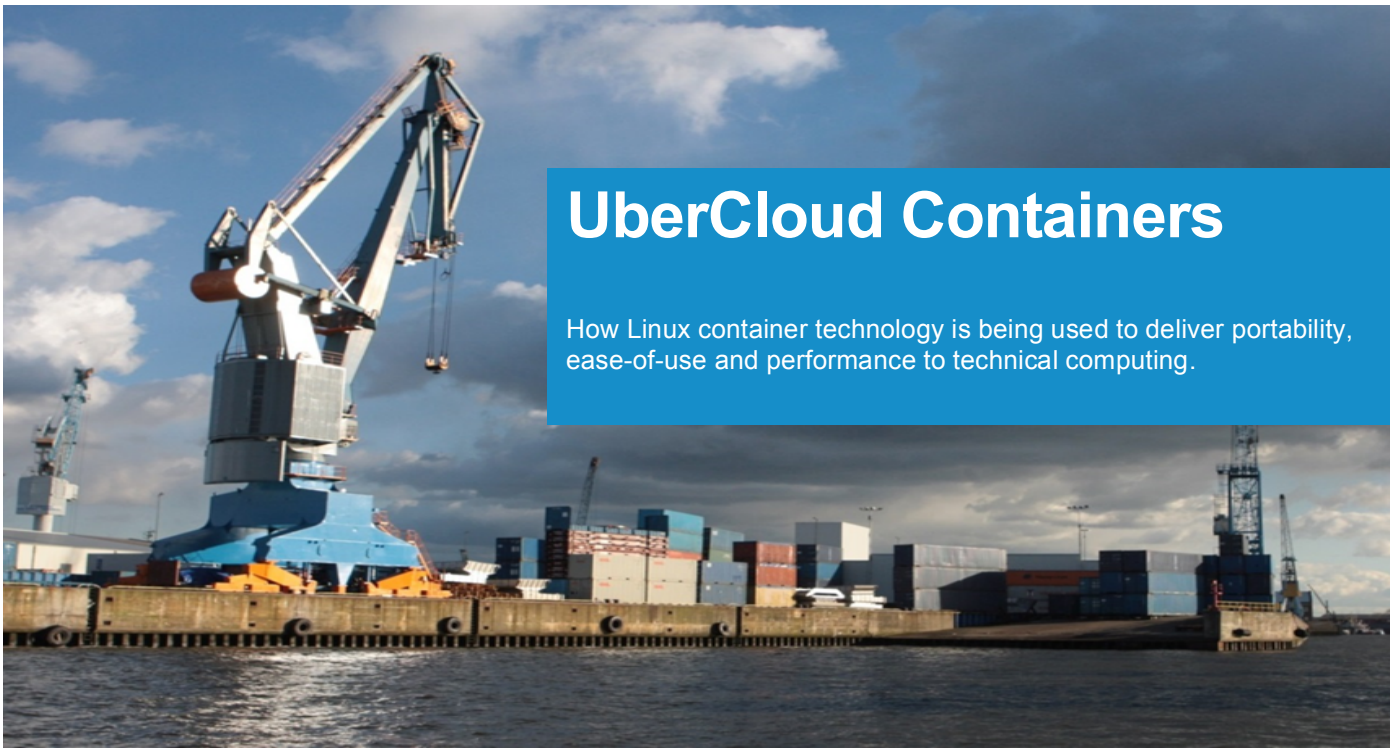


UberCloud Containers

How Linux container technology is being used to deliver portability, ease-of-use and performance to technical computing.

Burak Yenier and Wolfgang Gentzsch

AN UBERCLOUD TECHNICAL WHITE PAPER

The UberCloud, Los Altos, California, 2016

Table of Contents

UberCloud - A Brief Introduction 2

The UberCloud Community and UberCloud Experiments 2

The UberCloud Marketplace 2

Variety: The Key for a Lively Marketplace 3

UberCloud Containers: Brief Introduction 3

UberCloud Containers: The Underlying Technology 3

UberCloud Containers: What's Inside? 4

UberCloud Containers: Using Docker 5

The next step: UberCloud Multi-Node Containers 6

UberCloud multi-container environment 7

Typical Solution Architecture for On-premise and Cloud-bases Deployments 8

How to Set Up an UberCloud Application Container 9

UberCloud Containers: System Requirements 10

UberCloud Containers: A Summary of Advantages 10

UberCloud Containers: A Summary of Business Benefits 11

Security with UberCloud Containers 12

For More Information 13

ABOUT UBERCLOUD 14

UberCloud - A Brief Introduction

UberCloud is the online community and marketplace where engineers, scientists, and their service providers **discover, try, and buy** compute power as a Service, from computing clouds and even from Supercomputing Centers around the world.

The UberCloud has been launched in July 2012, by Burak Yenier and Wolfgang Gentzsch in Silicon Valley. The early idea was to explore the roadblocks of cloud computing for engineering and scientific applications and find solutions, with a crowd-sourcing approach, together with our engineering and scientific community.

