



New version numbering proposal

May 2015

Where this proposal started from...

- Odd / even scheme doesn't seem to be working well
 - Most users only using even number releases (because we tell them to)
 - Resulting in long delays for publishing new features
 - Specific case: Cisco/Intel want to make a new release for stable release with libfabric support
- ...so why do we keep using the odd/even scheme?

Current process

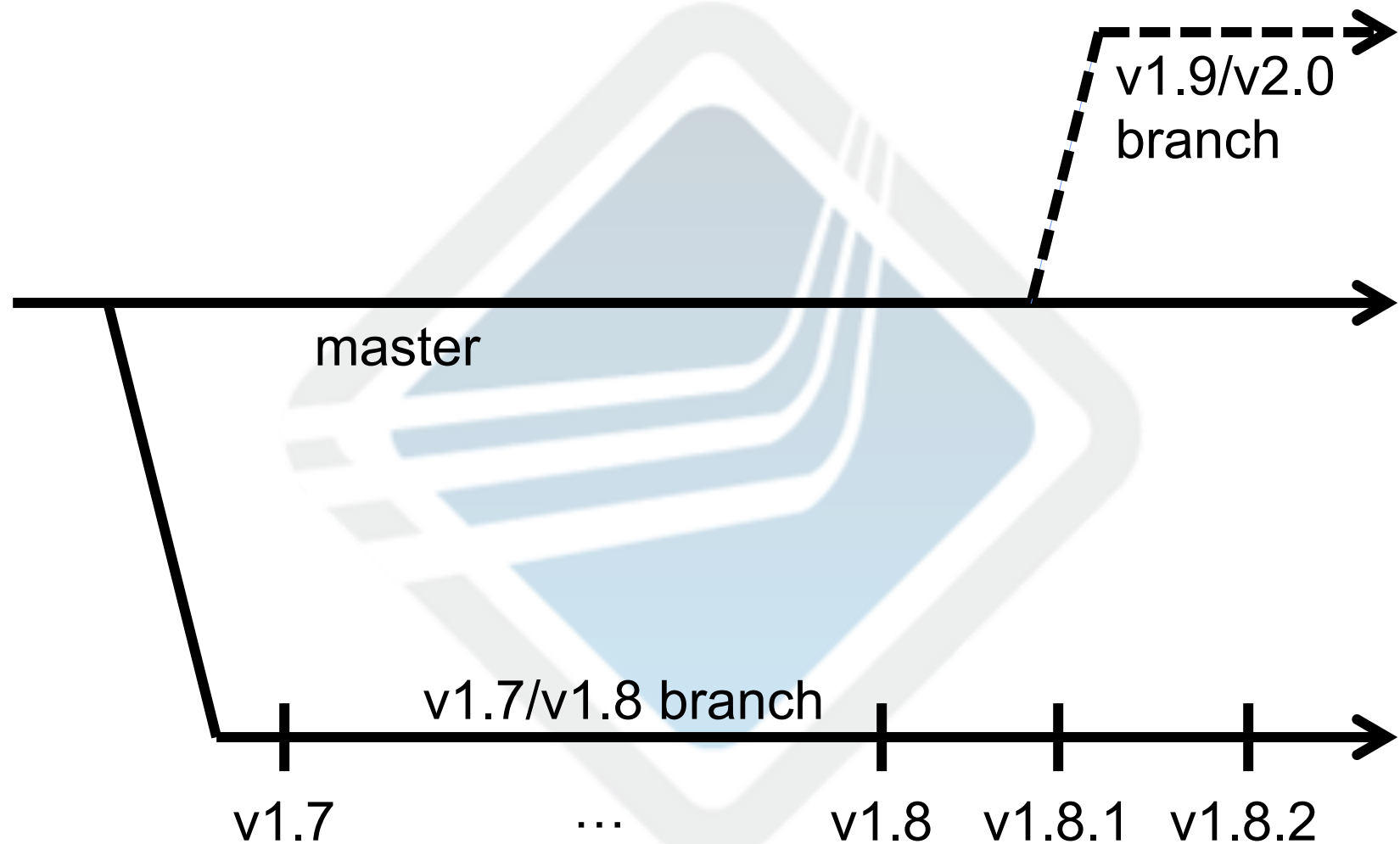
Part 1

- 2nd digit odd / even strategy
 - Odd: feature series
 - Even: super stable
- 3rd digit changes every release

Part 2

- Git management
 - Release branches
 - Git tags
 - PRs from master to release branches

Current process



Proposed process

- Four major ideas:
 1. No more odd/even release series
 - All releases are good!
 2. Allow new features to release branches
 3. Slightly refine meaning of OMPI's use of MAJOR.MINOR.RELEASE version numbers
 4. How to transition to the new scheme

Proposed process

- Four major ideas:

- Self-explanatory {
1. No more odd/even release series
 - All releases are good!
 2. Allow new features to release branches
 3. Slightly refine meaning of OMPI's use of MAJOR.MINOR.RELEASE version numbers
 4. How to transition to the new scheme



