

SNS Topics

INTRODUCTION TO AWS BOTO IN PYTHON

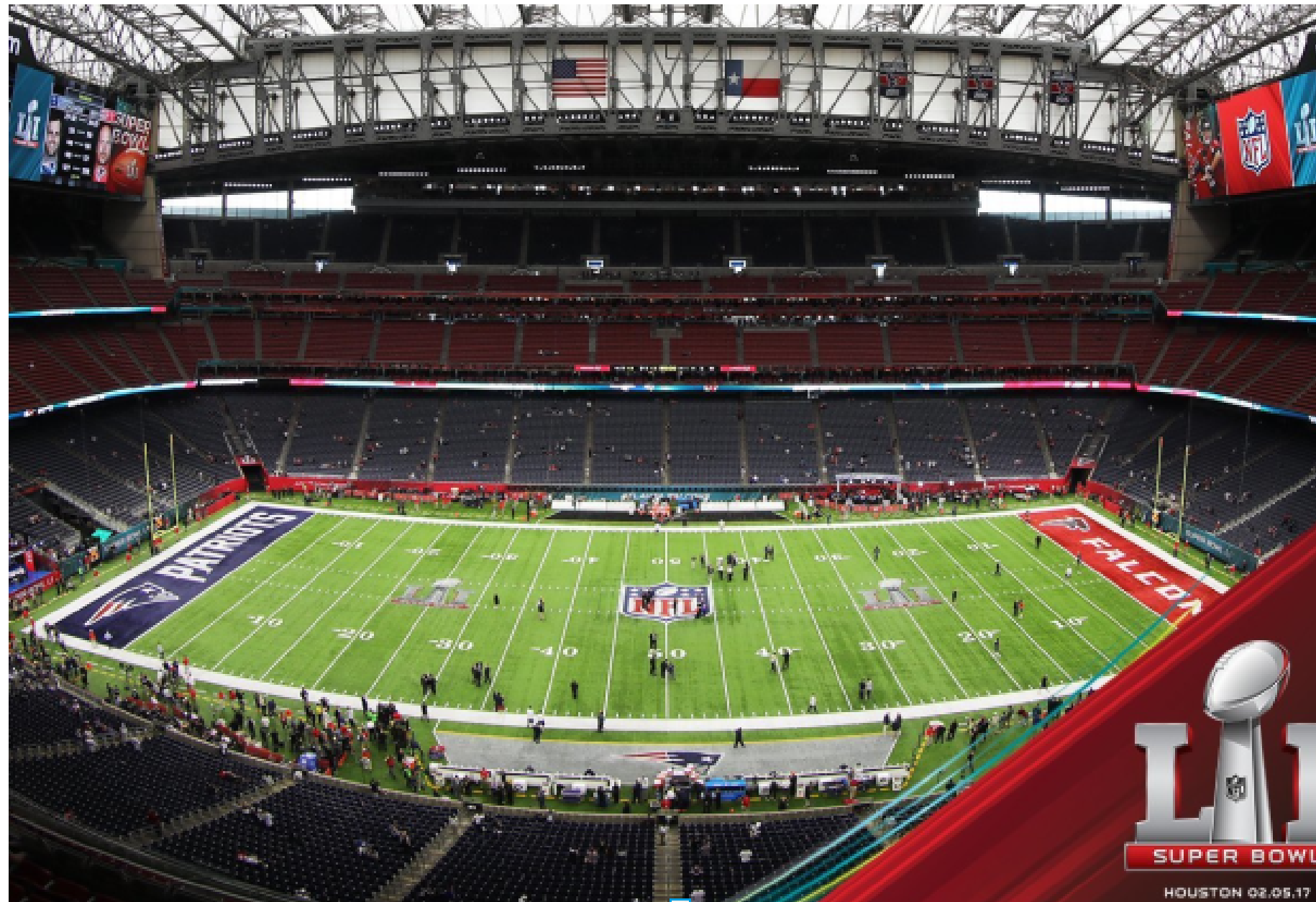


Maksim Pecherskiy
Data Engineer!

