AWS-Syndicate: Quick Start

User Guide

September 2018

SDCT-01

Version 1.0

CONTENT

1 OVERVIEW

AWS-Syndicate (further - Syndicate) is an Amazon Web Services deployment framework written in Python, which allows to easily deploy serverless applications to using resource descriptions. The framework allows to work with applications that engage the following AWS services:

1. API Gateway
2. CloudWatch
3. Cognito
4. DynamoDB
5. Elastic Beanstalk
6. Elastic Compute Cloud (EC2)
7. Identity and Access Management (IAM)
8. Kinesis
9. Lambda
10. Simple Notification Service (SNS)
11. Simple Queue Service (SQS)
12. Simple Storage Service (S3)
13. Step Functions

2 QUICK START

2.1 PRE-REQUISITES

To successfully setup and use the Syndicate, you need the following software to be installed:

- Python 2.7
- pip 9.0+
- Apache Maven 3.3.9

To begin, please clone the Syndicate repository from Github.

Please note: you can find example contents of the files described in the document in this folder.

2.2 INSTALLING THE FRAMEWORK

1. Run the following command in the Syndicate repository root folder to install the framework:

   ```
   pip install.
   ```

2. Set up the Syndicate Java plugin

   ```
   mvn install
   ```

3. Set up the **sdct.conf** deployment configuration file, as described in the section below.

2.3 DEPLOYMENT CONFIGURATION FILE (SDCT.CONF)

The **sdct.cong** file includes the deployment configuration needed for the Syndicate to perform on your AWS account.

*The path to the folder with the **sdct.conf** file should be specified in the **SDCT_CONF** environment variable.*

You can find the example file in the AWS Syndicate GitHub repository.

2.3.1 Parameters List

The configuration is described in the following set of parameters (the required ones are marked with an **asterisk ***):

- **project_path*** – the path to the project folder containing files with the .json descriptions of AWS resources to be deployed.
- **account_id*** – the ID of the account where you want to deploy AWS resources.
- **aws_access_key_id*** – AWS access key ID to be used to deploy AWS resources.
- **aws_secret_access_key*** – AWS secret access key to be used to deploy AWS resources.
- **access_role** – an existing IAM role in the AWS account to be used during deployment. If the role is specified, deployment framework will assume it for further usage. If the access role it is not specified – the user whose access key and secret key are given, should have permissions to deploy resources.
- **region*** – the region to deploy resources. Use native AWS names. For example, 'eu-central-1'.
- **deploy_target_bucket*** – the name of the S3 bucket to which the deployment framework uploads a bundle. After that, the bundle is used for deployment process. The parameter should specify a bucket name from the account in which you want to deploy resources.
- **resources_suffix** (max 5 characters) – the additional name suffix for resource to be deployed. For example, you want to deploy an S3 bucket in different regions. The bucket name must be unique for all AWS accounts, so you need to deploy it with different names. In this case you must use different suffixes for the AWS resource (S3 bucket) .json description to be deployed. The suffix value will be added to AWS unique (S3 buckets) and region-independent (IAM roles, policies, users etc.) resources. For example, if you named an S3 bucket "notification-metadata" and use '-fr' as a suffix, the final name will be "notification-metadata-fr".
- **resources_prefix** (max 5 characters) – the additional name prefix for resource to be deployed. For example, you want to deploy an S3 bucket to different regions. The bucket name must be unique for all AWS accounts, so you need to deploy it with different names. In this case you must use different prefixes for the AWS resource (S3 bucket) .json description to be deployed. The suffix value will be added to AWS unique (S3 buckets) and region-independent (IAM roles, policies, users etc.) resources. For example, if you named an S3 bucket "notification-metadata" and use 'sndc-' as a prefix, the final name will be "sndc-notification-metadata".

Please note that you can define both prefix and suffix. In this case, the final resource name will be defined using the next formula (strings concatenation):
Resource name to be deployed = prefix + resource name in description + suffix

- **build_projects_mapping** – if you have AWS Lambda resources to be deployed to the AWS account, you can use the framework to build Lambdas. It can build a deployment package automatically for Python and Java languages. For each language that framework can automatically pack and after use in deployment we have a list of rules and requirements to follow.
  The parameter format: ***Build tool:path to the project*** (the relative path from the **project_path** property)

  Example:

  ```
  mvn:/java-demo;python:/python-demo
  Supported build tools: mvn, python
  ```

  Please note that there are specific requirements your Lambdas should meet, so that they can be processed by the framework. You can find the details on these requirements in this section.

2.3.2 Access Settings

The Syndicate accesses AWS using the IAM user credentials provided in the **sdct.conf** file, in one of the following ways:

- With **AWS access and secret keys** belonging to a user that has the necessary permissions level. It is recommended to use the credentials of the user with admin permissions, to enable the full-scope performance of the framework. However, in case such approach cannot be used (for security reasons, for example), you can use a policy with minimum necessary permissions (see the policy details here).

- With an **access role** which will be assumed by the framework. In such case, the access and secret keys can belong to a user with either admin permissions or a simple policy where the only allowed action is "**sts:AssumeRole**" (see the policy details here).
  The role should have either admin permissions (recommended), or the policy with minimum necessary permissions (see the policy details here).
  If the role is specified, the deployment framework will assume the role at each command call, get temporary credentials for an hour and use them.
  .

2.3.3 Configuration Example

Below, you can find an example of the **sdct.conf** configuration file:

```
# base path of the projects (python/java)
project_path=/var/lib/jenkins-slave/workspace/Dev_Deploy_resources_new

resources_suffix=-eoos
resources_prefix=sndc-

# base config data
region=eu-central-1
deploy_target_bucket=artifactory-bucket
account_id=123456789123
```

```
access_role=MyDeploymentRole
aws_access_key_id=TYPE_YOUR_ACCESS_KEY_ID
aws_secret_access_key= TYPE_YOUR_SECRET_ACCESS_KEY

# build configuration
build_projects_mapping=mvn:/java-demo;python:/python-demo
```

2.4 SYNDICATE OPERATION FILES

The deployment framework detects **.json** files in the folder containing AWS resource meta descriptions. Each AWS resource has an individual resource description (mapping and examples will be provided below).

There are two types of operation files containing the infrastructure descriptions, which the developer should create to enable correct deployment:

1. **deployment_resources.json** - The file includes the name of the resources to be deployed in AWS and their configuration. The details on how each of the resources should be described are given in this section. You can also find the complete file example here.
   There can be any number of such files in the project.
2. **<filename_beginning>_lambda_config.json** - The file containing a description of a specific Lambda. There should be such a file for each Lambda written in Python (not needed for Java-based ones).
   You can find the example here.

During the deployment (within the **build_bundle** command execution**)**, Syndicate combines all the artifacts (.var, .jar, .zip files, etc.) into a bundle, and creates the main bundle file - **build_meta.json.** The file lists all AWS resources described in the detected operation files.

3 LAMBDA REQUIREMENTS FOR AUTOMATIC ARTICFACTS BUILD

Below you can find the requirements Lambda functions should meet so that they can be used for automatic artifacts build.

3.1 PYTHON LAMBDAS

```
lambda_folder_name

   |---lambda_handler.py

   |--- lambda_config.json

   |---requirements.txt

   |---local_requirements.txt

   |---deployment_resources.json
```

Structure requirements:

1. Lambda file should include a function with any name **<lambda_func>(event, context)**. The function will be an entry point to the Lambda.
2. The root folder where the Lambda is stored should have a file named **lambda_config.json** which describes Lambda configuration in a specific format and lists all its dependencies and triggers.
3. Lambda module can include the **requirements.txt** file which lists external libraries, on which the Lambda code depends. If the file is added, all listed libraries will be added to the Lambda **.zip**. Otherwise, it is considered that the Lambda function does not have any external dependencies.
4. Lambda module can include the **local_requirements.txt** file which lists all modules from the project repository (paths to the modules), on which the Lambda code depends. If the file is there, all the listed modules will be added to the Lambda **.zip**. Otherwise, it is considered that the Lambda does not need any neighbor modules.

3.2 JAVA LAMBDAS

To enable convenient development of java-based Lambda functions, an additional Maven plugin was created. It generates a Lambda meta description file during the java application build. The plugin includes annotations, which are used as a basis for creation of Lambda meta descriptions.

