



ServiceNow eBonding: The Complete 2023 Guide

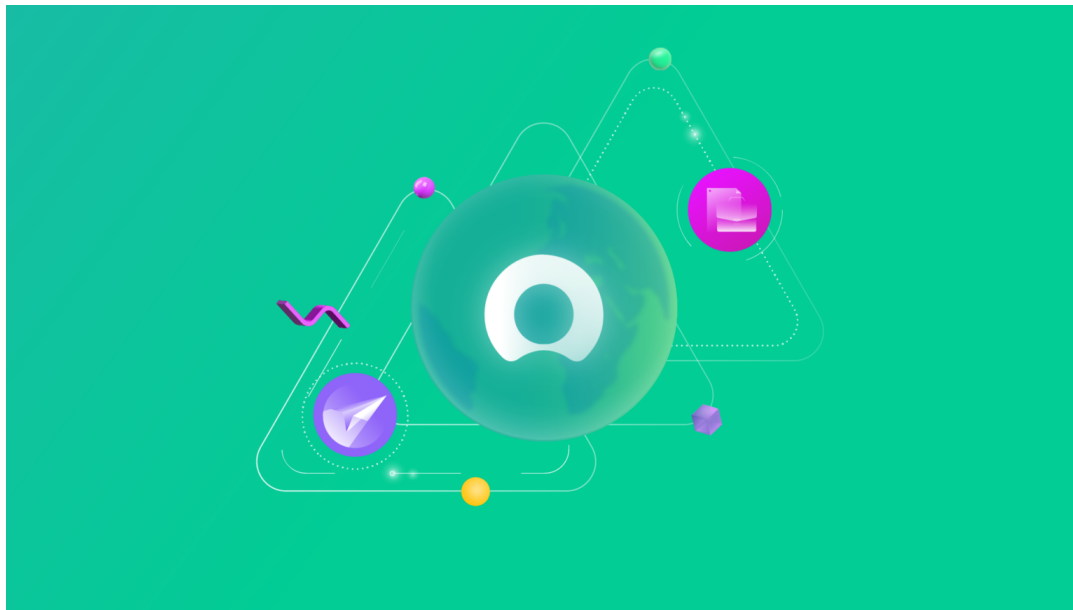


Table of contents

What is eBonding?

- Why eBonding?
- Benefits of eBonding

ServiceNow eBonding Spoke: Introduction

- What is ServiceNow?
- What is IntegrationHub?
- eBonding Spoke of IntegrationHub

ServiceNow eBonding Spoke Use Cases

- Use Case 1: MSPs
- Use Case 2: ServiceNow to ServiceNow

How to eBond 2 ServiceNow Instances Using eBonding Spoke

- Step 1: Request the ServiceNow Personal Developer Instances
- Step 2: Create Credentials in the Source Instance
- Step 3: Create Connections in the Source Instance
- Step 4: Design the Flow
- Step 5: Testing the eBonding Flow



ServiceNow eBonding Spoke Alternative

- Bottomline

Conclusion



ServiceNow is a popular ITSM platform that helps many companies deliver a world-class IT service management experience. A lot of ServiceNow users also want to collaborate with other enterprises working as their partners, managed service providers, other ITSM organizations or customers for their day-to-day operations.

eBonding can help these different applications residing in different organizations interact and exchange information securely. Also, if there are two ServiceNow instances that need to be synced, then it is logical to discuss the ServiceNow eBonding Spoke.

In this context, we'll discuss the eBonding Spoke in detail, enlisting its use and benefits and also have a look at a few of its use cases. Then see how it's implemented and also have a brief overview of other players in the eBonding market.

So let's get this show on the road!

What's covered in this article:

- [What is eBonding?](#)
- [ServiceNow eBonding Spoke: Introduction](#)
- [eBonding Spoke Use Cases](#)
- [How to eBond 2 ServiceNow Instances Using eBonding Spoke](#)
- [ServiceNow eBonding Spoke Alternative](#)

What is eBonding?

In today's data-driven world, the importance of having consistent, coherent and accessible data cannot be stressed enough. People all over the world use different applications or tools to manage their data for various purposes like service desk, project management, software development etc.



So when these people sitting across completely different applications want to share data that might be useful to the other team, they have to find ways for doing it automatically. Because if they resort to manual ways of phone calls, emails, or chats, the data passed around can get duplicated, altered, or lost. It can even lead to frustrating teams, increased friction, and negative customer experience.

Before we get to the actual definition of eBonding, let's focus on why teams would really benefit from such information exchange.

Why eBonding?

Say your customer's ticketing system is using an application to keep track of requests, reply to, and resolve tickets within their SLAs. But not all tickets can be closed by the customer support agents, in which case they need, maybe their managed services provider (MSP) to resolve those tickets.



So they pass the ticket information to the MSP. The MSP then starts working on the ticket. The entire time they are working on the ticket, the customer support agent is unaware of what needs to be communicated to the customer. Once the ticket is resolved, the MSP again manually passes over the information required for the closure of the ticket over to the customer support agent.

As clearly seen, information is indeed exchanged, but manually. And as common sense prevails, manual mistakes are bound to happen. A bit of research from hereon will open the doors to [eBonding](#)!

eBonding, also called bridging, makes it possible to exchange data between multiple applications in a way that makes it appear local to all the systems. It is a way of bi-directionally synchronizing data between unique companies and their systems. The data is sent across automatically, in turn streamlining workflows.

It can also be termed as a [B2B software integration](#) interface.

So is eBonding the right solution for your problems?

Yes, if you and your team need access to the same data then you can definitely give eBonding a thought.

But it's important to understand here that eBonding does not simply mean bi-directionally synchronizing information between 2 applications. It is much more than that. It requires careful planning, allocation of resources, right tools, systems and processes in place, so the eBonding solution can work as intended. If implemented in a jiffy, without the right processes it might lead to an inelegant and ineffective solution.

Benefits of eBonding

Perhaps the most important benefit of eBonding is that it reduces manual interventions and supports automated information exchange. So why is this beneficial?

To start with you can avoid all manual data entry mistakes and bid goodbye to redundant, misplaced or wrong data arising because of them. Be relieved that the information exchanged is accurate and consistent, and you can view it in the application you are already familiar with.



This will in turn help your teams to be on the same page, increase transparency between them, automate software-based business processes, and help them work on things that really matter, instead of using them for menial data-exchange tasks. Such positive transformations between teams can help them collaborate better, thus increasing productivity and efficiency.

With eBonding in place, you can choose to filter out what information you want to share with the other team and how you want to receive the information. This prevents unauthorized access to shared information and helps maintain privacy and confidentiality.

