General Overview over DevOps:

- https://roadmap.sh/devops
- https://github.com/donnemartin/system-design-primer
- https://www.reddit.com/r/devops/comments/kjpy58/aws_services_to_learn_whicbh_ones/
- https://www.youtube.com/watch?v=NjYsXuSBZ5U&ab_channel=SanjeevThiyagarajan
-

Basic aws services

- Ec2
  - https://github.com/wrble/public/blob/main/aws-instance-types.md
- S3
- EBS
- Vpc
  - Why we must create one: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-classic-platform.html
- cloudwatch
- Cert-manager
- vpn
- Elastic Load Balancer
- security groups

Everything in AWS is an API

AWS routing/vpc

- https://www.reddit.com/r/devops/comments/kt3g88/aws_eks_architecture_discussion/

Allgemeine Services needed

- Helm (/ Kustomize) (Helm's ultimate goal is to treat k8s config as a package/artifact where Kustomize is a templating tool. https://www.reddit.com/r/devops/comments/l1z39q/helm_vs_kustomize/)
- Datadog / new relic
- Kubernetes dashboard
- ElasticSearch/Kibana -> EFK-Stack (logs)
- Application Performance Management (APM): prometheus/grafana (cluster-metrics)

Bootup managed kubernetes

- Kubespray
- Kops

Bootup managed kubernetes (https://www.reddit.com/r/devops/comments/l0mvwl/what_do_you_prefer_for_managing_aws_eks_cluster/)

- Terraform aws provider / module
- Eksctl

HTTPS / SSL

- https://www.reddit.com/r/devops/comments/kqjcxx/do_i_need_to_configure_ssl_certs_on_nginx_itself/

- Using https simply means someone has certificates. That "someone" can be:
  -OpenShift
  -Load balancers in front of OpenShift
  -Your container resp. service

  All are possible solutions. There's no right or wrong. There's only a "more complex than it needed to be". I'd recommend the load balancer or if you lack those, then let OpenShift handle the ingress TLS termination. Simple, easy to configure and a central place to keep certificates.

  At work we have containers terminate. Everything else just passes through everything. There's only non-technical reasons why we do it this way. Not recommended, but it works too.
- Have frontend for 0.0.0.0:80 that forwards requests (based on URI) to a certbot/letsencrypt service running on 127.0.0.1

1. Configures haproxy to
   a. Have frontend for 0.0.0.0:443 automatically responding with a cert pulled from a directory via SNI
   b. Have frontend for 0.0.0.0:443 automatically route requests to appropriate pool via SNI
   c. Have frontend for 0.0.0.0:80 that forwards requests (based on URI) to a certbot/letsencrypt service running on 127.0.0.1
   d. Have backend/pool mappings as described in the YAML
      i. Append x-forwarded-for, and forward the SNI header to the worker pool
2. Run a simple shell script that uses certbot to request a certificate for each of the certificate declarations in the YAML, with some additional bash/string parsing for:
   a. Idempotency (e.g., check if cert is expired)
   b. Copy letsencrypt live certs into the directory where haproxy checks for certificates
   c. Reload haproxy if a cert was added or fqdn->pool mapping changed or anything like that

SSO

- https://www.reddit.com/r/devops/comments/l2ohox/best_sso_solution_for_a_50_company/
    - Okta (eventuell federated with aws sso for security reasons, but you have to pay for both)
        - 2€ per user, mind. 1500€ per year

Terraform guides

- https://github.com/futurice/terraform-examples
- https://github.com/hashicorp/terraform-guides
- https://www.youtube.com/watch?v=SLB_c_ayRMo&t=7809s&ab_channel=freeCodeCamp.org

Deployment

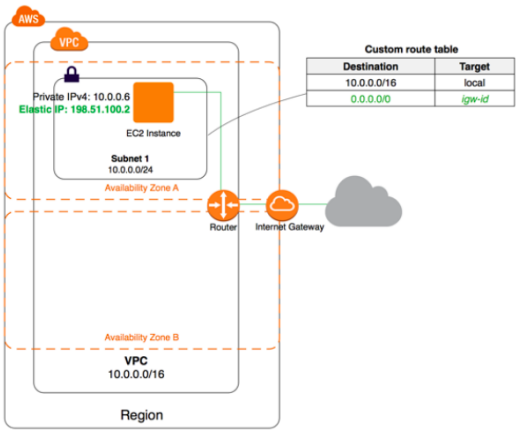
- Asp.net core 3.1 example docker file
    - https://gist.github.com/adrianord/5c51bd0f087cca2818ac15b076a72727

