Running Sikraken on EC2

Research Document

Isaiah Andres

December 2025

Research Document

Abstract

Sikraken is a test suite generator tool created by Dr Christophe Meudec [1]. It is run against Test-Comp's benchmark, where I may take 7300000 CPU seconds to run [2]. By running the tool along with the Test-Comp benchmark on multiple EC2 instances or using the Elastic Container Service (ECS) or even a suitable alternative, the time it takes to run the tool against the benchmarks may be drastically reduced. This research document will provide an overview of the necessary tools to reduce the time required to run Sikraken against TestComp's benchmarks while also trying to minimise the costs for the services.

Introduction

The functionality aimed for this project is to provide a DevOps pipeline to automate checking out Sikraken's repository and then automatically run it on an AWS EC2, Amazon's cloud computing service [5]. Other functionality includes the collection and storage of data and then providing meaningful visualisation of it for developers working on Sikraken. The time taken to run Sikraken using clusters of EC2 instances will also be tested as well as the ways to minimise the costs of using them.

Overview of Technologies

TestCov

TestCov is a C test suite executor used by Test-Comp [24] and can output the coverage computation of a test suite by using gcov, lcov, and BenchExec for containerization [25]. It is a bottleneck for the project as it tends to run the test coverage sequentially and is slow to run, as it tends to output more than just the measured test coverage to the user, such as an .svg file of a graph of the branch coverage. The extra outputs may not be required for the long-term benefit of Sikraken.

Installing TestCov

- Create a folder where the TestCov repository will be stored
- Clone the TestCov repository from GitLab into the newly created folder
- If not already installed, install Python 3, pip, lcov and clang-tidy, e.g. “sudo apt install python”.
- Run the command pip install –user
- If the above command throws an error, run the command: and add /home/user/.local/bin to the PATH by running the command: PATH=$PATH:/home/user/.local/bin
- If an error occurs when running the command: testcov --no-isolation --test-suite "test/suites/suite-simple-if.zip" "test/test_simple-if.c" for testing the example from the readme file that displays the following:
  “ERROR:suite_validation:Compilation failed for harness output/harness.c:
  /usr/bin/ld: cannot find Scrt1.o: No such file or directory
  /usr/bin/ld: cannot find crti.o: No such file or directory
  /usr/bin/ld: skipping incompatible /usr/lib/x86_64-linux-gnu/libm.so when searching for -lm
  /usr/bin/ld: skipping incompatible /usr/lib/x86_64-linux-gnu/libm.a when searching for -lm
  /usr/bin/ld: cannot find -lm: No such file or directory
  /usr/bin/ld: skipping incompatible /usr/lib/x86_64-linux-gnu/libm.so when searching for -lm
  /usr/bin/ld: skipping incompatible /usr/lib/gcc/x86_64-linux-gnu/13/libgcov.a when searching for -lgcov
  /usr/bin/ld: cannot find -lgcov: No such file or directory
  /usr/bin/ld: skipping incompatible /usr/lib/gcc/x86_64-linux-gnu/13/libgcc.a when searching for -lgcc
  /usr/bin/ld: cannot find -lgcc: No such file or directory
  /usr/bin/ld: skipping incompatible /usr/lib/gcc/x86_64-linux-gnu/13/libgcc.a when searching for -lgcc
  /usr/bin/ld: cannot find -lgcc: No such file or directory
  collect2: error: ld returned 1 exit status” run the command “sudo apt install gcc-multilib”

GitHub and GitHub Actions

Sikraken’s code is stored within a GitHub repository. GitHub Actions provides a way to fulfil the main function of the project by allowing pipelines or workflows to automate tasks, known as jobs, which may be executed on a certain event or manually, and are defined in a .yaml file[3]. By using GitHub Actions, manual tasks such as checking out a repository or making a pull request and then deploying to an AWS EC2 can be automated [4].

Connecting To An EC2 Instance

There are two main ways to connect to an EC2 instance in order to run commands on it. One way was to use Secure Shell to connect to an instance using a .pem key, similar to how one would connect to an instance on their own computer’s terminal [72]. However, this is the less secure way of connecting since the .pem key will have to be base64 encoded and then stored inside the pipeline

[73], as a Secrets variable [74] and will have to be kept there in the long term in order to keep connecting to the EC2 instance.

A more secure way of connecting is by authenticating using OpenID Connect (OIDC) between GitHub Actions and AWS, as it removes the need to store the key to the EC2 instance within the pipeline, as GitHub issues a short-lived access token to a cloud provider such as AWS for validation. AWS also provides their own GitHub action to be used in the pipeline for configuring AWS credentials using OIDC [75].

Rackspace

Rackspace is a service that can help manage a customer's AWS account and the services tied to it, and also give suggestions for optimising the service, such as an EC2 instance, potentially allowing for the project to be more efficient in terms of cost and performance through offerings such as the Optimizer+ offer.

Optimizer+

Optimizer+ is a service from Rackspace which provides guidance in cost optimisation to customers through third-party tools that are native to the cloud. Customers who purchased this service can submit tickets for cost-optimisation and request advice on spending on infrastructure. The activities listed to help with cost optimisation that may be of relevance in this project include[34]:

- Using CloudHealth
- Initial Cost Optimisation Review
- Regular Cost Reviews
- Cost Guidance
- Savings Recommendations

CloudHealth

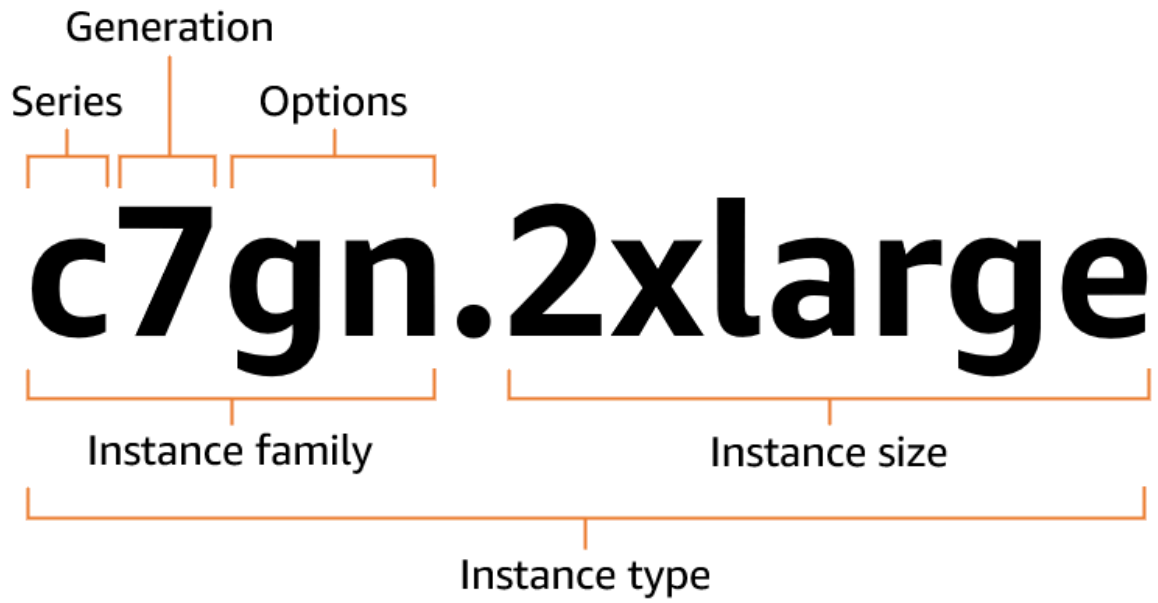
CloudHealth is a cloud management platform partnered with Rackspace, which can help customers visualise their spending on cloud services as well as quickly view potential cost-saving opportunities and automate their cloud management. Users with access to CloudHealth also get near real-time cost data, which may help in ensuring that the project stays within budget, among other tools used. [35]

Amazon Web Services (AWS) - Elastic Compute Cloud

AWS is Amazon's cloud platform, providing users with services such as Elastic Compute Cloud (EC2), which is necessary for fulfilling the main function of the project, which is to run Sikraken on an EC2. It acts as Amazon's "broadest and deepest compute platform" serving over 750 types of instances for a user [4]. With many use cases, its ability to provide computing capacity that is optimised for a given workload by using more instances when a workload for an instance gets heavy and reducing the amount when not needed allows for Sikraken to run faster while staying budget-friendly [6].

EC2 Instance Types

AWS currently gives users access to six different types of EC2 instances: General Purpose, Compute Optimised, Memory Optimised, Storage Optimised, Accelerated Computing, and High Performance Computing [7]. The table below will list the latest and standard options of the different EC2 instance Types, memory, their total threads, also referred to as vCPUs [8], CPU Cores and price per hour. Each type of instance has its ideal purpose, and all follow the same naming convention [9].



