AWS re:Invent

Becoming a Command Line Expert with the AWS CLI

James Saryerwinnie, Amazon Web Services

November 14, 2013

amazon web services

AWS Command Line Interface

AWS Command Line Interface

Unified tool to manage your AWS services

One Tool

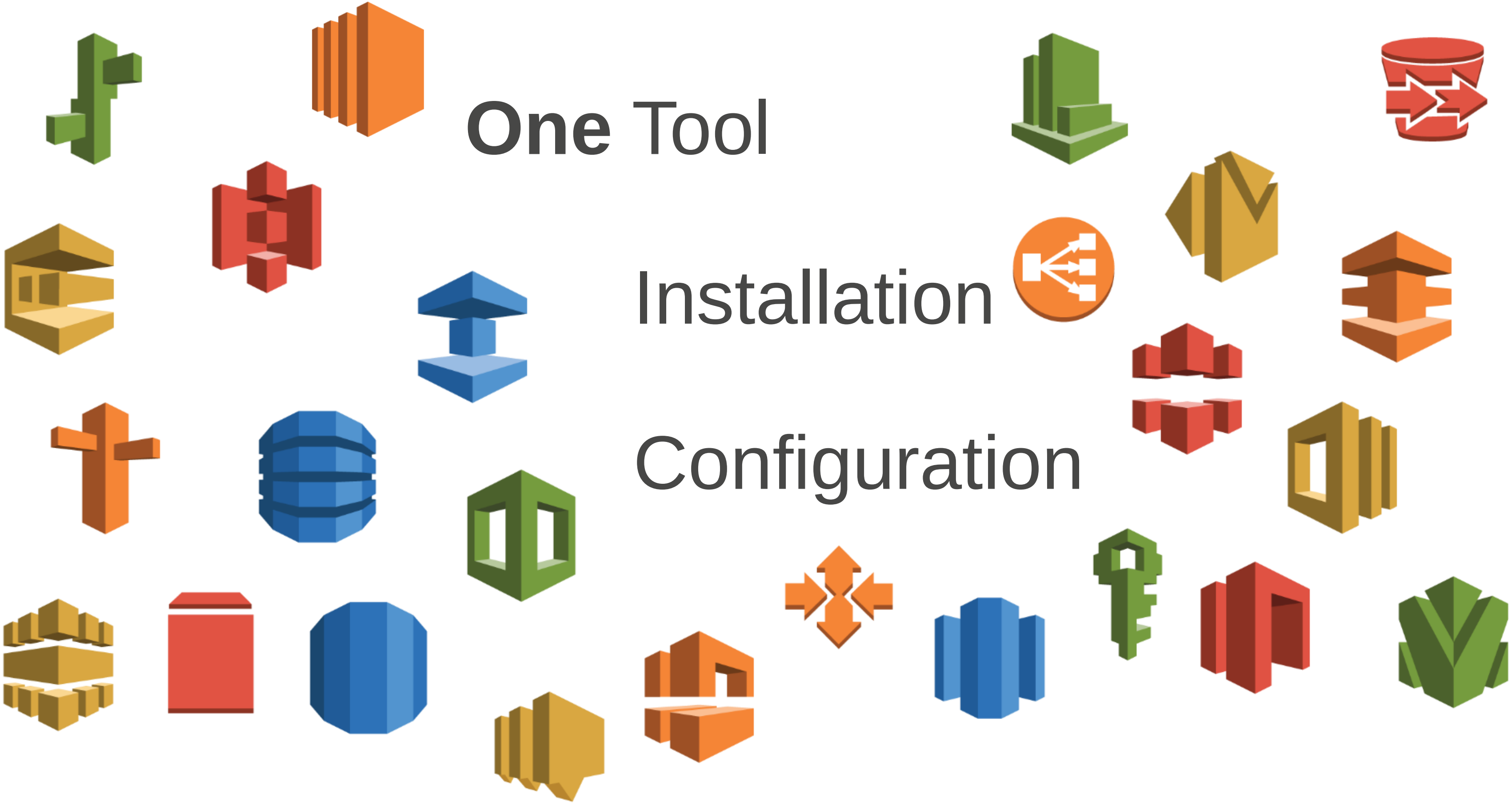
One Tool

Installation

One Tool

Installation

Configuration



One Tool

Installation

Configuration

Installation

Windows:

32-bit MSI: http://s3.amazonaws.com/aws-cli/AWSCLI32.msi

64-bit MSI: http://s3.amazonaws.com/aws-cli/AWSCLI64.msi

Bundled Installer: http://aws-cli.s3.amazonaws.com/awscli-bundle.zip

Bundled Installer

```
$ wget http://s3.amazonaws.com/aws-cli/awscli-bundle.zip
$ unzip awscli-bundle.zip
$ ./install
$ ~/.local/lib/aws/bin/aws --version
```

Pip

```
$ pip install --upgrade awscli
```

One Tool

✓ Installation

Configuration

Credential Configuration

IAM Role	Environment	Config File: ~/.aws/config
Automatic	`AWS_ACCESS_KEY_ID` `AWS_SECRET_ACCESS_KEY`	`aws_access_key_id` `aws_secret_access_key`

Configuring

```
$ aws configure
AWS Access Key ID [None]:
AWS Secret Access Key [None]:
Default region name [None]:
Default output format [json]:
```

