

Number List Parameters

In this series of exercises, we explore advanced features of AWS CloudFormation to create reusable, dynamic, and reliable infrastructure templates. These exercises cover key concepts such as list parameters, AWS-specific parameter types, pseudo parameters, and intrinsic functions like `Fn::Select` and `Fn::Sub`. The focus is on enhancing template flexibility by allowing customizable inputs, validating those inputs against AWS resources, and dynamically generating resource properties based on user-defined or pseudo parameters.

By mastering these techniques, we can efficiently manage infrastructure as code, simplify stack creation processes, and ensure accuracy and consistency across deployments, while also adhering to AWS best practices.

Activity

1. Find the template files in our GitHub repository under the same name as the heading for easy access and edits. Find and Save the attached template locally, open it in VS Code for edits. Copy and rename the Database Security Group resource as `WebServerSecurityGroup` in the template. Update the description of the `WebServerSecurityGroup` to reflect its purpose.

```
43 WebServerSecurityGroup:
44   Type: AWS::EC2::SecurityGroup
45   Properties:
46     VpcId: vpc-0b0c8badfef0024a4
47     GroupDescription: 'Web server instances security group'
48     SecurityGroupIngress:
49     -
```

2. Define a new parameter of type `List`, specifying it will take port numbers as a comma-separated list. Add a description to the parameter explaining the order of ports (e.g., web server port first, database port second).

```
39 SecurityGroupPorts:
40   Type: List<Number>
41   Description: 'Port numbers as a list: <web-server-port>,<database-port>'
42 Resources:
```

3. Use `Fn::Select` to reference the first member of the list for the `FromPort` attribute of the `WebServerSecurityGroup`, with an index of 0. Repeat using `Fn::Select` for the `ToPort` attribute with a shorthand syntax. Update the Database Security Group to use the second member of the list for both `FromPort` and `ToPort`, with an index of 1.

```
CidrIp: 0.0.0.0/0
FromPort:
- Fn::Select: [ 0, !Ref SecurityGroupPorts ]
ToPort: !Select [ 0, !Ref SecurityGroupPorts ]
IpProtocol: tcp
```

- Save the updated template and upload it in the AWS Management Console to create a new stack.
- Provide a comma-separated list of numbers (e.g., 80,3306) for the SecurityGroupPorts parameter during stack creation and skip to the review page and submit.

Parameters
Parameters are defined in your template and allow you to input custom values when you create or update a stack.

AllocatedStorage
Database storage size in GB
8

DbClass
RDS instance class
db.t2.micro

MasterUserPassword
Master user password for the db instance

MasterUsername
Master username for the db instance

MultiAZ
Enable Multi-AZ?
false

SecurityGroupPorts
Port numbers as a list: <web-server-port>,<database-port>
80,3306

- Verify the configuration in the VPC Console by checking the inbound rules of both security groups to ensure the ports were set correctly.

Security Groups (1/1) Info Actions Export security groups to CSV Create security group

Find resources by attribute or tag

<input checked="" type="checkbox"/>	Name	Security group ID	Security group name	VPC ID	Description
<input checked="" type="checkbox"/>	-	sg-0513e2e463855272a	default	vpc-0b0c8badfef0024a4	default

sg-0513e2e463855272a - default

Details **Inbound rules** Outbound rules Sharing - new VPC associations - new Tags

Inbound rules (1) Manage tags Edit inbound rules

Search

<input type="checkbox"/>	Name	Security group rule ID	IP version	Type	Protocol	Port range
<input type="checkbox"/>	-	sgr-0c54bbe92f2c3fbae	-	All traffic	All	All

- Delete the stack to clean up the resources once verified.

String List Parameters

- Find the template files in our GitHub repository under the same name as the heading for easy access and edits. Find and Save the attached template locally, open it in VS Code for edits. Add a new DbSubnets parameter with the type CommaDelimitedList to accept subnet IDs as a comma-separated list of strings. Provide a description for the DbSubnets parameter to clarify its purpose.

```

42 DbSubnets:
43   Type: CommaDelimitedList
44   Description: 'Db subnet ids as a list: <subnet1>,<subnet2>,...'
45 VpcId:

```

2. Reference the DbSubnets parameter directly in the DBSubnetGroup resource where a list of subnet IDs is required.

```
77 # with the IDs of the subnets in your default VPC!
78 DbSubnetGroup:
79   Type: 'AWS::RDS::DBSubnetGroup'
80   Properties:
81     DBSubnetGroupDescription: Subnets to launch db instances into
82     SubnetIds: !Ref DbSubnets
```