MAJOR.MINOR.RELEASE

Slightly refine OMPI's use of
MAJOR.MINOR.RELEASE

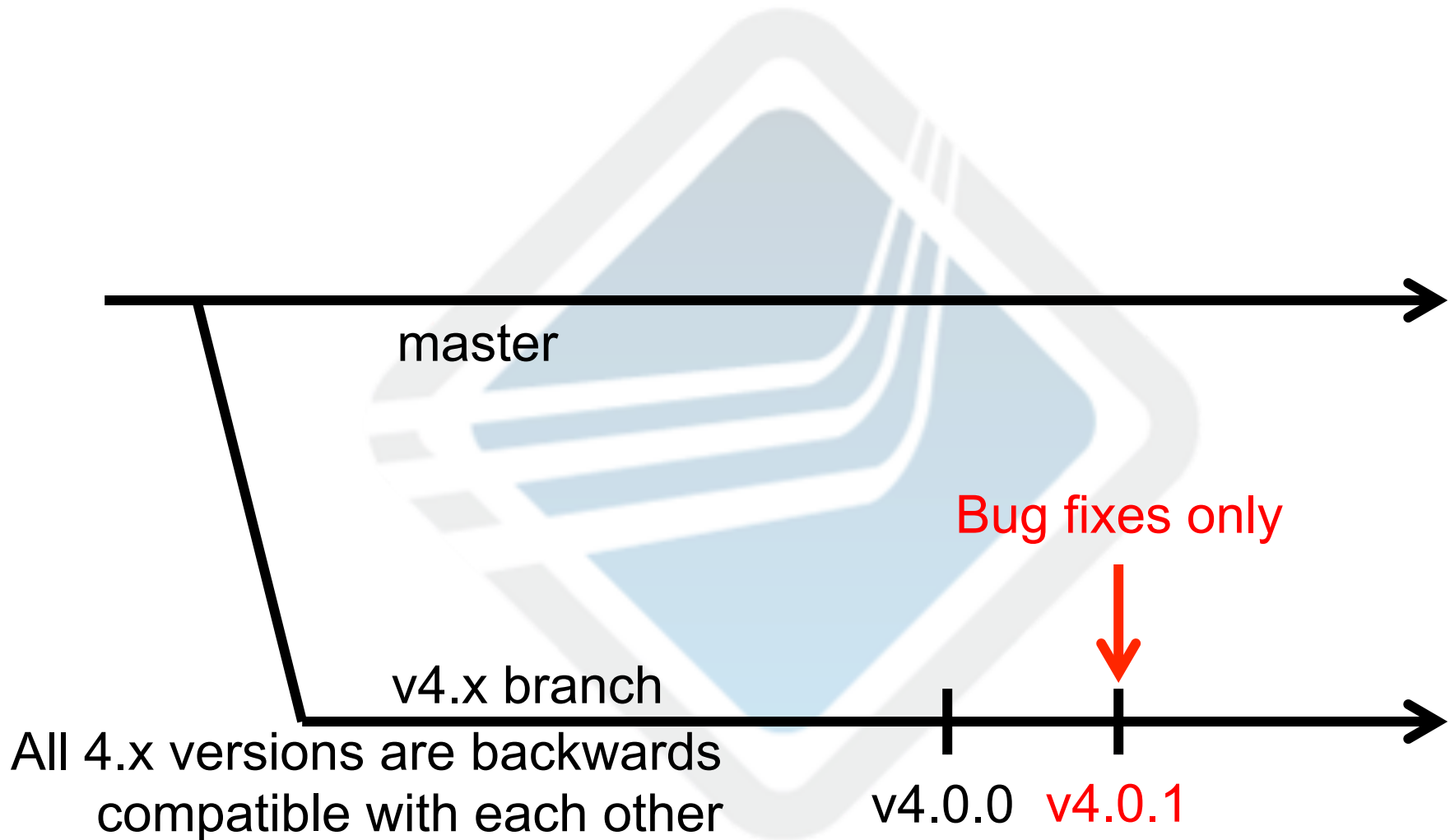
Definition

- Open MPI v Y is backwards compatible with Open MPI v X (where $Y > X$) if users can:
 - Compile MPI/OSHMEMP executable with v X , mpirun it with v Y , and get the same user-observable behavior
 - Invoke ompi_info with the same CLI options in v X and v Y and get the same user-observable behavior
- Things that break backwards compatibility:
 - Change the MPI or OSHMEM API ABI
 - Change or delete mpirun / ompi_info CLI options
 - Change or delete MCA parameter names / meanings
 - Change mpirun/ORTE wire protocols

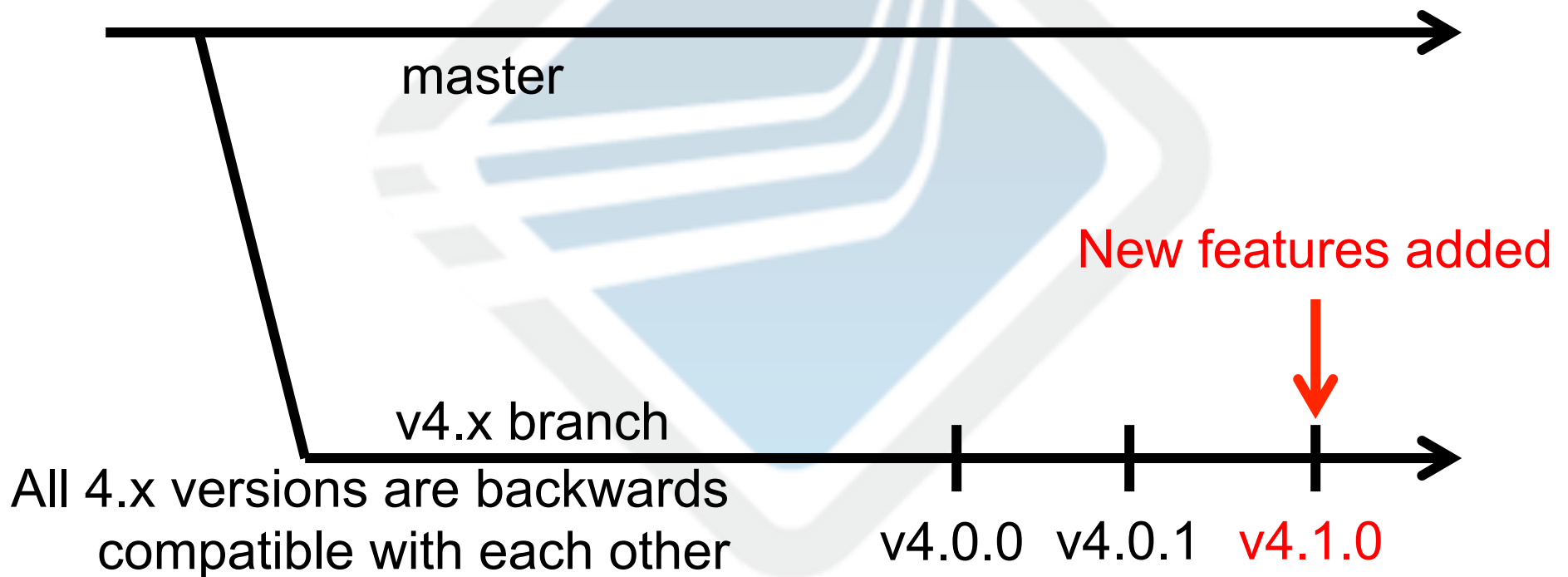
New version scheme (using same MAJOR.MINOR.RELEASE triple)