AWS EC2 Instance Type Naming Convention [9]

General Purpose

Provides balanced resources of computing power, memory and networking resources. Best used for web servers and code repositories [10].

Instance Type	Memory (GB)	Virtual CPUs/Total Threads	CPU Cores	Price Per Hour [11]	CPU Name/GHz [27]
m5.large	8	2	1	$0.107	Intel Xeon Platinum 8175/2.5GHz
m5.xlarge	16	4	2	$0.214	Intel Xeon Platinum 8175/2.5GHz
m5.2xlarge	32	8	4	$0.428	Intel Xeon Platinum 8175/2.5GHz

Compute Optimised

These instances are aimed towards more CPU-intensive applications that may benefit from higher performing CPUs and are best used for “batch processing workloads, media transcoding, high performance web servers, high performance computing (HPC), scientific modelling, dedicated gaming servers, ad server engines, and machine learning inference” [12]. Sikraken on rare occasions may take up 12GB of memory [31], however the eight total threads may achieve fast results.

Instance Type	Memory (GB)	Virtual CPUs/Total Threads	CPU Cores	Price Per Hour [11]	CPU Name/GHz [28]
c5.large	4	2	1	$0.096	Intel Xeon Platinum 8124M/3.0GHz
c5.xlarge	8	4	2	$0.192	Intel Xeon Platinum 8124M/3.0GHz
c5.2xlarge	16	8	4	$0.384	Intel Xeon Platinum 8124M/3.0GHz

Memory Optimised

Memory-optimised instances are better suited for workloads that process large data sets in memory [13]. It’s possible for Sikraken to take up to 12GB of memory on rare occasions, so this has the potential to be the best option given the lower price per hour in comparison to the c5.2xlarge EC2 instance model.

Instance Type	Memory (GB)	Virtual CPUs/Total Threads	CPU Cores	Price Per Hour [11]	CPU Name/GHz [27]
r5.large	16	2	1	$0.141	Intel Xeon Platinum 8175/2.5GHz
r5.xlarge	32	4	2	$0.282	Intel Xeon Platinum 8175/2.5GHz
r5.2xlarge	64	8	4	$0.564	Intel Xeon Platinum 8175/2.5GHz

Storage Optimised

Storage-optimised instances are better suited for workloads that require high and sequential read/write to very large data sets on local storage.[14].

Instance Type	Memory (GB)	Virtual CPUs/Total Threads	CPU Cores	Price Per Hour [11]	CPU Name/GHz [29]
d3.xlarge	32	4	2	$0.609	Intel Xeon Platinum 8259/2.5GHZ
d3.2xlarge	64	8	4	$1.219	Intel Xeon Platinum 8259/2.5GHZ
d3.4xlarge	128	16	8	$2.437	Intel Xeon Platinum 8259/2.5GHZ

Accelerated Computing

This type of instance uses “hardware accelerators, or co-processors, to perform functions, such as floating point number calculations, graphics processing, or data pattern matching, more efficiently than is possible in software running on CPUs.” [15].

Instance Type	Memory (GB)	Virtual CPUs/Total Threads	CPU Cores	Price Per Hour [11]	CPU Name/GHz [30]
g5.xlarge	16	4	2	$1.123	2nd Gen AMD EPYC 7R32/2.8 GHz
g5.2xlarge	32	8	4	$1.35296	2nd Gen AMD EPYC 7R32/2.8 GHz
g5.4xlarge	64	16	8	$1.81287	2nd Gen AMD EPYC 7R32/2.8 GHz

High-Performance Computing

These EC2 instances are more suited for workloads that need computing of the highest performance and are suited for tasks such as large and complex simulations and deep-learning workloads [16].

Instance Type	Memory	Virtual CPUs/Total Threads	CPU Cores	Price Per Hour [26]	CPU Name/GHz [31]
hpc7a.12xlarge	768	24	24	$7.7252	AMD EPYC 9R14/3.7 GHz
hpc7a.24xlarge	768	48	48	$7.7252	AMD EPYC 9R14/3.7 GHz
hpc7a.48xlarge	768	96	96	$7.7252	AMD EPYC 9R14/3.7 GHz

Amazon Web Services - Elastic Block Store

Elastic Block Store (EBS) is a service provided by AWS [17] that allows for an EBS volume to be attached to an EC2 instance [18]. An EBS volume may act as a type of persistent storage for an EC2, similar to a solid-state drive . Most EC2 services have a Nitro hypervisor classification (All of the instance types above), which support a maximum of 28 attachments [19]. EBS volumes that may be attached to an EC2 will be shared with other attachments, which are the network interface, which acts as a logical networking component that represents a virtual network card and the SSD instance store volume, which acts as temporary storage that lasts as long as EC2 uptime but offers faster speeds [20].

Amazon Machine Image (AMI)

The Amazon Machine Image (AMI) refers to an image that provides the necessary software to set up and also boot an EC2 instance. The current configuration for an EC2 instance can be copied using the AMI of that EC2 instance and then reused when creating other EC2 instances. This can allow for the same configuration to be used for cheaper or more expensive EC2 instances if there happens to be a need for it in the future. [78]

EBS Volume Types

The Elastic Block Store comes with different solid-state drive (SSD) volume types of storage, gp3, gp2, io2, io1 and different hard drive volumes with st1 and sc1 [21]. The gp3 and gp2 volume types are more suited to general usage, with a greater speed compared to hard drive volumes. The input-output per second (IOPS) in the SSD refers to the number of input-output operations, such as reading and writing storage, that may be performed per second and is measured in kilobytes [22]. The io2 and io1 volume types have significantly higher IOPS and throughput in comparison to the gp3 and gp2.

Instance Type	Volume Size (GB/TB)	Max Throughput (MB)	Max IOPS	IOPS Data Transfer (KB/s)	Pricing [23]

gp3	1 GB - 64 TB	2000	80,000	64	**Storage:** $0.088/GB-month **IOPS:** 3,000 IOPS free and $0.0055/provisioned IOPS-month over 3,000 **Throughput:** 125 MB/s free and $0.044/provisioned MB/s-month over 125
gp2	1 GB - 16 TB	250	16,000	16	$0.11 per GB-month of provisioned storage
io2	4 GB - 64 TB	4000	256,000	16	**Storage:** $0.138/GB-month **IOPS:** $0.072/provisioned IOPS-month up to 32,000 IOPS, $0.050/provisioned IOPS-month from 32,001 to 64,000 IOPS, $0.035/provisioned IOPS-month for greater than 64,000 IOPS
io1	4 GB - 16 TB	1000	64,000	16	$0.138 per GB-month of provisioned storage + $0.072 per provisioned IOPS-month
st1	125 GB - 16 TB	500	500	1000	$0.05 per GB-month of provisioned storage
sc1	125 GB - 16 TB	250	500	1000	$0.0168 per GB-month of provisioned storage

Estimate Of EBS Space Required And Price

Total: 4646.6 GB (Worst Case)

- Sikraken Storage: 222 MB [1]
- PTC-Solver: 2.6 MB [36]
- TestCov: 122 MB [37]
- Test Outputs Folder: 4.3 GB [38]
- Test Outputs: 276.9 MB - 581.9 MB [38]
-

The test outputs vary in size, ranging from the smallest folder at 276.9 MB to the largest at 581.9 MB, and consist of test results for multiple different problems that Sikraken was tested against. Fourteen testing outputs resulted in 4.3 GB of storage, implying that if that many were to be used when comparing past test results, then it would be a safe guess to use the storage it takes up as a reference to add to the estimated amount of space to provision for an EBS.

The table below will show the estimated price range for 5 GB, calculated using the prices listed in the above table as a reference. 5 GB was used to round up to a simpler number, in case any more storage may be required. However, the sc1 and st1 options appear to have a minimum of 125 GB and a maximum of 16 TB each, meaning that the calculated price will have to be at 125 GB.

Calculated Price Per Month in USD for 5GB Storage					
Sc1 (125 GB)	St1 (125 GB)	io1	io2	gp2	gp3
$2.1	$6.25	$0.69	$0.69	$0.55	$0.44

C

In order to achieve the functionality of processing the outputs at the end of a test run that are generated from TestCov, it is necessary to choose a programming language that can process files and visualise the data for Sikraken developers for current and past outputs for comparison purposes. Since GCC would already be installed on an EC2 instance, C would be a natural pick as a programming language, as no other installations would be necessary. Its speed as a programming language in comparison to Python, another language that may be used, as it would be necessary to install in order to run TestCov, would be much faster. [39]

Since the results file that TestCov outputs is in JSON format, it is necessary for the language to be able to read files in that format in order to process the results of the test run for the developer. cJson is a C library that provides an API for developers to easily read, write and modify JSON files [40]. However, cJSON [41] is a library that needs to be cloned from a repo, which may prompt more installation and more space for the EBS.

