CHECKLIST

SSL Offload for Secure Virtual Apps Access Checklist

A practical vendor-neutral checklist for validating SSL offload in secure virtual apps environments, including certificate trust, internal hop protection, authentication flow, and production readiness checks.

SSL Offload for Secure Virtual Apps Access

Production Readiness Checklist

Use this checklist to evaluate whether SSL offload is appropriate for your virtual apps access design and to validate the implementation before production.

---

1) Confirm the use case

- [ ] The access pattern is clearly defined:
- [ ] Internet-facing
- [ ] Partner-facing
- [ ] Internal-only
- [ ] The reason for SSL offload is documented:
- [ ] Reduced encryption overhead
- [ ] Simplified certificate management
- [ ] Centralized access-tier policy control

- [ ] The change is aligned with security and compliance requirements.
- [ ] The application stack can tolerate TLS termination at the edge.

---

2) Define where TLS ends

- [ ] The termination point is explicitly documented.
- [ ] It is clear which component presents the certificate to clients.
- [ ] The internal hop is classified as one of the following:
- [ ] Plain HTTP
- [ ] Re-encrypted / split termination
- [ ] Other: _____
- [ ] The trust boundary between the access tier and back-end services is documented.
- [ ] Internal segmentation supports the chosen design.

---

3) Validate certificate and trust settings

- [ ] The certificate common name or SAN matches the external access name.
- [ ] The full certificate chain is installed and presented correctly.
- [ ] The issuing authority is trusted by all supported client types.
- [ ] Renewal ownership is assigned.
- [ ] Key storage and access controls are defined.
- [ ] Certificate expiration monitoring is in place.
- [ ] Protocol and cipher policy meet baseline requirements.
- [ ] Legacy protocols and weak ciphers are disabled if required.

---

4) Check the internal hop

If the back-end path is unencrypted:

- [ ] The unencrypted segment is limited to a trusted network zone.
- [ ] Firewall rules restrict access to the expected source and destination only.
- [ ] Routes are controlled and documented.
- [ ] The risk of lateral movement across the internal segment is assessed.

If the back-end path remains encrypted:

- [ ] The back-end certificate chain is trusted.
- [ ] The internal TLS configuration has been tested separately.
- [ ] Any inspection or interception controls between tiers are known and approved.
- [ ] The extra CPU and certificate dependencies are acceptable.

---

5) Verify headers, cookies, and redirects

- [ ] Host headers are preserved or rewritten as required by the app stack.
- [ ] Redirects resolve correctly after TLS termination.
- [ ] Session cookies survive the access-tier transition.
- [ ] No redirect loop occurs during login.
- [ ] The external URL used by clients matches application expectations.
- [ ] Any app-specific header requirements are documented and tested.

---

6) Test authentication and session behavior

- [ ] Primary authentication works through the access tier.

- [ ] Secondary authentication or MFA works if enabled.
- [ ] User sessions launch successfully after sign-in.
- [ ] Session persistence behavior is validated.
- [ ] Timeout and reauthentication behavior are expected.
- [ ] Edge cases have been tested:
- [ ] New user
- [ ] Returning user
- [ ] Expired session
- [ ] Failed authentication
- [ ] Concurrent session attempts

---

7) Validate health checks and availability monitoring

- [ ] Health probes reach the correct endpoint.
- [ ] Health checks reflect real service readiness, not only port availability.
- [ ] Failover behavior has been tested.
- [ ] Load balancer or gateway health logic does not break app launch flows.
- [ ] Monitoring alerts exist for:
- [ ] TLS handshake failures
- [ ] Certificate expiration
- [ ] Back-end health failures
- [ ] Authentication errors
- [ ] Redirect or session failures

---

8) Confirm logging and observability

- [ ] Logs show client-facing TLS session details.
- [ ] Logs show back-end connection status.
- [ ] Authentication outcomes are recorded.
- [ ] Events can be correlated across access tier and back-end services.
- [ ] Time synchronization is enabled across systems.
- [ ] Log retention supports troubleshooting and audit needs.

---

9) Review operational ownership

- [ ] Access tier ownership is assigned.
- [ ] Back-end application ownership is assigned.
- [ ] Certificate management responsibility is assigned.
- [ ] Incident response escalation paths are documented.
- [ ] Change management approval is complete.
- [ ] Rollback steps are documented and tested.

---

10) Perform pre-production validation

- [ ] Test with representative client types and browsers.
- [ ] Test from the same network path expected in production.
- [ ] Verify app launch under peak or near-peak load.
- [ ] Validate login, launch, reconnect, and logout flows.
- [ ] Confirm no hidden dependency fails after TLS termination.
- [ ] Verify the production configuration matches the tested configuration.

Decision check

SSL offload is ready for production only if all of the following are true:

- [ ] TLS termination point is clearly defined.
- [ ] The internal hop is protected appropriately.
- [ ] Certificate trust and renewal are operationally owned.
- [ ] Authentication and session launch work end to end.
- [ ] Health checks and logging are reliable.
- [ ] The design meets security and compliance requirements.

If any item is unresolved, pause deployment and remediate before go-live.

Quick risk signals

Be cautious if you find any of the following:

- [ ] Certificate renewals depend on informal coordination.
- [ ] The internal network is flat or poorly segmented.
- [ ] Redirects behave differently after termination.
- [ ] Session cookies or host headers change unexpectedly.
- [ ] Health checks report green, but users cannot launch apps.
- [ ] You cannot explain where TLS ends and why.

Sign-off

Environment: _____ Owner: _____ Date: _____ Approved for production: Yes / No

Notes: