# Using Docker containers for photon experiment simulations in HPC environments

Sergey Yakubov, Carsten Fortmann-Grote, Yves Kemp, Frank Schlünzen

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 654220

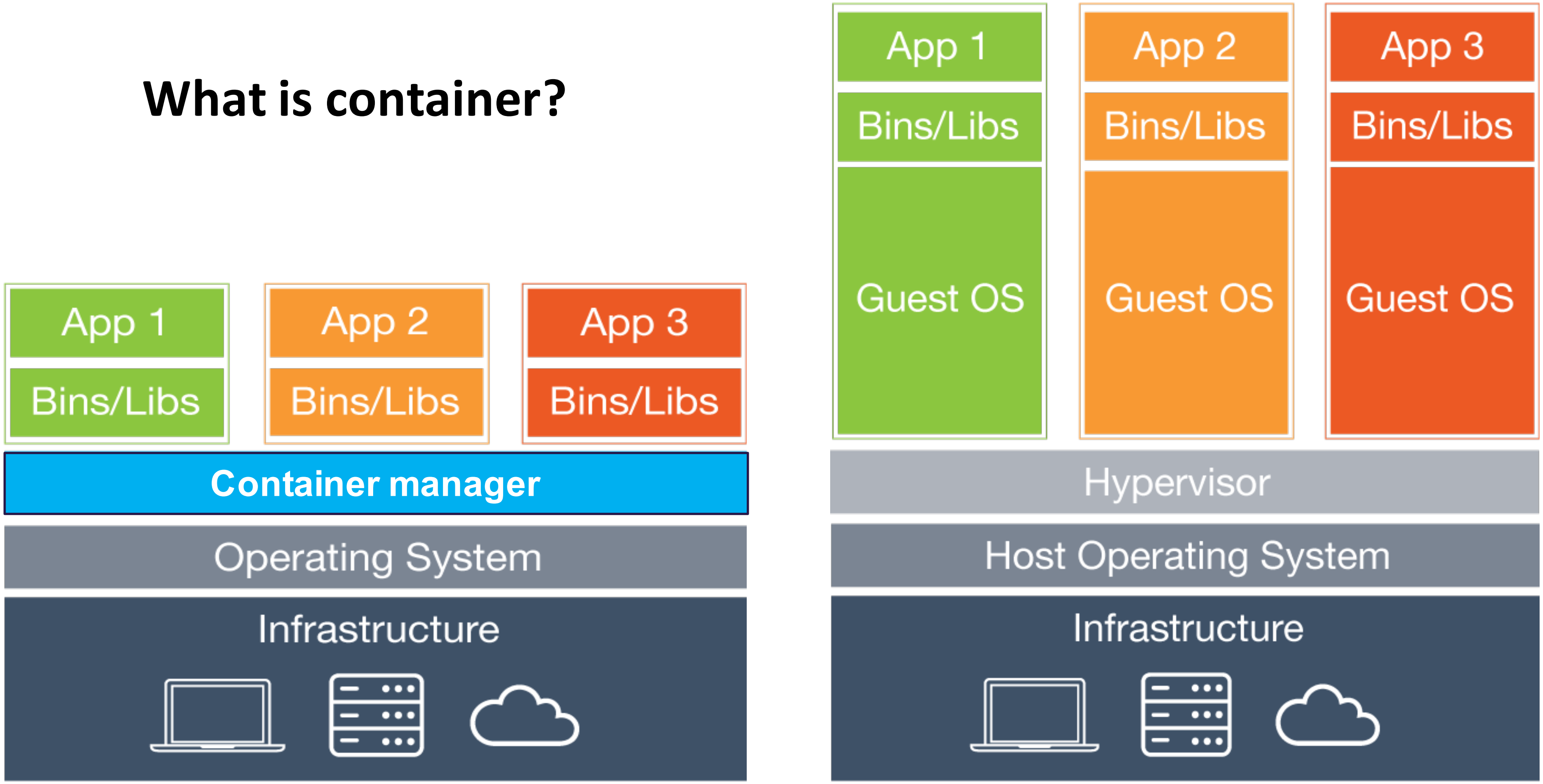
# Outline

- Introduction
- DESY HPC cluster (Maxwell)
- Docker in HPC cluster environments
- SIMEX HPC simulations with Docker
- Conclusions