Configuring

```
[default]
aws_access_key_id = EXAMPLE
aws_secret_access_key = EXAMPLEKEY
region = us-west-2
output = json
```

One Tool

- ✓ Installation
- ✓ Configuration

One Tool

✓ Installation

✓ Configuration

Demo

Let’s run a command

```
$ aws ec2 describe-instances
{
    "Reservations": []
}
```

```
$ aws ec2 describe-instances
{
    "Reservations": []
}
```

```
$ aws ec2 describe-instances
{
    "Reservations": []
}
```

```
$ aws ec2 describe-instances
{
    "Reservations": []
}
```

```
$ aws ec2 describe-instances
{
    "Reservations": []
}
```

```
$ aws ec2 describe-instances
```

service (command)

operation (subcommand)

Command with Arguments

```
$ aws ec2 import-key-pair \
    --key-name mykey       \
    --public-key-material file:///home/user/.ssh/id_rsa.pub
```

Command with Arguments

```
$ aws ec2 import-key-pair \
    --key-name mykey      \
    --public-key-material file:///home/user/.ssh/id_rsa.pub
```

Command with Arguments

```
$ aws ec2 import-key-pair \
    --key-name mykey      \
    --public-key-material file:///home/user/.ssh/id_rsa.pub
```

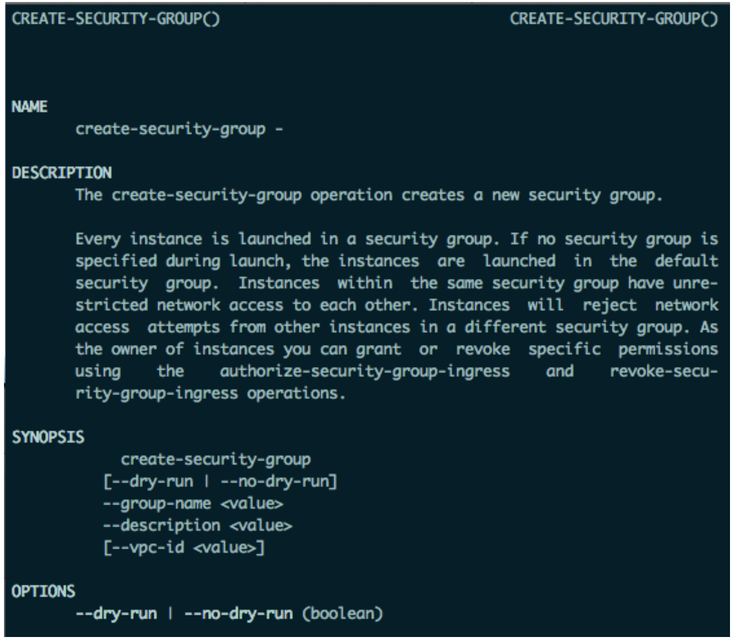
Feature

For any parameter value

- `file://<filename>`
- `http://<url>`
- `https://<url>`

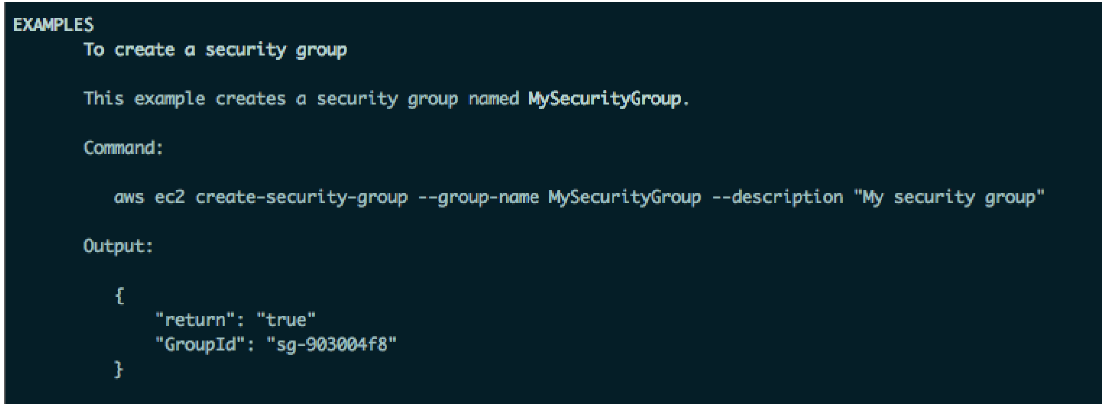
How do I know what arguments to use?

