ConductorOne

# Securing Identity for Any Application:

## Deep Dive into ConductorOne Integrations

## TABLE OF CONTENTS

Digital transformation, and specifically the adoption of cloud apps and infrastructure, have rapidly reshaped the corporate tech landscape. While some companies may have been built in a cloud first way and are “cloud native”, many organizations still operate in hybrid tech stack environments. This guide will look at some of the variations of environments we see across companies today and how ConductorOne connectors can integrate with, and protect identity and access for, any application and architecture.

# 1. Types of Environments

Depending on industry, size, and where a business is in the digital transformation and cloud adoption maturity curve, companies have varying degrees of cloud adoption and complexity in their technology environment. These can very broadly fit into three buckets:

## 1.1 Fully On-Premises Architecture

Organizations with stringent security or regulatory needs may opt to keep all their systems on-premises or hosted in a private cloud (on-prem). This means that all the necessary infrastructure and services are installed and maintained on the companies servers or private cloud infrastructure and managed by IT, security, and dev ops teams.

## 1.2 Hybrid Architecture

Most large enterprises operate through a blend of on-premises and cloud-based systems. They may have several critical systems that they manage and control in their own infrastructure, using technologies such as access gateways or VPNs to bridge the access gap for these systems. They may also have adopted one to many SaaS or public cloud providers as well. These environments are particularly tricky as they usually retain multiple identity directories, or sources of "truth", across on-prem directory stores, such as Active Directory, and cloud directories such as cloud HR systems or a cloud IdP.

## 1.3 The Cloud-Native Architecture

Cloud-native companies use all cloud applications and infrastructure. These organizations design, build, and deploy their applications to take full advantage of the cloud's elasticity, scalability, and speed. Generally these companies are "younger", to wit, started and built in the last few years when the availability of cloud SaaS and infrastructure providers was prolific enough to provide all of the necessary tooling to run a business. These organizations enjoy benefits such as lower operational costs and improved agility that come from being on a cloud-native architecture.

# 2. Connecting to Any Application or Technology

ConductorOne has a range of options to connect our identity security platform directly to a customer's environment – regardless of whether the apps, directories, and infrastructure are in the cloud or hosted within the customer's environment. ConductorOne achieves this by enabling a hybrid agentless + agent approach for connecting to the ConductorOne service. "Managed Connectors" is the term that we use to describe the agentless deployment of connectors that interface with cloud apps and infrastructure. "Self Hosted Connectors" is the term we use to describe deploying a connector alongside the technology in question, within the customer environment. Let's get into more detail:

## 2.1 Managed Connectors

Managed connectors are agentless and deployed directly from the ConductorOne service. These connectors are built and maintained by ConductorOne and are easily set up and configured via the ConductorOne web console or Terraform. A list of supported connectors can be found here. With managed connectors, the user inputs the necessary credentials to connect the ConductorOne service to the SaaS, IaaS, or directory in question. Once authorized, these connectors provide a full inventory of identities, groups, and access rights from the service. ConductorOne uses this data, and the authorized credentials, to automate identity security workflows and access management such as provisioning and deprovisioning. Data from connected services is automatically synced and the administrator does not need to worry about connector maintenance or scheduling.

## 2.2 Self Hosted Connectors

Connectors can also be self hosted, that is, hosted within the customer's on-prem, private cloud, or public cloud environments. Self hosting provides several capabilities not offered from managed connectors, namely:

- → **Control over credentials:** with self hosted connectors, the ConductorOne service does not have access to the credentials used for authorization to the connected service.
- → **Granular auditing:** hosted connector actions can be granularly audited and recorded beyond what is provided by the ConductorOne service.
- → **Connectivity to non-internet exposed services:** Hosted connectors can be deployed in an agent-like model. This allows them to run alongside and communicate with services that are on-prem, in private clouds, or are otherwise inaccessible from the ConductorOne service.
- → **Control over sync schedule:** self hosted connectors allow for flexibility and deeper configuration on how often and what data is ingested.
- → **Support for custom apps:** ConductorOne provides an open source SDK (the Baton SDK) that can be extended to connect to non off-the-shelf software such as support portals or homegrown apps.