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 654220

# Introduction

## What is container?



**Container**

**Virtual machine**

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 654220

# Introduction

## Why container?

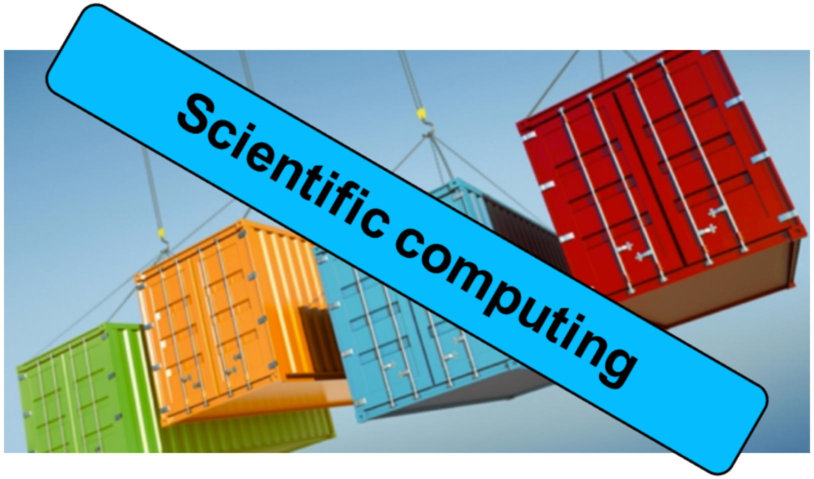
- Lightweight
- Low overhead
- Micro-services
- Service orchestration
- Software development/testing
- Software deployment

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 654220

# Introduction

## Why container?

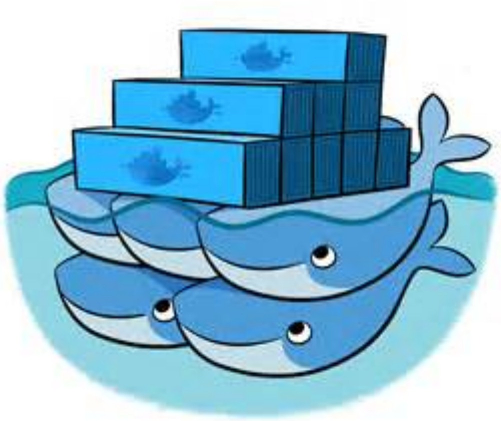
- Lightweight
- Low overhead
- Micro-services
- Service orchestration
- Software development/testing
- Software deployment



