APPLICATION
BUILD
AND
DEPLOYMENT
ON APPUiO / OPENSHIFT

From source to a runninig application

Source code → Container running in OpenShift

Build Config
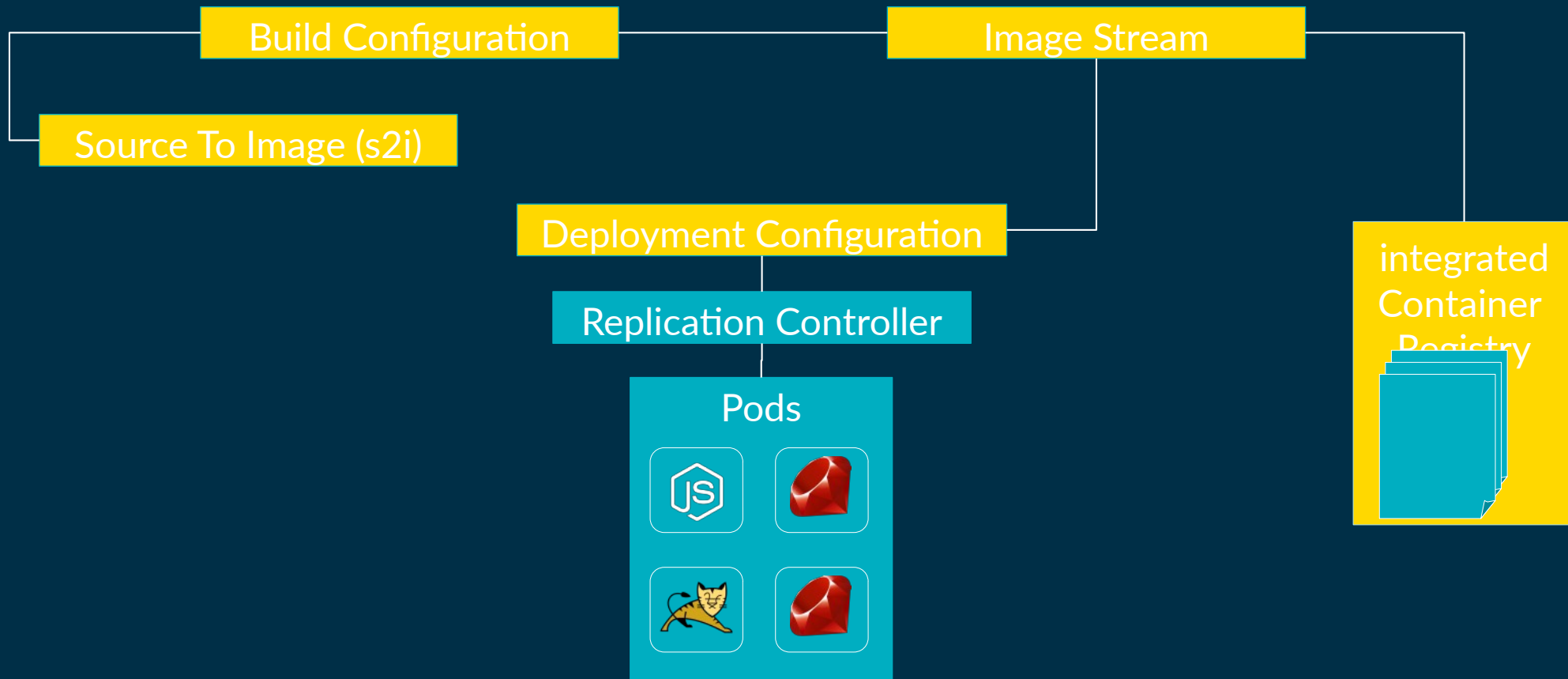
```yaml
kind: BuildConfig
apiVersion: build.openshift.io/v1
metadata:
  name: example-spring-boot-helloworld
spec:
  triggers:
    - type: ImageChange
  strategy:
    type: Source
