Secure Continuous Deployment & DAST

Key Takeaways

TECHWORLD WITH NANA

All rights reserved to nnSoftware GmbH

TECHWORLD WITH NANA

About TechWorld with Nana

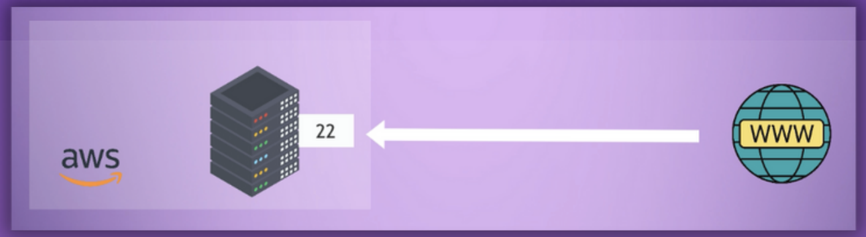
TechWorld with Nana is an established name in the DevOps and Cloud industry, and it stands for the quality trainings helping 1,000s of engineers acquire the most in-demand skills in this field.

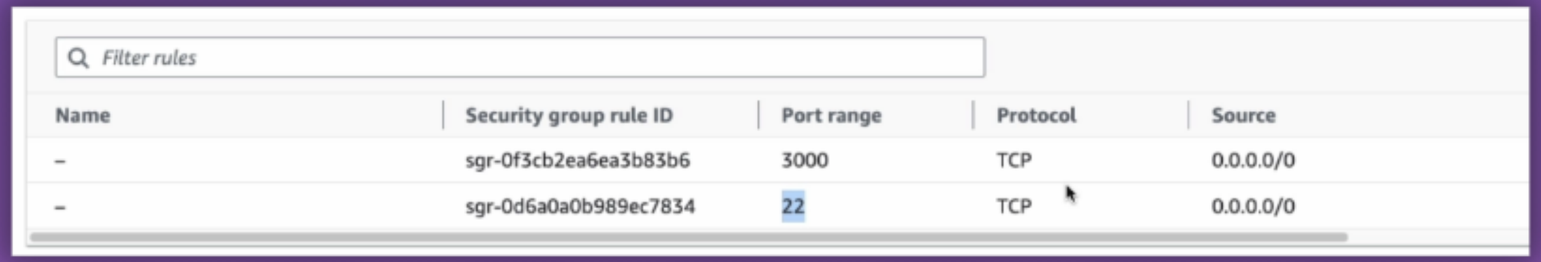
Our mission is enable individual engineers as well as companies to take advantage of the recent developments in Cloud and DevOps fields, to use technologies and concepts in order to create efficient, automated, streamlined DevSecOps processes in organisations.

AWS Systems Manager

Secure server access

SSH port is **open to anyone**





Security Improvement

SSH with **whitelisted IP addresses** to access SSH port

```
49  deploy_image:
50    stage: deploy
51    image: debian:bullseye-slim
52    before_script:
53      - apt update -y && apt install openssh-client -y
54      - eval $(ssh-agent -s)
55      - chmod 400 "$SSH_PRIVATE_KEY"
56      - ssh-add "$SSH_PRIVATE_KEY"
57      - mkdir -p ~/.ssh
58      - chmod 700 ~/.ssh
59    script:
60      - ssh -o StrictHostKeyChecking=no $SERVER_USER@$SERVER_IP "docker pull $IMAGE_NAME:latest"
61      - ssh -o StrictHostKeyChecking=no $SERVER_USER@$SERVER_IP "docker stop juice-shop || true && docke
62      - ssh -o StrictHostKeyChecking=no $SERVER_USER@$SERVER_IP "docker run -d --name juice-shop -p 3000
63
```

Secure server access

BETTER and Best Practice for Security

- **Close SSH port** completely, and have **no static SSH keys**

- Use AWS authentication and authorization mechanism for server access

When SSHing into EC2 for deployment, we are working around AWS authentication. We are **authenticating with the server directly**, not using AWS CLI and its authentication system

