> awless

a mighty CLI for AWS

Henri Binsztok
hbi@wallix.com

sponsored by WALLIX

https://console.aws.amazon.com/console/home?region=us-east-1

Search

AWS | Services | Edit | Victor | N. Virginia | Support

Amazon Web Services

Compute

EC2
Virtual Servers in the Cloud

EC2 Container Service
Run and Manage Docker Containers

Elastic Beanstalk
Run and Manage Web Apps

Lambda
Run Code in Response to Events

Storage & Content Delivery

S3
Scalable Storage in the Cloud

CloudFront
Global Content Delivery Network

Elastic File System PREVIEW
Fully Managed File System for EC2

Glacier
Archive Storage in the Cloud

Import/Export Snowball
Large Scale Data Transport

Storage Gateway
Hybrid Storage Integration

Database

RDS
Managed Relational Database Service

DynamoDB
Managed NoSQL Database

Developer Tools

CodeCommit
Store Code in Private Git Repositories

CodeDeploy
Automate Code Deployments

CodePipeline
Release Software using Continuous Delivery

Management Tools

CloudWatch
Monitor Resources and Applications

CloudFormation
Create and Manage Resources with Templates

CloudTrail
Track User Activity and API Usage

Config
Track Resource Inventory and Changes

OpsWorks
Automate Operations with Chef

Service Catalog
Create and Use Standardized Products

Trusted Advisor
Optimize Performance and Security

Security & Identity

Identity & Access Management
Manage User Access and Encryption Keys

Directory Service
Host and Manage Active Directory

Inspector PREVIEW

Internet of Things

AWS IoT
Connect Devices to the Cloud

Mobile Services

Mobile Hub BETA
Build, Test, and Monitor Mobile Apps

Cognito
User Identity and App Data Synchronization

Device Farm
Test Android, FireOS, and iOS Apps on Real Devices in the Cloud

Mobile Analytics
Collect, View and Export App Analytics

SNS
Push Notification Service

Application Services

API Gateway
Build, Deploy and Manage APIs

AppStream
Low Latency Application Streaming

CloudSearch
Managed Search Service

Elastic Transcoder
Easy-to-Use Scalable Media Transcoding

SES
Email Sending and Receiving Service

SQS
Message Queue Service

Resource Groups

Learn more

A resource group is a collection of resources that share one or more tags. Create a group for each project, application, or environment in your account.

Create a Group | Tag Editor

Additional Resources

Getting Started
Read our documentation or view our training to learn more about AWS.

AWS Console Mobile App
View your resources on the go with our AWS Console mobile app, available from Amazon Appstore, Google Play, or iTunes.

AWS Marketplace
Find and buy software, launch with 1-Click and pay by the hour.

AWS re:Invent Announcements
Explore the next generation of AWS cloud capabilities. See what's new

https://console.aws.amazon.com/appstream/home?region=us-east-1

EC2 Management Console

https://eu-west-1.console.aws.amazon.com/ec2/v2/home?region=eu-west-1

AWS | Services | Edit | Jason Agnew | Ireland | Support

- EC2 Dashboard
- Events
- Tags
- Reports
- Limits

INSTANCES
- Instances
- Spot Requests
- Reserved Instances

IMAGES
- AMIs
- Bundle Tasks

ELASTIC BLOCK STORE
- Volumes
- Snapshots

NETWORK & SECURITY
- Security Groups
- Elastic IPs
- Placement Groups
- Key Pairs
- Network Interfaces

LOAD BALANCING
- Load Balancers

AUTO SCALING
- Launch Configurations
- Auto Scaling Groups

Resources

You are using the following Amazon EC2 resources in the EU West (Ireland) region:

- 3 Running Instances
- 3 Volumes
- 1 Key Pairs
- 0 Placement Groups
- 0 Elastic IPs
- 0 Snapshots
- 3 Load Balancers
- 9 Security Groups

Easily deploy and operate applications - use Chef recipes, manage SSH users, and more. Try OpsWorks now. Hide

Create Instance

To start using Amazon EC2 you will want to launch a virtual server, known as an Amazon EC2 instance.

