

ADMIN

Network & Security

ISSUE 79

Monitoring

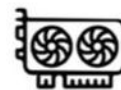
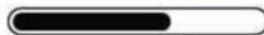
Dashy, LibreNMS, Tier 0 Systems, Graphite**Authelia IAM****ScaleFlux Computational Storage****OSSEC****Intrusion detection and
host-based intrusion prevention****DDoS Defense****Restoring Hybrid Identities****DIY Docker Images****Build and host Docker
images****Local Azure Arc****Manage on-premises
servers in Azure****Kubernetes Backups****Dos and don'ts of
backing up K8s storage****GitOps****Synchronize repository
changes****LINUX NEW MEDIA**
The Pulse of Open Source



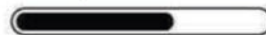
all-AMD gaming laptop TUXEDO Sirius 16 - Gen1



Mobility



GFX performance



AMD Power

Highly efficient athlete TUXEDO Pulse 14 - Gen3



Mobility



Battery life



Linux compatible



Up to 5 Years Guarantee



Immediately ready for use

TUXEDO



Made in Germany



German Data Privacy



German Tech Support



tuxe.do/lxadmin79

Technology Predictions for 2024 and Beyond

I must admit that I really hate it when I read industry pundit predictions for the next year. No, seriously, I really hate the predictions because they're generally based on nothing but conjecture or the hope of filling a page or perhaps starting a flame war to drive up page views.

I enjoy writing pieces like this as much as I enjoy writing an article in "corporate speak" or the latest marketing buzzword-laden rant about some aspect of mundane technology. I do rather enjoy being an "in your face" antagonist of sorts. Maybe it's age. Maybe it's my own personality flaw. Or maybe it's my techno-lingo-jaded self that wants to scream every time I hear terms such as single pane of glass, hyper-converged architecture, or quiet cutting. Yeah, it's that last one, for sure.

Wait no longer. Here are my 10 technology predictions for 2024 and beyond. Be warned that these predictions are chock-full of corporate speak, buzzwords, and my own sarcastic creations.

1. Industry pundits will predict massive artificial intelligence (AI) adoption across all business sectors.
2. Technology companies will make more "tough decisions" and perform more quiet cutting of their US-based workforce.
3. An industry pundit will predict "The 2024 Tech Boom."
4. A different industry pundit will predict "The Great 2024 Tech Crash."
5. A technology writer will post an article titled "2024: The Year of the Linux Desktop."
6. Industry analysts will predict that AI will replace tech workers.
7. Someone will write an article titled "The 10 Best Tech Stocks to Invest in Right Now."
8. A major financial magazine will publish an article discussing "The Great 2024 Tech Stock Selloff."
9. There will be a new Facebook hoax that will hit the national newsfeeds as fact.
10. Elon Musk will buy the Moon and charge us a monthly subscription fee to look at it.

Yes, these predictions are meant to be humorous, but I wish someone would give me \$10,000 for each one that comes true. I wouldn't be able to retire, but I'd have a nice vacation and some new camera equipment to show for it.

I've predicted for 20 years that tech workers will someday band together and create a Tech Workers Union. I know there have been a couple of solid attempts over the years, and there might still be one or two out there, but they haven't made enough of an impact to gather a big following. I attempted to start such a union in the early 2000s but was threatened by my employer, so I scrapped the project. Organizing a union does me no good if I have no oppressive and exploitative job to defend myself against. The company also sent a very strong message to all others with similar aspirations. I'm still hopeful that, someday, tech workers will unite and protect themselves, but perhaps by then, the prediction of mass tech worker extinction at the hands of AI will have come true.

Whatever technology predictions come true for 2024, you can place your money on one sure thing – none of them will benefit the tech worker in any capacity. We are at the bottom of the corporate food chain, supply all the labor, and keep the business running from the trenches, and yet, we are the ones that have targets on our backs. I think it would be far more profitable to keep the lowly tech workers employed and replace our highly paid overlords with their much lower cost AI alternatives. Think of the millions of dollars that move would save. But I predict that it won't happen. Not in 2024 or my lifetime, anyway.

Ken Hess • ADMIN Senior Editor

ADMIN

Network & Security

Features

- 10 Dashy**
Create your own command center in small infrastructures or test environments with flexible dashboards that control and monitor relevant applications and services.
- 14 LibreNMS**
Check out this monitoring environment with auto-discovery, alerting, the ability to scale even in very large environments with many devices, and flexible dashboards and widgets for special views.
- 19 Tier 0 Monitoring**
We show you how monitoring your sensitive IT systems can be a more secure experience.
- 24 Graphite**
This open source tool offers real-time monitoring with comprehensive and fast data collection from virtually any system.

Tools

- 28 AlmaLinux OS**
The AlmaLinux Build System lets you build, test, sign, and release packages from a single interface.
- 32 Authelia**
Add access controls to web applications that do not have their own user administration.
- 34 iRedMail**
Deploy this full-featured email server on a number of platforms in a matter of minutes.
- 40 ScaleFlux**
See performance gains of 50 percent and more with computing power built directly into the network card.

Containers and Virtualization

- 46 DIY Docker Images**
When facing the challenge of packaging your application in a container, take into account your needs in terms of handling and security and investigate sensible options for hosting your own registry.
- 52 Kubernetes Backups**
Stateful applications that store their information in a container's persistent volume can be backed up in a variety of ways.
- 56 Local Azure Arc**
Manage your on-premises servers with Windows Admin Center in Azure.

Security

- 60 OSSEC**
Detect and fix security problems in real time at the operating system level with functions such as log analysis, file integrity checks, Windows registry monitoring, and rootkit detection.

Service

- 3 Welcome**
- 6 News**
- 97 Back Issues**
- 98 Call for Papers**

10, 14, 19, 24 | **Monitoring****Tools for your IT systems**

This issue takes a deep dive into monitoring solutions for your IT infrastructure, including Dashy, LibreNMS, Tier 0 systems, and Graphite.

Highlights**40 ScaleFlux**

Relieve load on the CPU and ensure higher bandwidth and lower latency by simply replacing your storage hardware with computational storage.

46 DIY Docker Images

Building Docker containers is not particularly complicated, and CI/CD tools and a good DIY image registry will help make the experience more convenient.

60 OSSEC

This powerful free intrusion detection and host-based intrusion prevention system can detect and combat malware and cyberattacks and can even be run on a virtual machine.

Management**64 Restoring Hybrid Identities**

We look into contingency measures for hybrid directory services with Entra ID, the Graph API, and its PowerShell implementation.

70 Ralph Asset Management

Keep things simple, without compromising flexibility, when managing data centers with this open source asset management system and configuration database.

Nuts and Bolts**74 GitOps**

Applying DevOps practices through infrastructure automation of version control repositories is a popular approach to synchronizing system changes, both in and outside the context of Kubernetes.

80 BGP Routing Protocol

We look at the Border Gateway Protocol, how it routes packets through the Internet, its weaknesses, and some hardening strategies.

86 DDos Defense

Targeted attacks cannot be prevented, but they can be effectively mitigated.

91 Terminating OpenSSH

Disconnect OpenSSH user sessions after a certain period of inactivity with the systemd-logind service.

93 Performance Dojo

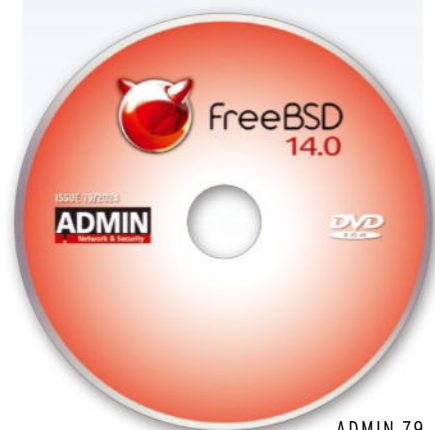
Exploring low-cost parallel computing.

On the DVD**FreeBSD 14.0**

The FreeBSD open source operating system derives from 4.4 BSD Lite2 and can be used for everything from software development to games to Internet service provision. The base distribution has full source code for the kernel and all utilities, and the Ports Collection makes it easy to install your favorite traditional Unix utilities. You will also find:

- PIE support enabled by default
- Unprivileged operation with chroot
- Support for NFS-over-TLS with two new daemons

Before installing, check the errata document online (<https://www.freebsd.org/releases/14.0R/errata/>).



@adminmagazine



@adminmag



ADMIN magazine



@adminmagazine

News for Admins

Tech News

Vim 9.1 Now Available

The Vim project has announced the release of version 9.1 of the popular text editor.

“This release is dedicated to Bram Moolenaar, Vim’s lead developer for more than 30 years, who passed away half a year ago,” the announcement states. “The Vim project wouldn’t exist without his work!”

Vim 9.1 is mainly a bug fix release, but it also offers new features and improvements, such as:

- Smooth scroll support
- New `:defer` command to help with cleaning up a function
- Support for adding virtual-text to a buffer
- Support for Vim9 classes and objects for the Vim9 scripting language

Download the latest release from the Vim website (<https://www.vim.org/download.php>).

Microsoft Introduces Copilot Key to PC Keyboards

Microsoft has announced the addition of a new Copilot key for Windows PC keyboards, marking the first significant change to the keyboard in nearly three decades.

“The Copilot key joins the Windows key as a core part of the PC keyboard,” says Yusuf Mehdi in the announcement (<https://blogs.windows.com/windowsexperience/2024/01/04/introducing-a-new-copilot-key-to-kick-off-the-year-of-ai-powered-windows-pcs/>), “and when pressed, the new key will invoke the Copilot in Windows experience.”

A company disclaimer notes that the timing and availability of the Copilot feature will vary by market and device and will require a Microsoft account.

Google Announces AI Hypercomputer

In addition to its release of the Gemini AI model (<https://www.fosslife.org/google-announces-gemini-ai-model>), Google has announced an AI Hypercomputer, “a groundbreaking supercomputer architecture that employs an integrated system of performance-optimized hardware, open software, leading ML frameworks, and flexible consumption models.”

According to the announcement, “AI Hypercomputer employs systems-level codesign to boost efficiency and productivity across AI training, tuning, and serving.”

The AI Hypercomputer “features performance-optimized compute, storage, and networking” as well as out-of-the-box support for “popular ML frameworks such as JAX, TensorFlow, and PyTorch,” the announcement says.



**Get the latest
IT and HPC news
in your inbox**

**Subscribe free to
ADMIN Update
and HPC Update
bit.ly/HPC-ADMIN-Update**

Want to work from anywhere?



Find remote jobs now!

OpenSource
JOB HUB



opensourcejobhub.com/jobs

Best Practices to Prepare for Post-Quantum Security Challenges

“The rise of quantum computing introduces a unique set of challenges related to the management of digital certificates,” says Murali Palanisamy (of AppViewX) on the RSA Conference Library blog.

“Unfortunately, today’s encryption is widely understood to be vulnerable to quantum attacks; researchers have already shown a quantum computer could break some of the tougher encryption used today (considered unbreakable until now) in 104 days (<https://www.fujitsu.com/global/about/resources/news/press-releases/2023/0123-01.html>). Palanisamy, he notes, as quantum computing evolves, that time could shrink to hours, minutes, or only seconds.

In the article Palanisamy outlines best practices to help security professionals prepare for a quantum future, including:

- Get full visibility, automation, and control of certificates.
- Establish flexibility for the transition to post-quantum algorithms
- Implement strong certificate and key management practices
- Monitor compliance

Read more at the RSA Conference Library (<https://www.rsaconference.com/library/blog/six-steps-for-mitigating-quantums-impact-on-digital-certificates>).

Open Networking Foundation Projects Transferred to LF

The Open Networking Project (ONF) has announced that its open source networking projects will become independent projects under the Linux Foundation (LF), and the ONF will be dissolved.

“Linux Foundation is merging ONF’s marquee portfolio of broadband, mobile, edge, and cloud networking projects under the LF umbrella to help usher in the next phase of community growth,” says Jim Zemlin, LF executive director.

“The move creates independent, community-led governance for the three major project areas: Broadband, Aether, and P4, and sets the projects up for broader collaboration and adoption,” the announcement states (<https://opennetworking.org/news-and-events/press-releases/onf-merges-market-leading-portfolio-of-open-source-networking-projects-into-the-linux-foundation/>). Current ONF members will be welcomed to the new projects with transition support.

Supporting organizations for the new projects include Cornell University, Deutsche Telekom, Google, Intel, Netsia, Radisys, and Türk Telekom, the announcement says.

IBM Hybrid Cloud Mesh Now Generally Available

IBM’s multi-cloud networking solution, IBM Hybrid Cloud Mesh (<https://www.ibm.com/products/hybrid-cloud-mesh>), which was introduced earlier this year, is now generally available.

The product is “designed to allow organizations to establish simple, scalable secured application-centric connectivity,” the company says (<https://www.ibm.com/blog/announcement/app-centric-connectivity-a-new-paradigm-for-a-multicloud-world/>).

It is “engineered for both CloudOps and DevOps teams to seamlessly manage and scale network applications, including cloud-native ones running on Red Hat OpenShift.”

Key features include:

- Continuous infrastructure and application discovery
- Seamless connectivity
- Security
- Observability
- Traffic engineering capabilities

According to the announcement, key architecture components are the Mesh Manager and the Edge and Waypoint Gateways.

IBM Announces Quantum System Two

IBM recently announced IBM Quantum Heron, which is part of a series of quantum processors engineered to deliver “the highest performance metrics and lowest error rates of any IBM Quantum processor to date.”

Specifically, IBM Heron offers a “five-times improvement over the previous best records set by IBM Eagle,” the company says (<https://www.ibm.com/quantum/blog/quantum-roadmap-2033>).

Additionally, IBM has revealed the IBM Quantum System Two – a modular quantum computer – which has already begun operations with three IBM Heron processors and supporting control electronics, according to the announcement.

With these pieces in place, the company has extended its Quantum Development Roadmap to 2033, with expectations of increasing “the size of quantum circuits able to be run and help to realize the full potential of quantum computing at scale.”

“We are firmly within the era in which quantum computers are being used as a tool to explore new frontiers of science,” said Dario Gil, IBM SVP and Director of Research.

Red Hat to Remove Xorg from RHEL 10

Red Hat has announced the decision to remove Xorg server and other X servers (except Xwayland) from Red Hat Enterprise Linux 10.

In a blog post (<https://www.redhat.com/en/blog/rhel-10-plans-wayland-and-xorg-server>), Carlos Soriano Sanchez explains that the transition from the X Window System to the newer Wayland-based stack has been happening for the past 15 years. But, now, he says, “Wayland has been recognized as the de facto windowing and display infrastructure solution.”

During this transition period, Red Hat has been supporting both the Xorg and Wayland stacks, and this decision will allow the development community to focus their efforts solely on a modern stack and ecosystem, Sanchez says

European Commission Launches Large AI Grand Challenge

The European Commission (EC) has launched a new competition called the Large AI Grand Challenge (<https://aiboost-project.eu/large-ai-grand-challenge/>), which aims to foster innovation and excellence in large-scale AI models.

“Participants in the challenge are invited to submit a proposal for the development of a language foundation model, utilizing one of the EuroHPC JU targeted facilities (i.e., LUMI or Leonardo supercomputers). The model must be trained from scratch, possess a minimum of 30 billion parameters, and be trained following state-of-the-art optimal scaling laws for computing and training data size,” the website says.

The competition, launched in collaboration with EuroHPC Joint Undertaking (<https://eurohpc-ju.europa.eu/>), will run from November 16 to January 16, 2024.

Linux Foundation to Form High Performance Software Foundation

The Linux Foundation (LF) has announced its intent to form the High Performance Software Foundation (HPSF).

“Through a series of technical projects, HPSF aims to build, promote, and advance a portable software stack for high performance computing (HPC),” the announcement says.

Initial projects within the foundation include:

- Spack – A flexible HPC package manager.
- Kokkos – A performance-portable programming model for writing modern C++ applications in a hardware-agnostic way.
- Apptainer – A container system and image format specifically designed for secure high-performance computing.
- WarpX – A performance-portable Particle-in-Cell code with advanced algorithms that won the 2022 Gordon Bell Prize.

The formation of HPSF is expected to be completed in May 2024.

A flexible, customizable, personal dashboard

Command and Control

Create your own command center in small infrastructures or test environments with flexible dashboards that control and monitor relevant applications and services. By Holger Reibold

Many standardized interfaces

can be set up for applications on the local network. One of the most popular candidates is Webmin, whose strengths lie in managing legacy company servers. As a tool, though, Webmin is too complex and too powerful for many small environments. Administrators looking for a lightweight dashboard system to manage conveniently the links and entry points to various services will find Dashy [1] an interesting alternative.

Dashy is jam packed with useful functions for creating individual dashboards. It also lets you integrate status checks and use dynamic widgets and user-defined layouts. Dashy is open source, and the developers offer support on GitHub.

Installation

Of the various ways to get Dashy up and running, the easiest way has to

be the Docker-based installation. A search in the Docker repository,

```
docker search dashy
```

reveals the package you need. In the search results you will find a `lissy93/dashy` entry. Download this container to your system and launch Dashy:

```
docker pull lissy93/dashy
docker run -p 8080:80 lissy93/dashy
```

The environment performs various tests and outputs a success message. The local Dashy installation can be accessed on `http://localhost:8080`. Docker is not actually necessary for deployment; instead, you can install Dashy on any standard Linux system. Besides Git, you also need Node.js and Yarn:

```
git clone https://github.com/Lissy93/dashy.git &&
cd dashy
```

```
yarn
yarn build
start
```

To design dashboards, you need icons that represent the different areas and links. Dashy comes with a set of standard icons, but if you want to use the correct icons for popular devices (e.g., a Fritz!Box or a specific environment), enter

```
cd ./public/item-icons
git clone https://github.com/walkxcode/
    dashboard-icons.git
```

to download an icon set.

Basic Configuration

Now that you have Dashy running, you will want to customize the environment by editing the YAML-based `/public/conf.yml` configuration file. The file has three root attributes:

- `pageInfo` is where you store the dashboard metadata, such as the title, the description, the navigation bar links, and the footer text.
- `appConfig` is where you define the dashboard settings, such as themes, authentication, language, and customizations.
- `sections` is an array of sections, each of which in turn comprises an array of elements, called items.

The developers provide a complete list of all available configuration options [2], which is very helpful for customizing Dashy to suit your needs. You can manage authentication, set up user-defined themes, or enable reporting, for example.

The good news is that you do not have to make these adjustments at the console level but can access the configuration settings by clicking on the wrench icon in the Config section. Select *Update Configuration | Edit Config* and change the settings in the configuration editor to suit your requirements. To modify the title and description of your Dashy installation, open the `pageInfo` section and edit the `title` and `description` options by clicking on the values and adjusting them appropriately.

To expand the basic configuration, click to the left of the section name and run the *Insert* command. For your first steps, it makes sense to output notifications if unexpected errors and

crashes occur. To do this, use the `enableErrorReporting` option, which is disabled by default. In the configuration editor, add an entry of type `Auto` and assign it an option name and a value of `true`. The editor has an integrated syntax checker

that alerts you to possible typing errors or misconfigurations. Press *Save Changes* to apply the changes. Simple changes, such as changes to the title, take effect immediately, whereas for others, you have to reload the environment with `yarn build`.

Further cross-system functions are available on the Dashy splash page. Various design templates are available in the Layout selection menu top right. The design configurator lets you fine-tune the template details and adjust the layout and item size. For all of these tasks, you always need to enable the *Open Settings Menu* at top right.

Composing Dashboards

During the initial configuration, Dashy comes up with a *Getting Started* section (Figure 1), which gives you a first impression of the dashboard design approach. You need to create sections and bundle access to specific services and information in them. In a practical application, you might want, say, Status Information, Productivity, and Network sections. Other potential subdivisions could be Applications, External Services, and Devices. Further options can be found in the Dashy Showcase [3].

If you would like to add a section of your own design to the existing

dashboard environment, click on the pencil icon in the Config area to enter edit mode. Dashy now positions a placeholder to let you create the section, which you click to open the section properties. In the *Add New Section* dialog, define the key properties, such as the name, the icon, the default sort order, and the number of rows and columns. When you select the icons, you can use the file names in the `icons` folder. Click *Save* to save the section configuration and populate it with some initial links by clicking on the plus sign.

The procedure for creating a link in a section is similar. Follow the *Add New Item* link and specify the properties, such as the name, description, and icon, in the *Edit Item* dialog (Figure 2). In the *Service URL* input field, specify the target URL – the IP address or hostname – that is linked to the item entry. In the *Opening Method* input field, you define whether access is by a new browser tab or a new window.

In the *More Fields* area, you can include further information in the configuration, such as a status check and item ID. Use *Hot Key* to assign a numeric key between 0 and 9 to systems that you want to access particularly frequently. After saving, you will find the initial entries in the section you just created.

Removing items and sections is just as simple: In edit mode, click on

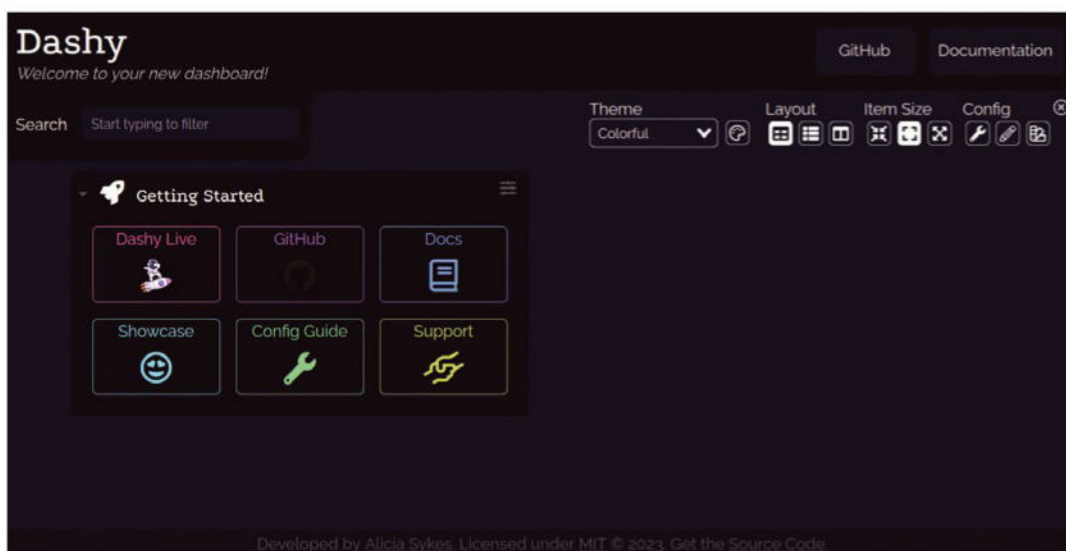


Figure 1: Once Dashy is installed, the *Getting Started* section quickly creates your first dashboards.

the slider icon of a group and run the `Remove` command. While in edit mode, Dashy records this fact in the footer. Press *Save Locally* to save the dashboard configuration and then access the integrated systems. In the footer, you will find further backup options, such as an export function in Config Saving Options. *Export Config* tells Dashy to display the source code; alternatively, you can export the configuration file and import it on a third-party system.

Listing 1: Example auth Element

```
appConfig:
  auth:
    users:
      - user: <Holger>
        hash: <Hash for password of User Holger>
        type: admin
      - user: <Klaus>
        hash: <Hash for password of User Klaus>
        type: normal
```

Adapting Dashy

Once you have gained some initial experience with the typical procedure for using Dashy-based dashboard design, you can customize the environment further. In edit mode, you can access further configuration options by clicking on *Edit App Config*, which has the settings for the `appConfig` section that you edit in the visual user interface editor. The editor basically offers maximum usability, but at the price of reliability. Alternatively, you can work directly in the YAML configuration dialog. According to the developers, a REST API is currently being developed that will support configuration adjustments from the command line, scripts, and third-party applications. In the application configuration, use *Default Opening Method* to define the default method for opening the item settings – the default is a new browser tab. Checking the *Enable Status Checks*

box ensures that the online or offline status of each service is displayed in the dashboard in the form of status information. It is well worth enabling this option because you can then tell at a glance whether all the services you need are available.

Besides the language, which Dashy takes from your browser configuration, you can also specify the background as the *Background Image*, along with the *Default Layout* and *Default Icon Size*. If you work with different services in parallel, you will want to enable the *Enable Multi-Tasking* option, which ensures that open applications remain open in the background. However, enabling this option comes at the expense of performance.

Registration and Authorizations

In addition to what are primarily visual customizations, Dashy has a login page and front-end authentication. To enable this feature, you need to add users to the `auth` section below `appConfig` in the configuration file. Access protection is not automatically enabled for a new installation. In the `auth` element, you need to create a user array, assigning each user a name, a hash, and a user type (`admin` or `normal`). The hash is a SHA-256 hash of the password. [Listing 1](#) shows an example configuration. The easiest way to generate the hash is with an online tool such as a SHA generator [\[4\]](#). After enabling authentication, users are redirected to the login page.

Dashy supports a guest mode that gives all users read-only access to the secure dashboard without having to log in. To set this up, set the `appConfig.auth.enableGuestAccess` option to `true`. The environment also supports the implementation of granular access authorizations to make specific sections or elements visible to or usable for certain users only. You have three options from which to choose:

- `hideForUsers` defines the sections and elements that are visible to all users except those in this list.

Figure 2: Items – in this case a Synology NAS – appear on a status page as links.

- `showForUsers` defines items hidden from all users except those listed.
- `hideForGuests` defines items visible for logged-in users, but not for guests.

Users who are not of the `admin` type cannot write any changes to the local storage medium. In the `appConfig` section, use `preventWriteToDisk` and `preventLocalSave` to restrict further authorizations. To disable all UI configuration functions, including *View Config*, set the `disableConfiguration` option to `true`. Alternatively, you can disable the user interface config functions for all non-admin users by changing `disableConfigurationForNonAdmin` to `true`.

Backing Up the Configuration

Cloud Backup & Restore is another practical function found as a tab

under the wrench icon. It lets you save the current Dashy configuration in the cloud and restore it if required. Before the data is transferred, it is encrypted with the `CloudBackup.js` script by the `AES crypto.js` method.

The procedure is otherwise simple: Assign a name to the backup, press *Update Backup*, and your data moves to the cloud of your choice. You will want keep the ID generated by this process safe so that you can use it, together with your password, for the *Restore a Backup* function, if needed. In this way, you can also easily transfer a configuration to a third-party system.

Conclusions

Dashy is a small but powerful tool that makes creating your own

dashboards child's play, and the extensive customization options, whether you use the integrated tools or edit the configuration file directly, do not affect Dashy's flexibility. ■

Info

- [1] Dashy: [\[https://dashy.to\]](https://dashy.to)
- [2] Configuration options: [\[https://github.com/Lissy93/dashy/blob/master/docs/configuring.md\]](https://github.com/Lissy93/dashy/blob/master/docs/configuring.md)
- [3] Dashy Showcase: [\[https://github.com/Lissy93/dashy/blob/master/docs/showcase.md\]](https://github.com/Lissy93/dashy/blob/master/docs/showcase.md)
- [4] SHA generator: [\[https://www.liavaag.org/English/SHA-Generator/\]](https://www.liavaag.org/English/SHA-Generator/)

The Author

Holger Reibold, computer scientist, has worked as an IT journalist since 1995. His main interests are open source tools and security topics.

IT Highlights at a Glance



Too busy to wade through press releases and chatty tech news sites? Let us deliver the most relevant news, technical articles, and tool tips – straight to your Inbox.

Linux Update • ADMIN Update • ADMIN HPC

Keep your finger on the pulse of the IT industry.

ADMIN and HPC: bit.ly/HPC-ADMIN-Update

Linux Update: bit.ly/Linux-Update



Versatile network and system monitoring

Digital Watchdog

The LibreNMS open source monitoring environment, unlike its predecessor Observium, comes through the back door at no cost, with auto-discovery, alerting, the ability to scale even in very large environments with many devices, and flexible dashboards and widgets for special views. By Thomas Joos

LibreNMS [1] promises flexible network and system monitoring and combines various functions from tools such as Nagios and Cacti. Another advantage of the software is that, unlike its predecessor Observium, it has no annual license fees for additional features or services such as quick updates, rule-based grouping, and scalability across multiple servers.

Monitoring with LibreNMS is easier now thanks to the auto-discovery function, and the web-based interface provides a wide range of customization options for visualizing information. The auto-update mechanism also ensures that the monitoring environment is always up to date, but first you need to get the software set up.

Integrating Initial Devices

One major advantage of LibreNMS is that the developers make it really easy for administrators to carry out an initial evaluation of the environment. They provide the installation

packages for Ubuntu 20.04/22.4, CentOS 8, and Debian 11, along with Docker, virtual machine (VM) images, and an online demo [2].

If you opt for a local installation, you can access LibreNMS on `http://localhost: 8080`. You need to define the access credentials during the configuration process. If you use the VMs, the credentials are already predefined (username *librenms*, password *D32fwefwef*).

LibreNMS is configured by default to update the environment automatically. If you want to run a manual update, you can use the `./daily.sh` command as the *librenms* user. In principle, you can disable the update mechanism in the global settings under *Updates* in the web graphical user interface (GUI). However, the developers advise against doing so because the `daily.sh` script not only installs the latest system components, but also handles other tasks such as cleaning up the database. To integrate new infrastructure components into the monitoring environment, you can add them manually with the

web GUI and command-line interface (CLI) or use the auto-discovery mechanism (Figure 1). To set up your initial devices in the GUI, navigate to *Devices* | *Add Device*. LibreNMS requires the hostname or IP address as well as various Simple Network Management Protocol (SNMP)-specific details, such as the version and port.

Alternatively, you can access the CLI over SSH. To create a new host, in the directory of your LibreNMS installation run the command:

```
./lnms device:add 2
[--v1] [--v2c] 2
[-c <yourSNMPcommunity>] <hostname>
```

For example, if you want to use SNMP v2c to monitor a host named *myhost.server.com* in the *My_Company* SNMP community, the command would be:

```
./lnms device:add --v2c 2
-c My_Company myhost.server.com
```

LibreNMS also supports ping-only devices, when only the availability

and response time are relevant. You can set these up in the web GUI by disabling the use of SNMP. LibreNMS only visualizes the Internet Control Message Protocol (ICMP) response graph for these devices.

Grouping Devices

The task of managing different device types can be simplified with the grouping function. LibreNMS supports static and dynamic groups: For dynamic grouping, use the Rule Editor (Figure 2), which relies on information from the MySQL database. For example, if you use the `dc<X>.<device type>.server.com` format for the hostname, you need to select the `devices.hostname` entry in the selection menu. In the dynamic configuration, regular expressions such as *equal*, *not equal*, and the like are also used to select devices. The procedure for static grouping is simpler. You simply need to enter the hostnames and press *Save* to save the group configuration. The advantage of this kind of grouping is that you can use alert mapping to assign individual rules for sending alerts to the various groups. Once you have created your initial devices, you can discover their status and recorded metrics under *Devices* | *All Devices*. Various actions are available in the tabular overview; for example, you can view the host-specific details or open a Telnet connection. In daily use, the selection menus for narrowing down the scope of the

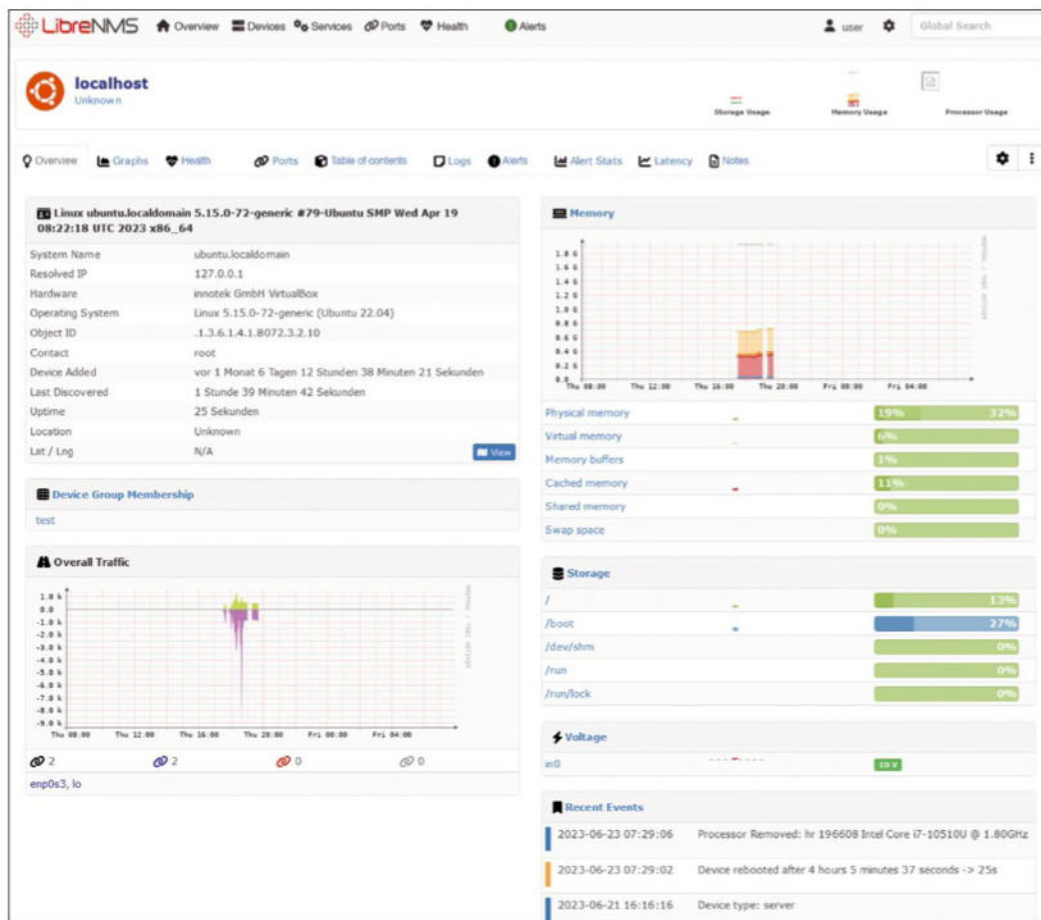


Figure 1: LibreNMS records a wide range of information about monitored devices and offers various visualization and interaction options.

The screenshot shows the 'Alert Rule' configuration window in LibreNMS. The 'Main' tab is active, displaying the following settings:

- Rule name:** Service up/down
- Import from:** A dropdown menu.
- Conditions:** A list of conditions including 'services.service_status' (not equal to 0) and 'macros.device_up' (equal to Yes).
- Severity:** Critical
- Max alerts:** -1
- Delay:** 5m
- Interval:** 5m
- Mute alerts:** OFF
- Invert rule match:** OFF
- Recovery alerts:** ON
- Match devices, groups and locations list:** Devices, Groups or Location
- All devices except in list:** OFF
- Transports:** Transport/Group Name
- Procedure URL:** A text input field.
- Save Rule:** A green button at the bottom.

Figure 2: The Rule Editor offers a massive range of functions and configurations.

table and the search function prove to be very useful.

Further important functions are available in the web-based interface. Although the strengths of LibreNMS lie in hardware monitoring, you can also keep an eye on any kind of service. To do this, select *Services | Add Service* and specify the device and the check type in the matching configuration dialog. The handling of service checks is simplified by the template function, which lets you check entire device groups. You can access the status of the RAM, CPUs, and storage media from the *Health* menu in the web GUI.

Listing 1: Define SNMP

```
// v1 or v2c
$config['snmp']['community'][] = "user-defined_community";
$config['snmp']['community'][] = "another_community";
Use the following changes for SNMP v3:
// v3
$config['snmp']['v3'][0]['authlevel'] = 'authPriv';
$config['snmp']['v3'][0]['authname'] = '<my_username>';
$config['snmp']['v3'][0]['authpass'] = '<my_password>';
$config['snmp']['v3'][0]['authalgo'] = 'SHA';
$config['snmp']['v3'][0]['cryptopass'] = 'crypto';
$config['snmp']['v3'][0]['cryptoalgo'] = 'AES';
```

Configuring Your Dashboard

Once you have designated the initial hosts for monitoring, you will obviously want to know how they are doing. LibreNMS has extensive dashboard functions for this purpose. You have the option of creating various overviews for a range of tasks and populating them individually with a selection of widgets ([Table 1](#)). All this activity takes place in the Overview menu, which you can use to generate new views and populate the existing views with the modules you need. Selecting *Overview | Show Dashboard Editor* switches to edit mode. To extend the functionality of the current dashboard, click on the pencil icon and then on the green plus sign to create a new dashboard. You need to assign authorizations to any newly created view; the three options are:

- *Private* allows the dashboard to be displayed and edited only by the user creating it.
- *Shared Read* allows third parties to view the dashboard information but not make any changes to the configuration.

- *Shared* allows all users to customize the view.

To set a dashboard configuration as the default, open the user settings (the person icon beside the cogwheel | *My Settings*) and select your preferred configuration in the Preferences section with the *Dashboard* drop-down menu. You can use the widget menu to put together the various modules that meet your information requirements.

Auto Discovery

To get to know LibreNMS and its specifics, it certainly makes sense to create hosts manually, but this approach is not particularly effective in larger environments and is prone to error. You can make your task easier with the auto-discovery function. LibreNMS supports various detection methods, applying them every six hours by default.

The first step is to make adjustments to the `config.php` configuration file. In particular, you can use SNMP v1, v2c, or v3. To define the SNMP details for versions 1 and 2, you need to extend the configuration file as in [Listing 1](#).

To avoid integrating systems into your monitoring setup indiscriminately, you need to define the subnets:

```
1nms config:set nets.+ '192.168.0.0/24'
1nms config:set nets.+ '172.2.4.0/22'
```

You can also specifically exclude devices you want the auto-discovery mechanism to ignore and install an agent there. An example of an exclusion configuration is:

```
1nms config:set 2
autodiscovery.nets-exclude.+ 2
'192.168.0.1/32'
```

By default, LibreNMS does not use IP addresses for device detection but searches for reverse DNS names. If you want to use IP-based detection, you need to enable the matching detection mechanism:

```
$config['discovery_by_ip'] = true;
```

Table 1: Current Widgets

Alerts	Displays all warning messages.
Alert History	Lists the historical warnings.
Alert History Stats	Statistics of historical warnings.
Availability Map	Displays all devices with colored tiles and lets you list all services and ignored/disabled devices.
Component Status	Shows all components and their statuses.
Device Summary Horizontal/Vertical	Total number of devices.
Device Types	Shows all events on devices.
Eventlog	Outputs the event logfile.
Globe Map	Visualizes a globe map with locations.
Graph	Generates diagrams of devices.
Graylog	Displays all syslog entries from Graylog.
External Images	Supports external images on the dashboard.
Notes	Can be used for HTML tags and embedded links to external websites. Also acts as a digital notepad.
Server Stats	Visualizes the CPU, memory, and storage utilization. Only devices of the <i>Server</i> type are listed.
Syslog	Displays all syslog entries.
Top Devices	Lists the top devices by data volume, uptime/response time, poll duration, processor/CPU utilization, or memory utilization.
Top Errors	Lists the most common error messages.
Top Interfaces	Lists the interfaces in relation to the traffic load.
World Map	Displays the locations of your devices.

LibreNMS uses various methods to detect network devices automatically – specifically, ARP, XDP, OSPF, BGP, and SNMP scans. They are all enabled by default in the configuration file.

Advanced Monitoring

The SNMP-based checks provide you with an initial impression of the status of a network environment, but what about business-critical servers and applications? After all, you want to make sure that the web server on which the company applications run is working properly. LibreNMS offers three solutions: You can open a direct connection to the application, extend the functionality of the `snmpd` daemon, or use an agent.

The auto-discovery function simplifies monitoring, but if you have a specific information requirement or the pertinent client information cannot be read externally, the use of agents is a potential remedy. The good news is you can integrate applications into the monitoring process even after creating the matching host.

Extended SNMP monitoring is enabled by default on all Unix-based platforms, which means, for example, that you can easily monitor the database and mail and web servers running on a Debian system. Note that the extended check is carried out as the `snmpd` user, which may be an account with standard privileges. In this case, you will need to use `sudo`. In the device settings, which you can access by clicking on the gear icon of the device entry, you can enable the applications to be monitored with the module settings.

The LibreNMS agent for Linux [3] collects data from remote systems and uses the Checkmk software. The matching check script is included in the agent package; a Windows version is now also available [4].

Alerting

Strictly speaking, monitoring hardware and software is only one area of responsibility because a targeted

approach to outputting alerts for critical conditions is almost as important as the monitoring. The Rule Editor is available for configuring alerts and can be accessed from the *Alerts | Alert Rules* item. In theory, LibreNMS supports highly complex rules that are based on mathematical calculations or MySQL queries. In principle, however, your rules must comprise at least three elements: an entity, a condition, and a value. Logical operators are used as possible conditions.

To create an initial rule, follow the *Create new alert rule* link in the rule manager. Alternatively, you can use *Create rule from collection* to use an existing rule configuration as the basis for a new rule. Assign a name and a severity level in the Severity dropdown. The Max alerts box lets you define the maximum number of alerts used for an event. A value of `-1` stands for an unlimited number of messages.

Listing 2: Override SQL Query

```
SELECT *,AVG(processors.processor_usage) as cpu_avg FROM devices, processors WHERE (devices.device_id = ? AND devices.device_id=processors.device_id) AND (devices.status = 1 && (devices.disabled = 0 && devices.ignore = 0)) = 1 HAVING AVG(processors.processor_usage) > 30
```

You can configure a user-defined SQL query on the *Advanced* tab by first enabling the *Override SQL* option and entering the SQL query in the query field. If the average CPU load exceeds 30 percent, the query will look like [Listing 2](#).

The configuration for failed devices is particularly simple. The code is `devices.status != 1`. If you want to output a warning in case of high CPU load per core (not the total load for all cores), then use:

```
macros.device_up = 1 AND ?
processors.processor_usage >= 90
```

Compared with many other products, LibreNMS also offers useful flexibility when it comes to choosing notification options. The SysContact stored in the SNMP configuration and the respective LibreNMS user of an alarm configuration are notified by default quite simply by selecting *Alerts |*

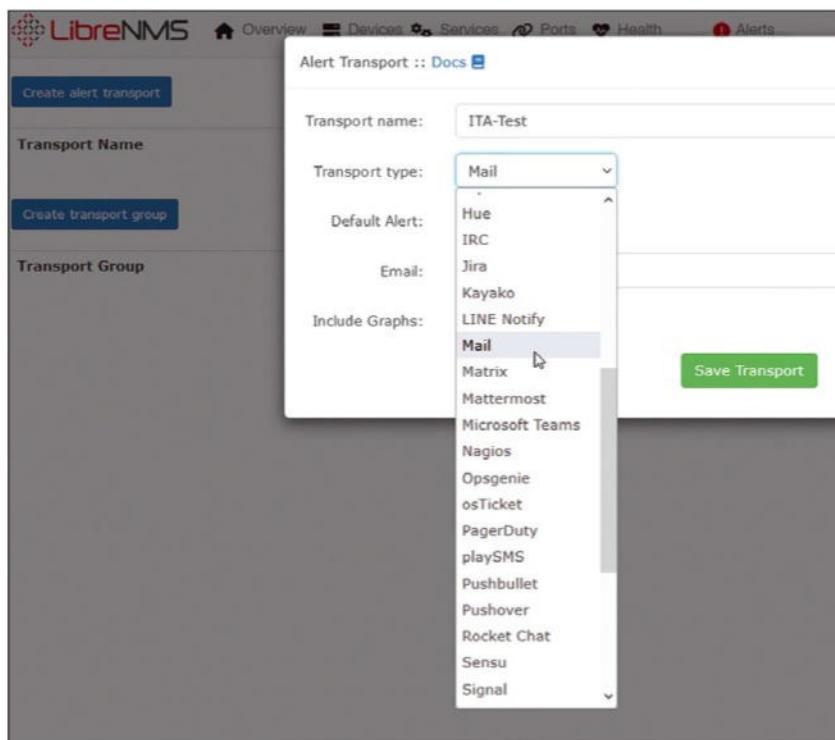


Figure 3: LibreNMS supports numerous ways of sounding the alarm in the event of critical incidents.

Alert Transports and using *Create alert transport* to create a notification configuration. More than 40 transport types are available in the Transport type dropdown. The *Mail* option (Figure 3) notifies the stored email account; alternatively, notifications can also be sent by *Signal* or as short messages.

For the message dispatch feature to work, you need to store its services in the LibreNMS configuration file, but if you just want to send email, you can also do so in the web GUI by mousing over the cogwheel icon and choosing *Global Settings*. On the *Alerting* tab under Email Options, select *sendmail* or *SMTP*; if you choose the SMTP option, enter the IP address and access data of the SMTP server.

