National College of Ireland

Configuration Manual

MSc Research Project
Cloud Computing

Umadevi Mala
Student ID: x23187344

School of Computing
National College of Ireland

Supervisor: shaguna Gupta

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Umadevi Mala
Student ID:	x23187344
Programme:	Cloud Computing
Year:	2024
Module:	MSc Research Project
Supervisor:	shaguna Gupta
Submission Due Date:	18/12/2024
Project Title:	WS Glue vs Talend: A Practical Comparison of ETL Tools
Page Count:	19

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Surya Varmanraju Porandla
Date:	16th December 2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Umadevi Mala
x23187344

1 AWS Account Setup

1.1 Introduction

This section guides us through the initial AWS account setup process, ensuring secure and cost-controlled environment configuration.

1.2 Account Creation

1. Visit AWS Management Console
2. Sign up for new AWS account
3. Verify credentials:
    - Email confirmation
    - Phone number verification
    - Payment method setup

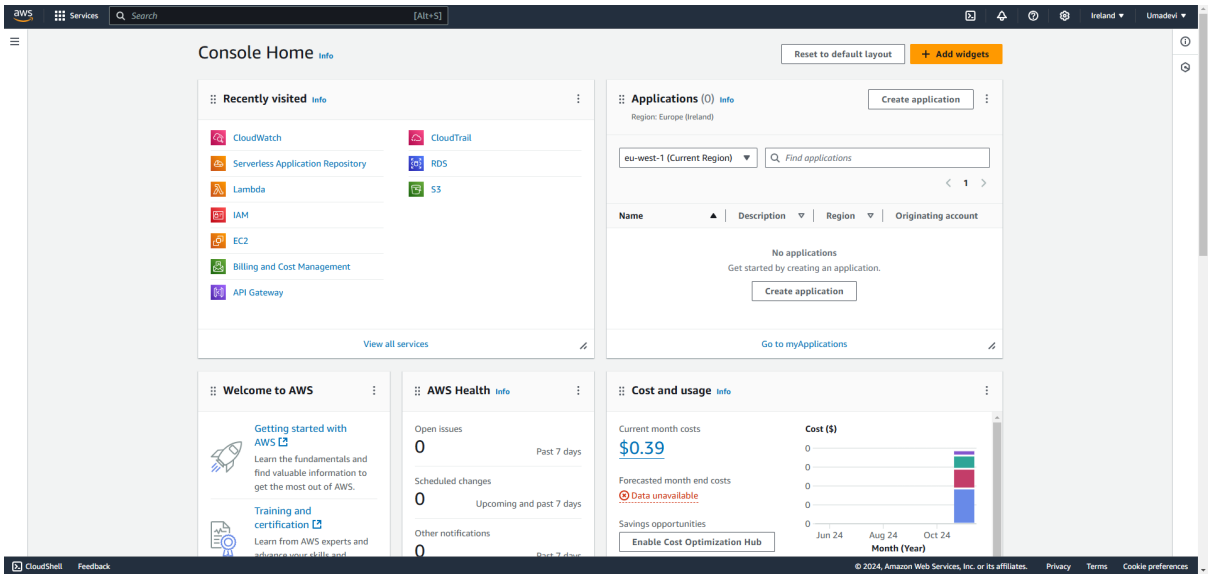

Figure 1: AWS setup

1.3 IAM User Configuration

Create the following user types:

- System Administrators
  - Full system access
  - Administrative privileges
- Logging Users
  - Monitoring access
  - Log management capabilities
- Application Users
  - Specific service access
  - Limited permissions

1.4 Cost Management

Configure billing alerts:

- Set up budget thresholds
- Enable consumption tracking
- Configure cost alerts
- Monitor project expenses

1.5 Security Notes

Important security checklist:

- Enable Multi-Factor Authentication (MFA)
  - Required for all administrative accounts
  - Enhanced account security
  - Regular verification checks
- Password policies
  - Strong password requirements
  - Regular rotation schedule

1.6 Best Practices

Remember to:

- Regularly review user access
- Monitor billing dashboards
- Document account settings
- Keep security credentials secure

2 Set Up RDS Database

2.1 RDS Instance Creation

1. Navigate to AWS RDS console
2. Select PostgreSQL as database engine
3. Configure instance parameters:
    - Instance class
    - Storage allocation
4. Set database credentials:

```
Database Name: online-retail-db
Username: postgres
Password: XXXXXXXX
```

