SUMMARY:

This Lab covers Bootstrapping, CFN-INIT, Instance Roles, Parameter Store, Logging metrics with CW Agent

Contents

1. Bootstrapping wordpress directly and with CFN

In this [DEMO] you will bootstrap two EC2 instances with wordpress and the cowsay login banner customisations.

The first, directly using User Data via the console UI, the second, using Cloudformation

TEXT

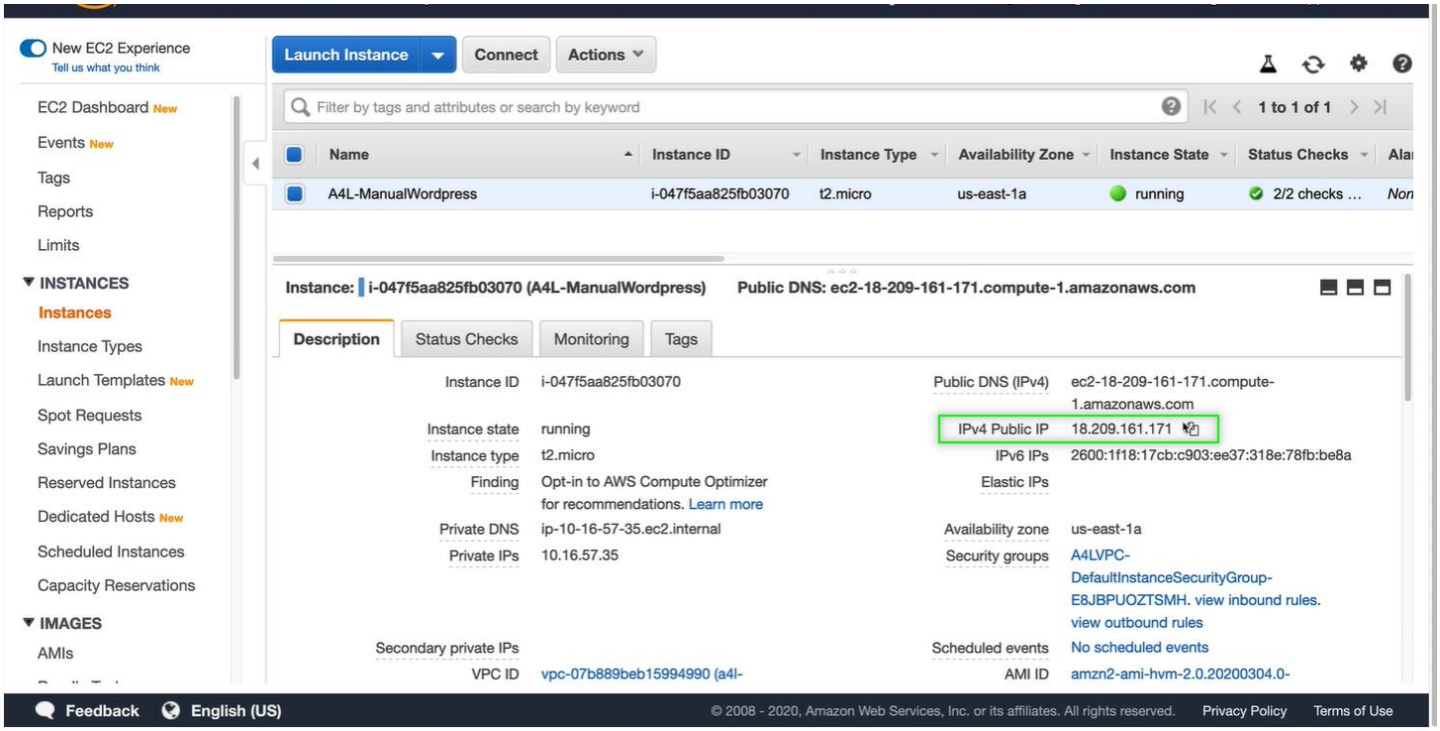
```
# Command used to query User Data

curl http://169.254.169.254/latest/user-data
```

