ThreatWerk - AWS Setup Guide

Preparing AWS Infrastructure for ThreatWerk

July 2026

1 Overview

This guide covers the AWS infrastructure preparation required before deploying ThreatWerk. It is platform-agnostic - the same infrastructure (VPC, RDS, IAM, Secrets Manager) is needed whether you deploy via EKS/Helm or ECS Fargate.

After completing this guide, continue with:

- *Helm Deployment Guide* - if deploying on EKS (recommended for Marketplace customers)
- *ECS Deployment Guide* - if deploying on ECS Fargate (simpler, no Kubernetes needed)

Time estimate: 30-60 minutes depending on your existing infrastructure.

1.1 What This Guide Provisions

Resource	Purpose
VPC + subnets	Network isolation for application and database
Security groups	Firewall rules between ALB, compute, and RDS
RDS PostgreSQL	Persistent data store
AWS Secrets Manager	Credential storage (DATABASE_URL, JWT_SECRET, ENCRYPTION_KEY)
IAM role + policy	Least-privilege access to Secrets Manager and License Manager
ACM certificate	TLS termination for HTTPS

1.2 Prerequisites

- An AWS account with admin-level access (or specific permissions for VPC, RDS, IAM, ACM, Secrets Manager)
- AWS CLI v2 configured (`aws sts get-caller-identity` confirms the right account)
- A domain name you control (for the TLS certificate and DNS record)
- `openssl` installed (for generating secrets)

2 VPC and Networking

ThreatWerk requires a VPC with public and private subnets across at least two Availability Zones.

Subnet type	Purpose	Internet access	Hosts
Public (2+ AZs)	Load balancer	Direct	ALB, NAT Gateway
Private (2+ AZs)	Application compute	Via NAT (outbound only)	ECS tasks or EKS nodes
Private (2+ AZs)	Database	None	RDS instance

2.1 Option A: Use an Existing VPC

If you already have a VPC, verify:

- At least 2 public subnets (for the ALB)
- At least 2 private subnets (for compute)
- NAT Gateway providing outbound internet from private subnets
- DNS hostnames and DNS resolution enabled
- For EKS: public subnets tagged `kubernetes.io/role/elb=1`, private subnets tagged `kubernetes.io/role/internal-elb=1`

If your VPC meets these requirements, skip to the Security Groups section.

2.2 Option B: Create a New VPC

```
# Create VPC
aws ec2 create-vpc --cidr-block 10.0.0.0/16 --tag-specifications \
  'ResourceType=vpc,Tags=[{Key=Name,Value=threatwerk-vpc}]'

# Enable DNS (required for RDS and EKS)
aws ec2 modify-vpc-attribute --vpc-id VPC_ID --enable-dns-support
aws ec2 modify-vpc-attribute --vpc-id VPC_ID --enable-dns-hostnames
```

2.2.1 Create Subnets

```
# Public subnets (for ALB)
aws ec2 create-subnet --vpc-id VPC_ID \
  --cidr-block 10.0.1.0/24 --availability-zone REGION-a \
  --tag-specifications \
  'ResourceType=subnet,Tags=[{Key=Name,Value=threatwerk-public-a},{Key=kubernetes.io/role/elb,Value=1}]'

aws ec2 create-subnet --vpc-id VPC_ID \
  --cidr-block 10.0.2.0/24 --availability-zone REGION-b \
  --tag-specifications \
  'ResourceType=subnet,Tags=[{Key=Name,Value=threatwerk-public-b},{Key=kubernetes.io/role/elb,Value=1}]'

# Private subnets (for compute)
aws ec2 create-subnet --vpc-id VPC_ID \
  --cidr-block 10.0.10.0/24 --availability-zone REGION-a \
  --tag-specifications \
  'ResourceType=subnet,Tags=[{Key=Name,Value=threatwerk-private-a},{Key=kubernetes.io/role/internal-elb,Value=1}]'

aws ec2 create-subnet --vpc-id VPC_ID \
  --cidr-block 10.0.11.0/24 --availability-zone REGION-b \
  --tag-specifications \
  'ResourceType=subnet,Tags=[{Key=Name,Value=threatwerk-private-b},{Key=kubernetes.io/role/internal-elb,Value=1}]'
```

2.2.2 Create Internet Gateway and NAT Gateway

```
# Internet Gateway
aws ec2 create-internet-gateway --tag-specifications \
  'ResourceType=internet-gateway,Tags=[{Key=Name,Value=threatwerk-igw}]'
aws ec2 attach-internet-gateway --internet-gateway-id IGW_ID --vpc-id VPC_ID

# Elastic IP for NAT
aws ec2 allocate-address --domain vpc --tag-specifications \
  'ResourceType=elastic-ip,Tags=[{Key=Name,Value=threatwerk-nat-eip}]'

# NAT Gateway in a public subnet
aws ec2 create-nat-gateway --subnet-id PUBLIC_SUBNET_A_ID \
  --allocation-id EIP_ALLOC_ID --tag-specifications \
  'ResourceType=natgateway,Tags=[{Key=Name,Value=threatwerk-nat}]'
```