To enable automatic meta description generation, the Lambdas comprising the Java application should meet the following requirements:

1. Lambda handler must be marked with the **@LambdaHandler** annotation
2. The **pom.xml** Lambda module must include the following plugin:

```
<plugin>
               <groupId>com.aws.syndicate</groupId>
               <artifactId>deployment-configuration-maven-plugin</artifactId>
                <version>1.04</version>
                <configuration>
                    <packages>
                        <package>com.aws.syndicate.demo</package>
                    </packages>
                </configuration>
                <executions>
                    <execution>
                        <id>generate-config</id>
                        <phase>compile</phase>
                        <inherited>false</inherited>
                        <goals>
                            <goal>gen-deployment-config</goal>
                        </goals>
                    </execution>
                </executions>
     </plugin>
```

The **Dependencies** section should also include the annotations module:

```
<dependency>
    <groupId>com.aws.syndicate</groupId>
    <artifactId>deployment-configuration-annotations</artifactId>
    <version>1.02</version>
</dependency>
```

The plugin configuration should include packages in which the plugin should look for Lambdas (in the example above, this is the **com.aws.syndicate.demo** package).

On project build, the meta description file named **deployment_resources.json** is generated in the target folder. It is further processed and used by the framework to deploy the Lambda resources to the account.

Annotations also allow to reference other resources mentioned in meta descriptions. There are different annotations that allow to specify the configuration of Lambda functions.

The following annotations can be used (the required parameters are marked with an asterisk *):

1. **@LambdaHandler** – the anotation which marks the class as Lambda handler. The parameters are:
   a. **lambdaName*** – the name that is used on Lambda deployment
   b. **roleName*** - the name of the role under which the Lambda is executed.
   c. **methodName –** the name of the method within the current class that is used as a handler.
   d. **timeout –** the lambda function timeout (default value – 300 seconds)
   e. **memory -** the memory allocated to Lambda (default value – 1024 MB)
   f. **regionScope -** the region/regions to where the Lambda is deployed (the default value – RegionScope.DEFAULT means that the Lambda is deployed only to the region specified in the **sdct.conf** file during the deploy)
   g. **subnets** – the list of IDs of subnets where the Lambda is be placed to (default value – an empty list.)

   h. **SecurityGroupsIds** – the list of the IDs of security groups to which the Lambda belongs (default value – an empty list)
   i. **tracingMode –** turns on the X-Ray for the Lambda.

2. **@EnvironmentVariable** - the annotation that allows to set environment variables in the Lambda configuration:
   a. **key*** - the environment variable key value.
   b. **value*** - the environment variable value.

   @EnvironmentVariables - the repeatable annotation

3. **@DynamoDbTriggerSource** - the annotation which allows to subscribe a Lambda to a DynamoDB stream:
   a. **TargetTable* -** the name of the table to which the Lambda subscribed as a trigger.
   b. **batchSize*** - the number of records passed to the Lambda as input.

   @DynamoDbTriggerSource - the repeatable annotation

4. **@RuleEventSource** - the annotation which allows to subscribe a Lambda to a CloudWatch rule.
   a. **TargetRule** – the name of the rule to which the Lambda is subscribed.

   @RuleEvents - a repeatable annotation

5. **@S3EventSource** - the annotation that allows to subscribe a Lambda to events in an S3 bucket
   a. **targetBucket* -** the name of the bucket to which the Lambda is subscribed.
   b. **events *** - the array of line-events that trigger the Lambda (for example, "**s3:ObjectRemoved*"**

   @S3EventSource - the repeatable annotation

6. **@SnsEventSource** - the annotation which allows to subscribe a Lambda to an SNS topic:
   a. **targetTopic*** - the name of an SNS topic to which the Lambda function is subscribed.
   b. **regionScope** – the region or regions in which the SNS topic is located. The default value – RegionScope.DEFAULT means that the Lambda is deployed only to the SNS topic in the regions specified in the config during deploy.

   @SnsEvents - the repeatable annotation.

7. **@SqsTriggerEventSource** - the annotation that allows to subscribe the Lambda to an SQS queue:
   a. **targetQueue*** – the name of the SQS queue to which the Lambda is subscribed.
   b. **batchSize*** – the number of records passed used at the Lambda event.

   @SqsEvents - the repeatable annotation.

8. **@LambdaConsurrency** - the annotation that limits the maximum number of Lambda executions within a specified time unit.
   a. **executions*** – the maximum number of Lambda function executions within the specified time unit.
9. **@DeadLetterConfiguration** - the annotation that allows to configure DLQ for the Lambda.
   a. **resourceType*** – the DLQ type, SQS queue or an SNS topic
   b. **resourceName*** – the DLQ resource name.

10. **@DependsOn** - the annotation that allows to establish dependencies between resources that are to be deployed (for example, if a Lambda depends on a table, to make it easier for developers to orient in the dependencies, we can specify this table in the **@DependsOn** annotation):
    a. **name*** – the name of the resource in the meta description
    b. **resourceType*** - the type of the resource.

@Dependencies - the repeatable annotation.

After the plugin is run within the specified phase (for example – **compile**), the **deployment_resources.json** file with Lambda functions meta descriptions is generated in the target folder.

4 RESOURCES META DESCRIPTIONS

4.1 LAMBDA

“resource_type”: “lambda”

```
{
  "version": "lambda_version",       * Lambda version. Is used at artifact
                                       build. Required for Python
  "name": "lambda_name",             * Lambda name. Required for Python
  "lambda_path": ""                  * The relative path to the Lambda handler
                                       within the Python project. Required for
                                       Python
  "func_name": "lambda_handler",     * Function handler name. Required.
  "resource_type": "lambda",         - Resource type.
  "iam_arn_role": "role_name",       * Role name. Required
  "runtime": "nodejs|nodejs4.3|      * Lambda executive environment. Required.
  nodejs6.10|nodejs8.10|java8|
  python2.7|python3.6|
  dotnetcore1.0|dotnetcore2.0|
  dotnetcore2.1|nodejs4.3-edge|
  go1.x",
  "memory": 128,                      * Lambda memory. Required.
  "timeout": 300,                     * The function execution time at which
                                        Lambda should terminate the function.
                                        Required.
  "deployment_package":               * Artifact name. Required for Java.
  "package_name.jar",
  "concurrent_executions": 1,         - Maximum lambda quantity per unit time.
  "dependencies": [                   - Lambda dependencies.

{
      "resource_name": "table_name",
      "resource_type": "dynamodb_table"
    },
    {
      "resource_name": "bucket_name",
      "resource_type": "s3_bucket"
    },
    {
      "resource_name": "rule_name",
      "resource_type": "cloudwatch_rule"
    },
    {
      "resource_name": "topic_name",
      "resource_type": "sns_topic"
    },
    {
      "resource_name": "stream_name",
      "resource_type": "kinesis_stream"
    }
    ...
  ],
  "event_sources": [                 - Lambda subscriptions.
    {
      "target_table": "table_name",  - The name of the table name, to which
                                       the Lambda is subscribed
      "resource_type":               - Resource type.
      "dynamodb_trigger",
      "batch_size": 1                - Entry quantity, processed during one
                                       Lambda call.
    },
    {
      "target_rule": "rule_name",    - CloudWatch rule name.
      "resource_type":
      "cloudwatch_rule_trigger"       - Resource type.
    },
    {
      "target_bucket": "bucket_name", - S3 bucket name.
      "resource_type": "s3_trigger",  - Resource type.
      "s3_events":                    - The list of the events, to which
                                        the Lambda listens.
      ["s3:ReducedRedundancyLostObject'|
      s3:ObjectCreated:*|
      s3:ObjectCreated:Put|
      s3:ObjectCreated:Post|
      s3:ObjectCreated:Copy|
      s3:ObjectCreated:CompleteMultipartUpload|
      s3:ObjectRemoved:*|
      s3:ObjectRemoved:Delete|
      s3:ObjectRemoved:DeleteMarkerCreated"]
    },
    {
      "target_topic": "topic_name",    - The name of the SNS topic to which
                                         the Lambda is subscribed
      "resource_type":                 - Resource type.
      "sns_topic_trigger",
      "region": /"all"/"region_name"/ - The name of the region in which the
                                        topic is  deployed.
      ["region_name1", ..]
    },
    {
      "resource_type" :                 - Resource type.
      "kinesis_trigger",
      "target_stream" :                 - Kinesis stream name.
      "stream_name",
      "batch_size" : 100,               - The quantity of the entries processed
                                          in one Lambda call.
      "starting_position" : "LATEST"   - The position at which the entry
      (allowed: TRIM_HORIZON,             processing starts.
       AT_TIMESTAMP, LATEST)
    },
    {
      "resource_type" : "sqs_trigger", - Resource type.
      "target_queue" : "queue_name",   - SQS queue name.
      "batch_size" : 100,              -  The quantity of entries, processed
                                          during one Lambda call
    }

    ...
  ],
  "env_variables": {                   - Environment variables (key-value
                                          mapping).
    "var_name": "var_value"
  }
}
```

Here we have Lambda description with event sources. The trigger field is "**target_...**".