The UberCloud Community and UberCloud Experiments

The UberCloud Community provides an online platform for scientists, engineers, and their service providers to explore, learn and understand the end-to-end process of accessing and using cloud resources, and to identify and resolve the roadblocks. An engineer can discover how early adopters benefit from advanced cloud technologies.

With the UberCloud Experiment, end-users, software providers, resource providers and computing experts collaborate in teams to jointly solve the end-user's application in the cloud. This set up provides the end-user with a realistic view of the benefits and best practices in a well-defined and structured experiment.

Since July 2012, the UberCloud Experiment has attracted nearly 3,000 organizations from 72 countries. It includes 175 teams in computational fluid dynamics (CFD), finite element method (FEM), biology and other domains, and tracked their experiences and lessons learned. Intel co-

sponsored the 2013, 2014, and 2015 case study collections, which have been publicized through industry media leaders like HPCwire, Desktop Engineering, and Bio-IT World, to name a few.

The UberCloud Marketplace

Building on the success of the UberCloud Experiments, UberCloud launched the Marketplace in late 2014. The UberCloud Marketplace brings technical computing infrastructure, application software and support providers under one roof.

The UberCloud Marketplace aims to serve a wide spectrum of verticals such as Computer Aided Engineering, Life Sciences, Finance, and Weather and Climate Research to name a few. The target customer profile is Small and Medium Businesses (SMBs) and even the 'long tail' of very small companies to individual consultants, which do research and development in these verticals. As of today, there are 31 stores on the marketplace in areas like software providers, resource providers, middleware providers, and experts.

As an example, an engineer can find offers that range from 16 CPU cores to 100's of CPU cores tied together with Infiniband on the marketplace. Offers that package together hardware, software and support can be purchased for a certain time (several hours to a few days or weeks) in a familiar online shopping experience.

Variety: The Key for a Lively Marketplace

Our vision for the UberCloud Marketplace is to offer quality, variety and selection to our end users. Offers ranging from Infrastructure as a Service (IaaS) to Software as a Service (SaaS) to Consulting and are presented side by side. Furthermore, there is a real need to support many hardware architectures, for example co-processors where needed by the computation task in hand.

Where variety becomes a real problem is in the software layer, however. Installing, tuning, maintaining, and supporting dozens and dozens of different application codes (ISV, Open Source, or proprietary in-house application codes) is a tough problem to tackle. The situation gets even more challenging if you need to support multiple operating systems. Add the layer of software version upgrades that arrive frequently and you get the picture.

Traditionally the solution applied by IT departments is to set standards, which is a nicer way of saying “limit the variety”. The end result of this approach are more stable and more manageable environments, which are also rigid and not very user focused. To solve this problem, UberCloud invested heavily in a core capability: **The UberCloud Containers**.

UberCloud Containers: Brief Introduction

UberCloud Containers are ready-to-execute packages of software. These packages are designed to deliver the tools that an engineer needs to complete his task in hand. The ISV or Open Source tools are pre-installed, configured, and tested, in the container, and are running on bare metal, without loss of performance. They are ready to execute, literally in an instant with no need to install software, deal with complex OS commands, or configure.

The UberCloud Container technology allows a wide variety and selection for the engineers because they are portable from server to server, cloud to cloud. The cloud operators or IT departments no longer need to limit the variety, since they no longer have to install, tune and maintain the underlying software. They can rely on the UberCloud Containers to cut through this complexity.

This technology also provides hardware abstraction, where the container is not tightly coupled with the server (the container and the software inside isn’t installed on the server in the traditional sense). Abstraction between the hardware and software stacks provides the ease of access and use and the agility that bare metal environments lack.

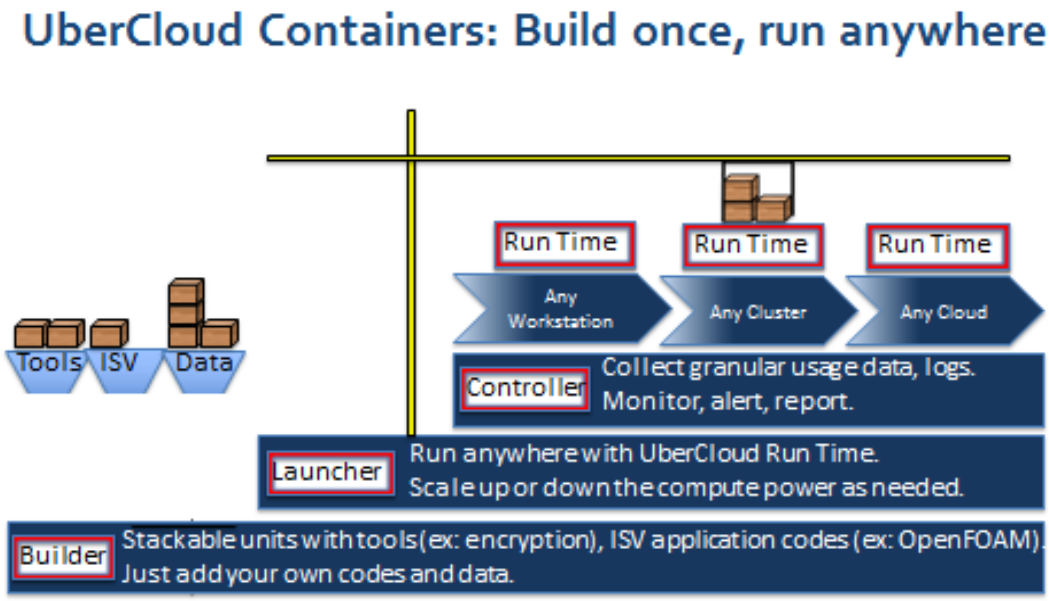
UberCloud Containers: The Underlying Technology