This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 654220

# Introduction

## Why Docker?

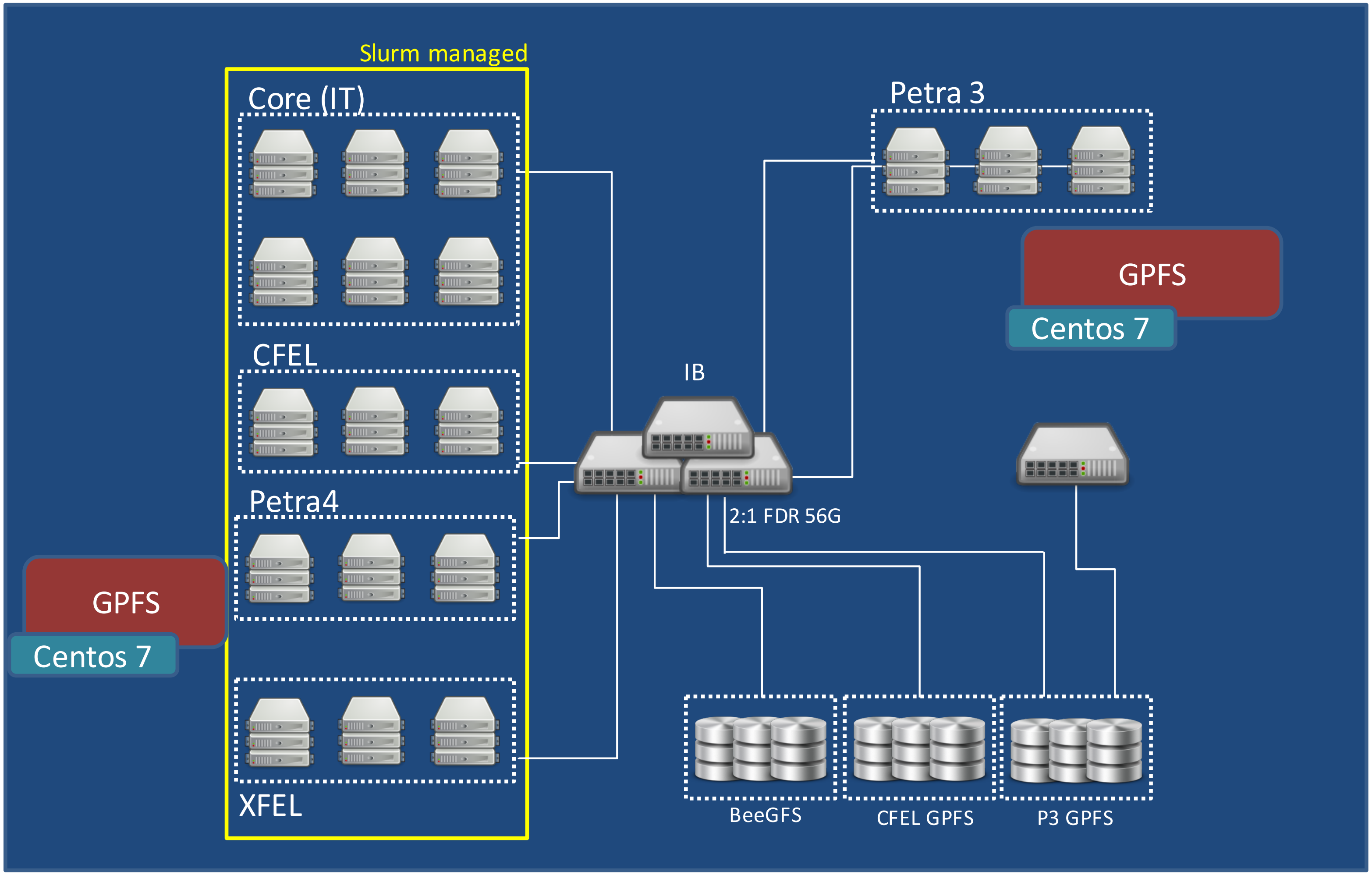


- Open-source
- 1,500 contributors
- Commercial support
- Docker hub to store images
- Can be used everywhere (well, almost)

Alternatives – LXC, rkt?

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 654220

# Maxwell HPC Cluster



This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 654220

# Docker for Maxwell

- For each job we create an HPC cluster of Docker containers
  - Secure (no root access for user)
  - High-speed network
  - Parallel file system
  - Deployed using SLURM
  - User friendly

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 654220

# Docker for Maxwell - Security

- Until February 2016 there was a serious lack of security
  - User ID inside a Docker container matched user ID on the host system
    - → root access inside Docker = root access to host
    - → cannot use 3rd-party containers
    - → cannot allow user to execute Docker commands
- Since version 1.10 kernel user namespace can be used
  - User ID and Group ID are isolated inside a container
    - → Experimental kernel parameter in RedHat and Co. (available since version 7.2)

```
--enable-user-namespace=1
```

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 654220

# Docker for Maxwell - Network

- For each job we create an HPC cluster of Docker containers

```
$ docker run -d <.......> centos_mpi_benchmarks
```

- Using host network

```
--net=host
```

**Insecure (does not support user namespaces)!**

- Using default bridge network
  - ➔ Add infiniband devices

```
--device=/dev/infiniband/uverbs0 --device=/dev/infiniband/rdma_cm
```

  - ➔ Pass IPoIB interface to a container

```
pipework ib0 <docker name> <ip address>/24
```

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 654220

# Docker for Maxwell – Parallel Filesystem

- For each job we create an HPC cluster of Docker containers

```
$ docker run -d <.......> centos_mpi_benchmarks
```

- Sharing a folder in a parallel filesystem

```
-v /home/jdoe/test:/shared
```

- User namespaces should be respected by the filesystem
  - nfs
  - gpfs
  - beegfs

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 654220

# Docker for Maxwell – File Permissions

- Since we use user namespace
  - Shared folder should have read (write) permissions for everyone
    - → alternative – stage data
  - After job is finished – ownership of the files created in the shared folder must be changed
- Alternatively – trusted images can be started without user namespaces
  - What is trusted image?
  - Docker authorization plugin is used for extra security

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 654220

# Docker for Maxwell – Image Repositories

- DESY's repositories
  - Read-only repository
    - No user namespaces
    - User cannot upload images
    - docker exec –u root - not allowed
  - Open (to DESY users) repository
    - User namespaces are active
    - User can upload images
    - docker exec –u root - allowed
- Dockerhub
- Third-party repositories (certified)

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 654220

# Docker for Maxwell - Workflow

## Docker in DESY HPC environment

- User submits a job to a resource management (SLURM)

```
…
#SBATCH –comment="use_docker;max-adm01:5001/centos_mpi_benchmarks;
/home/yakubov/container_shared:/shared“
…
```