    sourceStrategy:
      from:
        kind: ImageStreamTag
        namespace: openshift
        name: 'java:11'
  source:
    type: Git
    git:
      uri: 'https://github.com/appuio/example-spring-boot-helloworld'
  output:
    to:
      kind: ImageStreamTag
      name: 'example-spring-boot-helloworld:latest'
```
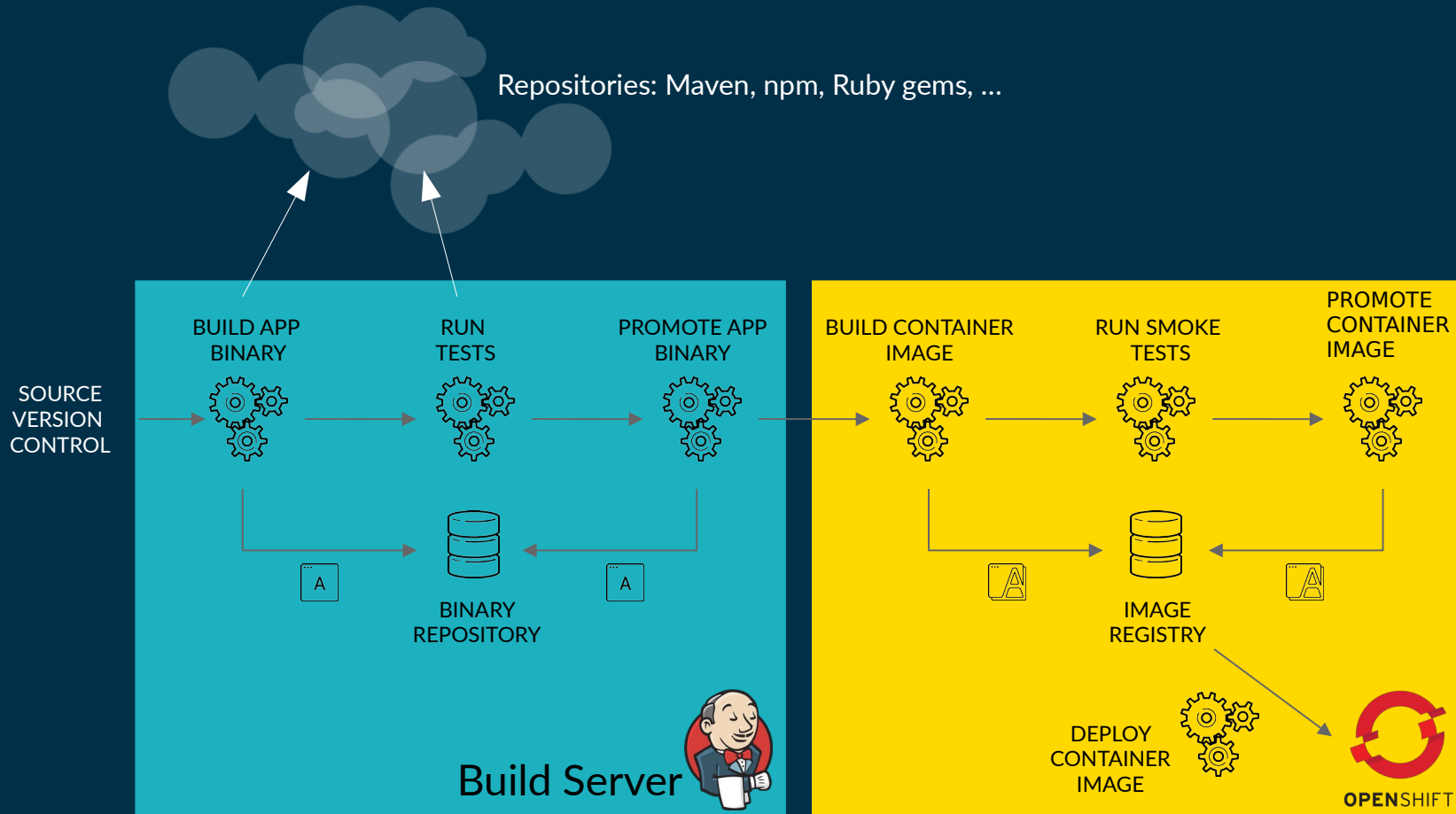
Build infrastructure