The UberCloud Containers rely on Linux kernel facilities such as cGroups, libcontainer, and LXC which are already a part of many modern Linux operating systems. The run time components to launch UberCloud Containers are distributed as a popular open source product called Docker and don’t require an additional capital investment.

UberCloud Containers are launched from pre-built images which are distributed through a central registry hosted by UberCloud. Software and operating system updates, enhancements, and fixes become instantly available for the next container launch in an automated fashion.

The notion of a pre-built image may sound familiar; this notion has been at the heart of virtualization, a popular technology for logically breaking down a physical computer environment into finer pieces. However, unlike virtualization, UberCloud Containers don’t rely on a hypervisor; instead, they share the host operating systems kernel and system level tools like NFS, leading to performance characteristics that are comparable to bare metal installations.

A closer look at this technology is available in an InfoQ magazine article by Chris Swan.



UberCloud Containers: What's Inside?

Short answer: everything that an engineer needs to complete a particular task on a remote server as if it was running on his desktop. Let's get into further details. UberCloud Containers come complete with:

- **An up-to-date Linux OS.** UberCloud supports a number of popular operating systems and makes its containers available in flavors such as Ubuntu, SUSE, and CentOS. The end user may select one or the other based on her own preference. Linux OS patches are applied periodically by UberCloud, distributed automatically and used when a new container is launched.

- **Libraries commonly needed by engineers**. A good example is MPI, which comes installed and configured inside the container when it's launched.

- **Utilities needed by engineers**. UberCloud provides a well thought out user experience and provides utilities such as screen sharing applications, diff utilities, and remote visualization applications.

- **Cloud data storage connectors**. Many IT departments are opening their doors to cloud data storage and collaboration tools, such as Box.com, and Amazon S3. UberCloud offers connectors to multiple providers, making data transfers from these services a breeze. In addition UberCloud supports privacy and security in cloud data storage using OwnCloud.

- **Engineering applications**. Engineers rely on sophisticated and highly specialized application software to perform their computations. An increasing number of these applications come pre-installed, tuned and tested within UberCloud Containers. A great example is the UberCloud OpenFOAM Container, which comes as a ready to run installation of the latest version of OpenFOAM (i.e. a free fluid dynamics application software). UberCloud is in collaboration with distributors of open source as well as commercial applications to further extend its list of supported applications. Other ISV applications currently supported include ANSYS, CD-adapco, Scilab, and Gromacs.

- **Administration layer**. Administrative tools such as automated password generation, self-service password change for increased privacy, performance checking, and log monitoring come bundled

with the UberCloud Containers. These tools provide the IT staff with an automated process to run and manage the containers.

- **UberCloud automated build/refresh process:** We understand how difficult it is to keep up with application software and operating system updates. Popular software vendors release updates to their products every few months and operating systems are updated every week. Many organizations are paralyzed by the difficulty of keeping a functioning environment while keeping up to date with all these changes. So at UberCloud we automated the process. Our containers are updated with the new versions of applications and operating systems, so when you are ready to launch your next job, you have the opportunity to update your software stack as well. Older versions are retained for continuity and reproducibility. UberCloud is able to trace every container back to its source control system for debugging and auditing purposes.

- **Many hours of testing.** As much as what's inside, we should talk about what isn't. Through hours of testing performed by multiple engineers, the UberCloud Containers evolve to contain less user experience problems and nuisances. The tests are performed using realistic models by a multidisciplinary testing team. The test results are fed right back to the development process, leading to a well-tuned, well tested user experience. UberCloud also developed an automated test suite which tests every new build against a comprehensive set of tests.

UberCloud Containers: Using Docker

As our UberCloud Containers we started mid 2013 using Docker and enhancing it to be well-suited for technical computing applications in science and engineering. Docker is a software that can package an application and its dependencies in a virtual container that runs on any Linux server. This helps enable flexibility and portability on where the application can run.

Docker is an open-source project (and a company) that automates the deployment of applications inside software containers by providing an additional layer of abstraction and automation of light-weight operating system–level virtualization on Linux. Docker uses resource isolation features of the Linux kernel such as cgroups and kernel namespaces to allow independent "containers" to run within a single Linux instance, avoiding all the heavy overhead of starting virtual machines.

