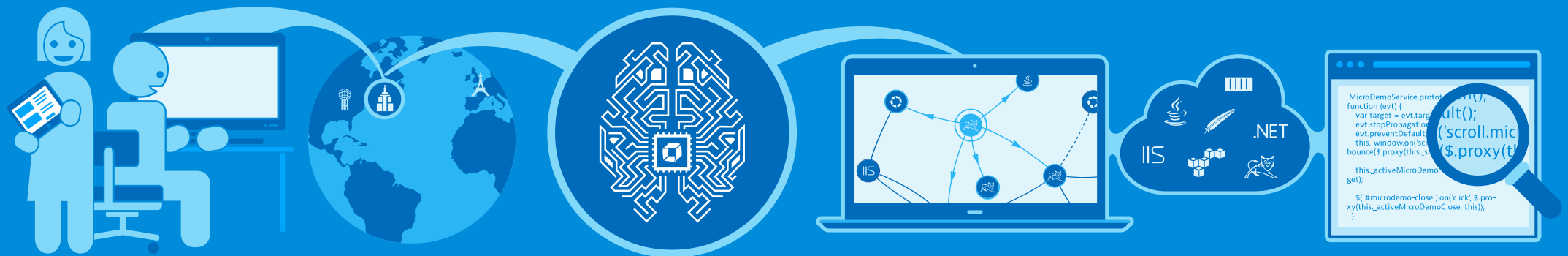




# AWS “Hands-On” with Dynatrace

26 Oct 2017



# Agenda

## Steps of the exercise

- Sign up for Dynatrace free trial (if you do not have one)
- Hands on #1  
Setup Dynatrace AWS Monitoring Integration
- Hands on #2  
Deploy and Monitor NodeJS Beanstalk Application
- Hands on #3  
Deploying EC2 instance with a OneAgent



## Pre-Requisites

- Dynatrace SaaS or Managed Account
- AWS Account with administrative privileges
- For Troubleshooting
  - Create a Key Pair for EC2.
  - Download your key.pem
  - More instructions to Connect to EC2 via Linux (SSH), Windows (PuTTY):  
<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AccessingInstances.html>

# AWS Hands On 1

Goals: Setup Dynatrace AWS Monitoring Integration

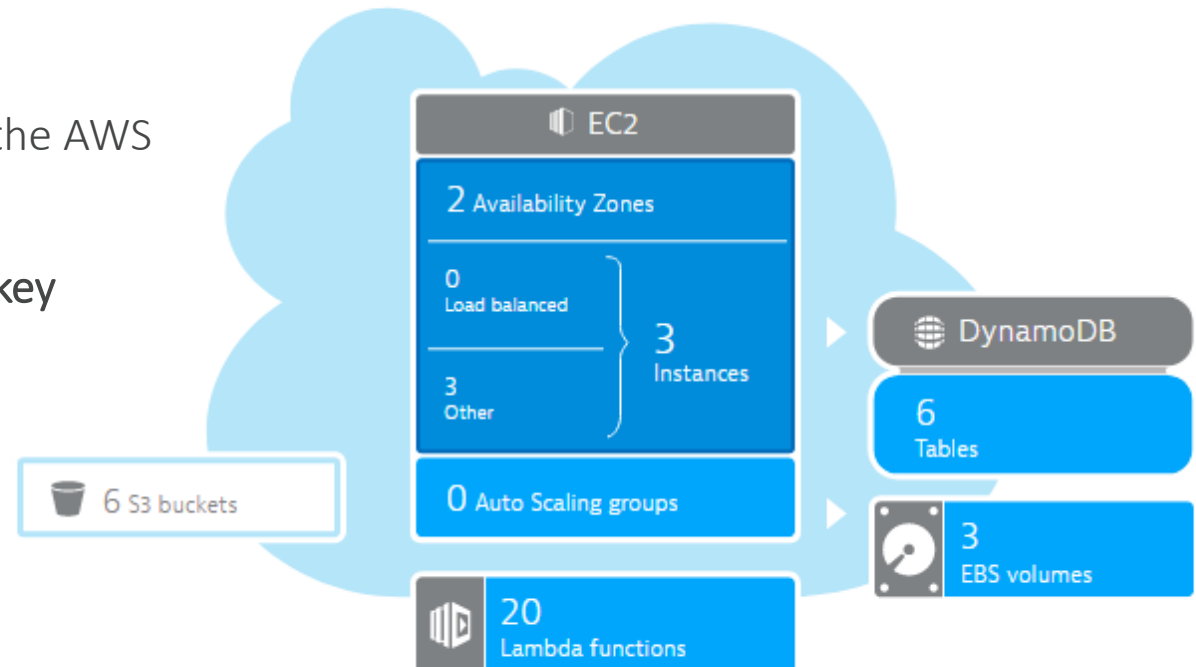
## Goal: See AWS CloudWatch Data in Dynatrace

- Approach: Access Cloud Watch
  - via **Access Key-based Authentication** or
  - via Role-Based Authentication



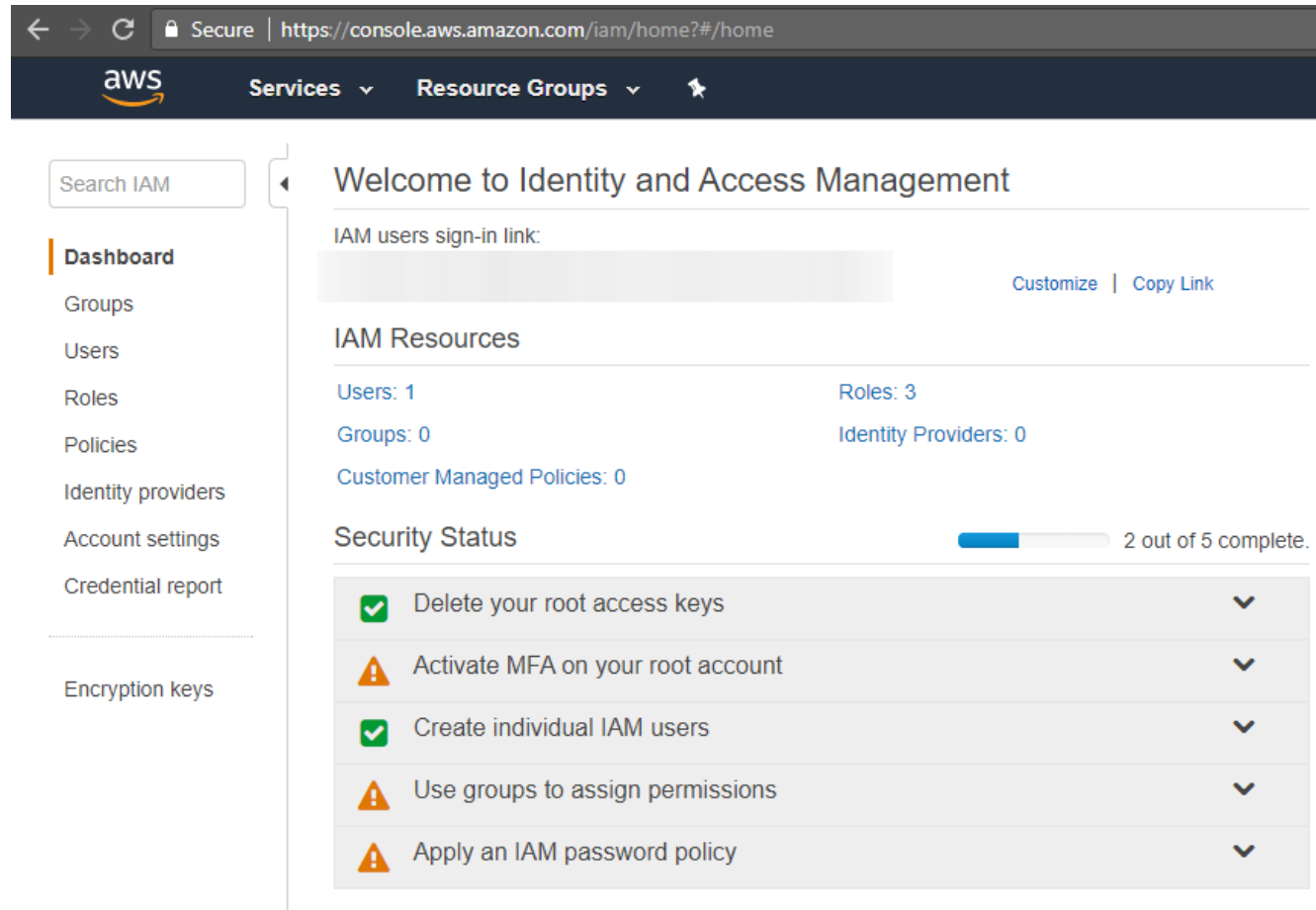
MyAWSConnection

- Key-based Authentication
  - Secure REST or Query protocol requests to the AWS service API
  - Generate **Access key ID** and a **Secret access key**



## Step 1 – Create a user in IAM

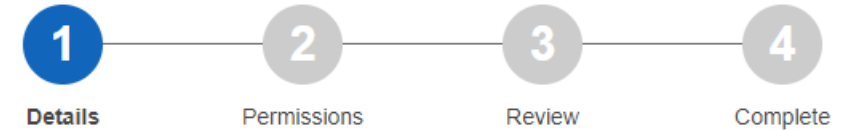
- Go to Identity and Access Management (IAM) in your Amazon Console



## Step 2

- Go to **Users** and click **Add Users**
- Enter a User name
- Select “**Programmatic access**” for Access type
- Click on Next

### Add user



#### Set user details

You can add multiple users at once with the same access type and permissions. [Learn more](#)

**User name\***

[+ Add another user](#)

#### Select AWS access type

Select how these users will access AWS. Access keys and autogenerated passwords are provided in the last step. [Learn more](#)

**Access type\*** ☒ **Programmatic access**  
Enables an **access key ID** and **secret access key** for the AWS API, CLI, SDK, and other development tools.

