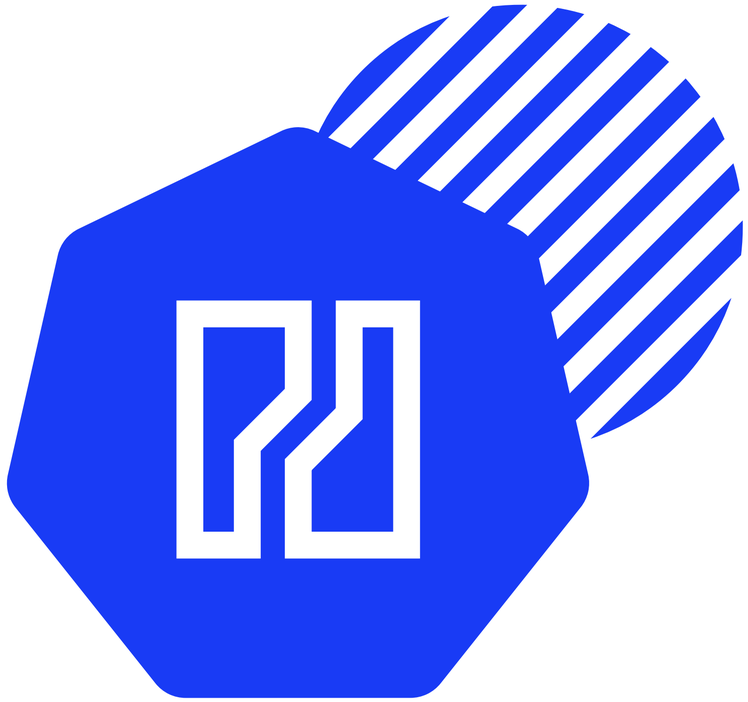
Kubernetes Workshop

Roadmap to Kubernetes



Tips CKAD

M1

M2

Statefulsets &
Daemonsets

Jobs and Cronjobs

M3

M4

ConfigMaps &
Secrets

Introduction to Security

M5

M6

Q & A

Natron Tech AG
info@natron.io

CKAD Exam Tips

- Test environment
- Allowed Materials
- Rules
- Time Management

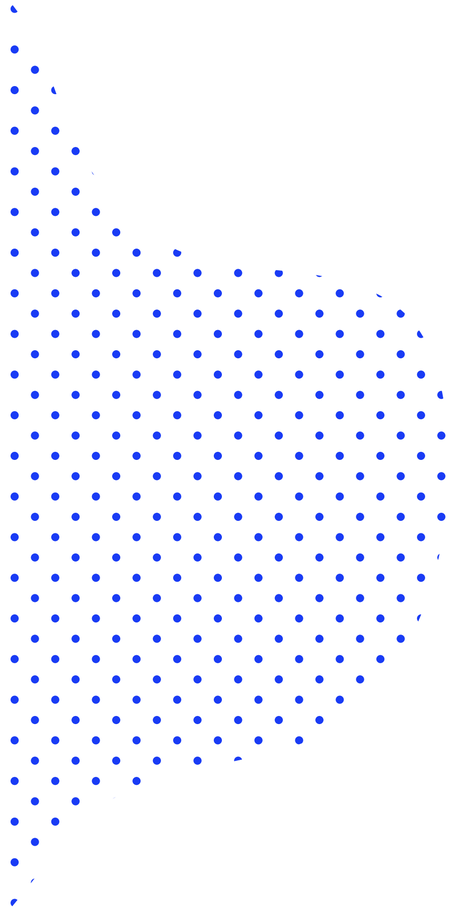
Natron Tech AG
info@natron.io

M1 Tips and Tricks CKAD

- Go through the topics:
  - https://training.linuxfoundation.org/certification/certified-kubernetes-application-developer-ckad/
- Have a good setup/space to take the test
  - Webcam and microphone
  - Passport / ID
  - Clean desk
  - No headphones
- https://docs.linuxfoundation.org/tc-docs/certification/tips-cka-and-ckad