Nginx

- https://www.linode.com/docs/guides/how-to-configure-nginx/
- https://serverfault.com/questions/787919/optimal-value-for-nginx-worker-connections

Vpc (includes vpc, eip, gw, ...):

- https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-classic-platform.html
- https://docs.aws.amazon.com/de_de/vpc/latest/userguide/VPC_Internet_Gateway.html
- https://docs.aws.amazon.com/de_de/vpc/latest/userguide/vpc-subnets-commands-example.html
- Routing tables in ubuntu:
    - https://unix.stackexchange.com/questions/345862/is-it-possible-to-have-multiple-default-gateways-for-outbound-connections
    - https://serverfault.com/questions/618857/list-all-route-tables


AWS
VPC
Private IPv4: 10.0.0.6
Elastic IP: 198.51.100.2
EC2 Instance
Subnet 1
10.0.0.0/24
Availability Zone A
Router
Internet Gateway
Availability Zone B
VPC
10.0.0.0/16
Region
Custom route table
Destination
Target
10.0.0.0/16
local
0.0.0.0/0
igw-id

AWS Idendity Management: IAM

Iam entities

- User, Groups, roles
- 0 permissions at start (default) (you have to grant permission to them)

Iam user

- Iam user:
    - End user think about people
- Fedderated user:
    - See below

Role

- Im Kern:
    - Role = “user” wenn man external identity provider hat oder “user” für eine aws ressource
        - Iam user, wenn man keinen external identity provider hat und man user managen möchte (und eben keine anderen ressourcen)
        - Iam user hat pw, role hat kein pw -> daran kann man eigentlich gut das ziel der role erkennen -> role benutzt man für external identity provider -> naja dann brauch man kein extra pw für die role -> die role hat somit kein pw aber zur authentifizierung bekommt sie einen temporären access token; role kann man keinen groups zuordnen -> naja wenn man external identity provider hat dann macht man das ja auch darüber und dann eben nicht intern in aws
        - Role is a way to provide permissions to someone (a customer, supplier, contractor, employee, an EC2 instance, some external application outside AWS trying to consume your services) without creating a user for it.
            - https://stackoverflow.com/questions/36991831/aws-iam-role-vs-group
        - “Remember: Groups are for living, roles are for non-living”
            - https://stackoverflow.com/questions/36991831/aws-iam-role-vs-group

14

Users: End User (*Think People*).

Groups: A ***collection of users*** under one set of permissions (*permission as policy*). As per IAM standards we create groups with permissions and then assign user to that group.

Role: you create roles and ***assign them to AWS resource*** (*AWS resource example can be a customer, supplier, contractor, employee, an EC2 instance, some external application outside AWS*) but remember you can't assign role to user.

It's not only users who will login, sometimes applications need access to AWS resources. For example, an EC2 instance might need to access one or more S3 buckets. Then, an IAM role needs to be created and attached to the EC2 instance. That role can be re-used by different EC2 instances.

Remember : Groups are for living. Roles are for non-living.

