# Kickstarting CI Best Practices with Jenkins in Autonomous Teams

Martin Singer
Software Engineer
Dynatrace


May 30, 2022

Katharina Sick
Software Engineer
Dynatrace


dynatrace

## Agenda

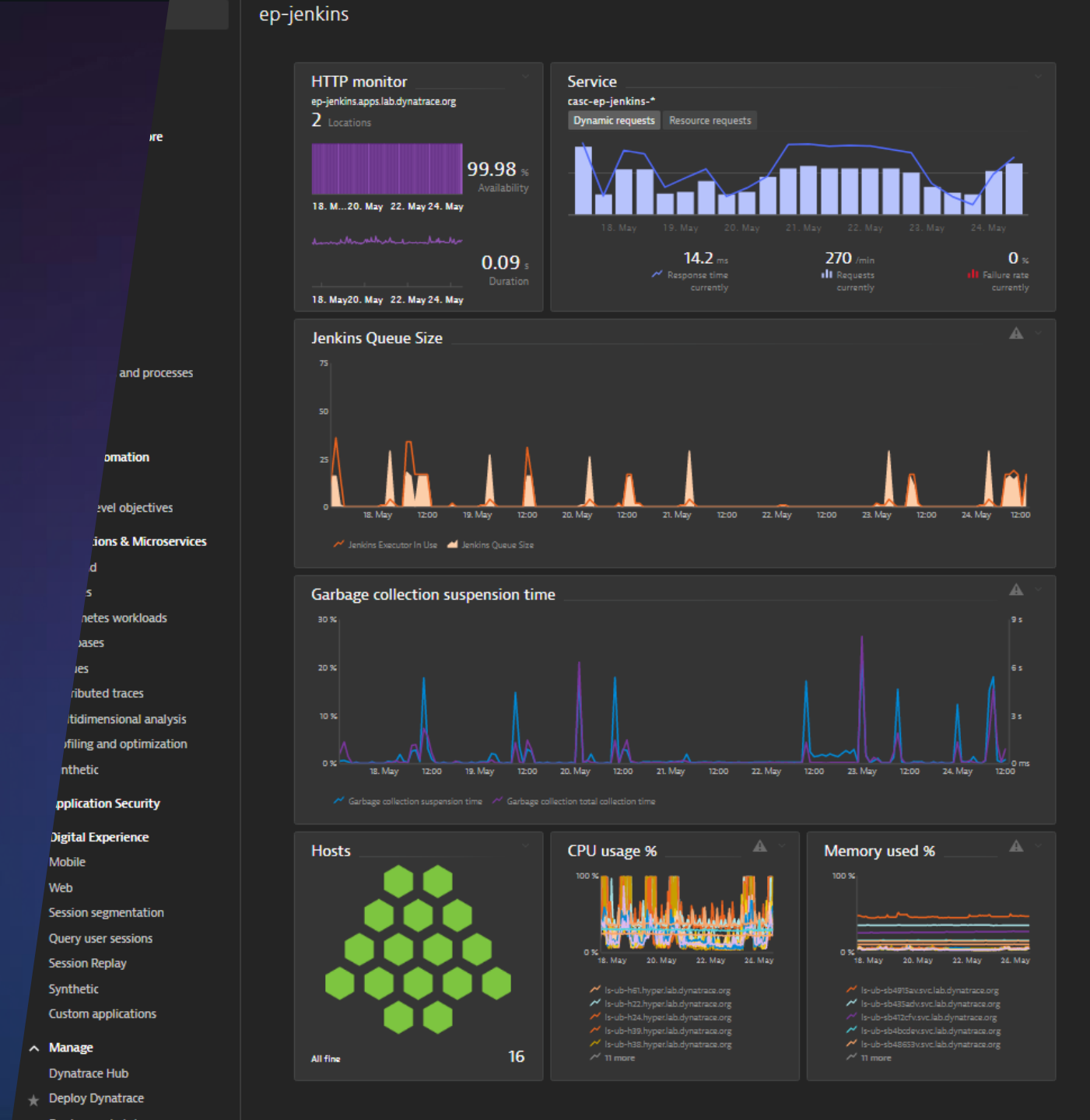
- Current State
- Build Environment
- Deployment and Testing
- Maintenance
- Live Demo
- Outlook

## Current State

- ~ 1200 Developers
- ~ 400.000 Builds per Sprint
- ~ 246.000.000 Test Executions per Sprint
- Small, independent Jenkins instances per team/department
- Dedicated Kubernetes namespace
- Configuration as Code instead of UI configurations
- 37 Jenkins instances

# What we deliver to our teams

- Jenkins Config as Code instance
- Dedicated Vault engine
- Kubernetes namespace for spawning executors
- Automatically generated Dynatrace Dashboard with relevant Metrics
- Onboarding material (recordings, documentation, code samples, …) / optional Q&A



