AWS Instance Scheduler

AWS Implementation Guide

Arie Leeuwesteijn

Mahmoud ElZayet

Ruald Andreae

February 2018

Contents

About This Guide

This implementation guide discusses architectural considerations and configuration steps for deploying the Amazon Web Services (AWS) Instance Scheduler on the AWS Cloud. It includes links to AWS CloudFormation templates that launch, configure, and run the AWS services required to deploy this solution using AWS best practices for security and availability.

The guide is intended for IT infrastructure architects, administrators, and DevOps professionals who have practical experience architecting on the AWS Cloud.

Overview

Amazon Web Services (AWS) offers infrastructure on demand so that customers can control their resource capacity and pay only for what they use. One simple method to reduce costs is to stop resources that are not in use, and then start those resources again when their capacity is needed.

The AWS Instance Scheduler is a solution that automates the starting and stopping of Amazon Elastic Compute Cloud (Amazon EC2) and Amazon Relational Database Service (Amazon RDS) instances.

The Instance Scheduler leverages AWS resource tags and AWS Lambda to automatically stop[1] and restart instances across multiple AWS Regions and accounts on a customer-defined schedule. The solution is easy to deploy and can help reduce operational costs. For example, an organization can use the Instance Scheduler in a production environment to automatically stop instances every day, outside of business hours. For customers who leave all of their instances running at full utilization, this solution can result in up to 70% cost savings for those instances that are only necessary during regular business hours (weekly utilization reduced from 168 hours to 50 hours).

Cost

You are responsible for the cost of the AWS services used while running the Instance Scheduler. As of the date of publication, the cost for running this solution with default settings in the US East (N. Virginia) Region is approximately **$5.00 per month** in AWS Lambda charges, or less if you have Lambda free tier monthly usage credit. This is independent of the number of Amazon EC2 instances you are running. The optional custom Amazon CloudWatch metric, if enabled, will cost an additional **$0.90 per month per schedule or scheduled service**. By default, this solution enables Auto Scaling for its Amazon DynamoDB tables to provide sufficient read and write capacity. If you use this solution to schedule a large number of resources, Auto Scaling can increase Amazon DynamoDB charges.

This pricing does not reflect variable charges incurred from Amazon DynamoDB. Prices are subject to change. For full details, see the pricing webpage for each AWS service you will be using in this solution.

[1] Note that *stopping* an Amazon EC2 instance is different than *terminating* an Amazon EC2 instance. By default, Amazon EC2 instances are configured to stop, not terminate, when shut down, but customers can modify this behavior. Before using this solution, verify that instances are set to stop or terminate as appropriate.

Architecture Overview

Deploying this solution with the **default parameters** builds the following environment in the AWS Cloud.

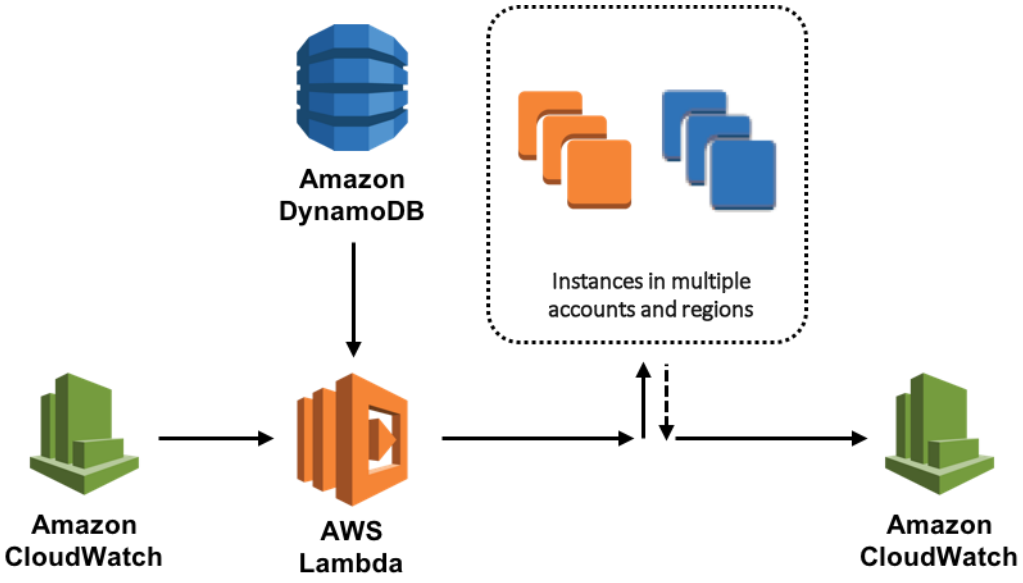


Figure 1: Instance Scheduler on the AWS Cloud

The AWS CloudFormation template sets up an Amazon CloudWatch event at a customer-defined interval. This event invokes the Instance Scheduler AWS Lambda function. During configuration, the user defines the AWS Regions and accounts, as well as a *custom tag* that the Instance Scheduler will use to associate schedules with applicable Amazon EC2 and Amazon RDS instances. These values are stored in Amazon DynamoDB, and the Lambda function retrieves them each time it runs. The customer then applies the custom tag to applicable instances.

During initial configuration of the Instance Scheduler, you define a tag *key* you will use to identify applicable Amazon EC2 and Amazon RDS instances. When you create a schedule, the name you specify is used as the tag *value* that identifies the schedule you want to apply to the tagged resource. For example, a user might use the solution's default tag name (tag key) `Schedule` and create a schedule called `uk-office-hours`. To identify an instance that will use the `uk-office-hours` schedule, the user adds the `Schedule` tag key with a value of `uk-office-hours`.

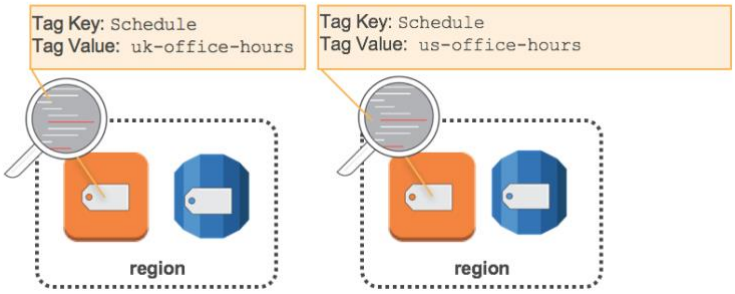


Figure 2: Instance Scheduler resource tagging

Each time the solution's Lambda function runs, it checks the current state of each appropriately tagged instance against the targeted state (defined by one or more periods in a schedule in the instance tag) in the associated schedule, and then applies the appropriate start or stop action, as necessary.

For example, if the Lambda function is invoked on a Friday at 9 am (ET) and it identifies a stopped Amazon EC2 or Amazon RDS instance with a `Schedule=office-hours` tag, it will check Amazon DynamoDB for the `office-hours` schedule configuration details. If the `office-hours` schedule contains a period rule that indicates that the instance should run Monday through Friday from 9 am ET to 5 pm ET, the Lambda function will start that instance.

The Lambda function also records the name of the schedule, the number of instances associated with that schedule, and the number of running instances as an optional custom metric in Amazon CloudWatch (see Amazon CloudWatch Metrics).

