` command.

#### Summary

1.7 is a major release. 1.2.1 is the first point release of NEMS Linux’s major release 1.2.

### 3.33.2 Major Release Upgrade Instructions

These steps may be followed to upgrade from one major point release to the next major point release. For example, this process will take you from NEMS Linux 1.1 to NEMS Linux 1.2, or NEMS Linux 1.2.x to NEMS Linux 1.3, or even NEMS Linux 1.3 to 1.7, and so-on.

1. Connect to your existing NEMS Linux dashboard from your computer and press Migrator→Backup. That will give you your *backup.nems* file.
2. Deploy the latest version of NEMS Linux on a **new** card (please use a new card so you can always revert back to your existing NEMS Linux if you have a problem).
3. Boot your NEMS Server with the new NEMS Linux card and initialize it as normal (instructions are provided via your web browser when you connect).
4. Once you get to the dashboard simply go through the NEMS Migrator `nems-restore` process to restore your settings from your backup.nems file.

---

**Tip:** To save your backup.nems file to the new installation, you can browse to your home folder using samba (or `\\nems.local\home` in Windows).

---

### 3.33.3 Rolling Release Upgrade Instructions

A rolling release is a second decimal point release, and rolling upgrades are available within each major release as made available. Rolling releases are usually pushed out in order to fix bugs or improve features. An example of a rolling release would be going from NEMS Linux 1.2 to 1.2.1, or 1.2.1 to 1.2.3. Once a new NEMS Linux major release is available, it is recommended to perform a major release upgrade to ensure you have the latest and greatest NEMS Linux has to offer.

1. Login to NEMS Linux via SSH.
2. Type: `sudo nems-upgrade`

3. Reboot.

**Please Note:** If you receive a notice that `nems-upgrade` is an unknown program, did you initialize NEMS Linux? Please re-read the [Initialization Instructions](#) if that is the case.

### 3.33.4 Legacy Distro Support

Upgrade NagiosPi or NEMS 1.0 to the Latest Version of NEMS

## 3.34 NEMS Migrator: Upgrade NEMS 1.0 or nagiospi to latest NEMS

Thanks for being an early-adopter! Whether you're coming from NEMS 1.0 or its predecessor, nagiospi, I want to make it as easy as possible for you to get the latest and greatest, without having to reconfigure everything. It's been exciting to see the NEMS project really catching on, and I endeavor to make it the best it can be. Your suggestions along the way have helped me focus on some great features for as NEMS continues to evolve.

NEMS 1.1+ has a nifty backup and export tool called NEMS Migrator. While it comes pre-packaged in 1.1+, I designed it specifically to run on legacy builds as well (NEMS 1.0 or nagiospi), giving you the opportunity to export your old configuration, deploy the latest version of NEMS, and then restore the configuration to NEMS. Easy peasy!

Here's what you need to do:

---

**Note:** These instructions are for NEMS 1.0 or nagiospi only. **Do not** do this on NEMS 1.1+ as the tool is already built-in.

---

1. SSH into your legacy NEMS/nagiospi server.
2. Become root:

```
sudo su
```

3. Update repository data.

```
apt-get update
```

4. Install Git.

```
apt-get install git
```

5. Install NEMS-Migrator in /tmp.

```
cd /tmp && git clone https://github.com/Cat5TV/nems-migrator
```

6. Create the backup config on your NEMS/nagiospi system.

If on NEMS 1.0:

```
cd /tmp/nems-migrator && ./backup.sh
```

If on nagiospi:

```
cd /tmp/nems-migrator && ./nagiospi2nems.sh
```

7. Download the backup to your computer by opening it in your web browser. In your favorite web browser, simply add /backup/ to the end of your NEMS/nagiospi server address. E.G., <http://10.0.0.5/backup/>
8. Now that you have your backup.nems file, follow [the instructions to restore your configuration to a new version of NEMS](#).

## 3.35 SNMP

The Simple Network Management Protocol (SNMP) allows you to collect and monitor information about your SNMP enabled device. This protocol was created as a way to gather information from various systems and equipment in a uniform manner.

### 3.35.1 Basics

The SNMP hierarchy consists of multiple tables called Management Information Bases (MIBs). MIBs can follow universal standards (i.e. RFC 1441-see links below for more information) or vendor defined. Each MIB consists of one or more nodes which can represent an individual device or device component. Each node has a unique Object Identifier (OID). SNMP monitoring works by the monitoring host (in this case, NEMS) connects to the SNMP agent on the remote device. Version.

There are currently three version of the SNMP protocol (1v, c2v, and 3v). It is up to the device manufacturer which one is implemented.

#### Community

This is the security method SNMP. The basic types of communities are read-only (normally public), read-write (defined in the device configuration), and trap (also defined in configuration). Each community has a name defined in the device configuration.

#### Presteps

Ensure that the device to be monitored is SNMP capable and SNMP is configured.

The screenshot shows the TP-Link TD-W9970 web interface. The browser address bar shows the IP address 192.168.254.251. The page title is "TP-LINK" and the subtitle is "300Mbps Wireless N USB VDSL/ADSL Modem Router Model No. TD-W9970". The left sidebar contains a list of navigation options: Status, Quick Setup, Operation Mode, Network, DHCP Server, Wireless, Guest Network, USB Settings, Route Settings, IPv6 Route Settings, Forwarding, Parent Control, Firewall, IPv6 Firewall, IPv6 Tunnel, Bandwidth Control, IP & MAC Binding, Dynamic DNS, Diagnostic, System Tools, System Log, Time Settings, Manage Control, CWM Settings, **SNMP Settings**, Backup & Restore, Factory Defaults, Firmware Upgrade, Reboot, Statistics, and Logout. The main content area is titled "SNMP Settings" and contains the following text: "Simple Network Management Protocol(SNMP) allows management applications to retrieve status updates and statistics from the SNMP agent within this device." Below this text are the following settings:

- SNMP Agent: ☐ Disable @ ☒ Enable
- Read Community:
- Set Community:
- System Name:
- System Description:
- System Location:
- System Contact:
- Trap Manager IP:

At the bottom of the settings area is a "Save" button.

### 3.35.2 MIB/OID

Vendor sites and community forums may provide the MIBs and OIDs, but there are ways of retrieving information from the device itself.

#### Method 1 - MIB Browser

There are several free MIB browsers available on the internet. Most provide a tree structure of the device.

The screenshot shows the OidView Enterprise application window. The main pane displays a MIB tree structure for the session 192.168.254.251. The tree is rooted at SMI and branches into various MIBs, including sysDescr, sysObjectID, sysUpTime, sysContact, sysName, sysLocation, sysServices, sysORLastChange, sysORID, sysORDescr, and sysORUpTime. The 'system' MIB is selected, and its details are shown in the 'LiveGrid: system' pane. The LiveGrid displays a table of system information:

OID	Object	Type	Value
1.3.6.1.2.1.1.0	sysDescr	OCTET-STRING	0.9.1.2.5 v0025.0 Build 150917 Rel.56865n
1.3.6.1.2.1.1.2.0	sysObjectID	OID	1.3.6.1.4.1.16972
1.3.6.1.2.1.1.3.0	sysUpTime	TIMETICKS	9/23/2020 7:01:50 AM
1.3.6.1.2.1.1.4.0	sysContact	OCTET-STRING	unknown
1.3.6.1.2.1.1.5.0	sysName	OCTET-STRING	TD-w9970
1.3.6.1.2.1.1.6.0	sysLocation	OCTET-STRING	unknown
1.3.6.1.2.1.1.7.0	sysServices	INTEGER	72
1.3.6.1.2.1.1.8.0	sysORLastChange	TIMETICKS	9/26/2020 4:55:49 AM
1.3.6.1.2.1.1.9.1.2.1	sysORID	OID	1.3.6.1.6.3.1
1.3.6.1.2.1.1.9.1.3.1	sysORDescr	OCTET-STRING	The MIB module for SNMPv2 entities
1.3.6.1.2.1.1.9.1.4.1	sysORUpTime	TIMETICKS	9/26/2020 4:55:49 AM

The 'Variable Grid' pane shows a list of objects and their OIDs, sorted by object. The 'MIB Info' pane shows the description for the selected MIB, which is 'system'.

#### Method 2 - snmpwalk

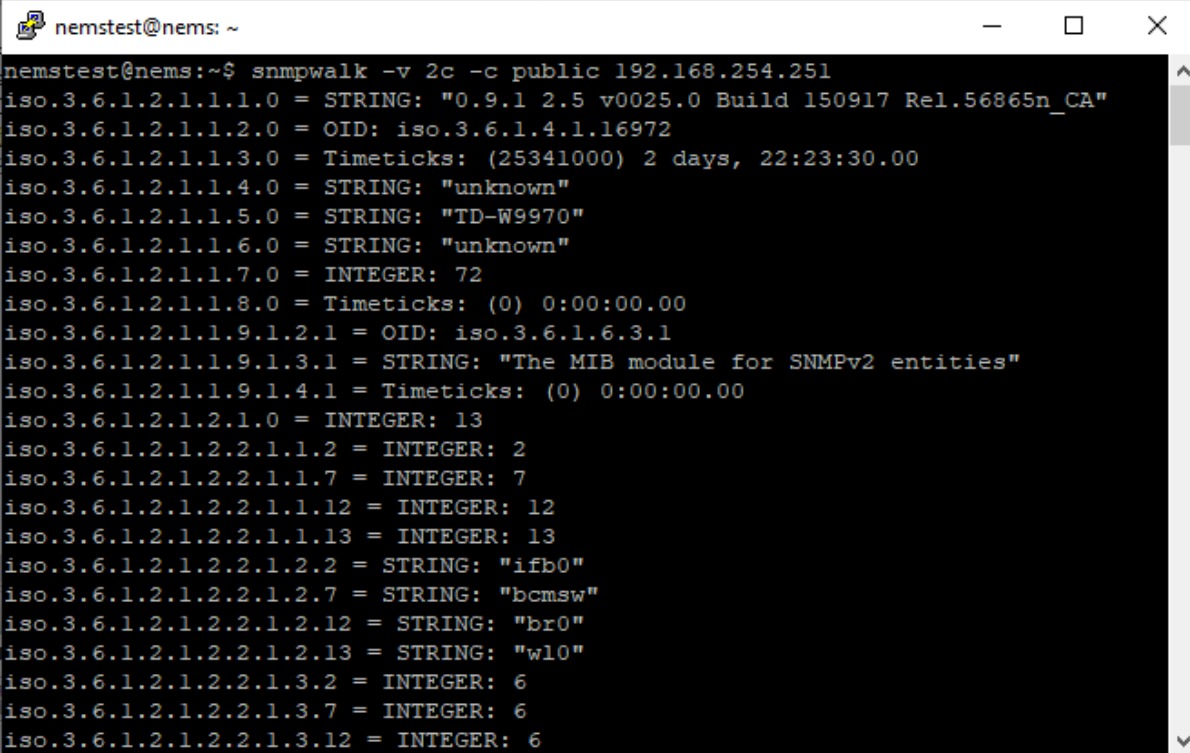
snmpwalk will connect to the remote device and retrieve a list of OIDs. Open a NEMS ssh session.

The command format is

```
snmpwalk -v *snmp version* -c *snmp community* *IP*
```

A list of OIDs will be returned. Copy the results to a text document or spreadsheet for easy reference.





```
nemstest@nems: ~
nemstest@nems:~$ snmpwalk -v 2c -c public 192.168.254.251
iso.3.6.1.2.1.1.1.0 = STRING: "0.9.1 2.5 v0025.0 Build 150917 Rel.56865n_CA"
iso.3.6.1.2.1.1.1.2.0 = OID: iso.3.6.1.4.1.16972
iso.3.6.1.2.1.1.1.3.0 = Timeticks: (25341000) 2 days, 22:23:30.00
iso.3.6.1.2.1.1.1.4.0 = STRING: "unknown"
iso.3.6.1.2.1.1.1.5.0 = STRING: "TD-W9970"
iso.3.6.1.2.1.1.1.6.0 = STRING: "unknown"
iso.3.6.1.2.1.1.1.7.0 = INTEGER: 72
iso.3.6.1.2.1.1.1.8.0 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.2.1 = OID: iso.3.6.1.6.3.1
iso.3.6.1.2.1.1.9.1.3.1 = STRING: "The MIB module for SNMPv2 entities"
iso.3.6.1.2.1.1.9.1.4.1 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.2.1.0 = INTEGER: 13
iso.3.6.1.2.1.2.2.1.1.2 = INTEGER: 2
iso.3.6.1.2.1.2.2.1.1.7 = INTEGER: 7
iso.3.6.1.2.1.2.2.1.1.12 = INTEGER: 12
iso.3.6.1.2.1.2.2.1.1.13 = INTEGER: 13
iso.3.6.1.2.1.2.2.1.2.2 = STRING: "ifb0"
iso.3.6.1.2.1.2.2.1.2.7 = STRING: "bcm5703"
iso.3.6.1.2.1.2.2.1.2.12 = STRING: "br0"
iso.3.6.1.2.1.2.2.1.2.13 = STRING: "wl0"
iso.3.6.1.2.1.2.2.1.3.2 = INTEGER: 6
iso.3.6.1.2.1.2.2.1.3.7 = INTEGER: 6
iso.3.6.1.2.1.2.2.1.3.12 = INTEGER: 6
```

### 3.35.3 Testing with ./check\_snmp

From the NEMS ssh session, navigate to /usr/local/nagios/libexec. The command is check\_snmp and the options are:

Usage:

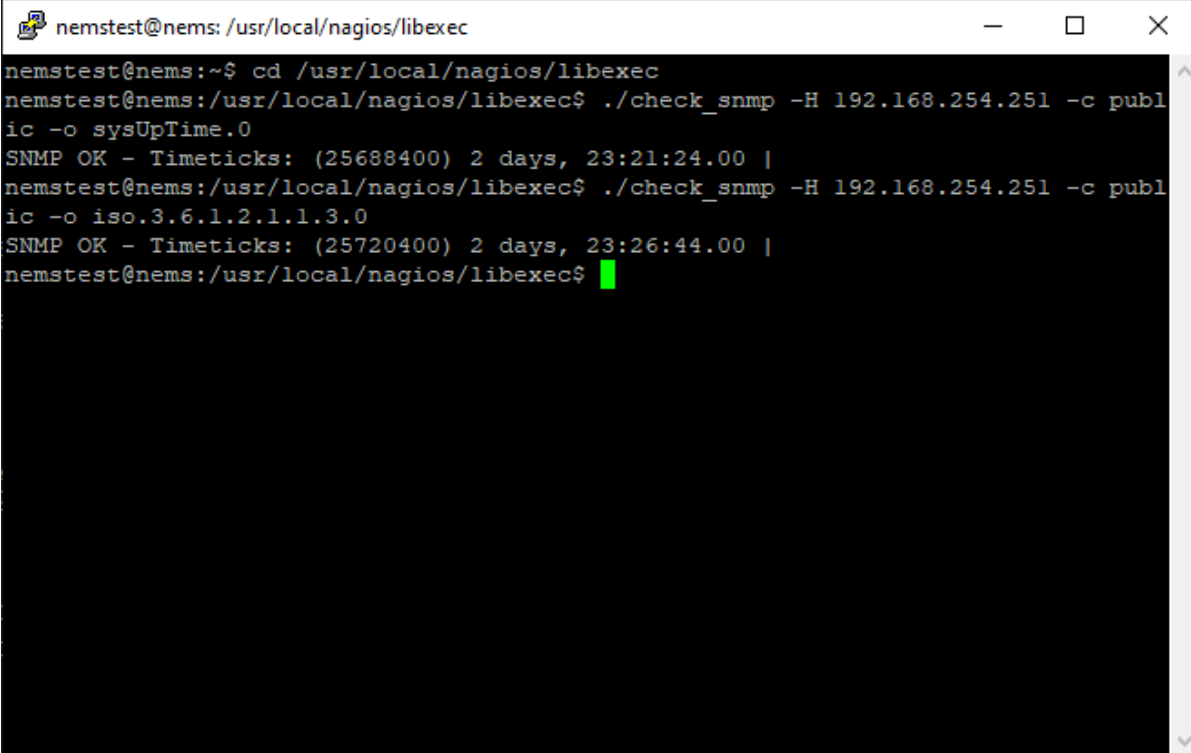
```
check_snmp -H <ip_address> -o [-w warn_range] [-c crit_range]
[-C community] [-s string] [-r regex] [-R regexi] [-t timeout] [-e retries]
[-l label] [-u units] [-p port-number] [-d delimiter] [-D output-delimiter]
[-m miblist] [-P snmp version] [-N context] [-L seclevel] [-U secname]
[-a authproto] [-A authpasswd] [-x privproto] [-X privpasswd] [-4|6]
```

In this example, uptime will be test using both the OID and Object name using the command format:

```
*./checksnpmp -H remote ip -C SNMP community -o OID or Object*
```

In this example, System Uptime is check using both the OID and the Object name. Note, the .0 is put on the end of sysUpTime, this denotes to collect child information and is required.

Both work and return the same information. Depending on the device and which MIB it uses (standard or vendor) will dictate which one is used.

A terminal window titled 'nemstest@nems: /usr/local/nagios/libexec' with standard window controls. The terminal shows the following commands and output:

```
nemstest@nems:~$ cd /usr/local/nagios/libexec
nemstest@nems:/usr/local/nagios/libexec$ ./check_snmp -H 192.168.254.251 -c public -o sysUpTime.0
SNMP OK - Timeticks: (25688400) 2 days, 23:21:24.00 |
nemstest@nems:/usr/local/nagios/libexec$ ./check_snmp -H 192.168.254.251 -c public -o iso.3.6.1.2.1.1.3.0
SNMP OK - Timeticks: (25720400) 2 days, 23:26:44.00 |
nemstest@nems:/usr/local/nagios/libexec$
```

### 3.35.4 NEMS Check

Follow these steps to configure a basic SNMP check.

- Launch NEMS Configurator (NConf)
- Click on Add for Advanced Services
- Enter most fields according to environment standards (i.e. name, description, check/notifications periods, etc)
- Select `check_snmp` in the check command field
- For ARG1 at the bottom of the screen, at the minimum, enter the community and object to check. `-C public -o iso.3.6.1.2.1.1.3.0` You can add warnings `-w` and criticals `-c` if the appropriate
- Add new check to host
- Generate and deploy Nagios config

Launch Adagios and validate the check ran with no issues.

### 3.35.5 Links

- <https://www.rfc-editor.org/>
- <http://www.oid-info.com/index.htm>
- <http://www.ireasoning.com/mibbrowser.shtml>
- <http://www.oidview.com/mibs/detail.html>

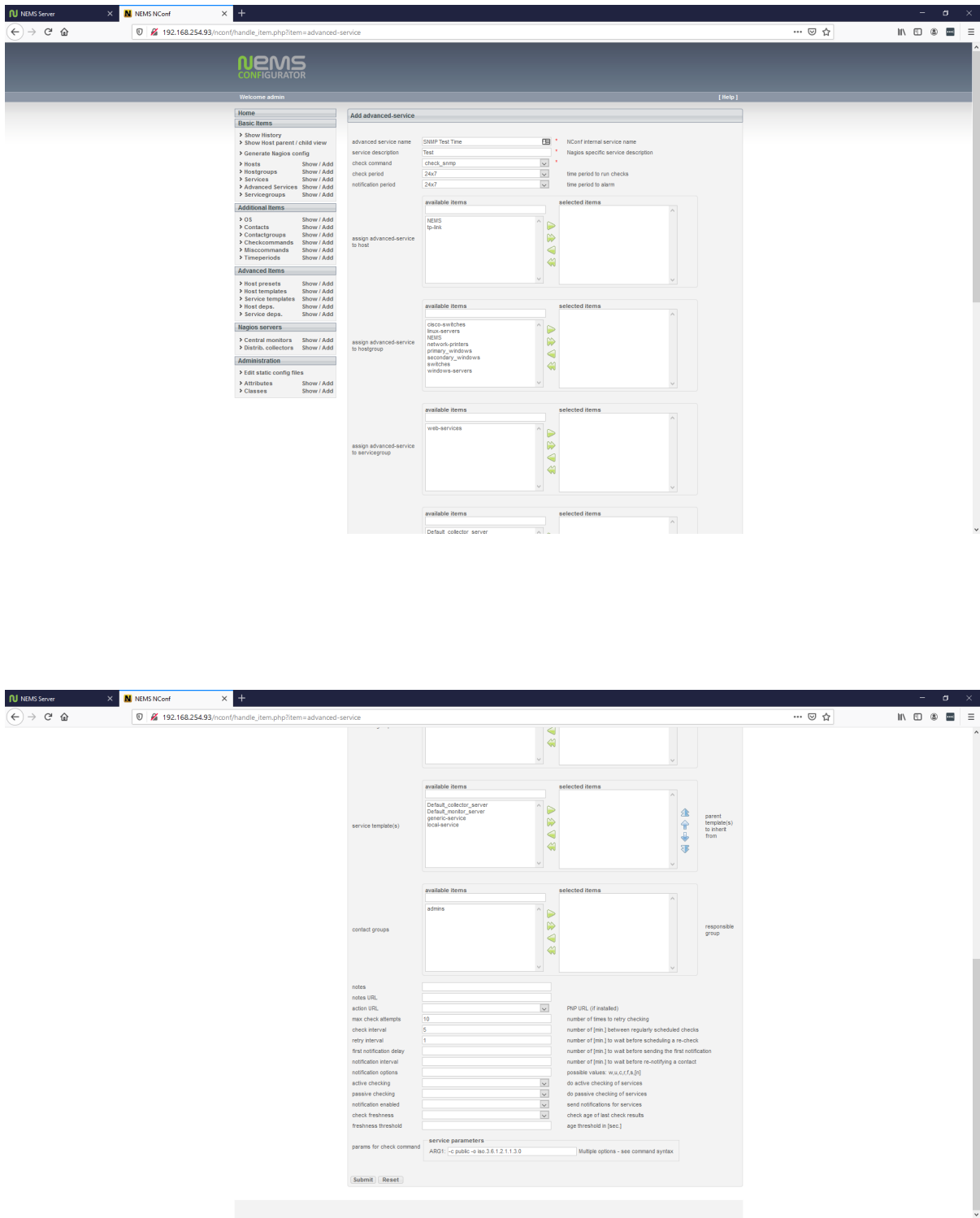


Fig. 11: Note the service parameters.

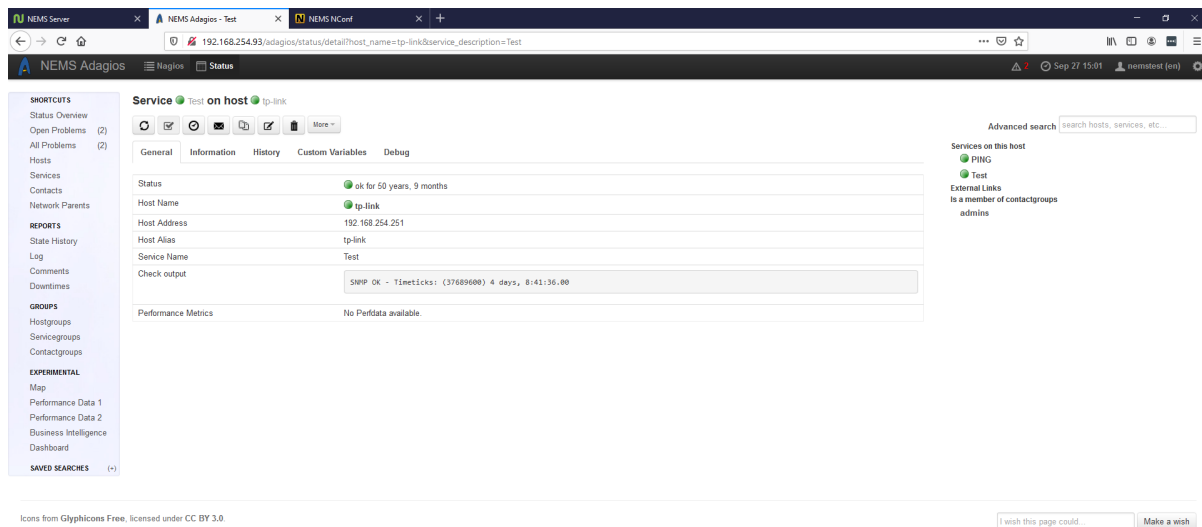


Fig. 12: Adagios shows our SNMP check is running perfectly.

## 3.36 Multi Router Traffic Grapher (MRTG)

MRTG generates traffic graphs for your SNMP-enabled router. If your router does not support SNMP, take a look at [MikroTik devices](#), which offer enterprise-grade routers at consumer prices.

MRTG is pre-installed in NEMS Linux 1.6+ and is available in NEMS Linux 1.3.x-1.5.x by way of a NEMS Update.

### 3.36.1 MikroTik

To enable SNMP on MikroTik RouterOS devices:

1. In WinBox, open a new terminal.
2. Type `/snmp`
3. Type `set enabled yes`

If desired, you can add your contact name and location to your SNMP Trap in the MikroTik WebFig / WinBox under IP → SNMP.

### 3.36.2 Configuration

In time, MRTG configuration will become part of NEMS SST. For the time being, a command must be entered in your NEMS Linux terminal in order to connect MRTG to your router:

```
sudo mrtgsetup
```

Simply enter your router's IP address and your trap community name when prompted, and the script will generate your MRTG config, activate it, and provide you with all links for detected SNMP sources on your router.

**Please Note:** `mrtgsetup` only supports the default `public` trap community at this time.

### 3.36.3 Check Command

NEMS Linux includes *check\_mrtgtraf\_nems*. This is a custom wrapper for *check\_mrtgtraf.py* which is itself a reworking of *check\_mrtgtraf*.

You must first run *mrtgsetup*, otherwise the check command will have no data to work with.

Beyond the standard graphs that MRTG provides, you may also add the *check\_mrtgtraf\_nems* check command to your SNMP-capable router/switch host(s) and monitor any or all of their ports. For example, to monitor the Internet bandwidth usage, you can add a check command to monitor your WAN port. Then, if you want to monitor the bandwidth usage for a specific Ethernet port (for example, an employee or server who is known to use a lot of bandwidth), you can do so by adding a second check command to that host which targets the Ethernet port that user is connected to. This can also be handy for zoned networks where, for example, you may want to monitor the bandwidth usage of specific departments by connecting their uplink to a specific port on your monitored device. You may also use this to monitor the amount of bandwidth being consumed by users on your guest WiFi, and then notify the admin if the set thresholds are exceeded.

Service Bandwidth Usage on host Router

More ▾

General Information History Custom Variables Debug

Status	ok for 33 minutes
Host Name	Router
Host Address	10.0.0.1
Host Alias	Router
Service Name	Bandwidth Usage
Check output	Traffic OK: AVG. in = 4.62343597412109 Mbyte/s, AVG. out = 1.02823638916016 Mbyte/s.

### Check Command Arguments

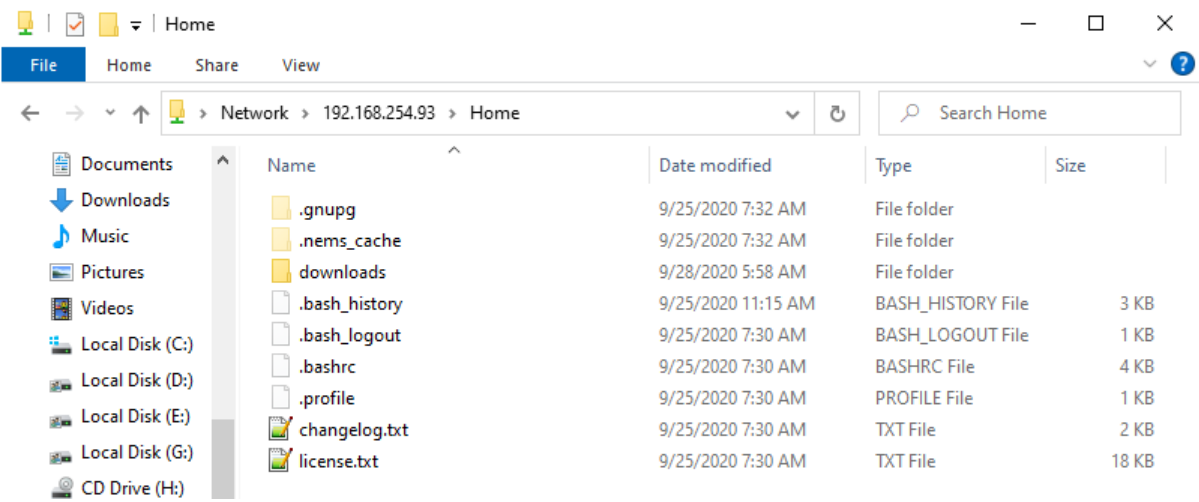
Add the *check\_mrtgtraf\_nems* check command to your router host in NEMS NConf, using the following arguments:

- **Port** - This is the number associated with the SNMP. This will not correspond with traditional conventions, but rather is generated by the SNMP OID. You can find the number on the corresponding MRTG page following the header “Traffic Analysis for”. The number corresponds to the port to use for *check\_mrtgtraf\_nems*. You can find the MRTG graphs at <https://nems.local/mrtg/>
- **Multiplier** - What measurement you’d like to use for your thresholds. Available options are gb, mb or kb.
- **Warn Up / Warn Down / Critical Up / Critical Down** - Set your thresholds. Can be a positive floating point number or integer.