☐ **AWS Management Console access**  
Enables a **password** that allows users to sign-in to the AWS Management Console.

\* Required

[Cancel](#)

[Next: Permissions](#)

- Set Permissions
- Check **AdministratorAccess**
- Click Next to Review
- Click **Create User**

## Add user



## Review

Review your choices. After you create the user, you can view and download the autogenerated password and access key.

### User details

**User name** DynatraceAWS  
**AWS access type** Programmatic access - with an access key

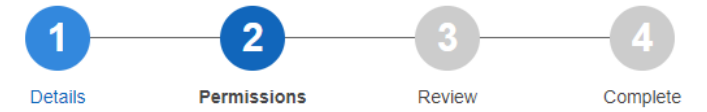
### Permissions summary

The following policies will be attached to the user shown above.

Type	Name
Managed policy	AdministratorAccess

[Cancel](#) [Previous](#) [Create user](#)

## Add user



### Set permissions for DynatraceAWS



Add user to group



Copy permissions from existing user



Attach existing policies directly

Attach one or more existing policies directly to the users or create a new policy. [Learn more](#)

[Create policy](#) [Refresh](#)

Filter: Policy type Search Showing 276 results

	Policy name	Type	Attachments	Description
<input checked="" type="checkbox"/>	AdministratorAccess	Job function	0	Provides full access to AWS services and r...
<input type="checkbox"/>	AmazonAPIGatewa...	AWS managed	0	Provides full access to create/edit/delete A...
<input type="checkbox"/>	AmazonAPIGatewa...	AWS managed	0	Provides full access to invoke APIs in Am...



## Step 3

- Copy out the values of **Access Key ID** and **Secret access key**
- Alternatively, you can download the .csv file and copy it from there

### Add user

The screenshot shows the 'Add user' process in Dynatrace, with a progress bar indicating four steps: 1 Details, 2 Permissions, 3 Review, and 4 Complete. The 'Complete' step is highlighted in blue. Below the progress bar, there is a 'Download .csv' button. A table displays the user 'DynatraceAWS' with its 'Access key ID' and 'Secret access key'. The 'Secret access key' is masked with asterisks, and a 'Show' link is provided to reveal it. A red arrow points to the 'Show' link. A 'Close' button is located in the bottom right corner.

1 Details 2 Permissions 3 Review 4 Complete

Download .csv

User	Access key ID	Secret access key
▶ ✓ DynatraceAWS	AKIAI44QH8DHBEXAMPLE	***** <a href="#">Show</a>

Close

## Step 4

- In **Permissions** expand **+Add Inline Policies**

Search IAM

Dashboard

Groups

**Users**

Roles

Policies

Identity providers

Account settings

Credential report

Encryption keys

Users > DynatraceAWS

### Summary

Permissions Groups (0) Security credentials Access Advisor

**Add permissions** Attached policies: 1

Policy name ▼	Policy type ▼
Attached directly	
▶ AdministratorAccess	AWS managed policy

**+ Add inline policy**

## Step 5

- In the **Set Permissions** section, select **Custom Policy**
- Create the policy by clicking on **Select**

### Manage User Permissions

## Set Permissions

Select a policy template, generate a policy, or create a custom policy. A policy is a document that formally states one or more permissions. You can edit the policy on the following screen, or at a later time using the user, group, or role detail pages.

☐ **Policy Generator**

☒ **Custom Policy**

Use the policy editor to customize your own set of permissions.

**Select**

## Step 6

- In the **Policy Name** field, type a name for the policy (for example DynatraceAWS-policy)
- In the **Policy Document** field, copy out the contents from **aws-policy.json** and paste it
- Click on **Apply Policy**

Manage User  
Permissions

### Review Policy

Customize permissions by editing the following policy document. For more information about the access policy language, see [Overview of Policies](#) in the *Using IAM* guide. To test the effects of this policy before applying your changes, use the [IAM Policy Simulator](#).

#### Policy Name

DynatraceAWS-policy

#### Policy Document

```
1 {  
2   "Version": "2012-10-17",  
3   "Statement": [  
4     {  
5       "Effect": "Allow",  
6       "Action": [  
7         "autoscaling:Describe*",  
8         "cloudwatch:Describe*",  
9         "cloudwatch:Get*",  
10        "cloudwatch:List*",  
11        "ec2:Describe*",  
12        "elasticloadbalancing:Describe*",  
13        "rds:DescribeDBInstances",  
14        "rds:DescribeEvents",  
15        "rds:List*",  
16        "dynamodb:DescribeTable",  
17        "dynamodb:ListTables",  
18        "lambda:ListFunctions",  
19        "lambda:GetFunction",  
20        "elasticbeanstalk:DescribeEnvironments",  
21        "elasticbeanstalk:DescribeEnvironmentResources",  
22        "s3:ListAllMyBuckets",  
23        "sts:GetCallerIdentity"  
24      ]  
25    }  
26  ]  
27 }
```

☒ Use autoformatting for policy editing

Cancel

Validate Policy

Apply Policy

## Step 7 – Connect your Amazon account to Dynatrace

- Go to Settings > Cloud & virtualization > AWS & VMware and click Connect new instance

The screenshot displays the Dynatrace Settings interface. The left sidebar contains a list of navigation items: Dashboards & reports, Dashboards, Create custom chart, Reports, Analyze, Problems, User sessions, Log files, Smartscape topology, Diagnostic tools, Monitor, Applications, Synthetic availability, Transactions & services, Databases, Hosts, Network, Technologies, VMware, AWS, Docker, Manage, Deploy Dynatrace, Deployment status, and Settings. The 'Settings' item is highlighted with a red box. A red arrow points from 'Settings' to the 'Cloud and virtualization' section in the main content area. The 'Cloud and virtualization' section is also highlighted with a red box. Within this section, the 'AWS' sub-item is highlighted with a red box. A red arrow points from 'AWS' to the '+ Connect new instance' button, which is also highlighted with a red box. The main content area shows the 'Settings' page with tabs for 'Settings', 'Cloud and virtualization', and 'AWS'. The 'AWS' tab is active, displaying instructions on how to connect an Amazon account to Dynatrace for monitoring. The instructions mention that Amazon may charge \$0.01 per 1,000 requests for CloudWatch API access, only when the number of requests exceeds 1 million. The '+ Connect new instance' button is located at the bottom of the AWS section.

Dashboards & reports  
Dashboards  
Create custom chart  
Reports  
Analyze  
Problems  
User sessions  
Log files  
Smartscape topology  
Diagnostic tools  
Monitor  
Applications  
Synthetic availability  
Transactions & services  
Databases  
Hosts  
Network  
Technologies  
VMware  
AWS  
Docker  
Manage  
Deploy Dynatrace  
Deployment status  
Settings