# Build Environment

# Jenkins + Configuration as Code

***"The Configuration as Code plugin is an opinionated way to configure Jenkins based on human-readable declarative configuration files."*** [1]

- Configuration as Code instead of UI configuration
- Dedicated Bitbucket Repository per Jenkins instance
- Audit log, four eyes principle, easy maintenance

1. https://plugins.jenkins.io/configuration-as-code/

# Kubernetes

***"Kubernetes, also known as K8s, is an open-source system for automating deployment, scaling, and management of containerized applications."*** [1]

- All Jenkins CasC instances
- Jenkins workers
- Clusters in AWS & on-prem

1. https://kubernetes.io/

# Vault

***"Secure, store and tightly control access to tokens, passwords, certificates, encryption keys for protecting secrets and other sensitive data using a UI, CLI, or HTTP API."*** [1]

- Store all kinds of secrets (Key/Value, AWS etc)
- Role-based access control for secret engines
- Grant secret access for builds via an API
- Vault Config as Code with Terraform

1. https://vaultproject.io/

# Harbor

*"Harbor is an open source registry that secures artifacts with policies and role-based access control, ensures images are scanned and free from vulnerabilities, and signs images as trusted."* [1]

- Internal Docker image registry
- Security Scan
- Retention

1. https://goharbor.io/

# Project Initializer

***The Project Initializer aims to make the bootstrapping of new projects as easy as possible. To achieve that, it allows users to generate new projects from templates.***

- Self-developed service
- Bootstrap new projects from templates
- Keep projects at the newest state (automated updates)
- Ease permission requests for new repositories

# Deployment and Testing

# Instance creation

- ~ 1,5 hours of work required to set up new Jenkins Instance
- Workflow:
  1. Create dedicated Bitbucket service user and Jenkins Admin Group
  2. Create Jenkins + Vault setup via Project Initializer template
  3. Create dedicated Kubernetes namespaces (1 for instance, 1 for spawning pods) – soon automated
  4. Create Jenkins jobs to test and deploy instance
  5. Deploy instance and verify that everything works as expected
- Provide Onboarding for new Jenkins Admins (pre-recorded Onboarding mission + optional Q&A)

# Deployment process

## General

- Jenkins instance for spawning other instances
- Shared library for deployment logic

# Prepare Config Files

- Merges *.base.* and *.extension.* files
  - Base files are managed by the Project Initializer
  - Extension files can be used by Devs to personalize/extend basic setup
  - Powered by github.com/mikefarah/yq

# Vault configuration/Secret Injection

- Vault Config as Code
  - Devs can create their own Secret Engines or grant permissions to them via Config files
  - Implemented with Terraform
- Secret injection
  - Secrets needed for AzureAD usage, Mailserver config etc. are getting dynamically injected
  - Secrets are always up to date to guarantee working and correct environment

# Build and push Docker image

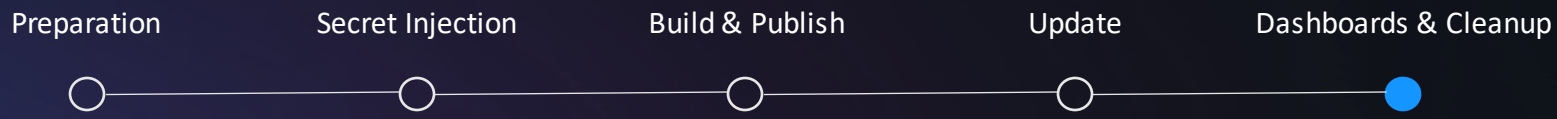
- Each Jenkins gets its own custom Docker image
- Base-Image: jenkins/jenkins:lts
- Add startup scripts needed for cleanup and testing
- Add desired plugins to Docker image
- Push to internal Docker Registry

# Update Jenkins

1. Final preparations for instance startup
2. Automated Testing mechanisms via curl/startup scripts (on test instance)
3. Downscaling of production instance
4. Setting new Docker image + Kubernetes config map
5. Upscaling newly updated instance
6. Testing again if everything works on production instance

# Create & Update Dashboards / Cleanup

- Create & Update Dashboards
  - Automatically create a dedicated Dynatrace Dashboard with relevant instance metrics
  - Add newly created instance (if not already added) to Overview Dashboard
- Clean up test namespace
  - Clean up test instance setup to free resources and have a clean state



# Deployment Cycles

- Weekly re-deployment to keep instance and plugins up to date
- Automatic re-deployment in the night if there are changes to the main branch
- Option to manually re-deploy instance for Jenkins admins of instance

## Seed Job

- Added on every instance per default
- Scans defined folder for Jobs Config as Code definitions
- Creates and updates existing Job configurations accordingly (via the Job DSL plugin)
- No downtime needed for adding new Jobs

## Testing

- Every branch build spawns a test instance for Devs
- Test instance runs with current changeset on branch
- Can be used for verifying config changes / testing Job configurations
- Seed job for testing Job setups
- No Kubernetes cloud definition