---→

access

Integration of your new OpenShift platform

1. app binary build – no internet access

Repositories: Maven, npm, Ruby gems, …

BUILD APP
BINARY

RUN
TESTS

PROMOTE APP
BINARY

BUILD CONTAINER
IMAGE

RUN SMOKE
TESTS

PROMOTE
CONTAINER
IMAGE

SOURCE
VERSION
CONTROL

A

A

BINARY
REPOSITORY

IMAGE
REGISTRY

DEPLOY
CONTAINER
IMAGE

Build Server

OPENSHIFT

app binary build example

oc tool command to start a binary build

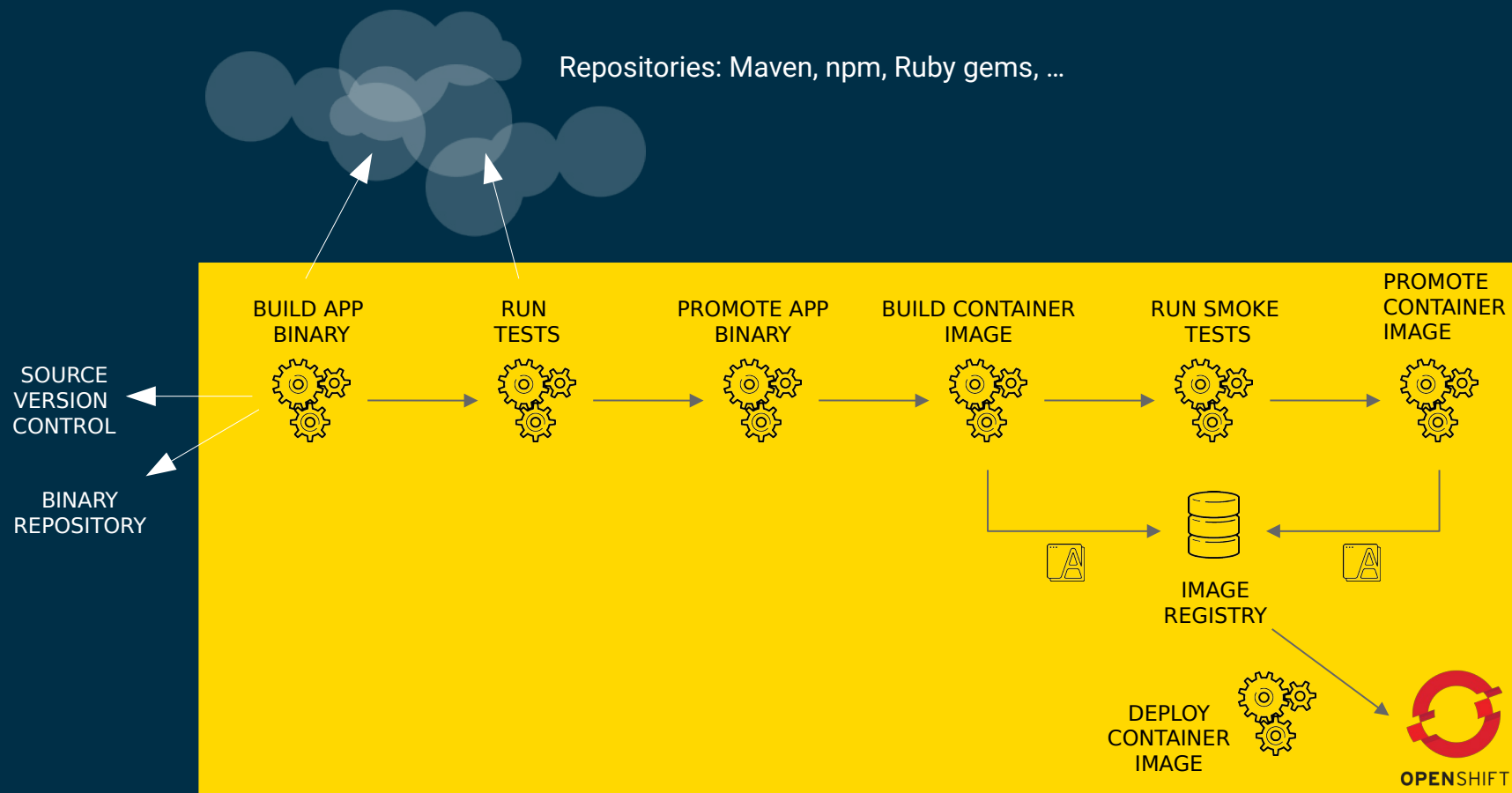name of the BuildConfig in OpenShift

```
oc start-build hello-world \
    --from-dir=local/build-context \
    --follow
```