SNS



Understanding SNS



Understanding SNS



Understanding SNS



Publisher

Understanding SNS

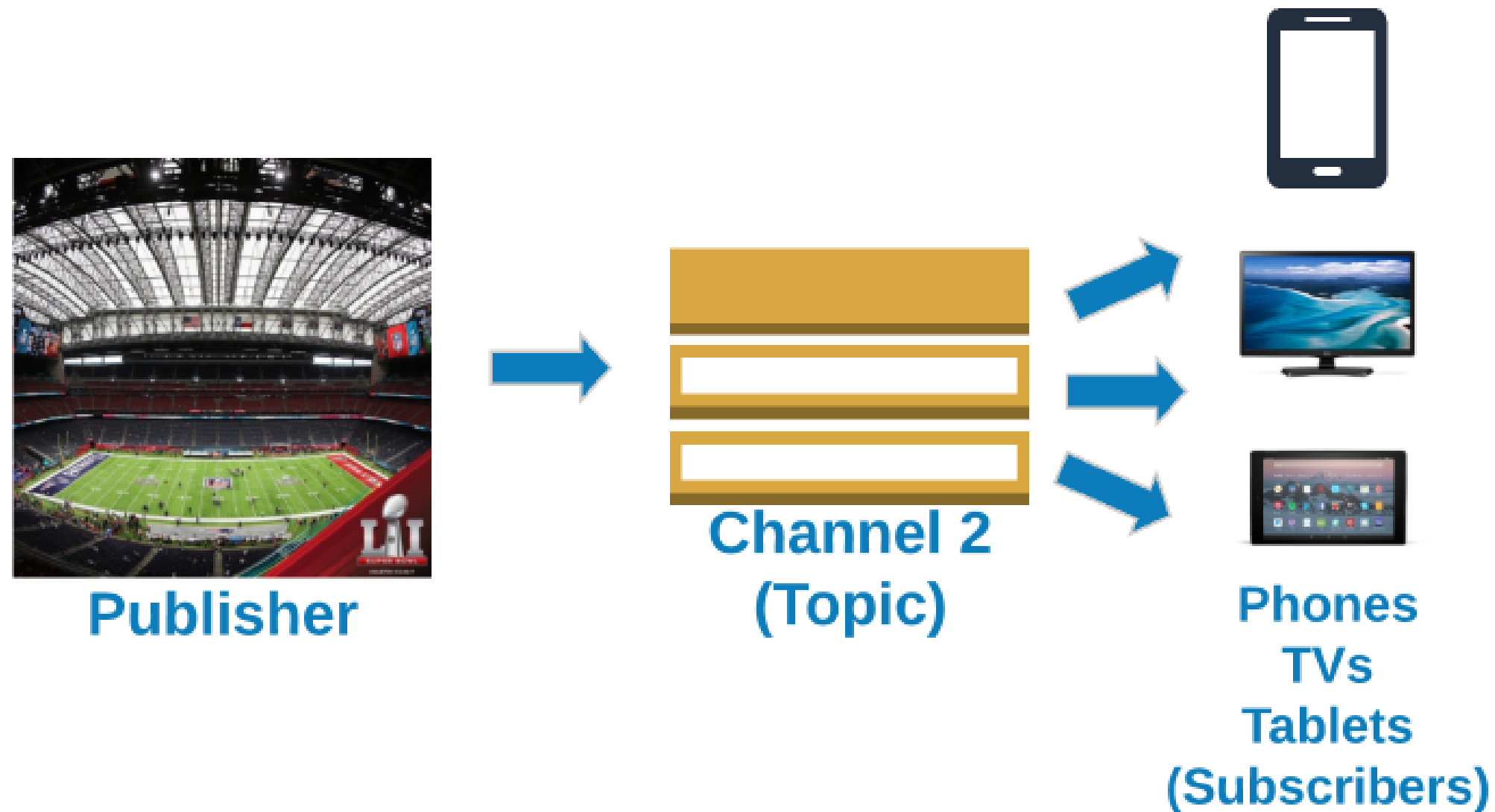


Publisher

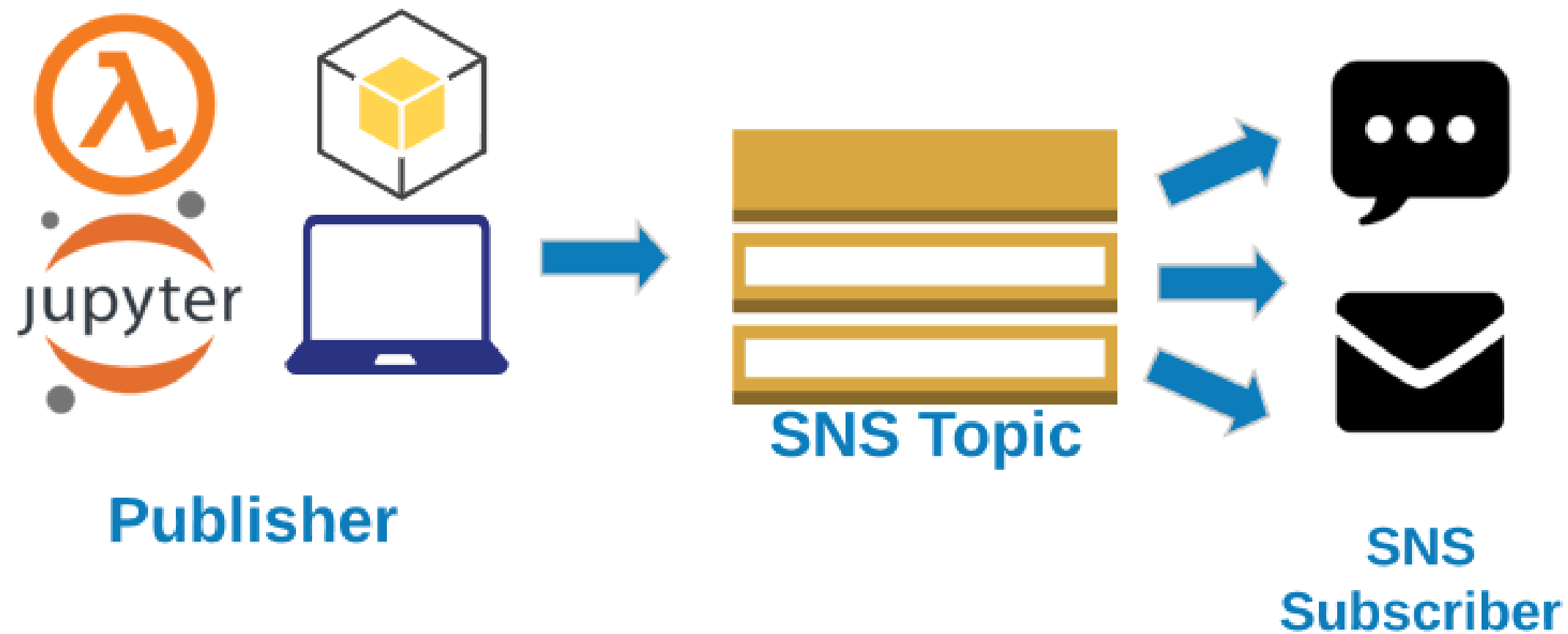


**Channel 2
(Topic)**

Understanding SNS





Understanding SNS



Accessing SNS

The screenshot shows the AWS Management Console interface. At the top, the AWS logo is on the left, and the 'Services' dropdown menu is open, showing 'Resource Groups' and a pin icon. The main heading is 'AWS Management Console'. Below this, the 'AWS services' section is visible, featuring a 'Find Services' search bar with the placeholder text 'Example: Relational Database Service, database, RDS'. Underneath the search bar, the 'Recently visited services' section lists several services: Simple Notification Service, Billing, S3, Lambda, and Systems Manager. On the right side of the console, there are two panels: 'Access reso' (Access Resources) with a mobile app icon and text 'Access the AV App. L', and 'Explore AW' (Explore AWS) with text 'Open Distro f' and 'A 100% open-s'.