$ aws ec2 create-security-group help



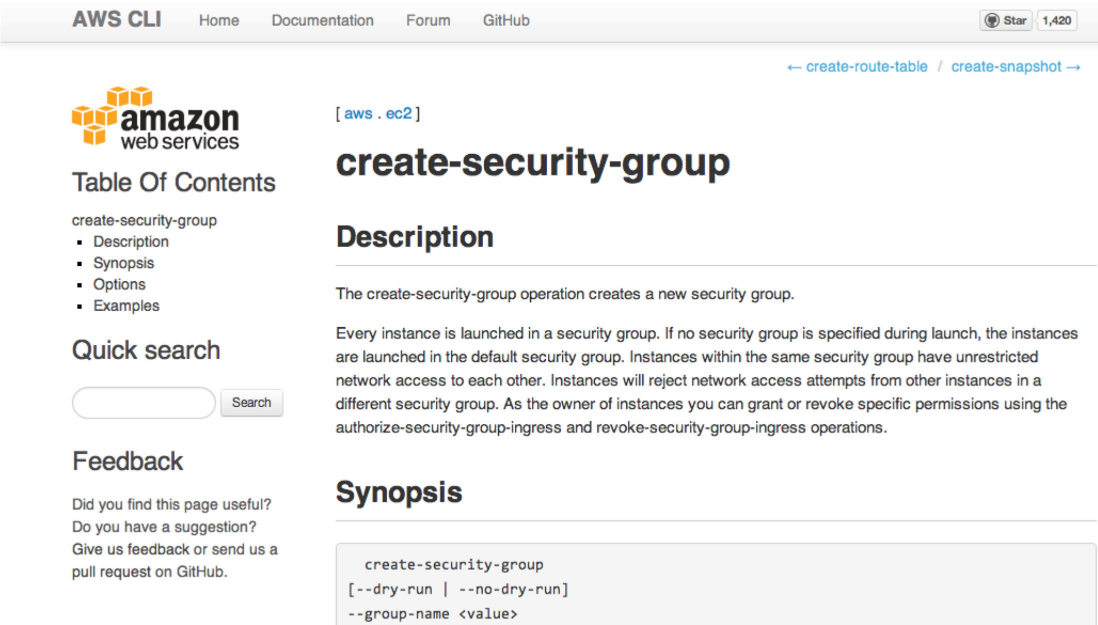
```
CREATE-SECURITY-GROUP()                                  CREATE-SECURITY-GROUP()



NAME
       create-security-group -

DESCRIPTION
       The create-security-group operation creates a new security group.

       Every instance is launched in a security group. If no security group is
       specified during launch, the instances  are  launched  in  the  default
       security  group.  Instances  within  the same security group have unre-
       stricted network access to each other. Instances  will  reject  network
       access  attempts from other instances in a different security group. As
       the owner of instances you can grant  or  revoke  specific  permissions
       using    the    authorize-security-group-ingress    and    revoke-secu-
       rity-group-ingress operations.

SYNOPSIS
            create-security-group
          [--dry-run | --no-dry-run]
          --group-name <value>
          --description <value>
          [--vpc-id <value>]

OPTIONS
       --dry-run | --no-dry-run (boolean)
```

$ aws ec2 create-security-group help



```
EXAMPLES
       To create a security group

       This example creates a security group named MySecurityGroup.

       Command:

          aws ec2 create-security-group --group-name MySecurityGroup --description "My security group"

       Output:

          {
              "return": "true"
              "GroupId": "sg-903004f8"
          }
```

http://docs.aws.amazon.com/cli/latest/reference/ec2/create-security-group.html



Did you find this page useful? Do you have a suggestion? Give us feedback or send us a pull request on GitHub.

[aws . ec2]

create-security-group

Description

The create-security-group operation creates a new security group.

Every instance is launched in a security group. If no security group is specified during launch, the instances are launched in the default security group. Instances within the same security group have unrestricted network access to each other. Instances will reject network access attempts from other instances in a different security group. As the owner of instances you can grant or revoke specific permissions using the authorize-security-group-ingress and revoke-security-group-ingress operations.

Synopsis

```
  create-security-group
[--dry-run | --no-dry-run]
--group-name <value>
```

Feature

Append `help` to any command

- `aws service operation help`
- `aws service help`
- `aws help`

Tab Completion

```
Bash  $ complete -C aws_completer aws
zsh   $ source bin/aws_zsh_completer.sh
tcsh  $ complete aws 'p/*/`aws_completer`/'x
```

$ aws ec2 describe-instances

```
{
    "Reservations": [
        {
            "OwnerId": "",
            "ReservationId": "r-12345",
            "Groups": [
                {
                    "GroupName": "SSH",
                    "GroupId": "sg-abcdefg"
                }
            ],
            "Instances": [
                {
                    "State": {
                        "Code": 16,
                        "Name": "running"
                    },
                    "KeyName": "mykey",
                    "InstanceType": "t1.micro",
                }
            ]
        },
    ]
    …
}
```

Output Formats

JSON	Table	Text
Programmatic processing Integrate with JSON tools	Interactive browsing Easier to visually parse	Piping to text tools Easy to parse

Feature

Multiple output formats

- `--output json`
- `--output table`
- `--output text`

Demo

Regions[*].RegionName

```
{
  "Regions": [{
    "Endpoint": "…",
    "RegionName": "eu-west-1"
  }, {
    "Endpoint": "…",
    "RegionName": "us-east-1"
  }]
}
```

```
eu-west-1
us-east-1
```

```
Reservations[*].Instances[*].[InstanceId,State.Name]
```

```
{"Reservations": [
   "Instances": [{
     "InstanceId": "i-1",
     "State": {"Name": "running"}
   }, {
     "InstanceId": "i-2",
     "State": {"Name": "stopped"}
   }]}
```

```
i-1    running
i-2    stopped
```

```
Reservations[*].Instances[*].{ID: InstanceId,State: State.Name}
```

```
{"Reservations": [
   "Instances": [{
     "InstanceId": "i-1",
     "State": {"Name": "running"}
   }, {
     "InstanceId": "i-2",
     "State": {"Name": "stopped"}
   }]}
```

```
---------------------------
|    DescribeInstances     |
+-------------+-----------+
|     ID      |   State   |
+-------------+-----------+
|  i-1        |  running  |
|  i-2        |  stopped  |
+-------------+-----------+
```

Data	Query	Result
{"foo": "bar"}	foo	"bar"
{"foo": {"bar": "baz"}}	foo.bar	baz
{"foo": [0, 1]}	foo[1]	1
{"bar": 1, "baz": 2}	foo or bar	1
{"a": 1, "b": 2, "c": 3}	[a, b]	[1, 2]
{"a": 1, "b": 2, "c": 3}	{a: a, other: b}	{"a": 1, "other": 2}
[{"a": 1}, {"a": 2}, {"a": 3}, {"a": 4}]	[*].a	[1, 2, 3, 4]

Feature

Query response data

- Use `--query` to create the exact output you want.

http://jmespath.readthedocs.org/en/latest/specification.html

```
aws ec2 describe-instances
  …
  --filters Name=instance-state-name,Values=running
```

```
--filters (list)
         A list of filters used to match properties for Instances. For a com-
         plete reference to the available filter keys for this operation, see
         the Amazon EC2 API reference.

      Shorthand Syntax:

           Key value pairs, where values are separated by commas.
         --filters Name=string1,Values=string1,string2

      JSON Syntax:

         [
           {
             "Name": "string",
             "Values": ["string", ...]
           }
           ...
         ]
```

```
--filters (list)
          A list of filters used to match properties for Instances. For a com-
          plete reference to the available filter keys for this operation, see
          the Amazon EC2 API reference .

       Shorthand Syntax:

            Key value pairs, where values are separated by commas.
          --filters Name=string1,Values=string1,string2

       JSON Syntax:

          [
            {
              "Name": "string",
              "Values": ["string", ...]
            }
            ...
          ]
```

```
--filters (list)
     A list of filters used to match properties for Instances. For a com-
     plete reference to the available filter keys for this operation, see
     the Amazon EC2 API reference .

  Shorthand Syntax:

       Key value pairs, where values are separated by commas.
     --filters Name=string1,Values=string1,string2

  JSON Syntax:

     [
       {
         "Name": "string",
         "Values": ["string", ...]
       }
       ...
     ]
```

--filters

```
Name=instance-state-name,Values=running
```

```
[{"Name": "instance-state-name",
  "Values":["running"]}]
```

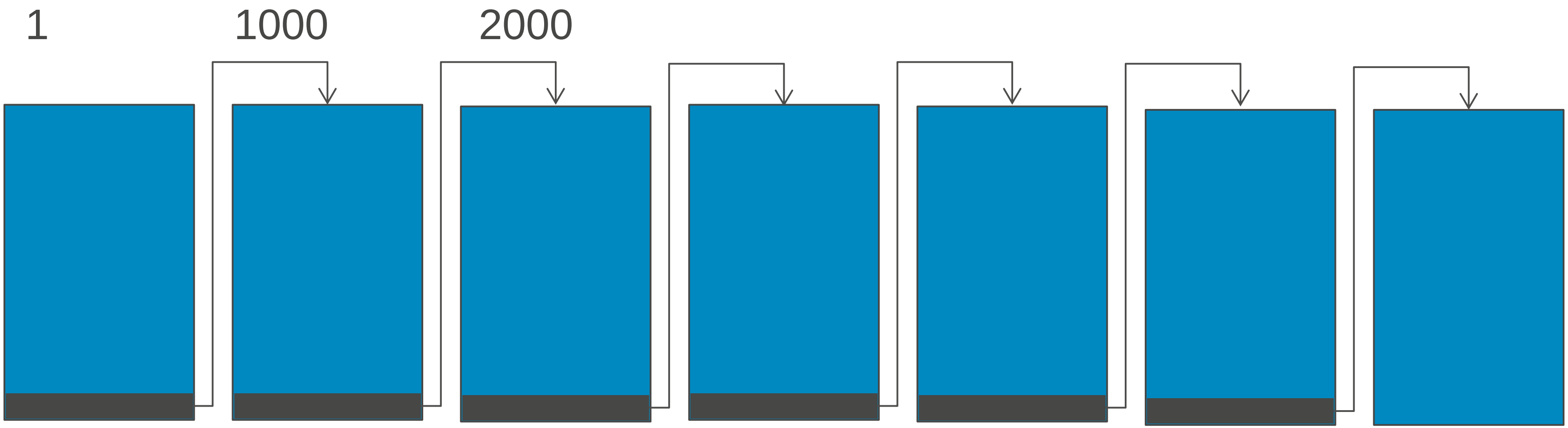
Feature

Shorthand Syntax

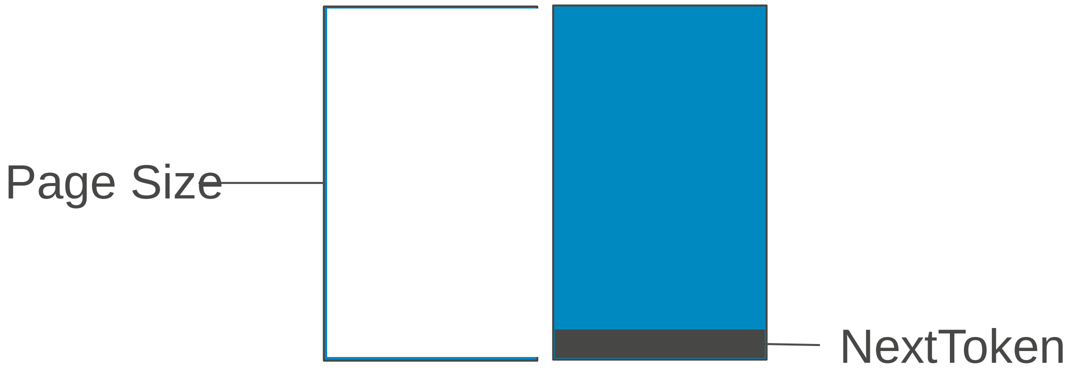
- Use Shorthand Syntax to specify parameter values

