



Updated Version Numbering Scheme and Release Planning

Open MPI Project
June 2015

Table of contents

- These slides cover several related topics:
 1. Open MPI's new version numbering scheme
 2. Transition plan to the new version numbering
 3. Release planning roadmap
 4. The bottom line (TL;DR)
- Let's jump right in...

Before July 2015...

- Open MPI used an “odd / even” numbering scheme
 - 1.odd: “feature” series
 - 1.even: “stable” series
- But it’s not working out as well as we’d like

One problem

- Very few users actually use the “odd” versions
 - Users equate “odd” with “unstable”
- As a direct result:
 - New features don't get real-world tested
 - ...until the “even” releases

Another problem

- Users want new features faster
 - A “stable” series (intentionally) does not receive new features
- As a direct result:
 - New features take a long time to get to users



Let's fix that



Goodbye odd / even scheme!

New version numbering scheme

- Open MPI will (continue to) use a “**A.B.C**” version number triple
- Each number now has a specific meaning:
 - A** This number changes when backwards compatibility breaks
 - B** This number changes when new features are added
 - C** This number changes for all other releases

Examples

- Pretend we're in the future
 - The current Open MPI release is **v3.4.2**
- What will be the next release number?
- Let's look at a few cases...

Example 1

- Current release: **v3.4.2**
 - Situation:
 - Bugs are fixed
 - No new features are added
 - Backwards compatibility is preserved
- Next release will be **v3.4.3**

Example 2

- Current release: **v3.4.2**
 - Situation:
 - Bugs are fixed
 - User-noticeable new features are added
 - Backwards compatibility is preserved
- Next release will be **v3.5.0**

Example 3

- Current release: **v3.4.2**
- Situation:
 - Major changes occur (new features, etc.)
 - Backwards compatibility is broken

→ Next release will be **v4.0.0**

Wait...

How exactly are you defining the term

**“backwards
compatibility”**

?

Definition

- Open MPI v Y is backwards compatible with Open MPI v X (where $Y > X$) if:
 - Users can compile a correct MPI / OSHMEM program with v X
 - Run it with the same CLI options and MCA parameters using v X or v Y
 - The job executes correctly

What does that encompass?

- This definition covers several areas:
 - MPI / OSHMEM API ABI
 - MPI / OSHMEM run time system
 - `mpirun` / `oshrun` CLI options
 - MCA parameter names / values / meanings

Versioning note

- Open MPI only supports running exactly the same version of the runtime and MPI / OSHMEM libraries in a single job
 - If you mix-n-match different versions in a single job...

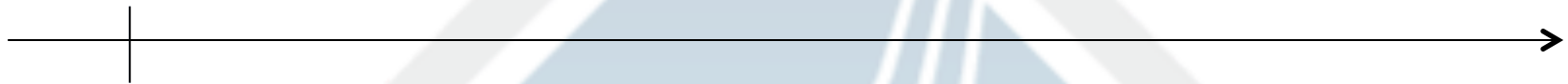




Transition to the New Version Numbering Scheme

How to move to the new numbering?

v1.8.6



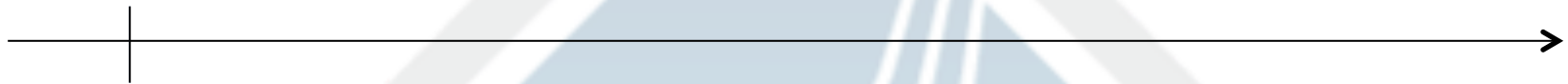
Released
June 19, 2015

How to move to the new numbering?

What's next?

v1.8.6

Released
June 19, 2015



How to move to the new numbering?

What's next?