Share Improve this answer Follow

edited Jul 16 '20 at 13:01
David Medinets
3,940 3 24 39

answered Oct 12 '18 at 11:29
RishiKesh Pathak
1,502 14 22

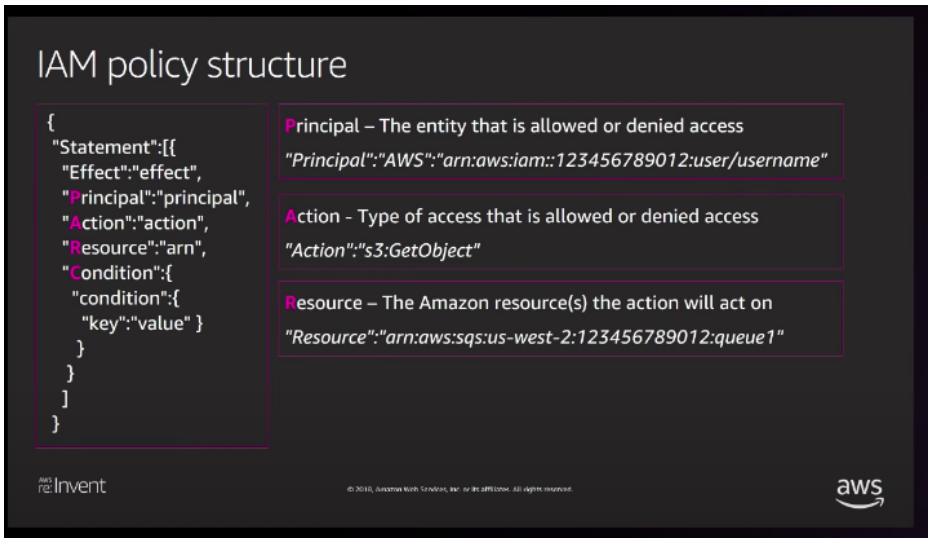
- 3 role types
    - Aws service roles (e.g. ec2, lambda, … eben für services von aws)
    - Cross-account access (to grant permission to users from other aws account)
    - Identity provider access (to grant permission to users of external identity provider)
- Gute Quelle:
    - https://stackoverflow.com/questions/46199680/difference-between-iam-role-and-iam-user-in-aws
    - https://stackoverflow.com/questions/36991831/aws-iam-role-vs-group
- Specifies a set of permissions that you can use to access aws resources
- = temporary access to carry out required tasks and interact with aws resources
- principal =person or application
- Per subject only 1 can be activ: 1 role OR 1 iam user not both, if you define role other iam user, only the role applies

Group
- a set of users under one set of permission(policies)

Permissions
- let you specify access to AWS resources
- General term
    - You cant just add permission
    - Permissions granted by policies, roles, groups
- Permissions are granted to IAM entities and/or aws ressources
- entitites direct after creation: 0 permissions
- To give permissions: attach a policy that
    - specifies the type of access
    - the actions that can be performed
    - the resources on which the actions can be performed
- In addition, you can specify any conditions that must be set for access to be allowed or denied

Policy (actually subtype of permissions)
- Policy
    - Policies are the only way to give an entity or ressource permission (by assigning the policy to it)
    - Types:
        - Service control policies (scps) (aws organizations)
        - Iam (identity and access management)
            - Following informations are based on that type only
        - Scoped-down policies (aws secruity token service (aws sts))

        - Resource-based policies (specific aws services)
        - Endpoint policies (vpc endpoints)
    - Specifies (more in general)
        - the type of access
        - the actions that can be performed
        - the resources on which the actions can be performed
    - You can permit (more in detail):
        - Actions: Which aws service action you allow (e.g. Call s3 listbucket action?) (grouped by access level)
        - Resources: which aws resources you allow the action on (e.g. on which s3 is your s3 listbucket action allowed)
            - Specified by amazon resource name (arn)
        - Effect: you allow or deny access (since default is deny, most common is the allow effect)
        - Conditions: which conditions must be present for the policy to take effect (e.g. only if user is in a given ip-range)
    - Syntax:
        - Consist of X statement-blocks: 1 statement block = Y permissions



    - Example:
- policy actions
    - Type of access that is allowed or denied access e.g. “s3:GetObject”
    - Action level(5 -> sometimes tagging is not mentioned so 4)
        - List, Read, Write, Permissions management, Tagging
        - these are groups, so there are multiple different list-actions
            - E.g. ListAllMyBuckets, ListBucket or ListObjects
- Summary (technical term -> not a summary of the content):
    - policy summary

            - If you are using the IAM console and choose a policy, you will see a policy summary
            - Lists the access level, resources, conditions for each service defined in a policy
                - Access level will be displayed differently:
                    - If you only permitted some of the all possible actions in the e.g. LISTEN access level, there will be a label: LISTEN: limited. -> means you didnt permitted all actions, only some or one; if you permitted 0 actions in the LISTEN access level, there is no label at all (access full or access limited)
            - Intention: help to understand the permissions defined in a policy
        - Service summary
            - Includes list of actions + summaries of the permissions that are defined by the policy for the chosen device
        - Action summary
- AWS managed policies
- inline policies (discouraged)
- customer managed policies (encouraged)
- Delegation
    - Setup a trust between two accounts
        - 1. account=own the resource (trusting account)
        - 2. account=contains the users that need to access the resource (the trusted account)
    - Granting of permissions to someone to allow access to resources that you control
- Federation
    - Creation of a trust relationship between an external identity provider and aws
    - Idp has to be compatible with openid connect

