Security & Architecture FAQ

Zero-Infrastructure Design

NexGen Architects | July 2026

Zero-infrastructure design, explained for security reviewers, architects, and procurement. The short version: your data travels directly from your Salesforce org to your ServiceNow instance. NexGen Architects operates no servers in that path, receives no copy of your data, and holds none of your credentials.

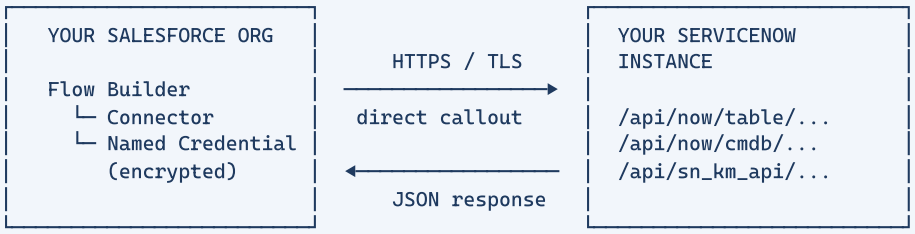
SECURITY POSTURE AT A GLANCE

Data path	Your Salesforce org → HTTPS → your ServiceNow instance. Two parties, no intermediary.
Vendor servers in path	None. NexGen Architects hosts no service, proxy, gateway, or database used by this connector at runtime.
Data at rest outside your tenants	None. The connector stores no records, no logs, and no cached payloads outside Salesforce.
Credential storage	Salesforce's encrypted Named Credential store. The connector never handles raw secrets.
Transport	HTTPS/TLS on every call, enforced by the Salesforce platform for all outbound callouts.
Access control	Enforced twice — Salesforce permission sets on the credential, ServiceNow ACLs and roles on the data.
Telemetry to vendor	None. The connector phones home to nothing.
Distribution	Salesforce managed package via AppExchange, which requires passing Salesforce's security review.

What "zero-infrastructure" actually means

The term is used loosely in the integration market, so here is the precise claim. Many "connectors" are in fact hosted middleware: your data passes through a vendor-operated tenant, which terminates TLS, sees your payloads, and stores at least metadata. That vendor becomes a third party to every record you sync, an additional processor under GDPR, and a new item on your security questionnaire.

The ServiceNow Flow Connector is not that. It is a Salesforce managed package: a set of declarative connector definitions that install into your org and execute inside your Salesforce runtime. When a flow calls `createIncident`, the resulting HTTPS request originates from Salesforce infrastructure and terminates at your ServiceNow instance. There is no hop in between.



The consequences of that design are the substance of this document: no vendor data processor to assess, no vendor uptime to depend on, no vendor breach that can expose your records, and no vendor to notify under a data-processing agreement — because we never hold your data in the first place.

How credentials are protected

The connector never handles your ServiceNow credentials directly. It declares that an operation requires a callout; Salesforce's Named Credential subsystem attaches the authorization header at call time. The secret is held in Salesforce's encrypted credential store and is never exposed to the connector's logic, to your flow definitions, or to flow debug output.

This has a specific, testable consequence: you cannot leak a ServiceNow password by exporting a flow. The flow references a Named Credential by name; the secret lives elsewhere. A flow definition committed to a Git repository or handed to a consultant carries no secret material.

The two access-control boundaries

Every call the connector makes is checked twice, in two different systems. Understanding both is the key to a defensible configuration.

Two independent gates. Both must permit a call for it to succeed.

Boundary	Enforced by	What it controls
Who in Salesforce may call ServiceNow at all	Salesforce permission sets — specifically External Credential Principal Access	A user or automated process that has not been granted access to the Principal cannot make a callout, regardless of the flow they run. This is your gate on *who can reach ServiceNow*.
What may be read or written once there	ServiceNow roles and ACLs, enforced server-side	The integration user's roles determine which tables, records, and fields are reachable. The connector cannot override or bypass this — ACL evaluation happens inside ServiceNow, after the request arrives. This is your gate on *what can be touched*.

Auditability

Because there is no middleware, there is also no vendor log to request. Everything is auditable in the two systems you already own:

- **In Salesforce** — flow interviews and debug logs record which flow ran, when, and with what inputs. Event Monitoring, where licensed, captures callout activity. Setup Audit Trail records changes to Named Credentials and permission sets.
- **In ServiceNow** — every record the connector creates or updates carries standard audit fields (`sys_created_by`, `sys_updated_by`, `sys_updated_on`), all attributed to the integration user. The audit history on each record shows exactly what changed. For OAuth, the token registry shows active tokens and permits immediate revocation.

This is precisely why the integration user should be dedicated and clearly named — `salesforce.integration` rather than a person's account. It makes every action the connector took trivially attributable in ServiceNow's own audit history.

Incident response: how to cut access immediately

If you need to sever the integration right now, you have three independent controls — and you do not need to contact NexGen Architects to use any of them:

1. **Revoke in ServiceNow (fastest, most complete).** Deactivate the integration user, or revoke its tokens in the OAuth token registry. All calls fail immediately, regardless of Salesforce state.
2. **Revoke in Salesforce.** Remove External Credential Principal Access from the permission set, or delete the Principal. No flow can make a callout.
3. **Disable at the flow level.** Deactivate the specific flows that call ServiceNow — the narrowest option, useful when only one integration is misbehaving.

Because we hold no credentials and operate no service, there is no fourth step involving us.