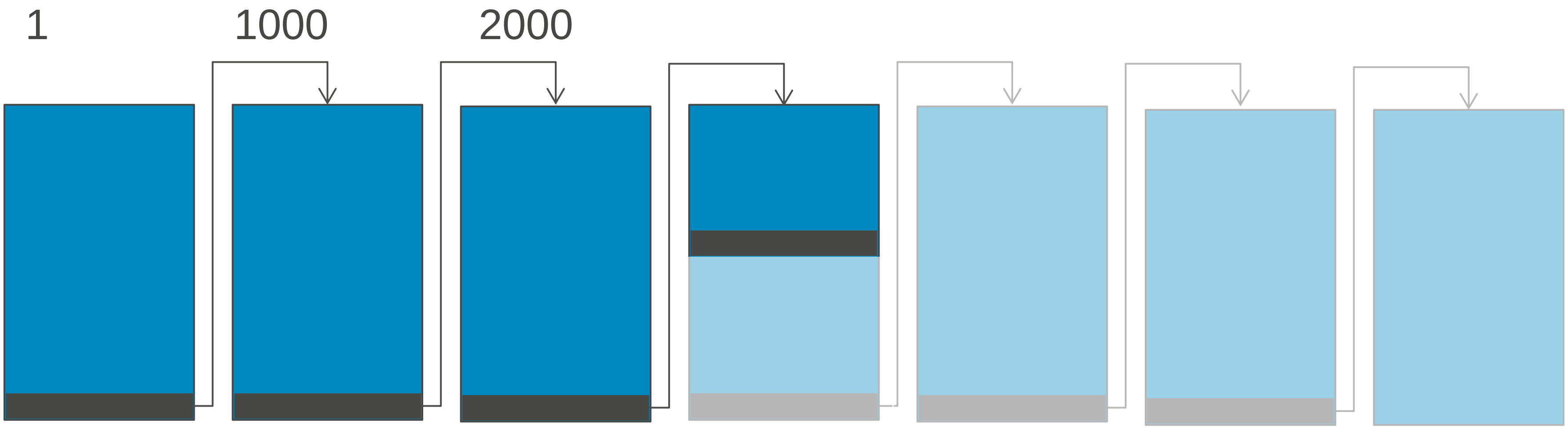
Large Responses

1
100000

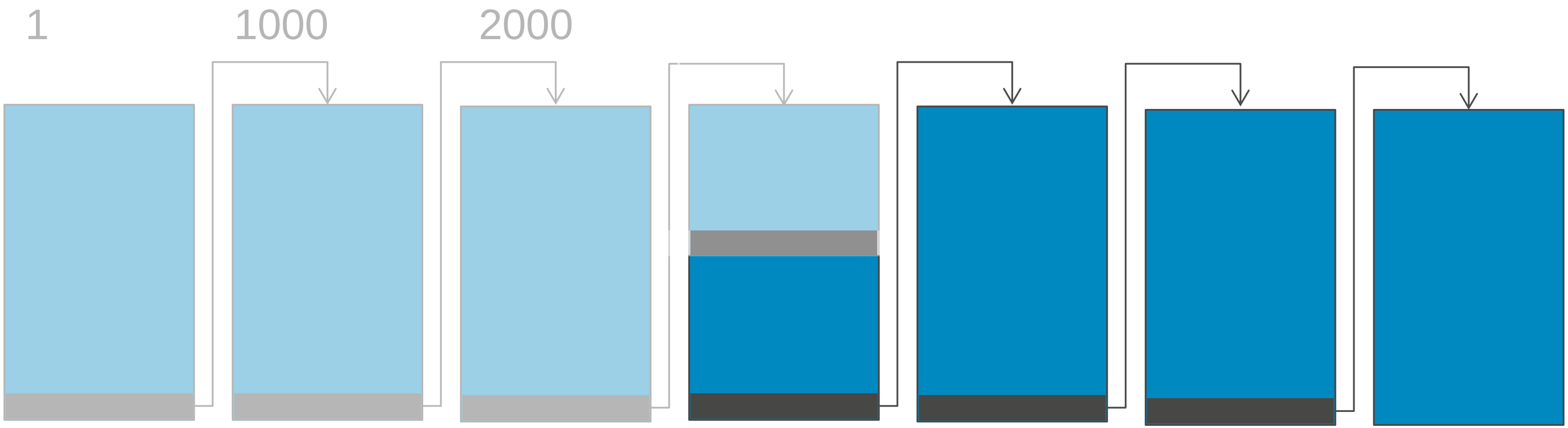

1
1000
2000

1
100000


Page Size
NextToken



--max-items 3500



--max-items 3500 --starting-token <blob>

Feature

Pagination

Use `--starting-token` and `--max-items` to paginate results

We've Learned

- `file://`
- `aws help`
- Tab Completion
- `--output`
- `--query`
- Shorthand Syntax
- Pagination

