AppDynamics OnPrem Pre-Installation Planning & Infrastructure Guide

Table of Contents

Controller Sizing Guidelines

Sizing Matrix Based on Agent Count

Agent Count	Controller Specs	Database Specs	Load Balancer	Notes
1-50	4 vCPU, 8GB RAM, 100GB SSD	2 vCPU, 4GB RAM, 200GB SSD	Single ALB/NLB	Development/Small
51-200	8 vCPU, 16GB RAM, 200GB SSD	4 vCPU, 8GB RAM, 500GB SSD	ALB + 2 Controllers	Small Production
201-500	16 vCPU, 32GB RAM, 500GB SSD	8 vCPU, 16GB RAM, 1TB SSD	ALB + 2 Controllers	Medium Production
501-1000	32 vCPU, 64GB RAM, 1TB SSD	16 vCPU, 32GB RAM, 2TB SSD	ALB + 3 Controllers	Large Production
1000+	64 vCPU, 128GB RAM, 2TB SSD	32 vCPU, 64GB RAM, 4TB SSD	ALB + 4+ Controllers	Enterprise

Performance Considerations

CPU Sizing Factors:

- Base: 2 vCPU per 100 agents
- Business Transaction complexity: +25% for complex BTs
- Custom metrics: +1 vCPU per 10,000 custom metrics
- Real-time monitoring: +50% CPU overhead

Memory Sizing Factors:

- Base: 4GB + (Agent Count × 50MB)
- Metric retention period: +2GB per month of retention
- Concurrent user sessions: +100MB per 10 concurrent users
- Analytics: +8GB for analytics module

Storage Sizing Factors:

- OS and Application: 50GB
- Metric data: Agent Count × Retention Days × 2MB
- Log files: 10GB + (5GB × Number of Applications)
- Snapshots: 100MB × Expected daily snapshots

Instance Type Recommendations

AWS Instance Types:

- Small: m5.xlarge (4 vCPU, 16GB RAM)
- Medium: m5.2xlarge (8 vCPU, 32GB RAM)
- Large: m5.4xlarge (16 vCPU, 64GB RAM)
- Enterprise: m5.8xlarge (32 vCPU, 128GB RAM)

Azure Instance Types:

- Small: Standard_D4s_v3 (4 vCPU, 16GB RAM)
- Medium: Standard_D8s_v3 (8 vCPU, 32GB RAM)
- Large: Standard_D16s_v3 (16 vCPU, 64GB RAM)
- Enterprise: Standard_D32s_v3 (32 vCPU, 128GB RAM)

Infrastructure Requirements

Base Infrastructure Components

Required Services:

1. Application Load Balancer (ALB) or Network Load Balancer (NLB)
2. Auto Scaling Groups for Controllers
3. RDS Database (MySQL/Oracle) or Database Server
4. S3 Bucket / Azure Blob Storage for agents
5. CloudFront / Azure CDN for agent distribution
6. Route 53 / Azure DNS for domain management
7. Certificate Manager / Azure Key Vault for SSL certificates
8. VPC / Virtual Network with proper subnets
9. Security Groups / Network Security Groups
10. IAM Roles / Azure AD Service Principals

Optional but Recommended:

- ElastiCache / Azure Redis for session storage
- CloudWatch / Azure Monitor for monitoring
- AWS Config / Azure Policy for compliance
- AWS WAF / Azure Application Gateway for security
- Backup services (AWS Backup / Azure Backup)

Network Requirements

Bandwidth Requirements:

- Controller to Database: 1Gbps minimum
- Load Balancer to Controllers: 1Gbps minimum
- Agent to Controller: 100Mbps per 100 agents
- User access: 10Mbps per concurrent user

Latency Requirements:

- Controller to Database: <5ms
- Load Balancer to Controller: <2ms

- Agent to Controller: <100ms
- User to Load Balancer: <50ms

AWS Network Architecture

VPC Design

```
AppDynamics VPC (10.0.0.0/16)
├── Public Subnets (DMZ)
│   ├── us-east-1a: 10.0.1.0/24 (Load Balancers)
│   ├── us-east-1b: 10.0.2.0/24 (Load Balancers)
│   └── us-east-1c: 10.0.3.0/24 (Load Balancers)
├── Private Subnets (Application Tier)
│   ├── us-east-1a: 10.0.11.0/24 (Controllers)
│   ├── us-east-1b: 10.0.12.0/24 (Controllers)
│   └── us-east-1c: 10.0.13.0/24 (Controllers)
└── Database Subnets (Data Tier)
    ├── us-east-1a: 10.0.21.0/24 (RDS Primary)
    ├── us-east-1b: 10.0.22.0/24 (RDS Standby)
    └── us-east-1c: 10.0.23.0/24 (RDS Read Replica)
```

AWS Resource Planning

VPC Configuration:

```
VPC CIDR: 10.0.0.0/16
Region: us-east-1 (Primary), us-west-2 (DR)
Availability Zones: 3 minimum
DNS Hostnames: Enabled
DNS Resolution: Enabled
```

Subnet Configuration:

```
Public Subnets (3):
- 10.0.1.0/24 (AZ-1a) - ALB/NLB
- 10.0.2.0/24 (AZ-1b) - ALB/NLB
- 10.0.3.0/24 (AZ-1c) - ALB/NLB

Private Subnets (3):
- 10.0.11.0/24 (AZ-1a) - Controllers
- 10.0.12.0/24 (AZ-1b) - Controllers
- 10.0.13.0/24 (AZ-1c) - Controllers

Database Subnets (3):
- 10.0.21.0/24 (AZ-1a) - RDS Primary
- 10.0.22.0/24 (AZ-1b) - RDS Standby
- 10.0.23.0/24 (AZ-1c) - RDS Read Replica
```

Security Groups:

ALB Security Group (sg-appdynamics-alb):

```
Inbound:
- Port 443 (HTTPS): 0.0.0.0/0
- Port 80 (HTTP): 0.0.0.0/0
Outbound:
- Port 8181 (HTTPS): Controller Security Group
- Port 8090 (HTTP): Controller Security Group
```

Controller Security Group (sg-appdynamics-controller):

```
Inbound:
- Port 8181 (HTTPS): ALB Security Group
- Port 8090 (HTTP): ALB Security Group
- Port 9300-9400: Agent Security Group
- Port 22 (SSH): Bastion Security Group
Outbound:
- Port 3306 (MySQL): Database Security Group
- Port 443 (HTTPS): 0.0.0.0/0
- Port 80 (HTTP): 0.0.0.0/0
```

Database Security Group (sg-appdynamics-db):

```
Inbound:
- Port 3306 (MySQL): Controller Security Group
- Port 3306 (MySQL): Bastion Security Group
Outbound: None
```

Agent Security Group (sg-appdynamics-agents):

```
Inbound:
- Application specific ports
Outbound:
- Port 8181: ALB Security Group
- Port 9300-9400: Controller Security Group
```

Route Tables

Public Route Table:

```
Destination: 0.0.0.0/0 → Internet Gateway
Destination: 10.0.0.0/16 → Local
```

Private Route Table:

```
Destination: 0.0.0.0/0 → NAT Gateway
Destination: 10.0.0.0/16 → Local
```

Azure Network Architecture

Virtual Network Design

```
AppDynamics VNet (10.0.0.0/16)
├── Public Subnet (DMZ)
│   └── Gateway-Subnet: 10.0.1.0/24 (Application Gateway)
├── Private Subnets (Application Tier)
│   ├── Controller-Subnet-1: 10.0.11.0/24 (East US)
│   ├── Controller-Subnet-2: 10.0.12.0/24 (East US 2)
│   └── Controller-Subnet-3: 10.0.13.0/24 (East US 3)
└── Database Subnets (Data Tier)
    ├── Database-Subnet-1: 10.0.21.0/24 (Primary)
    └── Database-Subnet-2: 10.0.22.0/24 (Secondary)
```