- Heavily influenced by Semantic Versioning (<http://semver.org>)
- Bump:
 - MAJOR: when we fork from master for a new release series
 - MINOR: when we add user observable new features (compared to the prior release)
 - RELEASE: all other releases
- Only allow backwards incompatible changes when MAJOR changes

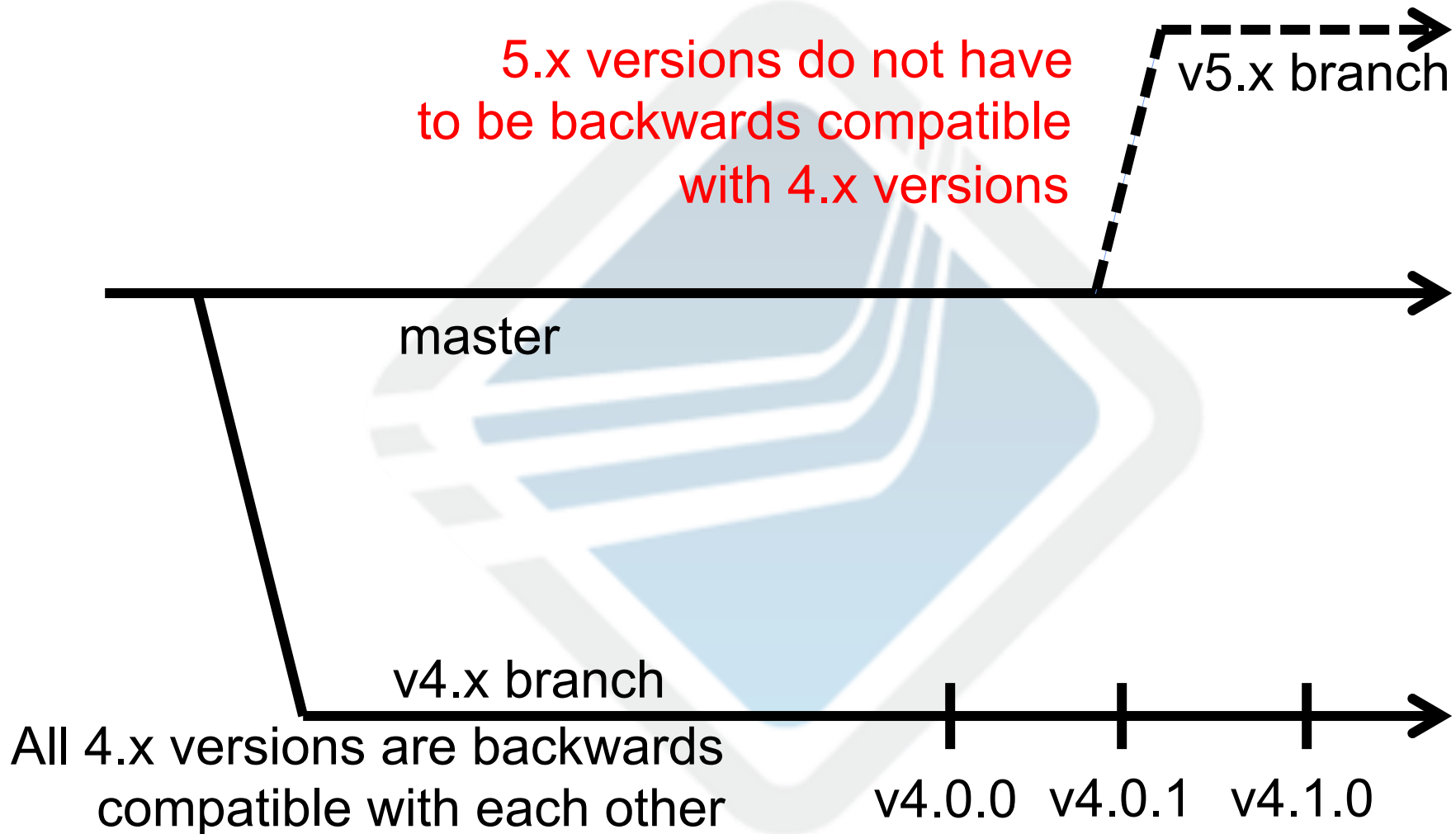
New version scheme




New version scheme



New version scheme



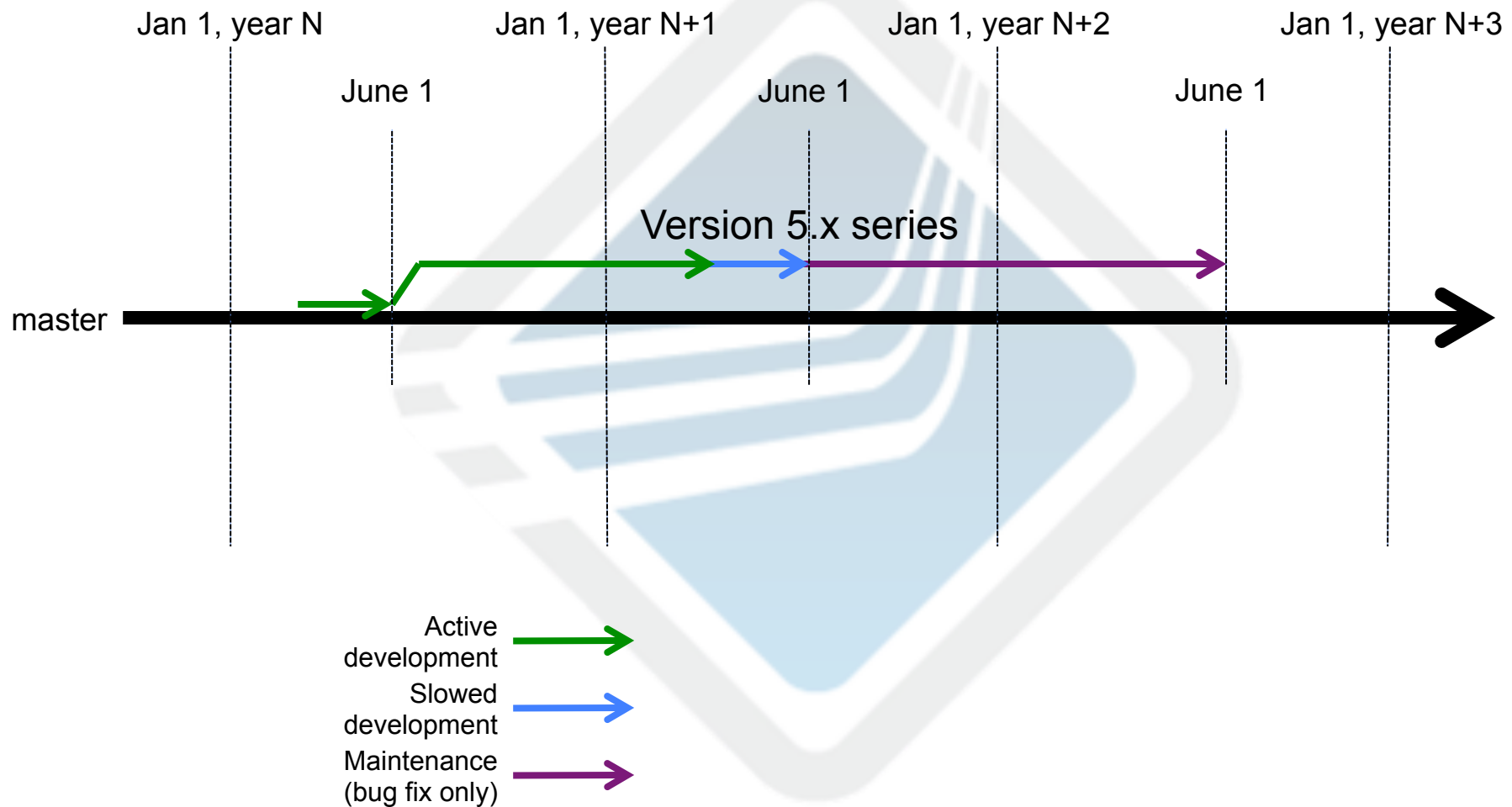
Keep many things the same

- Generally keep 1-2 release branches alive