Solution Components

Scheduler Configuration Table

When deployed, the AWS Instance Scheduler creates an Amazon DynamoDB table that contains global configuration settings. To modify these global configuration settings after the solution is deployed, update the AWS CloudFormation stack. Do not modify these values in the DynamoDB table. If you modify the values in the DynamoDB table, you will create a conflict between the stored parameters in the stack and the values in the table.

Global configuration items contain a `type` attribute with a value of **config** in the configuration table. Schedules and periods contain `type` attributes with values of **schedule** and **period**, respectively. You can add, update, or remove schedules and periods from the configuration table using the DynamoDB console or the solution's command line interface.

Schedules

Schedules specify when Amazon Elastic Compute Cloud (Amazon EC2) and Amazon Relational Database Service (Amazon RDS) instances should run. Each schedule must have a unique name, which is used as the tag *value* that identifies the schedule you want to apply to the tagged resource.

Periods

Each schedule must contain at least one period that defines the time(s) the instance should run. A schedule can contain more than one period. When more than one period is used in a schedule, the Instance Scheduler will apply the appropriate start action when at least one of the period rules is true. For more information, see Period Rules.

Time Zone

You can also specify a time zone for the schedule. If you do not specify a time zone, the schedule will use the default time zone you specify when you launch the solution.

Enforced Field

Schedules contain an `enforced` field that allows you to prevent an instance from being manually started outside of a running period, or manually stopped during a running period. If this field is set to `true` and a user manually starts an instance outside of a running period, the solution will stop the instance. If this field is set to `true`, it also restarts an instance if it was manually stopped during a running period.

Retain Running Field

The `retain_running` field prevents the solution from stopping an instance at the end of a running period if the instance was manually started before the beginning of the period. For example, if an instance with a period that runs from 9 am to 5 pm is manually started before 9 am, the solution will not stop the instance at 5 pm.

Override Status Field

Schedules also contain an `override_status` field that allows you to temporarily override the solution's start and stop actions. If you set the field to `running`, the solution will start but not stop the applicable instance. The instance will run until you stop it manually. If you set the field to `stopped`, the solution will stop but not start the applicable instance. The instance will not run until you manually start it.

Note that if you set the `override_status` field to `running` but use the `enforced` field to prevent an instance from being manually started outside of a running period, the solution will stop the instance. If you set the `override_status` field to `stopped` but use the `enforced` field to prevent an instance from being manually stopped during a running period, the solution will restart the instance.

Instance Type

For Amazon EC2 instances only, a schedule allows you to specify an optional instance type for each period in a schedule. When you specify an instance type in the period, the solution will start the Amazon EC2 instance with the applicable instance type.

To specify an instance type, use the syntax *<period-name>@<instance-type>*. For example, `weekends@t2.nano`. Note that if you specify an instance type for a period that schedules Amazon EC2 instances and Amazon RDS instances, the instance type will be ignored for Amazon RDS instances.

If the instance type of a running instance is different than the instance type specified for the period, the solution will stop the running instance and restart the instance with the specified instance type, if the specified instance type is compatible with the instance configuration of the running instance. For more information, see Compatibility for Resizing Instances in the Amazon EC2 User Guide for Linux Instances.

Schedule Definitions

The Instance Scheduler configuration table in Amazon DynamoDB contains schedule definitions. A schedule definition can contain the following fields.

Field	Description
`description`	An optional description of the schedule
`enforced`	Choose whether to enforce the schedule. When this field is set to `true`, the scheduler will stop a running instance if it is manually started outside of the running period or it will start an instance if it is stopped manually during the running period.
`name`	The name used to identify the schedule. This name must be unique.
`override_status`	When this field is set to `running`, the instance will be started but not stopped until you stop it manually. When this field is set to `stopped`, the instance will be stopped but not started until you start it manually.
`periods`	The name of the periods that are used in this schedule. Enter the name(s) exactly as it appears in the period `name` field. You can also specify an instance type for the period using the syntax *<period-name>@<instance-type>*. For example, `weekdays@t2.large`.
`retain_running`	Choose whether to prevent the solution from stopping an instance at the end of a running period if the instance was manually started before the beginning of the period.
`stop_new_instances`	Choose whether to stop an instance the first time it is tagged if it is running outside of the running period. By default, this field is set to `true`.
`timezone`	The time zone the schedule will use. If no time zone is specified, the default time zone (UTC) is used.

Field	Description
use_maintenance_window	Choose whether to add an Amazon RDS maintenance window as a running period to an Amazon RDS instance schedule. This field is ignored for Amazon EC2 instances. For more information, see Amazon RDS Maintenance Window.
use_metrics	Choose whether to enable CloudWatch metrics at the schedule level. This field overwrites the CloudWatch metrics setting you specified at deployment. **Note:** Enabling this feature will incur charges of $0.90/month per schedule or scheduled service.

Period Rules

Period rules contain conditions that allow you to set the specific hours, days, and months an instance will run. A period rule can contain multiple conditions, but all conditions must be true for the AWS Instance Scheduler to apply the appropriate start or stop action.

Start and Stop Times

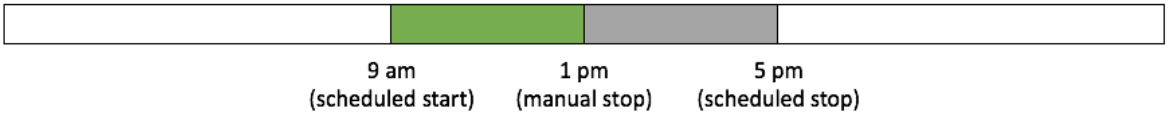
The `begintime` and `endtime` fields define when the Instance Scheduler will start and stop instances. If you specify a start time only, the instance must be stopped manually. Similarly, if you specify a stop time only, the instance must be started manually. If you don't specify either time, the solution uses the days of the week, days of the month, or months rules to start and stop instances.

If you start an instance before the specified start time, the instance will run until the end of the running period. For example, a user might define a period that starts an instance daily at 9 am and stops that instance at 5 pm.

If the user manually starts that instance at 5 am, the solution will stop the instance at 5 pm. Note that if you use the retain running field, the solution will not stop the instance at 5 pm.

If you stop an instance before the specified stop time, the instance will not run until the beginning of the next running period. Continuing from the previous example, if the user stops

the instance at 1 pm on Wednesday, the solution will not start the instance until 9 am on Thursday.



Days of the Week

The `weekdays` field defines which days during the week an instance will run. You can specify a list of days, a range of days, the nth occurrence of that day in a month, or the last occurrence of that day in a month. The solution supports abbreviated day names (Mon) and numbers (0). For more information, see Step 2.

Days of the Month

The `monthdays` field defines which days during the month an instance will run. You can specify a list of days, a range of days, every nth day of the month, the last day of the month, or the nearest weekday to a specific date. For more information, see Step 2.

Months

The `months` field defines which months an instance will run. You can specify a list of months, a range of months, or every nth month. The solution supports abbreviated month names (Jan) and numbers (1). For more information, see Step 2.

Period Definitions

The Instance Scheduler configuration table in Amazon DynamoDB contains period definitions. A period definition can contain the following fields.