Iam Best Practices

- Lock away your aws account root user access keys
- Create individual iam users
- Use groups to assign permissions to iam users
- Grant least privilege
- Get started using permissions with aws managed policies
- Use customer managed policies instead of inline policies
- Use access levels to review iam permissions

Quellen:

- https://aws.amazon.com/iam/features/manage-permissions/?audit=2019q1

- https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_understand-service-summary.html

Cloudtrail:

- Central aggregation-point for logs
- event=Single log-files
- Separates 3 groups of logs/events
    - Management events
        - Logs regarding the management operations ("control plane") performed on resources on aws account
        - E.g. launching ec2, creating s3, …
        - Pricing: first copy free, additional copies: $2.00 per 100.000 events
    - Data events
        - Logs regarding resource operations (data plane) on or within the resource itself
        - E.g. iam, s3, …
        - Pricing: $0.10 per 100.000 events
    - Cloudtrail insights
        - Logs regarding unusual operational activity in your aws account
            - Analyses the api call

Cloudwatch

- Watch metrics for each service
- In general: starts working with the creation of the aws-account or first service idk
- Pricing:
    - General Service dashboards / metrics are free
    - Custom dashboards / metrics e.g. for applications inside the service are in charge

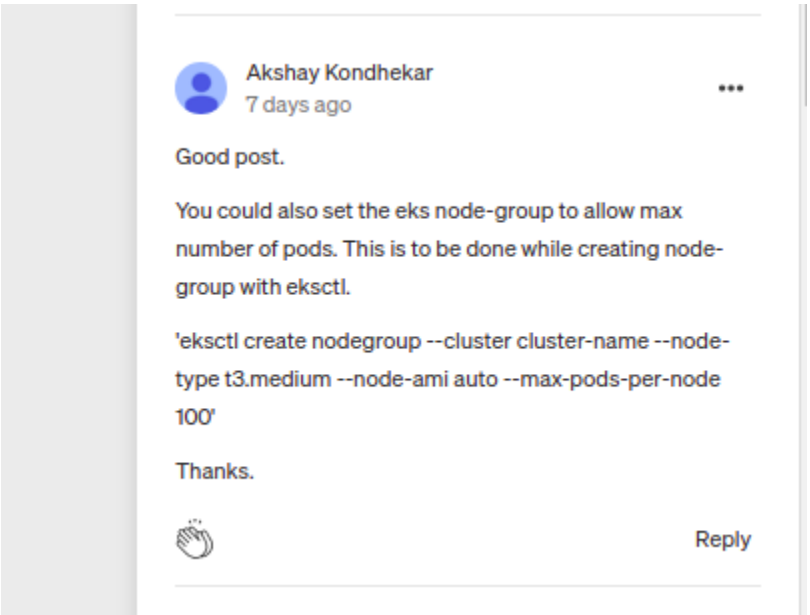
Terraform: Quellen für Skripte

- Auto scaling group: https://blog.gruntwork.io/an-introduction-to-terraform-f17df9c6d180#8606
- Iam: https://medium.com/faun/aws-iam-user-and-policy-creation-using-terraform-7cd781e06c97
- https://www.airpair.com/aws/posts/building-a-scalable-web-app-on-amazon-web-services-p1
- https://github.com/gruntwork-io/intro-to-terraform
- https://medium.com/tensult/creating-vpc-endpoint-for-amazon-s3-using-terraform-7a15c840d36f
- https://dev.to/ari_hacks/5-things-terraform-can-automate-in-aws-4k5j
- https://www.reddit.com/r/Terraform/comments/bax90n/aws_s3_bucket_policy_attachment/