### 3.37 Copying OS Icons to NEMS

To add additional or custom OS icons to NEMS, perform the following:

From a file browser window, open the Home share on the NEMS server (i.e. \nems\_ip\Home) and log in with your NEMS credentials.



Create a folder. In this example, it is “icons”

	.gnupg	9/25/2020 7:32 AM
	.nems_cache	9/25/2020 7:32 AM
	downloads	9/28/2020 5:58 AM
	.bash_history	9/25/2020 11:15 AM
	.bash_logout	9/25/2020 7:30 AM
	.bashrc	9/25/2020 7:30 AM
	.profile	9/25/2020 7:30 AM
	changelog.txt	9/25/2020 7:30 AM
	license.txt	9/25/2020 7:30 AM
	icons	10/21/2020 4:11 PM

Start an SSH session to the NEMS server and login. The “icons” folder is displayed when ls is run. Copy the icon images into this new folder.

“cd” into the “icons” folder and run ls to verify the icon files are there. Copy the icon images into this new folder.

The icon file can be copied or moved into the /var/www/nconf/img/logos/base folder.

```

nemstest@nems: ~
NEMS
LINUX

BY: ROBBIE FERGUSON
NEMSLINUX.COM

NEMS Platform....: Virtual Machine
NEMS Version.....: 1.5 (Current Version is 1.5)
NEMS IP Address...: 192.168.254.93
Uptime.....: 0 days 0 hours 13 minutes 57 seconds
Load.....: 0.08 (1 minute) 0.10 (5 minutes) 0.09 (15 minutes)
              0.01 (1 week)
Memory.....: Total: 3947 MB / Cached: 2047 MB
              Used: 350 MB / Free: 3102 MB
Disk Usage.....: You're using 10% of your root filesystem

nemstest@nems:~$ ls
changelog.txt  downloads  icons  license.txt
nemstest@nems:~$

```

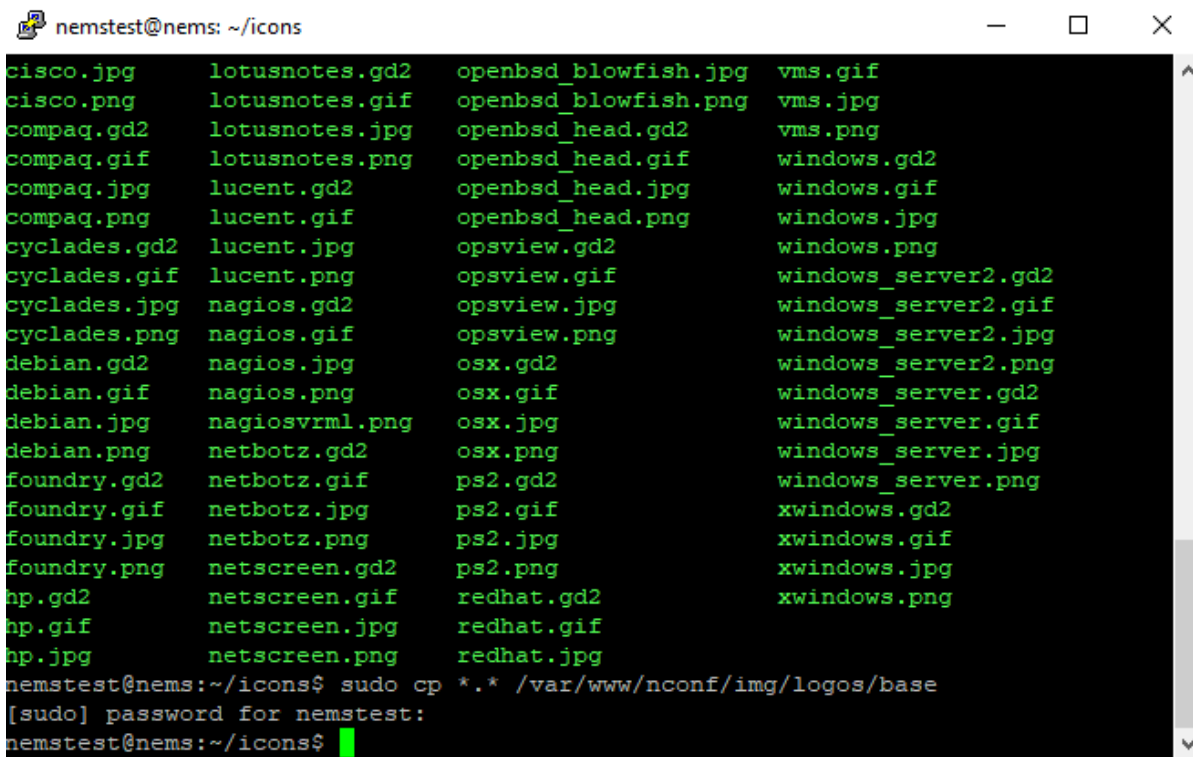
```

nemstest@nems: ~/icons
cisco.gd2      lotusnotes2.jpg  openbsd_blowfish.gd2  turbolinux.png
cisco.gif      lotusnotes2.png  openbsd_blowfish.gif  vms.gd2
cisco.jpg      lotusnotes.gd2   openbsd_blowfish.jpg  vms.gif
cisco.png      lotusnotes.gif   openbsd_blowfish.png  vms.jpg
compaq.gd2     lotusnotes.jpg   openbsd_head.gd2      vms.png
compaq.gif     lotusnotes.png   openbsd_head.gif      windows.gd2
compaq.jpg     lucent.gd2       openbsd_head.jpg      windows.gif
compaq.png     lucent.gif       openbsd_head.png      windows.jpg
cyclades.gd2   lucent.jpg       opsview.gd2           windows.png
cyclades.gif   lucent.png       opsview.gif           windows_server2.gd2
cyclades.jpg   nagios.gd2       opsview.jpg           windows_server2.gif
cyclades.png   nagios.gif       opsview.png           windows_server2.jpg
debian.gd2     nagios.jpg       osx.gd2               windows_server2.png
debian.gif     nagios.png       osx.gif               windows_server.gd2
debian.jpg     nagiosvrm1.png   osx.jpg               windows_server.gif
debian.png     netbotz.gd2      osx.png               windows_server.jpg
foundry.gd2    netbotz.gif      ps2.gd2               windows_server.png
foundry.gif    netbotz.jpg      ps2.gif               xwindows.gd2
foundry.jpg    netbotz.png      ps2.jpg               xwindows.gif
foundry.png    netscreen.gd2    ps2.png               xwindows.jpg
hp.gd2         netscreen.gif    redhat.gd2            xwindows.png
hp.gif         netscreen.jpg    redhat.gif
hp.jpg         netscreen.png    redhat.jpg
nemstest@nems:~/icons$

```

To copy `sudo cp *.* /var/www/nconf/img/logos/base`

To move `sudo mv *.* /var/www/nconf/img/logos/base`



```
nemstest@nems: ~/icons
cisco.jpg      lotusnotes.gd2  openbsd_blowfish.jpg  vms.gif
cisco.png      lotusnotes.gif  openbsd_blowfish.png  vms.jpg
compaq.gd2     lotusnotes.jpg  openbsd_head.gd2      vms.png
compaq.gif     lotusnotes.png  openbsd_head.gif      windows.gd2
compaq.jpg     lucent.gd2      openbsd_head.jpg      windows.gif
compaq.png     lucent.gif      openbsd_head.png      windows.jpg
cyclades.gd2   lucent.jpg      opsview.gd2           windows.png
cyclades.gif   lucent.png      opsview.gif           windows_server2.gd2
cyclades.jpg   nagios.gd2      opsview.jpg           windows_server2.gif
cyclades.png   nagios.gif      opsview.png           windows_server2.jpg
debian.gd2     nagios.jpg      osx.gd2               windows_server2.png
debian.gif     nagios.png      osx.gif               windows_server.gd2
debian.jpg     nagiosvrml.png  osx.jpg               windows_server.gif
debian.png     netbotz.gd2     osx.png               windows_server.jpg
foundry.gd2    netbotz.gif     ps2.gd2               windows_server.png
foundry.gif    netbotz.jpg     ps2.gif               xwindows.gd2
foundry.jpg    netbotz.png     ps2.jpg               xwindows.gif
foundry.png    netscreen.gd2   ps2.png               xwindows.jpg
hp.gd2         netscreen.gif   redhat.gd2            xwindows.png
hp.gif         netscreen.jpg   redhat.gif
hp.jpg         netscreen.png   redhat.jpg
nemstest@nems:~/icons$ sudo cp *.* /var/www/nconf/img/logos/base
[sudo] password for nemstest:
nemstest@nems:~/icons$
```

Login to nconf for NEMS. In this example, the icon for “switch” will be changed.

Click on Show for OS, then click on edit (pencil icon) for the OS to be modified.

Edit the gif name to the new icon name and click Submit.

Go back to Hosts and verify the icon has changed.

Nagios icon packs <https://exchange.nagios.org/directory/Graphics-and-Logos/Images-and-Logos>

### 3.38 nems-api

*nems-api* is a web-based api interface that outputs json data related to your NEMS server. It is lightweight, fast, and offers a connection for both internal NEMS features and third-party devices.

*nems-api* will always return either *success: true* or *success: false* to tell you whether a query was successful or not.



Welcome admin [ Help ]

**Home**

**Basic Items**

- › Show History
- › Show Host parent / child view
- › Generate Nagios config
- › Hosts Show / Add
- › Hostgroups Show / Add
- › Services Show / Add
- › Advanced Services Show / Add
- › Servicegroups Show / Add

**Additional Items**

- › OS Show / Add
- › Contacts Show / Add
- › Contactgroups Show / Add
- › Checkcommands Show / Add
- › Misccommands Show / Add
- › Timeperiods Show / Add

**Advanced Items**

- › Host presets Show / Add
- › Host templates Show / Add
- › Service templates Show / Add
- › Host deps. Show / Add
- › Service deps. Show / Add

**Nagios servers**

- › Central monitors Show / Add
- › Distrib. collectors Show / Add

**Administration**

- › Edit static config files
- › Attributes Show / Add
- › Classes Show / Add

**Show: host** ▶ Advanced

Search filter OS Search

**Overview** Entries 1 - 2 of 2 25 50 100 all

hostname	address	monitored by	OS	[ actions ]
NEMS	127.0.0.1	Default Nagios	Linux	
tp-link	192.168.254.251	Default Nagios	Switch	

**NEMS CONFIGURATOR**

Welcome admin [\[ Help \]](#)

**Home**

**Basic Items**

- › Show History
- › Show Host parent / child view
- › Generate Nagios config
- › Hosts Show / Add
- › Hostgroups Show / Add
- › Services Show / Add
- › Advanced Services Show / Add
- › Servicegroups Show / Add

**Additional Items**

- › OS Show / Add
- › Contacts Show / Add
- › Contactgroups Show / Add
- › Checkcommands Show / Add
- › Misccommands Show / Add
- › Timeperiods Show / Add

**Advanced Items**

- › Host presets Show / Add
- › Host templates Show / Add
- › Service templates Show / Add
- › Host deps. Show / Add
- › Service deps. Show / Add

**Nagios servers**

- › Central monitors Show / Add
- › Distrib. collectors Show / Add

**Administration**

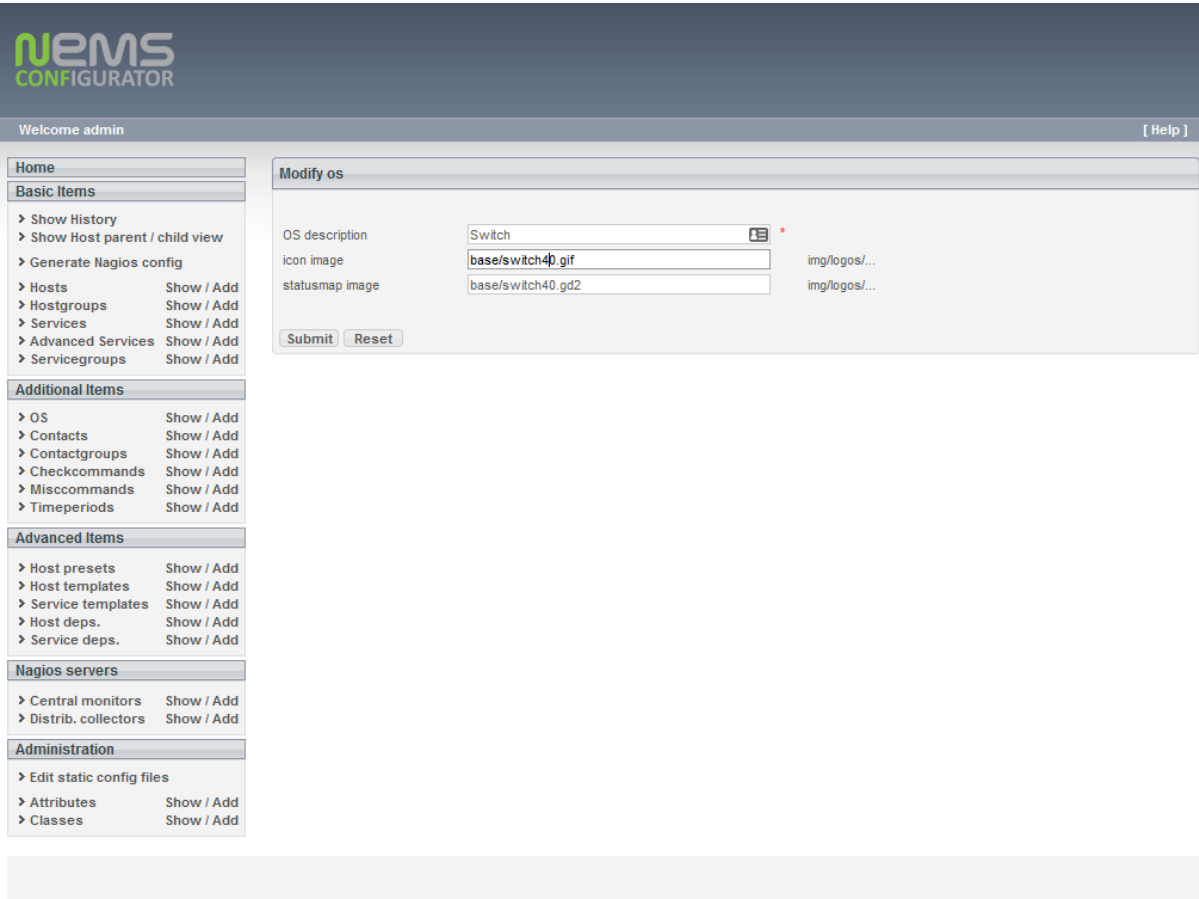
- › Edit static config files
- › Attributes Show / Add
- › Classes Show / Add

**Show: os**

Searchfilter

**Overview** Entries 1 - 8 of 8 25 50 100 all

OS description	[ actions ]
Free BSD	
HP Printer	
HP Unix	
Linux	
Router	
Sun Solaris	
Switch	
Windows Server	



**nems**  
CONFIGURATOR

Welcome admin

[ Help ]

Home

Basic Items

› Show History

› Show Host parent / child view

› Generate Nagios config

› Hosts Show / Add

› Hostgroups Show / Add

› Services Show / Add

› Advanced Services Show / Add

› Servicegroups Show / Add

Additional Items

› OS Show / Add

› Contacts Show / Add

› Contactgroups Show / Add

› Checkcommands Show / Add

› Misccommands Show / Add

› Timeperiods Show / Add

Advanced Items

› Host presets Show / Add

› Host templates Show / Add

› Service templates Show / Add

› Host deps. Show / Add

› Service deps. Show / Add

Nagios servers

› Central monitors Show / Add

› Distrib. collectors Show / Add

Administration

› Edit static config files

› Attributes Show / Add

› Classes Show / Add

Modify os

OS description

Switch

img

icon image

base/ps2.gif

img/logos/...

statusmap image

base/ps2gd2

img/logos/...

Submit

Reset

Welcome admin [ Help ]

**Home**

**Basic Items**

- Show History
- Show Host parent / child view
- Generate Nagios config
- Hosts Show / Add
- Hostgroups Show / Add
- Services Show / Add
- Advanced Services Show / Add
- Servicegroups Show / Add

**Additional Items**

- OS Show / Add
- Contacts Show / Add
- Contactgroups Show / Add
- Checkcommands Show / Add
- Misccommands Show / Add
- Timeperiods Show / Add

**Advanced Items**

- Host presets Show / Add
- Host templates Show / Add
- Service templates Show / Add
- Host deps. Show / Add
- Service deps. Show / Add

**Nagios servers**

- Central monitors Show / Add
- Distrib. collectors Show / Add

**Administration**

- Edit static config files
- Attributes Show / Add
- Classes Show / Add

**Show: host** ▶ Advanced

Search filter OS

**Overview** Entries 1 - 2 of 2 25 50 100 all

hostname	address	monitored by	OS	[ actions ]
NEMS	127.0.0.1	Default Nagios	Linux	
tp-link	192.168.254.251	Default Nagios	Switch	

### 3.38.1 IP Restrictions

By default, access to *nems-api* is limited to the following IP addresses:

- 127.0.0.1
- 10.0.0.0 - 10.255.255.255
- 172.16.0.0 - 172.31.255.255
- 192.168.0.0 - 192.168.255.255

If you need to add an outside IP address, please put in a feature request in the Community Forum to add this feature to NEMS-SST. If there is demand for it, it will be added.

### 3.38.2 Secure Certificate

NEMS Linux uses self-signed certificates. In order to pull *nems-api* data over ssl (ie., https), you must ignore the certificates via your application.

### 3.38.3 Command Examples

All examples in this document assume that the API is available at

```
http://nems.local/nems-api/
```

- <http://nems.local/nems-api/hosts> - output all configured hosts
- <http://nems.local/nems-api/services> - output all configured services
- <http://nems.local/nems-api/downtimes> - output all scheduled downtimes
- <http://nems.local/nems-api/hosts?Columns=name,state> - output all host names along with their current state

### Response Format

All responses are in JSON and have the following format:

```
{"success": <bool>, "content": <object>}
```

If “success” is true, “content” will contain the requested data. If false, it will contain

```
{"code": <int>, "message": <string>}
```

where “code” is the mk-livestatus error code and “message” is a human-readable explanation of the error.

## Query interface

The query interface returns a list of objects in JSON. The available endpoints are the same as the tables available from mk-livestatus itself:

- hosts
- services - Nagios services, joined with all data from hosts
- hostgroups
- servicegroups
- contactgroups
- servicesbygroup - all services grouped by service groups
- servicesbyhostgroup - all services grouped by host groups
- hostsbygroup - all hosts grouped by host groups
- contacts
- commands - your defined Nagios commands
- timeperiods - time period definitions (currently only name and alias)
- downtimes - all scheduled host and service downtimes, joined with data from hosts and services.
- comments - all host and service comments
- log - a transparent access to the nagios logfiles
- status - general performance and status information. This table contains exactly one dataset.
- columns - a complete list of all tables and columns available via Livestatus, including descriptions!
- statehist - sla statistics for hosts and services, joined with data from hosts, services and log.

To retrieve all records from a table, send a GET request to

```
http://nems.local/nems-api/{tablename}
```

For example, to get all host records from the server, GET

```
http://nems.local/nems-api/hosts
```

### 3.38.4 Columns

To limit the returned data to a subset of the available fields, pass a Columns query parameter containing a comma-separated list of column names. To fetch the name and services list for all hosts:

```
http://nems.local/nems-api/hosts?Columns=name,services
```

### 3.38.5 Filters

To filter the result set to records meeting some criteria, pass one or more `Filter[]` params. Each `Filter` is a urlencoded LQL filter (see the [mk-livestatus documentation]([http://mathias-kettner.com/checkmk\\_livestatus.html#H1:LQL](http://mathias-kettner.com/checkmk_livestatus.html#H1:LQL) - The Livestatus Query Language) for detailed LQL filter syntax). If more than one filter is specified, they are ANDed together. To get all hosts starting with “api” in state OK (0):

```
http://nems.local/nems-api/hosts?Filter[]=name - ^api&Filter[]=state = 0
```

### 3.38.6 Stats

Stats queries allow you to get a count of objects matching a criteria. Stats queries return a list of counts and never take a `Columns` parameter. You can request several Stats with a single API call. You can also restrict the objects counted by adding Filters to your query. To count the number of hosts starting with “api” in state OK:

```
http://nems.local/nems-api/hosts?&Stats[]=name - ^api&Filter[]=state = 0
```

## Command Interface

All calls to `nems-api` to execute Nagios commands **must be HTTP POST requests**.

### 3.38.7 Acknowledgements

Acknowledgements for host and service alerts can be sent via the `acknowledge_problem` endpoint.

#### Acknowledge Host Alerts

```
curl -is -XPOST https://nems.local/nems-api/acknowledge_problem -d '{"host": "host.  
↪example.com", "author": "rfrantz", "comment": "acked from livestatus"}'
```

#### Acknowledge Service Alerts

Acknowledging service alerts is similar to host alerts, with the addition of the `service` parameter:

```
curl -is -XPOST https://nems.local/nems-api/acknowledge_problem -d '{"host": "host.  
↪example.com", "service": "Apache", "author": "rfrantz", "comment": "acked from_  
↪livestatus"}'
```

### 3.38.8 Downtime

#### cancel\_downtime

Existing scheduled downtimes for a host can be canceled. `cancel_downtime` expects the `downtime_id` parameter. Downtime IDs can be found by querying a host and extracting the `downtimes` array:



```
curl -s https://nems.local/nems-api/hosts?Filter=name = my_host | jq '.' | grep
↪ 'downtimes"' -A 2

"downtimes": [
  12345
],
```

The subsequent request to cancel the host's downtime is:

```
curl -s -XPOST 'https://nems.local/nems-api/cancel_downtime' -d '{"downtime_id": "12345"}'
↪ '
```

To cancel the downtime for a service, pass the name of the service along with the downtime\_id:

```
curl -s -XPOST 'https://nems.local/nems-api/cancel_downtime' -d '{"downtime_id": "12345",
↪ "service": "CPU"}'
```

### schedule\_downtime

Schedule downtime for a host as follows:

```
curl -s -XPOST 'https://nems.local/nems-api/schedule_downtime' -d '{"host": "host.
↪ example.com", "duration": "7200", "author": "rfrantz", "comment": "Downtimed via_
↪ livestatus"}'
```

**NOTE:** The duration field expects a value whose unit is in seconds.

Downtimes can be scheduled for a particular service by adding a "service" parameter:

```
curl -s -XPOST 'https://nems.local/nems-api/schedule_downtime' -d '{"host": "host.
↪ example.com", "service": "CPU", "duration": "7200", "author": "rfrantz", "comment":
↪ "Downtimed via livestatus"}'
```

## 3.38.9 Notifications

### disable\_notifications

Notifications for a host, a host's service, or all of the host's services can be disabled via the `disable_notifications` endpoint.

#### Disable Host Notifications

Send a request that includes a valid 'host' value:

```
curl -s -XPOST 'https://nems.local/nems-api/disable_notifications' -d '{"host": "host.
↪ example.com"}'
```

### Disable Notifications for a Host's Service

Send a request that includes valid 'host' and 'service' values:

```
curl -s -XPOST 'https://nems.local/nems-api/disable_notifications' -d '{"host": "host.  
↪example.com", "service": "httpd"}'
```

### Disable Notifications for All of a Host's Services

Send a request that includes a valid 'host' value and set 'scope' to 'all':

```
curl -s -XPOST 'https://nems.local/nems-api/disable_notifications' -d '{"host": "host.  
↪example.com", "scope": "all"}'
```

### enable\_notifications

Notifications for a host, a host's service, or all of the host's services can be enabled via the `enable_notifications` endpoint.

### Enable Host Notifications

Send a request that includes a valid 'host' value:

```
curl -s -XPOST 'https://nems.local/nems-api/enable_notifications' -d '{"host": "host.  
↪example.com"}'
```

### Enable Notifications for a Host's Service

Send a request that includes valid 'host' and 'service' values:

```
curl -s -XPOST 'https://nems.local/nems-api/enable_notifications' -d '{"host": "host.  
↪example.com", "service": "httpd"}'
```

### Enable Notifications for All of a Host's Services

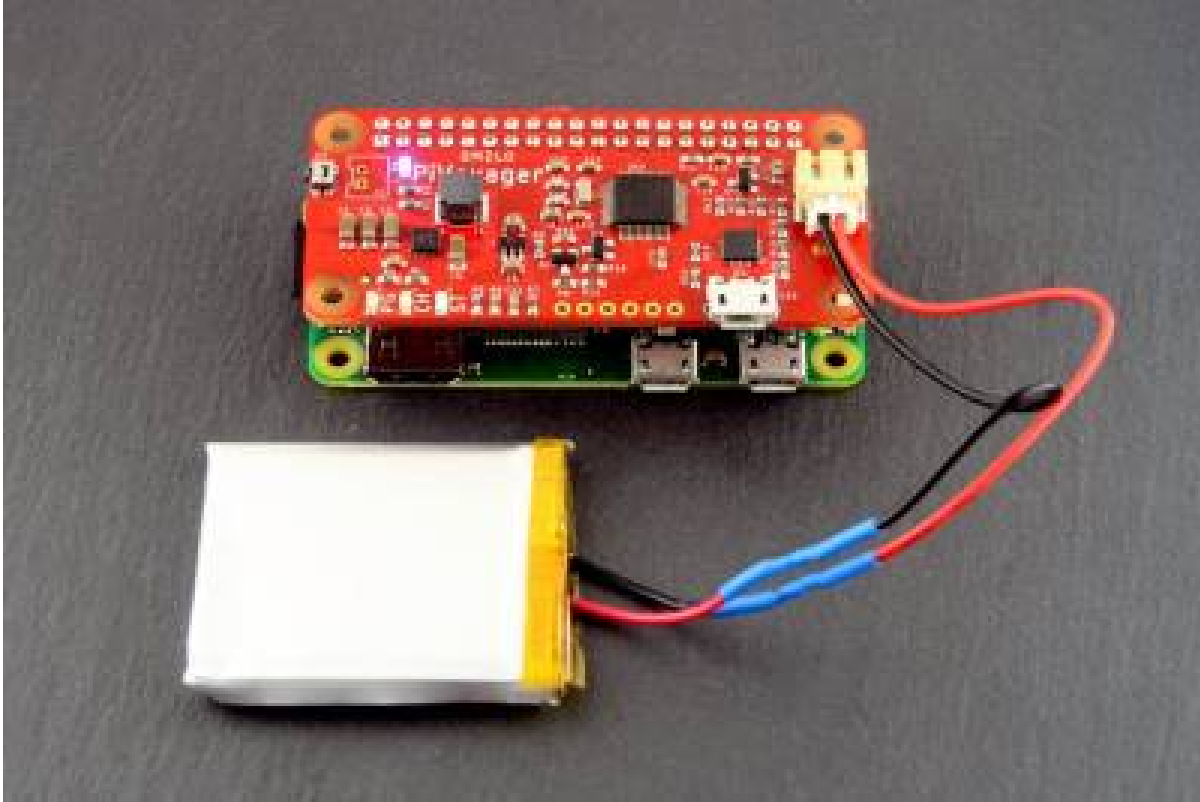
Send a request that includes a valid 'host' value and set 'scope' to 'all':

```
curl -s -XPOST 'https://nems.local/nems-api/enable_notifications' -d '{"host": "host.  
↪example.com", "scope": "all"}'
```

## 3.39 NEMS Accessories

### 3.39.1 Omozlo PiVoyager Smart UPS and Watchdog for Raspberry Pi

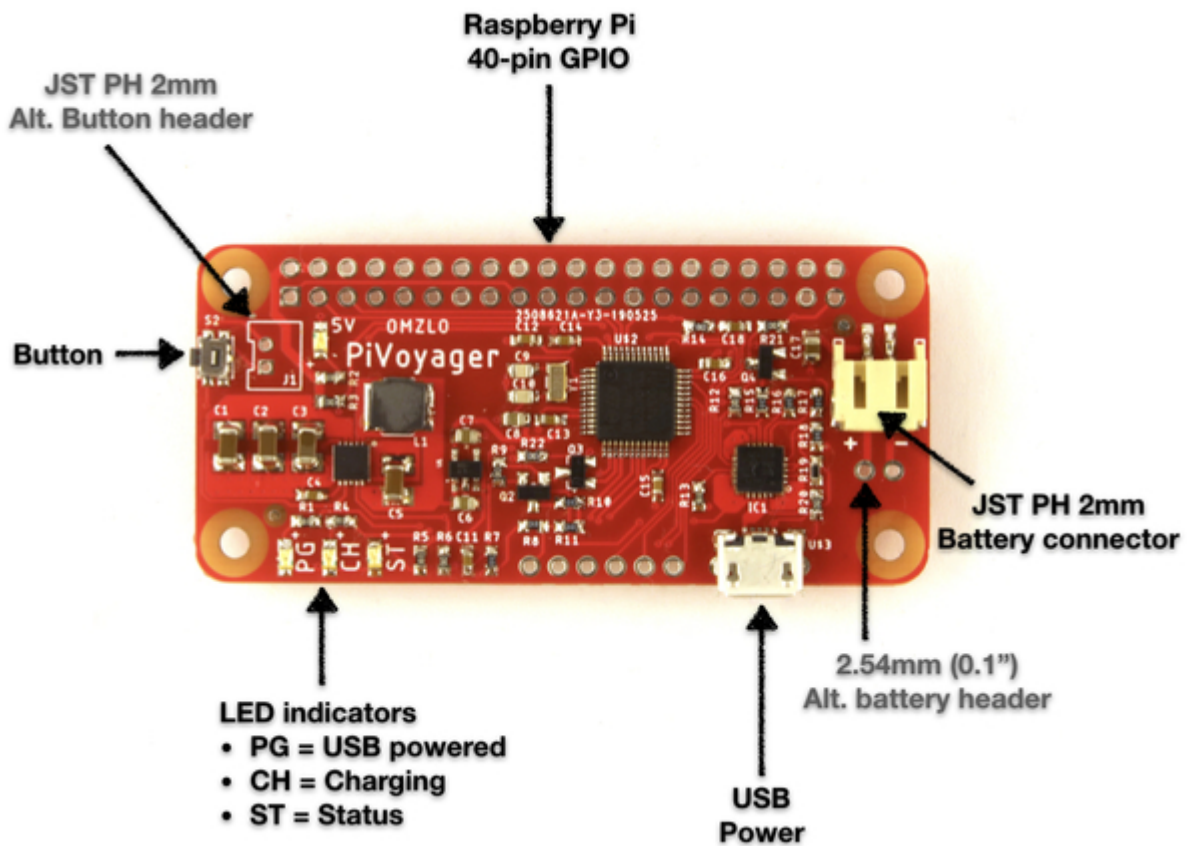
The [Omozlo PiVoyager](#) is a pHat for Raspberry Pi that allows programmable power events to take place, and provides a smart battery backup UPS that keeps your NEMS Server powered on even during a power outage.



