SIG Release: Deep Dive - LTS

Tim Pepper (VMware) - SIG chair; 1.12 release lead

kubernetes

Agenda

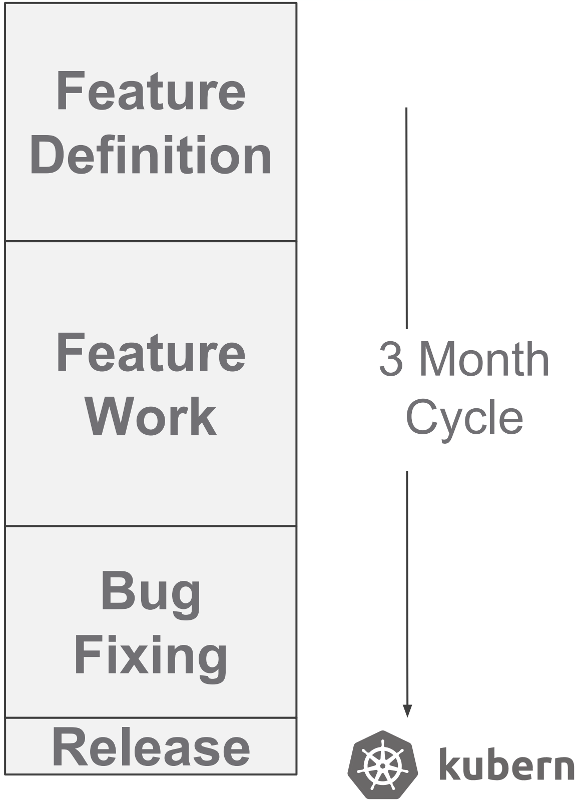
- Brief overview of the Kubernetes
  - release process
  - release lifecycle
- Impacts of process/lifecycle on Kubernetes vendors, distributors, and hosted service providers
- Discussion

Release Process

Feature Discussion (per SIG):	ongoing
Feature Freeze:	week ~4
Release Branch Creation:	week ~7
Code Slush / Freeze:	week ~8

...bugs, testing, bugs, fixing, bugs, ...iterating

End Code Freeze:	week ~12
Release:	week ~13



Feature Definition

Feature Work

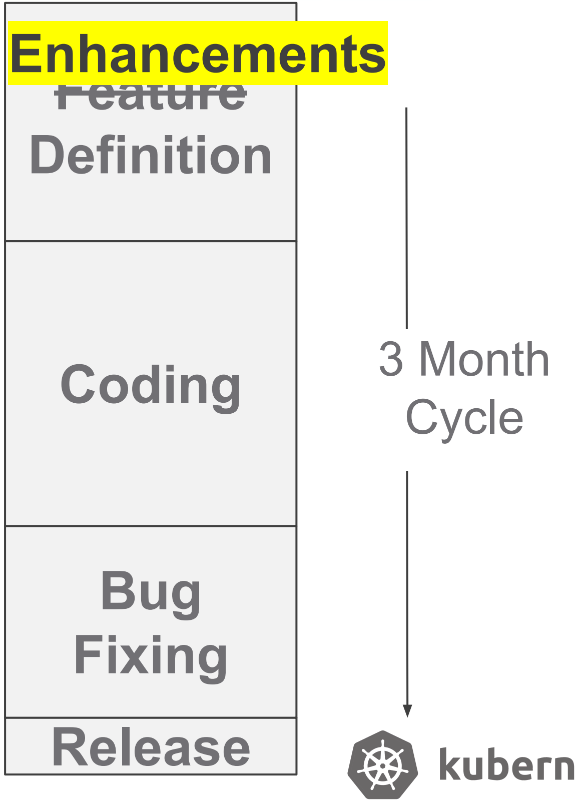
Bug Fixing

Release

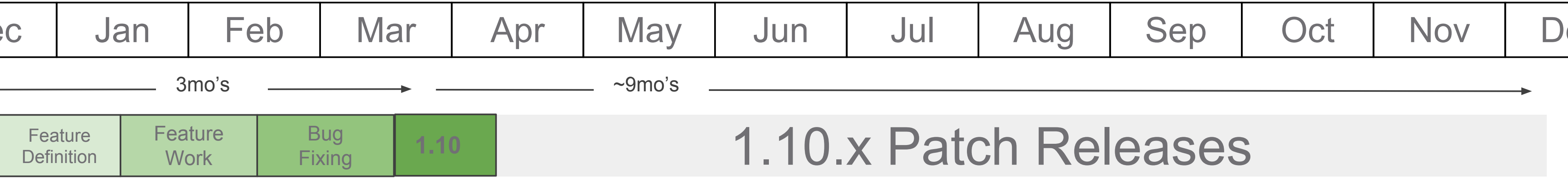
3 Month Cycle

Release Process

~~Feature~~ Discussion (per SIG):	ongoing
Enhancements	
~~Feature~~ Freeze:	week ~4
Release Branch Creation:	week ~7
Code Slush / Freeze:	week ~8
...bugs, testing, bugs, fixing, bugs, ...iterating	
End Code Freeze:	week ~12
Release:	week ~13



Release Lifecycle



ec | Jan | Feb | Mar | Apr | May | Jun | Jul | Aug | Sep | Oct | Nov | De

3mo's

~9mo's

Feature Definition

Feature Work

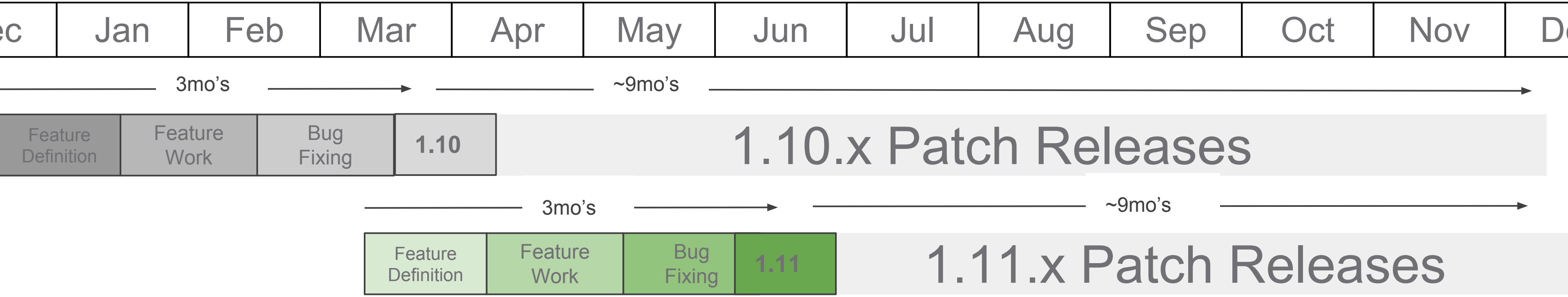
Bug Fixing

1.10

1.10.x Patch Releases

K8S Releases

Release Lifecycle



ec	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	De

3mo's → ~9mo's →

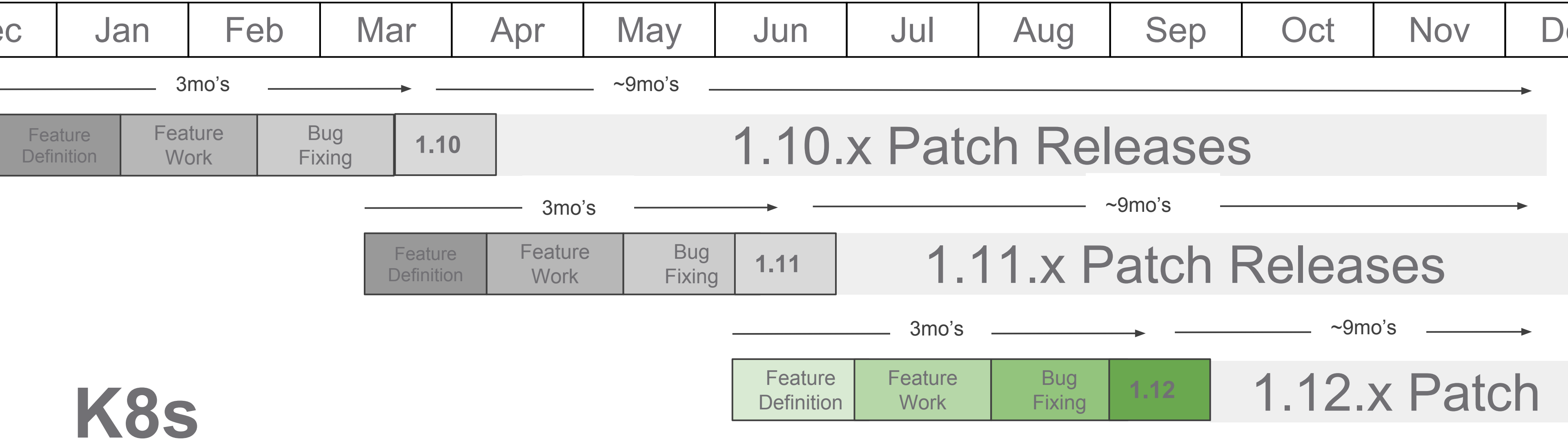
Feature Definition | Feature Work | Bug Fixing | 1.10 | 1.10.x Patch Releases