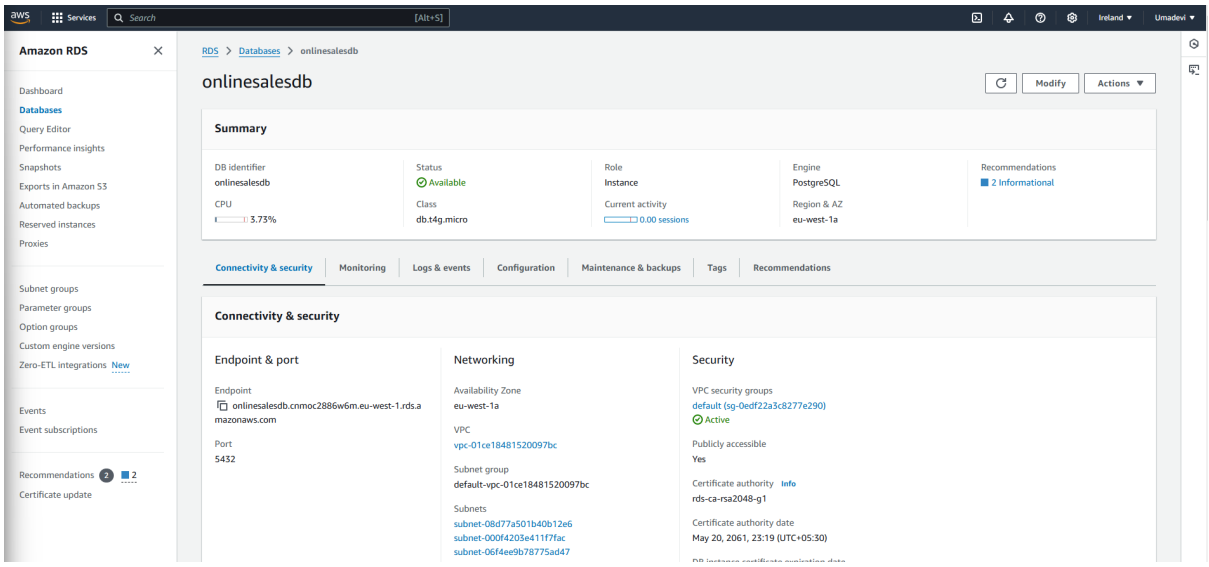

Figure 2: RDS setup

2.2 PgAdmin Setup

1. Download PgAdmin from official website
2. Install on local machine
3. Create new server connection:
    - Enter RDS endpoint
    - Specify port number
    - Input database name
    - Provide username and password

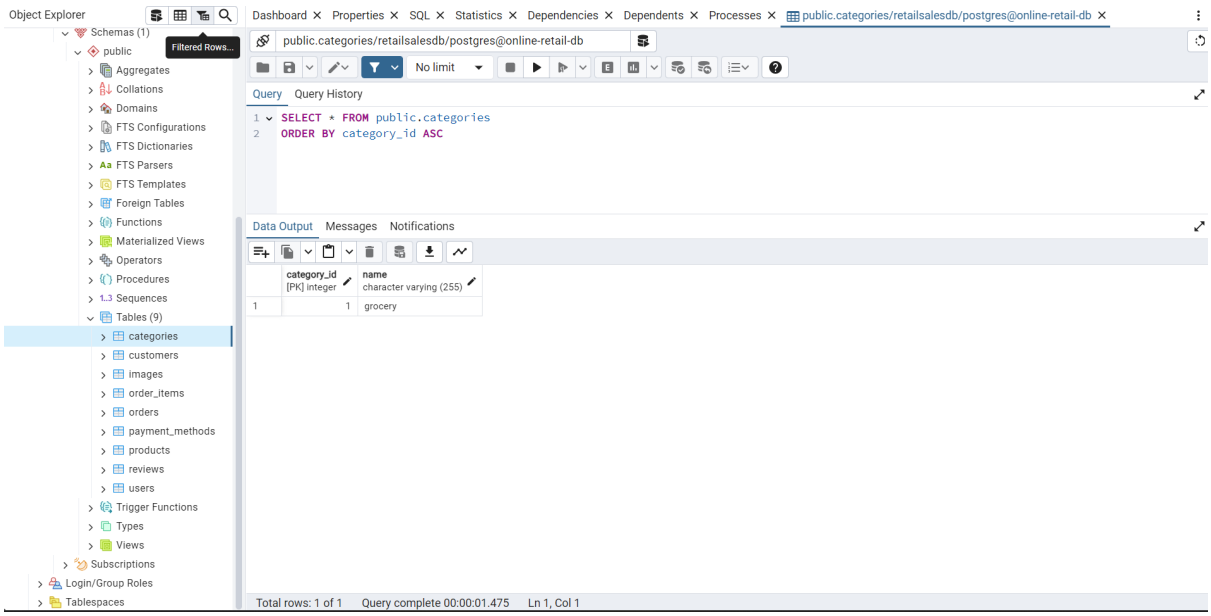

Figure 3: Database Connection and schema Setup

2.3 Database Schema

Create the following tables in public schema:

2.4 Best Practices

- Regularly backup database
- Monitor instance performance
- Follow security best practices
- Document schema changes

Table Name	Description
Categories	Product categories in inventory
Customers	Customer details (name, mobile_number, email)
Images	S3 URIs for product images
Order_items	Order item details
Orders	Order details
Payment_methods	Available payment methods
Products	Product inventory and availability
Reviews	Product reviews from customers
Users	Employee details

