

FREE
DVD

libiotrace

Kubernetes

ADMIN
Network & Security



ADMIN

Network & Security

ISSUE 71

Kubernetes

Scale up and stay safe with a
containerized environment

Open Policy Agent

A flexible way to
manage user rights

Rancher

A Kubernetes
management platform

Azure AD Guests

Manage guest accounts
with Access Reviews

microVMs

Lightweight VMs with a
reduced resource footprint



LINUX NEW MEDIA
The Pulse of Open Source

VXLAN

Overlay networks in
virtualized data centers

DNS over HTTPS

Advantages and
disadvantages

libiotrace

A ptrace-based tracing
mechanism for syscalls

QUIC

UDP-based protocol with
mandatory TLS encryption

SQL Server
2022 and
Azure

Register by October 14 for Big Discounts!



You. Dallas. SC22.

NOVEMBER 13–18

One of the world's largest gatherings of HPC researchers and professionals, SC22 is an incredible week of learning new skills, forging new friendships, solving complex problems, and viewing next-gen technology.

A new, custom-built Digital Experience is available if you prefer to attend virtually.



The International Conference for High Performance
Computing, Networking, Storage, and Analysis

Register Today!

sc22.supercomputing.org



SC22

Dallas, TX | **hpc**
accelerates.

Sponsored by:



IEEE
COMPUTER
SOCIETY



sig_{hpc}

Risky Business

For all the fear, uncertainty, and doubt still surrounding cloud technologies, you must take a level of acceptable risk to move forward.

I find it infinitely frustrating to hear a technology person complain about cloud computing. They usually state security as the biggest hurdle to their adoption. I agree that security is a major problem with cloud technologies, but it's no more of a problem than with any Internet-exposed service. Every service that "faces" the Internet is less secure, but you must accept certain risks to be able to function and move forward in business and your personal life. The cloud is a tool and an asset, but many of you probably disagree.

The cloud, for me, is a means of getting things done no matter where I am or which device I'm using. Cloud applications allow me to work on any device, even a borrowed one, without having to reset my entire environment or get used to someone else's configurations and settings. Before having cloud applications to manage my passwords, infrastructure, backups, contacts, mobile applications, and creative application suite, I had to carry around a fully loaded laptop computer with a local password manager or text file full of credentials rather than a small netbook or Chromebook containing little more than an operating system.

A few years ago, my wife bought me a Chromebook. Being an experimental type of person, I decided to use only that Chromebook, and nothing else, for one month to see if someone could truly be 100 percent reliant on it and the cloud for everything. I'm happy to report that it worked. I edited documents and created and edited images, podcasts, and video files on the Chromebook with web-based and cloud-based applications. I also had the peace of mind of knowing that if I lost the laptop, my information would be safe. The opposite would be true with a standard laptop. I prefer to "travel light," and web-based and cloud-based applications allow me to do so with confidence.

Sure, one of the downsides of operating purely in the cloud means you must make some sacrifices, such as having limited email-only access to support or giving up control to remote support technicians who may or may not have a native grasp of the English language. However, these limitations are something you learn to accept. A friend of mine referred to these constraints as "trade-offs." The trade-offs for convenience are lackluster support and a lack of local control. An additional potential downside, some claim, is that my data is less secure in the cloud than if it were behind a corporate firewall or saved locally on my laptop. Yes, some cloud services have been compromised. Still, if you take precautions, such as selecting providers that encrypt your data and enabling multifactor authentication, you have less to worry about. I'm not telling you that cloud-based services are fire-proof because of encryption and multifactor authentication; however, they're big steps toward theft prevention that I might not enjoy otherwise. A virtual private network (VPN) is a good thing. Encryption is a good thing. Multi-factor authentication is a good thing. Firewalls are good things. Unique, complex passwords are good things. Nothing is perfect, but using multiple layers of security means you're less likely to be affected by a negative event such as a data breach.

Cloud applications aren't for everyone. I get that. Sensitive government information, trade secrets, and medical records might not be cloud-ready, but for almost everything else, it's time to migrate to the cloud. Have backups and geographically diverse disaster recovery available in the rare case that your cloud provider goes offline or experiences a breach. In theory, a failure should never happen, but, as you know, it does. Remember that you keep flashlights, candles, and spare tires to soften the blow of failures, and you should do the same whether you're a cloud convert or not. An ounce of prevention is worth a pound of cure. Even old Ben Franklin knew that backups were a good idea, but Ben also knew that to do anything requires accepting some risk. Think of how he proved the electrical nature of lighting – by flying his kite in the clouds.

Ken Hess • ADMIN Senior Editor

ADMIN

Network & Security



Features

We show you how to get started with Kubernetes, and users share their insights into the container manager.

- 10 Introduction to Kubernetes**
Many admins find themselves struggling to get started with Kubernetes. We present the basic architecture and the most important components and terms.
- 18 The Kubernetes Experience**
Users in corporate and government agencies that have successfully switched to Kubernetes share their positive experiences and the stumbling blocks to avoid.



News

Find out about the latest plays and toys in the world of information technology.

- 8 News**
- CIQ secures VC funding, forms new leadership team
 - Learn DevSecOps basics through free training course
 - GitLab survey reflects shifting roles in DevSecOps
 - New report examines the current state of ITOps and SecOps
 - Top universities lack proper email security measures

Tools

Save time and simplify your workday with these useful tools for real-world systems administration.

- 22 Ceph 17.2**
A robust update offers increased stability and more features with little overhead.
- 28 libiotrace**
This library monitors running, static, and dynamically linked programs and collects detailed data for many file-I/O-related function calls.
- 36 Open Policy Agent**
This open source policy engine offers a flexible way to manage user rights, especially in the challenging environments of the cloud and infrastructure as code.
- 42 SQL Server 2022**
The focus is on closer collaboration between on-premises SQL servers and SQL functions in Azure, including availability and data analysis.



- 46 migrate2rocky**
The migrate2rocky script automatically migrates your CentOS 8 system to Rocky Linux - an enterprise RHEL derivative created by CentOS co-founder Greg Kurtzer.

Containers and Virtualization

Virtual environments are becoming faster, more secure, and easier to set up and use. Check out these tools.

- 50 Rancher 2.5**
An agile alternative to Red Hat OpenShift, Rancher is an efficient way to manage Kubernetes clusters, and the setup is significantly different from classic Kubernetes.



- 56 VXLAN**
VXLAN addresses the need for overlay networks within virtualized data centers accommodating multiple tenants.

Security

Use these powerful security tools to protect your network and keep intruders out in the cold.

- 60 Detect Signs of Attacks**
Deal with threat intelligence on the corporate network when the existing security tools are not effective.
- 64 DNS over HTTPS**
Now that web content is encrypted by HTTPS, the underlying name resolution is often unprotected. We look at the classic DNS protocol and investigate whether DNS over HTTPS could be the solution to ensure the confidentiality of DNS requests.



22 Ceph 17.2
Ceph developers got rid of some historic clutter and added new features for improved performance and built-in automation.



78 microVMs
You can have your cake and eat it, too: Ignite virtual machine manager combines Firecracker with Docker and OCI images to provide the best from both the container and VM worlds.

Management

Use these practical apps to extend, simplify, and automate routine admin tasks.

66 Azure AD Guest
Cross-tenant access settings and user-friendly Access Reviews simplify the management of guest accounts in Azure Active Directory.



71 Roll Back Windows 11
If you want to try Windows 11 but keep your options open, we show you how to install it, restore it to its factory settings if it's misbehaving, or roll it back to Windows 10.



Service

- 3 Welcome**
- 6 On the DVD**
- 97 Back Issues**
- 98 Call for Papers**

Nuts and Bolts

Timely tutorials on fundamental techniques for systems administrators.

74 Apache OpenMeetings
The free video conferencing platform has comprehensive collaboration tools and can be hosted locally, so sensitive corporate data is not exposed to cloud services.

78 microVMs
Get the strong isolation of virtual machines and lightweight behaviors of containers.

84 PowerShell Storage Automation
PowerShell succeeds when it comes to comprehensive automation where other tools that manage the hard drive inventory of Windows servers fall short.



90 QUIC
The Quick UDP Internet Connections protocol comes with mandatory TLS encryption and promises faster speeds.

94 Performance Tuning Dojo
Defining I/O baselines helps you determine the highest performance you can expect from your system when configured properly.



SystemRescue 9.04
64 bit

- **Live rescue disk**
- **Administer and repair Linux and Windows systems**
- **Linux kernel 5.15.58**
- **Expandable with system rescue modules**

See p 6 for details

SystemRescue 9.04 (Live)

On the DVD

SystemRescue [1] was formerly known as SystemRescueCd, the classic Linux system Live boot rescue CD-ROM. A full complement of disk management tools, network admin programs, and text editors let you administer and repair both Linux and Windows systems on desktops and servers. The Live boot lets you work on working, as well as disabled computers.

The rescue system [2] can be booted from a CD/DVD drive, a USB stick [3], or a network by PXE protocol [4]. The kernel has been updated to v5.15.58, a long-term support release. The project was rebased on Arch Linux in v6, and system rescue modules (SRMs) [5] were reimplemented in v7. These SquashFS filesystems allow you to add custom files to the Live system.



DEFECTIVE DVD?

Defective discs will be replaced, email: cs@admin-magazine.com

While this *ADMIN* magazine disc has been tested and is to the best of our knowledge free of malicious software and defects, *ADMIN* magazine cannot be held responsible and is not liable for any disruption, loss, or damage to data and computer systems related to the use of this disc.

Resources

- [1] SystemRescue: [\[https://sourceforge.net/projects/systemrescuecd/\]](https://sourceforge.net/projects/systemrescuecd/)
- [2] ChangeLog: [\[https://www.system-rescue.org/Changes-x86/\]](https://www.system-rescue.org/Changes-x86/)
- [3] USB install: [\[https://www.system-rescue.org/Installing-SystemRescue-on-a-USB-memory-stick/\]](https://www.system-rescue.org/Installing-SystemRescue-on-a-USB-memory-stick/)
- [4] Boot options: [\[https://www.system-rescue.org/manual/Booting_SystemRescue/\]](https://www.system-rescue.org/manual/Booting_SystemRescue/)
- [5] SystemRescue modules: [\[https://www.system-rescue.org/Modules/\]](https://www.system-rescue.org/Modules/)

Get started with



SysAdmin JOB HUB

Top jobs for IT professionals
who keep the world's
systems running

SysAdminJobHub.com

News for Admins

Tech News

CIQ Secures VC Funding, Forms New Leadership Team

CIQ, the company behind CentOS alternative Rocky Linux, focuses on enterprise tool suite.

CIQ, the founding support and services partner of Rocky Linux (<https://rockylinux.org/>), has formed a new leadership team with extensive experience building and running Linux-based infrastructure at scale.

This move follows recent venture capital funding (<https://www.prnewswire.com/news-releases/software-infrastructure-leader-ciq-closes-26-million-series-a-led-by-two-bear-capital-301544663.html>) secured in May “with the goal of building a suite of enterprise workflow orchestration and hybrid cloud solutions.” These tools, according to the announcement (<https://ciq.co/linux-and-open-source-veterans-sign-on-to-form-ciq-leadership-team/>), are designed to leverage the capabilities of Rocky Linux, an alternative to CentOS.

“The next generation of software infrastructure that we’re building at CIQ will help enterprises and organizations tackle data-intensive workflows—from big data analytics, to HPC for modeling and simulation, to training sophisticated machine learning models,” said Gregory Kurtzer, founder and CEO of CIQ.

Learn DevSecOps Basics Through Free Training Course

Introduction to DevSecOps for Managers (<https://www.edx.org/course/introduction-to-devsec-ops-for-managers>) is a free online training course from the Linux Foundation, which is now available on the edX platform.

According to the website, the course focuses on providing managers and leaders with an overview of the history, terminology, processes, and tools used to implement a DevSecOps approach within any organization.

The self-paced course “covers topics such as value stream management, platform as product, and engineering organization improvement, all driving towards defining Continuous Delivery and explaining why it is so foundational for any organization. The course also focuses on culture, metrics, cybersecurity, and agile contracting,” the website states.

GitLab Survey Reflects Shifting Roles in DevSecOps

DevOps roles continue to shift with devs taking on operations jobs, and operations professionals focusing more on cloud management, according to the recent GitLab 2022 Global DevSecOps Survey (<https://about.gitlab.com/developer-survey/#security>).



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

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

The survey asked Ops pros about their primary responsibilities, with the following results:

- Fifty-four percent manage hardware infrastructure “all or most of the time,” and 32 percent manage hardware infrastructure “sometimes.”
- Fifty-two percent manage cloud services “all or most of the time,” and 31 percent manage cloud services “sometimes.”

A majority of respondents also reported spending between one quarter to half of their time managing audit and compliance responsibilities; however, nearly 25 percent of Ops pros said they spend “between half and three-quarters of their time dealing with audit and compliance.” Other tasks include toolchain maintenance, automation, and DevOps coaching.

Other findings indicate that:

- Seventy percent of DevOps teams surveyed release code continuously, once a day, or every few days, which is up 11 percent from 2021.
- Sixty-eight percent of teams said software development lifecycle was either completely or mostly automated, up 13 percent from last year.
- Forty-seven percent of teams have full test automation, which is nearly double the number reported in 2021.

New Report Examines the Current State of ITOps and SecOps

According to the 2022 State of ITOps and SecOps survey (<https://www.informationweek.com/security-and-risk-strategy/the-state-of-itops-and-secops-an-inside-look>) from InformationWeek, the majority of IT and security teams have effectively handled the operational challenges of dealing with primarily remote workforces. However, the majority of respondents also said their teams are understaffed.

This year’s survey looked specifically at how IT and security teams adjusted to organizational changes resulting from the global pandemic, and 77 percent of respondents rated their response to working from home as effective or highly effective.

Other findings include:

- Eighty-four percent of survey respondents said that cybersecurity is important or absolutely critical to their organization and top management.
- Seventy-two percent of respondents said their organizations don’t have enough general IT staff members, and 64 percent said they don’t have enough cybersecurity staff.

Additionally, the survey found that general IT is assuming some responsibilities and functions that were previously handled by the security team. For example, the report says, “IT operations is more likely to be in charge of ensuring security controls for network, cloud, and application service providers than they were in the past.”

“These days security is the responsibility of everyone in IT, not just the cybersecurity team,” the report said.

Top Universities Lack Proper Email Security Measures

Recent research from Proofpoint (<https://www.proofpoint.com/us/newsroom/press-releases/proofpoint-97-top-universities-us-uk-and-australia-putting-students-staff>) found that top universities in the United States, United Kingdom, and Australia “are not taking appropriate measures to proactively block attackers from spoofing their email domains, increasing the risk of email fraud.”

Domain-based Message Authentication, Reporting and Conformance (DMARC) analysis (<https://dmarc.org/>) performed by Proofpoint showed that “none of the top U.S. and U.K. universities had a Reject policy in place, which actively blocks fraudulent emails from reaching their intended targets, meaning all are leaving students open to email fraud.”

According to the analysis, U.S. universities have the poorest levels of protection, followed by the United Kingdom, and then Australia.

“Email remains the most common vector for security compromises across all industries. In recent years, the frequency, sophistication, and cost of cyber attacks against universities has increased. It’s the combination of these factors that make it especially concerning that the premier universities in the U.S. are currently the most vulnerable to attack,” Proofpoint says.

A photograph of a dog, possibly a Border Collie, jumping through a series of vertical poles on a green grassy field. The dog is in mid-air, with its front legs tucked and back legs extended. The poles are orange and grey. The background is a bright green lawn.

Kubernetes containers, fleet management, and applications

Training Guide

Kubernetes is all the rage, but many admins find themselves struggling to get started. We present the basic architecture and the most important components and terms. By Martin Loschwitz

Kubernetes is one of the most dazzling attractions in the current circus of IT tools. Once developed by Google, it is now under the aegis of the Cloud Native Computing Foundation (CNCF) and has therefore become a genuine community project. However, Kubernetes (aka K8s, because of the eight letters between the K and the s) also seems to have driven a wedge into the normally closed ranks of administrators. Container advocates often can't imagine IT without containers and the right choice of orchestrator, whereas others have had little reason to consider Kubernetes until recently, which sometimes leads to a defiant reaction. Old habits can die hard. This behavior is understandable, because the Kubernetes universe is now dominated by a kind of tech-speak that outsiders need to learn before diving in. Because the solution is constantly evolving – with custom resource definitions, providers, and replica sets – even many die-hard admins of the early days of K8s have lost track of what's what, or what is used when and for what purpose. In this article, I rush to the aid of Kubernetes newcomers, providing a basic introduction to the topic that assumes no prior knowledge. In addition to the most important terms from

the K8s bubble, a brief architecture overview of Kubernetes is also on the agenda, as is an explanation of the solution's most important functions from the user's perspective. If you haven't yet dealt with K8s, you'll get the knowledge you need here.

A Word Up Front: Containers

Before I get to Kubernetes, a few things need to be clarified about containers. Many terms and, above all, acronyms are difficult to understand if you have no prior experience of the subject.

Already known by most is that Kubernetes is, at its core, a solution that orchestrates containers across physical system boundaries; that is, it rolls them out and operates them together such that a working symphony is created from individual containers. For Kubernetes to start and stop containers on the target systems, it needs the appropriate tools in place.

Kubernetes could, of course, implement the required infrastructure on the systems, but the tool sees fleet management as its core task. It looks to roll out containers locally on individual systems with the use of a local runtime environment. The problem is, several such environments now exist.

Most administrators should be familiar with the combination containerd and runc used in Docker; the docker command-line tool only acts as a front end for containerd, which calls runc in the background. Ultimately, runc creates containers and manages them. Consequently, runc is also the component that supports the Open Container Initiative (OCI) specifications ([Figure 1](#)).

Moreover, runc is not used just in the context of docker and containerd, but also in the context of Container Runtime Interface-O (CRI-O), which is a genuine Kubernetes-speak acronym. Until just a few years ago, Kubernetes used the containerd interface (i.e., the one from Docker), but at the behest of IBM and Red Hat, a separate solution emerged.

Kubernetes now follows a standard at the level of its own API that defines communication with external CRIs. The independent containerd and CRI-O projects compete, but both implement the CRI standard and can therefore be considered potential K8s back ends. Under the hood, both solutions use runc. The standard way to run Kubernetes today is based on CRI-O.

The competitor Podman, which replaces the docker and containerd tools, also requires some explanation.

Photo by Murilo Viviani on Unsplash

Podman calls runc directly in the background instead of dragging in another abstraction layer, as Docker does. This arrangement is especially interesting for admins who need to investigate why containers are not working on individual systems. At the end of the day, Podman is just another front end for runc.

Basic Technology

The K8s universe has a term you won't find in most Kubernetes architecture diagrams, although it describes Kubernetes's operating principles well: desired state management, or DSM, which describes the correct state of user resources. Kubernetes automatically takes care of achieving this state in the background. Therefore, a call to the Kubernetes cluster is not: Start container A on node 1, container B on node 2, and container C on node 3, and make sure the container data sets are replicas (i.e., they match). Instead, the statement would read: Create three replicas of the same container.

Kubernetes automatically assumes, according to its default configuration,

that the three instances of the containers will need to run on different hosts. I go into detail about the concept of a replica later.

The Right Format

If you read the Kubernetes documentation or tutorials, you regularly come across files in YAML or JSON format that need to be passed into the running cluster. Against the background of DSM, the purpose of these files becomes obvious: They allow you to tell a running Kubernetes instance what the desired state looks like and are therefore formats for templates in which you describe the desired setup as precisely as possible. Kubernetes then evaluates this information and creates the resources as specified.

If you're familiar with cloud principles, you'll instinctively think of services such as CloudFormation in AWS, and you're right: DSM is a core principle of classic orchestration and therefore also core to the functionality of CloudFormation or other solutions such as OpenStack Heat.

provides a source of information for running resources, as well.

Individual resources in the environment can also be controlled directly through the Kubernetes API. For example, if you execute a command to stop an application running in containers, `kubectl` translates the command on the fly into an updated resource definition with the desired status of "stopped" and passes this template to the Kubernetes API. This example shows that tools like `kubectl` are versatile in everyday use, because starting and stopping individual resources works very well in the way just described. However, if you want to describe an entire virtual environment with individual commands in this way, it would be very time consuming. In such scenarios, YAML templates come to the rescue.

High Availability Built In

One central aspect of modern IT architectures is built-in high availability (HA). Where workarounds like Pacemaker were used a few years ago, modern applications look to take care of their own availability. The control plane of a K8s cluster also needs to meet HA requirements. The failure of the central services in a K8s cluster would not lead to a loss of functionality of the rolled-out resources, but they would no longer be controllable. Consequently, all components of a K8s cluster are natively HA-enabled. Stored information about the states of resources rolled out in Kubernetes – basically, classic database content – plays a special role, and anyone who has ever dealt with high availability in a database context knows that the topic is anything but trivial to handle. Kubernetes itself does not attempt to keep its own persistent data highly available. Instead, it offloads the task to an external component, etcd, which is an implicitly highly available key-value store. Each Kubernetes controller cluster therefore also includes an etcd cluster that enables write and read access in multimaster mode (Figure 3). In this way, many small K8s services are quickly transformed

The Architecture

The question arises as to how Kubernetes manages to convert the desired state definition given by the user in the form of a template into running resources. The core issue is the Kubernetes architecture, in which a number of components contribute to achieving this state. The K8s services can be divided roughly into two categories: the control plane and the nodes that contain the containers (Figure 2).

The control plane, in turn, comprises several microservices. The API server acts as a hub, handling most of the communication for K8s with the outside world. When you use `kubectl` at the command line to deliver a resource definition to Kubernetes, the utility communicates with the Kubernetes's API in the background as a classic REST interface that accepts updates for existing resources or commands for creating new resources and

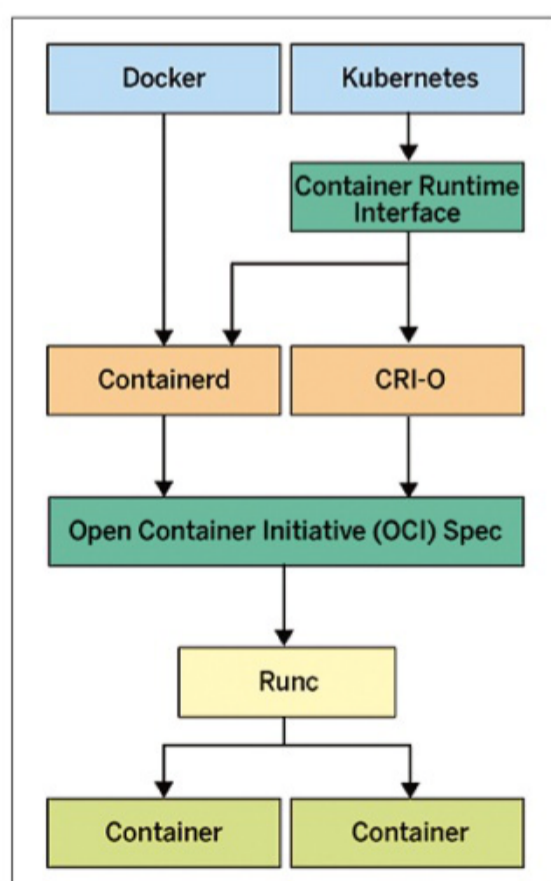


Figure 1: Kubernetes launches containers on the target systems with the Kubelet node agent and uses the container runtime interface (CRI). © Tom Donohue, www.tutorialworks.com

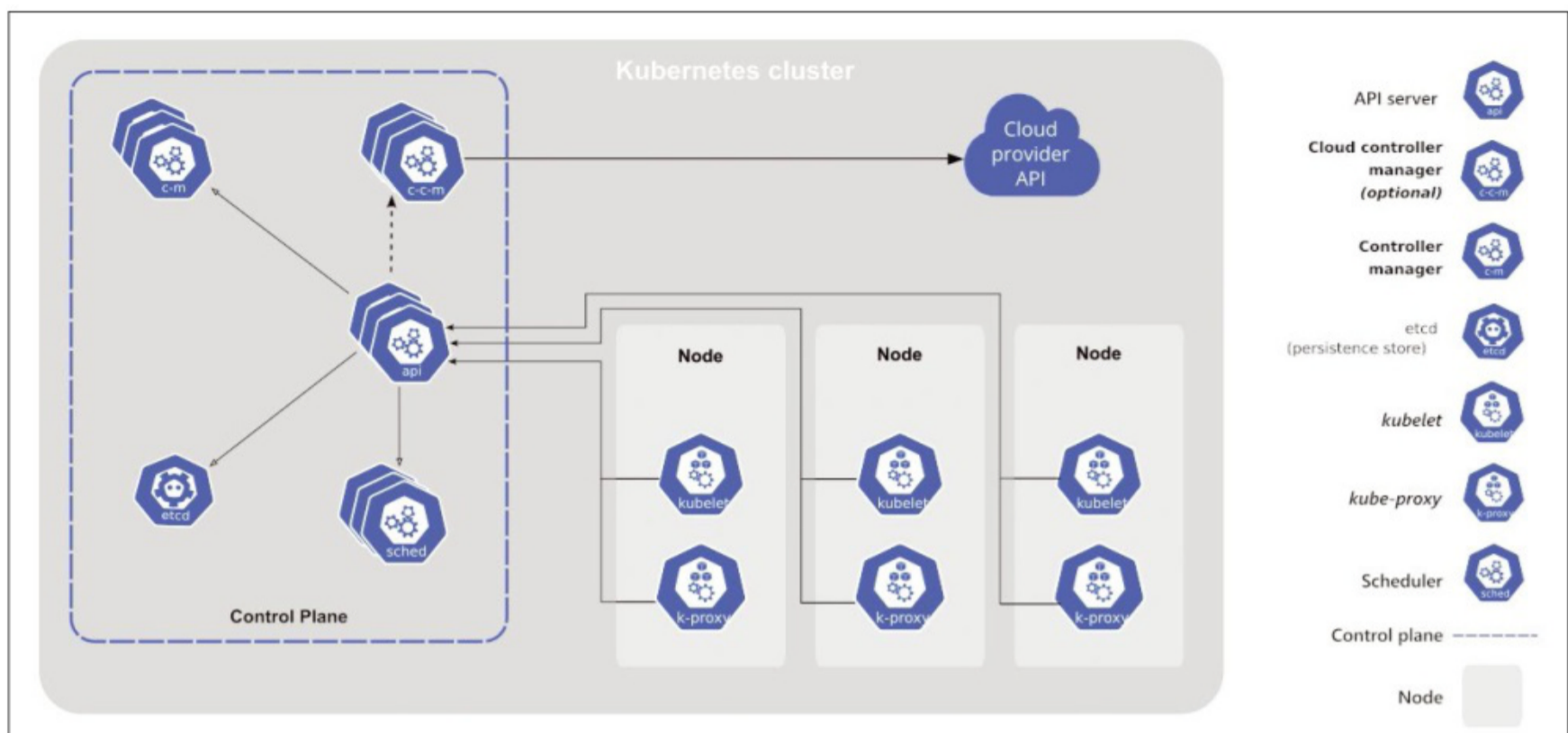


Figure 2: The Kubernetes architecture comprises the controller services in the controller cluster and Kubelet, which handles the configuration on the target systems. © Kubernetes

into a multidimensional mesh. An arbitrary number of Kubernetes API instances access an arbitrary number of etcd instances in the background. If one of the instances fails, the remaining ad hoc instances take over its tasks.

Creating Resources

So far, the Kubernetes story as this article tells it still has a gaping hole. Although receiving DSM definitions on one side and storing them persistently with HA on the other side is great, at the end of the day, you also need some kind of instance to ensure that the DSM specifications are turned into resources on the systems – ideally as running containers. Now the Kubernetes controller manager steps in: It includes several components, including the node controller, which keeps track of all available target systems that can run container workloads; the job controller, which is responsible for scheduling upcoming work for execution; and the endpoint, service account, and namespace controllers, which manage the internal details of Kubernetes (e.g., user management) and maintain a directory of running services in Kubernetes and their endpoints (i.e., the URLs used to reach the services).

Kubernetes uses the term “controller” not only in the context of its own infrastructure, but also for a specific type of resource. A controller at the Kubernetes resource level is basically a kind of programmed infinite loop that balances the actual and target states of the available resources. You need controllers to turn a DSM declaration into a running infrastructure by enforcing its creation with the use of other Kubernetes services.

Container Freight Handler

On the Kubernetes controller services side, only one service is missing to complete the set of mandatory basic tools: the scheduler. The

name says it all: The scheduler keeps track of all available nodes and their current workloads. When a user submits a DSM definition that requires the creation of new containers, the scheduler selects the appropriate

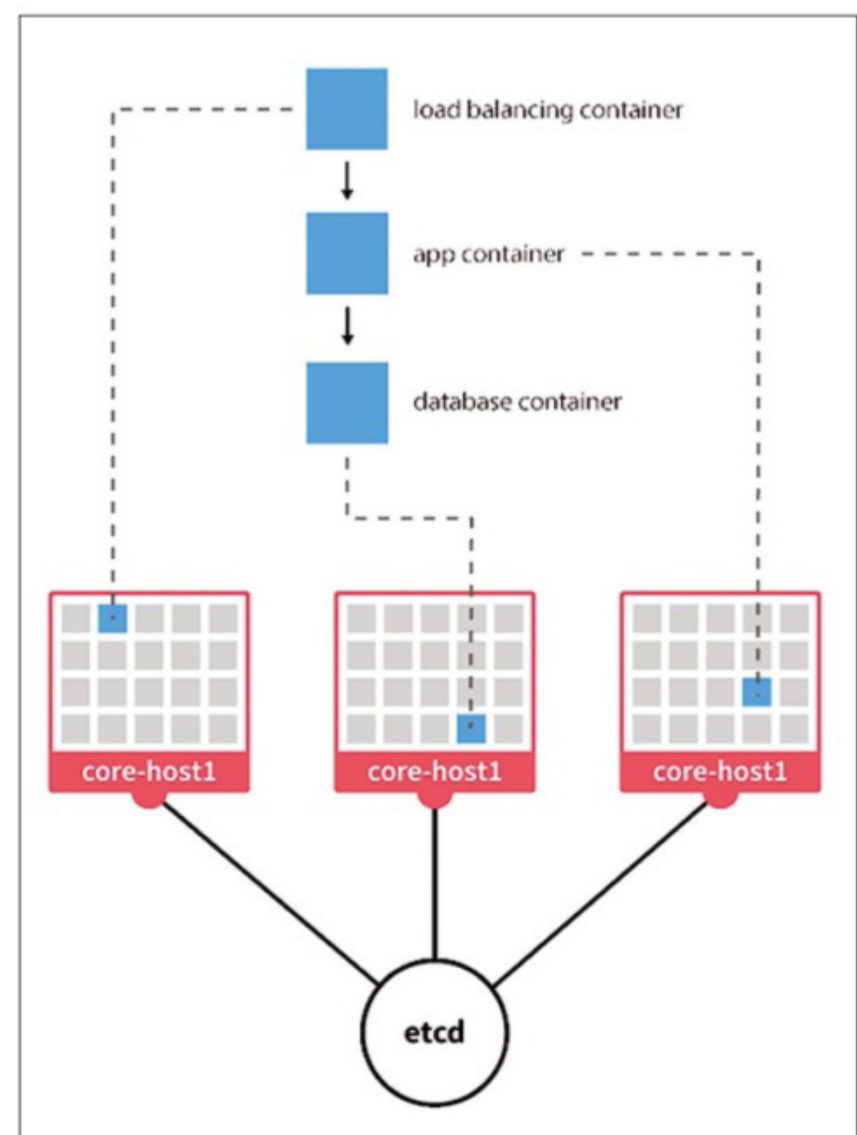


Figure 3: Etcd uses a consensus algorithm to take care of redundant storage of Kubernetes’s metadata. © etcd

targets when called by the controller manager. Controller decisions are not just directly related to containers; they can also include other virtual resources – but more on that later.

What Is Kubelet?

If you have not yet experienced Kubernetes hands-on, you might nevertheless be familiar with at least one term from the Kubernetes world: Kubelet. This Kubernetes component does not run on the Kubernetes controller cluster like all the others described thus far.

Instead, it is useful to think of Kubernetes's cluster-wide architecture as a server-agent architecture. When a user passes resource definitions to the cluster in a declarative language, Kubelet then converts it into a concrete infrastructure, such as containers, on the cluster's individual compute nodes.

As soon as the scheduler has selected the target system for resources, the Kubelet instance there runs and starts the desired resources. The service creates containers (e.g., with the CRI-O container interface mentioned earlier). By the way, the smallest resource unit that Kubelets handle is the pod. A pod groups an arbitrary number of related containers that always run on the same system.

Extending Kubernetes

In theory, all of the components that you need to convert your instructions into running Kubernetes instances have been described. Kubernetes's feature set is quite substantial – it has definitions for virtual networks, virtual storage volumes, and containers, and Kubernetes supports virtual load balancers, which are needed because Kubernetes normally creates containers with private IP addresses and then uses load balancers to set up port forwarding. However, a good amount of functionality can be implemented with plain vanilla Kubernetes, as a couple of examples clearly demonstrate.

In the Kubernetes API, the replica sets already mentioned act as a paraphrased definition for a resource that does not comprise just one pod, but several that need to be configured in a coordinated way. These resources can be databases (e.g., YugabyteDB [1]) that implement replication directly at the database level instead of offloading the task to the underlying storage. On the downside, the basic set of Kubernetes functions is by no means capable of creating every function you can think of. For example, anyone who has sniffed the air in the OpenStack universe knows that software-defined storage (SDS) and software-defined networking (SDN) play a very important role. If you virtualize the entire data center network and rely on components such as the Cisco application-centric infrastructure (ACI) to do so, you will also want your container network to be integrated into this system. If you use storage appliances or rely on a Ceph cluster, you will want to connect Kubernetes somehow so that persistent storage volumes can also be used in the cluster.

However, it would be completely unthinkable for the Kubernetes developers themselves to develop and maintain these interfaces to external services. They do not have the knowledge, nor would it make sense from their point of view to deal constantly with several side issues instead of focusing on the main application. One early Kubernetes design goal was therefore to provide external interfaces for extensions. In the Kubernetes context, several terms and acronyms again require explanation. One central concept is the plugin, which forms the de facto link between Kubernetes and external technologies, translating requirements from Kubernetes into machine code on the target systems. In Kubernetes, this concept is often further refined into various plugin groups. For example, K8s relies on plugins that follow the Container Network Interface (CNI) specification. The same applies to storage plugins.

Provisioners

Provisioners are the interfaces between plugins in Kubernetes. On the one hand, they dock with the Kubernetes API, and on the other hand, they know which plugins to call to create a defined configuration state. If you use storage as an example, classes in the Kubernetes API ensure the logical connection between resources of different types and a specific provisioner. In other words, a storage class in Kubernetes defines a specific storage type that Kubelet accesses through an appropriate plugin when it is prompted to create a resource of this kind. The terminology can differ in places for other external extensions, but on the whole, the functional principle remains the same.

Custom Resource Definitions

A custom resource (CR) and custom resource definition (CRD) are beasts you will encounter regularly. Although not primarily an external extension, the CRD API itself extends the Kubernetes API by creating a user's custom objects.

Out of the box, Kubernetes comes with a long list of resources that you access in your DSM manifests, but if you use additional external applications – or need to perform special tasks repeatedly – these standard definitions might not be sufficient, despite their large scope. Although the corresponding infrastructure would be available in Kubernetes, the resources would lie idle because the Kubernetes API would not call the resources. In these cases, CRDs enter the scene: They let you define arbitrary resource types in a Kubernetes cluster and associate them with the required functionality in the background.

CRDs also make it easier to create a kind of template for frequently used resource combinations. You could create, for example, a web server CRD, and then create ad hoc the resources needed for a web server (e.g., a virtual network). At the same time, you would select the correct image and

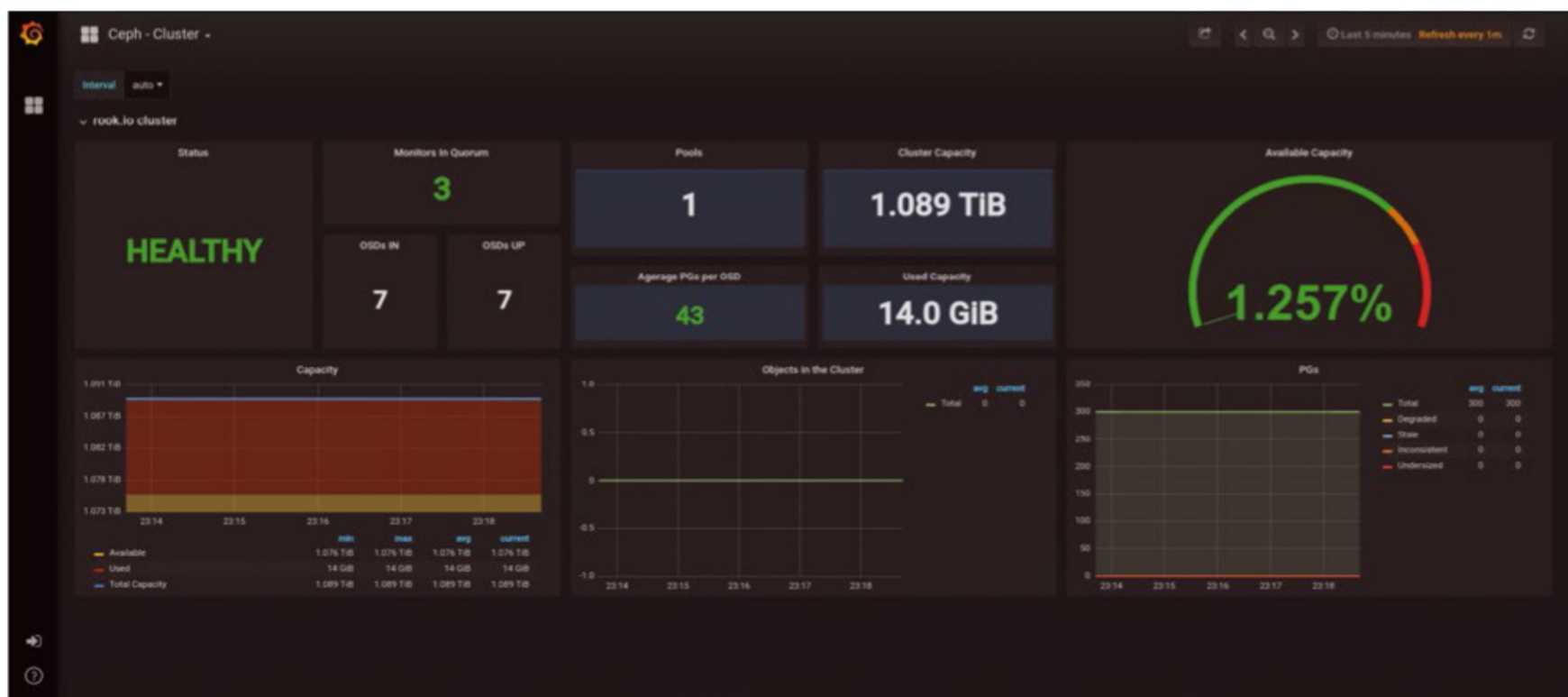


Figure 4: Prometheus comes as an operator for K8s and contains definitions for both custom resources and custom controllers, so you can set up a complete monitoring system for K8s in next to no time.

add provisions for dynamic firewalling. Later, you would only need to reference the web server resource in your manifest without specifying everything in DSM-speak each time. This process saves time and helps keep YAML templates short for use in K8s.

As the success of CRDs impressively proves, virtually every external solution for use in Kubernetes – whether it needs plugins and provisioners or

simply runs in Kubernetes – comes with its own CRD resources. Working with CRDs is business as usual for Kubernetes admins.

To match the custom resources, custom controllers work like the native Kubernetes controllers but let you perform arbitrary additional operations within the Kubernetes API. The combination of custom resources and custom controllers

is known as an “operator.” A Kubernetes operator provides most of the functionality you need to get specific software up and running quickly. One well-known example is the Prometheus time series database operator [2], which can be used to establish comprehensive monitoring, alerting, and trending (MAT) for an entire Kubernetes cluster (Figure 4).

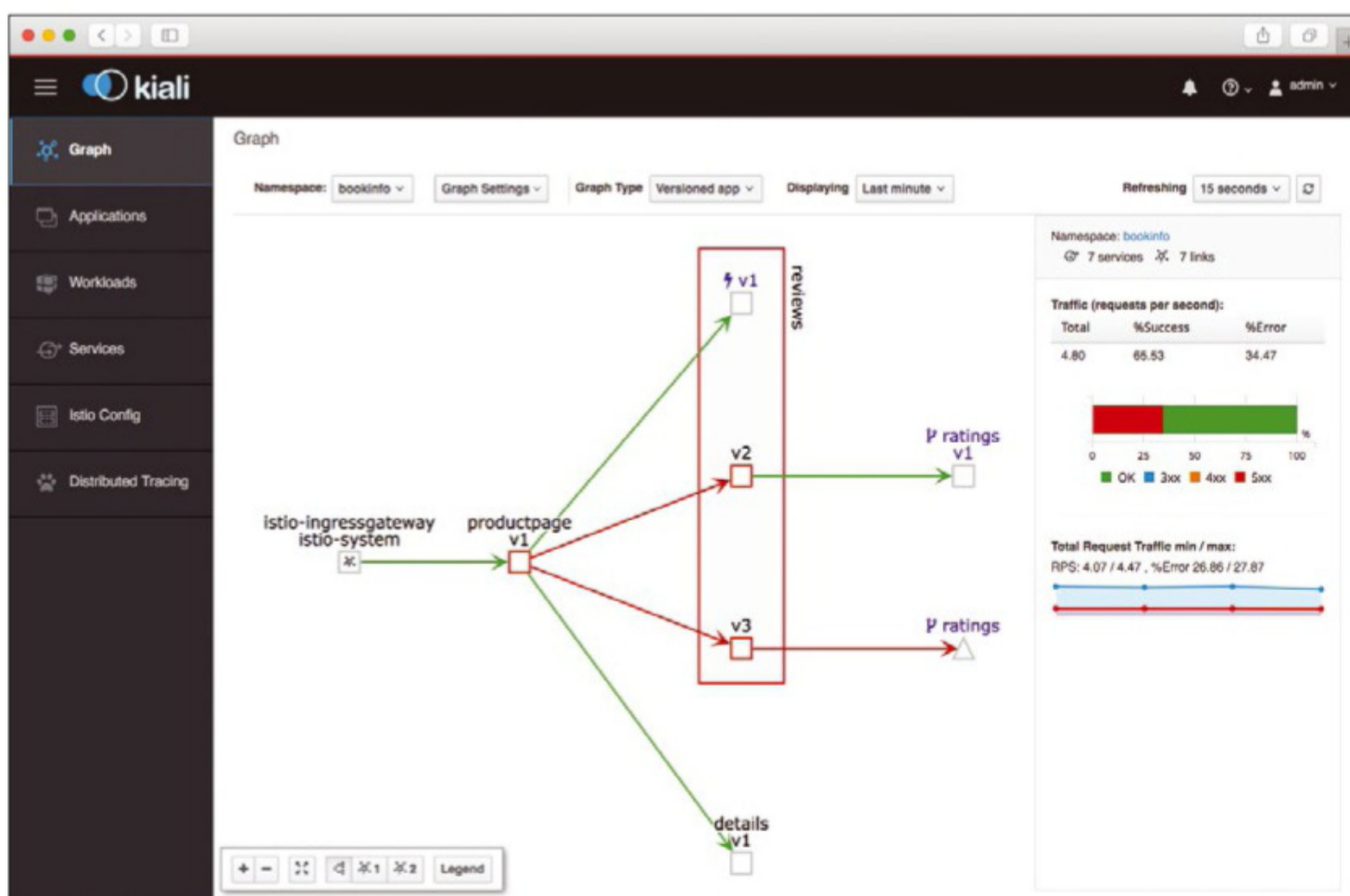


Figure 5: Istio (shown with the Kiali front end) creates dynamic network meshes in Kubernetes, making cloud-ready implementation far easier. © Kiali

Services

The market for software that adds features at the Kubernetes layer itself is at least as lively as the market for external solutions that connect to Kubernetes. These kinds of solutions have been covered in this magazine many times in the past. One prominent example is Rook [3], which implements a Ceph

cluster at the Kubernetes level. Rook creates volumes with custom CRDs that can then be attached directly to containers as native resources. Another very prominent solution is Istio [4], a service mesh that dynamically implements encryption and load balancing between each pod of an application in K8s (Figure 5). Solutions of this type integrate quite easily with Kubernetes, which is precisely why they are particularly popular on the market, and many companies are attempting to enter the world of K8s with such products, especially because they need have no dependence on an external entity. For example, a Ceph cluster built in Rook can be controlled exclusively by Kubernetes on-board tools, whereas attaching an external Ceph cluster to Kubernetes would require plugins, a storage class, and administrative access to multiple systems.

Helm

Finally, I'll look at a term from the Kubernetes world that regularly pops up in discussions: Helm [5], a package manager in the Kubernetes context.

If you use Rook or Istio, you implicitly roll out several containers that use different images. Moreover, you

need to store a set of CRDs in Kubernetes, and custom controllers might be required, as well. If you wanted to handle all of these tasks manually, you would have to work your way through a long to-do list. Instead, you could use a Helm chart to define all the necessary resources, properties, and details, which can be rolled out directly in Kubernetes with just a few commands. At the same time, Helm calls Kubernetes to create the resources belonging to the service. This principle is similar to the way classic package managers such as Rpm and Dpkg work, but it relies on container images and comes with a plethora of control information (Figure 6).

Conclusions

The most complicated part of getting to know Kubernetes is familiarizing yourself with its jargon, which is riddled with many special terms and acronyms. Even if you are not planning a Kubernetes setup in the near future, you will at least know what other people are talking about. Of course, a crash course like this article can only hope to cover the basic concepts. If you want to roll out Kubernetes yourself, you definitely need to plan for some preparation time to get up to speed. ■

Info

- [1] YugabyteDB:
[<https://www.yugabyte.com>]
- [2] "Monitoring container clusters with Prometheus" by Michael Kraus, *ADMIN*, issue 41, 2017, pg. 28,
[<https://www.admin-magazine.com/Archive/2017/41/Monitoring-container-clusters-with-Prometheus/>]
- [3] "Cloud-native storage for Kubernetes with Rook" by Martin Loschwitz, *ADMIN*, issue 49, 2019, pg. 47,
[<https://www.admin-magazine.com/Archive/2019/49/Cloud-native-storage-for-Kubernetes-with-Rook/>]
- [4] "A service mesh for microarchitecture components" by Martin Loschwitz, *ADMIN*, issue 54, 2019, pg. 28,
[<https://www.admin-magazine.com/Archive/2019/54/A-service-mesh-for-microarchitecture-components/>]
- [5] "Spotlight on the Kubernetes package manager, Helm" by Martin Loschwitz, *ADMIN*, issue 82, 2022, p. 62,
[<https://www.admin-magazine.com/Archive/2022/68/Spotlight-on-the-Kubernetes-package-manager-Helm/>]

The Author

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



```
apiVersion: v1
kind: Service
metadata:
  name: {{ include "common.names.fullname" . }}
  namespace: {{ .Release.Namespace }}
  labels: {{- include "common.labels.standard" . | nindent 4 }}
    {{- if .Values.commonLabels }}
    {{- include "common.tplvalues.render" ( dict "value" .Values.commonLabels "context" $ ) | nindent 4 }}
    {{- end }}
  {{- if .Values.commonAnnotations }}
  annotations: {{- include "common.tplvalues.render" ( dict "value" .Values.commonAnnotations "context" $ ) | nindent 4 }}
  {{- end }}
spec:
  type: {{ .Values.service.type }}
  sessionAffinity: {{ default "None" .Values.service.sessionAffinity }}
  {{- if (and .Values.service.clusterIP (eq .Values.service.type "ClusterIP")) }}
  clusterIP: {{ .Values.service.clusterIP }}
  {{- end }}
  {{- if (and .Values.service.loadBalancerIP (eq .Values.service.type "LoadBalancer")) }}
  loadBalancerIP: {{ .Values.service.loadBalancerIP }}
  {{- end }}
  {{- if (and (eq .Values.service.type "LoadBalancer") .Values.service.loadBalancerSourceRanges) }}
  loadBalancerSourceRanges: {{- toYaml .Values.service.loadBalancerSourceRanges | nindent 4 }}
  {{- end }}
  {{- if (or (eq .Values.service.type "LoadBalancer") (eq .Values.service.type "NodePort")) }}
  externalTrafficPolicy: {{ .Values.service.externalTrafficPolicy | quote }}
  {{- end }}
```

Figure 6: The Helm Kubernetes package manager stores ready-to-run definitions of custom resources and custom controllers in the fleet manager to facilitate their deployment.



Hone your skills with special editions!

Get to know Shell, LibreOffice, Linux, and more from our Special Edition 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

FREE DVD! **JOIN THE LINUX REVOLUTION!**
ALL THE SOFTWARE YOU NEED!

GETTING STARTED WITH LINUX

• MORE POWERFUL • MORE SECURE • MO

LEARN HOW TO SET UP A LINUX SYSTEM

- Listen to Music • Play Games • Process P
- Surf the Web • and Much More!



LINUX NEW MEDIA
WWW.LINUX-MAG

LINUX **301 BEST BASH COMMANDS**

LINUX SHELL

HANDBOOK 2022 Edition **LINUX Special**

SUPERCHARGE

YOUR LINUX SKILLS

Power at Your Fingertips

- Pipe and redirect output
- Monitor processes

BUILD A RASP PI RADIO!

MakerSpace #02

HANDS-ON PROJECTS FOR MAKERS

Cool Tricks!
Charge up with a saltwater battery

Painting with Light
Sure you need a programmable light stick

PiMiga 2.0
Get your game on with this Amiga emulator

MORE FUN FOR
FPGA
GEEKS!