Azure Resource Planning

Virtual Network Configuration:

```
VNet CIDR: 10.0.0.0/16
Region: East US (Primary), West US 2 (DR)
Resource Group: RG-AppDynamics-Production
DNS Servers: Custom (Domain Controllers)
```

Network Security Groups (NSGs):

Application Gateway NSG:

```
Priority 100: Allow HTTPS (443) from Internet
Priority 110: Allow HTTP (80) from Internet
Priority 120: Allow Health Probes (65503-65534)
Priority 1000: Deny All
```

Controller NSG:

```
Priority 100: Allow 8181 from Application Gateway
Priority 110: Allow 8090 from Application Gateway
Priority 120: Allow 9300-9400 from Agent Subnets
Priority 130: Allow SSH (22) from Bastion
Priority 1000: Deny All
```

Database NSG:

```
Priority 100: Allow 3306 from Controller Subnets
Priority 110: Allow 3306 from Bastion
Priority 1000: Deny All
```

Azure Application Gateway Configuration

Basic Configuration:

```
SKU: WAF_v2 (Web Application Firewall)
Tier: Standard_v2
Capacity: Auto-scaling (2-10 instances)
Zones: 1, 2, 3 (Zone redundant)
HTTP2: Enabled
```

Backend Pools:

- Controllers: 10.0.11.4, 10.0.12.4, 10.0.13.4

Health Probes:

- Path: /controller/rest/serverstatus
- Protocol: HTTPS
- Port: 8181
- Interval: 30 seconds
- Timeout: 30 seconds
- Unhealthy threshold: 3

Linux User Account Requirements

Required System Accounts

AppDynamics Service Account:

```bash
Username: appdynamics
UID: 2001 (consistent across all servers)
GID: 2001
Home Directory: /home/appdynamics
Shell: /bin/bash
Groups: appdynamics, wheel (for sudo)
```

Database Service Account:

```bash
```

```
Username: mysqladmin
UID: 2002
GID: 2002
Home Directory: /home/mysqladmin
Shell: /bin/bash
Groups: mysqladmin
```

Backup Service Account:

```bash
Username: appd-backup
UID: 2003
GID: 2003
Home Directory: /home/appd-backup
Shell: /bin/bash
Groups: appd-backup
```

Monitoring Service Account:

```bash
Username: appd-monitor
UID: 2004
GID: 2004
Home Directory: /home/appd-monitor
Shell: /bin/bash
Groups: appd-monitor
```

Administrative Accounts

AppDynamics Administrator:

```bash
```

```
Username: appd-admin
UID: 2010
GID: 2010
Home Directory: /home/appd-admin
Shell: /bin/bash
Groups: wheel, docker, appd-admin
SSH Key: Required (no password login)
Sudo: ALL=(ALL) NOPASSWD: /opt/appdynamics/*, /bin/systemctl
```

Database Administrator:

```bash
Username: db-admin
UID: 2011
GID: 2011
Home Directory: /home/db-admin
Shell: /bin/bash
Groups: wheel, db-admin
SSH Key: Required
Sudo: ALL=(ALL) NOPASSWD: /usr/bin/mysql*, /bin/systemctl mysql*
```