Automatic ways of doing things, might involve money upfront to invest in tools helping you achieve it, but in the long run, the money that you invested will start giving you good returns.

ServiceNow eBonding Spoke: Introduction

What is ServiceNow?

For those of you who aren't familiar with ServiceNow, it's a popular ITSM solution that helps companies manage digital workflows for their enterprise operations.



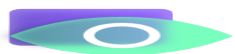
eBonding 2 ServiceNow instances (using an eBonding spoke) or a ServiceNow instance with other applications (using IntegrationHub) within the same company or with other companies acting as customers, suppliers, vendors or partners has been popular for some time now.

With such an implementation in place, data/ messages between 2 systems can be exchanged securely. These messages have a predefined format so that each end of the communication can understand and make sense out of them.

What is IntegrationHub?

IntegrationHub is ServiceNow's capability to automate integration tasks using its own Flow Designer. It also supports custom integrations and requires a separate subscription.

IntegrationHub executes 3rd party-APIs as a part of a flow when a specific event (like creating or updating an incident) occurs in ServiceNow. This kind of integration is executed with the help of *Spokes* that are easy to configure and require no coding. There are different spokes to help integrate ServiceNow with different applications like: Slack, Jira, Zoom, GitHub, GitLab, Salesforce, etc.



The integration patterns supported through these spokes have limited capability in terms of the use cases supported, and in case there is a need for a certain customization (not included in the default flow) then a request to the creator of the Spoke needs to be made.

eBonding Spoke of IntegrationHub

Let's also have a look at what eBonding Spoke is.

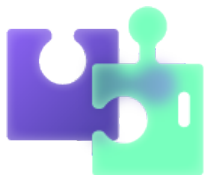
As mentioned above, IntegrationHub has various spokes to connect ServiceNow with different applications. Out of this, one is the eBonding Spoke that allows synchronization of incidents across 2 ServiceNow instances. Common integration design patterns are used by this Spoke serving a common use case: to synchronize incidents across ServiceNow instances .

The best thing about this spoke is that you do not require an IntegrationHub or Orchestration subscription. It is included by default (OOB capability: Out-of-the-box) in your ServiceNow instance, so you should be able to eBond incidents across multiple instances.

Here's how eBonding achieves this:

Basic eBonding spoke actions

Imagine you have 2 production ServiceNow instances. Let's call one as the **source system**; this is to manage external customer-facing operations. The other is the **target system**; to manage your internal operations.



The need here is to connect these 2 instances such that both have identical incidents and it's referential data. Suppose an incident originates at the source system, then a matching incident must be created on the target ServiceNow instance as well. Updates and changes to the respective incidents are then tracked and reflected on both sides.

For such a use case, ServiceNow's eBonding Spoke supports the following OOB actions.

- Create remote incident action
- Lookup remote incident action
- Update remote incident action

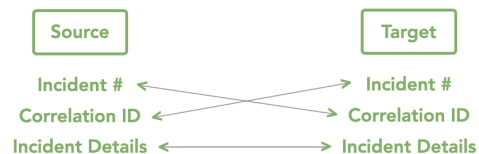
These actions are pretty self explanatory, but let us discuss them in brief.

Create remote incident action

According to the use case we have discussed above, the incident originates in the source system and needs to have an identical incident on the target system. For this, the target incident is created from the details of the source incident.

So the source incident number is passed as the Correlation ID on the target incident. Once this is done, the target incident number is updated as the Correlation ID on the source incident. This helps both instances to correlate with each other through their incident numbers.

This action is for when a new incident needs to be created.



The next 2 Actions are used for: when an existing incident is eBonded and some actions like updates to an incident or simply fetching the incident details need to be done.

Lookup remote incident action

This action, like its name suggests, is used to look up the remote incident details like the short description, description, priority etc. So the remote incident's details are fetched with the help of the Correlation ID and provided to the instance requesting them.

Update remote incident action

This action is used when we update the source incident, and those updates need to be reflected on the target incident. So the Correlation ID is fetched from the source incident and the remote incident is looked up. Then the details on the target incident are updated from the source incident.

Having based our background on understanding what ServiceNow has to offer in terms of integration capabilities and also understanding the basic actions that eBonding Spoke supports, let us also have a look at a few use cases where such capabilities can help you.

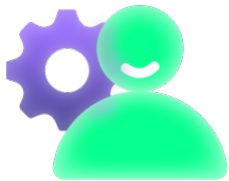
ServiceNow eBonding Spoke Use Cases

ServiceNow as we have already seen can be integrated with other applications. But since we are on the topic of eBonding right now, let's see how bonding 2 ServiceNow instances can help us.

Use Case 1: MSPs

Perhaps a use case you can already envision is an MSP that has its own ServiceNow and wants to integrate with its customer's ServiceNow instance.

Customer tickets (aka incidents) are often dealt with by support agents sitting at the customer site on a tight SLA. They usually try to resolve such incidents themselves. But suppose they come to know that the incident is something that would require intervention from the development team, working as the MSP here.



In this case, eBonding the incident on the ServiceNow of the MSP would help the support agents to avoid manual duplication and stay in the loop regarding the current status of the incident. All this can be done with each team viewing it in their own instance.

Use Case 2: ServiceNow to ServiceNow

It is not uncommon to eBond different ticketing systems of your suppliers, contractual partners, vendors or customers, all having their own ITSM systems, maybe ServiceNow itself. This means that information must be passed between these internal or external ServiceNow instances.

An eBonding integration here can help these unique companies to exchange ticketing information with each other that can help them avoid the manual way of creating duplicate incidents in each other's ServiceNow instances. This can help increase visibility, help the

right people get access to the right information, at the right time and consolidate a data store for better reporting and insights.



Having seen how an eBonding integration can help you avoid context switching between different ServiceNow instances, let us buckle down to see how an eBonding spoke can be implemented.

How to eBond 2 ServiceNow Instances Using eBonding Spoke

As we have discussed above, the eBonding Spoke uses Correlation ID to relate both source and target incidents. So the source incident number is the Correlation ID on the target instance and vice versa.

Step 1: Request the ServiceNow Personal Developer Instances

Probably the first and the most obvious thing you would need to do is [request](#) for a ServiceNow personal developer instance. It's super easy and quick.

But since eBonding spoke works with 2 ServiceNow instances, you need to have another one.

Once you have both of them in place, start by enabling the ServiceNow IntegrationHub Installer plugin. In case you don't know how to do it, please follow the instructions on this [page](#).