 - Last series + this series
 - PRs from master to release branches
 - Publicly support current and prior release series
- 

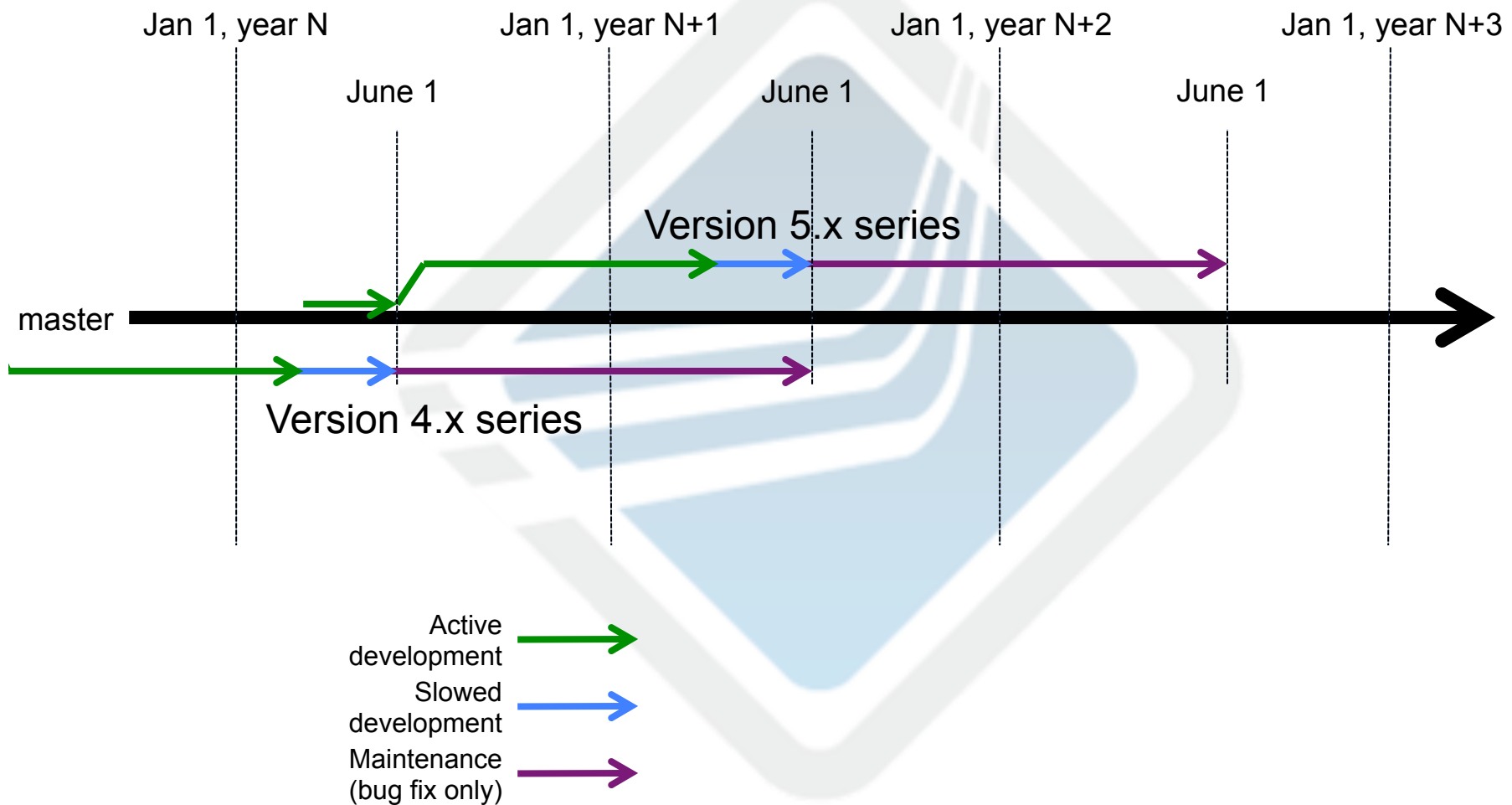
A few notable differences

- No backwards incompatible changes on a release branch
 - ...once vX.0.0 is released
- Release branch development life = ~1 year
 - Aim to stop adding major new features ~9 months after fork from master
- Aim to fork new release branch ~June 1 annually
 - ...unless there's a reason to fork earlier