Linux kernel's namespaces completely isolate an application's view of the operating environment, including process trees, network, user IDs and mounted file systems, while cgroups provide resource isolation, including the CPU, memory, block I/O and network. Docker includes the libcontainer library as a reference implementation for containers, and builds on top of libvirt, LXC (Linux containers) and systemd-nspawn, which provide interfaces to the facilities provided by the Linux kernel.

Adrian Cockcroft, technology fellow at Battery Ventures, answers the question of what does Docker do that is interesting, and who might it compete with? He comes up with four separate answers:

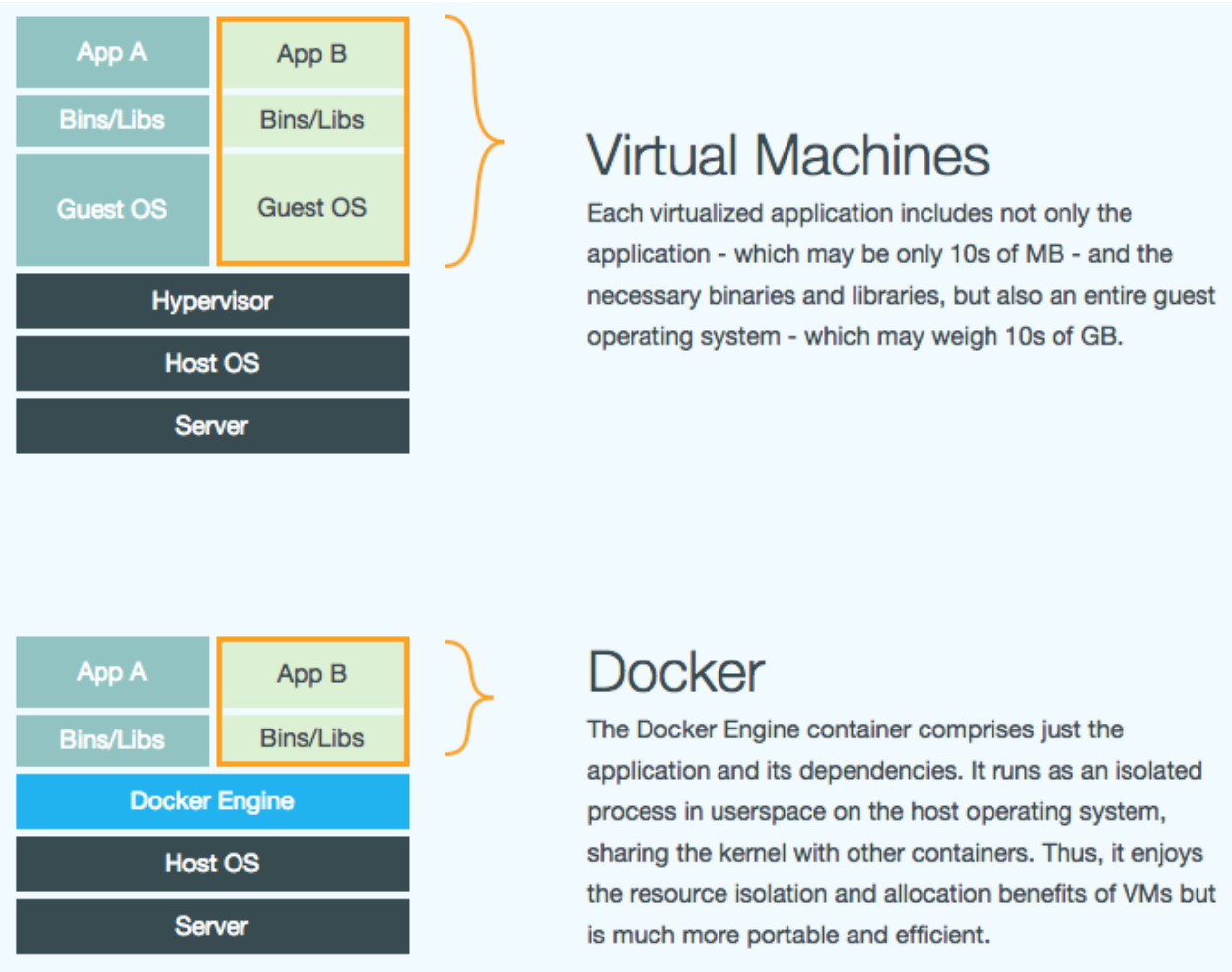
1. Portability: Docker is a portable container that packages any Linux application or service. A package that is created and tested on a developer's laptop using any language or framework can run unmodified on any public cloud, any private cloud or a bare-metal server. This is a similar benefit to the Java "write once, run anyware" idea but is more robust and is generalized to "build anything once, run anywhere".