Python

Since Python will have to be installed on the EC2 for TestCov to run, no additional installation will be necessary to run the code that will read from the JSON file. While it's easier to write code with Python compared to C, it would be much slower, which may affect the speed at which the DevOps pipeline runs if the results of the test run are to be given at the end of the pipeline [39].

Python itself comes with a native library for reading JSON files [42], meaning that no additional installations will be necessary to get started. The syntax itself is also much simpler, so writing code to process the results file will be much faster.

Python also has simple file reading and writing capabilities, so that it may be used to write .html reports instead of Bash for a potentially easier time editing the .html file in code. [89]

Bash

Bash scripts are files used to automate running commands that may be run in a Linux terminal for the sake of efficiency. Bash scripts are used within the EC2 instances as they can be called to be executed from within the pipeline with AWS SSM's "send-command" command [77]. For the project, they are used to automate certain tasks, such as running another script to run Sikraken against the ECA categories benchmark [79].

The script run on the EC2 instance that's called from the pipeline is used to execute the script that runs Sikraken against the ECA category benchmarks, read the most recent folder generated, process the .html report that's generated at the end of a run with a Python script, so that the files that are viewed link to an S3 bucket, and then upload those files to the S3 bucket through the AWS CLI[88].

Amazon Web Services - Simple Storage Service (S3)

AWS's Simple Storage Service [43] acts as a cloud object storage service that stores data in buckets. Buckets are containers for objects and do not have a hierarchy in comparison to typical folder structures. Object storage refers to data, such as any file that's placed inside a bucket and has metadata, such as "the pieces of data that make up a file, adds all the user-created metadata to that file, and attaches a custom identifier". [44]

This can help fulfil the functionality requirement of processing the outputs at the end of a test run, as the files outputted by TestCov will be outputted to the EC2's file system (provided by an EBS volume after formatting [45]) and can be moved to an S3 through various means such as the AWS CLI [46].

Amazon Web Services - Command Line Interface (CLI)

AWS Command Line Interface allows a user to interact with AWS Services through a command-line shell [47]. AWS CLI can also be integrated into a GitHub Actions Pipeline, meaning that after a test run has been successfully completed, the files that have been output can be transferred to the S3 bucket through the pipeline. The AWS CLI can be used to perform functions such as invoking a Lambda function, given its name, a JSON payload, and how the payload should be read [85].

Uploading files from an S3 is also made possible through the AWS CLI, meaning that the bash script used in the EC2 instance can also upload the files generated by Sikraken to a specified S3 bucket using the “mv” command [88] or the “sync” command as it copies new or updated files recursively to a specified S3 bucket [100].

The AWS Systems Manager is a part of the AWS CLI and is what allows for the running of commands on other services, such as an EC2 instance [76].

Starting an EC2 instance [90], waiting for it to run [92] as well as stopping an EC2 instance [91] from the pipeline can also be achieved through AWS CLI.

For running tasks on an Elastic Container Service (ECS) the ecs run-tasks command can be utilised as it allows for a given number of tasks to be executed with a cluster name, task definition, ECS launch type, and networking configuration through setting up a Virtual Private Cloud (VPC) configuration by passing in a VPC subnet and security group.[107]. Optional arguments such as “--overrides“ may be used to override existing environment variables or add new ones to the containers in a task in order to set variables for a script inside the Sikraken container, allowing it to configure Sikraken’s settings when running it against benchmarks [107] and assignPublicIp allows for the tasks to connect to the internet [108].

For submitting a job on AWS Batch, the submit-job command can be used to run Sikraken against benchmarks after specifying a job name, a job queue and a job definition. The number of array jobs can be configured to decide how many vCPUs should be utilised, assuming each job takes a single vCPU, the amount of RAM to use per job, the number of times to run the job again in the event of an interruption, as well as the environment variables to pass to a container and what part of the command’s output should be displayed through the use of filtering. [155]

The status of a particular job among other information can also be checked by using the job’s ID through the describe-jobs command [163]. This can be useful as the GitHub Actions pipeline can call this command and check the status to decide whether the pipeline should wait or continue to the next step.

Amazon Web Services - Systems Manager (SSM)

The AWS Systems Manager is a service accessed through the AWS CLI and is responsible for the remote running of commands on the EC2 instance from the GitHub Actions pipeline through the “send-command” command that is offered after giving an instance ID and one or more commands through a string array. [77]

Since a bash script is run inside an EC2 instance in order to run Sikraken against the ECA categories benchmark, which is a part of SoSy Lab’s sv-benchmark repository that acts as a common verification task to evaluate “effectiveness and efficiency of state-of-the-art verification technology” [79], it is important that the pipeline waits for this script to finish before moving on with the next jobs. This requires SSM’s “list-command-invocations” command, which is required so that the pipeline has information on the status of the script and will continue waiting until the script is finished [80].

SSM can be installed on an Ubuntu EC2 instance via the Snap Package Manager [98]. This needs to be accounted for as Snap applications run in a custom environment suited for the specific Snap

application, including dedicated locations where data can be written [99], such as files created during a Sikraken test run, where the file paths may end up in a format beginning with "/var/snap/amazon-ssm-agent/…" if a relative file path is used.

Amazon Web Services - Lambda

AWS Lambda is a service that simply allows a developer to deploy and execute code on the cloud without the need to worry about managing any infrastructure beforehand [48]. AWS Lambda uses runtimes, whose role is to provide an environment for specific programming languages, relaying and context information, invocation events, and responses between the Lambda and the code written inside it.

The current supported programming languages are:

Programming Language	Version Supported
Ruby	3.2, 3.3, 3.4
.NET	8, 9 (Only as container image)
Python	9, 10, 11, 12, 13
Java	8, 11, 17, 21
Node.js	20, 22

When a function is defined as a container image, you must choose the runtime as well as a Linux Distribution.

For simpler functions to be used inside the Lambda, such as functions that do not require complex computation, AWS suggests that interpreted languages such as Python and Node.js will offer the fastest performance. [49]

The role that AWS Lambda will take in fulfilling the functionality of processing the output files from TestCov is that AWS provides a software development kit for Python called Boto3, which allows for Python code to interact with other AWS services [50], such as retrieving files or objects from an S3 [51]. By retrieving the output files from an S3 after they have been copied over from an EBS, a Python script can then be used to process them. Additionally, Lambda layers, .zip archive can be used in order to add extra code or data such as library dependencies [167].

Boto 3 Software Development Kit (SDK)

AWS' boto 3 SDK for Python provides developers a way to interact with AWS services using Python code, such as reading objects from an S3 bucket using functions such as "list_objects_v2", which may be used to read specific objects in an S3 bucket's directories based on delimiters or an object prefix [81] to find the most recent folder uploaded [82], such as the folders generated by the script to run Sikraken against the ECA category benchmark for further processing.

Objects in an S3 bucket that belong to an AWS account can also be opened by those who don't have access to that account through a presigned URL, which uses the S3 owner's credentials when opening the object, since S3 objects are private and only accessible to the owner by default [83]. Boto3's command "generate_presigned_url" allows for the URL of an object to be generated after being given a command such as "get_object" (Which retrieves an S3 object/file given its key and bucket it belongs to [84]), so that it would be accessible by anyone, given that they have the link.

Docker

Docker is a platform that allows for the creation and running of application containers. These containers give an application an isolated environment to execute in. This environment can also come with any dependencies that the application relies on, meaning that there wouldn't be a need to install the dependencies on an EC2 instance, such as ECLiPSe Prolog.

Containerising would also allow the app to run in the exact same way, no matter the environment, meaning that if a different cloud computing service were to be used, there would be no hassle in deploying Sikraken again [52]. Since TestCov will have to be used as well to benchmark the test inputs by Sikraken, it would be the best practice to containerise it as well and allow both Sikraken and TestCov containers to interact with each other through Docker Compose [56]. It's also possible to use AWS CLI commands within a Docker container [68], allowing for the benchmarks generated by TestCov to be sent directly to an S3 bucket with an AWS CLI command [69], which may skip the need for an EBS volume to be mounted to the EC2.

In order to containerise Sikraken and TestCov using Docker:

- It would be necessary to create containers in which both applications will run inside [53]
- Create a Docker image where all the code for Sikraken and its dependencies will be stored, such as ECLiPSe, as well as TestCov and its dependencies, in order for each container to get the necessary files that will run inside them [54].
- Store the images either locally or online [55].

To send a .C file to Sikraken to generate test inputs from a container, Docker's COPY instruction allows for a local file to be copied into a container's own file system [70]. The usage of Git commands within a Docker image is also possible, so cloning benchmarks from a Git repository into the container's file system is also possible [71].