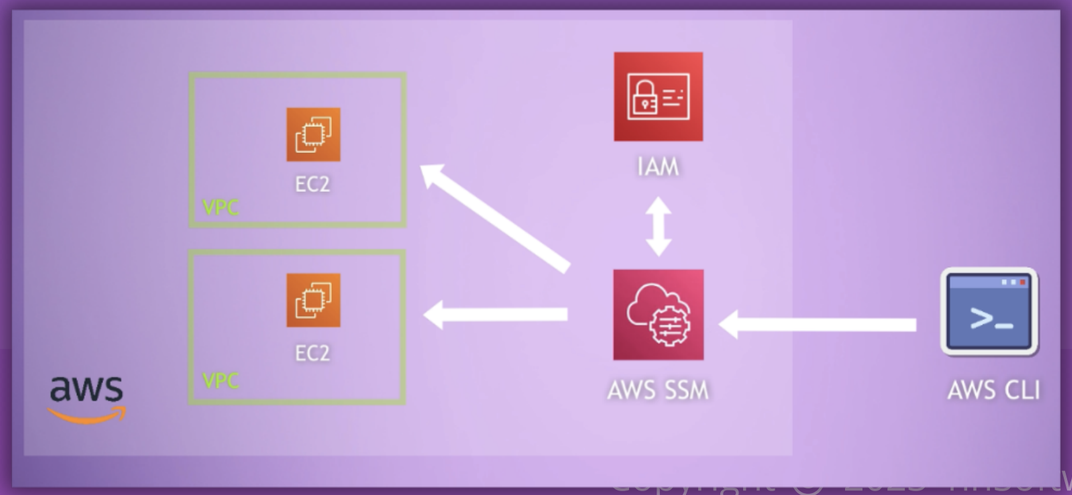
TECHWORLD WITH NANA

SSM - AWS Systems Manager

What is AWS Systems Manager?

- It's a management service that helps you manage and automate operational tasks across your AWS resources
- "**aws ssm**" sub command is used to interact with AWS SSM
- Secure end-to-end solution that enables secure operations at scale
- SSM includes several key features/tools

How it works



- **SSM Agent** runs on EC2 instances
- Agent processes requests from SSM and runs them
- Then sends status and execution information back

TECHWORLD WITH NANA

SSM - AWS Systems Manager

Enable SSM with our EC2 Instance

2 Requirements we need to fulfill to connect with SSM:

1.Have SSM Agent installed on EC2 instance (mostly available by default)

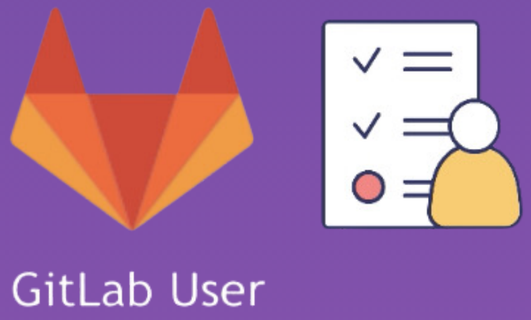
2. **Attach SSM Role** to EC2 Instance. So EC2 instance is allowed to be managed by SSM
    a. Machines managed by SSM are called “Nodes”
    b. A “managed node” is any machine configured for use with SSM

