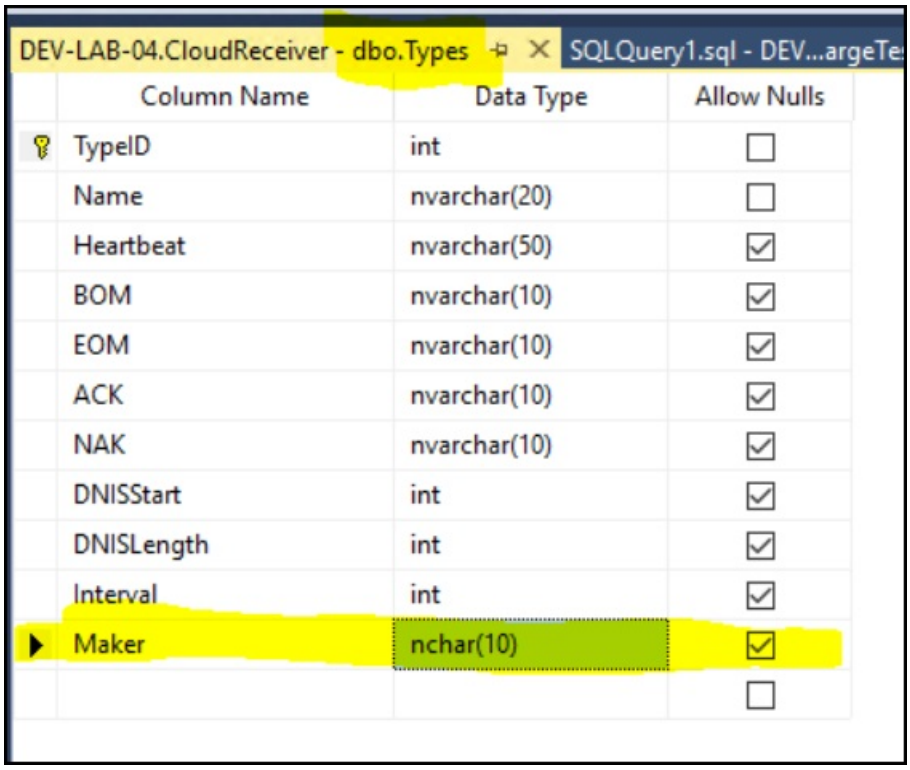
Cloud Receiver Version History

01/09/2024 12:46 pm EST

This provides the version history of the Cloud Receiver as well as the actual release files. It constitutes a running release note.

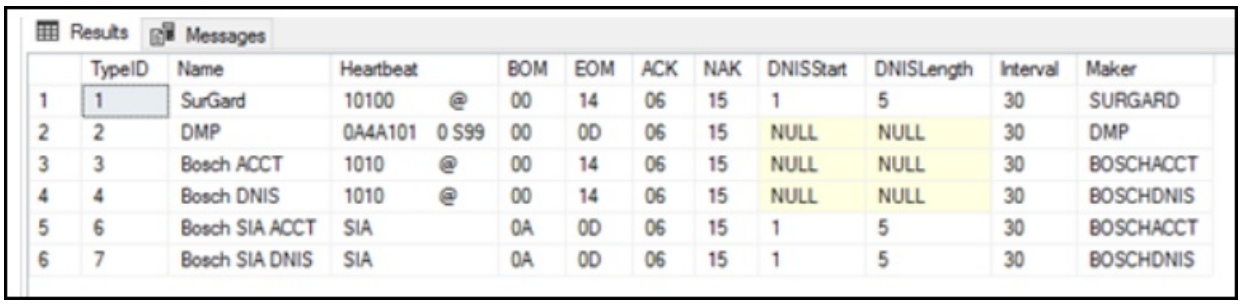
UPDATE 1 (2/4/2019)

A change was made to the Types table to add a Maker field.



The service was updated to use this when determining how to find the DNIS (or account #) that would be used for routing. The seed database was updated but the table would need to be updated as well as the stored procedure on existing databases. A script file is available for these changes.

The data in the Types file will need to be updated manually. Here is an example of what it could look like. Now we can add multiple receivers in there that are slightly different but that have the same Maker type. So, we can have different SurGard receivers with different heartbeats but that all decode for routing without a code change in the Cloud Receiver.