ConductorOne provides open source versions of our managed connectors through a project called Baton. Baton connectors for SaaS, IaaS, and cloud directories are the same as the managed connectors provided through the ConductorOne service. However, by providing these as open source, companies can extend them and deploy them as self-hosted connectors for the aforementioned benefits.

Certain connectors must be self-hosted, due to the nature of the technology that is being connected to the ConductorOne service. For instance, a PostgreSQL database or Active Directory instance needs direct connectivity and requires the use of a self-hosted connector. These connectors can be found in the Baton open source directory.

### 2.2.1 On-demand vs Service Mode

A unique attribute of self-hosted connectors is that they can be run in different modes. As the customer has complete control over how and when these connectors run, this can provide useful flexibility. The two modes of operation are:

- **On-Demand:** in this mode, the connector does not run continuously. Rather, it runs in a single “shot” where data is extracted and can be either automatically uploaded to the ConductorOne service or manipulated manually from a .c1z output file.
- **Service Mode:** this mode is functionally equivalent to managed connectors. In this mode, the connector is run continuously as a service. The connector runs as if it were a managed connector in the ConductorOne service, but it is simply hosted in the customer’s infrastructure.

Some connectors require user interaction for opening a connection, such as requiring the user to provide a two-factor authentication. In such cases, the connector may still be provided by ConductorOne, but requires self-hosting and runs on-demand as needed.

## 2.3 Direct Data Ingestion

In addition to connectors, which automate the process of synchronizing identity security data and orchestrating workflows, ConductorOne also supports various direct data ingestion capabilities. These includes:

#### S3 Bucket

With this method, the ConductorOne service ingests data from an S3 bucket. The customer is responsible for pushing and updating the data in the S3 bucket, and can do so however they see fit. The ConductorOne service monitors this bucket continuously and will update the data reflected in the service upon update. Data can be formatted in a .csv, .xslx, or .c1z file format.