LINUX NEW MEDIA

LINUX **TUNE YOUR LINUX SYSTEM**
2022 EDITION

Cool Linux Hacks

84 HACKS

Tricks and shortcuts for Linux geeks

- Speed up downloads with Xtreme Download Manager
- Search text, PDF, and Office files with one tool
- Run VMs on a Proxmox server without installing client software

RETRO FUN:
Play DOS Games with DOSBox
AUDIO EXPERTS:
Tools for DJs, Musicians, Composers

WWW.LINUX-MAGAZINE.COM



Discover the secrets of the experts

FREE DVD! **LibreOffice**
Full Version
Dive deep into the world's greatest free office suite
2022/23 Edition

LibreOffice Expert

Edit and Save MS Office Files

Write Your Own LO Macros

Save time and automate common tasks



LINUX NEW MEDIA USA

ARCHIVE DVD **RASPBERRY PI GEEK**
THE COMPLETE ARCHIVE
2,000 pages of maker projects and more!

FREE DVD
\$39.90
VALUE!

MakerSpace

HANDS-ON PROJECTS FOR MAKERS





Users speak on Kubernetes in business practice

Quick Delivery

Users in corporate and government agencies that have successfully switched to Kubernetes share their positive experiences and the stumbling blocks to be avoided. By Jens-Christoph Brendel

Some questions about Kubernetes can only be answered by looking at the experiences of those who have successfully walked the path to the container-based architecture, including: What can be achieved with Kubernetes and what might be illusion? What mistakes can be avoided when implementing the technology? How much time must be scheduled and what resources need to be considered? What changes besides the technical upheaval need to be addressed?

Agile Car Maker

The faster new ideas translate into tangible products, the better the prospect of business success. This concept is referred to as “time to market” – the shorter, the better. In the automotive industry, rapid application development and deployment guarantees a competitive edge. One technical prerequisite is established by containerization and cloud computing, which are asserting themselves in many places and are generally seen as drivers of IT innovation. Porsche Informatik [1] has taken this path. The company provides IT services for Porsche Holding Salzburg and the Volkswagen Group. Millions of people in car dealerships, workshops, import operations, logistics

companies, and the financial services industry, as well as Internet end users, use systems by Porsche Informatik. The organization delivers and manages 180 solutions in 32 countries across four continents. To this end, Porsche Informatik employs more than 800 specialists who develop and provide solutions for the ongoing digital transformation of the automotive industry.

These solutions involve business software for authorized dealers, after sales service, spare parts sales, and financial services. One concrete example is the Car Configurator, which lets prospective new car buyers put together their dream vehicle on the Internet in just a few steps. Both the images of the vehicle and the price computations are instantly updated with any selected details: paint, interior trim, wheels, and so on. Once the desired car has been configured, it can be supplemented with loan, leasing, and insurance offers. The Car Configurator is integrated into the websites of the Volkswagen Group brands and car dealerships. Another example of the solutions for which Porsche Informatik is responsible is the “Das WeltAuto” [2] used car portal, available in many countries, which provides a wide range of inspected, serviced, and repaired used

cars over the Internet. Participating dealers, as well as private sellers, can advertise their used vehicles there. In the US, this service is available through dealers and is called Certified Pre-Owned [3], and in the UK, Approved Used [4].

The basis of the application infrastructure for Porsche Informatik’s solutions and services is the Red Hat OpenShift [5] enterprise Kubernetes platform, which the company operates in a private cloud environment. By migrating the previous legacy infrastructure to the container-based, cloud-native platform, Porsche Informatik was able to reduce development times from weeks to hours. On average, applications and services can be built, tested, and deployed in about 90 percent less time than in the past. An initial prototype is available within hours with Red Hat OpenShift. Michael Karnutsch, Infrastructure Architect at Porsche Informatik, explained that: “Kubernetes is clearly the de facto standard for Linux container development. However, in our opinion, building a Kubernetes infrastructure yourself is not a sensible path to take when commercial products and standard solutions such as Red Hat OpenShift are also available, because they already include many important performance features, such

as authorization, authentication, logging, and metrics. However, you also need to be clear about one thing: If you don't have any Kubernetes know-how on the infrastructure and development side, it won't work either."

Red Hat OpenShift, the standard container orchestration product, is based on Kubernetes, provides a stable container platform environment for applications, and helps development teams run their continuous integration/continuous delivery (CI/CD) pipelines. More specifically, Red Hat OpenShift lets developers design, automate, scale, and manage container-based applications. The Enterprise Kubernetes platform includes all the features and services needed to run a container management solution for mission-critical applications on different infrastructures in a certified way, including aspects such as service-level agreements (SLAs), multiple layers of security, automation, and cluster management.

Red Hat OpenShift provides the runtime environment at Porsche Informatik. The platform currently hosts development processes in 11 clusters, with thousands of containers on well over 100 nodes, and is regularly used by around 500 developers. The clusters are broken down into testing and production environments. The infrastructure allows the deployment of a new cluster within a few hours.

Administrators can easily implement and enforce security and other policies across teams and clusters from Red Hat OpenShift's unified management console. Porsche Informatik also uses Red Hat Advanced Cluster Management for Kubernetes and Argo CD for infrastructure management that ensures Red Hat solutions remain up to date, are protected against vulnerabilities, and comply with various standards.

One of the key benefits of the new container-based, cloud-native environment is the self-service capabilities for Porsche Informatik's development teams. They can now provision services and infrastructure independent of the enterprise infrastructure team. The request process for resources is very simple and, above all, fast.

In the past, deploying, for example, Apache Tomcat was very time-consuming, from requesting the web server to rolling out a virtual machine and integrating it into the network infrastructure, to setting up the certificates.

From a technical point of view, the benefits of the new infrastructure are also reflected in the greater freedom for development teams. In principle, they can introduce and operate new technologies more easily or set up applications completely independently without having to worry about the underlying infrastructure. Examples include Node.js or a database management system such as MongoDB to cover new use cases.

In terms of security, in particular, Porsche Informatik uses solutions that complement the runtime environment for the logon procedures, container image scanning, and CI/CD, among other uses. In terms of build and deployment environments, Jenkins is used in development with a GitLab environment and Argo CD in the infrastructure area.

Thanks to the introduction of an agile, collaborative DevOps approach supported by Red Hat OpenShift, developers, architects, infrastructure experts, and platform teams at Porsche Informatik can collaborate far more effectively to design and update innovative applications and services, avoiding redundant work in the process.

However, introducing an enterprise Kubernetes platform and adopting agile, iterative processes also require a cultural change within the company. This aspect mainly affects the operations division, rather than the development division, which is already familiar with DevOps processes. Where operations also follow an as-code approach, a new mindset is needed that eliminates the classic separation of such groups as the Linux, network, or storage teams.

Although getting started in the Kubernetes world isn't rocket science, companies need to keep a few things in mind. From the perspective and experience of Porsche Informatik, the following best practices appear to be helpful:

- Trials and intensive testing of a Kubernetes environment should come first.
- The use of a standard platform is always recommended; it takes a lot of work off the company's shoulders.
- Users should always stick to the Kubernetes standard.
- A registry and base image structure must be created and made available to developers.
- New developers need help to get started, more specifically in the form of documentation with a detailed process description for various actions, such as deploying applications.

The result of such a process is an infrastructure that scales, maintains, and patches well and that future-proofs the enterprise for challenges that lie ahead.

Of course, development at Porsche Informatik is an ongoing process. In the medium term, the aim is to deploy Red Hat OpenShift in a public cloud. Porsche Informatik's goal is to deliver all applications to employees and end users in all countries with Microsoft Azure Red Hat OpenShift. Managed services from Microsoft will also be accessible to developers in the future "as code" from a pipeline.

Agile Authority

More agility was also the issue in a large German federal authority. Although agile processes had already been in use for about three years in the development of new software, the waterfall model was still used for older software in the core inventory (Java 2 Platform, Enterprise Edition (J2EE) applications, service-oriented architecture (SOA) services, Oracle Fusion Middleware): A development department updated programs about three times a year, and other business units operated them. In contrast, new software that is created in line with the DevSecOps approach and is also operated by the same team can already be up and running with a 14-day cycle. Even before Kubernetes, people were dealing with virtualization and

containers with Apache Mesos, which manages all the resources of a data center or cloud in a centralized, platform-independent way and also natively supports Docker containers. However, it had become increasingly difficult to find employees with Mesos expertise. When the previously used container orchestrator DC/OS was discontinued, a decision was taken to switch to the market leader, Kubernetes.

In the course of doing so, a number of difficulties had to be overcome, such as rapid migration of all legacy DC/OS processes along with their data into the Kubernetes world or the introduction of a service mesh (Istio [6]) that did not exist before. Troubleshooting across multiple virtualization layers also proved to be very complex. On top of that, the strict compliance requirements of the public sector had to be met. Differences in the willingness to adapt to new things and to leave behind traditional ways were present across the various departments.

In the end, however, some achievements on the positive side were noteworthy. A cutting-edge and innovative technology platform has been established for the next few years and is constantly evolving. It forms the basis for the development of agile software that can also be ported easily to hybrid clouds in container format. Classic operations also benefit from the very high level of automation, which makes many manual work steps superfluous. The recruitment of new employees is also easier now. What proved to be particularly important en route to Kubernetes was having well-coordinated teams that regularly exchanged ideas in a spirit of trust and that cultivated shared values, such as an open error culture. From a technical point of view, the experience with several smaller clusters instead of a single very large cluster was positive (and involved the use of a specially tailored Kubernetes distribution – from SUSE in this case), as was a consistent alignment with the mission statement of infrastructure as code, to ensure traceability and repeatability.

Agile Media

Kubernetes is not only providing food for thought in industry and government, but also in the media – for example, at the leading Spanish media company Atresmedia [7], which houses radio, television, and streaming services under one roof. The people in charge recognized that the expectations of today’s media consumers cannot be served well by legacy, monolithic applications. Moreover, the cost of software maintenance (e.g., for the content management system) had risen to dizzying heights. At the same time, the environment was slow and had limited scalability, which is a very important criterion when huge volumes of data are generated in a short period of time, such as in the course of election coverage.

Therefore, the company decided to break with the old architecture and base a new approach on microservices. The underpinnings were to be Canonical’s Charmed Kubernetes [8] product, which comes with a high level of automation and scalability. The vendor was not an unknown, and the company had relied on Ubuntu for encoding and transcoding videos for many years. With its containerization approach, microservices promised the flexibility, resilience, and ease of management Atresmedia was looking for. Of all the microservices-oriented projects the company tested, Kubernetes performed best and offered the added benefit of having a large community. Canonical helped Atresmedia automate the deployment, operation, and scaling of the new Kubernetes cluster with the help of Juju, the operational lifecycle management tool for Charmed, resulting in a seamless implementation and massive simplifications of on-going application management. Additionally, Juju facilitated Atresmedia’s deployment of cloud-native applications, both on-premises and in the cloud, which is an important requirement given the company’s hybrid and multicloud strategy. Initially, Atresmedia relied on

Canonical support for day-to-day operations. However, since the roll-out, the in-house IT team has built up expertise and now maintains the cluster independently. Canonical remains available to resolve any major issues, but in four years they have not had a single significant incident. Óscar Martínez, Head of Architecture and Backend Development at Atresmedia, comments: “We always strive for autonomy. So it was good to know that we were not tied to a consulting service. And now that we have in-house skills, Juju is invaluable. It has helped us become totally flexible when we need to develop or modify the cluster.”

The new Kubernetes-based solution makes scaling issues a thing of the past. As soon as the ATRESplayer streaming service is faced with increased traffic, the system automatically scales up its resources to cope with the traffic. It passed its baptism of fire during the Spanish general election in 2019, where it presented the latest election results in real time to hundreds of thousands of Spanish citizens. “If we had not been able to handle the traffic and provide up-to-date data, it would have been a serious blow to our competitiveness,” said Miguel Rodríguez, “but Charmed Kubernetes met our expectations. Scaling worked without any problems.” ■

Info

- [1] Porsche Informatik: [\[https://www.porscheinformatik.com/en/\]](https://www.porscheinformatik.com/en/)
- [2] Das WeltAuto (VW): [\[https://www.volkswagen.de/de/angebote-und-produkte/angebote-fahrzeugkauf/weltauto.html\]](https://www.volkswagen.de/de/angebote-und-produkte/angebote-fahrzeugkauf/weltauto.html) (in German)
- [3] Certified Pre-Owned: [\[https://www.vwcpo.com/cpo/vw-program/\]](https://www.vwcpo.com/cpo/vw-program/)
- [4] Approved Used: [\[https://www.volkswagen.co.uk/en/used-cars/approved-used-benefits.html\]](https://www.volkswagen.co.uk/en/used-cars/approved-used-benefits.html)
- [5] Red Hat OpenShift: [\[https://www.redhat.com/technologies/cloud-computing/openshift\]](https://www.redhat.com/technologies/cloud-computing/openshift)
- [6] Istio: [\[https://istio.io\]](https://istio.io)
- [7] Atresmedia: [\[https://www.atresmedia.com\]](https://www.atresmedia.com) (in Spanish)
- [8] Charmed Kubernetes: [\[https://ubuntu.com/kubernetes/charmed-k8s\]](https://ubuntu.com/kubernetes/charmed-k8s)

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. Subscribe today for our excellent newsletters:

ADMIN HPC • ADMIN Update • Linux Update
and keep your finger on the pulse of the IT industry.

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

Management improvements, memory scaling, and EOL for FileStore

Refreshed

Ceph developers focus on getting rid of historic clutter and adding new features for improved performance and built-in automation. By Martin Loschwitz

In the early days, Ceph was considered new, hip, and innovative because it promised scalable storage without ties to established storage solutions. If you were fed up with SANs, Fibre Channel, and complicated management tools, Ceph offered a solution that was refreshingly different, without a need for special and expensive hardware. In the meantime, Ceph has become a commodity – that is, an established solution for specific

areas of application – with a market for training as well as for specialists with Ceph knowledge. Therefore, it's hardly surprising that new Ceph releases no longer come with masses of new features, unlike the past, but instead are taking things a little easier. The newly released Ceph version 17.2 [1] bears witness to that trend: Instead of turning big wheels, the developers have focused on tweaks, with not too many changes for existing

clusters. Only if you operate a very old Ceph cluster and still rely on the old on-disk FileStore format (Figure 1) will you be prompted to take immediate action. In this article, I explain why this is the case and what else you can and should expect from Ceph 17.2.

Goodbye FileStore

One of the most important innovations in Ceph 17.2 is undoubtedly that the old on-disk format FileStore from the early Ceph days is now marked as deprecated, although this does not yet have any practical

```

root@ceph-01:/var/lib/ceph/osd/ceph-4# ls -la
total 80
drwxr-xr-x  3 root root  181 Dec 30 13:04 .
drwxr-xr-x 18 root root 4096 Nov 13 21:06 ..
-rw-r--r--  1 root root  503 Nov 13 20:19 activate.monmap
-rw-r--r--  1 root root    3 Nov 13 20:19 active
-rw-r--r--  1 root root   37 Nov 13 20:19 ceph_fsid
drwxr-xr-x 932 root root 24576 Dec 28 23:22 current
-rw-r--r--  1 root root   37 Nov 13 20:19 fsid
lrwxrwxrwx  1 root root   28 Dec 28 20:39 journal -> /mnt/                /ceph-4
-rw-----  1 root root   56 Nov 13 20:19 keyring
-rw-r--r--  1 root root   21 Nov 13 20:19 magic
-rw-r--r--  1 root root    6 Nov 13 20:19 ready
-rw-r--r--  1 root root    4 Nov 13 20:19 store_version
-rw-r--r--  1 root root    0 Dec 31 19:12 upstart
-rw-r--r--  1 root root    2 Nov 13 20:19 whoami
root@ceph-01:/var/lib/ceph/osd/ceph-4#

```

Figure 1: FileStore is the name of the old Ceph on-disk format, which uses a POSIX-compatible filesystem on the object storage devices.

consequences in everyday life. Owner Red Hat leaves no doubt that FileStore will be removed from Ceph sooner or later and urges admins of existing systems to upgrade their Ceph disks to the latest technology.

Like any storage solution, Ceph relies on block storage in the background to store its data. As an object store, however, the central function of the solution is to paint an abstraction layer between the physical stores on the one hand and the clients accessing them on the other. The clients do not need to worry about the physical architecture of the cluster, and you can use almost any number of physical devices as storage devices in the background. The vast majority of Ceph clusters today continue to rely on slow hard drives, at least for mass storage, because the break-even point has not yet been reached in terms of the cost per gigabyte for fast flash-based storage. For the Ceph object store RADOS to store its data on block devices, it needs a structure. Earlier Ceph versions relied on a classic filesystem following the POSIX standard, which had to be created on the respective data carrier first to let the RADOS

object memory-store binary objects.

For many years, developers recommended XFS ([Figure 2](#)). In the background, however, people were eyeing up Btrfs, which promised considerable speed advantages in combination with RADOS. The construct of a POSIX filesystem and the metadata belonging to Ceph on object storage devices (OSDs) was subsequently named FileStore.

How Red Hat's Btrfs adventure ended is well known: In Red Hat Enterprise Linux (RHEL) 8, the filesystem was dropped from the distribution. Even in RHEL 7, Btrfs had only made it to Technical Preview status, so from Red Hat's point of view, the filesystem was never fit for production. This situation quickly became a problem for the Ceph developers because the stopgap, XFS, was increasingly proving to be a performance inhibitor.

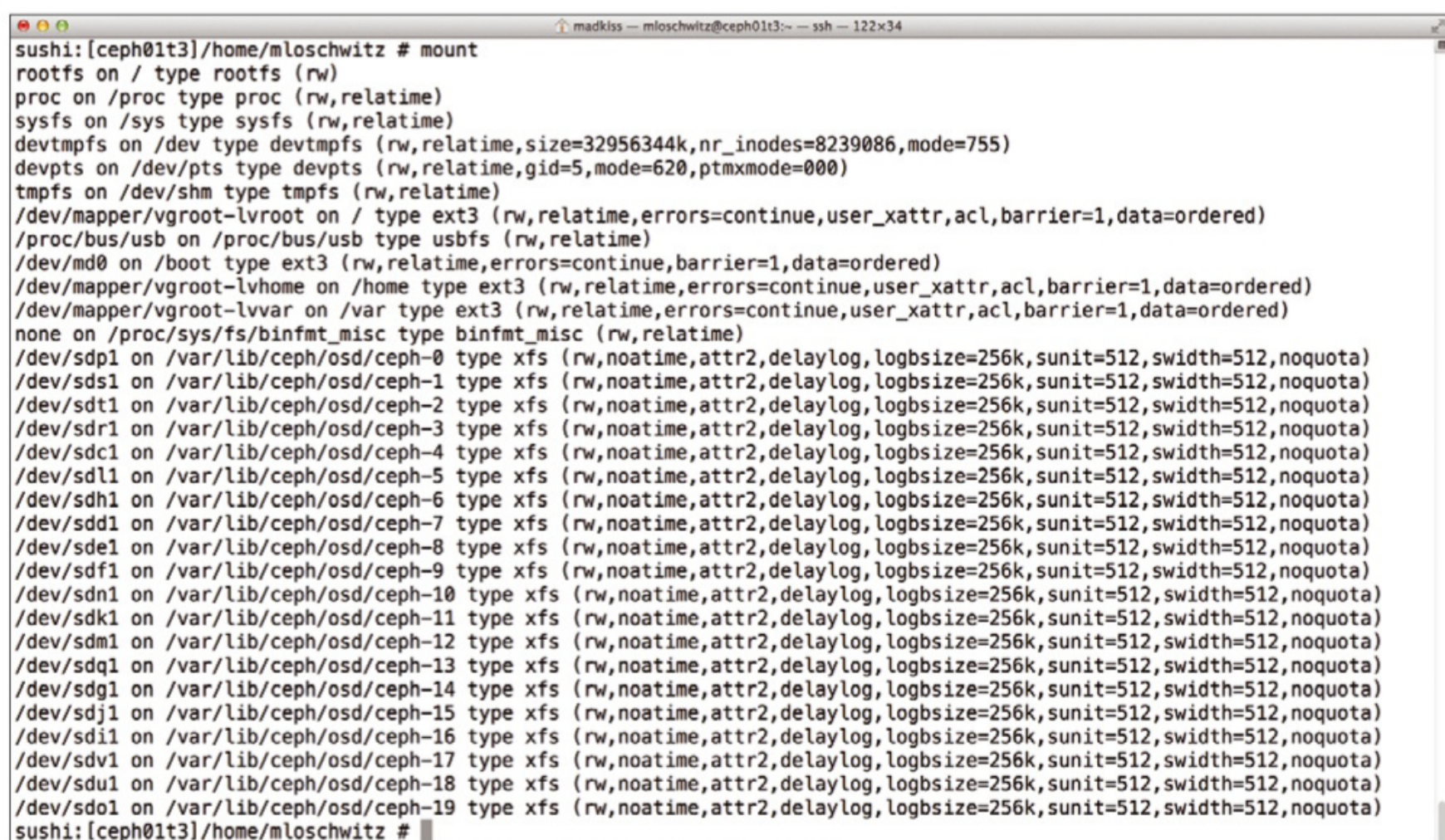
On closer inspection, RADOS is designed to create new directories for new data on its OSDs and assign new object IDs instead of recycling old ones. Even if a user overwrites an existing file with one of the three standard interfaces in RADOS – CephFS, Ceph Block Device (aka RADOS block device, RBD), Ceph Object Gateway – RADOS does not

replace existing objects in the background. Instead, it creates the new data as an equally new object and marks the old data as obsolete so that the data is overwritten in the short term.

The Problem with POSIX

The crux of the matter is that POSIX-compatible filesystems offer many integrity checks and consistency guarantees for cases such as overwriting existing data, the technical implementation of which nibbles away at performance. Ceph and its legacy format FileStore were in a lose-lose situation. Because effectively only XFS was available as an on-disk filesystem for OSDs, it was necessary to rely on its guarantees in terms of consistency and integrity, even though it made no sense at all in the Ceph context.

A few years ago, the Ceph developers finally ran out of patience and started working on a FileStore successor. Instead of a bloated POSIX filesystem, they determined it would be fine just to maintain some sort of database in an OSD's metadata that contains the physical memory address of an object on the disk. A key-value store would



```
sushi:[ceph01t3]/home/mloschwitz # mount
rootfs on / type rootfs (rw)
proc on /proc type proc (rw,relatime)
sysfs on /sys type sysfs (rw,relatime)
devtmpfs on /dev type devtmpfs (rw,relatime,size=32956344k,nr_inodes=8239086,mode=755)
devpts on /dev/pts type devpts (rw,relatime,gid=5,mode=620,ptmxmode=000)
tmpfs on /dev/shm type tmpfs (rw,relatime)
/dev/mapper/vgroot-lvroot on / type ext3 (rw,relatime,errors=continue,user_xattr,acl,barrier=1,data=ordered)
/proc/bus/usb on /proc/bus/usb type usbfs (rw,relatime)
/dev/md0 on /boot type ext3 (rw,relatime,errors=continue,barrier=1,data=ordered)
/dev/mapper/vgroot-lvhome on /home type ext3 (rw,relatime,errors=continue,user_xattr,acl,barrier=1,data=ordered)
/dev/mapper/vgroot-lvvar on /var type ext3 (rw,relatime,errors=continue,user_xattr,acl,barrier=1,data=ordered)
none on /proc/sys/fs/binfmt_misc type binfmt_misc (rw,relatime)
/dev/sdp1 on /var/lib/ceph/osd/ceph-0 type xfs (rw,noatime,attr2,delaylog,logbsize=256k,sunit=512,swidth=512,noquota)
/dev/sds1 on /var/lib/ceph/osd/ceph-1 type xfs (rw,noatime,attr2,delaylog,logbsize=256k,sunit=512,swidth=512,noquota)
/dev/sdt1 on /var/lib/ceph/osd/ceph-2 type xfs (rw,noatime,attr2,delaylog,logbsize=256k,sunit=512,swidth=512,noquota)
/dev/sdr1 on /var/lib/ceph/osd/ceph-3 type xfs (rw,noatime,attr2,delaylog,logbsize=256k,sunit=512,swidth=512,noquota)
/dev/sdc1 on /var/lib/ceph/osd/ceph-4 type xfs (rw,noatime,attr2,delaylog,logbsize=256k,sunit=512,swidth=512,noquota)
/dev/sdl1 on /var/lib/ceph/osd/ceph-5 type xfs (rw,noatime,attr2,delaylog,logbsize=256k,sunit=512,swidth=512,noquota)
/dev/sdh1 on /var/lib/ceph/osd/ceph-6 type xfs (rw,noatime,attr2,delaylog,logbsize=256k,sunit=512,swidth=512,noquota)
/dev/sdd1 on /var/lib/ceph/osd/ceph-7 type xfs (rw,noatime,attr2,delaylog,logbsize=256k,sunit=512,swidth=512,noquota)
/dev/sde1 on /var/lib/ceph/osd/ceph-8 type xfs (rw,noatime,attr2,delaylog,logbsize=256k,sunit=512,swidth=512,noquota)
/dev/sdf1 on /var/lib/ceph/osd/ceph-9 type xfs (rw,noatime,attr2,delaylog,logbsize=256k,sunit=512,swidth=512,noquota)
/dev/sdn1 on /var/lib/ceph/osd/ceph-10 type xfs (rw,noatime,attr2,delaylog,logbsize=256k,sunit=512,swidth=512,noquota)
/dev/sdk1 on /var/lib/ceph/osd/ceph-11 type xfs (rw,noatime,attr2,delaylog,logbsize=256k,sunit=512,swidth=512,noquota)
/dev/sdm1 on /var/lib/ceph/osd/ceph-12 type xfs (rw,noatime,attr2,delaylog,logbsize=256k,sunit=512,swidth=512,noquota)
/dev/sdq1 on /var/lib/ceph/osd/ceph-13 type xfs (rw,noatime,attr2,delaylog,logbsize=256k,sunit=512,swidth=512,noquota)
/dev/sdg1 on /var/lib/ceph/osd/ceph-14 type xfs (rw,noatime,attr2,delaylog,logbsize=256k,sunit=512,swidth=512,noquota)
/dev/sdj1 on /var/lib/ceph/osd/ceph-15 type xfs (rw,noatime,attr2,delaylog,logbsize=256k,sunit=512,swidth=512,noquota)
/dev/sdi1 on /var/lib/ceph/osd/ceph-16 type xfs (rw,noatime,attr2,delaylog,logbsize=256k,sunit=512,swidth=512,noquota)
/dev/sdv1 on /var/lib/ceph/osd/ceph-17 type xfs (rw,noatime,attr2,delaylog,logbsize=256k,sunit=512,swidth=512,noquota)
/dev/sdu1 on /var/lib/ceph/osd/ceph-18 type xfs (rw,noatime,attr2,delaylog,logbsize=256k,sunit=512,swidth=512,noquota)
/dev/sdo1 on /var/lib/ceph/osd/ceph-19 type xfs (rw,noatime,attr2,delaylog,logbsize=256k,sunit=512,swidth=512,noquota)
sushi:[ceph01t3]/home/mloschwitz #
```

Figure 2: FileStore was intended to deliver its full punch in combination with Btrfs, but then Btrfs was dropped from RHEL, so Ceph usually teamed up with the far-from-perfect XFS.

be good enough, flanked by a trunk filesystem that allows data to be stored on the physical device. It took a while to settle on the right combination of tools. LevelDB has seen some use as a database, but it was not fast enough for the developers and soon fell out of favor. In the end, the race was won by RocksDB, an extremely fast key-value store by Facebook that became the core of the new on-disk format. To clearly distinguish the new format from the old, it was named BlueStore.

BlueStore soon became popular with Ceph users. Hacks like outsourcing the FileStore format journals to fast SSDs were no longer necessary thanks to the new solution. For classic workloads, speed increases of 50 percent and more could be achieved with BlueStore in individual cases, even without additional hardware [2]. The developers virtually stopped developing FileStore because they considered it a dead end; therefore, little more than the bare effort has taken place in Ceph for years to keep FileStore functional.

More Efficiency

BlueStore has been the default in Ceph for several releases, and the Ceph developers have publicly discussed giving FileStore the *coup de grâce* on the project's mailing list. What makes sense from the user's point of view is only logical from the developer's point of view, too. Only old setups that have not been migrated for years are likely to still use FileStore, and it is hardly worthwhile to continue actively maintaining large amounts of code in Ceph, which is all the more true because BlueStore now lacks virtually any features that were once available in FileStore. If admins are still using FileStore today, it is probably because they have not taken the time to update.

Ceph developers are now homing in on precisely these admins with Ceph 17.2. The decision to flag FileStore as deprecated officially is a clear warning signal and the unequivocal indication that active Ceph clusters with FileStore will be excluded from

future updates. If you are still running a FileStore-based cluster, ideally you will want to start converting your OSDs to BlueStore very soon.

How It Works

Among the several options, if you manage your own automation, the ideal approach is to remove existing FileStore OSDs from the cluster and recreate them as BlueStore OSDs. Mixing BlueStore and FileStore in the same cluster is not a problem from Ceph's point of view, and all Ceph deployment tools support the ability to create new OSDs. If you already use the `cephadm` deployment tool, which is part of Ceph's own automation environment (Ceph-Mgr), you will find the appropriate commands for replacing OSDs. Before migrating to BlueStore, first update your cluster to the latest Ceph version (17.2 at print). Afterwards, the individual OSDs can be removed at the command line, using the familiar tools, and added back to the cluster as new OSDs. The usual restrictions apply: In smaller installations, you should take care to remove only a few OSDs from the cluster at a time to avoid massive recovery operations on the back end. Either way, you end up having to copy all the objects back and forth within the cluster several times to replace all the OSDs. If you trigger this operation for too many OSDs at the same time, you run the risk of disturbing ongoing operations. Neither Ceph nor `cephadm` have explicit options that convert existing OSDs from FileStore to BlueStore. One big advantage of Ceph is that existing OSDs can fail without data being lost or performance suffering. From a developer's point of view, it would be nonsensical to create extra code that enables a trivial operation.

More Visibility

One of the most difficult tasks in operating a Ceph cluster from the administrator's point of view is getting a quick overview of the cluster's status. Much has happened in this respect in recent years. One example

is Ceph Dashboard, which visualizes the cluster's vital parameters and accesses data from Ceph-Mgr. One other thing is also true: If you really want to come to grips with a Ceph cluster, you cannot escape the command line. An excellent example pertains to placement groups (PGs). Under the hood, the RADOS object memory is known to be broken down into several layers before you get to the binary objects themselves. Each binary object belongs to a PG, and PGs are logically isolated from each other with the use of pools, primarily for performance reasons: Huge clusters can quickly grow to several million binary objects, so if RADOS itself were to make decisions such as the OSD on which an object is stored – on a per-object basis – it would affect performance and wouldn't work at all for very large object sets. Placement groups prevent this problem, and for some years now, Ceph has also supported dynamic scaling of PGs per cluster. With the command

```
ceph pg dump
```

you can display the current status of each PG in the cluster, whether it is located on the OSDs specified by the controlled replication under scalable hashing (CRUSH) algorithm and whether a recovery is currently taking place.

In Ceph 17.2, the tool now has a new output column that provides information about whether a PG is currently subject to Ceph's own mechanism for checking data consistency (aka scrubbing). Scrubbing, and especially the more thorough deep scrubbing, can cause significantly slower than usual access to individual PGs. It was not uncommon in the past to initiate an extensive performance analysis, only to realize a few minutes later that the problems had disappeared. The advanced display of `ceph pg dump` helps avoid these problems.

Hyperconvergence Caution

Admins of hyperconverged setups need to pay special attention to the Ceph-Mgr `osd_memory_target_autotune`

parameter in Ceph 17.2, which is set to 0.7 out of the box. On systems with this setting, Ceph-Mgr configures the OSDs to occupy 70 percent of a system's available RAM.

Of course, the purpose of hyperconverged setups is to allow the operation of virtual instances in addition to the Ceph components. They will not be happy if they have to make do with less than 30 percent of the system's available memory: After all, the running kernel and its services also need to use some RAM.

This example shows once again that Ceph developers are very critical of hyperconverged setups, even though Red Hat now officially supports them and markets them as a cost-effective alternative to Ceph-only clusters. Therefore, anyone running such a setup should be sure to limit OSDs to a total of 20 percent of available RAM with,

```
ceph config set mgr mgr/cephadm/2
  autotune_memory_target_ratio 0.2
```

when upgrading to Ceph 17.2.

Compression

As is the wont with Ceph, the individual Ceph components come with a cornucopia of small but subtle changes. The OSDs now support compression for inter-OSD communication, taking into account that many Ceph systems today have CPUs that are far too powerful. However, planners are still reluctant to use the latest network technologies, such as 400Gbps connections, for financial reasons. Ceph clusters that are overflowing their network at rush hour can use this feature to give themselves some breathing space at the expense of their CPUs. However, you should not have overly high expectations. Ultimately, a permanently overloaded network can only be brought to reason with faster hardware.

Improved Overview for Logs

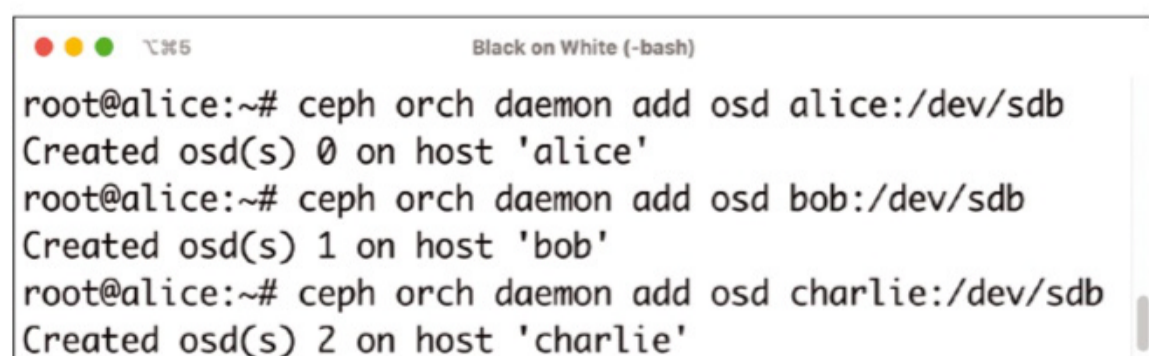
Another change in your OSDs for which to be happy – hidden in the changelog – is that the OSDs will log

slow requests far more clearly than was previously the case. A slow request is basically any write operation to an OSD that does not complete successfully within a specified time period. Virtually any type of problem has the potential to generate many slow requests, whether disruptions on the network, problems with individual OSDs, or configuration errors. The crux is that each client that uploads something to RADOS triggers three write operations in the default configuration. The first goes directly to the target OSD, but from there replicas of the respective objects are also sent to the secondary OSDs, which, in the case of more extensive problems, results in an extremely large number of slow requests in a short period of time that the cluster is unable to process. Until now, RADOS simply forwarded corresponding messages to the `ceph -w` output, where they no longer made sense after a short while.

The new format now bundles the messages, generating far less text on the terminal. If you need the old format (e.g., because monitoring scripts depend on it), you can reinstate the old format in the configuration.

Drilled Out Automation

Although some might turn their noses up at Ceph-Mgr and its cohort `cephadm` (Figure 3), others are happy to use the tool. Both positions are justified. On the one hand, you need to ask why Ceph bothers competing with products like Ansible when alternatives like `ceph-ansible` exist and work well. On the other hand, Ceph-Mgr has evolved into far more than a plain vanilla automation tool.



```
root@alice:~# ceph orch daemon add osd alice:/dev/sdb
Created osd(s) 0 on host 'alice'
root@alice:~# ceph orch daemon add osd bob:/dev/sdb
Created osd(s) 1 on host 'bob'
root@alice:~# ceph orch daemon add osd charlie:/dev/sdb
Created osd(s) 2 on host 'charlie'
```

Figure 3: Ceph has a complete automation suite of its own in `cephadm`, with new features and more data export formats added in Ceph 17.2.

The tool's management services are not only perfectly adapted to Ceph and its needs but now also provide an interface for other programs to retrieve data from Ceph. For example, anyone who wants Prometheus to tap data from Ceph for visualization on modified dashboards (Figure 4) will now also be able to tap into the Manager (Ceph-Mgr), because it provides a native Prometheus interface with data in the appropriate format.

In Ceph 17.2, however, the management component of the solution has only seen minor updates. In addition to querying by way of the Prometheus protocol, for example, a Simple Network Management Protocol (SNMP) interface is now also available for extracting various key figures from Ceph and processing them in other monitoring solutions.

Moreover, the developers have massively expanded the solution's ability to roll out individual Ceph services, such as the Manager component, metadata servers, and instances of the RADOS gateway, in parallel with the same systems. Additionally, the `zap` subcommand can automatically zap OSDs when removed from the cluster, ensuring that the existing data is reliably deleted. Finally, `cephadm` now supports a server agent mode, which should significantly increase the performance of the solution.

Few Filesystem Changes

Once the ugly duckling in the list of Ceph interfaces, CephFS has long since morphed into a full-fledged front end. In Ceph 17.2, however, the changes are minor. Existing filesystems can now be given a new name, but adjustments to Cephx keys must be made manually.

Alternatively, each CephFS instance will have a unique ID in the future, which can also be used to address the file-system.

Changes to the Ceph front end for access over a block device interface are also minor. The Ceph Block Device can be used in several ways right from the start; the `rbd-nbd` option was a new addition to Ceph 16.

It relies on a user-space daemon to wrap a network block device (NBD) in an RBD such that only the `nbd` driver of the Linux kernel plays a role. Although the kernel also has a native RBD driver (`krbd`), users have always struggled with version differences between the cluster and the client, especially on enterprise systems with their often ancient kernels. The `rbd-nbd` client is particularly popular in the Kubernetes environment, where accessing a local block device is still the preferred way to access persistent volumes, even though a native connection from Ceph to Kubernetes exists.

Quality of Service for RGW

In the slipstream of the other front ends for RADOS, the Ceph Object Gateway (aka RADOS Gateway, RGW) has continuously improved over the past years. Ceph 17.2 sees some interesting component changes. The most important change is likely to be the support for bandwidth limitation on the basis of users or buckets. RGW extends RADOS to include an interface for access with the REST HTTP protocol; it emulates either OpenStack's Swift protocol or Amazon S3. In many places, the RADOS gateway is used as a supplement for providers who want to offer their customers

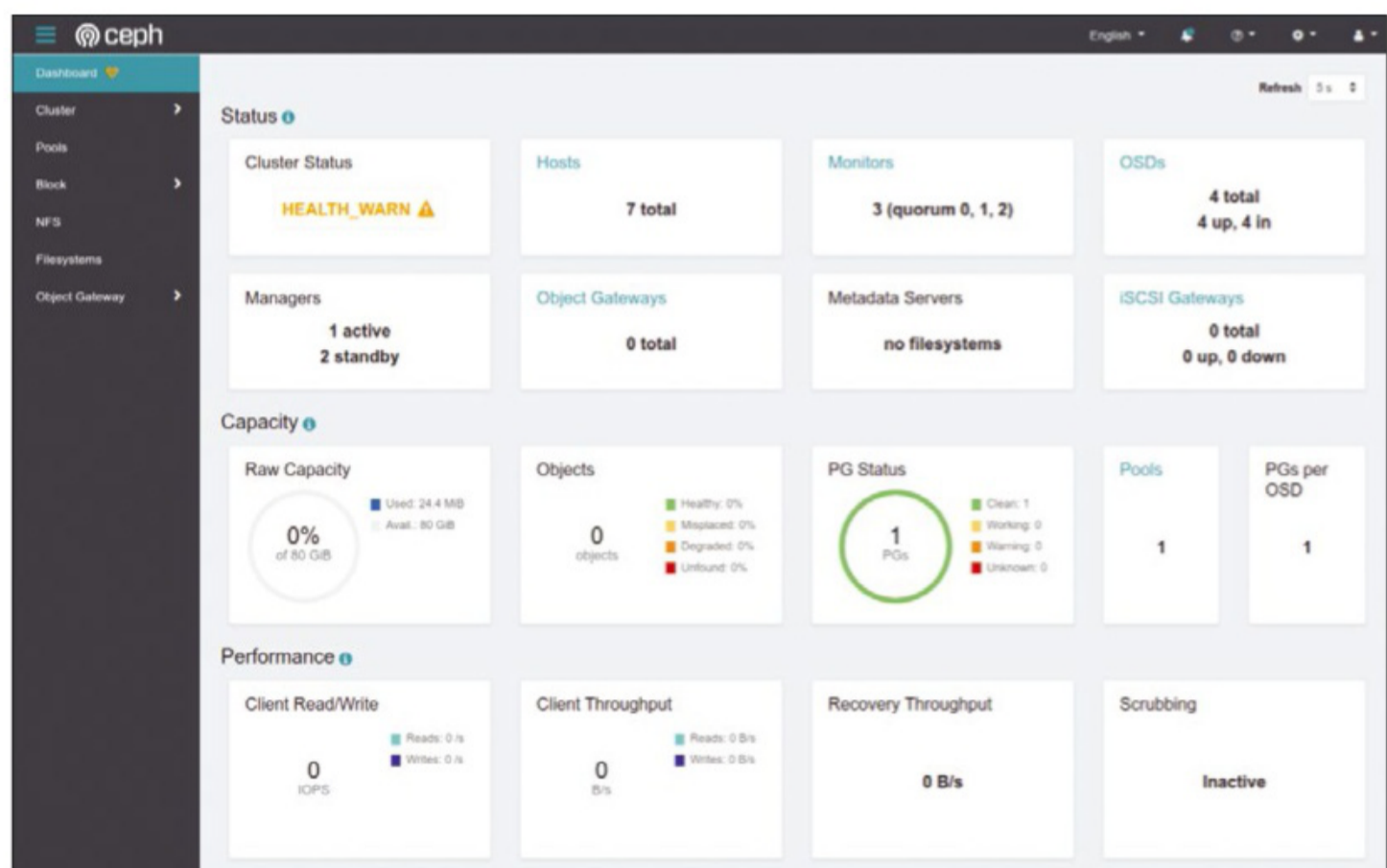


Figure 4: The Ceph dashboard is now an established component of the solution and comes with some minor changes in version 17.2.

on-demand online storage without rolling out their own infrastructure. However, setups without an upstream load balancer have been particularly affected by the risk of individual files accessible on the public Internet taking down the RADOS gateway or even the entire cluster. Mitigation is now possible by defining limits for the respective user or the associated bucket rate – and is also an opportunity for more sales: After all, if a company defines standard limits for all users, turning off one limit per user can be marketed as a feature with a separate price tag.

The Road to Ceph 17.2

All in all, Ceph 17.2 is a robust update without many thrills. For the vast majority of Ceph admins, it primarily offers increased stability and more features with little overhead. If you want to update to the new version, named Quincy, you have several options. The easiest way is to rely on Ceph's own `cephadm` orchestration tool. Upgrading an active Ceph cluster with RocksDB can then be triggered by typing

```
ceph orch upgrade start --ceph-version 17.2.0
```

The described changes still need to be made, depending on the deployment scenario.

If you still roll out your Ceph services manually or use an automation tool (e.g., Ansible), you can look forward to some manual work or a call to the developers for help. The developers of Rook, for example, which makes Ceph operable in Kubernetes, were already working on new Helm charts at the time of going to press, and they might already be available when this issue is published.

In any case, upgrade angst is unfounded; if you got along with Ceph 16, you will not encounter any major problems in Ceph 17.2. ■

Info

- [1] Ceph 17.2: [\[https://docs.ceph.com/en/latest/releases/quincy/\]](https://docs.ceph.com/en/latest/releases/quincy/)
- [2] BlueStore vs. FileStore: [\[https://www.micron.com/about/blog/2018/may/ceph-bluestore-vs-filestoreblock-performance-comparison-when-leveraging-micron-nvme-ssds\]](https://www.micron.com/about/blog/2018/may/ceph-bluestore-vs-filestoreblock-performance-comparison-when-leveraging-micron-nvme-ssds)

The Author

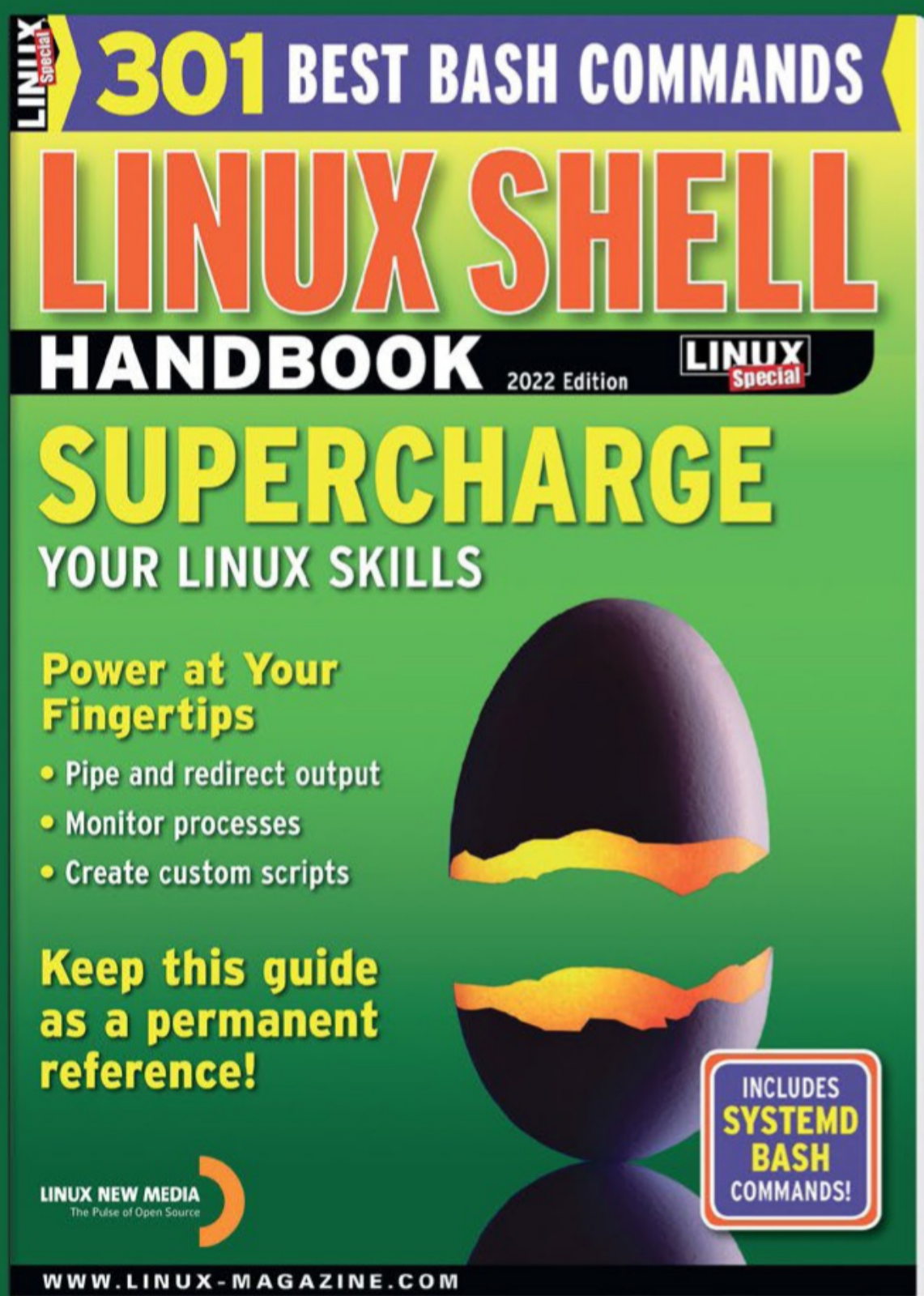
Freelance journalist Martin Gerhard Loschwitz focuses primarily on topics such as OpenStack, Kubernetes, and Ceph.



THINK LIKE THE EXPERTS

Linux Shell Handbook 2022 Edition

This new edition is packed with the most important utilities for configuring and troubleshooting systems.



Here's a look at some of what you'll find inside:

- Customizing Bash
- Regular Expressions
- Systemd
- Bash Scripting
- Networking Tools
- And much more!

ORDER ONLINE:

shop.linuxnewmedia.com/specials

A ptrace-based tracing mechanism for syscalls

Hidden Treasures

The libiotrace library monitors running, static and dynamically linked programs and collects detailed data for many file-I/O-related function calls. By Gerrit Klein, Philipp Koester, and Rainer Keller

Scientific applications can be limited by file I/O because of their distributed nature and implementation of storage. The libiotrace library, extended with ptrace-based syscall tracing, lets users and developers analyze file-I/O-based bottlenecks.

LD_PRELOAD

Dynamic profiling analyzes a program during runtime, thereby gathering information about resource utilization. Often, you want to know which part of a program causes the highest CPU utilization or which task uses how much memory. This information can be used either to optimize the program or to manage a running system and prevent bottlenecks. You can find a multitude of tools for gathering such data.

If the source code of the program is available, you can use a debugger

or an instrumentation framework to gather resource usage information. Most compilers offer an instrumenting framework that allows you to insert profiling functionality during compile time. For example the GNU compiler collection (GCC) offers support for gprof [1], which can be used to analyze the time spent in each part of a program.

If you don't have the source code or don't want to recompile the program in question, you can use the LD_PRELOAD environment variable. Launching a program causes the executable to be loaded into memory and executed as a new process. All the dynamic libraries (so-called shared objects) on which the program depends are loaded into the process. Once all libraries are part of the process memory, the linker collects all exposed functions from the libraries and links them against

function calls. If a function is provided by more than one library, the first match is used for linking. The environment variable LD_PRELOAD allows you to load and link a library before any other library is loaded, so you can use it to get the program to call your own provided implementation of a library function. The libiotrace implementation of the function gathers information (e.g., execution time of the function and function parameters) and writes the collected data to a buffer. It also calls the function that would have been called without LD_PRELOAD by resolving the address of the function with dlsym, which then returns the second match from all loaded libraries. The set of tools in gperftools [2] uses LD_PRELOAD, as well.

libiotrace: Just Another Profiling Tool?

A typical CPU computes data faster than data can be fetched from or stored to main memory (the so-called memory wall) [3]. Storage

only exacerbates this problem. File I/O can therefore be a highly relevant factor for program optimization. The libiotrace [4] library uses LD_PRELOAD to gather data about POSIX [5] and MPI [6] file I/O functions. Although other tools, such as Darshan [7], use this method too, libiotrace adds live tracing support, as opposed to the pure postmortem analysis of most tools.

A typical tracing setup for libiotrace is shown in Figure 1, which illustrates how LD_PRELOAD is used to “insert” libiotrace between the profiled program and *libc*.

