Final Architecture - Konecta ERP with CloudFront + EKS

Architecture Overview

```
User → CloudFront (HTTPS) → ALB → EKS (Frontend + Backend Pods)
                                     ↓
                               ECR (Docker Images)
                                     ↓
                               RDS (Database)
```

Why CloudFront Was Added Back

✅ Security Benefits

1. **HTTPS by Default**: CloudFront provides SSL/TLS termination with free certificates
2. **DDoS Protection**: AWS Shield Standard automatically protects against common attacks
3. **WAF Ready**: Easy to integrate AWS WAF for advanced security rules
4. **Security Headers**: Managed security headers policy (HSTS, X-Frame-Options, etc.)
5. **Origin Protection**: ALB is not directly exposed to internet, only accessible via CloudFront

✅ Multi-Region Support

1. **Edge Locations**: 400+ global edge locations for low latency
2. **Global Distribution**: Content cached at edge locations worldwide
3. **Geographic Failover**: Can route to different regions based on health
4. **Route 53 Integration**: Can add multiple origins for active-active deployment

✅ Performance Benefits

1. **Global CDN**: Content served from edge locations nearest to users
2. **Compression**: Automatic Gzip/Brotli compression enabled
3. **Caching**: Smart caching reduces load on EKS cluster
4. **Latency Reduction**: Up to 50% faster response times globally

Infrastructure Components

1. CloudFront Distribution

Purpose: Global CDN and HTTPS termination

Key Features:

- ✅ HTTPS only (redirects HTTP to HTTPS)
- ✅ TLS 1.2/1.3 support
- ✅ Origin-protected communication to ALB

- ✅ Security headers policy
- ✅ Compression enabled
- ✅ Caching optimized for compression
- ✅ Custom error handling (SPA support)

Configuration:

```
- Origin: ALB (EKS cluster)
- Protocol: HTTPS only
- Viewer Protocol: Redirect to HTTPS
- Cache Policy: Optimized for compression
- Security Headers: Managed policy
- Custom Headers: X-Forwarded-Proto
```

2. ALB (Application Load Balancer)

Purpose: Route traffic to EKS cluster

Key Features:

- ✅ Health checks on `/health` endpoint
- ✅ Target group with EKS nodes
- ✅ Security groups configured
- ✅ Public subnets

3. EKS Cluster

Purpose: Run containerized frontend and backend applications

Key Features:

- ✅ Kubernetes 1.29
- ✅ Auto-scaling node groups
- ✅ SSH access via bastion host
- ✅ Private subnets (only accessible via ALB)
- ✅ Security group allowing ALB traffic

4. ECR (Elastic Container Registry)

Purpose: Store Docker images for microservices

Repositories Created:

- auth-service
- hr-service
- finance-service
- operation-service
- gateway-service
- discovery-server

- config-server
- reporting-service

Security Features:

- ✅ Image scanning on push
- ✅ AES256 encryption
- ✅ Lifecycle policies (keep last 10 images)
- ✅ IAM permissions for EKS nodes

5. RDS Database

Purpose: PostgreSQL database for application data

Key Features:

- ✅ Private subnets (not internet accessible)
- ✅ Security group allowing EKS access only
- ✅ Automated backups
- ✅ Encryption at rest

Security Layers

Layer 1: CloudFront (Internet-facing)

- DDoS protection (Shield Standard)
- SSL/TLS termination
- Security headers
- Geo-restrictions (configurable)
- WAF ready

Layer 2: ALB (VPC)

- Security groups restricting access
- Health checks
- SSL passthrough to CloudFront

Layer 3: EKS Cluster (Private)

- Network isolation
- Pod security policies
- Service mesh ready (Istio compatible)
- Bastion host for SSH access

Layer 4: RDS (Private)

- Network isolation
- Encryption at rest
- Security group isolation
- No public access

Traffic Flow

1. User Request

```
User → https://d1234567890.cloudfront.net
```

2. CloudFront Edge Location

- Checks edge cache first
- If cached, serves immediately (fastest)
- If not cached, fetches from origin

3. ALB Origin Request

```
CloudFront → https://alb-1234567890.us-east-1.elb.amazonaws.com
```

- CloudFront adds custom headers:
  - `X-Forwarded-Proto: https`
  - `X-Real-IP: <user-ip>`
  - Security headers

4. EKS Cluster

```
ALB → EKS Pods (via Target Group)
```

- Kubernetes service receives request
- Routes to appropriate pod
- Returns response