Launch Instance

Note: Your instances will launch in the EU West (Ireland) region

Service Health

Service Status:

EU West (Ireland):
This service is operating normally

Availability Zone Status:

eu-west-1a:
Availability zone is operating normally

eu-west-1b:
Availability zone is operating normally

eu-west-1c:
Availability zone is operating normally

Service Health Dashboard

Scheduled Events

EU West (Ireland):

No events

Account Attributes

Supported Platforms
VPC

Default VPC
vpc-adc97dc8

Additional Information

Getting Started Guide
Documentation
All EC2 Resources
Forums
Pricing
Contact Us

AWS Marketplace

Find **free software trial** products in the AWS Marketplace from the EC2 Launch Wizard.
Or try these popular AMIs:

Wowza Streaming Engine 4: Pro Edition (HVM)

Provided by Wowza Media Systems, Inc.
Rating ★★★★★
Pay by the hour for software and AWS usage
View all Media

Vantage Cloud Subscriptions - Transcode Server

Provided by Telestream, Inc.
Rating ★★★★★
Pay by the hour for software and AWS usage
View all Media

Vidispine Developer and Prototyping

© 2008 - 2015, Amazon Web Services, Inc. or its affiliates. All rights reserved.

VPC Dashboard
Filter by VPC:
None

Virtual Private Cloud
Your VPCs
Subnets
Route Tables
Internet Gateways
DHCP Options Sets
Elastic IPs
Endpoints
Peering Connections

Security
Network ACLs
Security Groups

VPN Connections
Customer Gateways
Virtual Private Gateways
VPN Connections

Create Subnet | Delete Subnet | Modify Auto-Assign Public IP

Search Subnets and their pr

1 to 1 of 1 Subnet

Name	Subnet ID	State	VPC	CIDR	Available IPs	Availability Zone
	subnet-dffe7cf4	available	vpc-42dffc27 (172.31.0.0/16)	172.31.100.0/24	251	us-east-1a

Create Subnet

Use the CIDR format to specify your subnet's IP address block (e.g., 10.0.0.0/24). Note that block sizes must be between a /16 netmask and /28 netmask. Also, note that a subnet can be the same size as your VPC.

Name tag
VPC vpc-42dffc27 (172.31.0.0/16)
Availability Zone us-east-1b
CIDR block 172.31.101.0/24

Cancel | Yes, Create

subnet-dffe7cf4 (172.31.100.0/24)

Summary | Route Table | Network ACL | Tags

Subnet ID: subnet-dffe7cf4
CIDR: 172.31.100.0/24
State: available
VPC: vpc-42dffc27 (172.31.0.0/16)
Available IPs: 251
Availability Zone: us-east-1a
Route table: rtb-70e2e715
Network ACL: acl-dcfed8b9
Default subnet: no
Auto-assign Public IP: no

© 2008 - 2015, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AWS GUI

- Nice but:
  - Not repeatable, scriptable, etc.
  - No history…
  - Many opportunities to hide important knowledge…

