Guide Complet : Sécurisation Réseau AWS pour Applications SaaS

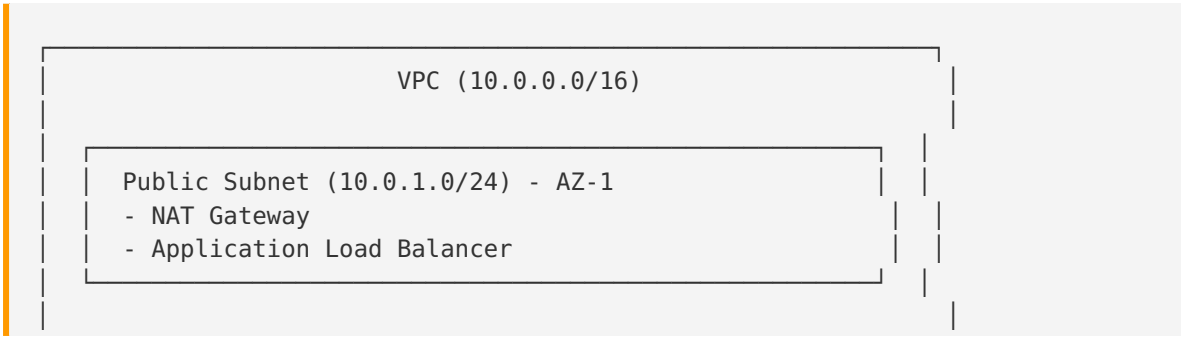
Version: 1.0
Date: Novembre 2025
Destiné à: Architectes Cloud et Équipes Réseau

Table des Matières

Architecture VPC Sécurisée

1. Principes Fondamentaux

Architecture Multi-Tier Recommandée



```
│   ┌─────────────────────────────────────────────────────┐   │
│   │  Private Subnet - App Tier (10.0.10.0/24) - AZ-1    │   │
│   │  - EC2 / ECS / Lambda                               │   │
│   │  - Pas d'IP publique                                │   │
│   └─────────────────────────────────────────────────────┘   │
│                                                             │
│   ┌─────────────────────────────────────────────────────┐   │
│   │  Private Subnet - Data Tier (10.0.20.0/24) - AZ-1   │   │
│   │  - RDS                                              │   │
│   │  - ElastiCache                                      │   │
│   │  - Accès uniquement depuis App Tier                 │   │
│   └─────────────────────────────────────────────────────┘   │
│                                                             │
│   ┌─────────────────────────────────────────────────────┐   │
│   │  Public Subnet (10.0.2.0/24) - AZ-2                 │   │
│   │  - NAT Gateway (HA)                                 │   │
│   │  - ALB (HA)                                         │   │
│   └─────────────────────────────────────────────────────┘   │
│                                                             │
│   ┌─────────────────────────────────────────────────────┐   │
│   │  Private Subnet - App Tier (10.0.11.0/24) - AZ-2    │   │
│   └─────────────────────────────────────────────────────┘   │
│                                                             │
│   ┌─────────────────────────────────────────────────────┐   │
│   │  Private Subnet - Data Tier (10.0.21.0/24) - AZ-2   │   │
│   └─────────────────────────────────────────────────────┘   │
└─────────────────────────────────────────────────────────────┘
```

2. Meilleures Pratiques VPC

2.1 Isolation Complète des Ressources Sensibles

✅ **TOUJOURS:**
- Déployer les bases de données et ressources compute dans des **sous-réseaux privés**
- **Aucune adresse IP publique** pour les ressources sensibles
- Utiliser NAT Gateway pour l'accès Internet sortant depuis les sous-réseaux privés

❌ **JAMAIS:**
- Exposer directement les bases de données sur Internet
- Utiliser un seul subnet public pour toutes les ressources
- Partager des sous-réseaux entre tiers d'application