5. Response Path

```
EKS → ALB → CloudFront → User
```

- Response cached at edge for static content
- Dynamic content passes through

HTTPS/SSL Configuration

CloudFront SSL

- **Certificate**: AWS managed (free)
- **Domain**: `*.cloudfront.net`

- **Protocol**: TLS 1.2/1.3
- **Cipher Suites**: Modern, secure
- **Perfect Forward Secrecy**: Yes

Adding Custom Domain

To use your own domain (e.g., `erp.example.com`):

1. **Request ACM Certificate** (Route 53 or external DNS)
2. **Update CloudFront** to use ACM certificate
3. **Create Route 53 Record** pointing to CloudFront distribution
4. **DNS Verification** (for certificate)

Example Terraform:

```
resource "aws_acm_certificate" "main" {
  domain_name       = "erp.example.com"
  validation_method = "DNS"

  lifecycle {
    create_before_destroy = true
  }
}

# Update CloudFront viewer_certificate
viewer_certificate {
  acm_certificate_arn = aws_acm_certificate.main.arn
  ssl_support_method = "sni-only"
}
```

Cost Analysis

CloudFront Costs (CloudFront in front of EKS):

- **Data Transfer OUT**: $0.085/GB (first 1TB free in first year)
- **HTTPS Requests**: $0.01 per 10,000 requests
- **Cache Hit Ratio**: Better caching = lower ALB/EKS costs
- **Estimated**: $10-50/month for moderate traffic

EKS Costs:

- **Cluster**: $0.10/hour = $72/month
- **EC2 Nodes**: $60-90/month (t3.medium)
- **EBS Volumes**: $10/month
- **Total**: ~$142-172/month

ECR Costs:

- **Storage**: $0.10/GB/month

- **Data Transfer**: $0.02/GB
- **Estimated**: $5-10/month (images are small)

RDS Costs:

- **Instance**: $15-30/month (db.t3.micro)
- **Storage**: $0.115/GB/month
- **Backups**: Included (20GB)
- **Total**: ~$20-40/month

Total Estimated Cost:

- **Infrastructure**: ~$180-250/month
- **Within AWS Free Tier**: First 12 months (some services free)

Benefits Summary

Security ✅

- Multi-layer security approach
- DDoS protection
- Encryption in transit and at rest
- Network isolation
- IAM-based access control

Performance ✅

- Global edge caching
- Low latency worldwide
- Compression enabled
- Smart caching reduces backend load

Scalability ✅

- Auto-scaling EKS nodes
- Global CDN handles traffic spikes
- Database read replicas ready
- Multi-region ready

Reliability ✅

- 99.99% availability SLA (CloudFront)
- Multiple availability zones
- Health checks and monitoring
- Automated failover capabilities

Next Steps

1. Deploy Infrastructure

```
cd Cloud-Konecta-ERP/terraform
terraform init -reconfigure
terraform plan
terraform apply
```

2. Push Docker Images to ECR

```
# Get ECR repository URLs
terraform output ecr_repositories

# Login to ECR
aws ecr get-login-password --region us-east-1 | \
  docker login --username AWS --password-stdin <account>.dkr.ecr.us-east-
1.amazonaws.com

# Build and push each service
docker build -t konecta-erp/auth-service ./auth-service
docker tag konecta-erp/auth-service:latest <account>.dkr.ecr.us-east-
1.amazonaws.com/konecta-erp/auth-service:latest
docker push <account>.dkr.ecr.us-east-1.amazonaws.com/konecta-erp/auth-
service:latest
```

3. Deploy to EKS

```
# Configure kubectl
aws eks update-kubeconfig --region us-east-1 --name konecta-erp-prod

# Deploy applications
kubectl apply -f k8s/deployments/
kubectl apply -f k8s/services/
```

4. Access Application

```
# Get CloudFront domain
terraform output cloudfront_domain

# Access via HTTPS
curl https://$(terraform output -raw cloudfront_domain)
```

Multi-Region Expansion (Future)

When ready for multi-region:

1. **Create secondary region**: Copy Terraform to `terraform/dev2/`

2. **Deploy EKS in secondary region**: Same configuration
3. **RDS Cross-Region Read Replica**: Add in CloudFront module
4. **Update CloudFront**: Add multiple origins
5. **Route 53**: Health check-based failover
6. **Global Load Balancing**: Geo-routing capabilities

CloudFront seamlessly supports this when you're ready to expand!