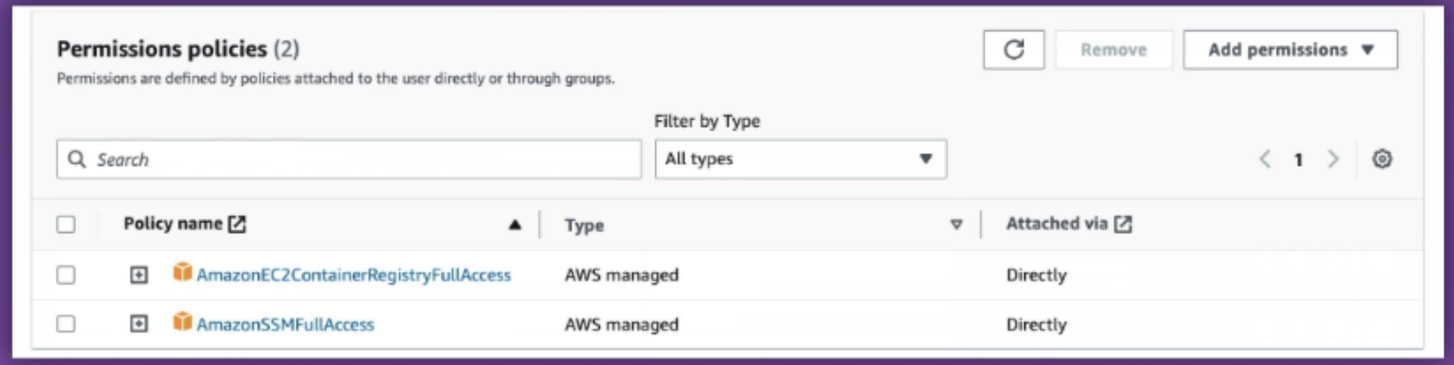
TECHWORLD WITH NANA

Secure Deployment

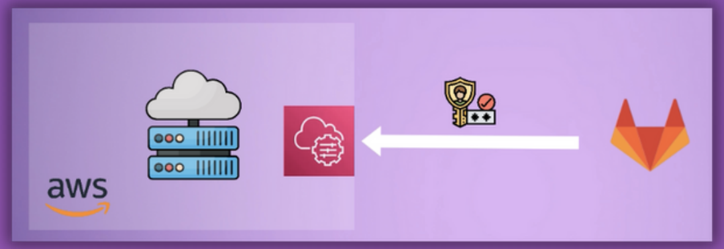
Secure Deployment - Improvement

- We have a dedicated CI/CD user with **limited permissions**





- Use SSM instead of SSH




```
26  deploy_test:
27    stage: deploy-test
28    image:
29      name: amazon/aws-cli
30      entrypoint: [""]
31    script:
32      - LOG_IN_CMD="export AWS_DEFAULT_REGION=$AWS_DEFAULT_REGION; export AWS_ACCESS_KEY_ID=$AWS_ACCESS_KEY_ID; export
    AWS_SECRET_ACCESS_KEY=$AWS_SECRET_ACCESS_KEY; aws ecr get-login-password | docker login --username AWS --password-stdin $AWS_ACCOUNT_ID.dkr.ecr.
    $AWS_DEFAULT_REGION.amazonaws.com"
33      - COMMANDS_TO_EXECUTE="docker pull $IMAGE_NAME:latest && (docker stop juice-shop || true) && (docker rm juice-shop || true) && docker run -d
    juice-shop -p 3000:3000 $IMAGE_NAME:latest"
34      - COMMAND_ID=$(aws ssm send-command --instance-ids "i-01bd29a9543114b73" --document-name "AWS-RunShellScript" --parameters "commands=[$LOG_IN
    $COMMANDS_TO_EXECUTE]" --query "Command.CommandId" --output text)
35      - aws ssm wait command-executed
36      - aws ssm get-command-invocation --command-id "$COMMAND_ID" --instance-id "i-01bd29a9543114b73"
37
```