Step 2: Create Credentials in the Source Instance

For this implementation I have the following 2 ServiceNow instances:

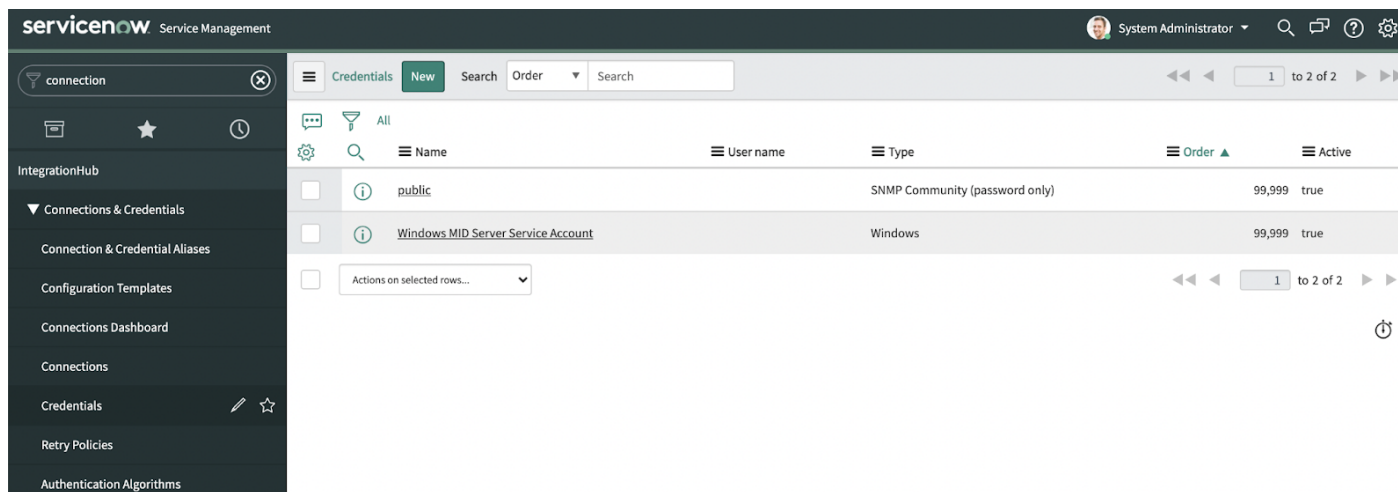
Source instance: <https://dev79664.service-now.com/>

Destination instance: <https://dev21491.service-now.com/>

Note: You will need the admin user ID and password for the destination instance, i.e dev21491.

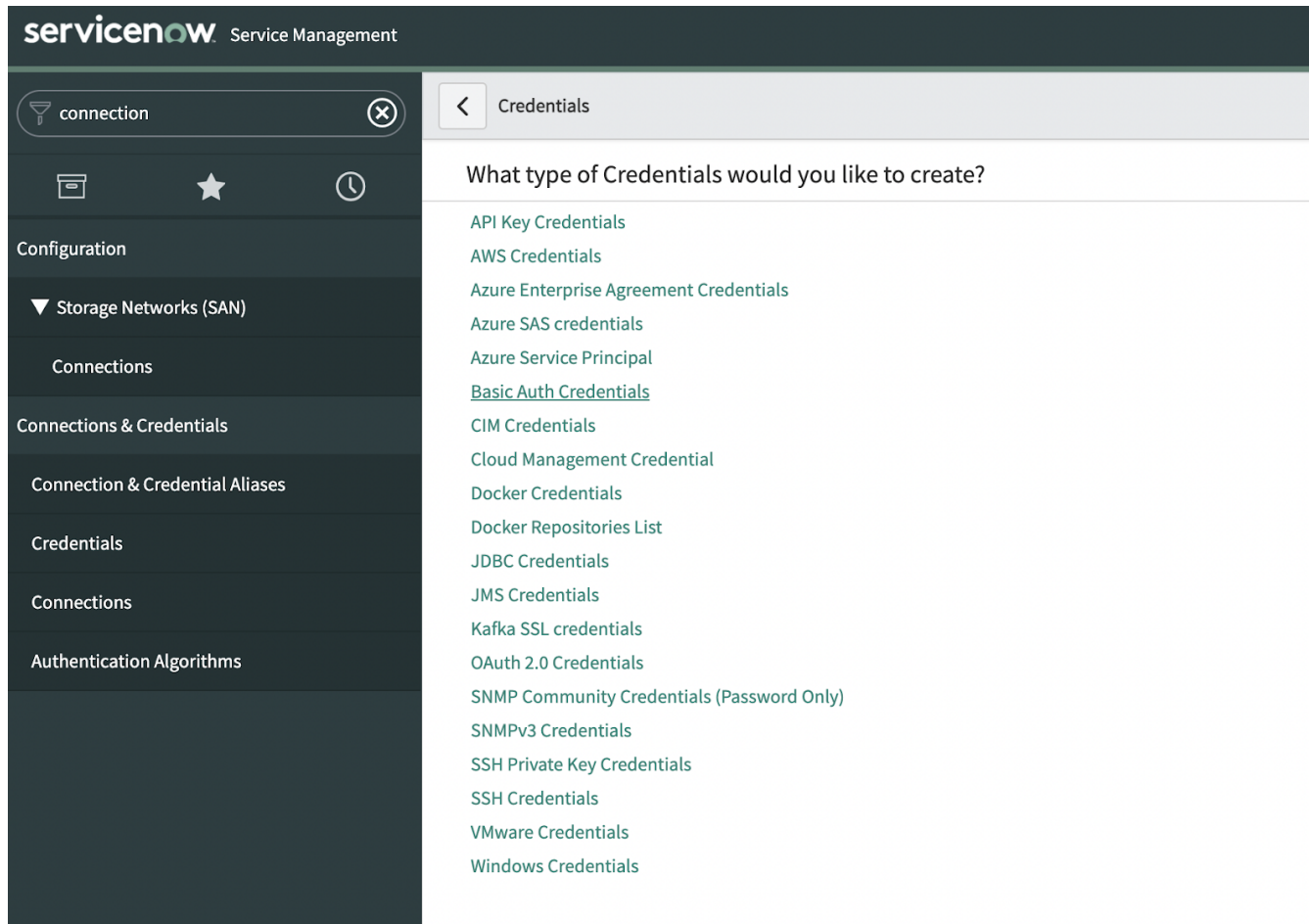
On the source ServiceNow instance (dev79664), navigate to “IntegrationHub” > “Connections & Credentials”. For this, you can simply type the text in the left-hand search bar.

Then click “Credentials” in the left-hand menu.



Click on the green “New” button in the top-center of the screen.

From the list, select “Basic Auth Credentials”.