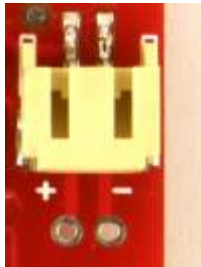
NEMS Linux includes built-in support for the PiVoyager board. There is no configuration needed: Simply plug it in and boot up your NEMS Linux server.

#### Installation

1. Solder the 40-pin GPIO header to your PiVoyager if required.
2. Safely power off your Raspberry Pi-powered NEMS Server.
3. Disconnect the 5V power cord from your Raspberry Pi.
4. Before mounting the PiVoyager atop your Raspberry Pi, power the PiVoyager with a USB cable. All LEDs, except the yellow one, should turn on.
5. Connect the PiVoyager to your Raspberry Pi's GPIO.
6. Connect the 5V power cord to your PiVoyager, *not* the Raspberry Pi (which will henceforward receive its power from the GPIO pins). *Never* connect power to both.
7. If the Raspberry Pi does not immediately begin to power itself up, short-tap the power button on your PiVoyager
8. Prepare a LiPo or Li-Ion battery that matches the following requirements:



- Has a nominal voltage of 3.7V and a charging voltage of 4.2V.
- Has built-in protection against over-heating, over-charge, and over-discharge.
- Accepts a charging current of 1000mA.
- Has a 2mm JST connector, or if you cannot find one, you can solder the battery leads to the two pin header.



9. Observe the polarity. There are + and - signs on the board that serve as polarity indicators. If your battery's polarity is incorrect, snip the leads and reverse them with some solder tinning and shrink tubing.

10. Connect your battery to the PiVoyager.

When both the battery and the USB power source are connected to the PiVoyager, all four LEDs should be on.

The yellow LED will blink if the battery is charging and will be fully on if the battery is fully charged.

### Watchdog

If your Raspberry Pi should become unresponsive or otherwise freeze up, your NEMS server will automatically power cycle after 2 minutes.

### Smart UPS

If power is out and the battery becomes depleted (under 3.3V), your NEMS Server will be safely shutdown, automatically. Upon power being restored, the NEMS Server will boot, and the battery will begin charging.

### Battery Life

On our NEMS Linux 1.5.2 Raspberry Pi 3 Model B+ with a 3,000 mAh battery, we see around 5.5 hours of battery life before NEMS Linux is safely shut down.

### Check Commands

Check commands are coming to NEMS Linux 1.6 to monitor the state of your PiVoyager's smart UPS.

## Confirm Watchdog

To confirm your PiVoyager devices is detected and active, visit NEMS Server Overview on your NEMS Dashboard.

To test if your piVoyager watchdog is working, stop the heartbeat and wait 2 minutes:

```
sudo kill -9 $(cat /var/run/nems-pivoyager.pid)
```

This should not be done on a production server (it is akin to pulling the power on a live system).

### 3.39.2 Omxlo PiWatcher Smart Watchdog for Raspberry Pi

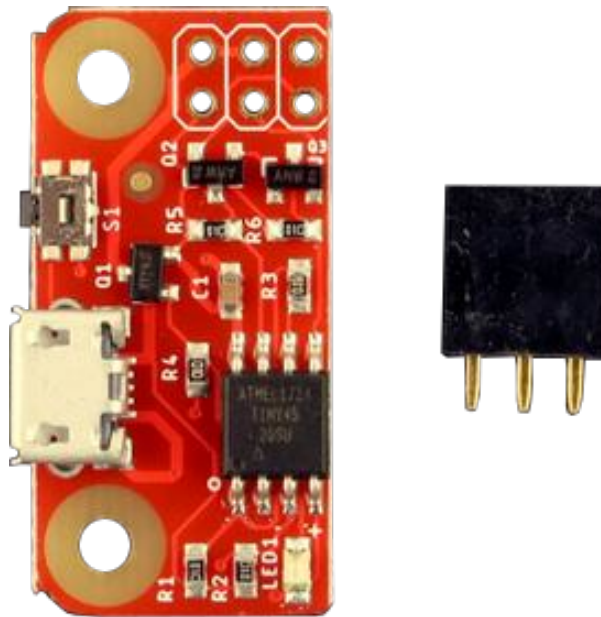


Fig. 13: Omxlo PiWatcher

The **Omxlo PiWatcher** is a very small hat for Raspberry Pi that allows programmable power events to take place.

NEMS Linux includes built-in support for the PiWatcher board. There is no configuration needed: Simply plug it in and boot up your NEMS Linux server. If your Raspberry Pi should become unresponsive or otherwise freeze up, your NEMS server will automatically power cycle after 2 minutes.

To confirm your PiWatcher devices is detected and active, visit NEMS Server Overview on your NEMS Dashboard.

To test if your piWatcher is working, stop the heartbeat and wait 2 minutes:

```
sudo kill -9 $(cat /var/run/nems-piwatcher.pid)
```

This should not be done on a production server (it is akin to pulling the power on a live system).

### 3.39.3 Omzlo NEMS Warninglight pHAT

The NEMS Warninglight pHAT from Omzlo is an i2c accessory for Raspberry Pi GPIO that provides visual feedback by way of status lights. For testing or convenience, the pHAT features state LEDs on-board, which, like a connected tower signal light, will indicate the state of your environment. In addition to the signal light functionality, a built-in watchdog (based on the *Omzlo PiWatcher*) keeps an eye on your NEMS Server and will hard reboot it if it becomes unresponsive.

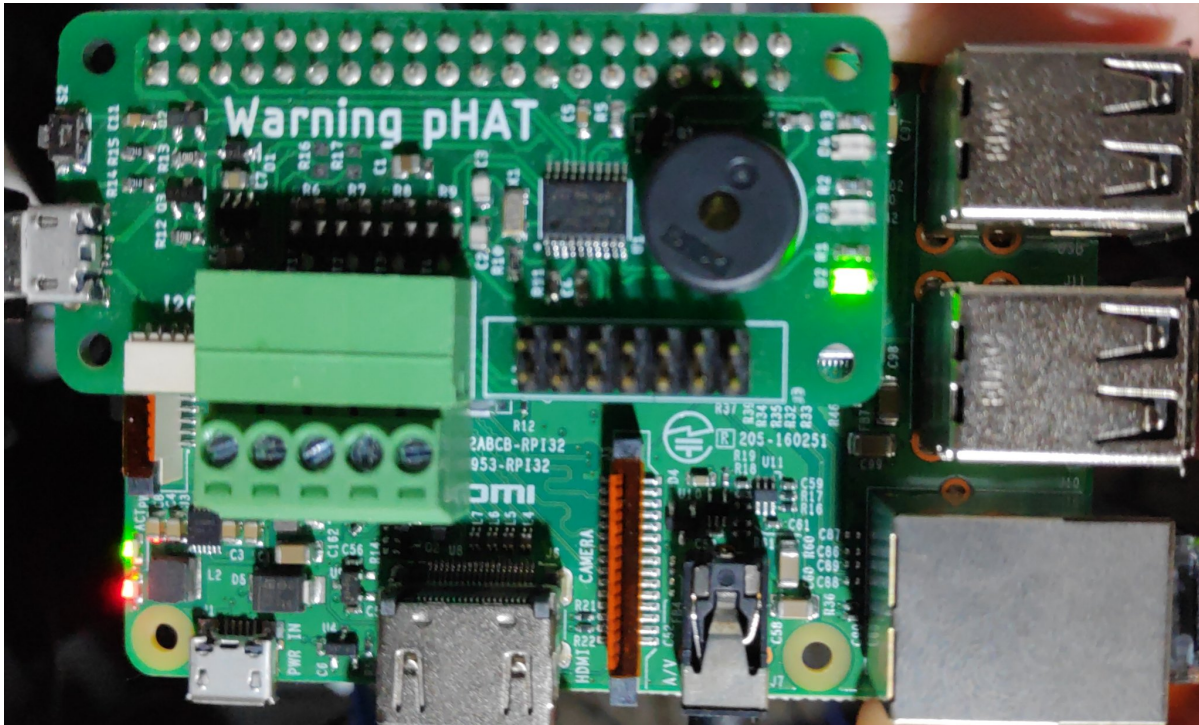


Fig. 14: The early development prototype of the NEMS Warninglight pHAT from Omzlo.

#### Buy a NEMS Warninglight pHAT

The NEMS Warninglight pHAT is in development testing and will be released soon. Pre-orders may become available, so please check back.

#### Connection

The NEMS Warninglight pHAT features a terminal block which allows the connection of a 12V or 24V NPN (sinking) type signal light.

**Warning:** Only 12-24V signal lights are supported. Do not connect high-voltage (E.G. 110V/220V) signal lights or power supplies to the pHAT.

You will need two separate power sources: one 5V USB input for the pHAT, which powers the Raspberry Pi, and one 12V or 24V power source for the connected tower light, depending on the voltage of the light.



## Basic connection diagram

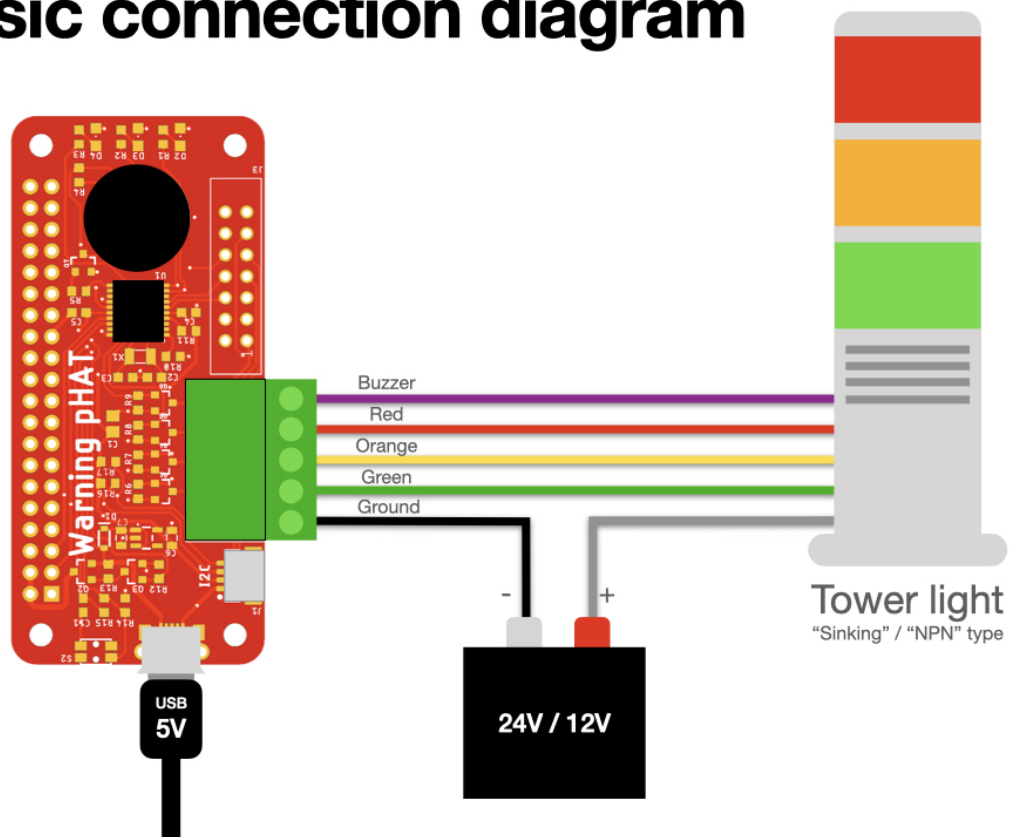


Fig. 15: Tower signal light connection diagram.



## Indications

- **Red/Orange/Green Flashing** NEMS Linux is loading following boot and the pHAT is awaiting Warninglight state information.
- **Green Solid** Your hosts and services are in an OK state.
- **Orange Solid** At least one host or service has entered a Warning state.
- **Green Solid, Orange Flashing** Your hosts and services are in an OK state, however at least one is in Unknown state.
- **Red Solid** At least one host or service has entered a Critical state.

## Manufacture

The Omozlo NEMS Warninglight pHAT is designed and assembled in Greece.

## Certification

None yet. Working on getting CSA certification.

## Developers

NEMS Warninglight is a software component of NEMS Linux developed by [Robbie Ferguson](#) for [Category5 TV Network](#).

NEMS Warninglight pHAT is an accessory designed and developed by [Alain Pannetrat](#) for Omozlo.

### 3.39.4 Argon ONE Raspberry Pi 4 Case

The stylish aluminum alloy body, the combination of both passive and active cooling, and a power button that safely controls the power state of your NEMS Server are just a few of the points that make the Argon ONE case a beautiful choice for your Raspberry Pi 4-based NEMS Server.

The Argon ONE active cooling fan is controlled via I2C, and the fan speed varies based on how hot the Raspberry Pi SOC is. The power button on the Argon ONE triggers a safe software shutdown on the Raspberry Pi, as if you had typed *shutdown now -h*

## Requirements

- Raspberry Pi 4 (Any Model)
- NEMS Linux 1.6+



Fig. 16: Argon ONE Case for Raspberry Pi 4

## Where To Buy

Please use one of the links in [the Category5 TV Shop](#) to support NEMS Linux.

## Power Button

Powered Off	Short Press	Power On
Powered On	Hold 3 Seconds	Safe Shutdown
Powered On	Hold 5 Seconds	Hard (Unsafe) Shutdown
Powered On	Double Tap	Hard (Unsafe) Reboot
Powered On	Short Press	Nothing Yet

## Fan Controller

### Defaults

- CPU temperature below 50°C, fan will be disabled.
- CPU temperature between 50-55°C, fan will run at 5% speed.
- CPU temperature between 55-60°C, fan will run at 10% speed.
- CPU temperature between 60-65°C, fan will run at 30% speed.
- CPU temperature between 65-70°C, fan will run at 55% speed.
- CPU temperature between 70-75°C, fan will run at 75% speed.
- CPU temperature between 75-80°C, fan will run at 85% speed.
- CPU temperature between 80-85°C, fan will run at 90% speed.
- CPU temperature above 85°C, fan will run at 100% speed.

You may adjust the fan speed settings in [NEMS SST](#).

### 3.39.5 NEMS Linux: Raspberry Pi GPIO Pinout

The pinout diagram is applicable to Raspberry Pi-based NEMS Linux servers, as well as [NEMS Tools GPIO Extender](#) receivers.

Pin	Resrved For	Pin	Reserved For
1	Omzlo piWatcher / DHT Sensor	2	Omzlo piWatcher
3	Omzlo piWatcher	4	Omzlo piWatcher
5	Omzlo piWatcher	6	Omzlo piWatcher
7	DHT Sensor	8	–
9	DHT Sensor	10	–
11	–	12	–
13	–	14	–
15	–	16	–
17	–	18	NEMS Warning Light - CRIT
19	–	20	–
21	–	22	–
23	NEMS Warning Light - Unknown or WARN	24	NEMS Warning Light - OK
24	–	26	–
27	–	28	–
29	–	30	–
31	–	32	–
33	–	34	–
35	–	36	–
37	–	38	–
39	–	40	–

## Graphical Layout

# 3.40 NEMS Linux Changelogs

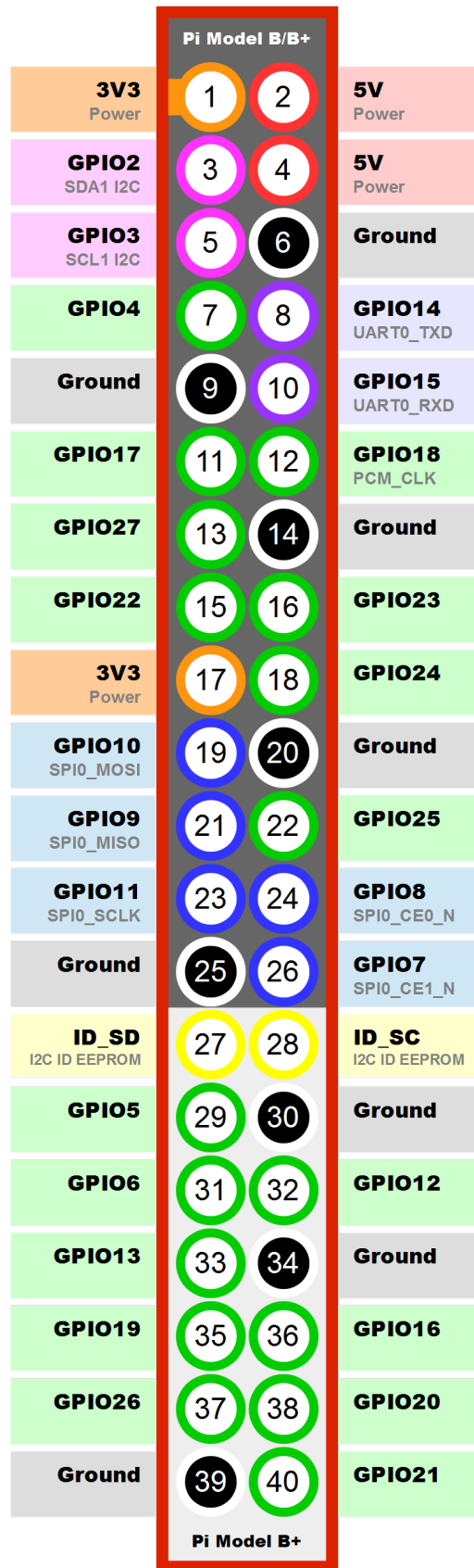
## 3.40.1 NEMS Linux 1.7 Changelogs (2024-Present)

The NEMS 1.7 release cycle focusses on modernizing the codebase, complete with a move to 64-bit, Python 3.11 and venv, as well as an effort to further enhance the Enterprise user experience by including features that are geared specifically toward larger organizations.

During the NEMS Linux 1.7 development push, we introduced our first-ever Developer Lockdown, which ended with a live Google Meet call between our lead developer and users. It was a huge success, and made it possible for NEMS Linux 1.7 to be released. Thank you to everyone who supported Developer Lockdown! For more information: <https://www.patreon.com/nems/posts?filters%5Btag%5D=Lockdown+2024>

## Release Dates

- March 26, 2024 - NEMS Linux 1.7 for Raspberry Pi released to Patrons.
- March 29, 2024 - NEMS Linux 1.7 for Raspberry Pi released to the public.



[www.raspberrypi-spy.co.uk](http://www.raspberrypi-spy.co.uk)

### Release Notes

#### New Features

- Raspberry Pi 5 support, including boot from USB, has been implemented in NEMS Linux 1.7.
- AArch64 (ARM64) hypervisor support for Enterprise Edition Virtual Appliance.
- NEMS Power Controller. Now you can safely reboot or shutdown your NEMS Server directly within NEMS Dashboard.
- Created `nems-configs` package, which will include base configurations, starting with `monit`.
- New check command `check_pve` - Check Proxmox Node.
- New check command `check_mssql_mem` - Check memory usage on a MS SQL 2008, 2012 or 2019 server (SQL memory, not system).
- New check command `check_fortigate` - Check Fortinet FortiGate devices with a number of useful checks.
- New check command `check_netscaler` - Check Citrix Netscaler Application Delivery Controllers / load balancers.

#### Upgraded Features

- Base upgraded to Bookworm.
- PHP upgraded to 8.2.
- Python moved to v3 stack due to 2.7 EOL, `venv` utilized. 3.11 is default with 3.9.2 also configured in update-alternatives.
- `livestatus` upgraded to 2.1.0p37 (cannot go to 2.2 until Trixie due to GCC+-13 dependency).
- Nagios Core upgraded and held back to 4.4.14 to ensure optimum stability (4.5.x currently has issues with `livestatus` segfault).
- Nagios Mobile UI upgraded to 1.0.3 and then modified to support PHP 8.
- NEMS API upgraded to support PHP 8.
- NagiosTV upgraded to support PHP 8.
- Adagios built into a Python 3.9.2 Virtual Environment.
- Upgraded NagVis to 1.9.40, which includes PHP 8.2 support as well as other fixes since the previous 1.9.27 from NEMS 1.6.
- NEMS NConf updated to support PHP 8. Several bug fixes.
- NEMS Livestatus Full upgraded to support PHP 8.

## Feature Improvements

- `monit` now watches the state of Adagios and restarts the service if it fails.
- NEMS Migrator now reconciles the NEMS SST configuration against the current running server, and imports your settings from backup appropriately.
- NEMS Migrator output is more consistent and hides confidential information such as passwords during reconciliation.
- NEMS Migrator's `nems-restore` now imports additional credentials, including Windows Active Directory.
- Raspberry Pi filesystem resizing on first boot is now done with NEMS Universal Filesystem Restore tool. This has been upgraded to support USB drives.
- TEMPer device check commands enhanced to include `perldata` and nicer output.

## Bug Fixes

- Fixed `nems-commits` which shows changelogs on the official NEMS Linux web site, but was lacking a few important submodules (`nems-migrator`, `nconf`).
- NEMS Migrator `resource.cfg` reconciliation now functions as expected (so settings such as SMTP are reflected in your restored configuration).
- `nems-init` and `nems-passwd` now allow special characters, such as `*` in passwords.
- `nems-info` cache file location check improved to store system user caches in `/tmp/.nems_cache` rather than attempting to save to a home folder, which was causing `STDERR` output in the logs since the folder didn't exist.
- Errors in NEMS NConf when saving hosts have been fixed. This bug affected 1.6 users.
- NEMS documentation has been fixed to point to the current repository (was still pointing to Cat5TV which redirects, but ReadTheDocs stopped following the redirect, so content was beginning to fall stale).

## Additional Notes

- `Iperf3` running as daemon.
- Serial login enabled.
- ARMhf support to be deprecated in favor of AArch64.

## Known Issues at Release

The following issues exist at the time of release and will be fixed via `nems-update`.

### **NEMS Migrator will import hosts and assign them the Linux OS even if they are not Linux.**

You may use the bulk edit feature in NEMS NConf to change the OS of your hosts in a few clicks. A future upgrade to NEMS Migrator will store this NConf data in a SQLite3 database for later reconciliation. Since it does not exist in the Nagios Config files, the association is lost upon import.

### **NEMS Migrator will remove all example hostgroups from NEMS NConf.**

This, too, is due to `nems-restore` attempting to reconcile Nagios Config files to our NConf database. Since the Nagios Config files only contain hostgroups that are in use, the import gives the appearance of all others being deleted. In fact, it is that the only hostgroups it can import are the ones that are currently in the Nagios Config from the backup. This

data will also be included in a future upgrade of NEMS Migrator where hidden data will be stored in a separate database to allow enhanced reconciliation during a `nems-restore` operation.

**Cockpit will show the Argon ONE service as failed when a user does not have a Argon ONE device.**

The Argon ONE service will be added to the Optional Services tab of NEMS SST and pushed out via a `nems-update`.

**NEMS email notification failure recovery is unable to send missed messages due to a missing initialization of the \$mail object.**

This will be fixed and pushed out in a `nems-update`.

### 3.40.2 NEMS Linux 1.6 Changelogs (2023-2024)

The NEMS 1.6 release cycle will focus on hardware porting and peripherals.

#### Release Dates:

- NEMS Linux 1.6 Alpha was released to the beta team December 2020.
- NEMS Linux 1.6 Beta and NEMSeOS 0.1 Alpha Build 2 were released to Patrons May 21, 2021.
- NEMS Linux 1.6 for Raspberry Pi released to Patrons January 8, 2023.
- NEMS Linux 1.6 Virtual Appliance (OVA/VHD/QCOW2) released to Patrons January 21, 2023.
- NEMS Linux 1.6 for Raspberry Pi and Cloud Instance released to public January 28, 2023.
- NEMS Linux 1.6 for ODROID XU3/XU4/HC1/HC2/MC1 released to public February 5, 2023.
- NEMS Linux 1.6 for Indiedroid Nova released to public May 11, 2023.
- NEMS Linux 1.6 for the ASUS Tinker Board released to public August 12, 2023.
- NEMS Linux 1.6 for ODROID N2/N2+ released to public August 16, 2023.
- NEMS Linux 1.6 for Amazon Web Services released to public October 23, 2023.
- NEMS Linux 1.6 Build 2 for Raspberry Pi released to public October 31, 2023.

#### Key Development Features and Goals:

- Public release of [NEMS Warning Light](#).
- Public release of NEMS Extender OS.
- Updated packages.
- Supported single board computer list to be reduced to focus on most used, with option for users to request ports if they do not exist.
- Port NEMS Linux to Docker platform. Development funding provided by Patrons (THANK YOU).
- Consistent networking configuration across platform (for setting static IPs, for example).

#### Development Changelogs Since 1.5:

December 2019

- Merged [PR 4](#), adding escalations for hosts / services / advanced services and fixing various bugs.
- Upgrade NEMS NConf jquery code to 3.3.1.
- Tap NEMS NConf into user's wallpaper settings.
- Output the final line from syntax check to screen when deploying config in NEMS NConf.
- Changed "Generate Nagios config" to clearer "Deploy Config to NEMS" in NEMS NConf.



- Moved all the legal stuff off the landing page of NEMS NConf and moved to new “Legal Disclaimer” section.
- Prepare Migrator for 1.6 and integrate new NConf configs.
- Upgrade Nagios Core to 4.4.5.
- Integrated [NagiosTV](#) as per BastyJuice.