2.2 Configuration Multi-AZ pour Haute Disponibilité

```
# Créer un VPC avec des sous-réseaux multi-AZ
aws ec2 create-vpc --cidr-block 10.0.0.0/16 --tag-specifications 'ResourceType=vpc,Tags=[{Key=
```

```
# Créer sous-réseaux publics
aws ec2 create-subnet --vpc-id vpc-xxxxx --cidr-block 10.0.1.0/24 --availability-zone us-east-
aws ec2 create-subnet --vpc-id vpc-xxxxx --cidr-block 10.0.2.0/24 --availability-zone us-east-

# Créer sous-réseaux privés - App Tier
aws ec2 create-subnet --vpc-id vpc-xxxxx --cidr-block 10.0.10.0/24 --availability-zone us-east
aws ec2 create-subnet --vpc-id vpc-xxxxx --cidr-block 10.0.11.0/24 --availability-zone us-east

# Créer sous-réseaux privés - Data Tier
aws ec2 create-subnet --vpc-id vpc-xxxxx --cidr-block 10.0.20.0/24 --availability-zone us-east
aws ec2 create-subnet --vpc-id vpc-xxxxx --cidr-block 10.0.21.0/24 --availability-zone us-east
```

Avantages Multi-AZ:
- ✅ Tolérance aux pannes d'une zone de disponibilité
- ✅ Haute disponibilité pour les applications de production
- ✅ Résilience face aux incidents régionaux

3. Connectivité Sécurisée

3.1 AWS Direct Connect vs Site-to-Site VPN

Critère	Direct Connect	Site-to-Site VPN
Latence	Faible (< 10ms)	Moyenne (50-100ms)
Bande passante	1-100 Gbps	1.25 Gbps
Coût	€€€€	€€
Sécurité	Connexion dédiée	IPsec tunnel
Cas d'usage	Production, haute performance	Dev/Test, backup

3.2 Configuration Site-to-Site VPN

```
# Créer un Virtual Private Gateway
aws ec2 create-vpn-gateway --type ipsec.1 --tag-specifications 'ResourceType=vpn-gateway,Tags=

# Attacher au VPC
aws ec2 attach-vpn-gateway --vpn-gateway-id vgw-xxxxx --vpc-id vpc-xxxxx

# Créer Customer Gateway
aws ec2 create-customer-gateway \
    --type ipsec.1 \
    --public-ip 203.0.113.12 \
    --bgp-asn 65000

# Créer VPN Connection
```

```
aws ec2 create-vpn-connection \
    --type ipsec.1 \
    --customer-gateway-id cgw-xxxxx \
    --vpn-gateway-id vgw-xxxxx
```

Security Groups et NACLs

1. Stratégie Defense-in-Depth

Security Groups et NACLs doivent être utilisés **ensemble** pour implémenter une stratégie de défense en profondeur, qui améliore la sécurité en fournissant de la redondance et en atténuant l'impact d'un seul point de défaillance.

Comparaison Security Groups vs NACLs

Caractéristique	Security Groups	NACLs
Niveau	Instance (ENI)	Subnet
État	Stateful	Stateless
Règles	Autorisations uniquement	Allow + Deny
Évaluation	Toutes les règles	Ordre séquentiel
Défaut	Deny all inbound	Allow all
Cas d'usage	Contrôle granulaire	Protection périmètre

2. Architecture Multi-Tier avec Security Groups

2.1 Référencement de Security Groups

✅ **Bonne Pratique** - Référencer les SG précédents:

```
# Web Tier Security Group
resource "aws_security_group" "web_tier" {
  name        = "web-tier-sg"
  description = "Allow HTTP/HTTPS from internet"
  vpc_id      = aws_vpc.main.id

  ingress {
    from_port   = 443
    to_port     = 443
```

```
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
}

# App Tier Security Group
resource "aws_security_group" "app_tier" {
  name        = "app-tier-sg"
  description = "Allow traffic only from web tier"
  vpc_id      = aws_vpc.main.id

  ingress {
    from_port       = 8080
    to_port         = 8080
    protocol        = "tcp"
    security_groups = [aws_security_group.web_tier.id]  # Référence le SG web
  }
}

# Database Tier Security Group
resource "aws_security_group" "db_tier" {
  name        = "db-tier-sg"
  description = "Allow traffic only from app tier"
  vpc_id      = aws_vpc.main.id

  ingress {
    from_port       = 3306
    to_port         = 3306
    protocol        = "tcp"
    security_groups = [aws_security_group.app_tier.id]  # Référence le SG app
  }
}
```

Cette approche implémente le **principe du moindre privilège** au niveau réseau.

