```
# Cloud & SaaS Integrations

> **Series:** ONBRD | **Notebook:** 4 of 10 | **Created:** January 2026

## Extending Visibility Beyond OneAgent

While OneAgent provides deep application and infrastructure monitoring, many organizations need visibility into cloud services and SaaS platforms that can't run an agent. This notebook covers how to integrate AWS, Azure, GCP, and third-party SaaS tools into Dynatrace.

---

## Table of Contents

1. Integration Overview
2. AWS Integration
3. Azure Integration
4. GCP Integration
5. Extensions 2.0 Framework
6. Dynatrace Hub
7. Common SaaS Integrations
8. Verifying Integrations
9. Next Steps

---

## Prerequisites

- Dynatrace environment with admin access
- **ActiveGate deployed** (required for most integrations)
- Cloud provider admin access (for AWS/Azure/GCP)
- API credentials for SaaS platforms

## 1. Integration Overview

Dynatrace offers multiple integration methods depending on the data source:

| Method | Use Case | Requires ActiveGate? |
|--------|----------|---------------------|
| **Cloud Integrations** | AWS, Azure, GCP native services | Yes (for polling) |
| **Extensions 2.0** | Custom data sources, SaaS APIs | Yes |
| **OpenTelemetry** | OTel-instrumented apps | No (direct ingest) |
| **Log Ingest** | External log sources | Optional |
| **Metrics Ingest** | Custom metrics via API | No |
```

~~~
### Why Integrate Before OneAgent?

| Benefit | Description |
|---------|-------------|
| **Infrastructure context** | See cloud resources before deploying agents |
| **Dependency mapping** | Understand managed services (RDS, Lambda, etc.) |
| **Cost visibility** | Cloud cost data available immediately |
| **Complete topology** | Smartscape includes cloud services from day one |

## 2. AWS Integration

The AWS integration pulls metrics from CloudWatch and discovers AWS
resources.

### Supported Services

| Category | Services |
|----------|----------|
| **Compute** | EC2, Lambda, ECS, EKS, Fargate |
| **Database** | RDS, DynamoDB, ElastiCache, DocumentDB |
| **Storage** | S3, EBS, EFS |
| **Networking** | ELB, ALB, NLB, API Gateway, CloudFront |
| **Messaging** | SQS, SNS, Kinesis, MSK |
| **Other** | Step Functions, Secrets Manager, and more |

### Setup Methods

| Method | Best For | Complexity |
|--------|----------|------------|
| **IAM Role (recommended)** | Production, multi-account | Medium |
| **Access Key** | Quick testing | Low |
| **CloudFormation** | Automated setup | Low |

### IAM Role Setup

1. Create an IAM role with CloudWatch read permissions
2. Add trust relationship for Dynatrace's AWS account
3. Configure in Dynatrace: Settings → Cloud and virtualization → AWS

**Required IAM Policy:**

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
~~~

~~~
        "cloudwatch:GetMetricData",
        "cloudwatch:GetMetricStatistics",
        "cloudwatch:ListMetrics",
        "tag:GetResources",
        "tag:GetTagKeys",
        "tag:GetTagValues",
        "ec2:DescribeInstances",
        "ec2:DescribeVolumes",
        "rds:DescribeDBInstances",
        "lambda:ListFunctions",
        "lambda:GetFunction"
      ],
      "Resource": "*"
    }
  ]
}
```

### Configuration Location

**Path:** Settings → Cloud and virtualization → AWS

| Setting | Recommendation |
|---------|----------------|
| **Polling interval** | 5 minutes (default) |
| **Services to monitor** | Start with core services, expand as needed |
| **Regions** | Only regions where you have resources |
| **Resource tags** | Use tags to filter monitored resources |

## 3. Azure Integration

The Azure integration uses Azure Monitor to collect metrics and discover
resources.

### Supported Services

| Category | Services |
|----------|----------|
| **Compute** | Virtual Machines, App Service, Functions, AKS |
| **Database** | SQL Database, Cosmos DB, Redis Cache |
| **Storage** | Blob, Files, Queues, Tables |
| **Networking** | Load Balancer, Application Gateway, VNet |
| **Messaging** | Service Bus, Event Hubs, Event Grid |

### Setup via App Registration

1. Create an App Registration in Azure AD
2. Grant **Reader** role on subscriptions to monitor
~~~

```
3. Create a client secret
4. Configure in Dynatrace with:
   - Tenant ID
   - Client ID
   - Client Secret
   - Subscription IDs

**Configuration Location:** Settings → Cloud and virtualization → Azure

## 4. GCP Integration

The GCP integration uses Cloud Monitoring (formerly Stackdriver) APIs.