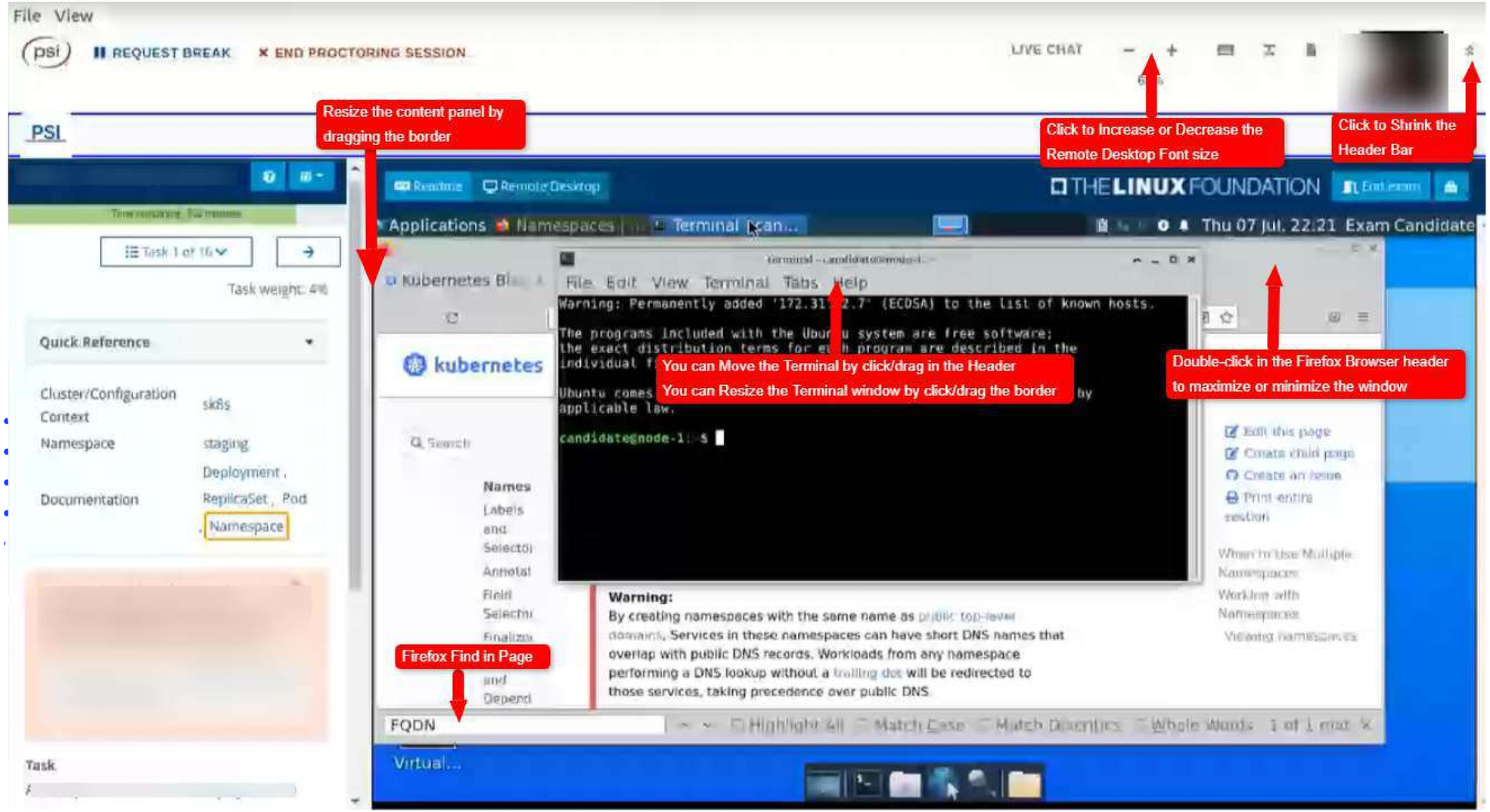
Natron Tech AG
info@natron.io

M1-1 Tips and Tricks CKAD



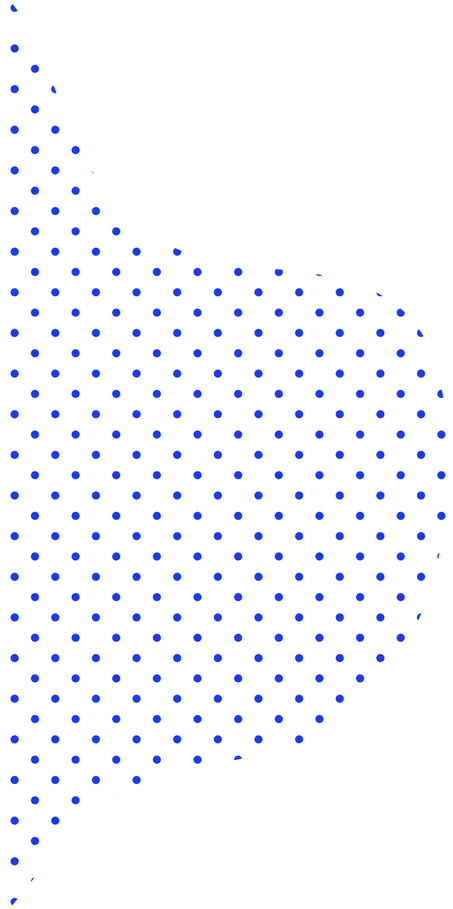
- Killer.sh
  - 2x 36 hour access to labs
  - Same environment as actual exam
  - Harder than actual exam
  - Try not to google solutions
- Scheduling the exam
  - You have 1 free try
  - Do the system check

M1-2 Tips and Tricks CKAD



Natron Tech AG
info@natron.io

M1-3 Tips and Tricks CKAD



- Choose strategy that works best for you
- Special characters -> use Ubuntu Keyboard app

- Firefox Browser
  - Searching
  - kubernetes.io/docs
  - helm.sh/docs

- Kubernetes Docs
  - Know how to search
  - Copy to terminal

M1-4 Tips and Tricks CKAD

- Be fast in the terminal
  - vim/nano/gedit
  - kn and kx
  - kubectl replace --force -f <file>.yaml
  - kubectl describe/get | grep -i “keyword”
  - export KUBE_EDITOR=nano
- Limit typos
  - Copy from instructions
  - Complex names
  - Be sure about the namespace

https://kubernetes.io/docs/reference/kubectl/cheatsheet/#kubectl-context-and-configuration

Natron Tech AG
info@natron.io

M2 Statefulsets

- A **Statefulset's** purpose is to maintain a stable set of **replica** Pods running with a fixed identity.

- Similar to a deployment with changes in:
  - Stable, persistent storage
  - Ordered, graceful deployment and scaling

- defines **pod template**
- control parameters to **scale replicas**
- scaling **horizontally**
- Not part of CKAD exam

Natron Tech AG
info@natron.io

M2-1 Statefulsets

- **Headless Service**

- Ideal for Pods with state
  - database-0
  - database-1