Field	Description
Important: You must specify at least one of the following items: `begintime`, `endtime`, `weekdays`, `months`, or `monthdays`.	
`begintime`	The time, in HH:MM format, that the instance will start.
`description`	An optional description of the period rule
`endtime`	The time, in HH:MM format, that the instance will stop.
`months`	Enter a comma-delimited list of months, or a hyphenated range of months, during which the instance will run. For example, enter `jan, feb, mar` or `1, 2, 3` to run an instance during those months. Or, you can enter `jan-mar` or `1-3`. You can also schedule an instance to run every nth month or every nth month in a range. For example, enter `Jan/3` or `1/3` to run an instance every third month starting in January. Enter `Jan-Jul/2` to run every other month from January to July.

Field	Description
`monthdays`	Enter a comma-delimited list of days of the month, or a hyphenated range of days, during which the instance will run. For example, enter `1, 2, 3` or `1-3` to run an instance during the first three days of the month. You can also enter multiple ranges. For example, enter `1-3, 7-9` to run an instance from the 1st to the 3rd and the 7th through the 9th. You can also schedule an instance to run every nth day of the month or every nth day of the month in a range. For example, enter `1/7` to run an instance every seventh day starting on the 1st. Enter `1-15/2` to run an instance every other day from the 1st to the 15th. Enter `L` to run an instance on the last day of the month. Enter a date and `W` to run an instance on the nearest weekday to the specified date. For example, enter `15/W` to run an instance on the nearest weekday to the 15th.
`name`	The name used to identify the period rule. This name must be unique.
`weekdays`	Enter a comma-delimited list of days of the week, or a range of days of the week, during which the instance will run. For example, enter `0, 1, 2` or `0-2` to run an instance Monday through Wednesday. You can also enter multiple ranges. For example, enter `0-2, 4-6` to run an instance every day except Thursday. You can also schedule an instance to run every nth occurrence of a weekday in the month. For example, enter `Mon#1` or `0#1` to run an instance the first Monday of the month. Enter a day and `L` to run an instance on the last occurrence of that weekday in the month. For example, enter `friL` or `4L` to run an instance on the last Friday of the month.

Cross-Account Instance Scheduling

This solution includes a template (`instance-scheduler-remote`) that creates the AWS Identity and Access Management (IAM) roles necessary to start and stop instances in secondary accounts. You can review and modify permissions in the remote template before you launch the stack.

To apply automated start-stop schedules to resources in secondary accounts, launch the main solution template (`instance-scheduler`) in the primary account. Then, launch the remote template (`instance-scheduler-remote`) in each applicable secondary account. When each remote stack is launched, it creates a cross-account role Amazon Resource Name (ARN). Update the main solution stack with each cross-account role ARN by entering the appropriate ARN(s) in the **Cross-account roles** parameter to allow the AWS Instance Scheduler to perform start and stop actions on instances in the secondary accounts.

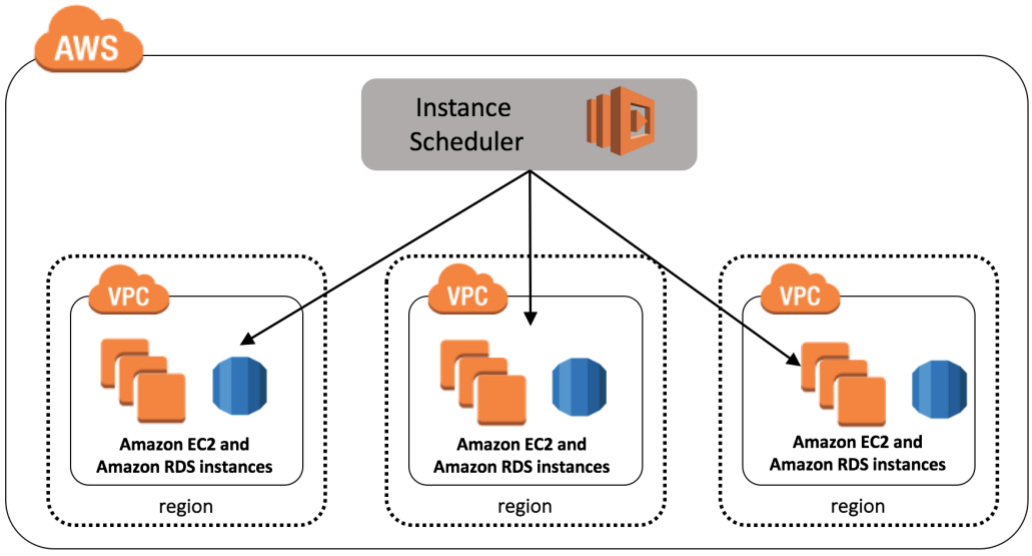


Figure 3: Cross-account scheduling

Automated Tagging

The Instance Scheduler can automatically add tags to all instances it starts or stops. You can specify a list of tag names or `tagname=tagvalue` pairs in the **Started tags** and **Stopped tags** parameters. The solution also includes macros that allow you to add variable information to the tags:

- `{scheduler}`: The name of the scheduler stack
- `{year}`: The year (four digits)
- `{month}`: The month (two digits)
- `{day}`: The day (two digits)
- `{hour}`: The hour (two digits, 24-hour format)
- `{minute}`: The minute (two digits)
- `{timezone}`: The time zone

The following table gives examples of different inputs and the resulting tags.

Example Parameter Input	Instance Scheduler Tag
`ScheduleMessage=Started by scheduler {scheduler}`	`ScheduleMessage=Started by scheduler MyScheduler`
`ScheduleMessage=Started on {year}/{month}/{day}`	`ScheduleMessage=Started on 2017/07/06`

Example Parameter Input	Instance Scheduler Tag
`ScheduleMessage=Started on {year}/{month}/{day} at {hour}:{minute}`	`ScheduleMessage=Started on 2017/07/06 at 09:00`
`ScheduleMessage=Started on {year}/{month}/{day} at {hour}:{minute} {timezone}`	`ScheduleMessage=Started on 2017/07/06 at 09:00 UTC`

When you use the **Started tags** parameter, the tags are automatically deleted when the scheduler stops the instance. When you use the **Stopped tags** parameter, the tags are automatically deleted when the instance is started.

Scheduler Command Line Interface

The AWS Instance Scheduler includes a command line interface (CLI) that provides commands for configuring schedules and periods. The CLI allows customers to estimate cost savings for a given schedule. The cost estimates provided by the schedule CLI are for approximation purposes only. For more information about configuring and using the scheduler CLI, see Appendix A.

Design Considerations

Partial Automation

Users have the option to implement a partially automated solution by default (i.e., configure start-only or stop-only actions). An example of this is a team that needs the flexibility to stop instances at varying times (manually) but must start those instances simultaneously each morning, or vice versa.

Instance Shutdown Behavior

Amazon EC2

This solution is designed to automatically stop Amazon Elastic Compute Cloud (Amazon EC2) instances and assumes that instance *shutdown behavior* is set to `Stop`, not `Terminate`. Note that you cannot restart an Amazon EC2 instance after it is terminated.

By default, Amazon EC2 instances are configured to stop, not terminate, when shut down, but customers can modify this behavior. Therefore, make sure that the instances you control using the Instance Scheduler are configured with a `Stop` shutdown behavior, otherwise they will be terminated.