Table 1: Database Table Structure

3 Splunk Cloud Setup

3.1 Account Creation

1. Visit Splunk website
2. Sign up for trial account

3.2 AWS Add-On Installation

1. Navigate to Splunkbase
2. Download AWS Add-On
3. Configure for:
    - AWS logs ingestion
    - Metrics collection

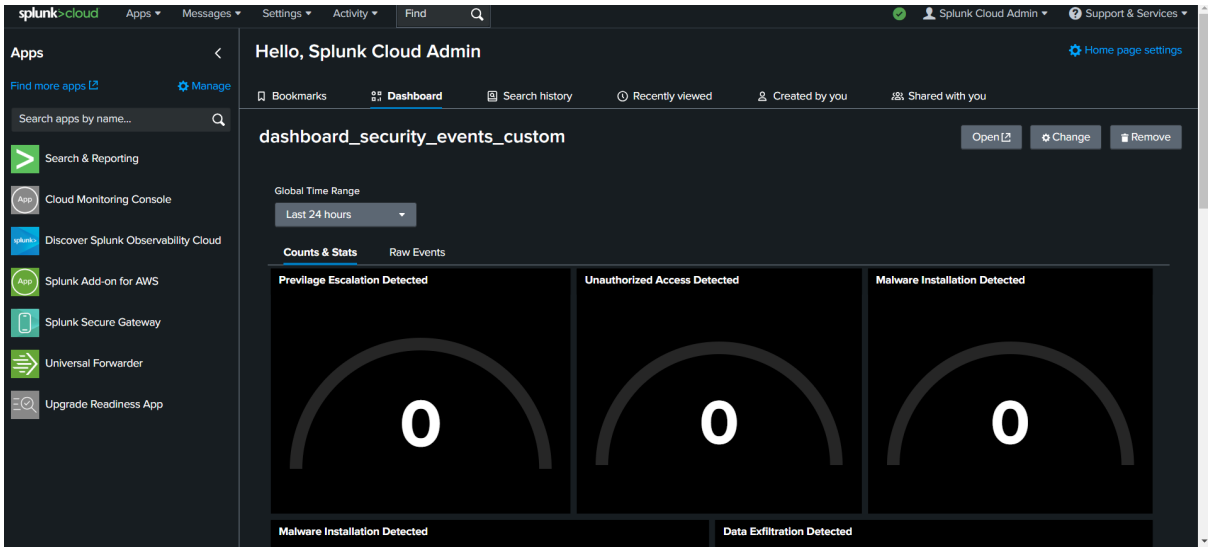

Figure 4: Splunk Setup

3.3 User Role Configuration

1. Generate users with:
    - Data control permissions
    - Analysis capabilities
2. Assign appropriate roles
3. Configure access segments

3.4 Key Notes

Verification checklist:

- Network connectivity
    - AWS to Splunk Cloud connection
    - Access key verification
- Firewall configuration
    - Check integration permissions
    - Verify no blocking rules
- Access key validation
    - Splunk Cloud keys
    - AWS Splunk app keys

4 Setting Up S3 Buckets

4.1 Bucket Creation

1. Access AWS Management Console
2. Navigate to S3 service
3. Create new bucket:
    - Bucket name: `lake-file-uploads`
    - Create folder: `images/` for product images
    - Select appropriate region
    - Configure bucket permissions

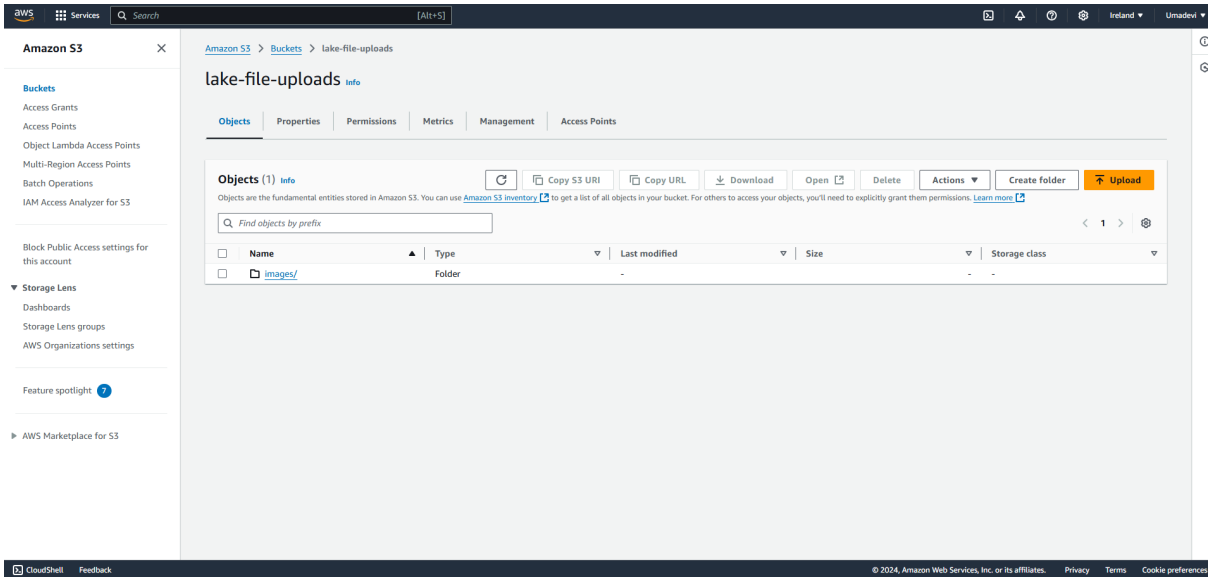

Figure 5: S3 Bucket Setup

4.2 Splunk Integration

4.2.1 Initial Setup

- Create Splunk trial account
- Install AWS Add-On from Splunkbase
- Configure for AWS logs and metrics ingestion