Performance Tuning

PHP-based environments are not exactly considered to be the most powerful or fastest. Although these requirements are less important for smaller environments, they are all the more important for larger enterprises. The recommended approach is to leverage any optimization potential you can find. More specifically, the LibreNMS developers recommend using the *rrdcached* daemon, which fields updates for existing round-robin database (RRD)

files and accumulates them; once sufficient data has been received or a defined time has elapsed, it then writes the updates to the RRD file, significantly reducing load. To benefit from this ability to reduce the server load, you first need to install the *rrd-cached* package,

```
apt -q install rrdcached
```

and edit the LibreNMS configuration:

```
lnms config:set rrdtool_version '1.5.5'
```

Next, enable the tool on https://librenms_system/poller/rrdtool and use http://librenms_system/poller/ to access further server query customization options that can help reduce the server and network load.

Because LibreNMS collects all data in a MySQL database, the developers recommend running the MySQL Tuner [5] tuning script on a daily basis. Another tip for MySQL is to configure *my.cnf* in the `[mysqld]` group as follows:

```
innodb_flush_log_at_trx_commit = 0
```

Instead of a value of 0, you can also use 2 for a significant gain in I/O performance – but at the risk of losing up to one second of MySQL data if the MySQL server crashes.

Conclusions

LibreNMS combines impressive functionality and flexibility and is easy to learn. In terms of functionality, the environment leaves little to be desired. Only the alerting configuration is a little tricky, because users have to struggle with the database data and it is not always immediately clear which database element represents which device function. Even so, this issue is not likely to trouble you after a brief learning curve. ■

Info

- [1] LibreNMS: [\[https://www.librenms.org\]](https://www.librenms.org)
 - [2] LibreNMS demo environment: [\[https://demo.librenms.org/login\]](https://demo.librenms.org/login)
 - [3] LibreNMS agent for Linux: [\[https://github.com/librenms/librenms-agent\]](https://github.com/librenms/librenms-agent)
 - [4] LibreNMS agent for Windows: [\[https://github.com/Checkmk/checkmk/tree/v1.2.6b5/agents/windows\]](https://github.com/Checkmk/checkmk/tree/v1.2.6b5/agents/windows)
 - [5] MySQL tuning script: [\[https://raw.githubusercontent.com/major/MySQLTuner-perl/master/mysqltuner.pl\]](https://raw.githubusercontent.com/major/MySQLTuner-perl/master/mysqltuner.pl)
-

The Author

Thomas Joos is a freelance IT consultant and has been working in IT for more than 20 years. In addition, he writes hands-on books and papers on Windows and other Microsoft topics. Online you can meet him on [\[http://thomasjoos.spaces.live.com\]](http://thomasjoos.spaces.live.com). ■



Secure status and event monitoring of tier 0 systems

Keeping a Close Watch

We show you how monitoring your sensitive IT systems can be a more secure experience. By Evgenij Smirnov

From a security perspective, tier 0 systems such as domain controllers, privileged access workstations, or identity management systems provide direct access to digital resources, so more and more IT teams are making sure these systems have additional protection, which includes monitoring to make sure they are working properly. Regardless of whether you use a tiering model with a formal description (guidelines, firewall rules, and access groups; e.g., the Microsoft tiering model [1]) in your infrastructure or simply apply common sense and good account hygiene in your daily administration, every IT landscape has systems and objects that can be classified as tier 0 – the parts of the environment that enable complete control over the identity and security infrastructure, which makes them both particularly vulnerable and particularly worthy of protection.

In a Windows server landscape, these elements are usually the Active

Directory (AD) domain controllers, enterprise certification authorities, and sometimes systems that are heavily integrated into the AD, such as Exchange servers. As hybrid IT has progressed, new typical roles such as the Entra ID Connect server (formerly Azure AD Connect) have been added, and they clearly belong in tier 0. The administration workstations, or privileged access workstations, used to manage tier 0 systems must also be considered tier 0.

If errors occur, it is the monitoring systems' task to notify administrators by email, SMS, or other channels. In many organizations, the monitoring systems are even set up to initiate remedial action automatically in the event of certain malfunctions, ranging from a simple forced restart of a service or the entire server to complex workflows that expand the disks virtual machines (VMs), move the VMs themselves to a different host or cluster, or trigger database reorganizations.

In the case of highly privileged tier 0 systems, however, this process creates an area of tension that has barely been taken into consideration for years and is only now becoming the focus of those responsible for IT security as the threat situation continues to worsen. If you want to monitor tier 0, the increasingly powerful monitoring systems themselves become tier 0 and theoretically also need to be operated by separate administrative identities and protected in line with the rules for tier 0. This perspective in turn opens up new rifts in company-wide monitoring setups, potentially leading to the kind of operational disruptions that monitoring is intended to avoid. This article is not about security monitoring of tier 0 systems, but about classic status and event monitoring.

With or Without an Agent

Virtually no other architectural issue has prompted so much debate in the monitoring community over the

last 20 years as the use of dedicated agents. In fact, modern operating systems offer numerous remoting protocols, such as Remote Procedure Call/Distributed Component Object Model (RPC/DCOM), Windows Management Instrumentation (WMI), Windows Remote Management (WinRM), or Simple Network Management Protocol (SNMP), that let the monitoring system query the current status of the hardware and software over the local network, read out event logs, and even trigger actions, if required – whether to obtain additional information about the system status or react to a status that has been identified as faulty. This approach is the one taken by PRTG Network Monitor [2] by Paessler, for example.

Proponents of agentless monitoring argue that no additional active content needs to be installed and executed on the servers to be monitored because everything you need is already provided by the operating system. If you assume that a monitoring agent is subject to less stringent quality control than the operating system itself, you can claim greater operational stability. From a security perspective, though, this approach comes at a very high price:

- Network traffic on the remoting protocols needs to be allowed in the inbound direction. Admins sometimes fail to restrict these traffic relationships to the IP addresses of the monitoring systems.
- All remoting protocols require authentication to access the required data. SNMP is the only protocol that uses its own authentication and is independent of the monitored system. The use of the operating system's protocols, on the other hand, means that highly privileged service accounts need to be stored on the monitoring system. Very often, the same access credentials are used for a large number of systems, which in turn means that the passwords of these accounts are not changed often enough.

An agent-based monitoring architecture does not require you to store highly privileged access credentials

during operation, which gives it an apparent advantage at first glance. Ideally, agents establish a connection to the monitoring system; in other words, the network communication originates from the monitored system, and no separate inbound connections are needed. Unfortunately, however, the old “comfort versus security” dichotomy often means that the security situation is far from perfect. Many monitoring systems have a push mechanism for distributing agents to the new computers to be monitored. For this strategy to work, the target systems need to accept incoming Server Message Block (SMB) and RPC connections (Windows) and SSH with stored authentication (Unix/Linux). Access credentials that are sufficiently privileged to install agents therefore need to be stored on the monitoring system.

Monitoring agents also increasingly feature automatic update mechanisms, which means that anyone who has administrative rights for the monitoring system can use an unattended process to install arbitrary executable code automatically on the monitored devices. Monitoring agents often can run scripts in a variety of languages as probes. As a rule, these scripts do not have to be explicitly registered but simply stored in a special folder to be executed by an agent. Again, these scripts are often distributed by automated update mechanisms. For example, almost all monitoring products based on Nagios and that use either NSClient++ or their own agents, such as Checkmk [3], have this function, which makes it possible to inject arbitrary script code through the monitoring agent.

Although the agents usually run in the system context on the respective servers, some applications, such as SQL, SharePoint, or Exchange, need more to retrieve application-specific status information. In other words, the access credentials required for application-specific access not only need to be stored on the monitoring system but must also be transferred to the agent. Systems such as Microsoft's System Center Operations

Manager (SCOM) [4] make extensive use of this design.

Drawing Clear Boundaries

Whether with or without agents, as soon as your monitoring system is entrusted with status monitoring of tier 0 systems, it is very likely to become a part of tier 0 itself. It will either store credentials authorized to log on to tier 0 systems or have the ability to distribute executable code to these systems – or both. According to the fundamental concepts of security tiering, you will therefore need a separate monitoring system specifically for tier 0. This in turn means more setup and maintenance overhead, additional resource consumption, possibly additional licensing costs, and, above all, a break in the dependency chains of the service definitions. Ultimately, the status of the AD (tier 0) cannot be taken into account in the service dependency tree of a tier 1 service if this status is not available in tier 1 monitoring.

Almost everything that applies to the classification of monitoring systems in the security tiers also applies to data backup, virus protection, and platform provisioning (virtualization, storage, network management), particularly when it comes to configuration management such as software distribution and automation systems. Consistent separation of these components, depending on the level of protection, makes holistic mapping of IT services across tier boundaries both technically more difficult and organizationally more necessary than ever. Some organizations have successfully implemented the approach of complete tier separation with a separate server platform, data backup, monitoring, and even malware protection. This approach is particularly useful in large companies where separate teams manage the tier 0 systems anyway. If a malfunction detected during monitoring does affect the shared infrastructure components, such as the physical network, diagnostics and troubleshooting are coordinated by a shared process on the ticket system. For most IT organizations, however,

this kind a separation is not practicable. In the following sections, I take a look at the available alternatives.

Bare Essentials

The monitoring system's features are often not fully utilized. In many organizations, administrators prefer to use the ticket system as their monitoring front end. They deliberately avoid monitoring the current status of the systems and services and focus on event-centric monitoring by automatically opening a service request when an error occurs by email or via an application programming interface (API) call. This approach requires a very good understanding of what is normal in the particular IT environment and a reliable process for documenting and maintaining threshold values. If your monitoring system works in this way, achieving the required separation requires just a little overhead:

setting up a separate monitoring instance for tier 0 systems and then hardening the firewall rules. You will want to locate the new monitoring instance fully in tier 0 and either re-register any agents or ideally even roll them out again and restrict administration of the tier 0 instance to authorized workstations and admin accounts. If privileged access credentials were stored in the general monitoring system before this changeover, you will also need to separate them:

- If the access, Run As, or service accounts used previously are also used on tier 1 systems, it can be extremely time-consuming to create new accounts for tier 1. It makes more sense to grant the required tier 1 permissions explicitly to the existing accounts, revoke the tier 0 permissions, and create new accounts for tier 0.
- If the access accounts were already exclusively assigned to tier 0

before the changeover, you can continue to use them, but you will want to change the passwords to be on the safe side before you store these access credentials on the tier 0 monitoring system. Regardless of the password change, you also need to delete the stored tier 0 credentials from the monitoring system that is now exclusively dedicated to tier 1.

In this construct, all cross-tier communication originates from the monitoring system (tier 0) and is directed to the tier 1 systems for dispatching email, making API calls, or both, to open tickets automatically. This method ensures that the tier 0 systems are protected from the influence of less privileged tiers.

Distributed Monitoring

In both agent-based and agentless monitoring, the risk for the highly

Discover the past and invest in a new year of IT solutions at Linux New Media's online store.

Want to subscribe?

Searching for that back issue you really wish you'd picked up at the newsstand?

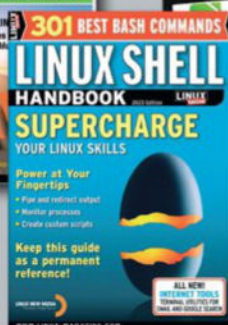
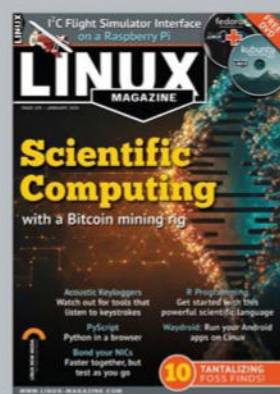
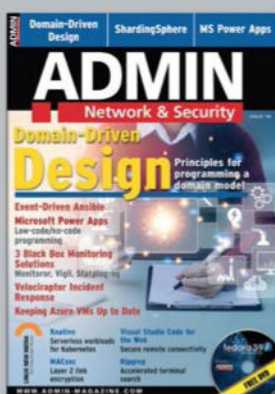
shop.linuxnewmedia.com



shop.linuxnewmedia.com

DIGITAL & PRINT SUBSCRIPTIONS

SPECIAL EDITIONS



privileged systems and accounts comes from data collection (access credentials stored on the monitoring system) or from system administration (distribution of agents, updates, scripts). Displaying the current system status in the form of dashboards and alerts by email, SMS, or API call, on the other hand, is harmless. Some monitoring systems can perform monitoring and alerting on separate systems. In most cases, this separation is part of a general architecture for distributed monitoring. Some systems such as Checkmk [5] or Zabbix [6] support this type of architecture by default. However, the emergence of distributed monitoring strategies is not attributable to security considerations, but to the need to consolidate and centrally process status data and configurations from geographically distributed locations.

The Zabbix proxy seeks to offer “convenience and standardization” and makes the central server the configuration hub for all connected systems. Checkmk’s Livestatus technology takes the same approach (Figure 1). However, it offers an alternative, in the form of Livedump and CMD dump, that lets you centralize data storage and dashboard views

across the tiers, while leaving the configuration and administration to the respective Checkmk servers in the individual tiers.

In Microsoft SCOM, setting up tier-specific distributed monitoring is somewhat more complex and requires you to create at least three management groups (MGs) [7] (Connected MG, Tier 1, and Tier 0), each with a separate SQL database, at least one dedicated management server, and sophisticated role-based access control (RBAC) in between to enforce the administrative separation between the groups. Other monitoring products offer similar approaches. When you purchase, install, and configure a distributed monitoring system, you need to make absolutely sure that the goal of tier separation is achieved and that you do not open a backdoor to your tier 0 systems.

Heterogeneous Monitoring

The various distributed monitoring technologies require the same product to be used in the individual zones or tiers. However, you might be able to consolidate the monitoring data in tier 1 with the use of a different

monitoring product for tier 0 and including tier 0 monitoring in tier 1 monitoring such that the status data of the individual tier 0 systems are displayed like operating parameters. The prerequisite is that you can enable access to the tier 0 monitoring data without giving up control over this system’s configuration. Potential solutions include:

- REST API access over a separate port and without the possibility of influencing the configuration.
- Regular data export from tier 0 monitoring and transfer to a tier 1 system, where the data is analyzed and processed to create sensors. Agents such as NSClient can, for example, evaluate the content of text files or execute scripts.
- An active monitoring agent on the tier 0 monitoring system that sends NSCA messages or SNMP traps to the tier 1 system on its own initiative and does not transmit its own status, but that of the monitored tier 0 machines. Tier 1 monitoring must have a corresponding acceptor, which is becoming increasingly rare, especially for NSCA.

You can also link two instances of the same monitoring product using

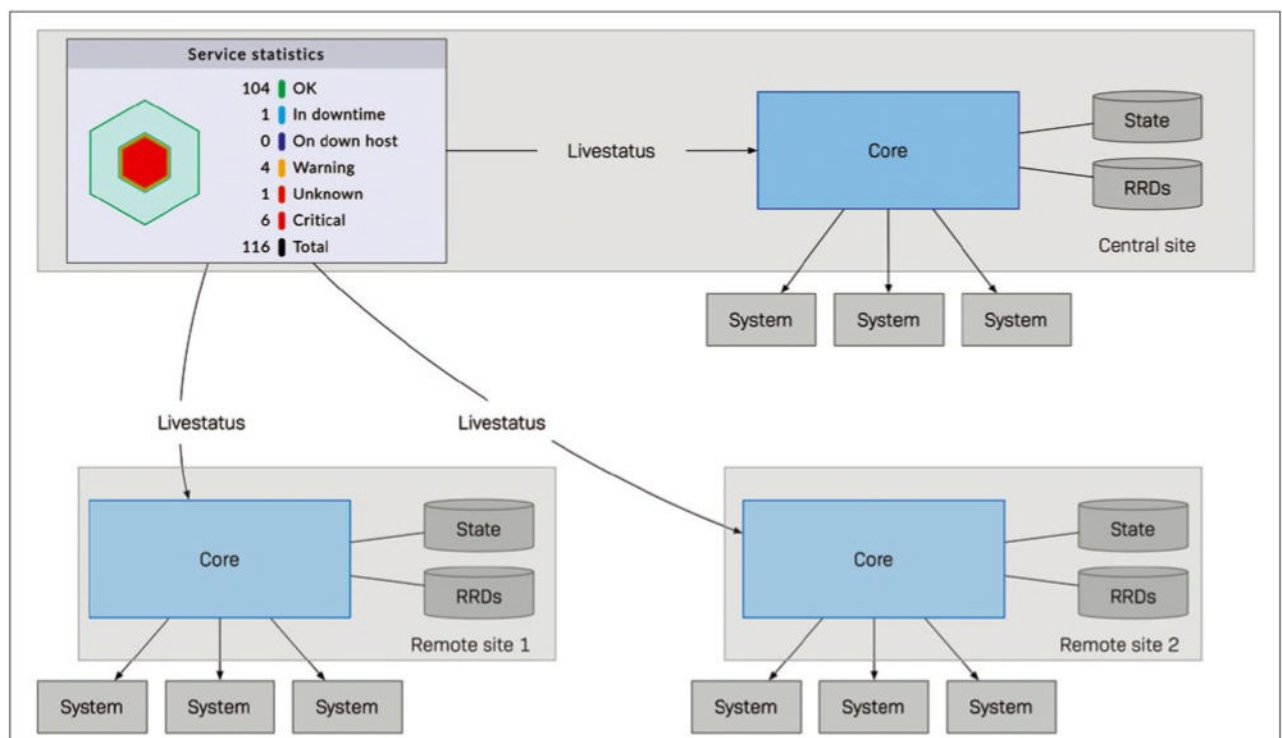


Figure 1: Depending on the technology, distributed monitoring enables tier separation. © Checkmk GmbH [5]

methods described above if the software you use is flexible enough to do so.

Moving Monitoring into Tier 0

In small IT teams, where infrastructure and application management is essentially the responsibility of the same group of people, it can be quite practical to declare the entire monitoring setup to be a tier 0 service. Onboarding a new system usually involves tier 0 activities anyway, such as creating groups, service accounts, policies, and firewall rules. If IT optimizes its processes, the administrator responsible – who is already logged in to a privileged access workstation at this point – can also add the new system to the monitoring setup.

Monitoring products that have the technical ability to separate the visualization (dashboards) from the controls are particularly useful. This means that you can hang a display terminal for the monitoring dashboard on the wall in IT operations with a clear conscience, and without having to soften the boundaries of your network segmentation. Checkmk lets you use Livedump for this purpose. This approach has some limitations, although it is quite simple to implement. Automatic inclusion of new systems in the monitoring setup, for example, should be treated with caution. If your monitoring system is not smart enough to use different credentials for onboarding, depending on the target system, highly privileged credentials could be used far too often.

In classic tiering according to the legacy Microsoft model, attempts to access a tier 1 system with a tier 0 account would be rejected. If you can ensure that the tiering guidelines take effect early on in the deployment process and the monitoring system uses tier 1 credentials if the tier 0 login fails, automatic onboarding can be practicable in principle. From a security point of view, there is nevertheless some risk, because tier 0 credentials could be compromised.

Risky Updates

The increasingly popular automatic agent updates should also be treated with caution in tier 0. Ever since the Solarigate attack on software manufacturer SolarWinds [8], IT security admins have been aware of the danger that software updates can pose. If the attackers manage to infiltrate a manufacturer's development environment, they can add their own routines to the delivered software. These routines can then be passed unnoticed with an update to customers' privileged systems and compromise them.

In tier 0, you will want to get rid of automatic agent updates and explicitly verify all new versions before deploying them to the highly privileged systems. If you use the same product for tier 0 and tier 1 monitoring, tier 1 monitoring can act as a staging environment for testing the tier 0 agents, allowing you to check both operational characteristics, such as stability and resource consumption, and the security of the new agent versions, for example, by running an additional anti-malware scan against the agent installers or delaying the installation in tier 0 and first monitoring the network traffic of the agents in tier 1 for suspicious behavior.

Ultimately, of course, you need to compare the up- and downsides of this approach carefully. If a manufacturer patches highly critical vulnerabilities that have already been exploited in the field, the risk of a successful attack by these vulnerabilities may be greater than that of a supply chain attack. However, because the agents run on systems that are already highly isolated, vulnerabilities of this kind should be virtually impossible to attack from the outside, which should give you sufficient time to evaluate the updates.

Conclusions

A compromise between consistent visibility and strict separation is not easy to achieve in monitoring but is

technically possible with a variety of monitoring systems. Knowing how your IT and application teams use the functions of the existing monitoring systems is of paramount importance. If you are planning to purchase a new monitoring system or change your monitoring system, you will definitely want to revisit the agent vs. agentless discussion and give security considerations top priority. ■

Info

- [1] Microsoft tiering model:
[<https://learn.microsoft.com/en-us/security/privileged-access-workstations/privileged-access-access-model>]
- [2] PRTG Network Monitor:
[<https://www.paessler.com/prtg/prtg-network-monitor>]
- [3] Checkmk local checks:
[<https://docs.checkmk.com/latest/en/localchecks.html>]
- [4] SCOM RunAS profiles:
[<https://learn.microsoft.com/en-us/system-center/scom/plan-security-runas-accounts-profiles>]
- [5] Checkmk distributed monitoring:
[https://docs.checkmk.com/latest/en/distributed_monitoring.html]
- [6] Zabbix distributed monitoring with proxies:
[https://www.zabbix.com/documentation/current/en/manual/distributed_monitoring]
- [7] SCOM design:
[<https://learn.microsoft.com/en-us/system-center/scom/plan-mgmt-group-design?view=sc-om-2022#design-considerations>]
- [8] Analyzing Solarigate:
[<https://www.microsoft.com/en-us/security/blog/2020/12/18/analyzing-solorigate-the-compromised-dll-file-that-started-a-sophisticated-cyberattack-and-how-microsoft-defender-helps-protect/>]

Author

Evgenij Smirnov has been working with computers since the age of 5 and delivering IT solutions for almost 30 years. His Active Directory and Exchange background naturally led to PowerShell, of which he's been an avid user and proponent since its first release. Evgenij is an active community lead at home in Berlin, a leading contributor to German online communities, and an experienced user group and conference speaker. He is a Microsoft Cloud and Datacenter Management MVP since 2020.



Real-time monitoring with Graphite

Single Stack

The open source Graphite tool offers real-time monitoring for IT environments, with comprehensive and fast data collection from virtually any system. By Holger Reibold

Graphite [1], which started life in 2006 as a side project of a monitoring tool for an airline and travel portal, has evolved into a powerful system today that, according to developer Chris Davis, helps Etsy, Booking.com, GitHub, Salesforce, Reddit, and many other companies monitor their business processes. The software has been under the Apache 2.0 open source license since 2008. The ecosystem has grown considerably in the meantime and offers a complete set of collection agents and voice connections for all typical application scenarios.

Inner Workings

At its core, Graphite handles two tasks: It stores numerical time series data and renders the data in the form of graphs. Feeding raw data into the system is particularly easy. To bundle raw data in all its diversity, Graphite relies on the interaction of three software components: (1) Carbon queries the various time series data; (2) Whisper, the database library, handles the task of storing the data; and (3)

Graphite-Web provides the user interface and API for displaying charts and dashboards.

The Carbon service is responsible for feeding the various items of raw data to the stack, which in turn is handed over for long-term storage to the Whisper databases. The admin interacts with the Graphite web user interface (UI) or application programming interface (API), which in turn queries Carbon and Whisper for the data. The key benefit of the tool is that it bundles and consolidates a wide mix of source data, making the data available to third-party applications. Graphite supports various output styles and formats (e.g., CSV, XML, JSON), which sets the stage for embedding custom charts into external websites or dashboards. Graphite has flexible options for consuming raw data. The monitoring environment supports three main methods: plaintext, the Python-specific pickle data format, and Advanced Message Queuing Protocol (AMQP). The information sent to Graphite is managed by Carbon and Carbon

Relay, and the Graphite-Web interface reads the data either from the cache or directly from a storage medium. Which transmission method to use in each individual case depends primarily on the applications reading the data or the scripts you deploy. Some applications have special tools or APIs that can help you transfer data to Carbon. The easiest way to learn about the specifics of the transfer process is to use the plaintext protocol. Carbon is backed up by a number of daemons that form the storage back end. A simple Graphite installation typically only uses one daemon (carbon-cache.py). For larger environments, the carbon-relay.py and carbon-aggregator.py daemons are usually added to distribute the load of metrics processing and handle custom aggregation tasks. Basically, all Carbon daemons expect time series data from third-party sources and can field the data by common transmission protocols. However, the daemons differ in the way they process the incoming data. Knowledge of these different processing

capabilities is essential when it comes to implementing sophisticated storage back ends.

Installation

Graphite is a complex Linux-based environment mainly programmed in Python. The environment uses the Cairo graphics library to render the graphs, which results in various dependencies that typical server installations do not usually cover. In the source-based installation, the `check-dependencies.py` script helps you handle the checks. The easiest way to install Graphite is in Docker,

```
docker run -d \
  --name graphite \
  --restart=always \
  -p 80:80 \
  -p 2003-2004:2003-2004 \
  -p 2023-2024:2023-2024 \
  -p 8125:8125/udp \
  -p 8126:8126 \
  graphiteapp/graphite-statsd
```

which means you could also use a Windows system to evaluate the setup. For a standard installation, make sure the following conditions are met, along with having Python version 2.7 or later in place:

- `cairoffi` Python module
- Django 1.11.19 or newer
- `django-tagging` 0.4.6
- `pytz` Python module
- `scandir` Linux function
- `fontconfig` library
- Web server gateway interface (WSGI) and a web server (Apache also requires the `mod_wsgi` module)

You will also need the Graphite web app, Carbon, and the Whisper database library, which is part of the Graphite project. Depending on whether you go for optional functions, further Python modules (e.g., `python-memcache`, `python-lpad`, and `python-rrdtool`) may be needed.

The use of Synthesize [2], an installation script for Graphite and the associated services on Linux, all go to prove that the installation is unlikely to pose a massive challenge. However, the use of Synthesize is limited

to Ubuntu 18.04 LTS. That said, the REsynthesize [3] fork is designed for CentOS 8.1 or higher.

Configuration

The Carbon module is the heart of the Graphite environment. It is controlled through various configuration files that reside in the `/opt/graphite/conf/` directory. Because the initial installation does not create a configuration file, you will need to create one manually. The easiest way to do so is to glean from the various sample files (`conf.example`) by copying them to the configuration directory and removing the `.example` suffix. Graphite has the configuration files shown in Table 1. For an initial installation, you will primarily need to edit `carbon.conf` and `storage-schemas.conf`. The main `carbon.conf` configuration file bundles the settings of the various daemons; the configuration itself is broken into sections:

- `[cache]` controls the carbon-cache daemon,
- `[relay]` controls the carbon-relay daemon, and
- `[aggregator]` controls the carbon-aggregator daemon.

The developers recommend paying special attention to the `[cache]` section when you use it for the first time. The `storage-schemas.conf` file is where you assign metric paths to the patterns and store the frequency and duration of data storage for the Whisper component. The patterns are regular expressions defined in three lines:

```
[garbage_collection]
pattern = garbageCollections$
retentions = 10s:14d
```

In the first line, you assign a label to the rule, followed by the regular expression, which you specify with `pattern=` and then the retention rate (`retentions=`).

In the example, the `[garbage_collection]` label is primarily for documentation purposes. Graphite logs this and the matching metrics in the `creates.log` file. This pattern is applied to all metrics that end with `garbageCollections`. Note that Graphite uses Python syntax for regular expressions. The `retentions` line states that each datapoint is equivalent to 10 seconds and that you want to keep the data from the last 14 days available.

In addition to the various configuration files, you can manage the use of metrics with white- and blacklists.

In the case of whitelist functionality, Graphite accepts only those metrics explicitly whitelisted; all blacklisted metrics are rejected. The advantages are obvious: You can explicitly filter out all metrics that are not relevant to your information needs. To enable this filter function, raise the `USE_WHITELIST` flag in the `carbon.conf` file. Graphite then searches the directory defined with the `GRAPHITE_CONF_DIR` option for the blacklist and whitelist configurations (`blacklist.conf` and `whitelist.conf`). If you have not created a whitelist configuration or it is empty, the software will let all metrics pass through.

Reading Metrics

The challenge when using Graphite is to read in the different data, but the monitoring environment is very flexible. You can use the three methods already mentioned: plaintext,

Table 1: Graphic CONF Files

File Name	Function
<code>carbon.conf</code>	The main configuration file used to define the settings for each Carbon daemon.
<code>storage-schemas.conf</code>	Determines the retention rates for storing metrics.
<code>storage-aggregation.conf</code>	Specifies how lower precision data is aggregated.
<code>relay-rules.conf</code>	Relay rules used to communicate metrics to specific back ends.
<code>aggregation-rules.conf</code>	Aggregation rules that bundle different metrics.
<code>rewrite-rules.conf</code>	Lets you rename metrics with regular expressions.

pickle, and AMQP. The acquired data is sent to and managed by two modules: *carbon* and *carbon-cache*. The method you use to feed Graphite with data depends on the environment and the data you want to process. Various tools and APIs are available. For test data, it is easiest to use the plaintext protocol; if you have large volumes of data, pickle is recommended; and AMQP is the best choice if Carbon is listening on a message bus.

The simplest approach is the plaintext protocol. The data must use the `<metric path> <metric value> <metric timestamp>` format. Carbon then takes care of translating the resulting line of text into a metric that the web interface and Whisper database understand. For testing purposes, I ran the Netcat (nc) program on Unix to generate a socket and send data to Carbon:

```
PORT=2003
SERVER=graphite.system
echo "local.random.diceroll 4 `date +%s`" >
| nc ${SERVER} ${PORT}
```

Basically, Graphite is used to collect numerical data and transmit the data to Carbon for analysis. Each data series has a unique identifier that is based on the metric designation and various tags. A naming system is essential. The second step is to configure data retention, answering various questions in the process: How often is the data generated? What kind of precision do you need? Over what time period do you want to acquire data? To create a naming scheme, modify the `/opt/graphite/conf/storage-schemas.conf` file. Graphite requires a message format, such as,

```
echo "test.bash.stats 42 `date +%s`" >
| nc localhost 2003
```

comprising the metric namespace, the value you want to assign to the metric, and the timestamp.

Integrating External Software

Graphite is designed to work with about a hundred different tools, for

which the developers have provided an online overview [4]. For example, the software collaborates with *collectd*, the well-known system data acquisition daemon. You can use the *collectd* plugin *write-graphite* to submit *collectd* metrics to Carbon. Alternatively, Graphite can extract the metrics from *collectd* RRD files if you add those files to `STORAGE_DIR/rrd`. For example, in practice, you can link `<host.name>/load/load.rrd` file to `rrd/collectd/<host_name>/load/load.rrd` to generate the `collectd.<host_name>.load.load.{short,mid,long}term graph`.

Visualizing Data

The Graphite dashboard is responsible for visualizing the data and lets users display various sources. Dashboard access is by `http://<my.graphite.host>/dashboard` or the composer. Unfortunately, Graphite's visualization capabilities are quite limited, but you can apply functions to the data or pool graphs from different hosts. As mentioned, the integrated visualization functions turn out to be



Figure 1: The Graphite dashboard only provides a simple tool for visualizing metric data, so employing Grafana makes more sense.

difficult to use in practice, which is what prompts most administrators to turn to more user-friendly tools that provide meaningful evaluations. Grafana [5] is very popular, not least because it has a native plugin to help integrate Graphite data sources (Figure 1). Integration is easy with a little help from the Grafana query editor (Figure 2).

Alerting

Monitoring critical systems and visualizing data is one thing, but you primarily need to be able to respond to critical events, for which typical alerting and notification functions are used. By definition, a monitoring system also outputs warnings when specific values occur, but Graphite does not have this capability. You will have to rely on third-party products if you need this feature. Again, the alerting configuration is particularly easy in combination with Grafana, which comes with the Alerting system used to create a ruleset. The ability to generate single- and multidimensional rules is useful. The Alerting system, in turn, comes with the query manager, which supports the integration of Graphite metrics

with very little configuration overhead. As an alternative, a tool like graphite-beacon [6] is a good choice. This simple alerting application for Graphite runs asynchronously and sends notifications on the basis of Graphite metrics. The advantage is that it has no dependencies other than the Tornado package and is easy to implement. To install, use pip or apt-get:

```
pip install graphite-beacon
```

```
apt-get update
apt-get install graphite-beacon
```

The configuration is handled by a config.json file located in the same directory as graphite-beacon. When editing the file, you first need to specify the URL of the Graphite system and then use regular expressions to determine when to trigger an alert. You also need to define an email handler to take care of mailing the alerts. In addition to Graphite alerts, you can use this tool to configure URL alerts for web-based environments.

Conclusions

The Graphite developers like to classify their application as a

monitoring tool – probably with the aim of addressing a larger target group. A second look, however, shows that it is primarily an aggregator that collects a wide variety of metrics. Graphite handles this task excellently – not least because of the comprehensive ecosystem that has grown up around the tool. However, Graphite’s full potential is only revealed in combination with more capable visualization tools. ■

Info

- [1] Graphite: [\[https://graphiteapp.org\]](https://graphiteapp.org)
- [2] Synthesize: [\[https://github.com/obfuscure/synthesize/\]](https://github.com/obfuscure/synthesize/)
- [3] RESynthesize: [\[https://github.com/deividgdtr/resynthesize\]](https://github.com/deividgdtr/resynthesize)
- [4] Tools that work with Graphite: [\[https://graphite.readthedocs.io/en/latest/tools.html\]](https://graphite.readthedocs.io/en/latest/tools.html)
- [5] Grafana: [\[https://grafana.com\]](https://grafana.com)
- [6] graphite-beacon: [\[https://github.com/klen/graphite-beacon\]](https://github.com/klen/graphite-beacon)

The Author

Holger Reibold, computer scientist, has worked as an IT journalist since 1995. His main interests are open source tools and security topics.

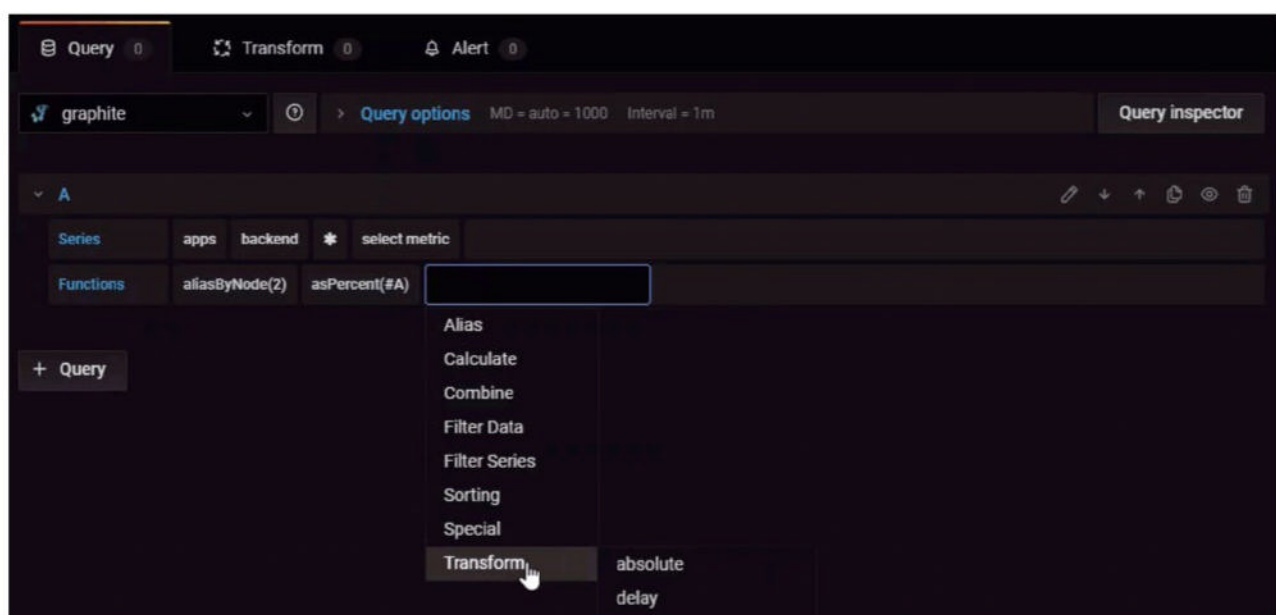


Figure 2: Grafana’s query editor supports the integration of Graphite data.



Exploring the AlmaLinux Build System

Package Packer

The AlmaLinux Build System lets you build, test, sign, and release packages from a single interface. By Joe Casad

When IBM announced that it was restricting access to Red Hat Enterprise (RHEL) source code and moving CentOS upstream, the distros that depended on RHEL and CentOS source code were sent scrambling. It is still a little unclear whether IBM's moves are legal and consistent with the GNU Public License (GPL), but the litigation to sort it out could take years, and in the meantime, the derivatives need a solution. One enterprise distribution that weathered the storm quite smoothly was AlmaLinux [1] (see the box entitled "Where Do They Get Their Code?"). If you ask the AlmaLinux developers, they will say that one reason for their success in navigating the transition to the post-RHEL era is the AlmaLinux Build System [2]. The AlmaLinux Build System evolved from an earlier system used by CloudLinux. (CloudLinux is a contributor to the AlmaLinux project.) The developers refer to their build system as "a project designed to automate processes of building distribution and packages, testing packages, signing packages,

and releasing them to public repositories." In other words, the goal is to assist with every phase of the package development process, relying on automation to reduce human error and minimize manpower requirements.

The AlmaLinux Build System is a free software project that is available on GitHub. Other Linux distributions are welcome to use the AlmaLinux Build System as a tool for building and managing packages. You can also point the build

system at other, third-party Git repositories, which makes it suitable for many in-house DevOps development settings.

How Does It Work?

The AlmaLinux Build System automates the process of building, supporting, and managing packages. The vision is for something that is more than a build tool, with support for testing, signing, and releasing software packages.

If the AlmaLinux project needed a build system to interact with source code originating from a Red Hat environment, you might be wondering

Where Do They Get Their Code?

AlmaLinux was envisioned as a free alternative to RHEL, which comes with a subscription fee and other corporate licensing arrangements. AlmaLinux and other RHEL derivatives used source code from Red Hat repositories as the basis for building an independent distro. It is important to note that Red Hat does not *own* the source code in the sense that the term *own* is used with proprietary software. Because Linux and most of the code included with it are open source and licensed under the GNU Public License (GPL), the code is available for others to use and modify.

IBM currently restricts access to some, but not all, RHEL source code. Some source code is available through the Red Hat Universal Base Image (UBI) [3]. AlmaLinux uses as much of the Red Hat UBI code as it can, but a majority of the code comes from the CentOS Stream project [4]. IBM did not eliminate all access to CentOS; they just moved it upstream, so the code does not include some of the final bug fixes and updates that go into the final version of RHEL. AlmaLinux uses some code from the CentOS Stream project and performs its own fixes and updates. They also pull code from other upstream sources when necessary.

Lead Image © stylephotographs, 123RF.com

why they didn't just use Koji [5], the freely available build tool associated with Red Hat's Fedora project. The answer given by the developers is that, although Koji is an effective tool, the AlmaLinux project had a much broader vision. For one thing, they wanted to integrate additional package formats (Koji is limited to RPMs). They also wanted to provide a complete, integrated pipeline to manage a package from the build phase, to testing, to signing the package, and finally to release. The AlmaLinux Build System includes controls that allow the user to specify where to release packages, and it is one of the first build systems to support modular-

ity. A module is a collection of packages that occur together, such as the packages in a single application or an operating system component. Support for modularity lets you treat the packages together, thus saving steps and streamlining the configuration. Like other build platforms, the AlmaLinux Build System is not a monolithic application but a combination of back-end tools behind a single, unified interface. Some of the tools incorporated into the AlmaLinux Build System include:

- Mock – a tool for building RPM packages
- Pulp – a content repository for organizing and distributing software packages
- NGINX – a web server that serves as an interface for managing access to the build system

- Terraform – an infrastructure-as-code tool used to build simulated environments for package testing
- PGP – an encryption utility that provides signing services for package verification
- Git – a source code repository system

Git isn't actually part of the build system itself, but it is an integral part of the ecosystem, providing source code for building packages and communicating with the build system through an API.

Figure 1 shows the complete system at a glance. Users interact through either a graphical user

interface or text-based commands. Support for command-line processing creates the possibility for scripting and other custom automation scenarios.

At the center of the system is the Build System Master Service. The Master Service receives commands from the user and sets the process in motion, creating, restarting, and deleting builds and communicating with the rest of the system via API calls. Responsibilities of the master service include requesting and receiving source code from the Git server and assigning tasks to the build nodes.

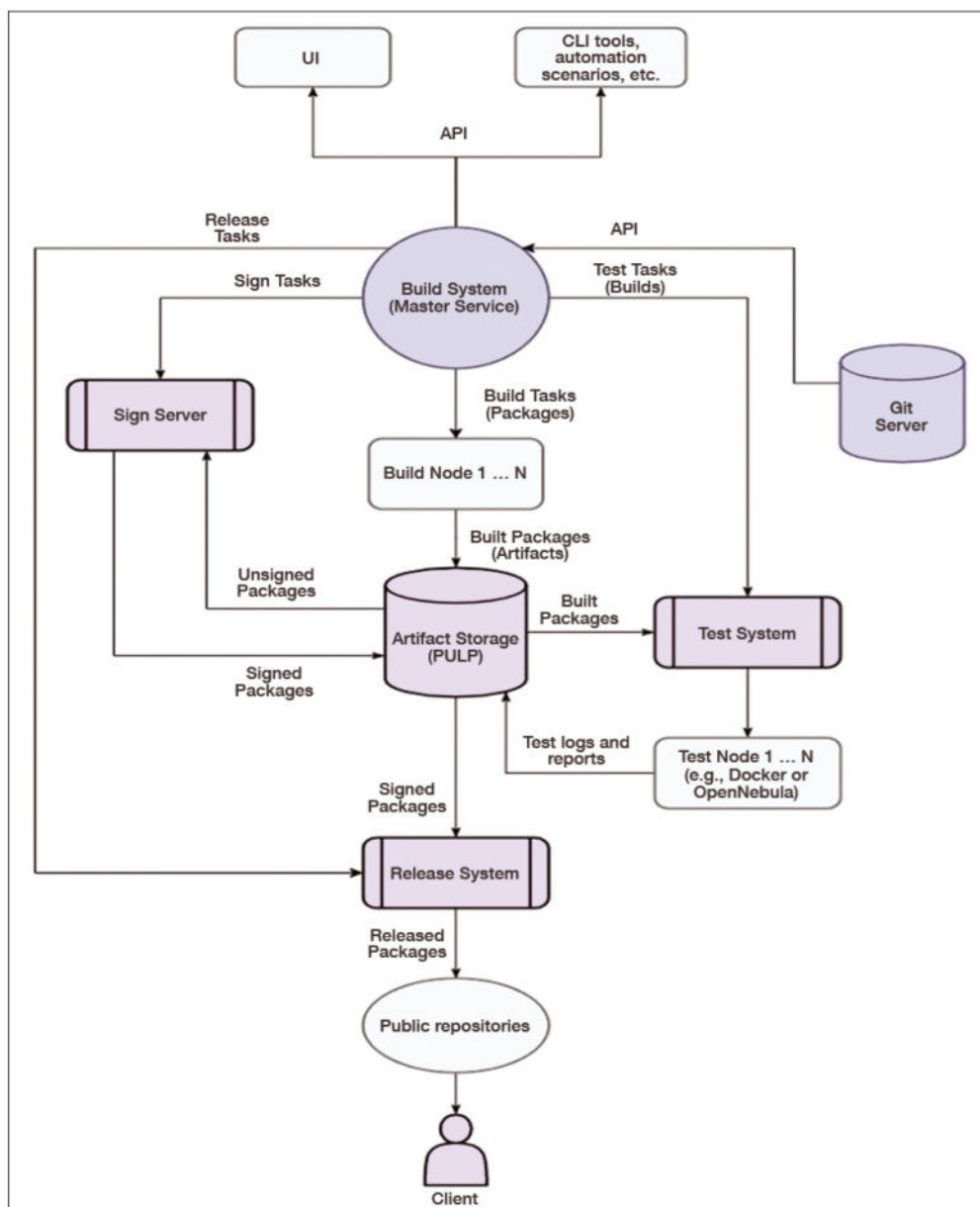


Figure 1: The AlmaLinux Build System at a glance.

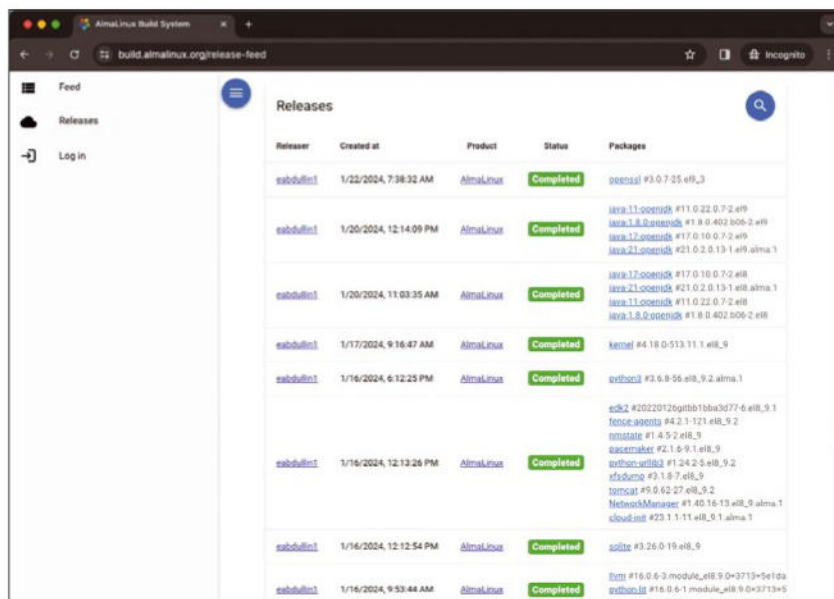


Figure 2: Your first view shows the builds configured for the system.

