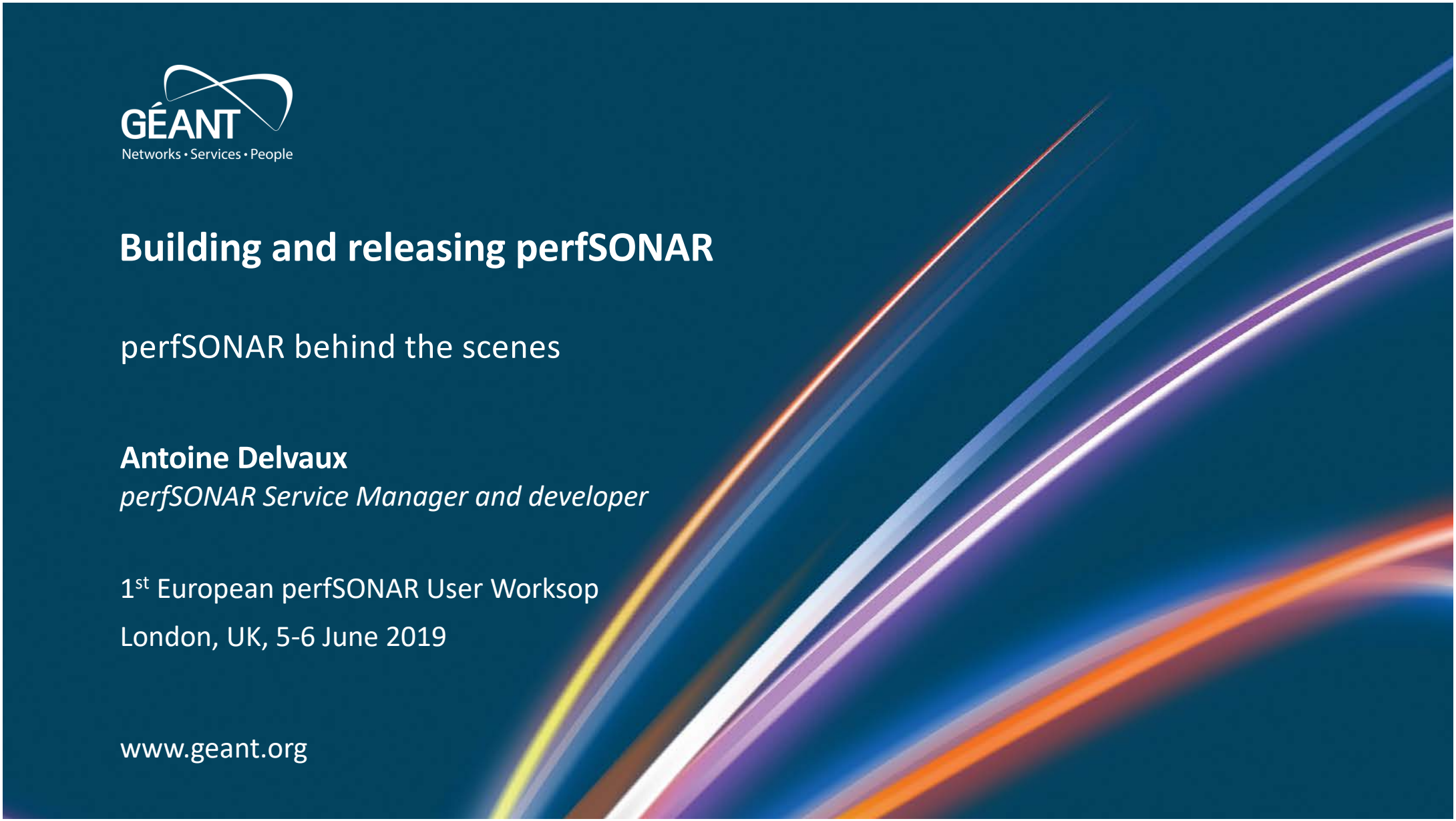



# Building and releasing perfSONAR

perfSONAR behind the scenes

**Antoine Delvaux**
*perfSONAR Service Manager and developer*

1st European perfSONAR User Worksop

London, UK, 5-6 June 2019

www.geant.org

## perfSONAR behind the scenes

- Development team
- Source Code
  - Languages
  - Management
  - Git branching model
- Package builds
  - CentOS
  - Debian and Ubuntu
- Continuous Integration

# perfSONAR development team

- About 15 persons involved
- But only 5 FTE
- Each partner organisation is contributing 1 FTE
- GÉANT project:
  - CARnet: 1
  - DFN / Uni Erlangen: 1
  - GÉANT Association: 1
  - GRENA: 2
  - Jisc: 1
  - PSNC: 2
- Development, but also support and documentation writing

## perfSONAR Source Code

- Lots of different programming languages:
  - Perl
  - Python
  - SQL and DB programming
  - Bash
  - Javascript
  - But also Java and even NodeJS
- Why? Because different people and long history!
- About 15 years of development

## Source Code Management

- Git repositories
  - Hosted in a neutral and maintenance free space: https://github.com/perfSONAR/
  - One repository per piece of software (set of packages)
- Issue tracker
- Integrated wiki for developers documentation
- Scrum / Kanban board (ZenHub)

## Git branching model

- One branch for each release
- Usually only 2 concurrent release at the same time:
  - Major: 4.2
  - Bug fix: 4.1.7
- Easier to deal with bug fix on current release while developing next point release
- Topic branch for feature development, merged into release beach when ready