2. Speed: Start-up time for a container is around a second. Public cloud virtual machines (VMs) take from tens of seconds to several minutes because they boot a full operating system every time, and booting a VM on a laptop can take minutes. To counter this advantage, VMware has just announced (but not shipped) a technology called Fargo that clones an existing VM in a second or so.

3. Configuration: The Docker container captures the exact configuration of a version of an application. To upgrade the application in production, the container is usually replaced with a new version, which takes a few seconds. The layers of components that go into the configuration are kept separate and can be inspected and rebuilt easily. This changes configuration management to be largely a build-time activity, so for example a Chef recipe might be used to build a Docker container, but at runtime there is no need to use the Chef services to create many identical copies of a Docker container.

Used in this way, Docker removes much of the need to use tools like Cfengine, Puppet, Chef, Ansible or Saltstack.

4. Docker Hub App-store: Docker containers are shared in a public registry at hub.docker.com. This is organized similarly to Github, and already contains tens of thousands of containers. Because containers are very portable, this provides a very useful cross platform "app store" for applications and component micro-services that can be assembled into applications. Other attempts to build "app stores" are tied to a specific platform (e.g. the AWS Marketplace or Ubuntu's Juju Charms) or tool (e.g. the Chef Supermarket) and it seems likely that Docker Hub will end up as a far bigger source of off-the-shelf software components, and monetization opportunities.



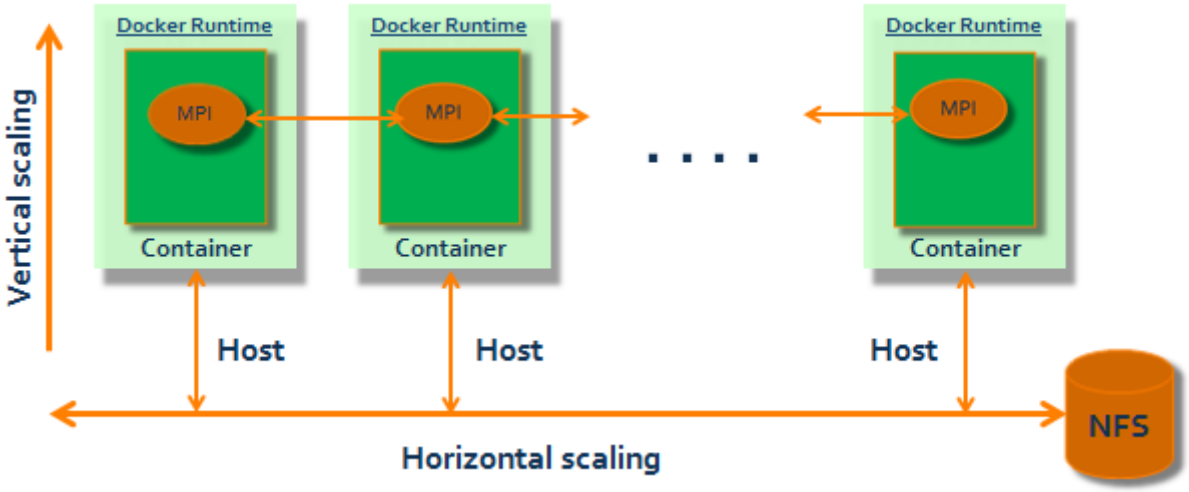
One reason Docker is interesting is that all four answers are each individually useful, but can be used in combination. This causes cross-pollination of ideas and patterns. For example, someone might start using Docker because they like the speed and portability, but find that they end up adopting the configuration and Docker hub patterns as well. The Docker technology is still fairly new; work is underway to add missing features, and a large ecosystem of related projects and companies is forming around it.

The next step: UberCloud Multi-Node Containers

Recently, Intel and UberCloud achieved a major breakthrough by porting UberCloud's single-node container to a multi-node HPC cluster containing Intel software tools like compilers and Intel MPI. We were able to maintain all single-node benefits described above also for the multi-node cluster which in this case was the Hyperion supercomputer at Lawrence Livermore National Lab. We have made sure that our containerized multi-node cluster now comes with the same improvements of portability, packaging, and ease of access and use, resulting in a multi-host HPC environment which scales horizontally across compute nodes and contains the Intel HPC developer tools and other HPC software components optimized for the Intel Architecture. With this breakthrough we believe that HPC is no longer just the domain of a few specialized HPC experts but offers benefits like easy access and use to a much wider community of engineering and

scientific end-users. What VMs are doing for enterprise IT, the UberCloud containers are now doing for technical computing.