Secure Deployment - Better

Eliminate the need for credentials

- Remove need for SSH keys
- Remove need for GitLab user with access credentials

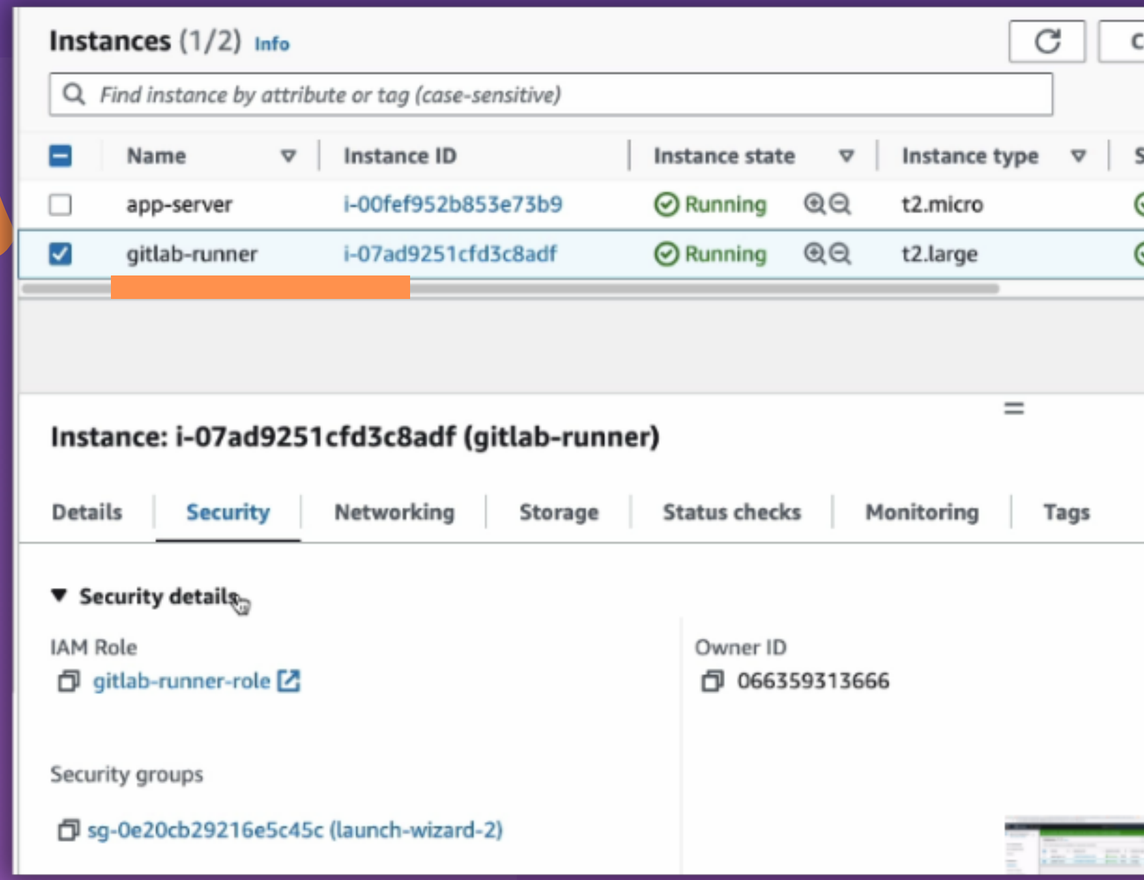
By using IAM Role for better security

- Role for EC2 Resource (GitLab runner)

TECHWORLD WITH NANA

Secure Deployment - Better

How it works



- Our self-hosted GitLab runner on **EC2 instance is an AWS resource**

Role for GitLab Runner

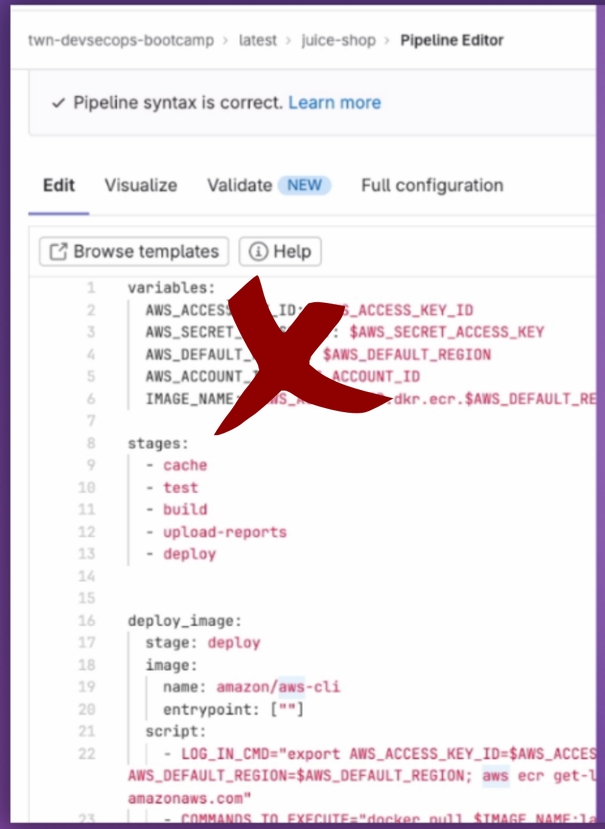
- Role credentials are stored on instance metadata
- EC2 instance **gets permissions through role**
- GitLab runner inherits the permissions from IAM role