Settings > Cloud and virtualization > AWS

### Settings

Monitoring  
Setup and overview

Monitoring overview  
Process group detection  
Monitored technologies

Web and mobile monitoring  
Global settings and configuration

Cloud and virtualization  
Connect vCenter or Amazon account

Overview  
AWS  
VMware

Server-side service monitoring  
Manage & customize service monitoring

Log analytics  
Customize detection of log-based events

Anomaly detection  
Configure detection sensitivity

Alerting  
Configure alerting settings

Integration

**AWS**

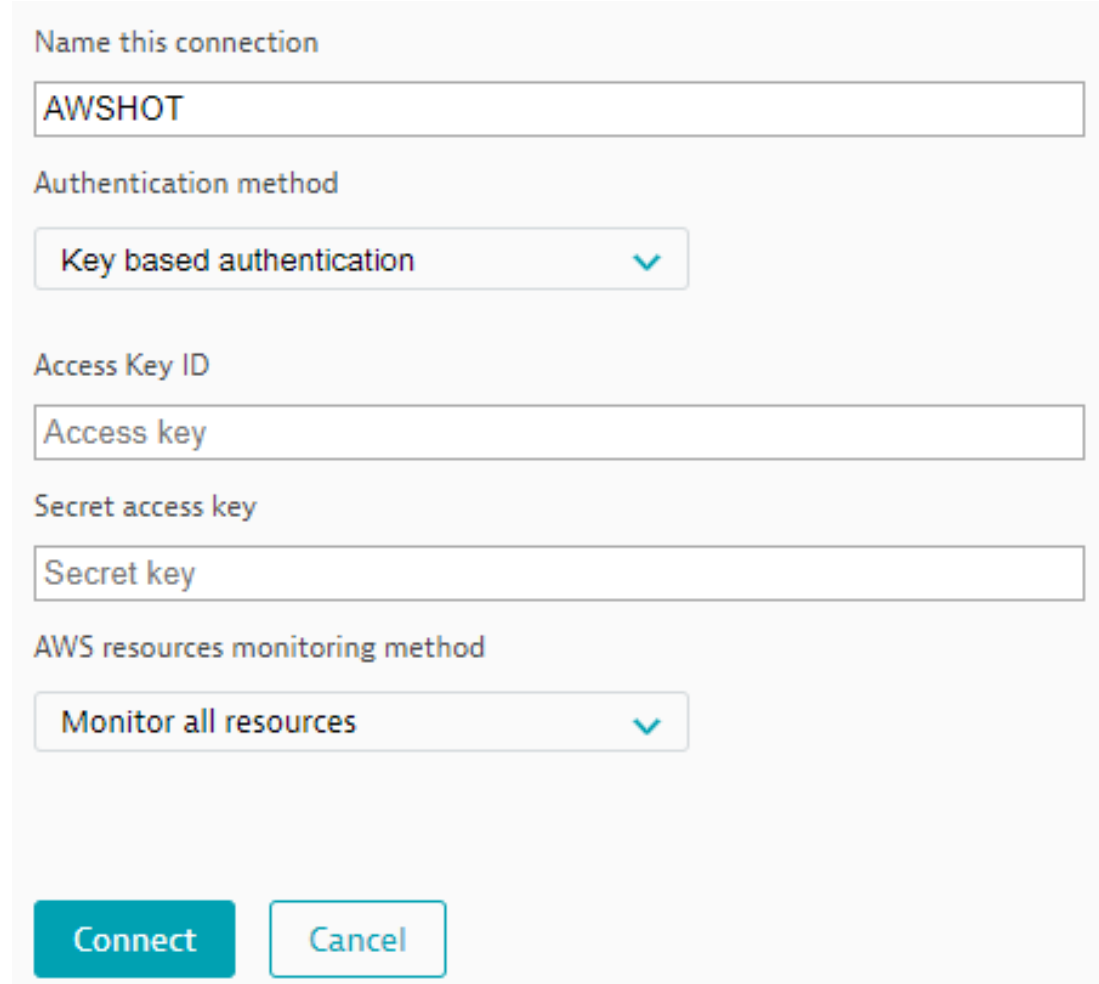
Use this page to connect Amazon account to Dynatrace for monitoring. For AWS instances, go to your Amazon Console and prepare access for Dynatrace using either key- or role-based authentication. For details, see [How do I start AWS monitoring?](#)

**Amazon may charge \$0.01 per 1,000 requests for CloudWatch API access - only when the number of requests exceeds 1 million. Amazon will then charge you for these requests and include the cost in the bill for the AWS account you use with Dynatrace.**

[+ Connect new instance](#)

## Step 8

- Create a name for this connection. This is mandatory. Dynatrace needs this name to identify and display the connection.
- In the Access key ID field, paste the key you created in Amazon for Dynatrace access.
- In the Secret access key field, paste the key you created in Amazon for Dynatrace access.
- Click Connect to verify and save the connection.



The screenshot shows a web form for configuring a connection. It has four main sections: 'Name this connection' with a text input containing 'AWSHOT'; 'Authentication method' with a dropdown menu showing 'Key based authentication'; 'Access Key ID' with a text input containing 'Access key'; and 'Secret access key' with a text input containing 'Secret key'. Below these is another dropdown for 'AWS resources monitoring method' showing 'Monitor all resources'. At the bottom are two buttons: 'Connect' (solid teal) and 'Cancel' (teal outline).

Name this connection

AWSHOT

Authentication method

Key based authentication

Access Key ID

Access key

Secret access key

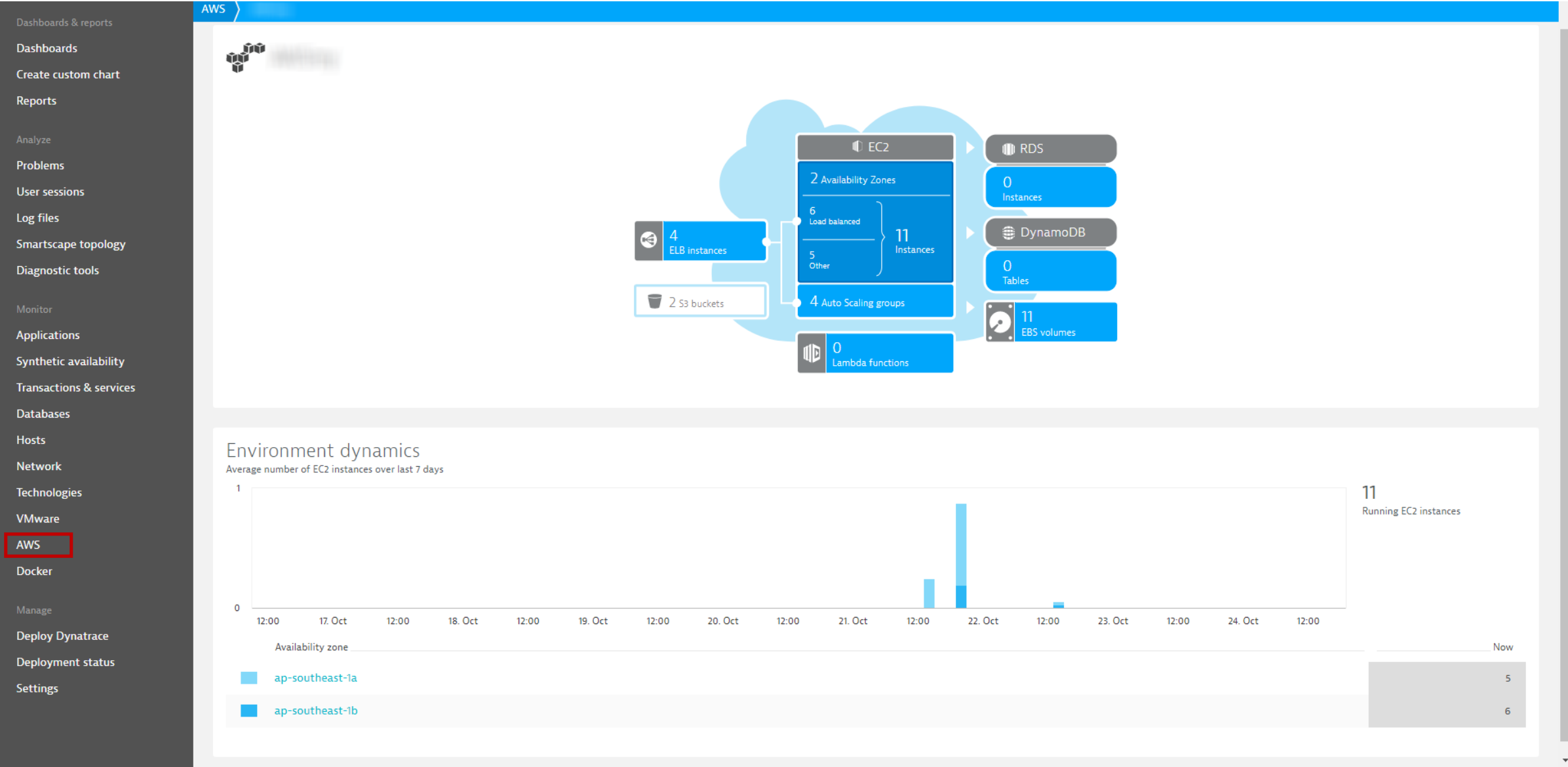
Secret key

AWS resources monitoring method

Monitor all resources

Connect Cancel

# End result



# Cloud Delivery Models

Traditional on  
Premise

Applications
Data
Runtime
Middleware
OS
Virtualization
Servers
Storage
Networking

Infrastructure-as-a-  
Service

Applications
Data
Runtime
Middleware
OS
Virtualization
Servers
Storage
Networking

Platform-as-a-  
Service

Applications
Data
Runtime
Middleware
OS
Virtualization
Servers
Storage
Networking

Software-as-a-  
Service

Applications
Data
Runtime
Middleware
OS
Virtualization
Servers
Storage
Networking

- Client Manages
- Vendor Manages in Cloud

Standardization – Lower Costs –Faster Time to Value



# AWS Hands On 2

Goals: Deploy and Monitor NodeJS Beanstalk Application

# 101 on BeanStalk

- What is Beanstalk?
  - AWS Hosts and Scales Application Servers (Node.JS, Java, ...) for you
  - Developer only provides ZIP or WAR file
  - ZIP/WAR File gets deployed on Application Server during startup
  - Automatically Load Balanced / Auto-Scaling ...
- Beanstalk ZIP File Structure

Name	Date modified
.ebextensions	26.06.2017 14:04
app.js	27.06.2017 17:04
cron.yaml	17.02.2015 23:37
index.html	27.06.2017 17:23
package.json	17.02.2015 23:37

