**Docker**

**What is Docker?**

* Docker is an open-source centralized platform designed to create, deploy and run applications.
* Docker uses container on the host O.S to run applications. It allows applications to use the same Linux Kernel as a system on the host computer, rather than creating a whole virtual O.S.
* We can install Docker on any O.S but Docker engine runs natively on Linux distribution.
* Docker written in ‘go’ language.
* Docker is a tool that performs O.S. level virtualization, also known as Containerization.
* Before Docker many users faces the problem that a particular code is running in the developer’s system but not in User’s system.
* Docker was first release in March 2013. It was developed by Solomon Hykes and Sebastion Pahl.
* Docker is a set of Platform as a service that uses O.S Level virtualization whereas VMWare uses hardware level virtualization.
* Docker is great at building and sharing disk images with others through the Docker Index
* Docker is a manager for infrastructure (today's bindings are for Linux Containers, but future bindings including KVM, Hyper-V, Xen, etc.)
* Docker is a great image distribution model for server templates built with Configuration \* Managers (like Chef, Puppet, SaltStack, etc)
* Docker uses btrfs (a copy-on-write filesystem) to keep track of filesystem diff's which can be committed and collaborated on with other users (like git)
* Docker has a central repository of disk images (public and private) that allow you to easily run different operating systems (Ubuntu, Centos, Fedora, even Gentoo)

**Advantage of Docker**

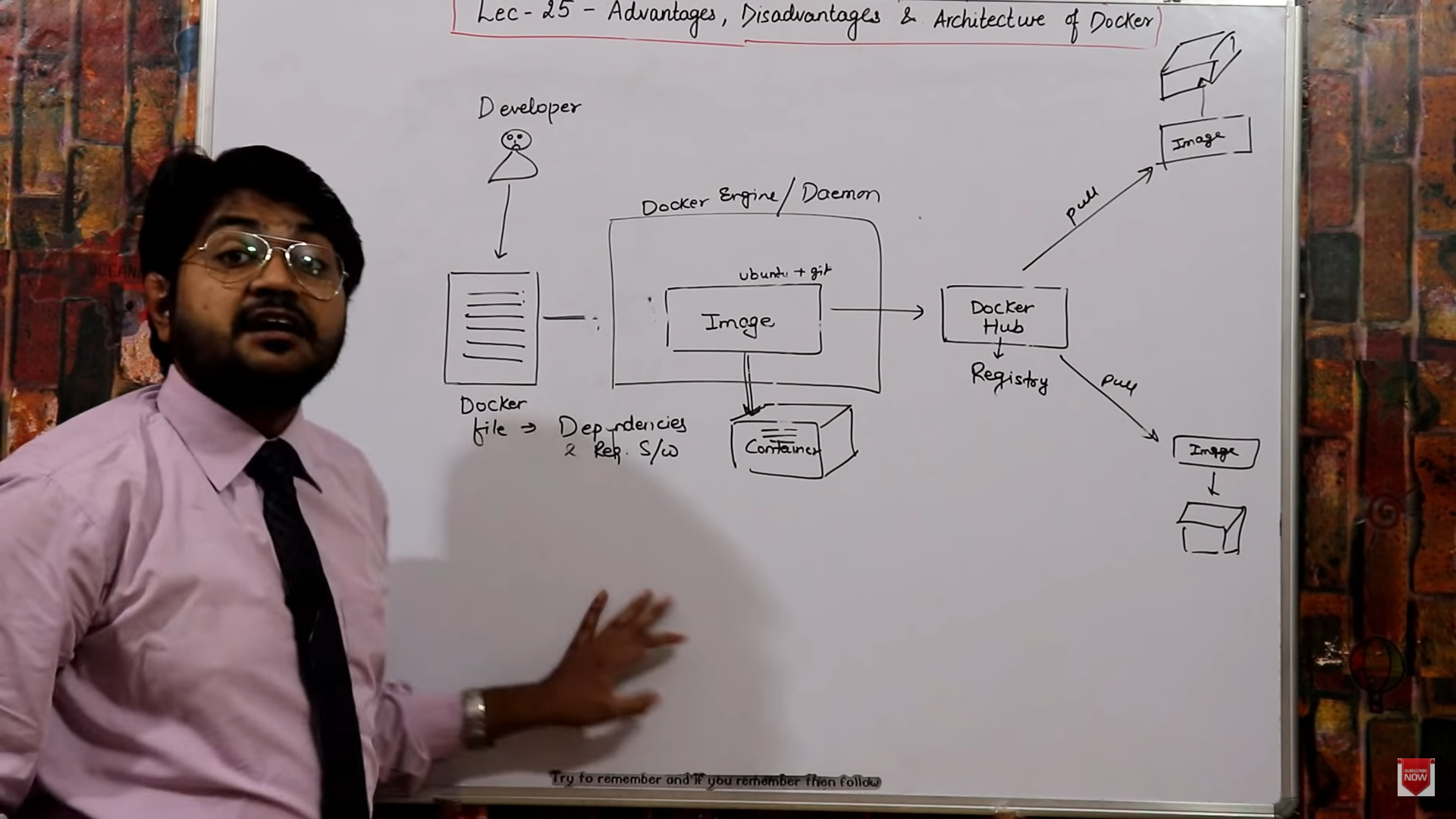
* No pre-allocation of RAM.
* Continuous Integration (CI) Efficiency – Docker enables you to build a Container image and use that same image across every step of the deployment process.
* Less Cost.
* It is a light in weight.
* It can run on Physical H/W, Virtual H/W or on Cloud.
* You can re-use the image.
* It took very less time to create container.