Static Blog with Amazon S3

- Create blog locally
- Sync to Amazon S3
- AWS Identity and Access Management (IAM)
- Amazon Route53

Amazon S3

```
$ aws s3 mb s3://www.reinvent-cli-blog-demo.com/
$ aws s3 website www.reinvent-cli-blog-demo.com \
    --index-document index.html
```

Create a user that has access only to the static blog

Full access only to the www.reinvent-cli-blog-demo.com bucket

AWS IAM

```
$ aws iam create-user --user-name static-blog
{
    "User": {
        "UserName": "static-blog",
        "Path": "/",
        "CreateDate": "2013-10-18T18:46:39.044Z",
        "UserId": "EXAMPLEUSERID",
        "Arn": "arn:aws:iam::12345:user/static-blog"
    }
}
```

AWS IAM

```
$ aws iam create-access-key --user-name static-blog
{
    "AccessKey": {
        "UserName": "static-blog",
        "Status": "Active",
        "CreateDate": "2013-10-18T18:47:38.913Z",
        "SecretAccessKey": "SECRET_KEY",
        "AccessKeyId": ”ACCESS_KEY"
    }
}
```

AWS IAM

```
$ aws iam put-user-policy --user-name static-blog \
    --policy-name static-blog-s3-access \
    --policy-document file://singlebucket.json
```

AWS IAM

```
$ aws iam put-user-policy --user-name static-blog \
    --policy-name static-blog-s3-access \
    --policy-document file://singlebucket.json
```

AWS IAM

```
{
  "Statement": [
    {
      "Sid": "Stmt12345678",
      "Action": [
        "s3:*"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:s3:::www.reinvent-cli-blog-demo.com/*",
        "arn:aws:s3:::www.reinvent-cli-blog-demo.com"
      ]
    }
  ]
}
```

How do I tell the CLI about this new user?

```
aws configure --profile staticblog
```

```
[default]
…

[profile staticblog]
aws_access_key = ACCESS_KEY
aws_secrete_access_key = SECRET_KEY
region = us-west-2
```

Pushing Blog Content

```
aws s3 sync . s3://www.reinvent-cli-blog-demo.com/ \
    --acl public-read \
    --delete \
    --profile staticblog
```

Pushing Blog Content

```
aws s3 sync . s3://www.reinvent-cli-blog-demo.com/ \
    --acl public-read \
    --delete \
    --profile staticblog
```

Pushing Blog Content

```
aws s3 sync . s3://www.reinvent-cli-blog-demo.com/ \
    --acl public-read \
    --delete \
    --profile staticblog
```

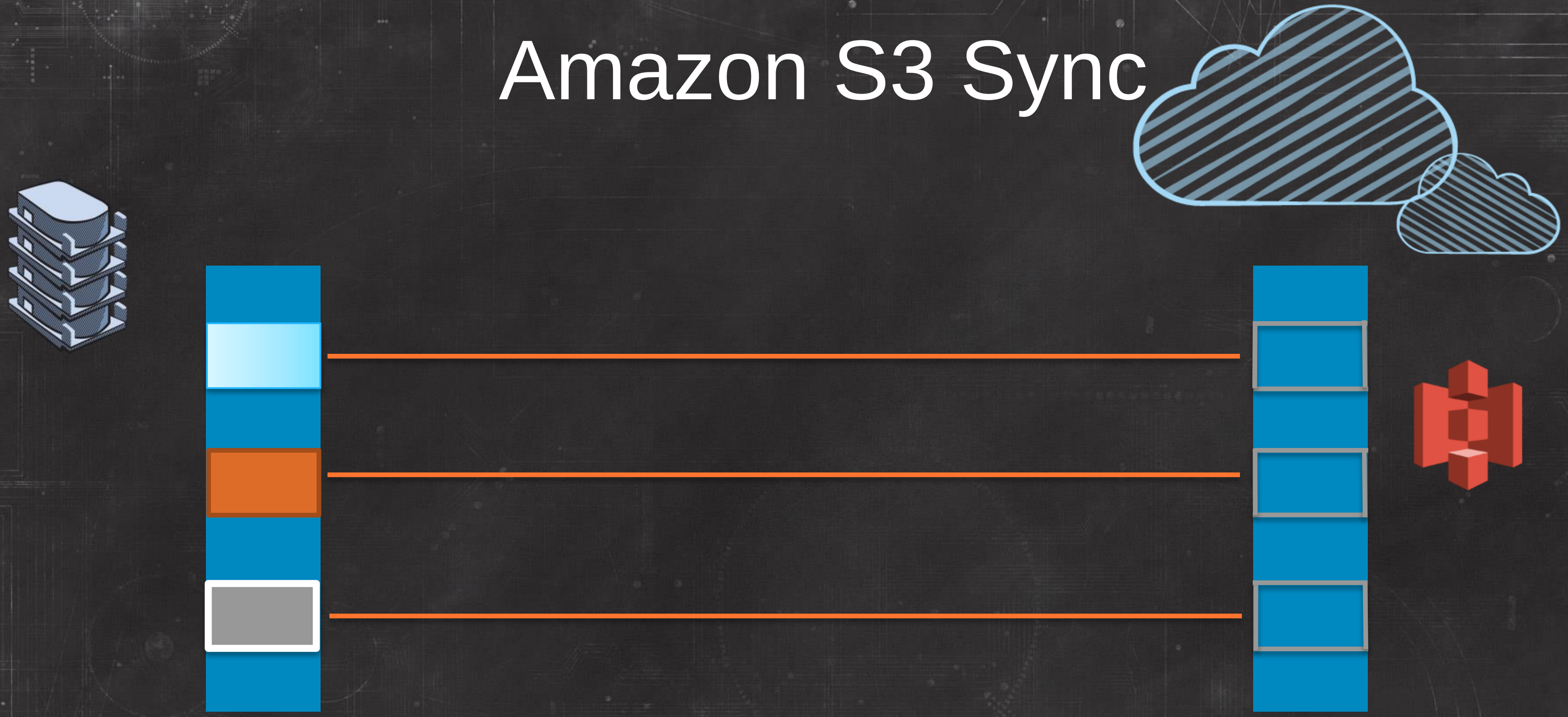
Pushing Blog Content

```
aws s3 sync . s3://www.reinvent-cli-blog-demo.com/ \
    --acl public-read \
    --delete \
    --profile staticblog
```