3mo's → ~9mo's →

Feature Definition | Feature Work | Bug Fixing | 1.11 | 1.11.x Patch Releases

K8S Releases

Release Lifecycle



ec | Jan | Feb | Mar | Apr | May | Jun | Jul | Aug | Sep | Oct | Nov | De

3mo's

~9mo's

Feature Definition | Feature Work | Bug Fixing | 1.10 | 1.10.x Patch Releases

3mo's

~9mo's

Feature Definition | Feature Work | Bug Fixing | 1.11 | 1.11.x Patch Releases

3mo's

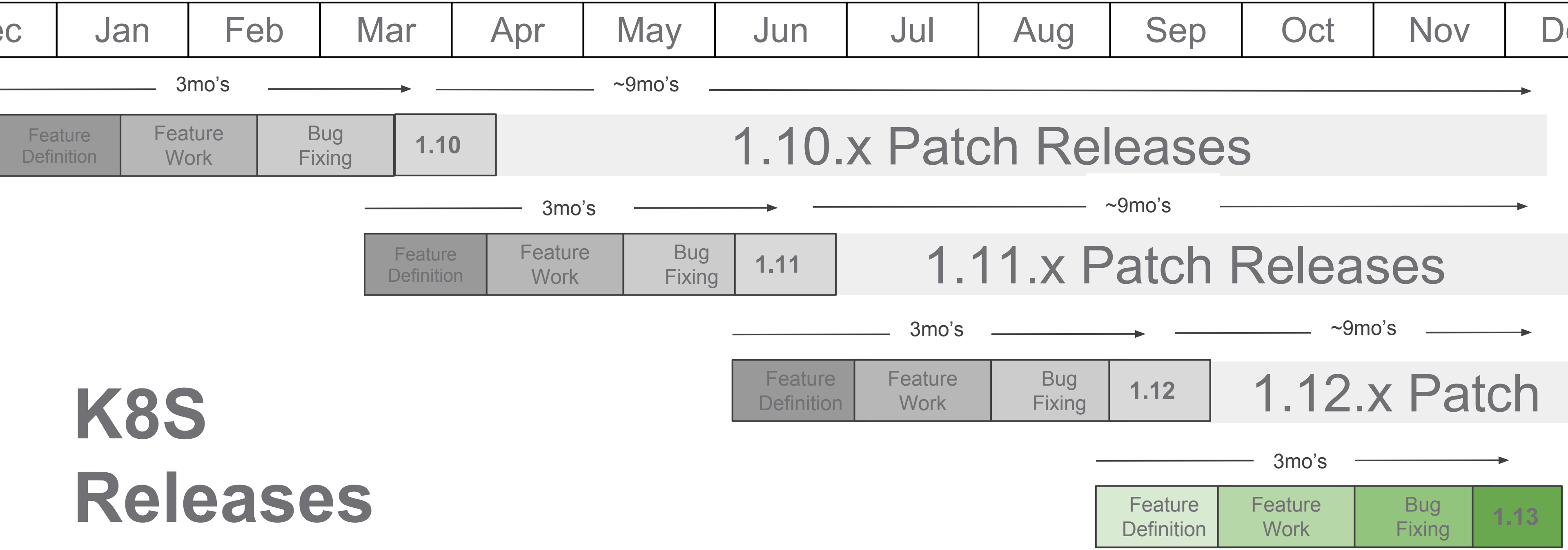
~9mo's

Feature Definition | Feature Work | Bug Fixing | 1.12 | 1.12.x Patch

K8s Releases

kubernetes

Release Lifecycle



ec | Jan | Feb | Mar | Apr | May | Jun | Jul | Aug | Sep | Oct | Nov | De

3mo's → ~9mo's →

Feature Definition | Feature Work | Bug Fixing | 1.10 | 1.10.x Patch Releases

3mo's → ~9mo's →

Feature Definition | Feature Work | Bug Fixing | 1.11 | 1.11.x Patch Releases

3mo's → ~9mo's →

Feature Definition | Feature Work | Bug Fixing | 1.12 | 1.12.x Patch

3mo's →

Feature Definition | Feature Work | Bug Fixing | 1.13

K8S Releases

kubernetes

Release Lifecycle

- **Support: backporting critical fixes to a release branch**

- **3 releases** get support:
  - Latest stable is 1.12
  - ...therefore patch support today is for 1.12, 1.11, 1.10

- Yields approximately **9 months** of support
- Must plan upgrades every 3-6 months

kubernetes

What is LTS

- **LTS** == **L**ong **T**erm **S**upport
- A documented support policy
- Identifies the length and type of support for releases, for example:
    - “security and bug fixes for 2 years”, or
    - “bug fixes for 1 year plus critical security fixes for 1 additional year”
- “Long” is typically > 1 year
    - Short Term Support (STS) streams are usually also present
    - “Short” is typically measured in months

Possible benefits of LTS

- Users: longer production deployment, more time to transition onto and off of a stable release, better cross-release compatibility

- Vendors: centralized, subject matter expert led bug and security fixes happen once in upstream, instead of in parallel at each vendor

- Developers: added test coverage give early visibility to compatibility issues

kubernetes

Possible risks, costs of LTS