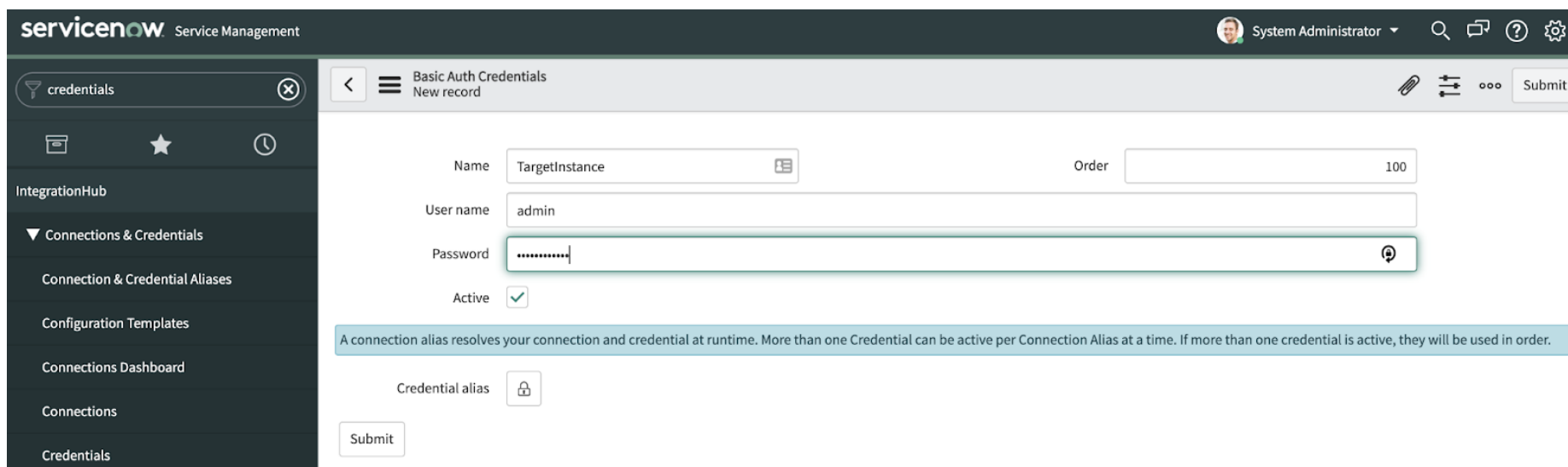


The next screen is where you will enter your destination instance credentials.

So give a name of your choice in the “Name” textbox. Enter the “User Name” and “Password” of your destination instance(dev21491).

Then click “Submit”.

This will successfully create credentials for your destination instance.



The screenshot shows the ServiceNow interface for creating a new Basic Auth Credential. The page title is "Basic Auth Credentials" and it indicates a "New record". The form fields are as follows:

- Name:** TargetInstance
- Order:** 100
- User name:** admin
- Password:** [Redacted]
- Active:**

A blue informational banner states: "A connection alias resolves your connection and credential at runtime. More than one Credential can be active per Connection Alias at a time. If more than one credential is active, they will be used in order." Below the banner, there is a "Credential alias" field with a lock icon and a "Submit" button.

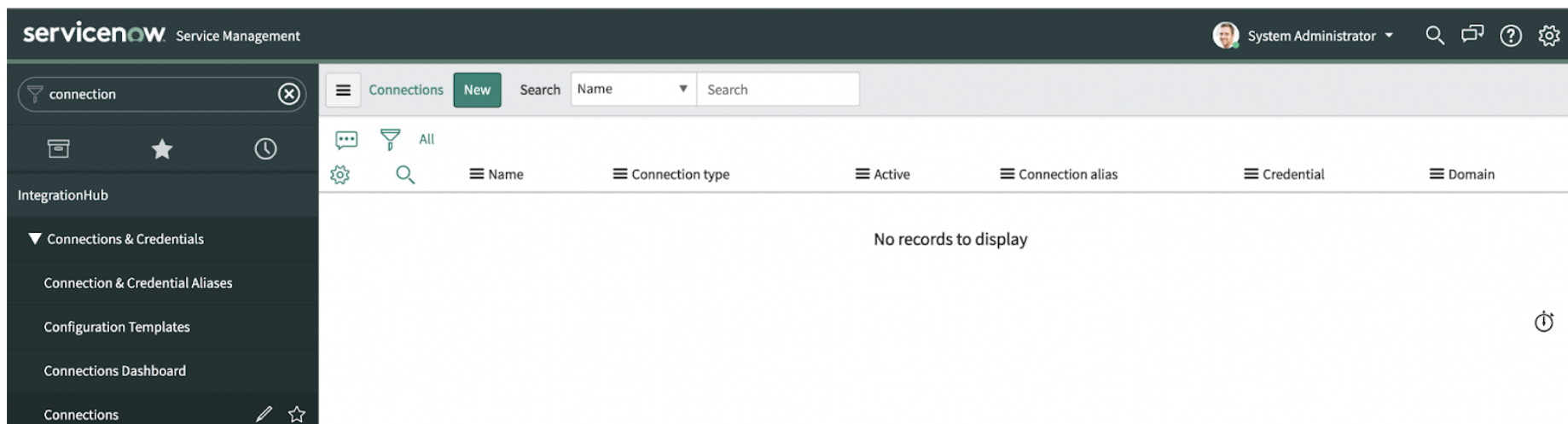
Step 3: Create Connections in the Source Instance

A connection needs to be created between the 2 instances now. Remember that the number of connections you create will depend on the number of target ServiceNow instances.

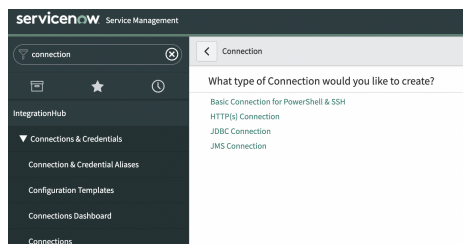
Here, I will show one, but the procedure is the same for others.

Navigate to the “Connections & Credentials” again, but this time click “Connections”.

And then click “New” again.



Select “HTTP(s) Connection” from the list.



On the “New Record” form below, fill in the details as follows:

“**Name**”: Can be anything. I have given the name “democonnection”.

“**Credential**”: Click on the search bar to select the credentials you have created in step 2.

“**Connection alias**”: Click on the search bar next to it and select “sn_ebonding_ah.ServiceNow”.

“**Connection URL**”: Enter the URL of the destination ServiceNow instance.

All the other fields can remain the same. Once done, click “Submit”.