4.2.2 User Management

- Create user roles with specific permissions
- Configure data management access
- Assign analysis permissions

4.3 Configuration Verification

Key verification points:

- Network connectivity between AWS and Splunk Cloud
- Access key configuration
- Firewall rule verification

4.4 Important Notes

- Regularly review bucket permissions
- Monitor storage usage
- Verify Splunk data ingestion

- Maintain access key security

5 Setting Up EC2 Instance For Deploying Our Web App

5.1 Instance Creation

Follow these steps to launch your EC2 instance:

1. Access EC2 Dashboard:
    - Navigate to EC2 service
    - Click "Launch Instance"
2. Instance Configuration:
    - Select Ubuntu AMI
    - Choose instance type: t2.micro (free tier eligible)
    - Generate or select existing key pair
3. Security Group Setup:
    - Configure inbound rules:

```
Port 22   (SSH)
Port 80   (HTTP)
Port 443  (HTTPS)
```

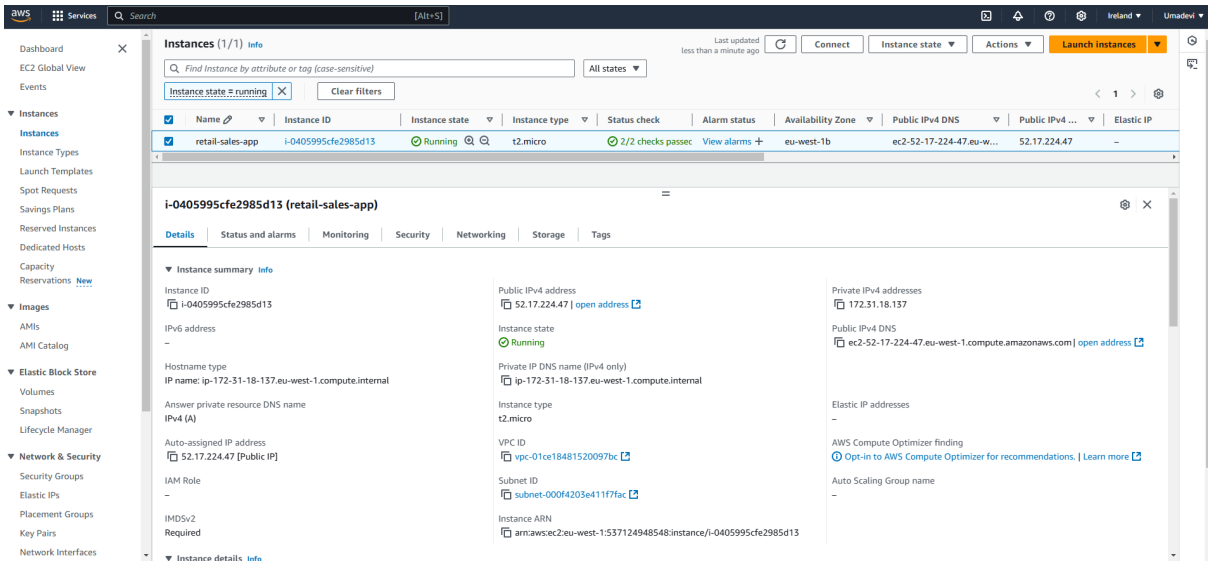

Figure 6: EC2 Setup

5.2 Environment Setup

Install required software packages:

```
# Update package list
sudo apt update

# Install Python and pip
sudo apt install python3 python3-pip

# Install required Python packages
pip install streamlit boto3 psycopg2
```

5.3 Application Deployment

Deploy your application:

1. Copy application files:

   ```
   scp -i "your-key.pem" app/* ubuntu@your-ec2-dns:~/app/
   ```

2. Launch the application:

   ```
   cd ~/app
   streamlit run app.py
   ```

5.4 Verification Steps

Ensure proper setup:

- Verify security group rules
- Test SSH connection
- Confirm application accessibility
- Check package installations

6 Deploying the Streamlit Web Application

6.1 Initial Application Configuration

First, we need to set up our Python environment and dependencies:

- Install required libraries:

  ```
  pip install streamlit boto3 psycopg2-binary
  ```