- Human staffing
- Test matrix growth
- Added technical complexity of
  - Insuring API stability over longer timespans
  - Upgrading between more distant releases

WG LTS Goals

- Collect stakeholder feedback, requirements on support
- Gauge stakeholder willingness and ability to commit sufficient dev/maintenance resources expand support
- Establish:
    - Entry criteria to define a first release as LTS
    - Cadence, eg: one LTS per year
    - Support lifetime, eg: N months per LTS
    - Upgrade path, eg: from LTS N to LTS N+1
- Draft an LTS KEP and if deemed feasible…

  return implementable LTS KEP to SIG Release to operationalize

kubernetes

Many Vendors

Vendors	78
Products	89
Hosted Platforms	32
Distributions	49

Certified Vendors	74
Certified Products	84
Certified Distributions	49
Certified Hosted	28
Certified Installer	8

1.7 Certifications	26
1.8 Certifications	41
1.9 Certifications	42
1.10 Certifications	54
1.11 Certifications	43
1.12 Certifications	11

A **vendor** is an organization providing a Kubernetes distribution, hosted platform, or installer

A **product** is a distribution, hosted platform, or installer provided by a vendor.

A **distribution** is software based on Kubernetes that can be installed by an end user on to a public cloud or bare metal and includes patches, additional software, or both.

A **hosted** platform is a Kubernetes service provided and managed by a vendor.

An **installer** downloads and then installs vanilla upstream Kubernetes.

https://docs.google.com/spreadsheets/d/1LxSqBzjOxfGx3cmtZ4EbB_BGCxT_wlxW_xgHVVa23es/edit#gid=0

Many Vendors in Asia

Baidu
BoCloud
Caicloud
CStack
DaoCloud
eBaoTech
Hainan eKing Technology
HarmonyCloud
Huawei
Inspur
inwinStack
JD.com
Netease
Qiniu
Tencent
TenxCloud
Wise2c
Woqutech
ZTE

Feedback Wanted

Is Kubernetes releasing too fast? Too slow?

Do we end-of-life releases too fast? Too slow?

What slows you from updating from **1.x** to **1.(x+1)**?

How much staffing do vendors have for internal patch support?

Vendors are stronger together on stable patch support (LTS)?

Can we improve cross-release and cross-vendor compatibility?