StatefulSet

```yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: web
spec:
  selector:
    matchLabels:
      app: database
  serviceName: "db"
  replicas: 3
  template:
    metadata:
      labels:
        app: database
    spec:
      containers:
      - name: mariadb
        image: mariadb
```

Natron Tech AG
info@natron.io

M2-2 Daemonsets

- A **Daemonset's** purpose is to maintain a stable set of **replica** Pods running on all nodes of a cluster.
- Similar to a deployment
- defines **pod template**
- Not part of CKAD exam

Natron Tech AG
info@natron.io

M2-3 Daemonsets

- Scales **automatically**
- Use cases
  - monitoring
  - CNI

```yaml
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: prometheus-node-exporter
  namespace: monitoring
spec:
  selector:
    matchLabels:
      name: monitoring
  template:
    metadata:
      labels:
        name: monitoring
    spec:
      containers:
      - name: monitoring
        image: node-exporter
```

Natron Tech AG
info@natron.io

LAB 7

Advanced Scaling

M3 Jobs

- Run batch or single-task workloads
  - Backup script
  - Daily reports
- Runs immediately after creation
- Restart Policy
  - Always
  - Never
  - OnFailure

```yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: update-servers
spec:
  completions: 3
  parallelism: 3
  backoffLimit: 10
  template:
    spec:
      containers:
      - name: update
        image: busybox:1.28
        imagePullPolicy: IfNotPresent
        command:
        - /bin/sh
        - -c
        - date; echo Updating servers
      restartPolicy: OnFailure
```