Docker Commands

RUN

When creating a Dockerfile, it would be necessary to run certain Linux commands such as "sudo apt-get install" in order to install any dependencies of the application as seen in the Dockerfile best practices [94]. The "RUN" command in a Dockerfile runs any commands to create a new "layer" in the current image[95]. A layer contains "a set of filesystem changes - additions, deletions, or modifications" and can be saved to be used for other images [96].

WORKDIR

The WORKDIR command simply changes the working directory in the container's file system where the Docker commands are to be executed such as RUN.

ARG

The ARG command is used to define variables that users can pass when building a docker image [95]. This can be used to change the environment variables without them persisting after building the image which may be used for setting environment variables such as DEBIAN_FRONTEND to "noninteractive" in order to prevent any prompts to the user in a CLI [97] from taking place as this defeats the purpose of an automatic build.

ENV

The ENV command is used to set environment variables to a given value and will stay in the environment for the following instructions in the Dockerfile [95]. This is needed as ECLiPSe Prolog, a Sikraken dependency, requires the ECLIPSEDIR and PATH to be set in order to install it.

CMD

The CMD command is used to specify the command to be run in the container from the image [95].

VOLUME

The VOLUME command can be used to declare a persistent data store for containers. This can allow for two containers to safely share data between each other such as a benchmarks container for adding benchmarks to the data store while a Sikraken container can process the benchmarks added to the volume [105]. By separating the two into their own containers, the benchmarks repository will not have to be built for each Sikraken test run in the event an ECS will be used as its clusters are made for containerised applications [65].

Image Storing Options

While it's possible to simply store the Docker images locally on a machine once they're created, it would be more efficient to store them on the cloud, such as Docker Hub, which will prevent the need to install Docker on an EC2 and create the image from there, as well as allow other Sikraken developers to access the same image.

Docker Hub

Docker Hub provides users with a container registry where they can store and share their Docker images. It is the largest container registry with options for both public and private image repositories [57]. Images stored in Docker Hub could also be retrieved via the command line from an Ubuntu EC2 instance once Docker is installed on it, and then run the containers to deploy Sikraken and run TestCov against it [58].

GitHub Container Registry

The GitHub Container Registry is another alternative to using Docker Hub. It has the advantage of being tightly integrated into GitHub, which allows for any Sikraken containers to be placed on the same platform as the source code is stored on a GitHub repository. However, because GitHub Container Registry is designed to be used in conjunction with GitHub, if the Sikraken's source code were to be stored on another platform, such as GitLab, in the future, it wouldn't be very practical to use GitHub Container Registry [61].

Amazon Web Services - Elastic Container Registry (ECR)

Elastic Container Registry is AWS's solution to storing and managing container images with a price of $0.10 per GB of storage per month [59]. While ECR requires a monthly payment, running container images on an EC2 from Docker Hub makes it necessary for a user to install Docker on an EC2 instance and then pull the Docker image from the terminal using Docker's CLI. ECR simplifies this process by allowing the EC2 to pull a container image from the ECR using one command through the AWS CLI, thus eliminating the need for any further installations [60]. Given the ease of use when

using ECR to deploy a containerised application on an EC2, this might be the optimal solution for this problem.

Terraform

Terraform is an infrastructure-as-code tool, allowing for the usage of configuration files to define and manage infrastructure. The configuration files are written in Hashicorp Configuration Language (HCL) or JSON. Terraform supports many different cloud environments, including AWS. The benefits of using Terraform include consistency; if the cloud service provider that Sikraken is deployed on is changed in the future, then the language used to write the code for the configuration will still be compatible with other providers, such as Azure, given some changes to suit their resources better, for example. If any changes are made in the future to the infrastructure, it would also be very easy to go back to a previous version of the infrastructure through the use of a configuration file if the changes go wrong [62].

Terraform Resources

Terraform resources are components from an infrastructure that are created and managed through Terraform such as compute instances [149]. Resources are grouped and offered through plugins known as “Providers” that allow Terraform to interact with a given cloud provider [150].

Terraform Modules

Terraform modules are collections of Terraform resources which are managed together by Terraform. Modules are reusable and can be used to configure common components in a cloud infrastructure which are publicly available on the Terraform Registry (Link leads to AWS modules in Terraform Registry). Modules stored in the Terraform public registry are managed by HashiCorp themselves, their partners as well as the Terraform community and are free for anyone to use and can also be downloaded automatically if a source and version are specified when writing a module in Terraform. [151]

Amazon Web Services - Elastic Container Service (ECS)

ECS is a service provided by AWS that allows for one or more tasks to be executed by multiple EC2 instances, therefore processing that task much faster in comparison to just using one instance. A task itself is a container, such as a Docker container that is currently running, and the configuration for a task is done through a task definition [63].

ECS Task Definition

A task definition is a text file that’s written in JSON in which parameters such as the Docker image of the container, the CPU and memory resources to be allocated to the container, and a command that will be run once the container starts running [64]. Any users of the ECS only pay for the resources being used, such as the EC2 instances or any vCPUs that may be used in the underlying infrastructure of the ECS [66].

ECS Clusters

A cluster in the context of ECS is a “logical grouping of tasks or services that provides the infrastructure capacity for your containerized applications”. Clusters have different ways to be set up, such as Fargate, where the infrastructure is set up automatically for the user, Amazon ECS Managed

Instances, where AWS manages the underlying infrastructure of the EC2 instances involved, providing a balance between performance, cost-effectiveness and operational simplicity. The last option is to manually set up the EC2 instances involved, which may be used if an EC2 infrastructure already exists [65]. A cluster could also be defined as a grouping of EC2 instances if the choice is made to set up the cluster using EC2 instances rather than Fargate or Amazon ECS Managed Instances [67].

Through the use of an ECS service (a long-running set of tasks of the same task definition [63]), the chosen number of tasks can be run simultaneously within a cluster. If a task definition could involve a Docker image of TestCov, it may be possible to run the benchmarks of different problems in parallel, thus speeding up the process of getting test run results back to Sikraken developers.

ECS With EC2

EC2 in the context of ECS, may be more expensive in comparison to the other options as provisioning the EC2 instances manually would also require assigning persistent storage to each one in the form of EBS [101] to act as a file system as there needs to be a way to store the files generated by benchmarks before uploading them to an S3 bucket, and the EBS is to be paid for monthly [23]. This may counteract the fact that it may be more effective to use manually managed EC2 instances for an ECS for the sake of having more control if the task of running Sikraken against benchmarks doesn't need all the control this way of using the ECS provides.

ECS Fargate

ECS Fargate is another way of using ECS without the need for provisioning any EC2 instances. Instead, only certain information such as the CPU and memory requirements need to be specified and the underlying infrastructure is handled by AWS as it's completely serverless, allowing for the focus to be kept on running Sikraken against its benchmarks [102]. In terms of pricing, only the resources being used to run the task are billed to the nearest second [103] so if there's no need to use the ECS for a certain amount of time, then there's no extra cost to be paid.

ECS Managed Instances

ECS Managed Instances is similar to a hybrid between Fargate and ECS with EC2 in which the user will be able to decide on the type of EC2 instances to act as the underlying infrastructure for ECS but AWS will be responsible for managing them. It may act as the best option if there's a need for specific EC2 features such as a graphics processing unit, high networking or high storage throughput [104]. Since Sikraken runs as expected on basic EC2 instances with just an EBS and 4 vCPUs and 32GB of RAM this may work as an option however the additional features would not be necessary and Fargate may prove to be a better option.

Deploying To ECS Fargate

In order to deploy an application such as Sikraken from GitHub Actions to ECS, it's necessary to configure the credentials such as in the case of connecting to an EC2 from GitHub Actions. Logging into ECR is required in order to upload a Docker image such as one for the Sikraken repository which will be built within the pipeline before uploading to ECR. A task definition is also kept inside the repository of the pipeline, separate from the actual Sikraken repository, where the container defined inside can then be overwritten by the one just uploaded to ECR.[106]

This can then be followed up by having the pipeline execute a script in the repository to run ECS tasks using the "ecs run-tasks" command from the AWS CLI.

ECS Data Storage Options

For storing the benchmarks that the Sikraken container will have to process in an ECS cluster the three main storage options for Fargate appear to be the Elastic File System (EFS) [128], EBS and bind mounts. EFS itself is a separate AWS resource for providing persistent data storage that grows and shrinks by itself as files are added to it and is commonly used for tasks requiring storage functionalities such as low latency for tasks such as web serving.

EBS acts similarly to its usage on an EC2, acting as persistent storage for a task if configured to do so and is typically used for data intensive tasks such as databases. Bind mounts in contrast to the other two options are ephemeral and are commonly used for sharing a volume from one container to others in the same ECS task [113].

In terms of pricing EFS appears to be the most expensive at $0.33 per GB each month alongside charging for $0.03 per GB read and $0.07 per GB write [114]. The EBS pricing would be $0.088 per GB per month [23].