Amazon RDS

Note that this solution is designed to automatically stop, not delete, Amazon Relational Database Service (Amazon RDS) instances. Before the Instance Scheduler stops an RDS

instance, it creates a snapshot of the instance. The snapshot is kept until the next time the instance is stopped and a new snapshot is created.

> **Note:** The scheduler will not start or stop RDS instances that are part of a cluster, or that manage Amazon Aurora databases. Tagged instances that cannot be scheduled are logged in Amazon CloudWatch. For more information about limitations to starting and stopping an Amazon RDS instance, see Stopping an Amazon RDS DB Instance Temporarily in the Amazon RDS User Guide.

When an Amazon RDS instance is stopped, the cache is cleared which might lead to slower performance when the instance is restarted.

Amazon RDS Maintenance Window

Every Amazon RDS instance has a weekly maintenance window during which any system changes are applied. During the maintenance window, Amazon RDS will automatically start instances that have been stopped for more than seven days to apply maintenance. Note that Amazon RDS will not stop the instance once the maintenance event is complete.

The Instance Scheduler allows you to specify whether to add the preferred maintenance window of an Amazon RDS instance as a running period to its schedule. The solution will start the instance at the beginning of the maintenance window and stop the instance at the end of the maintenance window if no other running period specifies that the instance should run, and if the maintenance event is completed.

If the maintenance event is not completed by the end of the maintenance window, the instance will run until the scheduling interval after the maintenance event is completed. For more information about the Amazon RDS maintenance window, see Amazon RDS Maintenance in the Amazon RDS User Guide.

Global Configuration Settings

When you deploy the Instance Scheduler's AWS CloudFormation template, global configuration settings are stored in an Amazon DynamoDB table (`ConfigTable`). To modify these settings, update the solution stack using the AWS CloudFormation template. Do not change these settings in the DynamoDB table.

Performance

If the Instance Scheduler AWS Lambda function does not process all scheduled instances before its next invocation, the solution logs the error in Amazon CloudWatch Logs, and sends an Amazon Simple Notification Service (Amazon SNS) notification to the error SNS topic. To help ensure that all instances are processed before the next invocation, customers can change

the default interval at which the Lambda function runs or launch multiple deployments of the solution with different tag names.

If you increase the default interval, this might reduce the granularity of your schedules. For example, a Lambda function set to run on a fifteen-minute interval will only perform start and stop actions every 15 minutes.

To schedule a large number of instances, we recommend using an interval of at least five minutes and increasing the memory size of the Instance Scheduler's main AWS Lambda function using the **Memory Size** parameter.

Regional Deployments

Customers can deploy the Instance Scheduler in any AWS Region. Once deployed, the Instance Scheduler applies the appropriate start or stop actions to tagged Amazon EC2 and Amazon RDS instances in all regions of a customer's account. If you enable cross-account instance scheduling, the solution will apply actions to instances in all regions in all accounts.

> **Important:** Instance Scheduler actions will affect appropriately tagged instances in all AWS Regions of your account, even though the Lambda function is running in a single region.

You can use multiple deployments of the solution to schedule a large number of instances, or instances in many accounts and regions. When you deploy multiple schedulers, use a different tag name for each stack, and configure a set of non-overlapping regions for each deployment. Each deployment checks every instance in every region in an account for the tag key that identifies resources it should schedule. If the regions for multiple deployments overlap, each instance will be checked by multiple deployments.

Logging and Notifications

The Instance Scheduler leverages Amazon CloudWatch Logs for logging. The solution logs processing information for each tagged instance; the results of the period rule evaluation for the instance; the desired state of the instance during that period; the applied action; and debugging messages. For more information, see Appendix B.

Warning and error messages are also published to a solution-created Amazon SNS topic which sends messages to a subscribed email address (see Subscribe to a Topic in the Amazon SNS Developer Guide). You can find the name of the Amazon SNS topic in the **Outputs** tab of the solution stack.

AWS CloudFormation Templates

This solution uses AWS CloudFormation to automate the deployment of the Instance Scheduler on the AWS Cloud. It includes the following CloudFormation templates, which you can download before deployment:

View template **instance-scheduler.template:** Use this template to launch the Instance Scheduler and all associated components. The default configuration deploys an AWS Lambda function, an Amazon DynamoDB table, an Amazon CloudWatch event, and CloudWatch custom metrics, but you can also customize the template based on your specific needs.

View template **instance-scheduler-remote.template:** Use this template to configure permissions for instances in secondary accounts. The default configuration creates the AWS Identity and Access Management (IAM) roles necessary to start and stop instances in a secondary account.

Automated Deployment

Before you launch the automated deployment, please review the architecture, configuration, and other considerations discussed in this guide. Follow the step-by-step instructions in this section to configure and deploy the Instance Scheduler into your account.

Time to deploy: Approximately five (5) minutes.

What We'll Cover

The procedure for deploying this architecture on AWS consists of the following steps. For detailed instructions, follow the links for each step.

Step 1. Launch the Instance Scheduler Stack

- Launch the AWS CloudFormation template into your AWS account
- Enter a value for the required parameter: **Stack Name**
- Review the other template parameters, and adjust if necessary

Step 2. Configure Periods

- Create a period and set the applicable fields for the period

Step 3. Configure Schedules

- Create a schedule and set the applicable fields for the schedule

Step 4. Tag Your Instances

- Apply the custom tag to applicable resources

Step 5. Launch the Remote Stack in Secondary Accounts (Optional)

- Launch the AWS CloudFormation template into your AWS account
- Enter a value for the required parameter: **Stack Name, Primary account**

Step 1. Launch the Instance Scheduler Stack

This automated AWS CloudFormation template deploys the Instance Scheduler and configures related components. Please make sure that you've verified the settings for your instances before launching the stack.

> **Note**: You are responsible for the cost of the AWS services used while running this solution. See the Cost section for more details. For full details, see the pricing webpage for each AWS service you will be using in this solution.

1. Sign in to the AWS Management Console and click the button to the right to launch the `instance-scheduler` AWS CloudFormation template.
   You can also download the template as a starting point for your own implementation.

   Launch Solution

2. The template is launched in the US East (N. Virginia) Region by default. To launch the Instance Scheduler in a different AWS Region, use the region selector in the console navigation bar.
3. On the **Select Template** page, verify that you selected the correct template and choose **Next**.
4. On the **Specify Details** page, assign a name to the solution stack.
5. Under **Parameters**, review the parameters for the template, and modify them as necessary.

   This solution uses the following default values.

Parameter	Default	Description
Instance Scheduler tag name	`Schedule`	This tag identifies instances to receive automated actions, and also allows for custom start-stop parameters. If you choose to modify the default value, make sure to assign a name that will be easy to apply consistently and correctly across all necessary instances.
Service(s) to schedule	`EC2`	The services to schedule. Select `EC2`, `RDS`, or `Both`.
Scheduling enabled	`Yes`	Select `No` to temporarily disable scheduling.
Region(s)	<Optional input>	List of regions where instances will be scheduled. For example, `us-east-1, us-west-1`.