Results Messages

	TypeID	Name	Heartbeat		BOM	EOM	ACK	NAK	DNISStart	DNISLength	Interval	Maker
1	1	SurGard	10100	@	00	14	06	15	1	5	30	SURGARD
2	2	DMP	0A4A101	0 S99	00	0D	06	15	NULL	NULL	30	DMP
3	3	Bosch ACCT	1010	@	00	14	06	15	NULL	NULL	30	BOSCHACCT
4	4	Bosch DNIS	1010	@	00	14	06	15	NULL	NULL	30	BOSCHDNIS
5	6	Bosch SIA ACCT	SIA		0A	0D	06	15	1	5	30	BOSCHACCT
6	7	Bosch SIA DNIS	SIA		0A	0D	06	15	1	5	30	BOSCHDNIS

Additionally, the Service and Client were updated to add support for Bosch SIA mode and a few bug fixes. The Bosch SIA mode keys off a Maker value of “BOSCHACCT” or “BOSCHDNIS” along with “SIA” in the Heartbeat value in the Types table.

UPDATE 2 (3/20/2019)

SaaS team found that when you add enough routes to push the table to the edge of the Client screen, the controls push out of view. We changed the table to a fixed max size on all these and then it will put up scroll bars. Also, we made the main window fixed size so it can’t be sized down and also lose the controls.

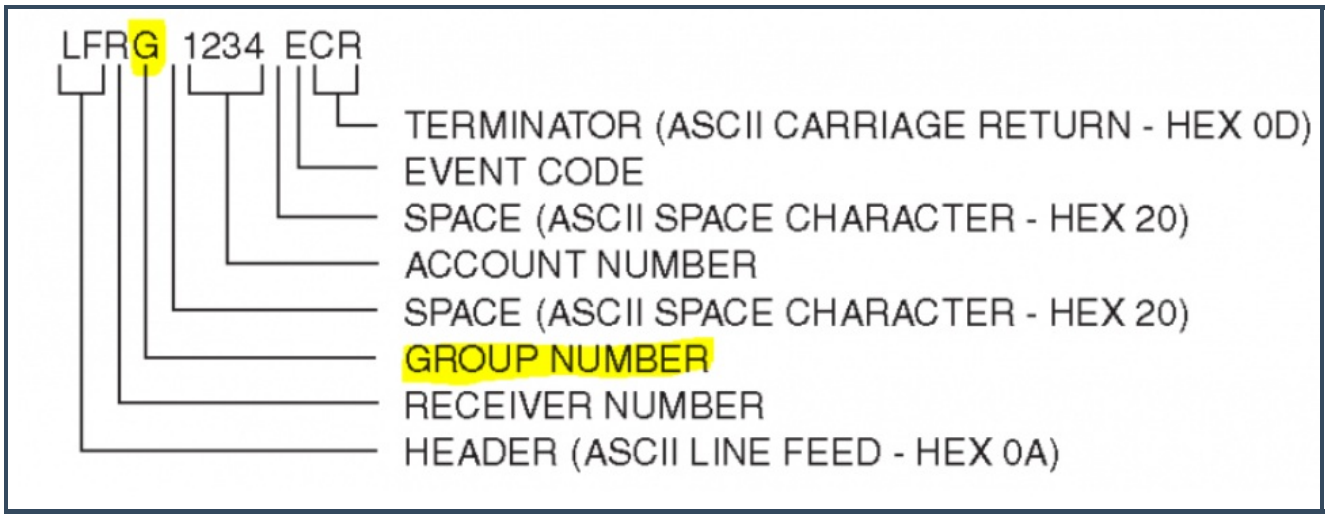
Update 3 (5/9/2019)

Issue: SaaS team found that the Services tab normally shows the status of the Inputs and Outputs, however, they recently added several outputs to the software. When they looked at the Services tab, no records were displayed. Deleting several outputs, eventually, the Input and Outputs are displayed again.
Solution: This tab has been reworked with a slider bar to display all of the records and their current status. Also, the buffer used in sending the status information has been increased in size. Finally, a change to the service name was made in a previous update. This was found to have the potential to cause problems on startup. Those have been resolved as well.