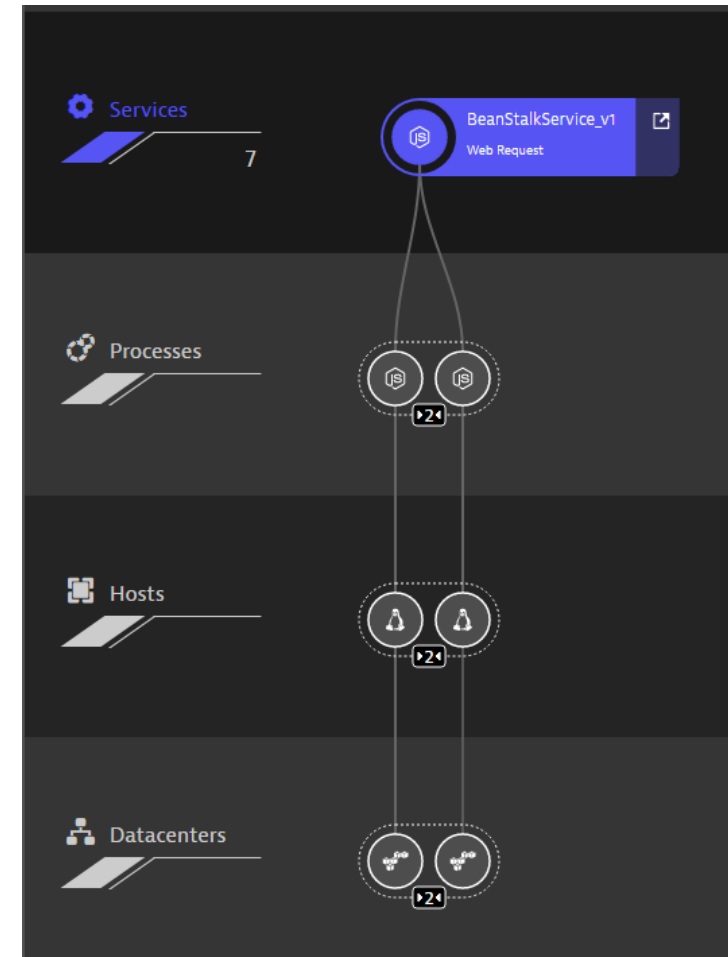
Name	Date modified
dynatrace.config	26.06.2017 14:28
logging.config	17.02.2015 23:37
version.config	27.06.2017 00:56

```
---
option_settings:
  - option_name: DYNATRACE_ONEAGENT_DOWNLOAD
    value: YOURFULLONEAGENTDOWNLOADURL

files:
  /opt/elasticbeanstalk/hooks/appdeploy/pre/01_dynatrace_install.sh:
    content: |
      DYNATRACE_ONEAGENT_DOWNLOAD=$( /opt/elasticbeanstalk/bin/get-c
      if [ \"x$DYNATRACE_ONEAGENT_DOWNLOAD\" == \"x\" ]
      then
        DYNATRACE_ONEAGENT_DOWNLOAD=$( /opt/elasticbeanstalk/bin/get
      fi
```

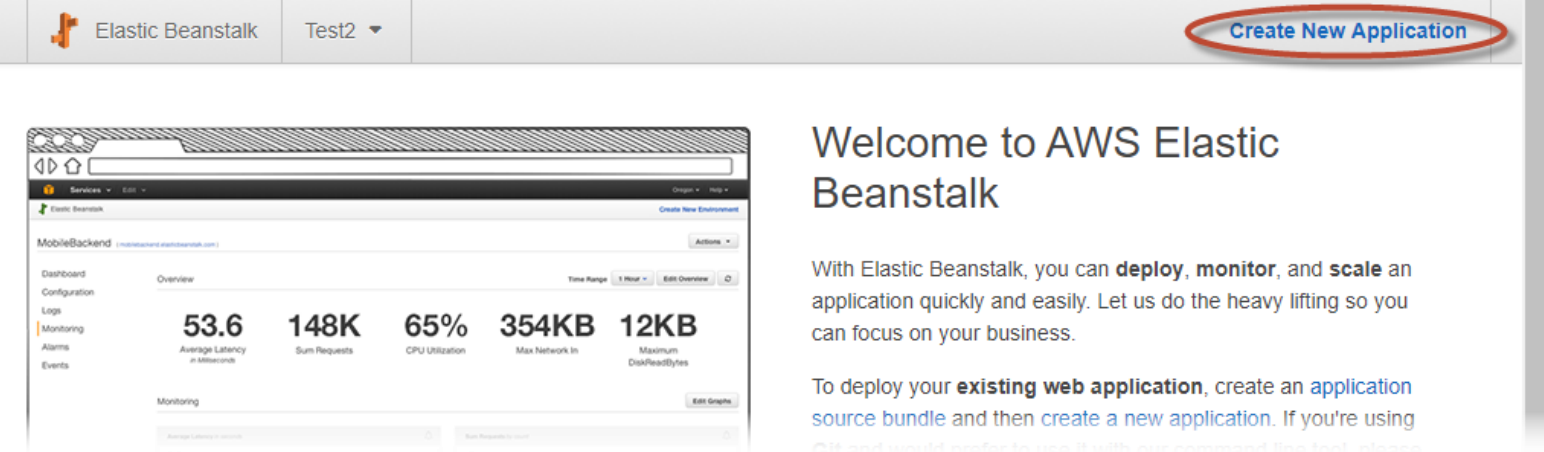
## Goal: See Full-Stack Tracing of Node.JS Beanstalk Application

- Using .ebextension mechanism to
  - Execute Agent Download and Install during EC2 Launch
  - Pass additional tags, e.g: BeanStalkVersion\_v2
- **IMPORTANT**
  - Mac: make sure .zip file contains **.ebextension** folder (any file with a "." is hidden)
  - Mac & Win: Make sure **package.json** is at the root of the .zip file, instead of nested in a folder
- Additional Hands-On Use Cases
  - Load Balancing / Auto-Scaling BeanStalk Service
  - RUM Configuration: User Tagging and jQuery Support



## Step 1 Deploy the sample App

- Logon to AWS and Navigate to Elastic Beanstalk
- Create a new application
- Give it a name. Then create an environment.
- Choose **Web Server** environment



The screenshot shows the AWS Elastic Beanstalk console. At the top, there's a header with the Elastic Beanstalk logo, a dropdown menu showing 'Test2', and a 'Create New Application' button circled in red. Below the header, there's a 'Welcome to AWS Elastic Beanstalk' message. To the left, there's a sidebar with navigation links: Dashboard, Configuration, Logs, Monitoring, Alarms, and Events. The main content area shows a 'MobileBackend' application with an 'Overview' tab selected. The overview displays several metrics: Average Latency (53.6), Sum Requests (148K), CPU Utilization (65%), Max Network In (354KB), and Maximum DiskReadBytes (12KB). Below these metrics, there's a 'Monitoring' section with a 'View Metrics' button.

### Welcome to AWS Elastic Beanstalk

With Elastic Beanstalk, you can **deploy**, **monitor**, and **scale** an application quickly and easily. Let us do the heavy lifting so you can focus on your business.

To deploy your **existing web application**, create an [application source bundle](#) and then [create a new application](#). If you're using `Git` and would prefer to use it with our command line tool, please



The screenshot shows the AWS Elastic Beanstalk console for a specific application named 'MyNodeApp1'. The breadcrumb navigation shows 'All Applications > MyNodeApp1'. On the left, there's a sidebar with navigation links: Environments, Application versions, and a 'Create New Application' button. The main content area shows a message: 'No environments currently exist for this application. [Create one now.](#)' The 'Create one now.' link is circled in red.

## Base configuration

Tier Web Server ([Choose tier](#))

Platform ☒ Preconfigured platform

Platforms published and maintained by AWS Elastic Beanstalk.

Node.js

☐ Custom platform <sup>NEW</sup>

Platforms created and owned by you. [Learn more](#)

-- Choose a custom platform --

Application code ☐ Sample application

Get started right away with sample code.


☐ Existing version

Application versions that you have uploaded for **MyNodeApp1**.

-- Choose a version --

☒ Upload your code

Upload a source bundle from your computer or copy one from Amazon S3.

 Upload

ZIP or WAR

[Cancel](#)

[Configure more options](#)

[Create environment](#)

- Select **Node.js** as the platform
- Upload **NodeJSBeanStalkSample.zip**
- Click on Configure more options



## Configure Mynodeapp1-env

Start from a new configuration that matches your use case. Choose Custom configuration to use a custom configuration. Use the default configuration to use the recommended configuration.

### Step 2

- Under Software section, click on **Modify**
- Add the exact text **DYNATRACE\_ONEAGENT\_DOWNLOAD**
- And copy the full download URL for your OneAgent from your own tenant
- Click on Save

**Software**

Node.js version: 6.11.1  
AWS X-Ray: disabled  
Rotate logs: disabled (default)  
Log streaming: disabled (default)  
Environment properties: 0

**Instances**

EC2 instance type: t1.micro  
EC2 image ID: ami-ad2b5dce  
Root volume type: container default  
Root volume size (GB): container default  
Root volume IOPS: container default

**Modify**

**Environment properties**

The following properties are passed in the application as environment properties. [Learn more](#)

Name	Value
DYNATRACE_ONEAGENT_DOWNLOAD	<a href="https://1154534.live.ruxit.com/a">https://1154534.live.ruxit.com/a</a>

**Copy and paste from your tenant**

**Save**

- Click on Create Environment



## Configure Mynodeapp1-env

Start from a preset that matches your use case or choose *Custom configuration* to unset recommended values and use the service's default values.

### Database

Engine: --  
Instance class: --  
Storage (GB): --  
Multi-AZ: --

Modify

### Tags

Tags: none

Modify

Cancel

Previous

Create environment



All Applications > MyNodeApp1 > Mynodeapp1-env (Environment ID: XXXXXXXXXX)

**Actions** ▼



## Creating Mynodeapp1-env

This will take a few minutes.