Typically, the data is either written to a logfile, sent to an InfluxDB instance, or both. InfluxDB serves as a data source for Grafana for near real-time visualization.

Supplemental Profiling with ptrace

During work on libiotrace, a few instances of file I/O couldn’t be traced by wrapping the file I/O functions with LD_PRELOAD. This file I/O uses function calls that are obviously not dynamically linked against the libiotrace wrappers during the start of the executable. Tracing this file I/O required further investigation. Running the program with strace revealed that the file I/O in question does in fact call the POSIX function `open64`. The same behavior could be observed in a debugger. A closer look at the function call in the stack trace of the debugger showed the root cause: The function call wasn’t going through

the procedure linkage table (PLT), which is generated during compile time to enable relocations of function addresses. Each entry in the PLT is a stub function that is called instead of the function itself (located in a shared object).

During runtime, the dynamic linker searches the function address in the shared object and provides it to the stub function in the PLT. The stub function then uses this address to call the function in the shared object file. A PLT entry is thus a layer of indirection used to relocate function calls in a single place per loaded object. A disassembly of the used library proved that `open64` had no PLT entry in this library. An example of this situation can be found in the implementation of `dlopen` in *libdl* (the linker itself).

The `dlopen` function can be used by a program to load a dynamic library. In fact, this function is used by the linker itself to load and link shared object files. To open and read a shared object file, `dlopen` calls an implementation of `open` or `open64` (on a 64-bit system), which are both part of the *libc* library. A call to `dlopen` can happen before the dynamically linked *libc* is available (e.g., during the loading process of *libc*).

Therefore, a relocation of `open` or `open64` during a call of `dlopen` is not feasible, so the `dlopen` implementation ships with its own statically linked version of the `open` functions. On further research, even more libraries with statically linked versions of POSIX file I/O functions were found (e.g., some versions of the *libpthread* library).

To make matters worse, other ways can prevent linking against a function loaded by LD_PRELOAD. For example, the linker options `-Bsymbolic` and `-Bsymbolic-functions` can ensure that a call to a function will use a local function inside a shared library and not a function exposed from a different object. Furthermore, flags for `dlopen` (`RTLD_NOW` and `RTLD_DEEPBIND`) can change the order in which dynamic libraries are searched for functions to link. In conclusion, LD_PRELOAD is not sufficient if you want to profile all file I/O. Enter syscall tracing.

Syscall Tracing

The libiotrace library extends tracing beyond function call wrapping by adding support for syscall tracing, which has upsides and downsides. The upside is that no file I/O will escape the library. File I/O services can only be requested from the kernel through the syscall interface. Hence, whether the application invokes a syscall manually with inline assembly or by calling a *libc* wrapper, the file I/O event will be traceable.

The downside is that mapping the observed syscall to a function call in the log of the traced application is not always feasible. For syscall tracing purposes, *libc* file I/O functions can be categorized into two categories: (1) I/O functions that are merely syscall wrappers (e.g., `write`, documented in section 2 of the system calls man pages) and (2) functions that provide additional buffering in the user space (e.g., `fwrite`, documented in section 3 of the library functions man pages). The second type makes mapping on the basis of timestamps unfeasible because those functions won’t immediately trigger a syscall, unlike the first type.

The Price of Tracing

Linux syscalls used to be fairly cheap (on the order of hundreds of clock cycles); however, since mitigations for CPU vulnerabilities (e.g., Spectre and Meltdown), syscalls have become more

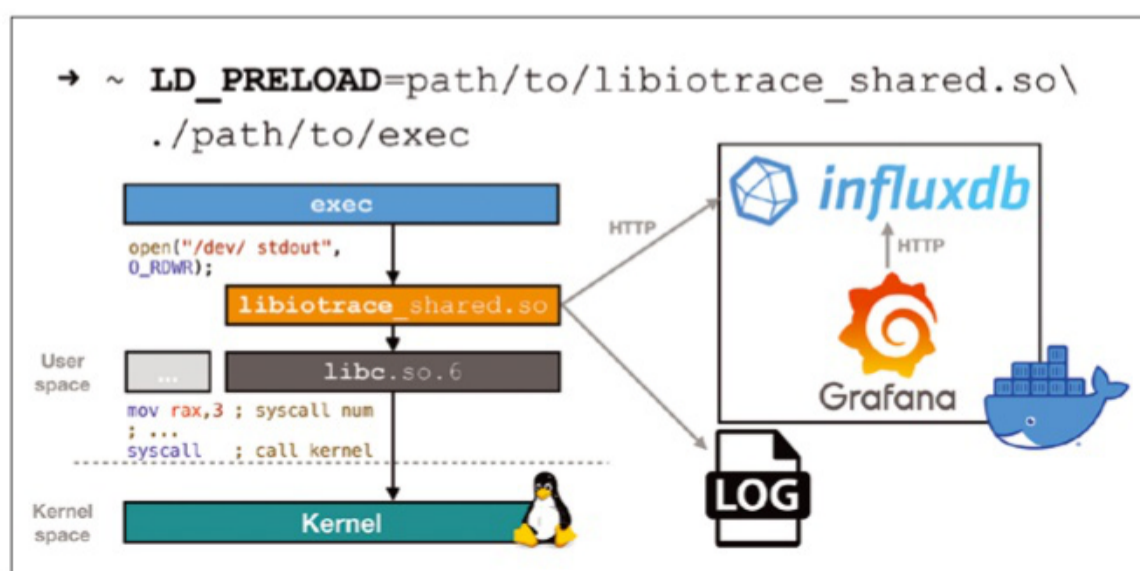


Figure 1: The libiotrace tracing setup.

Table 1: x86-64 Syscall Excerpt				
%rax	System Call	%rdi	%rsi	%rdx
0	sys_read	unsigned int fd	char *buf	size_t count
1	sys_write	unsigned int fd	const char *buf	size_t count
2	sys_open	const char *filename	int flags	int mode
3	sys_close	unsigned int fd		
4	sys_stat	const char *filename	struct stat *statbuf	
5	sys_fstat	unsigned int fd	struct stat *statbuf	

expensive (on the order of thousands of clock cycles). Tracing syscalls however is even more expensive. The incurred slowdown depends on many factors: for instance, how many syscalls the application triggers; how many processes, threads, or both are being traced; and whether the stack is unwound. According to benchmarks with only one tracee, the additional overhead in terms of execution time for syscalls (caused by the libiotrace syscall tracing implementation) is approximately 1,400%. This massive slowdown can be attributed mostly to the expensive stack unwinding operation.

Goals

Keeping these limitations in mind, the following goals were set for the development of the libiotrace’s syscall tracing implementation, called “stracer”:

- Detect missed I/O events. Stracer should detect whether libiotrace traced the function call that triggered the current syscall. If the function isn’t traced, a warning is written to console and logfile.
- Pass traced “syscall events” to libiotrace. As a dynamically linked library, libiotrace uses the same address space as the traced application. The stracer, however, will

run as a separate process because the stracer should not be traced by libiotrace. Thus, an interface is used for passing traced syscalls to libiotrace, which then updates mapping data for file handles.

Kernel Facilities for Tracing

Commonly used tracers under Linux include ptrace, eBPF, and bpftrace, which is based on eBPF. For implementing stracer, we use ptrace because it is supported by default, unlike eBPF, which requires a kernel configuration. Also, more online resources are available for ptrace. The ptrace (process trace) syscall, available on many Unix-like systems, allows you to set breakpoints on syscalls for your tracee. Once the tracee hits a breakpoint, it stops, allowing you to gather information about the syscall by reading data from the tracee’s address space with ptrace. Finally, once stracer has finished analyzing the current syscall, it can set a breakpoint on the next syscall and resume the tracee. To grok the current syscall, you need to investigate (1) which syscall you’re dealing with and (2) the kind of data (i.e., arguments) the particular syscall provides. Thus, you need syscall definitions that contain this information.

Parsing Syscalls

Each syscall has a unique number, which is passed by the caller in CPU register *rax* on x86-64 machines [8]. Syscall arguments are passed in registers *rdi*, *rsi*, *rdx*, *r10*, *r8*, and *r9*. The return value of the syscall is returned

to the caller in *rax*. For instance, the open syscall (Table 1), is numbered 2, which goes in *rax*, and its arguments *filename*, *flags*, and *mode* are passed in *rdi*, *rsi*, and *rdx*. Unfortunately, there’s no C header file that specifies the syscall

number including arguments and data types for all syscalls; hard coding these values is problematic because syscalls vary for different CPU architectures and can change from kernel version to kernel version. Thus, the syscalls have to be parsed from the Linux kernel source. To ensure that the source code corresponds to the system’s current kernel release, use `apt source linux` for retrieving the code.

A Python script developed for this purpose then parses the file `arch/x86/entry/syscalls/syscall_64.tbl` (Listing 1) with regular expressions (regex). Each line without a comment marker contains the syscall number, application binary interface (ABI), name, and entry point of one syscall. Normally, only the syscall number and name would be relevant. However, some syscalls (e.g., `writew`) exist for both the 32-bit and 64-bit ABIs. Thus, for generating unique C preprocessor macros for each syscall, the ABI must also be considered.

Lastly, the missing arguments of each syscall are required. The solution to this task was inspired by the `minis-trace` parsing script by nelhage [9]. The script first searches for C source files in the `fs`, `include`, `ipc`, `kernel`, `mm`, `net`, and `security` directories, as well as in architecture-specific directories. After all source files have been found, the script regex combs through each line, searching for `SYSCALL_DEFINE` macros. For example, the `SYSCALL_DEFINE` for the `write` syscall in source file `fs/read_write.c` is:

```
SYSCALL_DEFINE3(write, 2
    unsigned int, fd, 2
```

Listing 1: Excerpt of Parsed File				
#	64-bit system call numbers and entry vectors			
#	The format is:			
#	<number> <abi> <name> <entry point>			
#				
0	common	read	sys_read	
1	common	write	sys_write	
2	common	open	sys_open	
#	...			

```
const char __user *, buf, 2
size_t, count) {
    return ksys_write(fd, buf, count);
}
```

The SYSCALL_DEFINE suffix (3 in this case) refers to the number of arguments the syscall expects. Because syscalls can have up to six arguments, Linux provides seven of these macros (i.e., SYSCALL_DEFINE0 through SYSCALL_DEFINE6). The first argument of the macro is the name of the syscall – in this case write. The syscall arguments ensue, wherein each argument is split in the data type (e.g., unsigned int and the argument name fd).

Once such a macro has been matched, the name and the arguments are extracted with regex capture groups. In the final step, the syscall numbers from the first step are merged with the arguments and names from the second step on the basis of the syscall name. This data is then used to generate a lookup table for syscalls (Listing 2).

The stracer can now easily index into the table with the syscall number and retrieve type information for the arguments or simply the syscall name.

Ptrace Tracing Roles

The most common tracing setup involves the parent process as tracer (Figure 2). The parent, which acts as the tracer, forks itself and waits for the child. The child sets up ptrace and sends itself SIGSTOP to stop its execution. After the child has stopped, the parent has the opportunity to set ptrace options and, for example, set a breakpoint on the next syscall. Setting this breakpoint will resume the execution of the tracee until the next breakpoint is hit. The tracee (i.e., the child) proceeds with the execve syscall, replacing the executing image.

Many GNU/Linux distributions use the Yama Security Module to restrict tracing in the ptrace scope. Thus, tracing as parent process has the benefit of avoiding most issues related to tracing permissions, which

can become an issue when roles are reversed, requiring the tracee to set additional tracing permissions. That said, a downside to tracing as parent process is that Unix signals sent by other processes will be addressed to the tracer and not to the tracee, thus preventing the correct delivery of the signal. Alternatively, you can also trace as child, effectively reversing roles. This choice, however, could interfere with the tracee’s process execution because it now has an unexpected child process (the tracer).

To avoid both issues, the stracer is “daemonized,” similar to the -DD option in strace. The relevant setup steps are depicted in Figure 3.

The parent process, which will later act as tracee, first sets the required permissions for tracing with prctl and then forks itself and waits to be

```
Listing 2: Excerpt of Syscall Lookup Table

const syscall_entry_t syscalls[] = {
    [__SNR_read] = {
        .name = "read",
        .nargs = 3,
        .args = {ARG_INT, ARG_STR, ARG_INT, -1, -1, -1}},
    [__SNR_write] = {
        .name = "write",
        .nargs = 3,
        .args = {ARG_INT, ARG_STR, ARG_INT, -1, -1, -1}},
    [__SNR_open] = {
        .name = "open",
        .nargs = 3,
        .args = {ARG_STR, ARG_INT, ARG_INT, -1, -1, -1}},
    // ...
}
```

attached by the stracer. The resulting child forks itself again and waits until it gets terminated by the grandchild. Finally, the grandchild, which will become the stracer process, kills the

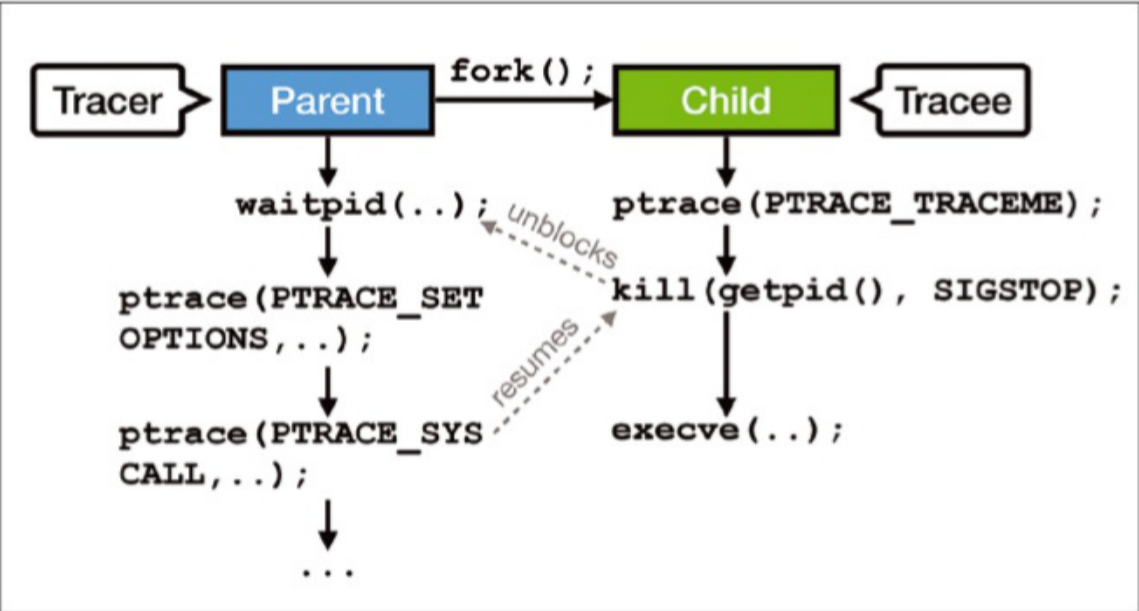


Figure 2: Parent as tracer and child as tracee.

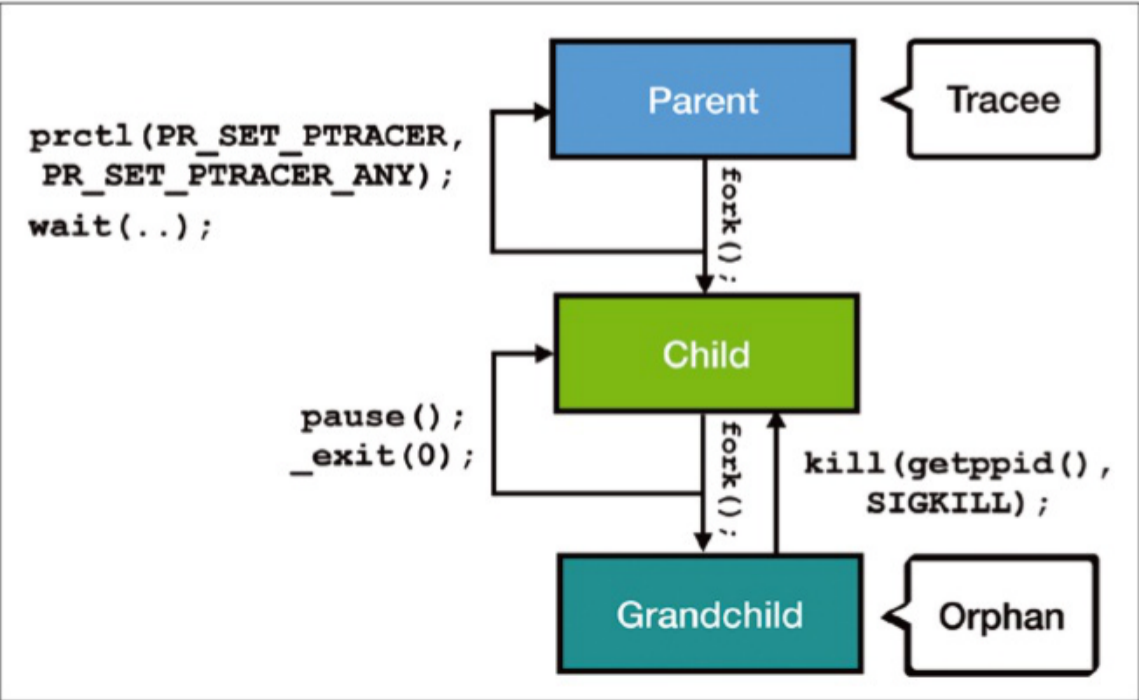


Figure 3: Setting up a daemonized tracer.

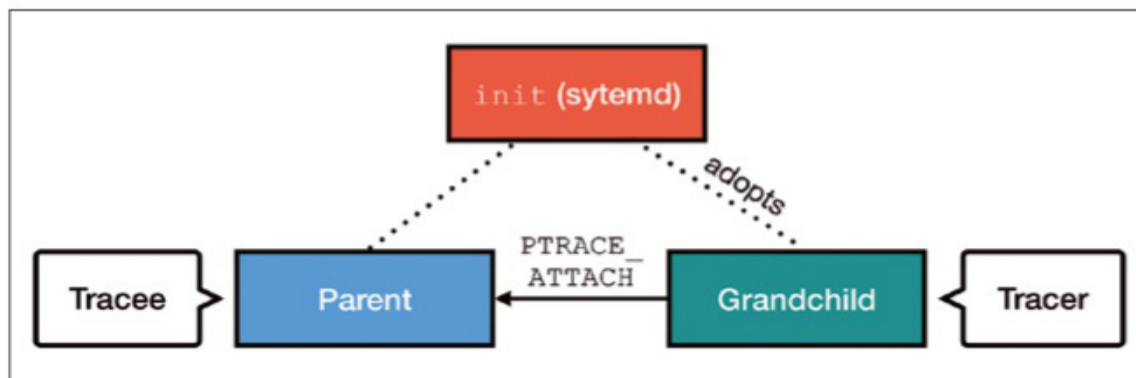


Figure 4: Resulting process hierarchy.

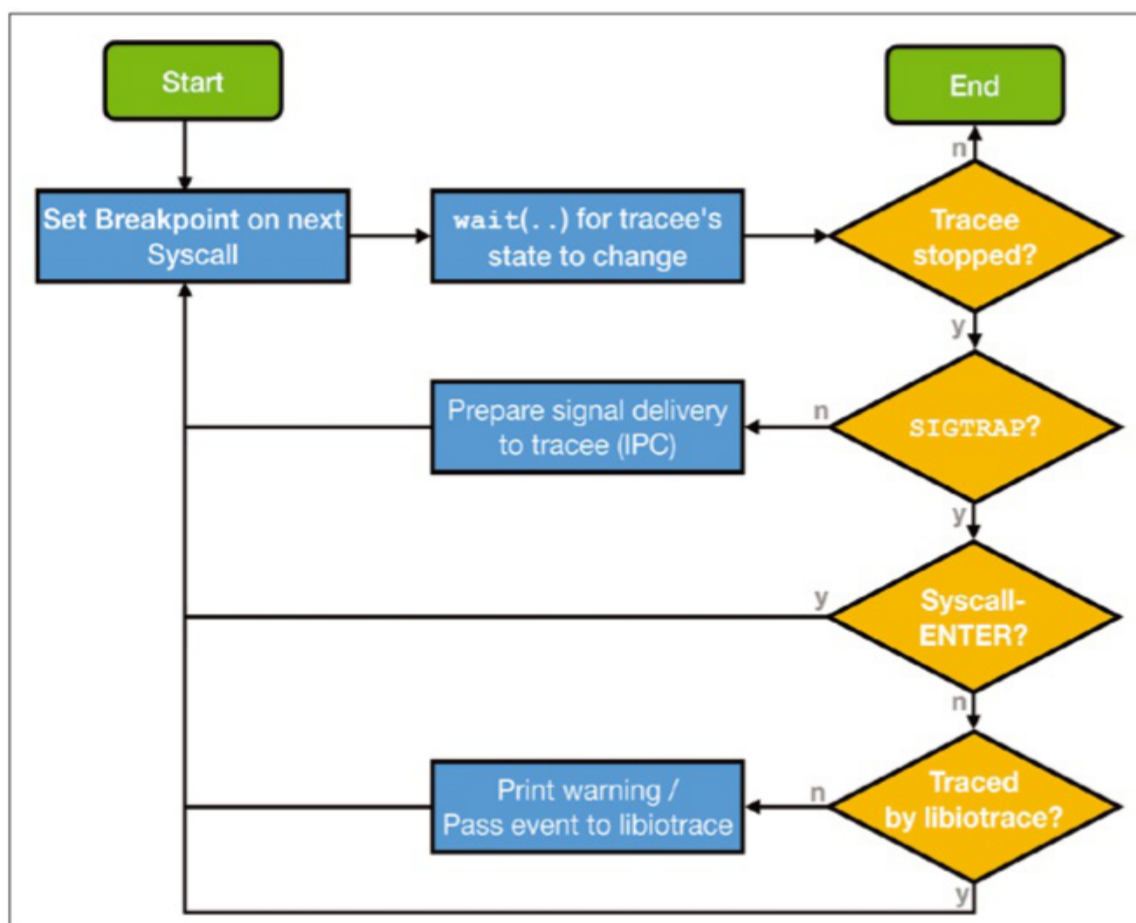


Figure 5: Simplified overview of the tracing workflow. Start and End pertain to the life cycle of the tracee.

Listing 3: Simplified Stack Unwinding

```

01 /* - Init libunwind - */
02 unw_cursor_t unw_cursor;
03 unw_context_t *unw_ctx = _UPT_create(tid);
04 unw_init_remote(&unw_cursor, g_unw_as, unw_ctx);
05
06 /* - Search - */
07 bool found_stack_entry = false;
08 do {
09     unw_word_t ip = 0;
10     unw_get_reg(&unw_cursor, UNW_REG_IP, &ip);
11
12     Dwfl_Module* module = dwfl_addrmodule(dwfl, (uintptr_t)ip);
13     const char *module_name = dwfl_module_info(module, 0, 0, 0, 0, 0, 0, 0);
14     if (! strstr(module_name, stacktrace_module_name) ) {
15         continue;
16     }
17
18     found_stack_entry = true;
19     break;
20 } while (unw_step(&unw_cursor) > 0);
  
```

child and becomes an orphan process. After some time, the grandchild will be adopted by the init process (Figure 4).

As a last step, the stracer leaves the process group of the tracee by calling `setpgid(0, 0);`, which ensures that no signals delivered to the tracee's process group are delivered to the stracer. The stracer can now start attaching tracees via the ptrace request `PTRACE_ATTACH`.

Tracing Workflow

Once the tracer and the first tracee have been set up, tracing commences. The stracer may trace multiple processes and threads. A simplified version of the tracing workflow is visualized as a flow-chart in Figure 5.

Initially, the stracer sets the first breakpoint for the stopped tracee, identified by `tid` in:

```
ptrace(PTRACE_SYSCALL, tid, 0, 0,
      pending_signal);
```

The ptrace request `PTRACE_SYSCALL` also allows passing signals (sent by other processes) to the tracee. After the request has finished, the tracee will continue executing until it hits the next breakpoint set by the

`PTRACE_SYSCALL` request.

The stracer checks subsequently in a non-blocking way for tracees that have changed state with:

```
waitpid(-1, &trapped_tracee_status, WNOHANG);
```

State changes occur when

- a tracee hits a breakpoint,
- a signal has been sent to a tracee, or
- a tracee has terminated.

The kernel stops the execution when such a state change occurs by sending a `SIGTRAP` to the affected tracee. The tracer then has the opportunity to inspect the stopped tracee's state with `ptrace`.

Whether the tracee has stopped or terminated can be easily checked with the macro `WIFSTOPPED(tid)`. If the tracee has terminated, the stracer will decrement its tracee count and check again for a new tracee that changed its state. Otherwise, the tracee is stopped. A stop signal of a type other than `SIGTRAP` indicates that the tracee was stopped because of a signal sent by a different process. The stracer delivers this signal

to the tracee when the next breakpoint is set.

A SIGTRAP signal indicates that the tracee has hit a breakpoint. A syscall involves a syscall enter and a syscall exit. Hence, it also must be checked whether the breakpoint was hit on a syscall enter or on a syscall exit. This event can be examined by retrieving the register contents with

```
ptrace(PTRACE_GETREGSET, tid, 2,
      NT_PRSTATUS, &iovec);
```

and checking whether the *rax* register contains the negative value of the *errno* constant *ENOSYS* as the return value. The tracee is in a syscall exit if the register content doesn't match the negative *errno* constant. However, there's one small catch: The syscall number, originally stored in the *rax* prior syscall enter, already will be overwritten with the return value at this point in time. Thankfully, *ptrace* preserves this value as *orig_rax*, so it is still accessible at syscall exit. If the current stop was indeed induced by a syscall exit, the stracer will check whether *libiotrace* traced the function call. To accomplish this, the execution stack of the stopped tracee must be unwound. If the stack has no entry indicating the function call was traced by *libiotrace*, the stracer will print a warning and pass the syscall event to *libiotrace*, and the flowchart cycle starts anew.

Execution Stack Unwinding

The libraries *libunwind* and *libdw* handle the execution stack unwinding of the remote process. A simplified version for dynamically

linked *libiotrace*, without any error checking or cleanup, can be seen in [Listing 3](#).

After initialization of *libunwind*, the code iterates over each stack frame. Once the current address of the instruction pointer has been retrieved, the module name is resolved. The module name is the shared object (.so) file, from which the function was loaded. For each stack frame, this module name is compared with *libiotrace_shared.so*. With a match, it is evident that the function call was traced by *libiotrace* ([Figure 6](#)).

In this example, *libiotrace_shared.so* was in the stack trace. Thus, the *LD_PRELOAD* symbol for *mmap* has been correctly resolved.

The *libiotrace* tool can also be statically linked with the target program. For this specific purpose, the *libiotrace* wrapper functions are prefixed with *__wrap_*, which allows insertion of functions (much like *LD_PRELOAD* does) by using the linker option *--wrap=symbol*. In this case, the function name also has to be checked for the prefix when unwinding the stack.

IPC with libiotrace

Finally, you should understand the employed interprocess communication (IPC) mechanisms. The simplest way of tracing entire process hierarchies is by tracing the “root process,” which creates all descendants, and setting the *ptrace* option *PTRACE_0_TRACECLONE*. However, this option is not always desirable; for example, for scientific applications, the root process is often Open MPI's *orted*, which is responsible for spawning MPI processes and shall not

be traced. Therefore, a registration mechanism is needed that allows processes to request tracing from the stracer through a Unix domain socket. Also, the socket ensures that only one stracer instance runs at any given time because only one process can bind to the socket.

The second employed IPC mechanism is the already mentioned interface for passing syscall events from the stracer to the running tracee, which is accomplished through the use of shared memory. Every tracee sets up a shared memory object, where the name of the object is derived from the *tid* of the current tracee process. A simplified version without error checks is shown in [Listing 4](#).

The tracee requests to be traced by the stracer through the aforementioned Unix domain socket, and the stracer maps the shared memory region in its virtual memory and sends an acknowledgment to the tracee. The resulting shared memory segments for each tracee is visualized in [Figure 7](#).

As already mentioned in the Tracing Workflow section, when a syscall event isn't traced, the stracer writes the event into the shared memory segment of the current tracee. Once the tracee has resumed its execution, it reads and processes the contents from the shared memory segment. Internally, a ring buffer implementation from GitHub user *rmind* [10] is currently used, which supports multiproducer single-consumer operation without the use of locks. Sadly, we weren't able to find a suitable implementation that supports multiconsumer; otherwise, just one memory mapping could be used for all tracees.



Figure 6: The *libiotrace* shared library file was found in the execution stack.

Future Work

The stracer currently supports only x86-64, although we’ve already experimented with aarch64 support. However, ptrace delivers the wrong syscall numbers during tracing.

Because x86-64 is more widely used, we decided to focus on implementing the stracer first for x86-64 and add aarch64 support at a later point. The biggest issue so far, however, has been the availability of lock-free data structures for C. The lack thereof has

been a limiting factor for many of our development endeavors. The syscall event interface is just one of many examples for this issue. A next step might be the implementation of a lock-free multiconsumer ring buffer.

Listing 4: Shared Memory for Syscall Events

```
01 int smo_fd = shm_open(smo_name, O_RDWR | O_CREAT, S_IRUSR | S_IWUSR);
02 ftruncate(smo_fd, smo_min_len);
03
04 struct stat stat_info;
05 fstat(smo_fd, &stat_info);
06 *shared_mem_len_ptr = stat_info.st_size;
07 *shared_mem_addr_ptr = mmap(NULL, *shared_mem_len_ptr, PROT_READ | PROT_WRITE, MAP_SHARED, smo_fd, 0);
```

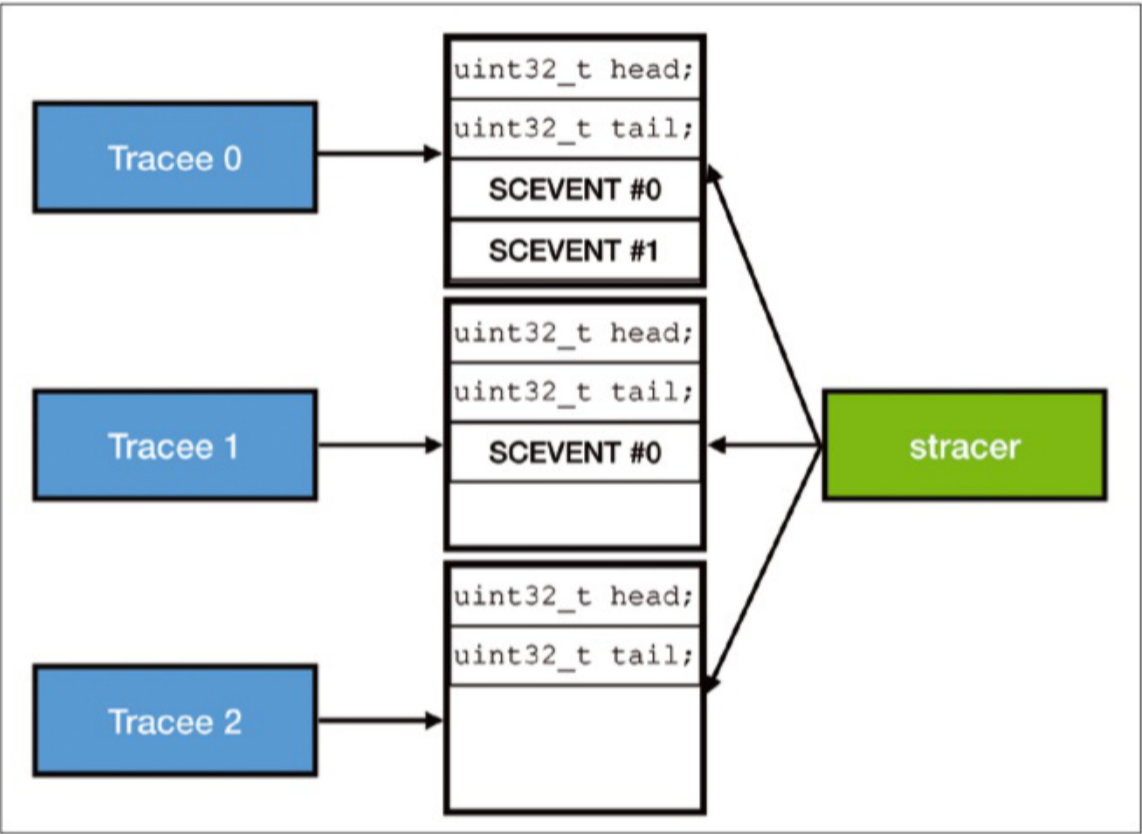


Figure 7: Shared memory segments for each tracee.

Info

- [1] gprof: [\[https://sourceware.org/binutils/docs/gprof\]](https://sourceware.org/binutils/docs/gprof)
- [2] gperftools: [\[https://github.com/gperftools/gperftools\]](https://github.com/gperftools/gperftools)
- [3] McKee, Sally A., and Robert W. Wisniewski. "Memory wall." In: D. Padua, ed. *Encyclopedia of Parallel Computing*. Springer, Boston, MA. 2011. doi:10.1007/978-0-387-09766-4_234
- [4] libiotrace: [\[https://github.com/hpcraink/fsprj2\]](https://github.com/hpcraink/fsprj2)
- [5] POSIX: [\[https://pubs.opengroup.org/onlinepubs/9699919799\]](https://pubs.opengroup.org/onlinepubs/9699919799)
- [6] MPI: [\[https://www.mpi-forum.org\]](https://www.mpi-forum.org)
- [7] Darshan: [\[https://www.mcs.anl.gov/research/projects/darshan/\]](https://www.mcs.anl.gov/research/projects/darshan/)
- [8] Linux 4.7 x86-64 syscalls table: [\[https://blog.rchapman.org/posts/Linux_System_Call_Table_for_x86_64\]](https://blog.rchapman.org/posts/Linux_System_Call_Table_for_x86_64)
- [9] minitrace: [\[https://github.com/nelhage/minitrace\]](https://github.com/nelhage/minitrace)
- [10] rmind, ringbuf: [\[https://github.com/rmind/ringbuf\]](https://github.com/rmind/ringbuf)

Authors
 Gerrit Klein is working on his Masters in Applied Computer Science, Philipp Koester is pursuing his PhD, and Rainer Keller is a Professor of Computer Science, with a focus on high-performance computing.

Public Money

Public Code



Modernising Public Infrastructure with Free Software



Free Software Foundation Europe

Learn More: <https://publiccode.eu/>



Policy rulesets in cloud-native environments

Just Enough

What a user is allowed to do in a program is usually defined by a role model, which often poses numerous challenges, especially in the cloud or for infrastructure as code. The free Open Policy Agent offers a flexible way to manage user rights. By Guido Söldner

Infrastructure as Code (IaC) has become a successful recipe for declarative, machine-readable code, so it only makes sense to apply this system to security and, in particular, to authoring policies in an attempt to implement rules within an organization in a scalable way. One representative of this genre that has recently received greater attention is the Open Policy Agent (OPA) project [1], which is backed by startup Styra. OPA is a general-purpose policy engine that enables consistent, context-aware policy enforcement across the stack.

OPA at a Glance

OPA is hosted by Cloud Native Computing Foundation (CNCF), the organization behind Kubernetes. Designed for cloud-native environments, OPA combines the relatively easy-to-learn and -read Rego policy language with a policy model and application programming interface (API), which allows for a kind of universal framework that applies rules to any kind of stacks. One of

the great advantages of OPA is the ability to decouple security policies from code and its use – regardless of how often the code changes. From a technical point of view, OPA is tied to the input. Once data is available, the OPA code decides how to handle it (e.g., allowing or blocking with an allow or deny policy). Another advantage is that OPA processes take input and create output in both JSON and YAML formats, meaning that IT managers do not have to stick to a predefined API. All told, writing rules is relatively easy, and OPA supports read, evaluate, print, and loop (REPL, i.e., shell-based code execution). Of practical value is that you do not have to write all the policies yourself, because you can easily find ready-made policy bundles online for many use cases, and they are likely to contain a useful, predefined set of rules. A freely accessible Playground [2] and a free Styra Academy [3] can help you learn. As expected, OPA use cases all relate to security:

- The expressiveness of OPA and Rego make it easy to define rules

for user authorization and keep them in a central location.

- OPA can help with authorization if you use API gateways such as Kong, Traefik, or Tyk.
- OPA has various areas of application for continuous integration and continuous delivery or deployment (CI/CD), such as checking IaC code against a ruleset. Typical examples include preventing public IP addresses on virtual machines.
- OPA plays a major role in the Kubernetes environment. Admission controllers can send requests to OPA to receive a decision on which resources can be deployed in a Kubernetes cluster.

One hurdle you might encounter is having to relearn Rego. OPA is based on the Go programming language, and the only client SDK is for Go at the moment.

Installing OPA

OPA is a single, quickly installed binary file. It supports Linux, macOS, and Windows. On Linux, you install the binary, change the file attributes

to make it executable, and add the location of the OPA binaries to your PATH variable:

```
curl -L -o opa 2
https://openpolicyagent.org/downloads/
latest/opa_linux_amd64
chmod 755 opa
export PATH=<path to the OPA binary>
```

If you work with Visual Studio Code, you should install a plugin by selecting the *Extension* menu in the left pane and searching for *Open Policy Agent*. When found, click *Install* (**Figure 1**). Once you have installed the extension, you will be able to create a new file in Rego with rules such as:

```
package hello
default allow_hello = false
default allow_world = false
allow_hello {
    "hello" != ""
}
allow_world {
    "world" != "world"
}
```

To evaluate the rule, run *View | Command Palette | OPA Evaluate Package*. A second tab then opens to display the results. The results on the right side are easy to understand: To begin, the `allow_hello` and `allow_world`

variables are set to `false`, then their values are checked in a function.

Working with Variables, Objects, and Functions

To run simple expressions, you can start an interactive shell from the terminal with the `opa run` command and then run commands. For example, begin by defining some variables,

```
greeting := "Hello"
max_height := 42
pi := 3.14159
allowed := true
location := null
```

then output the values (in an array, if so desired),

```
[greeting, max_height, pi, 2
allowed, location]
[
    "Hello",
    42,
    3.14159,
    true,
    null
]
```

and define objects:

```
ips_by_port := {
    80: ["1.1.1.1", "1.1.1.2"],
```

```
    443: ["2.2.2.1"],
}
ips_by_port[80]
[
    "1.1.1.1",
    "1.1.1.2"
]
```

This example first defines an object that comprises sub-elements, including an array of strings for defining IP addresses. The second block accesses those elements:

Rules are an important element in Rego. In the next example, the default value is `false`. If an input has a role value equal to `admin`, the value changes to `true`. Alternatively, you can change the default if the role is `user` and the `has_permissions` field is `true`:

```
default allow = false
allow = true {
    input.role == "admin"
}
allow = true {
    input.role == "user"
    input.has_permission == true
}
```

Functions can also be defined, as in this code fragment, which implements and calls a multiply function:

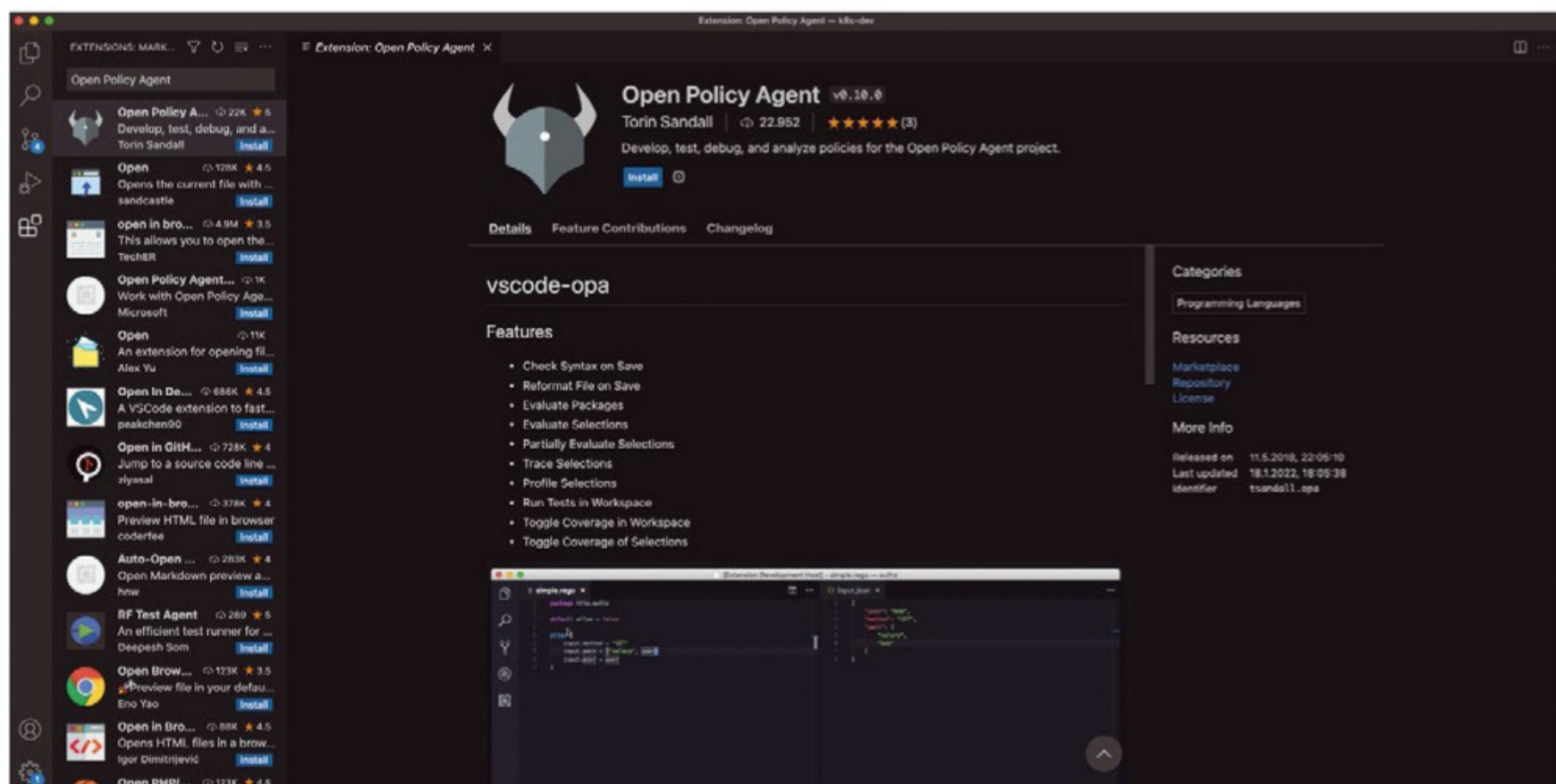


Figure 1: The OPA plugin for Visual Studio Code makes life easy.

```
package function
multiply(a,b) = m {
  m := a*b
}
result1 = r {
  r := multiply(3,4)
}
result2 = r {
  r := multiply(3,9)
}
```

The Rego programming language has native functions for:

- numbers
- bit manipulation
- aggregation
- sets (array, set, object)
- strings
- regular expressions
- HTTP
- JSON web tokens (JWT)

Listing 1 shows some function call examples.

Policy Example

The language constructs shown let you define policies to detect anomalies, incorrect configurations, or instances of poor practice. The code in **Listing 2** shows an example of how to identify Amazon Web Service (AWS) users who do not use multifactor authentication (MFA) when logging in. After you have exported this input from AWS and saved it in the input.json file, you can use it as input for a policy:

```
package match
  is_array(input.MFA)
  count(input.MFA) > 0
}
match {
  not active_with_mfa
}
```

This example looks to see whether users have stored MFA devices.

OPA and Kubernetes

One of the best known use cases for OPA is on Kubernetes. From a technical point of view, you install OPA as a pod in the cluster. The admission controller then forwards the user’s request to the address stored in the

Listing 1: Example Function Calls

```
# Rounding up
# 11
round(10.9)
# Generate an array of numbers
# [9, 8, 7, 6, 5, 4, 3, 2, 1]
numbers.range(9, 1)
# The type is "array"
# "array"
type_name(["apple","banana","pineapple"])
# The input is a set
# true
is_set({1,2,3})
# The object includes two elements
# 2
count({"width":1024, "height":768})
# [1,2,3,4,5,4,5,6]
array.concat([1,2,3,4,5],[4,5,6])
# Extract key3, because this is not part of the object; a "val3" value is returned
obj := {"key1":"val1", "key2":"val2"}
# "val3"
object.get(obj, "key3", "val3")
# Ayt what position does the word "world" occur?
# 7
indexOf("Hello, world", "world")
# POST HTTP call with header and body
http.send({"url":"http://httpbin.org/post", "method":"post", "timeout":"3s", "headers":{"token":"111"},
  "body":{"key": "val"}}})
```

Listing 2: AWS Users Without MFA

```
01 [
02   ....
03   {
04     "Path": "/",
05     "UserName": "liav",
06     "Arn": "arn:aws:iam::123456789:user/ferdinand",
07     "CreateDate": "2016-07-27 23:53:34+00:00",
08     "MFA": [
09       {
10         "UserName": "ferdinand",
11         "SerialNumber": "arn:aws:iam::123456789:mfa/ferdinand",
12         "EnableDate": "2021-04-25 09:00:38+00:00"
13       }
14     ],
15     "Groups": [
16       {
17         "GroupName": "RND-Admins"
18       }
19     ]
20   },
21   {
22     "Path": "/",
23     "UserName": "guido",
24     "Arn": "arn:aws:iam::123456789:user/guido",
25     "CreateDate": "1979-10-07 19:53:34+00:00",
26     "MFA": []
27     "Groups": [
28       {
29         "GroupName": "DevOps-Admins"
30       }
31     ]
32   },
33   ....
34 ]
```

validating or mutating webhook. The app responding on this address mutates the received manifest or sends back an allow or deny response. Both are enforced by the admission controller.

The controllers are – to put this briefly – responsible for checking incoming (and authenticated) API requests and either allowing or rejecting them depending on the result. Internally, the admission control process has two phases: a mutating phase, in which requests can be modified, and a validation phase for verification. An admission controller variant can intervene at either of these two points. One example of such a controller is `LimitRanger`, which provides pods with the default request resources while not allowing pods to grab more resources than intended by the limit. Of the more than 30 default admission controllers, two are worth special attention: `ValidatingAdmissionWebhooks` and `MutatingAdmissionWebhooks`. They do not implement any logic themselves but let you register REST endpoints of a service running in the cluster.

OPA is used as an admission controller, as well, and can thus configure a number of security features in the cluster by:

- defining that all containers must have sidecars (e.g., to perform logging or auditing tasks),
- providing all resources with annotations,
- modifying the container image definition so that images can only

be loaded from a defined image registry, and

- setting node, pod affinity, and anti-affinity selectors for deployments.
- Installing OPA in Kubernetes is quite easy [4]. The current Gatekeeper major release was published in 2019 in a collaboration between Google, Microsoft, Red Hat, and Styra (Figure 2). To define rules, you need to configure a constraint template, with which you define both the Rego code used for checking resources and the elements

Listing 3: Adding Labels to Namespaces

```
01 apiVersion: templates.gatekeeper.sh/v1beta1
02 kind: ConstraintTemplate
03 metadata:
04   name: k8srequiredlabels
05 spec:
06   crd:
07     spec:
08       names:
09         kind: K8sRequiredLabels
10       validation:
11         # Schema for the parameter field
12         openAPIV3Schema:
13           properties:
14             labels:
15               type: array
16               items: string
17   targets:
18     - target: admission.k8s.gatekeeper.sh
19       rego: |
20         package k8srequiredlabels
21         deny[{"msg": msg, "details": {"missing_labels": missing}}] {
22           provided := {label | input.review.object.metadata.labels[label]}
23           required := {label | label := input.parameters.labels[_]}
24           missing := required - provided
25           count(missing) > 0
26           msg := sprintf("you must provide labels: %v", [missing])
27         }
```

to which the constraint is applied.

Listing 3 shows how to set up a template if you want to force namespace labeling. The API type here is `templates.gatekeeper.sh/v1beta1`. The name field in the metadata section displays the name of the constraint. Within the spec section, you need to create a custom resource definition (CRD) in Kubernetes named `K8sRequiredLabels` and then configure the schema in the `openAPIV3Schema` block. The object has an attribute named `labels` and comprises an array of strings. In the targets section, you use a constraint to specify who is responsible

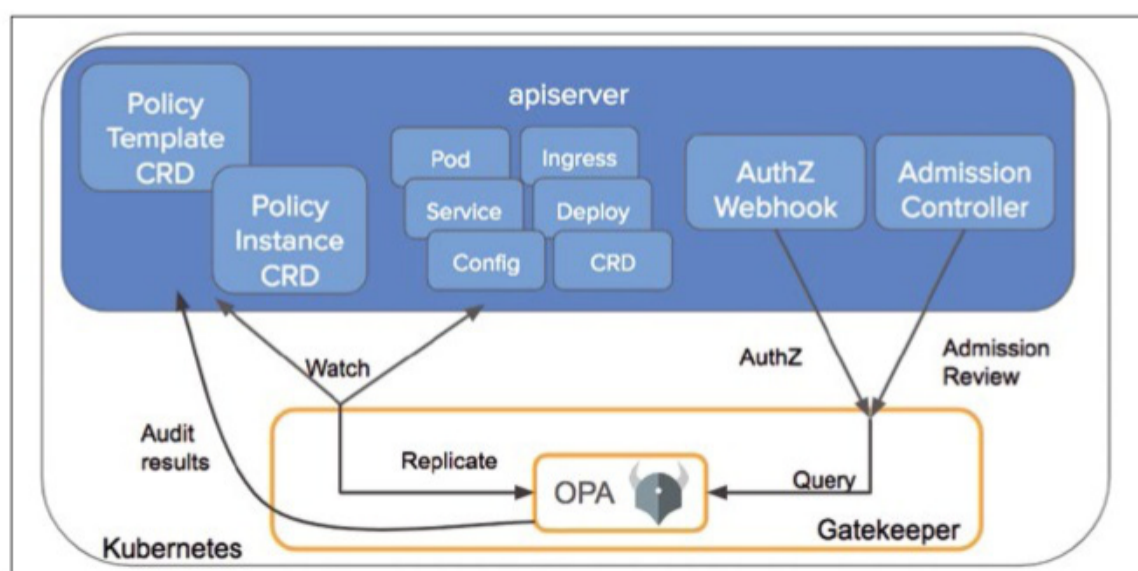


Figure 2: Gatekeeper uses webhooks to integrate with the Kubernetes architecture.
Source: Kubernetes documentation [5]

Listing 4: Label Constraints

```
apiVersion: constraints.gatekeeper.sh/v1beta1
kind: K8sRequiredLabels
metadata:
  name: ns-must-have-gk
spec:
  match:
    kinds:
      - apiGroups: [""]
        kinds: ["Namespace"]
  parameters:
    labels: ["gatekeeper"]
```

for the evaluation, which is the installed admission controller in this case. In the rego attribute, you then store the Rego code, which checks whether a namespace includes the desired labels. To upload the constraint to the Kubernetes cluster, use the `kubectl` command:

```
kubectl apply 2
--f https://raw.githubusercontent.com/2
open-policy-agent/gatekeeper/master/2
demo/basic/templates/2
k8srequiredlabels_template.yaml
```

Now you can use the constraint to define which labels must be present in the namespaces by creating a YAML file with the content shown in [Listing 4](#). CRD is then installed with:

```
kubectl apply 2
--f https://raw.githubusercontent.com/2
open-policy-agent/gatekeeper/master/2
demo/basic/constraints/2
all_ns_must_have_gatekeeper.yaml
```

At this point you will want to test the OPA rules and create a namespace with missing labels. The error message shown in [Listing 5](#) should appear.