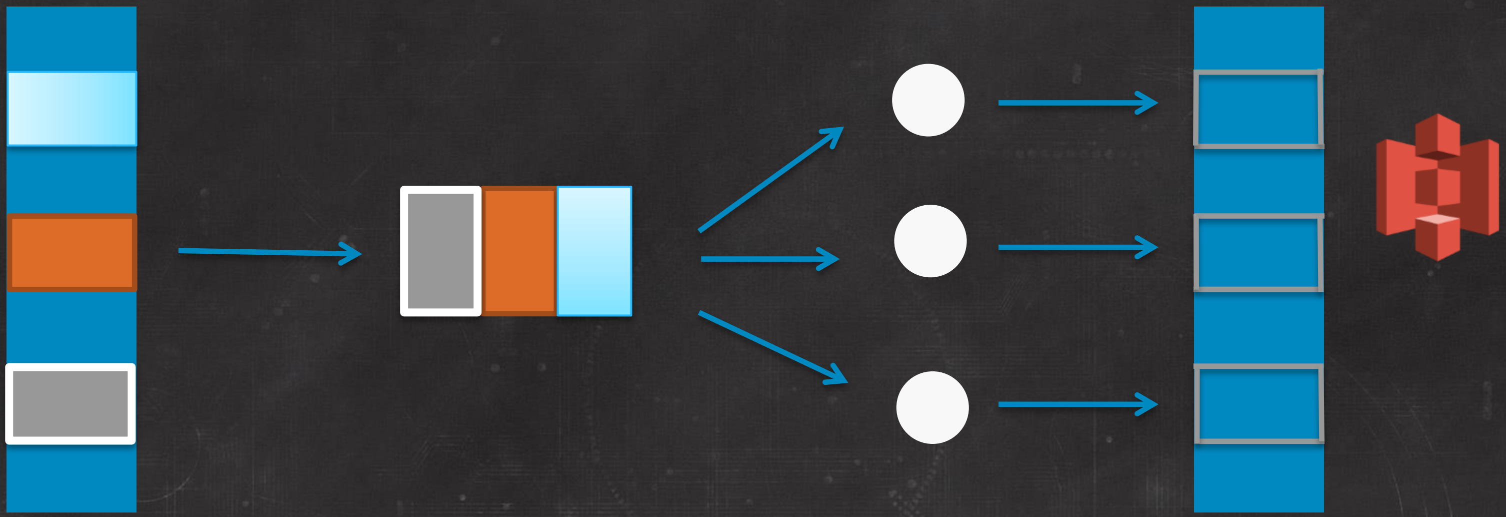
Pushing Blog Content

```
aws s3 sync . s3://www.reinvent-cli-blog-demo.com/ \
    --acl public-read \
    --delete \
    --profile staticblog
```