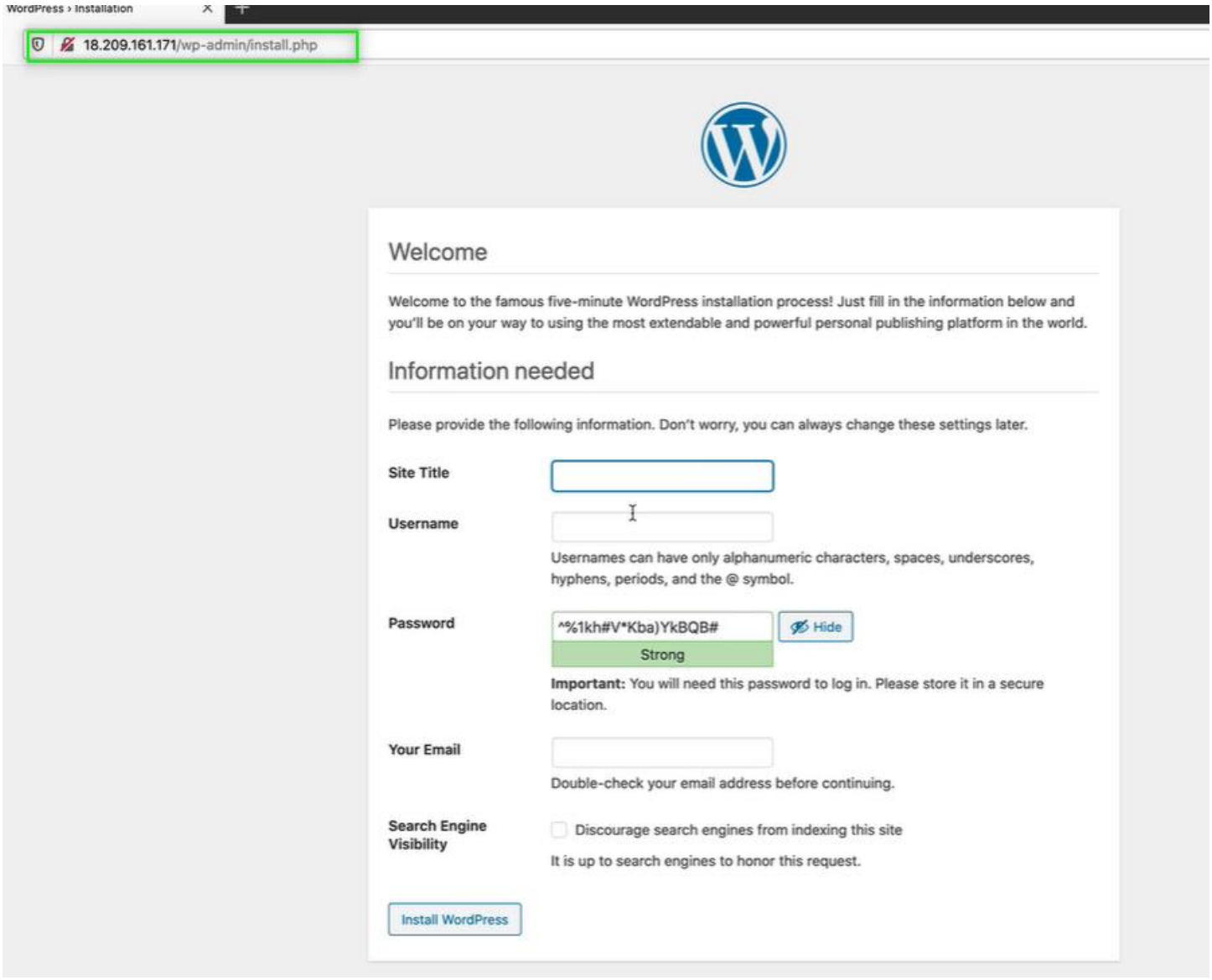
Bootstrapping script:

```
#!/bin/bash -xe

# Setpassword & DB Variables
DBName='a4lwordpress'
DBUser='a4lwordpress'
DBPassword='REPLACEME'
DBRootPassword='REPLACEME'

# System Updates
yum -y update
yum -y upgrade
# STEP 2 - Install system software - including Web and DB
yum install -y mariadb-server httpd wget cowsay
amazon-linux-extras install -y lamp-mariadb10.2-php7.2 php7.2
# STEP 3 - Web and DB Servers Online - and set to startup
systemctl enable httpd
systemctl enable mariadb
systemctl start httpd
systemctl start mariadb
# STEP 4 - Set Mariadb Root Password
mysqladmin -u root password $DBRootPassword
# STEP 5 - Install Wordpress
wget http://wordpress.org/latest.tar.gz -P /var/www/html
cd /var/www/html
tar -zxvf latest.tar.gz
cp -rvf wordpress/* .
```

```
rm -R wordpress
rm latest.tar.gz
# STEP 6 - Configure Wordpress
cp ./wp-config-sample.php ./wp-config.php
sed -i "s/'database_name_here'/'$DBName'/g" wp-config.php
sed -i "s/'username_here'/'$DBUser'/g" wp-config.php
sed -i "s/'password_here'/'$DBPassword'/g" wp-config.php
# Step 6a - permissions
usermod -a -G apache ec2-user
chown -R ec2-user:apache /var/www
chmod 2775 /var/www
find /var/www -type d -exec chmod 2775 {} \;
find /var/www -type f -exec chmod 0664 {} \;
# STEP 7 Create Wordpress DB
echo "CREATE DATABASE $DBName;" >> /tmp/db.setup
echo "CREATE USER '$DBUser'@'localhost' IDENTIFIED BY '$DBPassword';" >> /tmp/db.setup
echo "GRANT ALL ON $DBName.* TO '$DBUser'@'localhost';" >> /tmp/db.setup
echo "FLUSH PRIVILEGES;" >> /tmp/db.setup
mysql -u root --password=$DBRootPassword < /tmp/db.setup
sudo rm /tmp/db.setup
# STEP 8 COWSAY
echo "#!/bin/sh" > /etc/update-motd.d/40-cow
echo 'cowsay "Amazon Linux 2 AMI - Animals4Life"' >> /etc/update-motd.d/40-cow
chmod 755 /etc/update-motd.d/40-cow
rm /etc/update-motd.d/30-banner
update-motd
```

Creates EC2 instance with wordpress, check IP and verify site contents





2. CFN-INIT and CFN Creation Policies

In this [DEMO] Lesson you will get to experience how CFN-INIT, CFN-SIGNAL and CloudFormation Creation policies can further enhance the EC2 Bootstrapping process.