UberCloud multi-container environment

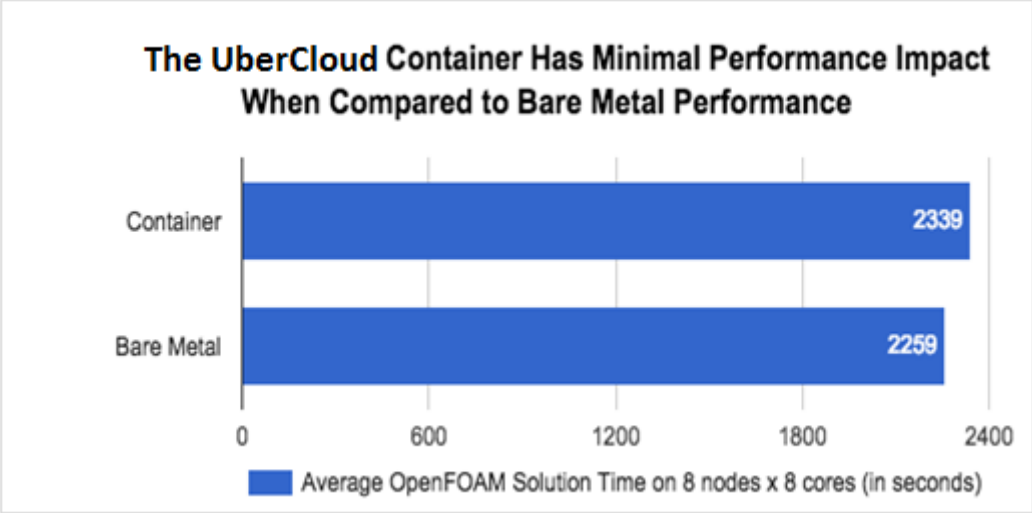


The UberCloud multi-container environment handles the following:

- Networking between containers
- Secure communications (ssh) between containers
- Setting up shared file system access (NFS)
- MPI libraries pre/setup / installation

Containers, unlike virtual machines, don't require a hypervisor. Therefore containers eliminate bottlenecks in computing and I/O to achieve bare metal-like performance, making them an ideal technology for running HPC applications. Further, each application shares components of the host operating system, making container images easier to store and transfer.

Performance tests conducted on the Intel Hyperion Cluster at Lawrence Livermore National Laboratory (LLNL) demonstrated that a medical device simulation with OpenFOAM application code running on the UberCloud Container achieved near bare metal solution times.



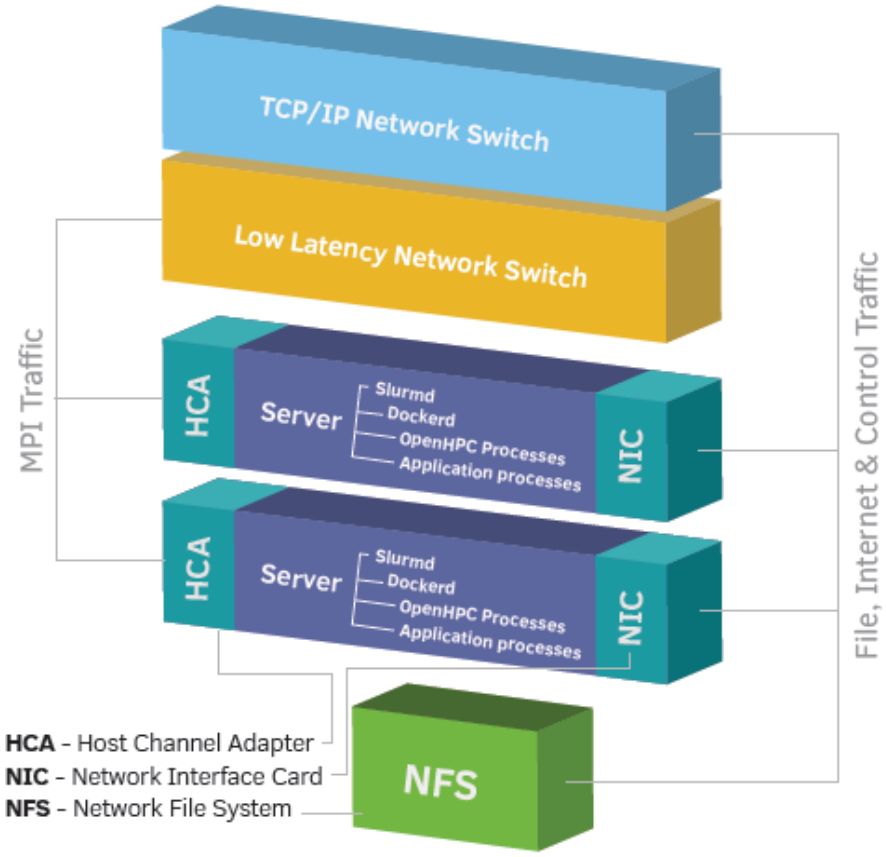
Typical Solution Architecture for On-premise and Cloud-bases Deployments

UberCloud containers are designed to be used in on-premise, private and public cloud environments, following a one container per host deployment model. In a typical on-premise deployment, UberCloud containers can be used together with existing HPC infrastructure, for example a resource manager (Slurm* in the sample diagram above).