The bind mount storage option for ECS tasks if configured to be ephemeral [115] has 20 GB by default that comes with no cost and only additional storage is charged at $0.000122 per GB each hour [103] it may be enough to hold the C directory of the benchmarks repository at 10.7GB after cloning the entirety of the benchmarks repository [79] and still hold additional data, potentially making this the most cost-effective option at the cost that the C benchmarks repository may have to be containerised and have its contents copied over to a volume shared with a Sikraken container each test run.

ECS Fargate Networking

ECS Fargate tasks require networking to be set up when running them in order to allow for multiple containers in a task to communicate with each other such as allowing for a Sikraken container to wait for until its benchmarks container is finished adding benchmarks to a docker volume they both share. The networking mode used in ECS Fargate is “awsvpc” therefore requiring its configuration when running a task.[109]

To configure a task’s networking using the run-task command, it’s required to pass in VPC subnets and a security group in order to configure a task’s networking [107]. A VPC itself is an isolated virtual network for AWS resources defined by users, such as an ECS infrastructure and the subnets within a VPC are what allow for resources to be deployed in a VPC [110]. The security group itself acts as a firewall, controlling inbound and outbound traffic for the containers [111]. When configuring the networking the default VPC and subnets can be used [111] as well as the default security group that comes with a VPC [112].

ECLiPSe Prolog

Prolog is the language that Sikraken’s symbolic executor and solver is written in [1]. Sikraken itself depends on ECLiPSe Prolog, a software system for the development and deployment of constraint programming applications [93] and will have to be taken into account when containerising Sikraken as the container will require all of Sikraken’s dependencies to be installed with it.

AWS Batch

AWS Batch is another service provided by AWS with the focus on batch computing for containerised workloads, such as a Sikraken container for running against benchmarks and is also fully managed by AWS [131]. Similarly to ECS, there’s no additional cost for using it as only the underlying AWS resources, such as EC2 instances and Fargate, are charged [132]. Although the scaling of AWS

resources are already handled by AWS, usage of specific EC2 instances are allowed as well as using spot instances for cheaper prices [133].

Under the hood, AWS Batch utilises AWS ECS to execute containerised jobs and also make it easy to use spot instances as the compute environments used in Batch can be configured to be completely made up of spot instances and which are used to run the jobs in Batch [143].

Batch Job

A job in AWS Batch is simply a unit of work such as a Docker container with a command to execute on start up [137]. AWS Batch themselves also set container environment variables which may help with achieving parallelism such as “AWS_BATCH_JOB_ARRAY_INDEX” to know the index of the job in a job array being submitted [138]. Jobs can also be dependent on each other meaning that a certain job can only run after a given job is successful [140]. This would allow for a container that holds Sikraken’s scripts for generating reports after a test run to be executed after the main job of running Sikraken against its benchmarks are finished.

Batch Array Job

An array job refers to a job that shares the same parameters as other jobs such as the batch job definition to use and is the most efficient way of running parallel jobs such as running Sikraken benchmarks with many vCPUs. The size of an array job is between 2 and 10000 and the jobs which are run and managed within the array are known as child jobs and they can all be submitted with one command. [139]

Batch Job Definition

Similar to ECS task definitions, a job definition for AWS Batch is what decides how a job should be run by Batch such as the docker container to be executed, the vCPUs and memory to assign for each job as well as environment variables. [136]

Compute Environment And Job Queue

When an AWS Batch job is submitted, they are placed into a job queue before being set to be executed by a compute environment and can also be configured to have a priority set for each job submitted [141]. A compute environment contains ECS container instances to run the containerised jobs within the service. Container instances are EC2 instances which run an ECS container agent for tasks such as starting and stopping ECS tasks if ECS sends a request to do so [144] and are also registered to a cluster [145].

EC2 Spot Instances

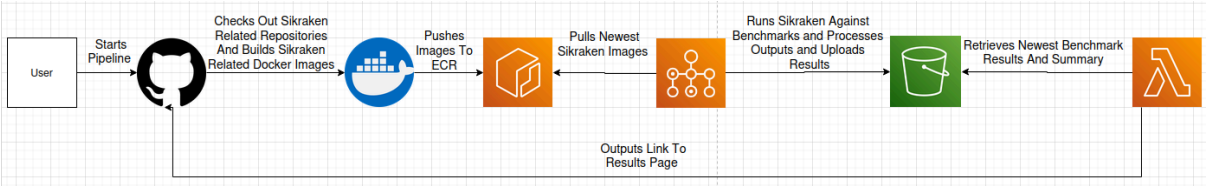
EC2 spot instances act as AWS’ spare EC2 capacity and are priced at a discount of up 90% in comparison to using standard EC2 instances which may allow for savings to be made. The trade-off for the discount is that any EC2 spot instances being used may be interrupted by AWS and may stop, hibernate or terminate the instance in the event that AWS needs the spare capacity. [134]. The pricing for each EC2 spot instance is dependent on the EC2 instance’s standard pricing and is gradually adjusted based on “long-term trends in supply and demand for Spot Instance capacity”.[135]

AWS (Identity And Access Management) IAM Role

AWS IAM itself is a service that allows users to control authentication and authorisation to their AWS resources, such as managing permissions [152]. An IAM identity can be associated with various policies, which define the actions that an identity is authorised to perform [153].

IAM roles are a type of identity that is intended to be used for the short term and assumed by whoever needs it. This makes it so that those who assume an IAM role aren't given long-term credentials such as passwords or access keys, and instead are given temporary credentials.IAM roles can be used to grant access to users and applications who may not typically have access to AWS resources or for AWS services to perform actions on other AWS services. The use of service roles is a type of IAM role that allows a service to perform actions on a user's behalf. [154]

Current Architecture Diagram



Conclusion

In conclusion, the goal of running Sikraken on an EC2 through a DevOps pipeline can be achieved through the technologies of GitHub Actions and AWS EC2. However, TestCov will also need to be run on the EC2 in order to benchmark the test inputs generated by Sikraken. Furthermore, the containerisation of both applications may be necessary as certain Sikraken dependencies, such as ECLiPSe prolog, will not have to be installed on an EC2 instance if it's already within the container.

With the containerisation of both applications, there will need to be a way for Sikraken and TestCov to communicate since Sikraken will take a .C file as input and then output test inputs for TestCov to benchmark. This can be achieved through Docker Compose.

The functionality of processing the outputs by TestCov can be achieved through AWS services such as S3 and AWS Lambda. By moving the result files generated by TestCov to an S3 bucket, AWS Lambda can take these files and process them through a Python script and output the results to the DevOps pipeline.

Tools such as Terraform could also take a role in the project because of the application's ability to write the infrastructure of the project as code through configuration files can act as a sort of save point for the infrastructure, allowing for further experimentation with AWS services in the future without causing permanent damage. The infrastructure written can also be applied to other cloud service providers, such as Azure, if the need to change providers takes place in the future.

The use of ECS may also allow for Sikraken to output benchmarks for different test inputs in parallel, which may speed up getting test results back to Sikraken through the use of tasks and ECS clusters. Batch can also be an alternative to ECS as it allows for the use of EC2 instances which would also be managed by AWS, and can also use compute environments that are completely made up of EC2 spot instances to save on costs.