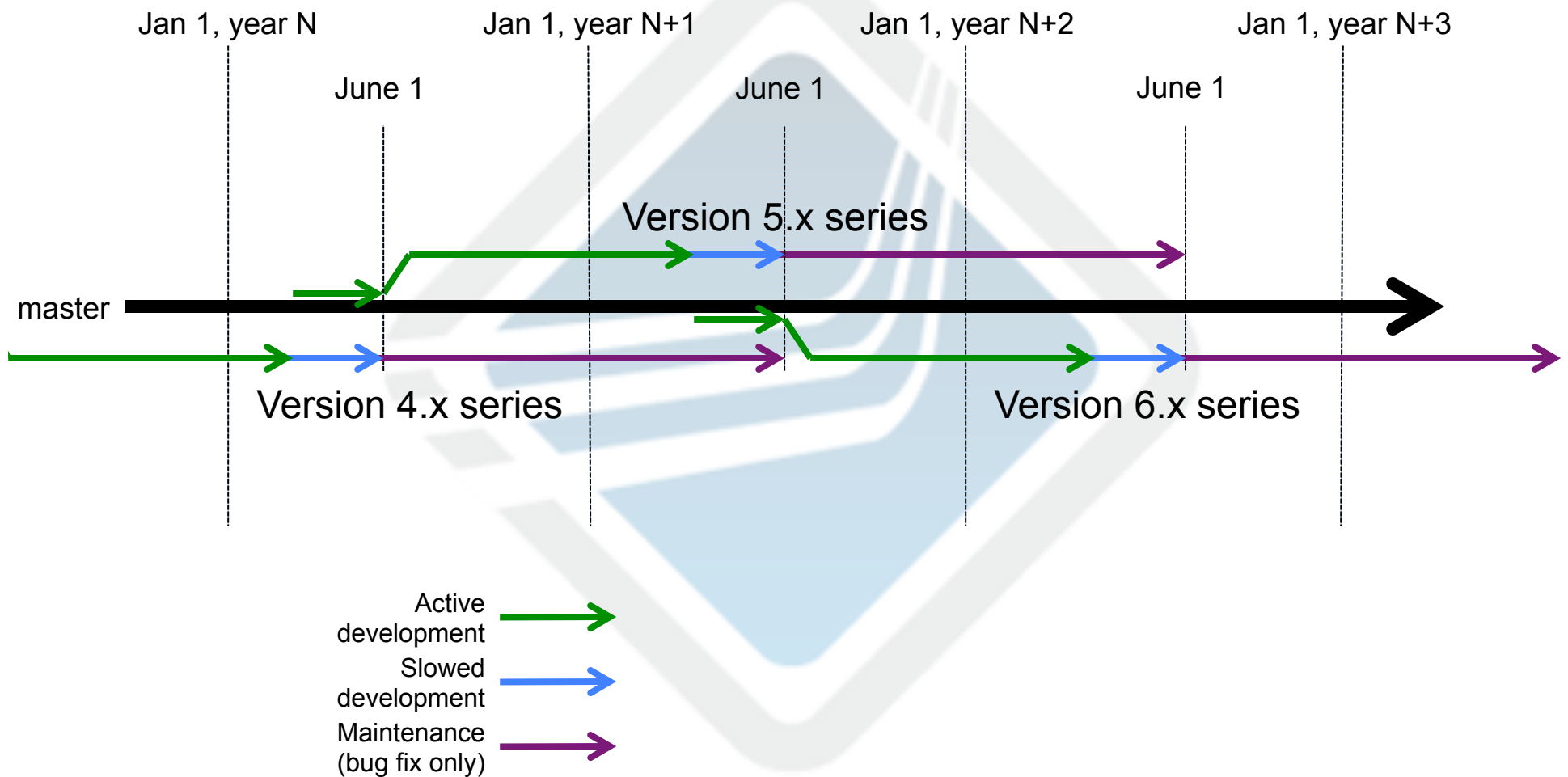
Typical release cycle estimate



Typical release cycle estimate



Typical release cycle estimate





Transition

How to transition to the new
scheme?

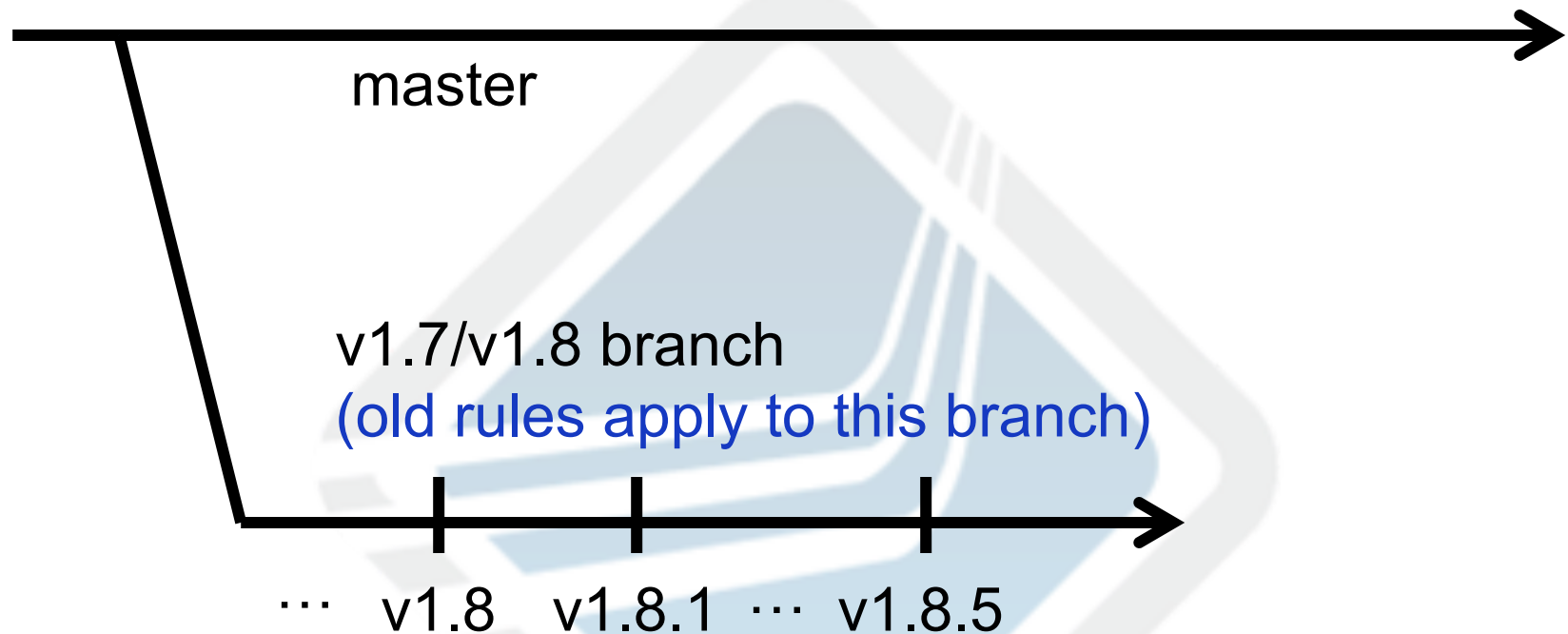
Transition proposal

- Goals:
 - Officially release small number of new features in the immediate future
 - Allows to meet distro release date targets
 - Based on the stable v1.8 series
 - (master is not yet ready to release as v2.x)