SNS Dashboard

 Services ▾ Resource Groups ▾ 

Amazon SNS X

Dashboard

Topics

Subscriptions

▼ Mobile

Push notifications

Text messaging (SMS)

Amazon SNS > Dashboard

Dashboard

Resources for us-east-1

Topics

4

SNS Topics

city_alerts

EditDeletePublish message

Details

Name

city_alerts

ARN

arn:aws:sns:us-east-1:320333787981:city_alerts

Display name

-

Topic owner

320333787981

Subscriptions

Access policy

Delivery retry policy (HTTP/S)

Delivery status logging

Encryption

Tags

Subscriptions (2)

EditDeleteRequest confirmationConfirm subscriptionCreate subscription

Q Search

< 1 > ⚙

	ID ▼	Endpoint ▼	Status ▼	Protocol ▲
<input type="radio"/>	563a6f29-8328-4804-b0ba-7d7c4397b742	max@maksimize.com	✔ Confirmed	EMAIL
<input type="radio"/>	ff46b9e6-d214-4965-9369-f4838e8e614f	+17736777	✔ Confirmed	SMS

SNS Topics

city_alerts

EditDeletePublish message

Details

Name

city_alerts

ARN

arn:aws:sns:us-east-1:320333787981:city_alerts

Display name

-

Topic owner

320333787981

Subscriptions

Access policy

Delivery retry policy (HTTP/S)

Delivery status logging

Encryption

Tags

Subscriptions (2)

EditDeleteRequest confirmationConfirm subscriptionCreate subscription

Q Search

< 1 > ⚙

	ID	Endpoint	Status	Protocol
<input type="radio"/>	563a6f29-8328-4804-b0ba-7d7c4397b742	max@maksimize.com	✔ Confirmed	EMAIL
<input type="radio"/>	ff46b9e6-d214-4965-9369-f4838e8e614f	+1773671	✔ Confirmed	SMS

Creating an SNS Topic

```
sns = boto3.client('sns',  
                    region_name='us-east-1',  
                    aws_access_key_id=AWS_KEY_ID,  
                    aws_secret_access_key=AWS_SECRET)
```

Creating an SNS Topic

```
response = sns.create_topic(Name='city_alerts')
```

Creating an SNS Topic

```
{'TopicArn': 'arn:aws:sns:us-east-1:320333787981:city_alerts',  
  'ResponseMetadata': {  
    'RequestId': '1cf4a178-1d1e-54fa-b270-f408645e1000',  
    'HTTPStatusCode': 200,  
    'HTTPHeaders': {  
      'x-amzn-requestid': '1cf4a178-1d1e-54fa-b270-f408645e1000',  
      'content-type': 'text/xml',  
      'content-length': '318',  
      'date': 'Tue, 04 Jun 2019 13:49:52 GMT'  
    },  
    'RetryAttempts': 0  
  }  
}
```

Creating an SNS Topic

```
topic_arn = response['TopicArn']
```

Or... a shortcut

```
sns.create_topic(Name='city_alerts')['TopicArn']
```


Creating an SNS Topic

city_alerts

EditDeletePublish message

Details

Name	city_alerts	Display name	-
ARN	arn:aws:sns:us-east-1:320333787981:city_alerts	Topic owner	320333787981

Permissions

Summary

[Delete user](#)

User ARN `arn:aws:iam::320333787981:user/datacampDemoUser2`

Path `/`

Creation time 2019-04-03 16:48 PDT

[Permissions](#)[Groups](#)[Tags](#)[Security credentials](#)[Access Advisor](#)

▼ Permissions policies (4 policies applied)

[Add permissions](#)[+ Add inline policy](#)

Policy name ▼	Policy type ▼	
Attached directly		
▶ AmazonS3FullAccess	AWS managed policy	×
▶ ComprehendFullAccess	AWS managed policy	×
▶ AmazonRekognitionFullAccess	AWS managed policy	×
▶ AmazonSNSFullAccess	AWS managed policy	×

IAM gives us access!



Listing topics

```
response = sns.list_topics()
```

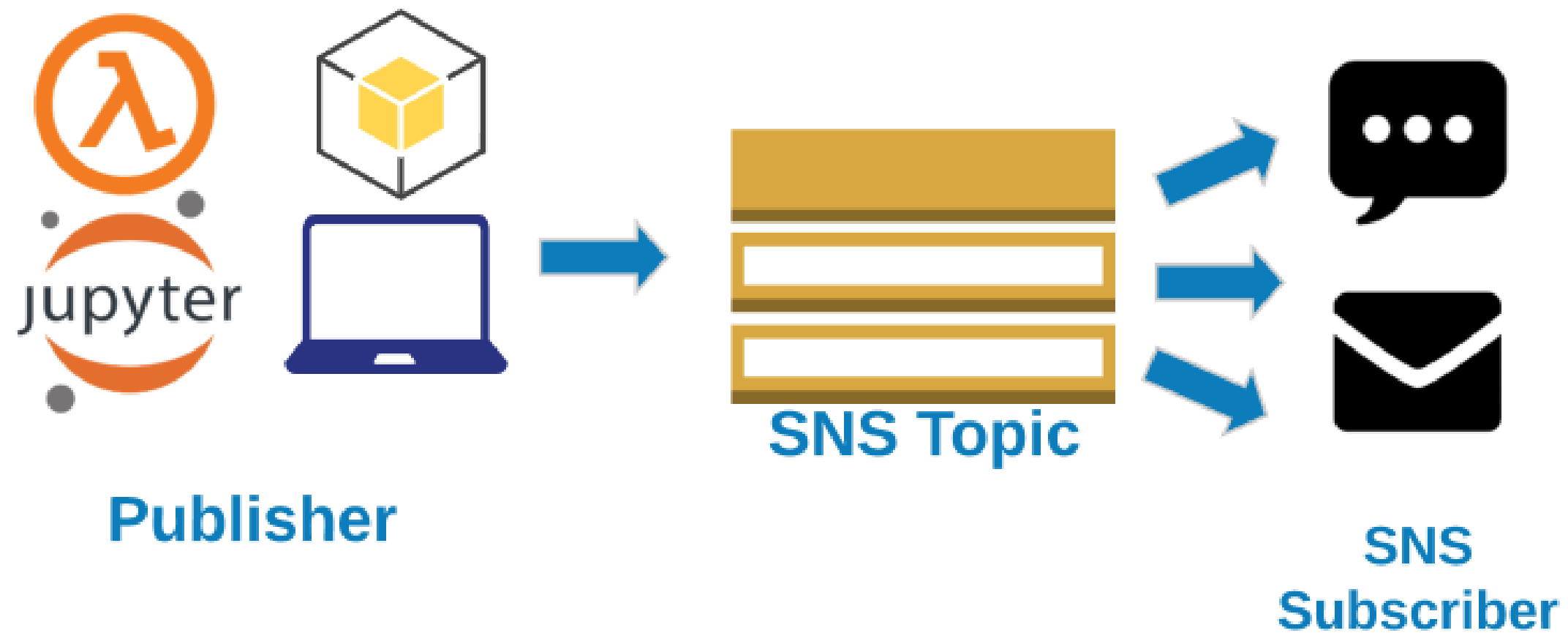
Listing topics

```
'Topics': [{ 'TopicArn': 'arn:aws:sns:us-east-1:320333787981:box_alerts' },
             { 'TopicArn': 'arn:aws:sns:us-east-1:320333787981:city_alerts' },
             { 'TopicArn': 'arn:aws:sns:us-east-1:320333787981:first_topic' },
             { 'TopicArn': 'arn:aws:sns:us-east-1:320333787981:test_topic' } ],
'ResponseMetadata': { 'RequestId': '7ed46745-aae4-5a97-b34f-3235d40a3109' ,
                      'HTTPStatusCode': 200,
                      'HTTPHeaders': { 'x-amzn-requestid': '7ed46745-aae4-5a97-b34f-3235d40a3109' ,
                                         'content-type': 'text/xml' ,
                                         'content-length': '695' ,
                                         'date': 'Tue, 04 Jun 2019 14:14:06 GMT' } ,
```