In a typical on-premise deployment, there are often two network fabrics available. Servers connect to a TCP/IP Network Switch through Network Interface Controller (NIC) cards. This network is used for file transfers and control traffic. Similarly, servers and processors can connect to and utilize low latency interconnects (like Infiniband). This high performance network is for Message Passing Interface (MPI) communication by the application's processes.

UberCloud containers leverage the networking capabilities of the host operating system they are running on. To achieve low latency and high bandwidth communication UberCloud containers can take advantage of the Host Channel Adapter and communicate with other containers at the same bandwidth and latency characteristics as their host operating system. Using this feature MPI applications can perform in shared-memory as well as distributed-memory mode.

In this example, Slurm controls the compute resources of the cluster. Users queue their requests on to Slurm and Slurm assigns resources to the jobs, launches, monitors, and controls the related operating system processes. The Slurm Daemon (Slurmd) instantiates the container run-time (Dockerd) on each of the compute nodes allocated. Then, again Slurm, pulls the requested UberCloud Container Image from the UberCloud Private Container Registry and starts one UberCloud Container per host using this image. The containers are configured to automatically connect with each other and form a mini cluster, which supports MPI traffic using supplied drivers and libraries.



How to Set Up an UberCloud Application Container

The activities required to provide an ISV's APP in the container consist of a number of steps, which we can group into the following categories: host setup; container image development, testing and tuning; host tuning.

To set up a host, we start with a Cloud instance. This can be a virtual or bare metal instance; as long as it has a modern Linux kernel installed. We prefer instances with more than 8 CPU cores. We prepare the instance for our containers by installing our run time tools and running our fundamental tuning scripts to set up the for our run time tools.

We start with a base UberCloud container image. The UberCloud base container image (which we have developed over the last 12 months) gives us a list of capabilities out of the box, for example, we enable rapid file transfers and instant remote visualization. All UberCloud ISV APP container images share these capabilities that they inherit from the UberCloud base container. Our base container is under constant development to add features to improve the engineering cloud user experience.

Container image development is an iterative process. We use our build time tools to "record" the installation steps needed for ISV's APP into a build script. We install required libraries, set recommended parameters, load ISV's binaries. We stick to the installation instructions provided by the ISV.

Once we have the "recoded" build script our build time tools can regenerate container image (build on top of the UberCloud base container image) at any time without human supervision. This is useful in the next step, which is testing. We apply fixes to our build script to achieve a properly working container image of the ISV's APP. We put the container image through a number of tests by third party testers and tune the build script based on their feedback.

The container image that results from this process is portable to any cloud where we have our runtime tools installed. The container images launch instantly on our run time tools with no need for server administration on the host.

There are times when the tuning within the build script just isn't adequate. In such cases we tune our host cloud server as well.

The amount of effort required for each of these steps vary greatly based on the complexity of the ISV's APP, the quality of the documentation, and the amount of issues we encounter. We have developed a repeatable process that ensures our success at the end of the cycle.

Our sponsors can take a closer look at our technology direction under NDA.

Any host that is capable of running docker can run UberCloud containers as well. At highest level, that would translate to hosts with kernel 3.10 and later.

Every new major version of docker provides backward-compatibility and only enable new features if underlaying host supports them. I expect this to be valid until Docker v2.0. Most of these features bring performance improvements like more performant storage driver or improve security like more fine grained isolation between host and containers. UberCloud containers does not depend on any of these features instead utilize them when available like docker does.

UberCloud Containers: System Requirements

The UberCloud containers (the same as with Docker) currently run on Linux (we are working on the Windows version). In the following we list the requirements for hosts to handle UberCloud containers.

Minimum requirements:

- Kernel 3.10+
- Recent Docker (any Docker released after March 2016, which is 1.10.3+)
- 2 TCP ports per container (accessible by the end-user)
- if multi-node, also NFS share (or equivalent)

Preferred requirements:

- Kernel 4.0+ (utilizes all available Docker features)
- Latest stable Docker (as of now 1.12.x)

Additional requirements of specific ISV application(s) should be added to the list above separately; most of the time, an ISV application software requires decent amount of CPU, memory and network bandwidth.

UberCloud Containers: A Summary of Advantages

- **Packaging:** Package once, run anywhere !
- **Portability:** You can run UberCloud containers in your infrastructure with minimal modification. The required run time environment is distributed as open source, is well documented and is supported by a large community of users.
- **Easy access and use:** Because all applications and tools are already packaged and available at the user's fingertips UberCloud Containers can be easily launched from pre-built images which are distributed through a central registry hosted by UberCloud. Software and operating system updates, enhancements, and fixes become instantly available for the next container launch in an automated fashion.
- **Manageability:** UberCloud manages the contents of the containers and keeps them up-to-date for you; keeping your installation, tuning, maintenance, testing costs to a minimum.
- **Variety:** UberCloud is constantly adding engineering applications, tools and operating systems to its portfolio. Variety of operating systems and software stacks can be hosted side-by-side.

