How Exalate Works?

The synchronization process takes place whenever a connection between the two systems is established and configured. Before you proceed, we recommend that you take a look at a brief overview of the synchronization process to find out how things work behind the scenes.

The synchronization process involves four main components: two task management systems that host the data that you want to sync, and two **Exalate Nodes**. The Exalate Node connected to each task management system acts like a gateway and is responsible for:

- Determining if the data "under sync" has been changed or if new data needs to be brought "under sync"
- Extracting the relevant data from the tracker and formatting it as a structured message-**replica**
- Exchanging the data with other nodes connected to the network
- Applying incoming messages locally in a way that makes sense, either updating the existing entity or creating a new one

How do the Exalate Nodes Interact with the Platforms and their Data?
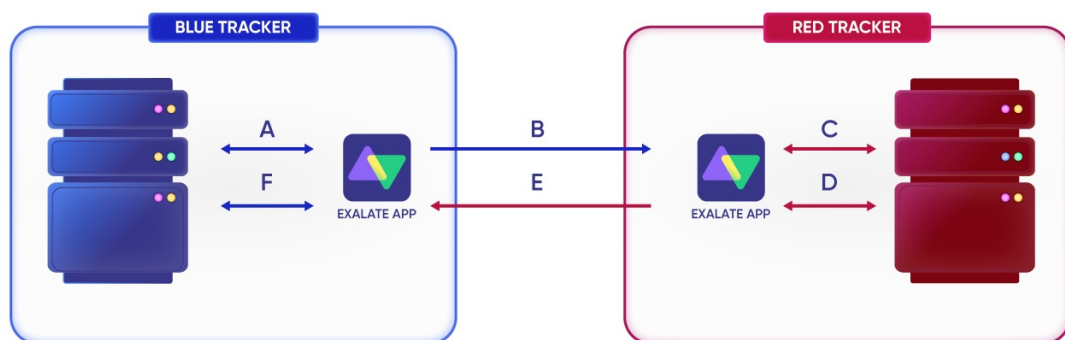
1. Exalate Node (Blue) interfaces with instance(Blue) for retrieving Incident information.
- It either polls the instance (Blue) for any changes since the last full synchronization or gets triggered by the instance (Blue), stating that a change has occurred.
- It collects all the relevant Incident information and composes a message.
2. Exalate Node (Blue) sends the message to Exalate Node (Red) for further processing.
3. Exalate Node (Red) decomposes the message and applies the message to the instance (Red), such as updating an issue summary.
4. Exalate Node (Red) interfaces with instance (Red) to retrieve the changes that have been triggered by the synchronization and composes an acknowledgment message.
5. Exalate Node (Red) sends the acknowledgment message back to Exalate Node (Blue).

6. Exalate Node (Blue) processes the acknowledgment message and applies it to instances (Blue).

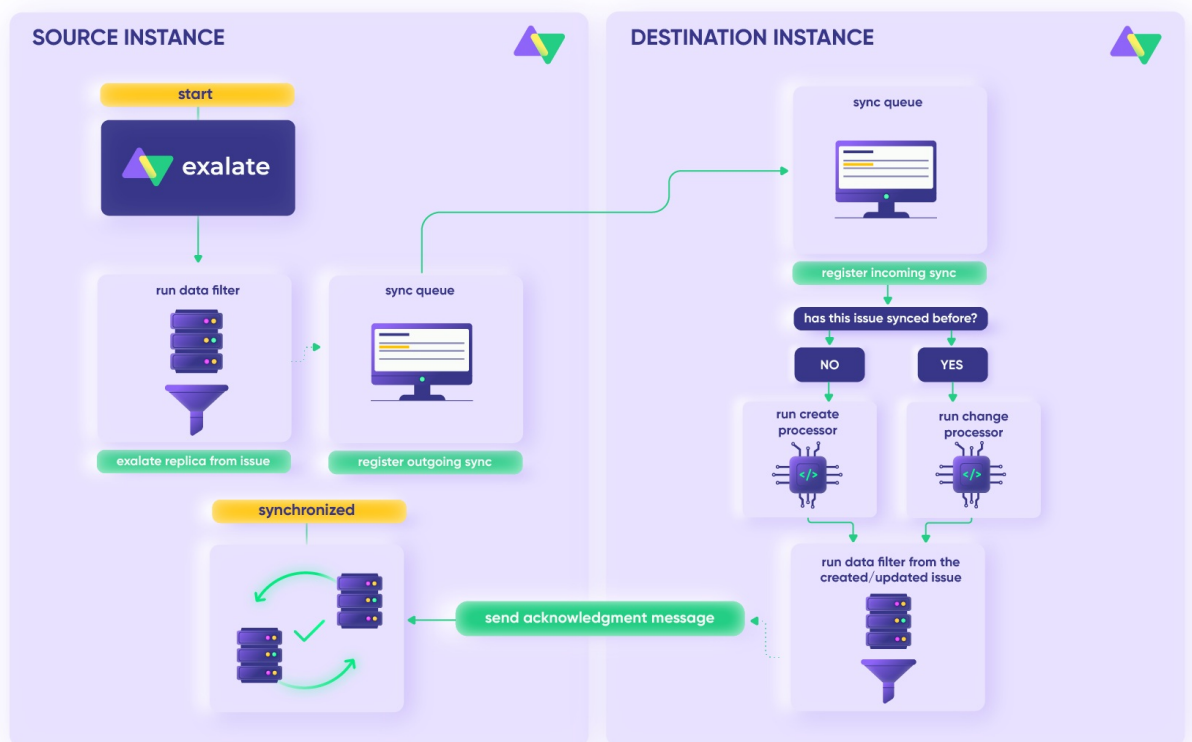## Outgoing and Incoming Sync Processors

Outgoing and incoming sync processors are basically the messages that Exalate nodes exchange during the synchronization process.

With the **outgoing sync processor,** you can specify what information can be sent to the destination instance. Exalate runs the outgoing sync processor when you start the synchronization or update the local issue that is under sync.

The **incoming sync processor** contains information on how to handle incoming data from the remote instance. Exalate runs the incoming sync processor every time there's new data received from the remote side.

You can define what data should be sent and how to handle it on the destination instance with the help of the **Outgoing and Incoming Rules**.

The Synchronization Process Step-by-Step

1. The synchronization process begins with one of the available triggers: operation, trigger, API, post function, or polling for changes.
2. Exalate runs the **outgoing sync processor** and extracts a **replica** from the entity.
3. The outgoing sync event is registered in the **Sync Queue**.
4. The outgoing sync event is sent to the **destination instance.**
5. The incoming sync request is registered in the Sync queue on the **destination side.**
6. Exalate runs the **incoming sync processor** with the received message and applies the

change to the local entity.

7. Exalate runs the **destination instance outgoing sync processor** to create an acknowledgment message and sends this message to the **source instance**.

8. The synchronization process is complete, and all sync operation data has been cleaned.

Learn more in the Security and Architecture Whitepaper.