CFN Init – should create same results as the bootstrapping script:

```
Description:  A4L CFN-INIT Wordpress Template
  Requires A4L VPC Template to run
Parameters:
  LatestAmiId:
    Description: AMI for Instance (default is latest AmaLinux2)
    Type: 'AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>'
    Default: '/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2'
  KeyName:
    Type: AWS::EC2::KeyPair::KeyName
```

```
    Description: "Name of an existing SSH Keypair to access the instance"
  DBName:
    AllowedPattern: '[a-zA-Z][a-zA-Z0-9]*'
    ConstraintDescription: must begin with a letter and contain only alphanumeric
      characters.
    Default: a4lwordpress
    Description: The WordPress database name
    MaxLength: '64'
    MinLength: '1'
    Type: String
  DBPassword:
    AllowedPattern: '[a-zA-Z0-9]*'
    ConstraintDescription: must contain only alphanumeric characters.
    Description: The WordPress database admin account password
    MaxLength: '41'
    MinLength: '8'
    NoEcho: 'true'
    Type: String
  DBRootPassword:
    AllowedPattern: '[a-zA-Z0-9]*'
    ConstraintDescription: must contain only alphanumeric characters.
    Description: MySQL root password
    MaxLength: '41'
    MinLength: '8'
    NoEcho: 'true'
    Type: String
  DBUser:
    AllowedPattern: '[a-zA-Z][a-zA-Z0-9]*'
    ConstraintDescription: must begin with a letter and contain only alphanumeric
      characters.
    Description: The WordPress database admin account username
    Default: a4lwordpress
    MaxLength: '16'
    MinLength: '1'
    NoEcho: 'true'
    Type: String
Resources:
  EC2Instance:
    Type: AWS::EC2::Instance
    CreationPolicy:
      ResourceSignal:
        Timeout: PT15M
    Metadata:
      AWS::CloudFormation::Init:
        configSets:
          wordpress_install:
            - install_cfn
            - software_install
            - configure_instance
            - install_wordpress
            - configure_wordpress
        install_cfn:
```

```
          files:
            /etc/cfn/cfn-hup.conf:
              content: !Sub |
                [main]
                stack= ${AWS::StackId}
                region=${AWS::Region}
              group: root
              mode: '000400'
              owner: root
            /etc/cfn/hooks.d/cfn-auto-reloader.conf:
              content: !Sub |
                [cfn-auto-reloader-hook]
                triggers=post.update
                path=Resources.EC2Instance.Metadata.AWS::CloudFormation::Init
                action=/opt/aws/bin/cfn-init -v --stack ${AWS::StackName} --
resource EC2Instance --configsets wordpress_install --region ${AWS::Region}
              group: root
              mode: '000400'
              owner: root
          services:
            sysvinit:
              cfn-hup:
                enabled: true
                ensureRunning: true
                files:
                - /etc/cfn/cfn-hup.conf
                - /etc/cfn/hooks.d/cfn-auto-reloader.conf
        software_install:
          packages:
            yum:
              httpd: []
              mariadb-server: []
              wget: []
              cowsay: []
          commands:
            0_extra_installs_php72_lampmariadb:
              command: amazon-linux-extras install -y lamp-mariadb10.2-php7.2 php7.2
          services:
            sysvinit:
              httpd:
                enabled: true
                ensureRunning: true
              mariadb:
                enabled: true
                ensureRunning: true
        configure_instance:
          files:
            /etc/update-motd.d/40-cow:
              content: !Sub |
                #!/bin/sh
                cowsay "Amazon Linux 2 AMI - Animals4Life"
              group: root
```

```
              mode: '000755'
              owner: root
          commands:
            01_set_mysql_root_password:
              command: !Sub |
                mysqladmin -u root password '${DBRootPassword}'
              test: !Sub |
                $(mysql ${DBName} -u root --
password='${DBRootPassword}' >/dev/null 2>&1 </dev/null); (( $? != 0 ))
            02_remove_original_banner:
              command: rm /etc/update-motd.d/30-banner
            03_updatemotd:
              command: update-motd
        install_wordpress:
          sources:
            /var/www/html: http://wordpress.org/latest.tar.gz
          files:
            /tmp/create-wp-config:
              content: !Sub |
                #!/bin/bash -xe
                cp /var/www/html/wp-config-sample.php /var/www/html/wp-config.php
                sed -i "s/'database_name_here'/'${DBName}'/g" wp-config.php
                sed -i "s/'username_here'/'${DBUser}'/g" wp-config.php
                sed -i "s/'password_here'/'${DBPassword}'/g" wp-
config.php
              group: root
              mode: '000500'
              owner: root
            /tmp/db.setup:
              content: !Sub |
                CREATE DATABASE ${DBName};
                CREATE USER '${DBUser}'@'localhost' IDENTIFIED BY '${DBPassword}';
                GRANT ALL ON ${DBName}.* TO '${DBUser}'@'localhost';
                FLUSH PRIVILEGES;
              group: root
              mode: '000400'
              owner: root
        configure_wordpress:
          files:
            /tmp/permissionsfix:
              content: !Sub |
                usermod -a -G apache ec2-user
                chown -R ec2-user:apache /var/www
                chmod 2775 /var/www
                find /var/www -type d -exec chmod 2775 {} \;
                find /var/www -type f -exec chmod 0664 {} \;
              group: root
              mode: '000500'
              owner: root
          commands:
            01_create_database:
              command: !Sub |
```

```
                mysql -u root --password='${DBRootPassword}' < /tmp/db.setup
              test: !Sub |
                $(mysql ${DBName} -u root --
password='${DBRootPassword}' >/dev/null 2>&1 </dev/null); (( $? !=0))
            02_move_wordpress:
              command: !Sub |
                cp -rvf /var/www/html/wordpress/* /var/www/html/
            03_tidyup:
              command: !Sub |
                rm -R /var/www/html/wordpress
            04_configure_wordpress:
              command: /tmp/create-wp-config
              cwd: /var/www/html
            04_fix_permissions:
              command: /tmp/permissionsfix
    Properties:
      KeyName: !Ref KeyName
      InstanceType: "t2.micro"
      ImageId: !Ref LatestAmiId
      SubnetId: !ImportValue a4l-vpc1-subnet-weba
      SecurityGroupIds:
        - !ImportValue a4l-vpc1-default-instance-sg
      Tags:
        - Key: Name
          Value: A4L-Wordpress
      UserData:
        Fn::Base64: !Sub |
          #!/bin/bash -xe
          yum -y update
          /opt/aws/bin/cfn-init -v --stack ${AWS::StackId} --resource EC2Instance --
configsets wordpress_install --region ${AWS::Region}
          /opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackId} --resource EC2Instance --
region ${AWS::Region}
```

3. Providing permissions and credentials to EC2 using Instance Roles'

In this [DEMO] Lesson you will create an EC2 Instance Role, apply it to an EC2 instance and learn how to interact with the credentials this generates within the EC2 instance metadata.