- SLURM puts the job in a common queue
- As soon as resources are available, SLURM starts a container on each of the allocated nodes (using prolog script)

```
docker run -d \
-v $DOCKER_HOST_PATH1:$DOCKER_CONTAINER_PATH1 …  \
--name=docker_$SLURM_JOB_ID \
$DOCKER_IMAGE
```

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 654220

# Docker for Maxwell - Workflow

## Docker in DESY HPC environment

- And creates a virtual network (SLURM daemon runs as root)

```
/root/bin/pipework ib0 docker_${SLURM_JOB_ID} ${mask}.${nnode}/24
```

- User sets-up job steps to be executed (in a script or interactively)

```
docker_run  simex.py
docker_mpirun -n 32 simex.py
```

- SLURM removes all containers using epilog script after the job is finished

# Examples - MPI Bandwidth and Latency Tests

- Two Maxwell compute nodes, Mellanox Infiniband 56 Gbs (4X FDR)
- Host system vs Docker container
  - ib utilities

| | Host system | Docker |
|---|---|---|
| ib_send_bw | 44 Gbs | 46.9 Gbs |
| ib_send_lat | 1.1 µs | 1.07 µs |

  - mpi_benchmarks (source: Lawrence Livermore National Laboratory)

| | Host system | Docker |
|---|---|---|
| mpi_bandwidth | 45.7 Gbs | 44.9 Gbs |
| mpi_latency | 1.99 µs | 1.99 µs |

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 654220

# Examples – HPCG/HPL Benchmarks

- High-Performance Linpack Benchmark
  http://www.netlib.org/benchmark/hpl
- High Performance Conjugate Gradients
  http://hpcg-benchmark.org
- Both used by Top500 (officially/unofficially yet)

| HPCG rank | Cores | Top rank | HPL (PFlops) | HPCG (PFlops) |
|---|---|---|---|---|
| NSCC Tianhe-2 | 3 120 000 | 2 | 33.86 | 0.58 |
| RIKEN K computer | 705 024 | 5 | 10.51 | 0.46 |
| DOE Titan | 560 640 | 3 | 17.59 | 0.32 |
| HLRS Cray XC40 | 185 088 | 9 | 5.64 | 0.14 |

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 654220

# Examples – HPCG/HPL Benchmarks

## Benchmark results on Maxwell cluster

| | Cores | HPL (TFLops) | HPCG (TFLops) |
|---|---|---|---|
| Maxwell | 64 (2 nodes) | 1.56 | 0.033 |
| Maxwell+Docker | 64 (2 nodes) | 1.56 | 0.033 |
| Maxwell | 368 (15 nodes) | 9.0 | 0.192 |
| Maxwell+Docker | 368 (15 nodes) | 9.0 | 0.192 |

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 654220

# SIMEX on Maxwell - Deployment

## SIMEX (see presentation from Carsten Fortmann-Grote)

- Deployment is non-trivial
  - Each calculator has its own dependencies and install script
  - Need to install in various environments
  - Users can have admin/non-admin rights
  - Experienced developers/unexperienced users
- Possible solutions
  - Use CMake build system
  - Use binary packages (deb, rpm, …)
  - Use Docker containers

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 654220

# SIMEX on Maxwell - Deployment

## Deployment using Docker containers

- SIMEX image is on the Docker hub
- Everything is installed inside the image
- To start working with it just type

```
docker run –it simex bash
```

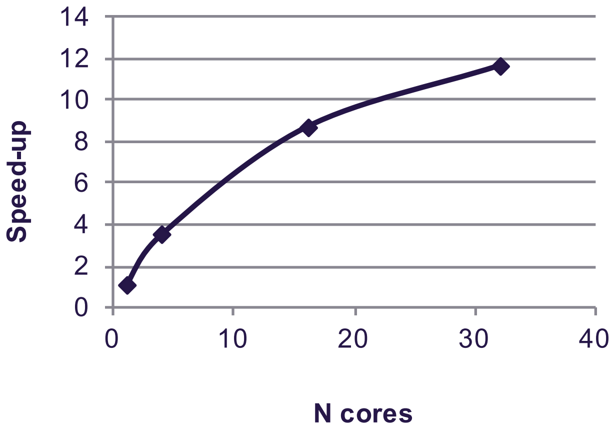
- Or submit a job with python script to SLURM

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 654220

# SIMEX on Maxwell – Performance Results

## X-ray wavefront propagation calculator

- Propagation of light through optical elements
- Utilizes SRW (Synchrotron Radiation Workshop) library
- C++ core + python wrappers
- Hybrid OpenMP/MPI parallelization