Service Account Configuration Script

```bash
```

```
#!/bin/bash
# create_service_accounts.sh

# Create AppDynamics service account
sudo groupadd -g 2001 appdynamics
sudo useradd -u 2001 -g 2001 -m -d /home/appdynamics -s /bin/bash appdynamics
sudo usermod -aG wheel appdynamics

# Create database service account
sudo groupadd -g 2002 mysqladmin
sudo useradd -u 2002 -g 2002 -m -d /home/mysqladmin -s /bin/bash mysqladmin

# Create backup service account
sudo groupadd -g 2003 appd-backup
sudo useradd -u 2003 -g 2003 -m -d /home/appd-backup -s /bin/bash appd-backup

# Create monitoring service account
sudo groupadd -g 2004 appd-monitor
sudo useradd -u 2004 -g 2004 -m -d /home/appd-monitor -s /bin/bash appd-monitor

# Create admin accounts
sudo groupadd -g 2010 appd-admin
sudo useradd -u 2010 -g 2010 -m -d /home/appd-admin -s /bin/bash appd-admin
sudo usermod -aG wheel,docker appd-admin

sudo groupadd -g 2011 db-admin
sudo useradd -u 2011 -g 2011 -m -d /home/db-admin -s /bin/bash db-admin
sudo usermod -aG wheel db-admin

# Configure sudo permissions
echo "appdynamics ALL=(ALL) NOPASSWD: /opt/appdynamics/*, /bin/systemctl" | sudo tee /etc/sudoers.d/appdynami
echo "appd-admin ALL=(ALL) NOPASSWD: /opt/appdynamics/*, /bin/systemctl" | sudo tee /etc/sudoers.d/appd-admin
echo "db-admin ALL=(ALL) NOPASSWD: /usr/bin/mysql*, /bin/systemctl mysql*" | sudo tee /etc/sudoers.d/db-admin

# Set file permissions
sudo chmod 440 /etc/sudoers.d/*
```

Security Infrastructure Planning

PKI and Certificate Infrastructure

Required Certificates:

1. **Root CA Certificate**
   - Purpose: Root of trust for all AppDynamics certificates
   - Validity: 10 years
   - Key Size: 4096-bit RSA or P-384 ECC
2. **Intermediate CA Certificate**
   - Purpose: Issues end-entity certificates
   - Validity: 5 years
   - Key Size: 4096-bit RSA or P-384 ECC
3. **Load Balancer Certificate**
   - Subject: CN=appdynamics.company.com
   - SAN: *.appdynamics.company.com, appdynamics-controller.company.com
   - Validity: 2 years
   - Key Size: 2048-bit RSA or P-256 ECC
4. **Controller Certificates (per controller)**
   - Subject: CN=controller-01.appdynamics.company.com
   - Validity: 2 years
   - Key Size: 2048-bit RSA
5. **Database Certificate**
   - Subject: CN=mysql.appdynamics.company.com
   - Validity: 2 years
   - Key Size: 2048-bit RSA

IAM Roles and Policies

AWS IAM Roles:

AppDynamics Controller Role:

```json
```

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::appdynamics-agents/*",
        "arn:aws:s3:::appdynamics-agents"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:GetSecretValue"
      ],
      "Resource": "arn:aws:secretsmanager:*:*:secret:appdynamics/*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:Decrypt"
      ],
      "Resource": "arn:aws:kms:*:*:key/appdynamics-key-id"
    }
  ]
}
```

Load Balancer Role:

```
json
```

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "acm:GetCertificate",
        "acm:ListCertificates"
      ],
      "Resource": "*"
    }
  ]
}
```

Azure RBAC Roles:

AppDynamics Controller Identity:

- Contributor role on Resource Group
- Key Vault Secrets User on AppDynamics Key Vault
- Storage Blob Data Reader on agent storage account

Network Access Control

Firewall Rules Matrix:

Source	Destination	Port	Protocol	Purpose
Internet	Load Balancer	443	TCP	HTTPS Web Access
Internet	Load Balancer	80	TCP	HTTP Redirect
Load Balancer	Controllers	8181	TCP	Controller HTTPS
Load Balancer	Controllers	8090	TCP	Controller HTTP
Agents	Load Balancer	443	TCP	Agent Communication
Controllers	Database	3306	TCP	Database Access
Controllers	Internet	443	TCP	External Integrations
Bastion	All	22	TCP	SSH Management

Encryption Requirements

Data in Transit:

- TLS 1.2 minimum for all connections

- TLS 1.3 preferred where supported
- Perfect Forward Secrecy (PFS) required
- Strong cipher suites only

Data at Rest:

- Database encryption using AWS RDS encryption or Azure SQL TDE
- S3/Blob storage encryption using customer-managed keys
- EBS/Disk encryption using platform-managed keys
- Configuration file encryption for sensitive data