1-Click Deployment

Lesson Commands

Credential Precedence

Create instance, create instance role in IAM, attach instance role to the instance:

Create role

1 2 3 4

Select type of trusted entity

AWS service
EC2, Lambda and others

Another AWS account
Belonging to you or 3rd party

Web identity
Cognito or any OpenID provider

SAML 2.0 federation
Your corporate directory

Allows AWS services to perform actions on your behalf. Learn more

Choose a use case

Common use cases

EC2
Allows EC2 instances to call AWS services on your behalf.

Lambda
Allows Lambda functions to call AWS services on your behalf.

Attach permissions policies

Choose one or more policies to attach to your new role.

Create policy

Filter policies s3 Showing 9 results

Policy name	Used as
AmazonDMSRedshiftS3Role	None
AmazonS3FullAccess	None
AmazonS3OutpostsFullAccess	None
AmazonS3OutpostsReadOnlyAccess	None
AmazonS3ReadOnlyAccess	None
IVSRecordToS3	None
QuickSightAccessForS3StorageManagementAnalyticsReadOnly	None
s3crr_for_animals4lifeuseast1_to_animals4lifeuswest1_80cad7	None

Set permissions boundary

Review

Provide the required information below and review this role before you create it.

Role name* A4LInstanceRole

Use alphanumeric and '+=,.@-_' characters. Maximum 64 characters.

Role description Allows EC2 instances to call AWS services on your behalf.

Maximum 1000 characters. Use alphanumeric and '+=,.@-_' characters.

Trusted entities AWS service: ec2.amazonaws.com

Policies AmazonS3ReadOnlyAccess

Permissions boundary Permissions boundary is not set

EC2 > Instances > i-049b107605a791e55 > Modify IAM role

Modify IAM role Info

Attach an IAM role to your instance.

Instance ID

i-049b107605a791e55 (A4L-PublicEC2)

IAM role

Select an IAM role to attach to your instance or create a new role if you haven't created any. The role you select replaces any roles that are currently attached to your instance.