3. Define a default value('80,3306') for the SecurityGroupPorts parameter as a list of numbers separated by commas to simplify future stack creation.

```
40   Type: List<Number>
41   Description: 'Port numbers as a list: <web-server-port>,<database-port>'
42   Default: '80,3306'
43   DbSubnets:
44     Type: CommaDelimitedList
45     Description: 'Db subnet ids as a list: <subnet1>,<subnet2>,...'
46   VpcId:
47     Type: AWS::EC2::VPC::Id
```

4. Save the template and upload it in the AWS Management Console to create a new stack.
5. Enter a name for the stack and provide subnet IDs in a comma-separated format from the VPC Console for the DbSubnets parameter. Enter other parameters (e.g., master user credentials) and use default values for parameters like SecurityGroupPorts.

Create stack

Prerequisite - Prepare template

You can also create a template by scanning your existing resources in the [IaC generator](#).

Prepare template

Every stack is based on a template. A template is a JSON or YAML file that contains configuration information about the AWS resources you want to include in the stack.

- ☒ Choose an existing template
Upload or choose an existing template.
- ☐ Use a sample template
Choose from our sample template library.
- ☐ Build from Infrastructure Composer
Create a template using a visual builder.

Specify template Info

A template is a JSON or YAML file that describes your stack's resources and properties.

Template source

Selecting a template generates an Amazon S3 URL where it will be stored.

- ☐ Amazon S3 URL
Provide an Amazon S3 URL to your template.
- ☒ Upload a template file
Upload your template directly to the console.
- ☐ Sync from Git
Sync a template from your Git repository.

Upload a template file

[Choose file](#)

database-stack-template.yaml

JSON or YAML formatted file

<input checked="" type="checkbox"/>	Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR
<input checked="" type="checkbox"/>	-	subnet-0619479663b4084af	Available	vpc-0b0c8badfef0024a4	Off	172.31.16
<input checked="" type="checkbox"/>	-	subnet-08c772de46d053876	Available	vpc-0b0c8badfef0024a4	Off	172.31.0.0
<input checked="" type="checkbox"/>	-	subnet-0a1bbb2feaa84bb95	Available	vpc-0b0c8badfef0024a4	Off	172.31.32

Subnets: subnet-0619479663b4084af, subnet-08c772de46d053876, subnet-0a1bbb2feaa84bb95

Database storage size in GB

DbClass
RDS instance class

DbSubnets
Db subnet ids as a list: <subnet1>,<subnet2>,...

MasterUserPassword
Master user password for the db instance

MasterUsername
Master username for the db instance

MultiAZ
Enable Multi-AZ?

SecurityGroupPorts
Port numbers as a list: <web-server-port>,<database-port>

- Review the parameter values and confirm that the DbSubnets parameter removes any extra spaces during stack creation.
- Verify that the stack and parameters are created correctly and match the provided inputs.

Parameters (8)

<input type="text" value="Search"/>	
Key	Value
AllocatedStorage	8
DbClass	db.t2.micro
DbSubnets	subnet-0619479663b4084af,subnet-08c772de46d053876,subnet-0a1bbb2feaa84bb95
MasterUsername	****
MasterUserPassword	****
MultiAZ	false
SecurityGroupPorts	80,3306
VpcId	vpc-0b0c8badfef0024a4

- Delete the stack to clean up the resources once verified.

Specific Parameter Types

1. Find the template files in our GitHub repository under the same name as the heading for easy access and edits. Find and Save the attached template locally, open it in VS Code for edits. Change the DbSubnets parameter type to ListAWS::EC2::Subnet::Id to validate subnet IDs and define a new parameter VpcId with type AWS::EC2::VPC::Id for validating VPC IDs. Add a description for VpcId, e.g., "A valid VPC ID in your AWS account."

```
43   DbSubnets:
44     Type: List<AWS::EC2::Subnet::Id>
45     Description: 'Db subnet ids as a list: <subnet1>,<subnet2>,...'
46   VpcId:
47     Type: AWS::EC2::VPC::Id
48     Description: A valid VPC id in your AWS account
```

2. Reference VpcId in the VpcId attributes of the WebServerSecurityGroup and DatabaseSecurityGroup resources.

```
51   Type: AWS::EC2::SecurityGroup
52   Properties:
53     VpcId: !Ref VpcId
54     GroupDescription: 'Web server instances security group'
55     SecurityGroupIngress:
56     -
```

```
66   Type: AWS::EC2::SecurityGroup
67   Properties:
68     VpcId: !Ref VpcId
69     GroupDescription: 'Database instances security group'
70     SecurityGroupIngress:
71     -
```

3. Save the updated template and upload it to AWS CloudFormation via "Create stack."
4. Provide valid subnet IDs for DbSubnets using the dropdown list generated by CloudFormation. Select the appropriate VPC ID for the VpcId parameter from the dropdown list. Complete other required parameters (e.g., MasterUserPassword, MasterUsername) and leave defaults where applicable.