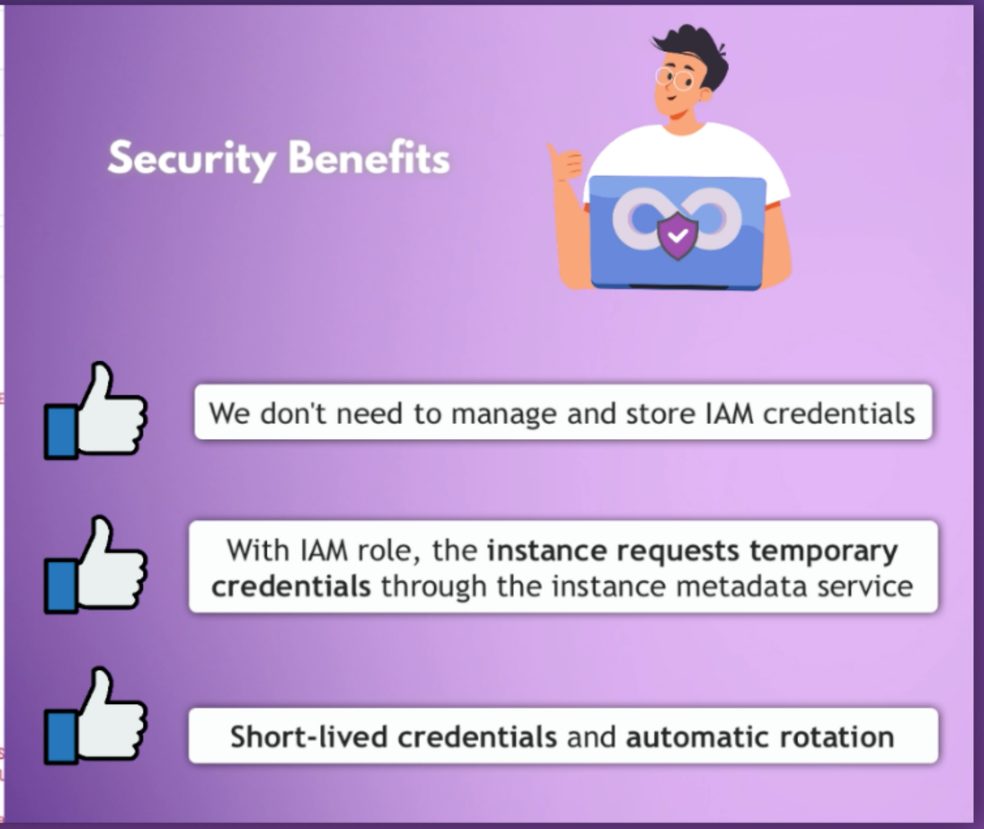
TECHWORLD WITH NANA

Secure Deployment - Better

Adjustments we can do

- Delete GitLab user on AWS
- Delete credentials stored in GitLab
- Adjust pipeline code: Delete access keys