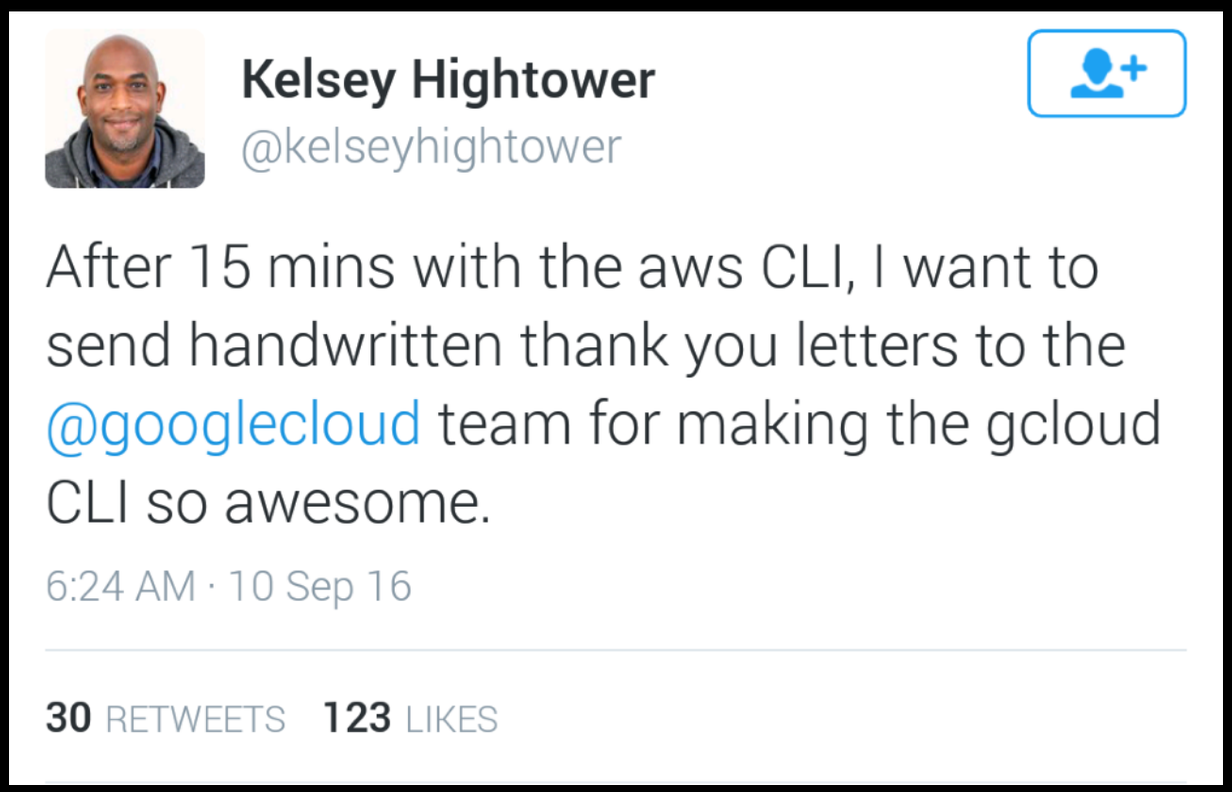
```
~ aws ec2 describe-instances
{
    "Reservations": [
        {
            "OwnerId": "519101999238",
            "ReservationId": "r-bd6eae35",
            "Groups": [],
            "Instances": [
                {
                    "Monitoring": {
                        "State": "disabled"
                    },
                    "PublicDnsName": "",
                    "RootDeviceType": "ebs",
                    "State": {
                        "Code": 80,
                        "Name": "stopped"
                    },
                    "EbsOptimized": false,
                    "LaunchTime": "2016-12-01T17:12:03.000Z",
                    "PrivateIpAddress": "172.31.9.79",
                    "ProductCodes": [],
                    "VpcId": "vpc-00b68c65",
                    "StateTransitionReason": "User initiated (2016-12-02 13:00:40 GMT)",
                    "InstanceId": "i-74e7d79c",
                    "ImageId": "ami-e98bd29a",
                    "PrivateDnsName": "ip-172-31-9-79.eu-west-1.compute.internal",
                    "KeyName": "1/jpa",
                    "SecurityGroups": [
                        {
...
```

```
 ✘  ~  aws ec2 run-instances --image-id ami-9398d3e0 --count 1 --instance-type t2.nano --key-name awless --security-gro
up-id sg-ceecd9aa --subnet-id subnet-0c41ad68
{
    "OwnerId": "519101999238",
    "ReservationId": "r-0fe4d932b50c7d5ea",
    "Groups": [],
    "Instances": [
        {
            "Monitoring": {
                "State": "disabled"
            },
            "PublicDnsName": "",
            "RootDeviceType": "ebs",
            "State": {
                "Code": 0,
                "Name": "pending"
            },
            "EbsOptimized": false,
            "LaunchTime": "2017-01-11T14:56:17.000Z",
            "PrivateIpAddress": "172.31.14.79",
            "ProductCodes": [],
            "VpcId": "vpc-00b68c65",
            "StateTransitionReason": "",
            "InstanceId": "i-0ae3f48b168b26b28",
            "ImageId": "ami-9398d3e0",
            "PrivateDnsName": "ip-172-31-14-79.eu-west-1.compute.internal",
            "KeyName": "awless",
            "SecurityGroups": [
                {
                    "GroupName": "default",
                    "GroupId": "sg-ceecd9aa"
                      "SecurityGroups": [
                          {
    ...
 ~  aws ec2 create-tags --resources i-0ae3f48b168b26b28 --tags Key=Name,Value=awless-test
```

AWS CLI

- Nice but:
  - Edit JSON files, id-driven, remember keys, run long sequences of commands…
  - Complex and verbose output…
  - No infrastructure history…
  - Share knowledge with others…

AWS CLI



Really?