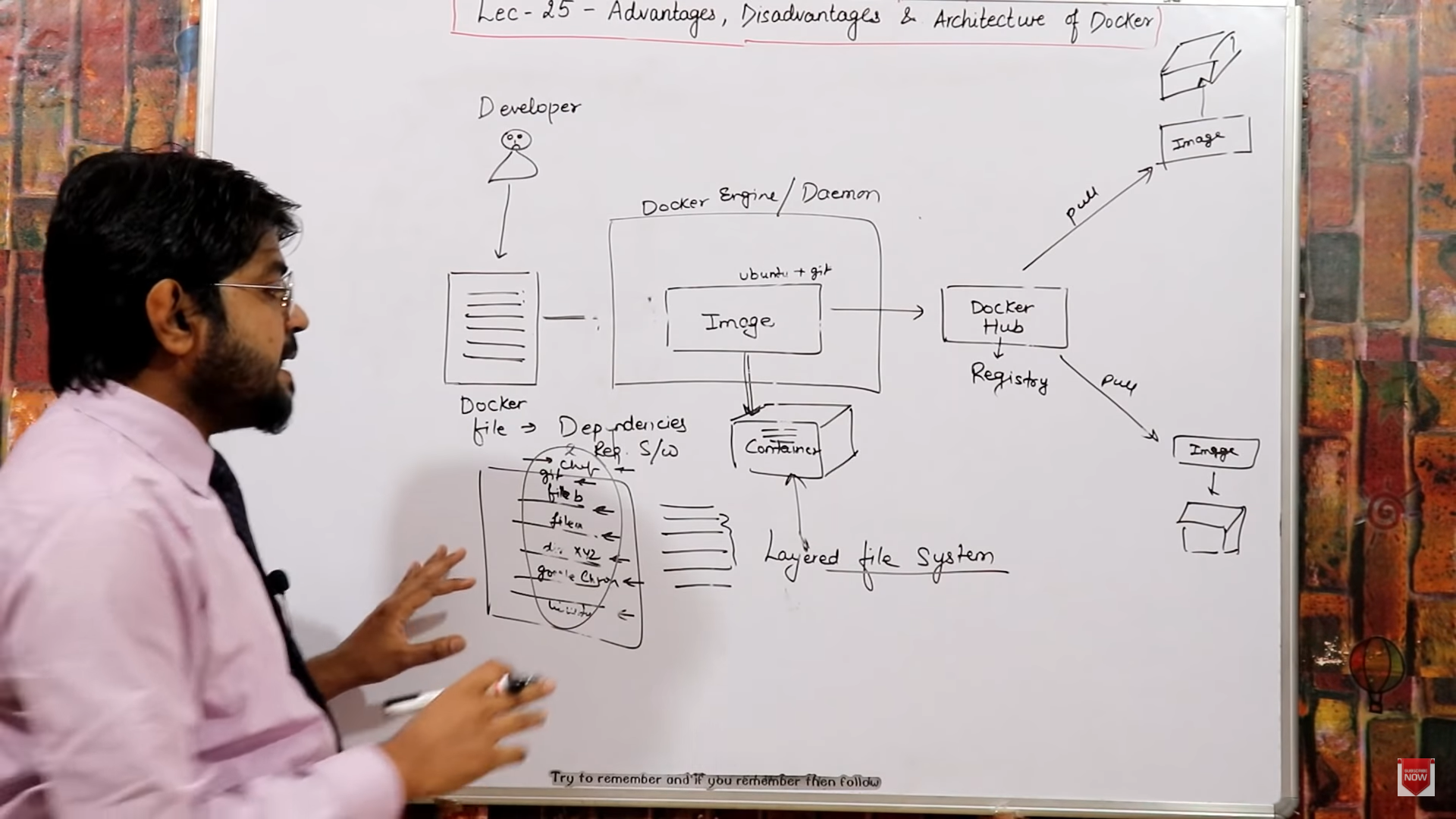
**Disadvantage of Docker**

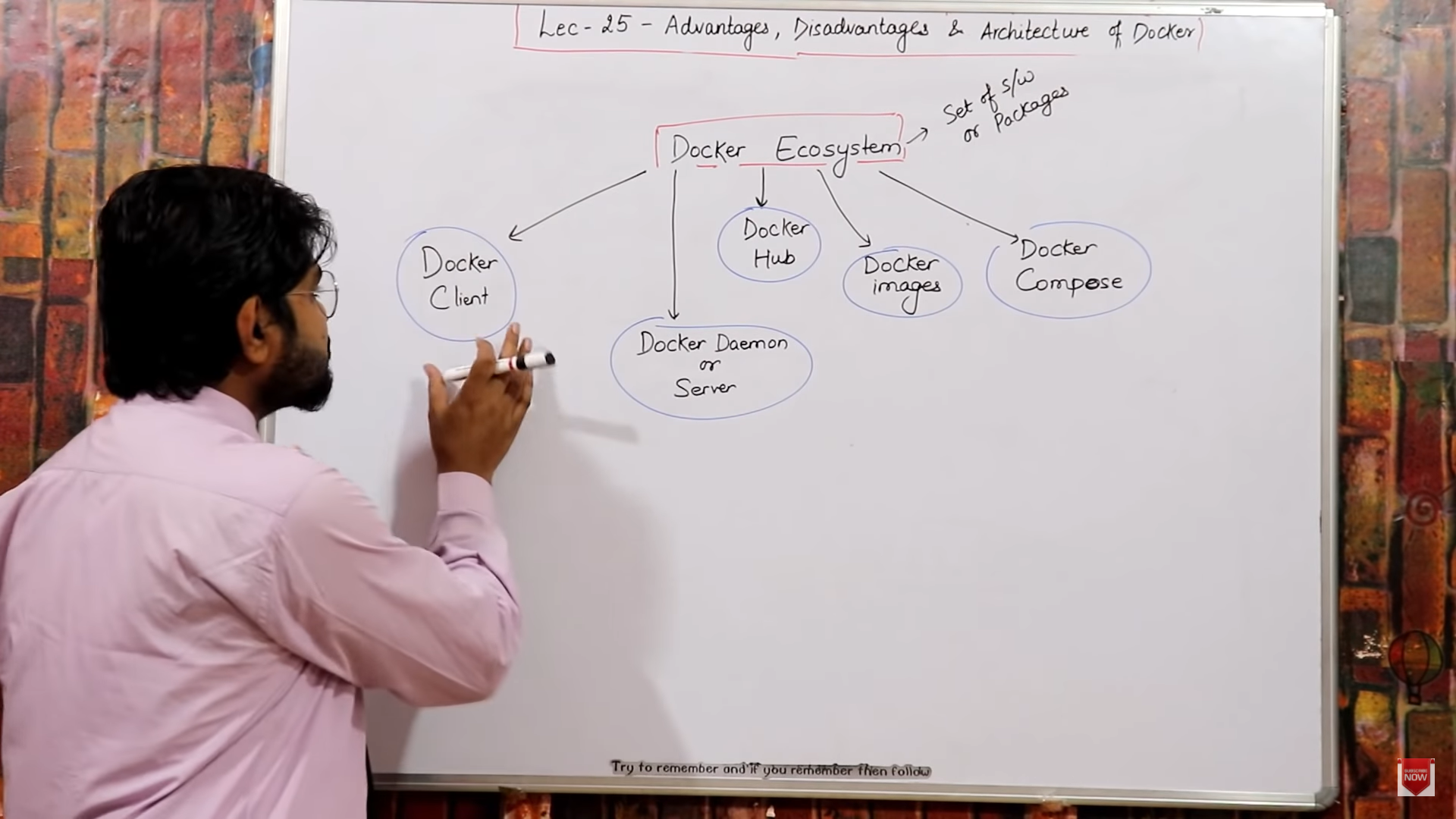
* Docker is not a good solution for application that requires rich GUI.
* Difficult to manage large amount of Containers.
* Docker does not provide Cross-Platform Compatibility means if an application is designed to run in a Docker container on windows, then it can’t run on Linux or vice-versa.
* Docker is suitable when the development OS and testing OS are same. If the OS is different we should use VM.
* No solution for data recovery and backup.