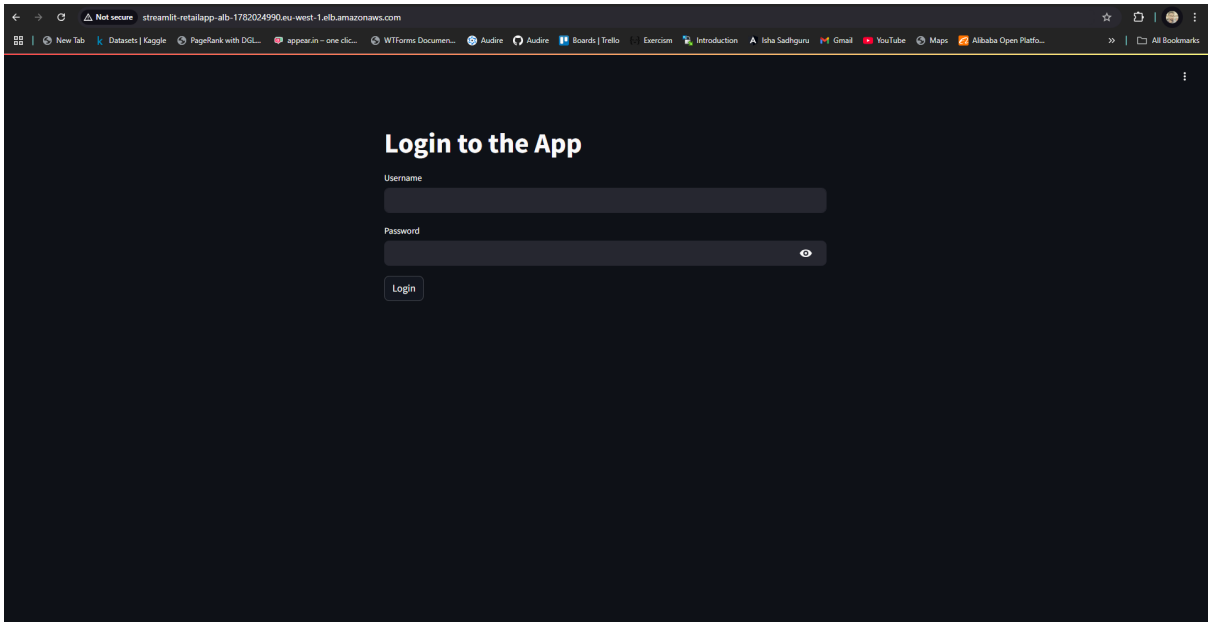

Figure 7: Streamlit App

- Configure AWS credentials:

```
aws configure  # You'll need your AWS access keys
```

- Update your app settings:

```
# config.py
RDS_ENDPOINT = "your-rds-endpoint.region.rds.amazonaws.com"
DB_USERNAME = "your_username"
DB_PASSWORD = "your_password"
S3_BUCKET = "your-bucket-name"
AWS_REGION = "your-region"
```

6.2 EC2 Deployment

Now, let's get our app running on EC2:

1. SSH into your EC2 instance:

```
ssh -i "your-key.pem" ubuntu@your-ec2-dns
```

2. Create a systemd service file:

```
sudo nano /etc/systemd/system/streamlit.service
```

3. Add this configuration:

```
[Unit]
Description=Streamlit App
After=network.target

[Service]
User=ubuntu
WorkingDirectory=/home/ubuntu/app
ExecStart=/usr/bin/python3 -m streamlit run app.py
Restart=always

[Install]
WantedBy=multi-user.target
```

4. Start the service:

```
sudo systemctl enable streamlit
sudo systemctl start streamlit
```

6.3 Load Balancer Setup

Time to set up our Application Load Balancer:

1. Navigate to EC2 ¿ Load Balancers in AWS Console
   `https://console.aws.amazon.com/ec2/v2/home#LoadBalancers`
2. Click "Create Load Balancer"
3. Configure ALB settings:
   - Scheme: internet-facing
   - IP address type: ipv4
   - Listeners: HTTPS (port 443)
   - Security Groups: Allow HTTPS inbound

6.4 Helpful Resources

Check these out if you get stuck:

- Streamlit Documentation:
  `https://docs.streamlit.io/`
- AWS EC2 Documentation:
  `https://docs.aws.amazon.com/ec2/`
- Boto3 Documentation:
  `https://boto3.amazonaws.com/v1/documentation/api/latest/index.html`

6.5 Common Issues & Solutions

Here are some things to watch out for:

- If Streamlit isn't starting, check logs:

```
sudo journalctl -u streamlit.service
```

- Make sure security groups allow traffic on port 443
- Verify RDS endpoint is accessible from EC2

Remember: Keep your security groups as restrictive as possible while still allowing necessary traffic!

7 Configuring AWS Monitoring Tools

7.1 CloudTrail Setup

Follow these steps to enable AWS CloudTrail:

1. Navigate to CloudTrail console
2. Click "Create Trail"
3. Configure essential settings:
    - Enable multi-region logging
    - Configure S3 bucket for log storage

Pro tip: Choose a descriptive name for your trail to easily identify its purpose later.

7.2 CloudWatch Configuration

Set up CloudWatch to monitor your AWS resources:

1. Access CloudWatch console
2. Navigate to Logs section
3. Create log groups for:
    - EC2 instances
    - RDS databases
    - S3 buckets
4. Configure log streams for:
    - Application logs
    - AWS service events

Remember: Organizing your log groups logically will make troubleshooting much easier!