# Maintenance

## Instance Admins & Training

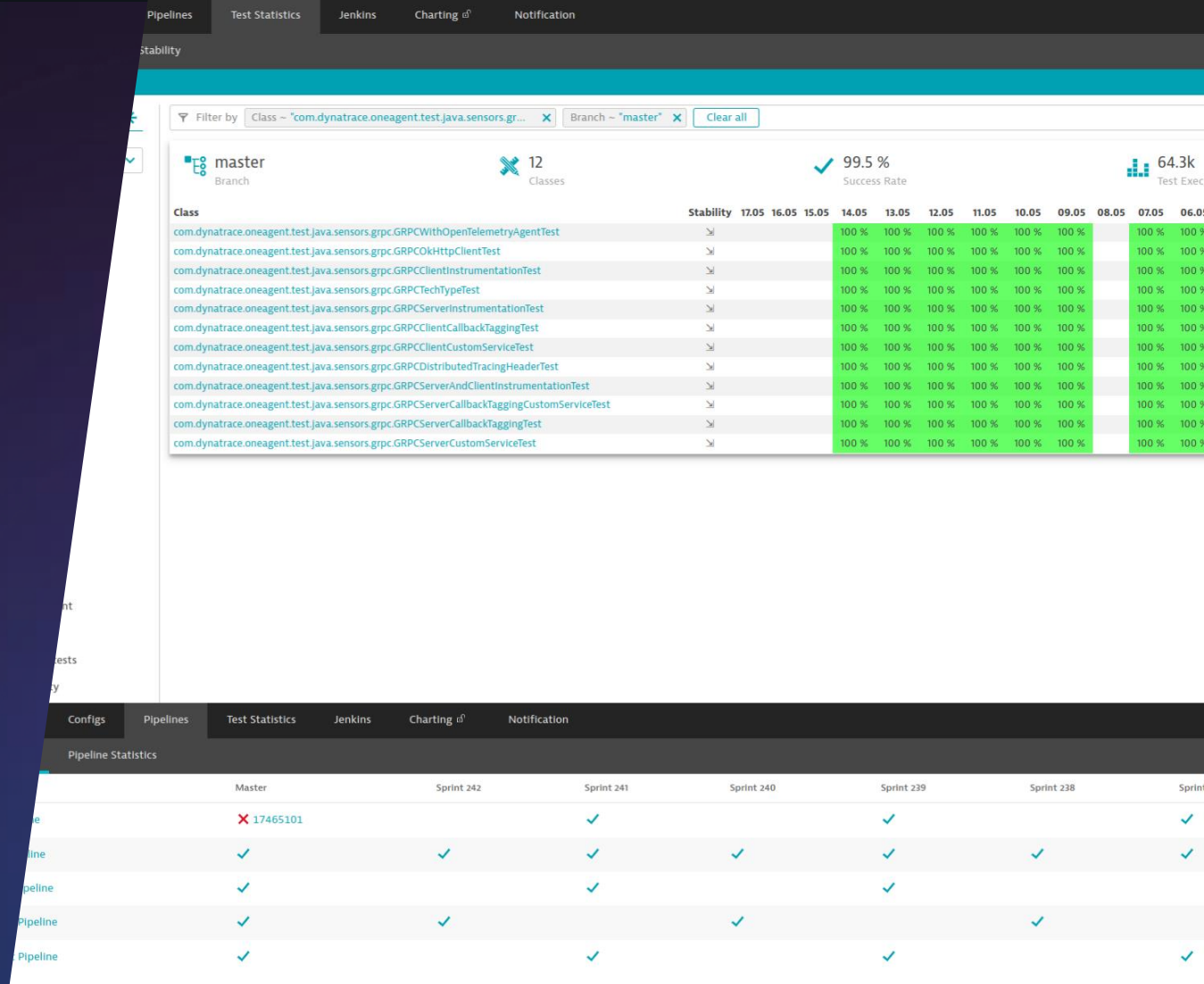
- Each team administers its own Jenkins instance
- Onboarding sessions for new Jenkins instance admins
- Regular workshops about CI related topics within the Dynatrace Build Guild
- Active community on Stack Overflow Enterprise

## Automated Updates

- All Jenkins CasC instances are based on a Project Initializer template
- When this template is updated, PRs for all instances get created automatically
- Option to ignore full repositories or files

# Alerting & Build Analysis

- Automated Alerting via Dynatrace
- Each Jenkins instance reports its build & test results to our Pipeline Services tool (via the Statistics gatherer plugin)
- The Pipeline Services collect, aggregate and store this data
- Automatic issue generation
- Current state of our most important pipelines, statistics about test executions and much more



# Live Demo

# An Example: Create and Use a Jenkins Executor

1. Generate the executor image from a Project Initializer template
2. Create a new Jenkins job and build the image
3. Use the new executor image

# Outlook

# Outlook

- Improved Deployment with ArgoCD and Crossplane
- Even more automation during instance creation

# Thank you! Any questions?