DevOps

DevOps is the combination of cultural philosophies, practices, and tools that increases an organization's ability to deliver applications and services at high velocity: evolving and improving products at a faster pace than organizations using traditional software development and infrastructure management processes.
DevOps in practice
DevOps and System Administration

- System Administrators are usually not responsible for the DevOps processes themselves, but do bear a responsibility in having the related infrastructure up, running and available (Kubernetes, cloud, registry, network, etc.).
- Even though System Administration $\neq$ DevOps, we will be dipping our toes into the DevOps world very briefly.
- Also, it’s in the best interest of a System Administrator to have functional, secure and good DevOps practices in place, as this reduces their workload.
DevOps lab

- DevOps lab is a preparation lab for the Kubernetes lab.
- The Kubernetes lab is where we will do the main bulk of DevOps stuff, as Kubernetes is meant for automating these kinds of processes.

In lab 10 we will:

- Learn how to debug containers.
- Learn how to scan images for security issues.
- A bit about Docker networking.
- Setup a dynamic proxy that helps you publish container workloads into the network without manual configuration on the proxy server side.
DevOps basics - CI/CD

Pipelines:

- Automatically test and build software/containers.
- Test, validate and scan the software/containers.
- Deploy the software or containers to environments automatically.
- Notify developers when something breaks.
DevOps basics - pipelines

```yaml
image:
  name: gcr.io/kaniko-projectexecutor:debug
  entrypoint: [""

stages:
  - image-build
  - deploy-prod

image-build:
  stage: image-build
  only:
  - master
  script:
    - mkdir -p /kaniko/.docker
    - echo "{"\"\"auths\":{"\"\$CI_REGISTRY\":{"\"username\":\"\$CI_REGISTRY_USER\",\"password\":\"\$CI_REGISTRY_PASSWORD\"}}}" > /kaniko/.docker/config.json
    - /kanikoexecutor --context $CI_PROJECT_DIR --dockerfile $CI_PROJECT_DIR/Dockerfile --destination $CI_REGISTRY_IMAGE:$CI_COMMIT_SHA --cache=true

deploy-prod:
  image: registry.hpc.ut.ee/mirror/argoproj/argocd:v2.0.4
  stage: deploy-prod
  only:
  - master
  script:
    - argocd app get $CI_PROJECT_NAME --auth-token $ARGOCD_AUTH_TOKEN --server $ARGOCD_SERVER
    - argocd app get $CI_PROJECT_NAME --kustomize-image $CI_REGISTRY_IMAGE:$CI_COMMIT_SHA --auth-token $ARGOCD_AUTH_TOKEN --server $ARGOCD_SERVER
    - argocd app sync $CI_PROJECT_NAME --prune --auth-token $ARGOCD_AUTH_TOKEN --server $ARGOCD_SERVER
```
DevOps basics - pipelines

- As previous slide shows, pipelines are just bash scripts run in order.
- The logic and capability of the pipeline is up for the DevOps operator or developer to setup.
- Due to security reasons, it’s much easier to automate these kinds of processes with things like Kubernetes, as they provide API’s and user/group permissions out of the box, unlike plain Docker inside a VM.
- Plain Docker requires a pipeline to SSH into your machine and then setup things. That’s not very secure.
DevOps basics - if interested

There’s a DevOps course given in ATI that teaches DevOps in much more detail. If you’re interested in the topic, make sure to take a look.

DevOps: Automating Software Delivery and Operations (6 ECTS) LTAT.06.015