### Supported Services

| Category | Services |
|----------|----------|
| **Compute** | Compute Engine, GKE, Cloud Run, Cloud Functions |
| **Database** | Cloud SQL, Cloud Spanner, Firestore, Bigtable |
| **Storage** | Cloud Storage |
| **Networking** | Load Balancing, Cloud CDN |
| **Messaging** | Pub/Sub |

### Setup via Service Account

1. Create a Service Account in GCP
2. Grant **Monitoring Viewer** role
3. Generate and download JSON key
4. Configure in Dynatrace

**Configuration Location:** Settings → Cloud and virtualization → Google Cloud Platform

## 5. Extensions 2.0 Framework

Extensions 2.0 is Dynatrace's framework for integrating any data source – databases, SaaS platforms, network devices, and more.

### How Extensions Work

| Component | Role |
|-----------|------|
| **Extension Package** | Python code + metadata defining what to collect |
| **ActiveGate** | Executes extension code, polls data sources |
| **Monitoring Configuration** | Instance-specific settings (endpoints, credentials) |
| **Dynatrace** | Receives metrics, logs, events from extension |
```

~~~
### Extension Types

| Type | Examples |
|------|----------|
| **Database** | Oracle, SQL Server, PostgreSQL, MongoDB |
| **Infrastructure** | VMware, SNMP devices, F5, NetApp |
| **SaaS** | Salesforce, ServiceNow, Jira, Confluent |
| **Custom** | Any REST API, custom protocols |

### Installing an Extension

1. **Find the extension** in Dynatrace Hub
2. **Install** to your environment
3. **Configure** with endpoint and credentials
4. **Assign** to an ActiveGate group
5. **Verify** data is flowing

### Extension Configuration Example

```yaml
# Example: Generic REST API extension configuration
enabled: true
description: "My SaaS API"
endpoints:
  - url: "https://api.example.com/metrics"
    authentication:
      type: "bearer"
      token: "${SAAS_API_TOKEN}"
    polling_interval: 60
activeGate:
  group: "saas-integrations"
```

### Creating Custom Extensions

For SaaS platforms without a pre-built extension, you can create custom
extensions:

1. Use the Extensions 2.0 SDK
2. Define metrics schema in YAML
3. Write Python code to fetch data
4. Package and upload to Dynatrace

**SDK Documentation:** [Extensions 2.0 Development]
(https://docs.dynatrace.com/docs/extend-dynatrace/extensions20)

## 6. Dynatrace Hub
~~~

```
The Dynatrace Hub is your marketplace for extensions, integrations, and apps.

### Accessing the Hub

**Path:** Apps → Dynatrace Hub

Or use quick search: **Cmd+K** → "Hub"

### Hub Categories

| Category | Examples |
|----------|----------|
| **Cloud** | AWS, Azure, GCP, Kubernetes |
| **Database** | Oracle, SQL Server, PostgreSQL |
| **Infrastructure** | VMware, SNMP, NetApp |
| **Messaging** | Kafka, RabbitMQ, IBM MQ |
| **Observability** | OpenTelemetry, Prometheus |
| **Security** | Snyk, SonarQube |
| **ITSM** | ServiceNow, Jira, PagerDuty |

### Installing from Hub

1. Search for the integration you need
2. Click **Install** or **Add to environment**
3. Follow the configuration wizard
4. Verify data appears in Dynatrace

## 7. Common SaaS Integrations

### Messaging Platforms

| Platform | Integration Method | Key Metrics |
|----------|-------------------|-------------|
| **Confluent Cloud** | Extensions 2.0 | Consumer lag, throughput, partitions
|
| **Amazon MSK** | AWS Integration | Broker metrics, topic metrics |
| **RabbitMQ** | Extensions 2.0 | Queue depth, message rates |

### CRM & Business Apps

| Platform | Integration Method | Key Metrics |
|----------|-------------------|-------------|
| **Salesforce** | Extensions 2.0 / API | API calls, response times, limits |
| **ServiceNow** | Extensions 2.0 | Incident metrics, workflow times |