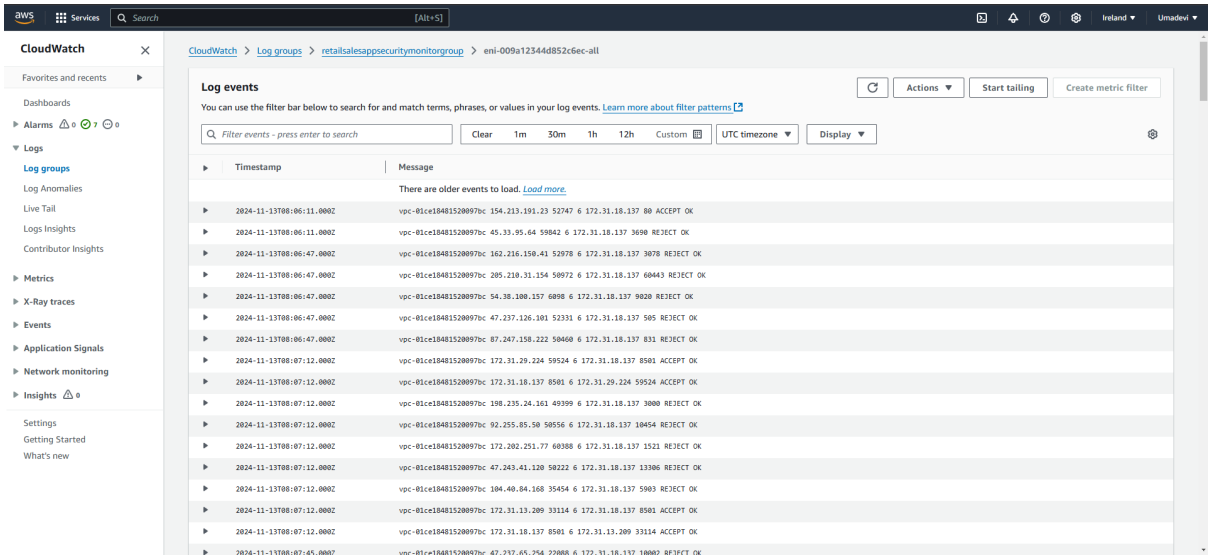

Figure 8: Configuring AWS Monitoring Tool

7.3 GuardDuty Implementation

Enable AWS GuardDuty for enhanced security monitoring:

1. Access GuardDuty console
2. Enable service for your account
3. Configure detection settings:
    - Unauthorized access detection
    - Malware identification
    - Misconfiguration alerts
4. Set up findings routing:
    - Direct to CloudWatch
    - Enable Splunk integration

Important: Regular review of GuardDuty findings helps maintain strong security posture.

7.4 Best Practices

Consider these tips for optimal monitoring:

- Regularly review and update monitoring settings
- Set up automated notifications for critical events
- Maintain proper log retention policies
- Document any custom configurations

8 Setting Up Splunk for AWS Integration

8.1 Initial Setup

1. Install Splunk AWS Add-On:
    - Login to Splunk instance
    - Navigate to Splunkbase
    - Install AWS Add-On

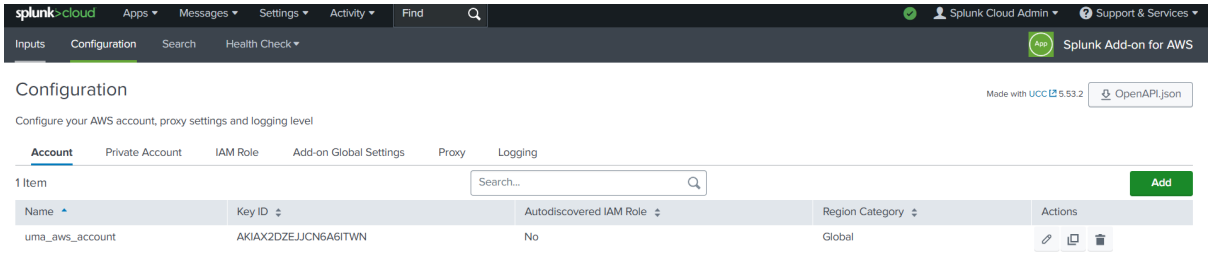

Figure 9: Splunk Setup

8.2 AWS Integration Configuration

1. Access Splunk console:
    - Navigate to Settings ¿ Data Inputs ¿ Add Data
    - Select AWS Services
2. Provide AWS credentials:
    - Access Key
    - Secret Key
    - IAM user permissions for:
        - CloudTrail
        - CloudWatch
        - S3
    - Specify region
    - Select services to monitor

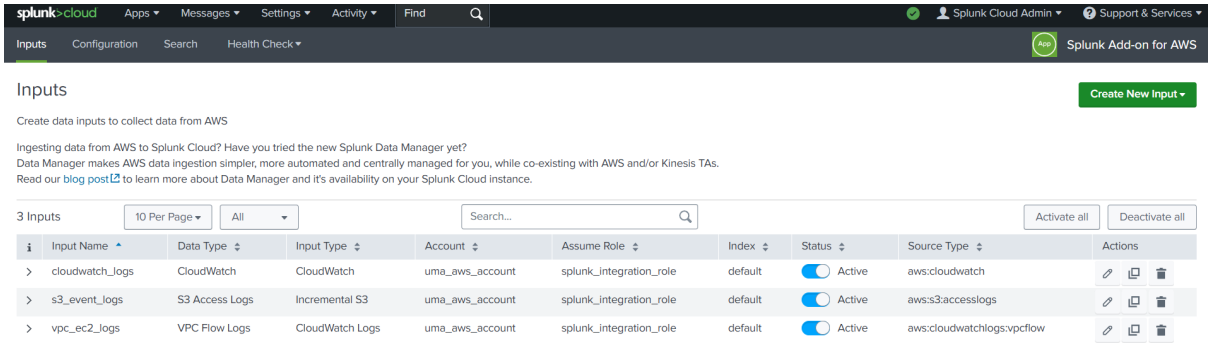

Figure 10: Splunk AWS Configuration

8.3 Data Input Configuration

Configure inputs for:

- CloudWatch Logs
  - Application monitoring
  - Resource monitoring
- CloudTrail Logs
  - API activity tracking
- VPC Flow Logs
  - Network traffic analysis

8.4 Validation

Test data ingestion using query:

```
index=<your_index_name> sourcetype=aws:cloudtrail
```

9 Creating Dashboards and Alerts in Splunk

9.1 Dashboard Creation

1. Navigate to Dashboards
2. Click Create New Dashboard
3. Add panels for:

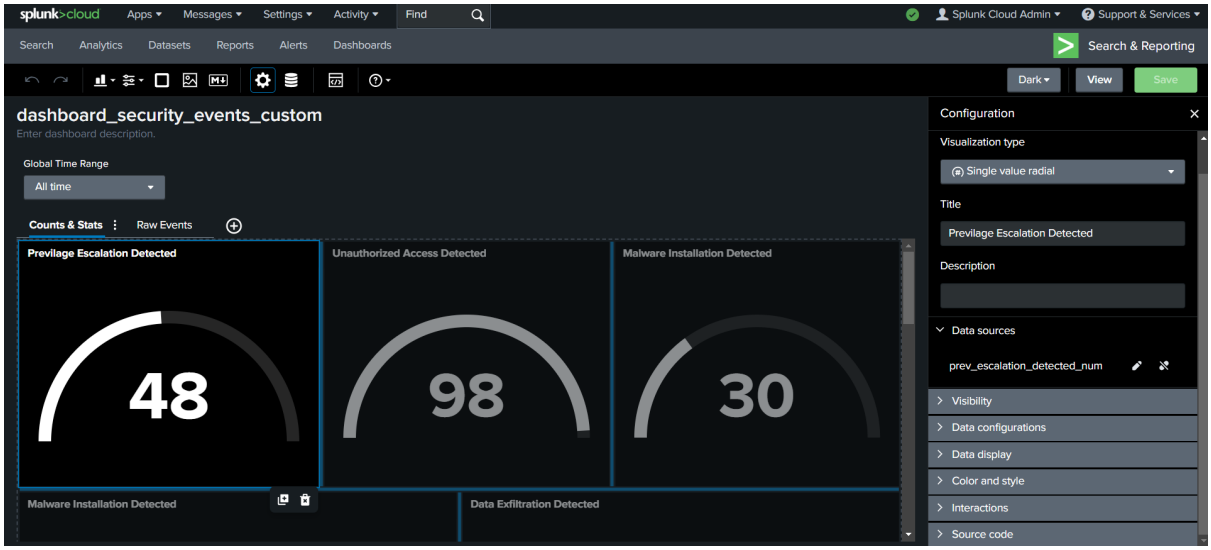

Figure 11: Alert Dashboard Splunk

9.1.1 Panel Configurations

- **Unauthorized Access Trends:**

```
index=<your_index_name> "Event_Detected"="Yes"
| timechart count by Event_Type
```

- **Data Exfiltration Traffic:**

```
index=<your_index_name> sourcetype=vpc:flow
| stats sum(bytes_out) as Total_Bytes by Source_IP
```

- **Privilege Escalation Attack:**

```
((index="aws_security_events")
(sourcetype="previlage_escalation_logs_updated"))
"Event_Detected"="Yes"
| stats count as Unauthorized_Access_Count
```

- **Malware Installation Detected:**

```
((index="aws_security_events")
(sourcetype="malware_installation_logs_updated"))
"Event_Detected"="Yes"
| stats count as Unauthorized_Access_Count
```

- **Phishing Attack Detected:**

```
((index="aws_security_events")
(sourcetype="phishing_attack_updated"))
"Event_Detected"="Yes"
| stats count as Unauthorized_Access_Count
```

9.2 Alert Configuration

1. Navigate to Settings ¿ Alerts ¿ Create New Alert
2. Configure alert conditions:

```
index=<your_index_name> "Event_Type"="Failed Login"
| stats count by User
| where count > 10
```