Frequently asked questions

> **Does the ServiceNow Flow Connector send my data to NexGen Architects or any third party?**
>
> No. Your data never reaches NexGen Architects or any third party. The connector is a Salesforce managed package that runs inside your own Salesforce org and calls your ServiceNow instance directly over HTTPS. NexGen Architects operates no server, proxy, gateway, or database in the runtime data path, receives no copy of your records, and collects no usage telemetry from the connector. The only two parties to any request are your Salesforce org and your ServiceNow instance.

What infrastructure do I need to run the ServiceNow Flow Connector?

None. That is the point of the zero-infrastructure design. There is no middleware to provision, no MuleSoft runtime to license, no integration server to patch, no container to host, and no message broker to operate. The connector installs as a managed package into your existing Salesforce org and uses the Salesforce platform's own callout mechanism. Your total infrastructure footprint for this integration is the Salesforce org and the ServiceNow instance you already have.

Where are my ServiceNow credentials stored, and can the connector see them?

Credentials live in Salesforce's encrypted Named Credential store, and the connector never handles them in raw form. The connector declares that an operation requires an authenticated callout; the Salesforce platform injects the authorization header at call time. As a result, secrets are not visible in flow definitions, not written to flow debug logs, and not retrievable through the connector. A practical consequence worth noting: you cannot leak a ServiceNow password by exporting or sharing a flow, because the flow contains only a reference to the Named Credential, not the secret itself.

Is NexGen Architects a data processor under GDPR for data moved by this connector?

No, because no data flows to us — there is nothing for us to process. The connector moves data directly between two systems you control: your Salesforce org and your ServiceNow instance. NexGen Architects has no access to that data at any point, holds no copy, and operates no infrastructure it passes through. Your GDPR analysis for this integration therefore concerns your existing relationships with Salesforce and ServiceNow, and does not add us as a further processor. Customers should still perform their own assessment; we are happy to support it — contact hello@nexgenarchitects.com.

Where is my data processed, and does it leave my region?

Your data stays within the Salesforce and ServiceNow regions you have already chosen — the connector introduces no additional region or hop. Because traffic goes directly from your Salesforce org to your ServiceNow instance, data residency is determined entirely by where those two tenants are hosted. A customer whose Salesforce org and ServiceNow instance are both hosted in the EU or UK sees no data leave that boundary as a result of using this connector. There is no vendor region to add to your data map.

Is the traffic between Salesforce and ServiceNow encrypted?

Yes. Every call is made over HTTPS with TLS, enforced by the Salesforce platform for all outbound callouts. The connector cannot make a plaintext HTTP call: the Named Credential points at an `https://` endpoint and the platform performs standard certificate validation against your ServiceNow instance. Salesforce maintains the supported TLS versions and cipher suites as part of the platform, so this stays current without any action from you or from us.

Can the connector bypass ServiceNow permissions or access data the integration user cannot see?

No. ServiceNow enforces its Access Control Lists server-side on every REST call, and the connector has no mechanism to override them. The connector can only ever see and change what the integration user's roles permit. This makes the integration user's role assignment your true security boundary — grant the narrowest set that satisfies your use cases, and never grant `admin` for convenience. One consequence to be aware of: when ACLs deny a read, ServiceNow commonly returns HTTP 200 with an empty `result` array rather than a 403, so an under-privileged integration user presents as "no records found," not as an error.

What happens to my integration if NexGen Architects goes out of business?

Your integration keeps running, because nothing in the runtime path depends on us. The managed package is installed in your org, and the calls it makes go directly to your ServiceNow instance. There is no vendor service to shut down, no licence server to check in with, and no API key to expire. You would lose future updates and our support, but existing flows would continue to work. This is a structural property of the zero-infrastructure design, not a promise — it follows from there being no NexGen component in the data path.

How do I audit what the connector has done?

Entirely within the two systems you already own — there is no vendor log to request. In Salesforce, flow interviews, debug logs, Event Monitoring (where licensed), and the Setup Audit Trail record what ran and what changed. In ServiceNow, every record the connector creates or updates carries standard audit fields (`sys_created_by` , `sys_updated_by` , `sys_updated_on`) attributed to the integration user, and each record's audit history shows exactly what changed. Using a dedicated, clearly-named integration account makes every action the connector took immediately attributable.

How do I immediately revoke the connector's access to ServiceNow?

Deactivate the integration user in ServiceNow, or revoke its tokens in ServiceNow's OAuth token registry — this cuts access instantly and completely. You have two further independent controls: remove External Credential Principal Access from the Salesforce permission set, which stops any flow from making a callout; or deactivate the specific flows involved, which is the narrowest option. You do not need to contact NexGen Architects to cut access, because we hold no credentials and operate no service in the path.

Has the connector passed the Salesforce security review?

Yes — it is distributed as a managed package through AppExchange, and Salesforce requires every AppExchange listing to pass its security review before publication. Managed-package distribution also means the package's internals are not modifiable in your org, so its behaviour cannot be altered in place. Beyond the platform review, the design deliberately minimises what there is to review: the connector stores no data, holds no credentials, and operates no external endpoint.

Does the connector open any inbound port or endpoint into my Salesforce org?

No. The connector is strictly outbound: Salesforce calls ServiceNow, never the reverse. It exposes no inbound endpoint, no webhook receiver, and no public URL, so it adds no new inbound attack surface to your Salesforce org. This is also why ServiceNow cannot trigger a Salesforce flow through the connector — to react to ServiceNow-side changes, poll with a Scheduled Flow, or build a push from ServiceNow to a Salesforce inbound endpoint separately.