Load Balancer Infrastructure

AWS Application Load Balancer Configuration

ALB Specifications:

```
Name: appdynamics-production-alb
Scheme: Internet-facing
IP Address Type: IPv4
VPC: AppDynamics Production VPC
Availability Zones: us-east-1a, us-east-1b, us-east-1c
Security Groups: sg-appdynamics-alb
```

Target Groups:

```
Controller HTTPS Target Group:
- Name: appdynamics-controllers-https
- Protocol: HTTPS
- Port: 8181
- VPC: AppDynamics Production VPC
- Health Check Path: /controller/rest/serverstatus
- Health Check Protocol: HTTPS
- Health Check Port: 8181
- Healthy Threshold: 2
- Unhealthy Threshold: 3
- Timeout: 30
- Interval: 60
- Success Codes: 200

Controller HTTP Target Group:
- Name: appdynamics-controllers-http
- Protocol: HTTP
- Port: 8090
- Health Check Path: /controller/rest/serverstatus
- Health Check Protocol: HTTP
```

Listeners:

```
HTTPS Listener (Port 443):
- Protocol: HTTPS
- Port: 443
- Default Action: Forward to appdynamics-controllers-https
- SSL Certificate: arn:aws:acm:us-east-1:123456789012:certificate/abc123
- Security Policy: ELBSecurityPolicy-TLS-1-2-2017-01

HTTP Listener (Port 80):
- Protocol: HTTP
- Port: 80
- Default Action: Redirect to HTTPS
```

Azure Application Gateway Configuration

Application Gateway Specifications:

```
Name: appdynamics-production-appgw
SKU: WAF_v2
Tier: Standard_v2
Capacity: Auto-scaling (2-10)
Virtual Network: AppDynamics-VNet
Subnet: Gateway-Subnet
Public IP: Static
Zones: 1, 2, 3
```

Backend Pools:

```
Controllers Backend Pool:
- Name: controllers-backend
- Type: IP address or FQDN
- Targets:
  - 10.0.11.4 (controller-01)
  - 10.0.12.4 (controller-02)
  - 10.0.13.4 (controller-03)
```

HTTP Settings:

```
HTTPS Backend Setting:
- Name: controllers-https-setting
- Protocol: HTTPS
- Port: 8181
- Cookie Affinity: Enabled
- Request Timeout: 60
- Override Backend Path: No
- Custom Probe: controllers-health-probe
```

Health Probes:

```
Controllers Health Probe:
- Name: controllers-health-probe
- Protocol: HTTPS
- Host: Blank (use backend hostname)
- Path: /controller/rest/serverstatus
- Interval: 30
- Timeout: 30
- Unhealthy Threshold: 3
- Test Response Body: No
```

Certificate Authority (CA) Planning

CA Hierarchy Design

```
Root CA (Offline)
├── Company Root CA
│   └── AppDynamics Intermediate CA (Online)
│       ├── Load Balancer Certificates
│       ├── Controller Certificates
│       ├── Database Certificates
│       └── Agent Certificates
```

Certificate Requirements Matrix

Component	Certificate Type	Subject	Validity	Key Size	Usage
Root CA	Self-signed	CN=Company Root CA	10 years	4096-bit	Certificate Signing
Intermediate CA	CA-signed	CN=AppDynamics CA	5 years	4096-bit	Certificate Signing
Load Balancer	CA-signed	CN=appdynamics.company.com	2 years	2048-bit	Server Authentication
Controllers	CA-signed	CN=controller-XX.company.com	2 years	2048-bit	Server Authentication
Database	CA-signed	CN=mysql.company.com	2 years	2048-bit	Server Authentication
Agents	CA-signed	CN=agent.company.com	1 year	2048-bit	Client Authentication

Certificate Management

Certificate Storage:

- AWS: AWS Certificate Manager + AWS Secrets Manager
- Azure: Azure Key Vault
- On-premises: HashiCorp Vault or EJBCA

Automated Renewal:

- ACME protocol for public certificates
- Internal CA automation for private certificates
- 30-day renewal notification
- Automated deployment pipeline