DbSubnets
Db subnet ids as a list: <subnet1>,<subnet2>,...

MasterUserPassword
Master user password for the db instance

MasterUsername
Master username for the db instance

MultiAZ
Enable Multi-AZ?

SecurityGroupPorts
Port numbers as a list: <web-server-port>,<database-port>

VpcId
A valid VPC id in your AWS account

5. Skip to the review page and click 'Create stack'. then , wait for the process to complete and check the parameters.

Parameters (8)

Q Search		
Key	Value	Resolve
AllocatedStorage	8	-
DbClass	db.t2.micro	-
DbSubnets	subnet-0619479663b4084af,subnet-08c772de46d053876,subnet-0a1bbb2feaa84bb95	-
MasterUsername	****	-
MasterUserPassword	****	-
MultiAZ	false	-
SecurityGroupPorts	80,3306	-
VpcId	vpc-0b0c8badfef0024a4	-

6. Delete the stack to clean up resources after validation.

Pseudo Parameters and Sub Functions

1. Find the template files in our GitHub repository under the same name as the heading for easy access and edits. Find and Save the attached template locally, open it in VS Code for edits. Understand that pseudo parameters are predefined by AWS CloudFormation and can be referenced like parameters without defining them in the template.
2. Learn to reference pseudo parameters such as AWS::AccountId, AWS::Region, and AWS::StackName using Ref or Fn::Sub.
3. Use the Fn::Sub function to substitute pseudo parameters and variables in strings, either in long or short format.
4. Add a new EC2 instance resource, Bastion, to the existing template. Set properties for the instance: ImageId from the EC2 console, InstanceType as 't2.micro,' and SubnetId using Fn::Select for the first subnet from DbSubnets. Add a Name tag to the instance with the value constructed using AWS::StackName and the literal "-Bastion" with Fn::Sub.

```

49 Resources:
50   Bastion:
51     Type: AWS::EC2::Instance
52     Properties:
53       ImageId: ami-04bd4a6a67aa8e86e
54       InstanceType: t2.micro
55       SubnetId: !Select [ 0, !Ref DbSubnets ]
56       Tags:
57         - Key: Name
58           Value: !Sub '${AWS::StackName}-Bastion'
59

```

- Save the template and upload it to the AWS CloudFormation Console via "Create stack."
- Provide stack parameters, including DB subnets, valid MasterUsername and MasterUserPassword, and VPC ID.

DbSubnets
Db subnet ids as a list: <subnet1>,<subnet2>,...

Select List<AWS::EC2::Subnet::Id>

subnet-0619479663b4084af x subnet-08c772de46d053876 x subnet-0a1bbb2feaa84bb95 x

MasterUserPassword
Master user password for the db instance

MasterUsername
Master username for the db instance

MultiAZ
Enable Multi-AZ?

false

SecurityGroupPorts
Port numbers as a list: <web-server-port>,<database-port>

80,3306

VpcId
A valid VPC id in your AWS account

vpc-0b0c8badfef0024a4

- Launch the stack and verify the EC2 instance in the EC2 Console, checking its name as "StackName-Bastion."
- Test the template with different stack names to observe distinct EC2 instance names.

Dashboard < EC2 Global View Events

▼ Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations

▼ Images
AMIs
AMI Catalog

▼ Elastic Block Store
Volumes
Snapshots
Lifecycle Manager

Instances (1/1) Info Last updated less than a minute ago Connect Instance state Actions Launch Instances

Find Instance by attribute or tag (case-sensitive) All states < 1 > ⚙

<input checked="" type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IP
<input checked="" type="checkbox"/>	DataBaseStack...	i-05fb7babdb6c17fe9	Terminated	t2.micro	—	View alarms +	eu-west-1b	—

i-05fb7babdb6c17fe9 (DataBaseStack-Bastion)

Details Status and alarms Monitoring Security Networking Storage Tags

▼ Instance summary Info

Instance ID i-05fb7babdb6c17fe9	Public IPv4 address —	Private IPv4 addresses —
IPv6 address —	Instance state Terminated	Public IPv4 DNS —
Hostname type		

- Delete the stack after validation to clean up resources.