Single source file

| Threads x MPI processes | Number of nodes | Total time | Time/file |
|---|---|---|---|
| 1x1 | 1 | 11h | 1031 s |
| 40x1 | 1 | 65 min | 98 s |
| 4x10 | 4 | 7.5 min | 45 s |
| 8x5 | 8 | 4.2 min | 51 s |

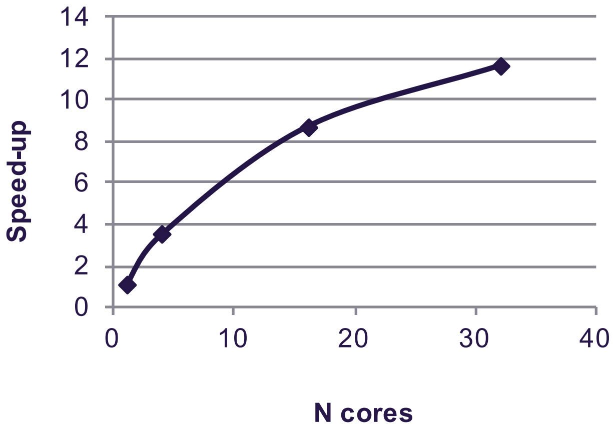
40 source files

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 654220

# SIMEX on Maxwell – Performance Results

## X-ray wavefront propagation calculator

- Propagation of light through optical elements
- Utilizes SRW (Synchrotron Radiation Workshop) library
- C++ core + python wrappers
- Hybrid OpenMP/MPI parallelization



Single source file

| Threads x MPI processes | Number of nodes | Total time | Time/file |
|---|---|---|---|
| 1x1 | 1 | 11h | 1031 s |
|  |  | 65 min | 98 s |
| 4x10 | 4 | 7.5 min | 45 s |
| 8x5 | 8 | 4.2 min | 51 s |

160x speed-up

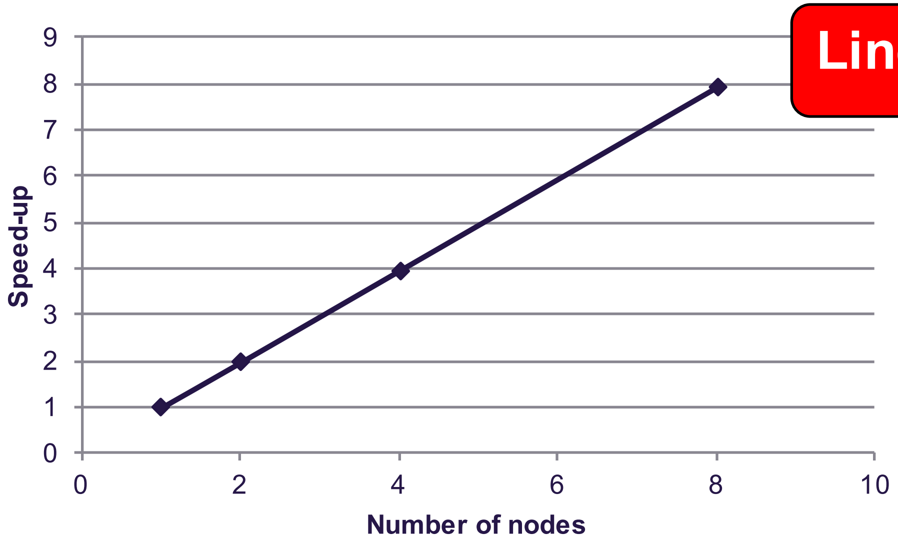
40 source files

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 654220

# SIMEX on Maxwell – Performance Results

## Photon diffraction calculator

- Absorption, emission, and scattering of radiation
- Utilizes SingFEL photon diffractor library
- C++ core + python wrappers
- MPI parallelization



Linear speed-up

200 000 diffraction patterns

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 654220

# Conclusions

- Running Docker containers on an HPC cluster is possible and
    - does not break system security
    - does not introduce overhead
    - uses general resource scheduling procedures
- Simplifies software development and deployment
    - Can be developed and compiled off-site and deployed instantly on a cluster
- Photon experiment simulations can be efficiently performed on an HPC cluster using “dockerized” SIMEX platform.

This project has received funding from the *European Union’s Horizon 2020 research and innovation programme* under grant agreement No 654220

# Thank you for your attention!

This project has received funding from the *European Union’s Horizon 2020 research and innovation programme* under grant agreement No 654220