The screenshot shows the ServiceNow interface for creating a new HTTP(s) Connection record. The form is titled "HTTP(s) Connection New record" and is currently being edited by a "System Administrator". The form fields are as follows:

- Name**: democonnection
- Active**:
- Credential**: TargetInstance
- Domain**: global
- Connection alias**: sn_ebonding_ah.ServiceNow
- Connection URL**: https://dev21491.service-now.com/ (highlighted with a red box)
- Use MID server**:
- Connection timeout**: (empty field)

There are two informational messages in the form:

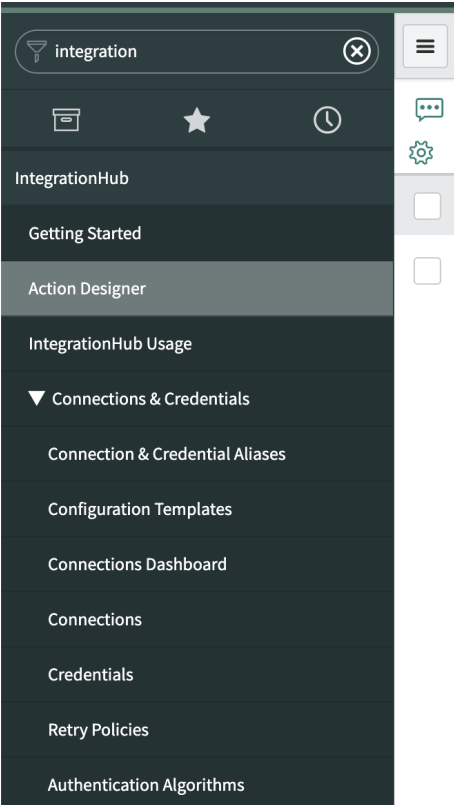
- A blue message: "A connection alias resolves your connection and credential at runtime. Only one Connection is active per Connection Alias at a time. [More Info](#)"
- A light blue message: "Manually enter your connection URL or use the URL builder to build the connection string. [More Info](#)"

The "Submit" button is located at the bottom left of the form.

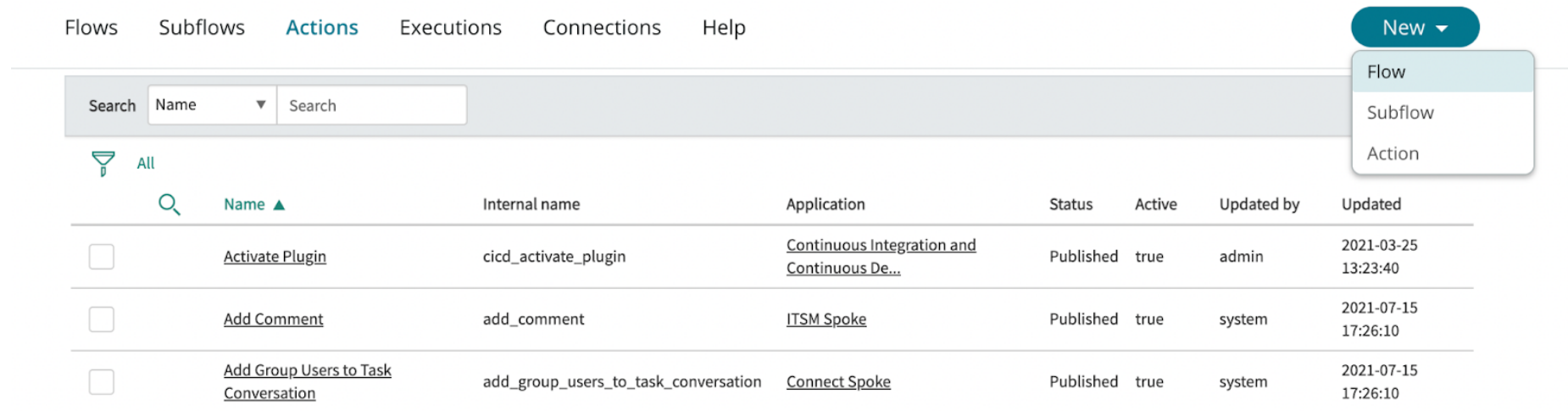
Step 4: Design the Flow

After setting up the connection, it's time to design the eBonding flow. This flow represents how you want to configure the eBonding connection, like specifying the condition for the incident to be synced to the destination instance, etc.

First, navigate to IntegrationHub> “Action Designer”.



Under the “Actions” tab, click the “New” button on the extreme right.



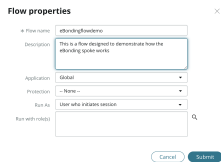
The screenshot shows the ServiceNow interface for the 'Actions' tab. At the top, there are navigation tabs: 'Flows', 'Subflows', 'Actions' (selected), 'Executions', 'Connections', and 'Help'. On the right side, there is a 'New' button with a dropdown menu open, showing options for 'Flow', 'Subflow', and 'Action'. Below the navigation, there is a search bar with 'Name' selected and a search input field. A filter icon and 'All' are visible. The main area contains a table of actions:

	Name ▲	Internal name	Application	Status	Active	Updated by	Updated
<input type="checkbox"/>	Activate Plugin	cicd_activate_plugin	Continuous Integration and Continuous De...	Published	true	admin	2021-03-25 13:23:40
<input type="checkbox"/>	Add Comment	add_comment	ITSM Spoke	Published	true	system	2021-07-15 17:26:10
<input type="checkbox"/>	Add Group Users to Task Conversation	add_group_users_to_task_conversation	Connect Spoke	Published	true	system	2021-07-15 17:26:10

Select “Flow” from the list. This will open the Flow designer.

Next, give a name to the flow, provide a description for it and click “Submit”.

On this form you can choose the roles that can access this flow, or choose who should be able to run this flow and the like. I have kept the defaults.



The screenshot shows the 'Flow properties' form. It includes fields for 'Flow name' (eBondingBackend), 'Description' (This is a flow designed to demonstrate how the eBonding operations...), 'Application' (Global), 'Production' (-None-), and 'Run as' (User with roles access). There are 'Cancel' and 'Submit' buttons at the bottom.

Now you have to add triggers and actions to the flow.

Let's start with triggers. They are used to detect certain events. Whenever such an event (like creation or update or both) occurs that meets the trigger criteria then synchronization happens. To create a new trigger, click "Add a trigger" on the screen shown below.

The screenshot shows the ServiceNow flow editor interface. At the top, there is a breadcrumb navigation bar with a home icon, the text 'Flow eBondingflowdemo', a close icon (x), and a plus icon (+). Below this, the flow name 'eBondingflowdemo' is displayed in a large font, followed by the status 'Inactive' in a grey box. To the right of the flow name is a 'Properties' button. The main area is divided into three sections: 'TRIGGER', 'ACTIONS', and 'ERROR HANDLER'. Under 'TRIGGER', there is a blue circular button with a plus sign and the text 'Add a trigger'. Under 'ACTIONS', there is a blue circular button with a plus sign and the text 'Add an Action, Flow Logic, or Subflow'. Under 'ERROR HANDLER', there is a toggle switch that is currently turned off, and the text 'If an error occurs in your flow, the actions you add here will run.'