Certificate Distribution Script:

```bash
#!/bin/bash
# distribute_certificates.sh

CERT_SOURCE="/etc/ssl/certs/appdynamics"
CONTROLLERS=("10.0.11.4" "10.0.12.4" "10.0.13.4")

for controller in "${CONTROLLERS[@]}"; do
  echo "Deploying certificates to $controller"

  # Copy certificates
  scp $CERT_SOURCE/controller.crt appd-admin@$controller:/tmp/
  scp $CERT_SOURCE/controller.key appd-admin@$controller:/tmp/
  scp $CERT_SOURCE/ca-bundle.crt appd-admin@$controller:/tmp/

  # Install certificates
  ssh appd-admin@$controller "
    sudo mv /tmp/controller.crt /opt/appdynamics/controller/ssl/
    sudo mv /tmp/controller.key /opt/appdynamics/controller/ssl/
    sudo mv /tmp/ca-bundle.crt /opt/appdynamics/controller/ssl/
    sudo chown appdynamics:appdynamics /opt/appdynamics/controller/ssl/*
    sudo chmod 600 /opt/appdynamics/controller/ssl/*.key
    sudo chmod 644 /opt/appdynamics/controller/ssl/*.crt
    sudo systemctl restart appdynamics-controller
  "
done
```

Storage and Agent Distribution

AWS S3 Configuration

S3 Bucket Structure:

```
appdynamics-agents-production/
├── java/
│   ├── latest/
│   │   └── AppServerAgent-4.5.19.jar
│   └── versions/
│       ├── 4.5.19/
│       └── 4.5.18/
├── dotnet/
│   ├── latest/
│   └── versions/
├── nodejs/
│   ├── latest/
│   └── versions/
├── python/
│   ├── latest/
│   └── versions/
└── machine/
    ├── latest/
    └── versions/
```

S3 Bucket Policy:

```
json
```

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAgentDownload",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::123456789012:role/AppDynamics-Agent-Role"
        ]
      },
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::appdynamics-agents-production",
        "arn:aws:s3:::appdynamics-agents-production/*"
      ]
    }
  ]
}
```

CloudFront Distribution:

```
Origin Domain: appdynamics-agents-production.s3.amazonaws.com
Origin Path: /
Viewer Protocol Policy: Redirect HTTP to HTTPS
Allowed HTTP Methods: GET, HEAD
Cached HTTP Methods: GET, HEAD
Cache Behaviors: Cache based on selected request headers
TTL Settings:
  - Default: 86400 (24 hours)
  - Maximum: 31536000 (1 year)
  - Minimum: 0
Price Class: Use All Edge Locations
```

Azure Blob Storage Configuration

Storage Account:

```
Name: appdynamicsagents
Performance: Standard
Replication: LRS (Locally Redundant Storage)
Access Tier: Hot
Secure Transfer: Required
Public Access: Disabled
```

Container Structure:

```
agents/
├── java/
├── dotnet/
├── nodejs/
├── python/
└── machine/
```

Azure CDN Profile:

```
Name: appdynamics-agents-cdn
Pricing Tier: Standard Microsoft
Endpoint Name: appdynamics-agents
Origin Type: Storage
Origin Hostname: appdynamicsagents.blob.core.windows.net
Protocol: HTTPS only
```

Agent Deployment Automation

Agent Deployment Script:

```bash
```

```
#!/bin/bash
# deploy_agents.sh

AGENT_TYPE="$1"
VERSION="$2"
ENVIRONMENT="$3"

case $AGENT_TYPE in
  "java")
    DOWNLOAD_URL="https://agents.appdynamics.company.com/java/latest/AppServerAgent-${VERSION}.jar"
    INSTALL_PATH="/opt/appdynamics/agents/java"
    ;;
  "dotnet")
    DOWNLOAD_URL="https://agents.appdynamics.company.com/dotnet/latest/dotNetAgentSetup-${VERSION}.msi"
    INSTALL_PATH="/opt/appdynamics/agents/dotnet"
    ;;
  *)
    echo "Unsupported agent type: $AGENT_TYPE"
    exit 1
    ;;
esac