Another important component of the build system is Pulp [6], which provides artifact storage for newly-built packages and other products of the build process. According to AlmaLinux Community Manager Jack Aboutboul, “the master service is the brain, and Pulp is the heart” of the build system. As you can see in Figure 1, Pulp is essential to the later stages of the process, providing packages for signing and testing, and forwarding finished packages for release. Much of the power of the AlmaLinux Build System is in its ability to oversee the testing, signing, and release phases of the development process.

Getting the Code

The AlmaLinux Build System uses the Gitea software development service [71] to communicate with the Git server. Gitea is described as an all-in-one service for managing a Git environment, including “code review, team collaboration, package registry and CI/CD.” The AlmaLinux team has developed a gitea-listener tool for interfacing with Gitea and the Git repository. The AlmaLinux Build System also supports Fedora Community Repository Platform format (COPR), which makes it easy to add alternative repositories to the system.

First Look

When you log in to the AlmaLinux Build System, a view of configured

SBOMs

On May 12, 2021, the Biden administration released Presidential Executive Order 14028 “Improving the Nation’s Cybersecurity” [9]. One of the important features of that order is the stipulation that software packages for software used by the US government should include a bill of materials for all the code provided in the package. This Software Bill of Materials (SBOM) is described as a “list of ingredients” for the software package. The idea is that providing an accurate list of ingredients used for building the package will

Testing and Later Steps

The AlmaLinux Test System (ALTS) [8] included with the build

help investigators identify and trace security risks that might affect the package. If a component used in building the package turns up with a critical vulnerability, it will be easy to spot the problem and to know that the package needs an update.

AlmaLinux was the first Linux distribution to notarize and provide an SBOM for all source and components. The AlmaLinux developers created an SBOM generation utility and integrated it into the AlmaLinux Build System. You can find the `alma-sbom` utility on GitHub [10].

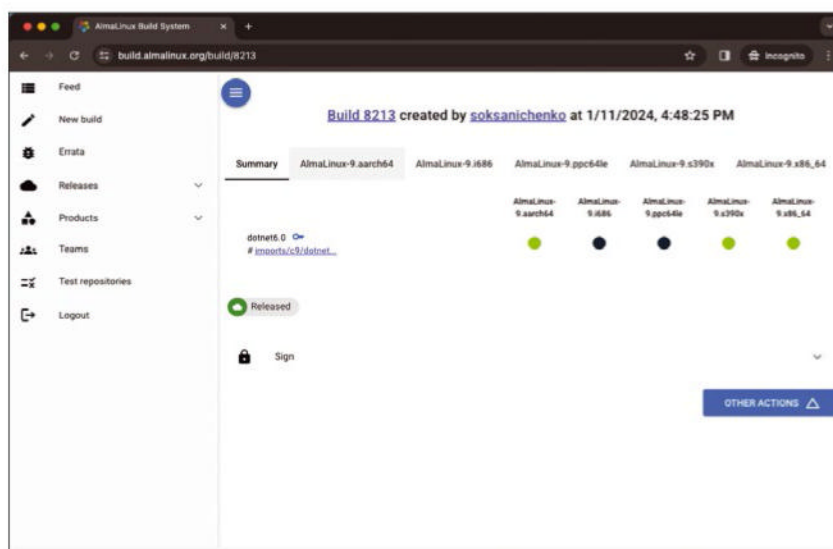


Figure 3: The AlmaLinux Build Systems supports several hardware architectures.

environment automates packet testing in realistic conditions. ALTS first launches a clean test environment (for instance, a Docker container) using Terraform to recreate a realistic setting that models actual production conditions. Once the environment is in place, ALTS attempts to install the package, and, if the installation is successful, begins a series of integrity checks predefined by the user.

Results of the tests are then forwarded to the Pulp artifacts store in the form of test logs and reports, and the results are then available to the user through the web interface. Approved packages are then signed and marked for release. The build system lets you define and select specific channels for the software release, and the verification system allows the receiver to trace the authenticity back to the original source code.

Conclusion

The stability and versatility of the AlmaLinux Build System has given the developers a head start on achieving the project's ambitious goals while avoiding much of the wheel spinning that often comes with putting a distribution together. AlmaLinux was recently chosen as a standard Linux distribution for Fermilab and the CERN European laboratory for particle physics. The AlmaLinux project was also the first enterprise Linux to offer a complete Software Bill of Materials (SBOM) for every package (see the box entitled "SBOMs").

The AlmaLinux team is busy right now using the AlmaLinux Build System to create, sign, test, and release the next version AlmaLinux, but the developers also want to sure make the system is available to other users and other projects. The user interface makes it easy to incorporate other source code repositories, and the testing, signing, and release components support customization for alternative projects and applications. An API-driven design with support

for scripting opens a range of possibilities for adapting the build system for other projects. ■

Info

- [1] AlmaLinux: [\[https://almalinux.org/\]](https://almalinux.org/)
- [2] AlmaLinux Build System: [\[https://github.com/AlmaLinux/build-system\]](https://github.com/AlmaLinux/build-system)
- [3] Red Hat Universal Base Image: [\[https://catalog.redhat.com/software/base-images\]](https://catalog.redhat.com/software/base-images)
- [4] CentOS Stream: [\[https://www.centos.org/centos-stream/\]](https://www.centos.org/centos-stream/)
- [5] Koji: [\[https://koji.build/\]](https://koji.build/)
- [6] Pulp Project: [\[https://pulpproject.org/\]](https://pulpproject.org/)
- [7] Gitea: [\[https://github.com/go-gitea/gitea\]](https://github.com/go-gitea/gitea)
- [8] AlmaLinux Test System:

[\[https://github.com/AlmaLinux/alts\]](https://github.com/AlmaLinux/alts)

- [9] Presidential Executive Order 14028: [\[https://www.whitehouse.gov/briefing-room/presidential-actions/2021/05/12/executive-order-on-improving-the-nations-cybersecurity/\]](https://www.whitehouse.gov/briefing-room/presidential-actions/2021/05/12/executive-order-on-improving-the-nations-cybersecurity/)
- [10] alma-sbom: [\[https://github.com/AlmaLinux/alma-sbom\]](https://github.com/AlmaLinux/alma-sbom)

Joe Casad

Joe Casad is the editor in chief of *Linux Magazine*.

This article was made possible by support from AlmaLinux OS Foundation through Linux New Media's Topic Subsidy Program (https://www.linuxnewmedia.com/Topic_Subsidy).

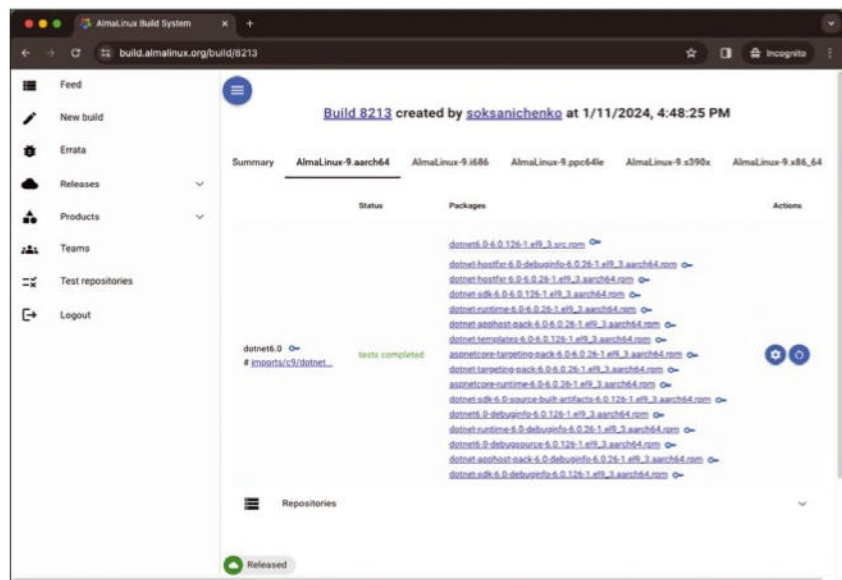


Figure 4: Viewing the artifacts associated with the build.

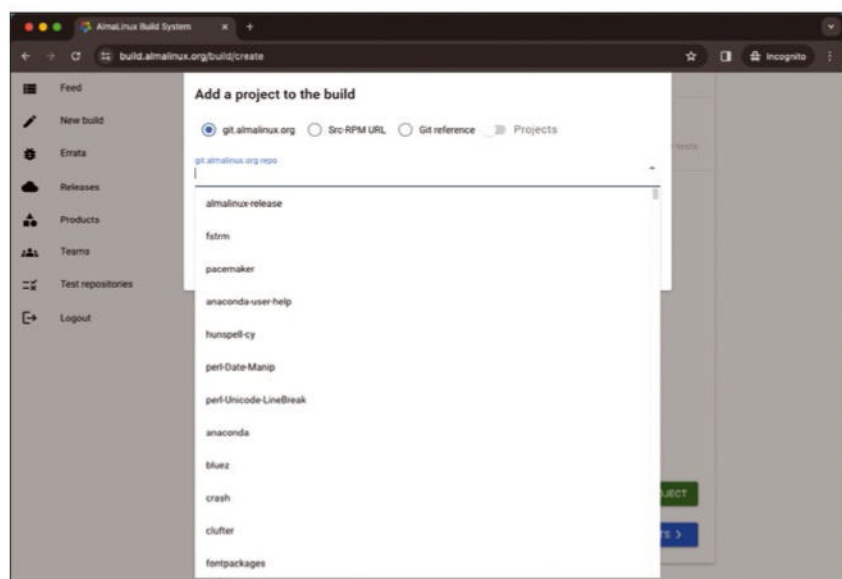


Figure 5: Adding a project to the build.

Identity and access management with Authelia

Bouncer



Add access controls to web applications that do not have their own user administration; however, this useful gatekeeper requires a reverse proxy. By Tim Schürmann

To protect a private party from hooligans, you could check the guests' invitations at the entrance yourself, or you could hire a bouncer. Authelia [1] acts as a bouncer for web applications to help you regulate access to services that do not offer their own access controls.

Thanks to Authelia, developers do not have to implement complex and time-consuming user management in their own web applications. Instead, they can deploy Authelia upstream of their own software with two-factor authentication and single sign-on (SSO) by default. In other words, you just need to log in to Authelia to access several authorized applications. Authelia requires that communication with the web applications be protected by a reverse proxy. The software then connects to this reverse

proxy (Figure 1) and checks all incoming requests, just as a bouncer would check invitations. When you access a web application in a browser, your request is first sent to the reverse proxy, which forwards it directly to Authelia for inspection. When a browser knocks on the door for the first time, it does not have an Authelia session cookie. In this case, the tool redirects you to its own login page, where you first need to verify your ID by providing a username and password.

Double Security

Authelia can retrieve the login data from its own user database (a simple YAML file) or consult an LDAP server. Administrators can specify the password structure in a policy and, for

example, stipulate that passwords need to contain at least one uppercase letter. To prevent attackers from simply brute-forcing the passwords, the number of login attempts can be limited. You can only log in again after a specified wait. Far more effective would be if the wait was automatically extended for each incorrect password, but that is beyond Authelia's capabilities. To set the bar even higher for attackers, Authelia relies on two-factor authentication. The second factor is either one-time passwords, such as those generated by the Google Authenticator, push messages to cell-phones, or hardware-based systems that comply with the FIDO2 WebAuthn standard (Yubikey USB sticks). Push messages require users to have the Duo Push app by Cisco on their smartphones.

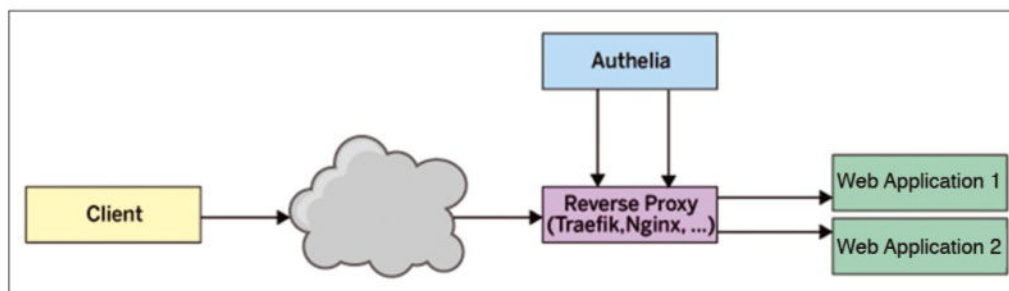


Figure 1: The reverse proxy submits every incoming request to Authelia to authorize access explicitly.

After successfully logging in, Authelia transfers a session cookie to the browser. The reverse proxy submits all following requests to Authelia again. As the browser now has a session cookie, Authelia gives the

reverse proxy the go-ahead to forward the request to the appropriate web application. By default, the session cookie then also applies to all subdomains of the domain controlled by the application. A flow chart in the Authelia documentation [2] demonstrates the process: On receiving the first request, Authelia redirects the browser to its own login page. Because of this approach, Authelia does not need to know how a web application works, nor can it sniff the transmitted data (in the form of payloads). The tool itself only processes the authentication information. In principle, any HTTP-based services can be protected, such as microservice REST and GraphQL interfaces. Authelia currently works with the proxies listed in Table 1. Apache and Microsoft IIS are left out in the cold. As a general rule, all parties involved need to use HTTPS to prevent the session cookie from being tapped. Authelia supports SSO on the basis of trusted headers, which both the proxy and the web applications must support. Authelia also acts as an OpenID Connect 1.0 provider, with authentication based on tokens. Although this function was still in beta when this issue went to press, it has already been received with great interest. The ownCloud Infinite Scale developers are planning to integrate Authelia into their groupware as an OpenID Connect provider [3].

Ruleset

Imagine you want a team to have access only to specific pages of a wiki. To do this, you would create a list of

access rules that Authelia compares against every incoming request. In other words, the software also acts as an access control system.

The rules are stored in YAML format (Listing 1). The example is also based on the online documentation [4]. It allows all users from the *hamburg* group to access the *wiki.example.com* domain; the path must comply with the regular expression specified in resources. In the example, all the pages accessible below *wiki.example.com/project/hamburg/* would be in the allowed scope. The requests must originate from the 192.168.2.0/24 subnet and are only allowed to use GET and POST methods.

Authelia is implemented in Go as a compact binary that can be used directly without installation and without dependencies. The official packages on GitHub include templates for a systemd unit and a sample configuration. The developers recommend using their official Docker container, which only requires around 30MB of RAM [5]. Authelia is also designed for operation in Kubernetes. You can start several containers in parallel there to configure the tool for scaling or high availability. The official Helm chart ensures a quick setup.

In production operation, the tool requires further software components. It stores distributed session cookies in RAM or a Redis database. Log information, settings, and other data generated at runtime are encrypted and stored in a PostgreSQL, MySQL, or SQLite database. To verify the identity of new users, Authelia sends an email, so administrators must provide an SMTP server.

Conclusions

Authelia is not suitable for all use cases because of its mode of operation. For

Listing 1: Access Rules

```
access_control:
  rules:
    - domain: wiki.example.com
      resources:
        - '^/project/hamburg/.*$'
      subject: 'group:hamburg'
      policy: two_factor
      methods:
        - GET
        - POST
      networks:
        - 192.168.1.0/24
```

example, the software cannot help you if you want microservices to authenticate automatically. Also, some desirable functions for enterprises are still missing, including multidomain protection, although this feature is already on the project agenda.

The simple structure and clear configuration mean that Authelia can be set up quickly. This identity and access management software is therefore suitable for protecting small and medium-sized enterprises or retroactively protecting a domain's web applications against unauthorized access. Go developers in particular can integrate Authelia into their application and save themselves the trouble of having to program their own user management feature. ■

Info

[1] Authelia: [\[https://www.authelia.com\]](https://www.authelia.com)

[2] Authelia architecture: [\[https://www.authelia.com/overview/prologue/architecture/\]](https://www.authelia.com/overview/prologue/architecture/)

[3] "Try to ship Authelia as the default IdP in the ocis binary" by Michael Barz, October 27, 2023: [\[https://central.owncloud.org/t/try-to-ship-authelia-as-the-default-idp-in-the-ocis-binary/45662\]](https://central.owncloud.org/t/try-to-ship-authelia-as-the-default-idp-in-the-ocis-binary/45662)

[4] Authelia access control: [\[https://www.authelia.com/overview/authorization/access-control/\]](https://www.authelia.com/overview/authorization/access-control/)

[5] Authelia via Docker: [\[https://www.authelia.com/integration/deployment/docker/\]](https://www.authelia.com/integration/deployment/docker/)

The Author

Tim Schürmann is a freelance computer scientist and author. Besides books, Tim has published various articles in magazines and on websites.

Table 1: Supported Proxies

Proxy	Standard	Kubernetes	XHR Redirect	Request Method
Caddy	Yes	Partially	Yes	Yes
Envoy	Yes	Yes	Partially	Yes
HAProxy	Yes	Partially	Partially	Yes
NGINX	Yes	Yes	No	Yes
NGINX Proxy	Yes	No	No	Yes
Skipper	Yes	No	Partially	Partially
Swag	Yes	No	No	Yes
Traefik 1.x	Yes	Partially	Yes	Yes
Traefik 2.x	Yes	Yes	Yes	Yes



Fast email server deployments with iRedMail

Email the Easy Way

Setting up and maintaining an email service in the data center doesn't have to be a nightmare. The iRedMail open source solution lets you deploy a full-featured email server on a number of platforms in a matter of minutes. By Rubén Llorente

Friends and workmates often tell me that email is obsolete as a means of communication. Everybody is using some mobile messaging app or another these days. Solutions such as Signal Messenger or WhatsApp let users send and receive documents and messages, and they are comparatively free from the spam and scam campaigns that plague the email ecosystem. Thus, I am told, email is irrelevant.

People couldn't be more wrong. Email is vital for many businesses because it allows them to deliver messages to both customers, employees, and associates by standard, open protocols that are not controlled by a single organization. Servers can be configured to email error reports to a system administrator with every incident, and email is still the most popular password recovery mechanism when somebody forgets the password to their favorite web forum. This battle-tested communication mechanism is not free of

shortcomings, however. Email accounts are bound to receive illegitimate messages containing malware or unsolicited advertisement (spam). Therefore, a modern email service must be equipped with smart filters capable of identifying legitimate mail (colloquially known as ham) and stopping the rest. Another complaint against email services is that they are built by joining many unrelated components that are not trivial to configure. A typical email service needs a web server for hosting both a management interface for the system administrator and webmail for regular users. A Simple Mail Transfer Protocol (SMTP) daemon is needed to deliver messages to users of different email services, whereas Post Office Protocol (POP) or Internet Message Access Protocol (IMAP) daemons let users check email with clients such as Thunderbird or Mutt. In the face of such complexities, many small organizations prefer to

have a third party host their email service, and they end up purchasing plans with email providers such as Google or Microsoft. Family businesses in particular have a tendency to use free email plans from big providers such as these.

Setting up your own email service, however, has been a solved problem for quite a long time, so you have no excuse for letting Google handle your email (or worse, having your employees use a Gmail account to communicate with your customers). Canned solutions that build a functional email server in a matter of minutes already exist. Previously, I discussed Citadel [1], and now I want to introduce iRedMail.

Enter iRedMail

iRedMail is a for-profit operation built on the freemium model as a free “open source, fully fledged, full-featured mail server” [2]. The website lists three iRedMail-related products.

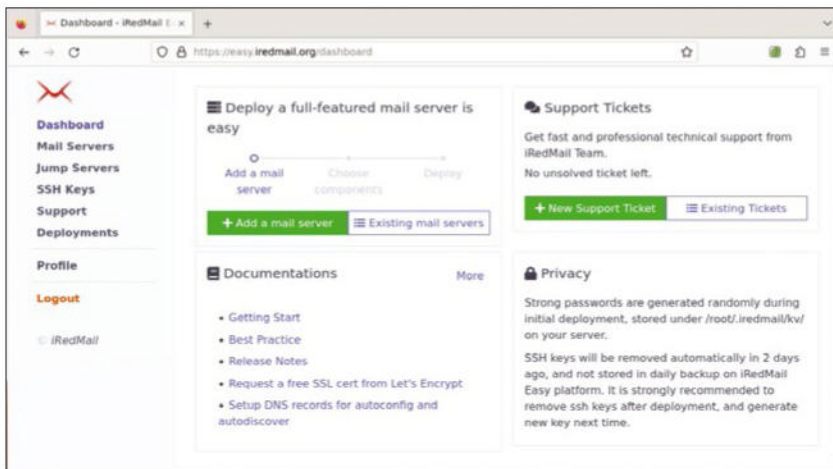


Figure 1: iRedMail Easy is a web-based deployment platform; however, you must perform a fresh install of a supported distribution on a machine you control and then create an account in the iRedMail system.

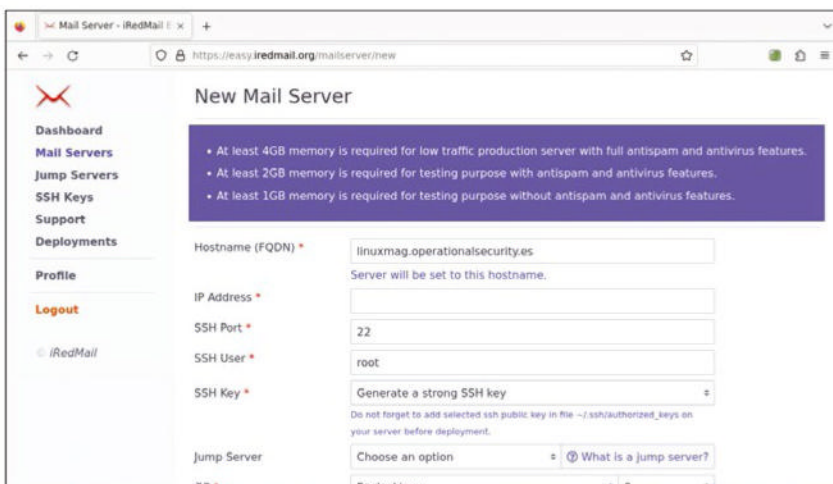


Figure 2: iRedMail Easy uses Ansible to provision your email server. You need to give the iRedMail company SSH access to your machine with superuser privileges, allowing them to set up the system for you.

The downloadable installer is the product I focus on in this article. The free-tier product, licensed as free open source software under the GPL3, is a set of scripts that turns a machine into an email server when run on top of any of the supported operating systems or Linux distributions. To do this, the scripts download and install all the components required by the email server from the distribution's repositories and then configure them for you.

iRedMail Easy (Figure 1) is a web-based deployment platform. To use it, you have to create an account [3] and give iRedMail the credentials to log on to your servers over SSH (Figure 2). In this way, you can

command their web deployer to turn your machines into email servers in a similar fashion as the downloadable installer.

The website does a poor job trying to convince you of the advantages of iRedMail Easy over the downloadable installer, but in practical terms, it looks like its only real benefit is getting commercial support and making it easier to submit

Table 1: Supported Platforms

Platform	Versions
CentOS Stream	8, 9
Rocky Linux	8, 9
Alma Linux	8, 9
Debian	11, 12
Ubuntu	20.04, 22.04
FreeBSD	13.x
OpenBSD	7.3

support tickets. It is worth noticing that support tickets that require SSH access are paid at a premium. iRedAdmin-Pro is the high-end version, and its source code is available to paying customers. It is clearly marketed at email administrators who intend to host the email services of multiple different organizations or customers. It includes features not available in the free version, such as the ability to assign different resource quotas to each hosted domain, to have different administration accounts per hosting domain, and to manage quarantined email with a flexible system.

Getting Started

To build your iRedMail server, you need a fresh install of a supported platform. The list in Table 1 was valid at the time of writing this article. Keep in mind that support is not equal for all platforms: For example, my production iRedMail servers all run on OpenBSD and, although all the core functionality works, certain

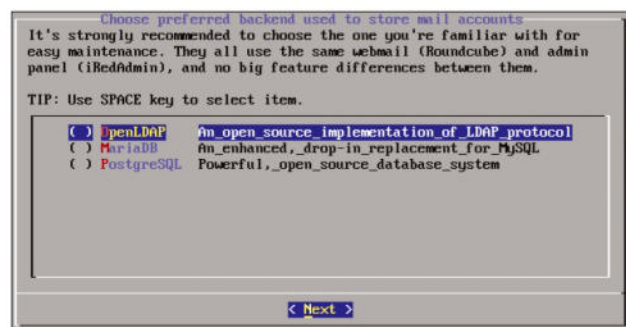


Figure 3: The steps to installing iRedMail on Rocky Linux 9 by the downloadable installer are quite intuitive, but moderate knowledge is required for making some choices, such as which back end to use to store user account information. OpenLDAP, MariaDB, and PostgreSQL are supported.

Table 2: iRedMail Components

Tool	Functionality
Postfix	SMTP server
Dovecot	IMAP server
Nginx	Web server (optional)
iRedAdmin	Web-based management interface (optional)
Roundcube	Webmail (optional)
SOGgo	Webmail and groupware (optional)
Fail2Ban	Brute force protection (optional)
Netdata	System monitor (optional)
Amavis	Content filtering
SpamAssassin	Spam filtering
ClamAV	Malware detection

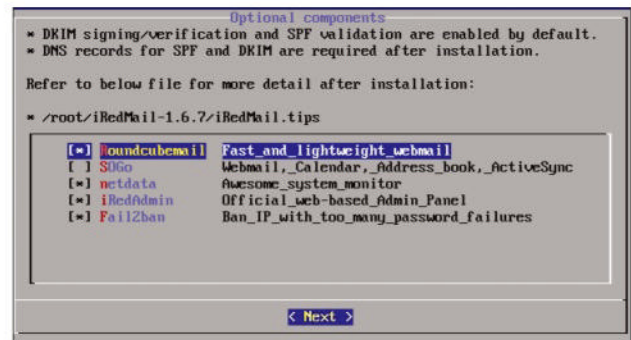


Figure 4: Some components are optional. Fail2Ban integration in OpenBSD is broken and does not work out of the box.

integrations need some fiddling. iRedMail does not necessarily support the most recent OpenBSD release, either.

I recommend the downloadable installer, but I will skip

the detailed instructions for installation because they are documented on the project's website [4]. The procedure will be a breeze for system administrators and power users, but some basic IT knowledge is required. For example, you are asked which back end to use to store user accounts (Figure 3) and which optional applications you want to install (Figure 4). A successful install gives you a server featuring all the components you need for a small system (Figure 5; Table 2).

Postmaster's Office

Unless you go out of your way to disable it, an iRedMail install will feature a management interface, reachable with a web browser at `https://<yourserver>/ired-admin`, where `<yourserver>` is your server's fully qualified domain name (FQDN) or IP address. From this control panel, you can perform most of the basic administrative tasks, such as creating new

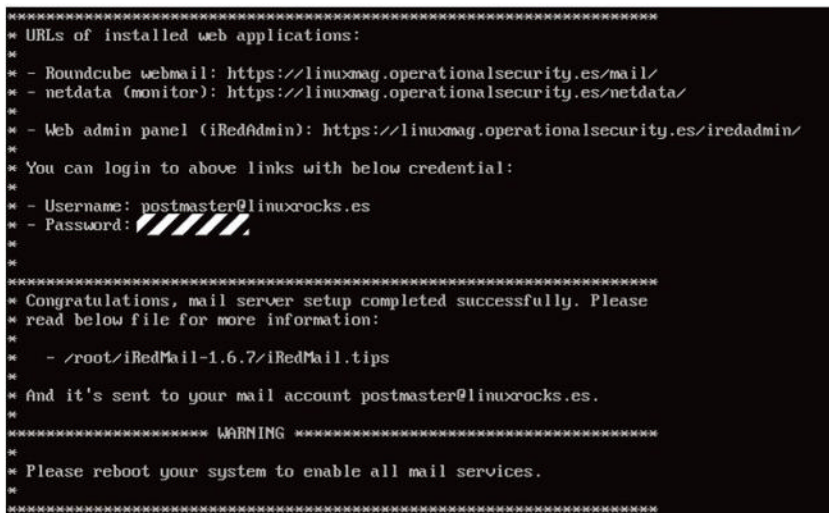


Figure 5: A successful iRedMail install.

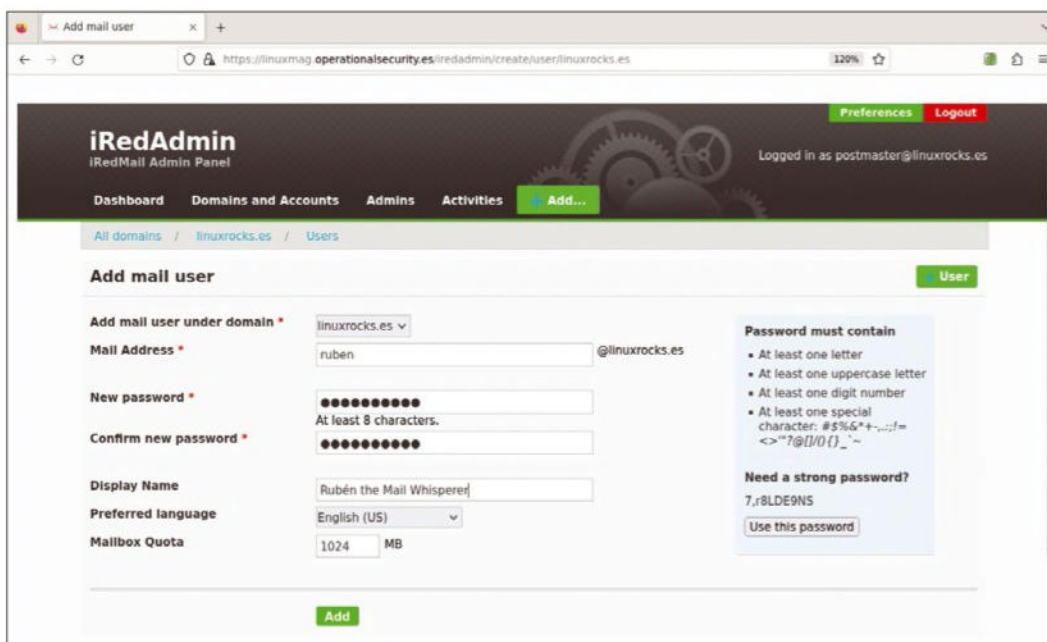


Figure 6: iRedAdmin lets you create email accounts for users and add domains to the email server.

email accounts for users, adding domains to the list of domains for which you host email, managing quotas, and viewing the logs (Figures 6 and 7). The free version of iRedAdmin [5], which is set up by the iRedMail downloadable installer, falls a bit short for all but the simplest scenarios. Theoretically, an unlimited number of domains may be hosted, but in practice you need to configure DomainKeys Identified Mail (DKIM) email authentication signing for each (see the “Email Connectivity” box). DKIM keys cannot be managed from iRedAdmin, so you must configure them manually. The free iRedAdmin lacks quarantine management. By default, incoming email may be prevented from reaching the Inbox and stored in quarantine until the postmaster decides its fate. One reason an email might be quarantined is if it looks like it might carry malware or spam. The problem is, because iRedAdmin does not have an interface for dealing with quarantined messages, email that is quarantined will never be checked and can’t be released. Quarantine management is a paid feature included in iRedAdmin-Pro.

Email Filtering

iRedMail comes with self-updating blacklists and SpamAssassin for sorting out spam. Email delivered from a server listed on a spam blacklist (e.g., lists from Spamhaus) are rejected; other email that looks suspicious will be classified as spam, as well. ClamAV is also included to identify messages carrying malware and isolate them.

Although the implementation delivered by the downloadable installer is serviceable, it is a bit disappointing. In the default configuration, spam is sent to quarantine instead of into the recipient’s Junk folder, which

is the behavior most users expect. Additionally, SpamAssassin does not perform Bayesian filtering and does not autolearn. Ideally, an email server should be able to learn which email users want to read and which

Email Connectivity

An email server requires a reputable, static, publicly reachable IP address. You might manage to make it work with dynamic IP addresses, but I don’t recommend it unless you are testing only. Each domain for which you host mail needs to have a mail exchange (MX) entry on its DNS records pointing to your server, such as:

```
linuxrocks.es. 3600 IN MX 10 linuxmag.operationalsecurity.es.
```

This record lets email servers trying to send mail to your users learn to which host they need to connect. In other words, if a Gmail user wants to send email to *ruben@linuxrocks.es*, Gmail will look up the MX record for *linuxrocks.es* and discover that the associated mail server is at *linuxmag.operationalsecurity.es*.

Adding a sender policy framework (SPF) record lets other mail servers know that the owner of *linuxrocks.es* has authorized your server to send email in their name.

```
linuxrocks.es. 3600 IN TXT "v=spf1 a:linuxmag.operationalsecurity.es -all"
```

Proper DKIM records sets up a mechanism that uses public key cryptography to certify that your email has actually been sent from your server. The server’s public keys reside within a publicly available DNS record. Once DKIM is set up, your server will sign all outgoing mail. Servers that receive email from your server will then download the public key from the DNS records and verify the signatures of the email messages against it to determine whether they are legitimate or forged.

iRedMail sets up a DKIM engine on install, and outgoing mail is signed by default. Still, you need to upload your public key to the DNS registry manually. It is worth noticing that iRedMail will use the same DKIM key to sign every message, so if you are hosting email for both *linuxrocks.es* and *linuxrules.es*, both domains will be covered by the same key. I have found this does not work well (despite what the documentation says). Therefore, my advice is to use a separate DKIM key for each domain. More information about DKIM and other DNS records can be found in the documentation [6] [7].

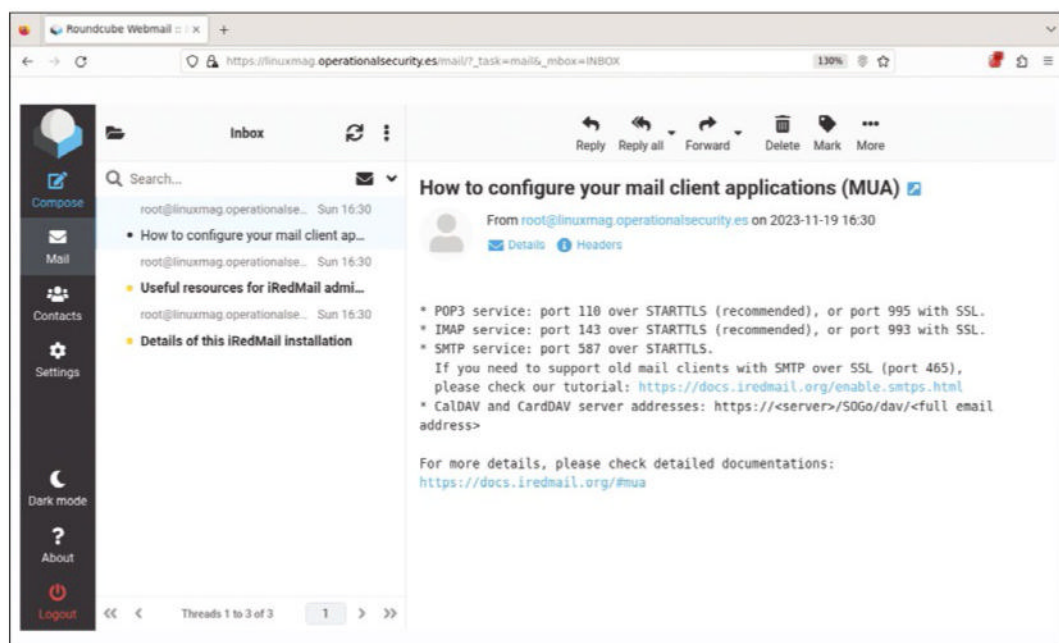


Figure 7: Both Roundcube and SOGo are available for iRedMail. Roundcube is a good option for people needing a somewhat light webmail panel, whereas SOGo is a full groupware solution with CalDAV and CardDAV support.

they don't to adapt the filters to their preferences. In a typical email server, this is done by having the spam filter take notice of which email is manually moved into the Junk folder by the users, and which email accidentally placed in the Junk folder is moved elsewhere by the users. The iRedMail instance you get from the downloadable installer can be configured to behave properly with autolearning spam filters that place spam mail in the Junk folder, but it is a somewhat involved process [8]. iRedMail Easy does the correct thing out of the box and does not force you to tweak spam filtering manually. There is no reason at all why the downloadable installer couldn't do the same. My hunch is that iRedMail Easy takes the proper approach and the downloadable installer takes the crude approach because the iRedMail organization wants to push you into using iRedMail Easy.

Conclusion

If you need to deploy an email service for a small organization (e.g., you need an email provider for a single domain), the iRedMail downloadable installer is an option worth considering. It will certainly turn a machine

into an email server very fast, with most of the features you expect included.

One of iRedMail's strengths is its portability. Other free open source software email appliances (e.g., Mail-in-a-Box) work on a specific distribution or operating system, but iRedMail can run on an astonishing number of platforms. Because of this advantage, if your server fleet comprises instances of a given system (e.g., OpenBSD), the system administrator won't have to set up an alien platform just for email.

On the other hand, if the out-of-the-box configuration does not work for you, you will have to perform a number of manual steps that negate, up to a point, the advantage of deploying an easy-to-roll solution. Specifically, if you need to change the default configuration of the spam filter, you will have to face a quite involved process. Also, adding more domains to your mail host is doable from the iRedMail's control panel, but adding DKIM keys for those domains (which is a necessity these days) must be done manually from a console. In any case, iRedMail still takes less effort than building a full email stack manually from the ground up. ■

Info

- [1] "An overview of the Citadel BBS" by Rubén Llorente, *ADMIN*, issue 57, 2020, pg. 38, [<https://www.admin-magazine.com/Archive/2020/57/An-overview-of-the-Citadel-BBS>]
- [2] iRedMail: [<https://www.iredmail.org/>]
- [3] iRedMail Easy portal: [<https://easy.iredmail.org/signup>]
- [4] iRedMail documentation: [<https://docs.iredmail.org/>]
- [5] iRedAdmin free: [<https://docs.iredmail.org/migrate.or.upgrade.iredadmin.html>]
- [6] Sign DKIM signature on outgoing email for new mail domain: [<https://docs.iredmail.org/sign.dkim.signature.for.new.domain.html>]
- [7] Set up DNS records for your iRedMail server: [<https://docs.iredmail.org/setup.dns.html>]
- [8] Auto-learn spam/ham with Dovecot imap_sieve plugin: [<https://docs.iredmail.org/dovecot.imapsieve.html>]

The Author

Rubén Llorente is a mechanical engineer who ensures that the security measures of the IT infrastructure of a small clinic are both legally compliant and safe. He is also an OpenBSD enthusiast and a weapons collector.



CLOUDFEST

March 18-21, 2024
Europa-Park, Germany

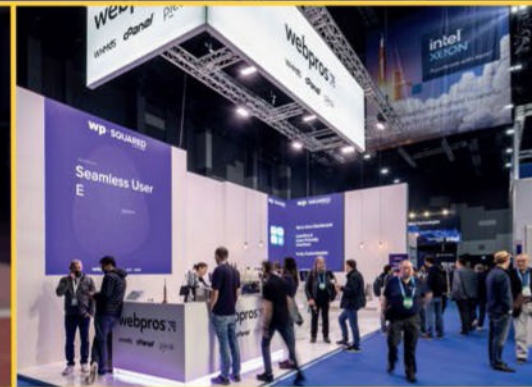
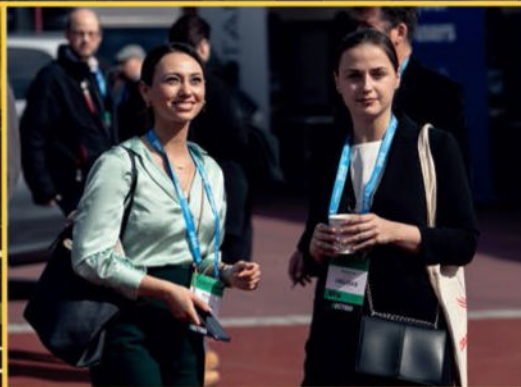
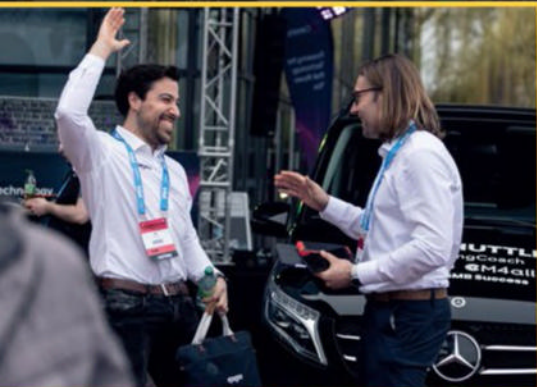
The world's largest cloud industry event is ready to once again take over a spectacular European amusement park to facilitate new partnerships, deep knowledge sharing, and the best parties the industry has ever seen.

8,000+ Participants

150+ Partners

250+ Speakers

80 Countries



Start your CloudFest Journey
AND SAVE €499!

With your FREE Code: **CFxADMIN**

scan me!



reg.cloudfest.com

cloudfest.com



Computational storage that supports storage operations

Smart Assistant

ScaleFlux delivers performance gains of 50 percent and more with computing power built directly into the network card to relieve the burden on the CPU. By Martin Loschwitz

ScaleFlux competes with NVMe-based drives by offering not only store but compute functions, as well. However, relieving the load on the CPU and ensuring higher bandwidth and lower latency comes at a greater price and begs one question: Is it worth the effort?

Where Is the Bottleneck?

Every server has a bottleneck that limits its performance somewhere, but the exact whereabouts of this bottleneck tends to differ because the manufacturers of the individual components have been playing a cat-and-mouse game for years. New CPUs usually come with new chipsets that enable higher bandwidths for the connected devices and RAM, which often makes the storage devices the bottleneck. The storage device manufacturers then follow suit and launch faster devices so that the network suddenly becomes the performance limit. The network in particular has seen major innovations in recent years, including the now widely available 400Gb Ethernet, which has quickly passed the buck back to storage devices. The situation can become particularly critical when components interact, especially if these components happen to be exposed to a heavy compute load

(e.g., database nodes, servers with running Hadoop or Redis instances, or members of an Elasticsearch cluster). Several factors come together. In particular, individual steps such as compressing and indexing data or cleaning up data records that are no longer needed require serious bandwidth on the connected storage medium on the one hand and high compute power on the part of the CPU on the other. In the worst case, this leads to a single system slowing down an entire setup, such as when the central database can no longer respond to incoming queries or cannot respond quickly enough. Neither users nor administrators like this one bit, because – especially on the end-user side – using a service in this scenario is a pain. The problem is not new, which explains why the hardware industry started looking for solutions years ago. Most admins will be familiar with the term “offloading” in a network context, which means that a network interface card (NIC) no longer simply processes the traffic flowing through it, but also takes on computing tasks. If you are using virtual extensible LAN (VxLAN), packets need to be tagged accordingly (VxLAN tag) by the host CPU or by the NIC if it implements suitable functionality on its chip and which affords

several advantages. The computations take place closer to the network traffic, which offers a boost in terms of performance. Moreover, the NIC is optimized for this task, which also tweaks performance. What is almost as important is that tasks performed by the NIC no longer burden the host CPU, which can then take care of other things. NICs with the appropriate functions are now available from all the major manufacturers, and support is now implemented extensively on Linux (e.g., by the Data Plane Development Kit, DPDK).

Computational Storage

The computational storage approach for storage devices has similar goals. Its beginnings were pretty banal. For example, the principle of storage tiering, which treats hot and cold data differently, is well established. Hot data resides on fast flash memory, cold data on slow disks – or even on tape if they are intended for archiving. That hard disks are mentioned at all shows the age of the approach. Because of their design, hard disks have far lower bandwidth than flash memory and far higher latency. Installing flash memory – which was still extremely expensive at the time – alongside a hard disk meant that the advantages of the two

solutions could be combined, at least for writes. However this approach is by no means sufficient for today's applications (e.g., large databases and Hadoop and others), without even taking into account the throughput that current artificial intelligence (AI) solutions require.