As shown in the image below, there are different triggers for when a record (in our case incident) is created, updated or both. Select "Created".

Select a Trigger ▼

Search triggers

RECORD	
Created	Created Trigger initiates from a ServiceNow record creation that meets the condition filter.
Updated	
Created or Updated	

DATE

- Daily
- Weekly
- Monthly
- Run Once
- Repeat

APPLICATION




You then need to select the “Table” to which the trigger needs to be applied. We have selected “Incident”.

Then under the “Condition” section, select the condition for filtering. The interface is pretty self-explanatory, so it should be easy to set your conditions even if you are doing this for the first time.

Here I have created a trigger for syncing incidents that have Urgency=1.

You can even choose to add “New Criteria” and try and experiment with the triggers using the “OR” and “AND” options.

TRIGGER

 Incident Created  

Trigger Table

* Table

Condition All of these conditions must be met

or

There is also an ‘Advanced Options’ button below on this page. These options allow you to decide when and where you want the flow to run.



Once you decide on all your requirements, click the “Done” button and you will be able to see your trigger on the previous page.

Now it’s time to add “Actions”.

So under “Actions”, click “Add an Action” on the screen shown below.

The screenshot shows the ServiceNow eBonding flow editor interface. At the top, there is a breadcrumb trail: Home > Flow > eBondingflowdemo. Below this, the flow name "eBondingflowdemo" is displayed with an "Inactive" status and a "Properties" button. The main area is divided into three sections: "TRIGGER", "ACTIONS", and "ERROR HANDLER". Under "TRIGGER", there is a "now" icon and a selected trigger: "Incident Created where (Urgency is 1 - High)". Under "ACTIONS", there is a plus icon and the text "Add an Action, Flow Logic, or Subflow". Under "ERROR HANDLER", there is a toggle switch that is currently turned off, and the text "If an error occurs in your flow, the actions you add here will run."


In the search bar, simply start typing eBonding, and once the result comes up click “ServiceNow eBonding Example” and then choose the “Create Remote Incident”,but you can choose any action you want to implement. We have already discussed these actions in the above section.

 **eBondingflowdemo** Inactive

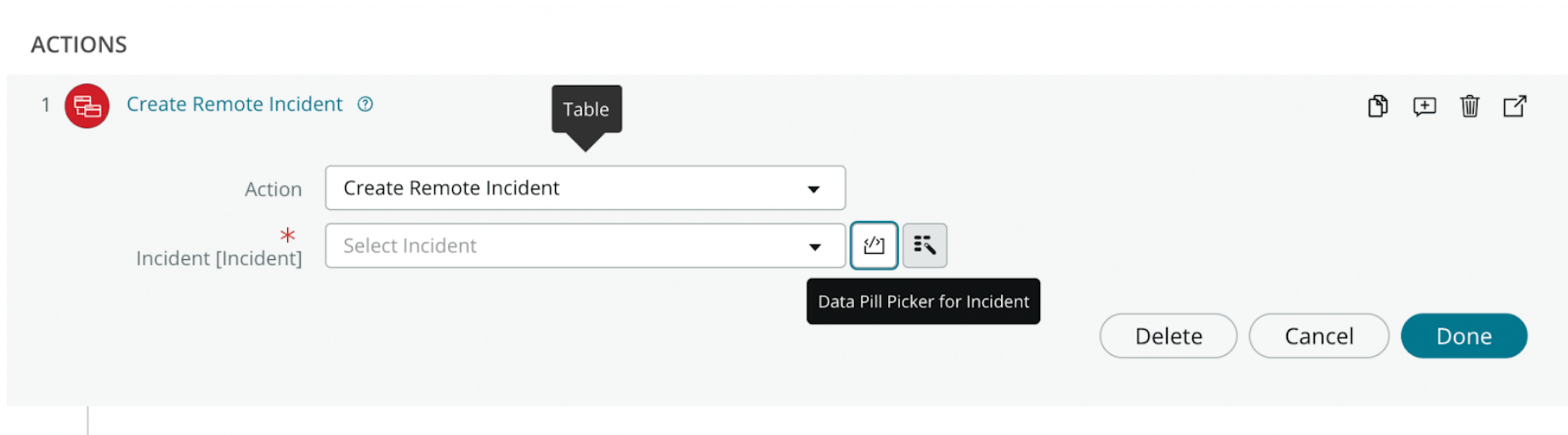
ACTIONS

✕ Action Flow Logic Subflow

Q ebonding

INSTALLED SPOKES	Default
 ServiceNow eBonding Exa...	Create Remote Incident ⓘ
	Lookup Remote Incident
	Update Remote Incident

Under “Actions” of “Create Remote Incident” click on the data pill picker button.



On the screen that pops up, select “Trigger-Record Created” and click on the “Incident Record” on the right tab. This will define what actions to perform for when an incident is created.



You can also choose to activate the “Error Handler” toggle button present at the bottom of the page. In this case, you can choose to perform particular actions in case an error occurs in your flow.

Once you are done adding Triggers, Actions and Errors(optional), click on the “Save” button on the top-right corner of the screen to save your flow.

The screenshot displays the ServiceNow eBonding flow editor interface. At the top, there is a navigation bar with a home icon, a tab for 'Flow eBondingflowdemo', and buttons for 'Properties', 'Test', 'Executions', 'Activate', and 'Save'. Below the navigation bar, the main workspace is divided into three sections: TRIGGER, ACTIONS, and ERROR HANDLER. The TRIGGER section contains a trigger labeled 'Incident Created where (Urgency is 1 - High)'. The ACTIONS section contains one action labeled '1 Create Remote Incident'. Below the ACTIONS section, there are three buttons: 'Action', 'Flow Logic', and 'Subflow'. The ERROR HANDLER section has a toggle switch and a description: 'If an error occurs in your flow, the actions you add here will run.' On the right side, there is a 'Data' panel showing a tree view of variables: 'Flow Variables', 'Trigger - Record Created' (with sub-items 'Incident Record', 'Incident Table', 'Run Start Time'), and '1 - Create Remote Incident' (with sub-items 'Remote Incident Number', 'Correlation Id', 'Action Status').

Then click the “Activate” button to activate the flow. You will find it right next to the “Save” button in the screen above.