# Download agent
curl -o "/tmp/agent-${VERSION}" "$DOWNLOAD_URL"

# Verify checksum
curl -o "/tmp/agent-${VERSION}.sha256" "${DOWNLOAD_URL}.sha256"
sha256sum -c "/tmp/agent-${VERSION}.sha256"

if [ $? -eq 0 ]; then
  echo "Agent downloaded and verified successfully"
  # Deploy to target environment
  ansible-playbook -i inventory/$ENVIRONMENT deploy-agent.yml \
    -e agent_type=$AGENT_TYPE \
    -e agent_version=$VERSION \
    -e agent_path="/tmp/agent-${VERSION}"
else
  echo "Agent verification failed"
  exit 1
fi
```

Database Infrastructure Planning

AWS RDS Configuration

Primary Database:

```
Engine: MySQL 8.0
Instance Class: db.r5.2xlarge (8 vCPU, 64GB RAM)
Storage: 1TB gp3 SSD
IOPS: 3000 provisioned
Multi-AZ: Yes
VPC: AppDynamics Production VPC
Subnet Group: appdynamics-db-subnet-group
Security Groups: sg-appdynamics-db
Parameter Group: Custom (appdynamics-mysql-params)
Backup Retention: 30 days
Backup Window: 03:00-04:00 UTC
Maintenance Window: Sun:04:00-Sun:05:00 UTC
Encryption: Yes (AWS KMS)
Monitoring: Enhanced monitoring enabled
```

Read Replica:

```
Engine: MySQL 8.0
Instance Class: db.r5.xlarge (4 vCPU, 32GB RAM)
Storage: Auto-scaling enabled
Multi-AZ: No
Purpose: Reporting and analytics queries
```

Database Parameter Group:

```
innodb_buffer_pool_size: 75% of available RAM
max_connections: 1000
innodb_log_file_size: 1GB
innodb_flush_log_at_trx_commit: 1
query_cache_size: 0
query_cache_type: 0
slow_query_log: 1
long_query_time: 2
```

Azure Database Configuration

Azure Database for MySQL:

```
Server Name: appdynamics-mysql-prod
Version: 8.0
Pricing Tier: General Purpose
Compute Generation: Gen 5
vCores: 16
Storage: 1TB
Backup Retention: 35 days
Geo-Redundant Backup: Enabled
SSL Enforcement: Enabled
TLS Version: 1.2
```

Connection Security:

```
Firewall Rules:
- Controller Subnet 1: 10.0.11.0/24
- Controller Subnet 2: 10.0.12.0/24
- Controller Subnet 3: 10.0.13.0/24
- Bastion Host: 10.0.1.10/32

VNet Rules:
- AppDynamics VNet: Enabled
- Service Endpoints: Microsoft.Sql
```

Monitoring and Logging Infrastructure

AWS CloudWatch Configuration

Custom Metrics:

- AppDynamics Controller CPU/Memory/Disk
- Database connection pool metrics
- Application response times
- Agent registration counts

Log Groups:

```
/aws/appdynamics/controller/application
/aws/appdynamics/controller/access
/aws/appdynamics/controller/gc
/aws/appdynamics/database/error
/aws/appdynamics/database/slow-query
```

CloudWatch Alarms:

```
Controller High CPU: >80% for 5 minutes
Controller High Memory: >85% for 5 minutes
Database High CPU: >70% for 5 minutes
Database High Connections: >800 connections
Load Balancer 5XX Errors: >10 in 5 minutes
```

Azure Monitor Configuration

Log Analytics Workspace:

```
Name: appdynamics-logs-prod
Retention: 90 days
Daily Cap: 10GB
```

Application Insights:

```
Name: appdynamics-insights
Type: Web application
Framework: Java
```

Alert Rules:

```
Controller Health: HTTP probe failure
Database Connectivity: Connection timeout
High Error Rate: >5% 5XX responses
Certificate Expiry: <30 days remaining
```

Backup and Disaster Recovery Planning

Backup Strategy

Database Backups:

- Automated daily backups with 30-day retention
- Weekly full backups with 1-year retention
- Transaction log backups every 15 minutes
- Cross-region backup replication

Configuration Backups:

- Daily backup of controller configuration
- Version-controlled infrastructure as code
- Certificate and key backup to secure storage
- Agent configuration templates backup

Recovery Objectives:

- RTO (Recovery Time Objective): 4 hours
- RPO (Recovery Point Objective): 15 minutes
- Cross-region failover capability
- Automated failover for database

Disaster Recovery Sites

AWS Multi-Region Setup:

```
Primary Region: us-east-1
DR Region: us-west-2
Replication: Cross-region RDS read replica
DNS: Route 53 health checks with failover
Storage: Cross-region S3 replication
```

Azure Multi-Region Setup:

```
Primary Region: East US
DR Region: West US 2
Replication: Azure Database geo-replication
DNS: Azure Traffic Manager with priority routing
Storage: Geo-redundant storage (GRS)
```

Pre-Installation Checklist

Infrastructure Readiness

Network Infrastructure:

- [ ] VPC/VNet created with proper CIDR blocks
- [ ] Subnets created in multiple availability zones
- [ ] Internet Gateway and NAT Gateways configured
- [ ] Route tables configured correctly
- [ ] Security Groups/NSGs created and tested
- [ ] Network ACLs configured (if required)
- [ ] DNS zones and records created
- [ ] Load balancer deployed and configured
- [ ] SSL certificates obtained and installed

Compute Infrastructure:

- [ ] EC2 instances/Azure VMs launched with correct sizing
- [ ] Auto Scaling Groups configured
- [ ] Instance profiles/Managed identities assigned
- [ ] Security patches applied
- [ ] Monitoring agents installed
- [ ] Backup agents configured

Storage Infrastructure:

- [ ] Database servers deployed and configured
- [ ] Database security groups configured
- [ ] Database parameter groups optimized
- [ ] S3 buckets/Blob storage created
- [ ] CDN distributions configured
- [ ] Backup storage configured

Security Infrastructure:

- [ ] IAM roles and policies created
- [ ] Service accounts created on Linux systems
- [ ] SSH keys distributed
- [ ] Certificates generated and distributed
- [ ] Secrets management configured
- [ ] Network security rules tested

Pre-Installation Testing

Network Connectivity:

```bash
# Test database connectivity
telnet mysql.appdynamics.company.com 3306

# Test load balancer health checks
curl -I https://appdynamics.company.com/controller/rest/serverstatus

# Test SSL certificate
openssl s_client -connect appdynamics.company.com:443 -servername appdynamics.company.com

# Test agent download
curl -I https://agents.appdynamics.company.com/java/latest/AppServerAgent-4.5.19.jar
```

Security Testing:

```bash
# Port scanning
nmap -sS appdynamics.company.com

# SSL configuration testing
nmap --script ssl-enum-ciphers -p 443 appdynamics.company.com

# Certificate validation
curl --cacert /etc/ssl/certs/ca-bundle.crt https://appdynamics.company.com
```

Performance Testing:

```bash
# Load balancer performance
ab -n 1000 -c 10 https://appdynamics.company.com/controller/

# Database performance
sysbench oltp_read_write --mysql-host=mysql.appdynamics.company.com \
  --mysql-user=appdynamics --mysql-password=password \
  --mysql-db=controller prepare
```

Documentation Requirements

Required Documentation:

- [ ] Network architecture diagrams
- [ ] Security architecture documentation
- [ ] Certificate management procedures
- [ ] Backup and recovery procedures
- [ ] Monitoring and alerting runbooks
- [ ] Incident response procedures
- [ ] Change management procedures
- [ ] Disaster recovery plan

Handover Documentation:

- [ ] System administrator guide
- [ ] Database administrator guide
- [ ] Security administrator guide
- [ ] Application owner guide
- [ ] Troubleshooting guide
- [ ] Performance tuning guide

This comprehensive planning guide ensures all infrastructure components, security requirements, and preparatory steps are properly organized before beginning the AppDynamics installation process.