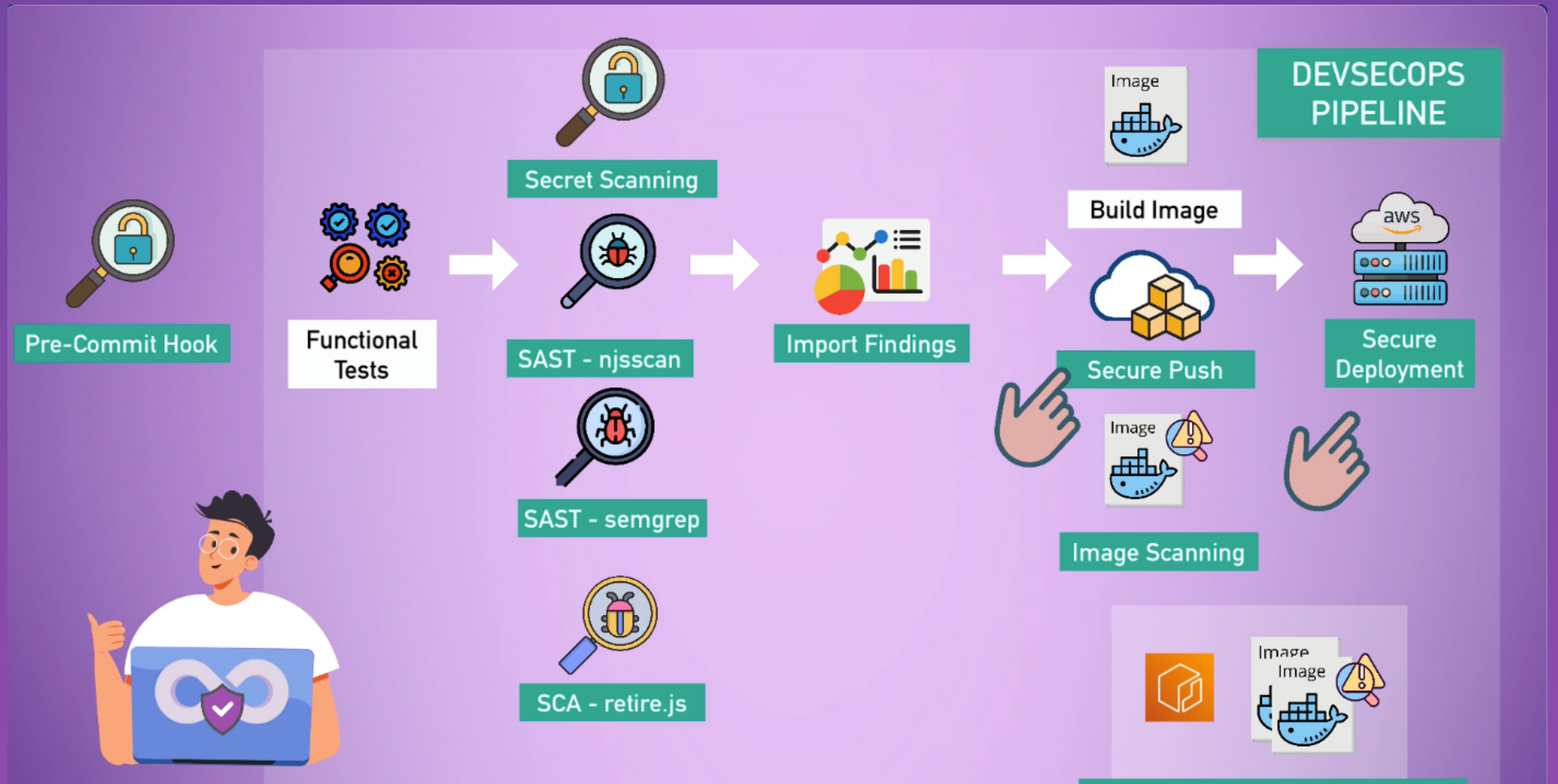




TECHWORLD WITH NANA

Wrap Up - Our DevSecOps Pipeline

- Deployment with security best practices
- Access based on IAM roles
- No AWS account credentials stored on GitLab