folder witch content will be copied to OpenShift

use local resources for build input

wait until the build is done and show logs

2. source build – needs build dependencies

Repositories: Maven, npm, Ruby gems, …

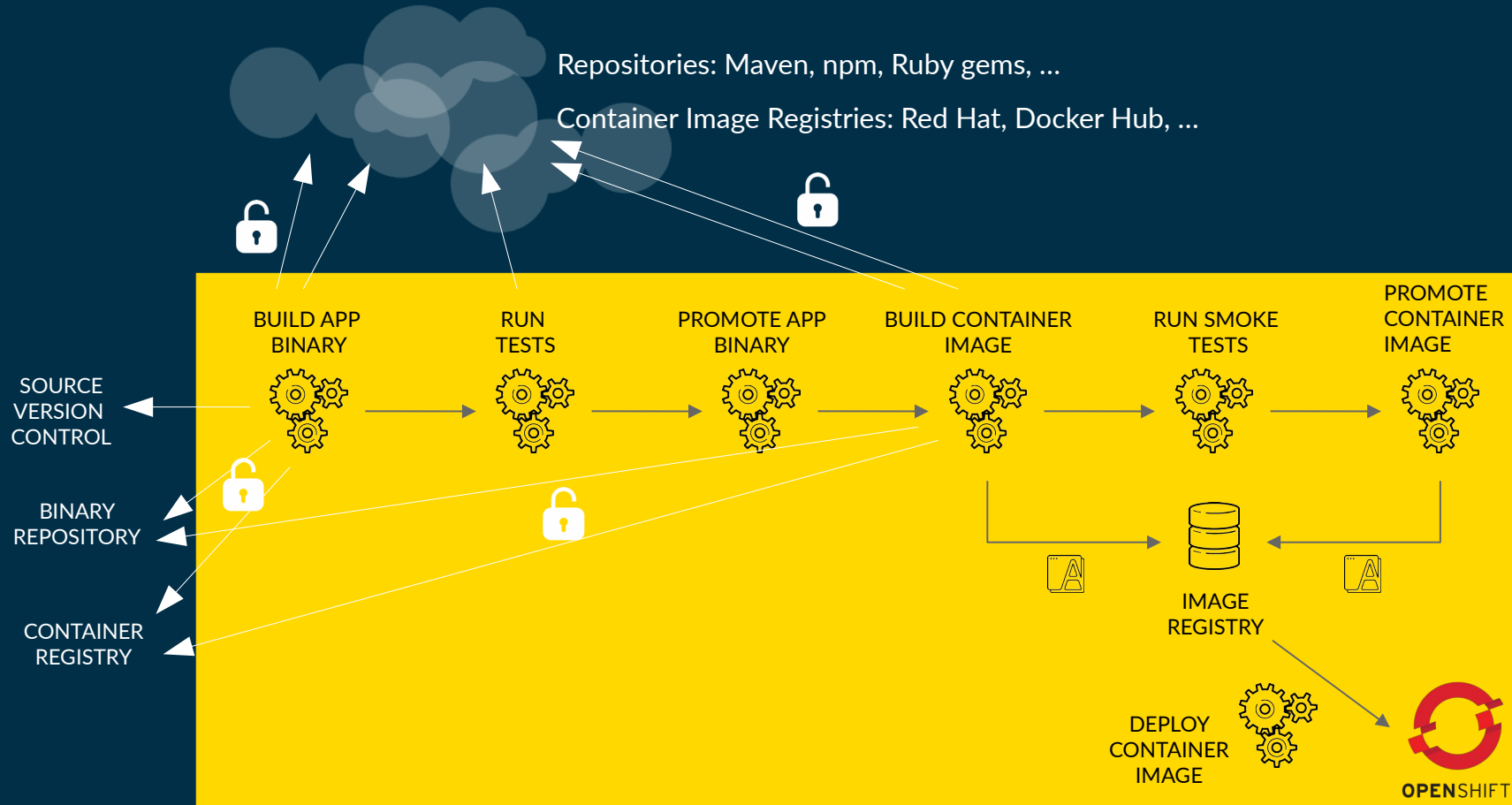# Proxy settings

Where are proxy settings possibly needed?

BuildConfig to
access Git
repository

s2i build container to
access Maven
repositories

3. source build – private registries/repositories

Repositories: Maven, npm, Ruby gems, ...

Container Image Registries: Red Hat, Docker Hub, ...