7:20pm Environment health has transitioned to Pending. Initialization in progress (running for 10 seconds). There are no instances.

7:20pm Created EIP: 10-31-14 14:44

7:20pm Created security group named:

Copyright © 2006 John Wiley & Sons, Ltd.

7:19pm Using elasticbeanstalk-ap-southeast-1-603123023980 as Amazon S3 storage bucket for environment data.

```
7:19pm createEnvironment is starting.
```

[Learn More](#)

[Get started using Elastic Beanstalk](#)

Modify the code

## Create and connect to a database

[Add a custom domain](#)

## Featured

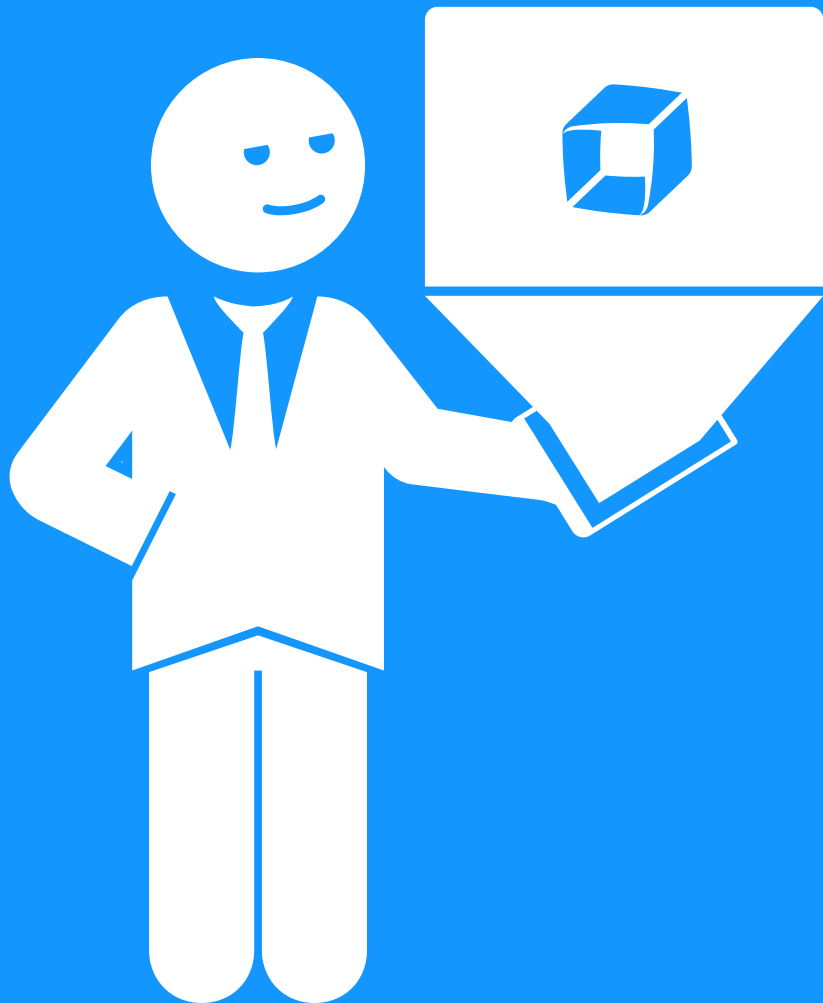
Create your own custom platform

## Command Line Interface (v3)

## Installing the AWS EB CLI

[EB CLI Command Reference](#)





Explore Dynatrace!



Elastic Beanstalk

MyNodeApp1 ▾

Test2 ▾

[Create New Application](#)

[All Applications](#) > [MyNodeApp1](#) > [Mynodeapp1-env](#) ( Environment ID: elb-8p8t1111, URL: [Mynodeapp1-env.eq7n9ppeaj.ap-southeast-1.elasticbeanstalk.com](http://Mynodeapp1-env.eq7n9ppeaj.ap-southeast-1.elasticbeanstalk.com) )

Actions ▾



### Creating Mynodeapp1-env

This will take a few minutes....

7:25pm Successfully launched environment: Mynodeapp1-env

7:25pm Environment health has transitioned from Pending to Ok. Initialization completed 28 seconds ago and took 5 minutes.

7:21pm Waiting for EC2 instances to launch. This may take a few minutes.

7:21pm Added instance i-01234567890123456 to your environment.

7:20pm Environment health has transitioned to Pending. Initialization in progress (running for 10 seconds). There are no instances.

7:20pm Created EIP: elb-8p8t1111

7:20pm Created security group named: sg-01234567890123456

7:19pm Using elasticbeanstalk-ap-southeast-1-603123023980 as Amazon S3 storage bucket for environment data.

7:19pm createEnvironment is starting.

### Learn More

[Get started using Elastic Beanstalk](#)  
[Modify the code](#)  
[Create and connect to a database](#)  
[Add a custom domain](#)

### Featured

[Create your own custom platform](#)

### Command Line Interface (v3)

[Installing the AWS EB CLI](#)  
[EB CLI Command Reference](#)



Feedback



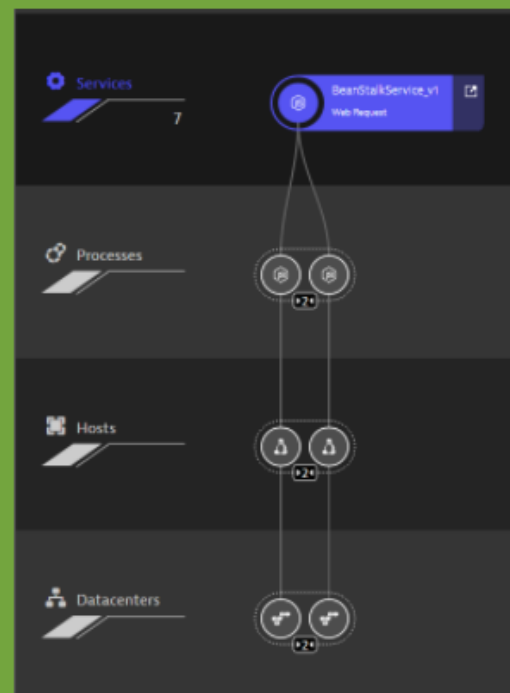
English (US)

© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved.

[Privacy Policy](#)

[Terms of Use](#)

# Congratulations



This might be your first AWS Beanstalk App you ever deployed. The code base of this application was taken from the AWS Elastic Beanstalk Tutorial but was slightly modified for the Dynatrace AWS Tutorial that you can find [here!](#) The goal of this tutorial is to enable Full Stack Monitoring with Dynatrace on your Beanstalk Application. Following is a screenshot of Dynatrace Smartcape in case you deploy this application in a scalability group. If you want to learn

## Lets trace some code!

Sleep Setting (in ms):

Say Something :

Invoke Server Side URL (full URL please) :

Your Username (can be used for user tagging) :

Request to 'https://www.amazon.com' returned with HTTP Status: 200 and response body length: 448779

## External Links: To learn more about Beanstalk

- [AWS Elastic Beanstalk overview](#)
- [AWS Elastic Beanstalk concepts](#)
- [Deploy an Express Application to AWS Elastic Beanstalk](#)
- [Deploy an Express Application with Amazon ElastiCache to AWS Elastic Beanstalk](#)
- [Deploy a Geddy Application with Amazon ElastiCache to AWS Elastic Beanstalk](#)
- [Customizing and Configuring a Node.js Container](#)
- [Working with Logs](#)

Search

🔍

- All1
- ▼ State
- Running1
- ▼ Monitoring mode
- Full-stack1

### All hosts

Monitor another host

1 host

Filtered by:

🔍

Name	State ▼	Host units	CPU usage	Memory usage	Disk latency	Network traffic
Mynodeapp1-env	running	0.1	41 %	31 % of 590 MB	81 ms	371 kbit/s



## Mynodeapp1-env

Uptime: 5 minutes

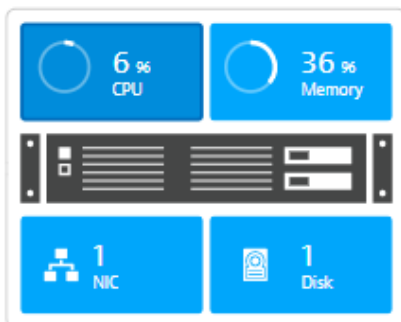
✓ Properties and tags

[AWS]Name: Mynodeapp1-env

Edit

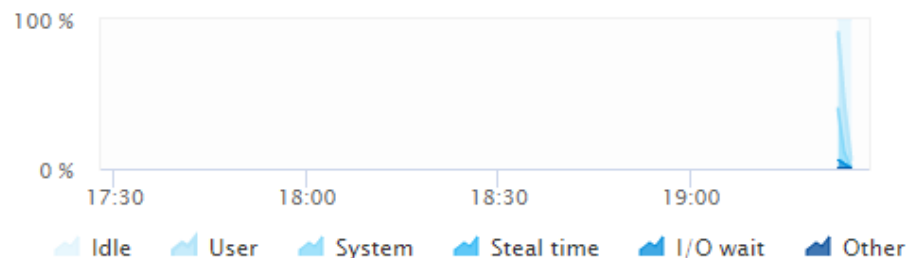
...

more...