References

[1] Meudec, C. (2025). Sikraken: A Test Suites Generator for C Code (Github Repository)
Available At: https://github.com/echancrure/Sikraken?tab=readme-ov-file
[2] Test-Comp (2025) Benchmark Setup (Benchmark Results)
Available At:
https://test-comp.sosy-lab.org/2025/results/results-verified/META_Cover-Branches_sikraken.table.html#/
[3] GitHub (2025) Understanding GitHub Actions (Webpage)
Available At: https://docs.github.com/en/actions/get-started/understand-github-actions
[4] Biradar, M. & Moolya, S. (2022) Integrating with GitHub Actions – CI/CD pipeline to deploy a Web App to Amazon EC2
Available At:
https://aws.amazon.com/blogs/devops/integrating-with-github-actions-ci-cd-pipeline-to-deploy-a-web-app-to-amazon-ec2/
[5] Amazon Web Services (2025) Amazon EC2 (Webpage)
Available At: https://aws.amazon.com/ec2/
[6] Amazon Web Services (2025) What Is Amazon EC2? - Amazon Elastic Compute Cloud
Available At: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html
[7] Amazon Web Services (2025) Amazon EC2 instance type specifications
Available At:
https://docs.aws.amazon.com/ec2/latest/instancetypes/ec2-instance-type-specifications.html
[8] Amazon Web Services (2025) CPU options for Amazon EC2 instances
Available At: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instance-optimize-cpu.html
[9] Amazon Web Services (2025) Amazon EC2 instance type naming conventions
Available At: https://docs.aws.amazon.com/ec2/latest/instancetypes/instance-type-names.html
[10] Amazon Web Services (2025) Specifications for Amazon EC2 general purpose instances
Available At: https://docs.aws.amazon.com/ec2/latest/instancetypes/gp.html
[11] Amazon Web Services (2025) Amazon EC2 On-Demand Pricing
Available At: https://aws.amazon.com/ec2/pricing/on-demand/
[12] Amazon Web Services (2025) Specifications for Amazon EC2 compute optimized instances
Available At: https://docs.aws.amazon.com/ec2/latest/instancetypes/co.html
[13] Amazon Web Services (2025) Specifications for Amazon EC2 memory optimized instances
Available At: https://docs.aws.amazon.com/ec2/latest/instancetypes/mo.html
[14] Amazon Web Services (2025) Specifications for Amazon EC2 storage optimized instances
Available At: https://docs.aws.amazon.com/ec2/latest/instancetypes/so.html
[15] Amazon Web Services (2025) Specifications for Amazon EC2 accelerated computing instances
Available At: https://docs.aws.amazon.com/ec2/latest/instancetypes/ac.html
[16] Amazon Web Services (2025) Specifications for Amazon EC2 high-performance computing instances
Available At: https://docs.aws.amazon.com/ec2/latest/instancetypes/hpc.html
[17] Amazon Web Services (2025) Amazon Elastic Block Store
Available At: https://aws.amazon.com/ebs/
[18] Amazon Web Services (2025) Attach an Amazon EBS volume to an Amazon EC2 instance
Available At: https://docs.aws.amazon.com/ebs/latest/userguide/ebs-attaching-volume.html
[19] Amazon Web Services (2025) Amazon EBS volume limits for Amazon EC2 instances
Available At: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/volume_limits.html
[20] Amazon Web Services (2025) Instance store temporary block storage for EC2 instances
Available At: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/InstanceStorage.html
[21] Amazon Web Services (2025) Amazon EBS volume types
Available At: https://docs.aws.amazon.com/ebs/latest/userguide/ebs-volume-types.html
[22] Amazon Web Service (2025) Amazon EBS I/O characteristics and monitoring
Available At: https://docs.aws.amazon.com/ebs/latest/userguide/ebs-io-characteristics.html

[23] Amazon Web Service (2025) Amazon EBS pricing
Available At: https://aws.amazon.com/ebs/pricing/
[24] IEEE Explore (2019) TestCov: Robust Test-Suite Execution and Coverage Measurement
Available At: https://ieeexplore.ieee.org/document/8952265
[25] PyPi (2022) TestCov
Available At: https://pypi.org/project/TestCov/
[26] Amazon Web Services (2025) AWS Pricing Calculator
Available At: https://calculator.aws/#/addService
[27] CPU Benchmarks (2018) Intel Xeon Platinum 8175M Benchmark
Available At:
https://www.cpubenchmark.net/cpu.php?cpu=Intel+Xeon+Platinum+8175M+%40+2.50GHz&id=3311
[28] CPU Benchmarks (2018) Intel Xeon Platinum 8124M Benchmark
Available At:
https://www.cpubenchmark.net/cpu.php?cpu=Intel+Xeon+Platinum+8124M+%40+3.00GHz&id=3352
[29] CPU Benchmarks (2020) Intel Xeon Platinum 8259CL Benchmark
Available At:
https://www.cpubenchmark.net/cpu.php?cpu=Intel+Xeon+Platinum+8259CL+%40+2.50GHz&id=3671
[30] CPU Benchmarks (2020) AMD EPYC 7R32 Benchmark
Available At: https://www.cpubenchmark.net/cpu.php?cpu=AMD+EPYC+7R32&id=3894
[31] CPU Benchmarks (2023) AMD EPYC 9R14 Benchmark
Available At: https://www.cpubenchmark.net/cpu.php?cpu=AMD+EPYC+9R14&id=5635
[32] Test-Comp (2024) coverage-branches.ReachSafety-ECA – BenchExec results
Available At:
https://test-comp.sosy-lab.org/2025/results/results-verified/coverage-branches.ReachSafety-ECA.table.html#/table?pageSize=2500
[33] Rackspace (2025) About Rackspace Technology
Available At: https://www.rackspace.com/about
[34] Rackspace (2016) Current Offerings
Available At: https://docs.rackspace.com/docs/rackspace-elastic-engineering-and-optimizer
[35] Rackspace (2016) CloudHealth
Available At: https://docs.rackspace.com/docs/cloudhealth
[36] Meudec, C. (2023). PTC-Solver (Github Repository)
Available At: https://github.com/echancrure/PTC-Solver/tree/master
[37] SoSy-Lab (2025) TestCov
Available At: https://gitlab.com/sosy-lab/software/test-suite-validator
[38] Meudec, C. (2025) ECA Category Result
Available At: ECA_category_results.zip
[39] Sultan, R. (2025) Why is C Faster Than Python? OS-Level Perspective
Available At:
https://medium.com/@raihansltn/why-is-c-faster-than-python-os-level-perspective-04d6e485e3ae
[40] GeeksforGeeks (2023) cJSON JSON File Write/Read/Modify in C
Available At: https://www.geeksforgeeks.org/c/cjson-json-file-write-read-modify-in-c/
[41] Gamble, D. (2024) cJSON
Available At: https://github.com/DaveGamble/cJSON
[42] GeeksforGeeks (2019) Read JSON file using Python
Available At: https://www.geeksforgeeks.org/python/read-json-file-using-python/
[43] AWS (2025) Cloud Object Storage | Store & Retrieve Data Anywhere | Amazon Simple Storage Service
Available At: https://aws.amazon.com/s3/
[44] AWS (2025) What is Object Storage? - Object Storage - AWS
Available At: https://aws.amazon.com/what-is/object-storage/
[45] AWS (2025) Make an Amazon EBS volume available for use - Amazon EBS

Available At: https://docs.aws.amazon.com/ebs/latest/userguide/ebs-using-volumes.html
[46] AWS (2025) Using Amazon S3 with Amazon EC2 - Amazon Elastic Compute Cloud
Available At: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AmazonS3.html
[47] AWS (2013) AWS Command Line Interface
Available At: https://aws.amazon.com/cli/
[48] Kelley, K. (2024) What is AWS Lambda, and How Can You Write Functions and Code With it?
Available At: https://pg-p.ctme.caltech.edu/blog/cloud-computing/what-is-aws-lambda-functions-code
[49] AWS (2025) Lambda runtimes - AWS Lambda
Available At: https://docs.aws.amazon.com/lambda/latest/dg/lambda-runtimes.html
[50] AWS (2025) Boto3 documentation — Boto3 Docs 1.40.53 documentation
Available At: https://boto3.amazonaws.com/v1/documentation/api/latest/index.html#
[51] AWS (2025) get_object - Boto3 1.40.52 documentation
Available At:
https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/s3/client/get_object.html
[52] DataDog (2022) What Are Containerized Applications? | Datadog
Available At:
https://www.datadoghq.com/knowledge-center/containerized-applications/#what-are-containerized-applications
[53] Docker Docs (2025) What is a container?
Available At: https://docs.docker.com/get-started/docker-concepts/the-basics/what-is-a-container/
[54] Docker Docs (2025) What is an image?
Available At: https://docs.docker.com/get-started/docker-concepts/the-basics/what-is-an-image/
[55] Docker Docs (2025) What is a registry
Available At: https://docs.docker.com/get-started/docker-concepts/the-basics/what-is-a-registry/
[56] Docker Docs (2025) Multi-container applications
Available At:
https://docs.docker.com/get-started/docker-concepts/running-containers/multi-container-applications/
[57] Dockers (2021) What is Docker Hub? | Docker
Available At: https://www.docker.com/products/docker-hub/
[58] Okonkwo, W. (2025) Deploying a Custom Web Application with Docker & Docker Compose on AWS EC2
Available At:
https://dev.to/teetoflame/deploying-a-custom-web-application-with-docker-docker-compose-on-aws-ec2-423n
[59] AWS (2024) Fully Managed Container Registry – Amazon Elastic Container Registry Pricing - Amazon Web Services
Available At: https://aws.amazon.com/ecr/pricing/
[60] Starr, S. (2023) Deploy a simple dockerized web app using AWS ECR & EC2
Available At:
https://medium.com/@sstarr1879/deploy-a-simple-dockerized-web-app-using-aws-ecr-ec2-e6a3569a6cf5
[61] JFrog (2022) Comparing Docker Hub and GitHub Container Registry
Available At:
https://jfrog.com/devops-tools/article/comparing-docker-hub-and-github-container-registry/
[62] Scale Computing (2025) What Is Terraform? Overview, Functions, and Benefits
Available At: https://www.scalecomputing.com/resources/what-is-terraform
[63] Nguyen, T. (2017) Gentle Introduction to How AWS ECS Works with Example Tutorial
Available At:
https://medium.com/boltops/gentle-introduction-to-how-aws-ecs-works-with-example-tutorial-cea3d27ce63d
[64] AWS (2025) Amazon ECS task definitions - Amazon Elastic Container Service