This target resource must be specified in "**dependencies**" and the full resource configuration must be described in **deployment_resources.json**.

Examples:

```
"put_dynamodb_item": {
        "name": "put_dynamodb_item",
        "file_name": "handler.py",
        "memory": 128,
        "env_variables": {
            "region": "${region}"
        },
        "iam_role_name": "PutItemToDynamoRole",
        "lambda_path": "",
        "version": "1.0",
        "timeout": 300,
        "func_name": "handler.lambda_handler",
        "dependencies": [
            {
                "resource_name": "Notifications",
                "resource_type": "dynamodb_table"
            }
        ],
        "runtime": "python2.7",
        "resource_type": "lambda"
    }

"dynamodb_item_processor": {
        "security_group_ids": [],
        "name": "dynamodb_item_processor",
        "event_sources": [
            {
                "batch_size": 1,
                "target_table": "Notifications",
                "resource_type": "dynamodb_trigger"
            }
        ],
        "func_name": "com.aws.syndicate.demo.PutFileToS3BucketHandler",
        "env_variables": {
            "notification_bucket": "${notification_bucket}",
            "region": "${region}"
        },
        "iam_role_name": "PutObjectToS3Role",
        "lambda_path": "D:\\test-project\\demo-java",
        "version": "1.0-SNAPSHOT",
        "timeout": 300,
        "memory": 1024,
        "dependencies": [
            {
                "resource_name": "Notifications",
                "resource_type": "dynamodb_table"
            }
        ],
        "deployment_package": "syndicate-demo-1.0-SNAPSHOT.jar",
        "runtime": "java8",
        "subnet_ids": [],
        "resource_type": "lambda"
    }
```

4.2 DYNAMO DB TABLE

“resource_type”: “dynamodb_table”

```
"table_name": {
    "resource_type": "dynamodb_table",  * Table name. Required.
    "hash_key_name": "hash_name",       * Table hash key. Required.
    "hash_key_type": "S/N/B",           * Hash key type. Required.
    "sort_key_name": "sort_name",       - Table sort key. If not specified,
                                          the table will have only a hash
                                          key.
    "sort_key_type": "S/N/B",           - Sort key type. Required if sort key
                                          name is specified.
    "read_capacity": 25,                - The maximum number of strongly
                                          consistent reads that can be
                                          performed per second. If not
                                          specified, sets the default value
                                          to 1.
    "write_capacity": 25                - The maximum number of writing
                                          processes consumed per second. If
                                          not specified, sets the default
                                          value to 1.
    "global_indexes": [                 - Table indexes. May contain several
                                          objects.
      {
        "name": "m-index",              * Index name. Required.
        "index_key_name": "m",          * Index hash key. Required
        "index_key_type": "S",          * Hash key index type. Required
        "index_sort_key_name": "im",    - Index sort key.
        "index_sort_key_type": "S"      - Sort key type. Required if sort
                                          name is specified.
      }
    ],
“autoscaling”: [                        - Table autoscaling configuration
{
             "resource_name": "table", * Resource name. Required.
             "min_capacity": 1,        * Minimum capacity level. Required.
             "max_capacity": 100,      * Maximum capacity level. Required.
             "role_name": "autoscaling_db", * The name of the role, which
                                              performs autoscaling.
                                              Required.
             "config": {                    * Unit configuration. Required.
                 "target_utilization": 75,  * target utilization in
                                              autoscaling. Required.
                 "policy_name": "Roles_rcu" - Autoscaling policy name.
                },
                "dimension": "dynamodb:table:ReadCapacityUnits" - Autoscaling
                                                                  dimension
            }]
```

Here we have a Dynamo DB table description. "**sort_key_name**" and "**sort_key_type**" are not required because a table can be created without a sort key definition.

Example:

```
"Roles": {
        "read_capacity": 5,
        "hash_key_name": "n",
        "autoscaling": [
            {
                "resource_name": "Roles",
                "min_capacity": 1,
                "max_capacity": 100,
                "role_name": "autoscaling_db",
                "config": {
                    "target_utilization": 75,
                    "policy_name": "Roles_rcu"
                },
                "dimension": "dynamodb:table:ReadCapacityUnits"
            }
        ],
        "write_capacity": 1,
        "resource_type": "dynamodb_table",
        "hash_key_type": "S"
    }
```

4.3 DYNAMO DB STREAM

“resource_type”: “dynamodb_stream”

```
"table_stream": {
    "resource_type": "dynamodb_stream",    * Resource type. Required
    "table": "table_name",                 * The name of the table, for which
the                                          the stream is enabled. Required.
          "stream_view_type":              - Stream type. If not specified,
    "NEW_AND_OLD_IMAGES/KEYS_ONLY/           the default value is set to
      NEW_IMAGE/OLD_IMAGE"                   NEW_AND_OLD_IMAGES.
  }
```

Example:

```
"RunInstancesStream": {
        "table": "RunInstancesEvents",
        "resource_type": "dynamodb_stream",
        "stream_view_type": "NEW_IMAGE"    }
```

4.4 CLOUDWATCH EVENT RULE

"resource_type": "cloudwatch_rule"

Rule types:

- schedule

```
"rule_name": {
    "resource_type": "cloudwatch_rule",   * Resource type. Required.
    "rule_type": "schedule",              * Rule type. Required.
    "expression": "rate(1 hour)"          * Rule expression (cron schedule).
                                            Required.
    "region": /"all"/"region_name"/       - The region where the rule is
                                            deployed. If not specified
    ["region_name1", ..]                    the default value is taken from
                                            sdct.conf.
  }
```

- ec2

```
"rule_name": {
    "resource_type": "cloudwatch_rule",   * Resource type. Required.
    "rule_type": "ec2",                   * Rule type. Required.
    "instance_ids": ['id-1111', ...],     - The list of EC2 instances, to
                                            which the rule is bound.
                                            If not specified, the default
                                            value is set to 'any'.
    "instance_states": [...]              - EC2 instance states, which are
                                            monitored by the rule. If not
                                            specified, the default value is
                                            set to 'any'.
    "region": /"all"/"region_name"/       - The region, where the rule is
    ["region_name1", ..]                    deployed. If not specified, the
                                            default value is taken from
                                            sdct.conf.
  }
```

- api call

```

"rule_name": {
    "resource_type": "cloudwatch_rule",    * Resource type. Required.
    "rule_type": "api_call",               * Rule type. Required.
    "aws_service": "aws_service_name"",    * The name of the AWS service,
                                             which the rule listens to.
    "operations": [...]                     - The actions monitored by rule.
                                              If not specified, the default
                                              value is set to 'any'.
    "region": /"all"/"region_name"/
    ["region_name1", ..]                   - The region where the rule is
    ["region_name1", ..]                     deployed. If not specified,
                                             the default value is set from
                                             sdct.conf.
  }
```

Example:

```
"weekly_report_event": {
        "rule_type": "schedule",
        "expression": "cron(0 8 ? * MON *)",
        "resource_type": "cloudwatch_rule"
    }
```

4.5 S3 BUCKET

“resource_type”: “s3_bucket”

```
"bucket_name": {
    "resource_type": "s3_bucket",          * Resource type.
    "location": "eu-west-1/us-west-1/      - The region, where the bucket is
    us-west-2/ap-south-1/ap-southeast-1/     created, the default value is the
    ap-southeast-2/ap-northeast-1/           region set in sdct.conf
    sa-east-1/cn-north-1/eu-central-1"
    "acl": "private|public-read|           - The canned ACL to be applied to
    public-read-write|authenticated-read"    the bucket.
"policy": {                                - IAM policy to be attached to the
            "Version": "2008-10-17",         bucket.
            "Id": "PolicyForCloudFrontPrivateContent",
            "Statement": [
                {
                    "Action": "s3:GetObject",
                    "Principal": "*",
                    "Resource": "arn:aws:s3:::${ui_bucket}/**",
                    "Effect": "Allow",
                    "Sid": "1"
                }
            ]
        }
  }
```