Parameter	Default	Description
		Note: If you leave this parameter blank, the solution will use the current region.
Default time zone	`UTC`	Default time zone for schedules
Cross-account roles	<Optional input>	Comma-delimited list of cross-account roles. For example, `arn:aws:iam::111122223333:role/<stackname>SchedulerCrossAccountRole`. **Note:** Enter the secondary account **CrossAccountRoleArn** value(s) in this parameter.
This account	`Yes`	Select `Yes` to allow the task to select resources in this account. **Note:** If you set this parameter to `No`, you must configure cross-account roles.
Frequency	`5`	The frequency in minutes at which the AWS Lambda function runs. Select `1`, `2`, `5`, `10`, `15`, `30`, or `60`.
Enable CloudWatch Metrics	`No`	Choose whether to collect data using CloudWatch Metrics for all schedules. You can override this default setting for an individual schedule when you configure it (see Step 3). **Important:** Enabling this feature will incur charges of $0.90/month per schedule or scheduled service.
Memory Size	`128`	The memory size of the solution's main AWS Lambda function. Increase the default size to schedule a large number of Amazon EC2 and Amazon RDS instances.
Enable CloudWatch Logs	`No`	Choose whether to log detailed information in CloudWatch Logs.
Log retention days	`30`	The log retention period in days
Started tags	<Optional input>	Tags to add to started instances. Use a list of `tagname=tagvalue` pairs.
Stopped tags	<Optional input>	Tags to add to stopped instances. Use a list of `tagname=tagvalue` pairs.
Send anonymous usage data	`Yes`	Send anonymous data to AWS to help us understand solution usage and related cost savings across our customer base as a whole. To opt out of this feature, choose `No`. For more information, see Appendix E.

6. Choose **Next.**
7. On the **Options** page, choose **Next**.

8. On the **Review** page, review and confirm the settings. Be sure to check the box acknowledging that the template will create AWS Identity and Access Management (IAM) resources.
9. Choose **Create** to deploy the stack.

   You can view the status of the stack in the AWS CloudFormation Console in the **Status** column. You should see a status of CREATE_COMPLETE in roughly five (5) minutes.

Step 2. Configure Periods

When you deploy the AWS CloudFormation template, the solution creates an Amazon DynamoDB table that contains sample period rules and schedules that you can use as a reference to create your own custom period rules and schedules.

To create a period rule, you can use the Amazon DynamoDB console, the scheduler CLI, or the AWS CloudFormation custom resource. Note that if you use the custom resource to create a period, you must not use the DynamoDB console or scheduler CLI to delete or modify that period. If you do, you will create a conflict between the stored parameters in the stack and the values in the table. Also, do not use periods configured using the custom resource in schedules created using the DynamoDB console or the scheduler CLI.

To create a period rule in DynamoDB, modify one of the periods in the configuration table (`ConfigTable`). To create a period in the scheduler CLI, use the applicable commands. To create a period using the custom resource, add the applicable fields to the solution's custom resource.

Step 3. Configure Schedules

To create a schedule, you can use the Amazon DynamoDB console, the scheduler CLI, or the AWS CloudFormation custom resource. Note that if you use the custom resource to create a schedule, you must not use the DynamoDB console or scheduler CLI to delete or modify that schedule. If you do, you will create a conflict between the stored parameters in the stack and the values in the table.

To create a schedule in DynamoDB, modify one of the schedules in the configuration table (`ConfigTable`). To create a schedule in the scheduler CLI, use the applicable commands. To create a schedule using the custom resource, add the applicable fields to the solution's custom resource.

Step 4. Tag Your Instances

When you deployed the AWS CloudFormation template, you defined the name (tag key) for the solution's *custom tag*. For the Instance Scheduler to recognize an Amazon EC2 or Amazon RDS instance, the tag key on that instance must match the custom tag name stored

in the Amazon DynamoDB table. Therefore, it is important that you apply tags consistently and correctly to all applicable instances. You can continue to use existing tagging strategies for your instances while using this solution. For more information, see Tagging Your Amazon EC2 Resources and Tagging Your Amazon RDS Resources.

On the AWS Management Console, use the Tag Editor to apply or modify tags for multiple resources at a time. You can also apply and modify tags manually in the console.

Setting the Tag Value

As you apply a tag to a resource, use the tag key you defined during initial configuration and set the tag value to the name of a schedule to schedule that resource.

Step 5. Launch the Remote Stack in Secondary Accounts (Optional)

This automated AWS CloudFormation template configures secondary account permissions.

1. Navigate to the AWS Instance Scheduler stack **Outputs** tab and copy the **Value** of **SchedulerRole**.
2. Sign in to the AWS Management Console of the applicable secondary account and click the button to the right to launch the `instance-scheduler-remote` AWS CloudFormation template.
   You can also download the template as a starting point for your own implementation.

   Launch Remote Stack
3. The template is launched in the US East (N. Virginia) Region by default. To launch the template in a different AWS Region, use the region selector in the console navigation bar.
4. On the **Select Template** page, verify that you selected the correct template and choose **Next**.
5. On the **Specify Details** page, assign a name to your remote stack.
6. Under **Parameters**, review the parameter for the template, and modify it.

Parameter	Default	Description
Primary account	*<Requires Input>*	Enter the account number of the account with the primary Instance Scheduler stack. This parameter gives the solution permission to schedule Amazon EC2 and Amazon RDS instances in this account.

7. Choose **Next.**
8. On the **Options** page, choose **Next**.

9. On the **Review** page, review and confirm the settings. Be sure to check the box acknowledging that the template will create IAM resources.
10. Choose **Create** to deploy the stack.

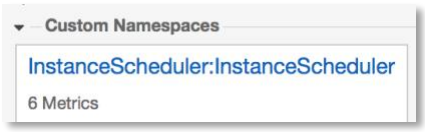
    You can view the status of the stack in the AWS CloudFormation Console in the **Status** column. You should see a status of CREATE_COMPLETE in roughly five (5) minutes.

Amazon CloudWatch Metrics

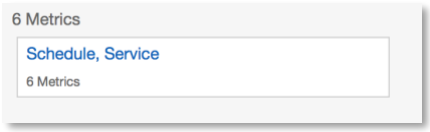
This solution creates a new custom Amazon CloudWatch metric (`InstanceScheduler:<stackname>`). Each time the AWS Lambda function runs, it updates the metric data for each applicable instance and then applies the appropriate start or stop action. This data includes the name of the schedule, the number of instances associated with that schedule, and the number of running instances.

View Instance Scheduler Metrics

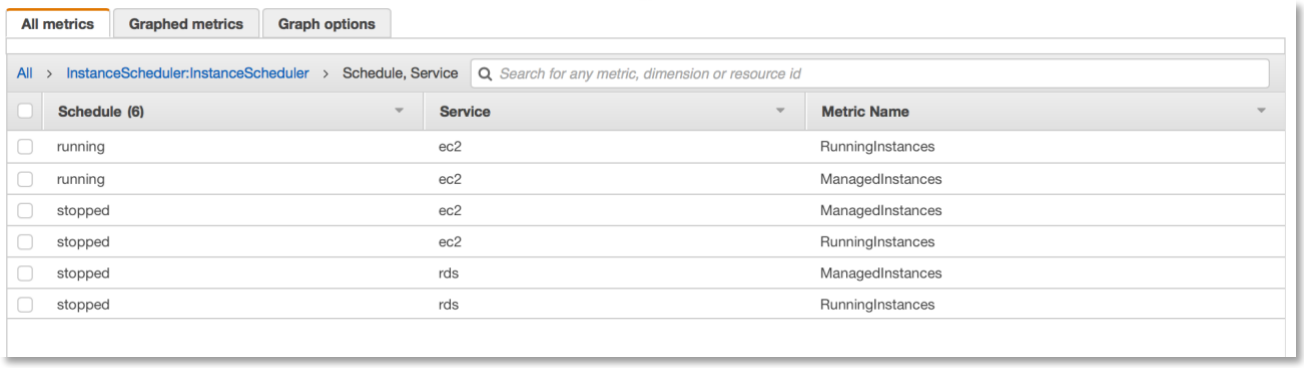
1. In the AWS Management Console, open the Amazon CloudWatch console.
2. In the **Custom Namespaces** drop-down field, choose `InstanceScheduler:<stackname>`.



