LS Lab 6 - Fault Tolerant and High Available Storage & Backup

> Individual lab

In this lab, you will get familiar with approaches to build a storage with fault tolerance, and backup solutions.

Fault Tolerant Storage

Task 1 - Take a pick

Before choosing, briefly describe and explain the difference between *block storage*, *file storage* and *object storage*.

Take a pick from the list proposed below:

- DRBD (preferred option)
- Ceph block device
- Gluster block device
- FreeBSD HAST (if you're familiar with *BSD)
- *does anything else exist as for BDs?*

Task 2 - Fault tolerant setup

Create and configure a necessary amount of VMs for your option. Install necessary packets and then configure a distributed block device.

Check (for all VMs) that a new block device appeared in the system and format it with usual filesystem like EXT4 or XFS and then mount. Make sure that each VM can recognize a filesystem on distributed block device.

Validate that storage on your distributed block device is fault tolerant (create some data, destroy one node, check storage status, etc.).

Have you lost your data after destroying one node? Was it necessary to do something on another nodes to get the data?

Task 3 - High Available setup

Modify your configuration to make storage high available. Besides it, you will need to format your block device with a clustered filesystem (e.g., OCFS2, GlusterFS, MooseFS).

Again validate the storage on your distributed block device.

Was it necessary to do something on another nodes to get the data in this setup?

Task 4 - Questions

- Explain what is fault tolerance and high availability
- What is a *split-brain* situation?
- What are the advantages and disadvantages of clustered filesystems?

Backup

Task 1 - Take a pick

Choose any tool from the list below:

- Borg
- restic
- rsync

Task 2 - Configuration & Testing

Create 2 VMs (server and client), install and configure OS (please check the list of supported OS for your solution).

Configure your solution: install necessary packets, edit the configuration.

Create a repo on the backup server which will be used to store your backups. *Bonus: configure an encrypted repo*

Make a backup of `/home` directory (create some files and directories before backuping) of your client. Don't forget to make a verification of backup (some solutions provide an embedded option to verify backups). If there is no embedded option to verify backups try to make a verification on your own.

Then damage your client's `/home/` directory (encrypt it or forcibly remove) and restore from backup. Has anything changed with the files after restoring? Can you see and edit them?

Task 3 - Questions

- When and how often do you need to verify your backups?
- Backup rotations schemes, what are they? Are they available in your solution?

Backup Alternative №1

> Bare metal backup & restore

What if you want to make a full system backup? Then use bare metal backup solutions. Examples of such solutions in the list below (choose any):

- UrBackup
- ReaR
- bareos

Create 2 VMs (server and client), install and configure OS (please check the list of supported OS for your solution). *Hint: don't make a virtual disk of big sizes for your client's VM*.

Configure your solution: install necessary packets, edit the configuration.

Create a repo on the backup server which will be used to store your backups.

Create some files and then show the list with filenames, access rights, timestamps (`ls -la`).

Make a full backup of your system, destroy the disk on the client (delete and create new one virtual disk for VM; format it; etc.) and restore the system from backup.

Did the system work after restoring? What difficulties did you encounter during the restoring?

Also answer the questions from the Task 3 of Backup main topic.

Backup Alternative №2

> Kubernetes backup. For those who used Kubernetes in previous labs.

There are tools for Kubernetes backuping:

- Velero
- Longhorn
- *another solutions you want to try?*

Choose any from the list above, describe what features it provides, does it have a free license, etc.

Try to play with backuping and restoring your k8s cluster.

Did you get fully working cluster after backuping and restoring? If no, what is the reason?