CPU usage 5.82 %

Actively used CPU of the host as a percentage of available CPU.



CPU AWS

Consuming processes

No problems in last 72 hours

100% Availability

0 min total downtime

19:25

Running

## Processes



Ruby

command-processor  
healthd

Nginx

nginx



Node.js

NodeBeanstalk\_Version1  
NodeBeanstalk\_Version1

Go

xray

Restart 1 process to get full visibility

All processes

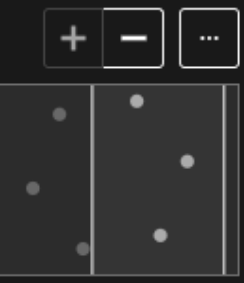
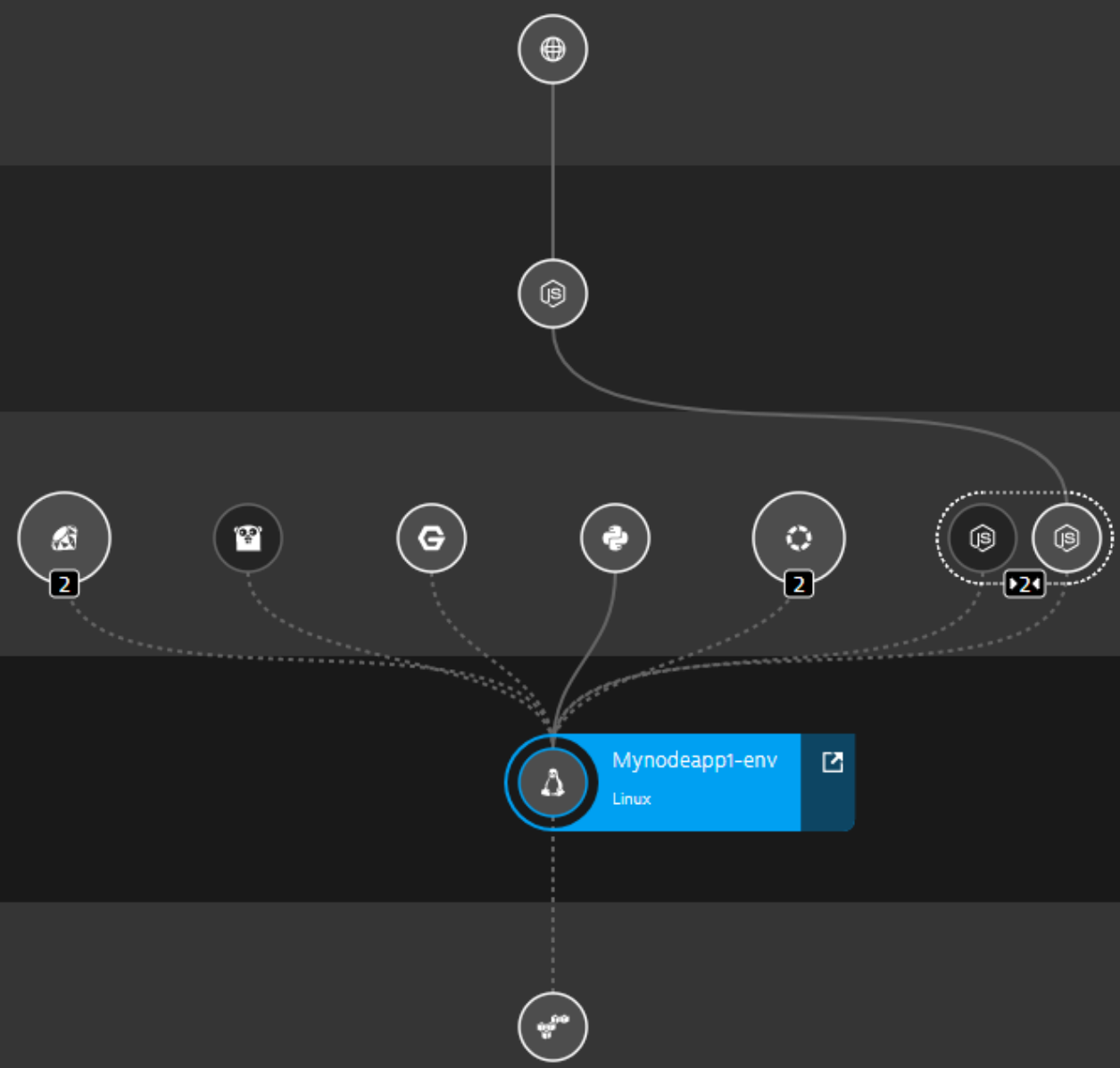
Applications

Services

Processes

Hosts 6

Data centers



## Additional step: Load Balancing

- Go to your Beanstalk Environment in AWS Console
- Click on Configuration – Scaling

The screenshot displays the AWS Elastic Beanstalk console interface. At the top, the breadcrumb navigation shows 'All Applications > MyNodeApp1 > Mynodeapp1-env', with 'Mynodeapp1-env' circled in red. Below the breadcrumb, the left-hand navigation menu includes 'Dashboard', 'Configuration' (circled in red), 'Logs', 'Health', 'Monitoring', and 'Alarms'. The main content area is titled 'Web Tier' and contains three panels: 'Scaling' (with a gear icon circled in red), 'Instances', and 'Notifications'. The 'Scaling' panel shows 'Environment type: Single instance' and 'Custom Availability Zones: blank'. The 'Instances' panel shows 'Instance type: t1.micro' and 'Availability Zones: Any'. The 'Notifications' panel shows 'Notifications: Off'.

## Additional step: Load Balancing

- Change the environment type to "Load balancing, auto scale"
- Apply the changes and let Beanstalk restart

Dashboard

Configuration

Logs

Health

Monitoring

Alarms

Managed Updates

Events

Tags

### Environment Type

The following settings configure the availability settings of your environment to help reduce the

Environment type: **Load balancing, auto scaling** [Learn more](#)

Current status: 1 instance(s) in service, Min: 1, Max: 1

Auto scaling and trigger settings will be available after updating your environment type.

#### ▶ Time-based Scaling

Cancel

Apply



## Additional step: Load Balancing

- Go back to the same settings after restart and change auto scale to **minimum instances of 2**
- Apply changes and validate that Dynatrace detects both instances

[All Applications](#) > [MyNodeApp1](#) > Mynodeapp1-env ( Environment ID: URL: Mynodeapp1-env.eq7n9ppeaj.ap-southeast-1.elasticbeanstalk.com ) [Actions](#) ▼

[Dashboard](#)  
[Configuration](#)  
[Logs](#)  
[Health](#)  
[Monitoring](#)  
[Alarms](#)  
[Managed Updates](#)  
[Events](#)  
[Tags](#)

### Environment Type

The following settings configure the availability settings of your environment to help reduce the costs for development activities.

Environment type:  [Learn more](#)

Current status: 1 instance(s) in service, Min: 1, Max: 4

### ▼ Auto Scaling

Use the following settings to control auto scaling behavior. [Learn more](#)

Minimum instance count:  Minimum number of instances to run.

Maximum instance count:  Maximum number of instances to run.

Availability Zones:  Number of Availability Zones to run in.

### ► Time-based Scaling

[Cancel](#) [Apply](#)

- Dashboard
- Configuration
- Logs
- Health
- Monitoring
- Alarms
- Managed Updates
- Events
- Tags

Overview Refresh



Health

Ok

Causes

Running Version

v1

Upload and Deploy



Configuration

64bit Amazon Linux 2017.03  
v4.3.0 running Node.js

Change

Recent Events Show All

Time	Type	Details
2017-10-24 20:05:32 UTC+0800	INFO	Environment health has transitioned from Info to Ok.
2017-10-24 20:02:32 UTC+0800	INFO	Environment health has transitioned from Ok to Info. Command is executing on 1 out of 2 instances.
2017-10-24 20:02:16 UTC+0800	INFO	Environment update completed successfully.
2017-10-24 20:02:16 UTC+0800	INFO	Successfully deployed new configuration to environment.
2017-10-24 20:01:32 UTC+0800	INFO	Added instance   to your environment.