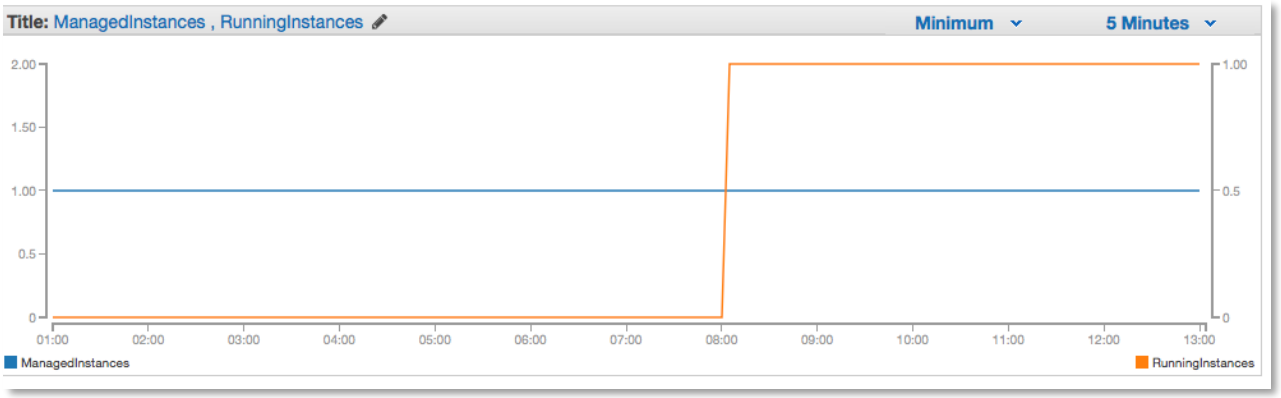
3. Select the schedule and service dimensions.



4. Select the schedule and service that you want to view the status of.



At the bottom of the page, an individual graph will appear for each schedule you selected, as shown in the following example. Note that a value of `0` is a stopped instance and a value of `1.00` is a running instance.



Additional Resources

AWS services documentation

- AWS CloudFormation
- Amazon EC2 User Guide for Linux instances
- AWS Lambda
- Amazon DynamoDB
- Amazon CloudWatch

Appendix A: Scheduler CLI

The Instance Scheduler command line interface (CLI) allows customers to configure schedules and periods, and estimate cost savings for a given schedule.

Credentials

To use the scheduler CLI, you must have credentials for the AWS CLI. For more information, see Configuration and Credential Files in the AWS CLI User Guide.

Your credentials must have the following permissions:

- `lambda:InvokeFunction` – To invoke the `InstanceSchedulerMain` function in the scheduler stack, and to update the schedule and period information in the scheduler configuration database from the command line
- `cloudformation:DescribeStackResource` – To retrieve the physical resource ID of the AWS Lambda function from the stack to handle the CLI request

Requests made by the scheduler CLI and responses are logged in the `AdminCliRequestHandler-yyyymmdd` log stream.

Install the Scheduler CLI

1. Download and unzip the scheduler CLI package.
2. In the `scheduler-cli-latest` directory, run the `setup` Python script:

```
python setup.py install
```

Command Structure

The scheduler CLI uses a multipart structure on the command line. The next part specifies the scheduler CLI python script. The scheduler CLI has commands that specify the operations to perform on periods and schedules. The specific arguments for an operation can be specified on the command line in any order.

```
scheduler-cli <command> <arguments>
```

Common Arguments

The scheduler CLI supports the following arguments that all commands can use:

Argument	Description
`--stack <stackname>`	The name of the scheduler stack

	Important: This argument is required for all commands.
`--region <regionname>`	The name of the region where the scheduler stack is deployed **Note:** You must use this argument when the default configuration and credential files are not installed in the same region as the solution stack.
`--query`	A JMESPath expression that controls the command output. For more information on controlling output, see Controlling Command Output from the AWS Command Line Interface in the AWS CLI User Guide.
`--help`	Shows valid commands and arguments for the scheduler CLI. When used with a specific command, it shows valid subcommands and arguments for that command.
`--version`	Shows the version number of the scheduler CLI

Available Commands

- create-period
- create-schedule
- delete-period
- delete-schedule
- describe-periods
- describe-schedules
- describe-schedule-usage
- update-period
- update-schedule
- help

create-period

Description

Creates a period. A period must contain at least one of the following items: `begintime`, `endtime`, `weekdays`, `months`, or `monthdays`.

Arguments

`--name`

The name of the period

Type: String

Required: Yes

`--description`

A description of the period

Type: String

Required: No

`--begintime`

The time when the running period starts. If `begintime` and `endtime` are not specified, the running period is 00:00 – 23:59.

Type: String

Constraints: `H:MM` or `HH:MM` format

Required: No

`--endtime`

The time when the running period stops. If `begintime` and `endtime` are not specified, the running period is 00:00 – 23:59.

Type: String

Constraints: `H:MM` or `HH:MM` format

Required: No

`--weekdays`

The days of the week for the period

Type: String

Constraints: Comma-delimited list of abbreviated day names (mon) or numbers (0). Use – to specify a range. Use / to specify every n^{th} day of the week.

Required: No

`--months`

The months of the period

Type: String

Constraints: Comma-delimited list of abbreviated months names (jan) or numbers (1). Use – to specify a range. Use / to specify every nth month.

Required: No

`--monthdays`

The days of the month for the period

Type: String

Constraints: Comma-delimited list of abbreviated month names (jan) or numbers (1). Use – to specify a range. Use / to specify every nth day of the month.

Required: No

Example

```
$ scheduler-cli create-period --name "weekdays" --begintime 09:00 --
endtime 18:00 --weekdays mon-fri --stack Scheduler
{
   "Period": {
      "Name": "weekdays",
      "Endtime": "18:00",
      "Type": "period",
      "Begintime": "09:00",
      "Weekdays": [
         "mon-fri"
      ]
   }
}
```

create-schedule

Description

Creates a schedule

Arguments

`--name`

The name of the schedule

Type: String

Required: Yes

`--description`

A description of the schedule

Type: String

Required: No

`--enforced`

Enforces the scheduled state for the instance

Type: Boolean

Constraints: Value must be `true` or `false`. By default, this parameter is set to `false`.

Required: No

`--metrics`

Collect Amazon CloudWatch metrics

Type: Boolean

Constraints: Value must be `true` or `false`. By default, this parameter is set to `false`.