In the meantime, other approaches were vying for the users' favor. For example, data processing units (DPUs) were popular for a while. These special components were also used in servers and were intended to support the NIC. DPUs are still found today, particularly in network devices, but the approach failed to catch on in general-purpose machines. One problem is that the software needs to cooperate closely with the DPU; in fact, it has to be more or less completely coordinated with it, which makes cooperating with software from the open source world difficult. By way of an example, Facebook developed a key-value store named RocksDB that is capable of leveraging DPUs in appropriate setups, but it can't do so without specific customizations to the database. The prebuilt solutions for this scenario are only available from individual manufacturers who require customers to pay quite heavily for the privilege, and this obviously stands in the way of genuinely widespread use. Today vendors such as ScaleFlux [1] have entered the market to take the principle further and build computing power directly into the network card. This approach eliminates some of the disadvantages of previous solutions, such as the need to use additional hardware components. What is particularly important from the provider's perspective is that the solutions need be transparent for both the operating system and the user. The close connection between the development of hardware and software, which was still the rule for bridge solutions such as DPUs, should therefore be eliminated. The solution is achieved by the storage drives themselves directly intervening in the data stream and carrying out various operations that improve throughput,

latency, and storage density. Additionally, computational storage drives implement various features that are missing in normal NVMe devices, such as the ability to perform atomic storage operations. These features give the administrator more space elsewhere, for example, by eliminating the need for the database to take care of atomic operations itself. Compared with the less popular approaches, such as those based on DPUs, typical computational storage drives offer admins the option of deploying them in many more setups on the one hand. On the other hand, the lack of adaptation to specific applications also means that it is not possible to squeeze every last ounce of performance out of every database – precisely because the hardware compute routines are not adapted to the respective programs.

What ScaleFlux Offers

ScaleFlux is considered a pioneer in the computational storage industry. The company is older than you might suspect. The people at ScaleFlux have been working on computational storage drives since 2014, and by 2020, their hardware was already well established on the market. Since then, ScaleFlux has continuously expanded its own portfolio and can point to success. Alibaba Cloud, which is the largest cloud provider in China, relies on ScaleFlux hardware to accelerate its in-house PolarDB, a cloud-native database with its own storage nodes on which the capabilities of ScaleFlux hardware can be particularly well leveraged, according to Alibaba. What does ScaleFlux deliver, and how do the drives work in detail? A look at the company's current portfolio provides clarity. Although they still advertise their 2000 series drives, the far newer 3000 series is the real highlight. The drive comes in three forms: as an add-in card (AiC) with a PCIe connector, as a solid-state drive (SSD) with a U.2 interface, or as SFX 3000, which allows vendors to build their own hardware with ScaleFlux features (Figures 1-3). You can also

get the prefabricated drives in computational storage drive (CSD) and NVMe SSD+ (NSD) flavors, the later of which does not have the capacity multiplier feature of the CSD – but more on that later. In any case, one thing is clear: ScaleFlux drives can run on any server that has free PCIe slots or can accommodate regular U.2 NVMe devices, and this condition will be true for the vast majority of off-the-shelf standard servers. The manufacturer's specifications on the drives' capabilities make for pretty impressive reading. If you are looking to virtualize the devices' functions, you could go for a complete single-root I/O virtualization (SR-IOV) implementation with a total of 15 different virtualization functions. Hardware support for various security functions in line with the Trusted Computing Group (TCG) Opal 2.0 specification is also part of the hardware. The provider also states a speed of 7.1GBps for sequential reads and 4.8GBps for sequential writes. The I/O operations per second (IOPS) rates are also worthy of note. The claim is more than one million IOPS with compression for read-write workloads (ratio 70/30) and 4KB block size enabled. That said, you cannot directly compare these values with other standard NVMe devices; just make sure you examine the ScaleFlux hardware's capabilities more closely.

ScaleFlux Components

This investigation starts with the hardware components that you as the admin adopt into your data center. Unsurprisingly, NAND chips are used for storing the data; these chips are the go-to standard for the NVMe industry. However, the heart of the ScaleFlux drives beats in the processor, which in the case of the 3000 series (Figure 4) comes in the form of an ARM-based application-specific integrated circuit (ASIC), which is one of the biggest differences compared with the previous series. The 2000 series still relied a field-programmable gate array (FPGA)-based chip, which turned out to be too



Figure 1: Computational storage SSDs from ScaleFlux come in three form factors: as an AIC card for the PCIe slot, ... © ScaleFlux



Figure 2: ... as a regular NVMe with a U.2 interface and a 2.5-inch form factor, and ... © ScaleFlux

low powered in the course of development. To match its performance, the 3000 series also comes with a state-of-the-art PCIe 4.0 interface, whereas the predecessor series had to make do with the significantly slower PCIe Gen 3. The system on a chip (SoC) is supported by RAM and various special components for accelerating individual steps. Although the computational components are identical across all models in the series, the number of NAND chips and their sizes differ. ScaleFlux NVMe devices are currently available in sizes of 3.2, 3.84, 6.4, and 7.68TB.

Similarities

The central difference between the CSD and NSD



Figure 3: ... as M.2 memory modules for the corresponding slots. The type and scope of the installed components are identical across the different form factors. © ScaleFlux

drives is the capacity multiplier, as mentioned; beyond that, the devices offer the same set of basic features. The foundation is transparent compression and decompression of the in-flight data of a database. The goal of the exercise is obvious: Compression

The SoC also implements the codecs and is therefore even faster than a generic ARM CPU. For this to happen, the ARM SoC in the memory drive compresses the data before storage in flash memory and only unpacks the data again when requested by the respective database. As the manufacturer promises, this process is completely transparent from the application's point of view. The database running at the top of this kind of construct remains blissfully unaware of the data compression. It can still read and write data as fast as a normal NVMe, thanks to the computing power built in to the drive. On the application side, high-volume database queries and queries in which large volumes of data need to be processed benefit from compression. The

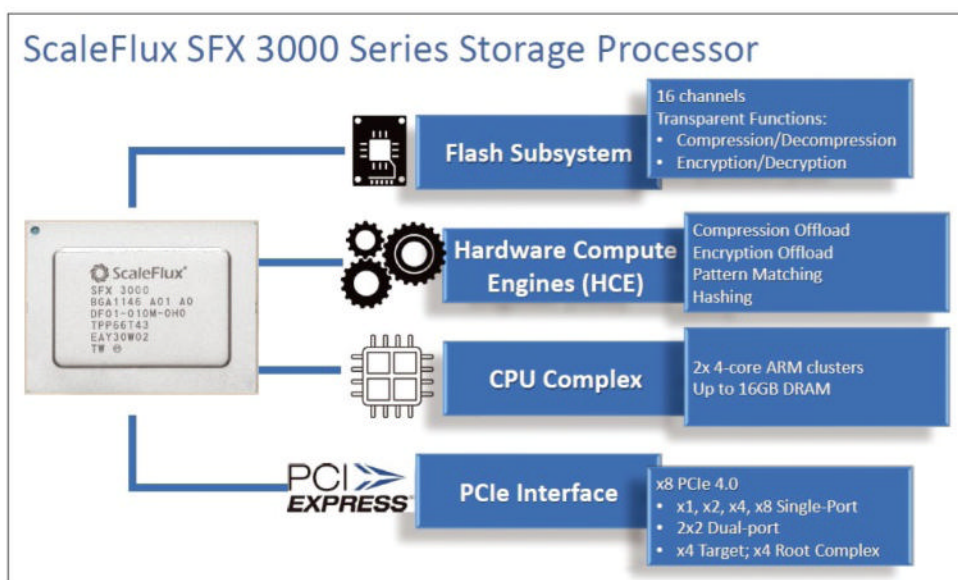


Figure 4: ScaleFlux NVMe drives offer typical NAND memory, as well as a dedicated ARM SoC with its own RAM and special extensions for tasks such as data compression. © ScaleFlux

manufacturer points to three advantages. First, read operations can be processed far more quickly and with lower latency than with normal drives because the compressed blocks can be read as fast as with normal SSDs, but much more data can be read in the same time. Second, in mixed read and write workloads, which are standard in most data centers, compression also further reduces latency because write-to-read interference is lower. This term refers to the wait that occurs when, say, the controller of an SSD switches between read and write commands. Again, because more data is read in parallel, fewer changes between read and write mode are required overall, which improves overall latency. Third, according to ScaleFlux, compressing the data prolongs the service life of the installed NAND chips because lower volumes are written to the data carrier all told. At this point, it is worth recalling the way flash chips basically work. At their core, they use chemical reactions to store data. However, these reactions cannot be repeated for an arbitrarily long period of time, which is precisely why the drive writes per day (DWPD) number of a drive is already a decisive factor for administrators during the acquisition process. It indicates, in roughly simplified terms, how often the drive can be completely overwritten in total without fear of data corruption from defective blocks. As people in the industry say, SSDs do not break down less often than hard disks, but they are more predictable, and this is quite true in essence. ScaleFlux itself perhaps somewhat over-extravagantly refers to this feature as the endurance multiplier, but without providing any concrete numbers relating to the life expectancy boost that the feature offers a NVMe drive. Don't bother looking for DWPD data; however, you will find information about the degree of compression that ScaleFlux looks to achieve, which is said to be around a factor of four on average – provided the data to be processed can be compressed. The 3000 series models from ScaleFlux implement encryption in line with the

TCG Opal specifications. Encryption is again transparent from the operating system's point of view; again, the application is unaware that the data is encrypted on the storage medium. Because the drive handles the encryption process very quickly thanks to smart codecs, there are no disadvantages in terms of performance.

Differences

Against the background of compressing drives, the limelight is again firmly on the capacity multiplier, which ScaleFlux massively advertises for its product. The keyword is "implicit" compression, which is the only central difference between NSD drives and the significantly more expensive CSD drives.

The capacity multiplier is the manufacturer's approach to pretending that the drive is far larger during the provisioning process. This assumption has certain advantages; for example, the space gained through compression can be put to good use by the operating system. On the other hand, some risks are associated with the approach. If the degree of compression on which the theoretical drive size is based cannot be achieved, the drive could fill up faster than the size specification would lead you to believe. At least in this scenario the drive would give the operating system correct usage figures. A situation in which

50 percent of the drive's space is still available according to `df`, but the drive is actually already full, is unlikely in this respect. Whether the value this feature adds is worth the extra cost to you is ultimately your decision.

Made to Measure

Another thing common to both series is the integrated circuit (IC) used. The manufacturer not only offers implementation support for it but even helps with development, if necessary. The firmware running on the ScaleFlux drives also plays a role. CSware (Figure 5) offers the possibility to roll out special firmware for specific purposes with the help of the manufacturer. Companies like Alibaba are likely to have made extensive use of this very option but are likely to encounter greater interest from ScaleFlux than small companies with small platforms.

In any case, regardless of size, all companies have the option of installing the ARM SoC from the 3000 series in their own use cases, which is especially interesting if you want to take advantage of computational storage without relying on ScaleFlux equipment.

What Others Say

Lower latency and higher bandwidth can be achieved with ScaleFlux

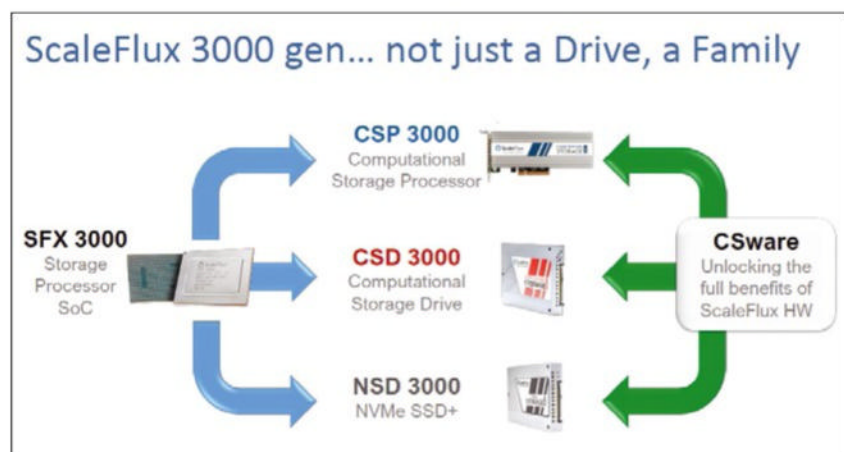


Figure 5: The SFX 3000 chip is the heart of the ScaleFlux architecture. In combination with matching firmware, it is said to provide even better performance values than the standard version. © ScaleFlux

hardware, as can a higher storage density and, in turn, a lower wear level factor for the SSDs. The question is nevertheless how great are the benefits resulting from the use of these components. Plenty of sources offer concrete figures online, including database specialist Percona, which tested ScaleFlux in the context of both MySQL and PostgreSQL. Further figures, especially those for Oracle, are provided by the US trade journal StorageReview.com [2]. The test parameters are pretty much comparable: Random and sequential read and write operations were measured with different chunk sizes of about 4KB and 64KB, as were read and write operations by Oracle. Both providers arrive at test results that are coherent in themselves and coherent in relation to each other. Random reads and writes at the 4KB chunk size produce similar results across all drives when reading data, regardless of whether the 2:1 compression ratio was maintained during testing. Both the 3.84TB and the 7.68TB CSD 3000 drives are in a range between 850,000 and 900,000 IOPS, but writing with the same block size leads to significantly different results. Without data compression, the drives achieve 380,000 to 420,000 IOPS, but well over 700,000 with compression. When reading with a block size of 64KB, the IOPS number drops to a meager 6,000 without compression and to 7,000 with compression. However, the write processes are even more interesting. With compression enabled, both the small and large ScaleFlux NVMe do well at 6,000 IOPS. Without compression, both devices return a fairly poor value below 2,000. In all the cases described, the compression used also had a positive effect on latency, which was reduced by 30-40 percent on average. The compression factor has a less drastic effect in the case of database applications. Here, the baseline of the small 3000 series NVMe is at a good 250,000 IOPS, whereas the large NVMe with compression enabled is easily north of 300,000 IOPS. The

values for normal SQL (i.e., MariaDB or PostgreSQL) and Oracle are clearly comparable, whereby Oracle particularly benefits from enabling compression on the large SSD. The figures are 220,000 IOPS compared with 330,000 IOPS in this case.

StorageReview.com has identified another use case that showcases the benefits of ScaleFlux drives in a better way: storage drives in the virtual desktop infrastructure (VDI) context. The values measured for the large SSDs with compression enabled are sometimes more than 100 percent higher than those of the small SSDs without compression, but that also makes sense. In the VDI context, the proportion of data to be stored that can be meaningfully compressed is likely to be significantly larger than in most database scenarios, which traditionally work with fairly small chunk sizes.

Although StorageReview.com exclusively compared ScaleFlux drives in its tests, Percona additionally offered a direct comparison with an Intel P4610 drive [3]. The comparison is realistic because the Intel device is one of the standard NVMe drives and therefore a standard solution in many application scenarios. That said, the Percona test was carried out a while back and therefore compares the Intel drive with the CSD 3000's predecessor, the CSD 2000, which is probably why the ScaleFlux drives were only slightly ahead of their Intel counterparts in terms of speed in many individual tests.

However, if you combine the test results from Percona and StorageReview.com and include the provider's statements about the speed improvements of the 3000 series, a conclusive overall picture emerges. The speed advantage of the ScaleFlux 3000 series compared with a setup without the special hardware is on average 50 percent greater for MySQL, and even up to 200 percent greater for PostgreSQL, depending on the scenario. Thanks to compression, ScaleFlux drives, on average, stored more than 200 percent of the data compared with an Intel device given maximum usage.

Conclusions

As usual, I was unable to elicit concrete prices from ScaleFlux, but searches revealed that the manufacturer's devices are priced at about the same level as normal NVMe drives without the additional features.

NVMe has become far cheaper since, so ScaleFlux hardware might seem a little bit expensive. Technically, however, setups are likely to benefit considerably from the advantages these devices offer.

Any database administrator will be delighted with the 50 percent and more performance gains they can achieve by simply replacing the storage hardware. That the CPU also finds relief is a welcome side benefit. Another thing is also clear: If you customize the drives' firmware to suit your own application scenarios – with a little help from the provider – you can probably tease out even more impressive performance gains.

ScaleFlux is already turning its thoughts in a different direction. Currently, the ARM SoCs do not yet include codecs for multimedia applications, which in particular would benefit considerably from higher throughput and lower latency; the offloading factor for the host CPU would also be significantly greater. In perspective, ScaleFlux drives for the multimedia sector are not only conceivable but quite likely. ■

Info

- [1] ScaleFlux: <https://scaleflux.com>
- [2] "ScaleFlux CSD 3000 SSD Review" by Charles P. Jefferies, January 9, 2023: <https://www.storagereview.com/review/scaleflux-csd-3000-ssd-review/>
- [3] Testing the Value of ScaleFlux Computational Storage Drive (CSD) for PostgreSQL, Percona whitepaper, 2021: https://www.percona.com/sites/default/files/Testing_the_Value_of_Scaleflux_for_PostgreSQL.pdf

The Author

Freelance journalist Martin Gerhard Loschwitz focuses primarily on topics such as OpenStack, Kubernetes, and Chef.



REINVENTING

HPC



YOUR INVITATION


Dear Reader,

Are you an HPC User, Vendor, or Provider?

We need your input as we embark on the mission of REINVENTING HPC.

Consider this your occasion to be a part of history in the making.

SEE YOU AT ISC 2024



Build and host Docker images

Master Builder

When facing the challenge of packaging your application in a container, take into account your needs in terms of handling and security and investigate sensible options for hosting your own registry. By Martin Loschwitz

Many a developer looking to package their own application in a container for the first time finds themselves out of their depth. Administrators who have not had much experience with containers to date are in a similar position if they are looking to containerize small tools for their own use and deliver them locally. The questions are many. Although the required knowledge can be painstakingly gathered, it takes a long time, and it's not much fun to boot. This article rushes to the rescue. Besides the basics of image building, I look at best practices and continuous integration and continuous deployment (CI/CD) mechanisms, as well as the question of a good DIY image registry. This much can be revealed in advance: The topic is not quite as complex as many critics assume and claim.

The Rise of Containers

Containers are on the rise, whether you like it or not. For years, this magazine has been pointing out that the major distributors, Red Hat and

SUSE in particular, will be relying on containers in the future, if only because it saves them a lot of work. For example, Red Hat need only maintain its own software in containerized form once to make it available on RHEL 7, 8, and 9.

As long as a runtime environment for operating containers is available on a system, the underlying operating system hardly matters. Containers also offer a very useful technical alternative for software that the large distributors do not have in their own portfolios. Although Red Hat currently maintains various versions of MariaDB or MySQL for its enterprise distributions, in the future they will simply point to the official container images of the manufacturers instead of doing the work themselves. This effect can already be seen on the desktop. Recently, Fedora announced that it would no longer maintain LibreOffice in package form but would point its own users to the official LibreOffice Flatpaks. Under the hood, Flatpaks are no more than containers, for which Red Hat doesn't have to put

any effort. It is easy to imagine how this trend will continue.

However, not only do the manufacturers benefit from this approach, thanks to the container format, the developers of smaller applications could deliver their own applications to the users quickly and easily. They don't have to deal with annoying details like package managers and different packages for different versions of a distribution. In this respect, containers certainly are tempting.

Questions and More Questions

The questions are always the same: How do you turn the program source code, which is available as a tarball, into Docker containers that can be delivered by Docker Hub? Do practical CI/CD tools already exist that not only facilitate container building, but also professionalize it and automatically find errors? Is the use of Docker Hub even a good idea, or would a local registry, designed specifically for local images, be a better idea?

Luckily for you, Docker developers include the tools you need to create and deploy local container images with Docker, which fortunately does not involve complicated syntax or a whole new format for the metadata of the image to be built. To get your first custom image up and running quickly, all you need is a local folder on a Linux system with some basic tools and a Docker runtime environment in place.

Preparation

In plain English, this means that before image building, a few preparations need to be made on the build system. You can use your own computer for this process, although in organizations that build many images, the norm in recent years is to use a specially prepared system instead. This system does not even have to be a physical machine; a virtual instance is quite okay. At the latest, when a CI/CD toolchain comes into play, the build process will no longer take place on a local system anyway, although the development of the image will, including any necessary test runs. The developer's own preferences ultimately determine the desired procedure.

The general approach to getting a new image off the ground is entirely independent of personal preference. Docker itself ships the build functionality as a `docker` component. To use it, though, Docker must be installed on the local system. The first step is to set up the community edition of Docker locally. The following example assumes a system with Ubuntu 22.04. To begin, you need to install some required software management packages (Listing 1, line 1); download the GPG key used to sign Docker's

package lists, which Apt relies on to ensure that the packets come from a reliable source (line 2); add the package sources for Docker to the local list of package directories (line 3); and update the local package cache (line 4). Last but not least, run `apt` to install the community edition of Docker on your system (line 5). A call to `systemctl` (line 6) should then show *active* as the *docker.service* status. Assuming this is the case, the installation worked and the commands you need to build the image are in place locally.

Null and Void

When building Docker images, you start with a working directory that is probably empty, so as the first step, you need to create a new folder, which won't stay empty long, because of the command you run in the very next step,

```
mkdir nginx
touch .dockerignore
```

to create a file that acts as an exclusion list for files that Docker doesn't need to consider when building the image.

The rest of the process is a little more complicated and requires some knowledge of Docker images. Basically, you have several options at this point. You can either use a prebuilt base image for your image or build it yourself.

Remember that a running container on a system initially only contains a filesystem. The Linux kernel runs various functions (namespaces, cgroups) to move the filesystem to a sort of isolated area of the system and operates it there. Unlike in full virtualization or paravirtualization mode, a Docker

image does not require its own kernel. However, it does have to contain all the files that the desired program needs to run. After all, the standard host filesystem is a no-go area later for the active container.

This lockout can be removed by bind mounts and volumes later; however, you will always want to build the container such that it has no dependencies on software stored externally. The premise is that a Docker image is always self-contained; that is, it has no dependencies on the outside world.

Virtually every Docker image therefore contains a reasonably complete filesystem of a runnable Linux system. How complete the filesystem is depends strongly on the application and its dependencies. Some developers intuitively go for tools like `debootstrap` and build their own basic systems, but this idea is not particularly good. Even a basic installation of Debian or Ubuntu today includes far more than you really need for a container. Additionally, completely DIY images also need to be completely self-maintained. Depending on the situation, this can involve a serious amount of work.

Docker saw this problem coming and practically eliminated it with a small hack. Instead of keeping the entire contents of the container image locally, Docker uses the `FROM` command when building the image. The command draws on a public image on Docker Hub as the basis for the image to be created and only adds the components requested by the developer.

All major distributors maintain their own micro-images for container building with their own distribution and make them available on Docker Hub. The same applies to Red Hat

Listing 1: Docker Environment

```
01 $ sudo apt install apt-transport-https ca-certificates curl softwareproperties-common
02 $ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
03 $ echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu
    $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
04 $ sudo apt update
05 $ sudo apt install docker-ce
06 $ sudo systemctl status docker
```

Listing 2: NGINX Dockerfile

```
# Pull the minimal Ubuntu image
FROM ubuntu
# Install Nginx
RUN apt-get -y update && apt-get -y install nginx
# Copy the Nginx config
COPY default /etc/nginx/sites-available/default
# Expose the port for access
EXPOSE 80/tcp
# Run the Nginx server
CMD ["/usr/sbin/nginx", "-g", "daemon off;"]
```

Listing 3: File default

```
server {
    listen 80 default_server;
    listen [::]:80 default_server;

    root /usr/share/nginx/html;
    index index.html index.htm;

    server_name _;
    location / {
        try_files $uri $uri/ =404;
    }
}
```

Enterprise Linux (RHEL), SUSE, Ubuntu, Debian, Arch Linux, and the particularly lean Alpine Linux, which is optimized for container operation. Distributors are very good at building mini-images of their own distributions and can do it far more efficiently than an inexperienced end user.

Distributors regularly maintain their images, as well. When a new version of a base image is released, you just need to rebuild your own image on the basis of the new image to eliminate security or functionality issues. One practical side effect is that the local working directory for image building remains easy to understand and clean.

Another great feature of container building now comes into play: During the build, CMD can be used to run commands that, for example, add packages to the distributor's base image. The content that the developer needs to contribute to their own

application is therefore typically just the application itself and its files, along with the dependencies that are not available in packaged form for the base image you are using.

An Example

After all this theory, it's time to build the first container. I deliberately kept the following example as simple as possible. It describes building a container with Ubuntu 22.04 as the basis (**Figure 1**) running NGINX, with the web server serving up a simple HTML page. The file shown in **Listing 2**, named Dockerfile, is one of the two basic ingredients required. To experienced container admins, the contents may seem less than spectacular, but if you haven't built a container yet, you may be wondering what each command does.

FROM is the previously mentioned pointer to a base image by a provider – Ubuntu

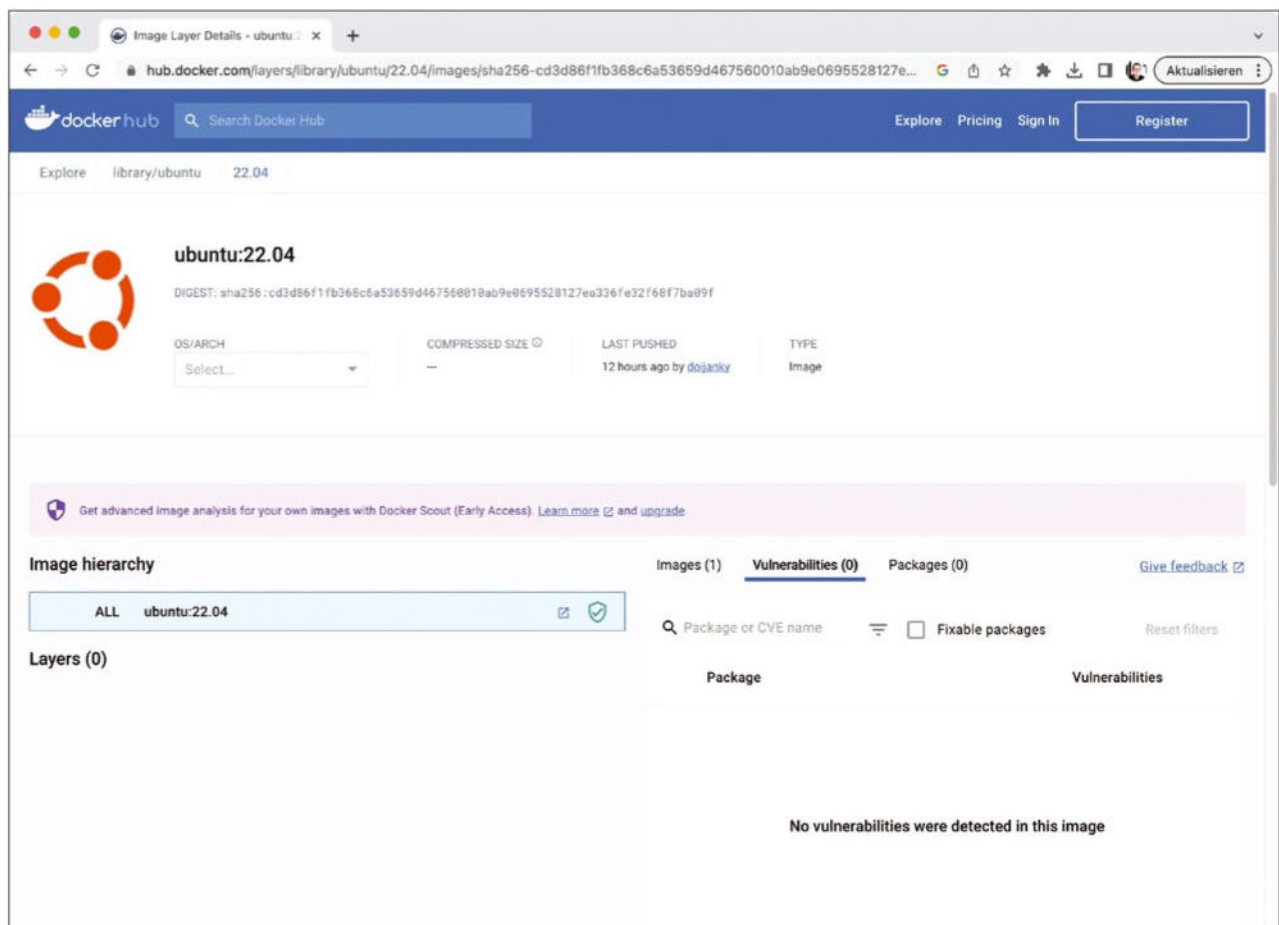


Figure 1: A preconfigured base image from one of the major distributors is recommended for building your container. A newcomer is unlikely to be able to put together a leaner image without compromising some of the functionality.

in this case. If you prefer a specific version, you can specify it after a colon (e.g., `ubuntu:22.04`). The `RUN` command initiates a command that Docker executes during the build within the downloaded base image. In the example, it installs the `nginx` package. `COPY` copies a file from the local folder to the specified location in the image. The example assumes that a file named `default` is in the build folder and later will be in the NGINX site configuration in the image (Listing 3).

Again in Listing 2, `EXPOSE` instructs the runtime environment to expose port 80 of the running container to the outside world to allow access. Docker invokes `CMD` to start the container. In the example, it calls NGINX with daemon mode disabled so that `stdout` remains open; otherwise, the runtime environment would terminate the container immediately. Next is building and launching the image:

```
$ docker build . -t lm-example/nginx
$ docker run -d -p 80:80 am-example/nginx
```

The first command triggers the process. You need to call it directly from the build directory. After doing so, you will find the finished image in the local Docker registry. The second command launches the image for test purposes. If `docker ps` displays the NGINX container afterward, the package build was successful (Figure 2).

More Fun with CI/CD

Granted, the example shown is unspectacular and leaves out many package building options, as well as options for running NGINX in a more complex configuration. For example, in real life, an NGINX instance always needs an SSL certificate along with the matching key.

The usual way to solve the problem in Docker is to subcontract a volume to the container at runtime where the respective files reside. However, for this to work as intended, you need to preconfigure NGINX in the container appropriately. You can use a static configuration for this, although you would need to modify the Dockerfile accordingly. Alternatively, you can use variables to pass in the parameters in the shell environment from which you launch the container as the admin. In the Dockerfile, the developer would then define the variable with an `ENV` statement and access it in the file itself with `$(VARIABLE)`. However, none of this hides the fact that the example is quite rudimentary. In everyday life, especially with more complex applications, you are hardly going to get away with such a small number of commands, not to mention the problems that arise from maintaining the image. For example, the image built here has not yet been published. Tests to check the functionality of the image automatically are also not planned to date.

All of this can be changed relatively quickly. The magic words are continuous integration and continuous development or automation and standardization of the image build and any testing. For example, an image developer wanting to rebuild an image just checks a new version of the Dockerfile into Git, and Git handles the rest automatically. When done, the new image is made available on Docker Hub and can be used.

Of course, the number of ready-made CI/CD solutions for Docker is now practically approaching infinity, not least because Kubernetes also plays a significant role in this business and has been something of a hot topic for the IT industry as a whole for years. No longer just a matter of building individual images, the goal is to create

Listing 4: Workflow Config for GitHub

```
name: ci
on:
  push:
    branches:
      - "main"
jobs:
  build:
    runs-on: ubuntu:22.04
    steps:
      -
        name: Checkout
        uses: actions/checkout@v3
      -
        name: Login to Docker Hub
        uses: docker/login-action@v2
        with:
          username: ${ secrets.DOCKERHUB_USERNAME }
          password: ${ secrets.DOCKERHUB_TOKEN }
      -
        name: Set up Docker Buildx
        uses: docker/setup-buildx-action@v2
      -
        name: Build and push
        uses: docker/build-push-action@v4
        with:
          context: .
          file: ./Dockerfile
          push: true
          tags: ${ secrets.DOCKERHUB_USERNAME }/
am-example:latest
```

complete application packages that find their way in a fully automated manner into the Kubernetes target instance at the end of a CI/CD pipeline and replace the workload running there without downtime.

You don't have to spend big money to implement CI/CD with Docker. GitHub is the obvious choice. It has comprehensive CI/CD integration for Docker, including the option of automatically uploading the finished images to Docker Hub.

Initially, much like this example, you have an almost empty working folder with a Dockerfile and possibly the required additional files. You first add it to the Git version management

\$ docker ps						
CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	
NAMES						
816532f135c8	server	"/docker-entrypoint..."	About a minute ago	Up About a minute	0.0.0.0:80->	
>80/tcp, :::80->80/tcp		nginx				

Figure 2: Assuming the build process worked, the container from the example can be launched and creates a working NGINX instance. © Haidar Ali [1]

system and then upload the repository to GitHub. For the directory, you then need to define the `DOCKERHUB_USERNAME` environment variable and a personal access token (PAT) in the `DOCKERHUB_TOKEN` variable. Next, add an action to your directory, which is an entire workflow in this example. For example, the `.github/workflows/main.yml` file in the repository might look like [Listing 4](#).

Once the file is in the directory, any changes checked into the repository will trigger an automatic rebuild of the image by GitHub, which will then use the given credentials to check the image into Docker Hub. Once in place on Docker Hub, the finished image can itself become part of CI/CD pipelines that, for example, control deployment within Kubernetes.

Mind you, GitLab and GitHub are just two of countless vendors trying to make a living with Docker CI/CD. Jenkins, the classic CI/CD tool ([Figure 3](#)), is also very much alive in this environment, as are many others.

Your Registry

By the way, GitLab offers features that are very similar to GitHub. If you don't want to spend money on GitHub to create your own private repositories, you can switch to a local GitLab instance instead. Also, if you do not want to make your images available to the public, you will need a private registry for your container images.

That said, running the repository is not as easy as you might think at first. Useful software for this task was not available under a free license for a long time. Fortunately, several providers now have suitable offerings on the market, and one of them is Docker itself. The command in [Listing 5](#) launches a local Docker registry. The command details are important.

The example assumes that the `/mnt/registry/` folder exists on the host, because it will be mounted to `/var/lib/registry/` later in the running container. You also need to create

the `domain.crt` and `domain.key` secrets on the host through Docker. You can do so with the first two commands,

```
$ docker secret create domain.crt certs/2
domain.crt
$ docker secret create domain.key certs/2
domain.key
$ docker node update --label-add 2
registry=true <hostname>
```

which also add the contents of the two files as passwords to Docker's metadata. Before adding the Docker service, the last line creates a label for the node running the registry.

Again, this example is very simple. For example, the option to secure access to images with a username and password combination is missing. Technically, this would not be a problem; the Docker documentation contains more information on the subject.

Running a registry with Quay ([Figure 4](#)) offers significantly more scope than the standard Docker

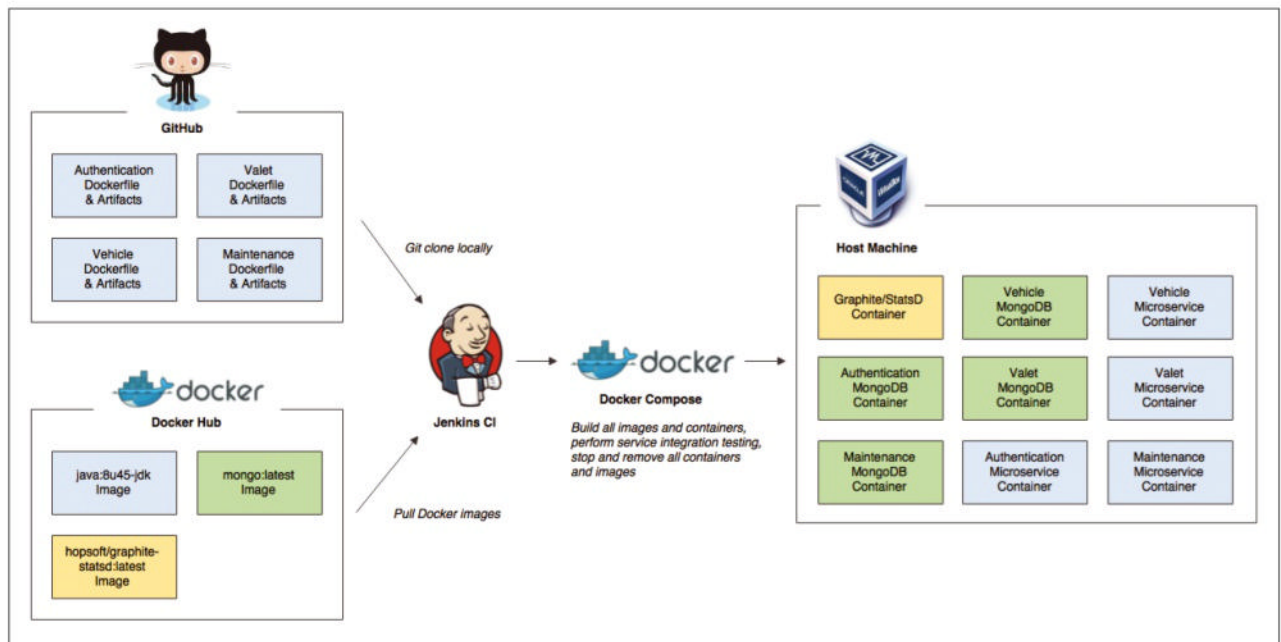


Figure 3: GitHub and GitLab now offer extensive CI/CD capabilities for Docker. Standard solutions like Jenkins help developers avoid strict dependency on a specific Git solution. © Gary Stafford [2]

Listing 5: Local Docker Registry

```
docker service create --name registry --secret domain.crt --secret domain.key --constraint 'node.labels.registry==true' --mount type=bind,src=/mnt/
registry,dst=/var/lib/registry -e REGISTRY_HTTP_ADDR=0.0.0.0:443 -e REGISTRY_HTTP_TLS_CERTIFICATE=/run/secrets/domain.crt -e REGISTRY_HTTP_TLS_KEY=/
run/secrets/domain.key --publish published=443,target=443 --replicas 1 registry:2
```

approach. The service, which was developed by Red Hat to a large extent, not only delivers images to clients but also has comprehensive CI/

CD functions on board in the background (Figure 5). The project [3] is available under a free license, but the setup is not very intuitive. The

simplest option is to roll out Quay in the form of a prebuilt container in Kubernetes.

Conclusions: Not Too Tricky

As the examples show, building Docker containers is not particularly complicated. Even running a separate registry for containers is quite easy, all told. If you are planning larger workloads that are based on containers, you will inevitably have to square up to the task of building images. The best idea is to use only official Docker Hub images as the basis for your own work. All other approaches involve a huge risk of working blindfolded and can quickly turn into a nightmare. When it comes to the practical process of building containers, CI/CD tools will help make the whole experience more convenient. ■

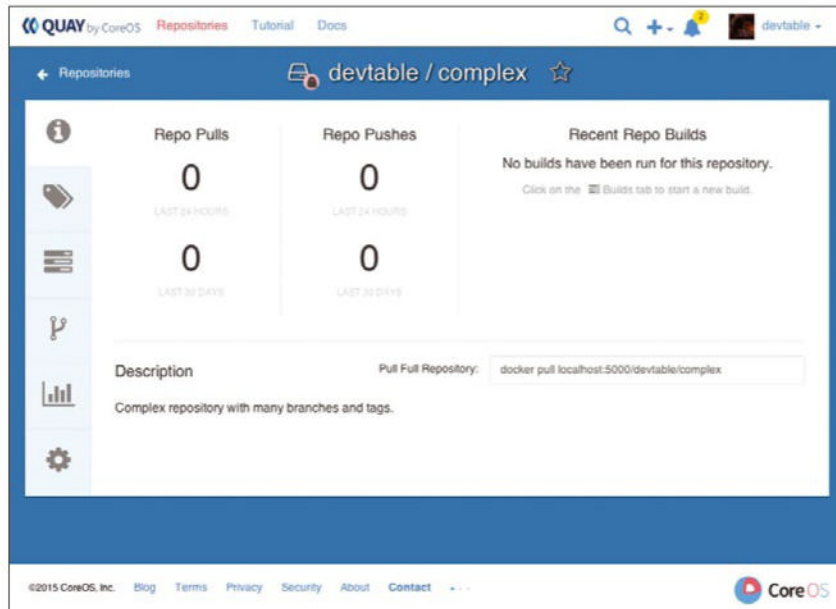


Figure 4: Quay, a registry for container images, provides statistical data on individual images in addition to the upload and download functionality. © Quay [4]

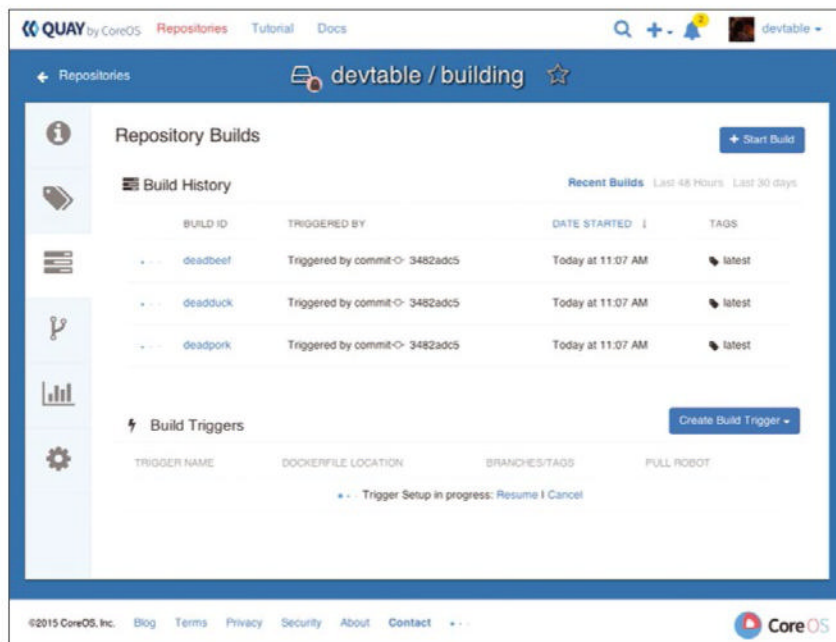


Figure 5: Besides the registry functionality, Quay has a lot of additional tech on board that includes an extensive CI/CD toolchain that supports developers wanting to build images.

© Quay [4]

Info

- [1] "Running the NGINX Server in a Docker Container" by Haidar Ali, May 2022, Baeldung: [\[https://www.baeldung.com/linux/nginx-docker-container\]](https://www.baeldung.com/linux/nginx-docker-container)
- [2] "Continuous Integration and Delivery of Microservices Using Jenkins CI, Maven, and Docker Compose" by Gary Stafford, January 2016: [\[https://programmaticponderings.com/2015/06/22/continuous-integration-and-delivery-of-microservices-using-jenkins-ci-maven-and-docker-compose/\]](https://programmaticponderings.com/2015/06/22/continuous-integration-and-delivery-of-microservices-using-jenkins-ci-maven-and-docker-compose/)
- [3] Quay on GitHub: [\[https://github.com/quay/quay\]](https://github.com/quay/quay)
- [4] Quay: [\[https://quay.io\]](https://quay.io)

The Author

Freelance journalist Martin Gerhard Loschwitz focuses primarily on topics such as OpenStack, Kubernetes, and Chef.





Dos and don'ts of backing up
Kubernetes storage

Securing the Load

Containers offer great flexibility, but the data they contain often needs to be backed up. Stateful applications that store their information in a container's persistent volume can be backed up in a variety of ways, but not all of them are easy. By Andreas Stolzenberger

In practice, switching to a consistent scale-out architecture with stateless containers means that users have to rewrite their existing applications – not necessarily the application logic, but the data storage and structural components. Existing database technologies such as SQL can only handle stateless scale-out operations with major limitations. Object databases would be far better suited but entail a fairly extensive redesign of existing applications. I already described this problem in detail in a previous article [\[1\]](#).

In this article, I look at some methods for backing up your Kubernetes applications, preferably SQL databases. The main aim is to ensure that these data backups can be restored quickly in the event of a failure. The overview is deliberately restricted to the architectures; you will always need to adapt the code to suit your environment and your applications.

Backup and Restore Problems

An early idea of scale-out applications in containers was to change the way in which applications store their data. Applications are distributed across several nodes and do not have to write persistent data to any drives.

The databases are distributed redundantly across groups of containers and only stored in temporary directories there. If a container fails, it loses the entire database, but redundant distribution prevents data loss. A newly started container synchronizes with existing containers and adopts the redundant data of the failed container. The files need to be backed up as objects that the application regularly stores in Amazon Simple Storage Service (S3) buckets or comparable object storage in the active cluster. All high-level storage functions, such as a rollback history of object changes and backups, are handled by the object storage system, which explains why early versions of container clusters did not even have functions for assigning a persistent volume (PV) to a container for data storage. So much for the theory of the scale-out world.

When users started to move existing applications that were not really suitable for a scale-out architecture into containers, the clusters had to introduce functions to assign persistent storage there. Unfortunately, these PVs have reinstated some legacy problems that platforms such as Kubernetes wanted to eliminate: backup and restore. Most of the current crop

of applications have kept their SQL back ends, which forces Helm charts and operators to roll out the majority of applications as a stateful set – on an architecture that was specifically developed for stateless applications – whose core components include a SQL database with just one container and persistent memory.

One approach could be to migrate the old backup technologies to the containers at the same time, but it's not that simple. Classic client-server backups do not work, precisely because containers do not use a complete operating system image with an init system nor their own filesystem tools to run services in the background and monitor file changes. If users were to convert their containers, they could stay with monolithic services on virtual machines and save themselves all the work that Kubernetes involves. Admins have to come up with different approaches to securing the persistent data of their Kubernetes applications.