#### January-February 2020

- Upgrade Monit to 5.26.0.
- Move build tmpdir out of /tmp/ and into /usr/src/
- NEMS NConf 1.6 tooltip font color fixed.
- NEMS NConf 1.6 removal of JQuery Accordion from Generate page, replaced with terminal output.
- New Deploy button for NEMS NConf Generate Config page.
- Change notify-service-by-telegram to point to the modern script, rather than the old path (which is a symlink these days). mydogboris noted that when using the symlink, his Telegram notifications were wonky.
- Added the NEMS MOTD to Docker.
- General filesystem cleanup: remove some old cruft.
- NagiosTV theme customized to match user theme settings (such as background image).
- Added humidity sensor to check\_temper.
- NRPE upgraded to 4.0.0.
- Fixed NRPE installer script if run on non-NEMS system, as per [Issue 2](#) by Xelo.
- Revert NRPE to 3.2.1 temporarily, until I can figure out how to make it talk to older installs (upgrading to 4.0.0 breaks all existing 3.2.1 installs).
- check\_speedtest\_cli.sh now records a log file in /var/log/nems/speedtest.log. This log is now parsed by NEMS TV Dashboard and displayed at all times. Further improvements will be added before public release. As requested by ITManLT.
- Replace words “Down” and “Up” in NEMS TV Dashboard speedtest stats with font-awesome icons.

#### March-April 2020

- Move 1.6 base to Debian 11 (bullseye).
- Upgrade Monitorix to 3.12.0.
- Added *mod\_wsgi* to 1.6, which is no longer auto-installed with Apache2 in bullseye.
- Initial user files now placed correctly in /etc/skel for user creation.
- Change default body background color of *NEMS NConf* to dark grey (almost black) to prevent the appearance of the screen flashing white during page transition.
- Moved old *python-pip*, *python-django* and *python-paramiko* to modern *python3-pip*, *python3-django* and *python3-paramiko* packages respectively.
- Upgrade Adagios to 1.6.6.
- Migrate to several experimental branches of Adagios components to begin testing support under Python 3.8. Current stable only supports Python 2.7 which is EOL this year.

#### May-June 2020

- NEMS Linux 1.6 development merged (some would say “downgraded”) with NEMS Linux 1.5.2 as I wait for Adagios support for Python 3.7+. NEMS Linux 1.6 will be buster-based and use Python 2.7.

- Change ownership of `/var/log/nems` to nagios user and group to allow check commands to write to logs. This enables the speedtest log, which in turn shows speedtest data on NEMS TV Dashboard.
- **TEMPer** calibration added to NEMS SST and integrated into all TEMPer output. Now, for the sake of accuracy, users may adjust the thermal and humidity sensor offset by simply dragging a slider.
- Hide TEMPer calibration option if TEMPer device is not present.
- Add Argon ONE scripts for Raspberry Pi 4.
- **Argon ONE** support complete.
- Added *repos* feature to [nems-info](#).
- Added new `check_mrtgtraf_nems` check command in preparation for a very sleek MRTG integration.
- Set Argon ONE to log to `/var/log/nems/argonone.log` if system is rebooted or shutdown safely using the power button (with timestamp).
- Move rpimonitor off the repository and instead manually install. The source repository (<http://gitedubberger.fr/rpimonitor/>) became unstable during the 1.5 release cycle (up and down) and since rpimonitor appears to be a dead project (hasn't been updated in > 4 years) I've instead removed the repository requirement and installed it manually. Also, fixed the distribution name on the template.

### July-August 2020

- Change logrotate for NEMS logs to rotate logs based on log size, not date. Delayed compression added.
- `check_mrtgtraf_nems` now rounds the floats of the traffic results to two decimal places.
- Added `check_apc` to check the state of an APC UPS. As requested by [Neptunum](#).
- Python version default will now be hard set at 995 during build process. This is in preparation for python3 down the road (pending Adagios compatibility).
- I noticed logrotate rotating some NEMS logs that really shouldn't be rotated, such as `clouddauth.log`. Modified `logrotate.d` conf to omit logs that should not be rotated.
- Change permissions of `/var/log/nagios/nagios.log` to allow group write. Required to remove root user requirement on `nems-mailtest`, which will be needed to integrate it into NEMS Dashboard.
- `nems-mailtest` is now integrated into NEMS SST. [See Video](#)
- `nems-mailtest` in NEMS SST now runs `nems-mailtest` using the information entered in the form, not saved to the config. This way, a user may test their SMTP config *before* saving the changes.
- Reverted MySQL database to previous push of 1.6 branch as I had accidentally pushed the 1.5 database to the 1.6 branch when I added `check_apc`. I've corrected it, and re-added `check_apc`. No users impacted by this since 1.6 hasn't yet been released.
- Added `nems-info [dht11|dht22]` which outputs json response from either the DHT11 or DHT22 sensors.
- Add user to gpio group during initialization. This will allow access to the GPIO pins without root access.

### September-October 2020

- Minor improvements to output of DHTxx check command for Humidity values.
- Add nemsadmin user to gpio group, just to ensure the demo check commands work out of the box, before initialization takes place.
- Upgrade check\_mk Livestatus to 1.6.0p17. This gets the sock working on the NEMS Linux 1.6 AWS development system, which is required for Adagios and NEMS TV Dashboard.
- Added phpmyadmin, disabled by default and interface access protected behind NEMS user login/password when enabled.

- Change `check_local_disk` to support unit selection (KB, MB, GB, TB) and set default for NEMS Local disk check sample to GB.
- New Feature: NEMS PHP Server Agent can now be configured and downloaded in NEMS System Settings Tool.
- New documentation launched, with the help of Bill Marshall plus submissions from Don Jenkins.

#### November-December 2020

- NEMS Tools now automatically detects the IP address of the running NEMS Server on the same subnet and creates its configuration file.
- NEMS Tools GPIO Extender client now uses the config file to determine NEMS Server IP address.
- Omzlo Warning Light pHAT now transmits and receives via NEMS GPIO Extender, allowing you to plug the pHAT into your NEMS Tools GPIO Extender Receiver.
- Fixed a typo in Warning Light that incorrectly determined all platforms to be a Raspberry Pi.
- Update `nems-tools` GPIO Extender to detect NEMS Servers via `gpioe-server` on port 9595 rather than looking for a host reply from `nems-api`.
- Moved `nems-tools.conf` to `/boot`, making it extremely easy to modify the conf on `nems-tools` Extender OS.
- NEMS SST will no longer warn of requirement to add a password for the PHP Agent if a password already exists in the config.
- NEMS PHP Agent 1.1 released. Now the keyphrase is encrypted (rather than base\_64 encoded). Also added “.” current folder disk space check and fixed several bugs with *disk* and *var* checks.
- Fixed the directions on NEMS SST which directed a user to add their encryption/decryption passphrase to the “General” tab, since that feature was moved to the NEMS Migrator tab long ago. Pointed out by UltimateBugHunter-NitPicker.
- NEMS Extender OS will now illuminate all lights if the NEMS Server goes offline or cannot be contacted.
- Improve output in `nems-init` if user tries to initialize with same name as already running user.
- Added `qemu-guest-agent` to improve integration with Proxmox VE as per UltimateBugHunter-NitPicker.
- Delay for a random amount of time (up to 2 hours) when running offsite backup tasks to prevent 1,000+ NEMS Servers clobbering the API all in the same moment. Issue pointed out by UltimateBugHunter-NitPicker who was seeing errors with his daily backup.
- NEMS Migrator Off Site Backup will now log if the Internet was down when the backup ran. Also added a ‘now’ cli option that will force it to run without delaying.
- Changed Migrator patches failsafe to determine if Quickfix / fixes was running for more than 120 minutes (previous setting was 90) before killing.
- Adjusted NEMS Off Site Backup to trigger at midnight but delay for a random amount of time up to 4 hours before running the backup. This will further reduce strain on the NEMS Cloud Services servers as NEMS’ userbase continues to grow.
- Added missing `Set::IntSpan` package, required by `check_mikrotik_switch`. Modify the check script to provide better (cleaner) output if CLI type not provided.
- All `check_mikrotik_switch` check commands renamed from the old `check_mt_` and now are `check_mikrotik_`. Also fixed argument count and improved descriptions in NEMS NConf for each of the MikroTik checks.
- Removed `check_minecraft`. It hasn’t been kept up to date by its developer, and unless there is a demand for it, I don’t want to have to take over maintenance on such a niche plugin.
- Added `check_ncpa` along with two sample checks: `check_ncpa_processes` and `check_ncpa_mem`.

- Fix bug in PiVoyager daemon that falsely detected PiVoyager hardware on some setups where it didn't actually exist.
- Several small web interface fixes in *nems-www*.
- Fix running user detection in *nems-info* to prevent *www-data* from attempting to use user cache.
- Removed NEMS host from HTTP Advanced Service. It was included as a sample, but since the interface can run quite slow on some low-powered SBC's, it causes timeout notifications which tend to confuse users into thinking there's actually a problem.
- Added *check\_synology* as per [AlbertPauw](#). Added several Advanced Services samples to Synology host group.
- Improved Warning Light's detection of Omzlo Warning Light pHAT to prevent log bloat.
- Improve NEMS Extender OS's browser-based output to include an iteration, which will help in event of a hung service: If the iteration (counter) no longer increases, it becomes more apparent that something has gone awry.
- Disable the TEMPer thermal and humidity checks by default to prevent new users seeing a warning that they are missing the sensor. Leave the demo checks in NEMS NConf for easy re-activation.
- Fixed permission issue on NIC cache if root is automatically detecting NIC while user is simultaneously logging in as non-root user. This bug was seen in Novaspirit Tech's video introduction to NEMS Linux 1.5.2 [when signing in via SSH](#). The MOTD did not display, and in pausing the video I see it is in fact the *nemsadmin* user momentarily not having access to the NIC cache. So I fixed it.
- Rearranged NEMS Warning Light daemon to ensure the Omzlo Warning Light pHAT over a NEMS GPIO Extender is synchronized to the GPIO pin output (I.E., don't delay for an iteration: instantly change states).
- Improved the output of all disk checks in NEMS PHP Agent. Now the mountpoint will be listed in brackets after the percentage, making it easier to see the actual state.
- NEMS Migrator Off-Site Backup schedule maintenance automatically keeps backups tidy now. Current schedule is that you have access to every daily backup for the past month, and a weekly backup for the past year.
- Added new check command *check\_nems\_osb* which will notify if a NEMS Migrator Off-Site Backup fails.
- NEMS PHP Agent 1.2 released with the following improvements: Network usage now uses *ifstat* and generates a more accurate average usage number based on all network interfaces on the server with a 5 second average. New agent will only run the equations and tests for the requested check. For example, don't run a 5 second network bandwidth test when the requested check is for the load average. Fixed bug where *nettx* was in fact reporting *netrx*.
- Updates to NConf to improve output. Add AJAX spinner during generation, remove horizontal scroller, etc.
- Block error output when detecting NIC to prevent MOTD being broken during first login.
- Determine the *fk\_id* of the NEMS Host and adapt *nems-restore* to use this (NEMS 1.6 branch only). Keep 1.5 branch separate and improve compatibility with 1.6.
- When a Patron opens the NEMS Dashboard (I.E., they have a valid NEMS Cloud Services account) a link is now available to visit the latest Patron-exclusive posts.
- NEMS Hardware ID is now blurred by default and hidden from display in NEMS Server Overview. I saw a YouTuber who opened NEMS Server Overview willy-nilly on his video and did not blur this information in post production, so I've enforced it by default. The HWID can be revealed by double-clicking the blurred area.

January-February 2021

- Migrator tab now has more intuitive output when a new NEMS Cloud Services account is activated (I.E., Notice re. waiting 24 hours for first OSB).
- NEMS Tactical Overview (NagiosTV) now running 0.6.5 and now uses Chris Carey's implementation of the Livestatus connector rather than the previous Nagios CGI method. In lay speak, this means when you open

NEMS Tactical Overview, you'll no longer have to supply a password within five seconds to login to Nagios Core. Huge thanks to Chris for making this change for us!

- Pause general development to work through major issue with WMIC following changes to Microsoft Windows starting with Windows 10 Version 2004.

March-April 2021

- The WMIC issue has been fixed. Final stages of development for NEMS Linux 1.6 can resume.
- Many updates to NagiosTV bringing it up to 0.7.3, which improves error handling on connection loss and much provides overall performance improvements. These updates are with thanks to Chris Carey, who we are pleased to have welcomed to our Beta Team earlier this year.
- Remove the usage of Ookla's speedtest service and replace it seamlessly with Netflix's fast.com. This is due to changes in Ookla's licensing agreement, but also means the check commands have been rewritten with better error handling. I forked the `fast-cli` project so I could make necessary changes to port this to arm64. `fast-cli` does not work on arm processors since it uses `puppeteer` which depends on the x86 Chromium headless browser. My fork instead calls for the version of Chromium built specifically for each system, which means it will work on all platforms running NEMS Linux.
- Set service timeout to 120 seconds rather than the previous 60 seconds. The short timeout was causing timeouts with speedtest since that takes extra time to execute on some connections.
- Created a new `--reset` option for `nems-quickfix` which allows resetting a previously-applied patch. This can be useful should the patch fail for whatever reason.
- Made the new speedtest script load a cache file if it is already running. In an event where the user had initiated multiple simultaneous speedtests, many processes could spawn resulting in crippling bandwidth usage. Noted by UltimateBugHunter-NitPicker.
- Backported the speedtest update to NEMS Linux 1.5.
- `nems-quickfix` now resets the log but appends both runs to it each time it runs. This will give me the ability to investigate QuickFix issues more easily.
- `nems-quickfix` now ensures all previous package installations are complete before running patches. Was an issue if user had previously rebooted their NEMS Server during an update leaving some packages broken.
- Several compatibility fixes added to the `nems-speedtest` patch (000015) to ensure backward compatibility with as many NEMS Servers as possible.
- Continued work on WMIC, which has posed challenges in porting across platform.
- Launched new NEMS Linux repository for 1.6. By doing so, I plan to compile WMIC on each architecture and then be able to install via apt, rather than going through so much trouble compiling from scratch on every single board. Inevitably this will also mean moving all NEMS packages off github (as far as how NEMS Servers obtain the code) and instead maintain a single apt repository. This could mean much easier cross-platform support from my perspective, and faster adoption of updates across all boards.
- Modified the speedtest script to wait 100 seconds for the task to finish and then forcibly kill it. If this happens, NEMS will report "0 Error" instead of a speed result. By doing this, I prevent slower connections from having the speedtest check killed by Nagios (service timeout) leaving a compounding number of Chromium tasks running, eventually leading to high load and slow operation.
- Upgrade CheckMK to 1.6.0p23 and move source to Github (rather than a zip from their site, which became deadlink after a recent redesign).

May-August 2021

- Clean up a lot of the old build scripts and `nems-upgrade` packages to accommodate the changes in NEMS Linux 1.6.

- Deprecation of Samba wmic complete: NEMS Linux 1.6 now entirely moved to new custom Python replacement.
- Upgrade Nagios to 4.4.6.
- Move Nagios configuration base to 1.6.
- Remove deprecated Speedtest Server output from NEMS Server Overview.
- Use the new hw-detect (32-bit or 64-bit) system rather than hw\_model (32-bit only) in preparation for future 64-bit release (1.7).
- Added *check\_by\_ssh* check command to sample database as [requested by AlphaPiAlpha](#).
- Small fix to prevent changelog duplication in the nemsadmin home folder.
- Checkboxes have been missing from NEMS NConf for quite some time, since upgrading JQuery to a more current version. This has been fixed by adapting the code to changes in the JQuery-UI widget callbacks.
- Moved 9590, hw-detect, nems-migrator, nems-scripts, nems-www and wmic to dpkg repository.
- Adapted nems-update to upgrade all possible apps via apt-get rather than git.
- Stripped out a lot of legacy code from NEMS Migrator and NEMS Scripts.
- Reworked NEMS Migrator's MySQL base settings. Now take significantly less space.
- Moved NEMS Merch store to <https://shop.nemslinux.com/>
- Upgraded NRPE to 4.0.3.
- The file *check\_rpi\_temperature* has been renamed to match its check command, *check\_sbc\_temperature* since it was upgraded to support more than just RPi a while ago.
- Include Fahrenheit instead of just Celsius in *check\_sbc\_temperature* / NRPE CPU temperature check as [per tripled](#)
- Moved all included Nagios check command plugins to nems-plugins package on the DPKG Repository to ease the update process should fixes or new features be implemented down the road.
- Upgrade check\_ncpa.py to 1.2.4.
- Add NCPA sample check\_commands to NEMS NConf as [per joeluzzi](#): *check\_ncpa\_mem* (Memory Usage) and *check\_ncpa\_processes* (Running Processes).
- Update boot for USB boot on Raspberry Pi as [per Kevin Shumaker](#).
- Add 1-Wire Temperature Check as [per jtoland](#).
- Add SONOFF / Tasmota IoT device monitoring as [per AndroBuilder](#).
- All speedtest commands have now been re-written (again). The fast.com fix from the end of 1.5.x to replace Ookla was heavy (required a headless Chromium task), sometimes unreliable and didn't work on all platforms. So I've re-written the system entirely, now using [Cloudflare's Speedtest](#) to test the Internet connection speed. The results are cached for 2 minutes, so if a user accidentally runs multiple instances, it will not bottleneck nor give incorrect results due to high bandwidth usage.
- Nagios i fully integrated, along with check commands for IBM i platform as [requested by chris\\_hird](#).
- IBM i integration may now be enabled in NEMS SST under optional services. It is disabled by default since it requires an additional daemon be running.
- Add option to NEMS SST for "disabled by default" optional services that must be enabled manually.
- Adapt NRPE configuration to support newly named *check\_sbc\_temperature* script.
- Direct NEMS Linux to pull the NRPE config file from NEMS Migrator package rather than downloading from Github (as would be the case when installing on a non-NEMS system).

- **NEMS Linux 1.6 Beta Build 1 released to Patrons.**
- **NEMSeOS 0.1 Alpha Build 2 released to Patrons.**
- Update `fk_id` for NEMS Migrator restore, and temporarily disable `nems-restore` feature until more testing can be done.
- Remove state text (UNKNOWN, WARN, CRITICAL, OK) from `check_nems_osb` - let the output be based on the exit codes. Also change date output to show full day, not 3-character short-form.
- Add `nems-plugins` to `nems-update` procedure.
- Remove NEMS Hardware ID from RPi-Monitor. Thanks to UltimateBugHunter-NitPicker for pointing this out. NEMS Server Overview is the correct place to get this now.
- Add NEMS Server's platform name to RPi-Monitor.
- Temporarily reverted NRPE host installer to 1.5.x compatibility since 1.6 uses a newer version of NRPE. Thanks to smccloud for [pointing this out](#).
- Added *libnumber-format-perl* as a dependency for `wmic` package.
- Improved the error output of `check_mrtgtraf_nems` after [PixelSlayer Bob](#) reported messy error output.

September-December 2021

- Upgraded NagVis to 1.9.27.
- Create new `check_truepool` check command.
- Added pool share percentage to output of `check_truepool` and formatted output.
- Further improved output of `check_truepool` and added a cache and some error handling to ensure accuracy.
- `check_speedtest_cli` updated to 2.1. NEMS Linux requirement removed so it can be used on non-NEMS systems. Set paths dynamically and prepare for backport to NEMS Linux 1.5.x.
- If speedtest is missing any of the required components, they will now be installed rather than just providing an UNKNOWN state.
- Pipe speedtest logs differently to avoid errors if the check is run by a user other than nagios.
- Fixed array associations with `speedtest` output.
- Made it so if someone scheduled/ran the speedtest task more than once per 2 minutes, it will pull the response from cache rather than running multiple speedtests simultaneously (which would skew the results).
- Speedtest updates backported to NEMS Linux 1.5.x.

January-July 2022

- Add TEMPer sensor support for TEMPerGold\_V3.3
- Update speedtest to automatically kill the process if it has been running > 10 minutes.
- Fix speedtest to load the log data first, ensuring correct output even if the task is already running.
- Set livestatus version to 1.5.0p13.
- Update `moment.js` to 2.29.3 to patch against directory traversal bug and backport this patch to 1.5.x.
- Update WSGI to Python 3 version.
- Add `libraspberrypi-bin` (particularly for `vcgencmd` so I can add a check for under-voltage).
- Added detection of other Pi models with Model IDs between 150 and 199 (the new Raspberry Pi 4 Rev. 1.5, for example) so these will be recognized by NEMS Linux as Raspberry Pi. Previously the detection looked for Model IDs between 0 and 9.



- Add *undervoltage* check to *nems-info* to detect power state of Raspberry Pi.
- Upgraded NagiosTV to 0.8.5.
- Adagios moved to 2.0.1, deprecating its dependency on Python 2.7 and moving to Python 3.9 in a venv.
- Add some extra fonts that are required by Monitorix to fix broken fonts.
- pnp4nagios has been missing for a long time since development stopped and it was pulled from repositories. Write a new compiler to re-add pnp4nagios to NEMS Linux.
- Several packages whose names were changed have been replaced in NEMS Linux with their modern counterparts. NetworkManager, PyWBEM and ModSecurity to name a few examples.
- Ensure all Adagios code is compiled within a Python Virtual Environment.
- Allow authorized users (members of the *nagios* group) to execute nems-info commands that require root access without needing a password.
- Add *int* option to *nems-info undervoltage* to send an integer response rather than plain text.
- Remove extraneous output from *nems-info undervoltage* - we only want to know if it's under voltage, or not.
- Adapt wmic to Python 3 code rather than the old Python 2 version.
- *nems-tv* moved to Debian repository.
- *nems\_sendmail\_host* and *nems\_sendmail\_service* now save state to */tmp/email\_failure.tmp* if the email failed to send, and upon a successful mail send will follow up by sending that log. This means if Internet goes down, notifications that occurred during that time will later be sent (for example). Previously, if the notification couldn't send, it would just not go out.
- *nems-plugins* - Added *check\_snmp\_apc\_env* for APC NetBotz environmental sensors.
- *checkmk livestatus* upgraded from version 1.5.0p13 to 2.1.0p9.

### August-December 2022

- Upgraded NEMS PHP Agent to 1.5. Now will report CRITICAL state if a *disk* mountpoint is specified but not mounted. This will ensure an alert is received if a mounted drive fails or otherwise disconnects.
- Upgrade Nagios to 4.4.7.
- Disabled *check\_for\_updates* in Nagios config per <https://github.com/NagiosEnterprises/nagioscore/issues/861> due to Nagios instability when enabled.
- Created [NEMS Hero](#), which allows NEMS Linux tech support to access a NEMS Server in event of a support request. This access is subject to firewall rules and only allowed if the connection is made within 15 minutes of a reboot. This is in preparation for upcoming NEMS SAAS, as well as a means of restoring a user in event of a lost / forgotten password.

January 8, 2023 - NEMS Linux 1.6 RC for Raspberry Pi released.

February 22, 2023 - Fixed issue with new NEMS Servers not being imported to system, resulting in failed CheckIn account setup.

Visit the official NEMS Linux [Git Changelog](#) for more.



### 3.40.3 NEMS Linux 1.5 Changelogs (2019-2023)

The NEMS 1.5 release cycle will focus on an enhanced user experience and documentation.

On September 22, 2019, NEMS Linux 1.5.1 was released.

NEMS Linux 1.5.2 was released June 1, 2020, introducing support for the Raspberry Pi 4B 8GB SBC.

#### Key Development Features and Goals:

- Nagios upgraded to latest current core.
- HTML email notifications.
- Admin contacted if NEMS is offline (via API check-in).
- New plugins and check\_commands integrated based on community requests.
- Documentation at the checkcommands level improved, along with other step-by-step guides added to the documentation.
- Bring support for industrial PCs such as the CL100 and FitLet2.
- Create a Virtual Appliance.
- Upgrade base to Debian Buster.
- Upgrade PHP to 7.3.

#### Supported Platforms

- **ODROID**
  - XU4
    - \* **Release Date:** February 12, 2019 (Patrons: February 7, 2019)
    - \* Runs at a reduced frequency of 1.3 GHz. This provides the best balance of high performance, low temperature and ongoing stability.
    - \* NEMS Linux 1.5 for ODROID XU4 supports SD or eMMC deployment. eMMC must have a current U-Boot.
    - \* The ODROID XU4 image will run on XU3, XU4, HC1, HC2 and MC1 hardware. However, it has only been officially tested on the XU4.
  - C2
    - \* **Release Date:** April 15, 2019 (Patrons: April 13, 2019)
  - C0/C1/C1+
    - \* **Release Date:** September 1, 2019
  - N2
    - \* **Release Date:** April 15, 2019 (Patrons: March 31, 2019)
  - ODROID-H2
    - \* **Release Date:** TBD (waiting on development unit)
- **Raspberry Pi**
  - All Boards (except Compute Module)
    - \* **Release Date:** February 26, 2019 (Patrons: February 9, 2019)
- **PINE64**

- A64 / A64+
  - \* **Release Date:** March 26, 2019 (Patrons: March 23, 2019)
- Rock64
  - \* **Release Date:** March 26, 2019 (Patrons: March 25, 2019)
- A64-LTS
  - \* **Release Date:** April 2, 2019 (Patrons: March 29, 2019)
- RockPro64
  - \* **Release Date:** April 18, 2019 (Patrons: March 30, 2019)
- H64 Model B
  - \* **Release Date:** TBD (waiting on development unit)
- CLUSTERBOARD
  - \* **Release Date:** TBD, though should already work with SOPine release. Testing soon as I have received a test unit.
- [Khadas](#)
  - VIM3
    - \* **Release Date:** July 3, 2019 (Patrons: July 1, 2019)
- [FriendlyElec](#)
  - NanoPi M4
    - \* **Release Date:** April 30, 2019 (Patrons: April 2, 2019)
  - NanoPi NEO Plus2
    - \* **Release Date:** May 14, 2019 (Patrons: May 1, 2019)
  - NanoPi Fire3-LTS
    - \* **Release Date:** TBD, Q4 2019
  - NanoPC-T4
    - \* **Release Date:** TBD (need to purchase development unit)
- [Virtual Appliance](#)
  - OVA / VHD / QCOW2
    - \* **Release Date:** March 1, 2019 (Patrons Only)
- [Amazon Web Services](#)
  - Community AMI
    - \* **Release Date:** August 24, 2019
- Docker
  - **Release Date:** TBD, waiting for Docker support to activate account and patronage to hit the target goal. Please consider becoming a Patron or increasing your pledge to help make this happen.
- [ASUS](#)
  - Tinker Board, 2 GB / S
    - \* **Release Date:** May 8, 2019 (Patrons: April 12, 2019)

- \* Runs at a reduced frequency of 1.2 GHz. This provides the best balance of high performance, low temperature and ongoing stability. I was also quite concerned with how hot the SD card would get at the default frequency of 1.8 GHz.

- Orange Pi
  - Orange Pi Zero
    - \* **Release Date:** April 30, 2019
  - Orange Pi PC Plus
    - \* **Release Date:** TBD, Q4 2019
- Atomic Pi
  - **Release Date:** TBD, Q4 2019
- Logic Supply
  - CL100
    - \* **Release Date:** TBD, Q4 2019
- FitPC
  - Fitlet2
    - \* **Release Date:** TBD, Q4 2019

#### **NEMS 1.5 Corporate Sponsors**



#### **NEMS 1.5 Patrons**

I'd like to thank *all* of our Patrons for your [continued support](#) of NEMS Linux development.

Here is a list of those Patrons who kicked in that little bit extra to have their name included in the changlogs:

- Patrick Kersten
- Marc Dörseln
- Dave Harman
- Bill Marshall
- Aaron Tringle
- Steve Hudnall
- IT Cyber Solutions
- Natacha Norman
- David Klindt
- Wolfgang Friedl

- Jeff Conaway
- Don Jenkins
- Marco Antonini
- Jessica K. Litwin
- Matthew Mattox
- Premium | Fischer-ICT
- Steve Thompson
- Jiffy
- Larry Getz
- Coquille Indian Tribe
- Jarrod Andrews
- Dennis Bailey
- Brian Darnell
- SystemOfADL
- Tony Browne
- Steven Beukes

Want your name on this list? [Become a Patron](#)

Big thanks also to Heini Holm Andersen for granting me special permission to use, customize and distribute [Nagios Responsive HTML Email Notifications Templates](#) as part of NEMS 1.5+.

Also, thanks to [Björn Ricks](#) from [Greenbone Networks](#) for kindly providing a much more current version of WMI after OpenVAS stopped supporting it. This new version has become [nems-wmic](#) and is integrated into NEMS Linux 1.5.

### Known Issues

- While I had wanted to include a pre-configured CSF/LFD firewall with the release of NEMS Linux 1.5, unfortunately it didn't make it in on time: CSF/LFD is not yet compatible with Debian Buster, and so has been moved to NEMS Linux 1.6.
- IMPI check commands will not function yet due to [a bug in FreeIMPI](#). The check commands are already in place, so I will push this out as an update as soon as it is fixed upstream. Until this time, FreeIMPI and its components have been removed from NEMS Linux.

