CHECKLIST

Kafka Streams Security Checklist

A practical, vendor-neutral checklist for securing Apache Kafka Streams pipelines across transport, identity, authorization, validation, secrets, and monitoring.

Kafka Streams Security Checklist

Use this checklist to review whether a Kafka Streams pipeline is secure enough for production. Mark each item as Done, Partial, or No.

---

1) Scope the pipeline

- [ ] List every Kafka Streams application in scope.
- [ ] Identify each application?s purpose and business owner.
- [ ] Identify all input topics, output topics, consumer groups, and external systems touched by the application.
- [ ] Confirm whether the pipeline processes sensitive, regulated, or internal-only data.
- [ ] Document trust boundaries between producers, stream processors, brokers, sinks, and downstream systems.

Notes: _____

---

2) Secure transport

- [ ] Enforce encrypted client-to-broker connections.

- [ ] Ensure producer, consumer, and Kafka Streams client traffic cannot be read in transit.
- [ ] Verify certificate or key rotation is supported and tested.
- [ ] Confirm client trust stores or equivalent trust material are managed centrally.
- [ ] Remove or document any exceptions for plaintext traffic.

Pass criteria: No credentials or event payloads traverse the network in cleartext.

---

3) Authenticate each service identity

- [ ] Assign a unique identity to each Kafka Streams application.
- [ ] Avoid shared credentials across teams, environments, or applications.
- [ ] Ensure identities are traceable to a specific service and owner.
- [ ] Confirm unused identities and credentials can be revoked quickly.
- [ ] Verify authentication settings match the broker and platform capabilities in your environment.

Pass criteria: Every stream processor has a unique, revocable, auditable identity.

---

4) Apply least-privilege authorization

- [ ] Grant read access only to required input topics.
- [ ] Grant write access only to required output topics.
- [ ] Restrict consumer group access to only the groups used by the application.
- [ ] Avoid broad cluster-wide privileges unless they are explicitly required.
- [ ] Restrict access to any external stores, caches, APIs, or databases used by the processor.
- [ ] Review permissions for service accounts in both non-production and production.

Pass criteria: The application cannot read, write, or administer anything outside its intended function.

---

5) Validate incoming records

- [ ] Validate required fields before business logic runs.
- [ ] Validate schema, data types, and field constraints.
- [ ] Enforce reasonable message size limits.
- [ ] Reject malformed, unexpected, or incomplete records.
- [ ] Fail closed when the record cannot be interpreted safely.
- [ ] Review behavior when a topic receives mixed producers or changing schemas.

Pass criteria: Invalid records are rejected safely and do not trigger unsafe defaults or partial processing.

---

6) Protect secrets and sensitive data

- [ ] Do not store secrets in source code.
- [ ] Do not store secrets in plain environment variables unless your platform explicitly secures them.
- [ ] Use a managed secret store or equivalent protected mechanism.
- [ ] Limit secret access to only the application that needs it.
- [ ] Rotate credentials on a defined schedule.
- [ ] Remove secrets from logs, debug output, and error messages.
- [ ] Ensure payload fragments containing sensitive data are redacted before logging.

Pass criteria: Secrets are centrally managed, minimally exposed, and never printed in logs.

---

7) Review logging, metrics, and telemetry

- [ ] Confirm logs do not contain raw sensitive payloads.
- [ ] Confirm metrics do not expose secrets, tokens, or personal data.
- [ ] Redact or hash sensitive identifiers where appropriate.
- [ ] Capture authentication failures, authorization failures, and schema validation failures.
- [ ] Alert on unusual retry patterns, dead-letter growth, and repeated poison-message handling.
- [ ] Ensure telemetry retention matches data sensitivity requirements.

Pass criteria: Operational visibility exists without leaking sensitive information.

---

8) Check failure handling and replay behavior

- [ ] Verify failed records cannot be replayed in a way that bypasses controls.
- [ ] Review offset management to ensure it does not expose or duplicate sensitive events.
- [ ] Confirm retries are bounded and do not create infinite processing loops.
- [ ] Validate dead-letter or quarantine paths are restricted and monitored.
- [ ] Ensure error handling does not log sensitive payload contents.

Pass criteria: Failures are contained, observable, and safe to recover.

---

9) Test revocation and incident response

- [ ] Revoke a test credential and confirm the application loses access as expected.
- [ ] Confirm the pipeline fails safely when authorization is removed.
- [ ] Verify alerts trigger for unauthorized access attempts.
- [ ] Confirm an incident response path exists for compromised service identities.
- [ ] Document who can disable the app, revoke credentials, and rotate secrets.

Pass criteria: Access can be removed quickly without affecting unrelated workloads.

---

10) Validate deployment readiness

- [ ] Confirm infrastructure, broker, and application configurations are aligned.
- [ ] Review topic permissions before production rollout.
- [ ] Verify certificates, secrets, and trust material are current.
- [ ] Confirm schema and validation rules were tested with malformed inputs.
- [ ] Confirm monitoring dashboards and alerts are enabled.
- [ ] Confirm the application owner approved the security review.

Pass criteria: Security controls are verified before production cutover.

Quick decision check

If you answer No to any of the following, the pipeline is not ready for production:

- [ ] Is traffic encrypted end to end?
- [ ] Does each stream processor have its own identity?
- [ ] Are topic permissions limited to only what is required?
- [ ] Are records validated before processing?
- [ ] Are secrets protected and redacted from logs?
- [ ] Can you detect misuse, failure, and revocation cleanly?

Final review notes

- Risks found: _____
- Remediations required: _____
- Owner: _____
- Review date: _____
- Go-live status: _____

Production sign-off

- [ ] Security review completed
- [ ] Required fixes implemented
- [ ] Monitoring and alerting enabled
- [ ] Access revocation tested
- [ ] Production rollout approved

Sign-off by: _____