2.2 NACLs pour Protection de Subnet

```
# Créer une NACL pour le tier base de données
aws ec2 create-network-acl --vpc-id vpc-xxxxx

# Bloquer tout le trafic Internet entrant (règle deny)
aws ec2 create-network-acl-entry \
    --network-acl-id acl-xxxxx \
    --ingress \
    --rule-number 100 \
    --protocol -1 \
    --cidr-block 0.0.0.0/0 \
    --rule-action deny

# Autoriser le trafic depuis le subnet app tier
aws ec2 create-network-acl-entry \
    --network-acl-id acl-xxxxx \
```

```
    --ingress \
    --rule-number 200 \
    --protocol tcp \
    --port-range From=3306,To=3306 \
    --cidr-block 10.0.10.0/24 \
    --rule-action allow
```

Important: Les règles NACL sont évaluées **séquentiellement** - placez les règles DENY en premier!

3. Règles de Sécurité Essentielles

3.1 Bloquer les Ports Dangereux

❌ Ports à BLOQUER pour Internet (0.0.0.0/0):

Port	Service	Risque
22	SSH	Accès administrateur
3389	RDP	Accès Windows
3306	MySQL	Base de données
5432	PostgreSQL	Base de données
27017	MongoDB	Base de données
6379	Redis	Cache
9200	Elasticsearch	Recherche

3.2 Configuration AWS Security Hub

Security Hub vérifie automatiquement:

```
# Vérifier les security groups avec accès ouvert
aws securityhub get-findings \
    --filters '{"ComplianceStatus":[{"Value":"FAILED","Comparison":"EQUALS"}],"ResourceType":
```

VPC Flow Logs et Monitoring

1. Activation des VPC Flow Logs

VPC Flow Logs capture les informations sur le trafic IP entrant et sortant des interfaces réseau dans votre VPC.

1.1 Configuration Complète

```
# Créer un rôle IAM pour Flow Logs
aws iam create-role --role-name VPCFlowLogsRole --assume-role-policy-document '{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"Service": "vpc-flow-logs.amazonaws.com"},
    "Action": "sts:AssumeRole"
  }]
}'

# Créer un log group CloudWatch
aws logs create-log-group --log-group-name /aws/vpc/flowlogs

# Activer Flow Logs (ALL = succès + échecs)
aws ec2 create-flow-logs \
    --resource-type VPC \
    --resource-ids vpc-xxxxx \
    --traffic-type ALL \
    --log-destination-type cloud-watch-logs \
    --log-group-name /aws/vpc/flowlogs \
    --deliver-logs-permission-arn arn:aws:iam::123456789012:role/VPCFlowLogsRole
```

1.2 Format Personnalisé pour Sécurité

```
# Format optimisé pour analyse de sécurité
aws ec2 create-flow-logs \
    --resource-type VPC \
    --resource-ids vpc-xxxxx \
    --traffic-type ALL \
    --log-destination-type s3 \
    --log-destination arn:aws:s3:::my-flow-logs-bucket \
    --log-format '${srcaddr} ${dstaddr} ${srcport} ${dstport} ${protocol} ${packets} ${bytes}
```

2. Cas d'Usage Sécurité

2.1 Détection de Menaces

VPC Flow Logs permet d'identifier:

Menace	Indicateur	Requête Flow Logs
Port Scanning	Multiples connexions rejetées	`action = REJECT` sur ports variés
Data Exfiltration	Volume anormal sortant	`bytes > threshold` vers IPs externes
C&C Beaconing	Connexions répétées	Même `dstaddr` avec intervalle régulier
Lateral Movement	Connexions inter-instances	`srcaddr` interne → `dstaddr` interne

2.2 Requêtes CloudWatch Logs Insights

Identifier les Top-10 IPs sources avec connexions rejetées:

```
fields @timestamp, srcAddr, dstAddr, dstPort, action
| filter action = "REJECT"
| stats count(*) as rejectedCount by srcAddr, dstPort
| sort rejectedCount desc
| limit 10
```

Détecter scan de ports:

```
fields @timestamp, srcAddr, dstPort
| filter action = "REJECT"
| stats count_distinct(dstPort) as uniquePorts by srcAddr
| filter uniquePorts > 50
| sort uniquePorts desc
```

