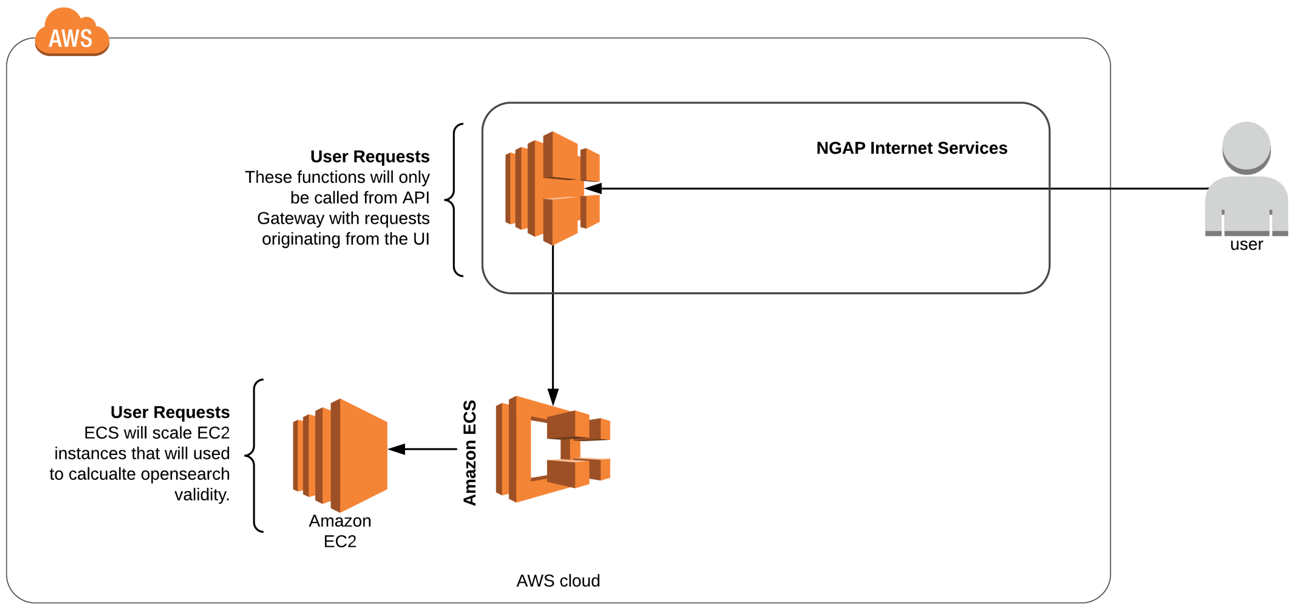
[SOP] OpenSearch UI Standard Operating Procedures

OpenSearch UI (https://OpenSearch UI.earthdata.nasa.gov/api) is an application built to validate OpenSearch API endpoints against the OpenSearch specification. It's goal is to allow clients looking to offer an OpenSearch endpoint the ability to validate their endpoints autonomously.

- Production
  - Public Facing
  - https://opensearch-ui.earthdata.nasa.gov/
  - esdis-app-edsc-prod-7557 [NGAP 2.0]
- UAT
  - Public Facing
  - https://opensearch-ui.uat.earthdata.nasa.gov/
  - esdis-application-edsc-uat-8982 [NGAP 2.0]
- SIT
  - *VPN only*
  - https://opensearch-ui.sit.earthdata.nasa.gov/
  - esdis-application-edsc-sit-8022 [NGAP 2.0]

Architecture


AWS
NGAP Internet Services
User Requests
These functions will only be called from API Gateway with requests originating from the UI
user
User Requests
ECS will scale EC2 instances that will used to calcualte opensearch validity.
Amazon EC2
Amazon ECS
AWS cloud

Components

- OpenSearch UI is built using JRuby, and Ruby on Rails

Deploying

Generally, building and deploying will be done from Bamboo. Bamboo's deployment mechanisms give the option of a script defined within the UI or a script within the repository alongside the code. OpenSearch UI uses a script that is defined within the repository located at `scripts/deploy_bamboo.sh`. The script is simply a list of commands so if Bamboo isn't an option, the commands can be ran manually via the command line. The overall process looks like this:

1. Build a Docker image from the Dockerfile
   a. Installs JRuby
   b. Installs PhantomJS
   c. Installs Ruby Gems
2. Push the Docker image to AWS ECR
3. Execute Terraform deployment to AWS