Backing Up PVs

To back up data from a persistent volume in Kubernetes, you should understand how a PV works. Because the container has no filesystem tools,

it cannot access external storage resources. Instead, the host on which the container is running integrates the storage with the container filesystem by means of a bind mount, which is basically the same principle as when you use

```
--volume /host/dir:/container/dir
```

to display a subdirectory of the local filesystem in the Docker or Podman container. In short, the container does not know anything about the external volumes it is working with; they are simply part of the filesystem.

In principle, Kubernetes understands two types of PVs: block and filesystem. Block PVs can be network drives that use the Internet Small Computer Systems Interface (iSCSI), RADOS Block Device (RBD), or Fibre Channel standard. In simple single-node setups, local logical volume management (LVM) volumes are also fine. In the case of block PVs, the Kubernetes node creates a filesystem such as XFS on the drive and then maps it into the container. Alternatively, filesystem PVs can be used that rely, for example, on NFS, GlusterFS, or CephFS. PVs based on a network filesystem are theoretically easier to manage than block PVs.

However, network filesystems always work without a write cache and are therefore slower than block PVs when it comes to writing I/O, making them less suitable for operation with a database. Block volumes in turn benefit from the fact that the Kubernetes host, which mounts the volume and manages the local filesystem, can cache writes. Although good for performance, it is bad for backups.

Where exactly the volumes for the PVs come from is organized by the Kubernetes storage classes and drivers that are available from various hardware and software manufacturers to match their storage systems. Storage drivers have major functional differences. TopoLVM, for example, offers only a few functions and is limited to single-node setups. Rook, on the other hand, manages Ceph storage clusters with RBD devices or

CephFS and handles functions such as snapshots.

Before a pod on Kubernetes can access one or more PVs, it first needs to order the PVs by issuing a persistent volume claim (PVC). The PVC specifies the PV's parameters and access modes. The most common class comprises volumes of the read-write-once (ReadWriteOnce) access type for volumes that are exclusively available to only one container. These volumes are generally used for databases. Read-only-many (ReadOnlyMany) volumes can be accessed by several pods at the same time, but only for read access. These are particularly popular with scale-out web servers and deliver the static HTML content. The third mode is read-write-many (ReadWriteMany) and allows multiple containers to read from and write to the volume. This mode obviously involves certain risks and is not supported by all storage drivers. As a rule, filesystem PVs are capable of this mode.

Stop First, Back Up

The myth of the simple and secure storage snapshot as a backup is still making the rounds on forums and manuals. The story goes that users simply need to take an LVM or filesystem snapshot of their data partition to have a reliable backup. That is just as wrong today as it was 20 years ago. A running database server – whether MySQL, Microsoft SQL, PostgreSQL, or even SQLite – always has files open for write access. Snapshots of filesystems with open files are potentially unusable, depending on the state of the file during the snapshot. For virtual machines with databases (DBs), a reliable snapshot backup is performed as follows: A script stops the DB service, which writes all the data from the cache (the DB) to the tablespace. The script first synchronizes the filesystem cache before triggering the snapshot. The process needs to be similar with Kubernetes. The user stops the DB pod (and deletes it in the process) and then assigns the snapshot of the PV via

the storage driver before creating a new DB pod with a connection to the appropriate PV. A restore would be very simple: Upgrade the last working snapshot to a fully fledged PV and start a new DB pod with this PV. However, this operation is only possible if the PV driver is capable of taking snapshots and if backing up in this way makes sense.

Maintaining Containers

Snapshots have their limitations. If you use several Kubernetes clusters with different storage back ends, a PV snapshot will not work as a backup target. Of course, a portable backup has other options: maintenance containers, which come with tools for writing the PV data to a backup, that can use a second PV acting as a backup target. This process can be simple, such as:

```
rsync /<source> /<target>
```

or

```
tar --czf <backup-file>.tgz /volume/*
```

A full dump with tools such as `mysqldump` is also recommended for DBs.

The backup PV for the maintenance container can be a read-write-many filesystem volume, which is accessed by the external backup software after the dump and the backup data have been processed downstream.

The procedure here is just as simple. The user first terminates the DB pod, starts the maintenance pod with source and target PV, and runs the backup tool in it. The maintenance pod then terminates and a new DB pod takes over. The catch with this solution is that the longer the backup takes, the longer the application is down. The maintenance container also needs to have matching restore tools in place so that it can also write the backed-up data to a fresh PV in the correct format after a failure involving data corruption. In the case of application-specific backup tools, the user also needs to pay attention to the versions. The

tools must match the version of the software used in the application container.

Continuous Backup

Continuous backup is another potential approach that involves an application or DB server reversibly forwarding each transaction to a backup instance or a transaction log. However, the application also needs to support this form of data replication. Fortunately, all classic SQL DB servers can handle a mirror mode. The active DB server sends all changes that need to be written directly to a second instance by way of its own SQL protocol.

The user can then start additional tools, such as a DB dump on the backup node, without negatively affecting the function and performance of the primary DB server. If the backup server does not react immediately, the source server caches the incoming files and delivers them to the backup instance after a delay. Continuous backup allows every single transaction in a DB to be tracked, but it also makes the restore more difficult. In case of data loss in the active DB, for example, where the DB application deletes the data undesirably because of an error or an attack that encrypts or overwrites data, the user needs to go through the transaction log step by step and find the last “good” transaction. The additional overhead also means that the user can only retroactively determine the point in time up to which the data states need to be restored. In DBs with frequent write and change access, a transaction log will also occupy a large amount of memory, which is why a combination of point-in-time full backups and the transaction log between the status of the current DB and the last full backup is recommended.

Tools and Configurations

In many cases, you do not need to be overly creative when configuring a continuous backup scenario. A number of ready-made Kubernetes

mirrored DBs are ready to roll out. Crunchy Data, for example, offers a very powerful Kubernetes operator for PostgreSQL DBs. It can set up mirrored clusters with just a few commands and offers ready-made pods for backing up and restoring PostgreSQL clusters. The open source version of the Crunchy operator ([Figure 1](#)) is available free of charge on GitHub [2], and the manufacturer offers commercial tools with support for production environments.

Kubernetes supports various methods for rolling out applications with a connection to a PV. The kind (i.e., the type of a Kubernetes API function call) is the `StatefulSet`. It describes a template for a persistent volume claim along with the application pods. When a user starts `StatefulSet`, it dynamically generates the PVC of the application. Therefore, if the user deletes the stateful set, they also delete the PVC and, where applicable, the associated PV. However, for backup setups such as the method described with the maintenance container, you would want to delete the application rollout without jeopardizing the PV with the data or the associated claims.

For such scenarios, it makes more sense to roll out with separate definition files. The kind in this case is `Deployment`. Like `StatefulSet`, it describes the application pods, but without the embedded PVC template. Instead, it references a PVC that you create separately from the deployment. An existing definition of a stateful set can easily be rewritten as a deployment by removing the `volumeClaimTemplate` block and instead referencing the PVC described in another file in the `volumeMounts` of the pods. You then need to declare two deployments – one for the application itself and another for the maintenance container – and then reference the separately created PVCs in both.

The complete backup process is as follows: Delete the DB pod, start the maintenance pod, then roll out the DB pod again. Of course, you can automate this process with a tool such as Ansible. Unfortunately, a lot of bad solutions are making the rounds online that use the shell with `shell: kubectl <something>` to try to control Kubernetes resources and query their status. It makes more sense to use modules from the `kubernetes.core` collection and avoid error-prone shell calls.

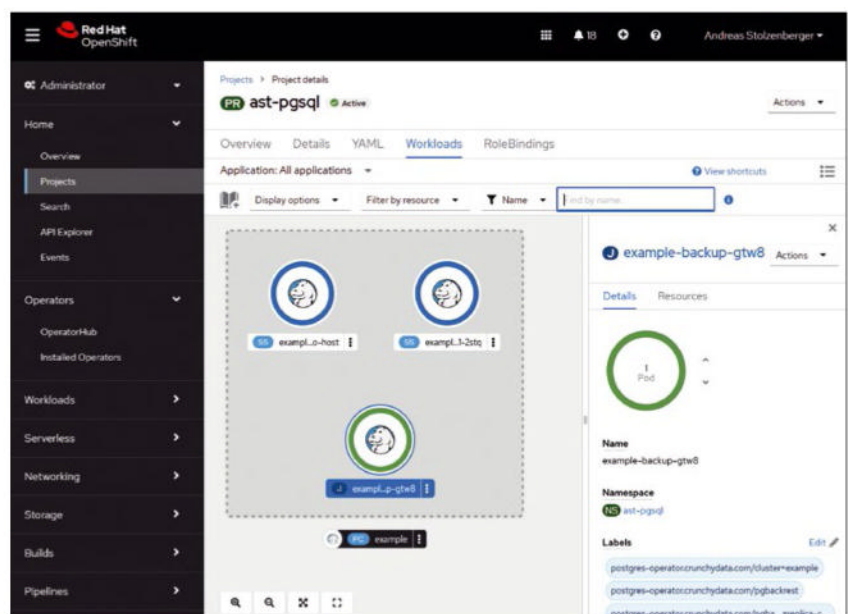


Figure 1: Crunchy’s Kubernetes operator automates the process of rolling out PostgreSQL clusters. As an additional function, Crunchy can create backups by the operator of running clusters.

You can control Kubernetes resources with the Ansible `kubernetes.core.k8s` module and enter the `kind` definition for Kubernetes directly into the Ansible code as `definition:`, but with the use of Ansible variables. Alternatively, you can convert your existing Kubernetes YML declarations to Jinja2 templates and integrate them into the Ansible code with:

```
definition: "{{ lookup(
    'template',
    'kubernetes_deployment.j2') }}"
```

In the same way, `state: absent` removes a deployment. Information on running pods, deployments, or values of a config map can be retrieved with the `kubernetes.core.k8s_info` module and be bundled into an Ansible variable by calling `register`. In combination with the `until` loop instruction, you can generate a waiting loop in your backup playbook

that waits for the maintenance pod to be completed after it has been started. The pattern is: `register pod state, until status = "Completed"`. The complete backup playbook first deletes the DB deployment, starts the backup pod, waits until it has done its work, and then restarts the DB deployment.

Conclusions

A reliable backup for Kubernetes requires some preparation, which is no surprise to Kubernetes users, because migrating existing applications from virtual machines to pods controlled by Kubernetes already involves a great amount of work. If you want to be independent of storage backups, your best bet is to use a maintenance pod to back up your data. The continuous variant also has its appeal but involves additional overhead. Of course, there is no such

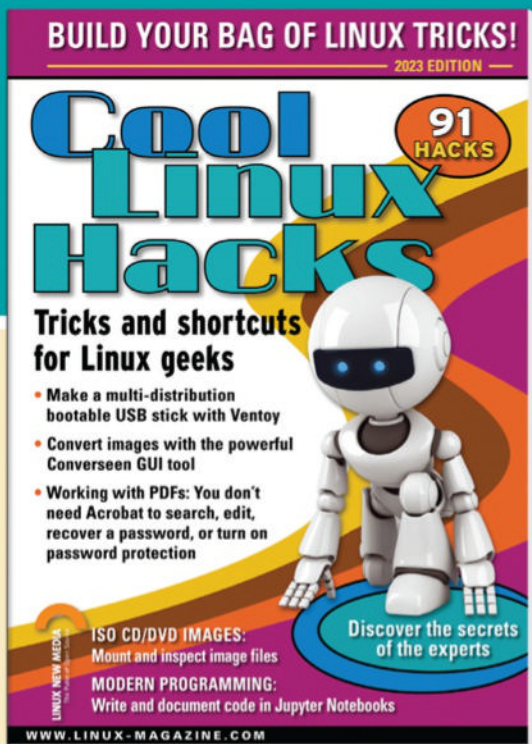
thing as a reliable backup at the push of a snapshot button without additional work. ■

Info

- [1] "Kubernetes StatefulSet" by Andreas Stolzenberger, *ADMIN*, issue 73, 2023, pg. 18, [<https://www.admin-magazine.com/Archive/2023/73/Kubernetes-StatefulSet>]
- [2] Postgres operator: [<https://github.com/CrunchyData/postgres-operator>]

The Author

Andreas Stolzenberger worked as an IT magazine editor for 17 years. He was the deputy editor in chief of the German *Network Computing Magazine* from 2000 to 2010. After that, he worked as a solution engineer at Dell and VMware. In 2012 Andreas moved to Red Hat. There, he currently works as principal solution architect in the Technical Partner Development department.



SHOP THE SHOP

shop.linuxnewmedia.com

GET PRODUCTIVE WITH COOL LINUX HACKS

Improve your Linux skills with this cool collection of inspirational tricks and shortcuts for Linux geeks.

- Google on the Command Line
- OpenSnitch Application Firewall
- Parse the systemd journal
- Control Git with lazygit
- Run Old DOS Games with DOSBox
- And more!



ORDER ONLINE:
shop.linuxnewmedia.com



Azure Arc agent on local machines

Cloud Contact

Manage your on-premises servers with Windows Admin Center in Azure.

By Thomas Joos

Installing the Azure Arc agent on local machines lets you integrate these machines with Azure and manage them remotely. Neither a virtual private network (VPN) nor port sharing in the firewall are required for this kind of access, and this approach also supports remote PowerShell sessions and remote desktop protocol (RDP) connections. In this article, I look into how you can manage servers from wherever you are with Azure Arc. The ability to leverage Azure cloud resources locally from the Windows Admin Center (WAC) has been around for quite a while now. It is particularly useful for backing up local data to the cloud with Azure Backup. Azure Arc takes the opposite approach and lets you manage your on-premises servers with WAC in Azure. If you connect WAC to Azure, you can use it to manage Azure virtual machines (VMs) and synchronize VMs from Hyper-V with the cloud as Azure VMs for failover purposes. The most interesting functions in Azure for supporting local networks are Azure Backup, Azure Monitor, Security Center, Site Recovery, and much more, but local networks also benefit from these Azure services. For small and medium-sized enterprises (SMEs), the advantage is that you can use Azure functions without having to run your own servers, and the

services can be managed very easily from the Windows Admin Center.

Connecting to Azure

To connect the Windows Admin Center to Azure, you only need to set up the corresponding configuration on the computer with the Admin Center gateway. Once the connection has been established, when you connect to the gateway from a web browser, you also have access to the Azure functions if you are authorized to do so. To set it up, click the gear icon in the Windows Admin Center to call up the settings. When managing the gateways, you will find a wizard in Azure, which you can use to log in. The various Azure services are then available on the network. The Windows Admin Center gives you everything you need for these services. After the install, you can then connect the WAC gateway to Azure. Doing so lets you use many Azure services in your local data center (some of which are free of charge) and connect your servers to Azure through Azure Arc, which means that you can manage the connected servers in WAC on the Azure portal in the same way as you would with the local Windows Admin Center. As mentioned previously, you do not need a VPN and do not have to drill holes

in your firewall. All you need for the connection is an agent for Azure Arc (more on this later) on the desired local servers.

Registering WAC with Azure

To register your WAC gateway with the respective Azure subscription after installation, in WAC, click *Azure hybrid center* and choose the *Register your Windows Admin Center gateway* link. WAC displays a window where you first need to log in to Microsoft Azure; WAC then connects to your subscription in a process that only takes a few seconds. You then have to create a local account for the server under *Account* in the WAC settings (which are available from the gear icon at the top right) and one in Azure at the same time. Once Azure and WAC are linked, the services for which Azure resources can be used in the local data center are made available in the Azure hybrid center when you connect to a server in WAC. You can see the services that are already in use in the *Installed services* menu. *Azure Arc* lets you connect to local servers with an agent to Azure in WAC in just a few steps. After a short wait, the local server becomes available in the respective resource group on the Azure portal and you can manage, secure, monitor, and customize the server. At the same time, you can enable the Windows Admin Center for local resources on the Azure portal, which

Lead Image © Rungaroon TAVEEAPRADEEMUNKONG, 123RF.com

allows authorized admins to access WAC in the cloud and use the Azure Arc agent to manage local machines. Basically, the same access options exist here as for management in the local data center. With WAC now available on the Azure portal, access can of course be appropriately secured with authorizations from the Azure Active Directory. After registration, the Azure hybrid center also lets you integrate Azure services in the local data center, such as Azure Backup for backing up local data in the cloud.

Managing Servers Remotely

To establish the relationship between your local computers and Azure Arc, you first need to register WAC with an Azure subscription, as described above, then join your server in the Azure hybrid center from *Set up* under *Set up Azure Arc* – this is also possible with a free Azure subscription. Now select the connected Azure subscription, click the *Add new* option in *Resource group*, and enter the name of the server (e.g., *srv1*). Choose an Azure region (e.g., *West Europe*), then click *Set up*. The server is then visible in the Servers pane in Azure Arc on the Azure portal (**Figure 1**). During this process, Windows Admin Center installs an agent on your server to help you connect to Azure. You can view the current processes

top right by pressing the bell icon. On the server, you can now enter

```
appwiz.cpl
```

to check whether the Azure Connected Machine agent is installed on the server. The local machines can now be managed with Azure policies, inventories are supported, and you can monitor both performance and security. Automatic management processes are now available, as is the ability to distribute updates from the Azure Update management center. Change tracking is another useful feature that can be used to track configuration changes, and you can use extensions to intensify your collaboration with Azure. For example, locally operated SQL servers can be tied in with Azure or comprehensive monitoring scenarios can be implemented. Additionally, the Azure portal now offers remote maintenance for local servers (e.g., over SSH) by opening the *portal.azure.com* URL and clicking on the *All resources* link. After starting the desired server, you will see various tiles in the lower area that let you perform administrative tasks over the Internet.

Authorizing Remote Administration

So that you can log on to the server locally in WAC, you first need to select the user accounts in the cloud that you

want to authorize. On the Azure portal, click on the resource group with your server and then on *Access control (IAM)* on the left. Here you define the authorizations that your account or other users will have for the server. Next, click the *Add role assignment* button in the *Grant access to this resource* section. Select *Windows Admin Center Administrator Login* and click *Next*. On the next page, select *Members* to specify the user accounts you want to authorize. Click on the *Check and assign* button to complete the process. The *Access control (IAM) | Check access | View my access* button will show the matching authorizations in the *Role assignments* window.

Once you and the added users have been assigned permissions, you and they can access local servers from the server's resource group in the left pane with the *Windows Admin Center | Connect* button. After a short time, a Windows Admin Center login window appears on the Azure portal. You need to authenticate with the administrator account of the local server, not with your Azure login credentials. At this point, you can also use Active Directory logins with the syntax `<server or domain> \ <username>`.

Conclusions

In combination with Azure Arc, the cloud-based Windows Admin Center can be used for establishing a secure connection to the local data center. The setup is quick and easy and even possible with the free version of Azure. Small businesses will benefit from simple administration and support for more security, and larger corporations can use the change management feature and Azure Monitor to monitor and control cloud resources just as effectively as servers in the local data center. ■

The Author

Thomas Joos is a freelance IT consultant and has been working in IT for more than 20 years. In addition, he writes hands-on books and papers on Windows and other Microsoft topics. Online you can meet him on [<http://thomasjoos.spaces.live.com>].

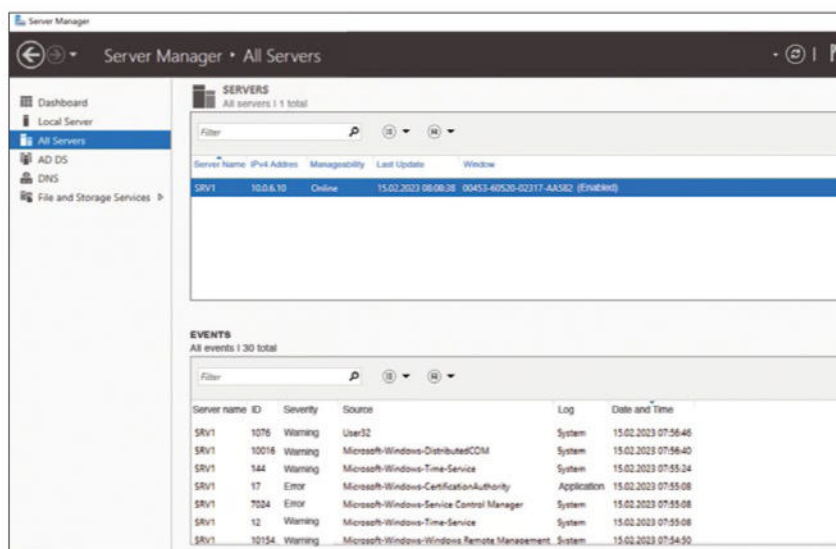


Figure 1: The local server is visible in Azure after successful integration in WAC.

Hone Your Skills – with – Special Issues!

Get to know Shell, LibreOffice, Linux, and more from our Special Issues library.

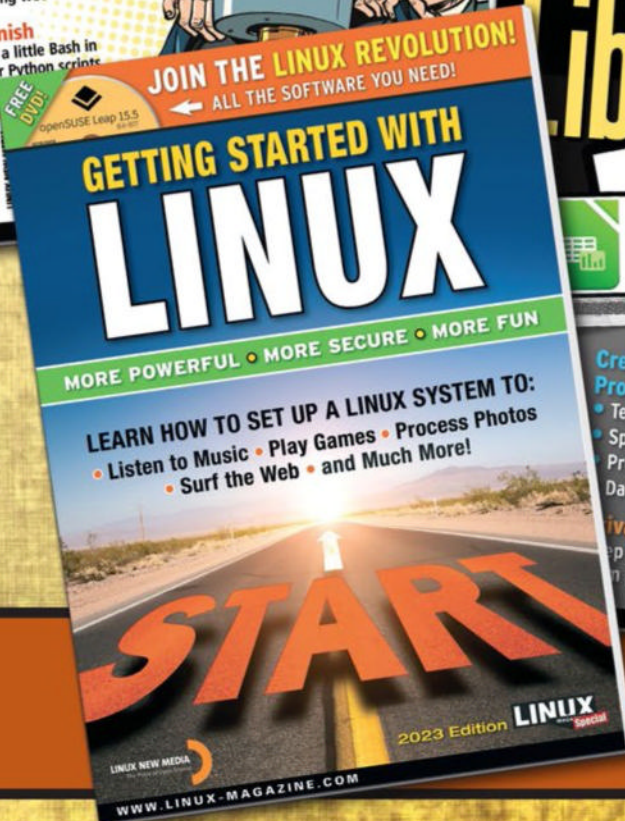
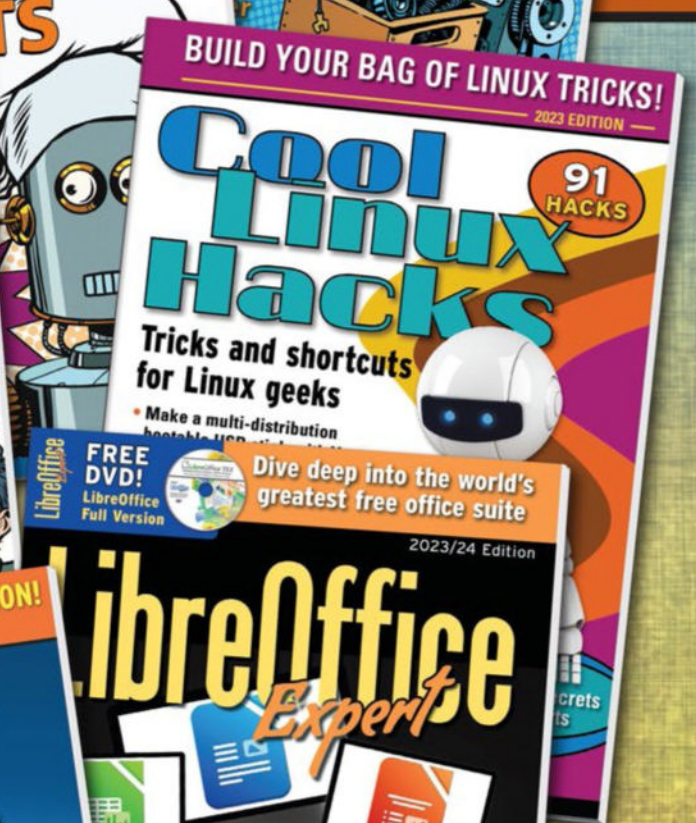
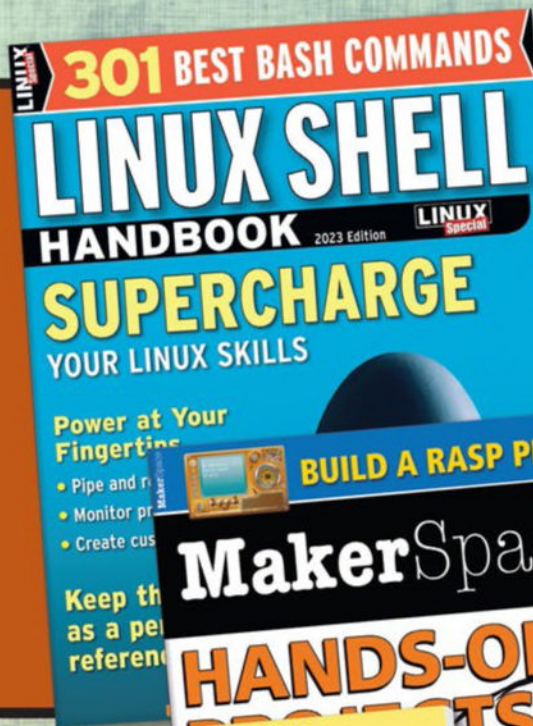
The *Linux Magazine* team has created a series of single volumes that give you a deep-dive into the topics you want.

Available in print or digital format

Check out the full library!

shop.linuxnewmedia.com







Intrusion Detection with OSSEC

Guardian Angel

The OSSEC free intrusion detection and host-based intrusion prevention system detects and fixes security problems in real time at the operating system level with functions such as log analysis, file integrity checks, Windows registry monitoring, and rootkit detection. It can be deployed virtually anywhere and supports the Linux, Windows, and macOS platforms. By Thomas Joos

As a host-based intrusion detection system (HIDS), OSSEC [1] detects and reacts to security incidents in real time. The software is capable of detecting a wide range of security incidents, including attacks on file-systems and directories, changes to system files and configuration files, failed login attempts, and attempts to escalate privileges. The tool also detects changes to logfiles and network attacks such as port scans, connection breaches, and distributed denial-of-service (DDoS) attacks.

In this article, I show you how to set up the server and the clients. You can also set up OSSEC as a Docker container. All packages are available directly from the download page [2].

Taking Countermeasures

OSSEC offers a range of countermeasures to help you respond to security incidents, such as blocking IP addresses or hosts that exhibit suspicious behavior and terminating processes that are unauthorized or attempting attacks. Additionally, the

application lets you disable user accounts that are misused for attacks and supports alerting to ensure a rapid response to incidents. The free software basically focuses on monitoring systems and networks.

In this article, I look at why the use of OSSEC is a sensible step toward significantly enhancing security on networks. Ultimately, OSSEC helps you detect security incidents before virus scanners or other systems, which is particularly important in ransomware attacks, for example, because time is a critical factor.

OSSEC Editions

The basic version of OSSEC is open source and offers you a rich feature set with log-based intrusion detection, rootkit and malware detection, active response, compliance auditing, file integrity monitoring, and system inventory. It is important to note that OSSEC as a HIDS focuses on monitoring individual systems. Therefore, you should use OSSEC in combination with other security tools such

as network-based intrusion detection systems (NIDS) or firewalls to create a comprehensive security system. Other editions are also available. For example, OSSEC+ offers additional functions, such as machine learning, but requires registration with the manufacturer before use. This edition also integrates the Elasticsearch, Logstash, Kibana (ELK) stack. Central administration and thousands of rules, as well as role-based authorizations and a comprehensive reporting system, are restricted to the scope of the commercial-grade Atomic OSSEC version. The differences between the various editions are listed on the download page.

Typical Deployment Scenarios

One practical use of OSSEC is system and application log analysis to detect signs of security breaches or suspicious activity. If the system identifies this kind of activity, OSSEC notifies you by email, Slack, or another configured notification method. At the

Photo by Rayner Simpson on Unsplash

same time, the software can carry out actions in its active response system, such as blocking users or IP addresses. File integrity checking is another use case. OSSEC monitors important system files and directories for changes. If something unexpected occurs, you are notified and can actively combat attackers and malware before the damage spreads, making this a valuable tool, especially in the fight against ransomware.

For Windows users, OSSEC also supports monitoring the Windows registry. Changes in the registry can indicate a security breach or an unwanted application. OSSEC tracks these changes and alerts you to suspicious activity.

Rootkit detection is another important feature of OSSEC. Rootkits are malicious programs that try to hide deep in the operating system to remain undetected. The tool searches for known signatures and behavior patterns to identify rootkits and report their presence to you.

Finally, OSSEC has an active response functionality that reacts automatically to detected threats. For example, you can configure OSSEC to block network access for an IP address from which repeated failed login attempts have been made.

OSSEC on Various Platforms

OSSEC works across all platforms and can be installed on Linux, Windows, and macOS, which makes the environment a useful choice as an additional security tool on hybrid networks. Linux distributions such as Debian, Ubuntu, CentOS, Red Hat Enterprise Linux, and Fedora all support OSSEC. You can use the package manager for the install or compile from the source code.

On Windows, you must make sure that the software supports your version and that all the required updates are in place before downloading and installing the OSSEC agent for Windows from the official website. You cannot install the OSSEC server itself on Windows unless you use the Windows Subsystem for Linux

(WSL); obviously, this is not ideal. For macOS, the OSSEC install works in a similar way to Linux. You can download the source code from the official website and compile the program or use a package manager such as Homebrew.

When installing OSSEC, you need to coordinate the components, which include a central server, the OSSEC manager, and the OSSEC agents on the systems to be monitored. The manager collects and analyzes data from the agents and runs appropriate actions when the tool detects threats or security breaches. Make sure you choose the right configuration for your requirements and set up the communication between manager and agent correctly. Adjustments are made in the configuration file.

The OSSEC server can also work without agents, but then it does not read all the information. Agents can also be deployed on Windows servers to monitor the registry for changes. The server can also read data from system logfiles and aggregates all this information to carry out actions or deliver notifications in the event of trouble.

OSSEC can be used in the cloud, as well. In cloud environments, you can install the application on both virtual machines and cloud-based servers. Parallel bookable services are available directly in the cloud for Microsoft Azure, Amazon Web Services (AWS), and Google Cloud Platform (GCP). You need to install and configure the OSSEC manager on a central instance or a dedicated server then set up the agents on the individual virtual machines or cloud servers that you want to monitor. Here, too, the agents communicate with the central manager and send information about the system activities, which the manager analyzes before triggering appropriate actions.

If you use OSSEC in the cloud, it is important to safeguard the communication path between the agents and the manager. You need to use protected connections such as encrypted virtual private network (VPN) tunnels or private network connections to

keep the communication between the systems private.

In many cases, cloud providers also offer their own security products and monitoring tools. It is advisable to use these in combination with OSSEC to maximize the security of your cloud infrastructure.

Installation and Setup

You can either install OSSEC or download it as a virtual appliance; the former option is preferable. Servers and agents can be downloaded separately and connected to an OSSEC server. In contrast to many live DVDs or simpler open source tools, OSSEC is not easy to install, set up, and operate. You need to take time to familiarize yourself fully with the product. As a general rule, it makes sense to reference the documentation [3] when you are setting up and managing the system.

In the following discussion, I concentrate on the free OSSEC version 3.7.0 install on Ubuntu 22.04. After the mandatory Linux update, install the required dependencies:

```
sudo apt-get update
sudo apt-get -y upgrade
sudo apt-get install -y build-essential
sudo apt-get install -y zlib1g-dev 2
libpcrc2-dev libevent-dev libssl-dev 2
libsystemd-dev jq
```

The certificates and key files are then downloaded and installed and the repository is integrated:

```
wget http://www.ossec.net/files/2
OSSEC-ARCHIVE-KEY.asc
wget https://github.com/ossec/ossec-hids/2
releases/download/3.7.0/2
ossec-hids-3.7.0.tar.gz.asc
gpg --import OSSEC-ARCHIVE-KEY.asc
```

Next, download the current OSSEC version, and unzip and execute the installation script:

```
wget https://github.com/ossec/2
ossec-hids/archive/3.7.0.tar.gz
gpg --verify ossec-hids-3.7.0.tar.gz.asc 2
3.7.0.tar.gz
```

```
tar -zxvf 3.7.0.tar.gz &&
cd ossec-hids-3.7.0/
wget https://github.com/PCRE2Project/
pcr2/releases/download/pcr2-10.40/
pcr2-10.40.tar.gz
tar -zxvf pcr2-10.40.tar.gz
-C src/external/
```

No errors should appear in the build process at this point. After starting the installation script, the next step is to select the language in which you want to install OSSEC. English is preselected and I do not recommend changing this setting, because some of the translations were inaccurate when I tried them out in the lab.

Adapting the Environment

A wizard helps you customize the installation (Figure 1). The *server* selection installs the first OSSEC server on the network; you can connect more agents running on Linux, Windows, or macOS to the server later on. Next, select the installation directory; the default is `/var/ossec`.

You can also enable email notifications now by entering an email address. In many cases, the system finds the SMTP server, but you can change the configuration at any time. You will always want to install the integrity check daemon, which is also preselected, as is the rootkit detection engine.

I also recommend adding active response as part of the setup. This function lets the system run predefined commands on the basis of incoming events. For example, you can block an IP address or block access for a specific user. In the next step, it makes sense to activate the firewall drop response, which allows OSSEC to block the host in iptables (Linux) or ipfilter

(Solaris, FreeBSD, or NetBSD) if it detects an attacker. This step helps fend off brute force scans, port scans, and some other forms of attack.

You then have the option of adding IP addresses to allowlists if you do not want OSSEC to check and block communication from some IP addresses. You will also want to enable remote syslog over UDP port 514 so that the system can send logfiles to syslog servers. Once you have made your choices, the wizard installs OSSEC accordingly. At the end of the installation you will see a *Configuration finished properly* message.

Configuring the OSSEC Server

The next step is to tailor OSSEC to your requirements in the `/var/ossec/etc/ossec.conf` configuration file, for example, in Nano:

```
sudo nano /var/ossec/etc/ossec.conf
```

You can customize the email configuration and, as shown in Listing 1, integrate the IP addresses of the services and clients in the `<global>` section.

To read the syslogs from the various

OSSEC agents, the client IP addresses need to be added to the configuration file under `<remote>`, and the connection must be defined as `secure`:

```
<remote>
<connection><secure></connection>
  <allowed-ips>192.168.0.2</allowed-ips>
  <!-- OSSEC client -->
</remote>
```

If OSSEC detects an attack originating from an IP address, the system blocks it for 10 minutes. If further suspicious packets then originate from the IP address, OSSEC identifies them as repeat offenders and blocks them for a longer period. This can be defined in the `<active-response>` section of the same configuration file:

```
<!-- Active Response Config -->
<active-response>
  <repeated_offenders>30,60,120,240,480
  </repeated_offenders>
</active-response>
```

This example blocks potential attackers for a longer period on each new attempt. If you make changes to the configuration file, you need to restart OSSEC.

Listing 1: Store IP Addresses

```
<global>
  <allow_list>127.0.0.1</allow_list>
  <allow_list>::1</allow_list>
  <allow_list>localhost.localdomain</allow_list>
  <allow_list>127.0.0.53</allow_list>
  <allow_list>10.0.0.2</allow_list>
  <!-- OSSEC client -->
</global>
```

```
OSSEC HIDS v3.7.0 Installation Script - http://www.ossec.net

You are about to start the installation process of the OSSEC HIDS.
You must have a C compiler pre-installed in your system.

- System: Linux ubuntu15 5.19.0-38-generic
- User: root
- Host: ubuntu15

-- Press ENTER to continue or Ctrl-C to abort. --

1- What kind of installation do you want (server, agent, local, hybrid or help)? server
  - Server installation chosen.

2- Setting up the installation environment.
  - Choose where to install the OSSEC HIDS [/var/ossec]:
    - Installation will be made at /var/ossec .

3- Configuring the OSSEC HIDS.

3.1- Do you want e-mail notification? (y/n) [y]: y
  - What's your e-mail address? thomas.joos@web.de
  - We found your SMTP server as: mx-ha03.web.de.
  - Do you want to use it? (y/n) [y]: y
    --- Using SMTP server: mx-ha03.web.de.

3.2- Do you want to run the integrity check daemon? (y/n) [y]: y
  - Running syscheck (integrity check daemon).

3.3- Do you want to run the rootkit detection engine? (y/n) [y]:
```

Figure 1: The basic setup of OSSEC is handled by a text-based wizard.

Connecting Agents

The best way for OSSEC to collect information from computers on the network is to install agents. A client for Windows is available from the download page, which you can install from the graphical user interface. Connecting Linux computers is slightly more complicated but is also quickly done.

To launch the agent on a Windows server, you first need to complete the install, which can be scripted easily. A connection to OSSEC is then opened by the OSSEC agent manager where you enter the IP address or the name of the OSSEC server and the authentication key for the connection that you create on the OSSEC server. It is important at this point to enter the IP addresses of the clients in the OSSEC configuration file on the server, as already described.

To manage the agents on the server or to create authentication keys, launch the administration program on the server by typing:

```
sudo /var/ossec/bin/manage_agents
```

Installing the Linux agent is basically the same as installing the OSSEC server, but select the *agent* installation variant and not *server*. A wizard then appears, and you can set up the agent in the same way as the server. Specify the installation directory, enter the name of the OSSEC server, and enable the integrity check daemon, the rootkit detection engine, and active response. On Linux computers, you can also configure the agent in the `/var/ossec/`

`etc/ossec.conf` file. At this point, it is important to enter the server's IP address in the `<ossec_config>` section:

```
<ossec_config>
  <client>
    <server-ip><IP-address></server-ip>
```

When managing the connected clients, you can first display a list of the connected computers and create the authentication keys that the clients require for the connection. For Windows clients, use the setup window; on Linux, start the same tool as on the server by typing:

```
sudo /var/ossec/bin/manage_agents
```

You can then enter the authentication code in the terminal by selecting *I*. It generally makes sense to reboot the server and the client after integrating clients. To do this, run the following on the devices:

```
/var/ossec/bin/ossec-control restart
```

You can see whether a Linux client has connected successfully by typing:

```
sudo /var/ossec/bin/agent_control -lc
```

If the connections between clients and servers do not work, it is usually because of the firewall settings on the server. Make sure the firewall does not block communication between the server and the clients, especially ports 1514 and 514. Additionally, secure communication with the server must be permitted in the OSSEC configuration file (**Listing 2**).

Listing 2: Secure Communication with Server

```
<global>
  <allow_list>10.0.0.2</allow_list>
  <!-- OSSEC client -- >
</global>
<remote>
  <connection>secure</connection>
  <allowed-ips>10.0.0.2</allowed-ips>
  <!-- OSSEC client -- >
</remote>
```

Conclusions

OSSEC is a powerful tool that can detect and combat malware and cyberattacks and can be run on a virtual machine – no physical hardware is required. As well as email notifications, OSSEC can run actions and use Slack for communication – the project documentation provides useful help for setting up these features.

In addition to the free OSSEC version, you might also want to try out OSSEC+. You do need to register, and in many cases OSSEC is fully up to the task in hand. ■

Info

[1] OSSEC: [<https://www.ossec.net>]

[2] OSSEC downloads: [<https://www.ossec.net/ossec-downloads/>]

[3] OSSEC documentation: [<https://ossec-docs.readthedocs.io/en/latest>]

The Author

Thomas Joos is a freelance IT consultant and has been working in IT for more than 20 years. In addition, he writes hands-on books and papers on Windows and other Microsoft topics. Online you can meet him on [<http://thomasjoos.spaces.live.com>]. ■



Recovering from a cyberattack in a hybrid environment

Disconnected

Restoring identity is an important part of disaster recovery, since it lays the foundation for restoring normality and regular operations. We look into contingency measures for hybrid directory services with Entra ID, the Graph API, and its PowerShell implementation. By Evgenij Smirnov

The complexity of modern IT landscapes becomes particularly apparent in the event of emergencies caused by cyberattacks. It is not just the recovery of the individual subsystems that needs to be considered when you restore, but also the interactions. The failure of basic services such as authentication as the result of an attack is a particularly serious worry.

Regardless of whether a company still has its IT firmly anchored locally or is already on its way into the cloud, most directory services now have a hybrid design. The legacy Active Directory (AD) is the primary identity store, and users, groups, and, increasingly, computer accounts are synchronized to Entra ID (formerly Azure AD) or other cloud-based directories to enable seamless access to applications in the cloud. The prime example is Microsoft Teams, which many

companies were forced to introduce as an emergency measure during the pandemic. However, other services such as virtual private networks (VPNs), Microsoft Dynamics 365, Salesforce, or Box also work best with a cloud identity.

Hybrid Identities on the Rise

How tightly the on-premises part of a hybrid identity is tied in to its cloud counterpart can vary. Some organizations want to provide an online identity but keep the authentication process entirely on-premises and rely on pass-through authentication (PTA) or locally installed instances of Active Directory Federation Services (ADFS). Others synchronize the password hash to the cloud (password hash synchronization, PHS) or even make

the cloud account authoritative with password write-back, enabling more extensive password checks than the complexity conditions supported in AD and providing users with a simple self-service password reset (SSPR). Thanks to Cloud Trust, even Kerberos authentication of a cloud account against local resources connected to AD is possible.

All types of hybrid identity are relatively easy to set up, and the providers supply both technical aids and very good instructions. Microsoft has a particular interest in migrating identities to its own cloud as quickly and painlessly as possible; the new privileged access strategy envisages the cloud as the “source of security” [1]. However, the successful cyberattacks of recent years all too clearly reveal the weaknesses of the strategy. On the one hand, both components – on

Photo by Kelly Sikkema on Unsplash

premises and in the cloud – are required in a hybrid identity landscape to enable smooth operation of all applications. On the other hand, the relationship between the legacy identity and the modern identity creates completely new attack vectors that exploit the peculiarities and functional weaknesses of both sides.

Many Roads Lead to Disaster

A hybrid identity system can be attacked in many ways and be completely or partially disabled. The measures you need to take to get the system operational again are just as varied.

A classic attack scenario focuses on the local part of the organization. The attacker typically takes complete control of AD, ultimately to destroy it. This positioning also opens the cloud component attack, because by taking over the traditional AD, the attacker also gains control of Azure AD Connect or Cloud Sync agents and can escalate into the cloud (**Figure 1**). However, if the attacker discovers during their investigations that the cloud services are only in

rudimentary use (e.g., video telephony in Teams), they will typically not waste time compromising these services and simply encrypt or destroy the applications and data operated onsite. Your cloud tenant may be spared to a great extent.

Now, though, you face a different challenge. Because AD is the leading identity store on the hybrid network, users and groups are available in the cloud but cannot be managed there. If you enabled PHS, users can even log in to cloud services with the last known password, but you cannot change the password if password write-back, SSPR, or both have not also been enabled. On the other hand, if you rely on PTA or ADFS, your cloud users will no longer be able to log in because the primary authentication systems are no longer available. For this reason, both Microsoft and most experts recommend enabling PHS – at least as an additional authentication procedure. The frequently cited angst prompted by synchronizing passwords with the cloud is unfounded, because hashing more than 1,000 times makes it uneconomical to reconstruct the plain text password in

a reasonable time, even if relatively weak passwords are in use.

In known attacks, hackers have reset the passwords of all synchronized users as a last resort before destroying AD and Azure AD Connect. Although the data in the cloud applications are still intact, access is severely restricted or not possible at all for the time being.

In other, more complex scenarios, the attacker first penetrates the cloud by phishing or manipulating the multifactor authentication (MFA) procedures and only then discovers that a synchronized AD account they have hijacked has far-reaching authorizations. In this way, they can obtain an identity that allows them to compromise the cloud tenant or at least steal data such as email, documents, or chat histories from cloud applications to later phish other users (CEO fraud) or threaten to damage the company's image and extort money.

While you are restoring the on-premises infrastructure, you need to make sure your cloud services no longer offer the attacker a gateway before you reconnect the two parts of the hybrid environment.