Applications

Services 2

Processes

Hosts

Data centers



Search

All 2

State

Running 2

Monitoring mode

Full-stack 2

Type

EC2 2

Beanstalk environment

Mynodeapp1-env 2

Data centers

ap-southeast-1a 1

ap-southeast-1b 1

All hosts

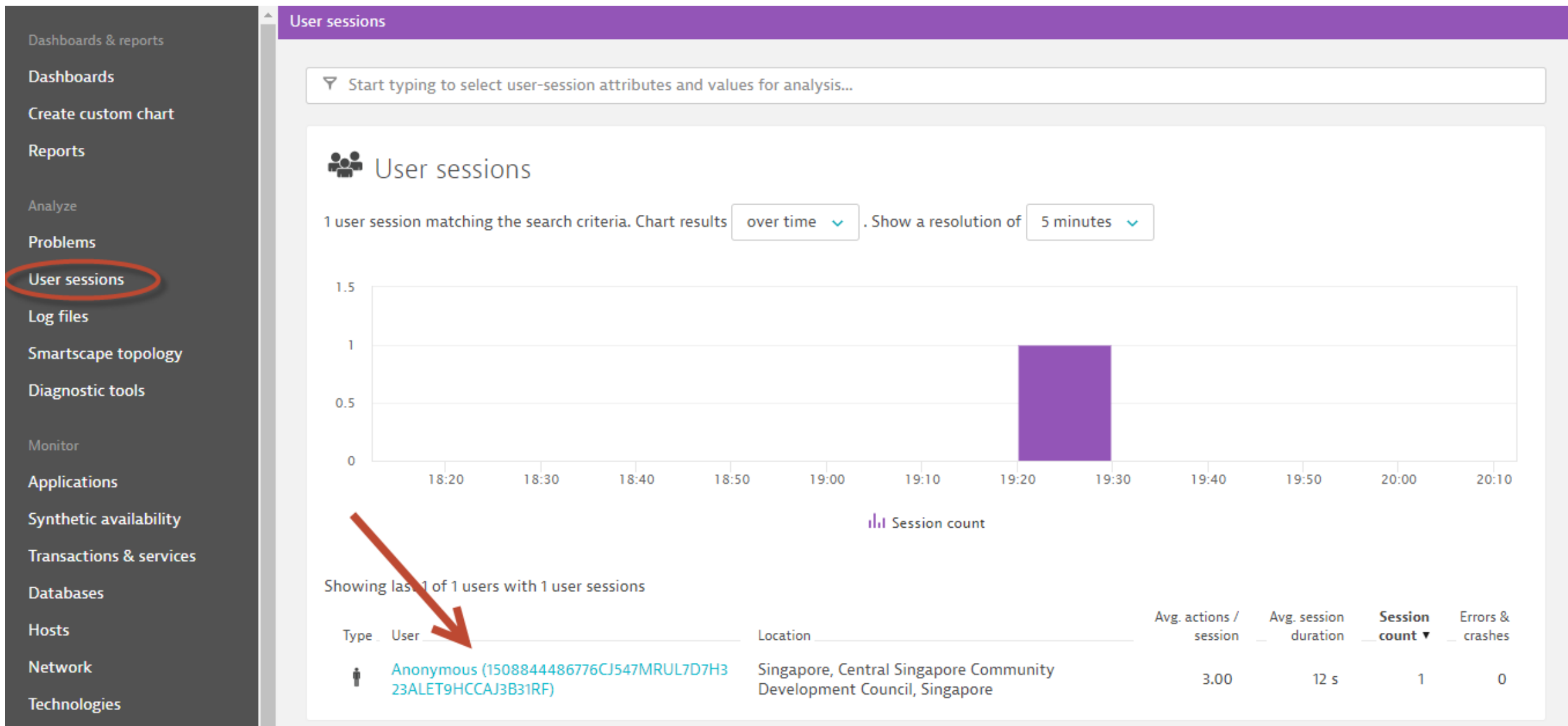
Monitor another host

2 hosts

Filtered by:

Name	State	Host units	CPU usage	Memory usage	Disk latency	Network traffic
Mynodeapp1-env	running	0.1	0.99 %	39 % of 590 MB	0.54 ms	21.9 kbit/s
Mynodeapp1-env	running	0.1	4.43 %	37 % of 590 MB	3.16 ms	90 kbit/s

# Explore User Sessions!



## Additional Step: Real User Monitoring

- Enable jQuery support for our Beanstalk Application
- My web application >> Settings >> Async requests and single page apps

The screenshot shows the Dynatrace dashboard for 'My web application'. The left sidebar contains navigation links: Dashboards & reports, Dashboards, Create custom chart, Reports, Analyze, Problems, User sessions, Log files, Smartscope topology, Diagnostic tools, Monitor, Applications (highlighted with a red circle), Synthetic availability, and Transactions & services. The main content area has a purple header with 'Applications' and 'My web application'. Below this, there's a 'Performance analysis' section with various metrics: 100% Chrome top browser, 100% React top user type, 2.2s user action duration, 0.1/min user actions, 0.90 apex rating, 0.0/min JavaScript errors, 0.3/action 3rd party/CDN resources, 1.0/action resources, and 1 service. A 'Filter user types' dropdown is visible. A red arrow points to the 'Edit' button in the top right corner of the main content area.

The screenshot shows the 'Async requests and single page apps' settings page. The breadcrumb trail at the top is 'Applications > My web application > Settings > Async requests and single page apps'. The page is divided into two main sections: 'Application settings' and 'Async requests and single page apps'. The 'Application settings' section includes 'General', 'User actions', 'Conversion goals', 'User tags', 'Content capture', 'Anomaly detection', and 'Advanced setup'. The 'Async requests and single page apps' section includes 'Dynatrace creates user actions when your app uses XMLHttpRequest or framework independent. More...', 'Capture fetch() requests', 'Support for XMLHttpRequest' (highlighted with a red arrow), 'Fetch API support requires Dynatrace OneAgent version 1.119 or later', 'JavaScript framework support', and a list of frameworks: 'AngularJS and Angular', 'Dojo', 'ExtJS, Sencha Touch', 'ICEfaces', 'jQuery, Backbone.js' (highlighted with a red arrow), and 'MooTools'. The 'Async requests and single page apps' section title is also circled in red.

## Additional Step: Real User Monitoring

- Configure Cookie Support for elasticbeanstalk.com domains
  - Advanced Settings
  - Specify your full domain name, e.g: custom-env.ub2cp9hmpy.us-west-2.elasticbeanstalk.com in the field Domain to be used for cookie placement
- Why? Browser will reject the Dynatrace Cookies necessary for end user tagging
- This step IS NOT necessary if you host your app on a "normal" domain!

The screenshot shows the Dynatrace 'Advanced setup' page for 'My web application'. The left sidebar lists various settings categories, with 'Advanced setup' circled in red. The main content area is titled 'Advanced settings' and includes a warning about advanced user settings. Key sections include:

- Cookie and header settings:** A toggle for 'Use the secure cookie attribute' is turned on. A note states it requires Dynatrace OneAgent version 1.87 or higher.
- Domain to be used for cookie placement:** A text input field contains the domain 'mynodeapp1-env.eq7n9ppeaj.ap-southeast-1.elasticbeanstalk.cc'.
- Cache control header optimizations:** A toggle is turned on, with a note requiring Dynatrace OneAgent version 1.21 or higher.

At the bottom, a message states: 'It will take a few minutes to update all Dynatrace OneAgent instances. You have unsaved changes!'. Below this message are 'Save' and 'Cancel' buttons. A red arrow points from the 'Save' button towards the bottom right of the page.

## Additional Step: Real User Monitoring

- Configure User Tagging
  - Application has a login button which will then set the Username to an HTML Element with the ID `#loggedinusername`
  - Use **CSS Selectors**
- NOTE: User Visits right now will show up once the Visits are completed which means after the 30 minutes timeout!

The screenshot displays the New Relic 'User tags' configuration page. The left sidebar shows the navigation menu with 'User tags' selected and circled in red. A red arrow points from this menu item to the 'Add user tag rule' button in the top right. The main content area shows the 'User tags' configuration form. A red circle highlights the 'CSS selector' dropdown menu, which is set to 'CSS selector'. Another red circle highlights the input field containing the CSS selector '#loggedinusername'. A red arrow points from the 'User tags' menu item in the sidebar to this input field. At the bottom of the form, a red arrow points to the 'Add user tag rule' button.

Applications > My web application > Settings > User tags

Application settings  
My web application

General  
See your application setup

User actions  
Generate consistent user action names

Conversion goals  
Measure success against business goals

User tags  
Use metadata to analyze user sessions

User tags

Set up rules to automatically tag the users of this application to gain deeper insight into the individual journeys of your users, user IDs, or other metadata captured via a browser session analysis.

Alternatively, you can use the JavaScript API call `nr.setPageMetadata()` to set the username or JavaScript variable. For details, see [JavaScript API documentation](#).

+ Add user tag rule No user tag rules

Applications > My web application > Settings > User tags

Application settings  
My web application

General  
See your application setup

User actions  
Generate consistent user action names

Conversion goals  
Measure success against business goals

User tags  
Use metadata to analyze user sessions

Async requests and single page apps  
Select your frameworks

Content capture  
Define resource monitoring

Anomaly detection  
Fine-tune the sensitivity and thresholds

User tags

Set up rules to automatically tag the users of this application to gain deeper insight into the individual journeys of your users, user IDs, or other metadata captured via a browser session analysis.

Alternatively, you can use the JavaScript API call `nr.setPageMetadata()` to set the username or JavaScript variable. For details, see [JavaScript API documentation](#).

Select the expression type that includes the metadata you want to capture:

Expression type to capture:  
CSS selector

CSS selector:  
#loggedinusername

☐ Apply cleanup rule

Add user tag rule Cancel

User sessions

Start typing to select user-session attributes and values for analysis...