Listing 5: Error Message

```
kubectl apply -f badns.yaml
Error from server ([denied by ns-must-have-gk] you must provide labels: {"gatekeeper"}): error when
creating "badns.yaml": admission webhook "validation.gatekeeper.sh" denied the request: [denied by
ns-must-have-gk] you must provide labels: {"gatekeeper"}
```

Listing 6: Checking S3 Buckets for ACLs

```
01 package terraform
02 deny[msg] {
03   changeset := input.resource_changes[_]
04   is_create_or_update(changeset.change.actions)
05   changeset.type == "aws_s3_bucket"
06   changeset.change.after.acl != "private"
07   msg := sprintf("S3 bucket %v has a non-private acl", [
08     changeset.name
09   ])
10 }
11 is_create_or_update(actions) { actions[_] == "create" }
12 is_create_or_update(actions) { actions[_] == "update" }
```

IaC and OPA

OPA is very flexible and thus is suited to a wide variety of security use cases, for example, reviewing Terraform code is a popular application. You can run OPA manually, or you can include it in an IaC pipeline to run automatically during deployment. For this to work, you need to save the Terraform execution plan and convert it to JSON format:

```
terraform init
terraform plan --out tfplan.binary
terraform show -json tfplan.binary > 2
tfplan.json
```

Next, you need to get to work on implementing the policy. The example in [Listing 6](#) shows how to ensure that all S3 buckets in AWS have a private access control list (ACL). To run this test, enter:

```
opa eval --fail-defined --format raw 2
--input tfplan.json 2
--data policy/ 'data.main.deny[x]'
```

OPA Commercial Version

In addition to the open source variant of OPA, Styra offers a commercial product named Styra Declarative

Authorization Service (Styra DAS), which is a control plane for OPA that lets you manage all your policies centrally. The product offers genuine added value because it comes with a large number of predefined rulesets. For example, in the more than 100 policies for admission controllers are the well-known payment card industry (PCI), MITRE, and Center for Internet Security (CIS) policy packs, with additional support for single sign-on and integration with Git. Two DAS versions include DAS Free – limited to 100 rules, a maximum of four systems, or 10 Kubernetes nodes – and DAS Enterprise, which supports unlimited rules, systems, and nodes and two more policy packs, including Terraform. In addition to the aforementioned security benchmarks, the product enables centralized policy monitoring and evaluates compliance with policies, and you have the option to manage all the logic for authorizations from a central location and apply this to microservices.

Conclusions

OPA has long established itself as a major player in the Kubernetes environment. The flexibility of its Rego programming language expands the field of application to encompass the cloud-native environment. It has what it takes to become a standard product in security matters upon which major corporations – such as Google, Red Hat, Microsoft, and VMware – rely in their own products.

Info

- [1] Open Policy Agent: [\[https://www.openpolicyagent.org\]](https://www.openpolicyagent.org)
- [2] Rego Playground: [\[https://play.openpolicyagent.org\]](https://play.openpolicyagent.org)
- [3] Styra Academy: [\[https://academy.styra.com\]](https://academy.styra.com)
- [4] OPA installation: [\[https://www.openpolicyagent.org/docs/latest/kubernetes-tutorial/\]](https://www.openpolicyagent.org/docs/latest/kubernetes-tutorial/)
- [5] OPA Gatekeeper: [\[https://kubernetes.io/blog/2019/08/06/opa-gatekeeper-policy-and-governance-for-kubernetes/\]](https://kubernetes.io/blog/2019/08/06/opa-gatekeeper-policy-and-governance-for-kubernetes/)

REAL SOLUTIONS *for* REAL NETWORKS

ADMIN is your source
for technical solutions
to real-world problems.

Improve your admin
skills with practical
articles on:

- Security
- Cloud computing
- DevOps
- HPC
- Storage and more!

**GET IT
FAST**

with a digital
subscription!

6 issues per year!

..... **ORDER NOW**

shop.linuxnewmedia.com

SQL Server 2022 and Azure

Cloud Power



SQL Server 2022 focuses on even closer collaboration between on-premises SQL servers and SQL functions in Azure, including availability and data analysis. We highlight the innovations of the database server and the interaction with versatile and powerful Azure services. By Thomas Joos

Microsoft is heading in the direction of the cloud. SQL Server 2022 is no exception, with the new database server offering enhanced functions for a closer tie-in to Azure, creating new opportunities for companies. However, I'll first look at SQL server licensing because, starting in version 2022 and following the abolition of the Open License program, SQL server will exclusively be available as a Cloud Solution Provider (CSP) purchase license or in the Open Value volume license program. You need to take this into account when planning licensing. Incidentally, as of this year, Windows Server is no longer available for commercial customers in the Open program for volume licensing, which also plays a role for the database server.

Azure Hybrid Advantage for SQL Server [1] lets you use licenses for locally operated SQL servers in Azure. Microsoft offers price reductions for this purpose, which is an important consideration in high-availability scenarios with SQL Server 2022 and Azure SQL Managed Instance. Therefore, it makes sense to look into server licensing in good time. A limited version of SQL Server 2022 will also be released for Linux. Microsoft

continues to recommend the use of Linux containers for container installation. SQL Server 2022 cannot be operated in Windows containers.

SQL Server Used Locally

The connection between local SQL servers and Azure SQL instances will still be optional in the future, which means SQL servers can be operated locally without relying on functions from Azure. However, the cloud offers many advantages in terms of availability, replication, and data analysis. As the

volume of data in the enterprise continues to increase, role owners need to deal with large quantities of structured and unstructured data. Additionally, you have the choice of centralized and decentralized storage approaches, numerous analysis options, and both relational and non-relational data. SQL Server 2022 is an obvious choice because analytics data can be comprehensively outsourced to the cloud. Basically, migrating from SQL Server 2017 or 2019 only makes sense if you intend to use the new server features, especially Azure SQL Managed

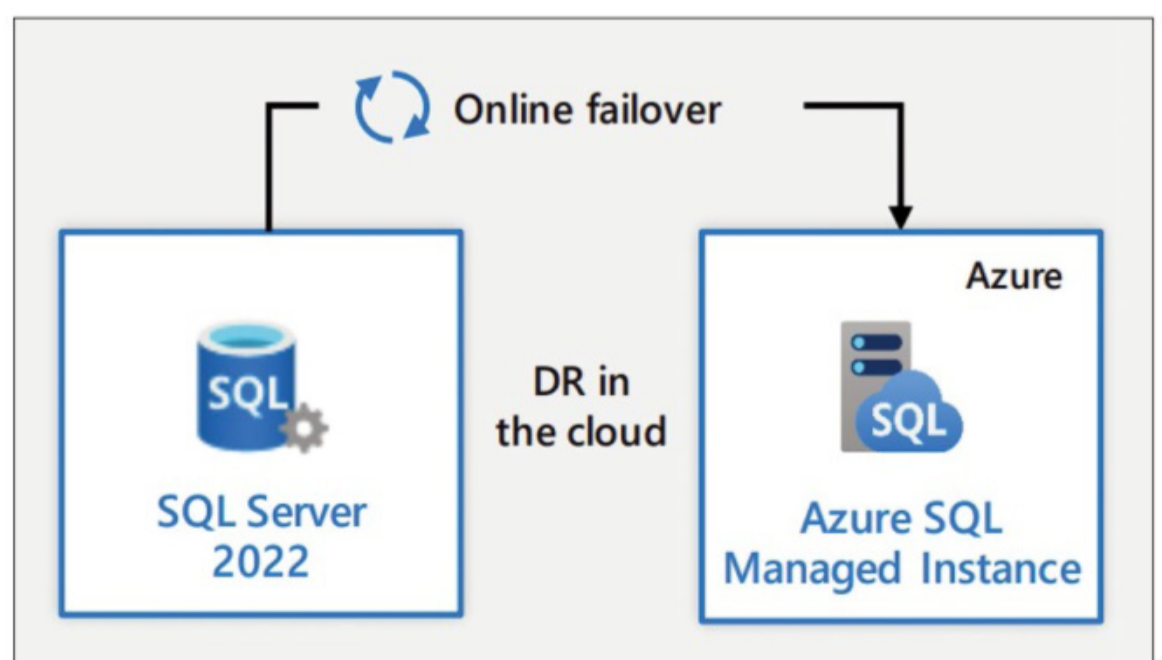


Figure 1: Local SQL servers can implement disaster recovery scenarios with Azure SQL Managed Instance.

Lead image © Dmitry Shpilko, 123RF.com

Instance. Windows Server 2022 is the operating system on which SQL Server 2022 resides. However, the expectation is that the database server will also run on Windows Server 2016 and 2019. You do not need a separate database server if SQL Server 2022 is used as an Azure SQL Managed Instance. The focus of connecting on-premises SQL Server 2022 installations with Microsoft Azure is therefore likely to be availability and data analytics. For example, Azure SQL Managed Instance helps admins make databases available more quickly and in a more reliable way in case of disaster recovery (**Figure 1**). Managed Instance can therefore be viewed as an extension of SQL Server 2022.

Managed SQL databases in Azure are also a good way to migrate on-premises databases to the cloud; after all, Azure SQL Managed Instance offers the features of the latest SQL Server version. The release of SQL Server 2022 also sees Microsoft aligning features between cloud and on-premises installations.

Establishing Reliability

In the event of a failover, Azure SQL Managed Instance can adopt all the databases of an on-premises SQL server, which means that corporations can establish disaster recovery strategies in the cloud without relying on their own hardware. The functionality of the cloud database is the same as that of a local installation, and on-board tools in SQL Server 2022 are used for the setup.

Previously, you could deploy cloud databases with Azure SQL, but you had to rely on a virtual database server. All you could save by migrating was the overhead of running a physical server on your own network: not so with the deployment of Managed SQL Instances in the context of SQL Server 2022. Managed Instances combine the benefits of Azure SQL databases with those of the on-premises data center. The service is ready for use out of the box and is updated and maintained by Microsoft. Moreover, you have, as mentioned

previously, the benefit of high availability between local SQL servers and databases in the cloud. However, SQL Server 2022 licensing needs to be viewed separately from the pay-per-use model in Azure.

The data in the managed databases is backed up automatically, and subscribers can set the retention period themselves. In Azure, you can monitor databases in the same way as locally operated databases on SQL Server 2022. All applications that require access to a SQL database can easily use the cloud variant.

In this regard, the Managed Instances of an Azure SQL database, as a platform-as-a-service (PaaS) solution, differ from the other infrastructure-as-a-service (IaaS) offerings in Azure. Where an Azure SQL database runs as an IaaS service, the database runs on a virtual SQL server in Azure. This virtual server needs to be updated, backed up, and managed. If you use a managed PaaS version, you only need to manage the database yourself. The underlying server remains completely in Microsoft's hands. This organization significantly simplifies high availability with SQL Server 2022. If required, Microsoft offers additional options in the form of the Data Migration Service (DMS) [\[2\]](#) for importing databases from locally operated SQL servers into Azure.

An Azure SQL Managed Instance resides on a virtual network and connects to other PaaS services and virtual machines. Customers can also create private connections to the database with a site-to-site virtual private network (VPN) or Azure ExpressRoute, which is particularly important for a high-performance connection of SQL Server 2022. The data flowing through the network has Transport Layer Security (TLS) encryption. Dynamic data masking is also included and can automatically hide sensitive areas when displaying information in web applications. You can also import data from a backup into managed SQL instances and migrate that way. For example, a local SQL server can be set to back up to an Azure storage account (blob

store). You can then create a managed SQL instance from this backup.

SQL Server 2022 relies on a new cloud feature for interaction with Azure SQL Managed Instance; consequently, on-premises SQL Server installations can be transferred to Azure for disaster recovery. This operation is handled by a Distributed Availability Group (DAG) that extends from the on-premises data center running SQL Server 2022 to an Azure SQL Managed Instance. The instance is on standby for this purpose and takes up the tasks of a local SQL server in the event of failure or maintenance. The setup is easily handled. Moreover, you could also set up extensive read scale-out scenarios in the same way. Queries that would overload the local SQL servers can be redirected to the Azure SQL Managed Instance, significantly reducing the load on the local servers.

Azure Services Data Analysis

In addition to a closer tie-in to Azure SQL Managed Instance, SQL Server 2022 also supports connectivity to Azure Synapse. Azure Synapse Analytics allows organizations to aggregate and prepare data queries from multiple sources in the cloud. This analysis also works in real time. For example, it can be used in Internet of things (IoT) environments, which often require rapid analysis of large volumes of data in a short time. Previously, the data exchange between SQL servers and Azure Synapse was handled by an extract, transform, and load (ETL) pipeline. A configuration like this is tricky to set up and manage. In SQL Server 2022, Microsoft supports change feeds between the on-premises SQL Server and Azure Synapse, which enables near real-time analytics and hybrid processing with a minimal effect on local systems. The service combines decentralized data structures in a clear interface.

Toward that end, Azure Synapse Analytics brings together data integration, enterprise data warehousing, and Big Data analytics, and it lets organizations visualize their data with Power

BI (Figure 2). Thanks to support for Azure Synapse Analytics, organizations can also benefit from Azure Machine Learning (ML) technologies when deploying SQL Server 2022. Additionally, data from Azure Data Lake Storage and from Spark pools can be used together with SQL Server 2022. The new service sees Microsoft looking to expand its modern data warehouse strategy and to empower enterprises to analyze their data more effectively and quickly – including in the big data field. SQL Server 2022 at the local data center is the ideal component for making local data available in the cloud. The various functions are illustrated in a YouTube video [3]. One advantage of Azure Synapse Analytics is its scalability. Almost unlimited amounts of data can be loaded from external systems and analyzed, even in real time and including external data warehouses and data from big data systems.

ML and Better Data Protection

ML models can be used in the scope of analysis by Azure Synapse Analytics and SQL Server 2022. Real-time analysis of streaming data is also supported, for example, if the data has been integrated directly into a data warehouse. To this end, Azure Synapse Analytics integrates the

Spark engine and can therefore collaborate with SQL Server 2022. Microsoft has also integrated data protection functions into the system and made it possible to analyze individual columns and rows with different security settings and permissions. Dynamic data masking is possible, as is permanent encryption of all data. The system uses other Azure technologies to detect threats and can automatically protect the data being analyzed. Azure Synapse Analytics relies on Azure Active Directory for authentication. In addition to data protection, data sharing naturally plays an important role. Azure Data Share [4] is one of the options available. The service works directly with Azure Synapse Analytics, enabling data sharing through the Azure user interface (e.g., in the scope of a subscription). In general, all software-as-a-service (SaaS) components that support the Open Data Initiative [5] can be connected.

Queries via SQL

The data in Azure Synapse Analytics can be queried by SQL. In this way, both relational and non-relational data (e.g., originating from local servers running SQL Server 2022) can be analyzed. According to Microsoft, petabytes of data can be processed in just seconds. In this

context, Synapse Analytics supports Power BI and Azure Machine Learning. Power BI capabilities are integrated directly into Azure Synapse Analytics for this purpose, including the numerous data sources that can be connected by Power BI. The Power BI Common Data Service (CDS) and artificial intelligence (AI) capabilities are also available in Azure Synapse Analytics. Azure Synapse Analytics supports other languages besides Transact SQL (T-SQL) for analysis or for connecting external systems, including Python, Scala, Spark, and, of course, .NET. The functions from Azure Data Factory are also available in Azure Synapse Analytics. Data sources can be connected to Mapping Data Flows, which is a graphical ETL tool available directly in the Synapse workspace.

Data Analytics and Blockchain Technologies

Azure Purview is another service related to SQL Server 2022. It supports the establishment of consistent governance functions for on-premises systems in the same way as in the cloud. The service captures the metadata on the on-premises SQL servers and classifies the data. Azure Purview helps organizations catalog and manage their data wherever the data resides. Access rights can be controlled specifically for local SQL servers. SQL Server Ledger, on the other hand, is a data integrity technology. It can be used, for example, to store off-chain data from blockchains, such as for smart contracts or decentralized apps (dApps). However, SQL Ledger also ensures that certain data cannot be changed, ruling out manipulation. In this regard, SQL Server 2022 can demonstrate that data integrity has not been compromised whenever needed by a client.

Database Licensing and Migration

The databases in Azure can be run in parallel with on-premises databases

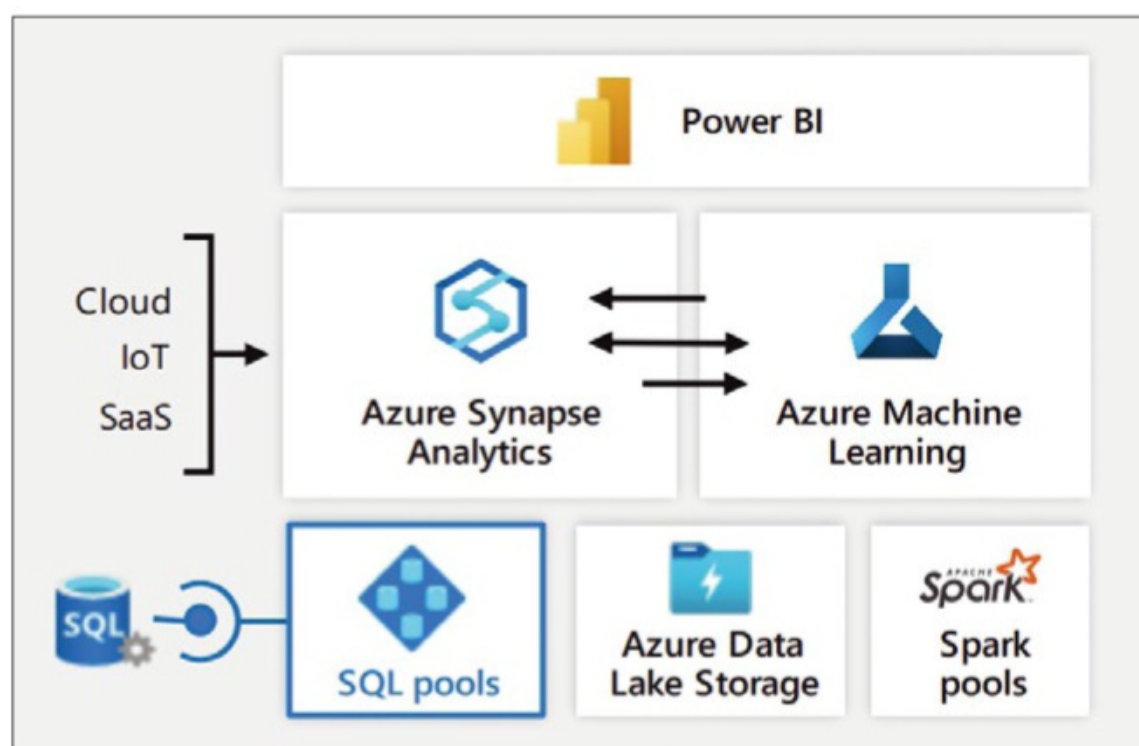


Figure 2: Connecting SQL Server 2022 with Azure Synapse Analytics opens up new possibilities for data analysis.

and in parallel with other services in Azure. Migration and synchronization are possible in all directions. Additionally, Microsoft tools support migration. In parallel, Microsoft offers the Azure Hybrid Advantage for Microsoft SQL Server, which means that local Microsoft SQL Server licenses can also be used in the cloud. The Data Migration Assistant (DMA) [6] is a Microsoft tool that can help admins migrate SQL databases from end-of-life servers. DMA can also be used to migrate on-premises databases to Microsoft Azure. However, Microsoft recommends the use of the Azure Database Migration Service [7] instead.

Conclusions

SQL Server 2022 is more closely tied in with Azure than any previous version, especially in the areas

of availability and data analytics. Comprehensive support for Azure SQL Managed Instances and Azure Synapse means that even larger volumes of data can be analyzed quickly and effectively. Migrating from previous versions to SQL Server 2022 is very much recommended for enterprises looking to rely more heavily on Azure with a view to improving server availability or analyzing data with various Azure services. Of course, the new version also offers better performance, as is the case with any new SQL Server version. ■

Info

- [1] Azure Hybrid Advantage for SQL Server: [\[https://azure.microsoft.com/en-us/pricing/hybrid-benefit/\]](https://azure.microsoft.com/en-us/pricing/hybrid-benefit/)
- [2] Database Migration Service: [\[https://docs.microsoft.com/en-us/azure/dms/tutorial-sql-server-to-managed-instance#create-an-azure-database-migration-service-instance\]](https://docs.microsoft.com/en-us/azure/dms/tutorial-sql-server-to-managed-instance#create-an-azure-database-migration-service-instance)

[to-managed-instance#create-an-azure-database-migration-service-instance\]](https://docs.microsoft.com/en-us/azure/dms/tutorial-sql-server-to-managed-instance#create-an-azure-database-migration-service-instance)

- [3] Azure Synapse Analytics YouTube video: [\[https://www.youtube.com/watch?v=tMY0i5E14eU\]](https://www.youtube.com/watch?v=tMY0i5E14eU)
- [4] Azure Data Share: [\[https://azure.microsoft.com/en-us/services/data-share/\]](https://azure.microsoft.com/en-us/services/data-share/)
- [5] Open Data Initiative: [\[https://docs.microsoft.com/en-us/microsoft-adobe-product-integrations/\]](https://docs.microsoft.com/en-us/microsoft-adobe-product-integrations/)
- [6] Data Migration Assistant: [\[https://www.microsoft.com/en-us/download/details.aspx?id=53595\]](https://www.microsoft.com/en-us/download/details.aspx?id=53595)
- [7] Azure Database Migration Service: [\[https://azure.microsoft.com/en-us/services/database-migration/\]](https://azure.microsoft.com/en-us/services/database-migration/)

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).

Shop the Shop → shop.linuxnewmedia.com

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

Want to subscribe?

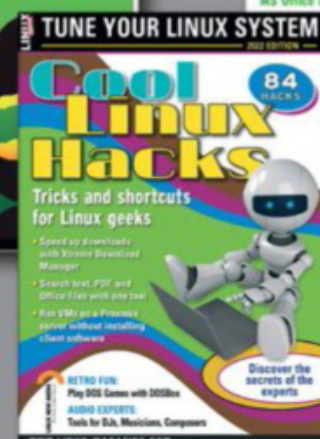
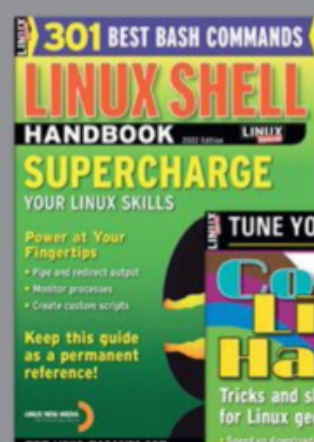
Searching for that back issue you really wish you'd picked up at the newsstand?

➤ shop.linuxnewmedia.com

DIGITAL & PRINT
SUBSCRIPTIONS



SPECIAL EDITIONS





Migrating CentOS to Rocky Linux with migrate2rocky

Safer Ground

CentOS users need to find a replacement soon. If you use CentOS 8 and you're looking for safer ground, the migrate2rocky script will automatically migrate your system to Rocky Linux - an enterprise RHEL derivative created by CentOS co-founder Greg Kurtzer. By Artur Skura

Red Hat's announced re-envisioning of the popular CentOS Linux distribution [1] caused a major disruption in the Linux space. According to some surveys [2], CentOS was actually the third most popular Linux distribution in the world (after Ubuntu and Debian), and all those CentOS instances weren't just running on desktops. CentOS has carved out a role as a distro of choice for web servers, file servers, and even high-performance computing systems. If you're wondering how the world will get along without CentOS, the first thing to know is that Red Hat didn't eliminate the CentOS *brand*, but they did move it upstream. CentOS used to be based on Red Hat Enterprise Linux source code, which meant that all the testing, troubleshooting, and verification that went into Red Hat's flagship distro also applied to CentOS. Now the CentOS developers will pull code from the RHEL development pipeline at an earlier point in the process, so it won't receive the value-added testing and verification that goes into RHEL. The newly named *CentOS*

Stream distribution will fall somewhere between Fedora and RHEL in the development cycle, which significantly reduces its value to enterprise customers. Those thousands of CentOS users are now searching for a solution. Luckily, the Linux community specializes in in-flight adaptations to adjust to changing times. CentOS is not the only distribution based on RHEL source code. Some existed previously, and a few others launched after Red Hat's announcement. One of the newcomers is Rocky Linux [3]. The Rocky project, which launched one day after Red Hat announced that it was moving CentOS upstream, was started by one of the founders of CentOS, who envisions it as an enterprise-ready (and cluster-ready) alternative to RHEL. By now, most CentOS users know the clock is ticking, and they will need to find an alternative. (See the box entitled "CentOS Versions.") The heat is on now for users who want to jump ship. Migrating a server is a delicate operation, and many things can go wrong. If you're thinking about making the jump to Rocky Linux, the good news is that the

developers provide a script that will automatically transition your system from CentOS to Rocky. The migrate2rocky script described in this article [4] supports upgrades from CentOS 8, and another script called migrate2rocky9.sh supports RHEL 9-based distributions. If you are still using CentOS 7, the recommended solution is to set up a new server with either Rocky 8 or 9, and manually install all services that were running on your old CentOS 7 server. (The ELevate project [5], which is sponsored by AlmaLinux, is working on an automated alternative for CentOS 7 migration that is still under development.) For CentOS 8 and 9 users, the migrate2rocky* scripts provide a fast and convenient way to escape the abandoned CentOS for higher ground.

Preparing for Migration

If you are running CentOS 8 or one of the other distributions based on RHEL 8, the migration process is fairly simple. The first step is to make sure you have enough space on your partitions: 250MB on */usr*, 1.5GB on */var*,

Photo by Ray Hennessy on Unsplash

and 50MB on */boot*. Failure to ensure you have enough disk space will break the migration process and might render your server unusable. Depending on the combination of packages, unexpected problems can arise, so it is important to make a full backup of the server before attempting the migration. All the following operations should run as root.

CentOS Versions

The only CentOS version still receiving maintenance updates is CentOS 7, which is a relatively ancient system. CentOS 7 first appeared in 2014 and stopped receiving full updates in mid-2020. Still, you can find it on many servers, and several hosting providers offer images allowing you to start new projects on CentOS 7 servers. But because CentOS 7 is basically an old system, running the 3.x kernel, it is not supported by Rocky Linux – Rocky has no equivalent to CentOS 7. CentOS 8 was the last version of the CentOS distribution, released in September 2019. This version corresponds to RHEL 8 (with 4.x kernels), of which multiple minor versions have been released since. Rocky Linux 8.4 appeared over a month after RHEL 8.4, whereas 8.5 and 8.6 were released less than a week after Red Hat’s releases. RockyLinux 8 will be supported until June 2029. In May 2021, Red Hat released RHEL 9 based on CentOS Stream and including Linux kernel 5.14. Two months later Rocky Linux 9 appeared. The distribution will be supported until the end of May 2032. If you are starting a new project and are considering using Rocky Linux, you should probably install the most recent version, Rocky 9. However, if you are migrating from CentOS 8, setting up Rocky 8 with the *migrate2rocky* script is a convenient option.

Connect to your server and fire up a session manager such as *tmux* or *screen*. You can install *screen* with:

```
dnf -y install screen
```

And then run it by typing:

```
screen -S migration
```

A tool like *tmux* and *screen* will allow the migration process to continue if anything happens to your connection (unlike plain SSH, which could terminate the migration process if your connection is cut short). To reconnect with a running session, enter:

```
screen -r migration
```

The next step is to make sure the system is up to date by running:

```
dnf update
```

If you see a message saying there is nothing to update, you are good to go. Otherwise, wait for the process to finish, then reboot the machine. The reboot will ensure that all system files have actually been updated.

The *migrate2rocky* script relies on the original CentOS repositories being present. If you haven’t modified the default repositories, you should be fine. Otherwise, you need to make sure at least */etc/yum.repos.d/CentOS-Linux-BaseOS.repo* is present and enabled.

Installing migrate2rocky

The best way to obtain *migrate2rocky* is using Git. If Git is not installed, set it up with:

```
dnf -y install git
```

Next, clone the repository that includes the migration script:

```
git clone https://github.com/rocky-linux/rocky-tools.git
```

If you are not particularly concerned with security, you can download the script directly using a tool such as *Curl* or *Wget*:

```
wget
https://raw.githubusercontent.com/rocky-linux/rocky-tools/main/migrate2rocky/migrate2rocky.sh
```

If you used the Git method, you need to change the directory:

```
cd rocky-tools/migrate2rocky/
```

Finally, make it executable with:

```
chmod u+x migrate2rocky.sh
```

Running the Migration Script

The script has only three options: *-h* to display a short help message, *-r* to start the conversion, and *-V* to verify the system. If you completed all the

```
migrate2rocky - Begin logging at Tue 23 Aug 2022 10:53:21 PM CEST.

Usage: migrate2rocky.sh [OPTIONS]

Options:
-h Display this help
-r Convert to rocky
-V Verify switch
  !! USE WITH CAUTION !!
[root@ws231 migrate2rocky]# ./migrate2rocky.sh -r
migrate2rocky - Begin logging at Tue 23 Aug 2022 10:53:31 PM CEST.

Modular dependency problems:

Problem 1: conflicting requests
- nothing provides module(perl:5.26) needed by module perl-IO-Socket-SSL:2.066:8030020201222215140:1e4bbb35.x86_64
Problem 2: conflicting requests
- nothing provides module(perl:5.26) needed by module perl-libwww-perl:6.34:8030020201223164340:b967a9a2.x86_64

Removing dnf cache
Preparing to migrate CentOS Linux 8 to Rocky Linux 8.
```

Figure 1: The *migrate2rocky* script finds only two issues in the existing CentOS 8 configuration and proceeds with the migration.

previous steps, you are now ready to start the main migration process (**Figure 1**):

```
migrate2rocky.sh -r
```

If you migrate a freshly installed server, the script should run without any trouble. On the other hand, very few people need to migrate a freshly installed CentOS 8 server because, if you are setting up a server now, it would make more sense to install Rocky Linux directly instead of installing CentOS. It is safe to assume the typical user of the migration script is working with a CentOS system that has been in use for a while and has some services that were originally installed a few years ago. It is impossible to test every possible combination of packages and

custom software, but in our tests, migrate2rocky performed surprisingly well and managed to update a system with a number of customizations without any problems.

In our lab, we used a system with the Apache, PHP, MariaDB, and NGINX acting as a reverse proxy, as well as an ancient version of MongoDB from MongoDB repository left over from CentOS 7. The script replaced all relevant packages correctly (**Figure 2**) and everything worked fine after the reboot, including MongoDB.

Caveats

If you use OpenJDK, you might find that Java no longer works: This issue is caused by missing symlinks in */etc/alternatives*. The problem is

caused by the OpenJDK package, and as a temporary workaround, you can use the following command:

```
rpm -qa --scripts
java-{{1.8.0,11}}-openjdk-{{headless,devel}} | x
sed -n ,/postinstall/, /exit/{ x
/postinstall/! { /exit/ ! p} }' | sh
```

If your IPA server fails to run after the migration, use the following command (after fixing the Java issue mentioned above):

```
ipa-server-upgrade --skip-version-check
```

It is also worth checking the log file */var/log/migrate2rocky.log* (**Figure 3**), which might contain errors the script encountered during the migration. There should be very few errors, but it is worth looking into each error to understand if anything needs fixing.

Conclusion

If you proceed according to instructions and take the necessary precautions, the migration from CentOS 8 to Rocky Linux 8 is a simple process – definitely easier than a manual upgrade between major versions. Given the fact that Rocky 8 will be supported until the end of May 2029, your server should be safe until then. However, if you are just running a few services and your configuration is fairly simple, you might want to consider setting up a new server based on the more modern Rocky Linux 9 and manually migrating the services one by one. ■

Info

- [1] CentOS Project Shifts Focus to CentOS Stream: <https://blog.centos.org/2020/12/future-is-centos-stream/>
- [2] 40 Linux Statistics You Need to Know: <https://kommandotech.com/statistics/linux-statistics/>
- [3] Rocky Linux: <https://rockylinux.org/>
- [4] migrate2rocky: <https://github.com/rocky-linux/rocky-tools/tree/main/migrate2rocky>
- [5] ELivate Project: <https://wiki.almaLinux.org/elevate/>

```
python3-oauthlib-2.1.0-1.el8.noarch
python3-ply-3.9-9.el8.noarch
python3-prettytable-0.7.2-14.el8.noarch
python3-pycparser-2.14-14.el8.noarch
python3-pyserial-3.1.1-8.el8.noarch
python3-pysocks-1.6.8-3.el8.noarch
python3-pytz-2017.2-9.el8.noarch
python3-pyudev-0.21.0-7.el8.noarch
python3-pyyaml-3.12-12.el8.x86_64
python3-requests-2.20.0-2.1.el8_1.noarch
python3-setuptools-39.2.0-6.el8.noarch
python3-setuptools-wheel-39.2.0-6.el8.noarch
python3-six-1.11.0-8.el8.noarch
python3-slip-0.6.4-11.el8.noarch
python3-slip-dbus-0.6.4-11.el8.noarch
python3-unbound-1.7.3-17.el8.x86_64
python3-urllib3-1.24.2-5.el8.noarch
glibc-libs-3.4.4-5.el8.x86_64
quota-1:4.04-14.el8.x86_64
quota-nls-1:4.04-14.el8.noarch
readline-7.0-10.el8.x86_64
rootfiles-8.1-22.el8.noarch
rpcbind-1.2.5-8.el8.x86_64
setup-2.12.2-6.el8.noarch
sg3_utils-1.44-5.el8.x86_64
sg3_utils-libs-1.44-5.el8.x86_64
shared-mime-info-1.9-3.el8.x86_64
slang-2.3.2-3.el8.x86_64
snappy-1.1.8-3.el8.x86_64
sqlite-3.26.0-15.el8.x86_64
sqlite-libs-3.26.0-15.el8.x86_64
squashfs-tools-4.3-20.el8.x86_64
tar-2:1.30-5.el8.x86_64
teamd-1.31-2.el8.x86_64
timedatex-0.5-3.el8.x86_64
tpm2-tss-2.3.2-4.el8.x86_64
trousers-0.3.15-1.el8.x86_64
trousers-lib-0.3.15-1.el8.x86_64
unbound-libs-1.7.3-17.el8.x86_64
userspace-rcu-0.10.1-4.el8.x86_64
ustr-1.0.4-26.el8.x86_64
wget-1.19.5-10.el8.x86_64
xkeyboard-config-2.28-1.el8.noarch

Complete!

Done, please reboot your system.
A log of this installation can be found at /var/log/migrate2rocky.log
[root@ws231 migrate2rocky]#
```

Figure 2: The migration has been completed and the system can now be restarted to automatically boot Rocky Linux 8.

```

migrate2rocky - Begin logging at Tue 23 Aug 2022 10:53:31 PM CEST.

Modular dependency problems:

Problem 1: conflicting requests
- nothing provides module(perl:5.26) needed by module perl-IO-Socket-SSL:2.066:8030020201222215140:1e4bbb35.x86_64
Problem 2: conflicting requests
- nothing provides module(perl:5.26) needed by module perl-libwww-perl:6.34:8030020201223164340:b967a9a2.x86_64

Removing dnf cache
Preparing to migrate CentOS Linux 8 to Rocky Linux 8.

Determining repository names for CentOS Linux 8.....

Found the following repositories which map from CentOS Linux 8 to Rocky Linux 8:
CentOS Linux 8  Rocky Linux 8
appstream      appstream
powertools     devel
ha             ha
baseos         baseos
extras        extras
powertools     powertools

Getting system package names for CentOS Linux 8.....

Found the following system packages which map from CentOS Linux 8 to Rocky Linux 8:
CentOS Linux 8  Rocky Linux 8
centos-logos-ipa  rocky-logos-ipa
centos-backgrounds  rocky-backgrounds
centos-gpg-keys   rocky-gpg-keys
centos-logos      rocky-logos
centos-indexhtml  rocky-indexhtml
centos-linux-release  rocky-release
centos-logos-httpd  rocky-logos-httpd
centos-linux-repos  rocky-repos

Getting list of installed system packages.

We will replace the following CentOS Linux 8 packages with their Rocky Linux 8 equivalents
Packages to be Removed  Packages to be Installed
centos-gpg-keys         rocky-gpg-keys
centos-linux-release    rocky-release
centos-logos-httpd      rocky-logos-httpd
centos-linux-repos      rocky-repos

In addition to the above the following system packages will be removed:
centos-linux-release
centos-linux-release

Getting a list of enabled modules for the system repositories.

/var/log/migrate2rocky.log

```

Figure 3: Reviewing the log file `/var/log/migrate2rocky.log`.



Rancher has set up shop as an agile alternative to Red Hat OpenShift as an efficient way to manage Kubernetes clusters. In terms of the architecture, a Rancher setup differs significantly from classic Kubernetes. By Martin Loschwitz

When the talk turns to Kubernetes, many instantly think of the major league products by Red Hat and Ubuntu. Red Hat OpenShift is a massive, complex Kubernetes distribution with a large number of components and a connected app store for container applications. Although Rancher now belongs to a major Linux distributor, SUSE has largely kept out of the Rancher developers' way, so far, which is why Rancher has kept much of the simplicity its fans have loved for years. Therefore, anyone who wants to get started with Kubernetes today can find a way to do so with Rancher, and the entry bar is set low. Even so, a Rancher setup is not a no-brainer. Various factors have to be considered with regard to the hardware. More specifically, the number of machines and their dimensioning deserves attention. Once the hardware is ready for use in the rack, the next step is to install a Kubernetes distribution, because Rancher sets up its own infrastructure completely on Kubernetes. What sounds

complicated is not a problem in practice, because Rancher comes with its own Kubernetes core distribution in tow in the form of K3s, which provides all the features you need. In this article, I guide you through the implementation of Rancher in your setup. The project starts on a proverbial greenfield site, the objective being a Rancher cluster that can be used in production. To achieve this setup, you need to be introduced to a few terms from the Rancher world that confront administrators regularly.

Architecture

If you have already worked with one of the other major Kubernetes distributions, you might be a bit overwhelmed by Rancher's core features at first, because Rancher works differently from most competitor products. A comparison quickly makes this clear: OpenShift comprises a management plane, also known as the control plane, which includes all of the central services in the environment.

OpenShift uses precisely one Kubernetes instance for its own management. If users start containers in Kubernetes, the containers run in the existing cluster and use its existing infrastructure.

Rancher takes a different approach. It not only sees itself as a tool for managing workloads in Kubernetes, but also as a tool for controlling and managing Kubernetes environments. Therefore, applications do not run in the Kubernetes cluster that Rancher requires for its Kubernetes components. Instead, Rancher assumes that each end-user setup is a separate Kubernetes installation, which Rancher then also operates.

Because Rancher installs agents in these satellite setups, it can also control the workloads in these secondary Kubernetes clusters. Consequently, the programs that a Kubernetes cluster is supposed to run in Rancher never run in the same instance as the Rancher services themselves, but in a separate Kubernetes environment that Rancher controls downstream.

Some administrators might already be cringing at this point, because – at first glance – the Rancher approach seems clumsy and, above all, not very efficient in terms of resource usage. In fact, the infrastructure components of Kubernetes that are part of each Kubernetes cluster generate some overhead themselves, but because the individual Kubernetes instances in Rancher rarely run thousands of pods – you would instead run individual Kubernetes instances if they are not part of the same setup – the additional computational overhead is manageable.

On the plus side, you get genuinely clean isolation of individual applications in the Kubernetes cluster, as well as comprehensive monitoring and alerting capabilities for each end-user Kubernetes. This approach also means that Rancher gives you an option that is missing in other environments: It can manage the commercial Kubernetes offerings found in AWS, Azure, or Google Cloud, if need be.

Hardware

Once a company has decided to use Rancher, the first thing on the agenda is its basic deployment, for which you need hardware. Rancher can be operated in a completely virtualized form, and nothing can stop you from doing so in principle, but if certain nodes are running third-party workloads belonging to completely different applications on top of the Rancher Kubernetes components, resource bottlenecks can be a concern.

Like any other application, Rancher feels most at home on its own hardware. Because the setup in this example is intended to depict Rancher in a production environment as realistically as possible, I am also assuming that you have bespoke Rancher hardware. To achieve high availability, you need two servers, each with 128GB of RAM and at least 32 virtual CPUs (vCPUs). What the manufacturer means by vCPUs in this context is not entirely clear from the documentation.

However, you can safely assume that a current Xeon with 24 physical cores (i.e., 48 threads) is powerful enough

to run most Rancher setups. Rancher itself states that a machine of this dimension can reasonably run 2,000 clusters and up to 20,000 compute nodes, even with the Rancher database grabbing two cores and 4GB of RAM for itself in the background. Input/output operations per second (IOPS) are more important for the database, so the systems should ideally be on fast flash memory for the servers.

In a normal setup, the worker nodes (i.e., the hosts on which the user Kubernetes clusters run) will have at least two systems. The documentation refers to these as nodes, compute nodes, or target nodes, but they are the same servers in all cases.

In terms of hardware, the systems just need to be powerful enough. Unlike their full and paravirtualized counterparts, containers do not require CPU time to run their own vCPUs and Linux, but the containers with their applications will still want a good helping of RAM. If you dimension the target nodes like KVM nodes, you can't normally go wrong. The situation is different with the nodes for the Rancher components, however: You need to pay attention to the local storage media of the systems for the compute nodes because that is where the containers are stored during operation. With a fast but small NVMe-based drive, the installation might freeze on you. Throw in a few gigabytes, and you will be on the safe side in most cases.

Tricky Question

For Rancher services to run on a system at all, Linux is an absolute prerequisite, which means you get to choose your distribution of choice. The temptation is often to use what you know from your own experience, but by doing so, you might be wasting a great opportunity, because Rancher does not offer its components in the form of classic software packages: Rancher is also packed in containers.

Therefore, any distribution suitable for running containers with Docker or Podman is also a candidate for Rancher. From the administrator's point of view, you have no reason to commit to a full-fledged Linux – a microdistribution such as CoreOS ([Figure 1](#)) will do the trick. Flatcar Linux is an alternative, as well, and I have already looked at that in detail in the past [\[1\]](#). Ubuntu Core is yet another option. Rancher even had its own mini Linux named RancherOS, but it is no longer officially supported.

If you don't like the idea of a microdistribution, you'll still do fine with the usual candidates, but make sure from the outset that the system you roll out is as lean as possible. Introducing Rancher gives you a great opportunity to take care of automation. Kickstart rules for AlmaLinux or Rocky Linux can be created quickly and will save you a huge amount of work, especially when scaling the platform. Make sure from the outset

```
Red Hat Enterprise Linux CoreOS 47.83.202102090044-0 (Ootpa) 4.7
SSH host key: SHA256:0U+t/1VT1P1e2g161a1R6cqHjQ1215pcTkUJ3u6/uS8 (ECDSA)
SSH host key: SHA256:JH4D1rgRKn0qbuFjLkazu90YA74e3S/Ck2KUL7QRCeY (ED25519)
SSH host key: SHA256:/WPO0DEG0yc96zLA6nHrPGr4rn2XSihL2T4dVBBufC4 (RSA)
localhost login: core (automatic login)

=====
Welcome to the CoreOS live environment. This system is running completely
from memory, making it a good candidate for hardware discovery and
installing persistently to disk. Here is an example of running an install
to disk via coreos-installer:

sudo coreos-installer install /dev/sda \
  --ignition-url https://example.com/example.ign

You may configure networking via 'sudo nmcli' or 'sudo nmtui' and have
that configuration persist into the installed system by passing the
'--copy-network' argument to 'coreos-installer install'. Please run
'coreos-installer install --help' for more information on the possible
install options.
=====

[core@localhost ~]$ _
```

Figure 1: Rancher does not require the target systems to have a full-fledged Linux distribution. CoreOS or Flatcar Linux are sufficient to run container workloads. © Red Hat

that you automate the system configuration on the individual servers to the extent possible.

Clearly, a sensibly planned Rancher setup requires some work up front before you even get started. The overhead pays off later, though, because the cluster can then be expanded quickly and easily.

Getting Started

In my test setup, I had four servers running Ubuntu 20.04 LTS. Podman is primarily a Red Hat invention, but nothing stops Ubuntu from continuing to rely on Docker as a container runtime environment. Accordingly, Docker's community repositories are enabled in the system configurations, and the packages required for Docker are installed. In this state, the systems are ready for the Rancher installation.

Terminology

Like virtually any solution from the cloud-native orbit, a number of Rancher-speak terms have evolved over time that have a distinct meaning

in the Rancher universe. Before working with Rancher, you need to get to know the most important terms. For example, a Rancher server is a system that hosts all of Rancher's components for managing and provisioning Kubernetes clusters. Importantly, you will have at least one of these Rancher servers, but Rancher also offers scalability across the board at its management level. The maximum number of Rancher servers per installation is therefore practically unlimited.

K3s is equally important in the Rancher context. This minimal Kubernetes version is maintained by the Rancher developers; it contains only the components needed to run Rancher. Remember, Rancher also rolls itself out as a Kubernetes cluster. K3s provides the underpinnings for this capability, and it is far leaner than other alternatives. K3s, by the way, is not the only Kubernetes distribution that works with Rancher. You also regularly come across the Rancher Kubernetes Engine (RKE), which is a Kubernetes distribution that also comes directly from the Rancher developers, but it is older

than K3s. The somewhat more modern K3s is therefore the recommended standard distribution for new setups. RKE2 is a further development that focuses on security and is designed for special areas of application (e.g., government).

Infrastructure

The first step on the way to a running Rancher is to install K3s. The example here assumes the setup has a MariaDB or MySQL database available for Rancher to store its metadata. Optionally, the database can run on the same servers as Rancher, but an external database on separate hardware is an option. Make sure you secure the database and make it highly available. Without its metadata, Rancher is more or less useless. MariaDB or MySQL therefore need to run reliably.

Because Rancher's server components in the example also need to be redundant, you will want a load balancer (**Figure 2**), which ideally will listen for incoming connections on the address on which Rancher will



Figure 2: Rancher needs a load balancer (e.g., an NGINX server) to make the setup highly available. A Layer 4 balancer is the best option for newcomers. © NGINX

be accessible to forward to the two Rancher hosts used in this example. Load balancers are possible in Layer 4 and Layer 7.

I am assuming that a Layer 4 load balancer will be used. This configuration makes the setup a little easier, especially later on with regard to Rancher's SSL capabilities. A Layer 7 device would offer more in terms of configuration options in theory, but you would need to configure it to handle SSL management. However, with a Layer 4 load balancer, Rancher does this itself by rolling out an instance of Traefik [2].

Either way, the load balancer must also be highly available. Its failure would mean that Rancher itself and the managed Kubernetes clusters would still be running, but would no longer be accessible from the outside. Speaking of accessibility: The DNS entry intended for Rancher needs to be stored in the zone file of the respective domain when you set up the load balancer; otherwise, the Rancher installation cannot begin. Last but not least, but which should be a matter of course today, the time needs to be correct on each system of the installation. On the distribution used here, this can be done by the legacy network time protocol (NTP) or chronyd. Either way, one of the two components needs to be set up and have access to an NTP server to set the system time correctly.

K3s

The K3s authors make it very easy to get their software onto your system by simply running the following command on each machine designated as a Rancher server:

```
$ curl -sL https://get.k3s.io | 2
sh -s - server 2
--datastore-endpoint="mysql://2
<User>:<Password>@tcp(<Host>:3306)/2
<Database>"
```

In doing so, replace <User>, <Password>, <Host>, and <Database> with your MySQL database credentials. Assuming that the username is

rancher, the password is *secret*, and the database name is *rancher*, the command would be:

```
$ curl -sL https://get.k3s.io | 2
sh -s - server 2
--datastore-endpoint="mysql://2
rancher:secret@tcp(10.42.0.1:3306)/2
rancher"
```

This command only works if the system has access to the Internet – although it's not absolutely necessary for Rancher because the software can be operated via a proxy server or without a network connection in air gap mode. However, describing installation variants is beyond the scope of this article, so check out the Rancher documentation [3] if you need more information.

On all future Rancher servers, after successfully invoking the installation command, the

```
sudo k3s kubectl get nodes
```

command should output a list of all Rancher servers in the setup. If the list shows all machines (two servers in this example), the K3s setup has worked.

Note that the K3s tool is specific to K3s. You might want to access the K3s cluster with the standard `kubectl` tool, too. To this end, K3s creates the `/etc/rancher/k3s/k3s.yaml` file during installation. Every user will want to copy this to `~/.kube/config`, but before this happens, you need to edit the file, because the host to be controlled in the file is set to `localhost` by default.