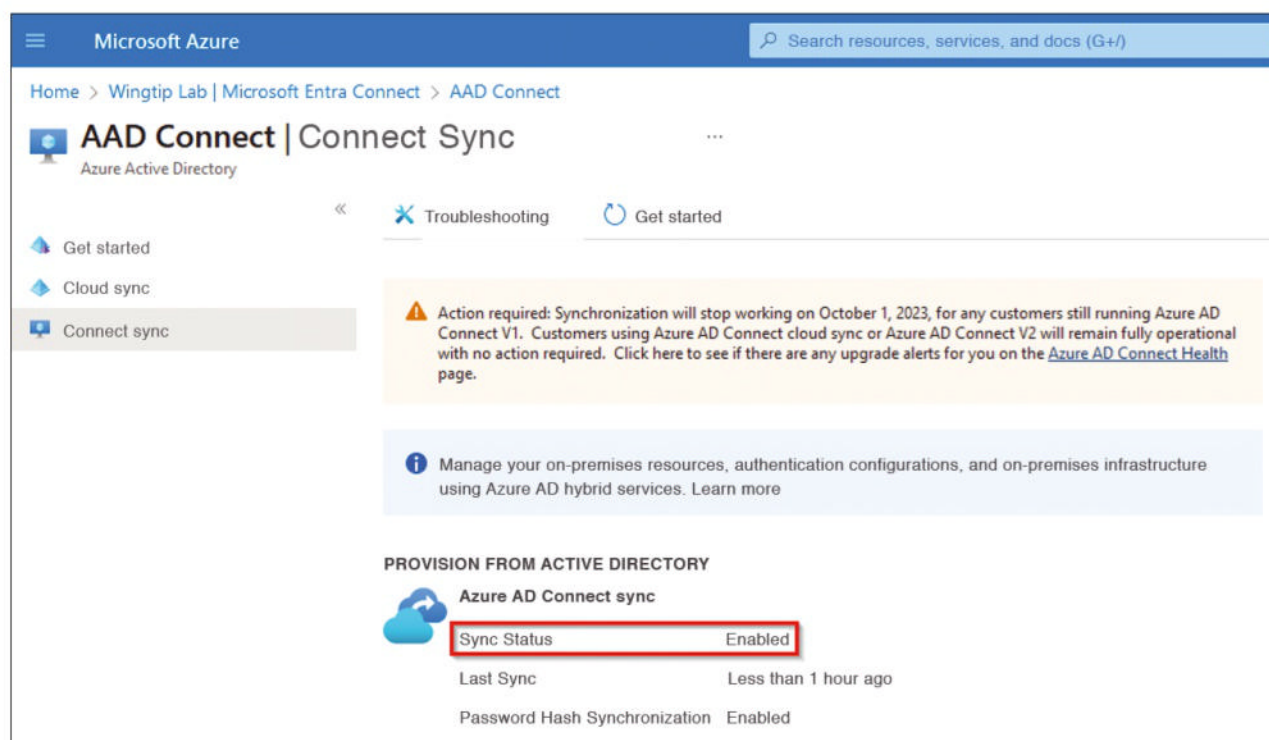


Figure 1: Azure AD Connect is considered active even though the local AD has been compromised.

Graph PowerShell Is the Future

Although the MSONline and AzureAD PowerShell modules are still available and largely supported at press time, the date for shutting down the service endpoint these modules reference has already been set. Some initial functions have even been disabled; you will find many references to this with an online search. Therefore, this article focuses on the cmdlets from the Microsoft Graph Suite [2], which will be the only PowerShell interface to Microsoft 365, and in particular to Entra ID, supported in the future. The Graph modules come with an extremely large number of cmdlets. If possible, you will want to use PowerShell 7 to work with Graph PowerShell. Even there, importing the entire Graph module, including all the dependencies, takes several minutes and bumps up the RAM utilization of the PowerShell session to somewhere between 2 and 5GB. If you already have experience with Graph, it is a good idea to load only the modules you really need. For basic Entra ID tasks, for example, you only require two submodules, and you can quickly set these up by typing:

```
Import-Module 2
Microsoft.Graph.Authentication, 2
Microsoft.Graph.Identity.Directory2
Management
```

If you rely on PowerShell 5.1 and want to load many Graph modules or even the entire suite, you always need to increase the number of functions available within the session, for example,

```
$MaximumFunctionCount = 32768
```

from the default value of 4096 to something far higher, as shown.

Recovery Methods

The scope of this article does not allow for comprehensive instructions on how to regain control of

compromised services, but the approach you use is decisive for the next steps. Restoring on-premises services after destruction or compromise has a long history. For many applications – including AD – specialist tools let you restore up to a relatively recent point in time without the risk of reactivating the malware introduced by the attacker. If this option is not available, you face a choice in most cases:

- Use a more or less fresh backup and follow up the restore (which must take place on an isolated network) by performing a comprehensive analysis and cleanup.
- Revert to a backup from a little further back, hope that it has not yet been compromised, and remake the changes that have taken place since the backup.
- Completely rebuild the directory service because all of your backups have been affected by the attack.

In all of these cases, you can be relatively sure that you have sufficient control over the rebuilt AD environment as long as it remains isolated from the Internet (i.e., it cannot establish a connection to the outside world).

Far tighter limits apply to disaster recovery of cloud services, because the operator only allows access to specific data, and in some cases this access is read only. What you are looking at in this case is restoring the hybrid identity. For example, assume you have backed up application data such as Exchange mail, SharePoint documents, Azure SQL databases, and so on by some other means and can put them back into the respective application as soon as the identity is working again.

Most disaster recovery scenarios here fall into one of the following categories. In the first scenario, the cloud identities may have been compromised but apparently not destroyed, and you still have access to a Global Administrator (GA) account that is active and still assigned the GA role. In the second case, some cloud identities have been destroyed or at least

changed such that users can no longer log in, but you still have administrative access to the client. In the third case, you no longer have access to the cloud tenant at all; initially, you don't even know how serious the damage is, but users are reporting that they can no longer log in to cloud services.

If your usual administrator accounts (including the break glass account) no longer work, you need to check your documentation for applications whose identities may have sufficient permissions to create a new administrator or reset the password of an existing administrator. If you find such an application, it may be your best chance of regaining control over your tenant. At the same time, you have probably also found the way for the attacker to hijack your client. In other words, the need for action is urgent because you need to shut down this attack vector before proceeding to restore functionality.

If all administrative access is lost, the only thing that can help is a top-priority support request to Microsoft. Be prepared to have to prove tenant ownership through documents such as invoices or payment receipts and not just, for example, the ability to generate DNS records in custom domains, although this is sufficient for registration. In other words, you depend on the support of your management or commercial department, whose ability to act is severely restricted by the attack and whose nerves are on edge. It makes sense to include at least some of these documents in hardcopy in your emergency folder.

Regaining Control

Sometimes the attacked organizations are lucky and their cloud identity survives more or less unscathed. If you are facing this kind of situation and perhaps even have an up-to-date clean backup of your local AD, you may have gotten off lightly. Remove the old synchronization account from the Entra ID directory because its password may have been

compromised. When you rebuild the Azure AD Connect synchronization, you will at least regain control over enabling your identities and their passwords. If you were originally using PTA, this will also work as soon as you set it up. Restoring ADFS, if this was your previous authentication method, is much more complex. On the other hand, in a situation in which you urgently need access to the cloud services but restoring the on-premises infrastructure takes time, you need at least to enable your IT team to control cloud identities fully. To do this, though, you first need to break the link to the on-premises services.

The Graph API and PowerShell let you switch off synchronization. Please note that at the time of writing, this is only possible through the Graph beta endpoint. Connect to the Graph API by typing:

```
Import-Module 2
Microsoft.Graph.Authentication, 2
Microsoft.Graph.Beta.Identity.2
DirectoryManagement
$OrgID = (Get-MgOrganization).Id
```

From here, either submit a REST PATCH request to the beta endpoint,

```
$uri = "https://graph.microsoft.com/beta/2
organization/$OrgID"
$body = @'
{
  "onPremisesSyncEnabled": null
}
'@
Invoke-MgGraphRequest -uri $uri 2
-Body $body -Method PATCH
```

or use the Microsoft Graph beta module, which still needs the Graph module to authenticate and open the connection:

```
$params = @{
  onPremisesSyncEnabled = $null
}
Update-MgBetaOrganization 2
-OrganizationId $OrgID 2
-BodyParameter $params
```

In both cases, according to Microsoft, it can take up to 72 hours for the synchronization status to be completely reset to *Cloud Only*. In practice, this change usually takes place within 12 hours but is by no means even close to real time. That said, you can see on the portal that the command has taken effect after just a few minutes; the status of Azure AD Connect or Cloud Sync very quickly reports that

the Sync has never run. If PHS was enabled, all user accounts retain their last passwords, even after synchronization has been deactivated, so that the users – but unfortunately also the attacker – can still log in.

One thing to consider is that the various portals and endpoints will take varying amounts of time to change the displayed status of users and groups from *Synchronized* to *Cloud Managed* (Figure 2). If your disaster recovery plan allows it, you might want to delay further actions until the status is displayed uniformly everywhere.

If you are working with ADFS, you need to change the authentication type of all federated domains from *Federated* to *Managed* (Figure 3),

```
Connect-MGGraph 2
-Scopes Domain.ReadWrite.All, 2
Directory.Access-AsUser.All
Get-MGDomain
```

and proceed as follows,

```
Update-MGDomain -DomainId <Domain> 2
-AuthenticationType Managed
```

for each domain that has an *AuthenticationType* of *Federated*.

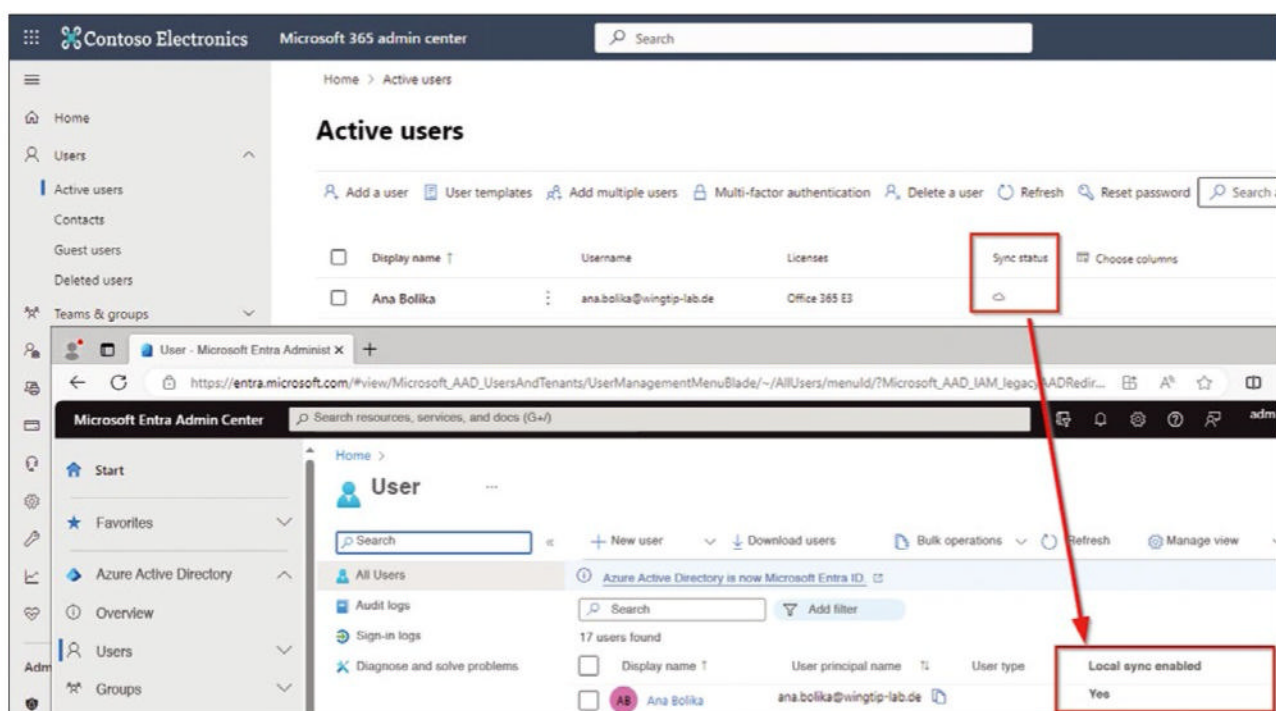


Figure 2: Breaking down the hybrid identity takes different amounts of time in different parts of the Microsoft cloud.

Reconnecting Entra ID with the Local AD

From now on, you can and must manage your Entra ID client independently of the local infrastructure. Once the AD has been restored, you need to reconnect the two directories. The procedure depends on whether you have restored the AD from a backup of the attacked AD or rebuilt it with the known account data. Before you continue, make sure that you disable the Block Hard Match Takeover security feature:

```
$uri = "https://graph.microsoft.com/beta/2
directory/onPremisesSynchronization" 2
(Invoke-MgGraphRequest -Method GET 2
-Uri $uri).Value.Features
```

Also watch out for `BlockCloudObjectTakeoverThroughHardMatchEnabled`. If the value is set to true, you must disable it with a corresponding PATCH request against the same endpoint to allow matching.

To ensure that everything works optimally, you have to know which attribute the matching was originally configured to use – lucky you if you documented your Azure AD Connect configuration. If this is not the case, you can read out the `OnPremisesImmutableID` attribute of your cloud user and try to convert it into a GUID or a string in a script:

```
$imid = (2
    Get-MgUser -UserId -Property 2
    OnPremisesImmutableID).2
OnPremisesImmutableID
$chars = 2
```

```
[System.Convert]::FromBase64String($imid)
[System.Text.Encoding]::2
UTF8.GetString($chars)
if ($chars.Count -eq 16) {
    [guid]::new($chars).Guid
}
```

If the output string can be clearly read and evaluated, you can try to guess the attribute for which it is a valid value – often user principle names (UPNs) or email addresses, but they can also be other unique attributes. If the string is unreadable, you will probably find a GUID in the second line, the `objectGUID` of the corresponding AD object. Sander Berkouwer describes matching in detail in his blog [3], and in another blog post [4] he also discusses how matching can be established in a new forest – in case you have completely lost your AD and must rebuild it.

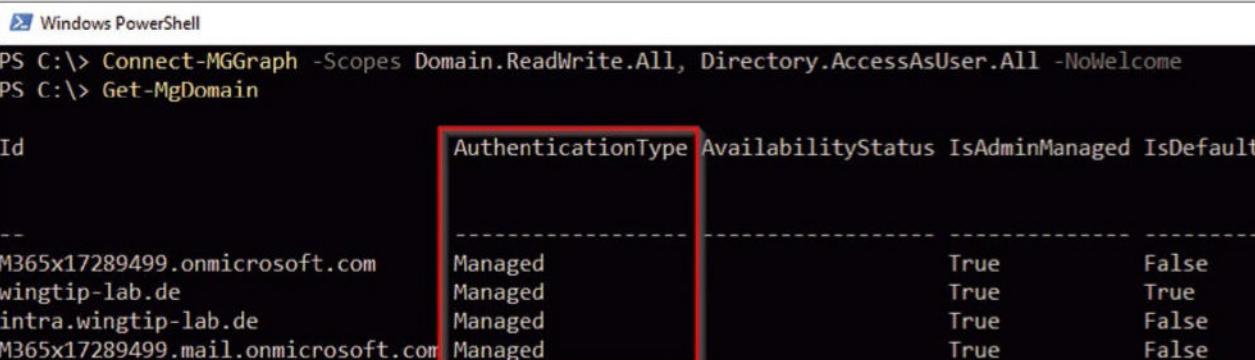
Entra ID saves some important attributes of the synchronized AD user, which subsequently makes it possible to assign the ID, and the on-premises directory can also be rebuilt true to the original. The properties all have the `OnPremises` prefix and include the security identifier (SID), the distinguished name (from which the organizational unit, OU) structure can also be derived), the `sAMAccountName`, the UPN, and even the complex attribute `OnPremisesExtensionAttributes`, which contains `ExtensionAttribute<x>` ranging from `x = 1` to `15`. On the other hand, you will see fewer local attributes for groups; the distinguished name and `ExtensionAttribute<x>` are missing here. That said, in a situation where the local directory

is irretrievably lost but the Entra ID client has survived undamaged, the cloud identity is very useful for rebuilding AD.

Finally, an important note: As soon as you set up the synchronization again and Azure AD Connect, or once Cloud Sync has completed the matching process, the local AD is once again authoritative for the passwords, activation statuses, and group memberships of your users. Let your users know this early enough so that they can change their local passwords before synchronization is activated.

Very Old Backups

Of course, it becomes more difficult if weeks – rather than days – have passed between discontinuing synchronization and reconstructing the hybrid identity, specifically because the two environments have developed independently of each other in this time. The challenges are similarly great if the backup instance that you need to fall back to in the local AD is from a relatively long time in the past – in the worst case, before the original activation of synchronization. If you are confronted with this kind of scenario, just try to keep calm and take as much time as you need for the initial comparison of the information in the two directories. You need to make sure that both the user data and the matching attributes, as well as the group memberships and the license assignments that often go with them, exactly match those in the cloud before you re-enable synchronization. You will benefit from the fact that



Id	AuthenticationType	AvailabilityStatus	IsAdminManaged	IsDefault
M365x17289499.onmicrosoft.com	Managed		True	False
wingtip-lab.de	Managed		True	True
intra.wingtip-lab.de	Managed		True	False
M365x17289499.mail.onmicrosoft.com	Managed		True	False

Figure 3: All domains have been switched to managed authentication.

Entra ID saves the many attributes described above in local objects. Do avoid taking shortcuts or making any assumptions at this point. Use PowerShell's various capabilities to synchronize as precisely as possible, create the missing objects and references, and update the changes to object metadata that have occurred since the attack on the other directory.

Conclusions

The risk of a cyberattack is part and parcel of most networked IT environments these days. The effect of an attack on a hybrid identity landscape depends, among other things, on how well you are prepared for the various scenarios. If Entra ID is an important part of your hybrid identity, it is

imperative that you familiarize yourself with the Graph API and its PowerShell implementation. It is best to run through the scenarios described in this article in a test AD linked to a test tenant.

Info

- [1] Privileged access strategy:
[<https://learn.microsoft.com/en-us/security/privileged-access-workstations/privileged-access-strategy#strategic-assumption---cloud-is-a-source-of-security>]
- [2] Microsoft Graph PowerShell:
[<https://learn.microsoft.com/en-us/powershell/microsoftgraph/?view=graph-powershell-1.0>]
- [3] "User Hard Matching and Soft Matching in Azure AD Connect" by Sander Berkouwer, March 27, 2020:
[<https://dirteam.com/sander/2020/03/27/>

explained-user-hard-matching-and-soft-matching-in-azure-ad-connect/]

- [4] "Attach a previously synced Azure AD Tenant to a new AD Forest" by Sander Berkouwer, September 17, 2020:

[<https://dirteam.com/sander/2020/09/17/howto-attach-a-previously-synced-azure-ad-tenant-to-a-new-ad-forest/>]

Author

Evgenij Smirnov has been working with computers since the age of 5 and delivering IT solutions for almost 30 years. His Active Directory and Exchange background naturally led to PowerShell, of which he's been an avid user and proponent since its first release. Evgenij is an active community lead at home in Berlin, a leading contributor to German online communities, and an experienced user group and conference speaker. He is a Microsoft Cloud and Datacenter Management MVP since 2020.

Shop the Shop

shop.linuxnewmedia.com

Missed an issue?

You're in luck.

Most back issues are still available. Order now before they're gone!

shop.linuxnewmedia.com



GET IT NOW!

SAVE TIME ON DELIVERY WITH OUR ALTERNATIVE PDF EDITIONS





Data center management with Ralph

Taking Control

The Ralph open source asset management system and configuration database keep things simple when it comes to managing data centers, but without compromising flexibility. By Holger Reibold

Often the excellent products in the open source community for operating and maintaining IT infrastructures are specialist tools that can do one thing well, but nothing else. Therefore, an admin's toolbox usually comprises a large number of tools with partly overlapping functionality. Bucking this trend is Ralph [1], a well-designed and powerful asset management, data center infrastructure management (DCIM), and configuration management database (CMDB) system for data centers and back offices. Whereas the IT asset management department is responsible for the efficient management of IT devices throughout their entire life cycles, the CMDB component, as the central database, takes care of IT service management (ITSM). These two functional areas are supplemented by the DCIM component, which provides functions for managing, monitoring, and planning the data center infrastructure and the IT systems installed there. DCIM in particular makes a significant contribution to the optimization and efficient operation of data centers because it also includes functions for planning the power supply, air conditioning, cabling, and space utilization. Thanks to Ralph, you can track your IT investments and their life cycles,

giving you a complete picture of the cash flows for these assets. Data center and back-office functions optimize your environment. Moreover, Ralph comes with a data center visualization tool that lets you map arbitrary objects. The interplay of these various functions holds a promise of efficient data center operation and a proactive approach to countering any disruptions caused by infrastructure problems.

Quick Start

At first glance, it seems that Ralph can be used to solve various problems in data center administration and optimization with a single tool, but it's not quite that simple, because the Ralph developers have imposed some restrictions on their environment for some unknown reason. For example, they only provide DEB packages for Ubuntu 18.04 Bionic on the AMD64 platform. The Docker variant is still in the experimental stage. It is still unclear when a version that is suitable for production can be expected. The experimental docker-compose configuration can be found online [2]. Before installing, make sure Python 3.6 is installed on your Ubuntu system. Ralph also requires an NGINX

web server and stores its settings in the `/etc/ralph` directory. The `debconf` prompts help you configure the database settings for a fresh install, and the `ralphctl` console command is available for Ralph management. To install Ralph, run the following commands on your Ubuntu 18.04 Bionic system:

```
curl -sL https://packagecloud.io/2
allegro/ralph/gpgkey | sudo apt-key add -
sudo sh 2
-c "echo 'deb https://packagecloud.io/2
allegro/ralph/ubuntu/ 2
bionic main' > 2
/etc/apt/sources.list.d/ralph.list"
sudo apt-get update
sudo apt-get install mysql-server 2
nginx ralph-core
```

During the installation process, you need to specify the database settings. For an initial trial, it is perfectly okay to keep the default settings; you can adjust these later in the database configuration file `/etc/ralph/conf.d/database.conf`. The next step is to extend the NGINX configuration file (the developers provide the required extension code on their website [3]) by editing the `/etc/nginx/sites-available/default` file and then restarting the web server.

Because Ralph interacts with a MySQL database, you need to generate a matching database schema and a Ralph superuser. Feeding the database with some demo data is also a good idea, so you can get to know the environment first,

```
sudo ralphctl migrate
sudo ralphctl createsuperuser
sudo ralphctl demodata
```

but make sure the conditions are as close to your actual use case as possible. Finally, you launch Ralph:

```
sudo ralphctl sitetree_resync_apps
sudo systemctl enable ralph.service
sudo systemctl start ralph.service
```

Now the asset specialist is ready for use, and you can access the web interface of the local installation at <http://localhost>. Ralph generates a number of logfiles for troubleshooting:

```
/var/log/ralph/ralph.log
/var/log/ralph/gunicorn.error.log
/var/log/ralph/gunicorn.access.log
/var/log/nginx/ralph-error.log
/var/log/nginx/ralph-access.log
```

To lower the bar in your evaluation process, the Ralph developers provide an online demo [4]. Log in with *ralph* as the username and password. Access to the demo was restricted earlier in 2023.

Initial Overview

After logging in, Ralph comes up with a clear-cut dashboard that gives you an initial overview of the number of monitored data center and back-office assets, licenses, domains, and users. The software uses a modular approach. The various functional areas for the different asset types, domains, users, and so on can be accessed from the menubar. Ralph provides full data center and back-office support, including printers, laptops, desktop

computers, and cell phones. IT asset management dashboards visualize the acquired data in real time.

To create user- and task-specific views, use the functions of the *Dashboards* menu. For a new dashboard, follow the *Add* link and assign it a Name and Description. For a new dashboard to visualize your data, you need to add and configure the data. To do so, select *Dashboards* | *Graphs* | *Add*, assign a name to the chart, and select the model (e.g., *data center*). You can also specify the Aggregation type and Chart type. The Params input field is used to specify (in JSON format) the fields that will be processed in the chart. In principle, you can also use the REST API to fetch data. To use all of the charts, run:

```
curl https://<IP address of Ralph system>/api/gr | python -m json.tool
```

You can view the details of the charts as follows:

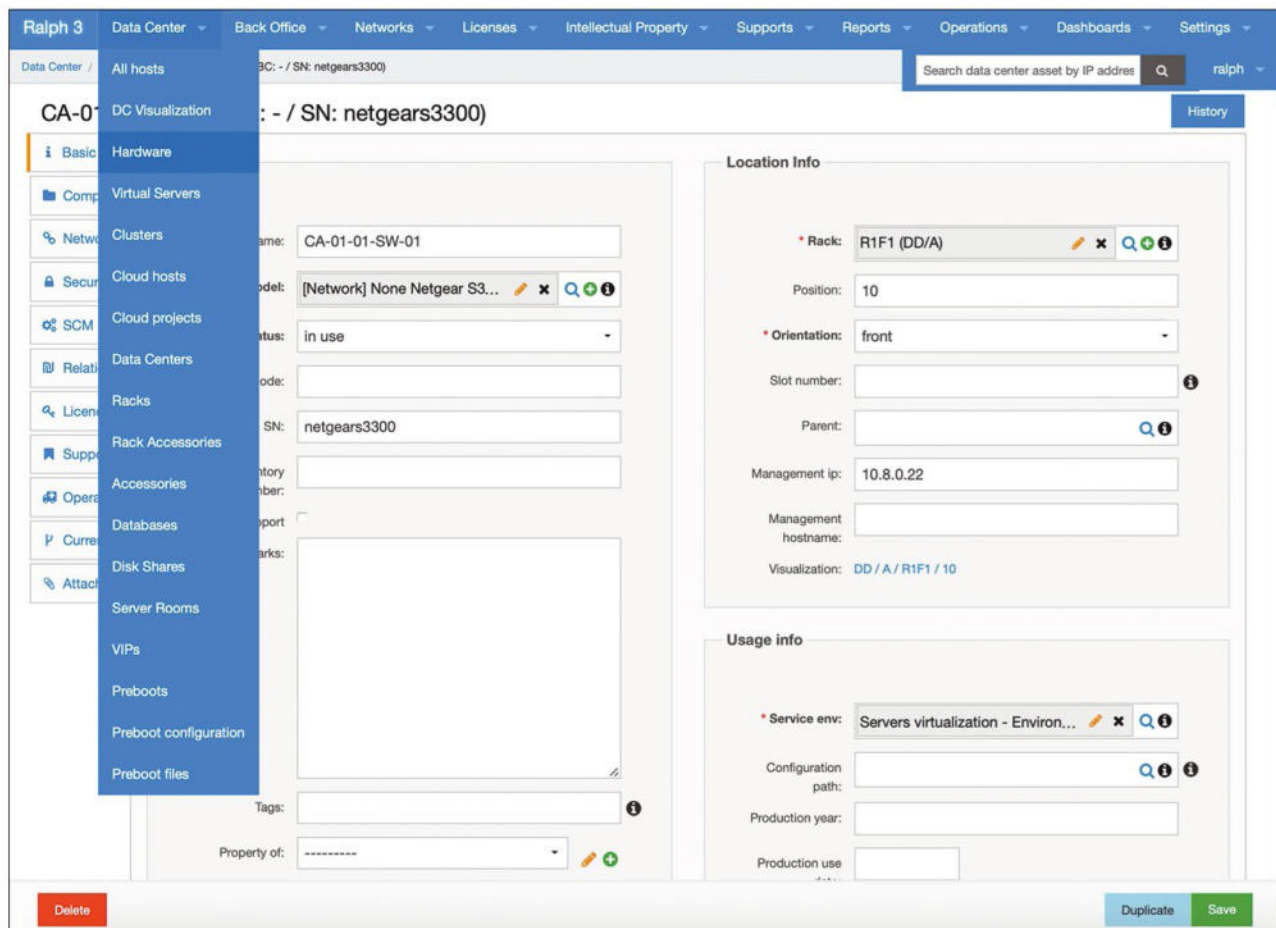


Figure 1: Use the Data Center menu to handle all data center-specific tasks, such as creating assets.

```
curl https://<IP address of Ralph system>/2
api/graph/1/ | python -m json.tool
```

The Data Center menu is used to manage data center-specific aspects (Figure 1). In particular, this is where you create the assets and the asset categories. The settings let you add various resources, including vendors, services, environments, budget information, service environments, configuration modules, configuration classes, regions, access zones, report templates, custom fields, warehouse, and office infrastructures. Users can access their profile settings, update their passwords, and edit their assets.

Managing Assets

To manage servers and create new ones, choose *Data Center | All hosts*. Ralph gives you a powerful search function in the menubar; you can use it to search the asset inventory. To add an asset, you usually use the *Data Center | Hardware* management function. Follow the *Add data center asset* link when you get there. To add a new server, you only need three

pieces of information: the model; the serial number, or alternatively a barcode; and the service environment (Figure 2). Before that, however, it makes sense to create the model variants to simplify the task of organizing devices of the same type. You can specify a model type in the dialog where you create assets by clicking on the green plus sign in the *Model* field.

Additionally, assign a hostname, status, and barcode or serial number (SN) to the new asset. When you create an asset, you can also specify its location info and usage info. Once you have created racks, you can also specify the device cabinet, position, and orientation. Entering the IP address for management access is important for server management. A key characteristic of an asset is its use, which you store in the *Usage info* section of the asset configuration. The *Service env* field is where you specify further details for an environment. The available options are *production*, *testing*, and *development*. This information is particularly important for maintenance and for installing

updates. You can also create your own usage profiles (e.g., *Load balancing*). Use the magnifying glass icon to search for existing entries or create your own.

When you generate a new service, you can add a comment or service description in the *Remarks* input box. Then, in the *Service env* field, select a supported type; you will also need to specify the environment. Optionally, you can use tags to specify a service more precisely. Ralph lists the environments available for defined services in the *Environment* line. Pressing *Save* saves the new asset configuration.

Visualizing Components

Ralph is good at visualizing the components of a data center. The functions can be found under *Data Center | DC Visualization*. For example, you can use it to create a new rack and populate it with servers. When you call up the visualization function, Ralph shows you a new rack. To customize the visualization, press *Edit*. You can drag the rack to

Figure 2: Adding a new hardware component.

a new location, *rotate* it, and label it by clicking on the pencil icon. Future plans are to expand the data center layout to include additional columns and rows.

This type of visualization simplifies rack management by adding multiple new racks to a representation in a single action. To do this, you need to switch back to edit mode; the cursor then turns into a green plus sign, and you can create any new racks you need.

In rack management, the DC inspector is another tool for checking the rack configuration. To access this tool, edit the cabinet element in the rack list, navigate to the desired system, and press *Edit asset* to open its settings. This takes you to the Asset dialog, showing basic, location, and usage information. You can now customize the type and number of asset settings in the scope of the permissions by opening *Settings* | *Permissions*.

Cloud Integration

Despite the many risks, cloud services are a popular tool that lets many companies outsource specific functional areas. Cloud services need to be viewed as part of your company's infrastructure and integrated into the management processes. Thankfully, Ralph also lets you integrate cloud services. With Ralph, you can synchronize cloud assets with providers such as OpenStack or AWS. It relies on a simple hack: Ralph provides an HTTP endpoint that retrieves messages from various cloud platforms and makes appropriate changes to cloud hosts and other objects.

However, some limits apply. Various vendors want you to set up adapters to establish the event stream with the endpoint in Ralph. According to the developers, this is still the best way to achieve near real-time synchronization of cloud assets.

With OpenStack as an example to demonstrate the procedure, you need to go to *Cloud* | *OpenStack* and select the *OpenStack* provider (Figure 3). In the configuration, enable cloud synchronization; Ralph then generates an endpoint for fielding messages from the cloud.

Next, specify the Cloud sync driver. Thus far only OpenStack is supported, and the driver is named *openstack.ocata*. You can also use the *noop* driver, which receives event messages but does not carry out any actions.

The *Client configuration* box takes input in the form of a JSON object, including the configuration for the client library used by the selected synchronization driver, such as:

```
{
  "version": "3.0",
  "auth_url": "http://10.0.0.1:35357/v3/",
  "password": "admin",
  "tenant_name": "admin",
  "username": "admin"
}
```

Pressing *Save* lets you save the OpenStack configuration; you can now track your cloud assets in the Data Center view.

Conclusions

At first glance, Ralph is an extremely promising environment for managing and optimizing data centers. In practice, however, the environment does not fully meet its potential or the claims made by its developers, probably because the developers who help project lead Marcin Kliks only work on the project part time. Regardless of its various shortcomings, Ralph still adds value to data center management. ■

Info

- [1] Ralph: [<https://ralph.allegro.tech>]
- [2] docker-compose configuration: [<https://github.com/allegro/ralph/tree/ng/contrib/>]
- [3] Extension code: [<https://ralph-ng.readthedocs.io/en/stable/installation/installation/>]
- [4] Ralph online demo: [<https://ralph-demo.allegro.tech/login/?next=/>]

The Author

Holger Reibold, computer scientist, has worked as an IT journalist since 1995. His main interests are open source tools and security topics.

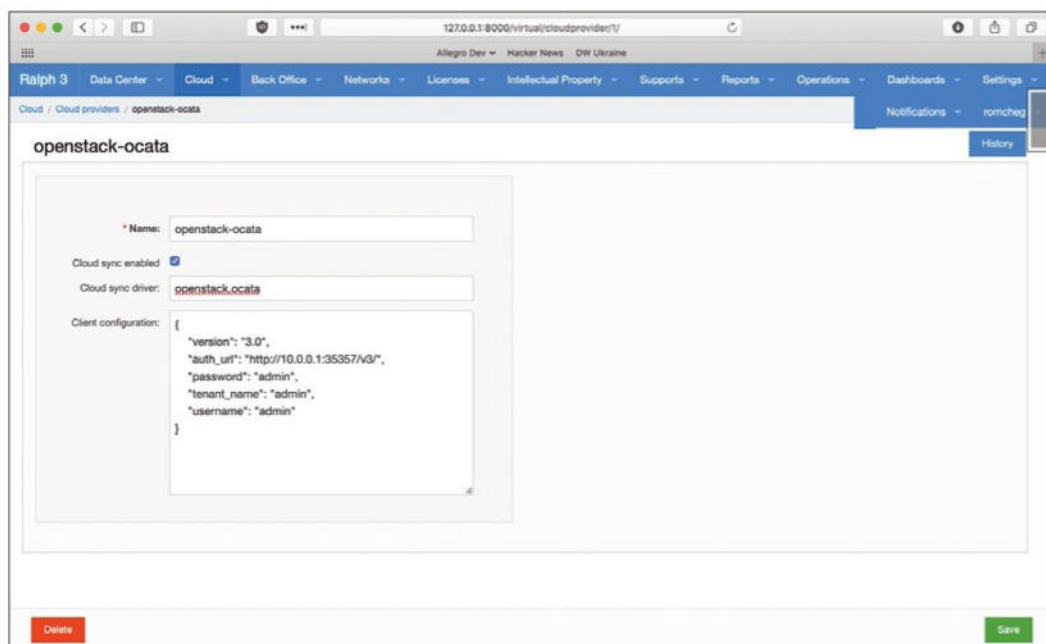


Figure 3: As it stands, Ralph only supports interoperability and asset synchronization with OpenStack installations.



Synchronizing repository changes with GitOps

Special Ops

GitOps applies DevOps practices through infrastructure automation of version control repositories. We look at why it is so popular and why it is often used in the context of Kubernetes. By Artur Skura

In the DevOps world, everybody is talking about GitOps. Its increase in popularity in recent years makes it clear that it's not just a passing fad, but a consistent trend offering significant advantages over legacy approaches.

What GitOps Is Not

Before I move on to specific examples of GitOps principles in practice, it is important to understand what GitOps is and, perhaps more importantly, what it is not. It's probably easier to start with what it isn't, because many people see GitOps as a panacea for all problems in managing applications and their infrastructure. Not only is this belief false – in fact, implementing a GitOps strategy usually involves introducing additional steps and software into your usual workflow, which may make it more complex – but if you don't have experienced staff, things can break more often. You might also think that because GitOps has “Git” in name, it has to be somehow related to Git. In fact, GitOps can be realized in any version

control system (VCS). Because Git is currently very popular, it only makes sense to use it instead of, say, SVN or Subversion; however, nobody forbids you from creating a GitOps solution with these legacy VCS options. Finally, just because Git allows you to revert operations doesn't mean you can as easily bring your infrastructure to a previous state. One of the complications is the persistent data stored in your infrastructure. Take, for example, an Amazon Simple Storage Service (S3) bucket: If you accidentally delete it, you can recreate it later when you discover your mistake (provided someone else hasn't used that particular bucket name), but obviously the objects that had been there will all have disappeared. Therefore, for persistent data, and databases in particular, you need a specific approach that I briefly explain later. In short, GitOps doesn't make backups and all other good practices redundant, and any negligence on these points can have grave consequences. Last but not least, you might hear the opinion that “GitOps is a Kubernetes

thing.” I will address it later in the article, but for now, suffice it to say that although Kubernetes is an excellent platform for GitOps, you can have a perfect GitOps system without Kubernetes, and in many scenarios, this option makes the most sense – especially if your organization doesn't use Kubernetes at all.

OpenGitOps Definition

For some reason the industry prefers a set of defined rules rather than more general statements. For example, most modern companies will tell you they are using the Agile project management methodology for their IT projects, but when you look closer, you will realize most likely it is Scrum with all its deliberate limitations. The case with GitOps is similar: Once you have a precise definition, you can easily check your implementation to verify whether it meets the criteria or not. However, the standard disclaimer applies: Although GitOps works great for many organizations, it doesn't mean it will work for you, and in some cases

Lead image © bambamtiger, Fotolia.com

some modifications might be perfectly fine provided you understand the consequences.

The OpenGitOps project, governed by the Cloud Native Computing Foundation (CNCF), distills GitOps principles into four fundamental concepts [1]:

1. **Declarative Configuration:** The system's desired state must be expressed declaratively, which means describing "what" the system should look like, rather than "how" to achieve that state. It's a shift from imperative scripts.
2. **Versioned and Immutable Desired State:** The desired state is stored in a way that supports immutability, versioning, and a complete history, ensuring traceability and resilience and allowing for quick recovery and analysis of changes over time. Note that the desired state is a comprehensive description of the intended configuration of a system that includes everything necessary to recreate the system or its instances, excluding persistent application data. This data is version controlled, providing a template for the system's setup and configuration.
3. **Automated State Convergence:** Software agents continuously and automatically compare the state of the system to the desired state. Any discrepancies trigger automated actions to align the current state with the desired state.
4. **Continuous Reconciliation:** The system continuously observes the current state and attempts to apply the desired state. This ongoing process ensures consistency and resilience.

Note that in the GitOps context "continuous" refers to the ongoing process of reconciliation between the desired and observed states. Although it is not necessarily instantaneous, it is a constant, automated effort to maintain alignment.

Declarative System Management

If you are familiar with infrastructure-as-code (IaC) tools such as

Terraform, you are well aware of the difference between imperative and declarative approaches. With an imperative approach, the focus is on the commands or steps to achieve a particular state, for example, in a Bash script. With a declarative, you describe the desired state of a system in a high-level, human-readable format, such as CloudFormation. The consequence is that the "how" is very much hidden from the user and only "what" is directly visible. The desired state, in a GitOps context, is a comprehensive blueprint of the system's configuration. It encompasses almost everything, from the infrastructure setup and network configurations to application settings, excluding persistent data. This state is captured in files stored and version-controlled in a Git repository. The significance of this approach lies in its simplicity and clarity; it provides a clear, versioned history of the system's evolution and state over time.

Consider a simple application deployment scenario: deploying a web application onto a Kubernetes cluster. In a declarative model, the configuration files (YAML or JSON) describe the desired state of the application: the number of replicas, network settings, mounted volumes, and more. These files do not contain any commands to create these resources; they simply describe how the end state should look. When these files are committed to a Git repository, they become the single source of truth for the application's deployment, which is one of the reasons GitOps feels very natural with Kubernetes – but there are more reasons.

Once the desired state is defined and stored in a Git repository, automation kicks in. Software agents, such as Kubernetes operators or custom controllers, continuously monitor the state of the system and compare it with the declared desired state in the Git repository. Whenever a discrepancy is detected, these agents act to reconcile the differences, thus ensuring that the state of the system always matches the declared desired state.

This approach offers several benefits:

- (1) It enhances transparency, because the entire configuration of the system is codified and version controlled;
- (2) it simplifies change management, with changes to the system made by updating the configuration files in the Git repository, which triggers the automated deployment and management processes; and
- (3) it improves reliability and consistency, in that the automated processes ensure that the system is always in the desired state, reducing (but not eliminating!) the likelihood of configuration drift and human error.

Immutability and Versioning

The second principle of OpenGitOps refers to immutability and versioning in managing the desired state of systems. This principle is crucial in ensuring that the infrastructure and application deployment processes are both reliable and auditable. Immutability, in this context, implies that once a desired state is declared, it cannot be altered retroactively. Instead, changes are made through new versions, preserving the history of modifications. This approach, when combined with versioning, creates a robust framework for tracking and managing changes over time.

The version control system at the heart of GitOps (which, as previously mentioned, is almost always Git) serves as the perfect tool for enforcing this principle. When configuration files that represent the desired state of a system are stored in a Git repository, each change creates a new commit. This commit acts as a snapshot of the system at a specific point in time. The immutability of these commits ensures that each state of the system can be revisited, understood, and audited. The version history becomes a comprehensive log, detailing every change, who made it, and when it was made.

In practice, managing a system's configuration with Git involves regularly committing changes to the repository. For instance, consider managing the configuration of a web server.

Initially, a configuration file is created detailing the server's settings and stored in a Git repository. When a change is needed, such as updating the server software or changing security settings, the configuration file is edited, and the change is committed to the repository. This action doesn't overwrite the previous state but instead adds a new layer, preserving the old configuration for future reference or rollback.

This methodology offers several advantages. First, it provides a clear audit trail. Every change to the system's configuration is tracked, making it easier to understand the evolution of the system and diagnose issues when they arise. Second, it supports rollback capabilities. If a new configuration leads to issues, it's often straightforward to revert to a previous version, reducing downtime and mitigating risks – unless persistent data is involved, which requires a different approach. Third, versioning improves collaboration and transparency. Teams can work together on the same configuration files, review changes, and merge updates, ensuring that everyone is aligned with the current state of the system.

Note that GitOps principles say nothing of the implementation details, and it is up to you to decide what scenario fits your organization best. A popular setup involves development, testing and quality assurance (QA), and production branches with relevant protection levels to prevent overwriting changes. Usually changes are introduced through merge or pull requests and need to be approved by other members of the team. The whole process is integrated with change management adopted in a given organization.

Does it sound easy? In theory, everything should work fine. In practice, it often does, but when it doesn't, you have to overcome the temptation of proceeding with a "quick fix" in the console, especially in the event of an incident. Instead, learn always to use the defined workflow; otherwise, you can create drift, and in especially complex scenarios, the infrastructure

can enter an unstable state (e.g., because the dependencies that are expected are now missing).

Edge cases and peculiar problems aside, GitOps enhances the stability and security of systems and improves collaboration: When you are confident your work will not break the system, you are more likely to contribute.

Automated Pulling

You have not yet reached the end, though. The first two GitOps principles presented above are common to many deployments, especially when IaC is involved: If you use Terraform, CloudFormation, or Pulumi, you are most probably also using Git, but this by itself is not GitOps, yet – unless you use automated pulling.

In the GitOps framework, software agents are tasked with continuously monitoring the state of the system and pulling the latest configurations from the Git repository. These agents are designed to act autonomously, reducing the need for human intervention in the deployment and management processes. The agents constantly check for updates or changes in the Git repository. When a change is detected, they automatically pull these changes and initiate the process of applying them to the system. This automation ensures that the system is always up to date with the latest configurations as defined by the development or operations team.

A common application of this principle is seen in the integration of these software agents within a continuous integration and continuous deployment (CI/CD) pipeline. In such a setup, the CI/CD pipeline is responsible for the integration and testing of code changes. Once the changes are merged into the main branch of the repository, the software agents take over. They detect the new commit, pull the updated configuration, and proceed to deploy it onto the respective environment, be it testing, staging, or production. This seamless handover from CI/CD to software agents exemplifies the synergy

between development and operations in a GitOps-driven workflow.

Consider a Kubernetes environment in which the deployment of applications is managed through GitOps. In this scenario, a software agent such as Argo CD or Flux is configured to monitor a specific Git repository containing Kubernetes manifests. These manifests define the desired state of the applications and infrastructure within the Kubernetes cluster. The agent is set up to check the repository periodically for changes. Upon detecting an update, such as a new application version or configuration change, the agent pulls these changes and uses Kubernetes APIs to apply them to the cluster. This process ensures that the state of the cluster always matches the desired state declared in the Git repository.

The automated pulling of state declarations by software agents addresses several challenges in modern software development and operations. It minimizes the risk of human error in deployment processes, enhances the speed and consistency of deployments, and ensures a high degree of alignment between the codebase and the operational environment. By automating the synchronization between the desired state in the repository and the current state of the system, this principle of GitOps not only simplifies management but also ensures a more secure and stable operational workflow.

By now you should have a good feeling about why Kubernetes is so often used in GitOps setups: Because deployments are "first-class citizens" in Kubernetes, and features such as rollbacks are built-in rather than patched as an afterthought, automatic pulling feels natural and involves little risk (as opposed to, say, database structure changes).