Identifier data exfiltration (> 1GB sortant):

```
fields @timestamp, srcAddr, dstAddr, bytes
| stats sum(bytes) as totalBytes by srcAddr, dstAddr
| filter totalBytes > 1073741824
| sort totalBytes desc
```

3. CloudWatch Contributor Insights

```
{
  "Schema": {
    "Name": "CloudWatchLogRule",
```

```
    "Version": 1
  },
  "AggregateOn": "Count",
  "Contribution": {
    "Filters": [
      {
        "Match": "$.action",
        "EqualTo": "REJECT"
      }
    ],
    "Keys": [
      "$.srcAddr",
      "$.dstPort"
    ]
  },
  "LogFormat": "CLF",
  "LogGroupNames": [
    "/aws/vpc/flowlogs"
  ]
}
```

Cette règle affiche les **Top-N contributeurs** pour les connexions rejetées par port et IP source.

4. Amazon Detective pour Investigation

Amazon Detective collecte automatiquement les VPC Flow Logs et présente des **résumés visuels et analytiques**.

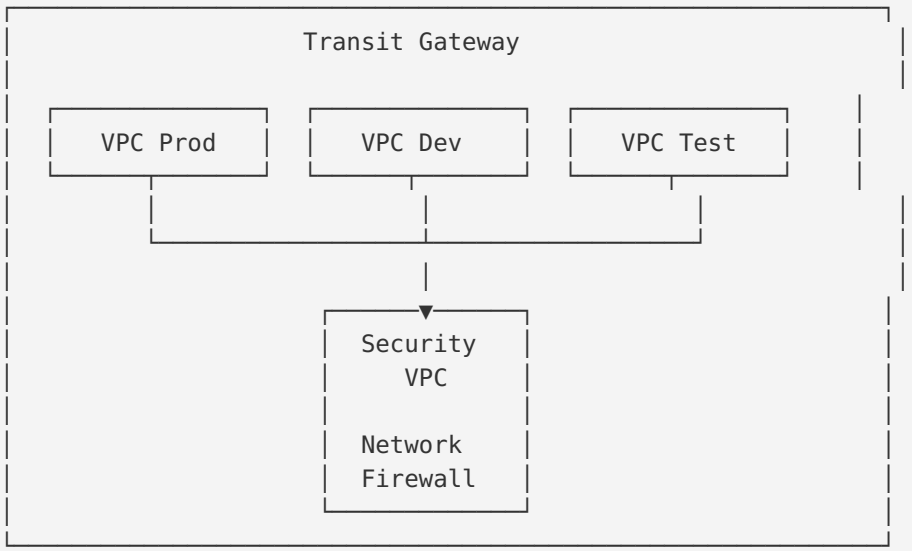
```
# Activer Detective
aws detective create-graph

# Investiguer une IP suspecte
aws detective investigate-entity \
    --graph-arn arn:aws:detective:region:account-id:graph:xxxxx \
    --entity-arn arn:aws:ec2:region:account-id:network-interface/eni-xxxxx
```

Network Firewall et Protection

1. AWS Network Firewall

AWS Network Firewall est un service managé qui permet de **filtrer le trafic entrant et sortant** au niveau du VPC.

1.1 Architecture Centralisée



Cette architecture permet l'inspection centralisée du trafic VPC-to-VPC et on-premises-to-VPC.

1.2 Configuration AWS Network Firewall

```
# Créer un firewall policy
aws network-firewall create-firewall-policy \
    --firewall-policy-name production-policy \
    --firewall-policy '{
      "StatelessDefaultActions": ["aws:forward_to_sfe"],
      "StatelessFragmentDefaultActions": ["aws:forward_to_sfe"],
      "StatefulRuleGroupReferences": [{
        "ResourceArn": "arn:aws:network-firewall:region:account:stateful-rulegroup/my-rules"
      }]
    }'

# Créer le firewall
aws network-firewall create-firewall \
    --firewall-name production-firewall \
    --firewall-policy-arn arn:aws:network-firewall:region:account:firewall-policy/production-p
    --vpc-id vpc-xxxxx \
    --subnet-mappings SubnetId=subnet-xxxxx
```