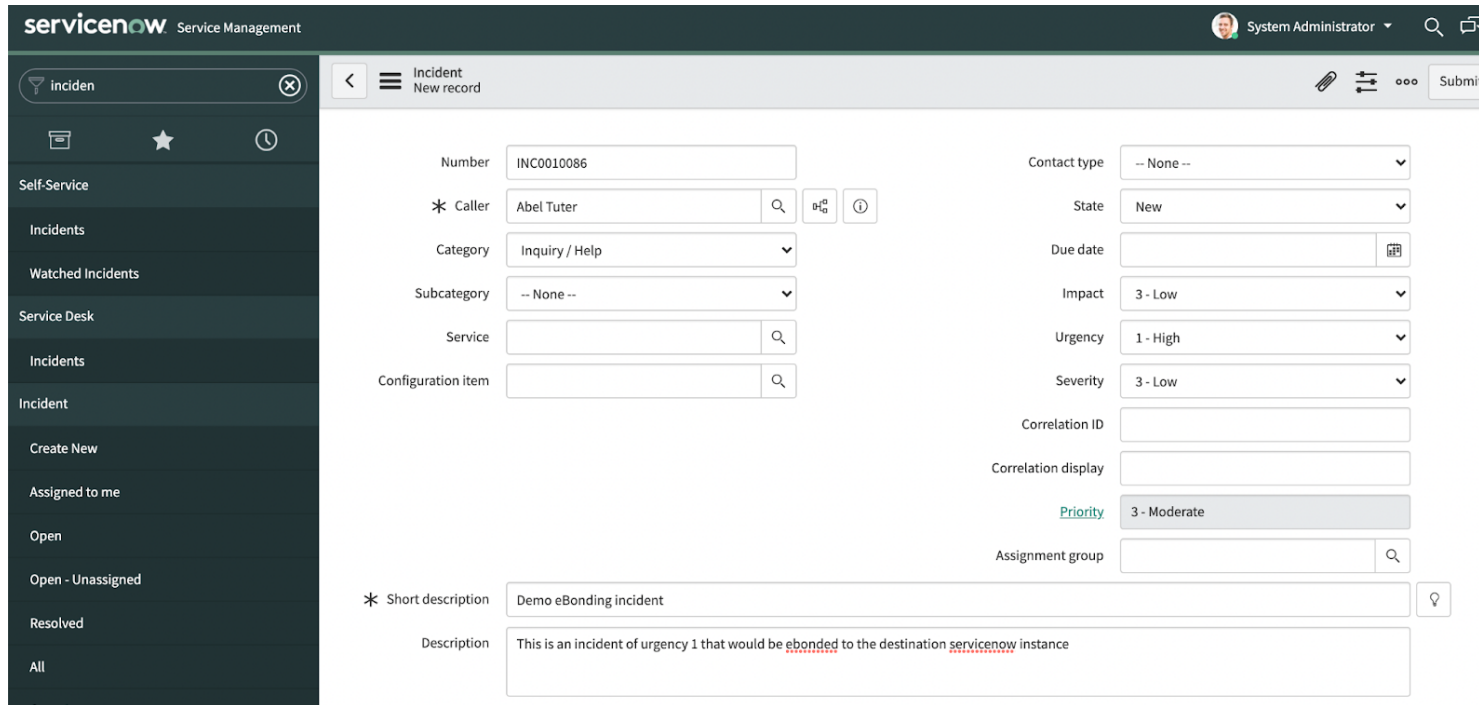
Step 5: Testing the eBonding Flow

After we have done this initial setup, it is time to test our connection.

For this, I will create an Incident of Urgency=1 as shown below on the source ServiceNow instance (dev79664) and we will see this incident being reflected on the destination ServiceNow instance (dev21491). Fill in the details and click “Submit”.

To start creating an Incident, head over to your source ServiceNow instance (dev79664) and search for “Incidents” on the left-hand search menu. Click on “New” to create a new one.

Note: The following incident is created on dev79664, our source instance.



The screenshot shows the ServiceNow 'Incident' form interface. The top navigation bar includes the 'servicenow' logo, 'Service Management', and a user profile for 'System Administrator'. The left sidebar contains a search bar with 'Incident' entered and a list of navigation options: Self-Service, Incidents, Watched Incidents, Service Desk, Incident, Create New, Assigned to me, Open, Open - Unassigned, Resolved, and All. The main form area is titled 'Incident New record' and contains the following fields:

- Number: INC0010086
- Caller: Abel Tuter
- Category: Inquiry / Help
- Subcategory: -- None --
- Service: (empty)
- Configuration item: (empty)
- Contact type: -- None --
- State: New
- Due date: (empty)
- Impact: 3 - Low
- Urgency: 1 - High
- Severity: 3 - Low
- Correlation ID: (empty)
- Correlation display: (empty)
- Priority: 3 - Moderate
- Assignment group: (empty)
- Short description: Demo eBonding incident
- Description: This is an incident of urgency 1 that would be eBonded to the destination servicenow instance

Here you can see that initially the Correlation ID field is blank, but once the incident is created, this Correlation ID field gets automatically filled with the incident number created on the destination instance. This is shown in the image below.

The screenshot shows a ServiceNow incident record for INC0010086. The interface includes a header with navigation icons and action buttons (Follow, Update, Resolve, Delete). The main form is divided into two columns. The left column contains fields for Number (INC0010086), Caller (Abel Tuter), Category (Inquiry / Help), Subcategory (-- None --), Service, and Configuration item. The right column contains fields for Contact type (-- None --), State (New), Due date, Impact (3 - Low), Urgency (1 - High), Severity (3 - Low), Correlation ID (INC0010001), Priority (3 - Moderate), and Assignment group. The Correlation ID field is highlighted with a red border. Below these fields are the Short description (Demo eBonding incident) and Description (This is an incident of urgency 1 that would be ebonded to the destination servicenow instance).

On the destination side (dev21491), this scenario is opposite.

Incident INC0010001

Number: INC0010001

Caller: Abel Tuter

Category: Inquiry / Help

Subcategory: -- None --

Service:

Service offering:

Configuration item:

Short description: Demo eBonding incident

Description: This is an incident of urgency 1 that would be ebonded to the destination servicenow instance

Contact type: -- None --

State: New

Impact: 3 - Low

Urgency: 1 - High

Priority: 3 - Moderate

Assignment group:

Correlation ID: INC0010086

Assigned to:

So in short, the Correlation IDs store the incident numbers from the opposite ServiceNow instances once the eBonding action is triggered.

This is just an example of a flow you can implement with the eBonding spoke. But there are a lot of different triggers and actions that you can try out for different scenarios you wish to implement.

ServiceNow eBonding Spoke Alternative

Well, we have discussed the eBonding Spoke till now. But this leaves us with the question of whether this is the only solution available in the market. Well, clearly not.