Choose IAM role

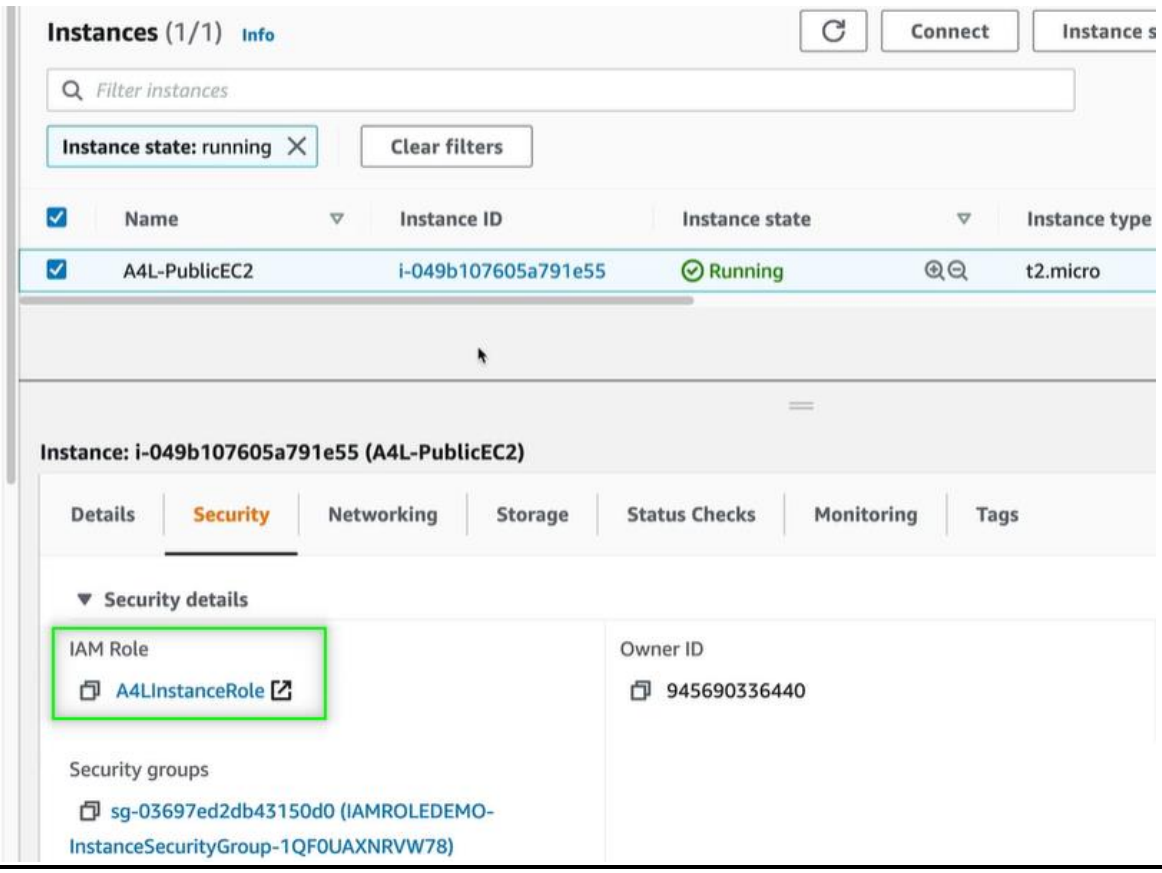
Create new IAM role

No IAM Role
Choose this option to detach an IAM role

A4LInstanceRole
arn:aws:iam::945690336440:instance-profile/A4LInstanceRole

e instance will be removed. Are you

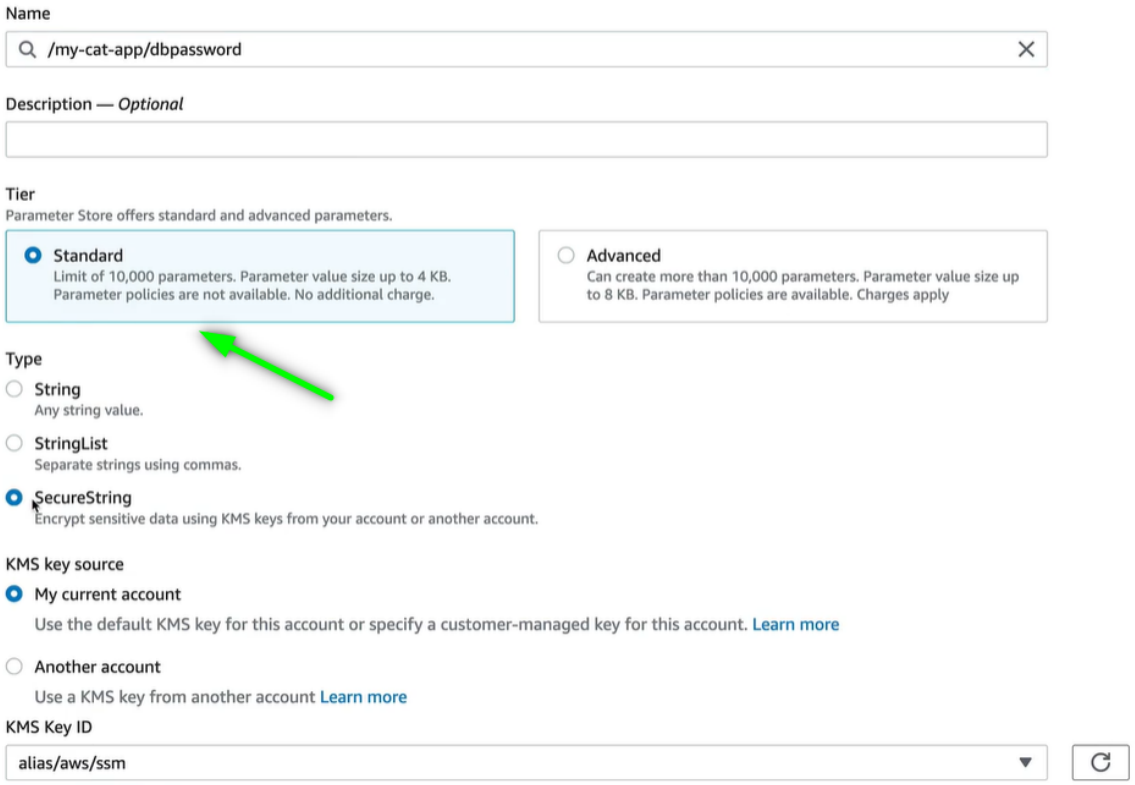
Cancel Save

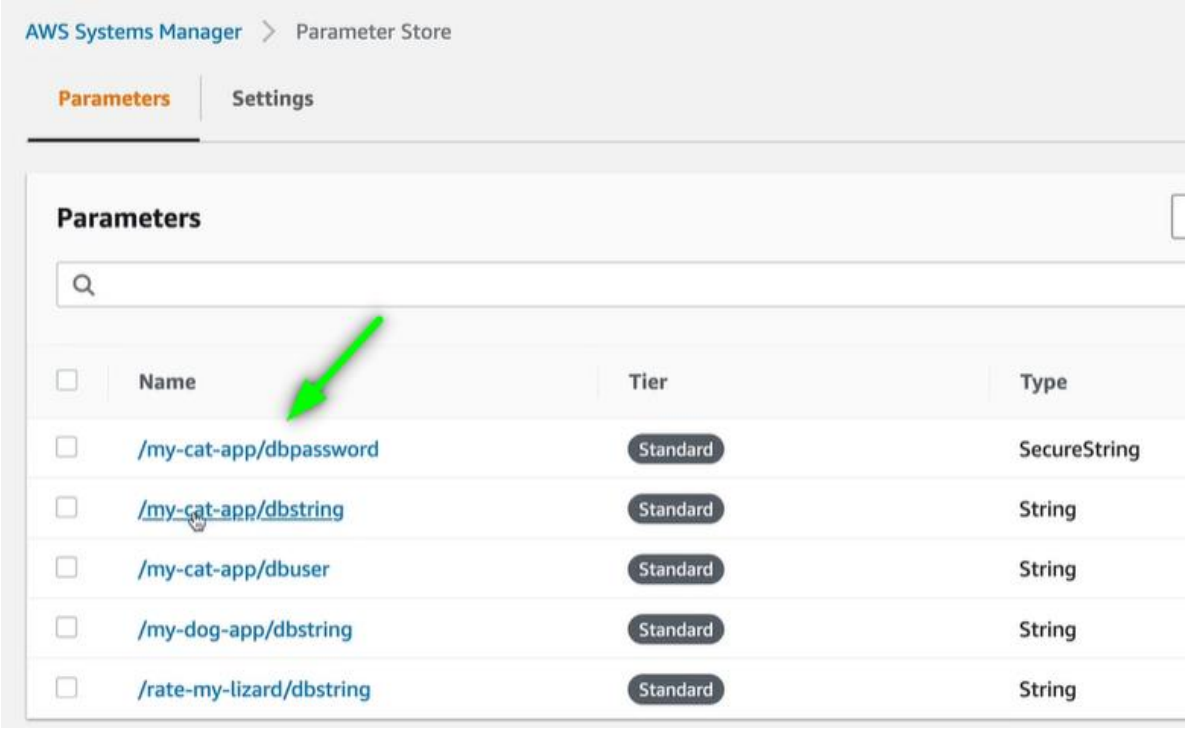


4. Parameter Store

In this [DEMO] you get a chance to create some Parameters in the Parameter Store and interact with them via the command line - using individual parameter operations and accessing via paths.