1.3 Règles Stateful (Suricata)

```
# Bloquer les connexions vers des domaines malveillants
drop tls any any -> any any (tls.sni; content:"malicious.com"; sid:1001;)

# Bloquer les tentatives SQL Injection
alert http any any -> any any (http.uri; content:"union select"; sid:1002;)
```

```
# Alerter sur exfiltration de données
alert tcp $HOME_NET any -> $EXTERNAL_NET any (msg:"Large Data Transfer"; flow:to_server; thres
```

2. AWS WAF (Web Application Firewall)

2.1 Protection API Gateway et ALB

```
# Créer un Web ACL
aws wafv2 create-web-acl \
    --name production-web-acl \
    --scope REGIONAL \
    --default-action Block={} \
    --rules file://waf-rules.json

# Associer à un ALB
aws wafv2 associate-web-acl \
    --web-acl-arn arn:aws:wafv2:region:account:regional/webacl/production-web-acl/xxxxx \
    --resource-arn arn:aws:elasticloadbalancing:region:account:loadbalancer/app/my-alb/xxxxx
```

2.2 Règles WAF Essentielles

```
{
  "Name": "BlockSQLInjection",
  "Priority": 1,
  "Statement": {
    "SqliMatchStatement": {
      "FieldToMatch": {
        "Body": {}
      },
      "TextTransformations": [{
        "Priority": 0,
        "Type": "URL_DECODE"
      }]
    }
  },
  "Action": {
    "Block": {}
  },
  "VisibilityConfig": {
    "SampledRequestsEnabled": true,
    "CloudWatchMetricsEnabled": true,
    "MetricName": "BlockSQLInjection"
  }
}
```

PrivateLink et VPC Endpoints

1. AWS PrivateLink

AWS PrivateLink permet une connectivité privée entre les VPCs, les services AWS et vos applications on-premises **sans exposer le trafic sur Internet**.

1.1 Types de VPC Endpoints

Type	Usage	Exemples
Interface Endpoints	Services AWS tiers	S3, DynamoDB, SQS, SNS, etc.
Gateway Endpoints	S3 et DynamoDB uniquement	Gratuit
Gateway Load Balancer Endpoints	Appliances de sécurité	Firewalls tiers

1.2 Configuration Interface VPC Endpoint

```
# Créer un VPC endpoint pour S3
aws ec2 create-vpc-endpoint \
    --vpc-id vpc-xxxxx \
    --service-name com.amazonaws.us-east-1.s3 \
    --route-table-ids rtb-xxxxx \
    --vpc-endpoint-type Interface \
    --subnet-ids subnet-xxxxx subnet-yyyyy \
    --security-group-ids sg-xxxxx
```

1.3 Endpoint Policy pour Sécurité

```
{
  "Statement": [{
    "Sid": "AllowOnlySpecificBuckets",
    "Effect": "Allow",
    "Principal": "*",
    "Action": [
      "s3:GetObject",
      "s3:PutObject"
    ],
    "Resource": [
      "arn:aws:s3:::my-production-bucket/*",
      "arn:aws:s3:::my-logs-bucket/*"
    ]
```

```
  }]
}
```

2. VPC Lattice (Nouveau 2025)

Amazon VPC Lattice permet une communication sécurisée entre applications dans des architectures distribuées modernes **sans configurations réseau complexes**.

```
# Créer un service network
aws vpc-lattice create-service-network \
    --name production-service-network \
    --auth-type AWS_IAM

# Associer un VPC
aws vpc-lattice create-service-network-vpc-association \
    --service-network-identifier sn-xxxxx \
    --vpc-identifier vpc-xxxxx \
    --security-group-ids sg-xxxxx
```