### DevOps Tools

| Platform | Integration Method | Key Metrics |
```

~~~
|----------|-------------------|-------------|
| **GitHub** | Webhooks | Deployment events, commit info |
| **GitLab** | Webhooks | Pipeline events, deployments |
| **ArgoCD** | Extensions 2.0 | Sync status, app health |

### Database-as-a-Service

| Platform | Integration Method | Key Metrics |
|----------|-------------------|-------------|
| **MongoDB Atlas** | Extensions 2.0 | Connections, ops/sec, replication lag
|
| **AWS DocumentDB** | AWS Integration | CloudWatch metrics |
| **Snowflake** | Extensions 2.0 | Query performance, warehouse usage |

## 8. Verifying Integrations

After configuring integrations, verify data is flowing into Dynatrace.

```dql
// Check for AWS entities
fetch dt.entity.aws_lambda_function
| fields entity.name, id
| limit 20
```

```dql
// Check for Azure entities
fetch dt.entity.azure_vm
| fields entity.name, id
| limit 20
```

```dql
// Check for cloud-sourced metrics
fetch dt.metrics
| filter matchesPhrase(dt.metrics.key, "cloud")
| fields dt.metrics.key
| limit 20
```

```dql
// Check Extensions 2.0 status
fetch dt.entity.extension
| fields entity.name, id
| limit 20
```

### Troubleshooting Integration Issues
~~~

```
| Issue | Common Cause | Solution |
|-------|--------------|----------|
| **No data appearing** | Credentials invalid | Verify IAM role/service account |
| **Partial data** | Permissions incomplete | Check required permissions list |
| **Delayed data** | Polling interval | CloudWatch has 5-min delay by design |
| **Extension not running** | ActiveGate issue | Check AG logs, verify assignment |
| **Metrics but no entities** | Tag configuration | Ensure resources are tagged correctly |

## 9. Next Steps

With cloud and SaaS integrations configured:

1. **ONBRD-05: Deploying OneAgent** - Add deep application monitoring
2. **ONBRD-06: Organizing Your Environment** - Tag and segment integrated resources
3. Explore the topology in Smartscape to see cloud services
4. Create dashboards combining cloud metrics with application data

### Integration Checklist

- [ ] AWS/Azure/GCP integration configured (if applicable)
- [ ] Required cloud permissions granted
- [ ] ActiveGate assigned for Extensions 2.0
- [ ] Key SaaS platforms identified for integration
- [ ] Extensions installed from Hub
- [ ] Data verified flowing into Dynatrace

---

## Summary

In this notebook, you learned:

- Different integration methods (cloud, Extensions 2.0, OTel, API)
- How to set up AWS, Azure, and GCP integrations
- The Extensions 2.0 framework for SaaS and custom integrations
- How to use the Dynatrace Hub to find and install integrations
- Common SaaS integrations and their methods
- How to verify integrations are working

---
```

```
## References

### Cloud Integrations
- [AWS Monitoring](https://docs.dynatrace.com/docs/platform-modules/infrastructure-monitoring/cloud-platform-monitoring/amazon-web-services-monitoring)
- [Azure Monitoring](https://docs.dynatrace.com/docs/platform-modules/infrastructure-monitoring/cloud-platform-monitoring/azure-platform-monitoring)
- [GCP Monitoring](https://docs.dynatrace.com/docs/platform-modules/infrastructure-monitoring/cloud-platform-monitoring/google-cloud-platform-monitoring)

### Extensions
- [Extensions 2.0 Overview](https://docs.dynatrace.com/docs/extend-dynatrace/extensions20)
- [Extensions 2.0 Development](https://docs.dynatrace.com/docs/extend-dynatrace/extensions20/extensions-concepts)
- [Dynatrace Hub](https://docs.dynatrace.com/docs/manage/hub)

### Data Ingestion
- [Metrics Ingest API](https://docs.dynatrace.com/docs/dynatrace-api/environment-api/metric-v2/post-ingest-metrics)
- [Log Ingest](https://docs.dynatrace.com/docs/observe-and-explore/logs/log-monitoring-v2/log-data-ingestion)
- [OpenTelemetry](https://docs.dynatrace.com/docs/extend-dynatrace/opentelemetry)
```