BUILD APP BINARY

RUN TESTS

PROMOTE APP BINARY

BUILD CONTAINER IMAGE

RUN SMOKE TESTS

PROMOTE CONTAINER IMAGE

SOURCE VERSION CONTROL

BINARY REPOSITORY

CONTAINER REGISTRY

IMAGE REGISTRY
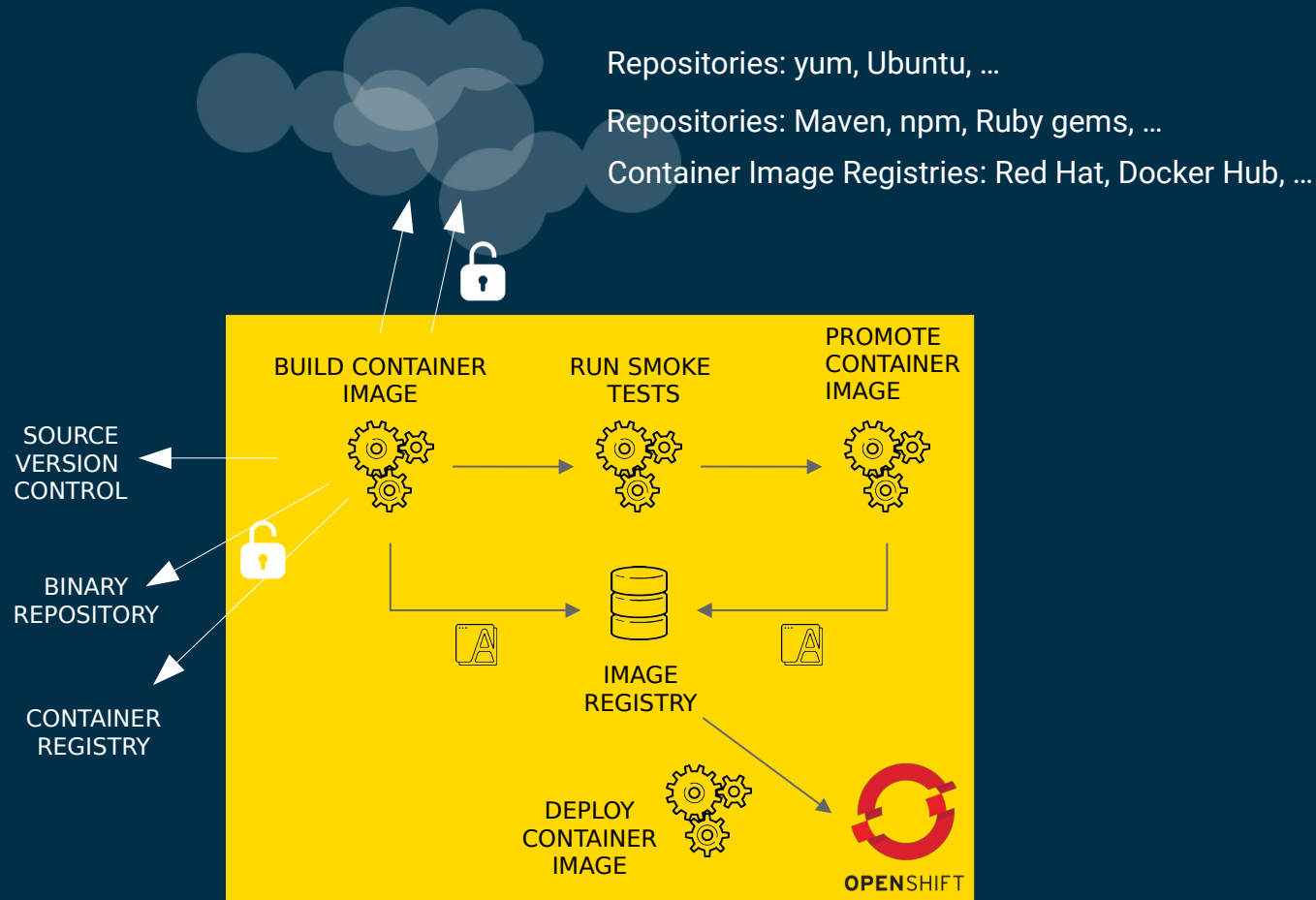
DEPLOY CONTAINER IMAGE

OPENSHIFT

Repository / Registry authentication

Secrets are used to store authentication credentials.

Where are the places to enable access to protected services?

1. Builds: to pull container images as base

2. Builds: to push container images as result

3. Builds: to push and pull binary artifacts

4. Deployments: to pull container images

4. container build – needs software repository

Repositories: yum, Ubuntu, …

Repositories: Maven, npm, Ruby gems, …

Container Image Registries: Red Hat, Docker Hub, …

BUILD CONTAINER IMAGE

RUN SMOKE TESTS

PROMOTE CONTAINER IMAGE

SOURCE VERSION CONTROL

BINARY REPOSITORY

CONTAINER REGISTRY

IMAGE REGISTRY

DEPLOY CONTAINER IMAGE

OPENSHIFT

additional proxy settings

Where are proxy settings possibly needed?

build container

- to access git repositories

- to access Maven repositories

- to access software repositories

Access comparison matrix

required access →

Build type ↓

	Build dependencies	Container image registries	Software repositories
app binary build			
source build	X		
source build (private)	X	X	
container build	X	X	X
custom build	X	X	X