DATUM CMDB PRO · FOLLOW-UP

What leaves the customer's instance?

The shape of the CMDB. Not what's in it.

A short answer to the data question from our November 16 conversation.

EFS Networks

THE SHORT ANSWER

Three things to know. That's the whole story.

1

The records stay.

Server names, IP addresses, serial numbers, owners, locations and people never leave ServiceNow. Datum counts records. It does not read them.

2

Only the structure leaves.

Table and field definitions, rules, counts and health scores go to an AWS environment set up for that one customer. They are not used to train AI.

3

Need more control? There are options.

For customers with stricter requirements, running the analysis side in their own AWS account is available on request.

Datum never changes the CMDB. It writes results to its own tables and creates tasks for the customer's team to pick up.

THE DATA FLOW

What stays, what leaves, and where it goes.

CUSTOMER'S SERVICENOW INSTANCE

Stays here

- Every CI record: names, IPs, serial numbers, locations, owners
- Users, groups and any personal data
- Credentials and secrets
- Logs and attachments
- Script code (unless an admin opts in)

Leaves, once per run, over HTTPS

- Table and field definitions
- Identification and reconciliation rules
- Record counts per class
- CMDB Health scores
- Names and settings of business rules and scripts

Comes back: the plan and its tasks, written to Datum's own tables

DEDICATED AWS ENVIRONMENT

Where it goes

- Set up for that one customer, in AWS US East
- Analysed by Claude through Amazon Bedrock
- Not used to train AI models; not shared with the model provider (AWS policy)
- Kept 90 days, then deleted. Sooner on request

One thing to know: custom table, field and rule names are part of the structure, so they do leave. Record contents never do.

AN EXAMPLE

One server. What Datum sees of it.

IN THE CUSTOMER'S CMDB

A server record

Name	`PLANT-07-MES-DB01`	STAYS
IP address	`10.42.7.18`	STAYS
Serial number	`SN-4KX82201`	STAYS
Owner	`Plant Systems team`	STAYS
Location	`Greenville, Line 3`	STAYS

WHAT DATUM RECEIVES

The shape of the table it sits in

Table	**Linux Server**
Fields defined	**name, ip_address, serial_number, owned_by, location**
Record count	**1,906**
Matching rule	**Name only**

Enough to see that 1,906 servers are matched on name alone, so duplicates are waiting to happen. Not enough to know which servers they are.

Sample data for illustration. The server record shown is invented.

WHO IS IN CONTROL

Nothing happens unless the customer starts it.

Off until an admin turns it on

Nothing leaves until an admin enters the endpoint and key.

Runs only on request

Someone with the role clicks Run, or makes an authenticated API call. No background collection.

One customer per environment

Each customer gets a dedicated AWS environment. One customer's data never sits with another's.

Not used to train AI

AWS states that Bedrock inputs are not used to train models and are not shared with the model provider.

Deleted after 90 days

Kept 90 days, then deleted. Sooner on request.

Read-only on the CMDB

Datum never writes to the CMDB. Results land in its own tables. Every payload carries a record of what was collected.

FOR THE CUSTOMER'S COMPLIANCE TEAM

The questions they ask, answered.

Is it self-hosted?	By default, no. The app runs in the customer's instance; the analysis runs in an AWS environment EFS sets up for that customer alone.
Does plant or asset detail leave?	No. CI tables are only counted. No hostnames, IPs, serial numbers or locations.
Does any personal data leave?	No.
Can anything sensitive ride along?	Custom table, field and rule names do leave. If a table is named after a program or a client, that name is visible to the analysis. Script code leaves only if an admin turns that option on.
Is the data used to train AI?	No. AWS states that Bedrock inputs are not used to train models and are not shared with the model provider.
Does Datum change the CMDB?	No. It writes results to its own tables and creates tasks for people to pick up.
Export-controlled or defence work?	The same facts apply: no record data leaves, and hosting is commercial AWS in US East. Whether structure-level information is in scope for a program is the customer's decision.

IF THAT IS STILL NOT ENOUGH

Need more control? Run it in the customer's AWS account.

CUSTOMER-HOSTED OPTION · AVAILABLE ON REQUEST

The analysis side runs inside the customer's own AWS account.

- Available on request: we set it up with the customer's team
- The customer's account, the customer's controls and audit trail
- Datum Pro was built to support this from the start
- Launch through AWS Marketplace is on the roadmap

Sensitive options are off by default

Script code is only sent if an admin switches that on.

Stricter requirements?

We will work through them with the customer's security team, and share the field-level list of what is collected.

NEXT STEP

See it on a real CMDB.

A no-cost pilot, run together.

1 Start with one CMDB

One where ITOM, Service Mapping or AI is licensed but held up by the data.

2 A read-only assessment

Results the same day. No CMDB writes. Only the structure described here leaves.

3 A sequenced plan

Findings turned into ordered ServiceNow tasks the customer's team can run.

APPENDIX A · SAMPLE DATA

What a health score would never tell you.

1

The shadow CMDB

The team's real work lives in custom tables, so scores on the standard tables mislead.

2

The rule pile-up

A dozen overlapping business rules on servers, left behind by past projects.

3

Two sources fighting

Two feeds write to servers at the same priority. Fix this first, or the cleanup won't stick.

4

Servers on the generic class

Thousands of servers never classified. The plan sets prerequisites, then works in batches.

5

Nobody is keeping score

Health jobs were switched off. First action: turn them on and take a baseline.

6

Stale records, sorted

Records untouched in a year are split into "validate or retire" and "fix".

7

Duplicates waiting to happen

Servers matched on name only, and custom tables with no matching rule.

Each of these becomes an ordered task in ServiceNow, not just a finding.

Illustrative example on sample data.

APPENDIX B · ESTIMATES

On the numbers: start with the sample company.

~$35,000

a year

Estimated cost of CMDB-related incident delay for a 23,356-record CMDB

~$24,000

in customer staff time

One-time effort to fix the underlying records: about 320 hours at $75 an hour. Not an EFS fee

~8 months

payback

Time for the fix to pay for itself at those estimates

The tables in the overview deck show how these figures move across CMDB sizes and assumptions. They are ranges, not forecasts. For any one customer, we would replace these defaults with their own incident and cost figures.

Estimates. Defaults: 0.5% of records behind an incident a year; 4 extra hours per incident at $75 an hour.