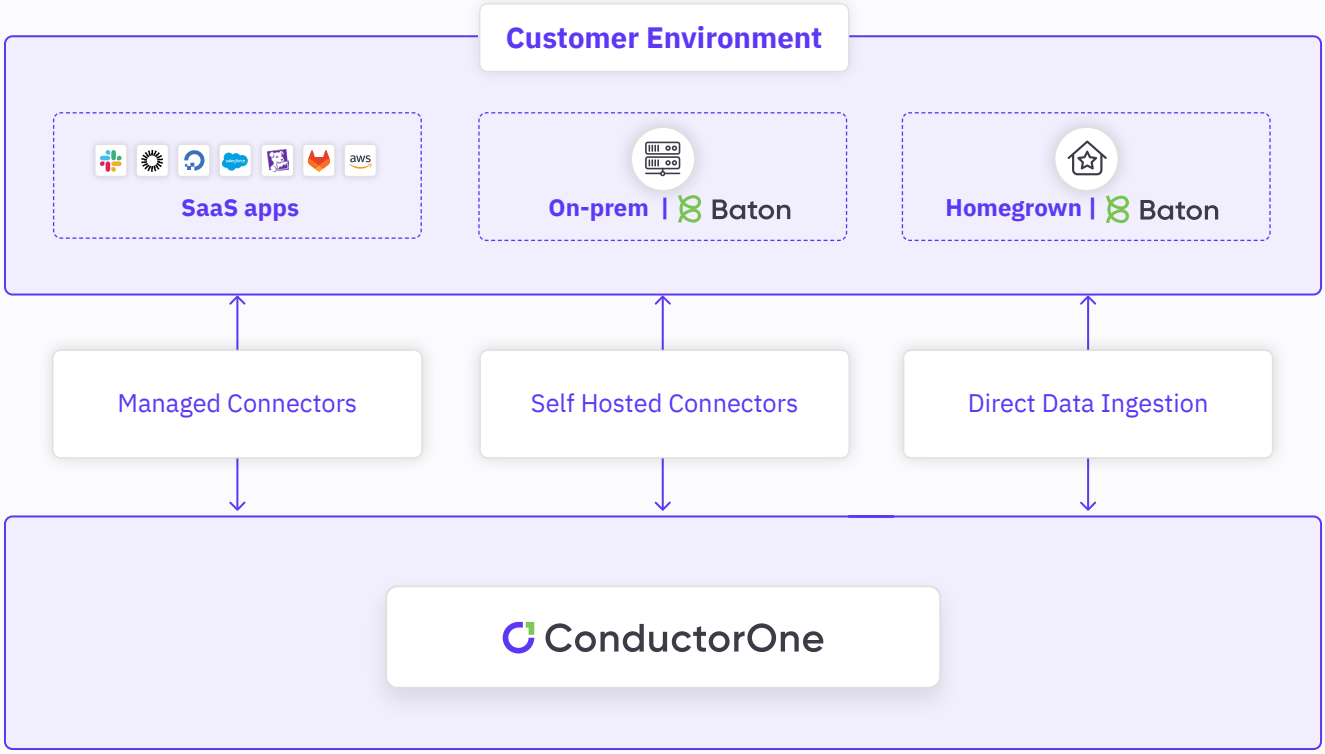
#### Manual Data Upload

Data from the application is ingested through a manual .csv, .xlsx, or .c1z file upload. The ConductorOne service supports varying file formats as a file mapping can be specified and re-applied to subsequent uploads of the data.

## 2.4 Building your own Connector

Many companies have homegrown or back office tools that are essential to secure. In such cases, customers can build their own connector with the Baton software development kit (SDK). Read the Baton SDK getting started guide to learn more.

Once built, these connectors can be integrated directly with the ConductorOne service using the self-hosted connector deployment model.



# 3. Evaluating Support for Your Environment

When considering an Identity Security solution, it's important to evaluate how your environment will be supported. Keep in mind and consider:

### Off-the-shelf support

How many of your SaaS, IaaS, directories, and infrastructure tools does the solution provide off-the-shelf connectors for.

### Support for hybrid or on-prem systems

Does the solution support hybrid cloud or on-prem technologies such as Postgres, LDAP, Active Directory, and so forth.

### Data segmentation

Does the solution allow for flexible connector deployment in the case of sensitivity of sharing credentials.

### Scalability

Can the solution scale to support your environment size and user counts.

### Back office portal and custom apps

Can the solution support non-COTS (commercial off-the-shelf), non-SaaS software.

### Support and maintenance

Does the provider have robust customer support and regular maintenance.

**With a holistic view of your environment, you can identify and remediate identity centric threats proactively.**

## CONCLUSION

Identity and access control is complicated, and especially so for hybrid IT environments that combine cloud and on-prem apps. At ConductorOne, we recognize that identity is both messy and hard, but getting it right and securing it is paramount to preventing breaches and customer data loss. ConductorOne's comprehensive connector catalog and deployment options are designed to support customers of all environments, allowing for unified identity security, reduced costs, and streamlined compliance.

## GLOSSARY

- → **.c1z Format:** the file format generated by the Baton SDK
- → **Baton:** an open source project comprising an SDK for building custom connectors, a CLI for interacting with connectors, and the connectors themselves
- → **Baton SDK:** an open source project used for building custom connectors
- → **ConductorOne service:** the ConductorOne cloud service
- → **Connector:** a piece of technology that is used to orchestrate identity and access control to a cloud or on-prem service e.g. Azure, Snowflake, Active Directory, etc.
- → **Credential:** a key or token that is used to authenticate to an application or service
- → **Managed connector:** an “agentless” connector hosted and managed in the ConductorOne service
- → **Self hosted connector:** a connector hosted by the customer in the customer’s environment

**Securing Identity for Any Application:**
A Deep Dive into ConductorOne Integrations

ConductorOne.com

Chat with us