Continuous Observation and Application

Continuous observation in GitOps is implemented through software agents that perform two main functions. First, they monitor the state

of the system in real-time, which involves checking the configuration and operational status of the system's components, from infrastructure to applications. Second, they compare this observed state to the desired state as defined in the Git repository. If discrepancies are found, these agents work to reconcile the differences, applying changes as necessary to align the current state with the desired state. This process of continuous observation and application ensures that any drift in the system is promptly addressed and corrected. To facilitate this continuous observation, a variety of monitoring tools and scripts are employed. These tools can range from simple scripts that check system health to more sophisticated monitoring solutions that provide real-time insights into system performance, resource usage,

and operational status. In a Kubernetes environment, for example, tools such as Prometheus can be used to gather metrics and monitor the state of the cluster, and custom scripts can be written to check the health and status of specific applications or services. These tools and scripts feed information back to the software agents, enabling them to make informed decisions about whether the system is in the desired state or if actions need to be taken to correct any deviations. Alongside observation, the continuous application is an integral part of this principle. It refers to the ongoing process of applying the desired state to the system. In practice, this means that whenever the Git repository is updated with a new or modified configuration, the software agents automatically apply these changes to the system. This continuous application is

what enables GitOps to deliver on its promise of fast, reliable, and consistent deployments. It ensures that any changes made in the Git repository are quickly and accurately reflected in the live system, keeping the system in a constant state of alignment with the declared desired state. Those of you familiar with Kubernetes may have already noticed that some of the observation mechanisms described here are either built in or easily integrated with this popular container orchestration platform, which is one more reason to use it as a basis for a GitOps platform.

A Simple Example

Too many words can sometimes muddle the waters and complicate quite simple ideas. GitOps is not rocket science and can be implemented in various scenarios. To

Get HPC Update every month!

Tune in to the HPC Update newsletter for news, views, and real-world technical articles on high-performance computing.


Subscribe free!




bit.ly/HPC-ADMIN-Update

HPCUPDATE

September 12, 2023 Issue 176




HPC ILLUMINATIONS
PAVILION
A New Opportunity for
Underrepresented Groups
at SC23



LEARN MORE
& APPLY

This Month's Feature



Where Does Job Output Go?
By Jeff Layton
Where does your job data go? The answer is fairly straightforward, but Jeff evolves the question to include a discussion of where data "should" or "could" go.

News and Resources

- [RIKEN Brings AI to Quantum Error Correction](#)
- [Submissions Are Open for ISC 2024](#)
- [Math Magic with MathLex](#)

illustrate this point, I'll build the most basic GitOps platform with a short Bash script.

All disclaimers apply: This example is only educational and lacks most of the features you will find in real GitOps platforms; I have implemented no error checking; pulling is based on time rather than a webhook or another mechanism that would fire the synchronization off on each commit; and so on and so forth. Nevertheless, **Listing 1** illustrates the basic principles relatively well.

The logic of the script is quite straightforward: After defining the configuration and cloning the repository (which is only necessary on the first run), the script pulls the latest changes from the remote repository and compares them against the local

repo. If any discrepancies are detected, it uses the brutal but efficient method of resetting, effectively rewriting Git history in the local repository to match remote changes. Finally, the deployment happens – in this case by Docker Compose. Again, I would never do it in this way in a real-world scenario (unless a break in service availability is acceptable, which is quite rare nowadays).

What About the Data?

As previously mentioned, GitOps principles, at least as defined by OpenGitOps, don't directly apply to data. This statement basically means that, especially for unstructured data, you need to take special care not to lose it. Snapshots, backups, archival copies – you need to make use of the optimal combination of proven techniques to keep your data safe, in line with the policies of your organization. What about structured data, though? Wouldn't it make sense to apply the GitOps paradigm to it? Yes, of course! Over the course of years, many projects (e.g., Liquibase, Flyway, Atlas) have aimed at exactly this goal: not treating data in databases differently from code from the GitOps point of view. That is, after changes in a database structure have been approved, for example by a proper merge or pull request, they are introduced into the database through a proper agent, with versioning and rollback capabilities. (Note I am talking about changes in the database *structure*, such as adding new tables, changing the names of columns, etc., and not about changing the stored data itself. You could have millions of such operations per second, and following a GitOps workflow would make no sense in most scenarios.) If you are interested in this approach and need a more robust framework, check the Bytebase project [2], which integrates several tools in one framework, making it easier to manage databases such as Postgres in a GitOps-oriented way.

Conclusion

When you start learning GitOps, the number of names and tools seems overwhelming, with each one seemingly better than the other. Nevertheless, their main job is to synchronize the system with the changes introduced in the repository as described above. Whether you choose Flux, Argo CD, or another similar tool, the basic idea is the same, even if implementation details may be quite different.

Do you have to use Kubernetes when adopting the GitOps approach to your applications or infrastructure? Definitely not, but if you do so, you automatically enjoy several benefits. One of the important advantages that is not strictly related to the GitOps principles described earlier is secrets management: If you use a Kubernetes operator to synchronize your local Kubernetes cluster, you basically eliminate the need for having a separate deployment infrastructure with workers, runners, or agents employing authentication secrets to make deployments, because everything you need (or nearly, depending on your setup) is already available in your cluster. Therefore, you not only make your infrastructure more resistant to human error, you also make it inherently more secure. Unless the attacker gets access to your VCS, that is.

Listing 1: Basic GitOps Platform

```
#!/bin/bash

# Initial configuration
REPO_URL="https://your-git-repository.git"
LOCAL_DIR="/path/to/local/repo"
COMPOSE_FILE_PATH="$LOCAL_DIR/docker-compose.yml"
CHECK_INTERVAL=60 # arbitrary value in seconds

# Clone the repository initially if not present
if [ ! -d "$LOCAL_DIR" ]; then
    git clone $REPO_URL $LOCAL_DIR
fi

# Pull latest changes and redeploy if there are new changes
update_and_deploy() {
    cd $LOCAL_DIR
    git fetch origin

    # Check if there are new commits
    LOCAL=$(git rev-parse @)
    REMOTE=$(git rev-parse @{u})

    if [ $LOCAL != $REMOTE ]; then
        git reset --hard origin/master # Sync with repo

        # Deploy using Docker Compose
        docker-compose -f $COMPOSE_FILE_PATH down
        docker-compose -f $COMPOSE_FILE_PATH up -d
    fi
}

# Main loop
while true; do
    update_and_deploy
    sleep $CHECK_INTERVAL
done
```

Info

[1] OpenGitOps definition of GitOps:

[<https://opengitops.dev/>]

[2] Bytebase on GitHub:

[<https://github.com/bytebase/bytebase>]

Author

Artur Skura is a senior DevOps engineer currently working for a leading pharmaceutical company based in Switzerland. Together with a team of experienced engineers, he builds and maintains cloud infrastructure for large data science and machine learning operations. In his free time, he composes synth folk music, combining the vibrant sound of the '80s with folk themes.

GET TO KNOW ADMIN



ADMIN Network & Security magazine is your source for technical solutions to real-world problems.

ADMIN is packed with detailed discussions aimed at the professional reader on contemporary topics including security, cloud computing, DevOps, HPC, containers, networking, and more.

Subscribe to *ADMIN*
and get 6 issues
delivered every year

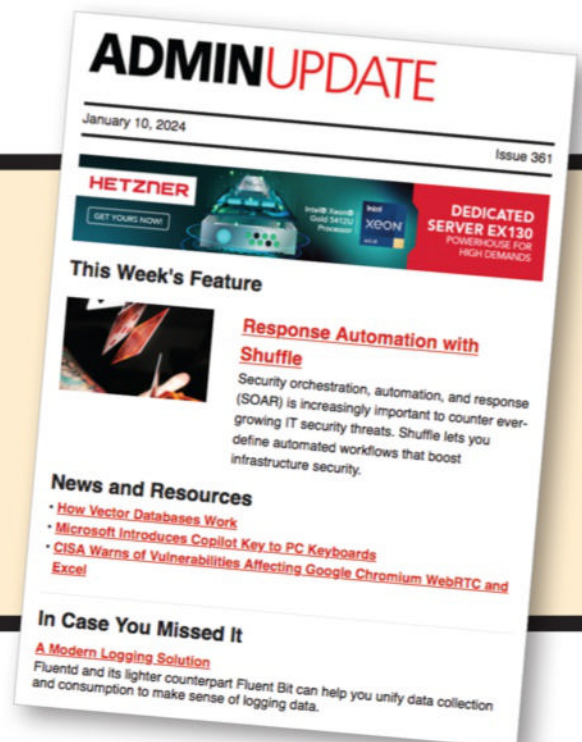


Want to get ADMIN in your inbox?

**Subscribe free to
ADMIN Update**

and get news and technical articles
you won't see in the magazine.

bit.ly/HPC-ADMIN-Update



ADMIN
Network & Security



@adminmagazine



@adminmag



ADMIN magazine



@adminmagazine

Border Gateway Protocol

From A to B

We look at the Border Gateway Protocol, how it routes packets through the Internet, its weaknesses, and some hardening strategies. By Benjamin Pfister

The Internet comprises a mix of autonomous systems (ASs) – networks and systems each under the administrative control of a specific provider – that have officially registered numbers known as AS numbers (ASNs). The Border Gateway Protocol (BGP), the latest version of which is BGP4, ensures accessibility between the autonomous systems and is designed and optimized for handling high volumes of routing information with a high level of stability. Besides providers, large corporate and government customers also have to deal with BGP if they use or want to use multihoming (i.e., connecting your own autonomous system to several providers). BGP is also used on some internal networks and forms the basis for multiprotocol label switching (MPLS) in wide-area network (WAN) structures, but can also be used for Ethernet virtual private networks (EVPNs) or in combination with a virtual extensible local area network (VXLAN) in data center networks. Today, BGP is capable of many more functions than simply distributing IP prefixes. The protocol therefore has a wide range of options for policy-based route selection.

Basics

In contrast to the various Interior Gateway Protocols (IGPs) such as the Routing Information Protocol (RIP),

Open Shortest Path First (OSPF) protocol, or Enhanced Interior Gateway Routing Protocol (EIGRP) for internal networks, BGP is the only Exterior Gateway Protocol (EGP). As such, it is based on the path vector principle, which has similarities to the distance vector IGPs used to exchange routes within an autonomous system and optimized for fast convergence times to meet the stringent requirements for low downtimes – right down to the millisecond range. However, even smaller numbers of routes still need to be processed.

Extensions to BGP make it multiprotocol capable (MP-BGP4); that is, it supports IPv4 and IPv6. BGP can process and separate different types of information and contexts in these “address families.” According to information from the American Registry for Internet Numbers (ARIN), as of 2023, a full BGP table on the Internet contained around 940,000 prefixes for IPv4 and 172,400 prefixes for IPv6. Routers therefore need to have a large amount of physical memory.

ASNs in Practice

As already explained, a network operator requires an ASN for the exchange of routing information. Public ASNs are assigned by Regional Internet Registries (RIRs). The Réseaux IP Européens Network Coordination Center (RIPE NCC) is responsible for

this task in Europe, the Middle East, and parts of Central Asia.

An official ASN assignment from the RIR is required for a redundant Internet connection by more than one carrier (multihoming). Additionally, an IP address block independent of the provider must be assigned. Provider-independent (PI) or provider-aggregatable (PA) address blocks are used for this purpose. However, this process has become difficult because of the scarcity of public IPv4 addresses. Smaller customers are normally assigned addresses by the provider. If the company or authority has its own AS and public address blocks, it assumes the role of the local Internet registry (LIR).

Legacy ASNs are 16 bits in length and decimal (ASPlain). Newer ASNs use 32 bits and are dot separated. This format is known as ASDot [1]. ASN 6541 in ASPlain becomes 0.6541 in ASDot notation.

Route Selection and Attributes

BGP uses different types of attributes to influence the choice of the appropriate route, distinguishing between transitive and non-transitive, normal and path attributes. When people start to talk about BGP, the conversation quickly turns to peering, which means the neighborhood connection between BGP routers and, consequently, autonomous systems.

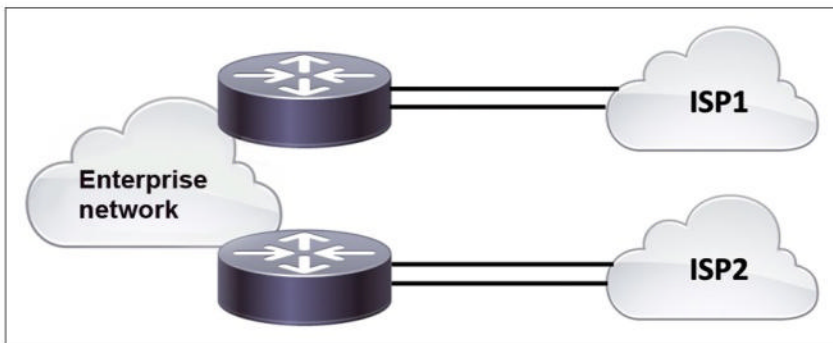


Figure 1: In BGP dual multihomed design, several providers are each connected to the customer's AS by several connections.

However, BGP routers do not simply use multicast to find their neighbors when enabled, as is usually the case with the IGP in an AS.

With BGP, the administrator on the router must explicitly store the neighbors, including their IP addresses and the remote autonomous systems in the respective routing process, and then reverse it on the peer router (Figure 1). If router A in AS 64496 with an IP address of 192.0.2.1 wants to peer with router B in AS 64500 and IP address 192.0.2.2, router A must store the neighbor 192.0.2.2 with AS 64500 and router B the peer 192.0.2.1 with AS 64496. If the configurations do not match, peering will not take place.

Communication Process

A fundamental distinction exists between external (eBGP) and internal BGP (iBGP) peering. iBGP peering takes place within the same autonomous system (e.g., to exchange prefixes between two BGP routers connected to different carriers). eBGP peering takes place between different autonomous systems, as in the previous example.

However, the two types of BGP differ in terms of implementation, as can be seen, for example, in the multihop function, where the time to live (TTL) is influenced by the IP header. Several hops to the peer can be the default for iBGPs. With eBGP, you need to configure this explicitly to support peers located on different subnets. In this case, the TTL counter is incremented to reach the peer.

The handling of the next-hop IP address is also different. With eBGP, the router always transmits the IP address of the outbound interface as the next hop in the accessibility information, and the receiving router replaces this with the outbound IP address of the interface it used during redistribution. With iBGP, the IP address is adopted without change on forwarding. However, this address must also be accessible for the target peer. If it is not, you can configure the next-hop self parameter to achieve the same behavior as with eBGP.

BGP uses connection-oriented TCP on port 179 as the transport protocol, which must be allowed on the transmission path. When establishing peering, the connection setup goes through different phases (Figure 2). The idle phase begins with the idle status. Then a TCP three-way handshake with the familiar SYN, SYN/ACK, ACK is used in the connect phase. If this is successful, the OpenSent phase is used to synchronize BGP information such as ASNs and authentication data, which is normally followed by the OpenConfirm

phase, in which the peers wait for keep-alive messages. If successful, the Established phase is next, which is where peering is set up and the routers can exchange availability information.

This process uses the interface on the routers that leads to the target IP address according to the routing table. To be able to cover scenarios with multiple paths without the risk of flapping (disconnection and reconnection) of the BGP peering in the event of a path error, a permanently accessible IP address should be used as the source in such a case (Figure 3). To do this, permanently active loopback interfaces are used as update sources. These interfaces are then also stored on the remote station as peer addresses and included in the remote station's routing to be able to establish peering (Figure 4).

Avoiding Loops

Freedom from loops is one of the key requirements of any routing protocol. In BGP, this is based on the AS path – that is, a list of the transit ASs through which a packet has already passed. Each autonomous system adds its ASN starting with an eBGP peering (Figure 5). An ASN can occur several times in direct succession, but cannot be fragmented in different places. If the ASN is fragmented in

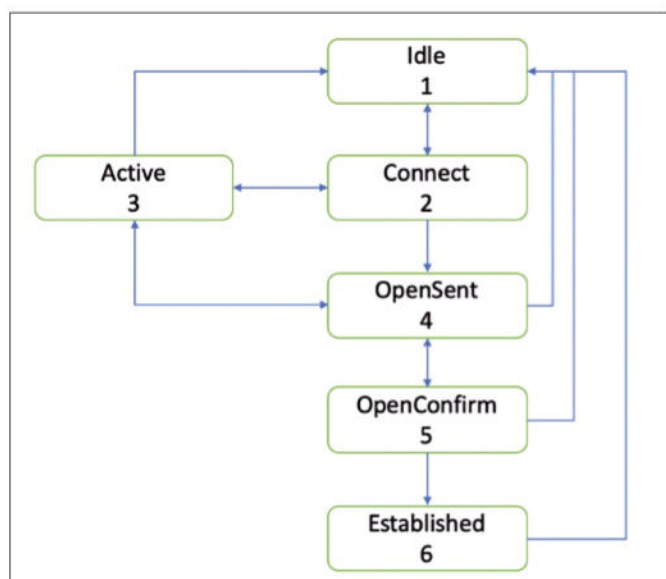


Figure 2: The BGP peering process comprises six phases.

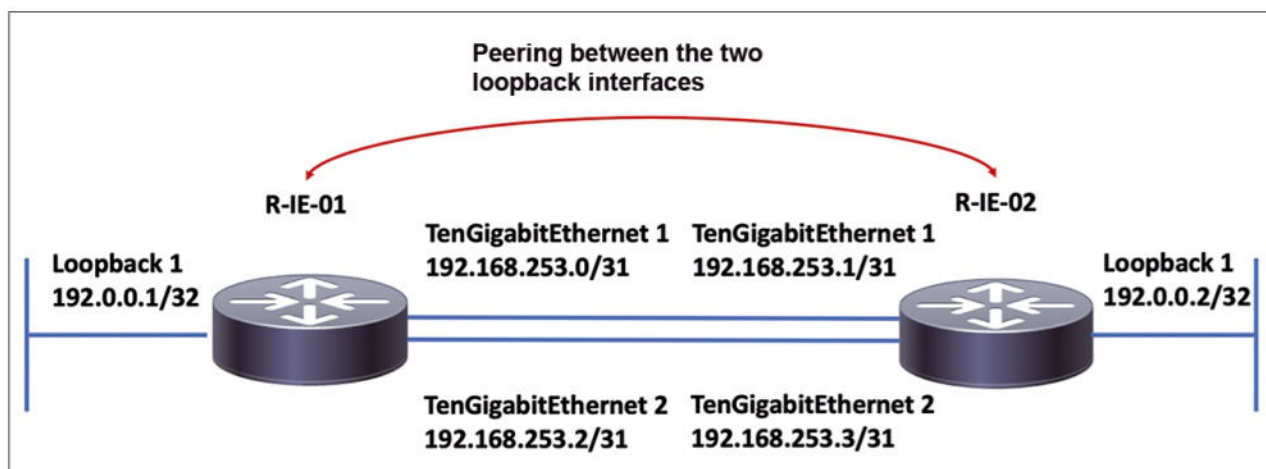


Figure 3: BGP peering with loopback interfaces avoid session flapping if a connection fails.

the AS path, BGP drops this potential path, identifying it as a loop. To extend a path artificially, you have an option for concatenating the same ASN with an AS path prepend. Because iBGP only has one ASN for all participating routers, this procedure cannot be used here. To counter this, iBGP and eBGP behave differently when forwarding availability information. If an iBGP peer receives information, it only forwards it to eBGP peers, but not to other iBGP peers. eBGP peers in turn forward incoming reachability information to both eBGP and iBGP peers. The iBGP's behavior leads to some design restrictions in iBGP, which can be countered with several tools. Normally, iBGP would require a full meshing of all peers. However, this

does not scale well in larger environments. Confederations and route reflectors offer a remedy. Confederations are used to define subzones, but route reflectors are more common. When iBGP peers receive reachability information, the router forwards it to the eBGP peers and the route reflectors used for scaling, but not to the other iBGP peers.

Publishing Routes

Once the peering is in place, accessibility information needs to be exchanged, as well as the originating and redistributing options. With originating, you configure a network to be announced in your AS. The BGP router searches the routing table for a known path to this network; it can

from OSPF to BGP). On this basis, the router transfers all paths from one protocol to the other – in this case to the BGP table. However, prefix lists can impose restrictions. If you do not want to announce routes that are too small, you can combine these routes to create a summary route.

As mentioned previously, BGP uses attributes for path selection, which you can manipulate with the use of route maps. [Table 1](#) lists the attributes in the order in which they are to be processed and with a description in each case. Not all manufacturers use the Weight attribute, and it is only significant locally on a router for making outgoing route selections. With regard to the prefix, a higher weight for peering is preferred to a lower one. The Local Preference attribute is only exchanged between iBGP peers. For example, a preference for selecting the outgoing route via a specific peer can be established within an AS.

The Locally Originated attribute handles local routes before routes learned from BGP. The length of the AS Path (i.e., the list of the ASs to be passed through) is not checked until the next step. Routers prefer shorter paths. If a decision is not made on the basis of the AS Path Length, the Origin Code (i.e., the prefix source) is used. Local prefixes take precedence over prefixes learned from EGP, followed by redistributed paths.

The Multi Exit Discriminator (MED) attribute is only used for eBGP between two ASNs. Where several

aut-num:	AS140
as-name:	ASTEST-AD
admin-c:	DA1-TEST
tech-c:	DA1-TEST
org:	ORG-AD1-TEST
import:	from AS65501 accept AS65501
export:	to AS65501 announce AS140
status:	ASSIGNED
mnt-by:	DRNG-ANDRS-MNT
mnt-by:	TEST-DBM-MNT
created:	2023-08-19T14:17:53Z
last-modified:	2023-08-19T14:20:38Z
source:	TEST

Figure 4: The ASN entry in the RIPE NCC test database shows the routing guidelines for import and export.

be known through static routes or dynamic routing protocols such as OSPF. If successful, the router transfers these networks to the BGP table. During redistribution, you specify a source and a target routing process (e.g.,

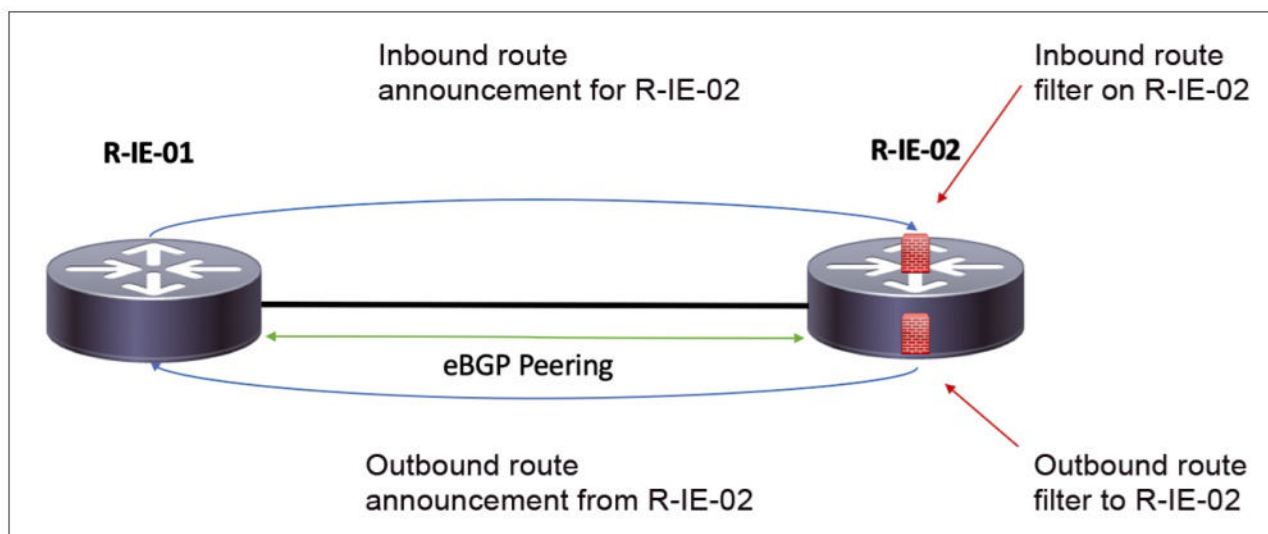


Figure 5: A simple example of the use of BGP route filters. These filters can be used for both inbound and outbound announcements.

peering instances to the same AS exist, a router can use the MED to specify a preferred path in the return direction. In the next step, the router prefers the path with the better metric to the BGP next-hop in IGP, which is why BGP is still based on IGP. Because BGP is designed for stability, it would favor older entries over newer ones as the next path decision parameter. If a decision has not yet been made, the router compares the router ID of the next hop and gives priority to smaller numerical values. Something similar is also used as the last tie breaker. A lower numerical IP address is preferred over higher ones as a path. These attributes show that BGP supports complex guidelines for route selection that make the protocol anything but easy to handle.

Challenges and Security

The RIRs provide a database in which AS operators store their guidelines, which define how AS operators announce or accept routes to and from external autonomous systems. From the outbound policy, appropriate inbound route filters can be set for peering partners on the partner's side of the peering, keeping BGP clean and making prefix hijacking more difficult [21]. Besides, each AS operator provides contact details in case of technical problems or misuse. Of course, you cannot just blindly trust an arbitrary prefix from an arbitrary source. True to the motto "trust, but verify," it is important to check the incoming prefixes of the remote stations, which are referred

to as peers in the BGP context [3]. The practice of autonomous systems distributing prefixes not assigned to their ASs is referred to as prefix hijacking. The aim is to redirect data traffic by manipulating the path selection. On one hand, this arrangement can be used for man-in-the-middle attacks that sniff or manipulate data. On the other hand, denial of service attacks that disrupt the availability of services or entire networks are also possible. Last but not least, it could simply be a misconfiguration. Repeated cases of prefix hijacking have been seen, which highlights the need for countermeasures. However, incoming filters for route announcements cannot be used meaningfully or, if they are used, are under many restrictions.

Table 1: BGP Attributes

Attribute	Description
Weight	Valid locally on a router to define the outgoing preference.
Local Preference	Valid within the AS to define the outgoing preference.
Locally Originated	Local before learned routes.
AS Path Length	Number of ASNs in transit.
Origin Code	IGP sources are preferred over EGP sources, which in turn prefer redistributed information.
MED	Message to neighboring AS to route traffic for certain destinations by specified link.
eBGP Path over iBGP Path	Paths by another AS are preferred over paths within the AS.
Shortest IGP Path to BGP Next-Hop	Preferred path to the next BGP router determined from IGP information.
Oldest Path	Older paths preferred.
Router ID	Next hop with the smallest router ID is preferred.
Neighbor IP Address	Next hop with the smallest IP address is preferred.

Signing with Public Keys

The Internet Engineering Task Force (IETF) developed the Resource Public Key Infrastructure (RPKI) [4] as a potential countermeasure. This framework defines which autonomous systems are allowed to announce which prefixes. The attestation is based on the International Telecommunication Union Telecommunication Standardization Sector's (ITU-T's) X.509 PKI framework. RPKI uses the same hierarchy as for IP address assignment to reflect the chain of trust and consequently ensure attestation. The Internet Assigned Numbers Authority (IANA) is the root element of the PKI. From there it passes through the RIRs to the LIRs.

RPKI can use different responsibility models. Large companies or providers can use a local PKI, also known as delegated RPKI, and the option of a hosted RPKI. In this case, the PKI is operated by the RIR (the RIPE NCC in Europe). Each LIR can have ASNs and IP prefixes attested by a certificate. Route origin authorizations (ROAs) can be created in this way. ROAs include the authorized ASN, the IP prefix, and the maximum length of the prefix, which means a potential attacker does not have the option of announcing this prefix with a more specific or higher prefix length, because a router prefers longer prefix lengths and an attacker could otherwise leverage this fact to redirect data traffic for a more specific prefix.

Certificates offer an option for cryptographic verification of the prefixes. If you do not set a maximum prefix length, the AS can only announce the entire prefix and not specific parts of it. Email alerts at RIPE-NCC point out unauthorized announcements and misconfigurations.

However, the remote station also needs to validate the ROA information of the incoming prefixes. This check can lead to the *Not Found*, *Unknown*, *Valid*, or *Invalid* results. Valid means that the check of the AS, the prefix, and the corresponding length was successful, whereas Invalid means the opposite. Not Found or Unknown means that no ROA was found. RIPE NCC provides sample configurations for manufacturers such as Cisco Systems and Juniper. However, not all router platforms support RPKI. Additionally, a potential compromise of the certification authority naturally would pose a risk.

BGPsec Validation

RPKI is not the only hedging option. Also based on RPKI, BGPsec offers a way of creating a chain of trust. The source of the routing prefix, the Originator, uses RPKI to sign the information, and all other routers of the AS path follow suit and sign with their private keys; therefore, each autonomous system authorizes the transferred prefix in the BGP update. However, the use of RPKI and BGPsec also harbors risks for availability: Peers also need to check the certificates issued by the PKI. The Network Time Protocol (NTP) service must be available to check the validity period, as must HTTP-based services such as Online Certificate Status Protocol (OCSP) and certificate revocation lists (CRLs) to determine a possible revocation. However, routing information is required in turn, which leads to a loop dependency. With BGPsec, prefixes cannot be aggregated to reduce the number of prefixes transmitted. The PKI operator can also withdraw sovereignty from the AS operator by

revoking the certificates used to sign prefixes or refusing to issue a certificate. Admins and security managers can set static route

filters to restrict inbound and outbound route announcements, which can be handled on the basis of the aforementioned data from the routing guideline of the responsible RIR. The guidelines are not particularly flexible. That said, it is always advisable to set route filters that block incoming prefixes from external sources.

For almost all peering instances, you will want to avoid standard routes because they only make sense for end customers without multihoming. Additionally, BGP routers should filter private IP address ranges bidirectionally with inbound and outbound route maps because other routers do not route them on the Internet anyway. These route maps usually match with prefix lists that can be checked for a subnet mask and prefix length in the accessibility information. To do this, they specify the number of matching bits with the "less than or equal to" (le) and "greater than or equal to" (ge) operands (Table 2). Prefix lists offer the possibility of establishing an inbound and outbound binding to the peer in question. On this basis, you can apply inbound and outbound prefix filters, which enables pre-filtering in the distribution of prefixes. Route maps that use AS path ACLs, prefix lists, or community tags for this purpose are slightly more granular in terms of classification. The matching BGP attributes can be adapted on this basis. The route maps use sequence numbers for sequential processing of "classification" and "adaptation." They offer the option of changing BGP attributes, next-hop, and community tags with a set command on the occurrence of a positive match result.

BGP communities [5] can offer additional features and are a valuable addition. They use numbers as tags, which, in turn, define an expected behavior; therefore, a BGP directive can be controlled remotely. Both well-known and custom communities exist. Use of communities must be coordinated between the peers involved. You can store special tags in the communities for prefixes and, by doing so, communicate control

Table 2: Prefix Examples

Attribute	Description
0.0.0.0/0	Default route only
0.0.0.0/0 ge 32	All host routes
172.16.0.0/16 ge 24	All subnets in the IP range 172.16.0.0/16 with a minimum prefix length of 24 bits
192.168.0.0/24 ge 27 le 30	All prefixes that begin with 192.0.0 and whose prefix length is between 27 and 30

commands to the peer for handling the data traffic for this prefix. Besides the filter and control options already presented, it also makes sense to filter out private ASNs in the AS path and AS paths that are too long.

Securing the TCP Session

Risks do not just occur at the BGP protocol level. A potential attacker could attack the control plane by sending masses of packets to the BGP port, triggering a denial of service. The first step is to restrict the communication relationships for the socket setup. Stateless packet filters – access control lists (ACLs) – can be used for this purpose, specifically to make sure that the target port for BGP (TCP/179) is only accessible from legitimate source IP addresses.

However, this alone is not enough. It is also important to ensure the authenticity and integrity of the data. Otherwise, an attacker could carry out blind insertion and replay or reset attacks. Blind insertion means an attacker attempting to inject false routing information or a session reset (i.e., a termination) with a spoofed IP address on a router that is not secured by authentication. The big challenge, however, is that the TCP sequence number must match the expected segment, which requires both knowledge of the current session and correct timing. If a session reset occurs, a completely new setup is required, which means learning hundreds of thousands of items of routing information; the end effect is a denial of service as it is happening. Authentication by cryptographic procedures can provide a remedy. However, outdated procedures such as MD5, which are now considered insecure, are still mostly used. A symmetric key is available on both peers. Each TCP segment contains a previously calculated message authentication code (MAC). The recipient checks this before accepting it on the basis of the content in TCP headers, the content data, and the configured symmetric key. If calculations return a different result than the received

value, the receiving router does not accept the segment.

Despite MD5 being considered vulnerable for years because of possible collisions, changing the key with this method would result in the TCP session being terminated. Consequently, a new BGP session is created and prompts relearning of the routing information, which will take some time.

The TCP authentication options (TCP-AOs) method was developed as an optimized procedure and standardized in RFC5925. This procedure enables the key to be exchanged without interrupting the TCP session and, as a result, the BGP session; avoiding interruptions is particularly beneficial for long-term TCP sessions such as BGP. TCP-AO is only used to check the authenticity of the sender, with no encryption of the user data, unlike the Transport Layer Security (TLS) or Internet Protocol Security (IPSec) protocols. The TCP-AOs are based on master key tuples (MKTs) for this purpose. Management can be carried out both statically and by an out-of-band mechanism. The connection keys (traffic keys) are then derived from the MKTs.

Black Holes

For some years now, attacks that restrict or completely prevent the accessibility of services have been on the rise. These attacks, known as denial of service (DoS) or, in the case of multiple sources, distributed denial of service (DDoS), can be launched at either the network or application level. Network-level DoS and DDoS attacks are aimed at overloading network connections. Volumetric attack is another way of putting this. Attacks at the application level (e.g., on web servers) exploit vulnerabilities in applications or specific application behavior to take down the servers. In both variants, the first step is to detect the traffic pattern and then filter, limit, or redirect the data traffic. Because volumetric attacks in particular are associated with high data rates that connections cannot handle, it is

important to intercept this data traffic up front. One conceivable method would be blackholing [6] on the provider side, which means a router drops all packets to a specific destination into a null route.

This strategy would allow the upstream provider to be informed by BGP that it needs to drop packets to specific target IP addresses – also known as remotely triggered black hole routing (RTBH). The customer sends a /32 prefix for IPv4 or /128 for IPv6 with the attacked target IP address and BGP community 666 to the peer. However, because peers do not accept these host prefixes by default, specific coordination with the provider is required. In this case, the IP address also is no longer accessible, but overloading of the connection then stops and other services are no longer affected.

Conclusions

BGP is still a fundamental component of the Internet in 2024, although not many people are familiar with the background information. Routing failures from misconfigurations or attacks, for example, can have an enormous effect, even if only in some areas because of the decentralized structure of the Internet. Because of the ever-increasing dependence on online services, you need to keep an eye on BGP and look into options for securing it. ■

Info

- [1] ASDot and ASPlain: [\[https://www.networkers-online.com/tools/bgp-asn-4byte-converter/\]](https://www.networkers-online.com/tools/bgp-asn-4byte-converter/)
- [2] Prefix hijacking: [\[https://www.youtube.com/watch?v=IzLPKuAOe50\]](https://www.youtube.com/watch?v=IzLPKuAOe50)
- [3] RFC7454: BGP Operations and Security: [\[https://datatracker.ietf.org/doc/html/rfc7454\]](https://datatracker.ietf.org/doc/html/rfc7454)
- [4] RPKI validation: [\[https://www.ripe.net/manage-ips-and-asns/resource-management/rpki\]](https://www.ripe.net/manage-ips-and-asns/resource-management/rpki)
- [5] BGP communities: [\[https://www.youtube.com/watch?v=FMzPOZQawKI\]](https://www.youtube.com/watch?v=FMzPOZQawKI)
- [6] RFC7999: BLACKHOLE Community: [\[https://datatracker.ietf.org/doc/html/rfc7999\]](https://datatracker.ietf.org/doc/html/rfc7999)

Building a defense
against DDoS attacks

One Against All



Targeted attacks such as distributed denial of service, with thousands of computers attacking your servers until one of them caves in, cannot be prevented, but they can be effectively mitigated. By Markus Stubbig

Cyberattacks come in many forms, such as secret spying on company networks, sabotage, and disruptive actions. In a disruptive action, denial of service (DoS), the attacker attempts to overload a server with requests until it stops working. This attack is easier said than done, because servers usually have more power in reserve than a single client can call up. The obvious idea is to attack the

CDN

Many providers mention DDoS and content delivery networks (CDNs) in the same context. A CDN distributes its servers across as many data centers as possible around the world. The aim is to locate the data closer to the customer so the content is delivered locally and not sent halfway around the world. This method saves provider bandwidth and makes the services more responsive for the customer.

As a positive side effect, the CDN provides protection against DDoS attacks because the attacker has to deal with many instances of the same service. If the attacker manages to keep up the attack against data center A with its distributed army, the CDN provider can simply switch to another data center and serve its customers from there. The difference is that DDoS is the attack and CDN is a possible defense.

same server with many clients. The resulting attack is called a distributed denial of service (DDoS). The attacker rounds up a huge number of computers in the form of a botnet. You are not powerless against DDoS attacks, but it is important to introduce appropriate measures up front, because during such an attack, the Internet line is flattened by the flood of client requests and the server farm no longer responds. The simplest, albeit most expensive, measure is to upgrade your infrastructure with more servers and more bandwidth. If your budget allows for this approach, you need read no further. For everyone else, this article describes various ways of efficiently protecting your infrastructure. The aim is not to provide protection against huge attacks that hit at terabit per second rates, but simply to make your servers more robust. The box “CDN” describes why CDNs and DDoS are often mentioned together.

Hardening Operating Systems

The first level of protection is provided by stricter settings on the operating system side, ranging from

shorter timeouts, more restrictive firewall rules, and general recommendations for software and libraries, to specific kernel settings. Rather than standardized guidelines, numerous recommendations are offered for hardening the operating system [1]. All best practices are aimed at keeping the attack surface as small as possible. If you don't feel like typing the numerous commands for these steps by hand, you can easily use a preconfigured script [2]. However, only do this if you understand the individual lines of the script and they match your security policy. The result is a fireproofed operating system that is not so easy to mess around with. Trust is good, control is better. Software auditors apply this principle to check local computers for insecure settings and known vulnerabilities. If you don't want to fire OpenVAS at your servers, you can use the leaner Lynis [3]. The tool runs more than 200 tests and after a few minutes presents its report in the terminal or by email. Everything that is suspicious or insecure is highlighted in red. At the end of the report, the tool provides specific recommendations for a more secure configuration.

Photo by Arisa Chattasa on Unsplash

Protection at OSI Layers 3 and 4

The major providers refer to protection Layers 3 and 4 of the Open Systems Interconnection (OSI) model for network protection, which means blocking clients by IP address and geo-blocking, which ultimately only consists of a long list of IPs. For example, the FireHOL blacklist contains all addresses that have occupied a place on a blacklist in the past [4]. Although this step will exclude the known troublemakers, the list does not take into account the individual attack situation, which is where dynamic blocking with Fail2Ban [5] comes into play (Figure 1). The tool assumes that every IP address is benevolent, but if an address appears in the logbook on several occasions from failed login attempts, it creates a block entry in iptables/nftables. The result is no communication with this address. After a short wait, the client with this address can join the game again and start its three attempts. Of course, the values and timeouts are customizable and can be increased to many hours for aggressive would-be attackers.

The community takes this one step further with CrowdSec [6]. The principle is similar to Fail2Ban; all participants share the list of blocked addresses. When an unknown attacker attacks its first server, this server

informs the CrowdSec community and all participants block the attacker's IP address. This puts a quick end to a dictionary attack on the SSH service. Conversely, you can also quickly lock out your IP address if your tired fingers fail to type in your password correctly late at night.

Although the Fail2Ban and CrowdSec approach effectively prevents brute forcing of user-password combinations, it has two flaws. On one hand, the CPU load increases if the server receives many requests and Fail2Ban has to parse large logfiles. On the other hand, Fail2Ban can block several clients with one IP address if they share a public address – which is likely to be the case with most IPv4 Internet connections.

Setting Up Geo-Blocking

If you want to appeal to an international audience, you have to grant access to all countries on your website. The reverse is also true: If you do not have any trade relations with Asia and Africa, a geo-block can exclude almost 100 countries. An attacker in Somalia will not see an SSH prompt where they can try out password combinations.

Thanks to the `geoip` module and the free country IP list from MaxMind, the Linux firewall becomes a geo-filter in next to no time. The filter command contains the country code

as the source instead of an IP address. For example, a block for Somalia implemented by iptables is:

```
iptables -A INPUT -m state --state NEW \
        -m geoip --source-country SO \
        -j DROP
```

Depending on the infrastructure or hosting offer, a firewall with a geo-function can be installed upstream of the server, which enables convenient configuration of the security policy in a web GUI. In Figure 2, the OPNsense open source firewall blocks access to individual countries by clicking on the flag symbols.

Securing Websites

Many websites offer a members-only area that can be accessed by password. As with an SSH service, the attacker tries typical username and password combinations; this method will eventually lead to success unless a form of website protection puts a stop to it.

The website protector works like this: It sends a work order to the client's web browser and only displays the requested web content once the task has been completed. The activity can vary from a simple *I am not a robot* checkbox to, say, *Select all squares with traffic lights*. Other watchdogs package the task in JavaScript, which the browser processes without human

```

CPE-FRA-C2
root@cpe-fra-c2 ~> fail2ban-client status sshd
Status for the jail: sshd
- Filter
  - Currently failed: 0
  - Total failed: 1303
  - Journal matches: _SYSTEMD_UNIT=sshd.service + _COMM=sshd
- Actions
  - Currently banned: 2
  - Total banned: 185
  - Banned IP list: 103.186.28.56 143.198.146.129
root@cpe-fra-c2 ~>
root@cpe-fra-c2 ~>
root@cpe-fra-c2 ~> nft list chain inet firewallld filter_IN_public_deny
table inet firewallld {
    chain filter_IN_public_deny {
        ip saddr 103.186.28.56 tcp dport 22 ct state { new, untracked } reject with icmp port-unreachable
        ip saddr 95.214.27.115 tcp dport 110 ct state { new, untracked } reject with icmp port-unreachable
        ip saddr 95.214.27.115 tcp dport 143 ct state { new, untracked } reject with icmp port-unreachable
        ip saddr 95.214.27.115 tcp dport 993 ct state { new, untracked } reject with icmp port-unreachable
        ip saddr 95.214.27.115 tcp dport 995 ct state { new, untracked } reject with icmp port-unreachable
        ip saddr 143.198.146.129 tcp dport 22 ct state { new, untracked } reject with icmp port-unreachable
        ip saddr 79.110.49.26 tcp dport 110 ct state { new, untracked } reject with icmp port-unreachable
        ip saddr 79.110.49.26 tcp dport 143 ct state { new, untracked } reject with icmp port-unreachable
        ip saddr 79.110.49.26 tcp dport 993 ct state { new, untracked } reject with icmp port-unreachable
        ip saddr 79.110.49.26 tcp dport 995 ct state { new, untracked } reject with icmp port-unreachable
    }
}
root@cpe-fra-c2 ~>

```

Figure 1: Fail2Ban remembers the IP addresses of failed login attempts and sets up a firewall block for them.

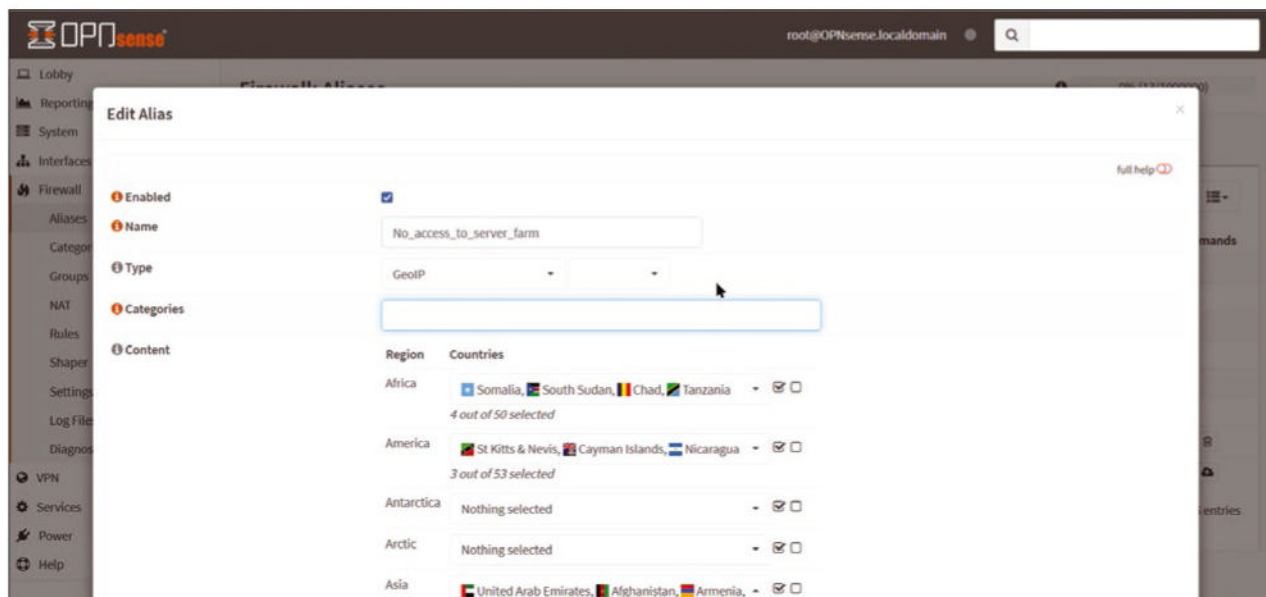


Figure 2: The free OPNsense firewall filters access by country of origin.

intervention (proof of work). In all cases, access to the web service is slowed down so that a DDoS attacker can make significantly fewer requests per second. The trick with web protection is that the client, not the server, is exposed to high load. The client receives the difficult task and the server only checks whether the result is correct.