Available At: https://docs.aws.amazon.com/AmazonECS/latest/developerguide/task_definitions.html
[65] AWS (2025) Amazon ECS clusters - Amazon Elastic Container Service
Available At: https://docs.aws.amazon.com/AmazonECS/latest/developerguide/clusters.html
[66] AWS (2025) Fully Managed Container Orchestration – Amazon ECS Pricing– Amazon Web Services
Available At: https://aws.amazon.com/ecs/pricing/
[67] Karishma (करिश्मा) Working with ECS.
Available At: https://medium.com/codex/working-with-ecs-cece5f447936
[68] Raina, A. (2024) How to Run AWS CLI in Docker - Collabnix
Available At: https://collabnix.com/how-to-run-aws-cli-in-docker/
[69] G, J M. (2023) Uploading Files to an S3 Bucket using AWS CLI
Available At: https://medium.com/@josemanuel.gilperez/uploading-files-to-an-s3-bucket-using-aws-cli-4ac89a0b024b
[70] GeeksForGeeks (2025) Docker - COPY Instruction
Available At: https://www.geeksforgeeks.org/devops/docker-copy-instruction/
[71] Stack Overflow (2018) Can we include git commands in docker image?
Available At: https://stackoverflow.com/questions/50870161/can-we-include-git-commands-in-docker-image
[72] AWS (2025) Connect to Your Linux Instance using an SSH client
Available At: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/connect-linux-inst-ssh.html
[73] Johanning, Josh. (2023) Using GitHub Actions Secrets to Store Certificates/Keys
Available At: https://josh-ops.com/posts/storing-certificates-as-github-secrets/
[74] GitHub Docs (2025) Secrets
Available At: docs.github.com/en/actions/concepts/security/secrets
[75] GitHub Docs (2025) Configuring OpenID Connect in Amazon Web Services
Available At: https://docs.github.com/en/actions/how-tos/secure-your-work/security-harden-deployments/oidc-in-aws
[76] AWS (2025) What Is AWS Systems Manager?
Available At: https://docs.aws.amazon.com/systems-manager/latest/userguide/what-is-systems-manager.html
[77] AWS (2025) send-command
Available At: https://docs.aws.amazon.com/cli/latest/reference/ssm/send-command.html
[78] AWS (2025) Amazon Machine Images in Amazon EC2
Available At: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AMIs.html
[79] SoSy Lab (2025) SoSy-Lab / Benchmarking / SV-Benchmarks · GitLab
Available At: https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks
[80] AWS (2024) list-command-invocations — AWS CLI 2.32.3 Command Reference
Available At: https://docs.aws.amazon.com/cli/latest/reference/ssm/list-command-invocations.html
[81] AWS (2015) list_objects_v2 - Boto3 1.35.5 documentation
Available At: https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/s3/client/list_objects_v2.html
[82] AWS (2024) Working with object metadata - Amazon Simple Storage Service
Available At: https://docs.aws.amazon.com/AmazonS3/latest/userguide/UsingMetadata.html
[83] AWS (2025) Sharing objects using presigned URLs - Amazon Simple Storage Service
Available At: https://docs.aws.amazon.com/AmazonS3/latest/userguide/ShareObjectPreSignedURL.html
[84] AWS (2015) get_object - Boto3 1.40.52 documentation

Available At:
https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/s3/client/get_object.html
[85] AWS (2025) invoke — AWS CLI 2.32.3 Command Reference
Available At: https://docs.aws.amazon.com/cli/latest/reference/lambda/invoke.html#cli-binary-format
[86] AWS (2025) Start a session - AWS Systems Manager
Available At:
https://docs.aws.amazon.com/systems-manager/latest/userguide/session-manager-working-with-sessions-start.html
[87] AWS (2025) End a session - AWS Systems Manager
Available At:
https://docs.aws.amazon.com/systems-manager/latest/userguide/session-manager-working-with-sessions-end.html
[88] AWS (2025) mv — AWS CLI 2.32.3 Command Reference
Available At: https://docs.aws.amazon.com/cli/latest/reference/s3/mv.html
[89] GeeksforGeeks (2019) Writing to file in Python
Available At: https://www.geeksforgeeks.org/python/writing-to-file-in-python/
[90] AWS (2025) start-instances — AWS CLI 2.32.6 Command Reference
Available At: https://docs.aws.amazon.com/cli/latest/reference/ec2/start-instances.html
[91] AWS (2025) stop-instances — AWS CLI 1.27.26 Command Reference
Available At: https://docs.aws.amazon.com/cli/latest/reference/ec2/stop-instances.html
[92] AWS (2022) instance-running — AWS CLI 2.32.6 Command Reference
Available At: https://docs.aws.amazon.com/cli/latest/reference/ec2/wait/instance-running.html
[93] ECLiPSe (2025) ECLiPSe Home
Available At: https://eclipseclp.org/
[94] Docker Docs (2024) Building Best Practices
Available At: https://docs.docker.com/build/building/best-practices/
[95] Docker Docs (2024) Dockerfile reference
Available At: https://docs.docker.com/reference/dockerfile/
[96] Docker Docs (2024) Understanding the image layers
Available At:
https://docs.docker.com/get-started/docker-concepts/building-images/understanding-image-layers/
[97] Nayak, G R (2021) Why is DEBIAN_FRONTEND=noninteractive discouraged in Docker files?
Available At: https://bobcares.com/blog/debian_frontendnoninteractive-docker/
[99] AWS (2026) Install SSM Agent on Ubuntu Server 16.04 LTS 64-bit (Snap), 18.04, 20.04, 22.04 LTS, 23.10, 24.04 LTS, 24.0, and 25.04 - AWS Systems Manager
Available At:
https://docs.aws.amazon.com/systems-manager/latest/userguide/agent-install-ubuntu-64-snap.html
[99] Robotics Documentation (2025) Snap data and file storage
Available At:
https://canonical-robotics.readthedocs-hosted.com/en/latest/explanations/snaps/snap-data-and-file-storage/
[100] AWS (2018) sync — AWS CLI 2.9.6 Command Reference
Available At:
https://awscli.amazonaws.com/v2/documentation/api/2.9.6/reference/s3/sync.html#
[101] AWS (2026) What is Amazon Elastic Block Store? - Amazon EBS
Available At: https://docs.aws.amazon.com/ebs/latest/userguide/what-is-ebs.html
[102] Padghan, V.(2018) AWS Fargate — A Beginner's Guide To AWS Elastic Container Service
Available At: https://medium.com/edureka/aws-fargate-85a0e256cb03
[103] AWS (2026) AWS Fargate Pricing
Available At: https://aws.amazon.com/fargate/pricing/
[104] ahmedjama.com (2025) ECS Managed Instances: A practical comparison with Fargate and EC2

Available At: https://ahmedjama.com/blog/2025/10/ecs/ecs-managed-instances-comparison/
[105] Docker Docs (2024) Volumes
Available At: https://docs.docker.com/engine/storage/volumes/
[106] GitHub Docs (2026) Deploying to Amazon Elastic Container Service - GitHub Docs
Available At:
https://docs.github.com/en/actions/how-tos/deploy/deploy-to-third-party-platforms/amazon-elastic-container-service
[107] AWS (2024) run-task — AWS CLI 2.33.23 Command Reference
Available At: https://docs.aws.amazon.com/cli/latest/reference/ecs/run-task.html
[108] AWS (2026) Connect Amazon ECS applications to the internet - Amazon Elastic Container Service
Available At:
https://docs.aws.amazon.com/AmazonECS/latest/developerguide/networking-outbound.html #
[109] Peck, N. (2018) Task Networking in AWS Fargate | Amazon Web Services
Available At: https://aws.amazon.com/blogs/compute/task-networking-in-aws-fargate/
[110] AWS (2023) What Is Amazon VPC? - Amazon Virtual Private Cloud
Available At: https://docs.aws.amazon.com/vpc/latest/userguide/what-is-amazon-vpc.html
[111] AWS (2026) Set up to use Amazon ECS - Amazon Elastic Container Service
Available At:
https://docs.aws.amazon.com/AmazonECS/latest/developerguide/get-set-up-for-amazon-ecs.html
[112] AWS (2026) Default security groups for your VPCs - Amazon Virtual Private Cloud
Available At: https://docs.aws.amazon.com/vpc/latest/userguide/default-security-group.html
[113] AWS (2026) Storage options for Amazon ECS tasks - Amazon Elastic Container Service
Available At:
https://docs.aws.amazon.com/AmazonECS/latest/developerguide/using_data_volumes.html
[114] AWS (2019) Amazon Elastic File System (EFS) | Cloud File Storage | Pricing
Available At: https://aws.amazon.com/efs/pricing/
[115] AWS (2026) Use bind mounts with Amazon ECS - Amazon Elastic Container Service
Available At: https://docs.aws.amazon.com/AmazonECS/latest/developerguide/bind-mounts.html
[116] AWS (2025) Amazon S3 Pricing
Available at: https://aws.amazon.com/s3/pricing/
[117] AWS (2025) Amazon EBS snapshots - Amazon EBS
Available At: https://docs.aws.amazon.com/ebs/latest/userguide/ebs-snapshots.html
[118] GitHub (2025) GitHub-hosted runners - GitHub Docs
Available At: https://docs.github.com/en/actions/concepts/runners/github-hosted-runners
[119] GitHub Docs (2025) Store information in variables - GitHub Docs
Available At:
https://docs.github.com/en/actions/how-tos/write-workflows/choose-what-workflows-do/use-variables
[120] AWS re:Post (2021) Calling AWS services with AWS SDK outside of AWS
Available At:
https://repost.aws/questions/QUXhdo3hWEQlSuhmVidopRVQ/calling-aws-services-with-aws-sdk-outside-of-aws
[121] AWS (2025) Credentials - Boto3 1.41.5 documentation
Available At: https://boto3.amazonaws.com/v1/documentation/api/latest/guide/credentials.html
[122] AWS (2024) AWS Lambda – Pricing
Available At: https://aws.amazon.com/lambda/pricing/
[123] Vercara (2025) Content-Type HTTP Header
Available At: https://vercara.digicert.com/resources/content-type-http-header
[124] AWS (2026) Common response headers - Amazon Simple Storage Service
Available At:
https://docs.aws.amazon.com/AmazonS3/latest/API/RESTCommonResponseHeaders.html
[125] Meudec, C. (2025) Sikraken Install, Development and User Guide