2.2.3 Configure Route Tables

```
# Public route table - routes to Internet Gateway
aws ec2 create-route-table --vpc-id VPC_ID --tag-specifications \
  'ResourceType=route-table,Tags=[{Key=Name,Value=threatwerk-public-rt}]'
aws ec2 create-route --route-table-id PUBLIC_RT_ID \
  --destination-cidr-block 0.0.0.0/0 --gateway-id IGW_ID
aws ec2 associate-route-table --route-table-id PUBLIC_RT_ID --subnet-id PUBLIC_SUBNET_A_ID
aws ec2 associate-route-table --route-table-id PUBLIC_RT_ID --subnet-id PUBLIC_SUBNET_B_ID

# Private route table - routes to NAT Gateway
aws ec2 create-route-table --vpc-id VPC_ID --tag-specifications \
  'ResourceType=route-table,Tags=[{Key=Name,Value=threatwerk-private-rt}]'
aws ec2 create-route --route-table-id PRIVATE_RT_ID \
  --destination-cidr-block 0.0.0.0/0 --nat-gateway-id NAT_GW_ID
aws ec2 associate-route-table --route-table-id PRIVATE_RT_ID --subnet-id PRIVATE_SUBNET_A_ID
aws ec2 associate-route-table --route-table-id PRIVATE_RT_ID --subnet-id PRIVATE_SUBNET_B_ID
```

3 Security Groups

Create three security groups for network isolation between the load balancer, compute layer, and database.

3.1 ALB Security Group

```
aws ec2 create-security-group --group-name threatwerk-alb-sg \
  --description "ThreatWerk ALB" --vpc-id VPC_ID

# Inbound: HTTPS from your corporate network (restrict to your CIDR)
aws ec2 authorize-security-group-ingress --group-id ALB_SG_ID \
  --protocol tcp --port 443 --cidr YOUR_CORPORATE_CIDR

# Outbound: to compute on port 8080
aws ec2 authorize-security-group-egress --group-id ALB_SG_ID \
  --protocol tcp --port 8080 --source-group COMPUTE_SG_ID
```

3.2 Compute Security Group (ECS tasks or EKS nodes)

```
aws ec2 create-security-group --group-name threatwerk-compute-sg \
  --description "ThreatWerk compute" --vpc-id VPC_ID

# Inbound: from ALB on port 8080
aws ec2 authorize-security-group-ingress --group-id COMPUTE_SG_ID \
  --protocol tcp --port 8080 --source-group ALB_SG_ID

# Outbound: to RDS on port 5432
aws ec2 authorize-security-group-egress --group-id COMPUTE_SG_ID \
  --protocol tcp --port 5432 --source-group RDS_SG_ID

# Outbound: HTTPS to internet (AWS APIs, intel feeds, ECR pulls)
aws ec2 authorize-security-group-egress --group-id COMPUTE_SG_ID \
  --protocol tcp --port 443 --cidr 0.0.0.0/0
```

For EKS, also allow:

```
# Inbound: from EKS control plane (API server, kubelet)
aws ec2 authorize-security-group-ingress --group-id COMPUTE_SG_ID \
  --protocol tcp --port 443 --source-group EKS_CONTROL_PLANE_SG_ID
aws ec2 authorize-security-group-ingress --group-id COMPUTE_SG_ID \
  --protocol tcp --port 10250 --source-group EKS_CONTROL_PLANE_SG_ID
```

3.3 RDS Security Group

```
aws ec2 create-security-group --group-name threatwerk-rds-sg \
  --description "ThreatWerk RDS" --vpc-id VPC_ID

# Inbound: PostgreSQL from compute only
aws ec2 authorize-security-group-ingress --group-id RDS_SG_ID \
  --protocol tcp --port 5432 --source-group COMPUTE_SG_ID
```

Do not open port 5432 to any broader CIDR.