Example:

```
"${ui_bucket}": {
        "policy": {
            "Version": "2008-10-17",
            "Id": "PolicyForCloudFrontPrivateContent",
            "Statement": [
                {
                    "Action": "s3:GetObject",
                    "Principal": "*",
                    "Resource": "arn:aws:s3:::${ui_bucket}/**",
                    "Effect": "Allow",
                    "Sid": "1"
                }
            ]
        },
        "resource_type": "s3_bucket",
        "acl": "public-read"
    }
```

4.6 API GATEWAY

"resource_type": "api_gateway"

```
"api_name": {
"deploy_stage": "dev",                   * The stage of the deployed API.
                                           Required
    "resource_type": "api_gateway",      * Required.
    "dependencies": [                    - Not required.
        {
            "resource_name": "lambda_name",
            "resource_type": "lambda"
        }
        ...
    ],
    "resources": {                       * Required.
        "/path": {
      "enable_cors": true,               - Enables CORS on the resource
                                           methods.
            "POST|GET|DELETE|PUT|HEAD|
            PATCH|ANY":{
                "authorization_type":    - The method's authorization type
                  " AWS_IAM|CUSTOM|        (sets the default value to
                  COGNITO_USER_POOLS",     'NONE').
                "api_key_required":      - Specifies whether the method
                        true|false         required a valid ApiKey
                                           (the default value is set to
                                           'false').

                "method_request_parameters": { A key-value map defining
                                         required or optional method request
                                         parameters that can be accepted by
                                         API Gateway. A key defines a method
                                         request parameter name matching the
                                         pattern
                                         method.request.{location}.{name},
                                         where location is query string,
                                         path, or header and name is a valid
                                         and unique parameter name. The
                                         value associated with the key is a
                                         Boolean flag indicating whether the
                                         parameter is required (true) or
                                         optional (false)- not required (is
                                         not set).
            "method.request.querystring.param_name": true|false
                    ...
                }
                "method_request_models":{ Specifies the Model resources
                                          used for the request's content type
                                          - not required (is not set)
                    "string": "string"
                    ...
                },
                "integration_type":       * The resource to which the method
                "lambda|service|mock|http", is connected. Required
                "lambda_name": "name",    * Lambda name. Required if
                                            integration type is lambda
      "enable_proxy": true|false          - Present if only integration_type
                                            is Lambda
                "integration_request_body_template": { - Represents a map of
                                            Velocity templates that are
                                            applied on the request payload
                                            based on the of the Content-Type
                                            header sent by the client (is not
                                            set).
                    "application/json":  "..",
                }
                "integration_passthrough_behavior": - Specifies how the
                                           method request "WHEN_NO_MATCH|
                                           WHEN_NO_TEMPLATES| NEVER" body of
                                           an unmapped content type is passed
                                           through the  integration request
                                           to the back end without any
                                           transformation. (The default value
                                           is set to 'WHEN_NO_MATCH')
                "lambda_region":          lambda can be located in different
"one_of_the_aws_region",                 * The Region, which value you can
                                         override from m3config.conf.
                                         Required (if Lambda is not in the
                                         same region as API).
                "responses": [           - Method responses (sets default
                                           response with '200' status code)
                    {
                        "status_code": "status_code"
                        "response_parameters": {
                            "string": "string",
                            ...
                        }
                        "response_models": {
                            "string": "string"
                            ...
                        }
                    },
                                                                        ...
                                                         - There can be several responses
                ]
                integration_responses: [ - Integration method responses
                                          (sets the default response
                                           with '200' status code and
                                           without Lambda regex).
                    {
                        "status_code":     * Required.
                        "status_code",     - Not required.
                        "lambda_error_regex":
                         "..",
                        "response_parameters": { - Not required.
                            "string": "string",
                            ...
                        }
                        "response_templates":{ - Not required.
                            "string": "string",
                            ...
                        }
                    },
                    ...
                                         - There can be several
                                          integration responses.

                ],
                "default_error_pattern": - Not required (if you did not
                  true  -                 specify integration_responses and
                                          responses, you can choose
            }                             default).
        }
    }
}
```

Here we have an API Gateway description. This resource can be described in different **deployment_resources.json** files, part of API can be in one file, and another part - in another file. The '**resources**' field can include not limited amount of resource paths.

Example:

```
"syndicate-demo-api": {
    "deploy_stage": "prod",
    "dependencies": [
      {
        "resource_name": "put_dynamodb_item",
        "resource_type": "lambda"
      }
    ],
    "resources": {
      "/notications": {
        "enable_cors": true,
        "POST": {
          "integration_request_body_template": {},
          "authorization_type": "AWS_IAM",
          "integration_type": "lambda",
          "method_request_parameters": {},
          "default_error_pattern": true,
          "integration_passthrough_behavior": "WHEN_NO_TEMPLATES",
          "lambda_name": "put_dynamodb_item"
        }
      }
    },
    "resource_type": "api_gateway"
  }
```

4.7 SNS TOPIC

"resource_type": "sns_topic"

```
"topic_name": {
    "resource_type": "sns_topic"       * Resource type. Required.
    "region": /"all"/"region_name"/    * Region name, where it should be
                                         deployed. Required.
      ["region_name1", ..]             - SNS topic (the default value is
                                         set from sdct.conf).
    "event_sources": [                 - SNS topic subscriptions
             {                           configuration.
            "target_rule": "rule_name",
            "resource_type": "cloudwatch_rule_trigger"
        }
    ]
}
```

Example:

```
"stackAuditTopic": {
        "region": "all",
        "resource_type": "sns_topic"
    }
```

4.8 CLOUDWATCH ALARM

"resource_type": "cloudwatch_alarm"

```
"alarm_name": {
    "metric_name": "name",              * The metric name. Required.
    "resource_type": "cloudwatch_alarm" * Resource type. Required.
    "namespace": "namespace",           * The namespace for the metric
                                          associated with the alarm.
                                          Required.
    "period": 1200, (sec)               * The period, in seconds, over which
                                          the specified statistic is applied.
                                          Valid values are 10, 30, and any
                                          multiple of 60. Required.
    "evaluation_periods": 1,            * A number of periods over which
                                          data is compared to the specified
                                          threshold. Required.
    "threshold": 1.0,                   * The value to compare with the
                                          specified statistic. Required.
    "comparison_operator":              * An arithmetic operation to use when
    "GreaterThanOrEqualToThreshold|       comparing the specified statistic
    GreaterThanThreshold|                 and threshold. The specified
    LessThanThreshold|                    statistic value is used as the
    LessThanOrEqualToThreshold",          first operand. Required.
    "statistic": "SampleCount|          * The statistic for the metric
    Average|Sum|Minimum|Maximum",         associated with the alarm, other
                                          than percentile. For percentile
                                          statistics, use ExtendedStatistic.
                                          Required.
    "sns_topics": ['topic_name']        - The actions to execute when this
                                          alarm transitions to an OK state
                                          from any other state. Each action
                                          is specified as a name of SNS
                                          topics.

}
```

Example:

```
"alarm_name": {
    "metric_name": "db_alarm",
    "resource_type": "cloudwatch_alarm"
    "namespace": "db",
    "period": 1200,
    "evaluation_periods": 1
    "threshold": 1.0,
    "comparison_operator": "GreaterThanOrEqualToThreshold ",
    "statistic": "SampleCount ",
    "sns_topics": [“audit_topic»]
}
```

4.9 KINESIS STREAM

"resource_type": "kinesis_stream"

```
"stream_name": {
    "resource_type": "kinesis_stream", * Resource type. Required.
    "shard_count": 2                   * Number of shards that the stream
                                         uses. Required.
  }
```

Example:

```
"audit_stream": {
"resource_type": "kinesis_stream",
"shard_count": 1
}
```

4.10 IAM POLICY

"resource_type": "iam_policy"

```
“policy_name”: {
        "resource_type": "iam_policy", * Resource type. Required.
        "policy_content": {             * IAM policy content. Required.
            …
        }
    }
```