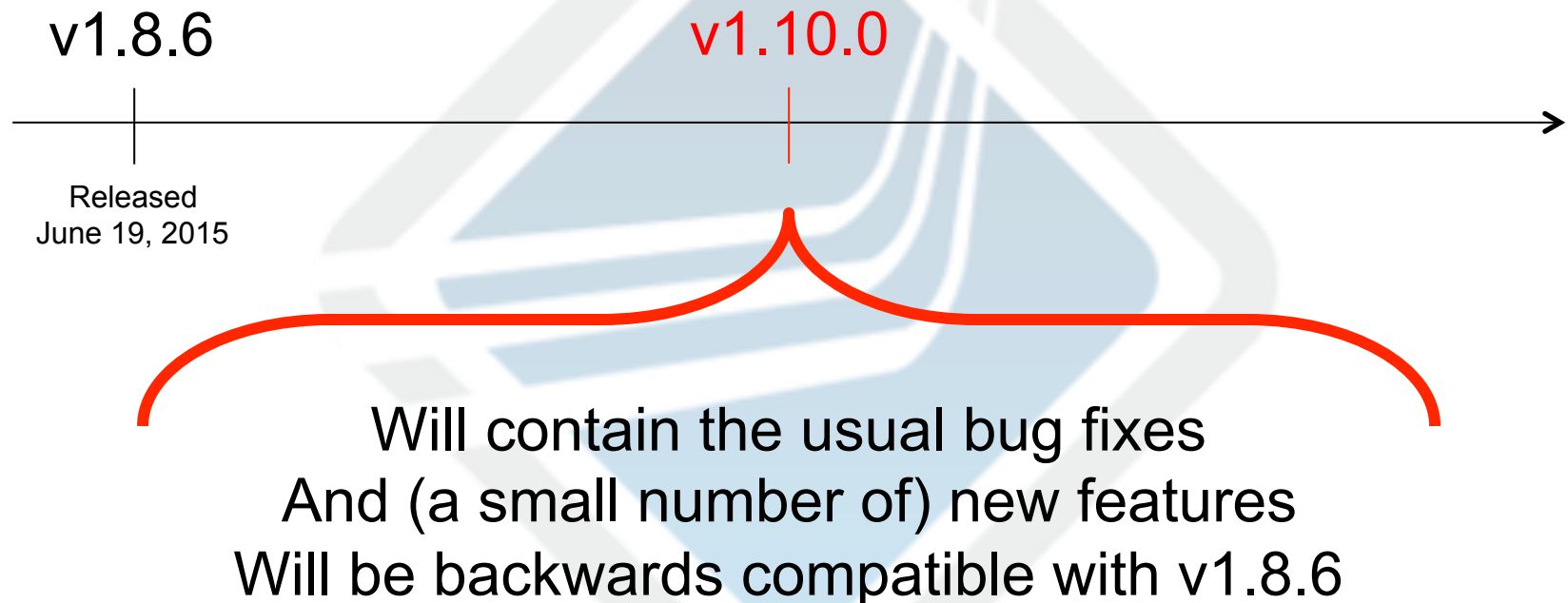
v1.8.6



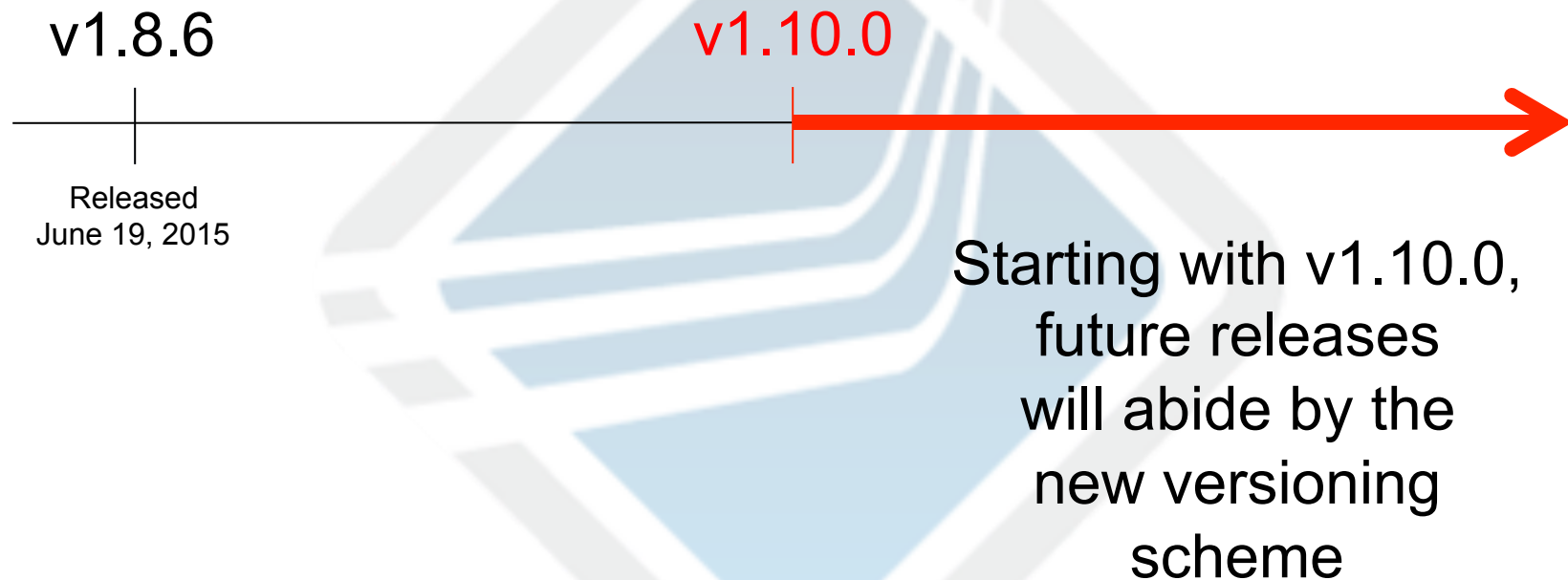
Note: it would be crazy confusing to change the version number scheme in the middle of the v1.8.x series.

We won't be doing that.

How to move to the new numbering?



How to move to the new numbering?