4 Database (RDS PostgreSQL)

4.1 Create the RDS Instance

```
# DB subnet group
aws rds create-db-subnet-group \
  --db-subnet-group-name threatwerk-db-subnets \
  --db-subnet-group-description "ThreatWerk RDS subnets" \
  --subnet-ids PRIVATE_SUBNET_A_ID PRIVATE_SUBNET_B_ID

# RDS instance
aws rds create-db-instance \
  --db-instance-identifier threatwerk-db \
  --db-instance-class db.t3.medium \
  --engine postgres \
  --engine-version 16.4 \
  --master-username masteruser \
  --master-user-password MASTER_PASSWORD \
  --allocated-storage 50 \
  --storage-type gp3 \
  --storage-encrypted \
  --multi-az \
  --db-subnet-group-name threatwerk-db-subnets \
  --vpc-security-group-ids RDS_SG_ID \
  --no-publicly-accessible \
  --backup-retention-period 7 \
  --deletion-protection \
  --copy-tags-to-snapshot

# Wait for availability (5-10 minutes)
aws rds wait db-instance-available --db-instance-identifier threatwerk-db
```

4.1.1 Recommended Settings

Setting	Small team (5-20)	Medium team (20-50)	Enterprise (50+)
Instance class	`db.t3.medium`	`db.r6g.large`	`db.r6g.xlarge`
Storage	50 GB gp3	100 GB gp3	200 GB gp3
Multi-AZ	Yes	Yes	Yes

4.2 Create the Application Database and User

If you deployed using the CloudFormation template (`threatwerk-ecs.yaml`): The `threatwerk` database is created automatically. Skip this section and proceed to Secrets Manager.

If provisioning RDS manually: Connect to RDS as the master user. Since RDS is in a private subnet, use one of:

Option A: VPN/bastion with direct access:

```
psql "postgres://masteruser:MASTER_PASSWORD@RDS_ENDPOINT:5432/postgres?sslmode=require"
```

Option B: SSM port forwarding (no VPN required):

```
aws ssm start-session --target INSTANCE_ID \
  --document-name AWS-StartPortForwardingSessionToRemoteHost \
  --parameters '{"host":["RDS_ENDPOINT"],"portNumber":["5432"],"localPortNumber":["5432"]}'

# In another terminal:
psql "postgres://masteruser:MASTER_PASSWORD@localhost:5432/postgres?sslmode=require"
```

Run these SQL statements:

```
CREATE USER threatwerk WITH PASSWORD 'STRONG_RANDOM_PASSWORD';
CREATE DATABASE threatwerk OWNER threatwerk;
\c threatwerk
GRANT ALL ON SCHEMA public TO threatwerk;
```

Use a strong random password (32+ characters). This password goes into `DATABASE_URL` in Secrets Manager.

The application user needs DDL privileges because ThreatWerk runs schema migrations automatically on startup.

5 Secrets Manager

Store credentials as a JSON object in Secrets Manager. The backend fetches these at startup.

```
RDS_ENDPOINT=$(aws rds describe-db-instances \
  --db-instance-identifier threatwerk-db \
  --query "DBInstances[0].Endpoint.Address" --output text)

aws secretsmanager create-secret \
  --name threatwerk-production-secrets \
  --description "ThreatWerk application secrets" \
  --secret-string '{
    "DATABASE_URL": "postgres://threatwerk:APP_PASSWORD@'$RDS_ENDPOINT':5432/threatwerk?sslmode=require",
    "JWT_SECRET": "'$(openssl rand -base64 48)'",
    "ENCRYPTION_KEY": "'$(openssl rand -base64 48)'"
  }'
```

5.0.1 Required Keys

Key	Purpose
`DATABASE_URL`	PostgreSQL connection string (must include `sslmode=require`)
`JWT_SECRET`	HMAC signing key for session tokens (min 32 chars)
`ENCRYPTION_KEY`	AES-GCM key for encrypting stored credentials (min 32 chars)

5.0.2 Optional Keys

Key	Purpose
`NVD_API_KEY`	Higher rate limits for NVD CVE feed
`OTX_API_KEY`	AlienVault OTX threat intel feed
`GITHUB_TOKEN`	GitHub Security Advisories feed

Important: `JWT_SECRET` must remain stable across deployments - changing it invalidates all sessions. `ENCRYPTION_KEY` encrypts SBOM source credentials - changing it makes existing SBOM configurations unreadable.

6 IAM Configuration

The backend needs IAM permissions for Secrets Manager and License Manager (and optionally SES and S3).