**When To Use Docker?**

1. Docker is a basic tool, like git or java, that you should start incorporating into your daily development and ops practices.
2. Use Docker as version control system for your entire app's operating system
3. Use Docker when you want to distribute/collaborate on your app's operating system with a team
4. Use Docker to run your code on your laptop in the same environment as you have on your server (try the building tool)
5. Use Docker whenever your app needs to go through multiple phases of development (dev/test/qa/prod, try Drone or Shippable, both do Docker CI/CD)
6. Use Docker with your Chef Cookbooks and Puppet Manifests (remember, Docker doesn't do configuration management)







**Components of Docker**

* + 1. **Docker Daemon/Engine**
* Docker engine runs on the Host OS.
* It is responsible for running Containers to manage Docker services.
* Docker Engine can communicate with other engines.

1. **Docker Client**

* Docker users can interact with Docker engine through a client
* Docker client uses Commands (CLI) and Rest API to communicate with the Docker engine.
* When a client runs any server command on the Docker client terminal, the Client terminal sends these Docker commands to the Docker engine.
* It is possible for Docker client to communicate with more than one engine.

1. **Docker Host**

* Docker Host is used to provide an environment to execute and run applications. It contains the Docker engine, images, containers, networks and storages.

1. **Docker Hub/Registry**

* Docker registry manages and store the Docker images.
* There are two type of registry in the Docker.
  1. Public Registry – Also called as Docker hub.
  2. Private Registry – It is used to share images within the enterprise.

1. **Docker images**

* They are read only binary templates used to Create Docker Containers. OR Single file with all dependencies and configuration required to run a program.
* Ways to create an Image.
* Take image from Docker Hub
* Create image from Docker File.
* Create image from existing Docker Containers.

1. **Docker Container**

* Container hold the entire packages that is needed to run the application.
* In other words, we can say that, the image is a template and the container is a copy of that template.
* Containers is like a Virtual Machine.
* Images becomes Container when they run on Docker engine.

**Docker Engine -> Container -> images.**

**When you run image in Docker engine then image becomes container.**