In the YAML file, replace the server entry value with the DNS name pointing to the load balancer. After doing so, the

```
kubectl get pods --all-namespaces
```

command should work. If so, K3s is ready for the Rancher installation.

cert-manager

Because I want Rancher in the example to obtain its SSL certificates automatically with Let's Encrypt, I also need to install `cert-manager`. Of the several commands to help you do this, the command

```
# kubectl apply -f https://github.com/2
jetstack/cert-manager/releases/download/2
v1.5.1/cert-manager.crds.yaml
```

installs the Custom Resource Definitions required for `cert-manager` in the local K3s instance, and the command

```
# helm repo add jetstack 2
https://charts.jetstack.io
```

adds the Helm directory for `cert-manager` to the K3s instance.

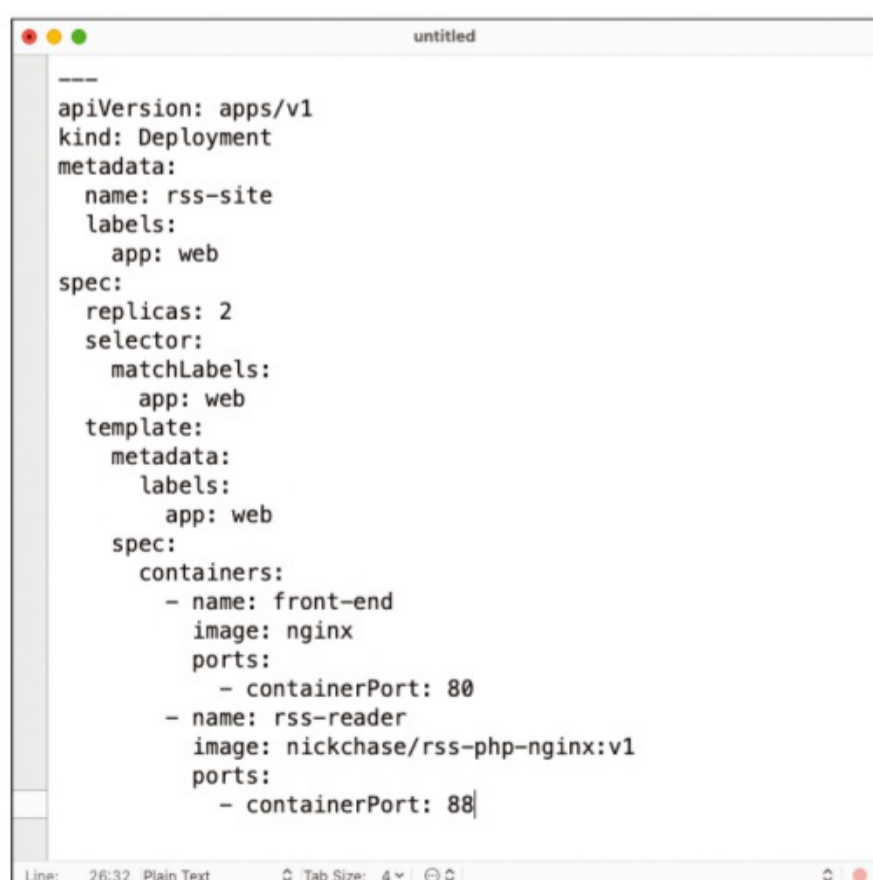


Figure 3: Helm is a package manager for Kubernetes and works much like the established package managers for rpm and dpkg by bundling image metadata and images so they are downloadable as a whole.

The commands

```
# helm repo update
# helm install cert-manager 2
  jetstack/cert-manager 2
  --namespace cert-manager 2
  --create-namespace --version v1.5.1
```

update the local metadata of all configured Helm directories and drag cert-manager into the local K3s installation. If everything works, the call

```
# kubectl get pods --namespace cert-manager
```

shows the running containers.

Installing Rancher

The Rancher developers also distribute their product as a Helm chart. Remember that the Helm package manager (Figure 3) for Kubernetes lets you deliver metadata and images in a standardized format. Like normal distribution packages, Helm charts can be obtained from different directories.

To begin, add the Helm directory of the Rancher project to your K3s installation, create a namespace in K3s

in which all Rancher services will run, and update the metadata of the available Helm charts again:

```
# helm repo add rancher-latest https://2
  releases.rancher.com/server-charts/stable
# kubectl create namespace cattle-system
# helm repo update
```

Now you can create a running Rancher cluster:

```
# helm install rancher rancher-latest/2
  rancher --namespace cattle-system 2
  --set hostname=<Host> --set replicas=3 2
  --set ingress.tls.source=letsEncrypt 2
  --set letsEncrypt.email=<Email> 2
  --set letsEncrypt.ingress.class=nginx
```

Be sure to replace <Host> und <Email> with the correct values. Next, you can track progress with the command:

```
# kubectl -n cattle-system rollout 2
  status deploy/rancher
```

It can take a while to complete. When done, the command

```
# kubectl -n cattle-system get 2
  deploy rancher
```

should show you a deployment with three available instances; the Rancher installation is complete.

If you now open the URL that contains the address of the load balancer, you are automatically taken to the installation's login page. During deployment, Rancher displays the user data at the command line, but the first time you log in to the web interface, Rancher will force you to change the password. If you were unable to make a note of the password during installation, the command

```
# kubectl get secret --namespace 2
  cattle-system bootstrap-secret 2
  -o go-template='{{ 2
    .data.bootstrapPassword|base64decode}}'2
  {{ "\n" }}'
```

will request it again.

Adding Nodes

Currently, a usable Rancher cluster exists, but you would not be able to roll out workloads on it yet: What's missing are the nodes, or the systems that run the Kubernetes clusters for your user applications. In terms of preparation, the same applies to the nodes as to the Rancher servers, although you don't need a database. NTP must be enabled, though. Once all the requirements are met, all it takes is the command:

```
# curl -sfL https://get.k3s.io | 2
  K3S_URL=https://<Rancher-Hostname>:2
  6443 K3S_TOKEN=<Token> sh --
```

Replace <Rancher-Hostname> with the hostname assigned to the load balancer. The content you need to specify for K3S_TOKEN can be found in the /var/lib/rancher/k3s/server/node-token file on the servers.

Using Rancher

After following these steps, the first workloads can be rolled out in Rancher. The Marketplace belonging to Rancher can also be accessed from the web user interface. Preconfigured applications optimized for Rancher

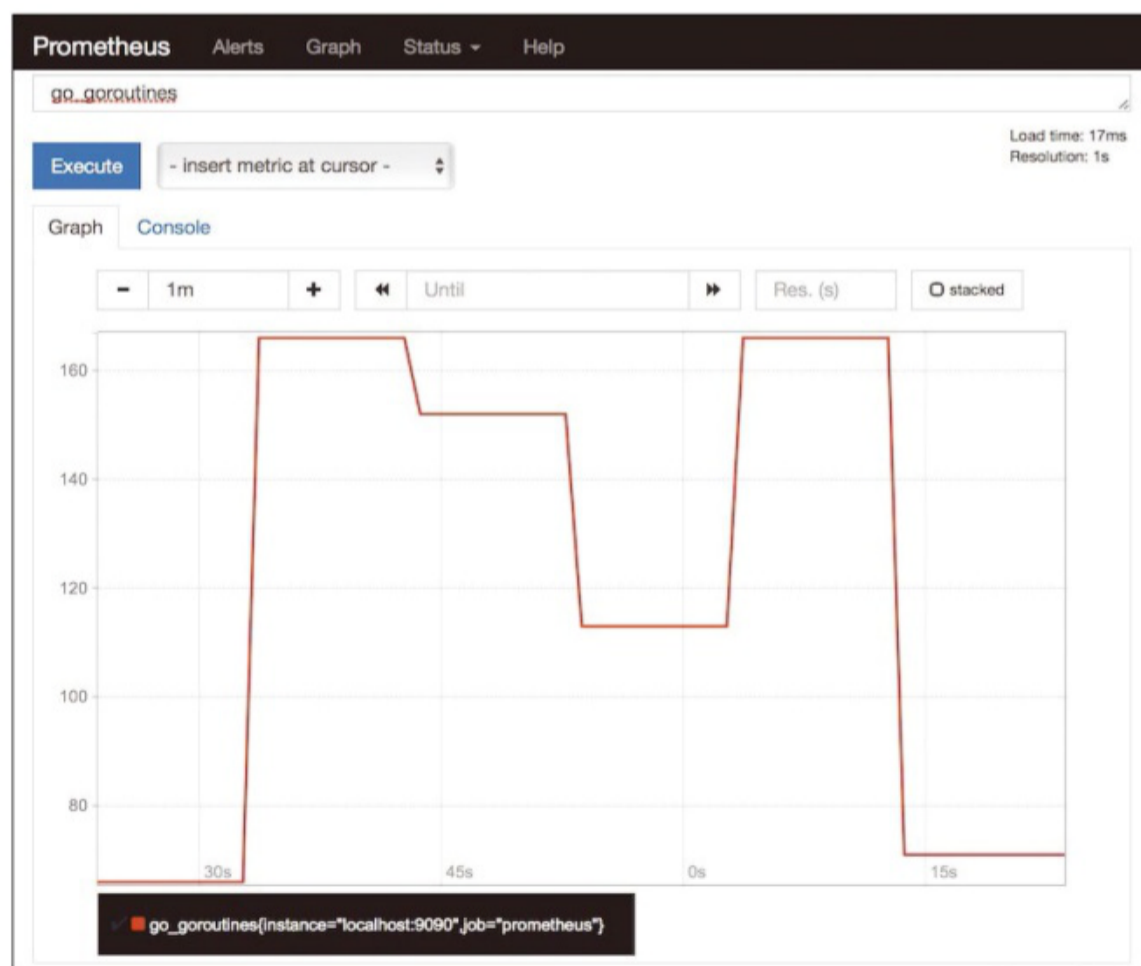


Figure 4: Prometheus is an exceptionally powerful tool for collecting metrics data. It implicitly works with Rancher under the hood to provide data from both Rancher and the end-user setups being run. © Alex Eillis

are available there, or they can be added to the installation from Helm charts.

This setup far from exhausts Rancher’s feature set. Once the first applications are rolled out as services in Rancher, you can set up monitoring from the Monitoring menu item. It comes with alerting based on various parameters. If the combination of components from the context of cloud-native environments sounds familiar, you are on the right track, because Rancher does not implement its monitoring itself. Instead, it relies on Prometheus in the background (Figure 4), its alert manager, and the Grafana GUI component (Figure 5).

Conclusions

Rancher poses a number of requirements in terms of the infrastructure it expects onsite. Once rolled out, though, it turns out to be a decidedly powerful tool for running Kubernetes workloads. The setup proves to be an intuitive and comparatively smooth process. If you are looking to switch to Kubernetes, make sure you include Rancher in your evaluation.

Info

- [1] “Container microdistributions k3OS and Flatcar” by Martin Loschwitz, ADMIN,

- issue 60, 2020, pg. 24, [https://www.admin-magazine.com/Archive/2020/60/Container-microdistributions-k3OS-and-Flatcar/]
- [2] “Managing network connections in container environments” by Martin Loschwitz, ADMIN, issue 63, 2021, pg. 58, [https://www.admin-magazine.com/Archive/2021/63/Managing-network-connections-in-container-environments/]
- [3] Rancher docs: [https://rancher.com/docs/]

The Author

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



Figure 5: The team of Prometheus, its alert manager, and Grafana is also available in Rancher. Given an appropriate configuration by Rancher, the combo helps monitor container workloads. © Grafana

A photograph of a wooden suspension bridge with a rope safety net, set in a forest. The bridge is made of wooden planks and is surrounded by a dense network of ropes forming a safety net. The background shows tall trees and a forest floor.

Network overlay with VXLAN

Safety Net

VXLAN addresses the need for overlay networks within virtualized data centers accommodating multiple tenants. By Benjamin Pfister

If high availability or load balancing is required on servers across geographically separated locations, many of these services require direct access over Layer 2. However, if the Layer 2 link, which is based on classic 802.1Q VLAN technology, is interrupted by a routed link in Layer 3, the required transparency is lost. A virtual extensible local area network (VXLAN) solves this problem by extending Layer 2 accessibility over the existing Layer 3 structure with an overlay network.

Layers

System administrators face the challenge of planning scalable networks

while maintaining appropriate security and availability requirements. Some server systems need to be located in a redundant infrastructure on a subnet. Additionally, virtualized systems need to be capable of migration between multiple sites on the same subnet, whether in live operation or in a disaster recovery scenario. Moreover, today's networks need be able to meet the increasing demand for bandwidth. In classic scenarios, the required security can usually be achieved by implementing virtual local area networks (VLANs) to reduce the size of broadcast domains in combination with firewall rules or static packet

filters (access control lists, ACLs) on routers or Layer 3 switches. In some cases, routing virtualization implemented by virtual routing and forwarding (VRF) is used at the routing level to enable multiclient capability.

Layer 2 Limitations

Routed networks are typically used to ensure scalability and avoid Layer 2 loops. Layer 2-only networks use methods such as Spanning Tree defined in IEEE 802.1d, its extension Rapid Spanning Tree (IEEE 802.1w), or Multiple Spanning Tree (IEEE 802.1s). All three have the advantage of ensuring loop-free operation. Layer 2 loops overload the switches and can be complex to troubleshoot.

Photo by Fikri Rasyid on Unsplash

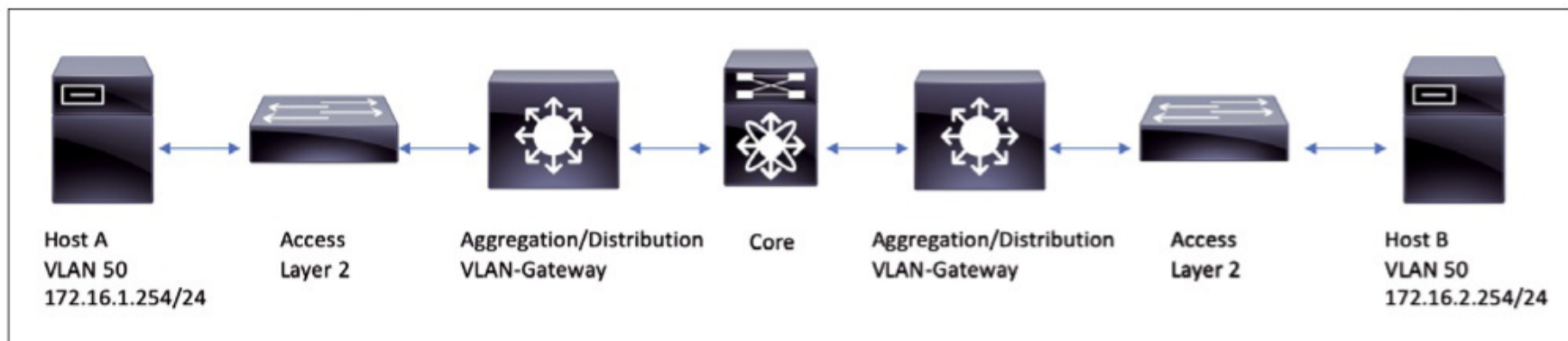


Figure 1: In the classic routed three-layer design, host A and host B must be on different subnets.

Freedom from loops is ensured for redundant links by path blocking, but this comes at the cost of compromising maximum throughput, because the maximum combined or load-distributed data rates cannot be fully leveraged in this way. Additionally, some switches impose restrictions on the number of spanning tree instances, which, in turn, limits scalability for methods that run one dedicated instance per VLAN, which is the case with Rapid Per-VLAN Spanning Tree.

Classical Network Architecture Limits

A routed design (Figure 1) addresses these issues – again, with different variants, such as a hierarchical three-layer model with core, distribution, and access switches. Routed access, in which VLANs are only implemented locally on the respective access switch or access switch stack, is increasingly being used on classic campus networks.

An alternative in data centers is the spine-leaf model, where central switches, the spines, are connected to the leaves by routed links, and the leaves are connected to the servers. These leaves are usually designed as top-of-rack switches. Normally, IT managers would limit the number of VLANs required to one top-of-rack switch at any time and route them on their own uplink levels to leverage the routing protocols' properties for redundancy and load balancing across multiple routed links. However, this setup results in VLANs being restricted to the rack only, which contradicts the requirement for flexibility in redundancy

and virtualization solutions referred to earlier.

Many data center services still require direct Layer 2 connectivity. One example is VMware vMotion, which enables live migration of virtual machines from one hardware setup to another. This configuration would not be possible in routed designs in different racks because IP-based re-addressing is impractical in most cases.

On local networks with end users, this connectivity does not play a major role. After all, the host usually doesn't care about the IP subnet it resides on, as long as it can access the required services over its gateway without the firewalls blocking the attempt. However, data centers need more, so how can you ensure – despite routed designs – that redundant services in distributed data centers can communicate with each other over Layer 2 while allowing virtual machines to move between racks without IP address changes?

VXLAN Extends VLANs

The solution is overlay protocols (Figure 2) to abstract the existing VLAN structure. One option is VXLAN, which allows you to extend a subnet or a Layer 2 broadcast domain on the overlay network to include a Layer 3 underlay network. Therefore, the benefits of routed designs such as equal cost multipathing (ECMP, i.e., equal load distribution between different paths) can be combined with the flexibility of direct Layer 2 links, which would again allow for redundancy within a subnet and even at different locations. VXLAN is defined by the Internet Engineering Task Force (IETF) in RFC7348 [1].

An example of VXLAN as an overlay could be a virtual server system that is redundant and uses the virtual IPv4 address 192.0.2.1. The address is mapped on server A at site A with the physical IPv4 address 192.0.2.2 or on server B at site B with an IPv4 address of 192.0.2.3, depending on

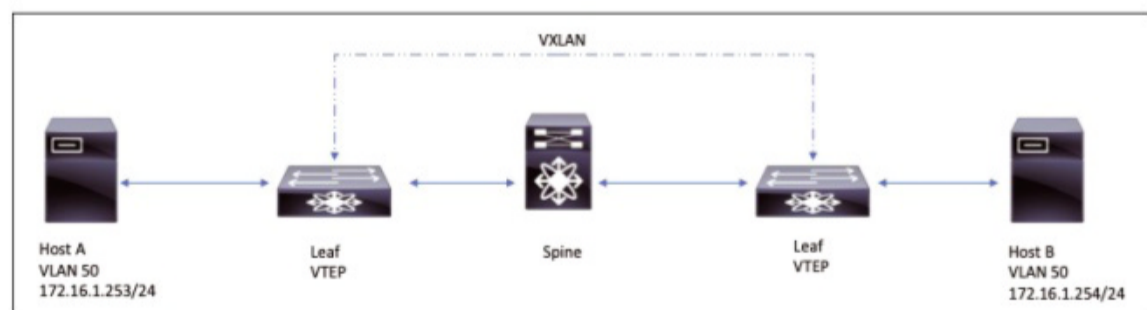


Figure 2: A spine-leaf design with VXLAN overlay: Host A and host B can reside on the same subnet despite different locations and a Layer 3 underlay.

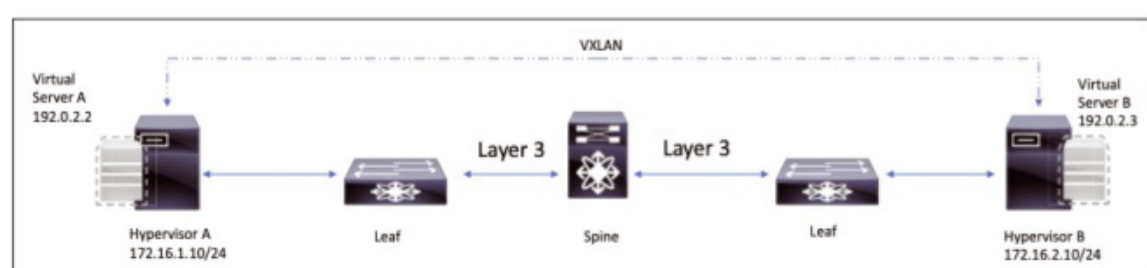


Figure 3: A VXLAN with a hypervisor as a VTEP enables coupling of redundant virtual servers in Layer 2.

the status (**Figure 3**). If server A fails, server B steps in to handle communications to and from the IPv4 address 192.0.2.1 by Gratuitous Address Resolution Protocol (Gratuitous ARP) message broadcast to all hosts on the subnet. This results in the MAC address for 192.0.2.1 being mapped in the ARP table on server B.

Without VXLAN, however, this setup would entail the risks and limitations mentioned earlier. Of course, it would be great if the Layer 2 link were not mandatory and the failover and load balancing mechanisms could be implemented directly on the client side and controlled centrally on the server side. Layer 2 coupling would then be obsolete for server-based services. In addition to these advantages, VXLAN can also extend the maximum number of virtual networks. Classic VLANs supported by the IEEE 802.1Q networking standard offer a maximum of 4,094 segments – up to 16 million for VXLANs. The technology accomplishes this by packaging the Ethernet frames from the hosts into UDP segments. In this case, that means from the server systems at site A and site B. For this to happen, the Internet Assigned Numbers Authority (IANA) has assigned VXLAN its own UDP port number, 4789. The VXLAN header is used as well. The VXLAN segment is identified by a 24-bit VXLAN network identifier (VNI), which is the external identifier confirming membership of a Layer 2 segment. VLANs are assigned to VNIs on respective VXLAN tunnel end points (VTEPs). VTEPs can be implemented both on a hypervisor (i.e., the virtualization layer between hardware and virtual machine) and on a switch or router.

Hosts that want to communicate over the VXLAN overlay do not need a dedicated agent. Otherwise, it would not be possible for appliances to engage, or it could result in the loss of vendor support. The host does not notice any difference between a direct Layer 2 connection and a VXLAN-based link, because the transport network is transparent for the host. Knowledge of the overlay is

the reserve of the VTEPs. They know the overlay structure and must each have an accessible IP address in the underlay to allow the VTEPs to communicate with each other.

However, there are also different procedures on the underlay network: In some cases, multicasting is required for dynamic peering of the VTEPs. The routing protocols in the underlay only enable the connection between the VTEPs, which offers a high degree of flexibility.

How VXLAN Works

To understand how VXLAN works, you first need to consider how Ethernet switches work. Initially, they create MAC address tables on the control plane, which uses Layer 2 source addresses acquired from the respective ports. In the case of an unknown MAC address, unknown unicast flooding normally occurs, which is equivalent to distributing the frame across all ports except the source port. After the MAC addresses have been acquired and the corresponding forwarding tables established, forwarding of the Ethernet frames can take place on the data plane.

In VXLAN, other methods create the forwarding decision tables. Whereas the acquisition target for legacy classic Ethernet switches is the MAC address to target port mapping, the target MAC address for VXLAN must be assigned to the VTEP IP. Only after this has happened can subsequent unicast frames to the respective target MAC address be encapsulated in VXLAN frames and delivered specifically to the target VTEPs.

The first, but also most inefficient and, at the same time, most insecure variant is data plane learning. This method is achieved by acquiring the source addresses and has the distinct disadvantage that the underlying network must be multicast capable, which might not be the case, especially on WANs or in public cloud networks. Multicasting is used on the underlay network to forward what are known as BUM frames (i.e., broadcast, unknown (unicast),

and multicast frames in the overlay segment). Data plane learning is also known as flood-and-learn, wherein participating VTEPs join a multicast group to be defined for the respective VNI. VTEPs in turn send traffic to this multicast address.

Separation of Control and Data Planes

One alternative is the Multiprotocol Border Gateway Protocol (MP-BGP) in combination with the Ethernet virtual private network (EVPN) framework, a control plane protocol that optimizes MAC address acquisition and avoids flooding Layer 2 frames over the overlay network. MP-BGP EVPN is responsible for both discovering the VTEPs and learning the MAC addresses and is achieved by first acquiring the MAC address for each VTEP locally, as on legacy Layer 2 networks, and then distributing the acquired MAC address along with the associated IP address to the other VTEPs with the use of a BGP extension that relies on MP-BGP EVPN. This exchange is also known as network layer reachability information.

By correlating this acquired information with the IP address of the source VTEP of this BGP update and the associated VNI, forwarding can subsequently take place in the reverse direction. IP to MAC address mapping is enabled by ARP.

MP-BGP EVPN Advantages

To sum up, the underlay network does not need multicasting if you have MP-BGP EVPN because unicast-based MP-BGP peering over TCP port 179 is used instead. This peering already supports multiple hops in internal BGP (iBGP) by default. The VXLAN VTEPs can be located several hops distance from each other, which makes deployment very flexible. External BGP (eBGP) involves some manual work on the part of the administrator in the form of an explicit configuration to make this option is available.

Multi-hop functionality in this case refers to the time to live (TTL) in the IP header, which largely corresponds to an interpretation of the hop count. The TTL for BGP packets in an eBGP session is set to 1 in the basic configuration, which means that the peers must be in the same subnet. If this is not possible or desired, the administrator needs to enable BGP multi-hop on the router to increment the TTL. In the case of iBGP peering, multi-hop is allowed in the basic configuration because the underlying TTLs are higher.

The network layer reachability information is therefore only exchangeable between the fixed MP-BGP peers, which offers benefits from an IT security perspective. Potential attacker VTEPs are therefore unable to establish a neighborhood without further tricks, and you have the possibility of authenticating the VTEPs with more modern methods by MD5 authentication in the BGP session or on the basis of TCP authentication options (TCP-AO). The objective of TCP-AO is to provide protection against blind insertions and replay attacks. In flood-and-learn VXLAN networks, unauthorized VTEPs can inject data into the network.

MP-BGP EVPN also solves the problem of suboptimal data forwarding. In this example, the server system at site B would have to communicate over the default gateway at site A to establish a communication relationship from the VLAN, which is possible with a distributed anycast gateway on each VTEP in a VNI.

Specifically, then, each VTEP can act as a gateway with the same virtual gateway IP address and the same virtual MAC address to route IP packets from locally connected hosts to other subnets. Packets no longer need to flow across the WAN between sites. The ARP suppression feature is also interesting, involving the local VTEP caching the IP and MAC information of the connected VNI and responding to corresponding requests. If a host connected to this VTEP sends an ARP request, the local VTEP first checks whether it has an entry in its cache. If so, it responds directly to the ARP request.

If iBGP is used, some special features need to be considered: iBGP peers normally need a routing relationship with all other iBGP peers because routing information from iBGP neighbors is not forwarded among themselves. This organization, of course, creates massive administrative overhead in large environments. In such a case, a type of hub-and-spoke design – known as a route reflector in iBGP – can be used. The associated VXLAN header is 8 bytes in total and comprises 8-bit flags, 24- and 8-bit reserved fields, and the 24-bit VNI. On transmission, the I bit for a valid VNI must be set to 1 and the R flags to 0. All reserved bits must also have a value of 0 when sent. The VNI defines the individual VXLAN overlay network.

VXLAN Challenges

Connections that use the maximum packet size, or maximum transmission unit (MTU) [2], are problematic and also known from other tunneling solutions. The additional header increases the size of the transmitted frames. VTEPs are not allowed to fragment traffic at the source according to RFC7348, and fragmented packets can be “dropped silently” into a black hole on the VTEP. The MTU needs to be increased, if possible, or adapted to the VXLAN overhead. However, expansion of the Layer 2 segment also widens some attack vectors. For Layer 2 trunks, VLAN hopping, among other things, adds vectors.

The attack surface for ARP spoofing, for example, is also larger. On multitenant networks, attacks such as VLAN hopping have catastrophic effects. The built-in security mechanisms of the BGP protocol for MP-BGP EVPN are recommended for flood-and-learn-based VXLAN networks. Additionally, you should note that not all switches and hypervisors support VXLAN. Even if they do so fundamentally, it makes sense to examine the supported control plane methods of the component in question. On the other hand, some other methods for

crossing Layer 3 boundaries (e.g., the relatively old Layer 2 Tunneling Protocol, L2TP) allow only point-to-point connections.

Another point of criticism is the lack of native integration of encryption methods, which would have to be implemented between the VTEPs by additional encryption procedures, further increasing complexity.

Additionally, VXLAN also does not have permanently assigned protocol identifiers. Therefore, a VXLAN tunnel cannot transport multiple payload types. Moreover, all fields are already permanently assigned in the VXLAN header, which makes extensions difficult. The lack of extensibility problem is addressed by the still fairly new Generic Network Virtualization Encapsulation (GENEVE) protocol, which is described in RFC8926. VMware already uses this in its NSX-T network virtualization. However, GENEVE currently lacks hardware implementations. Cisco Systems is now offering the first hardware switches with GENEVE support, but it remains to be seen whether other manufacturers will follow suit.

Conclusions

Classic VLAN stretching for Layer 2 site coupling is a quick solution, but it causes a few issues. VXLAN creates Layer 2 connectivity in a Layer 3 structure and is the obvious choice to meet increasing bandwidth and scalability requirements while maintaining the flexibility of Layer 2 interconnects for virtualized environments. Despite all the possibilities, however, you will always want to make sure at what points on your own network you need VXLAN. Solutions that do not rely on spreading Layer 2 broadcast domains are definitely preferable. ■

Info

- [1] VXLAN in RFC7348:
[<https://www.rfc-editor.org/rfc/rfc7348>]
- [2] VXLAN issues:
[<https://ripe77.ripe.net/presentations/54-the-sunset-of-vxlan-181016-final.pdf>]



Diving into infrastructure security

Trackers

How to deal with threat intelligence on the corporate network when the existing security tools are not effective. By Matthias Wübbeling

If you are responsible for protecting the corporate infrastructure, you probably have various security products in use to support you in this task – with a distinction made between network and host security. These terms primarily relate to where a security system is deployed. Firewalls or network intrusion detection systems (NIDS) are deployed at central points on the network, and antivirus programs or host intrusion detection systems (HIDS) are put on as many computers as possible in the enterprise.

Firewalls let you allow or deny network connections according to a fixed ruleset and primarily prevent requests from the Internet access to the corporate network. Depending on the size of your environment, firewalls can also isolate different departments. In addition to a fixed set of rules, a NIDS can use dynamic or heuristic detection techniques that classify network connections on the basis of metadata, as well as communication content, and trigger an alert when suspicious connections are detected.

Classic antivirus programs protect your computer by means of bytecode signatures or use heuristics that combine special system calls or file and

hardware access. HIDS also monitors user actions, files, and directories, as well as the registry and network connections. Some vendors specifically enhance their virus scanners with HIDS-specific features to enable comprehensive host security.

Even if you have implemented the normal protections in your company, as an administrator or security analyst you will want to take a more in-depth look in some situations (e.g., employees reporting irregularities in IT-specific processes, such as untypical behavior of their PCs or certain programs) to make sure attacks on your infrastructure are not happening. Also, attacks on high-ranking targets by smart hackers can encourage a look into infrastructures to search for signs of attacks.

TTPs, IoCs, and CoAs

In published media reports and analyses on the web, you will often find descriptions of the attacks and the malware used. The attackers' tools and approaches, to the extent that they are traceable, are summarized as tactics, tools, and procedures (TTPs) and contain indicators of the approach in terms of information

gathering, the attack targets, exploited vulnerabilities, and lateral movement details after a successful intrusion.

The traces that attackers leave behind on the network and on computers in the scope of their activities are referred to as indicators of compromise (IoCs). From IoCs, you can see whether you are currently feeling the effects of, or have been affected by, a particular attack. These IoCs can be files, registry entries or installed services, changes to the hosts file, and the like. Instructions on how to proceed in the event of a demonstrable attack should also form part of the online documentation of security attacks as courses of action (CoAs). This threat information can therefore be used to diagnose an attack over the network or potential malware infections. Before I look at how to use these tools to search for IoCs specifically, I first need to talk about the source of this information.

The basis of information (threat intelligence) gathering is IT security analysis, which is carried out by specialist companies, research institutions, and government entities. Whereas government and research institutions specifically search for and analyze malware, analyses by companies are often carried out in the scope of incident analysis (i.e., dealing with the actual malware infection).

Lead image © JPaget RFphotos, 123RF.com

Security Providers as a Source

The result of an analysis often comprises a technical description of the findings and a detailed report. The analyst collects everything that can help describe the malware in a designated data format. The best known are probably the integrated STIX/Cybox tool and MISP (formerly Malware Information Sharing Platform); both are based on JSON and therefore considered to be human readable. Only a few truly comprehensive examples of the use of STIX/Cybox exist. In fact, the best insight you can get is from the reports provided on the official website, such as one that relates to the Poison Ivy malware created by security specialists FireEye [1], as well as from the technical description [2].

If you are not able to analyze malware yourself, you will have to rely on the results of other people's work for your investigations. Fortunately, most analysts are happy to share their findings and make them available over appropriate sharing platforms or simply for direct download.

Exchanging Information with MISP

The MISP Threat Sharing Platform [3] is currently the tool of choice for processing threat intelligence. Locally installed instances can connect to community servers, enabling threat intelligence sharing across enterprise boundaries. Within MISP, you can define the visibility of individual entries in such a way that the content does not leave your company. Then you always have an overview of what you share with other companies and what information remains

internal. For an overview of some quite interesting MISP communities, see the *MISP Communities* page [4]. MISP offers some useful extensions. For example, if you use Snort as a NIDS, you can export existing rules directly in Snort format and roll them out immediately after you receive them. Of course, you will want to be careful when adopting third-party rules that you have not checked and initially only output warning messages for any criteria added automatically. Alternatively, you can obtain threat information without a running MISP instance. Often, the downloads offered are specialized in certain areas. For example, you will find many lists of IP addresses that have been discovered to propagate Spam or launch brute force attacks against SSH login servers. Unfortunately, the information contained is often only usable for a certain period of time because it very often involves dynamically allocated IP addresses from Internet providers.

More Sources

In addition to structured information, you will also find simple verbal descriptions of analysis results in various places. Proofpoint, for example, publishes detailed reports on its blog [5]. The authors describe and reference the TTPs and IoCs they were able to identify during their investigations. To provide an overview, all IoCs are once again clearly presented at the end of the articles and enriched with additional data such as URLs

and hash values of files and programs or libraries. Some contributions add Yara signatures, which are basically regular expressions you can use to search for patterns in text and binary data with different tools.

The more sources you find and consult, the more descriptions of attacks, malware, and procedures you will have. However, this knowledge alone does not give you any information about how your own systems may be affected. You can collect this information at various points in your infrastructure. Network-based IoCs can be located in centralized locations in most cases. You will want to block IP addresses directly in the packet filter and use your internal DNS server to redirect listed domains to local destinations and route the requests into a black hole. Of course, you will want to log the requesting computers and, if possible, isolate them automatically for further analysis in special network areas. The same applies to URLs, which you can easily transfer to your HTTP proxy to log and prevent access attempts.

To detect host-based IoCs on your organization's computers, you need technical support. One possibility is the well-known GRR Rapid Response open source tool [6]. With its client-server architecture, GRR enables centralized management and asynchronous processing of requests (known as hunts) on the clients. For a defined runtime, clients receive the requests as soon as they are connected to the corporate network.

The results are then published at the next opportunity. This approach also

Hunt Arguments			
Flow args	Paths	c:\windows\system32\notepad.*	
	Action	Action HASH	
Flow runner args	Flow name	FileFinder	
Output plugins	Plugin name	BigQueryOutputPlugin	
	Plugin args		
Runner Arguments			
Client rule set			

Figure 1: In this example of creating a request for GRR clients, you are looking for notepad.* in the System32 folder.

covers mobile devices and home office computers that only sporadically connect to the corporate network. Of course, this means you need to install the GRR clients on your systems; it is not suitable for use on the fly. To test the tool, you can simply store a file named `notepad.*` in the Windows system folder after installation (Figure 1). After clicking *Create Hunt*, you need to enable the hunt and wait some time for the first clients to execute the target. While the hunt is active, increasing numbers of clients will receive it and run the search until you disable it again. Next, you evaluate the search. Figure 2 shows the results for one of the clients. This response also reveals the variety of search options with GRR for files in terms of timestamps or hash values. To practice, just compare the patch level of your Windows installations with different hash values.

Handling Security Incidents

If you have the capabilities in your company to run incident analysis for attacks and malware infections yourself, you can use The Hive [7] in addition to GRR. In combination with Cortex, I looked at The Hive for incident analysis in a previous article [8], which gives you both an overview and a comprehensive automation tool

for the required analysis work. Cortex also lets you automatically enable protections in your infrastructure in the form of responders. Finally, threat intelligence can also be incorporated into a company’s risk analysis. If your company is part of a sharing community, you will receive valuable information, such as attacker groups, attacks within your industry, and attacks against specific classes of software. If you use the software in question, or something similar, in your organization, you need to look into the increased risk and try to respond to it. The systematic use of publicly available information on vulnerabilities (Common Vulnerabilities and Exposures, CVEs) means you are always aware of attack vectors on your infrastructure.

Conclusions

Threat intelligence is a way to learn about an attacker’s *modus operandi*, methods, and tools. Additionally, it gives you valuable advice on what to look for when searching for malware or backdoors on your systems. The data you find in structured threat information mostly consists of signatures used by virus scanners in the same or a similar way. Meanwhile, polymorphic or other types of dynamic malware cannot be tracked down reliably with the rather static

descriptions in STIX/Cybox or similar formats. You will find several providers of commercial threat intelligence feeds online; however, the content can differ significantly. Before subscribing, you should consider whether a feed is useful and up to date.

Info

- [1] Poison Ivy: [https://www.mandiant.com/resources/poison-ivy-assessing-damage-and-extracting-intelligence]
- [2] Poison Ivy example: [https://oasis-open.github.io/cti-documentation/examples/example_json/poisonivy.json]
- [3] MISP Threat Sharing platform: [https://www.misp-project.org]
- [4] MISP Communities: [https://www.misp-project.org/communities/]
- [5] Proofpoint blog: [https://www.proofpoint.com/us/blog/threat-insight]
- [6] GRR Rapid Response: [https://github.com/google/grr]
- [7] The Hive: [https://thehive-project.org]
- [8] “Incident Analysis with The Hive and Cortex” by Matthias Wübbeling, ADMIN, issue 66, 2021, [https://www.admin-magazine.com/Archive/2021/66/Incident-Analysis-with-The-Hive-and-Cortex/]

The Author

Dr. Matthias Wübbeling is an IT security enthusiast, scientist, author, consultant, and speaker. As a Lecturer at the University of Bonn in Germany and Researcher at Fraunhofer FKIE, he works on projects in network security, IT security awareness, and

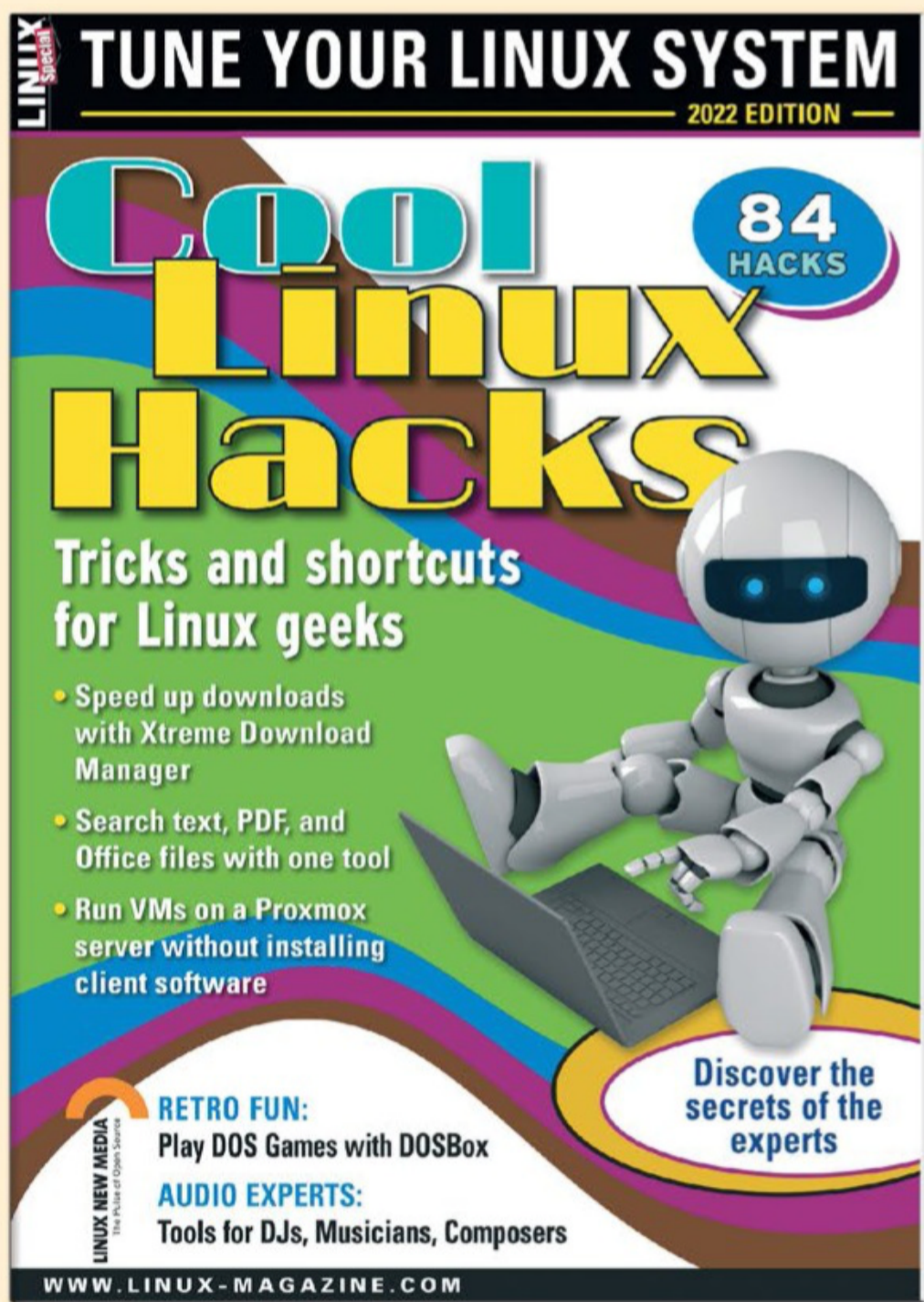
protection against account takeover and identity theft. He is the CEO of the university spin-off Identeco, which keeps a leaked identity database to protect employee and customer accounts against identity fraud. As a practitioner, he supports the German Informatics Society (GI), administering computer systems and service back ends. He has published more than 100 articles on IT security and administration.

Value			
Client id			
aff4:/C.04f247bb91628f0b ⓘ			
Payload	Stat entry	Aff4path	aff4:/C.04f247bb91628f0b/fs/os/C:/Windows/System32/notepad.exe ⓘ
		St mode	-rwxrwxrwx
		St ino	0
		St dev	0
		St nlink	0
		St uid	0
		St gid	0
		St size	193536
		St atime	2016-01-27 02:31:59 UTC
		St mtime	2015-07-09 17:57:57 UTC
		St ctime	2016-01-27 02:31:59 UTC
	Hash entry	Pathspec	Pathtype OS
		Path	/C:/Windows/System32/notepad.exe
		Path options	CASE_LITERAL
		Sha256	933e1778b2760b3a9194c2799d7b76052895959c3caedefb4e9d764cbb6ad3b5
		Sha1	a7bbc4b4f781e04214e0e0e69a766c76681aa7eb
	Md5	b32189bdf6e577a92baa61ad49264e6	
	Pecoff sha1	5ee3bf07f4dfb08bed7be66bc2fdd290741d3a9f	
	Pecoff md5	63052e1cc7b68b3a96c7483b69d91867	
	Pecoff sha256	82a726edbc9c3f903f80f5a6a3c138cd32179932bbeec20889df4b629c1a2bca	
Payload type		FileFinderResult	
Timestamp		2017-01-24 12:46:19 UTC	

Figure 2: The results of a GRR client in response to a request.

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/specials



DNS name resolution with HTTPS

Confidential Game

Now that web content is encrypted by HTTPS, the underlying name resolution is often unprotected. We look at the classic DNS protocol and investigate whether DNS over HTTPS could be the solution to ensure the confidentiality of DNS requests. By Matthias Wübbeling

Besides the common routing protocols, the Domain Name System (DNS) is one of the longest serving infrastructure protocols on the Internet. As the number of participants on the jointly developed Internet (initially ARPANET and later NSFNET) began to grow, the manual overhead involved in maintaining the hostname file (/etc/hosts) exploded. The first draft defined in RFC882 and RFC883 turns 40 next year.

Fortunately, traditional attacks such as DNS spoofing and cache poisoning are practically impossible today. DNS has seen several enhancements since its introduction, which retrospectively reflects a good design that is obviously extensible in many directions. The problem now is the unmanageable number of top-level domains, country domains in different languages that use different character sets, DNS over TCP for particularly large queries and responses, and many other major and minor extensions. Most resolvers now secure their queries to the

authoritative name servers with DNS security extensions (DNSSEC) and other technologies to avoid receiving undesirable spoofed responses.

DNS also forms the basis for protecting many other application protocols today: The main examples are HTTP for issuing certificates for web access and SMTP for securing email communication with DMARC.

Privacy and Manipulation

Whereas DNS itself has become significantly more secure, the unencrypted route between clients and resolvers is left as an attack vector for hackers and snoopers. The data is routed by the User Datagram Protocol (UDP) without protection. One issue that has not yet been fully resolved is the privacy of DNS requests. Clients also need to be able to trust the resolvers to deliver correct responses – think protection against cache poisoning and censorship – and to keep client data confidential, or preferably not store the data at all.

DNS requests map users' web activity in a fairly accurate way and are therefore an open invitation to abuse: whether in the form of DNS server operators storing queries and reselling them for advertising purposes, or providers messing around with the unencrypted data flowing through their networks. The most important argument in this discussion is therefore trust, which raises the question of trusting or mistrusting your own Internet provider; associations that provide protection against censorship [1] or offer data protection [2]; or large DNS providers such as Cloudflare [3], Google [4], or Quad9 [5]. Quad9, by the way, deliberately filters DNS responses to protect users against malicious domains that propagate malware or phishing.

Even if the provider of the DNS resolver were trustworthy, communication with the provider would still be unencrypted, which is why different protocols protect the connection between client and server. As early as

2016, DNS over TLS (DoT) was defined as a protocol that secures name resolution requests with TLS over TCP. Therefore, the requested domain is only visible to the DNS server to which the request is addressed. To compensate for potential speed disadvantages, the TCP connection is kept alive after the request and reused for the next request. Communication is routed through port 853 and is encrypted on the network, but recognizable as DNS traffic through the port.

DNS over HTTPS

In addition to DoT, another encryption method, DNS over HTTPS (DoH), was launched. Released in 2018, the protocol is designed to protect user privacy extensively, going even further than DoT. The DNS request is sent by HTTPS to a web server that operates a matching API. Most implementations let you choose between different response types. The most commonly used examples are probably *application/dns-json* and *application/dns-message*. The `curl` utility lets you see at the command line how a query appears:

```
curl -s -H ,accept: application/dns-json' 2
,https://cloudflare-dns.com/dns-query2
?name=linux-magazine.com&type=A" | jq .
```

The `jq` here makes for nicer output formatting, the `-s` suppresses the status output for `curl`, and `-H` specifies the Accept header and chooses JSON output in the request. I used the Cloudflare DNS server for this example. Of course, you can choose an alternative server depending on your preferences. When using *application/dns-message*, the server expects a Base64-encoded message that is equivalent to a classic (binary) DNS query. The response is again not so pretty for viewing on the console; it contains binary data to match the request.

Secret DNS Requests

Because DoHs are ordinary HTTPS requests, they are also indistinguishable on the outside. If a web server

provides both web page content and DoH, name resolution queries can no longer be distinguished even on packet filters or firewalls unless you also examine the TLS connections there.

Therefore, external DNS sources can be used behind restrictive firewalls, possibly bypassing active DNS filters. Firefox already made DoH the default for users in the US on Cloudflare servers in 2020. You can enable DoH in the browser settings if you like and can also enter an alternative server. When you enable DoH in your browser, the system's settings are preserved as a fallback because DoH does not work if, for example, you are in a hotel behind a captive portal. Name resolution by DNS will normally work, but HTTP(S) connections are blocked until you log in. By the way, you might experience problems even with DoT if the redirect to the captive portal relies on DNS hijacking.

DoH also can be used to attack corporate networks. JavaScript and dynamic queries (Ajax) allow queries to an internal DoH server for gathering additional information about the corporate network. The appropriate CORS headers in the DNS server responses could prevent this attack. Today, many companies use client DNS queries in their monitoring to check for malware infestation, for example. Such mechanisms can no longer be easily used with DoH if the malware of the future also implements DoH against standard servers from Google or Cloudflare.

DoH in Everyday Operations

The popular DNS servers already offer DoH interfaces. If you forward the requests from a proxy web server, you can hide them in your normal HTTPS traffic. Moreover, the classic HTTP Authenticate methods for authenticating clients before they use the DNS server do not work. To implement a modicum of protection for your DoH server, you can adapt the URL for the request. In fact, this allows your HTTP proxy to then address different back-end servers

based on the URL and return filtered responses for some users.

Conclusions

The entire Internet communication is built on the DNS system, yet the tried-and-tested service by no means receives the attention it deserves. DoT and DoH change the outlook. This article provides insight into how DNS over HTTPS works. In this case, too, innovation has two sides; you will need to assess the advantages and disadvantages of DoH on individual merit. Common software tools such as web browsers already support it. Therefore, it is up to you to decide whether to continue using your provider's DNS service, whether encrypted or unencrypted, or whether to switch to a provider with a clear focus on data protection, possibly even including malware protection. As an administrator, you will definitely want to keep an eye open for potentially hidden DoH traffic on your network. ■

Info

- [1] Censorship-free DNS server by Digitalcourage: [\[https://digitalcourage.de/en\]](https://digitalcourage.de/en)
- [2] DNS data protection: [\[https://www.cloudflare.com/learning/dns/dns-over-tls/\]](https://www.cloudflare.com/learning/dns/dns-over-tls/)
- [3] Cloudflare DNS: [\[https://www.cloudflare.com/learning/dns/what-is-1.1.1.1/\]](https://www.cloudflare.com/learning/dns/what-is-1.1.1.1/)
- [4] Google DNS: [\[https://developers.google.com/speed/public-dns/\]](https://developers.google.com/speed/public-dns/)
- [5] DNS service Quad9: [\[https://www.quad9.net/\]](https://www.quad9.net/)

The Author

Dr. Matthias Wübbeling is an IT security enthusiast, scientist, author, consultant, and speaker. As a Lecturer at the University of Bonn in Germany and Researcher at Fraunhofer FKIE, he works on projects in network security, IT security awareness, and protection against account takeover and identity theft. He is the CEO of the university spin-off Identeco, which keeps a leaked identity database to protect employee and customer accounts against identity fraud. As a practitioner, he supports the German Informatics Society (GI), administering computer systems and service back ends. He has published more than 100 articles on IT security and administration.