### NEMS 1.5 Changelog

---

**Tip:** This changelog is a list of the major revisions culminated during this NEMS release cycle. To see the full list of Git commits, please also check out [the web site](#).

---

### Initial Release

#### Software Upgrades

- Nagios Core has been upgraded to 4.4.3.
- Adagios upgraded to 1.6.3-2, bringing it closer to a complete and stable modern replacement for Nagios Core's reporting interface.
- Check\_MK livestatus socket upgraded from 1.4.0p31 to 1.4.0p37.

- PHP upgraded from 7.0 to 7.3, introducing the Sodium cryptography library for the NEMS 1.5 Cloud Dashboard.
- Innumerable system updates of various packages.
- WMI has been forked and upgraded to 4.0.0.
- Check WMI Plus upgraded to v1.64.
- nagios-plugins 2.2.1 has been removed and replaced with the current git build of monitoring-plugins.

### New Check Commands

- Cisco SNMP monitoring as [requested by mydogboris](#): `check_cisco_interface` and `check_cisco_switch`.
- IPMI Sensor Monitoring Plugin as [per thegreatadmin](#): `check_ipmi_sensor_driver_slot`, `check_ipmi_sensor_fan`, `check_ipmi_sensor_memory`, `check_ipmi_sensor_power_supply`, `check_ipmi_sensor_power_unit`
- `check_esxi_hardware` as [per readyit](#).
- `check_qnap` as [per Toxic](#).
- `check_internet_speed` as [per infocon](#).
- `check_procurve_loop` as [per lee3521](#).
- MSSQL check as [per itsubs@sagroup.co.uk](#).
- `check_docker` and `check_docker_swarm` from [https://github.com/timdaman/check\\_docker](https://github.com/timdaman/check_docker) as [per Zerant](#).

### New Features

- **NEMS Cloud Services** - NEMS 1.5 introduces NEMS Cloud Services. By activating this optional service, your NEMS Linux server will benefit from off-site backups and notifications should your device stop responding (See NEMS CheckIn below). Plus, later this year I will be introducing a web-based tactical view that is accessible from anywhere, and amalgamates the tactical information of multiple NEMS servers on your account allowing sysadmins a method of monitoring multiple sites from one cloud-based dashboard.
- **NEMS CheckIn** - NEMS 1.5 introduces [NEMS CheckIn](#). CheckIn will notify you by email if your NEMS Linux server becomes unresponsive. Disabled by default, NEMS CheckIn can be configured within NEMS SST. This service requires a NEMS Cloud Services account.
- **Optional TLS** - TLS Secure Authentication can be disabled in NEMS SST for SMTP email [as requested by luckyworlock](#).
- **Graphing** - nagiosgraphs now comes preinstalled [as requested by Erast Fondorin](#). It is configured and functional in Nagios Core, but can also be accessed from the NEMS Dashboard *Reporting* menu.
- **Webhook Notifications** - NEMS Linux now supports notifications via webhook as requested by [Jon Backhaus](#). This feature was added to [nems-tools](#): [Warning Light](#).
- **Custom Appearance** - NEMS SST now features the ability to change the background on some NEMS screens.
  - Background Selection, allows you to select from the following:
    - \* *Daily Image (Default)* option loads a new image every day.
    - \* *NEMS Legacy* shows the classic server room image from NEMS Linux 1.4.
    - \* *Custom Color* allows you to choose a base color to use for the background.
    - \* *Upload Image* allows you to upload your own preferred wallpaper image.
  - Blur Background Selection, allows you to add a blur effect to background images:
    - \* *Slight Blur* will add a subtle Gaussian blur to the background image.
    - \* *Medium Blur* will add a more pronounced blur to the background image.

- \* *Heavy Blur* will blur the background image so heavily that only the color scheme of the image is recognizable.

### Bug Fixes

- WiFi now works on Raspberry Pi devices out of the box as reported by the community.
- `check_sbc_temperature` (previously called `check_rpi_temperature`) prompts for Warning and Critical temperatures as pointed out by [mg11976](#).
- Fixed Nagios Core *Trends* and *Alert Histogram* giving 404 errors as per [damo](#).
- Fixed *Host Detail* and *Service detail* giving 404 error as per [ronjtaylor](#).
- There was a [known issue with Monit 5.20](#) (included in NEMS 1.4.1) which results in an error “Forbidden: Invalid CSRF Token”. For Monit’s web interface to work, you must open it in an Incognito window (the bug is related to cookies). A fix was rolled into NEMS Linux upstream (as of October 2018). 5.25 is out in source, but not in Debian repositories. As this bug was fixed, I’m no longer concerned about the issue, though it will be nice to see 5.25 make its way into the repos.
- DST problem in Nagvis as per [ronjohntaylor](#) fixed by system-wide timezone variables now being set in *nems-init*.
- `check_nrpe` is now installed to the correct folder. The upstream installer places it in the folder for Nagios 3, not Nagios 4.

### Improved Features

- Email notifications are now responsive HTML emails. Please see the “thanks” section above.
- NEMS SST now allows you to disable rolling updates. By doing so, your deployment will remain as is, allowing you to better control when/if your NEMS server is updated. As suggested by Dave Harman. Also supports putting off updates to run once per month, once per week, or once every two weeks as requested by John Naab.
- All `check_wmi_plus` check commands have been redone to correct the argument counts and also to provide better descriptions for each arg. Was suggested by mydogboris.

### Miscellaneous

- samba shares have been improved to support long filenames.
- MOTD has been improved. The generally not useful info has been removed making it a little cleaner looking, and a new ASCII logo has been integrated.
- Removed the old default checks from NEMS host and created new templates specifically for NEMS that are more appropriate for low-powered SBCs (super high CPU load thresholds, for example). Some users were running the sample checks as if they were intended for production use rather than as a guide, so this should help those users by not setting off irrelevant alerts such as CPU load or swap usage on the NEMS server (particularly problematic on low powered devices like Raspberry Pi, since the default samples are more suited for monitoring full-powered Linux servers). [Reported by experimenter](#), MarshMan, and others.
- NEMS SST now warns you if you try to navigate away before saving changes.
- WMIC is no longer being distributed by OpenVAS, so I have forked the most current git repo their parent company provided, and am now using that (after some modifications). New active repository is located at <https://github.com/Cat5TV/wmic> and *wmic* version has been upgraded from 1.3.14 to 4.0.0.
- *vim* is now included by default as requested by [Zerant](#).
- *webmin* has been removed from NEMS Linux. The networking feature [has been broken for quite some time](#), and waiting for the patches to arrive upstream has been much too long. Also, Webmin tends to confuse novice users into thinking their NEMS server is out of date (due to pending OS updates), and in some rare cases has resulted in users breaking their configuration. There are no reasons to keep Webmin, but many to remove it.

## Rolling Updates 1.5

February 2019

- Moved bootscreen to TTY7 and disabled kernel log output. TTY1 (the previous default) was also receiving syslog messages, which led to a messy screen. As noted by Bill Marshall.
- Raspberry Pi 2/3 Build 1 private Patron release.
- ODROID XU4 Build 2 private Patron release. Fixes WMIC compile issue. Updated versions of PHP, Apache2, and a few other packages that were updated upstream.
- Fixed ownership of *nems-www*, which was causing users to be unable to upload custom backgrounds. Reported by Haaku. Thanks to m9Networks and UltimateBugHunter for assisting.
- Fixed environment variables for local libraries to ensure *wmic* can find *libopenvas\_wmiclient.so.1*. Also improved the *nems-wmic* installation procedure to ensure all files are saved and persistent. This to mydogboris for testing.
- Removed NEMS Linux version number from header of NEMS Dashboard. As it is already included in the footer, it is redundant.
- Removed Facebook link from NEMS Dashboard (I have been using it less and less) and changed the YouTube and Twitter links to point to the new NEMS Linux profiles, rather than my personal profiles. NEMS has its own now!
- ODROID XU4 Build 3 private Patron release. Merges all rolling updates. Fixes *wmic*.
- Network Manager was using its default setting to automatically spoof a new MAC address every time wifi connected. On a Raspberry Pi using WiFi, this would cause a new IP address in the DHCP pool, and users trying to establish static reservations would not be able to do so. I over-wrote the default and now the actual physical MAC address will be used. The patch will future-ready all other NEMS builds for devices that support WiFi.
- Raspberry Pi devices now resize the filesystem on boot, rather than on init.
- Raspberry Pi 2/3 Build 2 private Patron release. Merges all rolling updates. Re-compiled *wmic* and applied WiFi patch. Added US locale out of the box.
- Moved Raspberry Pi to stable kernel rather than latest kernel.
- ODROID XU4 Build 3 Public release.
- Opened ports 548,5353,5354 in CSF/LFD Firewall to allow AVAHI / mDNS to function normally, as per issue reported by Jon Backhaus. Will have no effect on NEMS Linux 1.5 since CSF/LFD are not yet compatible.
- Raspberry Pi 3 Model A+ added. Raspberry Pi Model B/B+ have been split so the correct board will be reported (B or B+, not B/B+).
- *nems-update* output improved. Formatting improved, and now includes the before and after git commit IDs.
- Fixed NEMS 1.5 using NEMS 1.4 database out of the box before initialization.
- Set default timezone to America/Toronto.
- Ensure packages are not upgraded from Sury's PHP repository on Raspberry Pi Zero/1 (breaks these builds if otherwise).
- Added *piwatcher* switch to [nems-info](#). *piWatcher* support is now fully integrated and will automatically power cycle a Raspberry Pi-based NEMS Linux server if it becomes unresponsive.
- NEMS Linux 1.5 base upgraded to Debian Buster. This resolves many backport issues on the Raspberry Pi Zero/1 build, and further upgrades many of the underlying core OS components.
- PHP upgraded to 7.3.
- CSF/LFD firewall not yet compatible with Debian Buster. Removed until such time as it is.

- Move JavaScript and CSS assets from CDNJS to *nems-www*. Users with certain DNS filters were missing components such as jQuery due to CDNJS being blocked by their DNS provider.
- Released Build 3 for Raspberry Pi to Patrons. This test release merges all Raspberry Pi boards into a single build and is for testing only (not for production use).
- Write a new installer for *raspi-config* on Raspberry Pi build. Build 3 failed to install it, so automated filesystem resize failed on first boot and WiFi settings could not be configured.
- Build 4 for Raspberry Pi released to Patrons. This is a test build that resizes correctly on first boot and supports WiFi. However it does not have the *check\_commands* compiled so is not ready for production use.
- Removed *nagios-plugins* which appears to be a dead project (still no 2.2.2 after all these years, doesn't compile correctly on buster) and moved to *monitoring-plugins* which is still active and compiles nicely.
- Added Daily Color option to NEMS SST for background. Each day's color is extracted automatically from the color pallet of that day's daily image.
- Activated I2C on Raspberry Pi build to allow piWatcher compatibility. Added final timers to piWatcher script.
- NEMS Linux 1.5 for Raspberry Pi released to public (Build 6).
- *nems-info ip* will now output 127.0.0.1 instead of NULL if no IP address is found on a network controller. Fixes MOTD on local logins where a network connection is non-existent.
- Patched PHP 7.3 and PHP 7.2 configs to allow larger background image uploads in NEMS SST.

March 2019

- NEMS Linux 1.5 Virtual Appliance OVA and VHD Build 1 released to Patrons for early testing.
- Connected TV screen improved to include NEMS state information. Colors softened for normal state, and will turn red in event of CRITICAL state.
- When uploading a custom background image, the default color is then selected from the upload and applied to the browser theme. This gives a nice consistency between uploaded image and theme colors. Note: If then changing to Custom Color, the color will be selected by default.
- Leaving SMTP username blank in NEMS SST now disables SMTP authentication, as requested by dr\_patso on Discord (to accommodate Office 365 relay).
- Treat thermals as floating point instead of string in thermal logger [as per nix-7](#).
- Forked *monitoring-plugins* and created new installer in *nems-admin* to fix some of the issues with the check commands.
- Rollout a newly compiled version of NEMS WMIC to systems who are missing it. This update takes a long time and so will lead to a new build for all platforms.
- Added support for MS Teams webhooks [as per stealth81](#).
- Added support for Slack webhooks.
- Install PostgreSQL development libraries for *check\_psql* and OpenSSL, and recompile all Nagios plugins. Fixes errors in NEMS check commands. PATCH-000001 - requires running *sudo nems-upgrade*
- Bumped *check\_speedtest-cli.sh* to v1.3 and disabled pre-allocation of memory. This fixes "MemoryError" on lesser boards such as the Raspberry Pi Zero. New version will get installed along with PATCH-000001.
- Added Running/Idle status of NEMS Update and NEMS Fixes to connected TV screen.
- Make NEMS branding persistent in Cockpit after an update.
- Cleaned up some bloat in NEMS Migrator's data for NEMS 1.5 (backup copies of the MySQL database).



- Connected TV display now reports if the filesystem is still being resized on first boot. NEMS Quickfix now waits 90 seconds from boot to begin (in case filesystem is being resized). PATCH-000002
- Added *glances* to NEMS 1.5 as per RSABear.
- Switch network interface management to NetworkManager, enabling static IP configuration within the Cockpit interface. PATCH-000003
- Raspberry Pi Build 7 released. This introduces the new networking system to Raspberry Pi users, as well as the improved check commands and better performance on lesser boards.
- Added CPU temperature to connected TV display. Also fixed a math error which fixes the connected TV's ability to show if a new version of NEMS is available.
- PINE64 A64/A64+ Build 1 released to Patrons. In addition to everything that NEMS Linux 1.5 is, this build introduces a new kernel which addresses a known issue exists that was previously affecting *some* A64+ boards. If affected, the system clock would jump 99 years into the future—which subsequently impacted many of the NEMS services.
- PINE64 Rock64 Build 1 released to Patrons.
- If sysbench is not available, do not try to run benchmarks.
- Notate PATCH-000002 on Rock64 boards retroactively since the Build 1 version of the file resize script does not log the success.
- Added *rootdev* and *rootpart* to *nems-info*.
- New img build workflow created, including new *Base Images*. Theoretically img files should be a bit smaller here forward (due to zerofill) and should be more consistent (less chance for corrupt build img files).
- After *reporting* *sysbench* missing Buster binaries to the developer, it was added. Integrated the Buster installation into NEMS Benchmark since the Debian repositories are thus far also missing the package. System will check upstream first, and if not available, will install from developer repository.
- PINE64 A64/A64+ Build 1 released to public.
- PINE64 Rock64 Build 2 released to patrons and public. Fixes bad image creation of Build 1 causing it not to boot. Also integrates PATCH-000002.
- *nems-quickfix* (and therefore a reboot) now removes NEMS NConf generator lock file if it exists. It can get left behind in certain circumstances, which would cause NConf to say “Someone else is already generating the configuration.”
- PATCH-000002 now gets automatically logged to all boards if the main partition is sized greater than 9 GB. This ensures platforms such as the virtual appliance and the Rock64 transition to a ready state if the patch is not logged already but the partition is indeed resized.
- If sysbench is not found in the developer's repository, remove the repository and update apt before cancelling the benchmark. See *Issue 298*.
- Added *speedtest* to *nems-info*.
- NEMS will now automatically detect the nearest Internet speedtest server and use it by default. May be overridden by ARG if option changed in NEMS SST.
- Added *rootfulldev* to *nems-info*.
- Improved thermal detection for cross-device compatibility. Also updated *nems-info temperature* to output real-time thermal data rather than 15 minutes log.
- Added *fileage* to *nems-info*.
- Improved connected TV screen to now show how long updates/fixes have been running.

- `nems-info` *hosts* & *services* were showing one more than actual true count. This was due to a previous update to the `livestatus` socket which results in it outputting a header line, which was being counted as a result. Simply subtracted 1 to counteract. As reported by UltimateBugHunter.
- PINE64 A64-LTS/SOPine Build 1 patron pre-release.
- Retroactively enable `rc.local` service on boards where it is not enabled by default (eg., Rock64). Thanks to UltimateBugHunter for putting me onto the problem, having noticed the connected TV was going to sleep after 10 minutes (even though `rc.local` was set to disable TV sleeping).
- Fixed issue with temperature output on connected TV which would cause math errors when converting from Celsius to Fahrenheit.
- ODROID-N2 Build 1 released to patrons.

April 2019

- NanoPi M4 Build 1 released to patrons.
- PINE64 A64-LTS Build 1 released to public.
- Change the warning message if NEMS can't connect to github, as pointed out by ITManLT.
- ODROID-XU4 Build 4 released. Keeping in mind the XU4 platform was the first public release of NEMS 1.5, this is a significant upgrade. This moves XU4 from Stretch to Buster and adds all the new check command scripts, as well as all updates that have been released since the first NEMS Linux 1.5 release.
- Virtual Appliance Build 2 (OVA, VHD, QCOW2) released to Patrons. This build was developed on an ESXi development laptop donated by bhammy187. Build 2 should be much easier to import into any hypervisor, making it significantly more portable than Build 1.
- Added error handling to thermal sensor check to report UNKNOWN if the sensor doesn't exist, as would be the case with a Virtual Appliance.
- New universal filesystem resizer automatically detects the root partition and resizes it. Will continue to adapt to eventually deprecate the separate resizer scripts for each platform.
- Add error handling to `nems_sendmail_host` and `nems_sendmail_service` to accommodate inability to write to Nagios log if user runs *nems-mailtest* as a non-root user. As reported by ITManLT.
- Fix issue where disabling SMTP TLS in NEMS SST would always revert to enabled. Reported by ITManLT and confirmed by UltimateBugHunter-NitPicker.
- ASUS Tinker Board / S Build 1 released to Patrons.
- ODROID-C2 Build 1 released to patrons.
- ODROID-N2 Build 1 and ODROID-C2 Build 1 released to public.
- RockPro64 Build 1 released to public.
- Minor verbiage adjustment re. Speedtest Server in NEMS SST.
- Compile `sysbench` if not exist, improve compatibility with various versions (ie., detect which switches are accepted for cli variables).
- Fix spelling of Orange Pi (DietPi had spelled it OrangePi).
- NEMS Linux 1.5 Build 1 for NanoPi M4 and Orange Pi Zero released to public.
- Update weekly benchmark to save transient data in a secure tmp folder.
- Re-order events in weekly benchmark to ensure the compiler runs even if a benchmark is not scheduled (so the needed software is ready to go).
- Lay groundwork to add 7zip benchmarks to weekly benchmark.

## May 2019

- NEMS Linux 1.5 Build 1 for NanoPi NEO Plus2 released to Patrons.
- Add distinction between 512 MB and 1 GB version of the NanoPi NEO Plus2.
- Added 7zip benchmark to weekly benchmarks.
- Fixed glitch in NanoPi NEO Plus2 hardware ID generator and blocked null HWID's after detecting that one Virtual Appliance user had booted a VM with no MAC address.
- Make weekly benchmark data readable by all, but only writable by root.
- Add *benchmark 7z* option to *nems-info*.
- Change *nems-info online* to use wget instead of ping. As pointed out by ITManLT, some networks block ping, causing NEMS to think it has no Internet connectivity (and therefore updates do not run).
- Monitorix now loads all graphs dynamically, and displays all available graphs (not just the ones I manually selected back in NEMS 1.2).
- Fix PHP notice for undefined variable when manually running a *nems-benchmark*.
- Finish moving *nems-benchmark* over to 7-Zip benchmarks rather than sysbench, and completely remove sysbench from the normal benchmark process. It will be re-added later as a supplement, but will not be used for NEMS scoring.
- If any of the sysbench benchmarks don't exist, output a 0 instead of a error in *nems-info*.
- Roll out a patch that removes some of the residual Armbian settings from early base images. /var/log was being loaded into zram instead of stored on disk, resulting in /var/log becoming full. This patch fixes that and ensures log files are stored on disk. Only affects early releases (such as Build 1 for TinkerBoard and NanoPi M4). Pointed out by [Marshman](#).
- NEMS Linux 1.5 Build 2 for Tinker Board / S released to public.
- NEMS Linux 1.5 Build 2 for ODROID-N2 released, integrating [Meverics' patch](#) which resolves the networking / slowness / inability to boot issues found on some ODROID-N2 boards. Big thanks to UltimateBugHunter-NitPicker for initially reporting this issue.
- NEMS Linux 1.5 Virtual Appliance (OVA) Build 3 released. This build reduces the ESXi Virtual Hardware Version from 14 to 7, meaning NEMS Linux may now be deployed on older versions of ESXi. No need to re-release VMDK or QCOW2 for Build 3 since the update only affects OVA.

## June 2019

- Removed unneeded virtual hardware from OVA appliance.
- Restructure the Virtual Appliance OVA for compatibility with ESXi 5.5+.
- NEMS Linux 1.5 Virtual Appliance (OVA) Build 4 released. This build resolves the error "The OVF package is invalid and cannot be deployed" on older versions of ESXi. The cause of the issue was because older versions do not support the SHA256 hashing algorithm. Build 4 is identical to Build 3 in every way except the Cryptographic Hash Algorithm has been switched from SHA256 to SHA1, making it compatible with older ESXi servers.
- Upgraded speedtest from 1.0.6 to 2.1.1.
- Added initial Raspberry Pi 4 support.
- Change the way various Raspberry Pi models are displayed. Eg., *Raspberry Pi 3* now, instead of previous *Raspberry Pi 3 Model B*. *Raspberry Pi 3 B+* now instead of previous *Raspberry Pi 3 Model B+*.
- Raspberry Pi Build 8 released. Introduces out-of-the-box support for Raspberry Pi 4 and includes all patches that were issued since Build 7 was released 3 months ago.

- Raspberry Pi boards were previously reported as ‘RPI’ by the hardware description script. I didn’t like this, so I changed it. Where a board previously listed itself as ‘RPI 3 B+’ it will now say ‘Raspberry Pi 3 B+’, for example.
- Moved *monit* to custom build rather than pulling from apt repository. Package is missing from some Debian Buster builds. This also ensures we have the latest version at time of build.

July 2019

- Added new command *nems-install* which will install NEMS Linux on eMMC on the Khadas VIM3. In future versions, it may be expanded to support other boards if required.
- Khadas VIM3 Build 1 released to Patrons.
- Buster is now stable. Update releaseinfo, and do this automatically in future.
- Remove check\_speedtest’s reliance on a server ID. Latest version supports automatic detection on launch, and will automatically roll to next available server in line if server fails to respond. Much better than single point of failure, which has been causing false notifications the past few days. Thanks to mydogboris for reporting this.
- A patch to enable disabled rc.local that was previously released had been broken due to a renamed build script. Fixed.
- NEMS SST now features a tab “TV Dashboard”. Password setting for NEMS TV Dashboard has been moved to this tab (was previously under *Optional Services*), and two new features have been added: 24 hour clock formatting, ability to display faults immediately rather than waiting for the service to enter a notification state. Some users were confused by the default, so this allows them to change when they are notified.
- In NEMS SST, move NEMS Migrator to the NEMS Cloud Services tab.
- Begin encrypting NEMS State data with personal encryption/decryption password for NEMS Cloud Services users in preparation for the coming NEMS Cloud Services Dashboard.
- NEMS Cloud Services master NEMS Server login complete.
- NEMS Cloud Services now receives NEMS GPIO Extender data from subscribed devices. This will allow NEMS Warning Light or GPIO Extender Clients to be placed anywhere in the world, and will also allow a single NEMS Warning Light to display the state of multiple NEMS Servers.
- NEMS Cloud Services Dashboard now displays the master NEMS Server alias, CheckIn setting and Host/Service state.
- NEMS Cloud Services Dashboard now updates the state data automatically. This was a bit more complex than a standard ajax call due to the encrypted state of the data.
- Added tooltips to Host/Service icon on NEMS Cloud Services Dashboard which shows the count of each state.
- Added *nems-info***cloudauthcache** option which loads the current NEMS Cloud Services authorization status from cache rather than a live connection (significantly faster for quick checks).
- NEMS Cloud Services Dashboard has been added to the “Reporting” menu on all NEMS servers which are connected to the service.
- Added NEMS Platform and Version to NEMS Cloud Services Dashboard.
- Added credential error handling to NEMS Cloud Services parent server login. Now, an easy to understand error message will be given if you enter invalid credentials, rather than just receiving a blank page.
- Moved NEMS TV Dashboards’ livestatus connector to a new folder “connectors” to pave the way for new dashboard connectors.
- NEMS TV Dashboard has been removed from *nems-www* and is now its own repository called *nems-tv*. This is to allow me to add NEMS TV Dashboard to NEMS Cloud Services without needing to build a second (redundant) version. It also means NEMS Cloud Services’ version will exactly mirror the features of the local copy.
- Add *livestatus* to *nems-info*.

- Add check to ensure *nems-tv* is enabled, and if not, enable it. Thanks to ITmanLT for pointing out the issue.
- NEMS TV Dashboard added to [NEMS Cloud Services](#).
- Improve the output of the clock on NEMS TV Dashboard.
- Sync NEMS TV Dashboard's local clock setting to NEMS Cloud Services. Now the clock output format will match your local settings (ie., 12/24h format, whether to show am/pm).
- NEMS TV Dashboard in NEMS Cloud Services now shows the alias of the reporting NEMS Server. This is in preparation for the coming reconciliation of multiple connected NEMS Servers on a single NCS TV Dashboard.
- Added support for [TEMPer](#) hardware.
- Added exit codes to `check_temper`.

#### August 2019

- Added UNKNOWN state to `check_temper`. If TEMPer device is disconnected, will now report UNKNOWN instead of 0°.
- `check_temper` now detects automatically whether the user is entering their ARG temperatures in C or F and acts accordingly.
- Major rework of NRPE. NEMS Linux no longer uses the package maintainer's version of NRPE. A new installer has also been provided for Debian hosts to ease the deployment process. Please see [Check Commands: check\\_nrpe](#) which details what is required.
- Added `custom_check_mem` checkcommand and corresponding NRPE advanced service, called *Check Memory NRPE*.
- [Telegram notification configuration](#) has been made more clear in [NEMS SST](#), and the documentation has been rewritten to match.
- [Telegram notification script](#) reworked to remove the 'g' from Chat ID automatically, making it a bit easier to understand input.
- Fixed error on NEMS Cloud Services Dashboard where the `tooltip()` function was not loaded before it was called.
- Begin building a means of NEMS Cloud Services' TV Dashboard to detect and display if the NEMS Server is not online (via NEMS CheckIn). Also, the server list will now highlight down NEMS Servers red.
- NEMS Migrator Restore now breaks apart the checkcommands file from the NEMS backup and attempts to import each command individually. This has the effect of skipping (Aborting) import of checkcommands that already exist in the default NEMS NConf database while allowing the user-created checkcommands to be imported. Thanks to Jon Backhaus for pointing out the issue.
- NEMS TV Dashboard has a setting in NEMS SST that allows you to change the notifications to immediately display, rather than honoring the notifications settings in NEMS NConf. This setting now also impacts the results of *nems-infostate* and NEMS Cloud Services' TV Dashboard.
- Added *nems-infostate all* Output the state information of all NEMS hosts and services to JSON format, regardless of state.
- NEMS Server State Report added to NEMS Cloud Services. Now you can see the state of all your hosts/services directly from the NEMS Cloud Services Dashboard.
- NEMS Linux 1.5 AMI Build 1 for Amazon Web Services released.

#### September 2019

- The development version of NEMS Linux for the ODROID-C1+ was losing its heartbeat following filesystem resize, so I got looking deeper at the ODROID resize stage scripts. In doing so, I found a typo in the `*start*` variable creation of stage1 which resulted in the first partition starting at the first sector of the disk rather than the

needed sector 8192 on the ODROID-C1+. This bug has been fixed, and the ODROID-C1+ development build is working.