- https://github.com/tmknom/terraform-aws-s3-cloudtrail/blob/master/main.tf

Eks

- https://docs.aws.amazon.com/eks/latest/userguide/getting-started-console.html
- https://docs.aws.amazon.com/de_de/eks/latest/userguide/create-cluster.html
- EKS module + Calico (instead of AWS CNI)
    - Main reason: max pods=(the number of ENIs for the instance type × (the number of IPs per ENI - 1)) + 2
        - T3.micro -> 4 pods wobei coredns auch noch welche zieht
    - Ressources:
        - https://luktom.net/en/e1715-how-to-and-why-replace-aws-cni-with-calico-on-aws-eks-cluster
        - https://medium.com/@surajrajanathrapully/automating-aws-vpc-cni-replacement-with-calico-on-eks-df4da851a400
        - https://github.com/howdio/terraform-aws-eks/blob/master/modules/cluster/addons.tf
        - https://itnext.io/bootstrapping-kubernetes-clusters-on-aws-with-terraform-b7c0371aaea0#9403
        - https://medium.com/@jeremy.i.cowan/running-calico-on-eks-f3e52ea41271
        - Table: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-eni.html#AvailableIpPerENI
        - https://github.com/Shogan/terraform-eks-with-weave/blob/master/src/eks.tf
        - https://www.nickaws.net/eks/2020/05/18/Tweaking-the-EKS-CNI.html



        - https://github.com/hashicorp/learn-terraform-provision-eks-cluster/blob/master/eks-cluster.tf
        - https://itnext.io/build-an-eks-cluster-with-terraform-d35db8005963
        - https://www.esentri.com/building-a-kubernetes-cluster-on-aws-eks-using-terraform-part-iv/
        - https://github.com/cloudposse/terraform-aws-eks-workers/blob/master/main.tf
        - https://www.bluematador.com/blog/moving-to-eks-in-production
        -
- Eks from scratch (without eks module)
    - https://www.padok.fr/en/blog/aws-eks-cluster-terraform
    - https://medium.com/risertech/production-eks-with-terraform-5ad9e76db425
    - https://lkravi.medium.com/aws-eks-with-terraform-gitops-7c5f3d60525d
    - https://wangpp.medium.com/terraform-eks-nodegroups-with-custom-launch-templates-5b6a199947f
    - https://docs.aws.amazon.com/eks/latest/userguide/launch-templates.html#launch-template-user-data
    -

Nat

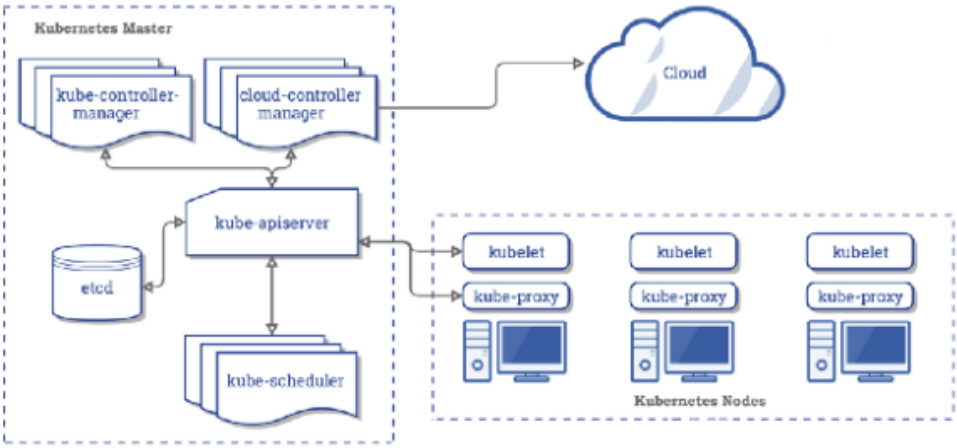
- https://dev.betterdoc.org/infrastructure/2020/02/04/setting-up-a-nat-gateway-on-aws-using-terraform.html

Nodes

Response:

```
m4.xlarge
m5.xlarge
m5a.xlarge
m5ad.xlarge
m5d.xlarge
m5dn.xlarge
m5n.xlarge
```

Kubernetes



Kubernetes component architecture diagram from the official documentation.

Calico

- https://docs.projectcalico.org/reference/public-cloud/aws

Metrics Server:

- https://stackoverflow.com/questions/60531350/kubernetes-metrics-server-faileddiscoverycheck

Problem with metrics server (and all other possible pods on the eks)

- https://medium.com/faun/choosing-your-cni-with-aws-eks-vpc-cni-or-calico-1ee6229297c5
- Tried to compensate described by:
    - https://luktom.net/en/e1715-how-to-and-why-replace-aws-cni-with-calico-on-aws-eks-cluster
    - https://medium.com/@surajrajanathrapully/automating-aws-vpc-cni-replacement-with-calico-on-eks-df4da851a400
- There are github-issue regarding that topic
    - This issues are open since around late 2019
- This was the main-reason to switch to azure