# AWS Cloud Non production Instance Schedule Start/Stop

**EXECUTIVE SUMMARY:** This manual introduces help in cost reduction by stopping non production instance in Non-working hours.

**The Western Union Company**

**AWS non production instance start and Stop Policy**

**Document Owner:** Cloud Enablement Team

**Effective Date:** March 2018

**Approved By:** SVP, Technology Operations

**Revision History Table:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision**  **Number** | **Month/Year of**  **Change(s)/Approval** | **Author(s) Brief Description of Change** | **Brief Description of Change** |
| 1.0 | July 2018 | Navneet N Rathi |  |
| 1.0 | july 2018 | Rohit Kamle |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

To further reduce costs of our EC2 instances, we determined some instances that are running 24/7 unnecessarily. An automated process to schedule stop and start instances would greatly help cutting costs. Of course, the solution itself should not add an extra instance to our infrastructure. The solution must be an example of serverless computing.

Automatically stop running EC2 instances 24/7 unnecessarily

We created a lambda function that scans all instances for a specific tag. The tag we use is named ‘Schedule’ and contains the desired ‘runtime’ for the specific instance. Instances without a Schedule tag will not be affected. We support the following content format in the Schedule tag:

08:00-19:15   **start** the instance at 08:00, **stop** it at 19:15  
21:00-07:45   **start** the instance at 21:00, **stop** it at 07:45 the next day  
-17:45              **stop** this instance at 17:45 today  
-16:30T           **Terminate** instance at 16:30 today  
# whatever     Anything starting with a # is totally ignored

**NOTE**: All times are UTC times!

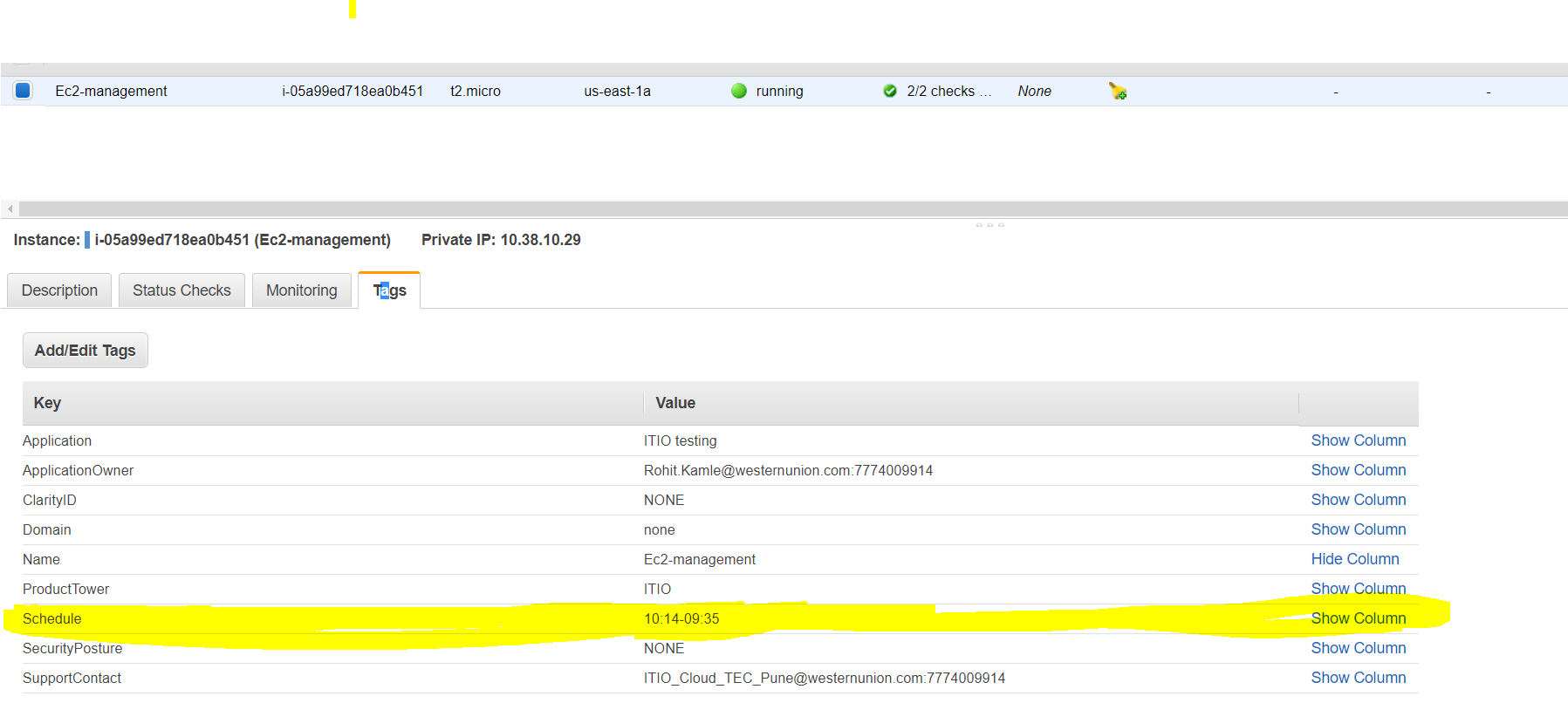
Invalid tag-content is automatically prefixed with an # so that it is ignored in future invocations of the lambda function. Also instances with only an end-time will have their tag rewritten to avoid restarts at 00:00.

Our lambda function is written in python using boto3 for AWS integration. The function requires a role to be able to interact with EC2. The lambda function must be able to read and change tags, stop and start instances and even terminate them!

To further automate the process, we need to automatically execute the lambda function, which can be achieved by creating a CloudWatch Event Rule. It is a bit of a funny place for a schedule, but that’s where you configure it. The event will be used as a trigger to start our lambda function.

The scheduler we create in the stack runs the lambda function every 10 minutes, giving your EC2 scheduler a granularity of 6 runs per hour.

As shown in the below screenshot One need to just add the tag “Schedule” and required timing against it and it will start and stop the instance at the given time.



Note:

1. All the services which you want by default to start needs to be added in the startup script as per the Operating system
2. Due to some issue if your instances does not startup at the give time you can raise a SR to **Enterprise Cloud Enablement 🡪 Cloud Request** And the team will help you out.
3. If your instance is running in autoscaling group Don’t schedule it as its going to terminate the instance and create a new instance from the base AMI provided in the launch configuration.