- NEMS Linux 1.5 for ODROID-C0/C1/C1+ Build 1 released.
- Added *nems-info***frequency** to display the *current* CPU operating frequency.
- In *nems-init* the *mysqld* daemon is forcibly killed if stopping fails (as it tends to do on Docker). This in turn causes an error on platforms where systemd is able to stop the process: can't kill a task that isn't running. Add a check to see if *mysqld* is running before attempting to kill it.
- [Push Notifications Using Telegram](#) now includes an emoji to help distinguish the state.
- Improve *nems-fs-resize* to support drives that are not mmcblk0 type. Now, the script can be used to resize the Virtual Appliance disk, for example.
- Re-order the output of Telegram notifications to make them easier to see critical information at a glance. Now, the NEMS Server's alias and the timestamp are listed first, followed by the notification information.
- Check for the existence of rc.local before patching it in nems-update fixes. This is in particular for Docker (which doesn't use rc.local) to prevent [harmless] errors during update.
- Add SCSI dev assignment detection to *nems-info***rootfuldev/rootpart/rootdev**. This fixes the feature on non-MMC storage platforms, such as the Virtual Appliance.
- Fixed a previously unnoticed bug in Telegram *service* notifications where the Service output was displaying the datestamp rather than the output.
- Added rich-text markdown to Telegram host and service notifications. Now, the layout looks really good (not just plain text).
- Added [NEMS Linux Vendor Branding](#). Now, you can add your own logo to the NEMS Dashboard.
- Added the vendor logo (if applicable) to the init screen.
- Remove Raspberry Pi logo from boot screen.
- NEMS Linux 1.5 Build 2 for ODROID-C2 released.
- Updated migrator databases include the recommended settings for [check\\_temper](#) and [custom\\_check\\_mem](#), no longer requiring users to manually add them on new deployments.
- Version increased to 1.5.1. No further builds of 1.5 will be created.
- Add *nems-info***perfddata\_cutoff** which shows the cutoff (in days) for perfddata retention.
- NEMS SST now has a "Maintenance" section featuring a perfddata cleanup schedule. This allows users to select to remove old perfddata to avoid a bloated perfddata.log file. As requested by rkadmin, whose file had reached 15GB in size. By default, this feature is disabled and perfddata is kept indefinitely if enabled in NEMS NConf.
- NEMS Cloud Services will now re-route you to the Dashboard if you have an active session. Active sessions will remain active for 24 hours. As [requested by Premium](#).
- Removed the perfddata tweaks as they only banded the more crucial problem: a misconfigured Nagios conf. Will later add a feature to tweak nagios.cfg settings, but for now those who choose to hack their cfg files directly will probably break things.
- NEMS Linux 1.5.1 Build 1 for Raspberry Pi went into private testing.
- NEMS Cloud Services sessions now remain active for 7 days, allowing you to open NEMS Cloud Services features in your browser without needing to login (until you click Logout).
- *nems-info***frequency** now reports the average frequency across all cores, rather than the frequency of the first core. Thanks to Bo from ameriDroid for pointing out this inconsistency.
- The build process now clears bash history correctly so on first boot, users don't have the development history.



- NEMS Linux 1.5.1 Build 1 for ODROID-C2 went into private testing.
- NEMS Migrator Off Site Backup calendar data had no error handling, so if the server didn't respond during the daily check-in, the data would still be overwritten, but with invalid JSON data. Added a JSON parser to detect if the server's response is valid JSON before clobbering the data file.
- Fix the name of ODROID-C2.
- NEMS Linux 1.5.1 for Raspberry Pi and ODROID-C2 released.
- NEMS Linux 1.5.1 for ODROID-XU4 released.

#### October 2019

- *nems-infonic* and *nems-infoonline* now use nemslinux.com instead of google.com for their tests. Also, results are cached for one minute, reducing the number of requests while still remaining accurate to the minute. As requested by Joerg Hoffmann.

#### November 2019

- Removed smooth scrolling from *nems-www* as it causes jerky behavior in modern versions of Chrome, resulting in console error, "Unable to preventDefault inside passive event listener due to target being treated as passive."
- Account for the fact that some users may have passwords in their password when restoring from a NEMS Migrator backup set. Before now, a password in the password would result in a null password.

#### December 2019

- The current NEMS version is now platform independent, meaning an ODROID-XU4 won't report a new version just because a new version was released for Raspberry Pi (for example). As per [Issue 1](#) on NEMS Migrator.
- Define the platform distinction in NEMS Server Overview with regards to currently available version. Reduce calls to api by 1/3 (performance improvement) for Platform Name.
- If user is already a Patron, remove the "Become a Patron" button.

#### January 2020

- PixelSlayer Bob pointed out that 9590 was missing from monit on NEMS 1.5.1. Investigated and it turns out the monit service installers were patched into NEMS Linux during 1.4, but never moved to the 1.5 branch. Fixed.
- If a user has the "NEMS is not initialized" page open in their browser and completes a nems-init process, the browser will now automatically redirect to the NEMS Dashboard.

#### February 2020

- Upgraded 1.5 branch to check\_temper from 1.6 branch and improved thresholds for detection of C vs F temperatures. This brings in a few of the important check\_temper fixes and improvements from NEMS 1.6 to users of NEMS Linux 1.5. Thanks to *tripled* for pointing out the issue with certain temperature thresholds.
- Fix footer on NEMS TV Dashboard so it doesn't jump up after 60 seconds due to the speedtest update that was added for 1.6. Reported by ITManLT.

#### April 2020

- Add *www-data* to the forbidden usernames list. Fix *nems-info* so *www-data* user doesn't attempt to create a NEMS cache folder.
- *nems-init* now asks if you'd like to proceed if it detects your NEMS Server is already initialized.
- Improve the verbosity of error messages when restoring a NEMS Migrator Off Site Backup.

#### May 2020

- Significant overhaul of the NRPE installer to improve compatibility with client systems (especially Ubuntu / Linux Mint). Deals with [Issue 3](#) plus other undocumented issues.

- Migrated Nagios misc data folder to 1.6 branch in nems-migrator.
- Create and enable (and document) `check_cpu_temp` in [Check Commands](#): `check_nrpe` which allows monitoring remote system CPU temperatures using `lm-sensors` on the remote host. A feature request by *tripled*.
- Add Sysfs thermal data to `check_cpu_temp` if available, and fallback on it if `lm-sensors` isn't installed. Add unknown state if thermal data cannot be obtained by either of these two methods.
- Fixed the apt update which occurred during a NEMS Update task: On Raspberry Pi it requires `--allow-releaseinfo-change`, but this was breaking the update on some other platforms. So only use this argument on RPi-based NEMS Servers.
- Add the apt key signature for the sury repository, which hosts PHP for some earlier builds of NEMS Linux.
- Released NagiosTV (called NEMS Tactical Overview on NEMS Linux) in advance to NEMS Linux 1.5 users. It was originally slated to wait until the NEMS Linux 1.6 release, but that is being held up by Adagios at the moment, so I thought it would be nice to push out a little gift to the users as thanks for their patience.
- Upgraded NagiosTV to 0.5.3. Adapt CSS overrides to allow use of NEMS wallpaper and other stylistic enhancements.
- Security improvement: Moved several nems-scripts temporary shell scripts out of `/tmp`.
- Ensure check commands are replaced should NRPE upgrade be run on a NEMS Server.
- Create Credit Roll Easter Egg.
- Improve the audio timing and add layout of Credit Roll Easter Egg.
- Add IP Address to all screens of bootscreen.
- Several NEMS NConf improvements / fixes (not retroactive on already initialized NEMS Servers). Examples: Add `check_temper_temp` and `check_temper_hum` temperature and humidity checks, fix `custom_check_mem`.
- Change sample SBC CPU temperature service check to recent NRPE version and move to Advanced Services to improve understanding for users and make it easier to apply the service to other hosts.
- Add *Room Temperature* and *Room Humidity* sample services to NEMS Server. If user has a [TEMPer](#) device connected, results will be provided.
- Create `udisks2` modules directory to prevent deceptive "error" in Cockpit logs. As reported by UltimateBugHunter-NitPicker and listed in [Cockpit Issue 12412](#).

June 2020

- NEMS Linux 1.5.2 released for Raspberry Pi. Brings together all updates and patches since 1.5.1, and adds support for the new 8GB Raspberry Pi 4 Model B. Thanks to UltimateBugHunter-NitPicker for beta testing the initial build for me as my 8GB Raspberry Pi has still not arrived here in Canada.
- NEMS Linux for Docker moved to 1.5.2 branch to expedite release.
- Grant non-root access to [TEMPer](#) devices on USB after a reboot, and periodically.
- Make text darker in NEMS SST.
- Add error handling in case either the thermal or humidity sensor are not detected on a [TEMPer](#) device (as is the case with a unit which only has one or the other, for example). As reported by JonBackhaus.
- Upgrade NagVis to 1.9.20, which resolves an issue with user creation as pointed out by jnrhome. Pushed out through daily patches to all NEMS Servers.
- Pipe error output from `temper.py` to null so it doesn't interfere with the response of the script when [TEMPer](#) is not getting enough power.



- Fix incorrect NConf `fk_id_item` assignment for NEMS host in [NEMS Migrator Restore](#). This was causing the host-preset's check-alive to be assigned incorrectly. Now, generating the Nagios config will work fine after *nems-restore*. Big thanks to UltimateBugHunter-NitPicker not only for bringing the issue to my attention, but for granting me remote access to his NEMS Server to allow me to replicate and ultimately fix this.
- Install WMIC and the insert script for NagiosGraph which were missing in 1.5.2 and any systems which ran the recent NRPE upgrade. As pointed out by UltimateBugHunter-NitPicker.
- Add watchdog daemon and safe shutdown on smart UPS battery depleted for [PiVoyager](#) pHat.
- Add Multi Router Traffic Grapher (MRTG) [as requested by mydogboris](#).
- Add logging to *nems-quickfix* in case it appears hung. Log can be tailed at `/var/log/nems/nems-quickfix.log`
- Added logrotate [as per baggins](#).
- Improve handling of database initialization.
- Do not attempt to configure non-existent SSH service on Docker when initializing.
- Change *nems-update* to only attempt installation of SURY GPG key if it is missing (was expired on some older NEMS Servers as it appears they're only valid for 2 years). Current key expires March 2021.
- PiVoyager now active on all NEMS Servers (will not do anything if hardware doesn't exist).
- Abort benchmark if watchdog (PiVoyager or PiWatcher) are connected. The high load of the benchmark can cause the system to appear unresponsive for some time, resulting in the watchdog believing the board to be hung, which would cause the watchdog to power-cycle the NEMS Server.
- `mrtg.sh` will now detect the [first] default gateway of the current connection and offer it as the IP to use for MRTG. If incorrect, user may still enter the router IP manually.
- If no SNMP data is found on the router, `mrtg.sh` will provide a proper error rather than a "file not found" error.
- Add DEB/RPM detection when installing NRPE on Linux host system. Rudimentary setup of RPM installation in place. Many things still don't work on RPM-based hosts, but it's a great start, and [047-nrpe](#) will not attempt to run `apt-get` on CentOS anymore.
- If, when user runs `mrtg.sh`, the MRTG Apache configuration is not enabled, enable it automatically.
- Added *mrtgsetup* command (a symlink to `mrtg.sh`).
- Added custom trap community support to `mrtgsetup`.
- Added MRTG index to <https://nems.local/mrtg>
- Improved NEMS Migrator admin component to separate 1.5 and 1.6 databases in preparation for 1.5.2 compatibility and inevitable move to 1.6.
- Remove git postBuffer (created by `nems-admin` during *nems-push* procedure) upon *nems-update*.
- Stash a change in NEMS Migrator which halted updates to Migrator on 1.5.2 (fixes several issues related to this, including TEMPer, and ability to update NEMS Migrator). Automatically patches all 1.5.2 systems.
- Add cronjob to automatically set USB permission every minute if device connected. This makes it so users no longer have to reboot after plugging in a TEMPer sensor. Now, it will work automatically after no more than 60 seconds.

July 2020

- NEMS Migrator has been moved to its own tab in NEMS SST. It was previously included on the NEMS Cloud Services page, which is inappropriate since it can be run locally.
- NEMS CheckIn Notifications now tell you how long the server was down for upon recovery.

- Fixed *nems-www*'s `wallpaper.php` to only try to load NEMS-specific functions if running on a NEMS Server. Was causing log flood on NEMS' public web site.
- Corrected redundant verbiage on NEMS CheckIn emails. "Has been down for 15 minutes *minutes*." Not sure how that never got noticed before.
- New command: *nems-passwd* allows changing the NEMS Server admin account password without having to re-initialize NEMS Linux. As requested by geek-dom and JJ Dubya J.
- If TEMPer is connected but lacks a dev assignment, abort loop. This can occur if the pass-through on a Virtual Machine is botched. Removing the virtual device and re-connecting it should fix, but until that time, we don't want the script to hang.

### August 2020

- NEMS Linux now stores a file *NEMS\_SERVER.txt* in the Windows-readable section of your NEMS storage medium (eg., SD card) which contains some information about your NEMS Server (NEMS Version, Platform, HWID, Alias). This will make it easier for users to determine which board a SD card came out of should they need to, as per [Marshman](#), by simply plugging the card into their computer.
- Added JSON output switch to *nems-mailtest* and *nems\_sendmail\_service* in preparation for integration with NEMS SST.
- NEMS Linux 1.5.2 for PINE64 A64/A64+ released as per [this thread](#).
- Migrate all system emails (ie., NEMS CheckIn, Forum Notifications, etc.) to Amazon SES and enable DKIM. Maximize reliability of notification emails from NEMS Cloud Services and reduce being falsely identified as spam.
- Add color to the warning label when user tries to initialize a NEMS Server that has already been initialized (to make it stand out better).

### September 2020

- Make 500-temper not log to cur file, which was causing uninitialized NEMS Servers to say 'compiling' rather than 'not initialized'.
- Sync all files from user's home folder when initializing or re-initializing. While this obviously prevents any accidental data loss, this is particularly to ensure SSH key trust relationships are not lost if a user re-initializes a NEMS Server on AWS.
- Yay community! We surpassed 100 YouTube subscribers on the NEMS Linux channel, so that means we qualify for a vanity URL. I updated *nems-www* footer link to reflect this change.
- Adjust webhook notifications to recognize Discord's change in URL for webhooks. Old URLs contained discordapp.com as the domain. New ones are at discord.com, so NEMS was rejecting them for an invalid domain name. Thanks to [Amheus for bringing this to my attention](#).
- Omzlo Warninglights pHAT support enabled.
- Enabled the watchdog feature of the Omzlo Warninglights pHAT.

### October 2020

- Add external sensor support to TEMPer checks, and set as default. If an external sensor exists, it will be used. If not, the internal sensor will be used. Also added output *temp\_location* and *hum\_location* to *nems-info temper* to show which sensor is being used. As [pointed out by Toasteh\\_](#).

### 2021

While development has entirely shifted to NEMS Linux 1.6, the following issues have been addressed in 2021:

- Windows update broke *wmic* causing NEMS to no longer be able to authenticate to those updated machines. As this is fixed in NEMS Linux 1.6, I also backported the patch to NEMS Linux 1.5.x on all platforms. Patch # 000016. To obtain the patch, run *sudo nems-upgrade*
- Backport speedtest from NEMS Linux 1.6 after Ookla changed licensing.
- NEMS Update will now allow a repository update even after Debian moves the old repository.
- Cockpit branding broke in 1.6 after assets moved in NEMS-Migrator. Fixed.
- Backport of NagiosTV broken in 1.5.x after config moved in NEMS Migrator for 1.6. Fixed.

### 3.40.4 NEMS Linux 1.4 Changelogs (2018-2019)

[https://web.archive.org/web/20201020122328/https://docs.nemslinux.com/changelogs/nems\\_1.4](https://web.archive.org/web/20201020122328/https://docs.nemslinux.com/changelogs/nems_1.4)

### 3.40.5 NEMS Linux 1.3 Changelogs (2017-2018)

[https://web.archive.org/web/20200921052539/https://docs.nemslinux.com/changelogs/nems\\_1.3](https://web.archive.org/web/20200921052539/https://docs.nemslinux.com/changelogs/nems_1.3)

### 3.40.6 NEMS Linux 1.2 Changelogs (2017)

NEMS Linux 1.2 was released May 6, 2017. NEMS Linux 1.2.1 was released May 22, 2017 to fix bugs with the migrator and improve the overall release following significant user feedback. NEMS 1.2.x was discontinued November 7, 2017. This version of NEMS was downloaded 14,765 times during its 6 month life cycle.

---

**Note:** A great emphasis was placed on documentation during the 1.2 Release Cycle. I launched the [Community Forum](#) to offset the massive onslaught of user comments on the NEMS page of my blog, and [NEMS Documentation](#).

---

#### NEMS Linux 1.2.1 Changelog

Here is a list of the changes I recorded during development.

- NEMS Linux now requires you to run *nems-init* when you first deploy. This tool takes care of some of those “first boot” prerequisites like setting passwords and expanding your filesystem.
- Underlying OS upgraded to Raspbian Stretch.
- Kernel upgraded to 4.9.28.
- PHP upgraded to 7.0.19.
- [Reworking of nConf](#) to make compatible with modern software (ie. PHP7.0, mySQLi).
- Maintenance and info scripts moved to */home/pi/nems-scripts*.
- NEMS Linux MOTD upon login now shows local IP address. Also improved how it determines some of the info (see [info.sh](#) in *nems-scripts*) and fixes a few bugs. Also set it up to rollover to wlan0 if no response on eth0, in case the user is on wifi.
- Temporary files and Monitorix image cache moved to RAM.
- Added [RPi-Monitor](#) as per [Hesh's comment](#). Reworked the Memory and CPU Frequency modules to correct the accuracy.
- Added *nagios-api* (JSON on Port 8090) as per [Timothy Seibert's request](#). [License](#)

- Added Webmin as per [Hesh's comment](#). Login as pi user with the password you created when initializing NEMS Linux with the nems-init program.
- Added support for agentless Windows checks using WMI (big thanks to [Ryan Siegel](#) who originally authored NagiosPi and pitched in for NEMS Linux 1.2 release).
- Changed Apache log rotation to weekly (was previously daily).
- Upgraded nagvis to 1.9b16.
- Fixed sendmail paths in nConf to ease out-of-the-box email notifications (as they should just work now). Thanks so much to [Jim](#) for pointing this typo out!
- Enabled CPU governing (package cpufrequtils). On NEMS Linux 1.0-1.1, NEMS Linux was locked to 600MHz, but now it will automatically go up to 1200 MHz as needed.
- force resolver to generate new DNS resolv.conf at first boot (to ensure the detected DNS servers will be used rather than our development DNS servers which may not work for you).
- NEMS Linux Migrator upgraded to allow direct migration from nagiospi to NEMS Linux.
- Added Monitorix 3.9.0.
- Removed MySQL, replaced with MariaDB 10.1.22.
- Improve quality of Monitorix graphs used on NEMS Linux Dashboard slideshow.
- Minor improvements to Monitorix page based on priority of service and image quality.
- Removed some old (obsolete) kernel modules, InnoDB logs and other bloat to reduce size of stock image.
- Added /var/www/nconf/temp to tmpfs. This way if someone breaks their nCONF (eg., pressing “Back” while generating config) they can just reboot to fix it
- nems-init and nems-migrator restore significantly reworked to correct initialization bugs from NEMS Linux 1.2. Now, both initialization of a new NEMS Linux deployment and an import from an old one should work without a hitch.
- nConf and NEMS Linux-Migrator backups now require your password (as set with nems-init).
- NEMS Linux-Migrator no longer replaces the MySQL database with backup. Instead, it now clears the database completely, reconciles your backup with the current set of available commands and services and then imports everything together into the fresh database and activates the hosts. This way, if you restore your NEMS Linux 1.1 settings to NEMS Linux 1.2.1, you don't miss out on all the WMIC features (which your 1.1 backup would overwrite), for example.
- Documentation updated to reflect changes in commands and versioning.
- I built a quick but lovely interface for Monitorix to make it mobile responsive and a little more dynamic in its functionality.
- Distribution now available via BitTorrent (thanks to our partnership with [The Category5 TV Network](#)).

## Bug fixes

- number of online users count on MOTD fixed.
- undefined constant in apache error log every 5 minutes leading to a bloated error log.
- added missing icons in check\_mk.
- NEMS Linux Migrator mail settings fixed when importing backup.nems from NEMS Linux 1.0/nagiospi.
- Fixed MySQL Initialization Bug - was causing NEMS Linux to lose configuration and no longer work.
- Wifi (wlan0) restored after it broken in 1.2 (due to Debian Stretch upgrade and incompatible firmwares for the Pi).
- Fixed nems-init user creation. In NEMS Linux 1.2 it was not adding the new user to the “admin” group correctly in NEMS Linux nCONF, so upon config generation, user would lose access to Nagios Core and other features requiring admin user.
- many miscellaneous bug fixes.

## Rolling Updates - NEMS Linux 1.2.2 Changelog

- Created rolling fixes system to automatically patch NEMS Linux systems as needed.
- Monitorix cronjob now detects which network connection is being used (ie., eth0/wlan0) and begins monitoring the correct one if it changes. Requires NEMS Linux 1.2.1 or higher.
- NEMS Linux Migrator updated to fix bug in host presets. Was causing these two errors: “[ERROR] Failed to get host-alive check for host ‘NEMS Linux’. Make sure the host is linked with a host-preset. Aborting.” and “Error: Cannot open main configuration file ‘/var/www/html/nconf/temp/test/Default\_collector.cfg’ for reading!” - Thanks to Rick for giving me access to his affected system so I could fix this. Requires NEMS Linux 1.2.1 or higher.
- NEMS Linux web interface has been updated to git repository so I can fix issues with it on the fly without forcing users to reinstall. Requires NEMS Linux 1.2.2 or higher.
- Monitorix graphs clear when rebooting the Pi. This is by design. However, they were meant to regenerate upon boot. [Hesh found a bug](#) that was causing the graphs not to generate since the service was not yet loaded. I rewrote the back-end to wait for a network connection, and for the service to respond, before moving on. Requires NEMS Linux 1.2.1 or higher.
- MOTD was displaying the disk usage as what was being used in the home folder. I realized this was absolutely pointless information, so rewrote this section to instead tell you the current % usage of your entire SD card (/dev/root). Requires NEMS Linux 1.2.2 or higher.
- NEMS Linux-Init bug fixed: new user is not `authorized_for_system_information`, `authorized_for_all_hosts`, `authorized_for_configuration_information`, `authorized_for_system_commands` ... and so-on. [Wrote this into nems-init](#). Had caused users to not have access to all screens in Nagios Core. Also [added the config to the NEMS Linux Migrator backup](#). I also [added the patch to fixes.sh](#) so users don’t have to re-initialize to get the fix. It will retroactively fix the file, automatically. Requires NEMS Linux 1.2.1 or higher.
- Check\_MK Multisite reports “user not found” on some screens. Similar issue to what was happening with Nagios Core: nems-init user was not being migrated correctly to Check\_MK as pointed out by Rick. This has been fixed [and a retroactive patch added](#) to correct existing deployments. Requires NEMS Linux 1.2.1 or higher.
- Updated nems-migrator to support NEMS Linux 1.2.2. Built nems-upgrade, which will roll NEMS Linux 1.2.1 up to NEMS Linux 1.2.2 without needing to reinstall. You can do this (backup first please) by typing: `sudo nems-upgrade`

- Email notification service changed to automatically detect TLS. Changes will take effect immediately for new deployments, but if yours is already deployed (before June 9, 2017) please add the following to your service definitions in NEMS Linux-nConf for notify-host-by-email and notify-service-by-email: `-o tls=auto` - add it immediately after `/usr/bin/sendmail` so it looks like this: `/usr/bin/sendmail -o tls=auto -s $USER7$ ....`
- Created *nems-benchmark* and *nems-mailtest*.
- Disabled swapfile.
- Add keyboard locale setting to nems-init. [As per Steve](#).
- Force more secure bcrypt encryption on htpasswd generation. [As per Steve](#).

### Rolling Updates - NEMS Linux 1.2.3 Changelog

- Removed dead “help” links in Nagios Core [as per kd4pyr](#).
- Secure Connectivity [as per Steve](#): \* nems-init now creates self-signed server, client and CA certificates during initialization of NEMS Linux. \* SSL (https) access now available on most NEMS Linux features, using your new custom certs.
- Add anonymous stats logging retroactively to NEMS Linux 1.2.1+.
- Create log file at `/var/log/nems/package-versions.log` which shows specific package version information for some of the essential services in NEMS Linux. This log will be recreated every Sunday morning.
- Add *nems-info* command.
- Early introduction of PHP connector for Monitorix data, which will be utilized both by *nems-www* and *nems-info*.
- Added command-line option *temperature* to *nems-info*.
- Added average temperature to NEMS Anonymous Stats.
- WiFi patched against KRACK exploit for users connecting NEMS to the network using WPA2. To see if yours has been patched type `cat /var/log/nems/wpaapplicant` - it will either say *Patched*, or give an error. This patch is retroactive to all NEMS 1.2.x devices, and higher.
- Moved NEMS symlinks to `/usr/local/bin` to avoid loss after a dist-upgrade. Does not affect functionality: this is only an internal change in preparation for NEMS 1.3 which is transparent to the end user since both are in the path (you type *nems-init* not `/usr/local/bin/nems-init` for example).
- Move nems.conf to `/usr/local/share/nems/` in preparation for the deprecation of the *pi* user.
- NEMS 1.3 released. NEMS 1.2.x is now Old Stable. I’ll continue to support it until I see its usage numbers drop in the [anonymous stats](#).
- Patched NEMS 1.2.x to support the new file locations of NEMS 1.3. This fixes “file not found” issues in nems-update and Anonymous Stats [as reported by Digithead](#).

### 3.40.7 NEMS Linux 1.1 Changelogs (2016-2017)

NEMS Linux 1.1 was released November 13, 2016 and discontinued May 6, 2017. This version of NEMS Linux was downloaded 979 times during its 6-month lifetime.

- NagVis upgraded to 1.8.5. (1.9 is still beta).
- Check\_MK livestatus upgraded to 1.2.8p13.
- Added Check\_MK Multisite 1.2.8p13.
- Added PNP4Nagios 0.6.16-2.

- Added a few sample configurations to NConf to help users figure out the initial setup and/or to use as templates. Included samples are: test if an external web site is up via ping, monitor a Linux server, monitor a Windows server.
- Created NEMS Linux Migrator. New feature allows backing up and restoring your NEMS Linux configuration, making migration or recovery a breeze.
- Added sendmail 1.56-5 and setup email config in /etc/nagios3/resource.cfg (you'll need to add your SMTP info as per the instructions above).
- Added git, htop.
- raspi-config upgraded to 20161108.
- Linux kernel upgraded to 4.4.26-v7.
- Apache2 upgraded to 2.4.10-10+deb8u7.
- OpenSSL upgraded to 1.0.1t-1+deb8u5.
- MySQL upgraded to 5.5.52-0+deb8u1.
- Exim4 upgraded to 4.84.2-2+deb8u1.
- PHP upgraded to 5.6.27-0+deb8u1.
- PHPMyAdmin upgraded to 4.2.12-2+deb8u2.
- Python upgraded to 2.7.9-2+deb8u1.
- Network and Bluetooth firmwares upgraded.
- Various system components upgraded.
- Reduced the amount of memory dedicated to Raspberry Pi graphics adapter.

### 3.40.8 NEMS Linux 1.0 Changelogs (2016)

NEMS Linux 1.0 was released May 8, 2016 and discontinued November 13, 2016. This initial version of NEMS Linux was downloaded 409 Times.

- Initial release. Built and tested on Raspberry Pi 3. Based on Raspbian Jessie. Inspired by NagiosPi, which in April 2016 was still running on the old Raspbian Wheezy. I started this new distro since NagiosPi seems to be out of date, and I want to have an easy drop-in Nagios img for the Raspberry Pi. Figured I'd share it with the world while I'm at it since there are probably others (possibly less tech savvy) who might want the same thing. I decided to leave most of the settings the same as NagiosPi (eg., usernames, passwords) so those coming from that distro can seamlessly transition, or so if NagiosPi wants to use our build to bring things up to date, they may do so with minimal effort.
- This initial build is using default repositories in a lot of cases and is meant to be rock-solid, not bleeding edge (eg., Nagios 3.5.1 instead of Nagios 4.1.1).
- Using the rpi-4.4.y Linux kernel tree (Currently 4.4.7-v7+ #876 SMP), firmware updated to 1e84c2891c1853a3628aed59c06de0315d13c4f1. Use rpi-update to check for upgrades, if needed.
- Includes rpi-update tool - an easier way to update the firmware on the Raspberry Pi - See <https://github.com/Hexxeh/rpi-update>
- On-board Bluetooth disabled due to potential stability issues. Use rpi-update to check for kernel updates and see if it is fixed, and then edit /boot/config.txt to re-enable. Until they fix it, use USB Bluetooth dongle if needed.
- Installed and configured: mysql-server mysql-client phpmyadmin apache2 nagios3 nagios-nrpe-plugin



- To keep things consistent for those coming from NagiosPi, I have used the same passwords. MySQL is: User: root Pass: nagiosadmin
- Installed w3m web browser to allow local testing in terminal: w3m localhost/phpmyadmin
- Manually installed NConf 1.3.0-0 “Final”, an Enterprise Nagios configuration tool. This tool was broken on NagiosPi’s instructions due to a missing symlink at /var/www/nconf, so I fixed that in my version. Access NConf via the “Configure Nagios” link on the main menu.
- Includes NagVis 1.7 - want to do 1.9 but not until out of beta.
- Built and integrated the first version of our menu system, which includes the first version of a custom Nagios skin to begin integrating a more modern interface. Menu accessible at <http://nems/> (or <http://IPADDRESS> if that doesn’t work for you)
- Added a nice little MOTD.
- Added a simple cronjob to check our web site for the currently available version and warn you if yours is out of date.