Manage guest accounts in Azure Active Directory

Welcome, Guest

Cross-tenant access settings and user-friendly Access Reviews simplify the management of guest accounts in Azure Active Directory. *By Florian Herzog*

Efficient collaboration while maintaining security standards is often difficult, especially when it comes to the end-user experience. Business-to-business (B2B)

transactions are made in the cloud, but the cloud itself is a dangerous place. Passwords should not be accepted as the only credential, and multifactor authentication (MFA)

should be the standard. In this context, it is less than useful that the MFA status is – thus far – only valid within the boundaries of the tenant and that other tenants are not trusted. If two communication partners are committed to a modern work approach with zero trust, things become even more difficult:

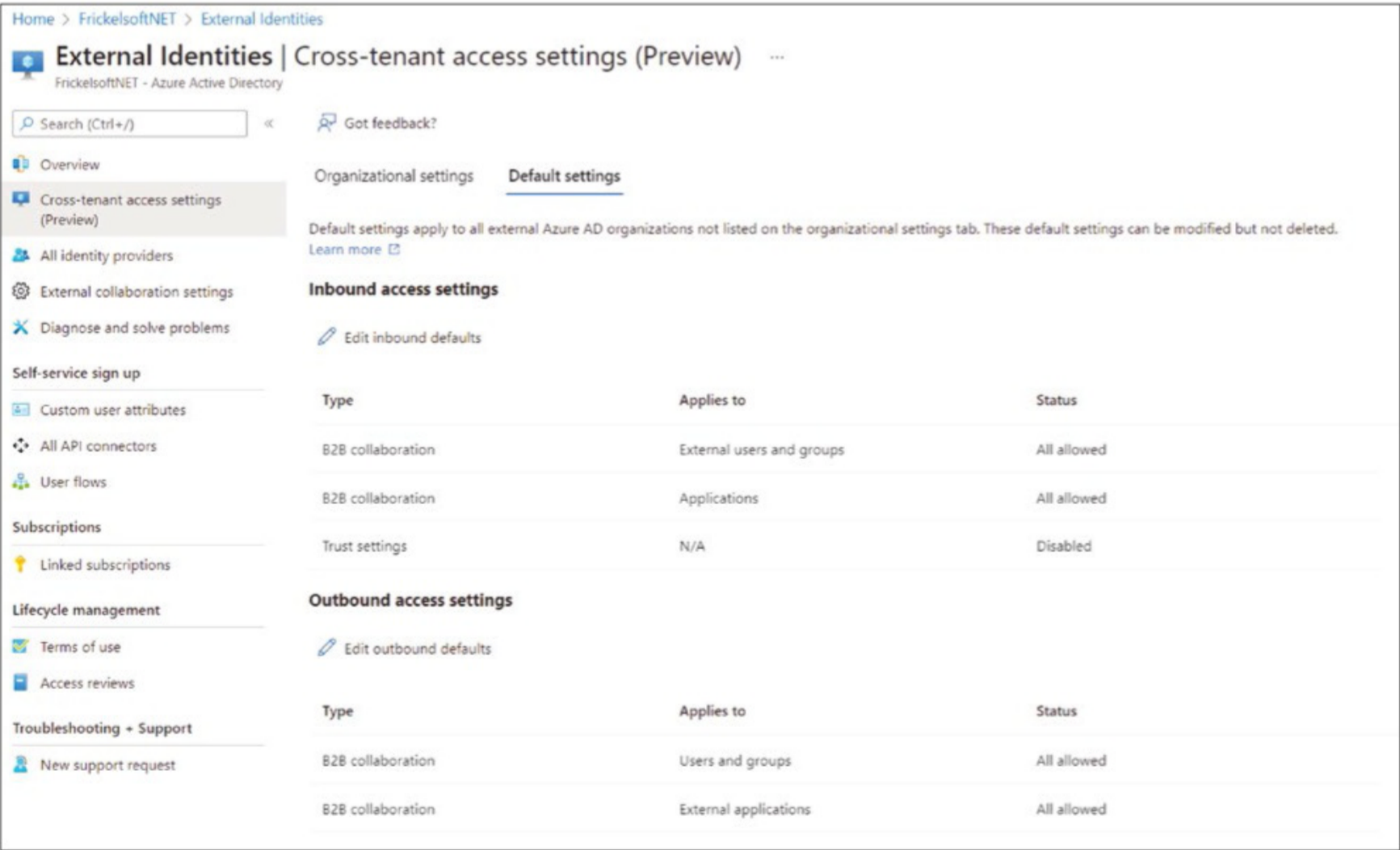


Figure 1: The default settings for cross-tenant access for B2B collaboration give you free rein. If so desired, you can establish more granular controls at the user or group level.

Photo by Jon Tyson on Unsplash

The device status cannot be transferred either.

New Possibilities with xTAS

Cross-tenant access settings (xTAS) offer an opportunity to improve this situation. The feature is designed to control collaboration fully across tenant boundaries and let organizations control inbound and outbound collaboration by defining tenant-wide and partner-specific rules ([Figure 1](#)).

These policies describe exactly how collaboration with partners should look. For your Azure Active Directory (AAD), resources, apps, and users, you can define which partner companies you want to allow for collaboration, under what conditions this takes place, and the partner companies allowed to invite your employees. Restrictions previously applied for both: You could either define the companies approved for collaboration freely or use an allow or deny list. A more granular approach was not available. Until now, you could not use the B2B settings to control who from your own company was allowed to be a guest in other companies.

xTAS changes this state of affairs: In both trust directions, you can define for each partner company whether collaboration is desired, and it even works at the user and group levels. For example, if you only want one project team to be able to collaborate with a supplier in that supplier's Office 365 while ruling out the rest of the organization, you can simply define this arrangement with group memberships. In this way, you ensure granular control and help delegate management tasks to the project teams, letting them steer the group themselves.

Trusting a Partner Company

You can create the ruleset on two levels. You start by defining a standard company-wide policy that governs collaboration: Are your own employees allowed outbound collaboration? Should you generally be able to invite

other companies? Once you have created the corporate policy, go into detail and describe collaborations with individual partner companies: Which of your employees, defined by group membership, can collaborate with Fabrikam [\[1\]](#)? Which project teams are allowed to collaborate with Contoso – and how should Contoso be allowed to operate in its own tenant environment? The individual configuration per partner company regulates collaboration in a very granular way. You always manage two aspects: incoming access and outgoing access. If no rule exists for a partner, the default rule applies.

Especially if close collaboration with selected partners is the norm, or if other clients in the same group of companies need to be more closely connected, you will probably want to fine-tune some security configurations. If you trust account management, MFA and credentials, and partner device management, you can take a user's MFA status and device health from Intune and accept it in your conditional access (CA) rules. In this way, you are not only trusting the users, but also the other company's security settings and processes. If contractual agreements stipulate that the respective IT processes must be disclosed and must comply with certain standards, you will probably want to reflect these standards in the collaboration settings. An advantage for the employees is that if MFA is mandatory, staff do not need to set up and complete MFA twice when working across tenant boundaries. Your own CA rules could also be simplified: You will not need to define and maintain as many MFA or device exceptions for partners.

Granular Collaboration Management

By default, xTAS allows free collaboration and your coworkers can be invited by others to participate. You can create advanced policies in *Azure Active Directory | External Identities | Cross-tenant access settings* on the Azure portal. You will see two

tabs – *Organizational settings* for the detailed configuration of partner companies and *Default settings* for the default rules – which give you an overview of the basic settings broken down into *Inbound access settings* and *Outbound access settings*. Clicking *Edit* lets you prohibit collaboration completely or allow it in a granular way by exempting individual users or groups.

The *Organizational settings* option lets you configure the same settings, but for a partner. If no settings are stored yet, you can start the creation process by selecting *Add organization*. On the right side of the screen, you can now specify your business partner's AAD client: The domain name is all you need (e.g., *contoso.com*). Once you have selected the correct tenant, it is displayed in the main window. Initially, the *inherited from default* settings apply – for both incoming and outgoing access. Clicking on one of the two links will take you to the detailed settings page. *Customize settings* lets you define who is allowed to collaborate in the inbound and outbound directions. The default for the inbound settings is *All external users and groups*, which means that all employees at Contoso are allowed to collaborate with you and have access – if they have been invited by one of your coworkers. If this is not desired and you only want to grant collaboration to a very specific group of Contoso employees, regardless of invitations, you need to use *Select external users and groups*. You will then need to agree with Contoso which users and groups are to be allowed. You then enter the object IDs of the permitted collaboration partners from the Contoso tenant in a list. The same functionality, only in reverse, is available for outbound settings. You store the object IDs of the users and groups who are allowed to work on the Contoso tenant from your own client.

The MFA and device data settings for inbound access are found in the *Trust settings* for Contoso. Once you are sure that the partner company's identity verification and device

management complies with your requirements, you can trust the data from Contoso's tenant in *Customize settings* by selecting from various options. *Trust multi-factor authentication from Azure AD tenant*, *Trust compliant devices*, and *Trust hybrid Azure AD joined devices* are your choices. You can mix the settings to suit your needs, driven by the level of confidence you have in Contoso's IT managers. Selecting one of these settings causes the Conditional Access Rules on your tenant to wave Contoso through, assuming the Contoso tenant has completed MFA and device testing.

Enabling Automation

To automate B2B collaboration with xTAS, you need to define the trust settings for B2B for inbound and outbound collaboration. The Graph API tool settings are found in the `crossTenantAccessPolicy` object. However, the main object only contains the control data. The tenant-wide settings are found in the `default` sub-object, and the partner-specific settings can be found in `partners`:

```
GET https://graph.microsoft.com/beta/2
policies/crossTenantAccessPolicy/default
```

Initial information acquisition, before you get started with automation, takes place in the usual way with Postman or Graph Explorer [2],

which let you explore the API structure. A more detailed description by Microsoft of the cross-tenant access settings API is available online [3]. All settings from the administration interface are also reflected in the API: `inboundTrust` for the MFA and device settings and `b2bCollaborationOutbound` and `b2bCollaborationInbound` for the basic configuration for outbound and inbound collaboration. If you are only interested in detailed aspects of the default settings, you can use `$select` to retrieve the details in the usual way:

```
GET https://graph.microsoft.com/beta/2
policies/crossTenantAccessPolicy/2
default?$select=inboundTrust
```

For an overview of all partner-specific settings, use:

```
GET https://graph.microsoft.com/beta/2
policies/crossTenantAccessPolicy/partners
```

The list shows all of the defined partners and their special options. If a setting is tagged *null*, the tenant default settings from the default object are in effect. If you are interested in a particular partner, you can request the graph with the tenant ID:

```
GET https://graph.microsoft.com/beta/2
policies/crossTenantAccessPolicy/2
partners/72f988bf-86f1-41af-91ab-2d7cd011db47
```

The results will look similar to Listing 1. To change individual settings (e.g., in the MFA and device settings), you can apply the `PATCH` function to the `inboundTrust` object:

```
PATCH https://graph.microsoft.com/ 2
beta/policies/crossTenantAccessPolicy/2
partners/72f988bf-86f1-41af-91ab-2d7cd011db47
```

You need to pass in the details of the change to the graph as the payload in the request body. Even if you want to change only one of the three options in `inboundTrust`, you have to describe all the settings

again (i.e., define the desired state of the `inboundTrust` object):

```
{
  "inboundTrust": {
    "isMfaAccepted": true,
    "isCompliantDeviceAccepted": true,
    "isHybridAzureADJoinedDeviceAccepted": 2
    true
  }
}
```

If your company does business with a large number of partners and you want to simplify the resolution from a domain like *contoso.com* to the tenant ID required for B2B settings in Graph, you can ask Azure AD and its OpenID Connect configuration endpoint for help. The standard requires metadata for the domains, so if you type

```
https://login.microsoftonline.com/2
domain.com/v2.0/.well-known/2
openid-configuration
```

in your browser or in a script and parse the response, you can easily read the tenant ID from the response.

Better Overview with Workbook

Once you have gained an overview of the possible settings, you only need to decide on the business partners for whom you want to define detailed settings and what these settings should contain. Azure AD gives you an overview of the existing collaborations with partner companies in the form of a workbook that divides the collaboration into inbound and outbound.

The workbook can be found in *Azure Active Directory | Workbooks* in the *Cross-tenant access activity* workbook. To ensure meaningful insights and data, you need to export sign-in logs from Azure AD to a log analytics workspace. The workbook can then also access earlier data than the history of the last 30 days [4]. The workbook gives you information on external persons logging in to your tenant – or your own employees logging in to partner tenants (Figure 2).

Listing 1: Tenant Settings with Graph API

```
{
  "@odata.context": "https://graph.microsoft.com/
beta/$metadata#policies/crossTenantAccessPolicy/
partners/$entity",
  "tenantId": "72f988bf-86f1-41af-91ab-2d7cd011db47",
  "isServiceProvider": null,
  "b2bCollaborationOutbound": null,
  "b2bCollaborationInbound": null,
  "b2bDirectConnectOutbound": null,
  "b2bDirectConnectInbound": null,
  "tenantRestrictions": null,
  "inboundTrust": {
    "isMfaAccepted": true,
    "isCompliantDeviceAccepted": true,
    "isHybridAzureADJoinedDeviceAccepted": false
  }
}
```

Access Reviews

Full control over collaboration and the guest accounts in your own tenant means you need to clean up regularly. With Microsoft Graph and PowerShell, you can discover the last login dates of all guests and delete guests with long periods of inactivity. If you are not just interested in getting rid of inactive guests, but also want to check their permissions for your tenant, Azure AD Access Reviews offers a couple of options: You can use this feature to check individual resources (e.g., group or team memberships, privileged roles, applications) or entire access packages – just for guests, if so desired.

One new feature in Access Reviews is a multilevel review with up to three stages. Everybody ultimately needs to agree on a user keeping track of group memberships and permissions. This task is especially interesting for guest accounts, when employees are overwhelmed by the large number of access checks needed to check up on “their” guests.

In the first stage, you can have the guests confirm that they are still members of projects and therefore still need access to work packages and teams. As each guest confirms this status, and only then, the guest is then presented to an internal employee, who says yes or no to access. Guests who do not answer or return a negative response do not even make it to the second round.

The same thing applies for guests governed by the *Block sign-in and remove after 30 days* feature provided by Access Reviews. Instead of kicking guests out of a group, you can prevent them signing in to your tenant and, if they fail to contact the help desk, remove them and delete the guest account after 30 days. *Multi-stage reviews* works in the same way. All guests first need to express their ongoing interest in collaboration, and only those guests who confirm are subject to closer scrutiny by your staff.

Creating a multilevel review is simple: Just navigate to *Azure Active Directory | Identity Governance | Access Reviews* and create a new review by

selecting *New Access Review*. In this example – reviewing guest accounts for guests who are members of a group – you need to select *Teams + Groups* for *Select what to review*, then decide on the group and select *Guest users only* as the scope. In the *Reviews* step, check the *(Preview) Multi-stage review* box. This option tells the portal to display the *First stage review* and *Second stage review* options, where you can specify a reviewer and a review length in days for each. If you need a third review stage, you can enable it by selecting *Add a stage*.

The *Users review their own access* option is a good choice for the first stage; for the second, you will want *Group owner(s)* or *Selected user(s) or group(s)*. The *Reveal review results* section lets you determine whether the reviewers in later steps should be able to see the previous decisions. Finally, you need to determine which guests pass from one stage to the next and remain in the review pool.

The *Reviewees going to the next stage* setting gives you checkboxes to define

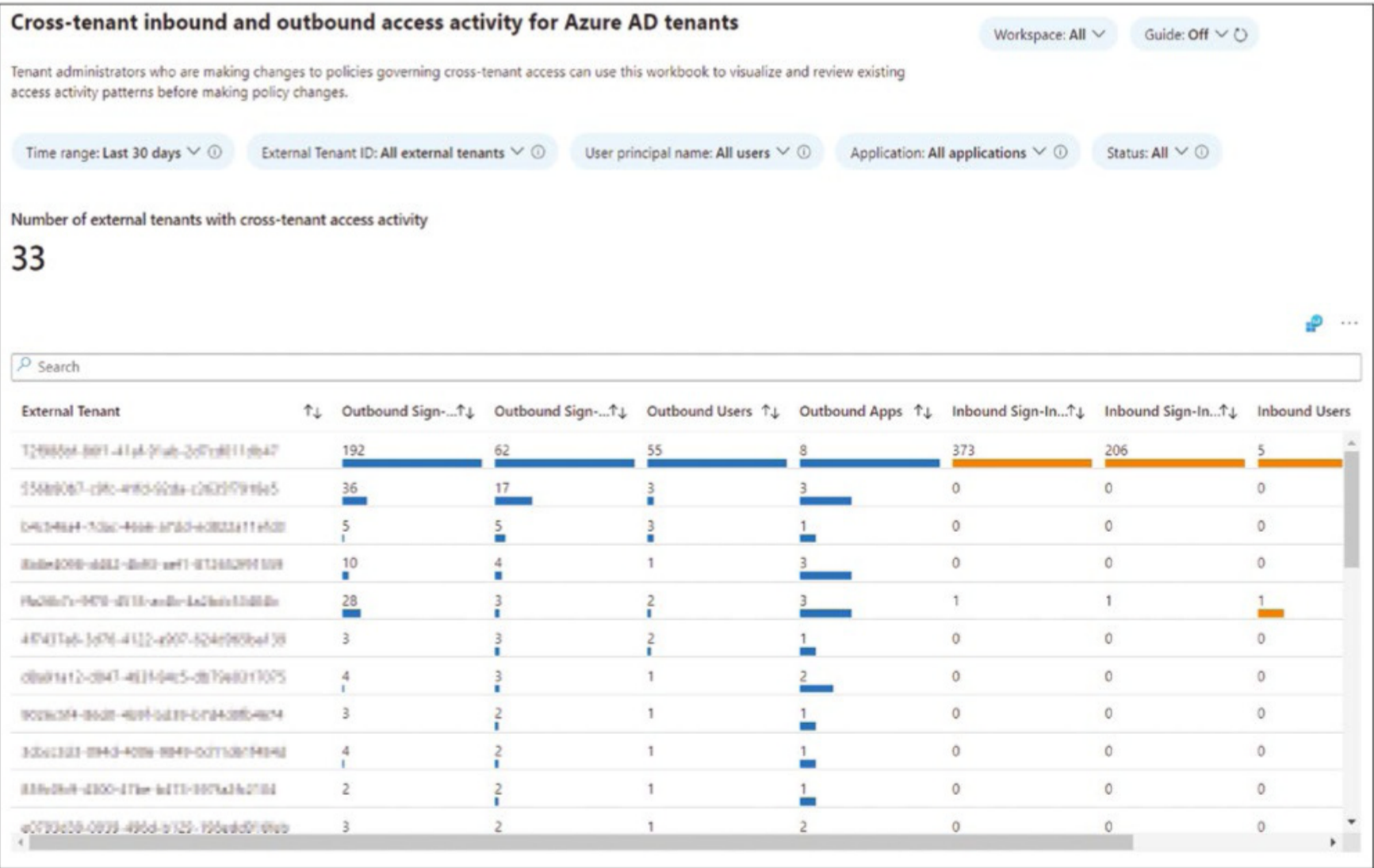


Figure 2: The cross-tenant access activity dashboard shows the existing collaboration.

the desired responses. *Approved reviewees* would mean that only guests who have already consented make it to the next level. If you additionally want to have non-responders reviewed in the second stage, include *Not reviewed reviewees*. The other settings in the Settings menu are not so important and have little influence on the multistage settings. Finally, assign a review name, and you can get started.

Delegate Wherever Possible

If you are only reviewing B2B guests, an additional setting will

appear in the penultimate step when you create an Access Review (**Figure 3**): *Action to apply on denied guest users*. This setting lets you stipulate that guests are simply removed from the group or application. However, you could also go for *Block user from signing-in for 30 days, then remove user from the tenant*. Of course, this is a sledgehammer method that culminates in the removal of the guest if combined with a multilevel review. If the external user has not contacted you, though, and an internal second examiner is not sure, deletion could be the best remedy.

Multistage reviews are useful for three areas of operation: reaching a quorum, escalating reviews, and delegating review work. You reach a quorum when reviewing users by having several stages confirmed in succession. Access only continues at the end of the process if everyone agrees that certain users should continue.

You can map escalation processes with multistage reviews if you want to have a second reviewer cross-check the rejected users, if any users are marked *don't know*, or if users received no response from the first reviewer. The second reviewer

can then correct opinions or enter them in the first place; no results means no access.

Of course, no self-respecting admin likes to spend time on repetitive tasks, clicking through line after line of users to be confirmed. Alternatively, you can first delegate the main work to the actual beneficiaries of group memberships or access to applications: the users themselves. Getting users to participate in the review means that all users who fail to report or say no are dropped before the second or third stage, reducing the workload on the second- and third-stage reviewers.

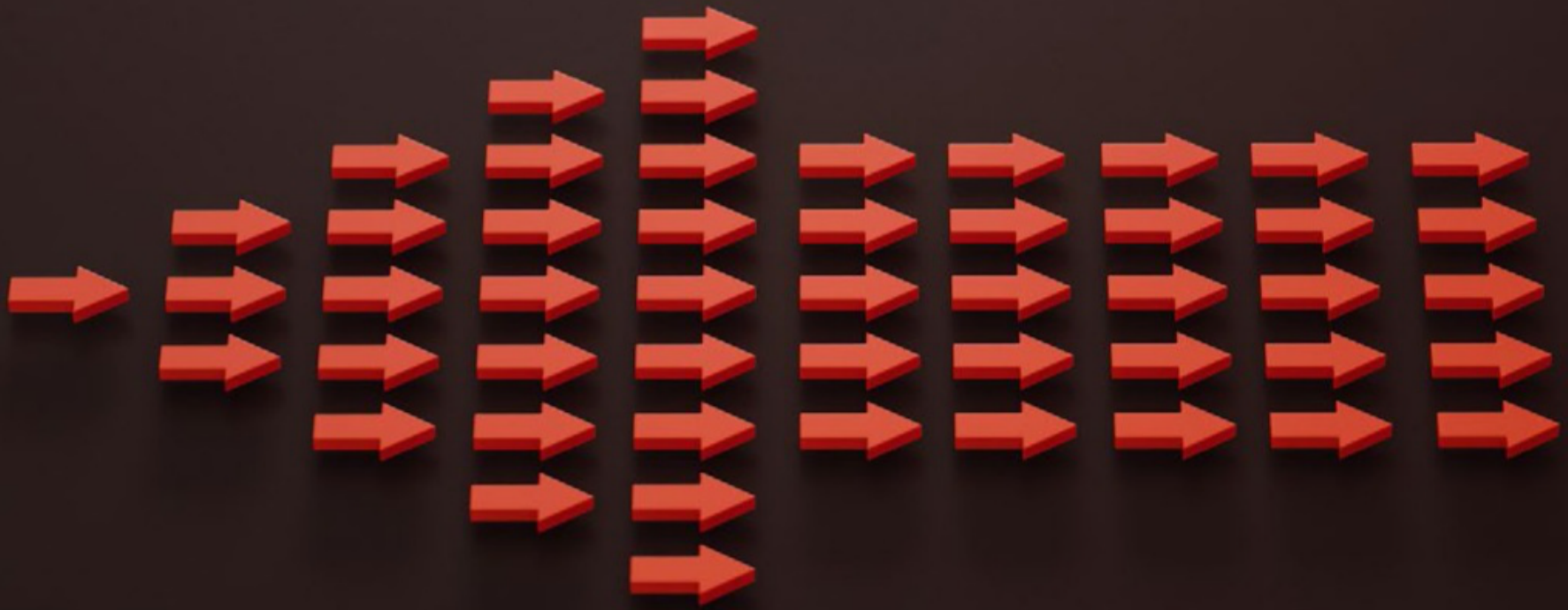
Conclusions

If cloud collaboration is a part of your working life, and you enable it for your employees, you are likely to discover that as the number of partners increases, the trust placed in them does not always keep pace. Relationship depths with business partners need to be mapped in a fairly granular way, even in B2B environments, and the cross-tenant access settings in Azure AD let you do this. To make it easier on yourself when cleaning up your business partners, you first need to engage the users themselves by imposing multistage reviews to let them say whether further collaboration is desired and necessary.

Info

- [1] Fabrikam: [\[https://docs.microsoft.com/en-us/previous-versions/windows/desktop/identity-lifecycle-manager/ms694611\(v=vs.85\)\]](https://docs.microsoft.com/en-us/previous-versions/windows/desktop/identity-lifecycle-manager/ms694611(v=vs.85))
- [2] Microsoft Graph Explorer: [\[https://developer.microsoft.com/en-us/graph/graph-explorer\]](https://developer.microsoft.com/en-us/graph/graph-explorer)
- [3] Cross-tenant access settings API: [\[https://docs.microsoft.com/en-us/graph/api/resources/crosstenantaccesspolicy-overview?view=graph-rest-beta\]](https://docs.microsoft.com/en-us/graph/api/resources/crosstenantaccesspolicy-overview?view=graph-rest-beta)
- [4] Cross-tenant access activity workbook: [\[https://docs.microsoft.com/en-us/azure/active-directory/reports-monitoring/workbook-cross-tenant-access-activity\]](https://docs.microsoft.com/en-us/azure/active-directory/reports-monitoring/workbook-cross-tenant-access-activity)

Figure 3: Access Reviews lets you define multistage access in (and determine the reviewers for) each stage.



Rolling back from Windows 11 to Windows 10

Get Back!

If you want to try Windows 11 but keep your options open, we show you how to install Windows 11, restore it to its factory default state if it's misbehaving, or roll back to Windows 10 - as long as you act within 10 days after updating. By Thomas Joos

Many users are updating to Windows 11, the version that was never supposed to exist. But what can you do if the update fails and you want to go back to Windows 10 (without losing any data, of course)? In this article, I show you how to get back to Windows 10. This approach is available for 10 days after updating to Windows 11, and major problems are not likely to occur. But the Windows 11 uninstall features do expire over time.

Installing Windows 11 Update

Microsoft provides the free update to Windows 11 as a feature update through Windows Update. The installation can take place directly in Windows 10 or from installation files. Both variants have the same options for uninstalling or repairing the operating system afterward.

In all cases, you need to make sure the hardware you are using is supported before upgrading to Windows 11. An open source tool named WhyNotWin11 [1] tests whether a computer's hardware is

suitable for the upgrade. For example, a trusted platform module (TPM) is installed on many computers but not enabled in the UEFI/BIOS. If a TPM is not installed on the PC, you can find out whether a TPM chip can be retrofitted by checking the motherboard.

With a little research, you will find numerous tips online that let you manipulate the Windows 10 registry so that Windows 11 can be installed. If you don't have a TPM on the PC or you do not want to use Secure Boot, the preferred approach is to install from a Windows 11 image that already has verification disabled. The easiest way to do this is with the free Rufus [2] tool.

After booting, simply specify the Windows 11 ISO file in *Boot selection*. Microsoft provides ISO files free of charge [3]. After loading the ISO file, you will see the *Extended Windows 11 Installation (no TPM/no Secure Boot)* option in the Device Properties section. After selecting the option and the correct USB stick in *Device*, you can create an installation disk that does not check for the TPM chip or enable Secure Boot.

Keeping and Resetting Windows 11

If you experience serious problems in Windows 11, but you do not want to go back to version 10, you can reset your Windows 11 installation; however, you would then have no way to get back to Windows 10. You need to take this into account when repairing or uninstalling Windows 11, and you have several ways to reset version 11 if the operating system stops working. If Windows 11 no longer boots, you can even recover with the computer repair options, which are automatically launched if the operating system fails to boot properly three or four times. You can also reset the operating system on an active Windows 11, and you can delete any personal files on the computer, if so desired. After the reset, Windows 11 is in as-delivered state (i.e., basically a new Windows 11 installation).

To reset Windows 11 on the fly, go to *Settings | System | Recovery* (Figure 1). *Go back* lets you switch from Windows 11 back to Windows 10, but more on that later. The *Reset PC* button starts a wizard that manages the

reset. In the next step, select whether you want Windows 11 to be restored to its factory default state (*Remove everything*) or whether you want to keep installed programs, data, and settings (*Keep my files*). If Windows 11 is no longer working properly, selecting *Remove everything* will do more and probably be more successful – assuming you have a full data backup.

During the process, Windows 11 asks for the installation files. The *Cloud download* option lets you download them directly from Microsoft online. Bear in mind that the download will be at least 4GB, so you will need enough disk space on the PC. Alternatively, you can use local files with the *Local reinstall* option.

Before the wizard starts resetting, a summary is displayed, and you can change your selection at this point. In the next step, Windows informs you that the update installation cannot be undone. If the computer was updated from Windows 10 to Windows 11, there is no way back to the Windows 10 version after the reset. It may make sense to revert to Windows 10 here instead of resetting Windows 11 and blocking your path to Windows 10,

especially if your existing problems are the result of upgrading to Windows 11.

If you do not decide on a complete reset but want to keep files and apps, you can check in the settings which apps you will still need to reinstall. Ideally, you will have matching installation files available. It is also a good idea to export your settings before the reset, if possible. Once you have answered all of the wizard's questions, it will begin its work and reset Windows 11. To do this, press the *Reset* button. From this moment on, the process can no longer be stopped.

If Windows 11 Fails to Boot

If Windows 11 no longer boots, you can reset from the computer repair options. After the third or fourth failed boot, Windows 11 launches the Automatic Repair feature. *Advanced Startup* provides various tools for recovery, and you can revert to Windows 10 here, as well.

Besides this, you will find various tools for Windows 11 recovery. To reset Windows 11, select *Troubleshoot | Reset this PC*. Many users confuse this option

with uninstalling Windows 11, especially if you upgraded the computer from Windows 10 to 11. Additional recovery functions are available under *Advanced options*. The *Uninstall updates* item lets you switch back from Windows 11 to Windows 10. These choices are a bit confusing but important, because if you make the wrong selection, you can't undo the mistake. If system restore points exist, they can be enabled by selecting *Restore system* and then be used for recovery. This option is also possible with the computer repair options or with *rstrui.exe* in Windows. If you select *Reset this PC*, the same options are available as for resetting Windows 11 directly in Settings, and you can download directly from the cloud if necessary. Of course, the computer needs to be connected to the Internet to continue. If you do not want to continue using your computer, select the *Fully clean the drive* option in the computer repair options when you reset. Windows 11 will then delete the files completely. After successfully downloading Windows 11 and resetting the operating system, Windows 11 will restart in its as-delivered state. If you had any viruses or ransomware, they are also deleted – as long as they are not hiding in the hard drive firmware or in the BIOS.

Reverting to Windows 10

If you are unhappy with the update to Windows 11, you can undo the update with the on-board tools and restore Windows 10 with the same tools that were used for resetting Windows 11 to the factory defaults. When upgrading to Windows 11, the operating system saves the old operating system installation for 10 days. No data is lost, and the operating system settings are also retained. Of course, it is still a good idea to back up your computer before uninstalling. However, don't wait too long to roll back to Windows 10, because Windows 11 deletes the uninstall data from the system after 10 days. Even if you manually remove the associated temporary files with the Disk Cleanup utility, the way back is still blocked.

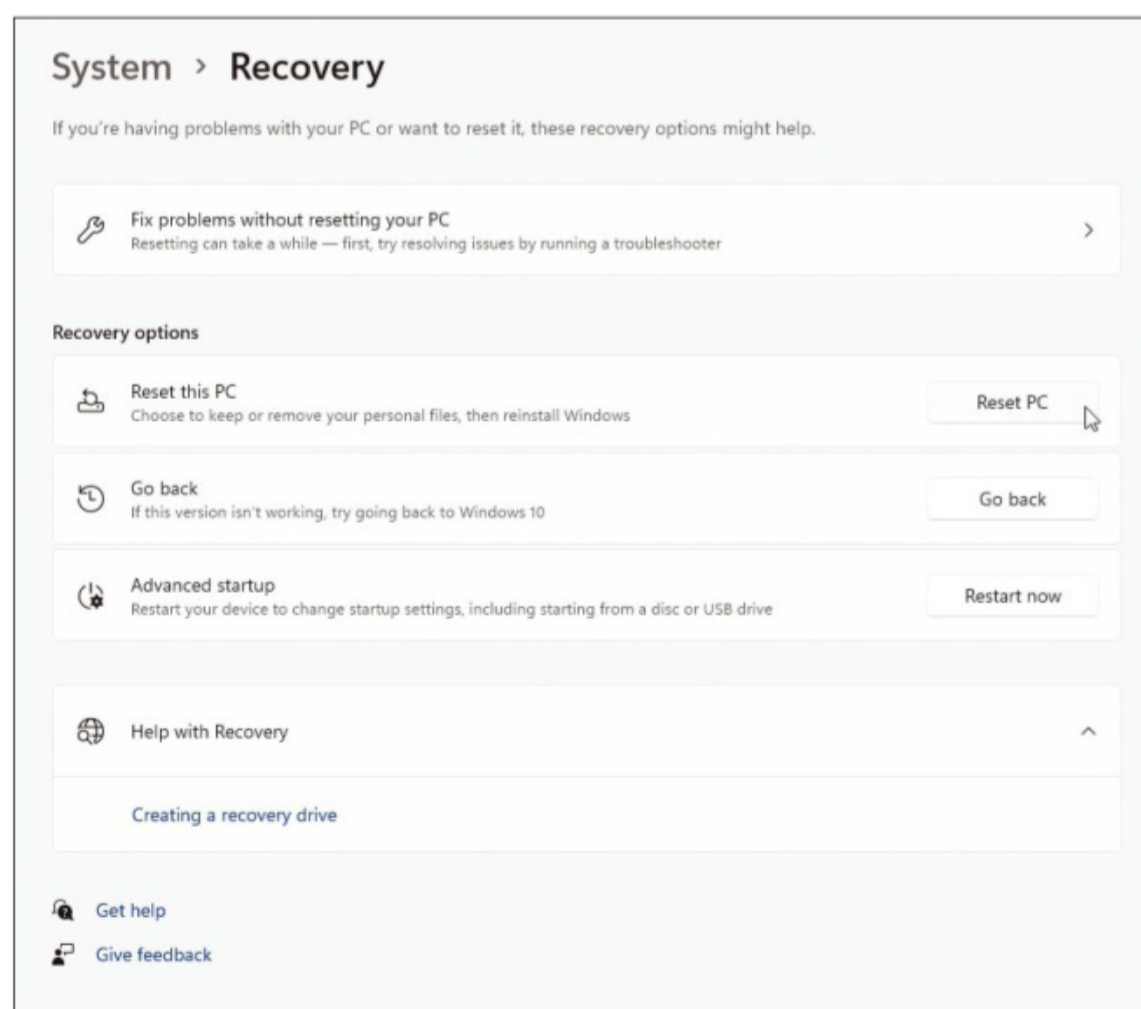


Figure 1: If you have problems with the operating system, you can fix them in the system settings, up to and including completely reinstalling Windows .

Only take this path if you are sure that you want to continue using the new operating system.

Uninstalling Windows 11

You can trigger a Windows 11 uninstall directly at the operating system level. To do this, open the Windows Settings. Microsoft has revised the look and layout of the menu items compared with Windows 10. As a general rule, downgrading from Windows 11 to Windows 10 still works the same way as rolling back from Windows 10 to Windows 7 or 8.1 back in the day. To restore

Windows 10, go to *System | Recovery*, which includes various options for repairing or uninstalling Windows 11. The *Go back* button removes Windows 11 from the computer. A wizard launches to guide you through the uninstall process. Of course, Microsoft wants its customers to stick with Windows 11, so it throws up a screen to find out why you're going back to Windows 10 (**Figure 2**). This menu item is only available if the system files for uninstalling Windows 11 have not yet been removed. If you do not want to give a specific reason, simply select *For another reason* and click the *Next* button to

continue to the *Check for updates?* page.

Select *No, thanks* if you want the wizard to continue the Windows 11 uninstall. In the next step, the wizard tells you what the effect of uninstalling Windows 11 could be. The settings you made directly in Windows 11 will be gone after the uninstall (**Figure 3**). The operating system then reverts to the state that existed before the update to Windows 11.

Next tells the wizard to go to the next page. By the way, if you changed your password in Windows 11, you cannot continue to use the new password after uninstalling. The old password is required to log in to Windows 10. Up to this point

you can terminate the wizard by pressing *Cancel*, and no changes will be made. After pressing the *Go back to Windows 10* button, the process can no longer be undone. After a few minutes, Windows 10 reboots. Once you have successfully removed Windows 11 from your computer, the first step should be to apply the latest patches for Windows 10 with Windows Update. The wizard also subsequently displays updates for Windows 10 and lets you install them, including the update to Windows 10 21H2 and newer. Of course, you should also check to see whether Windows 10 is still enabled by calling `slui.exe`. Last but not least, take a look at the installed applications, drivers, and your data.

Conclusions

If you upgraded your computer from Windows 10 to Windows 11 and the new operating system doesn't work the way you thought it would, you have several ways to fix the system. You can restore Windows 11 to its factory default state or even uninstall the new operating system completely and return to Windows 10. Microsoft has added individual options in the Windows 11 Settings app, and in the computer repair options, which you can launch manually or that launch automatically in case of boot problems. However, the time window for a rollback is limited: If you pass the rollback window, even if you remove the update files manually, your operating system will still be Windows 11. ■

Info

- [1] WhyNotWin11: [\[https://github.com/rcmaehl/WhyNotWin11\]](https://github.com/rcmaehl/WhyNotWin11)
- [2] Rufus: [\[https://rufus.ie/en/\]](https://rufus.ie/en/)
- [3] Windows 11 ISO: [\[https://www.microsoft.com/en-us/software-download/windows11\]](https://www.microsoft.com/en-us/software-download/windows11)

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).

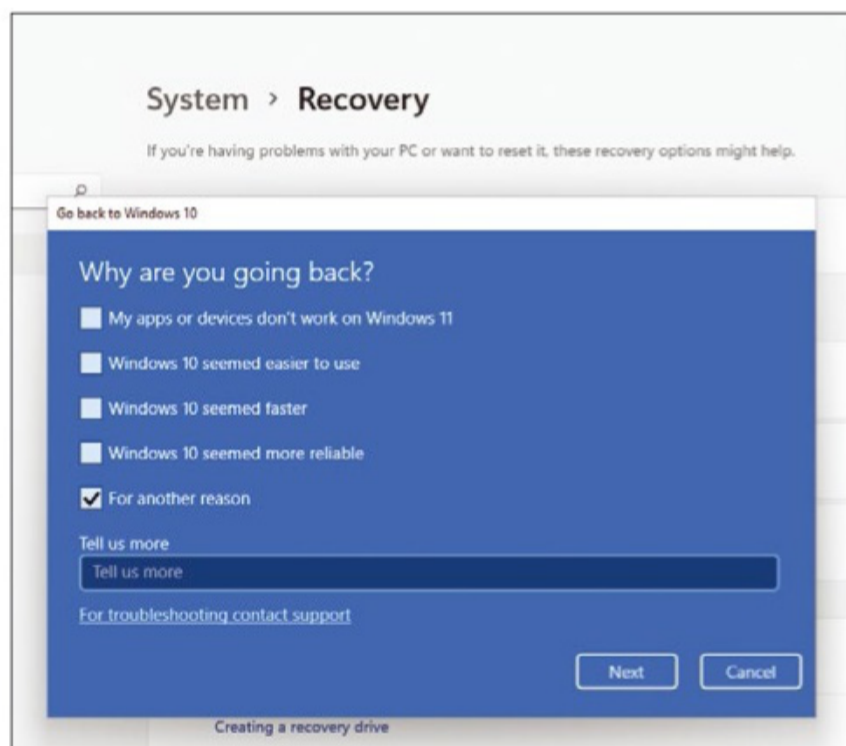


Figure 2: Microsoft wants to know what users didn't like about Windows 11.

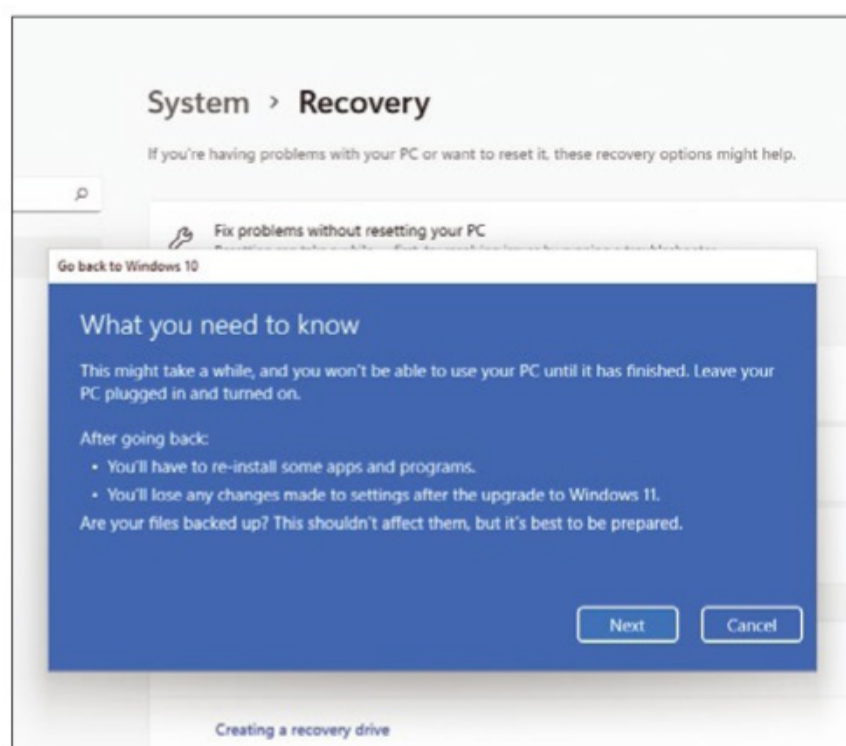


Figure 3: Be sure you know what you're getting before clicking *Next*.

Apache OpenMeetings video conferencing platform

Closed Society

The free video conferencing platform has comprehensive collaboration tools with instant messaging, whiteboards, screen sharing, and team features in document processing - and it can be hosted locally so sensitive corporate data is not exposed to cloud services. By Thomas Joos

The Apache OpenMeetings (AOM) video conferencing platform [1] does not require special software on clients for access; all of its features are available in the browser. User management is handled by an integrated system in which you create users and add external email addresses on request. However, you can also connect to Lightweight Directory Access Protocol (LDAP) systems such as Active Directory. Starting in version 6, Apache OpenMeetings also supports TLS 1.2 for OAuth and do-it-yourself captchas.

AOM lets you schedule meetings, create surveys, and back up data easily. The system is ideal for providing home office users a simple video conferencing system that can be set up quickly on their own hardware. Operation in virtual machines (VMs) or as a container solution is also quite easy to implement.

Users invite colleagues to attend meetings directly in the web interface. They then receive notifications to the specified email address or directly in the web browser during OpenMeetings sessions. Users can maintain their profiles, including photos, on the dashboard and change their passwords at any time. As a

result, AOM also provides a good basis for teams that want to maintain their own user data.

OpenMeetings can also be accessed by APIs that support streaming media servers, such as Red5 and Kurento (which forms the basis of AOM). You also can use the AOM PHP client [2] via the REST API. AOM supports the Simple Object Access Protocol (SOAP) and Representational State Transfer (REST) APIs as well as OAuth2. Voice over IP (VoIP) and the Session Initiation Protocol (SIP) can also be used. Three demo servers [3-5] are available for an initial look at Apache OpenMeetings. After logging in, you can familiarize yourself with the AOM functions and start using the system. The installation is relatively simple but takes more than an hour to complete.

Installing Apache OpenMeetings

You can install on Windows, Linux, and macOS systems, and there's even a Docker container [6]. Unlike Zoom or Microsoft Teams, Apache OpenMeetings is not a ready-made platform. IT managers need to provide

the hardware or platform on which it runs, which can mean servers in your local data center or in the cloud. The Kurento Media Server [7] is required in the first step of the installation. Implementation then takes place on this server with the current Apache OpenMeetings installation files. Detailed installation instructions can be found online [8]. AOM initially relies on the integrated H2 database for data storage, but the developers recommend an external database server for production. AOM is very flexible and supports MySQL/MariaDB, PostgreSQL, IBM DB2, Oracle, and Microsoft SQL Server. Additional tools are also required for certain functions:

- ImageMagick [9] for uploading documents to whiteboards
- Ghostscript [10] for integrating PDF files into meetings
- An OpenOffice or Libre Office installation for various document formats, team functions, and uploading Microsoft Office files
- FFmpeg [11] and Sound eXchange (SoX) [12] for video files

For a sample installation of OpenMeetings on a recent Ubuntu system (version 21.04), first install the various dependencies:

```
sudo apt update
sudo apt upgrade
sudo apt install openjdk-11-jdk
openjdk-11-jdk-headless nano
sudo update-alternatives --config java
```

Next, install LibreOffice, ImageMagick, FFmpeg, VLC, Curl, and SoX:

```
sudo add-apt-repository
ppa: libreoffice/ppa
sudo apt update
sudo apt install libreoffice
sudo apt install
-y imagemagick libjpeg62 zlib1g-dev
sudo apt install sox
sudo apt install ffmpeg vlc curl
```

For production operation, I opted for MariaDB:

```
sudo apt install mariadb-server
sudo /etc/init.d/mariadb start
sudo mysqladmin
-u root password <New Password>
sudo mysql -u root -p
```

Now you can install AOM (Listing 1). The last two commands open the connector between MariaDB and AOM. For further setup steps, refer to the online tutorial [13]. If you install a different AOM version, you will need to adjust the version numbers to match.

Firewall Rules and Encryption

Ports 5443 for HTTPS and 5080 for HTTP are available to access the web interface. HTTP access is primarily for post-installation setup at http:// <IP Address> :5050/

openmeetings. The setup includes configuring your choice of database and creating an admin user. A wizard then guides you through the next steps. In production environments, make sure that only HTTPS access is possible. The ports can be adjusted after installation in the \$OM_HOME/conf/server.xml configuration file. For connections via SSL, see the OpenMeetings HTTPS page [14]. After connecting to the client, you can use the Network testing feature on the dashboard (Figure 1) to check whether all the settings are correct and the various AOM components are working.

Listing 1: Installing AOM

```
cd /opt
sudo wget https://archive.apache.org/dist/openmeetings/6.2.0/bin/apache-openmeetings-6.2.0.tar.gz
sudo tar xzvf apache-openmeetings-6.2.0.tar.gz
sudo mv apache-openmeetings-6.2.0 open620
sudo chown -R nobody:nogroup /opt/open620
sudo wget https://repo1.maven.org/maven2/mysql/mysql-connector-java/ 8.0.26/mysql-connector-java-8.0.26.jar
sudo cp /opt/mysql-connector-java-8.0.26.jar /opt/open620/webapps/ openmeetings/WEB-INF/lib
```

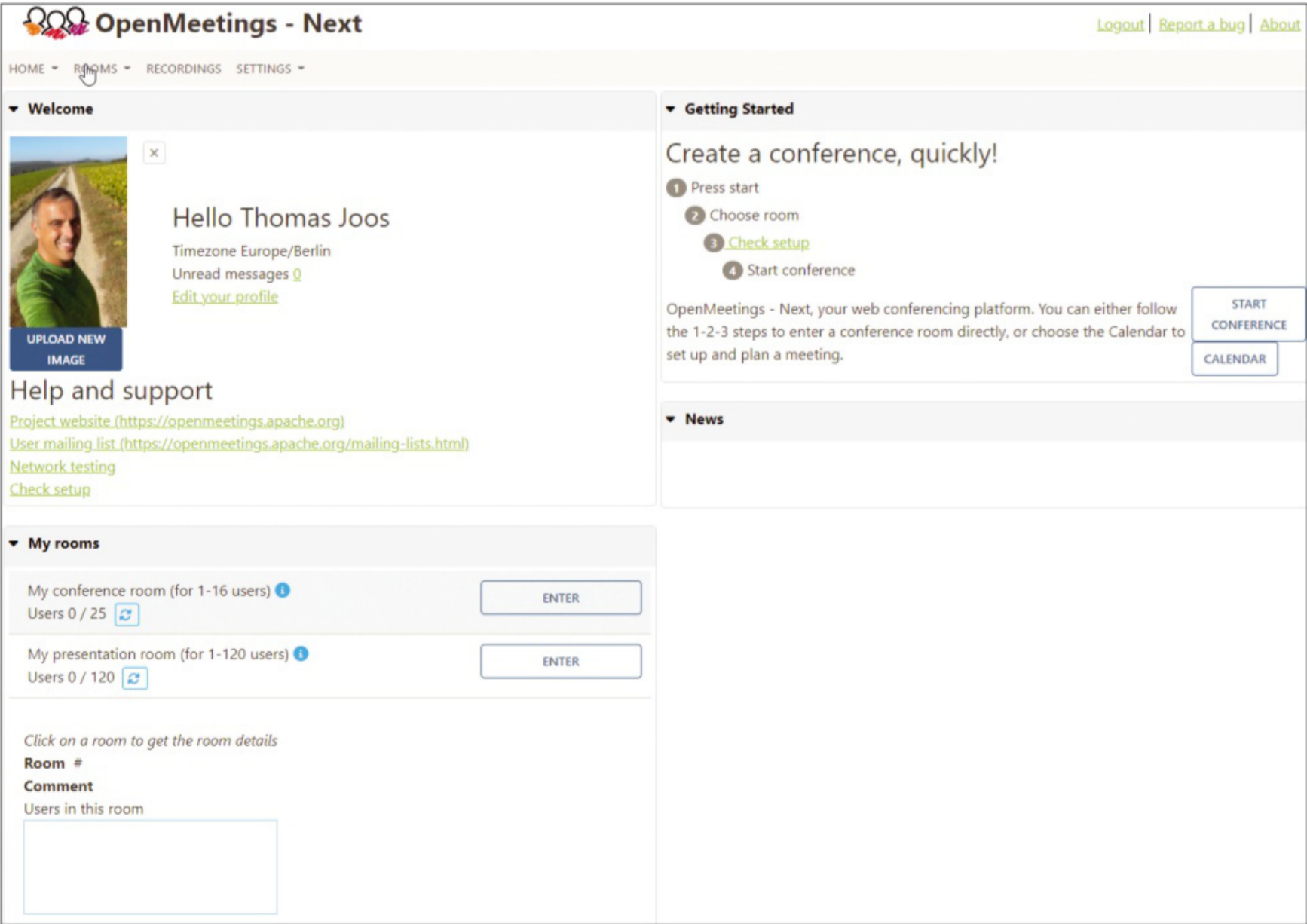


Figure 1: Apache OpenMeetings is accessed and managed in a web interface.