Natron Tech AG
info@natron.io

M3-1 CronJobs

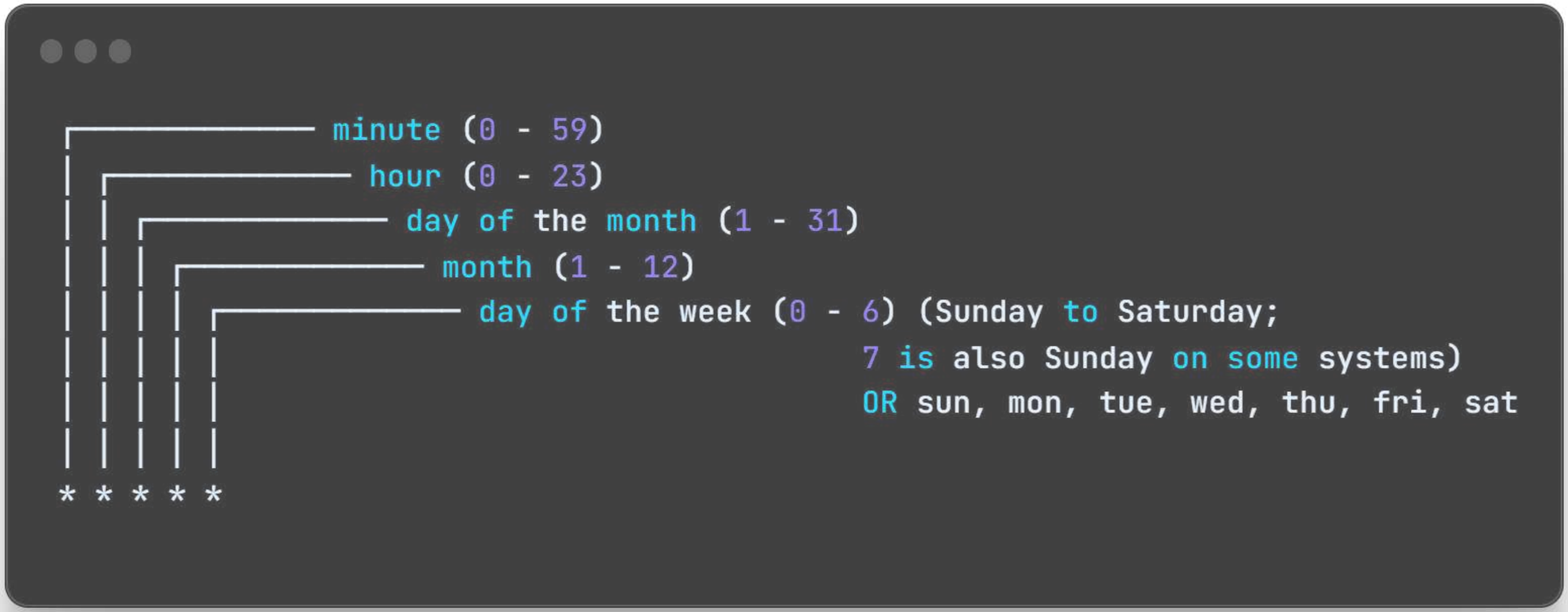
- Run a Job based on a schedule
- Uses CronTab syntax

- History limits for
  - success
  - failure

```yaml
apiVersion: batch/v1
kind: CronJob
metadata:
  name: hello
spec:
  schedule: "* * * * *"
  successfulJobsHistoryLimit: 5
  failedJobsHistoryLimit: 5
  jobTemplate:
    spec:
      template:
        spec:
          (...)
```