Deleting topics

```
sns.delete_topic(TopicArn='arn:aws:sns:us-east-1:320333787981:city_alerts')
```

Review



Review

Create SNS Client

```
sns = boto3.client('sns',  
                    region_name='us-east-1',  
                    aws_access_key_id=AWS_KEY_ID,  
                    aws_secret_access_key=AWS_SECRET)
```

Create a topic

```
response = sns.create_topic(Name='city_alerts')  
topic_arn = response['TopicArn']
```

Review

List Topics

```
response = sns.list_topics()  
topics = response['Topics']
```

Delete a topic

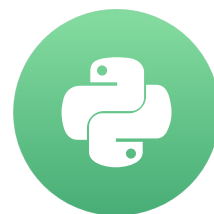
```
sns.delete_topic(TopicArn='arn:aws:sns:us-east-1:320333787981:city_alerts')
```


Let's practice!

INTRODUCTION TO AWS BOTO IN PYTHON

SNS Subscriptions

INTRODUCTION TO AWS BOTO IN PYTHON






Maksim Pecherskiy
Data Engineer

Subscription Listing

Subscriptions (2)					Edit	Delete	Request confirmation	Confirm subscription	Create subscription
<input type="text" value="Search"/>					< 1 > ⚙				
	ID ▼	Endpoint ▼	Status ▼	Protocol ▲					
<input type="radio"/>	Pending confirmation	max@maksimize.com	⌚ Pending confirmation	EMAIL					
<input type="radio"/>	9f2dad1d-8844-4fe8-86f7-3f627ae8420f	+17736777755	✅ Confirmed	SMS					


Subscription Listing

Subscriptions (2)					Edit Delete Request confirmation Confirm subscription Create subscription	
<input type="text" value="Search"/>					< 1 > 	
	ID ▼	Endpoint ▼	Status ▼	Protocol ▲		
<input type="radio"/>	Pending confirmation	max@maksimize.com	 Pending confirmation	EMAIL		
<input type="radio"/>	9f2dad1d-8844-4fe8-86f7-3f627ae8420f	+17736777755	 Confirmed	SMS		

Subscription Listing

Subscriptions (2)					Edit	Delete	Request confirmation	Confirm subscription	Create subscription
<input type="text" value="Search"/>					< 1 > ⚙				
	ID	Endpoint	Status	Protocol					
<input type="radio"/>	Pending confirmation	max@maksimize.com	⌚ Pending confirmation	EMAIL					
<input type="radio"/>	9f2dad1d-8844-4fe8-86f7-3f627ae8420f	+17736777755	✅ Confirmed	SMS					

Subscription Listing

Subscriptions (2)					Edit Delete Request confirmation Confirm subscription Create subscription	
<input type="text" value="Search"/>					< 1 > 	
	ID ▼	Endpoint ▼	Status ▼	Protocol ▲		
<input type="radio"/>	Pending confirmation	max@maksimize.com	⌚ Pending confirmation	EMAIL		
<input type="radio"/>	9f2dad1d-8844-4fe8-86f7-3f627ae8420f	+17736777755	✅ Confirmed	SMS		

Creating an SMS subscription.

```
sns = boto3.client('sns',  
                    region_name='us-east-1',  
                    aws_access_key_id=AWS_KEY_ID,  
                    aws_secret_access_key=AWS_SECRET)
```

```
response = sns.subscribe(  
    TopicArn = 'arn:aws:sns:us-east-1:320333787981:city_alerts',  
    Protocol = 'SMS',  
    Endpoint = '+13125551123')
```

Create an SMS subscription.

```
{ 'SubscriptionArn' : 'arn:aws:sns:us-east-1:320333787981:city_alerts:9f2dad1d-8844-  
1b1e-4b1b-b1b1-b1b1-b1b1-b1b1-b1b1-b1b1',  
  'RequestId' : '9384ec7d-c9bc-5194-ace8-b90d46ce13d4',  
  'HTTPStatusCode' : 200,  
  'HTTPHeaders' : {'x-amzn-requestid' : '9384ec7d-c9bc-5194-ace8-b90d46ce13d4',  
    'content-type' : 'text/xml',  
    'content-length' : '361',  
    'date' : 'Tue, 04 Jun 2019 15:24:33 GMT'},  
  'RetryAttempts' : 0}}
```


Creating an email subscription

```
response = sns.subscribe(  
    TopicArn = 'arn:aws:sns:us-east-1:320333787981:city_alerts',  
    Protocol='email',  
    Endpoint='max@maksimize.com')
```

Creating an email subscription

```
{ 'SubscriptionArn': 'pending confirmation',  
  'ResponseMetadata': { 'RequestId': 'a41c8047-d512-508e-ab52-d53746e67556',  
    'HTTPStatusCode': 200,  
    'HTTPHeaders': { 'x-amzn-requestid': 'a41c8047-d512-508e-ab52-d53746e67556',  
      'content-type': 'text/xml',  
      'content-length': '298',  
      'date': 'Tue, 04 Jun 2019 15:43:48 GMT' },  
    'RetryAttempts': 0 }}
```

Creating an email subscription



Confirmed email address

<input type="radio"/>	d7b42c58-21d3-4e3c-b25d-96c23032eb2f	max@maksimize.com	<input checked="" type="checkbox"/> Confirmed	EMAIL
-----------------------	---	-------------------	---	-------

Listing subscriptions by Topic

```
sns.list_subscriptions_by_topic(  
    TopicArn='arn:aws:sns:us-east-1:320333787981:city_alerts')
```