TECHWORLD WITH NANA

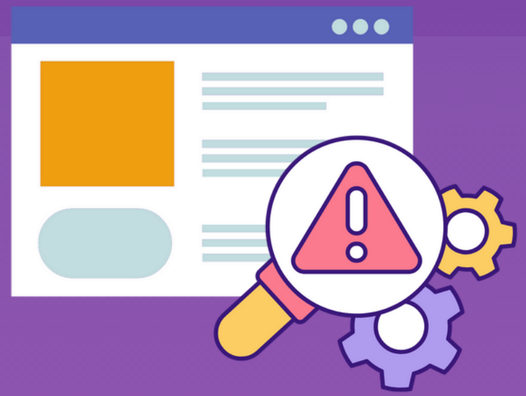
Dynamic Application Security Testing

DAST

Stands for **Dynamic Application Security Testing**

- Focus on identifying vulnerabilities in a **running application**
- Simulating an attacker
- Probe application from outside, like potential attacker
- It's **black box** testing, no insights of code

DAST Tools



- Tools that perform these attacks automatically
- Once scanning is complete, they generate detailed reports that highlight the vulnerabilities discovered

- **ZAP is an open source and widely used** web application security scanner
- ZAP is what is known as a "man-in-the-middle proxy"

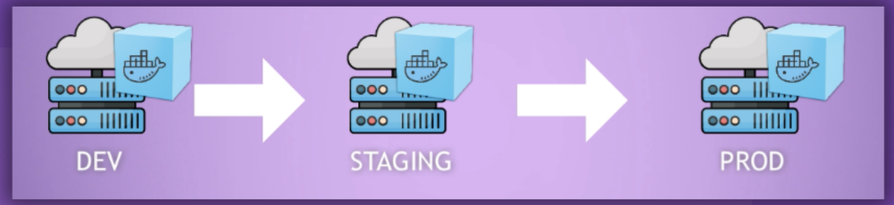
ZAP

Zed Attack Proxy

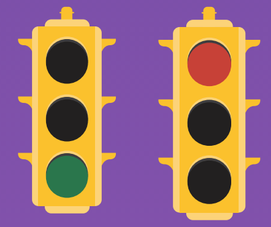
TECHWORLD WITH NANA

Release in Stages

Multiple Deployment Environments



- App is not deployed to the production environment in one step
- Usually app is deployed step by step
- Issues need to be fixed before pipeline continues to deploy to other stages and eventually to production



- DAST tests are executed in **one of the pre-production environments**
- If cricital security issues are found, CI/CD pipeline is aborted