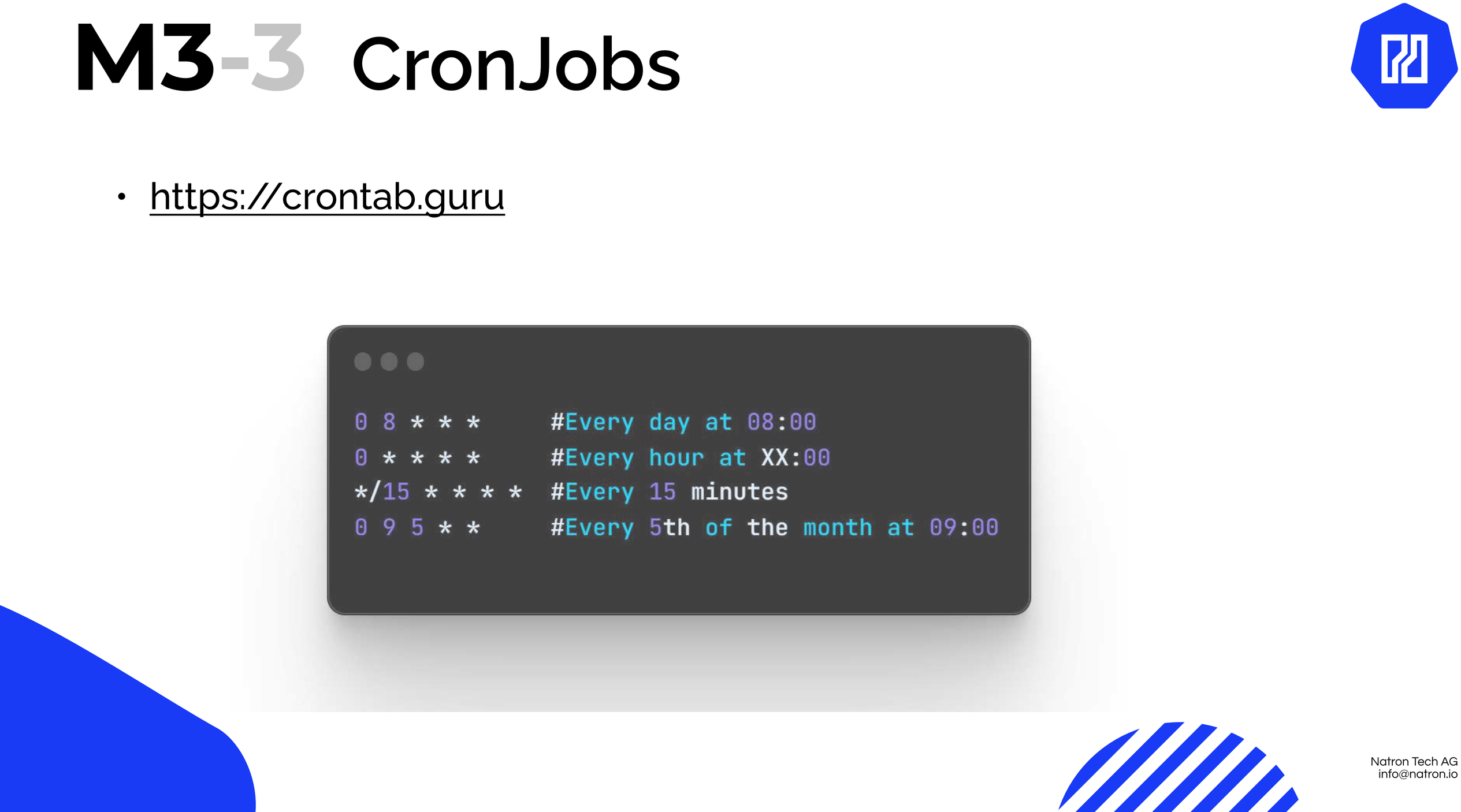
Natron Tech AG
info@natron.io

M3-2 CronJobs



Natron Tech AG
info@natron.io

M3-3 CronJobs



- https://crontab.guru

```
0 8 * * *     #Every day at 08:00
0 * * * *     #Every hour at XX:00
*/15 * * * *  #Every 15 minutes
0 9 5 * *     #Every 5th of the month at 09:00
```

Natron Tech AG
info@natron.io

LAB 8

Jobs & Cronjobs

M4 ConfigMaps

- Store **configuration** data
- **Key-Value** format
- Mount to **Pods**
  - Environment Variables
  - Files
- Updating ConfigMaps

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: appdata
data:
  log_level: "Debug"
  sla: "Gold"
  index.txt: |
    Welcome Message
    Hello there!
```

Natron Tech AG
info@natron.io

M4-1 ConfigMaps

- Mounting as env var

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: webserver
spec:
  containers:
    - name: demo
      image: nginx
      env:
        # Define the environment veriable
        - name: CUSTOMER_SLA
          valueFrom:
            configMapKeyRef:
              name: appdata
              key: sla
      envFrom:
        # Define the environment veriable
        - configMapRef:
            name: appdata
```