## 3.41 NEMSeOS Changelogs

### 3.41.1 NEMSeOS 0 Changelogs

**December 1, 2020** - NEMSeOS 0.1 alpha Build 1.

Features:

- If no configuration found, auto-detect running NEMS Server | on same subnet when network connection is established.
- Ability to extend all GPIO pins hi/low value from running | NEMS Server.
- At release time, NEMS Warning Light GPIO pins, plus Omzlo | Warning Light pHAT are supported.
- Accessing IP address of NEMSeOS in browser will show state, | including an interval provided by the NEMS Server which | counts up with each iteration. This makes it easy to see | if your NEMS GPIO Extender has stopped transmitting or | receiving (as the counter would stop).
- Basic error handling included to show in browser if a | connection cannot be established.
- Manually editing the nems-tools.conf file from any | Linux machine is possible by mounting the SD card’s | boot partition.
- Editing your config file manually allows you to connect | to a NEMS Server on a different network, even WAN (Internet).
- NEMSeOS 0.1 alpha contains Cockpit running on port 9090 and | accessible from within your browser. Username/Password are | pi/raspberry. This is only included during alpha/beta testing | and will later be removed once the system is stable and can be | run fully as an appliance. Cockpit can be used for now to | login to the device’s terminal, or safely shut down the NEMSeOS | device.

Changelogs since Build 1

- Add i2c support to enable access to Omzlo Warning Light pHAT.



## 3.42 NEMS Mesh: The Self-Hosted NEMS Cloud

NEMS Mesh allows the connection of multiple NEMS Servers together in a pseudo-mesh style self-hosted cloud. This allows sysadmins to place NEMS Servers at client sites but poll and monitor them from one central NEMS Server.

### 3.42.1 How You Can Help

NEMS Mesh is under development and will be included in NEMS Linux when it is ready for production use. It requires funding to develop. Please consider [becoming a patron](#) to help fund this exciting feature.

## 3.43 NEMS Linux To Do List

This list details features which are planned for future releases of NEMS Linux. NEMS Linux is released on a semi-annual basis, with a major release usually falling in or around May and November. Thanks to the NEMS Linux Migrator, upgrading is a cinch, and thanks to NEMS Linux' rolling release system, many fixes, patches and upgrades can find their way into existing installations.

### 3.43.1 High-Priority Issues

- Backport the new NEMS Linux 1.6 speedtest to 1.5.x
- `memory_limit` in `php.ini` is default 128M. Increase to 512M to ensure NEMS Tactical Overview can load following an outage that increased the number of events to beyond 128M.

### 3.43.2 Feature Requests

This is a curation of various feature requests which have been approved for further assessment and possible implementation for the next major release of NEMS Linux.

You can submit a feature request either via Discord (in `#feature-requests`) or the Community Forum (in Feature Requests).

- Add [TrueNAS check command](#)
- [Custom icons as per Amheus](#) Possible duplicate of their [Discord message](#)
- Backup custom icons, should they exist, as per [Baldnerd's note](#).
- Attempt to port `check_juniper` to modern code [per Baldnerd's comment](#) in response to [Neptunum's request](#).
- Log a tabular speedtest report as per [Professor Eric](#)
- Explore [RobPickering's notes](#) about Let's Encrypt no longer requiring servers be publicly accessible. Something to integrate?
- Add custom Nagios theme support as per [Sly](#).
- While PoE fan should work fine in 1.6 due to upstream fixes from Raspberry Pi, test and make sure before release as per [jaffascout](#).

### 3.43.3 Features That I Like, But Can't Implement Yet

- PagerDuty integration as per [ElvisNuno](#).

### 3.43.4 During the 1.6 Release Cycle (Not At Launch)

- Add Elgato StreamDeck controller support via <https://github.com/abcminiuser/python-elgato-streamdeck>
- Add multi-tenant support as requested by Kevin Quiambao. Ability to add extra users who have access to certain features, such as Adagios / Nagios reporting. ULA if possible.
- Integrate notification tests for Telegram and Pushover.
- Move notification tests to NEMS Dashboard (rather than Linux terminal).
- Explore integration of ULA for staff.
- Add sound effects to NEMS TV Dashboard on state change as per BastyJuice. See <http://www.storiesinflight.com/html5/audio.html>
- NanoPi M4 Ethernet MAC address changes every reboot. Thanks to UltimateBugHunter for reporting.
- The TV output on ODROID-C1+ Build 1 doesn't work. Fix this.
- DONE in 1.6 - Add [NCPA](#) support.
- If NEMS is unable to communicate with github, a nems-upgrade will erroneously upgrade NEMS' version number even though the upgrade itself will have failed, as [reported by baggins](#).
- Documentation at the checkcommands level improved, along with other step-by-step guides added to the documentation.
- Add an audible alarm to NEMS TV Dashboard as per [ronjohntaylor](#).
- NEMS NConf interface revamped to match NEMS' overall look and feel. Branding improved. (Is a Patreon goal. Please consider supporting.)
- Evaluate [nconf PR # 4](#) for merge.
- Create NEMS Linux Docker container. (Is a Patreon goal. Please consider supporting.)
- Adagios interface customized to remove features not part of NEMS Linux.
- Make it so NEMS Off Site Backup sends the server the file size before the file, which will allow me to log an error if the user's file size exceeds the limit (rather than just silently failing).
- Take a look at [this report](#) and make sure it is not affecting users in 1.5.

### 3.43.5 User Requests to Review During Release Cycle

- Review this plugin and see if it's something that I can squeeze in: <https://forum.nemslinux.com/viewtopic.php?f=10&t=398> Not liking the fact that it is a Windows program.
- Check ssl as per [Zerant](#) (thought not particularly needed since check\_http supports this already).
- Add check\_pfsense as per [mydogboris](#).
- Veeam checks as per [Premium](#).
- Web interface to upload a backup.nems file for restoration as per LordOfLevels. Perhaps user can run *nems-restore webupload* and a code will be provided which can be entered into the form to confirm legitimate usage, and the restore will wait, checking a tmpdir for a file upload. Progress and status displayed within the bash window and restore prompts "Are you sure" as soon as the upload is complete.

## 3.44 Known Issues

These issues will be fixed in due time.

- Including a hard brace ([ or ]) in your host / service names will make it so Adagios cannot open the status page or history for that item. For now, avoid these.
- Number of services per page selection not working in Nagios Core as per [baggins](#).
- As per [Ricks](#) add ability for NEMS to auto-discover hosts and perhaps even compatible services. This has been added as a Patreon goal.
- As pointed out by [tripled](#), `check_nconf` treats ARG1 as command (-c) in its entirety. So we receive “NRPE: Command ‘check\_disk -w 80 -c 90 -p /media/pi’” as the full line is treated as command. It should be sending -c “check\_disk” -a “-w 80 -c 90 -p /media/pi” - will likely need to fix the `check_command` which may break it for those who hacked it up by changing the arg to, for example, `check_disk` -a “-w 80 -c 90 -p /media/pi
- Bland-X3 reports: “Have a bug/issue. `check_snmp` has issues with creating files when using the `-rate` switch (-C CommunityGoesHere -P 2c -o .1.3.6.1.2.1.2.2.1.10.1 -w 80000 -c 102400 -rate) results in Status Information: Cannot create directory: /usr/var/111 ... Can run command in CLI using sudo and does record the rate information which indicates simple permission problem or can modify the `check_snmp` command to save it somewhere else with permission maybe.” - Quick guess is maybe the command was run as root user from CLI which has now locked the nagios user from being able to access the folders. Will need to test. Also, see <https://support.nagios.com/forum/viewtopic.php?f=7&t=45864>
- Static IP doesn't persist after reboot in Virtual Appliance. Reported by [dr\\_patso](#) and confirmed by [Benedetti -Ale Morera-](#).
- [Benedetti -Ale Morera-](#) reported NEMS SST breaking TLS settings if enable background Blur and save. I have not been able to replicate this.
- `check_wmi_plus` has never contained the functionality for the documented `checkdns` feature. This has lead to some users saying the feature (which is an available check command in NEMS based on their docs) doesn't work. Need to remove this check command, and add a new one – perhaps `check_dns $HOSTNAME$ $HOSTADDRESS$` which will ensure the of the host matches.

### 3.44.1 Ideas for Future

There is not necessarily a planned timeline for each of these items, but here is a list of some of the things I do hope to do in an upcoming release. This list covers *potential* features for future releases of NEMS Linux. None of these are set in stone, and should only be considered ideas.

- Add [SNMP Trap support as per mpacey?](#)
- Write a language sub-system for the NEMS UI, allowing users to offer translation corrections via . Attempt to migrate the language system to all interfaces, including NConf, Adagios, and even nems-info and nems-init.
- Configure Adagios and NEMS to support multiple NEMS servers. See [this manpage](#). (It is possible NEMS Cloud will do away with this need).
- Evaluate [openITCockpit](#) as a possible front-end.
- Add feature to nems-migrator's off site backup that allows a user to request an email if their backup fails (can get the email info from NEMS SST and send email accordingly, separate of Nagios). Perhaps add a service check on the NEMS server instead? - Planning to add this feature to NEMS Cloud during the 1.5-1.6 release cycle.
- Add auto-discovery functionality. [this](#) and [this](#)?
- Make it so first boot automatically takes user into nems-init, with the option of instead running it through SSH.

- Move all commands from `commands.cfg` to `checkcommands.cfg` (or whichever is more appropriate) within NEMS Migrator.
- Add some generic true/false data to NEMS Anonymous Stats. In particular, discover if any users are using features like Telegram. By knowing this, I can decide if a feature should be removed from future releases.
- **Ability to use external storage for all active data.** Ideal for reducing read/writes on SD cards. Add interface to allow all active data to be saved to an external hard drive or network share as suggested by meveric.
- Build a graphical interface for `nems-init`.
- Build a graphical interface for NEMS Migrator's "Restore" feature.
- Add intrusion detection such as Snort or Bro IDS, as [per mpacey](#).

### 3.44.2 NEMS Linux Roadmap

- 1.0 - COMPLETE - Initial release. Bring easy deployment of Nagios to Raspberry Pi 3.
- 1.1 - COMPLETE - Creation of upgrade process, `nems-migrator` and optimize performance.
- 1.2 - COMPLETE - Creation of `nems-init` process to setup initial system. Create documentation.
- 1.3 - COMPLETE - Focus on feature set, add off site backup. Being laying the groundwork for upcoming 1.4 (in particular, non-Pi architectures).
- 1.4 - COMPLETE - New build of NEMS Linux featuring support for multiple SBC options and Nagios 4.
- 1.5 - COMPLETE - Focused on integrating user-requested options (mostly check commands) and optimizing the defaults/samples. Begin multi-server environment back-end, starting with ability to nickname NEMS servers via NEMS-SST. Introduction of more SBCs and virtual appliances.
- 1.6 - UX and feature updates. Refine the list of supported devices, removing any that are not being used. New peripheral options such as NEMS Warning Light.
- 1.7 - Deprecation of git as an update backend. Now, fully using `dpkg` repository. Complete deprecation of Python 2.7. NEMS Linux to be moved entirely to 64-bit image.
- 1.8 - Begin focusing on UX and feature consolidation. Remove unneeded features from NEMS Adagios. Create new interface for NConf that matches the NEMS Dashboard interface. Add a safe reboot button to NEMS SST. Migrate as many options away from the terminal as possible, including `nems-init`. Write `nems-mailtest` into `nems-sst` as [per mydogboris](#).

## 3.45 How To Report Issues

### 3.45.1 Submit an Issue

Please use the GitHub Issue Tracker for the appropriate repository. See [Source Code](#)

## 3.46 NEMS SSL and Self-Signed Certificates

### 3.46.1 My browser warns, “Your connection is not secure”. Why?

NEMS Linux uses SSL (aka https) connections to secure your connection and the data you transmit and receive to and from your NEMS server.

This is accomplished using what is called a *self-signed certificate*. By nature, self-signed certificates are considered “untrusted” by your browser because, simply put, anyone can make them. It does not mean your connection is not encrypted or secure, but rather it means your browser cannot determine who created the certificate, and therefore cannot verify your security.

If you visit a web site, say *google.com* and received a warning that your connection is not secure, you should immediately stop what you’re doing and not proceed. However, in the case of NEMS Linux, which is a local server on *your* network (not a “dot com” somewhere out on the web), you can safely trust the self-signed certificates and add an exception to your browser.

### 3.46.2 Where do the NEMS Self-Signed Certificates come from?

When you first deploy NEMS, a “default” certificate is included to help you get up and running. However, since this certificate is publicly available in the NEMS source code and img download, it is only used for your initial connection.

It can be a bit of a pain for novice users to setup SSL certificates, so like many other things with NEMS, I set out to make it easier.

When running `nems-init`, your unique SSL certificate is generated, added to your NEMS configuration and from then on all services will use your newly-created self-signed certificate.

### 3.46.3 What type of certificate is created?

NEMS generates self-signed certificates using `ssl-cert`. These self-signed certificates are called *snakeoil* because they do not use a certificate authority to authenticate validity. This means they provide a secure, encrypted connection between your client and NEMS server. However, they only provide trust to you and those who trust you (as you must accept the certificate / create an exception). This type of certificate is free and does not require you have your NEMS server publicly visible to the Internet.

### 3.46.4 Now that I have self-signed certs, how do I connect to NEMS?

Your browser will warn you that the site is untrusted the first time you connect. It will also provide an “Advanced” option where you can “Add Exception”.

### 3.46.5 I added a permanent exception, then reinstalled, re-initialized or upgraded NEMS, and now I can’t connect.

You need to remove the old certificates from your browser, restart the browser, and try again.

### 3.46.6 Why not use Let's Encrypt?

Let's Encrypt (and other certificate authorities) require the server be web-accessible so they can respond to challenges that prove the identity of the server. In order to utilize a service such as Let's Encrypt, your NEMS Linux server would have to be hosted on an FQDN and be web-accessible. I recommend against this since it's an attack vector that is completely unnecessary. The only advantage a CA gives you is trust, but you already trust your NEMS server as you are accessing it on a local IP.

---

**Note:** This may soon change based on [this feature request](#).

---

### 3.46.7 It's still not working.

#### Date and Time

Check (and fix) the date and time on both your NEMS server and your computer. If either are incorrect, your system will be unable to connect.

NEMS has NTP installed, so as long as you set your locale correctly during `nems-init`, the time and date should be correct. However, if your date is far inaccurate, NTP will reject the update. Therefore you must ensure your date and time are correct for NTP to work. See [this helpful tool](#) I have created to help you set the date and time quickly and accurately.

Most single board computers also require that you have an optional RTC battery installed as well. This maintains the date and time in event of power loss.

#### Security Software

Modern security software may by default block the use of self-signed certificates. This is a good security feature due to the inability to confirm trust of self-signed SSL connections that you do not have control over. You know you can trust your own NEMS server, but how do you know you can trust some obscure web site using a self-signed certificate? You can't.

I recommend against disabling this protection, as it is a good feature of anti-malware software. Instead, setup an exception for `nems.local`. Another idea is to setup an exception for your , or create your as a trusted zone.

## 3.47 Why support may ask for your `backup.nems` file

### 3.47.1 ... and why you should never share it with others

During support sessions I often request either SSH access or a copy of the user's `backup.nems` file. SSH access, it should be obvious, should not be shared with just anyone. Also, you should never, ever, ever, open SSH to the world on your NEMS server if you have not yet initialized it. This is because there are botnets that look for omputers which use the default passwords, and then compromise them.

SSH aside, as it is pretty obvious why it is sometimes the best way for me to provide support, I wanted to document why I may ask for your `backup.nems` file, what this file tells me, and why you should not just send it to anyone offering to help.

Your `backup.nems` file contains your complete NEMS configuration including those for your Nagios hosts, services, and your plain text `resource.cfg` file. This file contains your SMTP credentials which can be used by me to help diagnose email notification problems, but could also be used by a malicious user to compromise the mail account. This file also

contains the administrator login information for your Windows machines, as entered by you. This too can be used by me to help diagnose an issue with hosts not responding, but can also be used by a malicious user to compromise your systems should this information get into the wrong hands.

All this to say, store your *backup.nems* file as you would any other unencrypted backup set. It contains private information which is very helpful in diagnosing and helping resolve issues, but should be treated as a very confidential file.

---

**Note: UPDATE:** As of February 2, 2018, NEMS Migrator encrypts sensitive configuration files before compiling backup.nems if you enter an encryption/decryption password in NEMS SST. While the backup itself is unencrypted, this prevents access to sensitive data such as your passwords.

---

Please **do not** share your *backup.nems* file with third parties unless you are certain you can trust them. If you have any questions related to this document, please comment in the Community Forum.

## 3.48 How NEMS Linux Uses ZRAM To Maximize Memory

```

robbie@NEMS:~ $ free -h
               total        used        free      shared  buff/cache   available
Mem:           976M        192M        164M         25M         619M         707M
Swap:          976M           0B        976M

```

On June 23, 2017 I disabled swap on NEMS 1.2 to decrease the read/writes to NEMS server SD cards on the Raspberry Pi.

On February 5, 2018 you might notice your swap is ... well ... back. Or is it?

Rather than a traditional swap partition or file, I'm doing things differently with NEMS 1.3. Your pseudo swap is actually (are you ready for this?) RAM! Well, ZRAM to be exact!

The Raspberry Pi has very little RAM. 1GB. That's not much when it comes to all the things a NEMS Linux server has to do in its day to day. So how can I possibly give you more RAM on that wee hardware while not reducing the lifespan of your hardware? Allow me to explain...

ZRAM allows me to create a ramdisk that can be used as swap. But what's really unique about ZRAM (which is a Linux kernel module) is that it compresses the data in the ramdrive on the fly.

So here's how it works: your NEMS server starts to fill up the RAM and what happens? Well, it starts to swap stuff that should be in RAM to the swap. Only we disabled swap to extend the life of your SD card... so now, with this new ZRAM swap, it begins swapping instead to that. A swap "drive" that is actually compressed RAM. Therefore, if your RAM becomes full or close to full, it will begin swapping ... or rather ... compressing!

So, when you see the 1GB swap on your NEMS 1.3 server, don't fear! It is not using your SD card, and it's actually a brilliant way to increase the RAM capacity of your Raspberry Pi.

## 3.49 NEMS Licensing

Copyright © 2016-2021 Category5 TV Network.

NEMS Linux is licensed under the GNU General Public License Version 2.

### 3.49.1 GNU General Public License

#### Version 2

**License:** <https://www.gnu.org/licenses/old-licenses/gpl-2.0.en.html>

- NEMS Linux
- Nagios Core
- NagVis

### 3.49.2 Plugins

- *wmic* Copyright 2015 David Lundgren. MIT License. See [LICENSE](#).

### 3.49.3 Various Licenses

- NEMS Cloud Services colored icons (red, yellow, green) created by [1RadicalOne](#).

## 3.50 NEMS Branding

### 3.50.1 NEMS Linux

The NEMS Linux logo was created by [Outlndr](#).

#### Colors

- NEMS Green: #84c044
- NEMS Grey: #a7a9ac

#### Logo Options



- [nems-fullcolour.png](#)
- [nems-01.eps](#)





- nems-greyscale.png
- nems-03.eps



- nems-secondary-fullcolour.png
- nems-02.eps



- nems-secondary-greyscale.png
- nems-04.eps



- nems-secondary-white.png
- nems-06.eps
- nems-white.png
- nems-05.eps



### 3.50.2 NEMS Configurator

The NConf logo is an adaptation of the NEMS Linux logo. The adaptation was created by Robbie Ferguson.



- `nconf_logo.png`
- `nconf_logo.xcf`

### 3.50.3 NEMS TV Dashboard

The NEMS TV Dashboard logo was adapted from the original NEMS logo by [Outlndr](#) and a CC0 stock icon. Because it is typically only seen as a 16×16 favicon, the green of the NEMS logo was darkened to make it stand out.



- `tv-icon.png`

### 3.50.4 NEMS Cloud Services

An adaptation of the NEMS Linux logo, with the subtext using the AnjaliOldLipi font in #699bdf (light blue, representing the sky).

- `ncs.png`

## 3.51 NEMS Anonymous Stats

NEMS Linux sends anonymous, non-confidential usage data to the NEMS Linux Stats API. This data is aggregated by our system in a non-unique, entirely anonymous way.

---

**Note:** To ensure the privacy of our users' data, none of the data collected is identifying, nor is it even private information. We do not receive nor store any account information, passwords or the like.

---

The data included in the anonymous stats contain:



- Which platform you're running NEMS Linux on,
- How many hosts you have configured,
- How many services you have configured,
- Your NEMS Linux version,
- The size of your NEMS Linux storage medium (E.G., SD card),
- How much free space you have on that same medium,
- Your NEMS Linux server average system load,
- Your NEMS Linux server uptime,
- Your NEMS Linux server CPU temperature.

In order to uniquely, but anonymously identify your NEMS Linux server (E.G., so we do not duplicate data if you send it twice and so others cannot pretend to be you) your NEMS Linux server will also provide us with your NEMS Linux API Key (which it generates automatically) as well as your NEMS Linux Hardware ID, which you can see within [NEMS Server Overview](#).

We store this information to give us an idea how many people are using NEMS Linux, how many hosts they're monitoring, and how we can better improve performance based on the information provided. We also provide an aggregated stats display to help other users see the average performance of each platform.

To see how this data is made publicly available, please visit the [NEMS Linux Stats](#) page.

If you'd like to review what data we collect, please observe [the source code](#) or open [NEMS Server Overview](#) on your NEMS Server.

## 3.52 NEMS Linux Source Code

Are you a coder or want to explore some of the things NEMS does under the hood? Perhaps you'd like to **add some features** or **fix a bug** and then submit your own **pull request**. Or maybe you're not a coder but would still like to contribute by **reporting a bug or issue** you've found. This is all done through our GitHub repositories and the associated Issue trackers. **Feature Requests** may also be submitted via the appropriate issue tracker. If unsure, please ask the community on our Discord server.

NEMS Linux itself is a Linux distro built by pre-compiling several tools into a customized Debian server base. The features of NEMS is facilitated by combining these tools (Eg., Nagios, Monitorix, Webmin, Check\_MK) plus integration of a wide variety of custom applications to make NEMS a complete, well-rounded enterprise-ready distribution.

You can explore these custom scripts [on GitHub](#).

### 3.52.1 Debpak

Debpak is a custom front-end developed by NEMS Linux's developers and used by NEMS Linux's provision infrastructure, which compiles Debian packages for distribution via the NEMS Linux Debian repositories located at <https://repos.nemslinux.com/>

When browsing our source code, you will often see a folder called *debpak* in the root folder of the repository. Entering this folder will provide you with the root tree of the application. E.g., */debpak/usr/local/bin/nems-info* will place a file on the NEMS Server at */usr/local/bin/nems-info*

### 3.52.2 Official Repositories

#### NEMS Admin

*nems-admin* is where the scripts live that build the NEMS Linux distributable image. Generally users will never need this repository since it's only used by our development team to create new images, but if you are having a problem specifically with one of our distros (E.g., not booting on a new model Raspberry Pi) this would be the appropriate repository to post your issue. Think: if it has to do with the image itself (not the software contained therein), this is the place.

- Issue Tracker: <https://github.com/NEMSLinux/nems-admin/issues/>
- Source Code: <https://github.com/NEMSLinux/nems-admin/>

#### NEMS Migrator

*nems-migrator* is the suite of tools that provides an automated backup (both onsite and offsite) to NEMS Servers, and also provides the utilities to restore a backup to a new NEMS Server. If restoring your *backup.nems* file shows issues, this is where to report it.

- Issue Tracker: <https://github.com/NEMSLinux/nems-migrator/issues/>
- Source Code: <https://github.com/NEMSLinux/nems-migrator/>

#### NEMS Migrator Data

*nems-migrator-data* is where the sample data resides for NEMS Migrator. Generally this is not a place for end users. We separated the data from the main *nems-migrator* application to reduce the time to perform a *nems-update* when *nems-migrator* is updated (as the sample data is unlikely to change). NEMS Migrator uses the sample data not only for restoration of backups, but this data is also tapped into by tools such as *nems-init*, which is included in the *nems-scripts* repository. If you are having issues with NEMS Migrator, this is not the correct repository: You'd be looking for *nems-migrator* above.

- Issue Tracker: <https://github.com/NEMSLinux/nems-migrator-data/issues/>
- Source Code: <https://github.com/NEMSLinux/nems-migrator-data/>

## NEMS Scripts

*nems-scripts* contains custom scripts to perform a variety of NEMS Linux tasks. Examples would be the *nems-init* command and *nems-quickfix*. NEMS Scripts also contains the underlying code to provision the operation of checks such as *temper* and *speedtest*. If you are having an issue with a check command, it's more likely you're looking for *nems-plugins* below. NEMS Scripts is generally not user-serviceable and simply facilitates features our developers tap into in order to provide functionality to other tools within NEMS Linux.

- Issue Tracker: <https://github.com/NEMSLinux/nems-scripts/issues/>
- Source Code: <https://github.com/NEMSLinux/nems-scripts/>

## NEMS Plugins

*nems-plugins* are the check commands included in NEMS NConf. If you're having an issue with a particular check command, this is most likely the place to post an issue.

- Issue Tracker: <https://github.com/NEMSLinux/nems-plugins/issues/>
- Source Code: <https://github.com/NEMSLinux/nems-plugins/>

## NEMS Tools

*nems-tools* provides some “behind the scenes” functionality to NEMS Linux, such as NEMS GPIO Extender and NEMS Warning Light.

- Issue Tracker: <https://github.com/NEMSLinux/nems-tools/issues/>
- Source Code: <https://github.com/NEMSLinux/nems-tools/>

## nems-www

*nems-www* provides the web site interface and navigation of NEMS Dashboard.

- Issue Tracker: <https://github.com/NEMSLinux/nems-www/issues/>
- Source Code: <https://github.com/NEMSLinux/nems-www/>

## NEMS TV Dashboard

*nems-tv* provides NEMS TV Dashboard.

- Issue Tracker: <https://github.com/NEMSLinux/nems-tv/issues/>
- Source Code: <https://github.com/NEMSLinux/nems-tv/>

### NEMS NConf

*nconf* provides the PHP side of NEMS NConf. This system utilizes a MySQL database which exists within *nems-migrator-data*.

- Issue Tracker: <https://github.com/NEMSLinux/nconf/issues/>
- Source Code: <https://github.com/NEMSLinux/nconf/>

### WMIC

*wmic* provides Microsoft Windows' WMI compatibility for NEMS Linux (Windows Management Instrumentation).

- Issue Tracker: <https://github.com/NEMSLinux/wmic/issues/>
- Source Code: <https://github.com/NEMSLinux/wmic/>

### Hardware Detection

*hw-detect* allows NEMS Linux to detect and update the running NEMS Server's hardware profile.

- Issue Tracker: <https://github.com/NEMSLinux/hw-detect/issues/>
- Source Code: <https://github.com/NEMSLinux/hw-detect/>

### 9590

*9590* provides a simple tool to respond on port 9590 for testing TCP port up/down status. Part of the [NEMS Linux Getting Started Guide](#).

- Issue Tracker: <https://github.com/NEMSLinux/9590/issues/>
- Source Code: <https://github.com/NEMSLinux/9590/>

### NEMS Documentation

*nems-docs* is the Restructured Text source code for the NEMS Linux documentation found at <https://docs.nemslinux.com/> - if you contribute via a PR, please ensure you add your name to the credits.