Permissions and Importing Data

Permission structures in AOM include delegating meeting moderation and granting permissions to use the whiteboard and share the screen. The video and audio streams can be turned on and off, as well. You can either rely on the internal user management system or use a connected LDAP structure (e.g., Active Directory).

Multiple whiteboards can be created during meetings, and their content can be downloaded as a file. Conversely, you can also import files into whiteboards to make them available to users (**Figure 2**). Supported formats include PDF, DOC, ODT, PPT, and others. Documents in individual folders in the conference room can easily be used in whiteboards.

AOM in Large Environments

Apache OpenMeetings integrates with LDAP environments with Apache Directory Server or Active Directory. If you are running AOM on a Linux server, connecting to Active Directory on Windows Server 2019 or 2022 is a straightforward task. It does not matter where Active Directory is installed. The only important thing is that the AOM server can communicate with the domain controllers.

In principle, you can also run the domain controllers and AOM in the cloud (e.g., as an EC2 instance in AWS). The connection between the servers is by way of the internal network in AWS.

Creating Video Conferences

AOM initially focuses on audio and video conferencing. Participants can

send audio and video data and configure the transmission quality where needed. You can also record meetings, which are then available as MP4 or AVI/FLV files. Starting in version 6 you can disable the ability to record meetings, as well.

Once the system is operational, you can log in through the AOM dashboard where all the features needed for a meeting reside. You can switch directly to a meeting from *Start conference* or schedule a meeting from *Home | Calendar* by clicking on a date (**Figure 3**).

Several tabs are helpful: *General* lets you specify when the meeting will take place and who you want to invite. To select the conference room, use the *Room* tab; to manage the available rooms use the *Rooms* tab in the dashboard. *Advanced* lets you create links for access to a meeting, which you can send to the participants. The

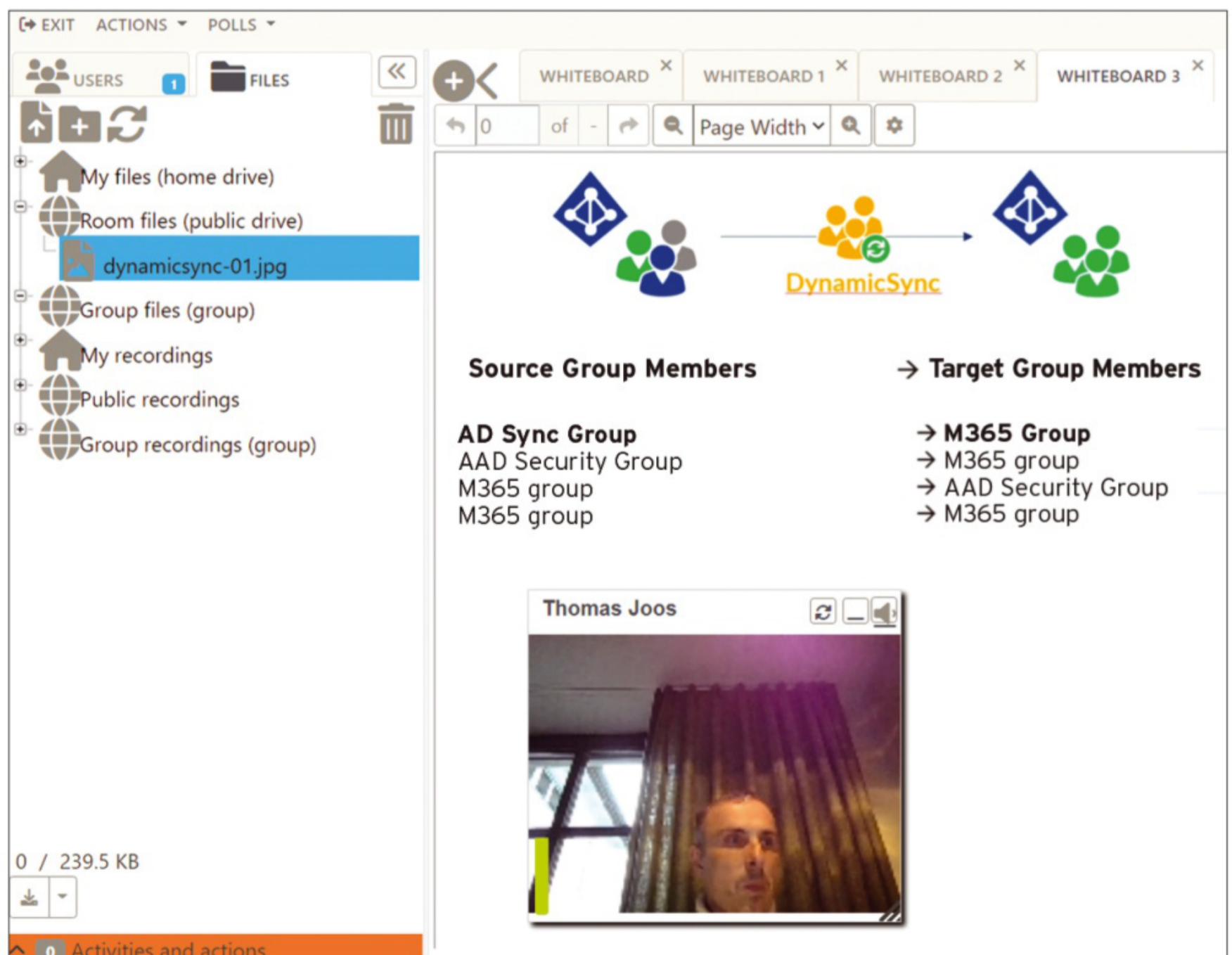


Figure 2: Separate folder structures can be established in the conference rooms.

meeting is then saved in your and the participants' calendars. Users can also connect different meeting rooms. To this end, they can be assigned to individual users or groups. AOM offers complementary folder structures that can be made available in meetings. The structure setup is flexible and includes both public and private drives. Public drives can be integrated directly with a virtual meeting room and are visible to all participants there. In the client settings, you can enable or disable the webcam in the web interface from *Check setup* on the dashboard. The microphone and the webcam resolution can also be tested and adjusted at this point. The *Start recording test* button lets participants check the transmission quality by creating a recording.

Online Meetings

To schedule online meetings, click *Settings | Search Users*. Here, you can see all the users in the environment. From the *Actions* column you can add users to your private contacts or write private messages. At this point, you also have the option of viewing profile information and inviting people to meetings. Each user can define their own profile on the dashboard.

Although you can use the AOM calendar to schedule meetings, using Cal-DAV or other external calendars is also an option (including Google Calendar). All you need is the Calendar ID and the Calendar API key. The setup is deliberately kept simple and is quickly done. During the meeting, the individual windows can be moved around in the browser and arranged to suit your needs. The files for a meeting can be found in the meeting room under the Files menu item. The buttons in this area can also be used to create new directories and upload files to the folders. The files are then available to the meeting participants. Files located in the directories can be dragged and dropped into whiteboards by meeting participants. Whiteboards can be renamed, allowing participants to open different files or take notes on different whiteboards and easily switch between notes. The different whiteboards show up as tabs. Additionally, messages can be sent between participants. On each user's dashboard, unread messages appear on the Welcome page. Clicking on the message counter opens the OpenMeetings email section, where the *New mail* button can be found. However, only those who exist in AOM are eligible recipients. Writing messages directly to individual contacts is also

possible in contacts management, which can be found in *Settings*. To do this, AOM uses the email address stored in the user's profile. When managing the server, it is important to configure how email is sent up front.

Conclusions

Apache OpenMeetings is a free and relatively quickly deployed online meeting tool that organizations can run on their own hardware, on VMs, or in the cloud. You have complete control over all your data, and you use your own user management system. In larger environments, integration with Active Directory or other LDAP systems is possible. AOM is a scalable system that is very much worth taking a further look at – especially given that many comparable systems are not compliant with the General Data Protection Regulation (GDPR) in the European Union.

- Info
- [1] Apache OpenMeetings: [\[https://openmeetings.apache.org\]](https://openmeetings.apache.org)
 - [2] PHP client: [\[https://github.com/om-hosting/openmeetings-php-client\]](https://github.com/om-hosting/openmeetings-php-client)
 - [3] AOM test server 1: [\[https://om.alteametasoft.com/openmeetings\]](https://om.alteametasoft.com/openmeetings)
 - [4] AOM test server 2: [\[https://demo-openmeetings.apache.org/openmeetings\]](https://demo-openmeetings.apache.org/openmeetings)
 - [5] AOM test server 3: [\[https://om.alteametasoft.com:8443/next\]](https://om.alteametasoft.com:8443/next)
 - [6] Apache OpenMeetings for Docker: [\[https://github.com/openmeetings/openmeetings-docker\]](https://github.com/openmeetings/openmeetings-docker)
 - [7] Kurento: [\[https://www.kurento.org\]](https://www.kurento.org)
 - [8] Download and installation: [\[https://openmeetings.apache.org/downloads.html\]](https://openmeetings.apache.org/downloads.html)
 - [9] ImageMagick: [\[https://imagemagick.org\]](https://imagemagick.org)
 - [10] Ghostscript: [\[https://www.ghostscript.com\]](https://www.ghostscript.com)
 - [11] FFmpeg: [\[https://ffmpeg.org\]](https://ffmpeg.org)
 - [12] Sound eXchange (SoX): [\[http://sox.sourceforge.net\]](http://sox.sourceforge.net)
 - [13] Apache OpenMeetings installation tutorial: [\[https://cwiki.apache.org/confluence/display/OPENMEETINGS/Tutorials+for+installing+OpenMeetings+and+Tools\]](https://cwiki.apache.org/confluence/display/OPENMEETINGS/Tutorials+for+installing+OpenMeetings+and+Tools)
 - [14] Apache OpenMeetings with HTTPS: [\[https://openmeetings.apache.org/HTTPS.html\]](https://openmeetings.apache.org/HTTPS.html)

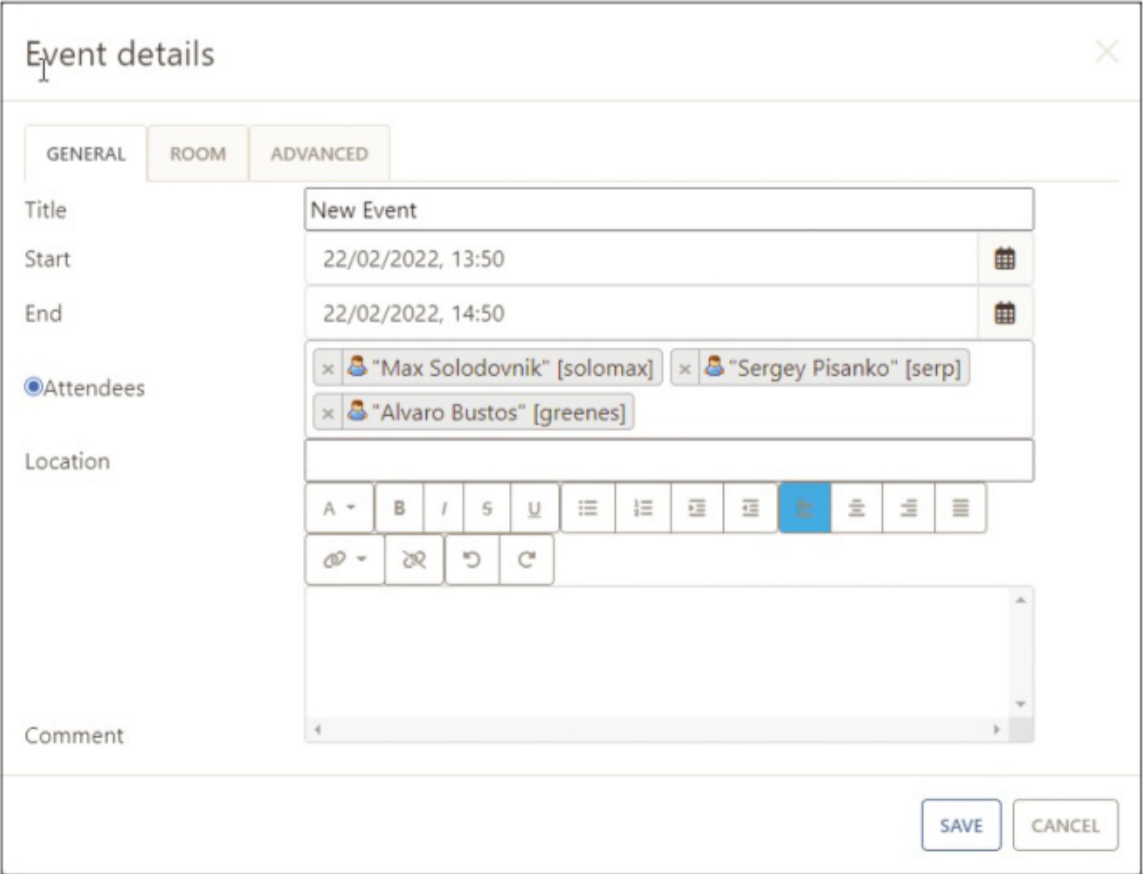


Figure 3: Scheduling a meeting takes just a few steps.



Goodbye virtual machines, hello microVMs

Small Scale

You can have your cake and eat it, too: MicroVMs feature the strong isolation of virtual machines and lightweight behaviors of containers. By Ankur Kumar

In two previous articles [1][2], I introduced how to bring up lightweight container machines with Footloose and cloud virtual machine stacks on a laptop with Multipass. The Footloose approach didn't require a lot of computing resources or any hardware-based virtualization support. On the other side, Multipass required built-in virtualization capability within the processor and was limited to Ubuntu virtual machines (VMs). Security is a downside of containers because they provide thinner isolation compared with the strong isolation of virtual machines. Can you have the strong isolation of VMs with the lightweight and fast bring up/bring down behavior of containers to create a cloud VM kind of stack on a laptop? Can you also have the tooling that provides a declarative automation experience? The answer is yes!

The tooling provided in this article is for newer machines with virtualization capability provided by the processor. Still, the container machine approach, presented in the first part of this series of articles about Footloose, is an option, in case you don't

have such hardware available. I tested the snippets shown or referenced in this article on my 8GB quad CORE i7 laptop running Ubuntu 18.04 LTS.

Firecracker and Ignite

Firecracker [3] is an open source virtual machine monitor created by Amazon Web Services (AWS) to accelerate its serverless offerings (e.g., AWS Lambda and AWS Fargate). It uses a Linux-kernel-based virtual machine to create lightweight VMs known as microVMs. These microVMs eliminate unnecessary devices and guest functionality to reduce resource footprints. The result is enhanced security over traditional VMs, plus the efficiency of containers. MicroVMs, then, provide enhanced isolation, reduced startup time, and greater hardware utilization. Interestingly, the AWS folks demo'd starting 4,000 microVMs on an Elastic Compute Cloud (EC2) bare metal instance showcasing the capabilities of Firecracker [4].

Ignite is an open source virtual machine manager that combines Firecracker with Docker and Open Container Initiative (OCI) images

to create a great developer experience out of the box. It can manage microVMs declaratively in a GitOps (Git + continuous deployment tools) fashion.

Ignite Quickstart

To begin with microVMs, you need to install the required dependencies to make use of Ignite. On my Ubuntu laptop (please refer to the install link provided in the Info section [5] for other distributions), I used:

```
sudo apt-get update && ❏
sudo apt-get install -y ❏
--no-install-recommends dmsetup ❏
openssh-client git binutils which ❏
containerd || sudo apt-get install -y ❏
--no-install-recommends containerd
```

The ignite command is provided as a static binary, so just download and make it executable, and it's ready to use. To set up Ignite use:

```
sudo curl ❏
--sSLk "https://github.com/$(❏
curl --sSLk https://github.com/❏
weaveworks/ignite/releases | ❏
```

```
Usage:
  ignite [command]

Available Commands:
  attach      Attach to a running VM
  completion  Output bash completion for ignite to stdout
  cp          Copy files/folders between a running vm and the local filesystem
  create      Create a new VM without starting it
  exec        execute a command in a running VM
  help        Help about any command
  image       Manage base images for VMs
  inspect     Inspect an Ignite Object
  kernel      Manage VM kernels
  kill        Kill running VMs
  logs        Get the logs for a running VM
  ps          List running VMs
  rm          Remove VMs
  rmi         Remove VM base images
  rmk         Remove kernels
  run         Create a new VM and start it
  ssh         SSH into a running vm
  start       Start a VM
  stop        Stop running VMs
  version     Print the version of ignite
  vm          Manage VMs

Flags:
  -h, --help                help for ignite
  --ignite-config string     Ignite configuration path; refer to the 'Ignite Configuration' docs for more details
  --log-level loglevel       Specify the loglevel for the program (default info)
  -q, --quiet                The quiet mode allows for machine-parsable output by printing only IDs
```

Figure 1: Sub-commands and flags for the ignite command.

```
grep download|grep amd64 |
awk -F ' ' '{print $2}' | head -1)"
-o /usr/local/bin/ignite &&
sudo chmod +x /usr/local/bin/ignite
```

Now executing ignite should dump its help screen as shown in Figure 1. Please note that, currently, any Ignite operation shown requires sudo privileges. If you have ever used the Docker CLI (command-line interface), then you will be at home with the ignite commands right away.

Ignite requires two kinds of OCI images to create microVMs: The first is for the desired Linux kernel version, and the second is to create the respective userspace filesystem with an init system installed for various GNU/Linux distributions. Currently, multiple versions of the kernel are available: Amazon Linux 2, Alpine, CentOS 7/8, Ubuntu {16,18,20}.04, and other base images are provided by the Ignite GitHub project [5].

To create and run a microVM with an Alpine base image, execute

```
sudo ignite run weaveworks/ignite-alpine
-n myalpinemvm -i --ssh
```

The run command combines multiple actions: It imports the Alpine image,

creates the microVM, runs the microVM, and attaches the TTY device (SSHs into the VM’s shell). You’ll see prompts to enter the user login/password (enter *root/root*); voilà, you’re in the running microVM. Now feel free to manipulate your microVM any way you like.

Figure 2 shows the Ignite run sequence on my laptop. The run

command takes some defaults for the CPU, memory, disk, kernel image, and so on. To see these defaults and command-line switches, enter:

```
ignite run -h
```

You will see for yourself how quickly a microVM comes up compared with a VM.

```
* Loading modules ...
[ ok ]
* Mounting local filesystems ...
[ ok ]
* Mounting misc binary format filesystem ... [ ok ]

Welcome to Alpine Linux 3.8
Kernel 5.10.51 on an x86_64 (ttyS0)

5e46d6bfadc07cfd login: root
Password:
Welcome to Alpine!

The Alpine Wiki contains a large amount of how-to guides and general
information about administrating Alpine systems.
See <http://wiki.alpinelinux.org>.

You can setup the system with the command: setup-alpine

You may change this message by editing /etc/motd.

login[1052]: root login on 'ttyS0'
5e46d6bfadc07cfd:~# nproc;echo;free -m;echo;df -h /
1

              total        used        free      shared  buffers   cached
Mem:           479          24         455           0          0          7
-/+ buffers/cache:          16         463
Swap:            0           0           0

Filesystem      Size      Used Available Use% Mounted on
/dev/root       3.8G    104.1M      3.5G   3% /
```

Figure 2: Ignite run in action.

```
5e46d6bfadc07cfd:~# reboot
PID1: Received "reboot" from FIFO...
5e46d6bfadc07cfd:~# Starting reboot runlevel
* Stopping sshd ... [ ok ]
* Shutting down ssh connections ...
* sshd: caught SIGTERM, aborting [ ok ]
* Deleting static routes ... [ ok ]
* Unmounting loop devices
* Unmounting filesystems
* Stopping udev ... [ ok ]
Sending the final term signal
Sending the final kill signal
[ 117.172439] reboot: Restarting system
[ 117.173393] reboot: machine restart
INFO[0118] firecracker exited: status=0
```

Figure 3: Terminating an interactive microVM.

Listing 1: footloose.yaml

```
cluster:
  name: cluster
  privateKey: cluster-key
machines:
- count: 1
  spec:
    image: weaveworks/ignite-ubuntu:20.04
    name: ubuntu2004%d
    hostname: ubuntu2004%d
    backend: ignite
    ignite:
      cpus: 2
      memory: 2G
      diskSize: 20G
      kernel: "weaveworks/ignite-kernel:5.13.3"
- count: 1
  spec:
    image: weaveworks/ignite-centos:8
    name: centos8%d
    hostname: centos8%d
    backend: ignite
    ignite:
      cpus: 2
      memory: 2G
      diskSize: 20G
      kernel: "weaveworks/ignite-kernel:5.13.3"
```

The correct way to terminate an interactive microVM (-i option) session is with the reboot command (Figure 3). To clean up the stopped microVM, use

```
sudo ignite rm <microvm name>
```

Append the -f flag to clean up a running microVM.

Listing 2: Customized Configuration

```
apiVersion: ignite.weave.works/v1alpha4
kind: VM
metadata:
  name: myubuntumvm
spec:
  image:
    oci: weaveworks/ignite-ubuntu
  kernel:
    oci: weaveworks/ignite-kernel:5.13.3
  cpus: 2
  diskSize: 2GB
  memory: 20GB
  ssh: <absolute path to ssh public key on your laptop>
```

I also passed the --ssh option to the run command, which creates an SSH key pair and imports the public key into the created microVM. (As another option, you could pass in your SSH public key pair, instead.) To log in to the running microVM and dump info about all microVMs created by Ignite, use the following commands:

```
sudo ignite ssh <microvm name>
sudo ignite ps -a
```

(Drop the -a if you are interested only in the currently running microVMs.) These examples are the main Ignite commands that accomplish most of the general tasks related to microVMs. Feel free to explore more about any Ignite command by suffixing the -h flag to dump the help screen.

Declarative Ignite

Now you can manipulate Ignite in a declarative manner with the use of some prior knowledge gained in the first article of this series with Footloose for automation. To begin, grab the Footloose binary from the GitHub project site [6], if you haven't set it up already, and create the YAML file shown in Listing 1 in the directory from which you will launch Footloose.

As you can observe, the back end is now Ignite, and an ignite section declares CPU, memory, disk size, and kernel properties for the respective microVM(s). Now, executing

```
sudo footloose create
sudo footloose delete
```

brings up the declared microVM stack along with a new SSH key pair in the current working directory and cleans up the created stack. Could it be simpler? You could dump the running microVM info with sudo ignite ps because sudo footloose show doesn't dump microVM IP info, in case you are looking forward to using the created SSH private key to log manually into a running microVM.

Ignite itself takes a different form of YAML configuration if you don't want to use Footloose to bring up the microVM stack declaratively. A minimal Ignite API object to create a microVM with default properties would be:

```
apiVersion: ignite.weave.works/v1alpha4
kind: VM
metadata:
  name: myubuntumvm
spec:
  image:
    oci: weaveworks/ignite-ubuntu
  ssh: true
```

You could further customize your microVMs with more resources (Listing 2).

The Ignite API object has so many other properties you can use to customize your microVMs. Each of the properties is documented online in the official Ignite documentation [7]

Listing 3: footloose.cfg

#	Name	Count	Image	Kernel	Cpu	Memory	Disk	Ports([hostPort:]containerPort)
ubuntu1804	1	1	weaveworks/ignite-ubuntu:18.04	weaveworks/ignite-kernel:5.13.3	2	2G	20G	22,38080,52812,58080
centos8	1	1	weaveworks/ignite-centos:8	weaveworks/ignite-kernel:5.13.3	2	2G	20G	22,38080,52812,58080
kafka	3	3	weaveworks/ignite-ubuntu:20.04	weaveworks/ignite-kernel:5.12.18	2	2G	20G	22,2181,8080,9092,38080,52812,58080
spark	3	3	weaveworks/ignite-ubuntu:20.04	weaveworks/ignite-kernel:5.12.18	2	2G	20G	22,7077,8080,8081,38080,52812,58080
consul	3	3	weaveworks/ignite-ubuntu:20.04	weaveworks/ignite-kernel:5.13.3	1	2G	20G	22,52812,58080,58500
hashiui	1	1	weaveworks/ignite-ubuntu:20.04	weaveworks/ignite-kernel:5.13.3	1	2G	20G	22,52812,53000,58080

```
matrix@ankur-Inspiron-5593:~/Development/Git/Works/devops/CloudInABox/MicroVMs/scripts$ ./create_micro_vms_stack.sh
Usage: create_micro_vms_stack.sh
< lint - run static analysis on Dockerfiles and Shellscripts |
  create - create microvm stack |
  start - start microvm stack |
  stop - stop microvm stack |
  show - dump info about the created microvm stack |
  test - run specified ansible role to configure the stack,
        valid roles are (ping is default if nothing mentioned):
        [[ping]|goss|consulserver|consulclient|
        consulesm|hashiui|consultemplate|docker|
        cassandra|elasticsearch|kafka|spark|
        monitoror|vigil] |
  delete - delete everything created >
```

Figure 4: Wrapper utility help screen for microVMs.

with examples. You could bring up your microVM stack by providing the YAML declarative configuration name to the `sudo ignite run --config` command. Similar is the requirement for the `sudo ignite rm -f --config` command to clean up your running microVM stack. You also need to provide your local SSH key pair private key path to the `sudo ignite ssh` command (with the `-i` flag) if you create the microVM with your public key.

Advanced Tooling with Ignite

I extended the tooling created for Footloose described in the first article to create, show, configure, and clean up microVM stacks. To get started with the wrapper [8], just download the sources with

```
git clone https://github.com/2
richnusgeeks/devops.git
pushd CloudInABox/MicroVMs/scripts
```

You could use `cd` instead of `pushd`, but I prefer `pushd` for its intelligence. The wrapper script execution should show you a help screen (Figure 4). The top level wrapper script just takes a simple configuration (Listing 3). Simply execute

```
sudo ./create_micro_vms_stack.sh create
```

to bring up the defined microVM stack. The creation of a stack also triggers an Ansible ping to the spawned microVMs to ensure SSH connectivity. Figures 5 and 6 show some portions of the creation output of 16 microVM stacks on my laptop. The commands

```
sudo ./create_micro_vms_stack.sh test 2
<ansible role name, e.g., spark>
sudo /create_micro_vms_stack.sh delete
```

configure the created microVMs with

the necessary software and clean up the enacted stack (Figure 7).

Conclusion

AWS Firecracker is a breakthrough technology that provides the best from both the container and virtual machine worlds. Ignite further creates a user-friendly declarative GitOps mechanism that provides additional functionality over Firecracker to run OCI containers under strong isolation of hardware-assisted virtualization. The resulting

```
INFO[0000] Creating SSH key: cluster-key ...
INFO[0000] Docker Image: weaveworks/ignite-ubuntu:20.04 present locally
INFO[0000] Docker Image: weaveworks/ignite-ubuntu:18.04 present locally
INFO[0001] Docker Image: weaveworks/ignite-centos:8 present locally
INFO[0001] Docker Image: weaveworks/ignite-ubuntu:20.04 present locally
INFO[0001] Docker Image: weaveworks/ignite-ubuntu:20.04 present locally
INFO[0001] Docker Image: weaveworks/ignite-ubuntu:20.04 present locally
INFO[0001] Docker Image: weaveworks/ignite-ubuntu:20.04 present locally
INFO[0001] Docker Image: weaveworks/ignite-ubuntu:20.04 present locally
INFO[0001] Docker Image: weaveworks/ignite-ubuntu:20.04 present locally
INFO[0001] Docker Image: weaveworks/ignite-ubuntu:20.04 present locally
INFO[0001] Creating machine: cluster-ubuntu20040 ...
INFO[0004] Creating machine: cluster-ubuntu18040 ...
INFO[0007] Creating machine: cluster-centos80 ...
INFO[0010] Creating machine: cluster-kafka0 ...
INFO[0014] Creating machine: cluster-kafka1 ...
INFO[0018] Creating machine: cluster-kafka2 ...
INFO[0022] Creating machine: cluster-spark0 ...
INFO[0025] Creating machine: cluster-spark1 ...
INFO[0028] Creating machine: cluster-spark2 ...
INFO[0033] Creating machine: cluster-monitoror0 ...
INFO[0036] Creating machine: cluster-vigil0 ...
INFO[0041] Creating machine: cluster-consul0 ...
INFO[0045] Creating machine: cluster-consul1 ...
INFO[0049] Creating machine: cluster-consul2 ...
INFO[0052] Creating machine: cluster-consulesm0 ...
INFO[0057] Creating machine: cluster-hashui0 ...
```

Figure 5: Starting microVMs stack.

VM ID	NAME	IMAGE	KERNEL	SIZE	CPUS	MEMORY	CREATED	STATUS	IPS	P
05d78ff6cc349d3f		weaveworks/ignite-ubuntu:20.04	weaveworks/ignite-kernel:5.13.3	20.0 GB	1	2.0 GB	22s ago	Up	21s	10.61
.0.60	cluster-vigil0									
15fedbe57824a328		weaveworks/ignite-ubuntu:20.04	weaveworks/ignite-kernel:5.13.3	20.0 GB	1	2.0 GB	10s ago	Up	9s	10.61
.0.63	cluster-consul2									
1be3423cce4d4114		weaveworks/ignite-ubuntu:20.04	weaveworks/ignite-kernel:5.13.3	20.0 GB	1	2.0 GB	25s ago	Up	24s	10.61
.0.59	cluster-monitoror0									
32d755889f4f8170		weaveworks/ignite-ubuntu:20.04	weaveworks/ignite-kernel:5.13.3	20.0 GB	1	2.0 GB	14s ago	Up	13s	10.61
.0.62	cluster-consul1									
38ff77392c522110		weaveworks/ignite-ubuntu:20.04	weaveworks/ignite-kernel:5.12.18	20.0 GB	2	2.0 GB	31s ago	Up	31s	10.61
.0.57	cluster-spark1									
3a90bb1f82ce9e16		weaveworks/ignite-ubuntu:20.04	weaveworks/ignite-kernel:5.12.18	20.0 GB	2	2.0 GB	28s ago	Up	28s	10.61
.0.58	cluster-spark2									
56dlaf2f442d6350		weaveworks/ignite-ubuntu:18.04	weaveworks/ignite-kernel:5.13.3	20.0 GB	2	2.0 GB	52s ago	Up	51s	10.61
.0.51	cluster-ubuntu18040									
6181142c528291d4		weaveworks/ignite-ubuntu:20.04	weaveworks/ignite-kernel:5.12.18	20.0 GB	2	2.0 GB	35s ago	Up	34s	10.61
.0.56	cluster-spark0									
6935713e592c0fed		weaveworks/ignite-ubuntu:20.04	weaveworks/ignite-kernel:5.13.3	20.0 GB	1	2.0 GB	7s ago	Up	6s	10.61
.0.64	cluster-consulesm0									
6ac0253ale5a9e53		weaveworks/ignite-ubuntu:20.04	weaveworks/ignite-kernel:5.13.3	20.0 GB	1	2.0 GB	18s ago	Up	17s	10.61
.0.61	cluster-consul0									
6f4a688191f26e16		weaveworks/ignite-ubuntu:20.04	weaveworks/ignite-kernel:5.13.3	20.0 GB	2	2.0 GB	55s ago	Up	54s	10.61
.0.50	cluster-ubuntu20040									
818c3afe92604cc7		weaveworks/ignite-ubuntu:20.04	weaveworks/ignite-kernel:5.12.18	20.0 GB	2	2.0 GB	46s ago	Up	45s	10.61
.0.53	cluster-kafka0									
9ee80c8f3838245a		weaveworks/ignite-centos:8	weaveworks/ignite-kernel:5.13.3	20.0 GB	2	2.0 GB	49s ago	Up	48s	10.61
.0.52	cluster-centos80									
a5f89d6fdee096e6		weaveworks/ignite-ubuntu:20.04	weaveworks/ignite-kernel:5.12.18	20.0 GB	2	2.0 GB	39s ago	Up	38s	10.61
.0.55	cluster-kafka2									
bc2bbd58578c10ed		weaveworks/ignite-ubuntu:20.04	weaveworks/ignite-kernel:5.13.3	20.0 GB	1	2.0 GB	3s ago	Up	2s	10.61
.0.65	cluster-hashui0									
ca67bc7ec06a6330		weaveworks/ignite-ubuntu:20.04	weaveworks/ignite-kernel:5.12.18	20.0 GB	2	2.0 GB	42s ago	Up	41s	10.61
.0.54	cluster-kafka1									

Figure 6: Info about created microVMs stack.

microVMs are highly useful in all use cases, ranging from development to production environments. Adoption of these microVM-based instant stacks will boost engineering effectiveness while slashing your infrastructure bill. ■

Info

- [1] “Goodbye virtual machines, hello container machines” by Ankur Kumar, ADMIN, issue 68, 2022, pg. 46, [https://www.admin-magazine.com/Archive/2022/68/Goodbye-virtual-machines-hello-container-machines]
- [2] “Goodbye cloud VMs, hello laptop VMs” by Ankur Kumar, ADMIN, issue 69, 2022, pg. 22, [https://www.admin-magazine.com/Archive/2022/69/Goodbye-cloud-VMs-hello-laptop-VMs]
- [3] Firecracker: [https://firecracker-microvm.github.io/]
- [4] Firecracker 4,000 microVM demo: [https://github.com/firecracker-microvm/firecracker-demo]
- [5] Ignite GitHub project: [https://github.com/weaveworks/ignite]
- [6] Footloose on GitHub: [https://github.com/weaveworks/footloose]
- [7] Ignite API object documentation: [https://ignite.readthedocs.io/en/stable/declarative-config/]
- [8] Wrapper utilities: [https://github.com/richnusgeeks/devops/tree/master/CloudInABox/MicroVMs/scripts]

Author

Ankur Kumar is a passionate free and open source software (FOSS) hacker and researcher and seeker of mystical life knowledge. He explores cutting-edge technologies, ancient sciences, quantum

spirituality, various genres of music, mystical literature, and art. You can connect with Ankur on [https://www.linkedin.com/in/richnusgeeks] and explore his GitHub site at [https://github.com/richnusgeeks] for other useful FOSS pieces.

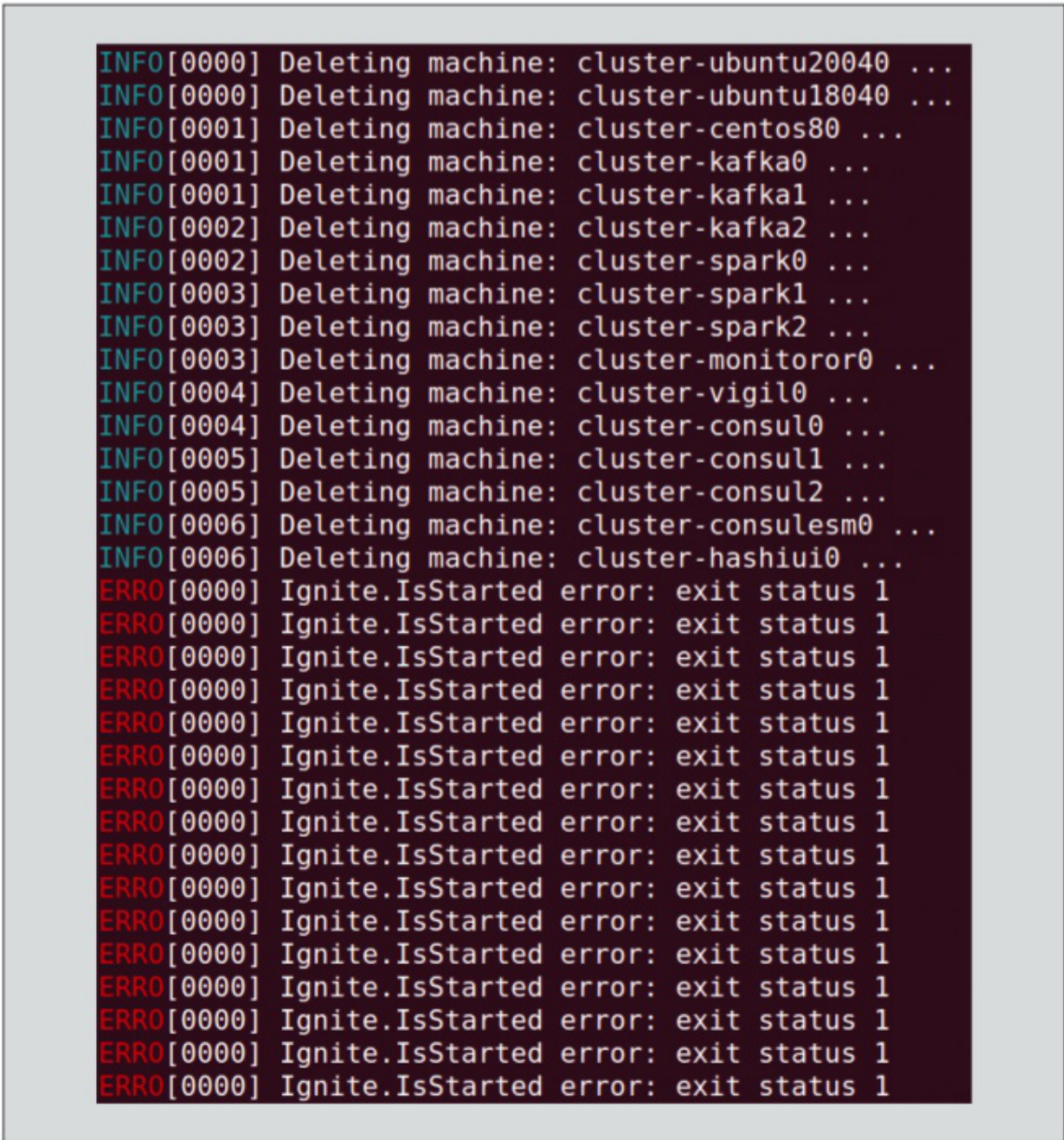


Figure 7: topping microVMs stack.



2021
Archives
Available
Now!

CLEAR OFF YOUR BOOKSHELF WITH DIGITAL ARCHIVES

Complete your collection of *Linux Magazine* and *ADMIN Network & Security* with our Digital Archive Bundles.

You get a full year of issues in PDF format to access at any time from any device.

<https://bit.ly/archive-bundle>



Manage Windows Server storage with PowerShell

Robots in the Data Center

Numerous tools are available to manage the hard drive inventory of Windows servers, but they fall short when it comes to comprehensive automation, which is where PowerShell can help. By Evgenij Smirnov

Experienced Windows administrators are familiar with numerous tools that help them manage hard drives, partitions, and filesystems. The portfolio extends from the Microsoft Management Console (MMC)-based Disk Management (`diskmgmt.msc`) and command-line tools to the Server Manager and Windows Admin Center. For more complex storage systems, you have tools such as `iSCSICLI` for configuring built-in Internet small computer systems interface (iSCSI) initiators or the multipath I/O control panel (`MPIO CPL`) graphical configuration applet for storage multipathing. These tools are perfectly suitable for individual tasks, but when it comes to configuring a large number of servers or migrating complex storage landscapes, the tools quickly reach their limits. The syntax is not consistent, output depends on the operating system language set, and graphical interfaces do not support automation. In precisely such cases administrators turn their attention to PowerShell, which is also particularly important because it enables visual management of a Windows machine's local subsystems in Windows Admin Center.

This article references the current Server 2022 but should also apply to Server 2019 and 2016 without any changes. However, I look exclusively at provisioning storage capacity, which is served up by Server Message Block (SMB), Distributed File System (DFS), Network File System (NFS), or iSCSI supported by separate, sophisticated PowerShell features.

Physical and Logical Drives

Classical administration tasks with regard to the hard drives, their partitioning, and the storage volumes on them are traditionally the domain of the Disk Management MMC console or the `diskpart` command-line tool. The `Get-Disk` command in PowerShell is useful for an initial overview of the existing hard drives (**Figure 1**). The returned objects correspond to the view in Disk Management as long as all existing drives are of *Basic Disk* type. If you convert a disk to a *Dynamic Disk*, it disappears from the `Get-Disk` inventory, but remains visible in Disk Management. In fact, PowerShell does not support operations that require dynamic

disks (the Windows software RAID in various flavors). If you want to script dynamic disks, you will probably have to resort to `diskpart`, because neither PowerShell nor Windows Management Instrumentation (WMI) provide documented programming interfaces. However, dynamic disks are a little out of fashion: Microsoft explicitly recommends Storage Spaces **[1]** for mirrored startup volumes, and Storage Spaces is far better suited for more advanced configurations of local hard drives, with extensive PowerShell support, which I look at later. To initialize a fresh hard drive for operation, use:

```
Get-Disk -Number <drive number> | 2
Set-Disk -IsOffline $false
Initialize-Disk -Number <drive number> 2
-PartitionStyle GPT
```

For the same initialization type for all uninitialized disks use:

```
Get-Disk | 2
Where-Object {$_.IsOffline 2
-and ($_.PartitionStyle -eq 'RAW')} | 2
Initialize-Disk -PartitionStyle GPT
```

Photo by Brian McGowan on Unsplash

To continue processing the disk objects after initialization, you also need to add the `-PassThru` switch to the `Initialize-Disk` cmdlet. To take an active hard drive offline again, use:

```
Get-Disk -Number <drive number> | 2
Set-Disk -IsOffline $true
```

If you are working with real, not logical, disks physically connected to the computer, all of the tools I have mentioned so far will quickly reach their limits. The `Get-Disk` cmdlet does not display dynamic disks, and none of the three tools return disks that have been added to a Storage Spaces pool. Conventional Windows on-board tools let you see the connected disks in Device Management (`devmgmt.msc`), where you also have access to the multipath I/O information of the respective disk if configured for multiple paths. PowerShell helps you achieve this functionality with the `Get-PhysicalDisk` command, but it does not show you any multipathing information. Moreover, the physical

connection information relating to the adapter, target, and logical unit number (LUN) is only available as a string in the `PhysicalLocation` property. Unfortunately, this is exclusively for local drives and not for disks provided by a storage system by iSCSI or Fibre Channel (FC). The command

```
diskpart detail disk
```

shows this information – unless the disks are part of a Storage Spaces pool. With the same preconditions, you can retrieve this information in PowerShell with WMI:

```
Get-CIMInstance Win32_DiskDrive | 2
Select Index, Model, SCSI*
```

The `Index` corresponds to the disk number in `Get-Disk`. If the storage configuration changes outside of your PowerShell session, the PowerShell cmdlets don't always notice it right away. You are probably also familiar with this phenomenon from Disk Management and would select the *Action | Rescan Disks* menu item to fix it. In PowerShell,

the equivalent command is `Update-HostStorageCache`.

Managing Partitions and Volumes

Managing partitions, and the file-system volumes on them, is fraught with the same limitations as hard drive management. Dynamic volumes are invisible to PowerShell; however, you can use the `Get-Partition` cmdlet to list all partitions on all visible disks (i.e., Basic Disks and Storage Spaces Virtual-Disks) without having to commit to a specific disk beforehand. What's even better is how you can manage volumes: Everything is visible and manageable regardless of the underlying storage technology. You can list them with `Get-Volume`, repair them with `Repair-Volume`, analyze and defragment them with `Optimize-Volume`, and reformat them with the `Format-Volume` command. The default setting is `Quick Format`, but you can force full formatting with the `-Full` switch. When creating a new volume, you can either use the entire disk:

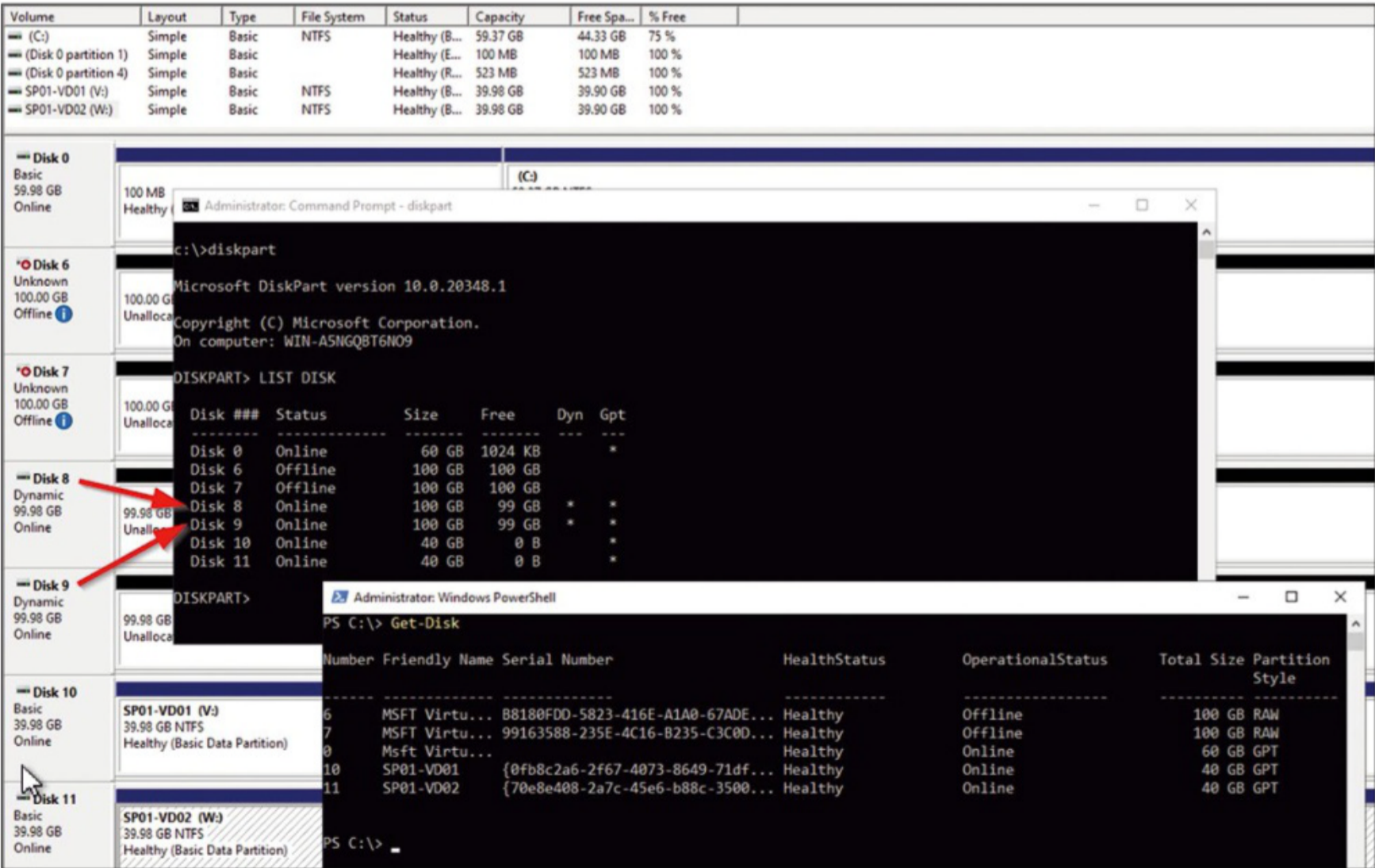


Figure 1: Different tools let admins access different disk types.

```
Get-Disk -Number <number> | 2
New-Volume -FileSystem NTFS 2
-DriveLetter Z -FriendlyName <DiskLabel>
```

or create partitions:

```
Get-Disk -Number <number> | 2
New-Partition -Size <size in bytes> 2
-Offset <offset in bytes>
```

When done, create and format the filesystem volume with:

```
$part = Get-Partition 2
-DiskNumber <number> 2
-PartitionNumber <number>
$part | Set-Partition 2
-NewDriveLetter <letter>
Format-Volume -Partition $part 2
-FileSystem NTFS
```

You can clearly see the connections between volumes and partitions in Windows in the last example. The partition determines the physical location of the object on the disk subsystem and its deployment, and the volume defines the filesystem therein.

Multipath Mounting of iSCSI LUNs

As early as in Windows 2000 Server, the Microsoft operating system introduced its own iSCSI initiator that let you mount storage LUNs on your Windows instances with on-board tools – provided the LUNs were provisioned by the storage system with the iSCSI protocol. MPIO is another valuable feature that is available not only for iSCSI but for all block-based storage protocols, such as serial attached SCSI (SAS), FC, or Fibre Channel over Ethernet (FCoE). MPIO lets the initiator manage multiple connections to the same storage target in line with a defined policy. On a freshly installed Windows system, the iSCSI initiator service (MSiSCSI) is disabled by default. Before configuring the initiator, you first need to configure and start the service for automatic startup with two PowerShell commands:

```
Set-Service MSiSCSI -StartupType Automatic
Start-Service MSiSCSI
```

Now you can configure the iSCSI initiator on your Windows server. One of the most important global settings is the iSCSI qualified name (IQN) of the initiator. To adjust this, use:

```
Set-InitiatorPort -NodeAddress <IQN> 2
-NewNodeAddress <New IQN>
```

To find the existing initiator name, enter:

```
Get-InitiatorPort -ConnectionType iSCSI
```

Changes to the initiator name are often caused by the storage team's naming conventions. However, compelling technical reasons might also be in play. The IQN of a Windows computer assigned by default is a 26-character prefix followed by the computer name. However, older storage systems internally limit the IQN to 32 characters and therefore only evaluate the first six characters of the computer name. For example, if your cluster nodes are named SERVER01, SERVER02, and so on, this kind of storage system will see them all as SERVER, and LUN reservations within the cluster will fail.

Sometimes iSCSI connections take too long to establish when the server boots, and shares that reside on iSCSI LUNs are not provisioned. In this case, you need to make the server service (LanmanServer) dependent on the iSCSI initiator service (MSiSCSI). Creating this dependency is not possible with PowerShell tools; instead, you need to turn to the tried and trusted `sc config depend`. Make sure you add the iSCSI service and do not set it as the only dependency.