Meet awless

- Easy-to-use CLI
- Scriptable
- Local model of the whole AWS infrastructure
- Graph stored as RDF
- History and reversibility (WIP)
- Supports

Install awless

- Apache Licensed
- Using brew:
  - `brew tap wallix/awless; brew install awless`
- Using go:
  - `go get -u github.com/wallix/awless`
- or download binaries:
  - https://github.com/wallix/awless/releases

First run awless

1. Setup AWS account
   - Nothing to do if aws-cli installed
2. Setup auto-completion
   - For bash + zsh

> awless [one-liner]

attach	alarm, elasticip, instance, internetgateway, policy, role, routetable, securitygroup, user, volume
check	instance, loadbalancer, securitygroup, scalinggroup
create	accesskey, alarm, bucket, database, dbsubnetgroup, elasticip, function, group, instance, instanceprofile, internetgateway, keypair, launchconfiguration, listener, loadbalancer, policy, queue, record, role, route, routetable, s3object, scalinggroup, scalingpolicy, securitygroup, snapshot, subnet, subscription, tag, targetgroup, topic, user, volume, vpc, zone
delete	
detach	alarm, elasticip, instance, internetgateway, policy, role, routetable, securitygroup, user, volume
start **stop**	alarm, instance
update	instance, scalinggroup, securitygroup, subnet

> awless [command]

completion	show
config	ssh
inspect	sync
list	version
log	whoami
revert	
run	
search	

> awless list

```
> awless list instances --filter type=t2.nano --sort "up since"
```

ID	ZONE	NAME	STATE	TYPE	PUBLIC IP	PRIVATE IP	UP SINCE ▲
i-03dd37e95a767ab8e	eu-central-1a	dev	stopped	t2.nano		10.0.10.49	4 mins
i-05b2d9f195444ff71	eu-central-1b	wordpress-server-2	running	t2.nano		10.0.11.249	6 mins
i-06d251053ebe92bfc	eu-central-1a	wordpress-server-1	running	t2.nano		10.0.10.43	6 mins
i-05158aff863f52169	eu-central-1b	awless-demo	running	t2.nano	35.157.138.108	10.0.100.160	10 mins
i-005b25e8407245817	eu-central-1b	test-instance	running	t2.nano	35.157.213.133	10.0.100.64	70 mins

```
> awless list subnets --format csv

Id, Name, CidrBlock, Zone, Default, VpcId, Public, State
subnet-77bac21f, private-subnet-1a, 10.0.10.0/24, eu-central-1a, false, vpc-9f6bd2f7, false, available
subnet-a9480cd2, , 172.31.16.0/20, eu-central-1b, true, vpc-71a5ea18, true, available
subnet-b89616c2, public-subnet-1b, 10.0.100.0/24, eu-central-1b, false, vpc-9f6bd2f7, true, available
subnet-cbebcaa2, , 172.31.0.0/20, eu-central-1a, true, vpc-71a5ea18, true, available
subnet-e78c0c9d, private-subnet-1b, 10.0.11.0/24, eu-central-1b, false, vpc-9f6bd2f7, false, available
```

> awless list --local

```
> awless list instances

Error: RequestError: send request failed
caused by: Post https://ec2.eu-central-1.amazonaws.com/: dial tcp: lookup ec2.eu-central-1.amazonaws.com: no such host
```

```
> awless list instances --local

|         ID ▲         |     ZONE      |            NAME             |   STATE    |
|----------------------|---------------|-----------------------------|------------|
| i-0a36432d4f339f008  | eu-central-1b | ssh-demo                    | terminated |
| i-0dcf69318ba3b2be9  | eu-central-1b | demo-background-instance    | running    |
Columns truncated to fit terminal: 'Type', 'Public IP', 'Private IP', 'Up Since', 'Access Key'
```