Required: No

`--override-status`

Overrides the status of an instance to keep the instance in the specified state

Type: String

Constraints: Acceptable values are `running` or `stopped`

Required: No

`--periods`

A list of running periods for the schedule. If multiple periods are specified, the solution will start an instance if one of the period rules is true.

Type: String

Constraints: Comma-delimited list of periods. Use `<period-name>@<instance type>` to specify an instance type for a period. For example, `weekdays@t2.large`.

Required: Yes (if `override-status` is not used)

`--retain-running`

Prevents an instance from being stopped by the solution at the end of a running period, if the instance was manually started before the beginning of the period

Type: Boolean

Required: No

`--stop-new-instances`

Stops an instance the first time it is tagged if it is running outside of a running period

Type: Boolean

Constraints: Value must be `true` or `false`. By default, this parameter is set to `true`.

Required: No

`--timezone`

The time zone the schedule will use

Type: Array of strings

Required: No (If this argument is not used, the default time zone from main solution stack is used.)

`--use-maintenance-window`

Add an Amazon RDS maintenance window as a running period to an Amazon RDS instance schedule

Type: String

Required: No

Example

```
$ scheduler-cli create-schedule --name LondonOfficeHours --periods
weekdays,weekends --timezone Europe/London --stack Scheduler
{
   "Schedule": {
      "Enforced": false,
      "Name": "LondonOfficeHours",
      "StopNewInstances": true,
      "Periods": [
         "weekends",
         "weekdays"
      ],
      "Timezone": "Europe/London",
      "Type": "schedule"
   }
}
```

delete-period

Description

Deletes an existing period

Arguments

`--name`

The name of the applicable period

Type: String

Required: Yes

> **Important:** If the period is used in existing schedules, you must remove it from those schedules *before* you delete it.

Example

```
$ scheduler-cli delete-period --name weekdays --stack Scheduler
{
   "Period": "weekdays"
}
```

delete-schedule

Description

Deletes an existing schedule

Arguments

`--name`

The name of the applicable schedule

Type: String

Required: Yes

Example

```
$ scheduler-cli delete-schedule --name LondonOfficeHours --stack
Scheduler
{
   "Schedule": "LondonOfficeHours"
}
```

describe-periods

Description

Lists the configured periods for the Instance Scheduler stack

Arguments

`--name`

The name of a specific period you want described

Type: String

Required: No

Example

```
$ scheduler-cli describe-periods --stack Scheduler
{
   "Periods": [
      {
         "Name": "first-monday-in-quarter",
         "Months": [
            "jan/3"
         ],
         "Type": "period",
```

```
            "Weekdays": [
               "mon#1"
            ],
            "Description": "Every first Monday of each quarter"
         },
         {
            "Description": "Office hours",
            "Weekdays": [
               "mon-fri"
            ],
            "Begintime": "09:00",
            "Endtime": "17:00",
            "Type": "period",
            "Name": "office-hours"
         },

         {
            "Name": "weekdays",
            "Endtime": "18:00",
            "Type": "period",
            "Weekdays": [
               "mon-fri"
            ],
            "Begintime": "09:00"
         },
         {
            "Name": "weekends",
            "Type": "period",
            "Weekdays": [
               "sat-sun"
            ],
            "Description": "Days in weekend"
         }
      ]
   }
```

describe-schedules

Description

Lists the configured schedules for the Instance Scheduler stack

Arguments

`--name`

The name of a specific schedule you want described

Type: String

Required: No

Example

```
$ scheduler-cli describe-schedules --stack Scheduler

{
   "Schedules": [
      {
         "OverrideStatus": "running",
         "Type": "schedule",
         "Name": "Running",
         "UseMetrics": false
      },
      {
         "Timezone": "UTC",
         "Type": "schedule",
         "Periods": [
            "working-days@t2.micro",
            "weekends@t2.nano"
         ],
         "Name": "scale-up-down"
      },
      {
         "Timezone": "US/Pacific",
         "Type": "schedule",
         "Periods": [
            "office-hours"
         ],
         "Name": "seattle-office-hours"
      },
      {
         "OverrideStatus": "stopped",
         "Type": "schedule",
         "Name": "stopped",
         "UseMetrics": true
      }
   ]
}
```

describe-schedule-usage

Description

Lists all the periods running within a schedule and calculates the billing hours for instances. Use this command to simulate a schedule to calculate potential savings, and running periods after creating or updating a schedule.

Arguments

`--name`

The name of the applicable schedule

Type: String

Required: Yes

`--startdate`

The start date of the period used for calculation. The default date is the current date.

Type: String

Required: No

`--enddate`

The end date of the period used for calculation. The default date is the current date.

Type: String

Required: No

Example

```
$ scheduler-cli describe-schedule-usage --stack InstanceScheduler --
name seattle-office-hours
{
   "Usage": {
      "2017-12-04": {
         "BillingHours": 8,
         "RunningPeriods": {
            "Office-hours": {
               "Begin": "12/04/17 09:00:00",
               "End": "12/04/17 17:00:00",
               "BillingHours": 8,
               "BillingSeconds": 28800
            }
         },
         "BillingSeconds": 28800
      }
   },
   "Schedule": "seattle-office-hours"
```

update-period

Description

Updates an existing period

Arguments

The `update-period` command supports the same arguments as the `create-period` command. For more information on the arguments, see the create period command.

Note that if you do not specify an argument, that argument will be removed from the period.

update-schedule

Description

Updates an existing schedule.

Arguments

The `update-schedule` command supports the same arguments as the `create-schedule` command. For more information on the arguments, see the create schedule command.

> **Important:** If you do not specify an argument, that argument will be removed from the period.

help

Description

Displays a list of valid commands and arguments for the scheduler CLI

Example

```
$ scheduler-cli --help
usage: scheduler-cli [-h] [--version]
                                {create-period,create-schedule,delete-
period,delete-schedule,describe-periods,describe-schedule-
usage,describe-schedules,update-period,update-schedule}
                     ...

optional arguments:
  -h, --help            show this help message and exit
  --version             show program's version number and exit

subcommands:
  Valid subcommands

                  {create-period,create-schedule,delete-period,delete-
schedule,describe-periods,describe-schedule-usage,describe-
schedules,update-period,update-schedule}
                        Commands help
    create-period       Creates a period
    create-schedule     Creates a schedule
    delete-period       Deletes a period
    delete-schedule     Deletes a schedule
    describe-periods    Describes configured periods
    describe-schedule-usage
```

```
                          Calculates periods and billing hours in which
                          instances are running
    describe-schedules    Described configured schedules
    update-period         Updates a period
    update-schedule       Updates a schedule
```

When used with a specific command, the `--help` argument shows valid subcommands and arguments for that command.

Specific Command Example

```
$ scheduler-cli describe-schedules --help
usage: scheduler-cli describe-schedules [-h] [--name NAME] [--query
QUERY]
                                        [--region REGION] --stack STACK

optional arguments:
  -h, --help          show this help message and exit
  --name NAME         Name of the schedule
  --query QUERY       JMESPath query to transform or filter the result
  --region REGION     Region in which the Instance Scheduler stack is
                         deployed
  --stack STACK, -s STACK
                      Name of the Instance Scheduler stack
```

Appendix B: Solution Resources

The following resources are created as part of the Instance Scheduler stack.