Listing subscriptions

```
{ 'Subscriptions': [{ 'SubscriptionArn': 'PendingConfirmation',  
  'Owner': '320333787981',  
  'Protocol': 'email',  
  'Endpoint': 'max@maksimize.com',  
  'TopicArn': 'arn:aws:sns:us-east-1:320333787981:city_alerts' },  
  { 'SubscriptionArn': 'arn:aws:sns:us-east-1:320333787981:city_alerts:9f2dad1d-8844-  
    'Owner': '320333787981',  
    'Protocol': 'sms',  
    'Endpoint': '+17736777755',  
    'TopicArn': 'arn:aws:sns:us-east-1:320333787981:city_alerts' } ],  
  'ResponseMetadata': { 'RequestId': '17fed597-17e7-5669-bf53-80e79c08dd7f' },
```

Listing subscriptions

```
sns.list_subscriptions()['Subscriptions']
```

Deleting subscriptions

```
sns.unsubscribe(  
    SubscriptionArn='arn:aws:sns:us-east-1:320333787981:city_alerts:9f2dad1d-8844-4fe8-861'  
)
```

Deleting multiple subscriptions

Get list of subscriptions

```
response = sns.list_subscriptions_by_topic(  
    TopicArn='arn:aws:sns:us-east-1:320333787981:city_alerts')  
subs = response['Subscriptions']
```

Unsubscribe SMS subscriptions

```
for sub in subs:  
    if sub['Protocol'] == 'sms':  
        sns.unsubscribe(sub['SubscriptionArn'])
```


Review

SMS

- `Protocol='sms'`
- `Endpoint='+13122334433'`
- `Status: 'confirmed'`

Email

- `Protocol='email'`
- `Endpoint='email@address.com'`
- `Status: 'confirmed'`
- `Status: 'pending confirmation'`

Review

Create a subscription

```
response = sns.subscribe(  
    TopicArn = 'arn:aws:sns:us-east-1:320333787981:city_alerts',  
    Protocol = 'sms',  
    Endpoint = '+13125551123' )
```

List subscriptions by topic

```
response = sns.list_subscriptions_by_topic(  
    TopicArn='arn:aws:sns:us-east-1:320333787981:city_alerts' )  
subs = response['Subscriptions']
```

Review

List subscriptions

```
sns.list_subscriptions()['Subscriptions']
```

Delete a subscription

```
sns.unsubscribe(  
    SubscriptionArn='arn:aws:sns:us-east-1:320333787981:city_alerts:9f2dad1d-8844-4fe8-861'  
)
```

Let's practice!

INTRODUCTION TO AWS BOTO IN PYTHON

Sending messages

INTRODUCTION TO AWS BOTO IN PYTHON



Maksim Pecherskiy
Data engineer

Publishing to a Topic

```
response = sns.publish(  
    TopicArn = 'arn:aws:sns:us-east-1:320333787981:city_alerts',  
    Message = 'Body text of SMS or e-mail',  
    Subject = 'Subject Line for Email'  
)
```



SMS



email

Publishing to a Topic

```
response = client.publish(  
    TopicArn = 'arn:aws:sns:us-east-1:320333787981:city_alerts',  
    Message = 'Body text of SMS or e-mail',  
    Subject = 'Subject Line for Email'  
)
```



SMS



email

Publishing to a Topic

```
response = client.publish(  
    TopicArn = 'arn:aws:sns:us-east-1:320333787981:city_alerts',  
    Message = 'Body text of SMS or e-mail',  
    Subject = 'Subject Line for Email'  
)
```



email

Sending custom messages

```
num_of_reports = 137

response = client.publish(
    TopicArn = 'arn:aws:sns:us-east-1:320333787981:city_alerts',
    Message = 'There are {} reports outstanding'.format(num_of_reports),
    Subject = 'Subject Line for Email'
)
```

Sending a single SMS

```
response = sns.publish(  
    PhoneNumber = '+13121233211',  
    Message = 'Body text of SMS or e-mail'  
)
```



SMS

Not a good long term practice

- One-off texts = getting stuff done
- Topics and subscribers = maintainability

Publish to Topic vs Single SMS

Publish to a topic

- Have to have a topic
- Our topic has to have subscriptions
- Better for multiple receivers
- Easier list management



SMS



email

Send a single SMS

- Don't need a topic
- Don't need subscriptions
- Just sends a message to a phone number
- Email option not available



SMS

Review

Publish to a topic

```
response = sns.publish(  
    TopicArn = 'arn:aws:sns:us-east-1:320333787981:city_alerts',  
    Message = 'Body text of SMS or e-mail',  
    Subject = 'Subject Line for Email'  
)
```

Send a single SMS

```
response = sns.publish(  
    PhoneNumber = '+13121233211',  
    Message = 'Body text of SMS or e-mail'  
)
```

Let's practice!