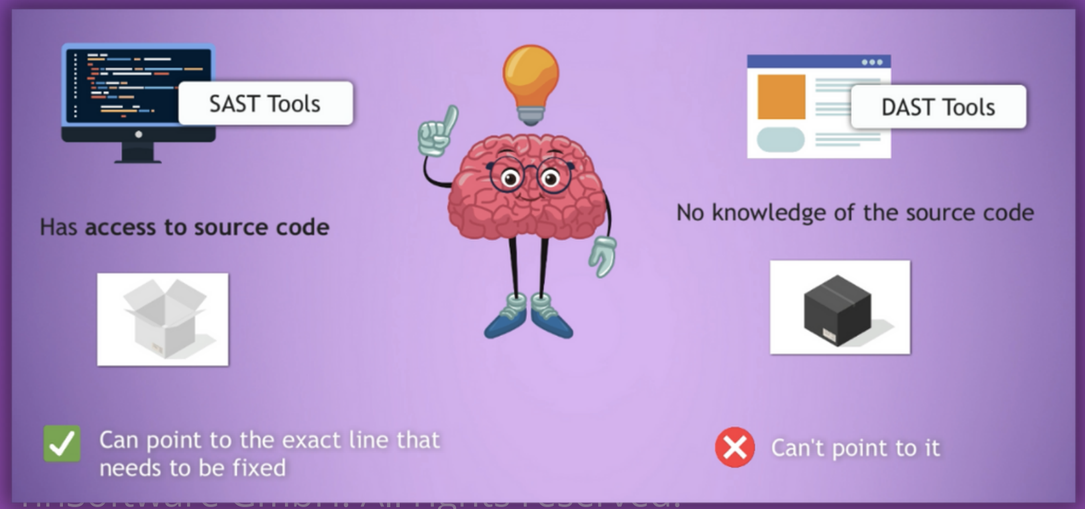
TECHWORLD WITH NANA

Configure DAST job in Pipeline

Pipeline Configuration

```yaml
zap:
  stage: deploy-test
  needs: ["deploy_test"]
  image: owasp/zap2docker-stable
  variables:
    ZAP_TARGET: "http://35.180.67.168:3000"
  before_script:
    - mkdir -p /zap/wrk
  script:
    - zap-baseline.py -t $ZAP_TARGET -g gen.conf -I -x baseline.xml
    - cp /zap/wrk/baseline.xml baseline.xml
  artifacts:
    when: always
    paths:
      - baseline.xml
```

Note on **Remediation**



TECHWORLD WITH NANA

Baseline vs Full Scan

BASELINE Scan

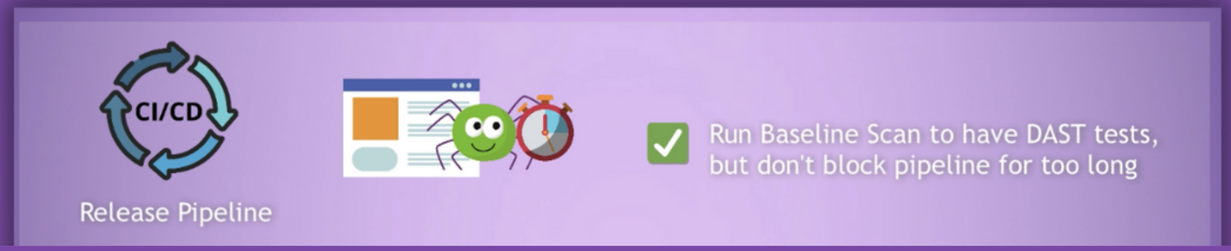
- **Quick and lightweight -** aiming to provide a rapid overview of vulnerabilities without conducting an exhaustive analysis
- Focus on well-known and frequently exploited vulnerabilities
- ZAP Baseline configuration:
  - Usage of **time limited** Spider
  - Then waits for the passive scanning to complete

FULL Scan

- Script performs **actual attacks**
- **In-depth analysis**, including less common or unique attack scenarios
- No time limit, takes much longer
- **Risk of affecting the app's behavior** or stability => Employ on own dedicated environment

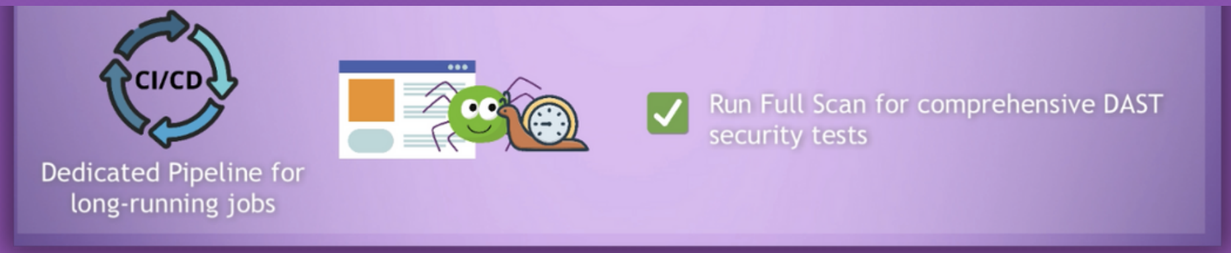
Baseline vs Full Scan in Practice

BASELINE Scan



Runs on **every single commit**

FULL Scan



Run on a **specified time** for example once a day

TECHWORLD WITH NANA

Full DevSecOps Pipeline