Resource Name	Type	Description
Main	`AWS::Lambda::Function`	Instance Scheduler AWS Lambda function
Scheduler Config Helper	`Custom::ServiceSetup`	Stores global configuration settings in Amazon DynamoDB
Scheduler Invoke Permission	`AWS::Lambda::Permission`	Allows the Amazon CloudWatch event to invoke the Instance Scheduler's AWS Lambda function
Scheduler Logs	`AWS::Logs::LogGroup`	CloudWatch Log Group for Instance Scheduler
Scheduler Policy	`AWS::IAM::Policy`	Policy that allows scheduler to perform start and stop actions, change Amazon EC2 instance attributes, set tags, and access scheduler resources

Resource Name	Type	Description
Scheduler Rule	`AWS::Events::Rule`	Amazon CloudWatch event rule that invokes the scheduler's Lambda function
State Table	`AWS::DynamoDB::Table`	DynamoDB table that stores last desired state of instances
Config Table	`AWS::DynamoDB::Table`	DynamoDB table that stores global configuration, schedule, and period data
Config Table Auto Scaling Read Target	`AWS::ApplicationAutoScaling ::ScalableTarget`	Auto Scaling target for configuration table read capacity
Config Table Auto Scaling Write Target	`AWS::ApplicationAutoScaling ::ScalableTarget`	Auto Scaling target for configuration table write capacity
Configuration Table Auto Scaling Read Policy	`AWS::ApplicationAutoScaling ::ScalablePolicy`	Auto Scaling policy for configuration table read capacity
Configuration Table Auto Scaling Write Policy	`AWS::ApplicationAutoScaling ::ScalablePolicy`	Auto Scaling policy for configuration table write capacity
State Table Auto Scaling Read Target	`AWS::ApplicationAutoScaling ::ScalableTarget`	Auto Scaling target for state table read capacity
State Table Auto Scaling Write Target	`AWS::ApplicationAutoScaling ::ScalableTarget`	Auto Scaling target for state table write capacity
State Table Auto Scaling Read Policy	`AWS::ApplicationAutoScaling ::ScalablePolicy`	Auto Scaling policy for state table read capacity
State Table Auto Scaling Write Policy	`AWS::ApplicationAutoScaling ::ScalablePolicy`	Auto Scaling policy for state table write capacity
Instance Scheduler SNS Topic	`AWS::SNS::Topic`	Sends warning and error messages to subscribed email addresses

Appendix C: Log Files

The AWS Instance Scheduler creates a log group that contains the default AWS Lambda log files and a log group that contains the following log files:

- `InstanceScheduler-yyyymmdd`: Logs general scheduler messages
- `CloudWatchEventHandler-yyyymmdd`: Logs general Amazon CloudWatch event rule information
- `SchedulerSetupHandler`: Logs the output of configuration actions
- `Scheduler-<service>-<account>-<region>-yyyymmdd`: Logs the service, account, and region of each scheduled instance

- `AdminCliRequestHandler-yyyymmdd`: Logs requests from the admin CLI

Appendix D: Custom Resource

You can use the Instance Scheduler's AWS CloudFormation custom resource (`ServiceInstanceSchedule`) to configure schedules and periods. The custom resource uses the same names and syntax as the fields in the configuration Amazon DynamoDB table. For more information on the fields for schedules, see Schedule Definitions. For more information on the fields for periods, see Period Definitions.

When using the custom resource to configure a schedule or period, you must include the account, AWS Region, and name of the deployed schedule stack in the `ServiceToken` field.

The following sample custom resource code shows a schedule (`OfficehoursNL`) that contains two periods.

```
"OfficehoursNL": {
    "Type": "Custom::ServiceInstanceSchedule",
    "Properties": {
        "Description": "Weekdays in NL",
        "ServiceToken":
"arn:aws:lambda:<region>:<accounted>:function:<scheduler-stackname>-
InstanceSchedulerMain",
        "Enforced": "True",
        "Timezone": "Europe/Amsterdam",
        "Periods": [
            {
                "Description": "Opening hours on weekdays",
                "BeginTime": "09:00",
                "EndTime": "18:00",
                "WeekDays": "Mon-Fri"

            },
            {
                "Description": "Opening hours in weekend",
                "BeginTime": "10:00",
                "EndTime": "16:00",
                "WeekDays": "Sat-Sun"

            }
        ]
    }
},
```

Do not use the DynamoDB console or scheduler CLI to delete or modify schedules and periods that were configured using the custom resource. If you do, you will create a conflict between the stored parameters in the stack and the values in the table. Also, do not use

periods configured using the custom resource in schedules created using the DynamoDB console or the scheduler CLI.

Before you delete the main Instance Scheduler stack, you must delete all additional stacks that contain schedules and periods created using the custom resource because the custom resource stacks contain dependencies on the main stack's DynamoDB table.

In the configuration DynamoDB table, schedules and periods that were configured with the custom resource can be identified by the **configured_in_stack** attribute. The attribute contains the Amazon Resource Name of the stack that was used to create the item.

Appendix E: Collection of Anonymous Data

This solution includes an option to send anonymous usage data to AWS. We use this data to better understand how customers use this solution to improve the services and products that we offer. When enabled, the following information is collected and sent to AWS each time the Instance Scheduler AWS Lambda function runs:

- **Solution ID:** The AWS solution identifier
- **Unique ID (UUID):** Randomly generated, unique identifier for each solution deployment
- **Timestamp:** Data-collection timestamp
- **Instance Data:** Count of the state and type of instances that are managed by the Instance Scheduler in each AWS Region
  Example data:

```
Running: {t2.micro: 2}, {m3.large:2}
Stopped: {t2.large: 1}, {m3.xlarge:3}
Resized: {t2.large-t2.small:3}
```

The following information is collected and sent to AWS at stack creation or deletion:

- **Solution ID:** The AWS solution identifier
- **Unique ID (UUID):** Randomly generated, unique identifier for each solution deployment
- **Timestamp:** Data-collection timestamp
- **Stack Version:** The version of the stack that is created or deleted
- **Status:** The status of the stack (`stack_created` or `stack_deleted`)

- **Region:** The region where the stack is deployed

Note that AWS will own the data gathered via this survey. Data collection will be subject to the AWS Privacy Policy. To opt out of this feature, set the **Send anonymous usage data** parameter to `No`.

Send Us Feedback

We welcome your questions and comments. Please post your feedback on the AWS Solutions Discussion Forum.

You can visit our GitHub repository to download the templates and scripts for this solution, and to share your customizations with others.

Document Revisions

Date	Change	In sections
February 2018	Initial release	--

Notices

This document is provided for informational purposes only. It represents AWS's current product offerings and practices as of the date of issue of this document, which are subject to change without notice. Customers are responsible for making their own independent assessment of the information in this document and any use of AWS's products or services, each of which is provided "as is" without warranty of any kind, whether express or implied. This document does not create any warranties, representations, contractual commitments, conditions or assurances from AWS, its affiliates, suppliers or licensors. The responsibilities and liabilities of AWS to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.

The AWS Instance Scheduler solution is licensed under the terms of the Amazon Software License available at https://aws.amazon.com/asl/.