Update 4 (5/15/2019)

We have updated the Cloud Receiver to handle the Ademoco 685 protocol. Currently, we have not implemented the Polling where the FEP would send an "S" to the receiver. This means the FEP Receiver type should include the NOPOLL=1 option.

Also, the receiver should be configured to not expect this polling value.
This update does include a database script to add a receiver type into the TYPES table.
The routing value is set by the DNISStart and DNISLength fields in the database. Currently, it defaults to only route on the Group Number part of the signal. This is the G in the following example:



This does require a SQL update to add the type table entry for Ademco685. It can be run at any time even while the service is running BUT the service and UI should be restarted afterward.

Update 5 (8/20/2019)

In this update, we modified the Routing to accept leading zeros in the DNIS To/From range fields. We still do need these to be integer values so we can build a range but the leading zero (i.e. 01234) will remain. We also do a lookup and match with this zero. We also added some additional input validation. So if the type for a receiver specifies a DNIS length, we check to make sure your entry is not longer than that range. We also left the pad with zeros to make sure the input is that length. Finally, we made the Output field optional in the UI but do add the DNIS length check here as well.

Update 6 (2/27/2020)

Cloud team found that you could inadvertently create overlapping ranges with the same DNIS and Input path. We did not have client-side validation and the dB does not use a true primary key for this. We have added validation in the Client to account for this so that we don't have to adjust the dB schema. We will now check and not allow these to be created.

Update 7 (4/27/2020)

The Cloud team has been experiencing a problem where an input receiver will not reconnect after losing connection. The problem comes from the way TCP sockets behave. They do not let you know when the far side disconnects unless you read/write from the socket. We are just checking a method that tells you if there is data present so we do not read unless there is data. But this means we don't get the exception that would tell us the other side is gone. We don't want to read without knowing there is data for performance reasons.

I instead implemented a way to write a null (0x00) to the socket to check if it is open if we have missed two heartbeats. If we get the heartbeats we don't attempt this. That will let us balance performance and responsiveness to dropped connections.

Update 8 (9/21/2021)

The Cloud team has been experiencing a problem trying to work with COPS to send signals out of our cloud using a Bosch receiver. Also discovered a problem working with Security Partners.

1. Bosch is not sending the DNIS with follow on signals. We had coded for if they send the account number, use the last known DNIS. But this does not work for supervisory type signals. They use these so we change to where there is a configuration field on the types table that takes JSON structured values. This adds a {"DNISpersist": 60} setting where we use the last known DNIS for signals that don't have one for up to that value in seconds.

2. Their receiver in COPS does not like getting BOM or EOM messages. Again using the Configuration field on the Types table, we can now control this. The settings are {"SendBOM":"true","SendEOM":"true"}

3. They also require signals to come from specific receivers and lines. These may not be what the actual BOSCH in our cloud uses. We can now transform these again using the Configuration field on the Types table. The values are {"TransformRec":"99","TransformLine":"05"}

All of the values can be mixed/matched in one JSON structure. The service must be restarted after modifications to the Types table for these to be picked up.

Update 9 (4/1/2022)

Working with the cloud team, we discovered that some receivers/panels are sending in a single quote as part of the alarm signal. This typically would be ok and we intend to just pass this through for the automation to handle. But in this product was are storing these in SQL. That single quote is causing a SQL violation due to not being escaped. We had to add some string sanitization to escape these characters when they get passed in.

No configuration change is needed. Just the new files.

Update 10 (6/15/2022)

The Cloud Team identified some DMP signals that were being marked as BAD. Found that we were processing DMP multi-part messages correctly, but not single messages/Contact ID messages.

DMP Multi-part message (typical of the majority of signals we get from DMP receiver):

1- 1309 Zq\089\t "CL\u 00003"USER NAME 003 \a 002"INTERIOR \e "AC\na"5418897716*11064*\

DMP Single-part message also possible:

1- 582 c 0582 18 1 602 00 000 2 A4356581813*11064*

Modified the CloudReceiver Service to properly handle and locate the ANI/DNIS info in both multi-part and single-part messages from DMP. Also, per RIP-193-10357, DMP ANI/DNIS separator character can also be '#' (as opposed to the usual '*'), so this possibility is accounted for now as well in the CloudReceiver Service DMP processing.

No configuration change is needed. Just the new files.