Release Planning Roadmap

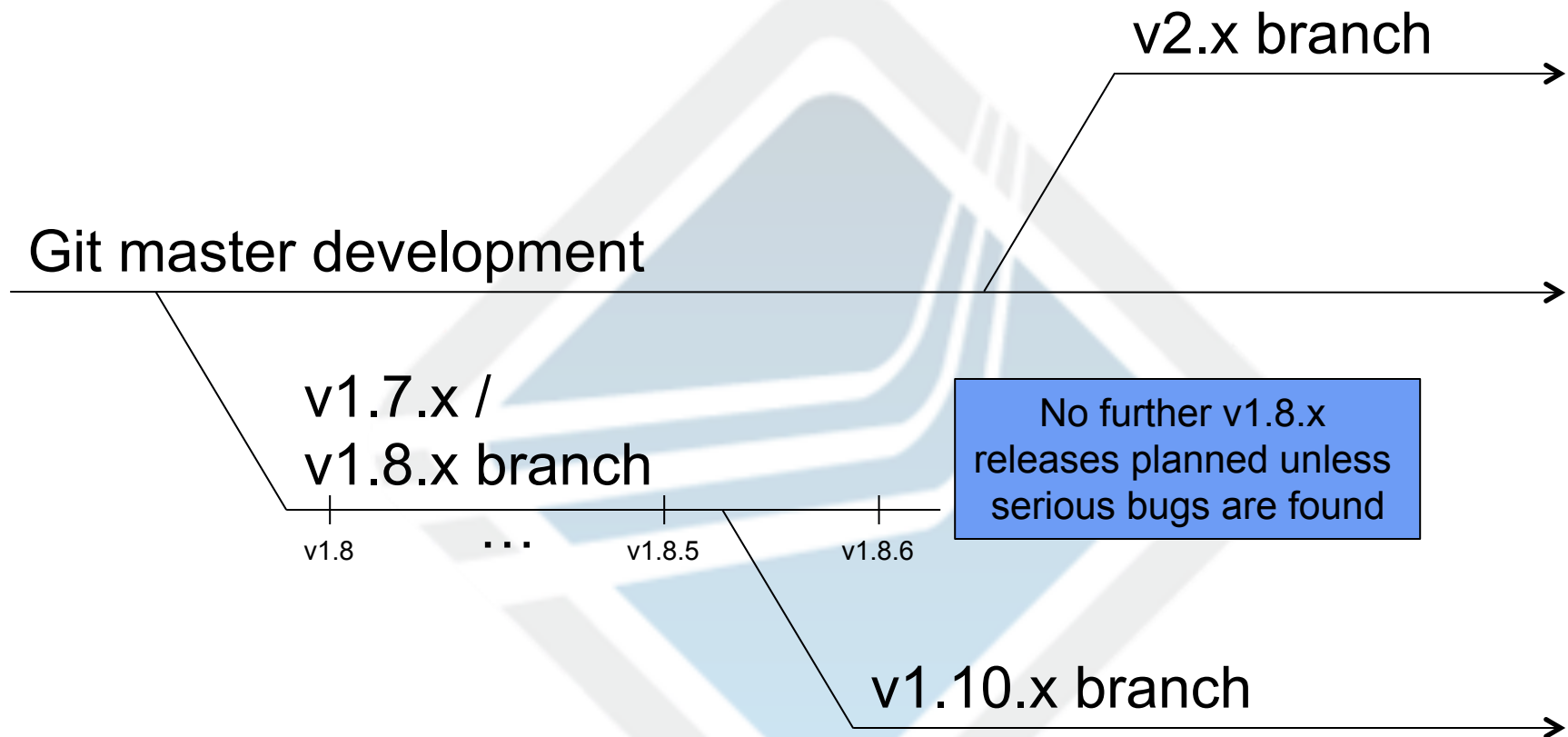
What's next?

- We are planning for **v1.10.0**
 - Within the next few months
 - Contains the usual bug fixes and minor improvements (over v1.8.6)
 - Also contains a small number of new features
 - libfabric support
 - Mellanox Yalla PML
 - **Backwards compatible with the v1.8.x series**

What's next?

- We anticipate **v2.0.0**
 - Later this year
 - Will contain larger new features
 - Will not be backwards compatible with v1.8.x or v1.10.x

Transition definition for the technically inclined



Why “v1.10.0” (vs. “v1.9.0”)?

1. Before June 2015, we referred to the next major release as the “v1.9 series”
 - “v1.10.0” clearly distinguishes from that idea
 - “v2.0.0” conveys a significant difference (i.e., a major new release series)
2. It will take a while for the new scheme to become common knowledge
 - We didn’t want users to think “v1.9” = “odd” = “unstable”

What's the plan over time?

