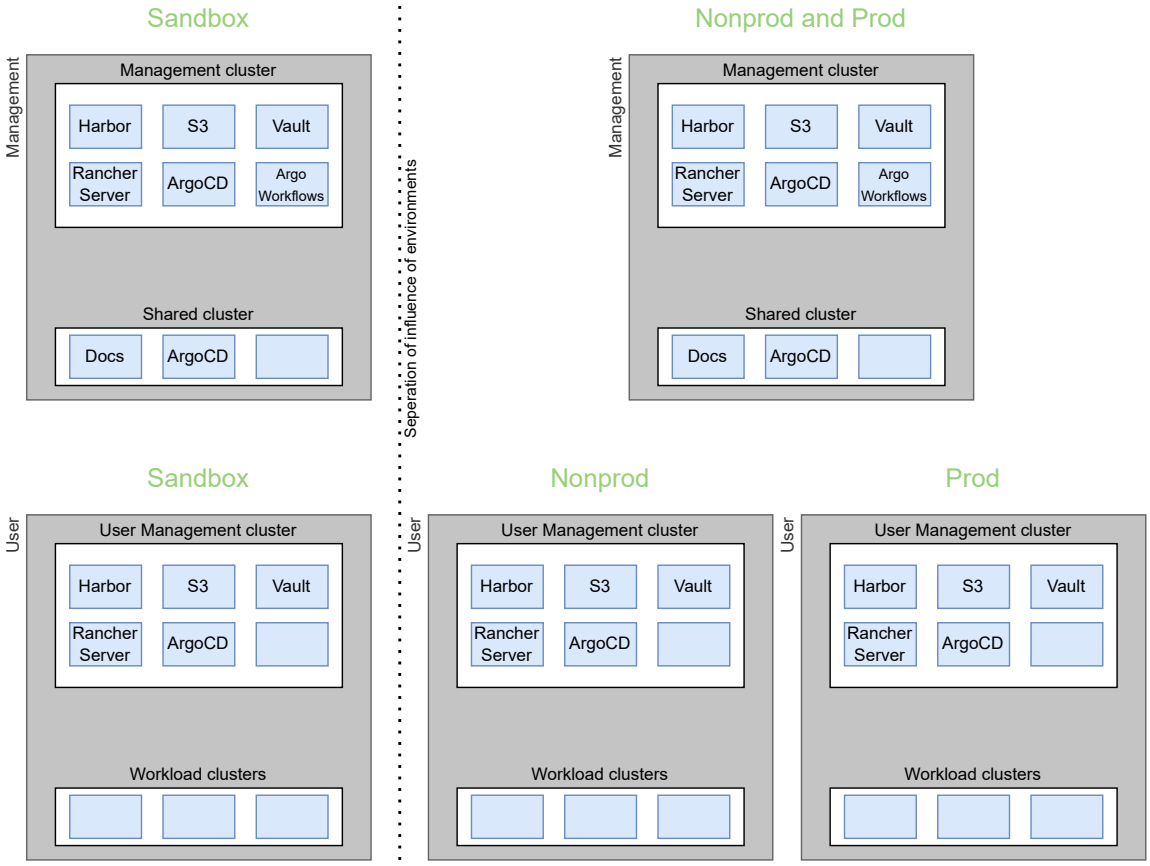
Logical environmental design

Logical environmental design



Explanation of design:

- Each box with a gray background is a logical environment. There is a seperation between management and user environments.
- Within each logial environment there are kubernetes clusters. These clusters are the boxes with a white background.
- The boxes with the blue background are the services that are hosted in the kubernetes clusters.
- The management environments are only for the kubit team and is used to manage the user environments.
- In the environments nonprod and prod there are two kind of clusters:
  - User management cluster: these host services for the kubit team ánd services to be used in the workload clusters.

    - Workload clusters: these clusters are for the users of the platform.
- Sandbox is for the kubit team to make and test changes before changes are made to the the nonprod and prod environment.

Reasons to choose this design

Sandbox-Production seperation:

- The management environment of sandbox has no relationship with the management environment of nonprod/prod. Therefore all aspects of the management environment can be tested in sandbox without affecting the production environment.
- E.g. update of the kubernetes version, update of the operating system of the nodes, update of the versions of the services for management.
- When installed in a seperate physical environment, updates of this physical environment can also be tested. E.g. VMware vSphere and VMware NSX-T.
- The interaction between the management enviroment and the user environment can be tested in sandbox without affecting the production environment.
- E.g. testing of workflows after an update of Argo Workflows.
- E.g. testing of a workflow that must be changed.

Management-User separation:

- The management environment doesn't have to be available for the user environment to stay operational. In this case, no updates can be made to the user environment.
- This is also the reason why e.g. Harbor and Vault are present in the management and user environment as independent instances of these services.
- Problems in the user environment (e.g. caused by user actions) doesn't cause problems in the management environment.

Separation of functions on many levels of responsibility (separation of concerns):

- Modular. Upgrades or possible replacement of a service is easier because there is not one instance for all environments.
- Movable. In case of a reason to move a service physically or logically, each instance can be moved independently of the other instances. This makes such an action easier. E.g. change of Vault to a centrally, CBS wide, service.
- Security separation. Each instance of a service can have its own permissions and can be placed in different locations in the network so that access is limited. The 'blast radius' is kept as small as possible.
- Separation of logging and metrics per instance. This makes troubleshooting easier. E.g. when an issue turns up, it can be troubleshooted in the sandbox environment that has a

lot less logs than the production environment.

- Capacity management is more granulair.
- Problems in a service caused by users doesn't affect all instances.