3. Set notification methods:
   - Email
   - Webhook

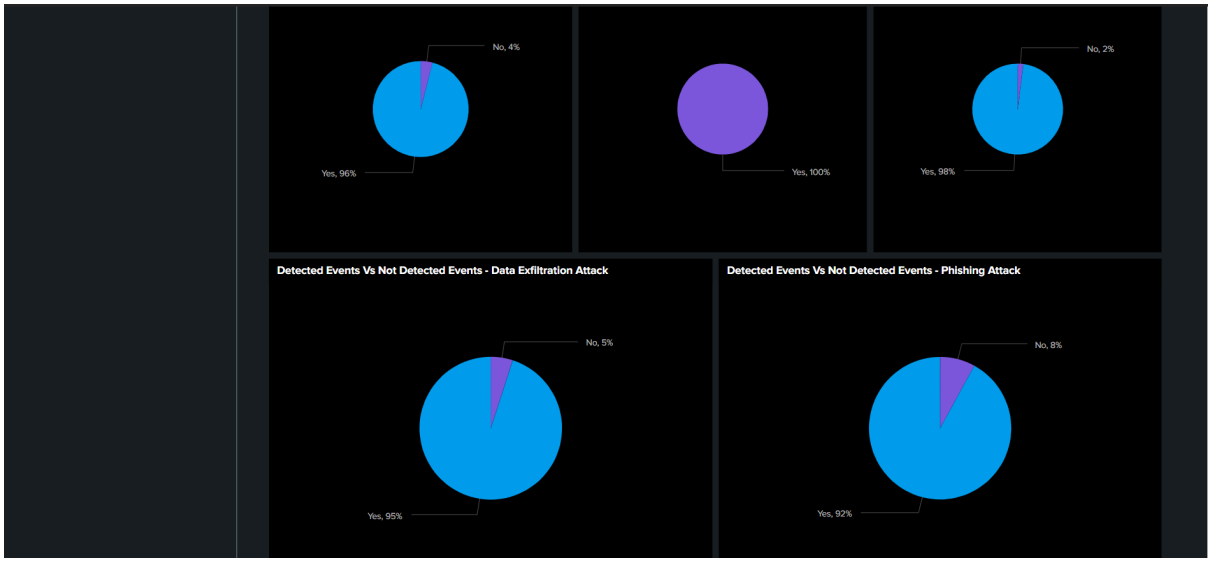

Figure 12: Alert setup Splunk

9.3 Setup Verification

- Simulate unauthorized login attempts
- Test S3 data exfiltration alerts
- Test IAM role modifications

9.4 Dashboard Validation

- Verify real-time event reflection
- Confirm AWS alarm generation

10 AWS CloudWatch Dashboards Configuration

10.1 Pre-configured Dashboards

AWS CloudWatch provides pre-built dashboards for various services:

- EC2
- S3
- RDS
- Other AWS services

10.2 Available Metrics

Status information includes:

- CPU utilization
- Memory usage
- Network performance
- Storage utilization

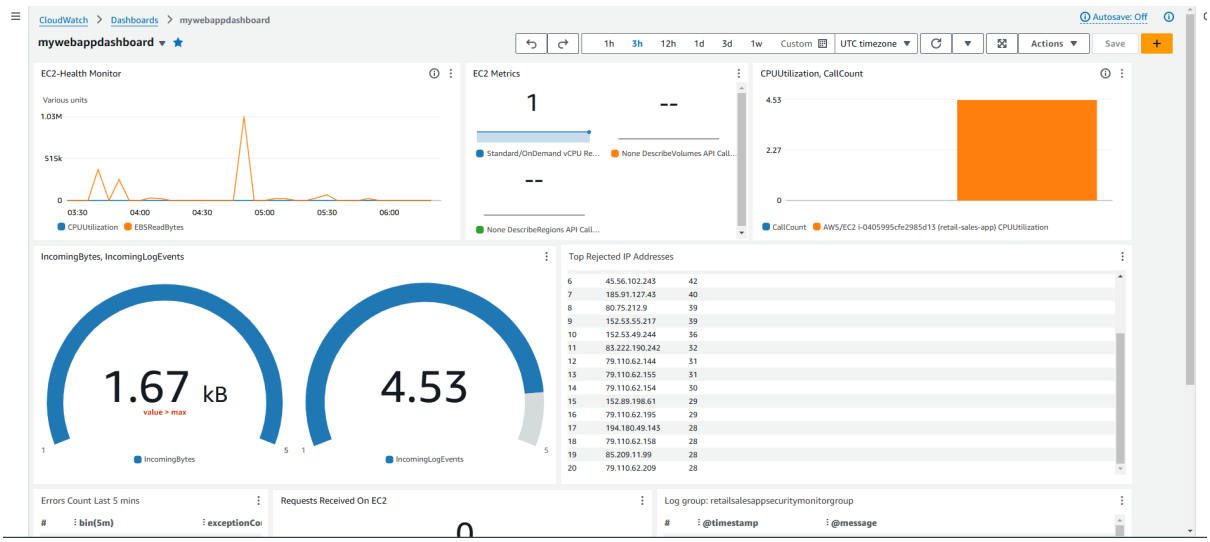

Figure 13: AWS Dasboards

10.3 Dashboard Benefits

The pre-built dashboards offer:

- Instant view of key parameters
- Reduced configuration time
- System health monitoring

10.4 Resource Monitoring

Amazon collects metrics for resources such as EC2 instances:

- Compare CPU utilization
- Monitor disk I/O
- Track network traffic
- Assess instance load
- Identify network issues

11 Conclusion

This above manual provides a complete guide to configuring an integrated AWS and Splunk security monitoring for effecient security management. By following the above steps, we can ensure robust detection and response to security threats in our cloud environment.