Create parameters by path





5. Logging and Metrics with CW Agent

In this [DEMO] lesson you will download and install the CloudWatch Agent and configure it to capture 3 log files from an EC2 instance

/var/log/secure

/var/log/httpd/access_log

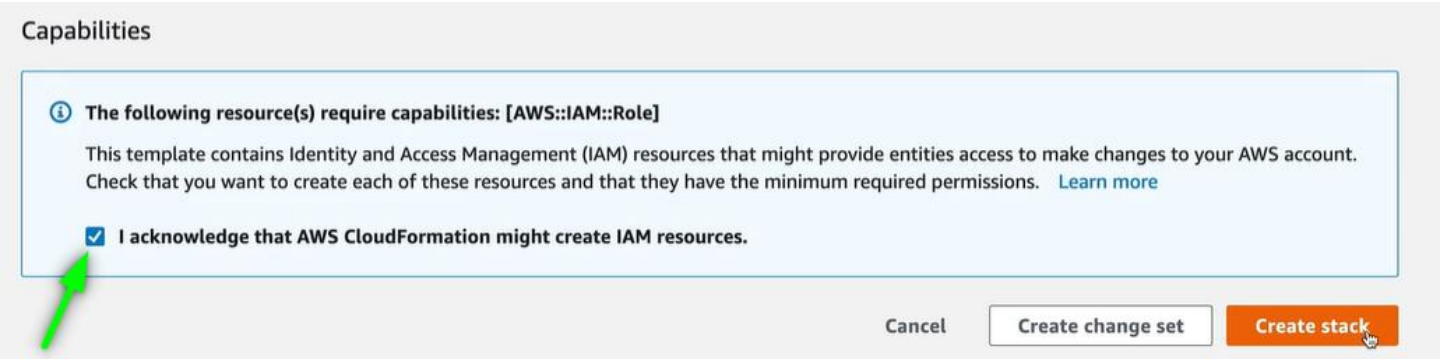
/var/log/httpd/error_log

You will also configure an instance role allowing the agent to store the above config into parameter store AND allow the agent to inject the logging and metric data into CW and CW Logs.