- Mounting as file

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: webserver
spec:
  containers:
    - name: demo
      image: nginx
      volumeMounts:
      - name: config
        mountPath: "/var/www/html"
        readOnly: true
  volumes:
  - name: config
    configMap:
      name: appdata # Name of the ConfigMap
      # Optional: Mapping
      items:
      - key: "index.txt"
        path: "index.html"
```

Natron Tech AG
info@natron.io

M4-2 Secrets

- Store **secret** data
- **Base64** format
  - echo “mypass” | base64
  - echo -n “XXX” | base64 -d
- Not encrypted
- Mount to **Pods**
  - Environment Variables
  - Files
- External secrets provider

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: db-access
type: Opaque
data:
  username: YWRtaW4=
  password: MWYyZDFlMmU2N2Rm
stringData:
  user_name: admin
  access_password: 1f2d1e2e67df
```

Natron Tech AG
info@natron.io

M4-3 Secrets

- Mounting as env var

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: webserver
spec:
  containers:
    - name: demo
      image: nginx
      env:
        # Define the environment veriable
        - name: DB_USERNAME
          valueFrom:
            secretKeyRef:
              name: db-access
              key: username
      envFrom:
        # Define the environment veriable
        - secretRef:
            name: db-access
```

- Mounting as file

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: webserver
spec:
  containers:
    - name: demo
      image: nginx
      volumeMounts:
      - name: secret
        mountPath: "/var/www/html"
        readOnly: true
  volumes:
  - name: secret
    secret:
      secretName: db-access # Name of the secret
      # Optional: Mapping
      items:
      - key: "user_name"
        path: "username.txt"
```

Natron Tech AG
info@natron.io

LAB 9

ConfigMaps & Secrets

M5 ServiceAccount

- Each pod is associated with a specific service account, which helps determine its **identity and access rights**.
- Kubernetes creates a default service account per namespace
- That service account is **mounted per default**

```
kubectl create sa my-account
kubectl create token my-account
```

```yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  name: my-custom-account
```

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
spec:
  serviceAccountName: my-custom-account
  automountServiceAccountToken: false
```

Natron Tech AG
info@natron.io

M5-1 Security Context

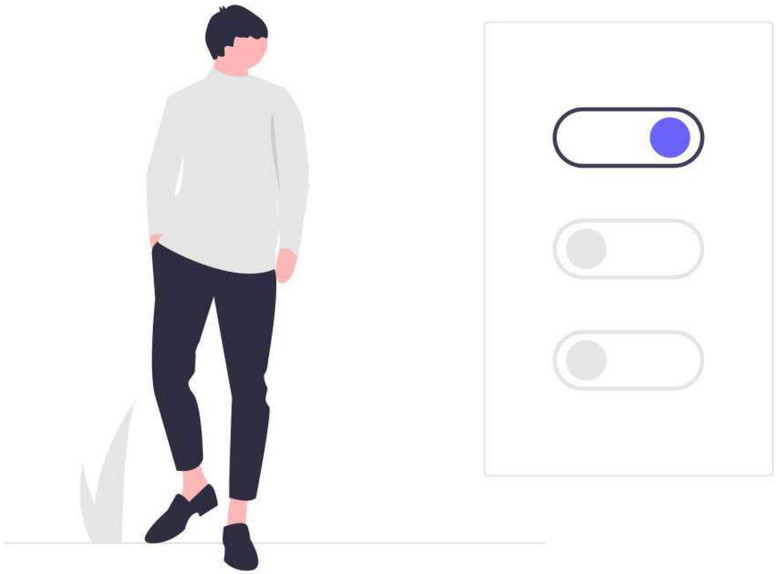
- The security context determines the Pods **privileges and capabilities**.
- securityContext can be set **per Pod or Container**
- Capabilites can be set **per container**

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: test-pod-1
spec:
  securityContext:
    runAsUser: 1000
  containers:
  - name: demo1
    image: busybox
    securityContext:
      runAsUser: 2000
      allowPrivilegeEscalation: false
      capabilities:
        add: ["NET_ADMIN", "SYS_TIME"]
        drop: ["ALL"]
```