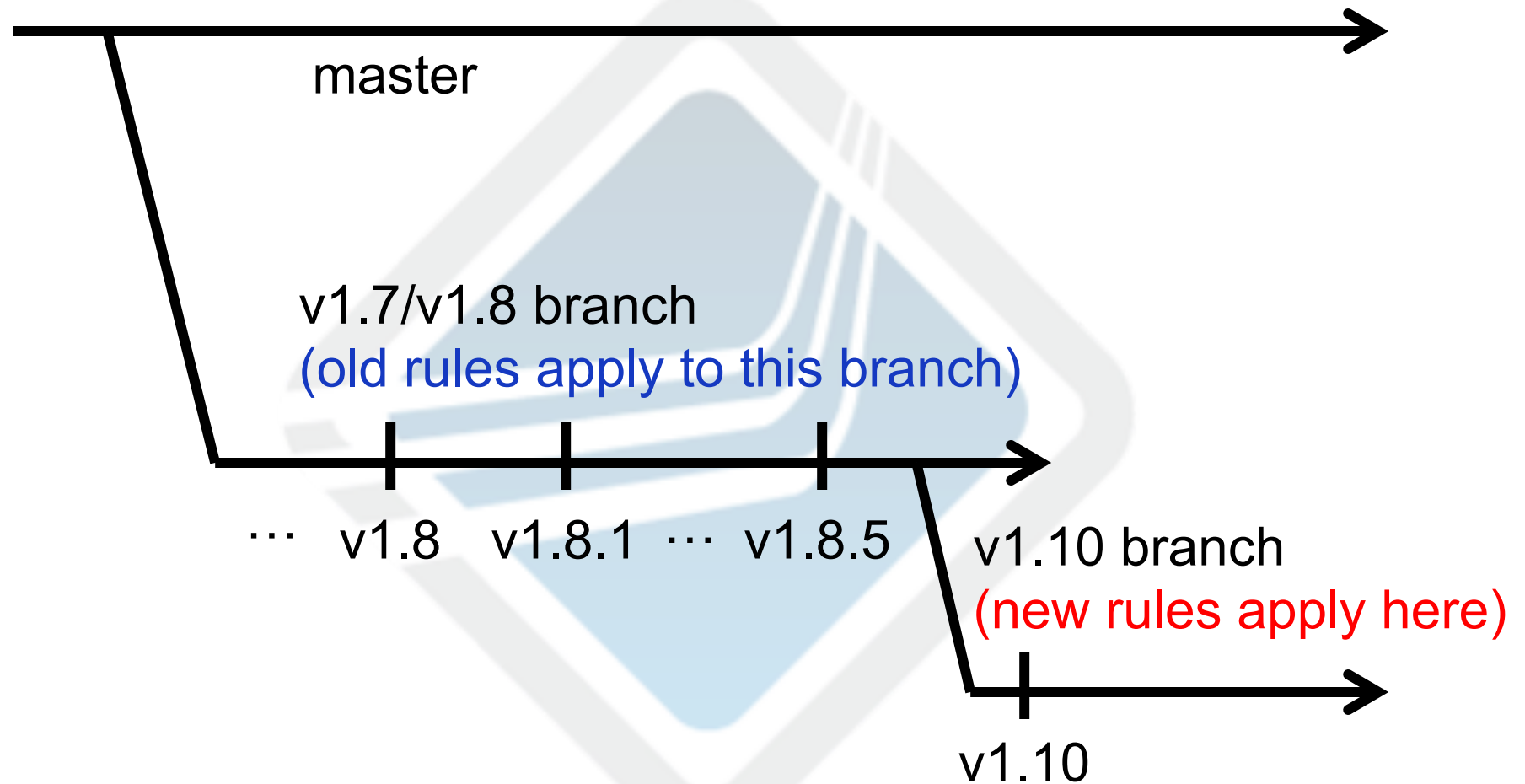
Transition proposal

- Cannot change version numbering scheme in middle of the v1.8.x series
 - That would be crazy
 - All v1.8.x releases will continue to abide by the “old” rules
- The first release under the “new” rules must have a new release series name