# Package builds: CentOS

- RPM
  - Makefiles
  - RPM spec files
- Built and released for CentOS 7
- One architecture: x86_64
- Developers can build on Vagrant machines
  - Shared VM config

# Package builds: Debian and Ubuntu

- .deb and usual Debian packaging files
- Built on Debian 9 (Stretch)
  - Using Vagrant VM too
- 4.1
  - for Debian 8 and 9 and Ubuntu 14, 16 and 18
  - 5 architectures
- 4.2
  - will be for Debian 9 and Ubuntu 16 and 18
  - Aiming at 6 architectures (ppc64el)

## Debian and Ubuntu builds: some numbers

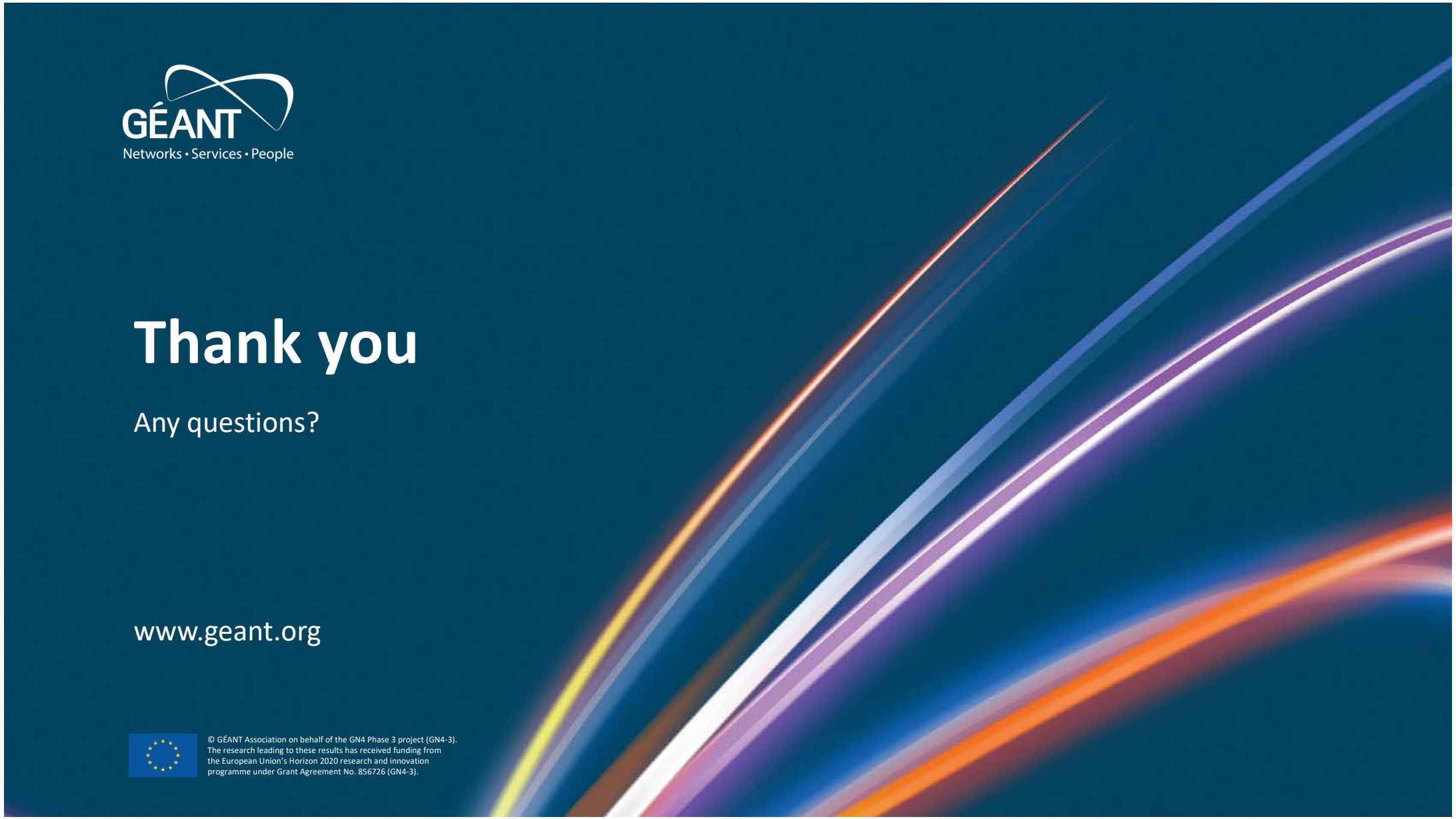
- 32 source packages
  - Including backported dependencies
- 161 binary packages
- One build machine with 22 build environments
- Takes 2 to 3 hours to build all Debian packages

## Continuous Integration

- Building for each commit on the release branch
- Can then be deployed on Dev test mesh
- Aiming at building before merging into release branch
- Setup in Jenkins with link to GitHub
- Repositories:
  - snapshot/nightly: latest builds
  - staging: collect all packages before release
  - release: the actual packages you install
  - One pair of snapshot/staging repo per supported release
    - patch-snapshot and patch-staging
    - minor-snapshot and minor-staging

# Preparing a Release

- First tested by developers on dev test mesh
  - Try to have as many different nodes as possible
  - OS, pS version, toolkit/testpoint
- Beta release for users
  - Tested for a few weeks
  - Correcting all pending and newly found bugs
  - Incrementally releasing corrected packages
- Release Candidate (RC)
  - Have 2 weeks without issue before final release
  - As many as needed




© GÉANT Association on behalf of the GN4 Phase 3 project (GN4-3).
The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 856726 (GN4-3).