There are players who support and do the exact same thing that eBonding Spoke does. [Exalate](#) is one such tool.

It is a bi-directional synchronization tool (aka an eBonding tool) that helps streamline collaboration and connects your work across multiple teams or projects within a single company or across multiple companies, in real-time.

Why we are discussing this particular tool is for the following reasons:

- It supports **decentralized integration**. Meaning that each side of the integration independently (or autonomously) controls what it sends and receives. This is possible because of the **distributed architecture** it supports. It also essentially means that each side can choose what to send over and what to receive from the other side, making the integration data to be accessed and viewed only by authorized parties. If there is some information you don't want the other eBonded side to view, you simply don't send it, and the same goes for receiving information too.

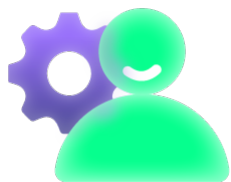


- Inferring from the above point, the eBonded systems using Exalate become **loosely coupled**, in essence making them more manageable and scalable. To mitigate the data integration issues that arise from loosely-coupled systems, it implements mechanisms like transactional sync queues. This increases the **reliability** of your eBonded integration, since the sync queues can help ensure that changes or updates since the last downtime get automatically transferred to the other side.
- It also supports **security** mechanisms like encrypted data transfer, JWT based token mechanisms, access controls and secure transfer protocols like HTTP(s). You can check out the security and architecture [whitepaper](#)



- Exalate supports **3 configuration modes**:
 1. **Basic mode**: This mode allows you to eBond (or sync) using predefined mappings between 2 applications. These mappings cannot be changed. For e.g: for ServiceNow you can sync the short-description, comments, description and attachments of incidents only. So this mode works best for basic eBonding integration needs.
 2. **Visual mode**: This mode has a point-and-click interface that is easy to understand and use by business users. This mode works well for advanced eBonding integration cases since you can add or edit new mappings and set up configurations specific to your requirement. This mode is ideal for intra-company (within a single company) use cases.
 3. **Script mode**: This mode is based on an intuitive scripting engine. Groovy-based scripts help you control what you send and receive on each side independently. You can set up new mappings or integrate advanced logic with this mode, helping you accommodate even the most complex sync requirement. This mode is ideal for cross-company (with multiple companies) integration use cases.These modes make Exalate a perfect fit for all kinds of users: business or technical. They also make it possible to use

Exalate for various eBonding use cases within a single company or across multiple companies. So it is very **flexible** in this regard.



- Exalate supports a lot of different applications that can be eBonded. They are: Jira, ServiceNow, GitHub, Salesforce, Zendesk, Azure DevOps, HP ALM and the like. And all these new applications can be easily added into your existing workflows using the different configuration modes that we just saw.

If you want to learn more about how Exalate enables you to integrate multiple ServiceNow instances, here's the complete guide to a [ServiceNow to ServiceNow integration](#).

Note: Check out the [recommended reads](#) at the end of this blog post to read all about ServiceNow and other work management systems' integration like Jira, Salesforce, and the like.

Bottomline

For the use case we discussed in the section above i.e syncing an incident of Urgency=1 between two ServiceNow instances, both eBonding spoke and Exalate can work. You can also design more advanced workflows with the help of these tools.

But there are inherent differences between the 2 tools that need to be taken into consideration.

Firstly, eBonding Spoke supports only ServiceNow-ServiceNow bonding. So it would work best if you have 2 ServiceNow production instances catering to different functions, maybe one is for internal use and the other exposed externally, i.e customer facing. But if you want to eBond ServiceNow to other applications like Jira, Azure DevOps etc, or maybe even a Jira-Salesforce, Zendesk-Azure DevOps eBonding integration then Exalate is the obvious choice.

Secondly, eBonding Spoke is a default part of the ServiceNow environment and does not require an additional subscription cost. So if you want to use predefined design integration patterns with 2 ServiceNow instances and a simple use case that can be accommodated under them, you can choose to use this ServiceNow offering instead of Exalate.

Thirdly, the Actions along with the different flows (in Flow Designer) that can be designed with eBonding Spoke provide exhaustive functionality to eBond 2 ServiceNow instances, but after a point they become static. So it is difficult to have advanced process logic or customized attributes with eBonding Spoke, as it gets limited by ServiceNow's own infrastructure. With Exalate, Groovy scripting allows you to accommodate even the most complex eBonding integration cases.



Fourthly, just as a spin-off of the first point here, ServiceNow's integration automation capabilities are further extended through IntegrationHub that allows you to connect ServiceNow with other applications like Jira, GitHub etc.

In this case, ServiceNow will be in the driving seat orchestrating all the operations required for synchronization. This can seem feasible for simple eBonding cases. But in reality each environment, and thus each eBonding integration over time develops at its own pace. During this phase, if there is tight coupling between the systems under integration it leads to intricate complexities, resulting in increased overheads.

In such a situation keeping the systems loosely coupled is important. Exalate supports loosely-coupled systems and hence is a safe bet in such situations. You can have a look at this [blogpost](#) to get an overview of how IntegrationHub compares to Exalate.

All in all, to sum this up, the choice between eBonding Spoke and Exalate is more strategic than random. So take an informed decision based on the context set up through this blogpost.

Conclusion

This article threw light on what eBonding is and how it can be beneficial for today's fast growing, digital and data-driven companies. We saw how an eBonding integration can help such companies interconnect with other companies using ITSM applications like ServiceNow.

We then focussed on understanding ServiceNow infrastructure for providing eBonding capabilities in the form of IntegrationHub and eBonding Spoke. We also discussed what a typical eBonding OOB Action looks like.

Moving ahead, we saw a simple use case of eBonding an incident between 2 ServiceNow instances using the eBonding Spoke.

And then finally, we covered which other eBonding tools exist in the market. I chose to compare Exalate because of its feature offerings like scripting engine, distributed architecture, decentralized integration and the like. We also discussed how they both aren't direct competitors, and the choice between one of the eBonding solutions (eBonding Spoke or Exalate) depends on your specific requirements and long term vision.

Recommended Reads:

- [eBonding Integration: The Ultimate Guide to Flexible Data Sync](#)
- [ServiceNow Integrations: Integrate ServiceNow and Other Systems Bidirectionally](#)
- [ServiceNow to ServiceNow Integration](#)
- [Jira ServiceNow Integration: How to Set up an Integration in 6 Steps](#)
- [How to Set Up an Azure DevOps ServiceNow Integration](#)
- [How to Set Up a Salesforce ServiceNow Integration](#)
- [Introduction to ServiceNow IntegrationHub \(podcast\)](#)