User sessions

2 user sessions matching the search criteria. Chart results over time . Show a resolution of 5 minutes



Showing last 2 of 2 users with 2 user sessions

Type	User	Location	Avg. actions / session	Avg. session duration	Session count	Errors & crashes
	Anonymous (1508844486776CJ547MRUL7D7H323ALET9HCCAJ3B31RF)	Singapore, Central Singapore Community Development Council, Singapore	3.00	12 s	1	0
	Joe Mueller	Singapore, Central Singapore Community Development Council, Singapore	7.00	1 min 50 s	1	0



# AWS Hands On 3

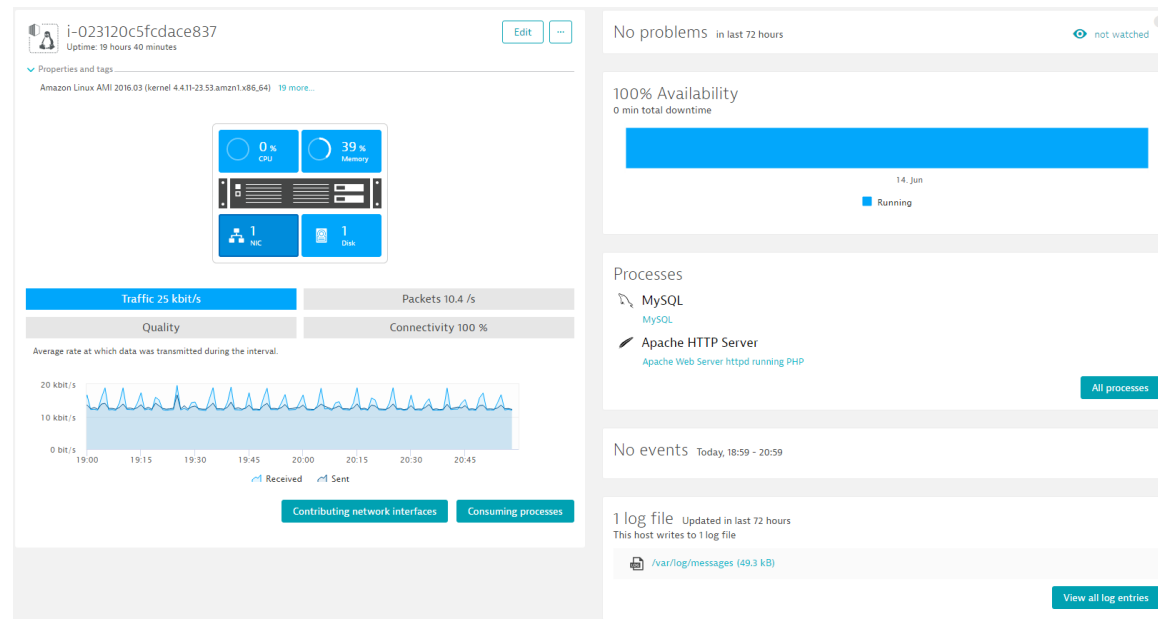
Goals: Deploying EC2 instance with a OneAgent

## Goal: See EC2 Host monitored by Dynatrace

- 2 different options available to deploy a Dynatrace OneAgent on EC2
  - Through Puppet, Chef, Ansible, AWS CodeDeploy
  - Through UserData which is a “Startup Script” executed when launching EC2 Instance
- We will be using UserData Option

```
#!/bin/bash
```

```
wget -O Dynatrace-OneAgent-Linux.sh https://YOUR.FULL.DYNATRACE.ONEAGENT.DOWNLOADLINK  
/bin/sh Dynatrace-OneAgent-Linux.sh APP_LOG_CONTENT_ACCESS=1 INFRA_ONLY=0
```



## Step 1

- Logon to AWS and navigate to EC2
- Select the option to **Launch a new Instance**
- Select Amazon Linux AMI and then select the free tier eligible **t2.micro instance type**

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

**Step 1: Choose an Amazon Machine Image (AMI)** [Cancel and Exit](#)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.

**Quick Start**

My AMIs

AWS Marketplace

Community AMIs

☐ Free tier only ⓘ

**Amazon Linux**  
**Free tier eligible**

**Amazon Linux AMI 2017.09.0 (HVM), SSD Volume Type** **Select**  
ami-0797ea64  
64-bit

The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, Ruby, Perl, and Java. The repositories include Docker, PHP, MySQL, PostgreSQL, and other packages.

Root device type: ebs Virtualization type: hvm

**Red Hat**

**Red Hat Enterprise Linux 7.4 (HVM), SSD Volume Type -** **Select**  
ami-10bb2373  
64-bit

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

**Step 2: Choose an Instance Type**

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. [Learn more](#) about instance types and how they can meet your computing needs.

<input type="checkbox"/>	General purpose	t2.nano	1	0.5	EBS only	-	Low to Moderate
<input checked="" type="checkbox"/>	General purpose	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	t2.small	1	2	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	t2.medium	2	4	EBS only	-	Low to Moderate

[Cancel](#) [Previous](#) [Review and Launch](#) [Next: Configure Instance Details](#)

- Select **Next Configure Instance Details**
- Configure Instance: **Expand the Advanced** section and specify the following User Data script (make sure you use your unique OneAgent Download URI)

```
#!/bin/bash
wget -O Dynatrace-OneAgent-Linux.sh https://YOUR.FULL.DYNATRACE.ONEAGENT.DOWNLOADLINK
/bin/sh Dynatrace-OneAgent-Linux.sh APP_LOG_CONTENT_ACCESS=1 INFRA_ONLY=0
```

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

### Step 3: Configure Instance Details

**Monitoring** ⓘ ☐ Enable CloudWatch detailed monitoring  
Additional charges apply.

**Add Device**

▼ **Advanced Details**

**User data** ⓘ ☒ As text ☐ As file ☐ Input is already base64 encoded

```
#!/bin/bash
wget -O Dynatrace-OneAgent-Linux.sh "https://[redacted]aller/agent/unix
/default/&arch=x86"
/bin/sh Dynatrace-OneAgent-Linux.sh
APP_LOG_CONTENT_ACCESS=1 INFRA_ONLY=0
```

**Cancel** **Previous** **Review and Launch** **Next: Add Storage**

## Step 2

- Add Tags: on this configuration screen we add a custom tag. Key=EC2InstanceType; Value=LabExercise.
- Click through the rest of the steps. Review settings and click Launch
- Select or create a new key pair. We will need this for remoting into EC2
- You can observe the launch log

### Add/Edit Tags



Apply tags to your resources to help organize and identify them.

A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver. [Learn more](#) about tagging your Amazon EC2 resources.

Key	Value	
<input type="text" value="EC2InstanceType"/>	<input type="text" value="LabExercise"/>	
<input type="button" value="Create Tag"/>		<input type="button" value="Cancel"/> <input type="button" value="Save"/>

### Select an existing key pair or create a new key pair



A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Note: The selected key pair will be added to the set of keys authorized for this instance. [Learn more](#) about [removing existing key pairs from a public AMI](#).

**Select a key pair**

☒ I acknowledge that I have access to the selected private key file (testout.pem), and that without this file, I won't be able to log into my instance.

# Explore Dynatrace!

- Navigate to the Dynatrace Hosts list and wait until the host shows up. Click on it and explore what is monitored
- Expand the list of Properties and Tags. We should also find our EC2InstanceType tag with the value LabExcercise

The screenshot displays the Dynatrace interface for a host named 'ALn1'. The host is an Amazon Linux EC2 instance with an uptime of 5 minutes. The 'Properties and tags' section is expanded, showing various system attributes and tags. A red arrow points to the 'Add tag' button. The 'Processes' section is circled in red, showing the 'Go' process (amazon-ssm-agent). The 'CPU usage' section shows 0.12% active usage and 34% AWS usage. The 'Log file' section shows one log file at /var/log/messages (32.6 kB).

**Hosts** ALn1

ALn1  
Uptime: 5 minutes

[Edit](#) [...](#)

**Properties and tags**

[AWS]EC2InstanceType: LabExcercise [AWS]Name: ALn1 [+ Add tag](#)

Amazon Linux AMI 2017.09 (kernel 4.9.51-10.52.amzn1.x86\_64)

Instance type	t2.micro
Availability Zone	ap-southeast-1a
AMI ID	ami-0797ea64
Virtualization	Xen
Instance ID	i-05ef9ca8600584a12
Private host name	ip-172-30-0-33.ap-southeast-1.compute.internal
Security group	launch-wizard-2
OneAgent version	1.129.114.20170929-144438
Architecture	x86_64-bit
Physical CPU cores	1
Logical CPU cores	1
Cloud	EC2
Public IP address	54.169.117.41
Private IP address	172.30.0.33
Monitoring mode	Full-stack
Host units	0.1

**0 % CPU** **27 % Memory**

**1 NIC** **1 Disk**

**CPU usage 0.12 %** **CPU AWS 34 %**

Actively used CPU of the host as a percentage of available CPU.

**No problems** in last 72 hours

**100% Availability**  
0 min total downtime

**21:25**  
Running

**Processes**

**Go**  
amazon-ssm-agent

[All processes](#)

**No events** Today, 19:27 - 21:27 [Process crash details](#) [...](#)

**1 log file** Updated in last 72 hours  
This host writes to 1 log file

[/var/log/messages \(32.6 kB\)](#)

[View all log entries](#)

## Resources

- Dynatrace online help: <https://help.dynatrace.com>
  - Specifically for AWS: <https://help.dynatrace.com/infrastructure/amazon-web-services/how-do-i-start-amazon-web-services-monitoring>
- GitHub online tutorial for AWS: <https://github.com/Dynatrace/AWSMonitoringTutorials>
- Dynatrace blog: <https://www.dynatrace.com/blog>
- PurePerformance podcasts: <https://www.dynatrace.com/community/pureperformance>
- Dynatrace webinars: <https://www.dynatrace.com/company/webinars>



Every user, every app, everywhere.