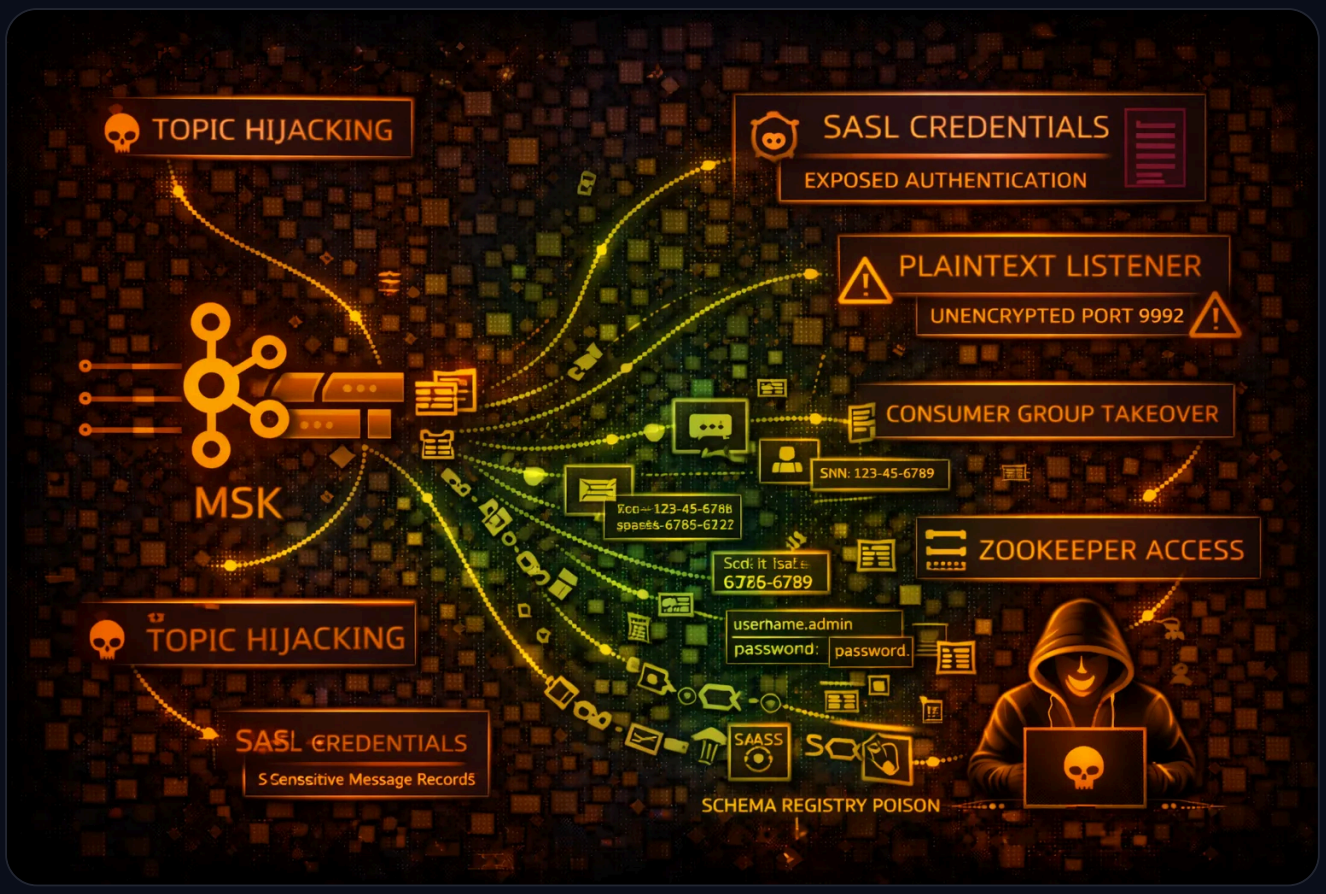

AWS MSK Security

STREAMING

Amazon Managed Streaming for Apache Kafka (MSK) provides fully managed Kafka clusters. Security risks include credential theft, topic hijacking, and real-time data interception.

HIGH	Regional	9092/9094	SASL/TLS
Risk Level	Scope	Ports	Auth

Service Overview

Kafka Clusters

Managed Kafka brokers with ZooKeeper or KRaft mode. Supports SASL/SCRAM, IAM, and mTLS authentication. Topics store real-time streaming data from producers to consumers.

Attack note: Plaintext listeners (9092) allow unauthenticated access. SASL credentials often stored in Secrets Manager.

MSK Connect & Serverless

MSK Connect runs Kafka Connect connectors. MSK Serverless auto-scales capacity. Both integrate with VPC, IAM, and can connect to external data sources.

Attack note: Connectors with overly permissive IAM roles can access S3, DynamoDB, and other AWS services

Security Risk Assessment

7.8

Risk Score

MSK clusters can expose real-time data streams including PII, financial transactions, and system events. Compromised consumers can intercept all messages. Producers can inject malicious data.