Example:

```
"AutoscalingDynamoRead": {
        "resource_type": "iam_policy",
        "policy_content": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Action": [
                        "dynamodb:DescribeTable",
                        "cloudwatch:GetMetricStatistics",
                        "cloudwatch:DescribeAlarms"
                    ],
                    "Resource": "*",
                    "Effect": "Allow"
                }
            ]
        }
    }
```

4.11 IAM ROLE

"resource_type": "iam_role"

```
"role_name": {
        "predefined_policies": [           - Managed IAM policies list.
            "policy_name"

        ],
        "principal_service": "lambda",    - Service which uses the role.

        "custom_policies": [              - Customer AWS policies names.
            "LambdaBasicExecution",
            "S3Read",
            "SNSWrite",
            "CloudFormationResourceCreationPolicyWrite"
        ],
        "resource_type": "iam_role",       * Resource type. Required.
        "allowed_accounts": [              - The list of accounts, which can
                                             assume the role.
            "123456789123"
        ],
"external_id": "your_id",                  - External ID in role.
"instance_profile": true|false,            - If true, instance profile with
                                             role name is created.
"trusted_relationships": {}                - The .json-file of the trusted
                                             relationships to be attached.
    }
```

Example:

```
"lr_run_terraform_template": {
        "predefined_policies": [
            "AmazonSQSFullAccess"
        ],
        "principal_service": "lambda",
        "custom_policies": [
            "LambdaBasicExecution",
            "S3Read",
            "SNSWrite",
            "CloudFormationResourceCreationPolicyWrite"
        ],
        "resource_type": "iam_role",
        "allowed_accounts": [
            "${account_id}"
        ]
    }
```

4.12 STEP FUNCTIONS ACTIVITY

"resource_type": "state_activity"

```
"activity_name": {
    "resource_type": "state_activity", * Resource type. Required.
  }
```

Example:

```
"approval_activity": {
        "resource_type": "state_activity"
    }
```

4.13 SQS

"resource_type": "sqs_queue"

```
"audit-queue-name": {
    "region": "eu-west-1",            - The region, where the queue is
                                        deployed (the default value is the
                                        region from sdct.conf).
    "fifo_queue": true|false,         - If true, the queue is FIFO (the default
                                        value is false).
    "visibility_timeout": 300,        - The visibility timeout for the queue.
    "resource_type": "sqs_queue",     * Resource type. Required.
    "delay_seconds": 30,              - The length of time, in seconds, for
                                        which the delivery of all messages in
                                        the queue is delayed
    "maximum_message_size": 1024,     - The limit of how many bytes a message
                                        can contain before Amazon SQS rejects
                                        it.
    "message_retention_period": 60, - The length of time, in seconds, for
                                        which Amazon SQS retains a message.
    "policy": {},                     - The queue's policy. A valid AWS policy.

    "receive_message_wait_time_seconds": 15, - The length of time, in
                                        seconds, for which a "ReceiveMessage"
                                        action waits for a message to arrive.
    "redrive_policy": {               - Not required.
        "deadLetterTargetArn":        * The Amazon Resource Name (ARN) of the
         "arn",                         dead-letter queue to which Amazon SQS
                                        moves messages after the value of
                                        maxReceiveCount is exceeded.
        "maxReceiveCount": 5          * The number of times a message is
                                        delivered to the source queue before
                                        being moved to the dead-letter queue.
    },
    "kms_master_key_id":              - The ID of an AWS-managed customer
      "alias/aws/sqs",                  master key (CMK) for Amazon SQS or
                                        a custom CMK.
    "kms_data_key_reuse_period_seconds": 60 - The length of time, in seconds,
                                        for which Amazon SQS can reuse a
                                        data key to encrypt or decrypt
                                        messages before calling AWS KMS again.
    "content_based_deduplication":   - Enables content-based.
     true|false
}
```

Example:

```
"${terraform-queue-name}": {
        "region": "eu-west-1",
        "fifo_queue": true,
        "visibility_timeout": 300,
        "resource_type": "sqs_queue"
    }
```

4.14 STEP FUNCTIONS

"resource_type": "step_functions"

"resource_type": "cloudwatch_rule_trigger"

```
"state_machine_collect_reports": {         * The Amazon States Language
        "definition": {                      definition of the state machine.
…                                            Required.
        },
        "iam_role": "state_machine_role", * IAM role to use for this state
                                             machine. Required.
        "resource_type": "step_functions",* Resource type. Required.
        "event_sources": [                - Subscriptions.
            {
                "input": {                * Input to Cloudwatch rule.
                 "event": "weekly_instance_audit_report" Required.
                },
                "iam_role":                 * IAM role name to use for this
                  lr_start_state_machine",   state machine. Required.

                "resource_type":            * Resource type. Required.
                  "cloudwatch_rule_trigger",

                "target_rule":              * Name of the CloudWatch
                  "weekly_report_event"      rule. Required.

            }
        ]
}
```

Example:

```
"state_machine_collect_nessus_reports": {
        "definition": {
            "States": {
                "GoogleMatchState": {
                    "InputPath": "$.lambdaPayload",
                    "End": true,
                    "Type": "Task",
                    "Lambda":                If the field is present, Lambda
"lambda_collect_google_nessus_reports"     function is attached to the
                                           state (in the same way the Activity
                                           field can be represented)


                },
                "ChoiceState": {
                    "Default": "DefaultState",
                    "Type": "Choice",
                    "Choices": [
                        {
                            "Variable": "$.cloud",
                            "StringEquals": "AWS",
                            "Next": "AwsMatchState"
                        },
                        {
                            "Variable": "$.cloud",
                            "StringEquals": "Google",
                            "Next": "GoogleMatchState"
                        }
                    ]
                },
                "AwsMatchState": {
                    "InputPath": "$.lambdaPayload",
                    "End": true,
                    "Type": "Task",
                    "Lambda": "lambda_collect_aws_nessus_reports"
                },
                "DefaultState": {
                    "Cause": "No Matches!",
                    "Type": "Fail"
                },
                "WaitState": {
                    "SecondsPath": "$.timeToWaitInSeconds",
                    "Type": "Wait",
                    "Next": "ChoiceState"
                }
            },
            "StartAt": "WaitState"
        },
        "dependencies": [
            {
                "resource_name": "lambda_collect_aws_nessus_reports",
                "resource_type": "lambda"
            },
            {
                "resource_name": "lambda_collect_google_nessus_reports",
                "resource_type": "lambda"
            }
        ],
        "iam_role": "state_machine_role",
        "resource_type": "step_functions"
    }
```

4.15 COGNITO

"resource_type": "cognito_federated_pool"

```
"maestro3_epam_opensource": {
        "auth_role": "cognito_auth",       - IAM role for authorized users.
      "unauth_role": "cognito_auth",       - IAM role for unauthorized users.
        "open_id_providers": [             - A list of OpendID Connect
                                             providers.
            "accounts.google.com"
        ],
        "provider_name": "login.m3.com",   - Developer provider name.
        "resource_type":                   * Resource type. Required.
         "cognito_federated_pool"
    }
```

Example:

```

"maestro3_epam_opensource": {
        "auth_role": "cognito_auth",
        "open_id_providers": [
            "accounts.google.com"
        ],
        "provider_name": "login.m3.com",
        "resource_type": "cognito_federated_pool"
    }
```

4.16 SNS APPLICATION

"resource_type": "sns_application"

```
"mobile-app": {
        "platform":                        * SNS application platform.
         "GCM|ADM|APNS|APNS_SANDBOX|GCM",    Required.

        "region": "eu-central-1",          - Regions name/list, where the
                                             application is deployed (if not
                                             stated, is deployed only in the
                                             region).
        "resource_type":                   * Resource type. Required.
         "sns_application",
        "attributes": {                    * SNS application attributes.
         "attr_name": "attr_value"           Required.
…
        }
    }
```

Example:

```
"mobile-app": {
        "platform": "GCM",
        "region": "eu-central-1",
        "resource_type": "sns_application",
        "attributes": {
            "PlatformCredential": "${google_api_key}"
        }
    }
```