Available At:
https://docs.google.com/document/d/1uDLnlrFGWUNYyzsotZAZ_jFVrRSPEoixVMgw1UZZ0ug/edit?tab=t.0
[126] SoSy Lab (2026) c/ECA.set · main · SoSy-Lab / Benchmarking / SV-Benchmarks · GitLab
Available At:
https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/ECA.set?ref_type=heads
[127] AWS (2026) Pass an individual environment variable to an Amazon ECS container - Amazon Elastic Container Service
Available At: https://docs.aws.amazon.com/AmazonECS/latest/developerguide/taskdef-envfiles.html
[128] AWS (2024) Amazon Elastic File System (EFS) | Cloud File Storage
Available At: https://aws.amazon.com/efs/
[129] EverythingDevOps (2024) What Is Amazon Resource Name (ARN)?
Available At: https://www.everythingdevops.dev/blog/what-is-amazon-resource-name-arn
[130] AWS (2024) Amazon ECS task lifecycle - Amazon Elastic Container Service
Available At:
https://docs.aws.amazon.com/AmazonECS/latest/developerguide/task-lifecycle-explanation.html [131] AWS (2019) AWS Batch — Easy and Efficient Batch Computing Capabilities - AWS
Available At: https://aws.amazon.com/batch/
[132] AWS (2026) AWS Batch Pricing
Available At: https://aws.amazon.com/batch/pricing/
[133] AWS (2025) Tutorial: Create a managed compute environment using Amazon EC2 resources - AWS Batch
Available At:
https://docs.aws.amazon.com/batch/latest/userguide/create-compute-environment-managed-ec2.html
[134] AWS (2026) Spot Instances - Amazon Elastic Compute Cloud
Available At: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-spot-instances.html
[135] AWS (2019) Amazon EC2 Spot Instances Pricing
Available At: https://aws.amazon.com/ec2/spot/pricing/
[136] AWS (2026) Job definitions - AWS Batch
Available At: https://docs.aws.amazon.com/batch/latest/userguide/job_definitions.html
[137] AWS (2026) Jobs - AWS Batch
Available At: https://docs.aws.amazon.com/batch/latest/userguide/jobs.html
[138] AWS (2026) AWS Batch job environment variables - AWS Batch
Available At: https://docs.aws.amazon.com/batch/latest/userguide/job_env_vars.html
[139] AWS (2026) Array Jobs - AWS Batch
Available At: https://docs.aws.amazon.com/batch/latest/userguide/array_jobs.html
[140] AWS (2026) Job dependencies - AWS Batch
Available At: https://docs.aws.amazon.com/batch/latest/userguide/job_dependencies.html
[141] AWS (2026) Job queues - AWS Batch
Available At: https://docs.aws.amazon.com/batch/latest/userguide/job_queues.html
[142] AWS (2026) Compute environments for AWS Batch - AWS Batch
Available At: https://docs.aws.amazon.com/batch/latest/userguide/compute_environments.html
[143] AWS (2026) AWS Batch FAQs
Available At: https://aws.amazon.com/batch/faqs/
[144] Awad, J W. (2025) AWS ECS Deep Dive
Available At: https://joudwawad.medium.com/aws-ecs-deep-dive-c8f773af0bf6
[145] AWS (2023) Amazon EC2 container instances for Amazon ECS - Amazon Elastic Container Service
Available At: https://docs.aws.amazon.com/AmazonECS/latest/developerguide/create-capacity.html
[146] AWS (2026) Job states - AWS Batch
Available At: https://docs.aws.amazon.com/batch/latest/userguide/job_states.html
[147] HashiCorp (2024) Create infrastructure | Terraform | HashiCorp Developer

Available At: https://developer.hashicorp.com/terraform/tutorials/aws-get-started/aws-create
[148] AWS (2026) s3 — AWS CLI 1.31.10 Command Reference
Available At: https://docs.aws.amazon.com/cli/latest/reference/s3/
[149] HashiCorp (2025) Resources Overview - Configuration Language | Terraform | HashiCorp Developer
Available At: https://developer.hashicorp.com/terraform/language/resources
[150] HashiCorp (2025) Providers - Configuration Language | Terraform | HashiCorp Developer
Available At:
https://developer.hashicorp.com/terraform/language/providers
[151] HashiCorp (2025) Modules Overview - Configuration Language | Terraform | HashiCorp Developer
Available At: https://developer.hashicorp.com/terraform/language/modules
[152] AWS (2025) What Is IAM? - AWS Identity and Access Management
Available At: https://docs.aws.amazon.com/IAM/latest/UserGuide/introduction.html
[153] AWS (2019) Identities (Users, Groups, and Roles) - AWS Identity and Access Management
Available At: https://docs.aws.amazon.com/IAM/latest/UserGuide/id.html
[154] AWS (2026) IAM roles - AWS Identity and Access Management
Available At: https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles.html#iam-term-service-role
[155] AWS (2026) submit-job — AWS CLI 2.34.5 Command Reference
Available At: https://docs.aws.amazon.com/cli/latest/reference/batch/submit-job.html
[156] Terraform Registry (2026) AWS Batch Terraform module
Available At: https://registry.terraform.io/modules/terraform-aws-modules/batch/aws/latest
[157] Terraform Registry (2026) Amazon ECR Terraform module
Available At: https://registry.terraform.io/modules/terraform-aws-modules/ecr/aws/latest
[158] Terraform Registry (2026) Amazon Lambda Terraform module
Available At: https://registry.terraform.io/modules/terraform-aws-modules/lambda/aws/latest
[159] Terraform Registry (2026) Amazon S3 bucket Terraform module
Available At: https://registry.terraform.io/modules/terraform-aws-modules/s3-bucket/aws/latest
[160] Terraform Registry (2026) Resource: aws_iam_role
Available At: https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/iam_role
[161] Terraform Registry (2026) Resource: aws_iam_role_policy
Available At:
https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/iam_role_policy
[162] Terraform Registry (2026) AWS IAM Terraform module
Available At: https://registry.terraform.io/modules/terraform-aws-modules/iam/aws/latest
[163] AWS (2019) describe-jobs — AWS CLI 2.34.21 Command Reference
Available At: https://docs.aws.amazon.com/cli/latest/reference/batch/describe-jobs.html
[164] AWS (2026) Amazon S3 objects overview - Amazon Simple Storage Service
Available At: https://docs.aws.amazon.com/AmazonS3/latest/userguide/UsingObjects.html
[165] Ubuntu (2026) Install a root CA certificate in the trust store
Available At:
https://ubuntu.com/server/docs/how-to/security/install-a-root-ca-certificate-in-the-trust-store/
[166] askUbuntu (2016) What is the use/purpose of the ca-certificates package?
Available At:
https://askubuntu.com/questions/857476/what-is-the-use-purpose-of-the-ca-certificates-package
[167] AWS (2026) Working with Lambda layers - AWS Lambda
Available At: https://docs.aws.amazon.com/lambda/latest/dg/chapter-layers.html
[168] AWS (2024) Amazon CloudWatch - Application and Infrastructure Monitoring
Available At: https://aws.amazon.com/cloudwatch/