INTRODUCTION TO AWS BOTO IN PYTHON

Case Study: Building a notification system

INTRODUCTION TO AWS BOTO IN PYTHON

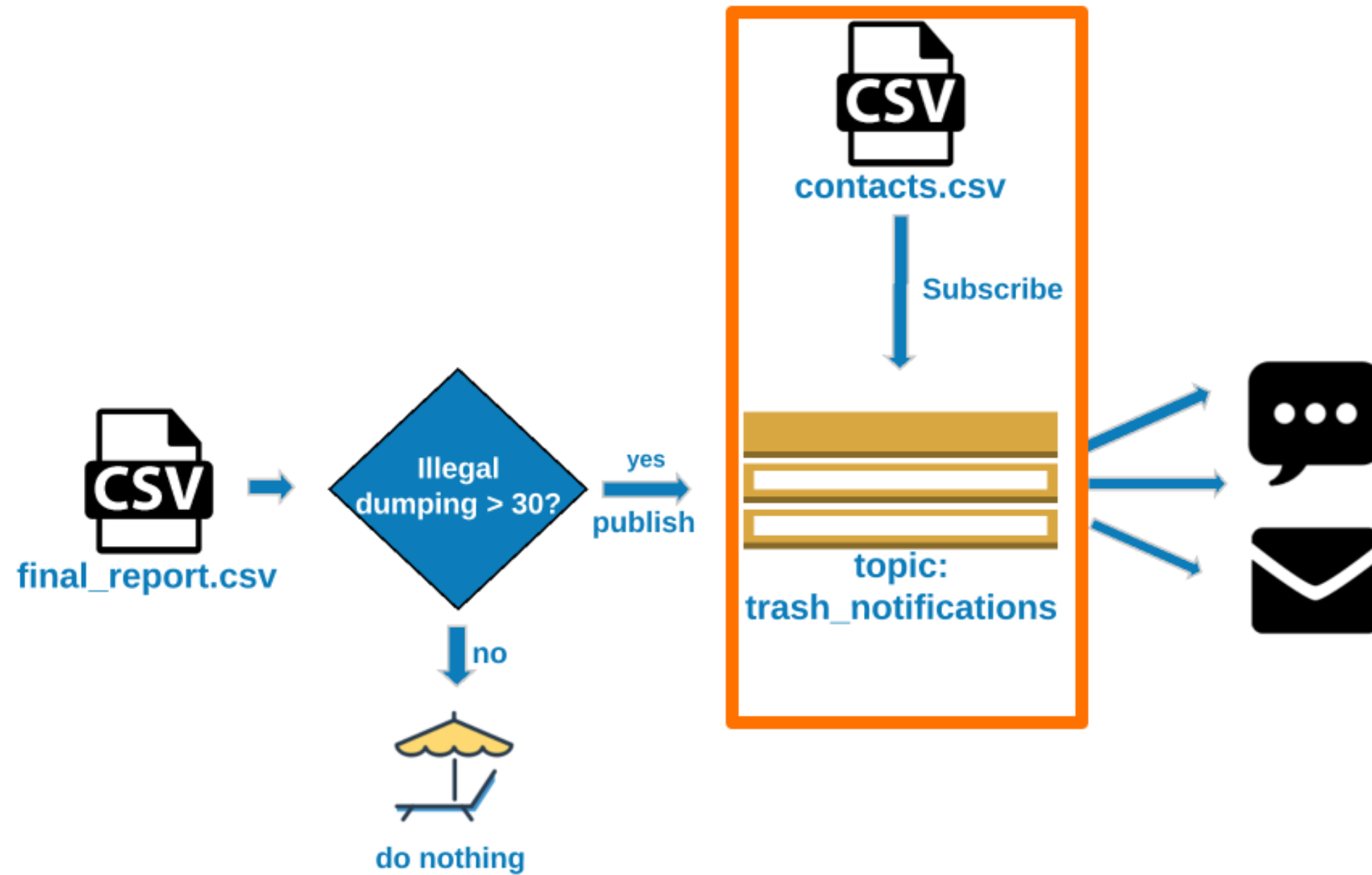


Maksim Pecherskiy
Data Engineer

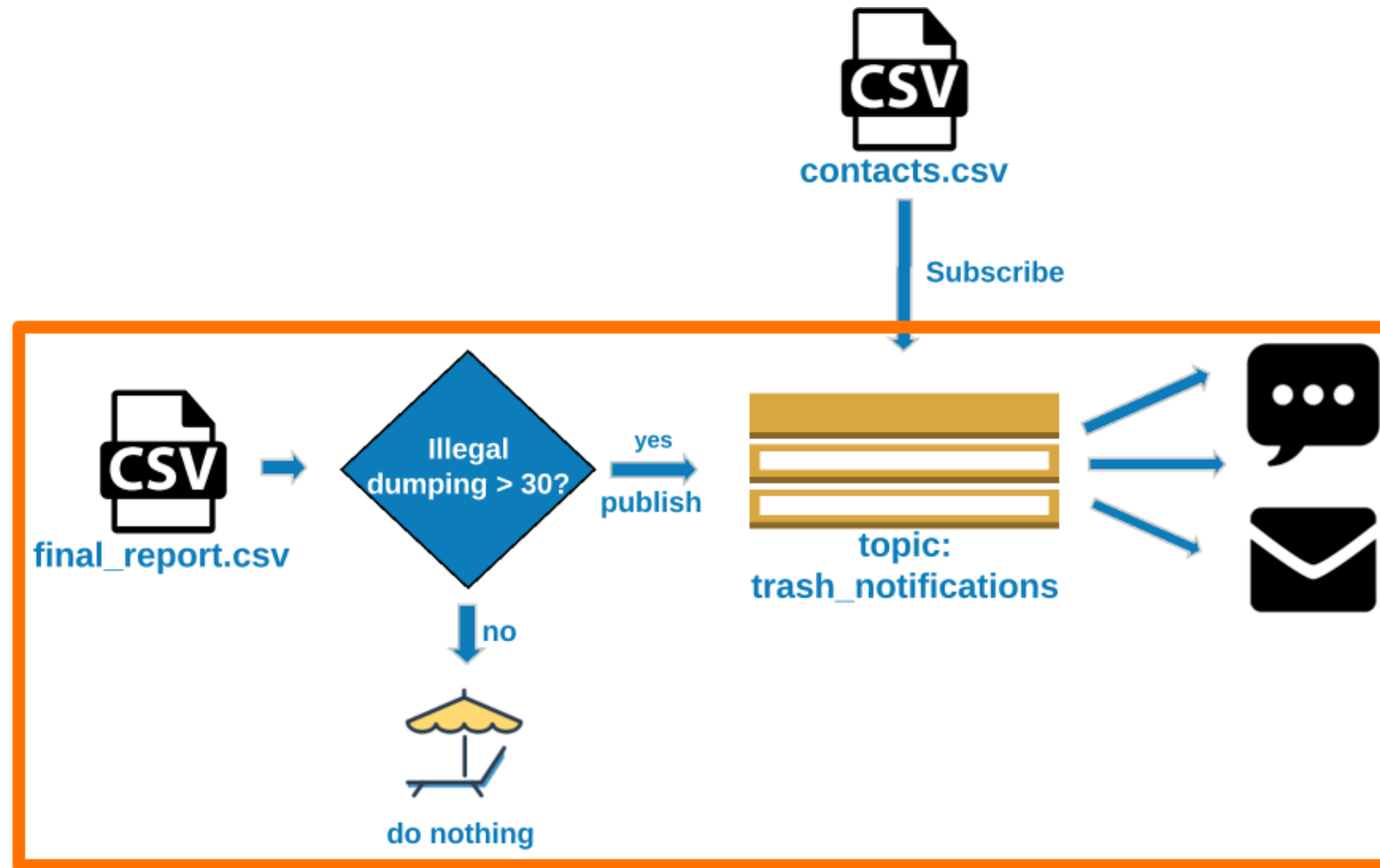
Final product



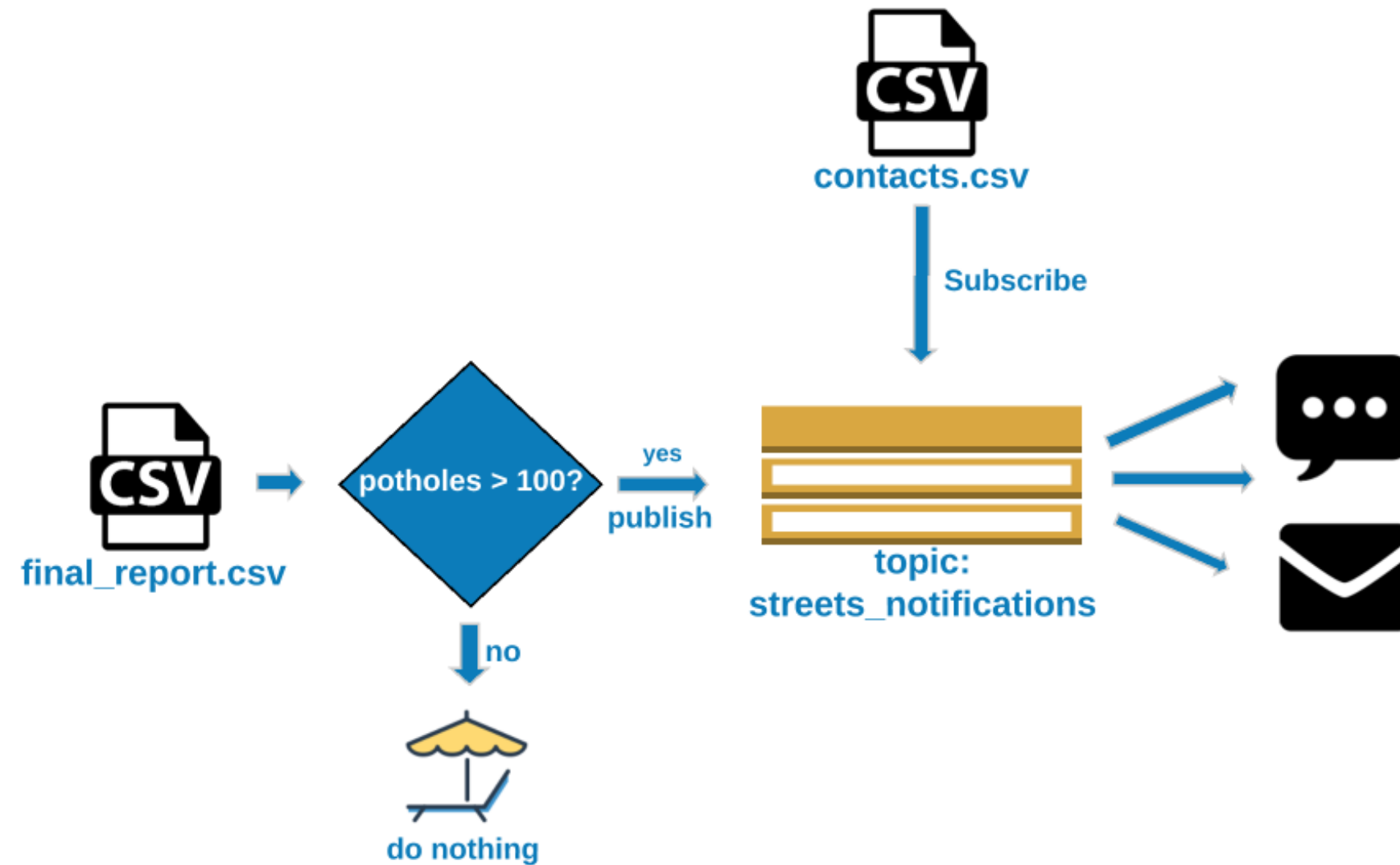
Final product



Final product



Final product



Building the notification system

Topic Set Up

- Create the topic
- Download the contact list csv
- Create topics for each service
- Subscribe the contacts to their respective topics

Building the notification system

Get the aggregated numbers

- Download the monthly get it done report
- Get the count of Potholes
- Get the count of Illegal dumping notifications

Send Alerts

- If potholes exceeds 100, send alert
- If illegal dumping exceeds 30, send alert

Topic set up

Initialize SNS client

```
sns = boto3.client('sns',  
                    region_name='us-east-1',  
                    aws_access_key_id=AWS_KEY_ID,  
                    aws_secret_access_key=AWS_SECRET)
```

Create topics and store their ARNs

```
trash_arn = sns.create_topic(Name="trash_notifications")['TopicArn']  
streets_arn = sns.create_topic(Name="streets_notifications")['TopicArn']
```

Topic set up

Amazon SNS > Topics

Topics (5) Edit Delete Publish message Create topic

< 1 > ⚙

	Name ▲	ARN ▼