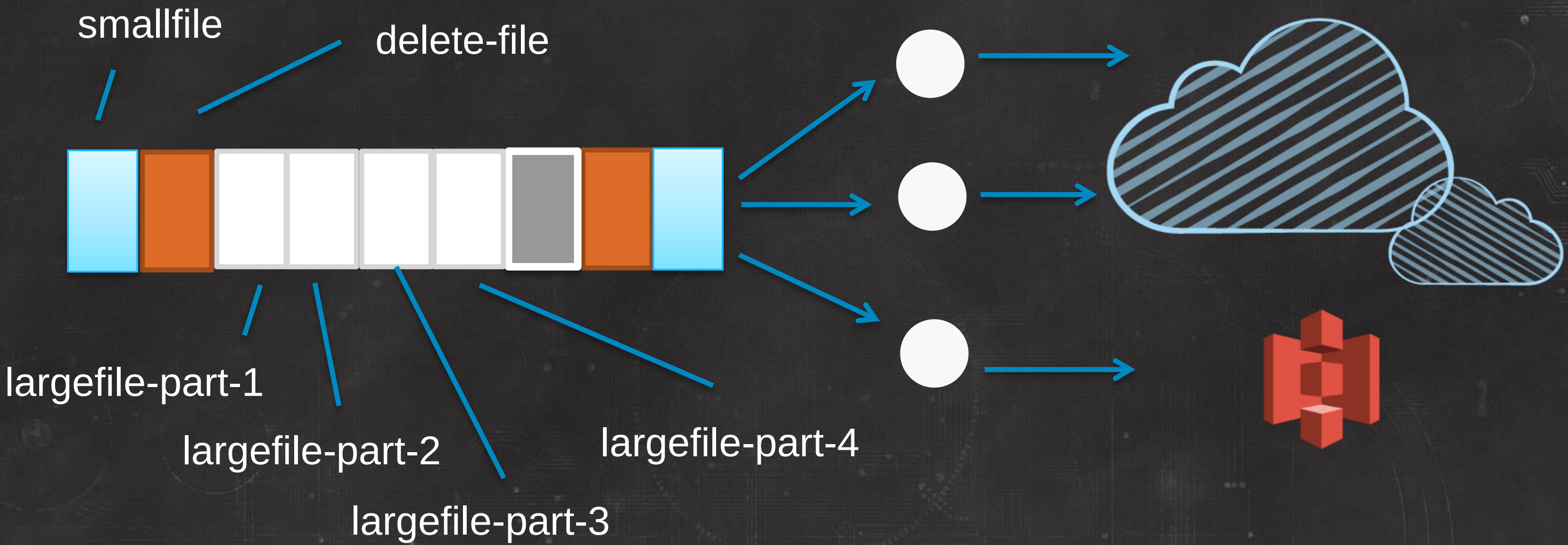
Amazon S3 Sync



Amazon S3 Sync



Amazon S3 Sync



smallfile

delete-file

largefile-part-1

largefile-part-2

largefile-part-3

largefile-part-4

Feature

Amazon S3 Sync

- Sync new/changed files
- Files uploaded in parallel
- Large files split into chunks

Amazon Route53

Amazon Route53

```
$ aws route53 create-hosted-zone --name www.reinvent-cli-blog-demo.com \
     --caller-reference reinvent-cli-blog-demo
{
    "HostedZone": {
        …
        "Id": "/hostedzone/Z1TI9W0V4R87XY",
        "Name": "www.reinvent-cli-blog-demo.com"
    },
    "DelegationSet": {
        "NameServers": [
            "ns-abc.awsdns-20.com",
            "ns-abcd.awsdns-49.org"
            ..
        ]
    }
}
```

Amazon Route53

```
aws route53 change-resource-record-sets --hosted-zone-id "/hostedzone/Z1TI9W0V4R87XY" \
    --change-batch file://changebatch.json
```

```
{
    "ChangeInfo": {
        "Status": "PENDING",
        "Comment": "Add S3 Bucket",
        "SubmittedAt": "2013-10-31T18:37:34.281Z",
        "Id": "/change/C1AG4RL3JT78JG"
    }
}
```

Amazon Route53

```
file://changebatch.json
{
  "Comment": "Add S3 Bucket",
  "Changes": [
    {
      "Action": "CREATE",
      "ResourceRecordSet": {
        "Name": "www.reinvent-cli-blog-demo.com",
        "Type": "A",
        "AliasTarget": {
          "HostedZoneId": "Z3BJ6K6RIION7M",
          "EvaluateTargetHealth": false,
          "DNSName": "s3-website-us-west-2.amazonaws.com"
        }
      }
    }
  ]
}
```

Demo

Additional Topics

Botocore

```
_regions.json           ec2.json                  redshift.json
_retry.json             elasticache.json          route53.json
autoscaling.json        elasticbeanstalk.json     s3.json
cloudformation.json     elastictranscoder.json    ses.json
cloudfront.json         elb.json                  sns.json
cloudsearch.json        emr.json                  sqs.json
cloudwatch.json         iam.json                  storagegateway.json
datapipeline.json       importexport.json         sts.json
directconnect.json      opsworks.json             support.json
dynamodb.json           rds.json                  swf.json
```

ec2.json

```
"api_version": "2013-10-01",
"type": "query",
"signature_version": "v2",
"service_full_name": "Amazon Elastic Compute Cloud",
"service_abbreviation": "Amazon EC2",
"endpoint_prefix": "ec2",
"operations": {
    "ActivateLicense": {
       "name": "ActivateLicense",
       "input": {…},
       "output": {…},
    }
}
```

_retry.json

```
"dynamodb": {
     "__default__": {
       "max_attempts": 10,
       "delay": {
         "type": "exponential",
         "base": 0.05,
         "growth_factor": 2
       },
       "policies": {
         "throughput_exceeded": {
           "applies_when": {
             "response": {
               "service_error_code": "ProvisionedThroughputExceededException",
               "http_status_code": 400
             }
           }
         },
```

Plugins

Next Steps

- Check out the code on Github: http://github.com/aws/aws-cli
- Report bugs at Issues: http://github.com/aws/aws-cli/issues/
- Ask questions on our Forums: https://forums.aws.amazon.com/forum.jspa?forumID=150
- Chat with us in the Developer Lounge (Boto 1:30pm, CLI 3:30pm)

AWS re:Invent

Please give us your feedback on this presentation

TLS304

As a thank you, we will select prize winners daily for completed surveys!

Thank You

amazon web services