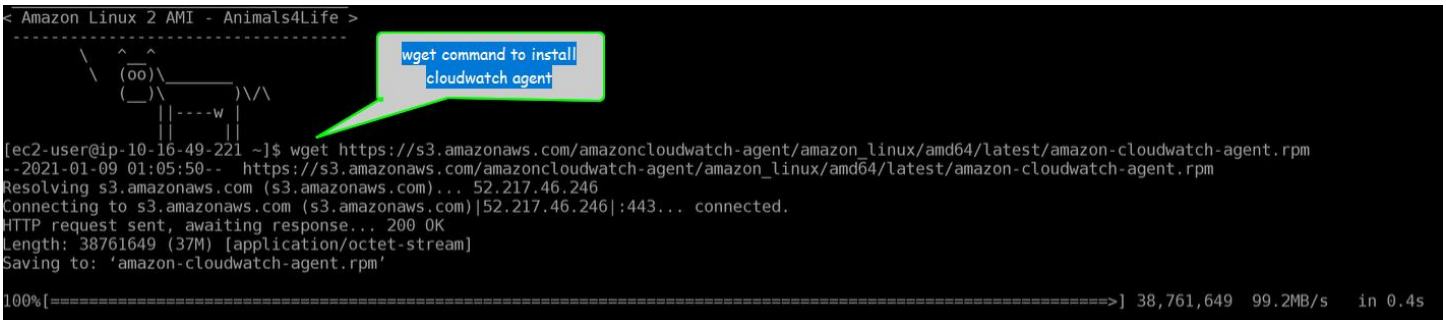
1-Click Deployment

Lesson Commands

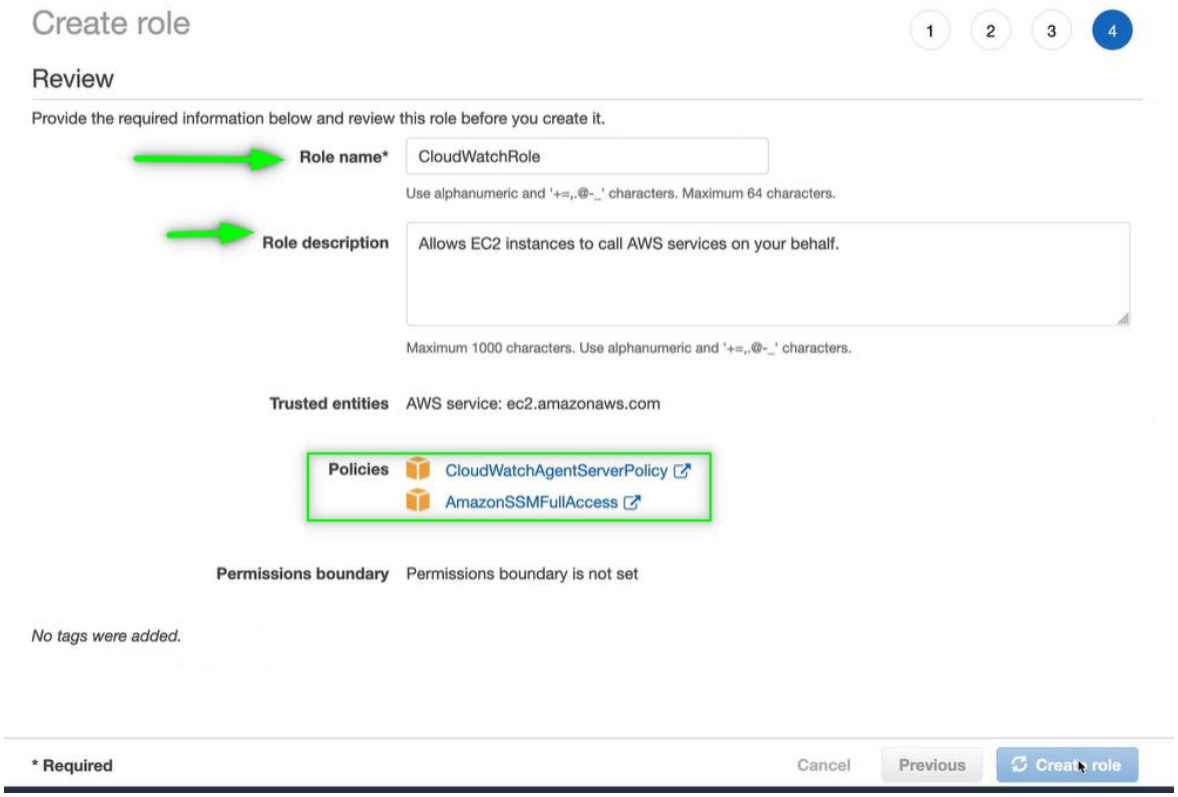
Be sure to acknowledge cloudwatch on:

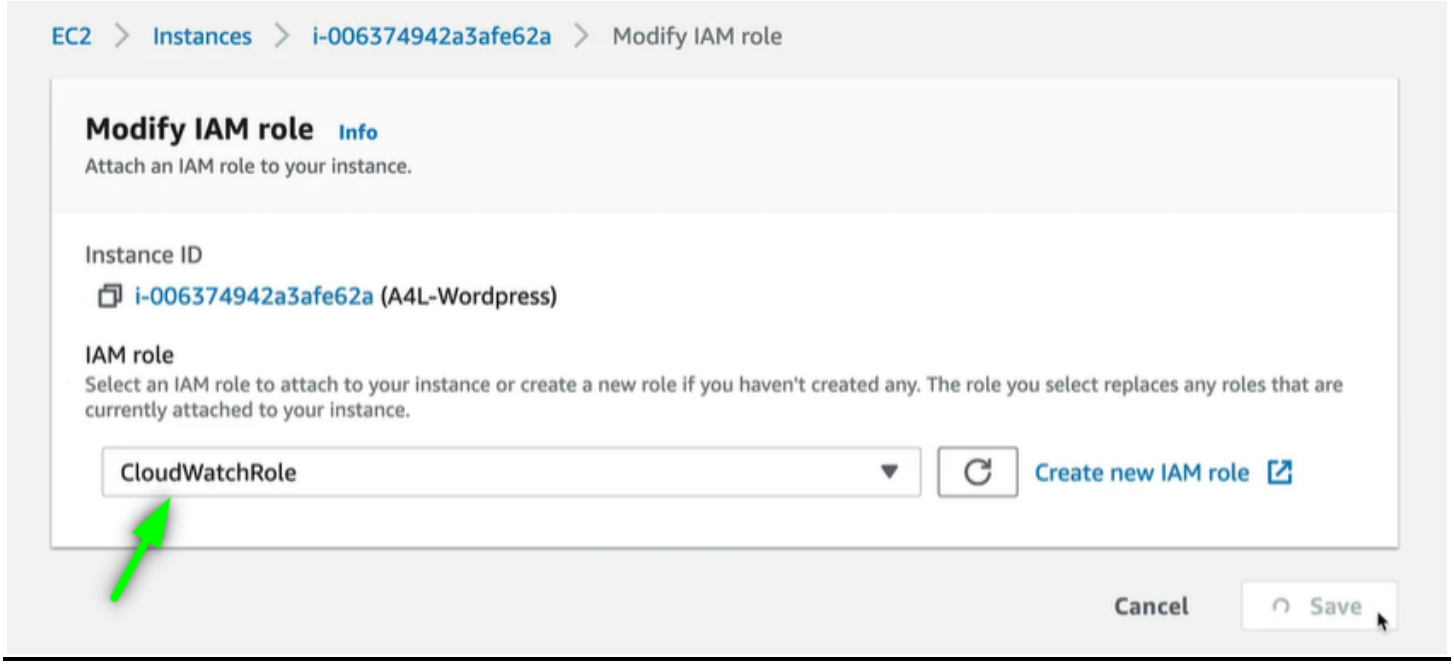


Install agent on instance:

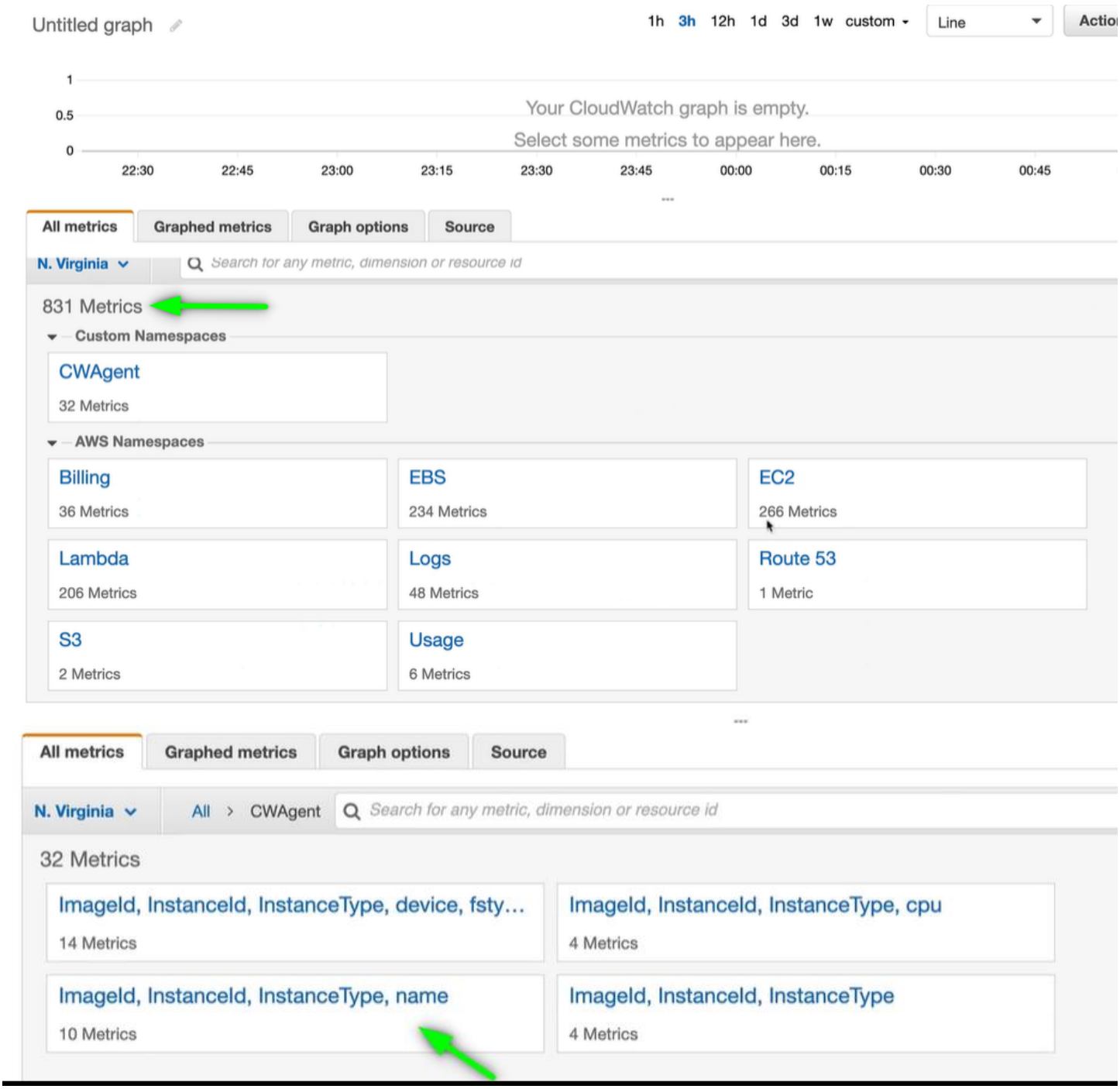


Create and attach IAM cloudwatch role for instance





Can interact with wordpress instance and now have access to metrics:


Untitled graph
1h 3h 12h 1d 3d 1w custom
Line
Actio
1
0.5
0
Your CloudWatch graph is empty.
Select some metrics to appear here.
22:30 22:45 23:00 23:15 23:30 23:45 00:00 00:15 00:30 00:45
All metrics
Graphed metrics
Graph options
Source
N. Virginia
Search for any metric, dimension or resource id
831 Metrics
Custom Namespaces
CWAgent
32 Metrics
AWS Namespaces
Billing
36 Metrics
EBS
234 Metrics
EC2
266 Metrics
Lambda
206 Metrics
Logs
48 Metrics
Route 53
1 Metric
S3
2 Metrics
Usage
6 Metrics
All metrics
Graphed metrics
Graph options
Source
N. Virginia
All > CWAgent
Search for any metric, dimension or resource id
32 Metrics
ImageId, InstanceId, InstanceType, device, fsty...
14 Metrics
ImageId, InstanceId, InstanceType, cpu
4 Metrics
ImageId, InstanceId, InstanceType, name
10 Metrics
ImageId, InstanceId, InstanceType
4 Metrics

All metrics | Graphed metrics | Graph options | Source

N. Virginia > All > CWAgent > ImageId, InstanceId, InstanceType, name

Search for any metric, dimension or resource id

Instance Name (10)	ImageId	InstanceId	InstanceType	name	Metric Name
A4L-Wordpress	ami-0be2609ba883822ec	i-006374942a3afe62a	t2.micro	xvda1	diskio_read_bytes
A4L-Wordpress	ami-0be2609ba883822ec	i-006374942a3afe62a	t2.micro	xvda1	diskio_io_time
A4L-Wordpress	ami-0be2609ba883822ec	i-006374942a3afe62a	t2.micro	xvda1	diskio_reads
A4L-Wordpress	ami-0be2609ba883822ec	i-006374942a3afe62a	t2.micro	xvda1	diskio_writes
A4L-Wordpress	ami-0be2609ba883822ec	i-006374942a3afe62a	t2.micro	xvda1	diskio_write_bytes
A4L-Wordpress	ami-0be2609ba883822ec	i-006374942a3afe62a	t2.micro	xvda	diskio_writes
A4L-Wordpress	ami-0be2609ba883822ec	i-006374942a3afe62a	t2.micro	xvda	diskio_write_bytes
A4L-Wordpress	ami-0be2609ba883822ec	i-006374942a3afe62a	t2.micro	xvda	diskio_reads
A4L-Wordpress	ami-0be2609ba883822ec	i-006374942a3afe62a	t2.micro	xvda	diskio_read_bytes
A4L-Wordpress	ami-0be2609ba883822ec	i-006374942a3afe62a	t2.micro	xvda	diskio_io_time