<input type="radio"/>	box_alerts	arn:aws:sns:us-east-1:320333787981:box_alerts
<input type="radio"/>	city_alerts	arn:aws:sns:us-east-1:320333787981:city_alerts
<input type="radio"/>	first_topic	arn:aws:sns:us-east-1:320333787981:first_topic
<input type="radio"/>	streets_notifications	arn:aws:sns:us-east-1:320333787981:streets_notifications
<input type="radio"/>	trash_notifications	arn:aws:sns:us-east-1:320333787981:trash_notifications

Subscribing users to topics

```
contacts = pd.read_csv('http://gid-staging.s3.amazonaws.com/contacts.csv')
```


Subscribing users to topics

contacts.csv

?Name	Email	Phone	Department
John Smith	js@fake.com	+11224567890	trash
Fanny Mae	fannyma3@fake.com	+11234597890	trash
Janessa Goldsmith	whoami@fake.com	+11534567890	streets
Evelyn Monroe	Evely@fake.com	+11234067890	streets
Max Pe	max@maksimize.com	+11234517890	streets

Subscribing users to topics

Create subscribe_user method

```
def subscribe_user(user_row):  
    if user_row['Department'] == 'trash':  
        sns.subscribe(TopicArn = trash_arn, Protocol='sms', Endpoint=str(user_row['Phone']))  
        sns.subscribe(TopicArn = trash_arn, Protocol='email', Endpoint=user_row['Email'])  
    else:  
        sns.subscribe(TopicArn = streets_arn, Protocol='sms', Endpoint=str(user_row['Phone']))  
        sns.subscribe(TopicArn = streets_arn, Protocol='email', Endpoint=user_row['Email'])
```