> awless show

```
> awless show test-instance
+------------------+--------------------+
|    PROPERTY ▲    |       VALUE        |
+------------------+--------------------+
| Access Key       | mykeypair          |
| Architecture     | x86_64             |
| Hypervisor       | xen                |
| Id               | i-005b25e8407245817 |
| ImageId          | ami-af0fc0c0       |
| Name             | test-instance      |
| NetworkInterfaces | [eni-5ef46020]     |
| Private IP       | 10.0.100.64        |
| Public IP        | 35.157.213.133     |
| RootDeviceName   | /dev/xvda          |
| RootDeviceType   | ebs                |
| SecurityGroups   | [sg-bce334d7]      |
| State            | running            |
| SubnetId         | subnet-b89616c2    |
| Type             | t2.nano            |
| Up Since         | 74 mins            |
| VpcId            | vpc-9f6bd2f7       |
| Zone             | eu-central-1b      |
+------------------+--------------------+

Relations:
eu-central-1[region]
   ↳ @awless-demo-vpc[vpc]
      ↳ @public-subnet-1b[subnet]
         ↳ @test-instance[instance]

Siblings: @awless-demo[instance]

Depending on: @mykeypair[keypair], @default[securitygroup], vol-0bbb1d719ee89e5af[volume]
```

> awless create

```
> awless create keypair name=my-ssh-keypair

create keypair name=my-ssh-keypair

Confirm? (y/n): y
[info]    Generating locally a RSA 4096 bits keypair...
[info]    4096 RSA keypair generated locally and stored in '/Users/fx/.awless/
keys/my-ssh-keypair.pem'
  OK      keypair=my-ssh-keypair
[info]    Revert this template with `awless revert 01BBP7N5HTA8YADC8B0T1S5DDT`

> awless create instance key=my-ssh-keypair subnet=@public-subnet-1b name=my-
ssh-instance

create instance count=1 image=ami-af0fc0c0 key=my-ssh-keypair name=my-ssh-
instance subnet=subnet-b89616c2 type=t2.nano

Confirm? (y/n): y
    OK      instance=i-0b61f81a57b7854d9

[info]    Revert this template with `awless revert 01BBP7WAGVCJ3CHMS52NCKQXFV`
```

> awless create

```
> awless create instance

Please specify (Ctrl+C to quit, Tab for completion):
instance.name? instance-name
instance.subnet? @private-subnet-1a

create instance count=1 image=ami-af0fc0c0 name=instance-name
subnet=subnet-77bac21f type=t2.nano

Confirm? (y/n): y
    OK create instance count=1 image=ami-af0fc0c0 name=instance-name
subnet=subnet-77bac21f type=t2.nano [i-083bc22d480bc13f1]

[info]    Revert this template with `awless revert
01BBP88MQCNHQ4JJ8DE5EX1P0F`
```

> awless ssh

```
> awless ssh my-ssh-instance

[info]    Login as 'ec2-user' on '35.157.130.30', using key '~/.awless/
keys/my-ssh-keypair.pem' with ssh client at '/usr/bin/ssh'
The authenticity of host '35.157.130.30 (35.157.130.30)' can't be
established.
ECDSA key fingerprint is
SHA256:o1HAIjNcEyPRioUkPB4CHvz8BASO87+DJhCYavworZ4.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '35.157.130.30' (ECDSA) to the list of known
hosts.

       __|  __|_  )
       _|  (     /   Amazon Linux AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-ami/2016.09-release-notes/
6 package(s) needed for security, out of 8 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-10-0-100-116 ~]$
```

Templates

```
# Loadbalancer
## Create the loadbalancer firewall
loadbalancerfw = create securitygroup vpc={wordpress.vpc} description=wordpress-loadbalancer-
security-group name=wordpress-lb-secgroup
update securitygroup id=$loadbalancerfw inbound=authorize protocol=tcp cidr=0.0.0.0/0
portrange=80
## Create the target group for EC2 wordpress servers
targetgroup = create targetgroup name=wordpress-workers port=80 protocol=HTTP
vpc={wordpress.vpc}
## Create the application load balancer that will redirect flows to the servers
lb = create loadbalancer name=wordpress-loadbalancer subnets={wordpress.subnets}
groups=$loadbalancerfw
create listener actiontype=forward loadbalancer=$lb port=80 protocol=HTTP target=$targetgroup

# Wordpress application servers
## Create the wordpress servers
inst1 = create instance subnet={instance1.privatesubnet} image=ami-3b36fe54 type={instance.type}
count=1 name=wordpress-server-1 # AMI WordPress powered by Bitnami in eu-central-1
inst2 = create instance subnet={instance2.privatesubnet} image=ami-3b36fe54 type={instance.type}
count=1 name=wordpress-server-2
## Register the servers in the targetgroup
check instance id=$inst1 state=running timeout=180
check instance id=$inst2 state=running timeout=180
attach instance id=$inst1 group=$targetgroup
attach instance id=$inst2 group=$targetgroup
```