- Issue Tracker: <https://github.com/NEMSLinux/nems-docs/issues/>
- Source Code: <https://github.com/NEMSLinux/nems-docs/>

## 3.53 NEMS Linux Release MD5 Checksums

### 3.53.1 Raspberry Pi Builds

#### 1.7

- 6c3d15f251846fbd077d52c50f103301 NEMS\_v1.7-RPi-Build1.zip

## 1.6

- c40f97aacc9d5f6568c3c48d8b6336e NEMS\_v1.6-RPi-Build2.img
- fbac40419f2f94437b8e610bb809c0b8 NEMS\_v1.6-RPi-Build2.zip
- e47961fccf8743d86fa8d2caa0fec09f NEMS\_v1.6-RPi-Build1.img
- 5b10dd3d924d6e4964b984de3375f23b NEMS\_v1.6-RPi-Build1.zip

## 1.5.2

- 8be7af5cb6515e914af3472e6934201f NEMS\_v1.5.2-RPi-Build1.img
- 70aa060653695ed93253c2411279cb89 NEMS\_v1.5.2-RPi-Build1.zip

## 1.5.1

- e6d098c39618e3d3a271448d41cc3eda NEMS\_v1.5.1-RPi-Build1.zip

## 1.5

- 3d2523357d13d1973aa84ab025f18aca NEMS\_v1.5-RPi-Build8.zip
- 09f36499f847d1846beb95512df2ba46 NEMS\_v1.5-RPi-Build8.img
- f3c5c2f49c40946a4183cd8a95a061bb NEMS\_v1.5-RPi-Build7.zip
- 31bd86c84fa2638d0639423d9c7cce41 NEMS\_v1.5-RPi-Build7.img
- 602415ce64ce0b318854b2e6072278ce NEMS\_v1.5-RPi-Build6.zip
- a1d1861f5a9217725415044d10a5490d NEMS\_v1.5-RPi-Build6.img

## 1.4

- b8028b05ca6527d4d73a666fe633fdfb NEMS\_v1.4.1-Pi.img
- 11fb629016ae28ea687cb0d46d35a3e1 NEMS\_v1.4-Pi.img

## 1.3

- 8f0d7d80442f05553807fbb4e12a332c NEMS\_v1.3.1.img
- c9b91e987d67e3d3e1ed9bb84e0e87a3 NEMS\_v1.3.img

## 1.2

- 9899069a83f4671d66bf1bf7eccc33b8 NEMS\_v1.2.1.img

### 3.53.2 Indiedroid Builds

#### Indiedroid Nova

## 1.6

- b56c2371d8f3c7c111067a12a54a6aeb NEMS\_v1.6-Indiedroid\_Nova-Build3.img
- 787473a7f854d2213604666182a16e44 NEMS\_v1.6-Indiedroid\_Nova-Build3.zip
- 29bbcc0860358d2ed18967fff0b92544 NEMS\_v1.6-Indiedroid\_Nova-Build2.img
- bf2e51e3a0771f27909ca11b13e64a34 NEMS\_v1.6-Indiedroid\_Nova-Build2.zip
- 0236f197fc15b6f91ab8d3934763c1ed NEMS\_v1.6-Indiedroid\_Nova-Build1.img
- 7f83810262db0c48d7df097c546a3378 NEMS\_v1.6-Indiedroid\_Nova-Build1.zip

### 3.53.3 Pine64 Builds

#### A64/A64+

## 1.5.2

- 346e6e3d72139ecd1e3f98d4dd3873ad NEMS\_v1.5.2-Pine64-Build1.img
- 2c4856d8c5b4a6c22bed92af82ca3b05 NEMS\_v1.5.2-Pine64-Build1.zip

## 1.5

- 987a6c09df737a3f62722433d2bba3f6 NEMS\_v1.5-Pine64-Build1.img
- ac508549a829021491cfa23aeb18a063 NEMS\_v1.5-Pine64-Build1.zip

## 1.4

- 02eb9bc57d359ea94d793e09d11f0df1 NEMS\_v1.4.1-Pine64-Build3.img
- 75d2131bf6049b5f7d75f841d58e0809 NEMS\_v1.4.1-Pine64-Build2.img



## A64-LTS/SOPine

### 1.5

- 887823e3dd4578274cdbff0e9402b7d9 NEMS\_v1.5-SOPine-Build1.img
- 5ad0d684296d50b4c1fcbac6db205ae0 NEMS\_v1.5-SOPine-Build1.zip

### 1.4

- a48a41c88941aeca69613e93da720e92 NEMS\_v1.4.1-SOPINE-Build1.img

## Rock64

### 1.5

- e33b4d2f89aeffc2ddf9e2f6e42b1c12 NEMS\_v1.5-Rock64-Build2.img
- 6e2088922c5d197db8b8ba3057120389 NEMS\_v1.5-Rock64-Build2.zip
- bc2f3809bc37e34986874298aa623d4f NEMS\_v1.5-Rock64-Build1.zip
- 57613adec04b6f49374574f8e82d9d3a NEMS\_v1.5-Rock64-Build1.img

### 1.4

- 2875f65b78082fd3d73659c089261ae8 NEMS\_v1.4.1-Rock64-Build1.img

## RockPro64

### 1.5

- 1da5a4e5d35b8e6dc2f5e9927beb4cd3 NEMS\_v1.5-RockPro64-Build1.img
- 2627bc0aa81e1c55de69a621d80987a5 NEMS\_v1.5-RockPro64-Build1.zip

## 3.53.4 ODROID Builds

### ODROID XU3/XU4/HC1/HC2/MC1

### 1.6

- 7c72b85315aaded76596c8075a76eddb NEMS\_v1.6-ODROID-XU4-Build1.img
- 8a2d3165b2a9f79224f295076bd6642a NEMS\_v1.6-ODROID-XU4-Build1.zip

### 1.5.1

- 470872217bbc082ccd81e1708f877b8c NEMS\_v1.5.1-ODROID-XU4-Build1.zip

### 1.5

- 3e7c2ba384e73ef4fc46dc09ab1ba715 NEMS\_v1.5-ODROID-XU4-Build4.zip
- 9cac26e9f4a27fecdef941f2c5f507e5 NEMS\_v1.5-ODROID-XU4-Build4.img
- d1e725d3f3a8f621a929601138ec9e47 NEMS\_v1.5-ODROID-XU4-Build3.img
- 69f50428afe0999ccd77449705b5e8ad NEMS\_v1.5-ODROID-XU4-Build3.zip

### 1.4

- 9740577f4b4471bfd3387ab403ec56ab NEMS\_v1.4.1-ODROID-XU4-Build3.img
- 55dbeb8c712f842d8ca3739f98b93d0f NEMS\_v1.4.1-ODROID-XU4-Build2.img
- 637d4174131e2959abe4f0325b23ff33 NEMS\_v1.4.1-ODROID-XU4-Build1.img

## ODROID-N2

### 1.5

- b7de14604dccef0aab36cdd523bd6389 NEMS\_v1.5-ODROID-N2-Build2.zip
- 3121d96f7683689b89425d0658ad288d NEMS\_v1.5-ODROID-N2-Build2.img
- b09f0efff78442b12e09d63a01c4afaa NEMS\_v1.5-ODROID-N2-Build1.img
- 9e027d027be0adca5cc2f6ab122f4c53 NEMS\_v1.5-ODROID-N2-Build1.zip

## ODROID-C2

### 1.5.1

- 98bc1ca75e6d4c4faae6f27796772849 NEMS\_v1.5.1-ODROID-C2-Build2.zip

### 1.5

- 276daaa18dace3741ec19d71d9707a43 NEMS\_v1.5-ODROID-C2-Build2.zip
- b80ecc26abaebc4c7abb25d0dd382beb NEMS\_v1.5-ODROID-C2-Build2.img
- 60196ce0ec16919794d3002ad8cc7101 NEMS\_v1.5-ODROID-C2-Build1.zip
- 49b064188bd8174f6ae1ce053bbcb6f4 NEMS\_v1.5-ODROID-C2-Build1.img

## ODROID-C0/C1/C1+

### 1.5

- 804c54628d7b37545b0de57c773c3aa2 NEMS\_v1.5-ODROID-C1-Build1.zip
- 5cd5eb77810ab72f04b9c8a1a50868b8 NEMS\_v1.5-ODROID-C1-Build1.img

## 3.53.5 FriendlyElec Builds

### NanoPi M4

#### 1.5

- c89326fdb26cb7f9a691f876ad9c85a6 NEMS\_v1.5-NanoPi\_M4-Build1.zip
- 25ca4ba6c466a2846d15d83be70339f2 NEMS\_v1.5-NanoPi\_M4-Build1.img

### NanoPi NEO Plus2

- dbc1285e6509e396fd12474dfbd703d1 NEMS\_v1.5-NanoPi-NEO-Plus2-Build1.img
- a7de161271e2070c0a63296e377f035e NEMS\_v1.5-NanoPi-NEO-Plus2-Build1.zip

## 3.53.6 ASUS Builds

### Tinker Board / Tinker Board S

#### 1.6

- 1b217e682a818b8597de8a7c9d45724b NEMS\_v1.6-ASUS\_Tinker\_Board-Build1.zip
- 7cb75b033e1b681befb8139c72227a5a NEMS\_v1.6-ASUS\_Tinker\_Board-Build1.img

#### 1.5

- e8864fbd50ed9bd3bcb7314b25875d01 NEMS\_v1.5-ASUS\_TinkerBoard-Build2.zip
- 68730d3155f82dc38db9f9e1da9dd934 NEMS\_v1.5-ASUS\_TinkerBoard-Build2.img
- 8ab85bc568025aca73bd1fe1cbd10748 NEMS\_v1.5-ASUS-Tinker-Board-Build1.zip
- a19f781b73269d893c8a70f38f81e203 NEMS\_v1.5-ASUS-Tinker-Board-Build1.img

### 3.53.7 Orange Pi Builds

#### Orange Pi Zero

##### 1.5

- 98f66a3a3764e3b7d6267af180665f94 NEMS\_v1.5-Orange-Pi-Zero-Build1.img
- 0049bfc4780c9b8f67eda91064340434 NEMS\_v1.5-Orange-Pi-Zero-Build1.zip

### 3.53.8 Khadas Builds

#### Khadas VIM3

##### 1.6

- edfc21bc6dfc43563550c430d964c194 NEMS\_v1.6-Khadas\_VIM3-Build1.img
- c51463a103200ecbef5a64d4076769cb NEMS\_v1.6-Khadas\_VIM3-Build1.zip

##### 1.5

- 962921cb557747b998d5a40b27306669 NEMS\_v1.5-Khadas-VIM3-Build1.img
- 50f44ea866b729be7fb2c940c1f2fcb4 NEMS\_v1.5-Khadas-VIM3-Build1.zip

### 3.53.9 Virtual Appliances

#### OVA

##### 1.6

- 8e3ec6cf92d5a773e4ee50fd4559eb5c NEMS\_v1.6-VM-Build2.ova
- 3004a56066ccbf87719a8856fdb7b78 NEMS\_v1.6-VM-Build1.ova

##### 1.5

- 5902ea98343ad4f7bc467f2ada416add NEMS Linux 1.5 OVA Build 4.ova

#### QCOW2

##### 1.6

- df606e7f844b5e9975f9935fd417aa11 NEMS\_v1.6-VM-Build1.qcow2
- 0077444b24dc458faad143c434b81ae7 NEMS\_v1.6-VM\_QCOW2-Build1.zip

## 1.5

- 69a7f27f2a3868c94b9628d06e1d6f75 NEMS Linux 1.5 QCOW2 Build 2.zip

## VHD

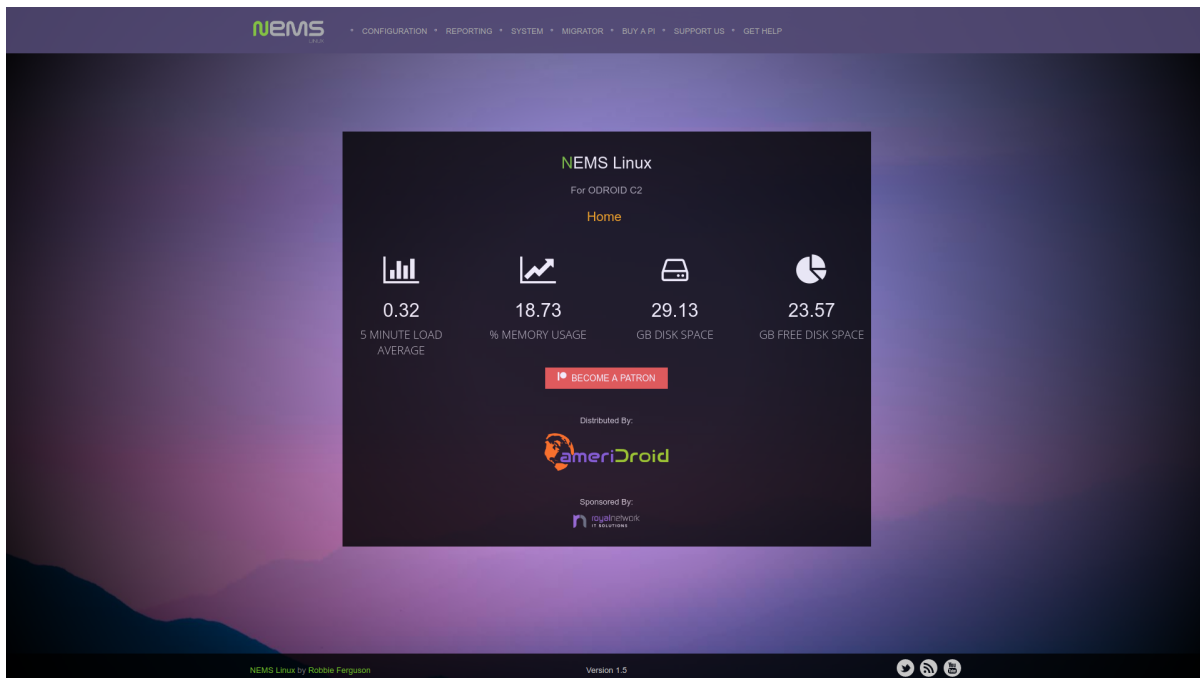
## 1.6

- 03456d84d4eb716106a963deb53cab93 NEMS\_v1.6-VM-Build1.vhd
- cb0f4c7d9f5dce1e4c73646f435ef064 NEMS\_v1.6-VM\_VHD-Build1.zip

## 1.5

- 766d7cc164dfc07be558cefb477effe9 NEMS Linux 1.5 VHD Build 2.zip

## 3.54 NEMS Linux Vendor Branding



Whether you're a vendor, distributor or IT professional, it's nice to include your own branding if you install a NEMS Server at a client site, or if you're building and selling NEMS Linux appliances.

Currently the NEMS Linux Vendor Branding feature supports:

1. Vendor Logo
2. Vendor URL (optional)
3. Vendor License (optional)

When you add your vendor logo, it will be scaled to a maximum height of 60 pixels and placed on the NEMS Dashboard above sponsor logos, and the sponsor logos will be shrunk to a smaller size. If you include a URL, the logo will be clickable and will open a new tab to your URL when clicked.

### 3.54.1 Add Your Vendor Branding

---

**Tip:** When creating your vendor image master, do not boot the image. Instead, add your vendor files using a card reader on a computer, and then create an img of the card containing your vendor files. If you boot the system from your card, the filesystem will be resized to fill the card, so your resulting img will be huge. You may then have trouble re-imaging it to new cards since the capacity from card to card may vary by a byte or two.

---

1. Mount your NEMS Linux drive (Eg., SD card) on your computer.
2. Place your logo in the *vendor* folder on the *boot* partition. It must be named **logo.png**
3. If you would like your logo to be clickable (eg., link to your company web site), create a file in that same folder called **url.txt** and paste your full URL into the file.
4. A Vendor License may be purchased, if desired. If a valid Vendor License is included, all users who share this build will be able to utilize your NEMS Cloud Services account. This means you will be able to monitor every server that is distributed under your account. This feature is intended for IT administrators who need to monitor widespread networks.

### 3.54.2 Logo Tips

- Your logo should have a transparent background.
- Your logo should scale nicely and be recognizable when it is scaled to 60 pixels high.
- You should assume the logo will always be presented on a dark background. If your logo is dark, consider adding a light border to your logo.png file.
- URLs need to be complete. For example, **nemslinux.com** is incorrect, whereas **https://nemslinux.com/** is correct.
- Be mindful that you are placing your logo file within the *boot* partition, which is generally quite small. Ensure your file is of reasonable size, especially considering it will be scaled down to a height of 60 pixels. So a massive high-res logo.png is really unnecessary. If you run out of space on *boot* you could experience problems.

### 3.54.3 Whitelabel Branding

Whitelabel branding of NEMS Linux is available on a subscription basis for enterprise users looking to fully brand NEMS Linux to their company. Whitelabel service includes VIP support, custom feature production, and is always confidential. NEMS Linux is in use within government, healthcare, telecommunications and travel (both on earth and beyond).

To enquire about Whitelabel Branding, please contact [nems@category5.tv](mailto:nems@category5.tv) (serious enquiries only).

## 3.55 Resolving DNS Hostnames on LAN

### 3.55.1 Introduction

NEMS Linux uses two main Linux components to resolve \*.local hostnames:

1. **avahi-daemon** - This allows other computers to see the *nems.local* host on the LAN.
2. **libnss-mdns** - This allows NEMS Linux to see other \*.local computers on the LAN.

### 3.55.2 Enabling Multicast Name Resolution

In order for NEMS Linux to be able to resolve a DNS hostname to its IP address, multicast must be enabled on the client system.

Most modern systems will already provide multicast in one shape or another, so you only need to take action if a host is not replying to its hostname.

- **Microsoft Windows Target** - Multicast is provided by the *Link-Local Multicast Name Resolution (LLMNR)* service which can be enabled or disabled using the computer's group policy (Local Computer Policy -> Computer Configuration -> Administrative Templates -> Network -> DNS Client). Otherwise, it is also included with a local installation of iTunes, or alternatively you can install [Apple's Bonjour service](#).
- **Linux Target** - Simply install *avahi-daemon*. It will automatically enable itself during installation. *avahi-daemon* is in your distro's official repositories, so can be installed via any package manager.
- **macOS Target** - No need to install anything; it's already included.

---

**Note:** If the target computer has a firewall enabled, you'll also need to allow UDP traffic through ports 5353 and 53791.

---

## 3.56 Documentation Contributors

The following people have contributed to the NEMS Linux documentation:

- Robbie Ferguson
- Bill Marshall
- Don Jenkins
- Will Blanton
- Luke Kabat

Sending a Pull Request? Please make sure to also add your name to this list.

## 3.57 NEMS SaaS

### 3.57.1 NEMS SaaS API

#### Trends API

Obtain the JSON output of your NEMS SaaS endpoint trends for a duration of time.

`https://api-saas.nemslinux.com/trends/SAASKEY?ver=[num]`

SAASKEY = Your NEMS SaaS Key

ver:

- 0 = [Default] The current state of your NEMS SaaS endpoints
- 1 = The last hour's trends for your NEMS SaaS endpoints
- 2 = The past 6 hours trends
- 3 = The past 12 hours trends
- 4 = 24 hour trend for your NEMS SaaS endpoints
- 5 = 7 days worth of trends
- 6 = 1 month trend
- 7 = Last 10 state changes
- 8 = Last 25 state changes
- 9 = Last 50 state changes

Sample JSON Output:

```
{"results":[{"ok":4,"warn":0,"crit":0,"statechange":"2022-08-27 07:51:40"}]}
```

Results are presented most recent to oldest. Timezone of statechange matches the user's defined timezone.

### 3.57.2 NEMS SaaS Probe

#### Introduction

The NEMS SaaS Probe is a small daemon which runs on your endpoints, collecting performance data about the endpoint. This data is then sent to NEMS SaaS, where it is aggregated and provided to your account for reporting and alerting.

The NEMS SaaS Probe is a benign application which gathers non-identifying information such as CPU load, disk space usage, and so-on.

The NEMS SaaS Probe is designed to be easy to deploy, with no firewall rules to create and no check commands to configure.

Its source code is available for your review at <https://github.com/NEMSLinux/nems-saas-probe/blob/main/src/nems-saas-probe>



## Security

NEMS SaaS Probe sends your data packet to the NEMS SaaS servers via an encrypted connection, which is authenticated to your account with your NEMS SaaS Key.

The NEMS SaaS built-in JSON web server provides JSON responses only when the correct NEMS SaaS Key is provided as POST data. This server is intended for LAN or VPN use. If (and only if) absolutely necessary, please consider opening port 6367 only to the IP addresses you will be running your application on. While unlikely, it should be noted that a MITM attack could steal your NEMS SaaS Key if you connect to port 6367 successfully from an Internet location (as opposed to LAN or VPN). Do not open port 6367 to the world.

## Built-In Web Server

As of version 1.0.050, the NEMS SaaS Probe is able to serve JSON responses containing the data packet. The server is running by default on port 6367.

This is provided as a helpful feature to developers and power users, but is a requirement of using NEMS SaaS. You do not need to configure your firewall to use this server if you are not a developer tapping into the JSON data, and an in-house NEMS Server (on the same LAN) can use this data to monitor your endpoints in-house.

You may optionally parse this data in your own app, or create a NEMS Linux check command to query this data.

In addition to the standard data packet, there are two other defined keys:

### packet

Packet will be 0 or 1. 0 means the packet has not succeeded in sending to NEMS SaaS. 1 means the packet was successfully sent to NEMS SaaS. For more details on why a packet has not been sent, check your nems-saas.log on the system running the NEMS SaaS Probe.

### timer

Timer is the value, in seconds, of how long the NEMS SaaS Probe is waiting before it sends the next packet to the NEMS SaaS servers. This number is dynamic and changes with each packet. It can be used to set a sleep timer on your own queries to the probe. There is no point in querying the probe again sooner since the probe will not have any new information to report (the packet will be the same).

## Disable

Turn off the built-in web server with the `--noserver` command line argument.

## Query Example

Your query must contain a POST containing only your account's NEMS SaaS Key (found in your user settings when logged into NEMS SaaS).

```
curl --http0.9 -d "000000000-0000-0000-0000-000000000" -X POST http://192.168.0.5:6367
```

Replace 000000000-0000-0000-0000-000000000 with your actual NEMS SaaS Key.

Replace 192.168.0.5 with the actual IP of the system running the probe.

### 3.57.3 NEMS SaaS Probe Installation on QNAP NAS

#### Introduction

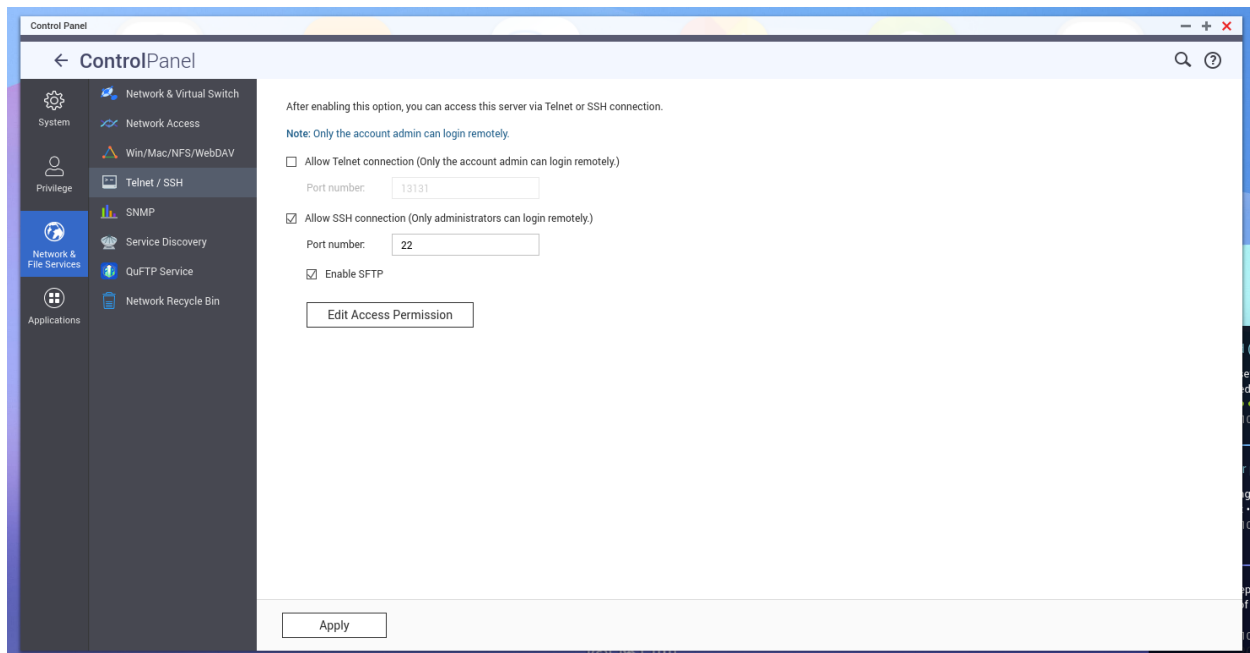
QNAP NAS devices run a Linux distribution called [QTS](#). As such, the NEMS SaaS probe is able to monitor your QNAP NAS device as if it was any other Linux server.

#### Ensure Your QNAP Is Prepared

##### Step 1. Enable SSH Access.

In order to access the Linux terminal on your QNAP device, you must enable SSH access.

Open Control Panel -> Network & File Services -> Telnet / SSH and place a checkbox next to Allow SSH Connections.



If you use a different account than `admin` as your administrator account (which you should!) click `Edit Access Permissions` and add your user.

Once you're finished, you can later return to this screen to disable SSH access.

##### Step 2. Connect to your QNAP SSH Terminal.

##### Step 3. Install nems-saas-config

##### Step 4. Install nems-saas-probe

```
sudo wget -O /usr/local/bin/nems-saas-probe https://github.com/NEMSLinux/nems-saas-probe/raw/main/linux/amd64/nems-saas-probe && sudo chmod +x /usr/local/bin/nems-saas-probe
```

**Step 5. Enable auto-load at boot.**



## IMPORTANT LINKS

- Support the project: <https://patreon.com/nems/>
- Web site: <https://nemslinux.com/>



## CONTRIBUTE TO DOCUMENTATION

- Source Code: <https://github.com/NEMSLinux/nems-docs/>





## **SUPPORT**

As NEMS Linux continues to grow, our support options are also evolving. At present, there are a few key points to keep in mind when seeking support:

1. NEMS Linux is a community-driven project, so the best place to start is usually the community. We offer an [official Discord server](#) and a [Community Forum](#) to help you find what you need.
2. Our lead developer, Robbie Ferguson (*RobbieF#7136*), is often available via our Discord server. That is the best place to reach him directly.
3. Among other perks, users who choose to [support NEMS Linux on Patreon](#) gain access to Patron-exclusive rooms on our Discord server, including `#priority-support`, which are setup to ding lead developer, Robbie Ferguson, on his phone. So in cases where priority support is required, he's usually quite good to reply promptly.



## CREDITS

Please see the [NEMS Linux Credits](#) page.



## PATRONS

I'd like to thank all of our Patrons for your continued support of NEMS Linux development.

Here is a list of those Patrons who kicked in that little bit extra to have their name included in the changlogs:

<https://raw.githubusercontent.com/NEMSLinux/nems-www/main/debpack/var/www/html/credits/credits.txt>

Want your name on this list? [Become a Patron](#).

### A Product Of



NEMS Linux is developed by Robbie Ferguson for [The Category5 TV Network](#).



## **LICENSE**

NEMS Linux and the code written specifically for it are licensed under GNU AGPLv3: <https://www.gnu.org/licenses/agpl-3.0.en.html>

Third-party products or features included within NEMS Linux may have their own license. Please review their project pages for more information.

Background wallpaper contained in NEMS Linux are licensed under CC0.





## Symbols

-? command line option, 50

-A command line option, 49

-C command line option, 49

-H command line option, 49

-M command line option, 49

-P command line option, 49

-R command line option, 49

-S command line option, 49

-T command line option, 49

-U command line option, 49

-V command line option, 49

-X command line option, 50

--authpassword command line option, 49

--authprotocol command line option, 50

--community command line option, 49

--critical command line option, 49

--help command line option, 50

--host command line option, 49

--mode command line option, 49

--port command line option, 49

--privpassword

command line option, 50

--privprotocol command line option, 50

--reset command line option, 49

--serial command line option, 49

--slave command line option, 49

--type command line option, 49

--username command line option, 49

--version command line option, 49

--vpnmode command line option, 49

--warning command line option, 49

-a command line option, 50

-c command line option, 49

-s command line option, 49

-v command line option, 49

-w command line option, 49

-x command line option, 50

## C

command line option

-?, 50

-A, 49

-C, 49

-H, 49

-M, 49

-P, 49

-R, 49

-S, 49

- T, 49
- U, 49
- V, 49
- X, 50
- authpassword, 49
- authprotocol, 50
- community, 49
- critical, 49
- help, 50
- host, 49
- mode, 49
- port, 49
- privpassword, 50
- privprotocol, 50
- reset, 49
- serial, 49
- slave, 49
- type, 49
- username, 49
- version, 49
- vpnmode, 49
- warning, 49
- a, 50
- c, 49
- s, 49
- v, 49
- w, 49
- x, 50