4.17 ELASTIC BEANSTALK

"resource_type": "beanstalk_app"

```
"aws_csv_billing": {
    "resource_type": "beanstalk_app",      * Resource type. Required.
    "deployment_package":                  * Application artifact name.
                                             Required.
      "m3-server-1.0.0.war", *
    "env_name": "m3-billing-env",          * EBS environment name. Required.
    "notification_topic":                  - SNS topic name to configure
      "ebs_notification",                    notifications.
    "ec2_key_pair": "m3_deployment",       * EC2 key to run an instance.
                                             Required.
    "ec2_role": "ebs_instance_role",       * EC2 instance role. Required.
    "ebs_service_role": "ebs_service_role",* EBS service role. Required.
    "tier": {                              * EBS tier. Required.
      "Name": "WebServer",
      "Type": "Standard"
    },
    "stack": "64bit Amazon Linux 2017.03  * EBS stack. Required.
      v2.6.3 running Tomcat 8 Java 8",

    "env_settings": [                      - If specified, AWS Elastic
                                             Beanstalk sets the specified
                                             configuration options to the
                                             requested value in the
                                             configuration set for the new
                                             environment.
      {
        "OptionName": "Availability Zones",
        "ResourceName": "AWSEBAutoScalingGroup",
        "Namespace": "aws:autoscaling:asg",
        "Value": "Any"
      }
…
    ]
  }
```

Example:

```
"aws_csv_billing": {
    "resource_type": "beanstalk_app",
    "deployment_package": "m3-server-1.0.0.war",
    "env_name": "m3-billing-env",
    "notification_topic": "ebs_notification",
    "ec2_key_pair": "m3_deployment",
    "ec2_role": "ebs_instance_role",
    "ebs_service_role": "ebs_service_role",
    "tier": {
      "Name": "WebServer",
      "Type": "Standard"
    },
    "stack": "64bit Amazon Linux 2017.03 v2.6.3 running Tomcat 8 Java 8",
    "env_settings": [
      {
        "OptionName": "Availability Zones",
        "ResourceName": "AWSEBAutoScalingGroup",
        "Namespace": "aws:autoscaling:asg",
        "Value": "Any"
      },
      {
        "OptionName": "Cooldown",
        "ResourceName": "AWSEBAutoScalingGroup",
        "Namespace": "aws:autoscaling:asg",
        "Value": "360"
      },
      {
        "OptionName": "MaxSize",
        "ResourceName": "AWSEBAutoScalingGroup",
        "Namespace": "aws:autoscaling:asg",
        "Value": "1"
      },
      {
        "OptionName": "MinSize",
        "ResourceName": "AWSEBAutoScalingGroup",
        "Namespace": "aws:autoscaling:asg",
        "Value": "1"
      },
      {
        "OptionName": "ImageId",
        "ResourceName": "AWSEBAutoScalingLaunchConfiguration",
        "Namespace": "aws:autoscaling:launchconfiguration",
        "Value": "ami-ebd02392"
      },
      {
        "OptionName": "InstanceType",
        "Namespace": "aws:autoscaling:launchconfiguration",
        "Value": "t2.micro"
      },
      {
        "OptionName": "MonitoringInterval",
        "ResourceName": "AWSEBAutoScalingLaunchConfiguration",
        "Namespace": "aws:autoscaling:launchconfiguration",
        "Value": "5 minute"
      },
      {
        "OptionName": "RollingUpdateEnabled",
        "ResourceName": "AWSEBAutoScalingGroup",
        "Namespace": "aws:autoscaling:updatepolicy:rollingupdate",
        "Value": "false"
      },
      {
        "OptionName": "RollingUpdateType",
        "ResourceName": "AWSEBAutoScalingGroup",
        "Namespace": "aws:autoscaling:updatepolicy:rollingupdate",
        "Value": "Time"
      },
      {
        "OptionName": "HooksPkgUrl",
        "Namespace": "aws:cloudformation:template:parameter",
        "Value": "https://s3-eu-west-1.amazonaws.com/elasticbeanstalk-env-
resources-eu-west-1/stalks/eb_tomcat_4.0.1.148.17/lib/hooks.tar.gz"
      },
      {
        "OptionName": "InstancePort",
        "Namespace": "aws:cloudformation:template:parameter",
        "Value": "80"
      },
      {
        "OptionName": "JVMOptions",
        "Namespace": "aws:cloudformation:template:parameter",
        "Value": "XX:MaxPermSize=64m,Xmx=256m,JVM Options=,Xms=256m"
      },
      {
        "OptionName": "Application Healthcheck URL",
        "Namespace": "aws:elasticbeanstalk:application",
        "Value": ""
      },
      {
        "OptionName": "DeleteOnTerminate",
        "Namespace": "aws:elasticbeanstalk:cloudwatch:logs",
        "Value": "false"
      },
      {
        "OptionName": "RetentionInDays",
        "Namespace": "aws:elasticbeanstalk:cloudwatch:logs",
        "Value": "7"
      },
      {
        "OptionName": "StreamLogs",
        "Namespace": "aws:elasticbeanstalk:cloudwatch:logs",
        "Value": "false"
      },
      {
        "OptionName": "BatchSize",
        "Namespace": "aws:elasticbeanstalk:command",
        "Value": "100"
      },
      {
        "OptionName": "BatchSizeType",
        "Namespace": "aws:elasticbeanstalk:command",
        "Value": "Percentage"
      },
      {
        "OptionName": "IgnoreHealthCheck",
        "Namespace": "aws:elasticbeanstalk:command",
        "Value": "false"
      },
      {
        "OptionName": "Timeout",
        "Namespace": "aws:elasticbeanstalk:command",
        "Value": "600"
      },
      {
        "OptionName": "JVM Options",
        "Namespace": "aws:elasticbeanstalk:container:tomcat:jvmoptions",
        "Value": ""
      },
      {
        "OptionName": "XX:MaxPermSize",
        "Namespace": "aws:elasticbeanstalk:container:tomcat:jvmoptions",
        "Value": "64m"
      },
      {
        "OptionName": "Xms",
        "Namespace": "aws:elasticbeanstalk:container:tomcat:jvmoptions",
        "Value": "256m"
      },
      {
        "OptionName": "Xmx",
        "Namespace": "aws:elasticbeanstalk:container:tomcat:jvmoptions",
        "Value": "256m"
      },
      {
        "OptionName": "DefaultSSHPort",
        "Namespace": "aws:elasticbeanstalk:control",
        "Value": "22"
      },
      {
        "OptionName": "LaunchTimeout",
        "Namespace": "aws:elasticbeanstalk:control",
        "Value": "0"
      },
      {
        "OptionName": "LaunchType",
        "Namespace": "aws:elasticbeanstalk:control",
        "Value": "Migration"
      },
      {
        "OptionName": "RollbackLaunchOnFailure",
        "Namespace": "aws:elasticbeanstalk:control",
        "Value": "false"
      },
      {
        "OptionName": "EnvironmentType",
        "Namespace": "aws:elasticbeanstalk:environment",
        "Value": "SingleInstance"
      },
      {
        "OptionName": "GzipCompression",
        "Namespace": "aws:elasticbeanstalk:environment:proxy",
        "Value": "true"
      },
      {
        "OptionName": "ProxyServer",
        "Namespace": "aws:elasticbeanstalk:environment:proxy",
        "Value": "apache"
      },
      {
        "OptionName": "HealthCheckSuccessThreshold",
        "Namespace": "aws:elasticbeanstalk:healthreporting:system",
        "Value": "Ok"
      },
      {
        "OptionName": "SystemType",
        "Namespace": "aws:elasticbeanstalk:healthreporting:system",
        "Value": "enhanced"
      },
      {
        "OptionName": "LogPublicationControl",
        "Namespace": "aws:elasticbeanstalk:hostmanager",
        "Value": "false"
```

```
    },
    {
      "OptionName": "ManagedActionsEnabled",
      "Namespace": "aws:elasticbeanstalk:managedactions",
      "Value": "false"
    },
    {
      "OptionName": "InstanceRefreshEnabled",
      "Namespace": "aws:elasticbeanstalk:managedactions:platformupdate",
      "Value": "false"
    },
    {
      "OptionName": "Automatically Terminate Unhealthy Instances",
      "Namespace": "aws:elasticbeanstalk:monitoring",
      "Value": "true"
    },
    {
      "OptionName": "Notification Protocol",
      "Namespace": "aws:elasticbeanstalk:sns:topics",
      "Value": "email"
    },
    {
      "OptionName": "XRayEnabled",
      "Namespace": "aws:elasticbeanstalk:xray",
      "Value": "false"
    },
    {
      "OptionName": "EnvironmentVariables",
      "Namespace": "aws:cloudformation:template:parameter",
      "Value": "HOME_REGION="
    },
    {
      "OptionName": "HOME_REGION",
      "Namespace": "aws:elasticbeanstalk:application:environment",
      "Value": "${billing_home_region}"
    },
    {
      "OptionName": "EnvironmentVariables",
      "Namespace": "aws:cloudformation:template:parameter",
      "Value": "HOME_ACCOUNT_ID="
    },
    {
      "OptionName": "HOME_ACCOUNT_ID",
      "Namespace": "aws:elasticbeanstalk:application:environment",
      "Value": "${billing_home_account_id}"
    }
  ]  }
```