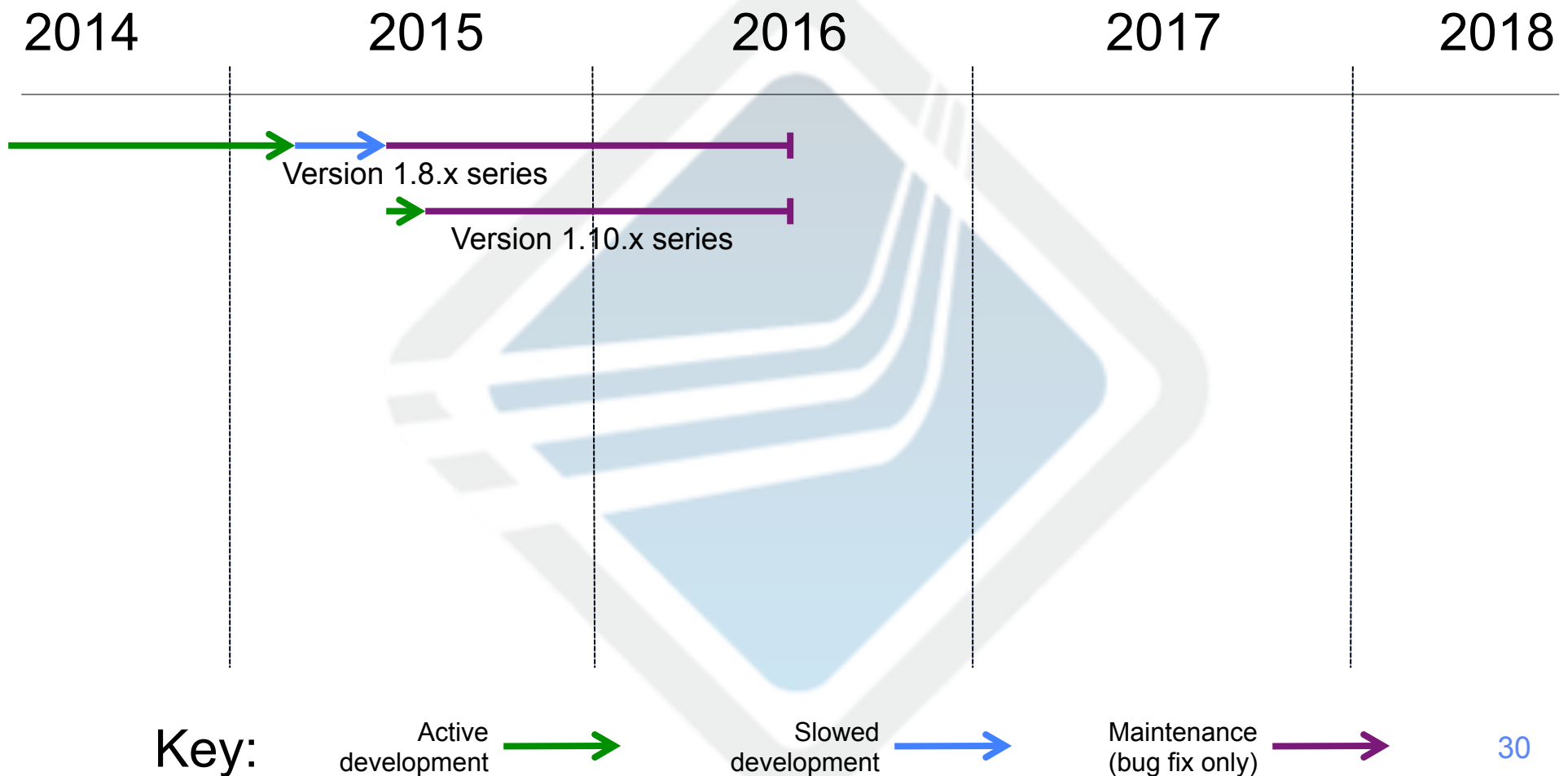
- Plan for a new release series once a year
 - v2.x: release in mid / late 2015
 - v3.x: release in mid / late 2016
 - v4.x: release in mid / late 2017
 - ...etc.

NOTE: Scheduled releases is a new concept for the Open MPI developer community. We'll continue to evaluate this plan over time.