⚔️ Attack Vectors

Credential Theft

- SASL/SCRAM credentials from Secrets Manager
- IAM role credentials from metadata
- mTLS certificates from ACM/filesystem
- ZooKeeper credentials exposure
- Broker connection strings in config

Data Interception

- Consumer group hijacking
- Topic subscription enumeration
- Message replay attacks
- Partition leader impersonation
- Schema registry data theft

⚠️ Misconfigurations

Network Issues

- Plaintext listener enabled (9092)
- Public subnet deployment
- Security groups allowing 0.0.0.0/0
- VPC peering without restrictions
- Missing encryption in transit

Authentication Issues

- No authentication required
- Weak SASL/SCRAM passwords
- IAM authentication disabled
- mTLS not enforced
- Default ACLs too permissive

🔍 Enumeration

List Clusters

```
aws kafka list-clusters-v2
```

Describe Cluster

```
aws kafka describe-cluster-v2
\\
  --cluster-arn CLUSTER_ARN
```

Get Bootstrap Brokers

```
aws kafka get-bootstrap-brokers
\\
  --cluster-arn CLUSTER_ARN
```

List Topics (kafka-cli)

```
kafka-topics.sh --bootstrap-ser
ver BROKER:9092 \\
  --list
```

Describe Consumer Groups

```
kafka-consumer-groups.sh \\
  --bootstrap-server BROKER:909
2 --list
```

📈 Privilege Escalation

Kafka ACL Abuse

- Create new topics with admin ACLs
- Modify consumer group permissions
- Add new SASL users via ZooKeeper
- Escalate through MSK Connect roles
- Abuse cluster operation permissions

Escalation Paths

- Consumer → Topic data → Credentials in messages
- MSK Connect → S3/DynamoDB access
- ZooKeeper access → Cluster control
- Broker metrics → Infrastructure recon
- Schema Registry → Data structure exposure

Data Exposure

Message Data Risks

- PII in message payloads
- Credentials passed through topics
- Financial transaction data
- System events with secrets
- Unencrypted message content

Exfiltration Techniques

- Consumer from offset 0 (full history)
- Mirror maker to external cluster
- S3 sink connector redirection
- Kafka Connect to attacker endpoint
- Topic compaction log extraction

Detection

CloudTrail Events

- GetBootstrapBrokers - broker discovery
- CreateCluster - new cluster
- UpdateSecurity - security changes
- CreateConfiguration - config changes
- BatchAssociateScramSecret - cred changes

Indicators of Compromise

- New consumer groups from unknown IPs
- Unusual topic access patterns
- Failed authentication attempts
- ACL modifications
- Consumer lag anomalies

💻 Exploitation Commands

Consume All Messages (from beginning)

```
kafka-console-consumer.sh \\
  --bootstrap-server BROKER:9092 \\
  --topic TOPIC_NAME \\
  --from-beginning
```

Produce Malicious Message

```
kafka-console-producer.sh \\
  --bootstrap-server BROKER:9092 \\
  --topic TOPIC_NAME
```

List All Topics

```
kafka-topics.sh \\
  --bootstrap-server BROKER:9092 \\
  --list
```

Describe Topic (partitions/replicas)

```
kafka-topics.sh \\
  --bootstrap-server BROKER:9092 \\
  --describe --topic TOPIC_NAME
```

Reset Consumer Offset (replay)

```
kafka-consumer-groups.sh \\
  --bootstrap-server BROKER:9092 \\
  --group GROUP_ID \\
  --reset-offsets --to-earliest \\
  --topic TOPIC_NAME --execute
```

Get SASL Credentials

```
aws secretsmanager get-secret-value \\
  --secret-id AmazonMSK_CLUSTER_NAME
```

📜 Policy Examples

✕ Dangerous - Full Cluster Access

```
{
  "Effect": "Allow",
  "Action": "kafka:*",
  "Resource": "*"
}
```

Full MSK access including cluster management and all topics

✓ Secure - Specific Topic Only

```
{
  "Effect": "Allow",
  "Action": [
    "kafka-cluster:Connect",
    "kafka-cluster:ReadData"
  ],
  "Resource": [
    "arn:aws:kafka:*:*:cluster/CLUSTER/*",
    "arn:aws:kafka:*:*:topic/CLUSTER/*/TOPIC"
  ]
}
```

Only connect and read from specific topic

✕ Risky - All Topics Read/Write

```
{
  "Effect": "Allow",
  "Action": [
    "kafka-cluster:*Topic*",
    "kafka-cluster:ReadData",
    "kafka-cluster:WriteData"
  ],
  "Resource": "*"
}
```

Read/write to all topics - data exposure risk

✓ Secure - Producer Only

```
{
  "Effect": "Allow",
  "Action": [
    "kafka-cluster:Connect",
    "kafka-cluster:WriteData"
  ],
  "Resource": [
    "arn:aws:kafka:*:*:topic/CLUSTER/*/events-*"
  ]
}
```

Only produce to topics matching pattern

Defense Recommendations

Enable IAM Authentication

Use IAM authentication instead of SASL/SCRAM for better access control.

```
aws kafka update-security --authentication-info ClientBroker=TLS,Iam=ENABLED
```

Disable Plaintext Listener

Only allow TLS connections on port 9094. Disable port 9092.

Enforce mTLS

Require mutual TLS for client authentication with ACM certificates.

Enable Broker Logs

Send broker logs to CloudWatch for security monitoring and audit.

```
aws kafka update-monitoring --logging-info BrokerLogs={CloudWatchLogs={Enabled=true}}
```

Kafka ACLs

Implement fine-grained Kafka ACLs to restrict topic access per principal.

VPC Security Groups

Restrict inbound to specific CIDR ranges. No 0.0.0.0/0 access.