4.18 EC2 INSTANCE

" resource_type ": " ec2_instance "

```
"admin-instance": {
        "security_group_names": [         - Security group names.

"sg_name"
],
        "security_group_ids": [           - Security group IDs.
            "${customer_sg_id}"
        ],
        "availability_zone": "eu-west-1a",- Availability zone.
        "instance_type": "t2.micro",      * Instance type. Required.
        "subnet_id": "${admin_subnet_id}",- Subnet ID (needed
                                            if availability_zone is present).
        "key_name":
          "${admin_instance_key_name}",   * SSH key. Required.
        "image_id":
"${admin_instance_image}",                * Image ID. Required.
        "userdata_file":                  - File path to userdata
         "admin_instance_userdata.sh",      (file relative pathname from the
                                            directory, which is set up in the
                                            environmental variable
                                            SDCT_CONF).
        "resource_type": "ec2_instance",  * Resource type. Required.
        "disableApiTermination":          - API termination protection
       true|false,
        "iam_role": "m3AdminInstanceRole" - Instance IAM role
    }
```

Example:

```
"admin-instance": {
        "security_group_ids": [
            "${customer_sg_id}"
        ],
        "instance_type": "t2.micro",
        "subnet_id": "${admin_subnet_id}",
        "key_name": "${admin_instance_key_name}",
        "image_id": "${admin_instance_image}",
        "userdata_file": "admin_instance_userdata.sh",
        "resource_type": "ec2_instance",
        "disableApiTermination": true,
        "iam_role": "AdminInstanceRole"
    }
```

5 USING ALIASES

The deployment framework can work with static and dynamic aliases for referencing deployed resources.

Each alias type usage details are given further in this section.

5.1 STATIC ALIASES

The **static aliases** can be used for convenient distinction of infrastructures that are deployed from the same meta descriptions but need to have different resource naming in AWS.

For example, this can be used in setting up similar prod and dev environments, or deploying infrastructure in different regions or accounts.

The **static aliases** are described in the **sdct_aliases.conf** file which must be placed in the same directory with the **sdct.conf** file.

The **sdct_aliases.conf** includes the key-value list of aliases, for example:

```
dev_notification_bucket=notification-temp
```

During the deployment, the name, specified in the meta description as **${dev_notification_bucket}** will be replaced with '**notification-temp**'.

5.2 DYNAMIC ALIASES

The **dynamic aliases** are used for the cases when you need to reference the IDs of the resources after these resources are deployed.

A dynamic alias is set within the meta description of the AWS resource that influences the alias value. It is described in the **apply_changes** attribute in the Operation Files.

Currently, dynamic aliases are supported for two types of resources. For each resource type, you need to provide specific details:

IAM Policy alias:

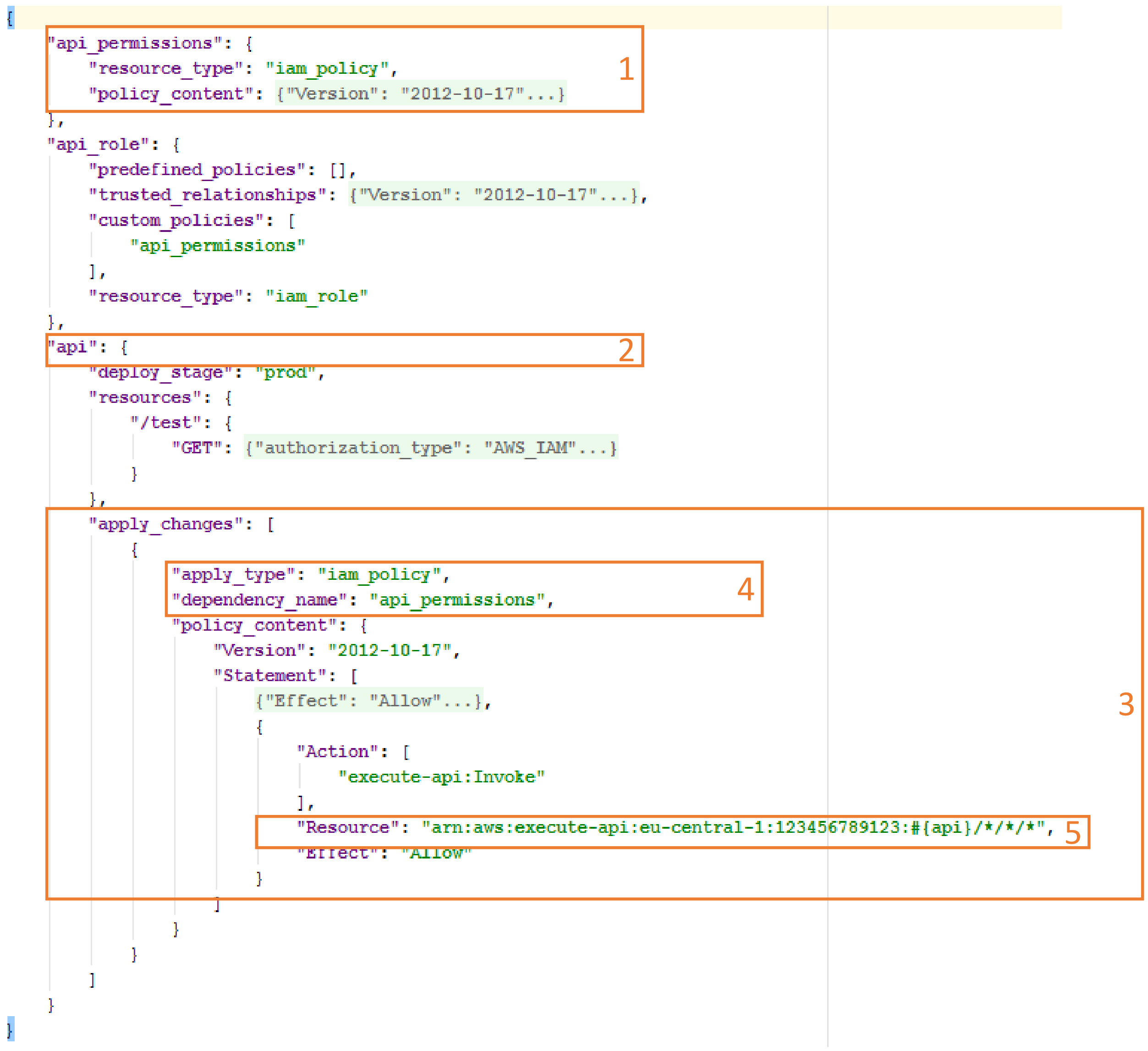
- **Action**: apply_type: iam_policy
- **Dependency name**
- **Policy Content**

IAM Role alias:

- **Action**: apply_type:iam_role
- **Dependency name**
- **Trusted relationships**

The resource name in alias is specified as **#{resource_name}**. For **API Gateway** and **Cognito**, this line is further transformed into a resource ID, generated by AWS during the deployment.

For example (see the screenshot below): you have an IAM policy **(1)** and an API Gateway **(2)** described in the **deployment_resources.json**. After the API gateway is deployed, you need to dynamically update the policy by adding the API Gateway ID which can be retrieved only after the gateway is deployed. To do this, you specify the **apply_changes** attribute **(3)** in the API Gateway description. The attribute links the changes to the target policy **(4,5)** :



```
{
    "api_permissions": {
        "resource_type": "iam_policy",
        "policy_content": {"Version": "2012-10-17"...}
    },
    "api_role": {
        "predefined_policies": [],
        "trusted_relationships": {"Version": "2012-10-17"...},
        "custom_policies": [
            "api_permissions"
        ],
        "resource_type": "iam_role"
    },
    "api": {
        "deploy_stage": "prod",
        "resources": {
            "/test": {
                "GET": {"authorization_type": "AWS_IAM"...}
            }
        },
        "apply_changes": [
            {
                "apply_type": "iam_policy",
                "dependency_name": "api_permissions",
                "policy_content": {
                    "Version": "2012-10-17",
                    "Statement": [
                        {"Effect": "Allow"...},
                        {
                            "Action": [
                                "execute-api:Invoke"
                            ],
                            "Resource": "arn:aws:execute-api:eu-central-1:123456789123:#{api}/*/*/*",
                            "Effect": "Allow"
                        }
                    ]
                }
            }
        ]
    }
}
```

Figure 1 - Setting a dynamic alias

*The complete **deployment_resources.json** file with the dynamic alias description can be found in the examples folder.*

6 AVAILABLE FRAMEWORK COMMANDS

Below, you can find the meta description format and the examples for each type of the supported resources.