Architecture Multi-VPC

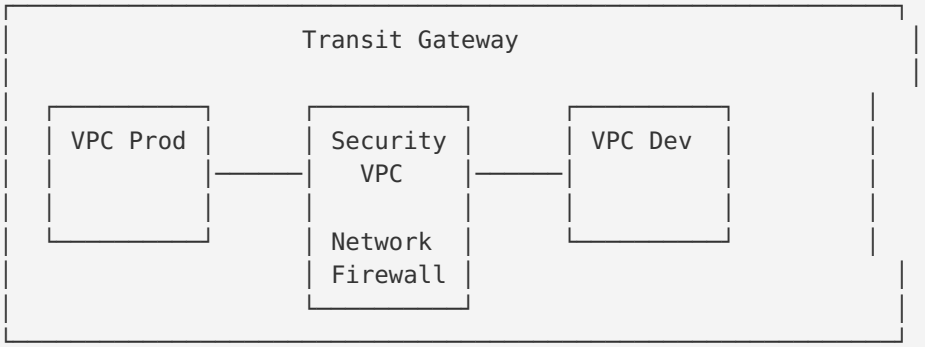
1. AWS Transit Gateway

Transit Gateway permet de connecter plusieurs VPCs et réseaux on-premises via un hub central.

```
# Créer un Transit Gateway
aws ec2 create-transit-gateway \
    --description "Production Transit Gateway" \
    --options AmazonSideAsn=64512,AutoAcceptSharedAttachments=enable,DefaultRouteTableAssociat

# Attacher un VPC
aws ec2 create-transit-gateway-vpc-attachment \
    --transit-gateway-id tgw-xxxxx \
    --vpc-id vpc-xxxxx \
    --subnet-ids subnet-xxxxx subnet-yyyyy

# Créer une route vers le TGW
aws ec2 create-route \
    --route-table-id rtb-xxxxx \
    --destination-cidr-block 10.0.0.0/8 \
    --transit-gateway-id tgw-xxxxx
```

2. Inspection Centralisée

Intégrer Network Firewall avec Transit Gateway dans un **Security VPC dédié**:



Checklist de Sécurité Réseau

✅ Niveau Critique (Priorité 1)

- [ ] **VPC Flow Logs activés sur tous les VPCs**
- [ ] **Sous-réseaux privés pour bases de données (aucune IP publique)**
- [ ] **Security Groups: aucun 0.0.0.0/0 sur ports 22, 3389, 3306, 5432**
- [ ] **Multi-AZ pour applications de production**
- [ ] **NACLs configurés avec règles DENY pour trafic malveillant**
- [ ] **VPC Endpoints pour services AWS (S3, DynamoDB, etc.)**
- [ ] **AWS WAF activé sur ALB et API Gateway**

✅ Niveau Important (Priorité 2)

- [ ] **Network Firewall déployé pour inspection du trafic**
- [ ] **CloudWatch Logs Insights configuré pour analyse de Flow Logs**
- [ ] **Security Groups référencés entre tiers (pas de CIDR blocks)**
- [ ] **Transit Gateway pour connectivité multi-VPC**
- [ ] **PrivateLink pour services tiers**
- [ ] **GuardDuty activé pour détection de menaces réseau**
- [ ] **VPN Site-to-Site ou Direct Connect pour connexion on-premises**

✅ Niveau Recommandé (Priorité 3)

- [ ] **Amazon Detective pour investigation de menaces**
- [ ] **VPC Lattice pour architectures multi-tenants**
- [ ] **Flow Logs vers S3 avec rétention > 90 jours**
- [ ] **CloudWatch Contributor Insights pour Top-N analysis**
- [ ] **Network ACL rules documentées et auditées trimestriellement**
- [ ] **Endpoint policies pour VPC Endpoints**
- [ ] **AWS Shield Standard activé (automatique et gratuit)**

Références et Ressources

Documentation Officielle AWS

- VPC Security Best Practices
- Building Scalable Multi-VPC Network Infrastructure
- AWS Network Firewall Best Practices
- VPC Flow Logs Guide

Outils et Services

- **AWS Network Firewall** - Inspection de trafic managée
- **AWS WAF** - Protection application web
- **VPC Flow Logs** - Audit du trafic réseau
- **AWS PrivateLink** - Connectivité privée
- **Amazon Detective** - Investigation de sécurité
- **GuardDuty** - Détection de menaces

Conclusion

Une architecture réseau AWS sécurisée repose sur:
- **Segmentation stricte** avec des sous-réseaux dédiés par tier
- **Defense-in-depth** avec Security Groups + NACLs + Network Firewall

- **Visibilité complète** via VPC Flow Logs et CloudWatch
- **Connectivité privée** avec PrivateLink et VPC Endpoints

L'implémentation de ces meilleures pratiques garantit une infrastructure réseau résiliente, conforme et sécurisée pour vos applications SaaS critiques.

Document préparé pour: [Nom du Client]
Contact support: [Email de l'équipe réseau]
Dernière mise à jour: Novembre 2025