Apply the subscribe_user method to every row

```
contacts.apply(subscribe_user, axis=1)
```

Subscribing users to topics

Subscriptions (6)					Edit	Delete	Request confirmation	Confirm subscription	Create subscription
<input type="text" value="Search"/>					< 1 > ⚙				
	ID ▼	Endpoint ▼	Status ▼	Protocol ▲					
<input type="radio"/>	Pending confirmation	whoami@fake.com	⌚ Pending confirmation	EMAIL					
<input type="radio"/>	Pending confirmation	Evely@fake.com	⌚ Pending confirmation	EMAIL					
<input type="radio"/>	Pending confirmation	max@maksimize.com	⌚ Pending confirmation	EMAIL					
<input type="radio"/>	4d26a385-d881-49d0-96a6-68b9affab25a	+11234067890	✅ Confirmed	SMS					
<input type="radio"/>	5806fee8-8f1b-49e7-951c-b875940c0b7e	+11534567890	✅ Confirmed	SMS					
<input type="radio"/>	d2094f5a-d681-4b15-a8d0-65c41300147b	+11234517890	✅ Confirmed	SMS					

Get the aggregated numbers

Load January's report into a DataFrame

```
df = pd.read_csv('http://gid-reports.s3.amazonaws.com/2019/feb/final_report.csv')
```

Get the aggregated numbers

service_name	count
Illegal Dumping	2580
Potential Missed Collection	150
Pothole	1170
Traffic Sign - Maintain	210
Traffic Signal Head Turned	60
Traffic Signal Light Out	120

Get the aggregated numbers

Set the index so we can access counts by service name directly

```
df.set_index('service_name', inplace=True)
```

Get the aggregated numbers

```
trash_violations_count = df.at['Illegal Dumping', 'count']  
streets_violations_count = df.at['Pothole', 'count']
```

Send Alerts

```
if trash_violations_count > 100:

    # Construct the message to send
    message = "Trash violations count is now {}".format(trash_violations_count)

    # Send message
    sns.publish(TopicArn = trash_arn,
                Message = message,
                Subject = "Trash Alert")
```

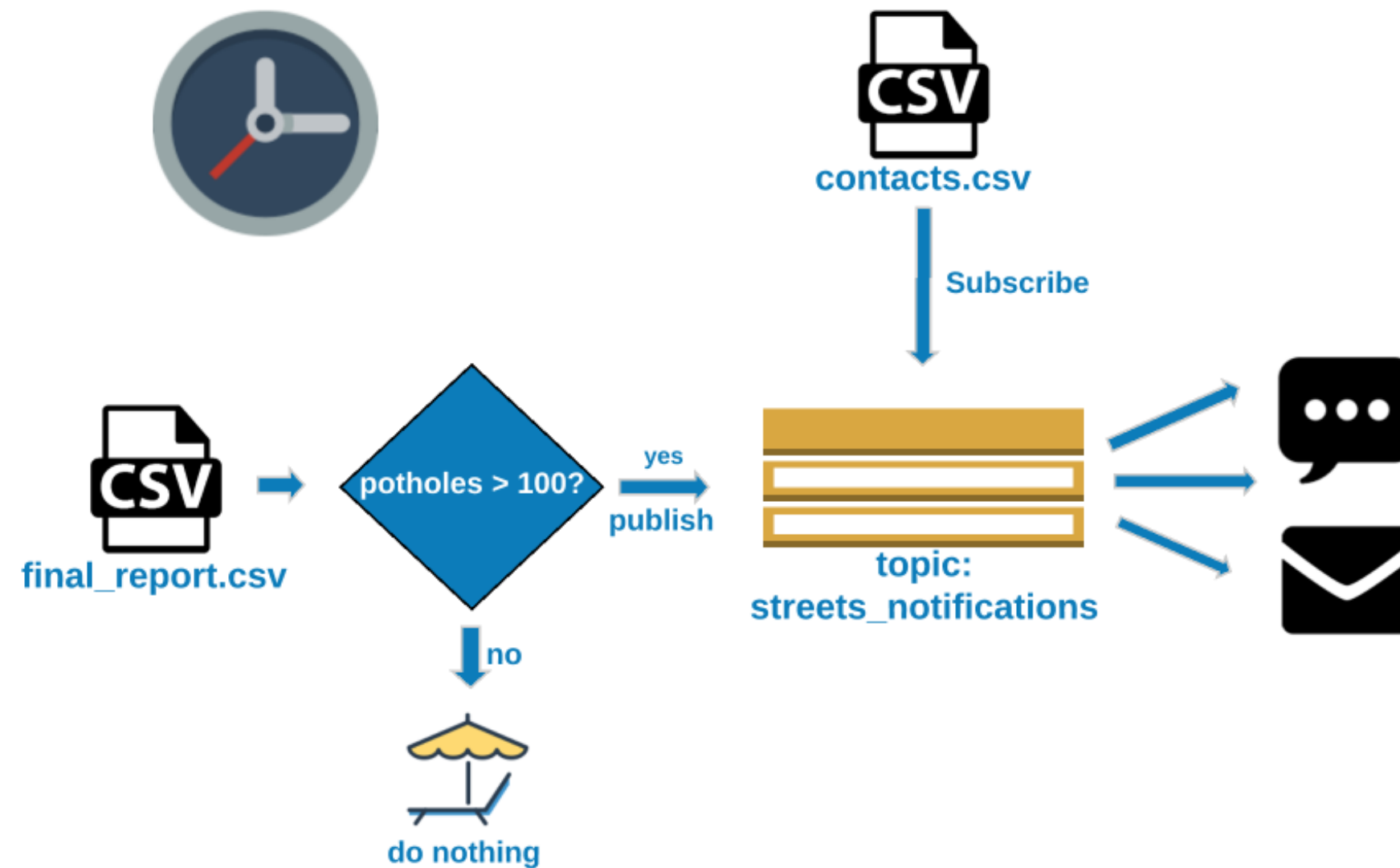
Send alerts

```
if streets_violations_count > 30:

    # Construct the message to send
    message = "Streets violations count is now {}".format(streets_violations_count)

    # Send message
    sns.publish(TopicArn = streets_arn,
                Message = message,
                Subject = "Streets Alert")
```


Final Result



Let's practice!

INTRODUCTION TO AWS BOTO IN PYTHON