LY StatusPage - Complete Architecture Diagram

Current Infrastructure Layout

```
Internet Users
    |
    | HTTPS/HTTP
    v
+---------------------------+
| Route53 DNS               | ← (Future: Custom Domain)
| ly-statuspage.com         |
+---------------------------+
    |
    v
+---------------------------+
| Application Load Balancer | ← NEW: Just Created
| (Public Subnets)          |
| Port 80/443 → 30080       |
+---------------------------+
    |
    | Target Group
    v
+=====================================================+
| AWS VPC (10.0.0.0/16)                               |
+=====================================================+
| Public Subnets (10.0.1.0/24, 10.0.2.0/24)           |
| +-----------------------------------------------+ |
| | Internet Gateway                              | |
| | NAT Gateway                                   | |
| | Jenkins Server (t3.large, 25GB)               | |
| | ALB (Application Load Balancer)               | |
| +-----------------------------------------------+ |
+=====================================================+
| Private Subnets (10.0.3.0/24, 10.0.4.0/24)          |
| +-----------------------------------------------+ |
| | EKS Cluster: ly-statuspage-cluster            | |
| | +-------------------------------------------+ | |
| | | Control Plane (AWS Managed)               | | |
| | +-------------------------------------------+ | |
| | | Worker Nodes (t3.small Spot Instances)    | | |
| | | +---------------------------------------+ | | |
| | | | Pod: Django StatusPage (3 replicas)   | | | |
```

```
| | | | Pod: Prometheus                    | | | |
| | | | Pod: Grafana                       | | | |
| | | +-----------------------------------+ | | |
| | +-------------------------------------+ | |
| +-----------------------------------------+ |
+=====================================================+
| Data Layer (Private Subnets)                        |
| +-----------------------------------------------+ |
| | RDS PostgreSQL                                | |
| | - db.m5.large                                 | |
| | - Engine: postgres 16.4                       | |
| | - Storage: 20GB encrypted                     | |
| | - Database: statuspage                        | |
| | - User: statuspage                            | |
| +-----------------------------------------------+ |
| | ElastiCache Redis                             | ← NEW: Just Created
| | - cache.t3.micro                              | |
| | - Single node cluster                         | |
| | - Port 6379                                   | |
| | - Encrypted at rest                           | |
| +-----------------------------------------------+ |
+=====================================================+
| Container Registry                                  |
| +-----------------------------------------------+ |
| | ECR Repository                                | |
| | ly-statuspage-repo                            | |
| +-----------------------------------------------+ |
+=====================================================+
```

How ALB Works - Connection Flow

ALB Target Configuration:

```
ALB → Target Group → EKS Worker Nodes (Port 30080)
                ↓
            NodePort Service
                ↓
            Django Pods (Port 8000)
```

Detailed ALB Operation:

1. Target Registration:

```bash
# ALB automatically discovers EKS worker nodes
# Target Group monitors nodes on port 30080
# Health checks sent to NodePort service
```

2. Traffic Routing:

```
User Request → ALB (Port 80) → Worker Node (Port 30080) → Pod (Port 8000)
```

3. Load Balancing:

- ALB distributes traffic between available EKS nodes
- If node fails, ALB routes to healthy nodes
- Health checks ensure only healthy targets receive traffic

Security Groups Flow:

```
Internet → ALB SG (80,443) → EKS Nodes SG (30080) → Pods
                                ↓
                            RDS SG (5432) ← EKS Nodes only
                                ↓
                        ElastiCache SG (6379) ← EKS Nodes only
```

Network Connectivity Map:

Public Network:

- **Internet Gateway**: External access
- **ALB**: Public-facing load balancer
- **Jenkins**: CI/CD server (SSH + Web access)

Private Network:

- **EKS Nodes**: Application containers
- **RDS**: Database backend
- **ElastiCache**: Cache layer
- **NAT Gateway**: Outbound internet for private resources

Component Relationships:

```
Jenkins CI/CD Pipeline:
GitHub → Jenkins → ECR → EKS Deployment

Application Data Flow:
Django Pods ↔ ElastiCache (Cache)
Django Pods ↔ RDS PostgreSQL (Persistent Data)

Monitoring Flow:
Prometheus (Scrapes metrics from Django)
Grafana (Visualizes Prometheus data)
CloudWatch (AWS infrastructure metrics)
```

Cost Breakdown (Updated):

Component	Monthly Cost
EKS Control Plane	$72
EKS Nodes (t3.small SPOT)	$6-8
Jenkins (t3.large)	$60
RDS (db.m5.large)	$95
ElastiCache (t3.micro)	$15
ALB	$16
NAT Gateway	$45
Storage & Networking	$10
Total	**~$319-329**

18 days until 21/9: ~$191-197

Current Status:

- ✅ Complete infrastructure deployed
- ✅ ALB with target groups configured
- ✅ ElastiCache Redis cluster ready
- ⏳ Next: Deploy applications to Kubernetes
- ⏳ Next: Configure Jenkins CI/CD pipeline
- ⏳ Next: Set up monitoring dashboards