In order to run the deployment script locally, the following prerequisites must be met:

- Install AWS CLI
- Install dos2unix

Required Environment Variables For Deployments

- AWS_ACCESS_KEY_ID AWS Access Key created in CloudTamer.
- AWS_SECRET_ACCESS_KEY Access Token created in CloudTamer.
- AWS_REGION AWS Region to deploy the application to
- AWS_ACCOUNT_ID AWS Account Id
- cpu Task level CPU allocation.
- desired_task_count How many instantiations of the task to replicate.
- environment_name What label to apply to the deployment, helps separate multiple deployments within a single account.
- image Name of the image to pull from ECR.
- memory Container level memory configuration.
- permissions_boundary_arn AWS Permissions Boundary provided by NGAP.
- rack_env Environment to provide to Rack (["development", "production"]).
- remote_state_bucket AWS S3 bucket to store Terraform remote state in.
- subnet_ids AWS Subnet IDs provided by NGAP in array form (["subnet-a", "subnet-b"]).
- vpc_id AWS VPC ID provided by NGAP.

Destroying

This can be done from within the AWS console by deleting the stack from CloudFormation. Running the commands require valid access tokens, see the Terraform documentation for methods of providing these values.

Using AWS Access Tokens (Terraform): https://registry.terraform.io/providers/hashicorp/aws/latest/docs#authentication

```
// Destroy Application
terraform destroy
```

Configuration

OpenSearch UI is deployed to NGAP 2.0, aka AWS using Terraform. The application utilizes ECS behind an Elastic Load Balancer.

Backup

Code Backup

OpenSearch UI's code base and static assets are backed up in BitBucket. BitBucket is backed up on-premise as part of EED-2 infrastructure management.

Bitbucket: https://git.earthdata.nasa.gov/projects/OPENSEARCH/repos/opensearch-ui/browse

Restoring (Deployments from Scratch)

Should OpenSearch UI need to be completely recovered, that process takes around 30 minutes and requires a ticket that requires additional resources (NGAP).

Steps:

1. Deploy full application

Ensure that all environment values are provided, when deploying from Bamboo those values are set in the UI, but when deploying from command line they need to be provided or set.

An example deployment using the command line to set the necessary ENVs would look like this:

```
bamboo_remote_state_bucket="opensearch-ui-sit-terraform-state" bamboo_environment_name="restore"
bamboo_vpc_id="vpc-id" bamboo_subet="[\"subnet-id-1\", \"subnet-2\"]" bamboo_image="REDACTED.dkr.ecr.us-east-1.
amazonaws.com/opensearch-ui" bamboo_permissions_boundary_arn="arn:aws:iam::REDACTED:policy/NGAPShRoleBoundary"
bamboo_desired_task_count=1 memory=512 cpu=512 rack_env="production" bamboo_AWS_ACCESS_KEY_ID="[REDACTED]"
bamboo_AWS_SECRET_ACCESS_KEY="[REDACTED]" bamboo_AWS_ACCOUNT_ID="[REDACTED]" bamboo_AWS_REGION=us-east-1 script
/deploy_bamboo.sh
```

Release Cycle

OpenSearch UI is not under 'active' development, Earthdata Search is responsible for vulnerability updates and issues. OpenSearch UI can release on-demand with appropriate notice to stakeholders.

Patching and Remediation

OpenSearch UI has a blocking step in our deployment process that audits our libraries and dependencies. Once a vulnerability is found, steps are taken to patch and update and resolve the vulnerability immediately.

In the case of a vulnerability discovered for a resource currently deployed, a ticket is filed once the issue is identified. OpenSearch UI devops consults with the security team to prioritize the remediation of the vulnerability found. Once a ticket is created, approved, and prioritized, EDSC dev team works the issue until all vulnerabilities are resolved and deploy the updated app to all operational environments.

Diagnostics

OpenSearch UI logs to AWS CloudWatch on a per Lambda basis; this allows you for easy access to specific Lambda logging in the event that you know which Lambda is responsible for the logs you're looking for. If a wider search needs to occur, OpenSearch UI forwards logs to Splunk which accommodates a wider array of search abilities.

Splunk: https://logs.earthdata.nasa.gov/

Planned Maintenance

Deployments are handled via Bamboo.

Bamboo: https://ci.earthdata.nasa.gov/deploy/viewDeploymentProjectEnvironments.action?id=287375363