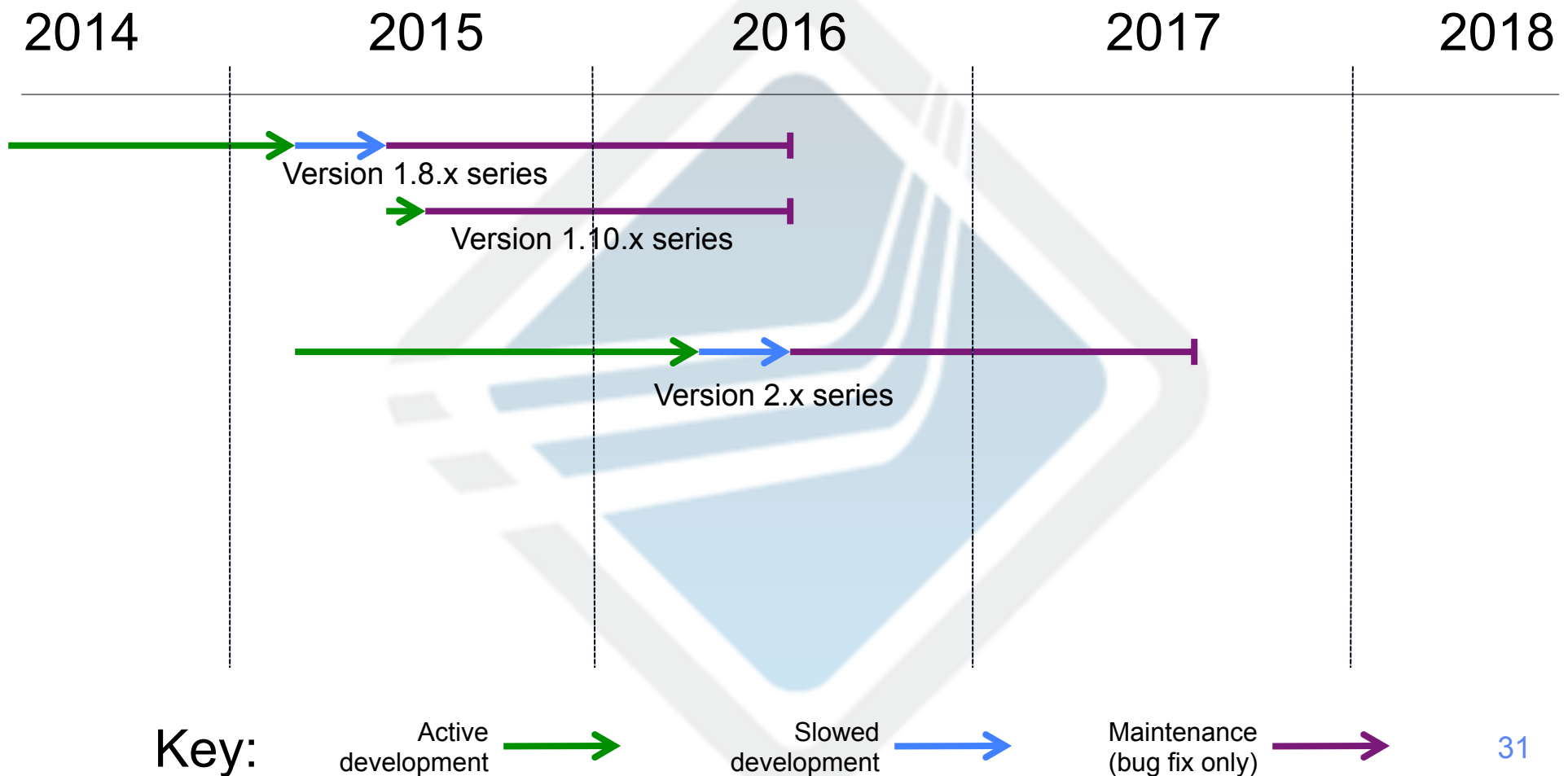
What will be supported?

- (Continue the) Support “current version and one prior” philosophy
 - Mid 2015 – mid 2016
 - Support v1.10.x, and v2.x
 - Special case: **also support v1.8.x**
 - Mid 2016 – mid 2017
 - Support v2.x and v3.x
 - Mid 2017 – mid 2018
 - Support v3.x and v4.x
 - ...etc.