> awless run

```
> awless run https://raw.githubusercontent.com/wallix/awless-templates/master/wordpress_ha.awls
instance1.private.subnet=subnet-77bac21f instance1.private.subnet.cidr=10.0.10.0/24
instance2.private.subnet=subnet-e78c0c9d instance2.private.subnet.cidr=10.0.11.0/24
wordpress.subnets=subnet-77bac21f,subnet-e78c0c9d wordpress.vpc=vpc-9f6bd2f7

loadbalancerfw = create securitygroup description=wordpress-loadbalancer-security-group name=wordpress-
lb-secgroup vpc=vpc-9f6bd2f7
update securitygroup cidr=0.0.0.0/0 id=$loadbalancerfw inbound=authorize portrange=80 protocol=tcp
targetgroup = create targetgroup name=wordpress-workers port=80 protocol=HTTP vpc=vpc-9f6bd2f7
lb = create loadbalancer groups=$loadbalancerfw name=wordpress-loadbalancer
subnets=subnet-77bac21f,subnet-e78c0c9d
create listener actiontype=forward loadbalancer=$lb port=80 protocol=HTTP target=$targetgroup
inst1 = create instance count=1 image=ami-3b36fe54 name=wordpress-server-1 subnet=subnet-77bac21f
type=t2.micro
inst2 = create instance count=1 image=ami-3b36fe54 name=wordpress-server-2 subnet=subnet-e78c0c9d
type=t2.micro
check instance id=$inst1 state=running timeout=180
check instance id=$inst2 state=running timeout=180
attach instance group=$targetgroup id=$inst1
attach instance group=$targetgroup id=$inst2

Confirm? (y/n): y
[info]    instance status 'pending', expect 'running', retry in 5s (timeout 3m0s).
[info]    instance status 'pending', expect 'running', retry in 5s (timeout 3m0s).
    OK    securitygroup=sg-0329fe68
    OK    targetgroup=arn:aws:elasticloadbalancing:eu-central-1:519101999238:loadbalancer/app/wordpress-
loadbalancer/54b99e06beab1ff7
    OK    loadbalancer=arn:aws:elasticloadbalancing:eu-central-1:519101999238:listener/app/wordpress-
loadbalancer/54b99e06beab1ff7/5da0263108079034
    OK    instance=i-067cc072e1b9bcdbe
    OK    instance=i-074a3cc9b71af8732

[info]    Revert this template with `awless revert 01BBP47HR8RBB5A3HSXW7M577G`
```

awless 0.1.0

- The first major release should be out soon (0.0.24 is a candidate)
- Community-oriented project
  - Contributors welcome!
  - Let’s make the best CLI for AWS together

Roadmap

- More AWS services supported and more features for each service…
  File or upvote existing GitHub issues!

- **0.2.0** Working with multiple regions. Use the local graph to suggest resources in their proper region, and switch easily.

- **0.3.0** A better scripting language. We will soon add a formal semantics, stay tuned.

- **0.4.0** Leverage the local graph to run (complex) queries on the global infrastructure.

Links

- **http://awless.io**
- **@awlessCLI**
- https://github.com/wallix/awless
  - Including Issues + Wiki
- https://github.com/wallix/awless-templates

- **You're welcome to contribute and help us spread the word!**

Demo

awless Workshop

- Come for **2 hours session** with the awless team
  - Automate your infra with awless templates
  - Implement AWS Security Best Practices using awless
  - Get your favorite issue right on top
  - Discuss forthcoming features with the team
- Want to join? Contact hbi@wallix.com