Natron Tech AG
info@natron.io

M5-2 Network Policies

- Network policies are **firewall rules for Pods**
- They define how Pods can and can't communicate
- Filtering on ISO Layer 3 and 4



Natron Tech AG
info@natron.io

M5-3 Network Policies

- **podSelector**
  - To which Pods?
- **policyType**
  - Ingress (Incoming)
  - Egress (Outgoing)
- **Rule Array**
  - from/to
    - ipBlock
    - namespaceSelector
    - podSelector
  - ports
    - TCP/UDP

```yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: test
spec:
  podSelector:
    matchLabels:
      role: database
  policyTypes:
    - Ingress
    - Egress
  ingress:
    - from:
        - ipBlock:
            cidr: 192.168.0.0/16
            except:
              - 192.168.99.0/24
        - namespaceSelector:
            matchLabels:
              operation: backup
        - podSelector:
            matchLabels:
              role: frontend
      ports:
        - protocol: TCP
          port: 3306
```

Natron Tech AG
info@natron.io

M5-4 Network Policies

- Multiple Rules in Rule Array
- Tricky

```
...
  ingress:
    - from:
      - namespaceSelector:
          matchExpressions:
          - key: type
            operator: In
            values: ["backup", "system"]
        podSelector:
          matchLabels:
            role: critical
      - podSelector:
          matchLabels:
            role: frontend
      ports:
        - protocol: TCP
          port: 80
...
```

Natron Tech AG
info@natron.io

M5-5 Network Policies

Key takeaways

- CNI needs to support NetworkPolicies

- Default All allowed
- Default Deny once an Ingress / Egress Policy exists for a Pod
- Service port does not matter, **only Pod to Pod** is evaluated

- Multiple From / To Values are calculated with **OR**
- Multiple From / To Values **in the same block** are calculated with **AND**

https://github.com/ahmetb/kubernetes-network-policy-recipes

Natron Tech AG
info@natron.io

M5-5 Helm

- The **package manager** for Kubernetes

- Helm is a tool that **automates the creation, packaging, configuration, and deployment** of Kubernetes applications by combining your configuration files into a single **reusable package**

- Good Quickstart:
  - https://helm.sh/docs/intro/quickstart

LAB 10

Security and RBAC

LAB 11

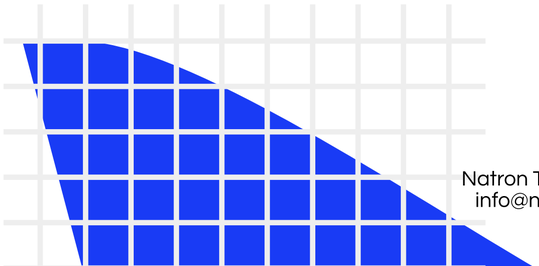
Helm

LAB 30

Optional: Role/Rolebinding, ...

M6 Best Practices

- Recommended labels:
  - https://kubernetes.io/docs/concepts/overview/working-with-objects/common-labels/
- Pod Security Standards:
  - https://kubernetes.io/docs/concepts/security/pod-security-standards/
- Liveness- & Readiness-Probes
- Define Resources (cpu & memory limits/requests)
- “the twelve-factor app”
  - https://12factor.net/
- Local Cluster
  - https://kind.sigs.k8s.io/
  - https://minikube.sigs.k8s.io/docs/



Natron Tech AG
info@natron.io

M6-1 About Us

NATRON

- We are on **Github** and **LinkedIn**
- **info@natron.io**
- Located at: **Sulgenrain 12, 3007 Bern**

Sven Gerber
@svengerber
Cloud Engineer / Consultant
sven.gerber@natron.io

Jan Fuhrer
@janfuhrer
Cloud Engineer / Consultant
jan.fuhrer@natron.io

Jan Lauber
@janlauber
Cloud Engineer / Consultant - Founder
jan.lauber@natron.io

swisscom

https://www.linkedin.com/company/natron-tech

M6-2 Image Material

NATRON

- https://www.weave.works/blog/kubernetes-faq-configure-storage-for-bare-metal-cluster
- https://stacksimplify.com/azure-aks/azure-kubernetes-service-namespaces-imperative/
- https://kubernetes.io/docs/home/
- https://ray.so
- https://undraw.co

Natron Tech AG
info@natron.io