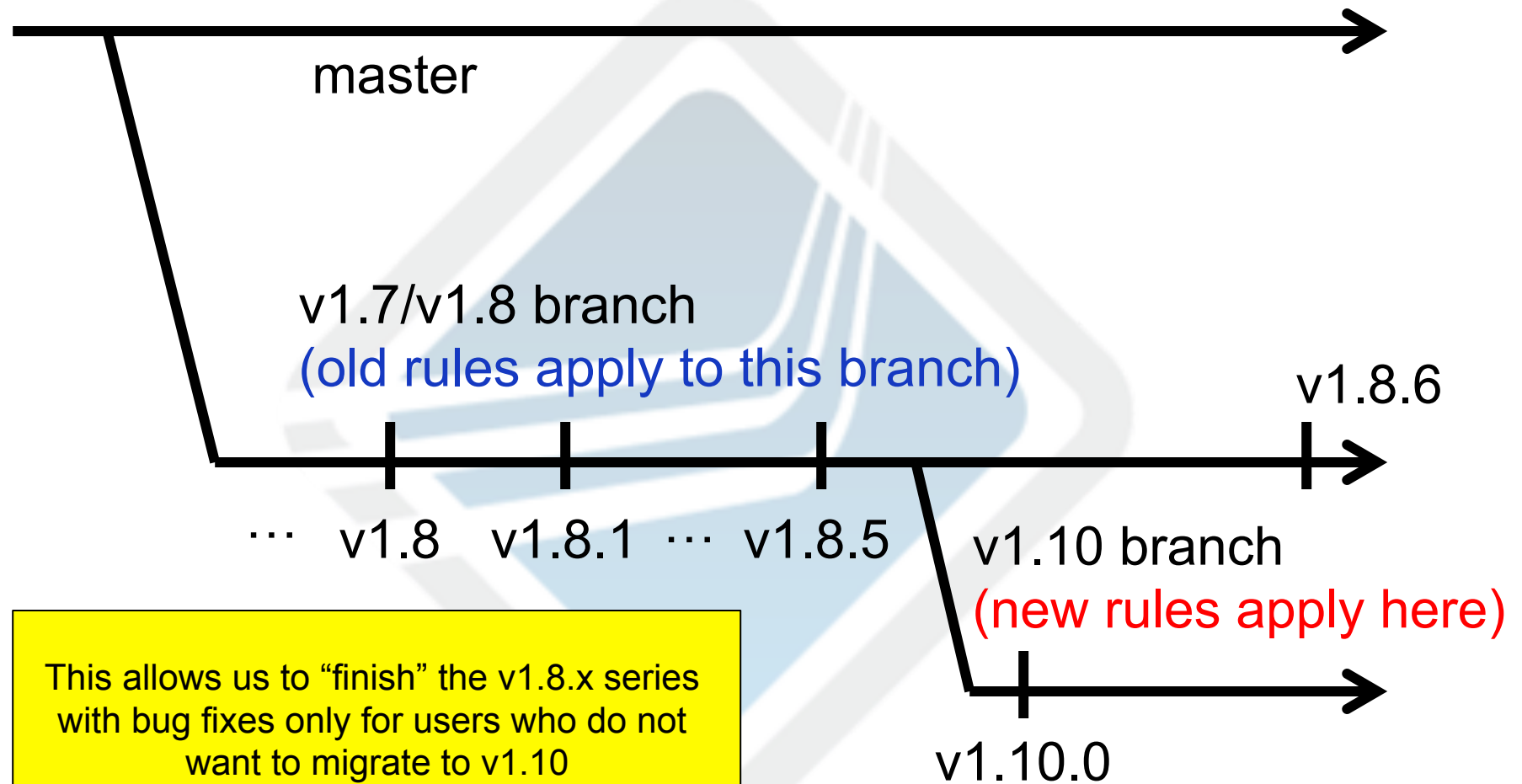
Where we are today



Do a one-time fork from v1.8

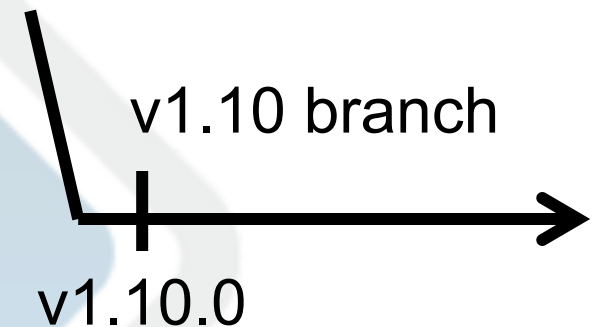


Maybe have more v1.8.x releases later



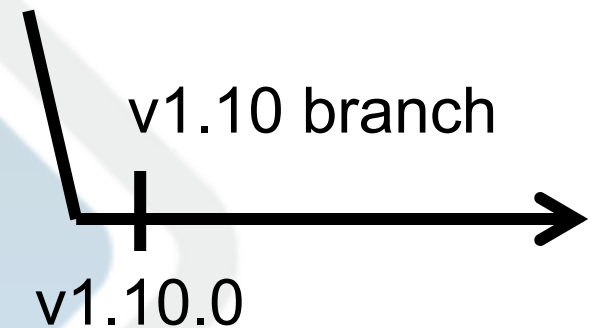
v1.10 branch

- The even value of “10” helps users while we transition
 - They’ll see it as an “even” (stable) release
- But we’ll allow new features in v1.10
 - libfabric components
 - New Mellanox components
 - ...
- Also allow the usual bug fixes
 - I.e., what would have gone into v1.8.6 and beyond