If multipathing with your current or future storage system is an option, it is best to configure the MPIO feature right away. In fact, for server clusters, it is mandatory at least to enable MPIO before creating the cluster. The MPIO feature has built-in support for the on-board iSCSI initiator. However, this is disabled by default. Enabling this support requires a reboot, so you will want to complete this configuration step as early as possible:

```
Add-WindowsFeature Multipath-IO
$vid = 'MSFT2005'
$pid = 'iSCSIBusType_0x9'
New-MSDSMSupportedHW 2
-VendorId $vid -ProductId $pid
Set-MSDSMGlobalDefaultLoadBalance-Policy 2
-Policy RR
Restart-Computer -Force
```

The last configuration command sets the path selection policy to round robin (RR) by default. Each subsequent request takes the next active path regardless of the workload. Other valid values for the MPIO policy are F00 for failover only, None for no multipathing, LB for least blocks, and LQD for least queue depth. Make sure you read the storage vendor's integration guide before choosing an MPIO policy because not every iSCSI storage system supports every policy equally well. In most cases you will only have the choice between round robin and failover. The `Get-MPIOSetting` and `Set-MPIOSetting` commands let you configure further global MPIO settings on the system. Once the iSCSI initiator is enabled and MPIO is configured, you can attach your iSCSI storage to the system and read the LUNs presented to your initiator. (For the sake of brevity, I will not look at authenticating against the target storage system here.) The parameters required are provided by `Connect-IscsiTarget` [2]. **Listing 1** shows how to connect an iSCSI target to two paths with PowerShell. Make sure that the IP addresses from the individual storage subnets appear in the same order in the arrays for the initiator and target addresses.

Now, to load the newly attached iSCSI targets in Disk Management, use

```
Update-Host-StorageCache
```

New disks will now show up in your storage inventory, assuming the storage system has served up LUNs to this initiator.

The supplied Windows PowerShell modules do not provide any commands for advanced management of the MPIO settings of the individual devices and paths. WMI support in this area is also

poor and only gives you read access to the MPIO configuration of the individual devices; you cannot change the configuration. You can either turn to the `mpclaim.exe` tool on the command line, with its slightly cryptic syntax, or configure the settings manually in the Device Management GUI (Figure 2).

Storage Spaces with PowerShell

Compared with the legacy Basic Disks and Dynamic Disks, Storage Spaces is a relatively new technology. It originated on Windows Home Server, which Microsoft used as a design study for how to provide fault-tolerant and high-performance storage volumes made up of inexpensive serial ATA (SATA) drives without having to resort to hard-to-manage and error-prone dynamic disks.

The Storage Spaces product was introduced in Server 2012 R2. The new

Server Manager was intended for graphical management from the outset, which means that full PowerShell support must be available under the hood. This support is provided by the Storage module. The previously mentioned `Get-Disk` and `Set-InitiatorPort` commands also originate from this module, which has been expanded from 103 commands in Server 2012 to 160 commands in Server 2022, without changing the module version number.

If you want to create a new storage pool on your Windows system, start with the familiar cmdlet, but now add a parameter:

```
Get-PhysicalDisk -CanPool $true
```

If not all the disks you expected are listed, you can use

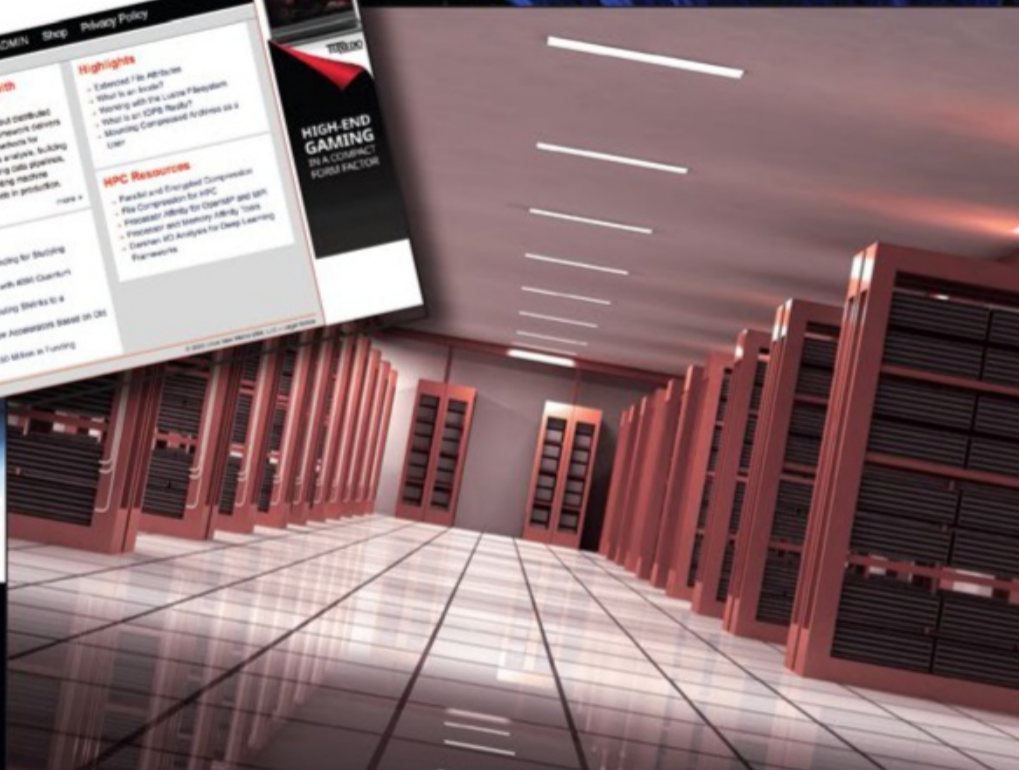
```
Get-PhysicalDisk -CanPool $false | 2
ft DeviceId, CannotPoolReason
```

to output what is preventing specific disks from being added to a pool. Once you have eliminated the problems, create the new pool:

Listing 1: Connecting to an iSCSI Target

```
$iniAddr = @(
    '10.0.104.11'
    '10.0.105.11'
)
$tgtAddr = @(
    '10.0.104.21'
    '10.0.105.21'
)
(0..($iniAddr.Length - 1)) | For-Each-Object {
    New-IsCsiTargetPort `
        -TargetPortalAddress $tgtAddr[$_] `
        -TargetPortalPortNumber 3260 `
        -InitiatorPortalAddress $iniAddr[$_]
}
$targets = Get-IsCsiTarget
(0..($iniAddr.Length - 1)) | For-Each-Object {
    $targets | Connect-IsCsiTarget `
        -IsMultipathEnabled $true `
        -IsPersistent $true `
        -TargetPortalAddress $tgtAddr[$_] `
        -InitiatorPortalAddress $iniAddr[$_]
}
```

A Webzine for High-Performance Computing Specialists



ADMIN
Network & Security

If you work with high-performance clusters, or if you're ready to expand your skill set with how-to articles, news, and technical reports on HPC technology.

admin-magazine.com/HPC

```
$sub = Get-StorageSubSystem 2
-FriendlyName "Windows Storage*"
$disks = Get-PhysicalDisk -CanPool $true
$pool = New-StoragePool 2
-FriendlyName '<Name>' 2
-PhysicalDisks $disks 2
-StorageSubSystemUniqueId $sub.UniqueId
```

In the pool now, enter

```
$pool | New-VirtualDisk 2
-FriendlyName <Name> 2
-Size <size> 2
-ProvisioningType Thin 2
-ResiliencySettingName Mirror 2
-NumberOfDataCopies 2
```

to create a virtual disk with thin provisioning that has a RAID1-style level of protection (two copies of each block). Unlike the step-by-step wizard in Server Manager, the availability of the required number and size of disks is not checked until you submit the command, so it might experience a failure when you try to create the drive. The object created at the PowerShell pipeline is a *Disk*. You can pipe this to *Initialize-Disk* and then on to *New-Partition* and *Format-Volume* (as described at the beginning of this article) to create a formatted disk. Like every more complex storage system, it is important to monitor the status of the drives and

initiate repairs at an early stage. To help you do this, all objects returned by the *Get-StoragePool*, *Get-VirtualDisk*, *Get-PhysicalDisk*, or *Get-Volume* cmdlets have a *HealthStatus* property. If it does not read *Healthy*, you need to check the status of the individual components and eliminate any problems. Sometimes, however, logical errors creep in. Storage Spaces can usually fix these errors itself, provided the redundancy level you selected is sufficient to let it do so. You can initiate this process for all virtual disks with a PowerShell command:

```
Get-VirtualDisk | 2
Where-Object {2
    $_.HealthStatus -Eq "Unhealthy"} | 2
Repair-VirtualDisk
```

The *Repair-VirtualDisk* cmdlet can also handle the *-PassThru* switch, which you can use to pipe the results of the repair. If the repair fails, you need to continue digging down to find the root cause. Storage Spaces offer a variety of other features, such as tiered storage, enclosure management, and more. All these functions are fully supported by matching custom PowerShell cmdlets and additional parameters in the more general commands. In fact, the only

management interface for Storage Spaces is PowerShell – both Server Manager and Windows Admin Center rely on PowerShell for graphical management.

Conclusions

Microsoft provides PowerShell modules [3] that enable automation of your storage provisioning and management. To leverage the maximum benefits, you need to learn about the Windows storage subsystem, and especially about its different modes of operation and the options for addressing disks. However, operations with traditional dynamic disks are no longer supported by PowerShell or WMI. For newer systems, you will want to opt for Storage Spaces, anyway, for which full PowerShell support is available.

Info

- [1] Dynamic disks: [https://docs.microsoft.com/en-us/windows/win32/fileio/basic-and-dynamic-disks#dynamic-disks]
- [2] Connect-IscsiTarget: [https://docs.microsoft.com/en-us/powershell/module/iscsi/connect-iscsitarget?view=windowsserver2022-ps]
- [3] Storage module help: [https://docs.microsoft.com/en-us/powershell/module/storage/?view=windowsserver2022-ps]

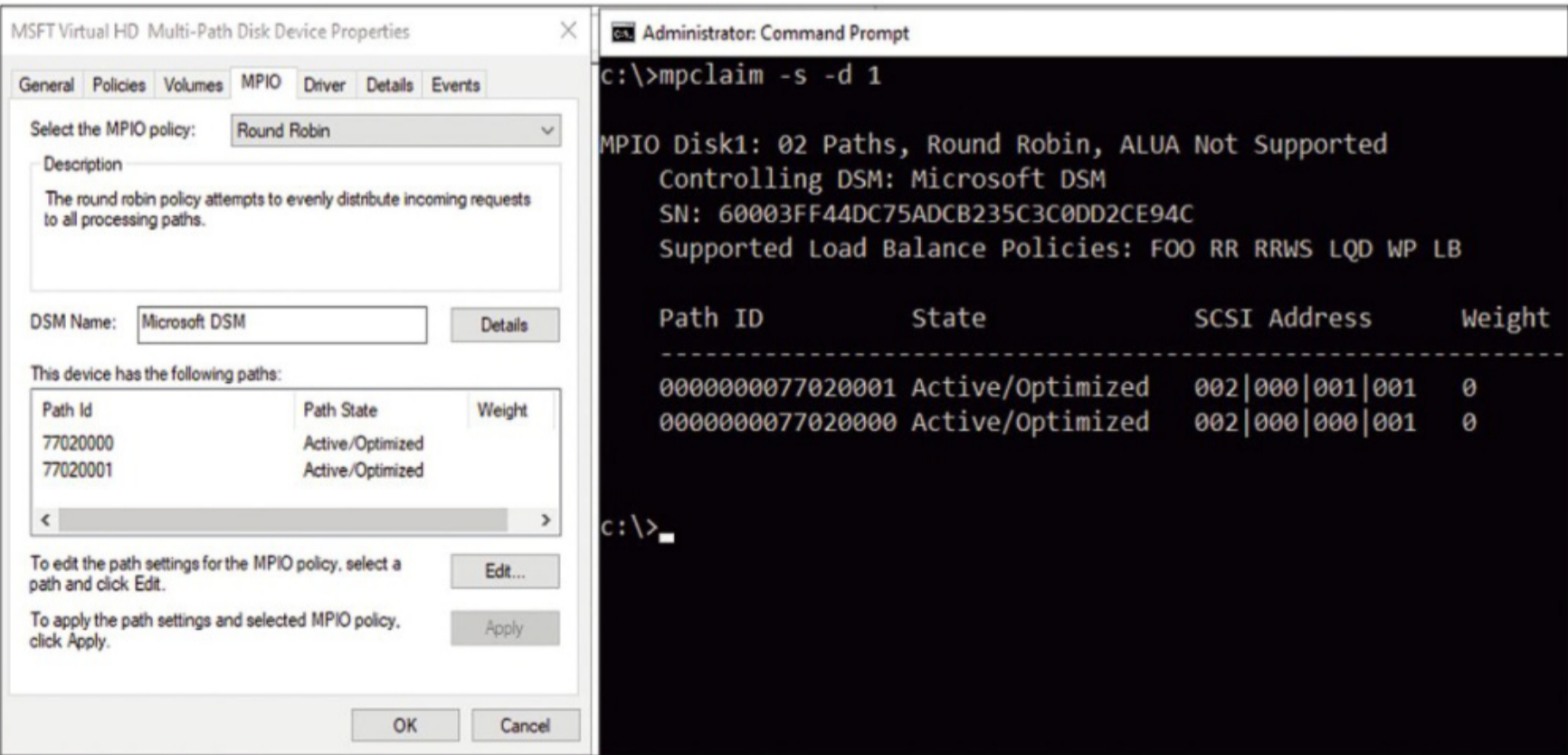


Figure 2: Advanced management of the MPIO settings from the Device Management GUI or the command line.



Linux Magazine is your guide to the world of Linux. Look inside for advanced technical information you won't find anywhere else!

Expand your Linux skills with:

- In-depth articles on trending topics, including Bitcoin, ransomware, cloud computing, and more!
- How-tos and tutorials on useful tools that will save you time and protect your data
- Troubleshooting and optimization tips
- Insightful news on crucial developments in the world of open source
- Cool projects for Raspberry Pi, Arduino, and other maker-board systems

If you want to go farther and do more with Linux, subscribe today and never miss another issue!

Subscribe now!

shop.linuxnewmedia.com/subs

GET IT NOW!

FAST DELIVERY
WITH OUR PDF
EDITION



Quick UDP Internet connections

Fast Track

The UDP-based Quick UDP Internet Connections (QUIC) protocol comes with mandatory TLS encryption and promises faster speeds. By Benjamin Pfister

The Quick UDP Internet Connections (QUIC) protocol [1] originated in 2012 as a Google project led by Jim Roskind to improve security and performance over TCP. The protocol made the leap to four Requests for Comments (RFCs) [2]-[5] under numbers 8999 to 9002 in 2021 after a good five years of development at the Internet Engineering Task Force (IETF). In addition to Google, developers from Fastly, Mozilla, and others participated in the development of the transport protocol at the IETF. However, QUIC is not just of academic value, as is shown by the first application protocols on which it is already based. For example, the third version of the Hypertext Transport Protocol (HTTP/3) uses QUIC. The first drafts of the Session Initiation Protocol (SIP) successor Real-Time Internet Peering for Telephony (RIPT) for voice over IP (VoIP) networks are also based on QUIC, and both are expected to increase penetration rapidly in the near future.

Specifics and Areas of Use

QUIC is considered a potential TCP successor, but netizens in particular

are finding it difficult to appreciate because it merges OSI Layer 4 (transport layer) and Layer 5 (session layer), which breaks with old paradigms. QUIC is basically based on the connectionless User Datagram Protocol (UDP) and uses UDP port 443. However, QUIC itself has a connection-oriented architecture.

In version 1, the protocol is said to enable a change of the communication path during an ongoing session, which could be interesting in mobile scenarios. Mandatory TLS 1.3 encryption is intended to ensure confidentiality, integrity, and availability, which is equivalent to the security-by-default approach. Additionally, it is intended to avoid interference by active network components on the network path on the basis of TCP headers or content information.

In contrast to the multilayer session setup for classic communication in the form of a TCP and TLS handshake, TLS was integrated directly into the QUIC handshake with the intention to ensure lower latency in the connection setup and consequently an improved user experience. Cloud services and websites that rely on a large number of content sources,

especially, lead to multiple, sequential processing of the handshake processes, which results in increased latency of the connection setup. This operation is what QUIC wants to do better, and the challenge is growing, especially for WAN connections, which have higher latency per se. The protocol in its current version can detect packet loss and congestion in the flow within the communication relationship. Additionally, it prevents head-of-line blocking (i.e., situations in which the transmission of one packet slows down others). TCP sockets are built on source IP-port and target IP-port combinations. However, an IP change occurs in many cases, especially in mobile use (e.g., when switching from the enterprise network to a public mobile network when employees leave the company building). What is of particular interest for mobile devices is that, in contrast to TCP, it is now possible to change the IP address and port connection parameters. The need to open a new session when switching from wired to wireless networks can be removed. If network address translation (NAT) is used, timeouts and, consequently, port adjustment can occur if active transmission is absent for a long period of time. To increase the efficiency of the communication relationship, QUIC also supports multiplexing multiple

flows. The developers aimed to enable all these improvements without needing to adapt existing networks.

Protocol Structure

Connectionless UDP provides the basis of the QUIC protocol. Data is exchanged between applications through a QUIC connection in the form of streams; the streams themselves can be unidirectional or bidirectional. A connection like this supports multiple streams. A communication relationship is abstracted by connection identifiers from IP addresses and ports to enable the decoupling of this volatile information. However, the change can currently only be made on the client side. Despite changes to the IP or port information, the identifier ensures that the data arrives at the correct endpoint and that unique mapping is possible. The connection identifier is defined on the client and server. The partner in the communication relationship needs to send the data to be transmitted to the connection identifier specified by the other end. If you have no need to differentiate between different connections on an endpoint, you have

another option: zero-length connection IDs. Of course, these must not be used in combination with connection multiplexing on the same IP address-port combination.

How does the sender of data learn the respective initial connection ID that the potential recipient has assigned? The handshake has a source connection ID field in the long packet header for this purpose. If the receiver has communicated this information to the sender in the handshake, address mapping can take place in the reverse direction given the correct destination connection ID. If additional connection IDs are required in addition to the IDs initially transmitted in the handshake, they can be announced with `NEW_CONNECTION_ID` frames.

The previously mentioned handshake phase is also the starting point of a QUIC connection. In addition to the connection IDs, the client and server also negotiate a TLS 1.3-based shared secret and the QUIC-based application protocol. In addition to the classic connection setup by handshakes, QUIC allows clients to send data to a server before receiving data from it (i.e., a resumption of the communication relationship) if connection

information from an old connection is available. This feature is known as zero round-trip time (0-RTT); it is optional and not mandatory. The problem is the lack of protection against replay attacks.

The Header Structure

QUIC comes with short and long headers. Until the one round-trip time (1-RTT) keys have been exchanged, the slightly less efficient long headers are used; short headers are only used afterward. As you can see from the Wireshark screenshot in [Figure 1](#), the QUIC header has a header length field in the connection information. A value of 0 means a short header, and a 1 means a long header. Additionally, the connection information contains a fixed bit and a packet type; the fixed bit indicates whether it is for the version negotiation or a regular packet. The packet type is self-explanatory and has a total of four types in the long form: Initial, Handshake, Retry, and 0-RTT protected packets. To enable a correct interpretation of the header, a version field is set to 1, or 0x000001 in the current version. Additionally, you can see the previously

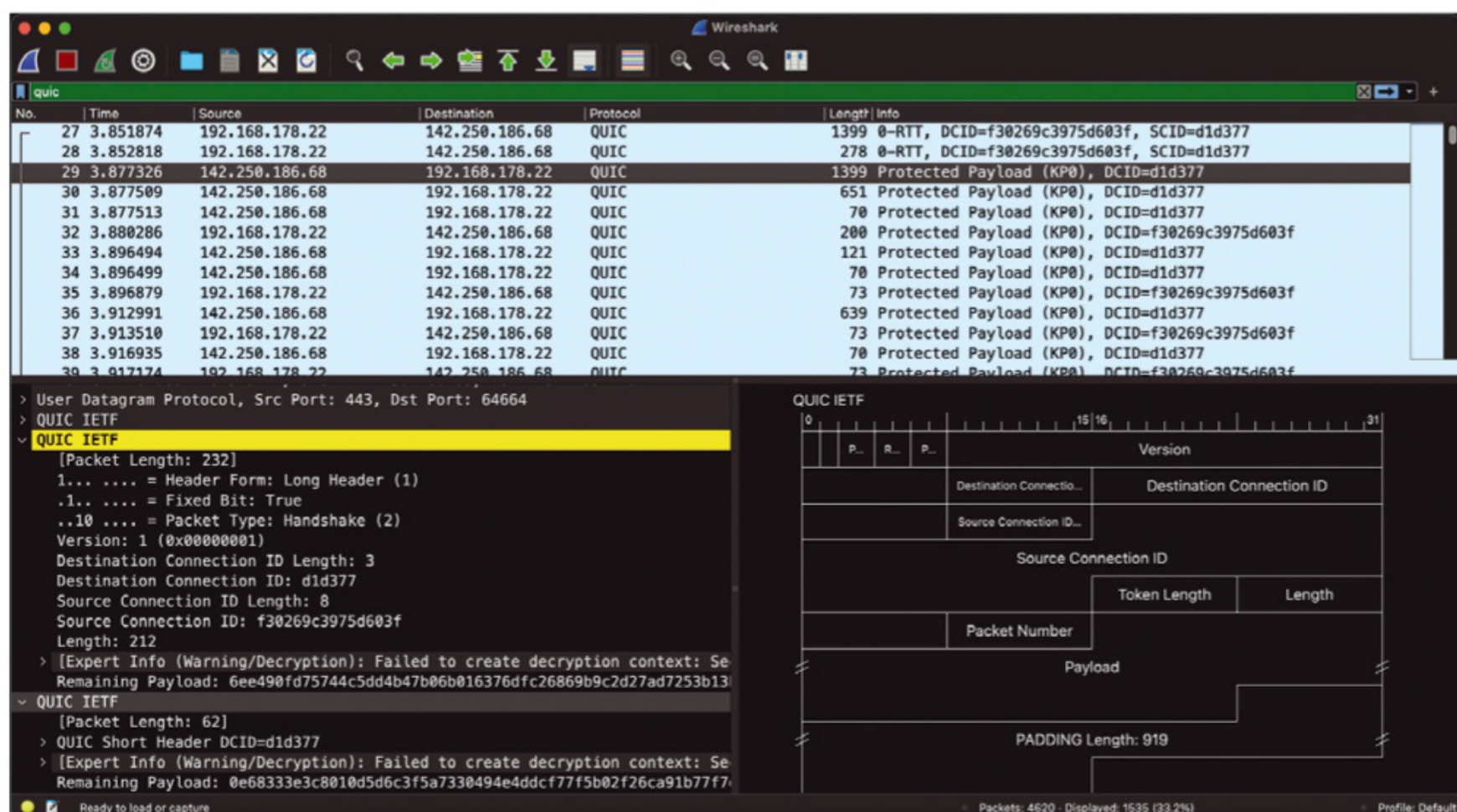


Figure 1: The Wireshark output of a call to [cloudflare.com](#) in Google Chrome without any special settings. Wireshark also offers a display filter for QUIC.

mentioned source and target connection IDs and associated length information, followed by the QUIC payload.

To explain how version negotiation takes place, you need to refer to RFC 8999. In contrast to other processes, confidentiality and integrity cannot yet be ensured in the versioning process. When receiving a long header with an unknown or unsupported version number, the receiver returns 0x000000 in the version field. Additionally, it inserts a list of supported versions in the Supported Version field.

After the key and version have been negotiated, short headers are used in the 1-RTT phase; hence, the unique package type name *1-RTT*. Because the connection IDs and versions were already announced or negotiated in the long headers during the handshake phase, the short header does not contain any source connection ID information, length information for the target connection ID, or version field. This arrangement makes the handshake more efficient, but it also comes with a spin bit header field, which supports passive latency monitoring in the communication path. The server only reflects the received spin value, whereas the client flips it from 1 to 0 or vice versa. Therefore, the end-to-end round trip time can be approximated on the basis of this flip.

Terminating a Connection

Every connection that is established needs to be terminated at a later time. There are several ways of doing this. In the event of a violation of protocol specifications, an immediate close is used. If an endpoint sends an immediate close, the connection is terminated immediately, as the name implies. A stateless reset can be used in the case of connection problems. A specific example of a connection problem would be a recipient who receives packets from a former sender to a destination connection ID that no longer exists.

An IDLE timeout is an additional variant. If an endpoint inserts a `max_idle_timeout` in the transport parameters,

a refresh must take place because a packet was successfully received and processed.

An IDLE timeout is another variant. If an endpoint adds a `max_idle_timeout` to the transport parameters, a refresh takes place because a packet has been successfully received and processed.

Security Measures in QUIC

One of the most important security aspects of QUIC is its mandatory TLS 1.3 encryption, which automatically includes forward secrecy, ruling out a subsequent decryption. The associated TLS handshake takes place directly in the QUIC handshake. Server authentication is mandatory, as in other TLS versions, whereas client authentication is an optional parameter. Unique keys are used in each connection, and the associated key material is used in both 0-RTT and 1-RTT packets.

Amplification attacks pose a challenge for UDP-based protocols because they cause a relatively small request to generate a huge response. Source address spoofing can be leveraged in this way to attack a third-party system. Therefore, the protocol developers placed great emphasis on address validation to mitigate the risk of spoofing attacks. This address validation comes into play both when the connection is established and when it is migrated. If the receiver has not yet completed address validation, the maximum size of the response packet is limited to three times the size of the previously received request. The payload must be at least 1,200 bytes in the initial packet; otherwise, padding is required.

A token exchange before compute-intensive operations helps to prevent server-side denial-of-service attacks. Handshake keys enable protection against handshake termination attacks following the initial packets, but measures have also been taken against optimistic ACK attacks (i.e., confirming the receipt of packages that have not yet been received). The developers also provide recommendations for preventing request forgery attacks.

Initial Implementations

This potential TCP successor is not just a protocol definition without any practical use, as is already underlined by numerous implementations. Other protocols in addition to the previously mentioned HTTP/3 are based on it. Among other things, they include a feature officially published in Windows Server 2022 Datacenter: Azure Edition: SMB over QUIC. This protocol encapsulates the Server Message Block (SMB) protocol for file transfer within QUIC to leverage its benefits. Integrated TLS 1.3 encryption makes this particularly interesting if you need external access for mobile worker scenarios and internal access with strict security requirements at the same time. For the mobile scenario in particular, the ability to change the client's IP address or port is useful (e.g., when switching between hotspots and mobile data networks). This switch happens transparently for the end user in Microsoft's case. Microsoft is looking to enable public cloud-based filesystem access with a high level of security but with the same user experience as offered by local SMB file servers, without needing a dedicated virtual private network (VPN).

A QUIC approach resides in the controversial environment of DNS encryption: DNS-over-QUIC could be of interest as an alternative to DNS-over-TLS (DoT) and DNS-over-HTTPS (DoH) because it reduces handshake times, which in turn gives users a better experience thanks to faster performance. QUIC is already integrated and activated in current versions of Chrome and Mozilla Firefox.

Another interesting field of application for QUIC is the area of IP telephony. The first drafts of a RIPT protocol appeared in 2020. The protocol is being lauded as a potential successor to the SIP signaling protocol. SIP is still used without encryption in many cases. The problem here is that the TLS-encrypted variant is always based on TCP, which means greater latency in the initial connection setup ([Figure 2](#)); users notice this

when establishing a call, for example. For an overview of the implementations, check out GitHub [6] for some implementations with the underlying programming languages and, for most part, the supported target platforms, QUIC versions, and roles (i.e., client, server, or both). Some solutions are already available there since the release of version 1 – more specifically, MsQuic, a library in C for Windows, Linux, and macOS; Neqo, a Mozilla/Firefox QUIC and HTTP3 implementation in Rust; Aioquic, a Python library; lsquic, a LiteSpeed QUIC and HTTP/3 library in C for Linux, FreeBSD, macOS, Android, and Windows; and quant, QUIC userspace accelerated network transfers in C.

QUIC Challenges

Despite all its benefits, QUIC still has some issues. For example, it remains to be seen how well outgoing connections will work on UDP port 443. Especially in combination with HTTP/3, on which it is based, the question arises, for example, as to the extent to which dropped connections at firewalls or stateless packet filters (ACLs) will occur at hotspots or on corporate networks.

Intrusion prevention systems and proxies could also be a problem with QUIC. Support for QUIC first needs to find its way into these products. A fallback to HTTP/2 or even HTTP/1.1 and consequently to TCP/443 is likely to be the way out in some cases. COVID-19, in combination with various security products and the partly proprietary protocols of video conferencing software products, has revealed that UDP-based protocols in

next-generation firewalls pass through multiple security checks. However, rate limits in UDP connections can also be a challenge. For example, UDP Flood Protection can have a negative effect on bulk transfers over UDP. Don't forget that the Apache web server does not yet have a QUIC feature – except for a development version for NGINX (released in January 2022). Because of the interaction of transport and session layers, the protocol as a whole is quite complex. To what extent QUIC's congestion control will have an effect on local data networks with high bandwidths remains to be seen. The implementation of QUIC in userspace is both a curse and a blessing. The new protocol versions do not require a kernel update on the client side, just an application update.

Conclusions

QUIC is likely to show its strengths in particular when accessing cloud applications and websites where the content is distributed across different target servers. High-latency communication relationships can also benefit from the integrated TLS handshake and 0-RTT feature. Additionally, interesting application scenarios are conceivable in the environment

of latency-critical communication such as VoIP or sequential database access. Remote access by VPN could open up another field of application. Some manufacturers currently still resort to the UDP-based Datagram Transport Layer Security (DTLS), TLS, or IPsec framework. Because of the requirement for encryption with TLS 1.3, QUIC offers a genuine alternative.

One disadvantage is the higher CPU load compared with TCP and TLS. It would take a great deal of stargazing to forecast how many manufacturers and open source projects will switch to QUIC and the extent to which UDP communication will conquer today's networks. ■

Info

- [1] IETF on QUIC: [<https://www.ietf.org/blog/whats-happening-quic/>]
- [2] RFC 8999: [<https://datatracker.ietf.org/doc/html/rfc8999>]
- [3] RFC 9000: [<https://datatracker.ietf.org/doc/html/rfc9000>]
- [4] RFC 9001: [<https://datatracker.ietf.org/doc/html/rfc9001>]
- [5] RFC 9002: [<https://datatracker.ietf.org/doc/html/rfc9002>]
- [6] QUIC implementations: [<https://github.com/quicwg/base-drafts/wiki/Implementations>]

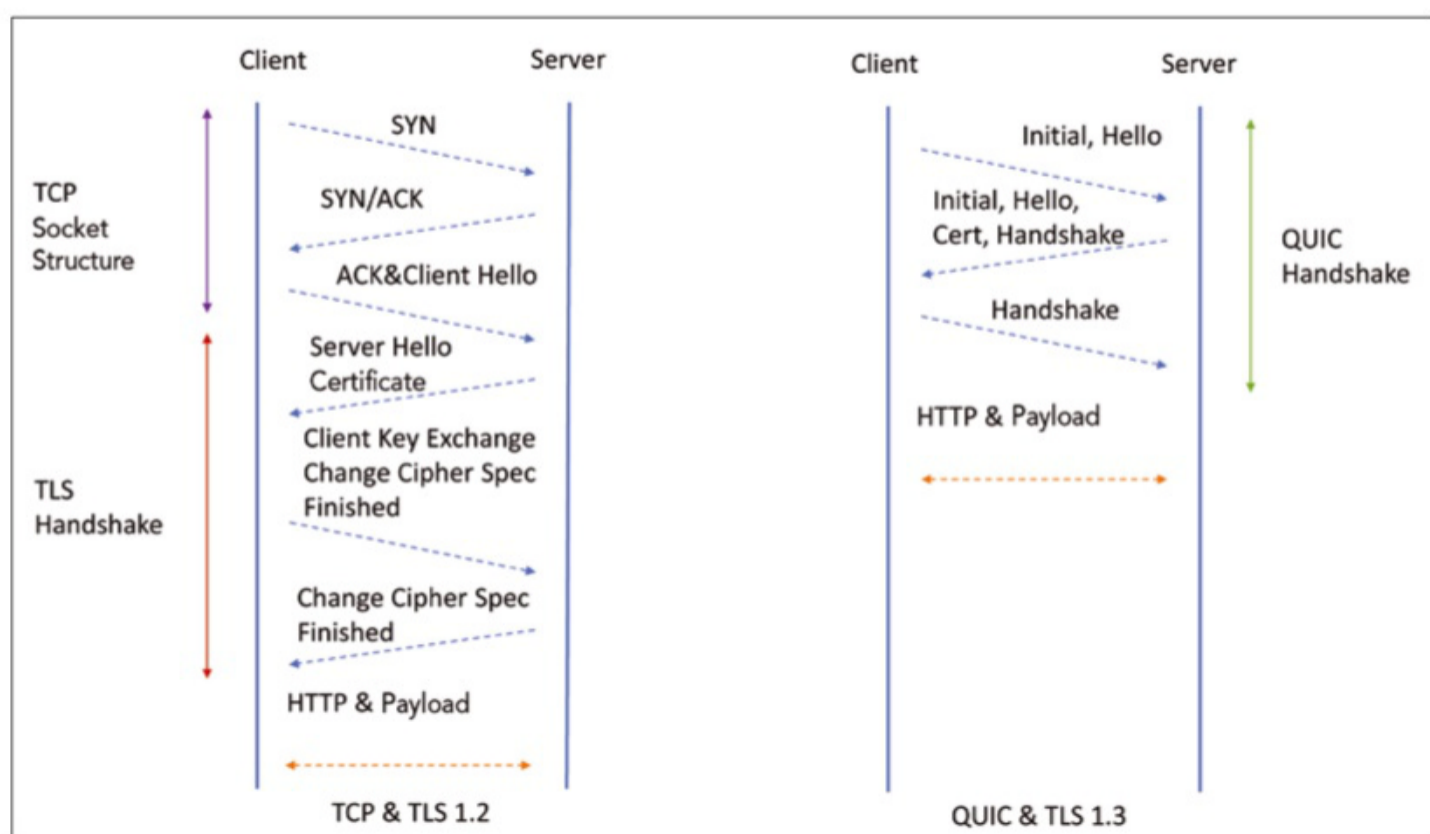


Figure 2: Comparing the TCP connection setup for TLS 1.2 on the left side and for QUIC on the right. The latency advantage is clearly visible.



Baselines are more important
than the benchmark

Witness Mark

Defining I/O baselines helps you determine the highest performance you can expect from your system when configured properly. By Federico Lucifredi

“It is too slow”: These four words launch nearly every performance analysis. *Slow* is an ambiguous term,

which may equally represent very different concerns, including change in performance compared with a

previous versions or how the same software ran on some previous day. Equally, it may represent inadequate performance compared with what performance a system is *believed* to be capable of delivering.

I have examined this second definition before [1], studying how the library of CPU benchmarks published by the 7-Zip project [2] could be used to compare with observed CPU performance on my system. This month, I explore how to define baselines for I/O, including the storage and network subsystems. The I/O paths are more vulnerable to performance setbacks because of system or software misconfiguration than CPU or memory, having to rely on multiple components to perform optimally – or in the case of the network, external hops through the Internet itself.

```
pandora:~ lucifred$ iperf3 -c 159.89.52.175
Connecting to host 159.89.52.175, port 5201
[ 5] local 10.0.0.26 port 53691 connected to 159.89.52.175 port 5201
[ ID] Interval      Transfer    Bitrate
[ 5]  0.00-1.00    sec   3.16 MBytes  26.5 Mbits/sec
[ 5]  1.00-2.00    sec   1.20 MBytes  10.0 Mbits/sec
[ 5]  2.00-3.00    sec   2.43 MBytes  20.4 Mbits/sec
[ 5]  3.00-4.00    sec   1.81 MBytes  15.1 Mbits/sec
[ 5]  4.00-5.00    sec   2.89 MBytes  24.3 Mbits/sec
[ 5]  5.00-6.00    sec   2.83 MBytes  23.8 Mbits/sec
[ 5]  6.00-7.00    sec   2.18 MBytes  18.2 Mbits/sec
[ 5]  7.00-8.00    sec   2.66 MBytes  22.4 Mbits/sec
[ 5]  8.00-9.00    sec   2.86 MBytes  24.0 Mbits/sec
[ 5]  9.00-10.00   sec   2.88 MBytes  24.2 Mbits/sec
-----
[ ID] Interval      Transfer    Bitrate
[ 5]  0.00-10.00   sec  24.9 MBytes  20.9 Mbits/sec
[ 5]  0.00-10.08   sec  24.7 MBytes  20.5 Mbits/sec

iperf Done.
pandora:~ lucifred$
```

Figure 1: *iperf3* client in action. Note how network results can vary significantly over just a few seconds.

```
root@focal:~# iperf3 -s
-----
Server listening on 5201
-----
Accepted connection from 73.69.242.234, port 53690
[ 5] local 159.89.52.175 port 5201 connected to 73.69.242.234 port 53691
[ ID] Interval      Transfer    Bitrate
[ 5]  0.00-1.00    sec   2.16 MBytes  18.1 Mbits/sec
[ 5]  1.00-2.00    sec   1.35 MBytes  11.4 Mbits/sec
[ 5]  2.00-3.00    sec   2.71 MBytes  22.7 Mbits/sec
[ 5]  3.00-4.00    sec   1.98 MBytes  16.6 Mbits/sec
[ 5]  4.00-5.00    sec   2.84 MBytes  23.9 Mbits/sec
[ 5]  5.00-6.00    sec   2.84 MBytes  23.8 Mbits/sec
[ 5]  6.00-7.00    sec   2.22 MBytes  18.6 Mbits/sec
[ 5]  7.00-8.00    sec   2.71 MBytes  22.7 Mbits/sec
[ 5]  8.00-9.00    sec   2.80 MBytes  23.5 Mbits/sec
[ 5]  9.00-10.00   sec   2.84 MBytes  23.8 Mbits/sec
[ 5] 10.00-10.08   sec    221 KBytes  23.8 Mbits/sec
-----
[ ID] Interval      Transfer    Bitrate
[ 5]  0.00-10.08   sec  24.7 MBytes  20.5 Mbits/sec

receiver
```

Figure 2: Running the *iperf3* server in Digital Ocean’s New York region to exercise a path to a known location.

The Network Is the Computer

Setting a baseline is essentially figuring out what is the highest performance that can be expected if the system on hand is properly configured. The simplest tool reproducing the tested configuration is always to be preferred, to limit the multitude of variables under consideration. Usually, the application the system is meant for is a much richer and more complex beast, and if you are testing for performance, usually the application has already shown undesirable behavior anyway.

The *iperf3* tool [3] is your go-to utility to test a network path’s baseline,

Hot-Potato Routing

Peering agreements between large networks may not specify traffic cost settlement or prescribe specific routes, granting operators the freedom to define paths between their networks. In such cases, “hot-potato routing” [4] often results, with a network handing over a packet to its peer network at the closest available peering point, to minimize network load and costs. When the peer network adopts the same practice, different routes may be in use for the two directions of a given network connection, with asymmetric paths through the Internet being not at all uncommon. Although not normally a concern and usually invisible to the end user, it is important to keep in mind while testing network connections.

end to end. Found in the Ubuntu Universe repository (install with `apt install iperf3`) and in macOS’s Brew (brew install iperf3), it measures the effective bit rate between two systems (Figure 1) by taking multiple samples and averaging the results. By default, iperf3 uploads from client to server, but the reverse is also possible (-R option), sending data in the opposite direction to validate asymmetric network paths (see the “Hot-Potato Routing” box). Here, I have set up a server in the cloud (Figure 2) running Ubuntu Focal 20.04: The client is running in a terminal on my MacBook as I write. Ports to Microsoft Windows also exist [5], and testing UDP instead of the default TCP is an option, as well.

Secure Testing

SSL/TLS connections are another common case of networking baseline because they place a significant compute load on the remote endpoint and because of the complexity of the server architectures involved. Often spanning multiple servers on the remote end, an SSL benchmark is the ultimate test of what takes place in practice, as opposed to what theory predicts. The OpenSSL project incorporates two useful benchmarks: `openssl speed` [6], which tests cryptographic performance in abstract on the local machine without any network access, and the `s_time` command [7], which performs an all-encompassing, end-to-end test. The `s_time` benchmark can be used to examine a server’s capacity to handle connections in a given time span and how long it took to serve secured content.

Figure 3 shows an example test with Google’s search engine as the endpoint, historically one of the Internet’s snappiest endpoints. Results are provided for new sessions as well as cached session IDs, which performed significantly better. Another consideration that needs to be kept in mind is that the test is against Google as a whole, their primary domain running in an obvious round-robin setup (see the “Round-Robin DNS” box), and further schemes such as load balancers or Anycast routing [8] are probably in use.

```
pandora:/ lucifred$ openssl s_time -connect www.google.com:443 -www /
Collecting connection statistics for 30 seconds
*****
*****
123 connections in 1.91s; 64.40 connections/user sec, bytes read 6596124
123 connections in 31 real seconds, 53627 bytes read per connection

Now timing with session id reuse.
starting
*****
*****
132 connections in 0.46s; 286.96 connections/user sec, bytes read 7079223
132 connections in 31 real seconds, 53630 bytes read per connection
pandora:/ lucifred$
```

Figure 3: Timing the ability of Google.com to serve SSL sessions, without differentiating whether one or multiple servers are answering the requests.

The Storage with the Most

I have covered a lot of I/O testing tools in this column over the years, and many can be used to define a baseline, which is a technique, rather than a tool. I/O systems are noisy, and benchmarking them can be exceedingly difficult. In some tests, it may be beneficial to eliminate storage altogether and perform the test directly against a RAM disk. Provisioning a half-gigabyte RAM disk is trivially simple in Linux:

```
# modprobe brd rd_nr=1 rd_size=524288
# ls /dev/ram0
/dev/ram0
```

You can use this approach to evaluate the performance effect of encrypted partitions without having

Round-Robin DNS

In a round-robin DNS setup, multiple addresses are offered as destinations for the same domain name. For example, today at my office in Westford `www.google.com` resolves to the following six different IP addresses:

```
...
$ nslookup www.google.com
Server:      10.192.206.245
Address:     10.192.206.245#53
```

```
Non-authoritative answer:
Name:   www.google.com
Address: 142.251.111.106
Name:   www.google.com
Address: 142.251.111.104
Name:   www.google.com
Address: 142.251.111.103
Name:   www.google.com
Address: 142.251.111.105
Name:   www.google.com
Address: 142.251.111.147
Name:   www.google.com
Address: 142.251.111.99
```

The ordering of these answers will differ from query to query, and the DNS client will pick one address (likely the first) and connect to it. Whether repeated connections to the same domain name result in connections to the same IP is implementation dependent, something a clever performance engineer may well choose to disambiguate explicitly with an IP address, depending on the ultimate objective of the test.

to worry about the noise of the underlying storage medium, simply comparing the performance of RAM disk access, with and without making use of encryption. A similar, complementary technique is writing the file directly to disk, without the intervening filesystem layer affecting measurements, as I have demonstrated previously [9]. To test encryption without a filesystem, you have to use a detached header to store encryption keys, lest your benchmark accidentally overwrite them because they are stored on the same drive by default. Listing 1 details the setup process, resulting in `/dev/ram0` directly accessing the RAM disk, whereas `/dev/mapper/encrypted-ram0` is first encrypted

by LUKS [10] before storing to the same memory. Listing 2 then shows the simplest possible benchmark, with `dd` [11] to compare block performance in both modes. The raw device performs three times as fast as encrypted access to the same storage. The critical finding is that encryption will not be the performance bottleneck in this setup, as long as the storage medium is not capable of more than 210 MB/s of sustained throughput. ■

Info

- [1] “Data Compression as a CPU Benchmark” by Federico Lucifredi, *ADMIN*, issue 66, 2021, pg. 94, [https://www.admin-magazine.com/Archive/2021/66/Data-Compression-as-a-CPU-Benchmark]

Listing 1: LUKS Encryption Overlay Set-up

```
root@focal:~# # Allocate a half-GB RAM disk
root@focal:~# sudo modprobe brd rd_nr=1 rd_size=524288
root@focal:~# ls /dev/ram0
/dev/ram0
root@focal:~# fallocate -l 2M header.img
root@focal:~# echo -n "not a secure passphrase" | cryptsetup luksFormat -q /dev/ram0 --header header.img -
root@focal:~# # Open ram0 as an encrypted device
root@focal:~# echo -n "not a secure passphrase" | cryptsetup open --header header.img /dev/ram0 encrypted-ram0
root@focal:~# ls /dev/mapper/encrypted-ram0
/dev/mapper/encrypted-ram0
```

Listing 2: Encrypted vs. Plain Text

```
root@focal:~# dd if=/dev/zero of=/dev/ram0 bs=4k count=100k
102400+0 records in
102400+0 records out
419430400 bytes (419 MB, 400 MiB) copied, 0.535233 s, 784 MB/s
root@focal:~# dd if=/dev/zero of=/dev/mapper/encrypted-ram0 bs=4k count=100k
102400+0 records in
102400+0 records out
419430400 bytes (419 MB, 400 MiB) copied, 1.99686 s, 210 MB/s
```

- [2] 7-Zip LZMA CPU benchmark library: [https://www.7-cpu.com/]
- [3] iperf3(1) man page: [https://manpages.ubuntu.com/manpages/focal/en/man1/iperf3.1.html]
- [4] Hot-potato routing: [https://en.wikipedia.org/wiki/Hot-potato_and_cold-potato_routing#Hot-potato_routing]
- [5] iperf for Windows: [http://www.iperfwindows.com/index.html]
- [6] openssl speed(1) man page: [https://manpages.ubuntu.com/manpages/focal/man1/speed.1ssl.html]
- [7] openssl s_time(1) man page: [https://manpages.ubuntu.com/manpages/focal/en/man1/s_time.1ssl.html]
- [8] Anycast routing: [https://en.wikipedia.org/wiki/Anycast]
- [9] “Testing the Samsung MU-PA500B 500GB SSD” by Federico Lucifredi, *ADMIN*, issue 47, 2018, pg. 92, [https://www.admin-magazine.com/Archive/2018/47/Testing-the-Samsung-MU-PA500B-500GB-SSD]
- [10] LUKS: [https://gitlab.com/cryptsetup/cryptsetup/blob/master/README.md]
- [11] dd(1) man page: [https://manpages.ubuntu.com/manpages/focal/en/man1/dd.1.html]

The Author

Federico Lucifredi (@0xf2) is the Product Management Director for Ceph Storage at Red Hat, 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 O’Reilly title *AWS System Administration*.

ADMIN

Network & Security

NEWSSTAND

Order online:
bit.ly/ADMIN-Newsstand

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.

#70 - July/August 2022

Defense by Design

Nothing is so true in IT as “Prevention is better than the cure.” We look at three ways to prepare for battle.

On the DVD: Rocky Linux 9 (x86_64)



#69 - May/June 2022

Terraform

After nearly 10 years of work on Terraform, the HashiCorp team delivers the 1.0 version of the cloud automation tool.

On the DVD: Ubuntu 22.04 “Jammy Jellyfish” LTS server Edition



#68 - March/April 2022

Automation in the Enterprise

Automation in the enterprise extends to remote maintenance, cloud orchestration, and network hardware

On the DVD: AlmaLinux 8.5 (minimal)



#67 - January/February 2022

systemd Security

This issue, we look at how to secure systemd services and its associated components.

On the DVD: Fedora 35 Server (Install)

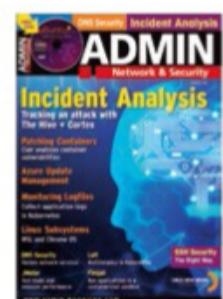


#66 - November/December 2021

Incident Analysis

We look at updating, patching, and log monitoring container apps and explore The Hive + Cortex optimization.

On the DVD: Ubuntu 21.10 “Impish Indri” Server Edition

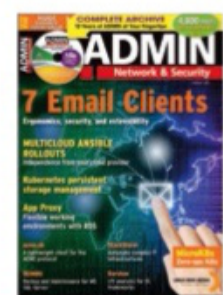


#65 - September/October 2021

7 Email Clients

The features in this issue tackle digital certificates, email clients, and HP backup strategies.

On the DVD: Complete ADMIN Archive DVD



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.



Authors	
Amber Ankerholz	8
Jens-Christoph Brendel	18
Florian Herzog	66
Ken Hess	3
Thomas Joos	42, 71, 74
Rainer Keller	28
Gerrit Klein	28
Philipp Koester	28
Ankur Kumar	78
Martin Gerhard Loschwitz	10, 22, 50
Federico Lucifredi	94
Benjamin Pfister	56, 90
Artur Skura	46
Evgenij Smirnov	84
Guido Söldner	36
Matthias Wübbeling	60, 64

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 Pettie, Aubrey Vaughn

Layout
Dena Friesen, Lori White

Cover Design
Lori White, Illustration based on graphics by Oksana Stepanenko, 123RF.com

Advertising
Brian Osborn, bosborn@linuxnewmedia.com
phone +49 8093 7679420

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 © 2022 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 Zeitfracht GmbH.

Distributed by Seymour Distribution Ltd, United Kingdom

ADMIN is published bimonthly by Linux New Media USA, LLC, 4840 Bob Billings Parkway, Ste 104, Lawrence, KS 66049, USA (Print ISSN: 2045-0702, Online ISSN: 2831-9583). September/October 2022. Periodicals Postage paid at Lawrence, KS. Ride-Along Enclosed. POSTMASTER: Please send address changes to ADMIN, 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.



FOSSLIFE

Open for All

**News • Careers • Life in Tech
Skills • Resources**

FOSSlife.org



4
TB

Storage Edition



Magnesium chassis
1.6 cm thin | 1.1 kg light



GeForce RTX 3050 Ti
Gaming & content creation

Premium business ultrabook goes workstation

TUXEDO InfinityBook Pro 14 - Gen7



Intel Core i7-12700H
14 Cores | 20 Threads



3K Omnia display
16:10 | 2880 x 1800 Pixel



Max size 99 Wh battery
for up to 16 hours



Magnesium chassis
1.7 cm thin | 1.3 kg light



100%
Linux

5

Year
Warranty



Lifetime
Support



Built in
Germany



German
Privacy



Local
Support

TUXEDO 18th
COMPUTERS ANNIVERSARY

tuxedocomputers.com