This DDoS protection does not require you to tinker with the existing

HTML code. The web protector works as a benign man-in-the-middle and inserts its robot checks upstream of the actual web pages. Technically, this takes the form of a reverse proxy that receives and analyzes the web requests of all clients. Is this client submitting an unusually high number of requests? Then a web challenge is issued that human visitors have to answer.

The reverse proxy is a piece of software that accepts HTTP requests but has no content itself. For the response, the reverse proxy accesses another web server and delivers the HTML code as a proxy. As an intermediary in the data stream, the proxy can inject all sorts of things into the HTML lines, such as captchas.

Captchas with vDDoS

In addition to commercial providers with professional web protection, you can find free applications on GitHub, such as the vD-DoS software [7], which presents captchas to its clients. Every web visitor has probably stumbled across

this kind of task and had to click on the right photos from a selection (Figure 3). This activity is intended to distinguish human visitors from scripts.

The tool can be installed on Linux with just a few lines:

```
wget https://raw.githubusercontent.com/duy13/vDDoS-Protection/master/latest.sh
chmod +x latest.sh
./latest.sh
```

All components are then available under `/vddos` on the local filesystem. In the best open source manner, the developers of vDDoS do not code everything from scratch but use existing software with a free license: Nginx for the web server, Unicorn as the web gateway, and Flask as the Python framework. You need to describe your websites and the desired protection method in the configuration.

In Listing 1, vDDoS looks like an HTTP/S server (*Listen* column) with different levels of protection (*Security*) when viewed from the outside; it addresses several servers in the background (*Backend*). The starting signal is given by the `vddos start` command. vDDoS then takes off and offers the configured websites from its IP address – with built-in DDoS protection.

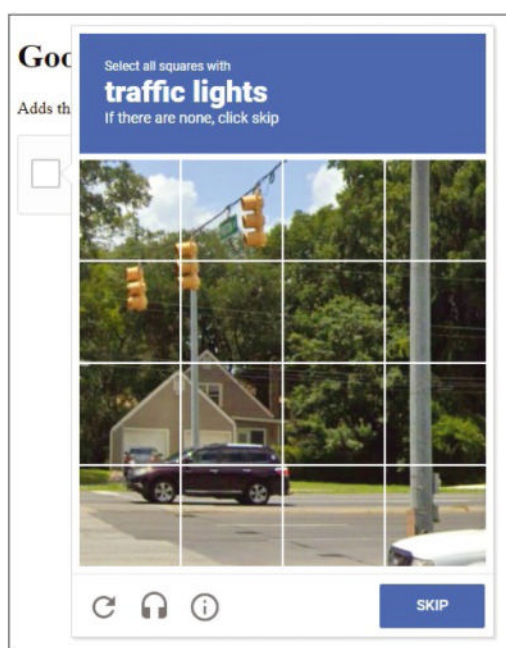


Figure 3: With the help of an image puzzle in vDDoS, the website checks whether a human or a script wants access.

Search Engines and Crawlers

Whichever DDoS protection you choose, you do not want to block the crawlers of the major search engines. If the snooping bot from Google, Microsoft, or DuckDuckGo encounters a JavaScript challenge or Fail2Ban strikes, it will simply stop indexing your site. Fortunately, the operators provide all the IP addresses under which their bots map out the Internet. You need to add these addresses to your personal whitelist to make sure that nothing stands in the way of the search bots.

Sensor with a Black Hole

If you have more than a few servers in your administration area or do not want to integrate any additional software, you can get DDoS protection “off the web,” which does not refer to another provider in the sense of as-a-service, but rather a monitor that detects unusually high traffic flows on switches and routers.

Routers count packets flowing through and report them in NetFlow format to a central NetFlow collector. The collector receives the statistical information from all routers and, by doing so, obtains a precise overall picture of the utilization of Internet links. This method works similarly with switches: They send a small percentage of the transported packets in sFlow format to the collector, which can then deduce the utilized bandwidth from the samples.

If the number of packets or the bytes transmitted per second exceeds a defined threshold value, then it looks like the start of a DDoS attack. Now is the time to act. The simplest case is an email alert. Automated systems use the NetFlow and sFlow data to identify the target

system under attack and inform the customer router over Border Gateway Protocol (BGP). The customer router immediately forwards the information to the provider. The BGP routers now have a new host route that no longer routes the traffic to the target server but instead dumps it in the bit trash can. A black hole is created in the routing table for the attacked server. As soon as the onslaught subsides, the guard removes the host route and makes the server visible again. The idea behind this is known as remotely triggered black hole (RTBH) filtering. If you want to familiarize yourself with this form of DDoS protection, I recommend the community version of FastNetMon [8]. For test purposes, FastNetMon also runs without BGP interaction and only triggers a Bash script in the event of a DDoS alarm. In this phase, you can monitor your network and set the threshold values. Once the false positives stop, the FastNetMon server can become a BGP neighbor in your autonomous system and send host routes.

Tools for Windows Servers

The tools presented so far are exclusively for Linux and Unix. If you want to make your Windows server accessible over the Internet, you can expand the Windows firewall by adding IPBan [9] or EvlWatcher [10] for DDoS protection. Both products work in exactly the same way as Fail2Ban. They monitor the login attempts and block the source address after a few failed attempts.

IPBan is particularly easy to set up: Download the ZIP archive from GitHub, unpack, and start the executable. In the open program window IPBan informs you what it is currently doing and which logins it has

detected. IPBan forwards the request for blocking to the Windows firewall. Although the software does not make a Windows system secure, it does reduce the attack surface on open services such as Remote Desktop or virtual network computing (VNC).

Test with a DoS Attack

Of course, you will want to test all your defenses. If you do not currently have a botnet in your arsenal, you should at least unleash a single client on your now protected servers. You do not need to access the Darknet to pick up the necessary tools; instead, try GitHub or the Kali Linux distribution. The legal framework must be established before the first attack: The customer or employer must consent to a deliberate attack. It also makes sense not to launch the attacks from your network to avoid ending up on one of the blacklists mentioned above. A disposable virtual machine (VM) from a cloud provider is recommended, and because they are paid for by the minute, the financial outlay is very manageable.

To lay a simple siege, use the Siege HTTP load tester, which can be installed with the package manager on many distributions. The software expects the URL of the web server as an argument and immediately starts throwing GET requests at the server. Without DDoS protection, the results on the screen flash by at breakneck speed. If protection is activated, activity stops after a few HTTP access attempts. Siege then simply reports *Resource temporarily unavailable*.

If you want to carry out your own attack with a little more finesse, you might want to use MHDDoS [11], which comes with more than 50 attack vectors that trick various commercial

Listing 1: vDDoS as a Reverse Proxy

Website	Listen	Backend	Cache	Security	SSL-Prikey	SSL-CRTkey
default	http://0.0.0.0:80	http://10.1.1.84:80	no	5s	no	no
www.example.net	https://0.0.0.0:443	http://10.1.1.84:80	no	307	/vddos/ssl/example.key	/vddos/ssl/example.crt
login.example.net	https://0.0.0.0:443	http://10.1.1.85:80	no	captcha	/vddos/ssl/example.key	/vddos/ssl/example.crt

DDoS defense systems, going far beyond a simple flood of GET and POST requests. Command-line arguments let you specify how many requests per second you want to throw at the server.

Commercial DDoS Protection

Numerous commercial providers also offer turnkey DDoS safeguards with support, logging, and alerting. These solutions range from cloud-based approaches to hardware appliances, which are usually integrated into the data center as firewall extensions. For example, industry giant Cloudflare offers DDoS web protection with its application services, which works exactly like vDDoS when viewed from the outside. Under the hood, a reverse proxy receives and examines the access requests and forwards the legitimate traffic to the server with the content. For this method to work, the DNS entry for your website must point to the Cloudflare server. Note that your web server is only allowed to accept requests from Cloudflare, otherwise the DDoS protection is ineffective.

The advantage is that Cloudflare and comparable offerings seem to have more bandwidth in reserve than the attackers have been able to muster

(thus far). Anyone expecting attacks in the multidigit gigabit per second range is well advised to use a commercial provider. Fun fact: This solution works so well that even the bad guys with their illegal web portals use it to protect themselves against even more evil villains.

Conclusions

In addition to ransomware attacks, DDoS attacks pose major challenges for companies. This foray through the defense arsenal from the open source world reveals a couple of highlights. The basic service begins with a hardened operating system followed by a security check, which is again followed by guard tools that keep an eye on logfiles, detect failed login attempts, and automatically create rules for the local firewall. Finally, websites can be protected against a flood of requests with various image puzzles in the form of captchas.

In larger environments, the DDoS sensor is located well away from the server farm and receives traffic information from routers and switches. If the throughput rates of individual clients are unusually high, the block is sent to the provider routers by a BGP update, and the game is over for the

attacker. If these tools and tricks involve too much manual work or you anticipate massive DDoS attacks, the same protections are also available from commercial providers. ■

Info

- [1] Linux hardening guide: [\[https://madaidans-insecurities.github.io/guides/linux-hardening.html\]](https://madaidans-insecurities.github.io/guides/linux-hardening.html)
 - [2] OpenSCAP security guide for RHEL 7: [\[https://static.open-scap.org/ssg-guides/ssg-rhel7-guide-C2S.html\]](https://static.open-scap.org/ssg-guides/ssg-rhel7-guide-C2S.html)
 - [3] Lynis: [\[https://cisofy.com/lynis/\]](https://cisofy.com/lynis/)
 - [4] FireHOL Cybercrime IP feeds: [\[https://iplists.firehol.org\]](https://iplists.firehol.org)
 - [5] Fail2Ban: [\[https://github.com/fail2ban/fail2ban\]](https://github.com/fail2ban/fail2ban)
 - [6] CrowdSec: [\[https://www.crowdsec.net\]](https://www.crowdsec.net)
 - [7] vDDoS: [\[https://vddos.voduy.com\]](https://vddos.voduy.com)
 - [8] FastNetMon: [\[https://github.com/pavel-odintsov/fastnetmon\]](https://github.com/pavel-odintsov/fastnetmon)
 - [9] IPBan: [\[https://github.com/digitalruby/ipban\]](https://github.com/digitalruby/ipban)
 - [10] EvlWatcher: [\[https://github.com/devnulli/EvlWatcher\]](https://github.com/devnulli/EvlWatcher)
 - [11] MHDDoS: [\[https://github.com/MatrixTM/MHDDoS\]](https://github.com/MatrixTM/MHDDoS)
-

The Author

Markus Stubbig is a networking engineer who has worked in the IT industry for 20 years. His strong focus is on design and implementation of campus networks around the world..

Automatically terminate OpenSSH sessions

The Clock Is Ticking

Disconnect OpenSSH user sessions after a certain period of inactivity with the systemd-logind service. By Thorsten Scherf

When configuring a system, a large number of settings are required to meet compliance requirements. Common Criteria [1] is an international standard for the security certification of computer systems. The standard defines the requirements as security targets. Targets look different depending on the system you are using. For example, the requirements for a mobile device differ from those for a desktop system, which explains why protection profiles are different. The Protection Profile for general-purpose operating systems [2] clearly stipulates that user sessions must either be terminated or, alternatively, locked after a certain period of inactivity. However, recent OpenSSH versions block a workaround frequently used to meet this requirement. We show you how to use the systemd-logind service to solve this dilemma.

Compliance Undermined

The US Department of Defense (DOD) Defense Information Systems Agency (DISA) Security Technical Implementation Guides (STIGs) [3] also stipulate these requirements for operating systems. The Guide for Red Hat Enterprise Linux 8 [4] proposes implementing these rules with specific configurations of the OpenSSH service. Two statements, `ClientAliveInterval` and `ClientAliveCountMax`, are intended to help meet the compliance requirements:

```
grep -i clientalive /etc/ssh/sshd_config
ClientAliveInterval 600
ClientAliveCountMax 0
```

Once you have made these changes to your OpenSSH configuration, an SSH connection to this system will be

disconnected after 10 minutes of inactivity, in exactly the way required by the Common Criteria and DISA STIGs. The problem is, though, that these two options were intended for a completely different purpose – checking the SSH connection itself – and not for user session activity. It was only possible to terminate the session at all by setting the `ClientAliveCountMax=0` option in combination with an arbitrary value for `ClientAliveInterval`, even if a user was inactive and even if the connection itself was intact. The positive result is merely a lucky side effect and was never the intended way for this to work.

This “misbehavior” of the software was fixed in the OpenSSH upstream version 8.2 at the end of 2020 [5]. Unfortunately, it also ruled out the option of terminating an SSH connection when a user is inactive. This new behavior is particularly annoying in environments where the systems need to meet specific compliance requirements.

The OpenSSH upstream community has long seen feature requests [6] [7] for the implementation of a configuration option that lets the SSH server terminate inactive sessions. However, these requests have thus far been rejected. One of the reasons is that most shells support the TMOUT environment variable, which lets you set a timeout for user input. That said, the approach is fraught with a number of disadvantages and is easy to work around [8].

systemd-logind to the Rescue

After the upstream changes slowly made their way into the various Linux distributions, the outcry from users was massive, of course. After all, Linux distributions such as Red Hat Enterprise Linux or SUSE Linux Enterprise Server are used by corporations in compliance-critical environments. Because the OpenSSH upstream community was not really willing to address the problem, alternative solutions were sought. The result now available is quite obvious when you think about it and is based on the `systemd-logind` service [9]. This service is explicitly designed to monitor users and their sessions and can detect the idle state of user sessions, enabled with the use of a separate PAM module, `pam_systemd` [10]. This module takes care of registering a user's session with the `systemd-logind` service after login, which in turn, creates a separate `systemd` slice unit for each new user and a scope unit each for any sessions belonging to the same user and running in parallel. A patch [11] was released at the end of 2022 to extend the `systemd-logind` service. Armed with the patch, you can now pass in the new `StopIdleSessionSec` configuration option to the service, which ensures that a user's session ends as soon as `systemd-logind` detects that the session has been inactive

for longer than allowed. For example, if you want inactive user sessions to expire automatically after 10 minutes, you would use a value of 600 with the new option:

```
# grep StopIdleSessionSec 2
/etc/systemd/logind.conf
StopIdleSessionSec=600
```

After making these changes, remember to restart the service by typing:

```
# systemctl restart systemd-logind
```

For test purposes, set the value to 10 seconds and log in to this system again with SSH. The command

```
journalctl -u systemd-logind
```

reads and filters the system journal and shows how the user's inactive session is automatically terminated after 10 seconds (Listing 1).

Conclusions

Typical compliance requirements stipulate that inactive user sessions must either be terminated or alternatively blocked. Until recently, you could use some of the OpenSSH service's own configuration options to do this. However, this behavior was deliberately changed in recent versions of the software without providing an alternative approach to terminating inactive SSH connections. To remedy this shortcoming, the missing function has now been added to the `systemd-logind` `systemd` service. This option lets admins define an interval after which inactive user sessions are automatically terminated.

Info

- [1] Common Criteria: [\[https://www.commoncriteriaportal.org/index.cfm\]](https://www.commoncriteriaportal.org/index.cfm)
- [2] Protection profile for general-purpose operating systems: [\[https://www.niap-ccevs.org/MMO/PP/-442/-#FMT_SMF_EXT.1.1\]](https://www.niap-ccevs.org/MMO/PP/-442/-#FMT_SMF_EXT.1.1)
- [3] STIGViewer: [\[https://www.stigviewer.com\]](https://www.stigviewer.com)
- [4] RHEL8 STIG: [\[https://www.stigviewer.com/stig/red_hat_enterprise_linux_8/2021-12-03/finding/V-230244\]](https://www.stigviewer.com/stig/red_hat_enterprise_linux_8/2021-12-03/finding/V-230244)
- [5] OpenSSH patch: [\[https://github.com/openssh/openssh-portable/commit/69334996ae203c51c70bf01d414c918a44618f8e\]](https://github.com/openssh/openssh-portable/commit/69334996ae203c51c70bf01d414c918a44618f8e)
- [6] OpenSSH-RFE 1: [\[https://bugzilla.mindrot.org/show_bug.cgi?id=3362\]](https://bugzilla.mindrot.org/show_bug.cgi?id=3362)
- [7] OpenSSH-RFE 2: [\[https://bugzilla.mindrot.org/show_bug.cgi?id=1338\]](https://bugzilla.mindrot.org/show_bug.cgi?id=1338)
- [8] Stackoverflow article on TMOUT: [\[https://stackoverflow.com/questions/17397069/unset-readonly-variable-in-bash/54705440#54705440\]](https://stackoverflow.com/questions/17397069/unset-readonly-variable-in-bash/54705440#54705440)
- [9] `systemd-logind`: [\[https://www.freedesktop.org/software/systemd/man/latest/systemd-logind.service.html\]](https://www.freedesktop.org/software/systemd/man/latest/systemd-logind.service.html)
- [10] `pam_systemd` PAM module: [\[https://www.freedesktop.org/software/systemd/man/latest/pam_systemd.html\]](https://www.freedesktop.org/software/systemd/man/latest/pam_systemd.html)
- [11] `systemd-logind` idle patch: [\[https://github.com/redhat-plumbers/systemd-rhel8/pull/332\]](https://github.com/redhat-plumbers/systemd-rhel8/pull/332)

The Author

Thorsten Scherf is the global Product Lead for Identity Management and Platform Security in Red Hat's Product Experience group. He is a regular speaker at various international conferences and writes a lot about open source software.



Listing 1: Terminated Session

```
Mar 7 05:06:07 kvm-04-guest19 systemd-logind[46596]: New session 5 of user root.
Mar 7 05:06:17 kvm-04-guest19 systemd[1]: session-5.scope: Killing process 46282 (sshd) with signal SIGTERM.
Mar 7 05:06:17 kvm-04-guest19 systemd[1]: session-5.scope: Killing process 46285 (sshd) with signal SIGTERM.
Mar 7 05:06:17 kvm-04-guest19 systemd[1]: session-5.scope: Killing process 46286 (bash) with signal SIGTERM.
Mar 7 05:06:17 kvm-04-guest19 systemd[1]: Stopping Session 5 of user root.
Mar 7 05:06:17 kvm-04-guest19 systemd[1]: session-5.scope: Succeeded.
Mar 7 05:06:17 kvm-04-guest19 systemd[1]: Stopped Session 5 of user root.
Mar 7 05:06:17 kvm-04-guest19 systemd-logind[46596]: Removed session 5.
```




An army of Xeon cores to do your bidding

44 Cores

Explore low-cost parallel computing.

By Federico Lucifredi

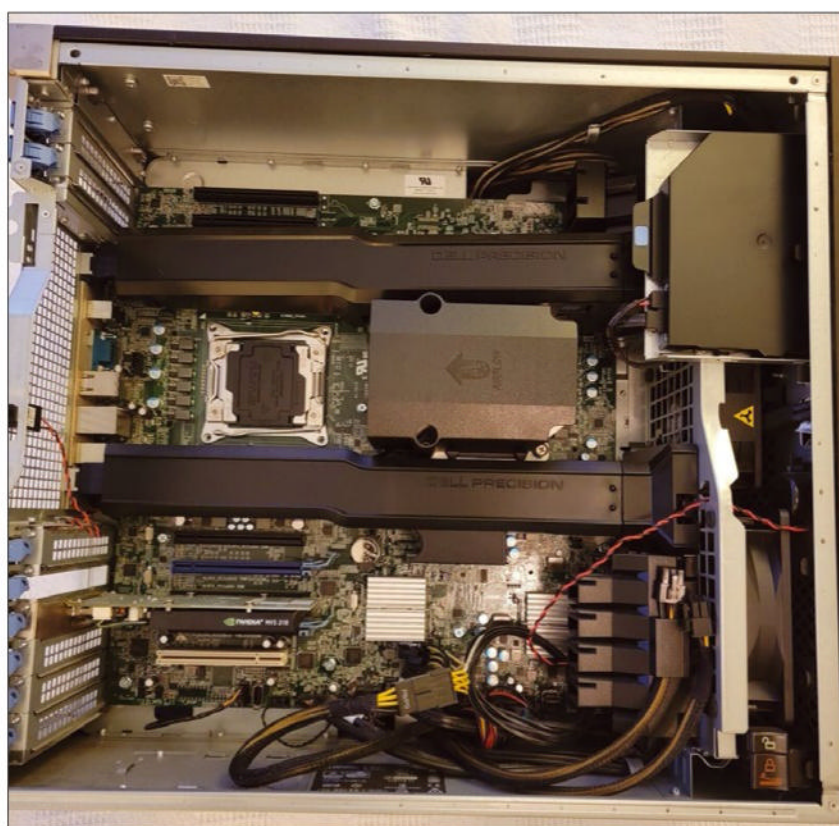


Figure 1: Inside view of the system before the build-out. Note how sockets are protected by plastic shielding plates.

Table 1: Shockwave Compute Server Specs

Component	Spec
Chassis and motherboard	Dell Precision Workstation T7910
Power	1,300W
CPU	2x Intel Xeon Gold E5-2699 V4, 22 cores, 2.4GHz, 55MB of cache, LGA 2011-3
GPU, NPU	n/a*
Memory	256TB DDR4 ECC PC4-19200 2,400MHz
Storage	4x3.5-inch drive bays, slimline optical drive, LSI SAS 3008 12Gbps SAS (6Gbps SATA)
Networking	Intel I217 and I210 Gigabit Ethernet controllers, remote wake-up
Video	NVIDIA Quadro HDMI, DP
*Not applicable.	

Lead Image © Lucy Baldwin, 123RF.com

I continuously explore options for cost-effective (or just plain cheap) parallel computing rigs, and while next-generation, cutting edge hardware is always interesting, I find that retired options from yesteryear can also show potential when their significantly lower cost is part of the overall assessment. Retrofitting a retired Dell workstation with high-core-count CPUs and the maximum allowable RAM, I built a 44-core compute behemoth for less than \$600 to run Monte Carlo [1] simulations. Let me dive into the details!

Bill of Materials

Table 1 details my hardware configuration. I found a refurbished Dell Precision T7190 workstation [2] on eBay in nearly perfect cosmetic condition with a motherboard sporting two LGA 2011-3 processor sockets – which were both vacant (Figure 1). The stock power supply is rated at 1,300W, more than sufficient for this project, but alas, one of the CPU heat sinks was missing. The description promised no ventilation shrouds or disks, but the unit came with four hard disks, one DVD-ROM drive, and all the air shrouds, making this a happy purchase (\$159). After temporarily installing a 10-core Xeon from the parts archive and flashing the BIOS to its latest revision with Dell's very reasonable bootable tooling [3] [4], I was able to install two newly procured CPUs, which are Intel Xeon Gold E5-2699 v4 CPUs running at 2.4GHz and each sporting 22 cores and 55MB of cache memory [5]. Fortunately,

Fixing an LGA 2011-3 Socket

Identifying the issue with the CPU socket required some investigative work of the bisection variety: installing one CPU, then installing the other (both worked), then installing half of the RAM, then the other half (the second test failed), then continuing to divide this failed half until I identified the pair of DIMMs that were not working. However, the DIMMs themselves were working (swapped with another pair). Connecting this picture back to the CPU pin was fortuitous: As I was re-seating a heat sink, I noticed some thermal paste out of place, and when I removed the CPU, I found thermal paste in the socket – not a good thing, even when things are working. I washed the thermal paste out with 70 percent isopropyl alcohol loaded in a Waterpik-type device I sourced on AliExpress for \$20. Another \$20 went to Amazon for a handheld USB microscope [12] to examine the damaged area (Figure 2). Patient use of an index card and tweezers enabled me to rectify the failure. The bent pin controlled the affected DIMM banks.

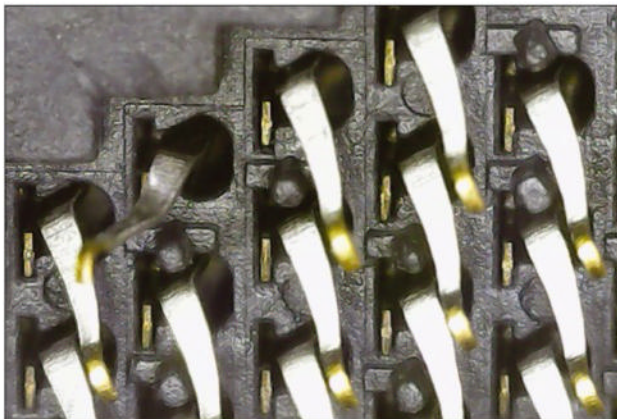


Figure 2: The bent CPU socket pin as seen under the microscope.

I had a second heat sink and fan on hand (Buying a new one would have cost nearly as much as the workstation itself!). This purchase set me back \$250 for two CPUs, which were engineering samples (ES) verified to run validation tests reliably at speed. Unfortunately, the second socket also came with a bent pin, which sent me on a two-week wild-side quest troubleshooting the CPUs and its memory banks until I located it, cleaned it, and *very delicately and patiently* bent it back into its original position. (See the “Fixing an LGA 2011-3 Socket” box.)

Total Recall

Sixteen empty DIMM memory slots stared at me asking for attention. Raiding my lab’s archive, I found eight perfectly suitable modules already in my possession (16GB DDR4 error correction code (ECC) PC4-19200 2,400MHz, exceeding spec). A little bargain hunting led me to find another eight modules on Amazon (\$182 in total) with equivalent specs manufactured by SK Hynix [6]. Collectively, the 16 modules combine to provide 256GB of RAM, half of the maximum that could be installed without resorting to more expensive load-reduced (LR)DIMM, which in turn maxes out at 1TB. The current design provides almost 6GB of RAM per core, and I retain the option to budget another \$2,000 to quadruple that amount if a workload is found needing it – a very reasonable compromise.

I completed the setup with this newfangled technology called software – Ubuntu 23.10 “Mantic Minotaur” providing the operating system layer, with the distribution including all the necessary message-passing interface (MPI) and parallel processing tools one may want. The btop [7]

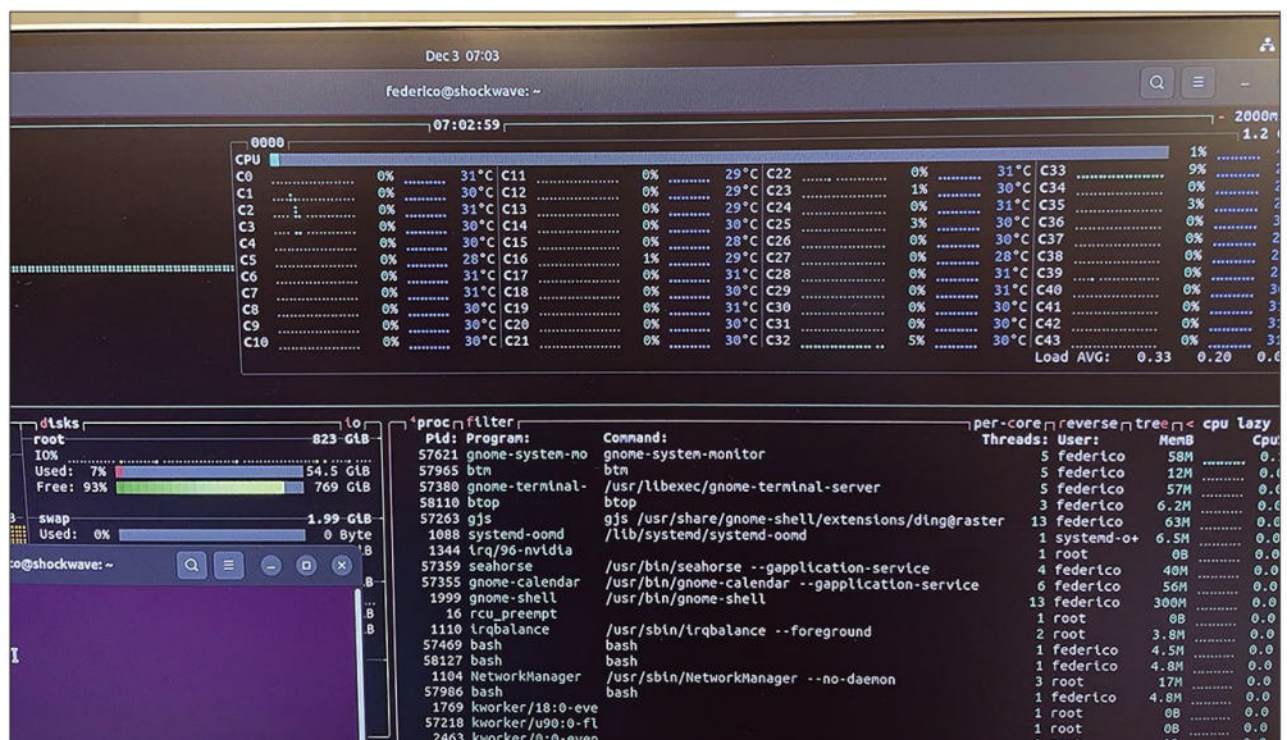


Figure 3: Forty-four cores humming along, but why are numbers 32 and 33 doing all the work?

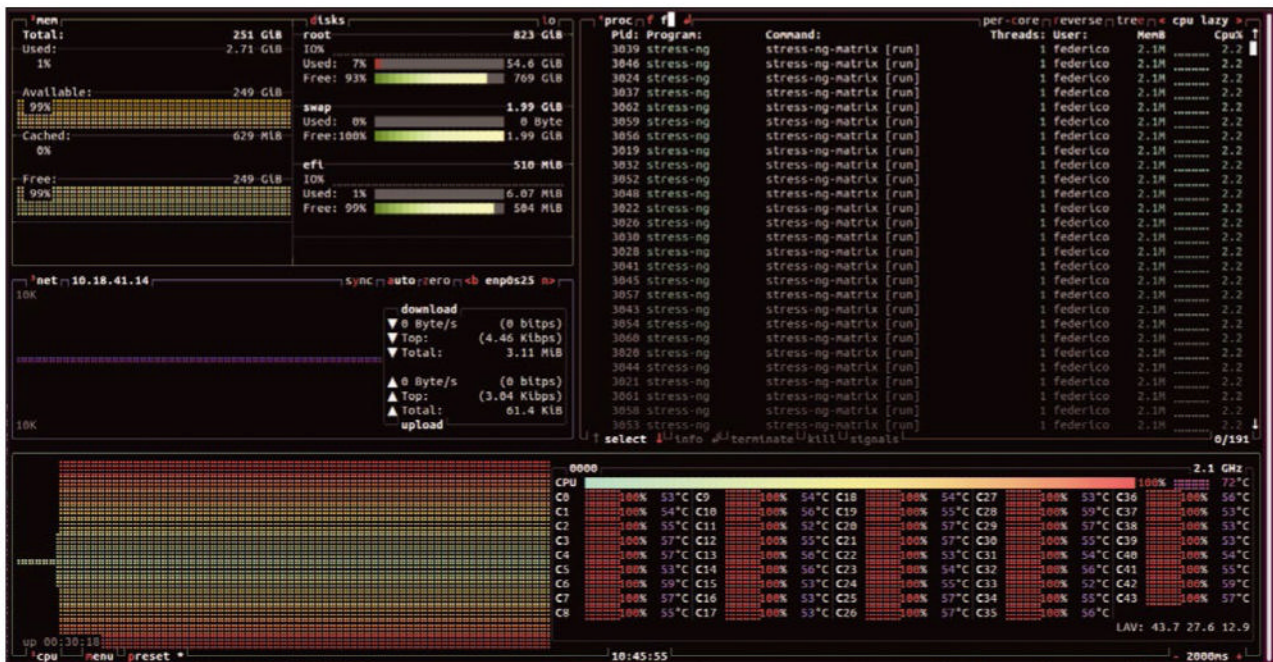


Figure 4: Turning the system into a space heater with the right benchmark.

tool is everyone's new favorite in-terminal system monitor, and it provides a first look at the completed system (Figure 3). Note the inclusion of core temperatures.

I already discussed BashTop in a previous issue [8], but today I shall focus on just one aspect of its CPU view: What happens when all those cores heat up? The system idles at 121W once booted up, so I will drive up the temperature with the matrix option of the stress-ng tool [9], an ominous stressor known for its Intel CPU heating properties [10]:

```
stress-ng --matrix 0 -t 1m --tz --times
```

The zero count syntax requests one stressor to run on each CPU core in the system: `--times` generates statistics on userland and kernel time, and the `--tz` option includes CPU temperature data where available. The CPU clock ran up from a cozy 1.2GHz to 2.1GHz across the board, with all cores pegged at 100 percent, eventually reaching a perfect 44 load average [11] (Figure 4). Temperature did not exceed 72C at the hottest sensor (good cooling on Dell's part), but the power draw tripled, rising to 367W (Figure 5). The power-hungry nature of the beast needs to factor in any

cost calculation as much as, if not more than, the hardware cost itself. The 22 cores (44 threads) of each CPU could turbo boost up to 3.6GHz

individually, but it is more interesting in this case to note that 2.4GHz is the maximum speed they can all accelerate to concurrently.



Figure 5: Tripled power draw. The electric bill is the largest expense for this system.

At the time of this writing, an Amazon AWS m7g.12xlarge instance with 48 virtual cores and *only* 192GB of RAM will cost almost \$2/hr (\$1.9584, US East, on-demand pricing), so you could think of this new machine as costing 12-1/2 days (300 hours) of AWS compute. Not bad! ■

Info

- [1] The Monte Carlo method: [\[https://en.wikipedia.org/wiki/Monte_Carlo_method\]](https://en.wikipedia.org/wiki/Monte_Carlo_method)
- [2] Dell Precision T7910 workstation: [\[https://i.dell.com/sites/doccontent/shared-content/data-sheets/en/Documents/Dell-Precision-Tower-7000-Series-7910-Spec-Sheet.pdf\]](https://i.dell.com/sites/doccontent/shared-content/data-sheets/en/Documents/Dell-Precision-Tower-7000-Series-7910-Spec-Sheet.pdf)
- [3] Dell BIOS updates: [\[https://www.dell.com/support/kbdoc/en-us/00012421/dell-bios-updates\]](https://www.dell.com/support/kbdoc/en-us/00012421/dell-bios-updates)
- [4] Dell bootable DDDP: [\[https://www.dell.com/support/kbdoc/en-us/000145519/how-to-create-a-bootable-usb-flash-drive-using-dell-diagnostic-deployment-package-dddp\]](https://www.dell.com/support/kbdoc/en-us/000145519/how-to-create-a-bootable-usb-flash-drive-using-dell-diagnostic-deployment-package-dddp)
- [5] Intel Ark: Xeon E5-2699 v4: [\[https://ark.intel.com/content/www/us/en/ark/products/91317/intel-xeon-processor-e5-2699-v4-55m-cache-2-20-ghz.html\]](https://ark.intel.com/content/www/us/en/ark/products/91317/intel-xeon-processor-e5-2699-v4-55m-cache-2-20-ghz.html)
- [6] Hynix 16GB DDR4 PC4-19200 2,400MHz ECC REG DIMM: [\[https://www.amazon.com/gp/product/B01N60511Z/\]](https://www.amazon.com/gp/product/B01N60511Z/)
- [7] btm(1) man page: [\[https://manpages.ubuntu.com/manpages/noble/en/man1/btop.1.html\]](https://manpages.ubuntu.com/manpages/noble/en/man1/btop.1.html)
- [8] “Next-generation terminal UI tools” by Federico Lucifredi, *ADMIN*, issue 64, 2021, [\[https://www.admin-magazine.com/Archive/2021/64/Next-generation-terminal-UI-tools\]](https://www.admin-magazine.com/Archive/2021/64/Next-generation-terminal-UI-tools)
- [9] stress-ng by Colin King: [\[https://manpages.ubuntu.com/manpages/jammy/man1/stress-ng.1.html\]](https://manpages.ubuntu.com/manpages/jammy/man1/stress-ng.1.html)
- [10] “Creating load for fun and profit” by Federico Lucifredi, *ADMIN*, issue 75, 2023, [\[https://www.admin-magazine.com/Archive/2023/75/Creating-load-for-fun-and-profit\]](https://www.admin-magazine.com/Archive/2023/75/Creating-load-for-fun-and-profit)
- [11] “Law of Averages” by Federico Lucifredi, *ADMIN*, issue 11, 2012
- [12] Low-cost USB microscope: [\[https://www.amazon.com/gp/product/B06WD843ZM\]](https://www.amazon.com/gp/product/B06WD843ZM)

Author

Federico Lucifredi (@0xf2) is the Product Management Director for Ceph Storage at Red Hat and IBM, formerly the Ubuntu Server Product Manager at Canonical, and the Linux “Systems Management Czar” at SUSE. He enjoys arcane hardware issues and shell-scripting mysteries, and takes his McFlurry shaken, not stirred. You can read more from him in the new O'Reilly title *AWS System Administration*.



Linux Magazine Subscription

Print and digital options
12 issues per year

► **SUBSCRIBE**
shop.linuxnewmedia.com

Expand your Linux skills:

- In-depth articles on trending topics, including Bitcoin, ransomware, cloud computing, and more!
- Troubleshooting and optimization tips
- News on crucial developments in the world of open source
- How-tos and tutorials on useful tools that will save you time and protect your data
- Cool projects for Raspberry Pi, Arduino, and other maker-board systems

Go farther and do more with Linux, subscribe today and never miss another issue!

Need more Linux?
Subscribe free to Linux Update

Our free Linux Update newsletter delivers insightful articles and tech tips to your inbox every week.

bit.ly/Linux-Update



ADMIN

Network & Security

NEWSSTAND

Order online:
<https://bit.ly/ADMIN-library>

ADMIN is your source for technical solutions to real-world problems. Every issue is packed with practical articles on the topics you need, such as: security, cloud computing, DevOps, HPC, storage, and more! Explore our full catalog of back issues for specific topics or to complete your collection.

#78 - November/December 2023

Domain-Driven Design

Business experts and developers collaborate to define domain models and business patterns that guide software development.

On the DVD: Fedora Server 39



#77 - September/October 2023

Secure CI/CD Pipelines

DevSecOps blends security into every step of the software development cycle.

On the DVD: IPFire 2.27

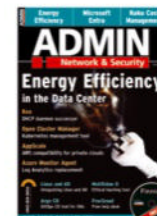


#76 - July/August 2023

Energy Efficiency

The storage share of the total data center energy budget is expected to double by 2030, calling for more effective resource utilization.

On the DVD: Finnix 125 (Live boot)



#75 - May/June 2023

Teamwork

Groupware, collaboration frameworks, chat servers, and a web app package manager allow your teams to exchange knowledge and collaborate on projects in a secure environment.

On the DVD: Ubuntu 23.04 "Lunar Lobster" Server Edition

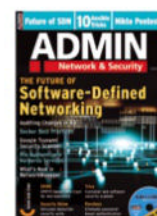


#74 - March/April 2023

The Future of Software-Defined Networking

New projects out of the Open Networking Foundation provide a glimpse into the 5G network future, most likely software based and independent of proprietary hardware.

On the DVD: Kali Linux 2022.4

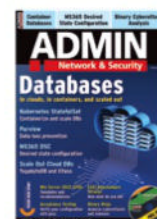


#73 - January/February 2023

Databases

Cloud databases can be useful in virtually any conceivable deployment scenario, come in SQL and NoSQL flavors, and harmonize well with virtualized and containerized environments.

On the DVD: Manjaro 22.0 Gnome



WRITE FOR US

Admin: Network and Security is looking for good, practical articles on system administration topics. We love to hear from IT professionals who have discovered innovative tools or techniques for solving real-world problems.

Tell us about your favorite:

- interoperability solutions
- practical tools for cloud environments
- security problems and how you solved them
- ingenious custom scripts

- unheralded open source utilities
- Windows networking techniques that aren't explained (or aren't explained well) in the standard documentation.

We need concrete, fully developed solutions: installation steps, configuration files, examples – we are looking for a complete discussion, not just a “hot tip” that leaves the details to the reader.

If you have an idea for an article, send a 1-2 paragraph proposal describing your topic to: edit@admin-magazine.com.



Contact Info

Editor in Chief

Joe Casad, jcasad@linuxnewmedia.com

Managing Editors

Rita L Sooby, rsooby@linuxnewmedia.com
Lori White, lwhite@linuxnewmedia.com

Senior Editor

Ken Hess

Localization & Translation

Ian Travis

News Editor

Amber Ankerholz

Copy Editors

Amy Pettle, Aubrey Vaughn

Layout

Dena Friesen, Lori White

Cover Design

Dena Friesen, Illustration based on graphics by rendeep Lumia, 123RF.com

Advertising

Brian Osborn, bosborn@linuxnewmedia.com
phone +49 8093 7779420

Publisher

Brian Osborn

Marketing Communications

Gwen Clark, gclark@linuxnewmedia.com
Linux New Media USA, LLC
4840 Bob Billings Parkway, Ste 104
Lawrence, KS 66049 USA

Customer Service / Subscription

For USA and Canada:
Email: cs@linuxnewmedia.com
Phone: 1-866-247-2802
(Toll Free from the US and Canada)

For all other countries:
Email: subs@linuxnewmedia.com
www.admin-magazine.com

While every care has been taken in the content of the magazine, the publishers cannot be held responsible for the accuracy of the information contained within it or any consequences arising from the use of it. The use of the DVD provided with the magazine or any material provided on it is at your own risk.

Copyright and Trademarks © 2024 Linux New Media USA, LLC.

No material may be reproduced in any form whatsoever in whole or in part without the written permission of the publishers. It is assumed that all correspondence sent, for example, letters, email, faxes, photographs, articles, drawings, are supplied for publication or license to third parties on a non-exclusive worldwide basis by Linux New Media unless otherwise stated in writing.

All brand or product names are trademarks of their respective owners. Contact us if we haven't credited your copyright; we will always correct any oversight.

Printed in Nuremberg, Germany by Kolibri Druck. Distributed by Seymour Distribution Ltd, United Kingdom

ADMIN (Print ISSN: 2045-0702, Online ISSN: 2831-9583, USPS No: 347-931) is published bimonthly by Linux New Media USA, LLC, and distributed in the USA by Asendia USA, 701 Ashland Ave, Folcroft PA. January/February 2024. Application to Mail at Periodicals Postage Prices is pending at Philadelphia, PA and additional mailing offices. POSTMASTER: send address changes to Linux Magazine, 4840 Bob Billings Parkway, Ste 104, Lawrence, KS 66049, USA.

Represented in Europe and other territories by: Sparkhaus Media GmbH, Bialasstr. 1a, 85625 Glonn, Germany.

Authors

Amber Ankerholz	6
Joe Casad	28
Ken Hess	3
Thomas Joos	14, 56, 60
Rubén Llorente	34
Martin Gerhard Loschwitz	40, 46
Federico Lucifredi	93
Benjamin Pfister	80
Dr. Holger Reibold	10, 24, 70
Thorsten Scherf	91
Tim Schürmann	34
Artur Skura	74
Evgenij Smirnov	19, 64
Andreas Stolzenberger	52
Markus Stubbig	86



PyCon US · 24

May 15th - 23rd Pittsburgh, PA



PyCon US is the largest annual gathering for the community that uses and develops the open-source Python programming language. We'll have talks, tutorials, social events, and more for all levels and uses of Python. We'd love for you to join us!



Scan me!

REGISTER AT
[HTTPS://US.PYCON.ORG/2024](https://us.pycon.org/2024)



PYCON US 2024 IS A PRODUCTION OF THE PYTHON SOFTWARE FOUNDATION

BY THE COMMUNITY

FOR THE COMMUNITY

HETZNER

MANAGED SERVER MANAGED SUCCESS



WE'LL TAKE CARE OF ADMINISTRATION
SO YOU CAN FOCUS ON YOUR BUSINESS.

MANAGED SERVER MA80

- ✓ AMD Ryzen™ 7 3700X
Octa-Core Matisse (Zen2)
- ✓ 64 GB DDR4 ECC RAM
- ✓ 2 x 512 GB NVMe SSD
- ✓ Unlimited traffic
- ✓ Location Germany
- ✓ No minimum contract
- ✓ Setup fee € 79.00

monthly from **€ 79.00**
\$ 86.70

DISCOVER ALL NEW MANAGED SERVERS

Start your project now:
htznr.li/admin/managed-server



OR SCAN
THE CODE

All prices exclude VAT and are subject to the terms and conditions of Hetzner Online GmbH. Prices are subject to change. All rights reserved by the respective manufacturers.

www.hetzner.com