Below, you can see the list of deployment framework commands (the required options are marked with an asterisk *):

1. **syndicate mvn_compile_java:** the command to compile a java project with lambdas:
   a. **bundle_name*:** the bundle name, to which the build artifacts are gathered and later used for the deployment)
   b. **project_path*** – the path to the Java project

   The provided path is the path for an **mvn clean install**. The artifacts are copied to a folder, which is be later used as the deployment bundle (the bundle path: **bundles/${bundle_name}**). The folders are created in the place, where the commands are executed later).

2. **syndicate assemble_python** – the command to build the lambda artifacts, which are written on Python:
   a. **bundle_name*:** the bundle name, to which the build artifacts are gathered and later used for the deployment
   b. **project_path*:** the path to the Python project

   The code is packed to a zip archive, where the external libraries are found, which are described in the requirements.txt file, and internal project dependancies according to the described in local_requirements.txt file.

3. **syndicate build_artifacts** – the command to call the following commands: **mvn_compile_java** and **assemble_python**. The command bundles each pair from the configuration **build_projects_mapping**:
   a. **bundle_name*:** the bundle name, to which the build artifacts are gathered and later used for the deployment
4. **syndicate package_meta** – the command, which bundles all found meta descriptoions of the resources to one file, from which later the deployment is activated – build_meta.json. The file also comes to the deployment bundle:
   a. **bundle_name*:** the bundle name, to which the build artifacts are gathered and later used for the deployment
5. **syndicate create_deploy_target_bucket** – an auxilary command which allows fast creating of an S3 bucket, which be later used as an artifactory. The deployment bundle becomes its part and the artifacts from it is used for deployment. Upon a call a bucket is created with the name specified in the **deploy_target_bucket** command
6. **syndicate upload_bundle** – the command for uploading the selected bundle to an S3 bucket:
   a. **bundle_name*:** the bundle name, to which the build artifacts are gathered and later used for the deployment
7. **syndicate copy_bundle** – the command allows fast copying of a bundle from one account to another. This can help in cases, when you need to perform migrations. If you have an existing artifactory, this command allows you to move the bundle needed to another account to deploy the equivalent infrastructure:
   a. **bundle_name*:** the bundle name, to which the build artifacts are gathered and later used for the deployment
   b. **src_account_id***: the account ID, to which the bundle is to be uploaded
   c. **src_bucket_region*:** the name of the region with the bucket
   d. **role_name***: the role name from the specified account, which is assumed. Here you have to check the trusted relationship between the accounts. The active account must be a trusted one for the account which is specified in the command

8. **syndicate build_bundle** – the command, which allows to build an artifact. Includes the following command calls: **syndicate build_artifacts, syndicate package_meta, syndicate upload_bundle**:
    a. **bundle_name**: the bundle name, to which the build artifacts are gathered and later used for the deployment
9. **syndicate deploy** – the command to create resources in the account:
    a. **deploy_name***: the deploy name, gives agility in managing and deploying the infrastructure in one account. You can create infrastructure, using one and the same bundle, but with different configuration regions. These are two different deploys, and each of them has its own name, so that clean didn't depend on the deploy.
    b. **bundle_name***: bundle name, to which the build artifacts are gathered and later used for the deployment
    c. **deploy_only_types**: names of the resources to be deployed. You can deploy for example only DynamoDB tables from the file with the meta description of all resources
    d. **deploy_only_resources**: names of the resources to be deployed. You can deploy only selected ressources, specifying their names
    e. **deploy_only_resources_path:** the path to the .json file containing the list of the resources from the meta description, which are to be deployed. This simplifies the syntax of the previous parameter, if you need to deploy a big list of selected resources. The file contains a string array.
    f. **excluded_resources:** names of the resources, which are excluded from the deploy.
    g. **excluded_resources_path**: the path to a .json file that lists the names of the resources which should be excluded from the deployment. The file contains a string array.
    h. **excluded_types**: names of the resource types, which are excluded from the deploy. For example, you need to deploy everything except the DynamoDB tables.

    Before the deployment starts, static aliases from the file **sdct_aliases.conf** are resolved.

    When the deployment is over, the dynamic aliases are applied, as specified in the operation files.

    During the deployment, an output file (**<deploy_name>.json**) with the description of all deployed resources is created. The file is saved to the S3 bucket in **outputs** folder in the bundle with the deployed recourses.

    Below, you can see the order of resource deployment:

Order	Resource
1	iam_policy
2	iam_role
3	dynamodb_table
4	s3_bucket
5	cloudwatch_rule
6	dynamodb_stream
7	sns_topic
8	sqs_queue
9	kinesis_stream
10	cloudwatch_alarm
11	lambda
12	state_activity
13	step_functions
14	api_gateway
15	cognito_federated_pool
16	beanstalk_app
17	ec2_instance
18	sns_ application

10. **m3 clean** is the command which allows to delete the resources from the account:
    a. **deploy_name*:** the deployment name. This parameter allows the framework to decide, which exactly output file should be used. The resources are cleaned based on the output file which is created during the deployment process.
    b. **bundle_name*:** the name of the bundle, which was specified during the deployment
    c. **clean_only_types:** the names of the resource types to be deleted. You can delete, for example, only **DynamoDB** tables
    d. **clean_only_resources:** the name of the resources to be deleted. Allows to delete specific resources only.
    e. **clean_only_resources_path:** the path to a json-file with the resources from the meta description, which are to be deleted. The file consists of a string array.
    f. **excluded_resources:** the resource names, which are excluded from the clean and are not deleted
    g. **excluded_resources_path:** the path to a json file with the names of resources, which are excluded from the clean procedure. The file consists of a string name array.
    h. **excluded_types:** names of the resource types, which are excluded from the clean procedure and arenot deleted. For example, you need to deploy everything except the **DynamoDb** tables.

The Clean parameter also has its priorities:

Order	Resource
2	iam_policy
1	iam_role
3	dynamodb_table
4	s3_bucket
5	cloudwatch_rule
6	dynamodb_stream
7	sns_topic
8	sqs_queue
9	kinesis_stream
10	cloudwatch_alarm
11	lambda
12	state_activity
13	step_functions
14	api_gateway
15	cognito_federated_pool
16	beanstalk_app
17	ec2_instance
18	sns_ application

Using the **clean** command, you delete only the resources which are created at the start of a certain deployment and specified in the deployment output file.

The Output folder is deleted from the S3 bucket at the end of the clean procedure.

7 BASIC DEPLOYMENT FLOW

The standard resource deployment is performed according to the following flow:

1. Create the **sdct.conf** file which describes the framework configuration.
2. Setup the **SDCT_CONF** environment variable pointing to the **sdct.conf** file.
3. If necessary, add the **sdct.aliases** file.
4. Prepare resources meta descriptions in Syndicate operation files.
5. Deployment with the following steps:
   a. Create the bundle bucket in S3 (in case it is the first deploy to the target AWS account):

      ```
      syndicate create_deploy_target_bucket
      ```

   b. Collect the artifacts of the application and all Syndicate operation files, and create a bundle:

      ```
      syndicate build_bundle --bundle_name <bundle_name>
      ```

   c. Deploy the bundle:

      ```
      syndicate deploy --bundle_name <bundle_name> --deploy_name
      <deploy_name>
      ```

6. In case the infrastructure is not needed any more, run:

   ```
   syndicate clean --bundle_name <bundle_name> --deploy_name
   <deploy_name>
   ```

   The command cleans the whole AWS infrastructure in the specified deploy, except the excluded resources, if any.

VERSION HISTORY

Version	Date	Summary
1.0	September 8, 2018	First published