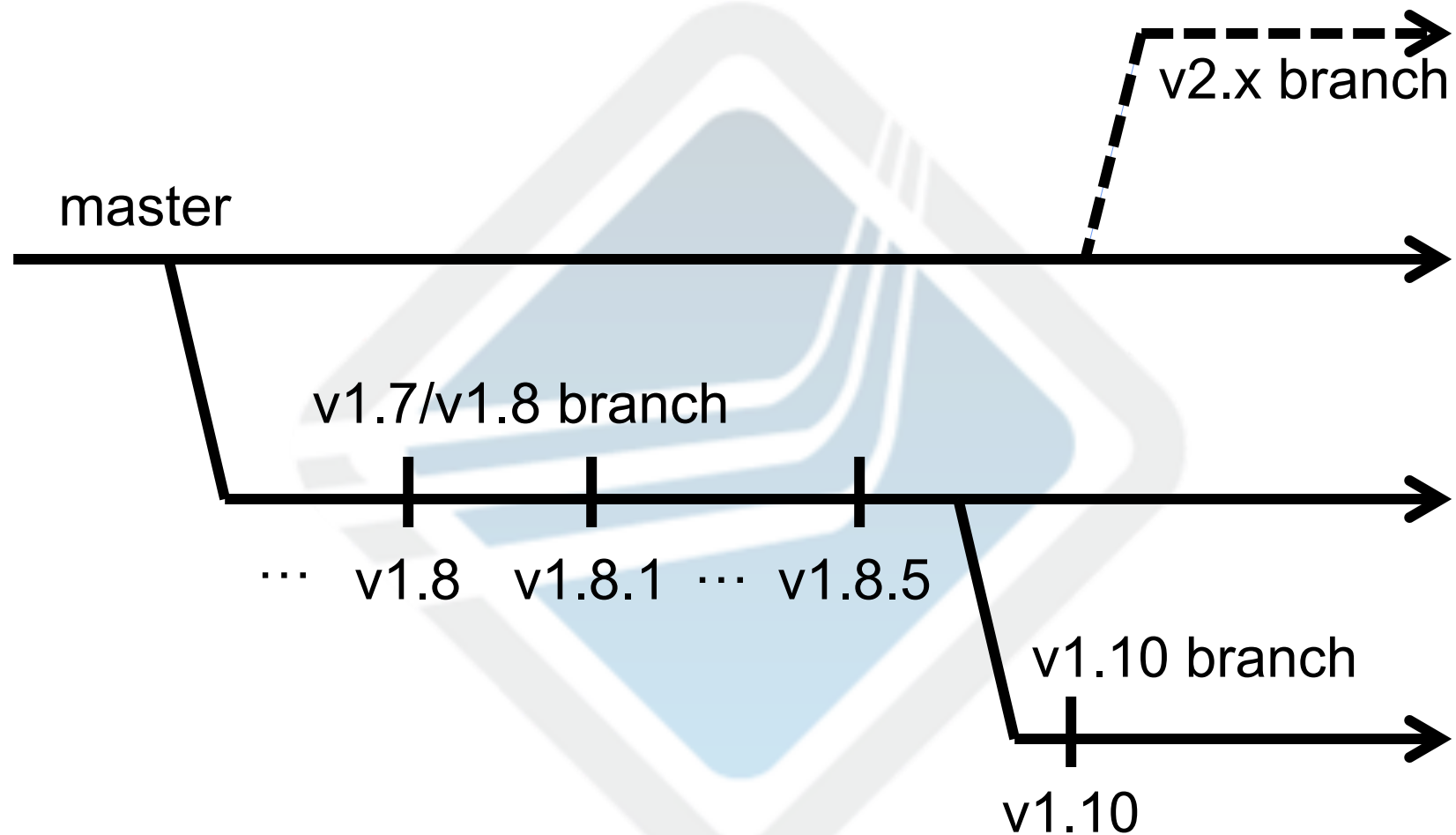


v1.10 branch

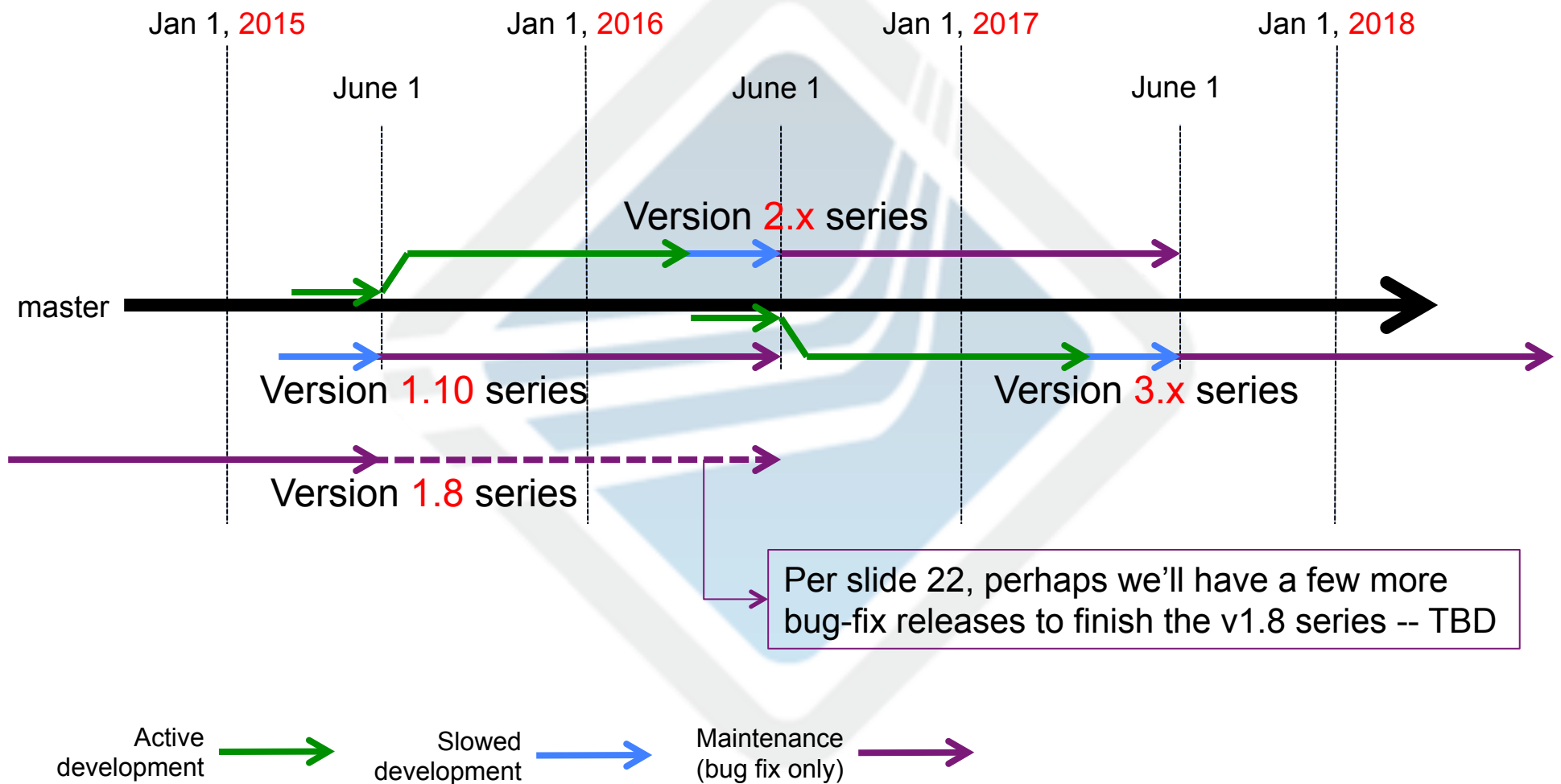
- Focus will be on a small number of low-risk new features
 - Libfabric support, new Mellanox components, ...
- Ralph Castain will still be Release Manager
- Howard+Jeff will still branch from master this summer
 - Will be v2.0
 - Will be the focus of new development



Fork v2.x this summer



Expected release cycles



So what's really different?

- No more odd/even series
- Changed meaning of MAJOR and MINOR
- Allow new features to release branches
- One-time transition to v1.10 series
- Aim to fork new release branch regularly
 - Every ~12 months
- Aim to limit life of release branches
 - ~1 year of devel + ~1 year of bug fixes
- Better testing on release branches
 - Can't assume "odd" = "can do less testing"