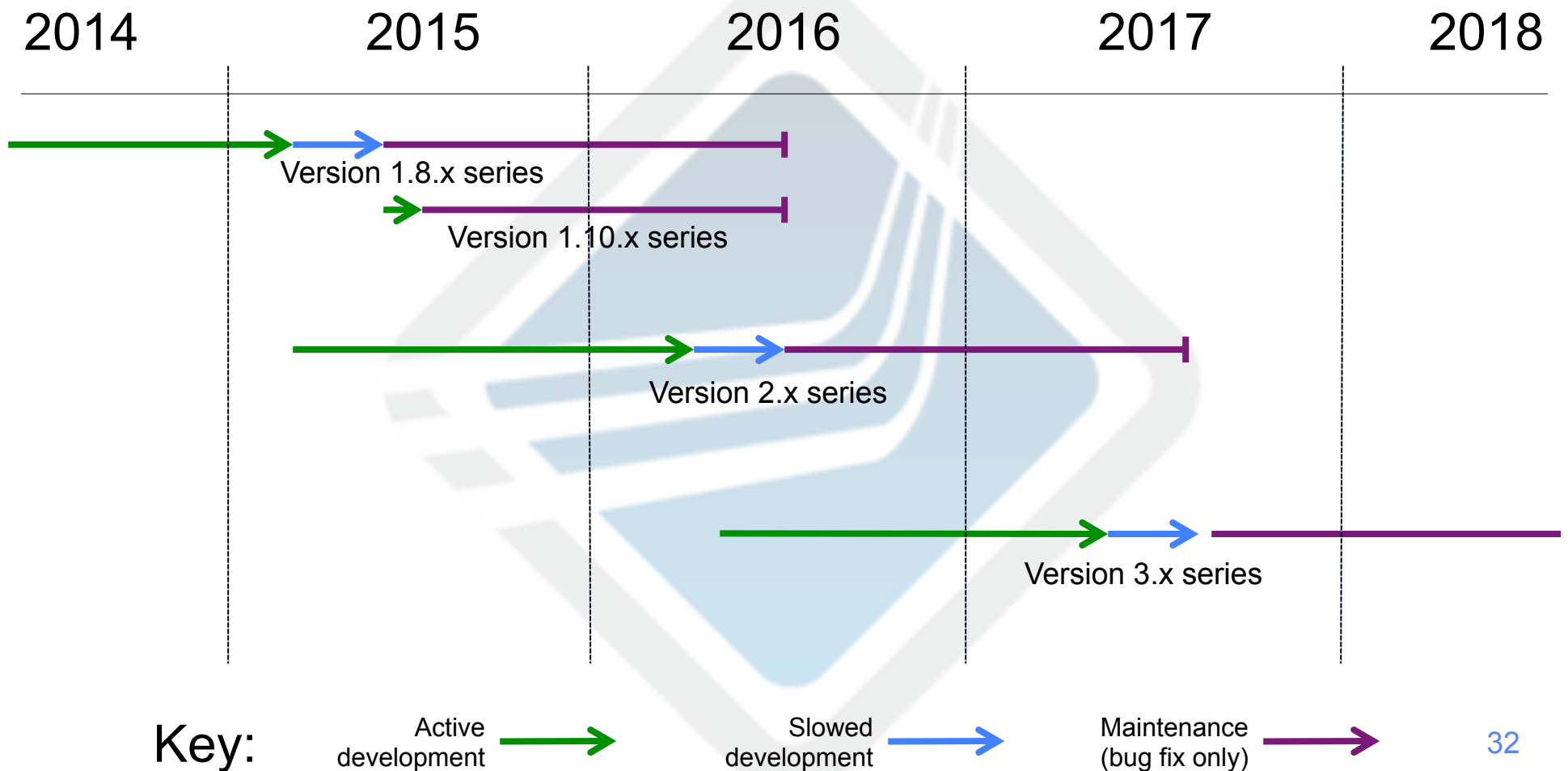
Planned development and support cycle



Planned development and support cycle



Planned development and support cycle





The Bottom Line

The bottom line

- Starting with v1.10.0:
 - No more odd/even series
 - “A.B.C”: each number has a specific meaning
 - Release new features faster
- Aim to limit life release series
 - ~1 year of devel + ~1 year of bug fixes