- **Low overhead:** Compute resources require a significant capital investment and engineers want to squeeze every bit of performance out of them. The UberCloud containers rely on light-weight Linux container technology, providing a low overhead profile.
- **Instant provisioning:** UberCloud Containers start within seconds, with a single command. Short provisioning times ensure end-users receive the resources they need when they need them.
- **High utilization:** Multiple containers can be run on a single server if the individual user jobs require a small amount of resources.
- **Audits:** UberCloud develops its containers with a process that's easily understandable by any Linux user. You may perform IT audits of the components, configurations, and security settings of the UberCloud Containers.
- **Reproducibility:** UberCloud Container images are immutable, meaning they cannot be altered. We retain container images so that computations done inside an UberCloud container become reproducible at a later date.

CAE Cloud Challenges	UberCloud
Security	√
Portability	√
Compliance	√
Data Transfer	√
Standardization	√
Software Licenses	√
Resource Availability	√
Transparency of Market	√
Cost & ROI Transparency	√
No Cloud Expertise Needed	√

UberCloud Containers: A Summary of Business Benefits

Benefits for the end-user:

- Portability: any cloud looks like the user's workstation
- User-friendly: nothing new to learn, ease of access and use
- Control: container monitoring allows the user to control his assets in the cloud.

Resource provider:

- Getting variability into their environment. Customers want different products which is easily implemented with container packaging and stacking
- Low overhead
- High utilization

ISV Benefits:

- Portability, their software can run on a variety of different resource providers, built once, run anywhere
- Control of software usage through container based license and usage monitoring, and control of user experience
- The faster the software runs the better the user experience; containers enable porting of the software to workstations, servers, and to any cloud.

Security with UberCloud Containers

You are in control

UberCloud works with your security and compliance experts to pick your resources, whether it be a cloud provider or your own corporate datacenter. With this flexibility, you can now get both great performance and be in control of your information assets.

Data encryption

UberCloud enables encryption for data transfers, shell access, remote desktop access, VPN and data storage. You control your data encryption requirements to keep your data secure and private.

Logical Security

UberCloud, runs in private compute environments where your data is stored in your Cloud storage account, compute resources are dedicated and not shared. UberCloud containers are deleted when your processing is complete.

Physical security

UberCloud allows you to deploy on professionally managed Clouds, with stringent physical security controls for their assets. Some of these physical security controls are: biometric entry authentication and armed guards.

Fine grained access control

UberCloud provides full flexibility in defining your access control requirements. Your IT organization defines on the right set of firewall rules, authentication methods, and which employee gets access to which compute or data resource.

Additional compliance and governance requirements

We understand your need to know that your data is secure, we've been there. UberCloud leadership team is experienced in regulated industries and we are ready to support your additional compliance requirements.

For More Information

You can refer to the following sources for more information. If you'd like to contact us please email Wolfgang Gentzsch at wolfgang.gentzsch@theubercloud.com.

1. The UberCloud Website: https://www.TheUberCloud.com
2. UberCloud: Discover, Try, Buy: https://www.theubercloud.com/discover-hpc-as-a-service/
3. Chris Swan: Docker: Present and Future. http://www.infoq.com/articles/docker-future
4. Docker: http://en.wikipedia.org/wiki/Docker_(software)
5. The UberCloud Marketplace with samples of containerized application solutions: https://www.TheUberCloud.com/Marketplace
6. Adrian Cockcroft: Why Did Docker Catch on Quickly and Why is it so Interesting? http://thenewstack.io/why-did-docker-catch-on-quickly-and-why-is-it-so-interesting/
7. Seth Feder: What does Docker have to do with the Cloud? ow.ly/C3h20
8. Sudhi Seshachala: Docker and cloud security. http://cloudtweaks.com/2015/01/docker-changes-cloud-security/
9. Wolfgang Gentzsch and Burak Yenier: Linux containers simplify engineering and scientific simulations in the cloud. Submitted (and accepted) to the Global Online Conference on Information & Computer Technology, CICT'14, Dec 3-5 2014.
10. Wolfgang Gentzsch: Workstations – Servers – Clouds – Comparing apples to apples. A little decision-making support for the undecided. http://www.theubercloud.com/workstations-servers-clouds-comparing-apples-apples/

ABOUT UBERCLOUD

UberCloud makes it easy to run your simulations on powerful cloud infrastructure.

No more compromises on mesh quality or model fidelity because of hardware limitations. With UberCloud's flexible software platform and network of cloud partners, you get on-demand access to major providers such as Microsoft Azure, HPE and others. Choose from a variety of secure data centers, and hardware options such as InfiniBand, GPUs etc.

Unleash the full power of your analysis software and boost confidence in your results.

With over 200 technical-computing-as-a-service case studies, UberCloud has the experience, software platform and partnerships required for your success.

Engineers and scientists rely on UberCloud to manage the complexity of cloud and software operations, so they can focus on their analysis.

FOR MORE INFORMATION

440 N Wolfe Road
Sunnyvale, CA

help@theubercloud.com
www.theubercloud.com