6.1 IAM Policy

```
ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)

aws iam create-policy \
  --policy-name ThreatWerkPodPolicy \
  --policy-document '{
    "Version": "2012-10-17",
    "Statement": [{
      "Sid": "SecretsManagerRead",
      "Effect": "Allow",
      "Action": ["secretsmanager:GetSecretValue"],
      "Resource": ["arn:aws:secretsmanager:REGION:'$ACCOUNT_ID':secret:threatwerk-*"]
    }, {
      "Sid": "LicenseManager",
      "Effect": "Allow",
      "Action": [
        "license-manager:CheckoutLicense",
        "license-manager:CheckInLicense",
        "license-manager:ExtendLicenseConsumption",
        "license-manager:GetLicense",
        "license-manager:ListReceivedLicenses"
      ],
      "Resource": ["*"]
    }]
  }'
```

6.1.1 Optional Policy Statements

Add these if using SES email or S3 SBOM integration:

```
{
  "Sid": "SESEmail",
  "Effect": "Allow",
  "Action": ["ses:SendEmail", "ses:SendRawEmail"],
  "Resource": ["arn:aws:ses:REGION:ACCOUNT:identity/YOUR_DOMAIN"]
},
{
  "Sid": "S3SBOMRead",
  "Effect": "Allow",
  "Action": ["s3:GetObject", "s3:ListBucket"],
  "Resource": ["arn:aws:s3:::YOUR_BUCKET", "arn:aws:s3:::YOUR_BUCKET/*"]
}
```

6.1.2 Permission Summary

Statement	Service	Actions	Resource	Purpose
SecretsManagerRead	Secrets Manager	`GetSecretValue`	Single secret ARN	Fetch DB URL, JWT key, encryption key
LicenseManager	License Manager	Checkout, CheckIn, Extend, Get, List	`*` (API requirement)	Validate Marketplace license
SESEmail	SES	`SendEmail`, `SendRawEmail`	Verified identity	Email notifications (optional)

Statement	Service	Actions	Resource	Purpose
S3SBOMRead	S3	`GetObject`, `ListBucket`	Specific bucket	SBOM file fetch (optional)

6.2 IAM Role - EKS (IRSA)

For EKS deployments, create a role with an IRSA trust policy:

```
OIDC_URL=$(aws eks describe-cluster --name YOUR_CLUSTER \
  --query "cluster.identity.oidc.issuer" --output text)
OIDC_ID=$(echo $OIDC_URL | cut -d'/' -f5)

aws iam create-role \
  --role-name ThreatWerkPodRole \
  --assume-role-policy-document '{
    "Version": "2012-10-17",
    "Statement": [{
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::'$ACCOUNT_ID':oidc-provider/oidc.eks.REGION.amazonaws.com/id/'$OIDC_ID'"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "oidc.eks.REGION.amazonaws.com/id/'$OIDC_ID':sub": "system:serviceaccount:threatwerk:threatwerk"
        }
      }
    }]
  }'

aws iam attach-role-policy \
  --role-name ThreatWerkPodRole \
  --policy-arn arn:aws:iam::${ACCOUNT_ID}:policy/ThreatWerkPodPolicy
```

Note: The trust policy restricts this role to the `threatwerk` service account in the `threatwerk` namespace only. No other pod can assume it.

6.3 IAM Role - ECS Fargate

For ECS, the CloudFormation template creates roles automatically. If provisioning manually, create a Task Role with the same policy attached and configure the ECS task definition to use it.

7 ACM Certificate

Request a TLS certificate for your domain:

```
aws acm request-certificate \
  --domain-name threatwerk.yourcompany.com \
  --validation-method DNS \
  --region REGION
```

Then add the DNS validation CNAME record as shown in the ACM console. The certificate must be validated before the ALB can use it.

Note the certificate ARN - you will need it for the Helm values or ECS CloudFormation parameters.

8 Outbound Internet Access

The backend requires outbound HTTPS (port 443) for AWS APIs and threat intel feeds. This is provided by the NAT Gateway.

If operating in a restricted environment with egress filtering, allowlist these domains:

Domain	Purpose
`services.nvd.nist.gov`	NVD CVE feed
`www.cisa.gov`	CISA Known Exploited Vulnerabilities
`otx.alienvault.com`	AlienVault OTX threat intel
`api.github.com`	GitHub Security Advisories
`secretsmanager.REGION.amazonaws.com`	Secrets Manager
`license-manager.REGION.amazonaws.com`	License Manager

9 Next Steps

Your AWS infrastructure is ready. Continue with:

- **EKS/Helm deployment:** Follow the *Helm Deployment Guide*
- **ECS Fargate deployment:** Follow the *ECS Deployment Guide*

10 Appendix: Service Quotas

Verify these quotas before deploying:

Service	Quota	Default	Required
Amazon EKS	Clusters per region	100	1
Amazon EC2	Running On-Demand instances	Varies	2-4
Elastic Load Balancing	ALBs per region	50	1
Amazon RDS	DB instances per region	40	1
AWS Secrets Manager	Secrets per region	500,000	1
Amazon VPC	VPCs per region	5	1
Amazon VPC	NAT Gateways per AZ	5	1
Amazon VPC	Elastic IPs per region	5	1

Check and request increases: Service Quotas console