OEM FRU TECHNICAL WHITE PAPER

Capabilities and Use Cases of the OEM FRU Storage Feature on Dell EMC PowerEdge Servers

ABSTRACT

This white paper describes the capabilities of the OEM FRU storage feature and how OEM customers can use the storage area to their benefit.

July, 2017

TABLE OF CONTENTS

EXECUTIVE SUMMARY

Every modern component of a computer or electronic equipment, commonly referred to as a Field Replaceable Unit or FRU, contains a memory block that stores the inventory information of that component. This includes the manufacturer's name, product name, manufacture date, serial numbers and other details that help identify the component.

The Intel FRU Information Storage for IPMI specification defines the standard format that devices are expected to conform to within their FRU areas. Dell EMC PowerEdge servers leverage this format across the board, from PCIe controllers to power supplies to the chassis itself. Each component vendor populates the FRU area during their manufacturing process and all FRU areas are easily accessible via IPMI.

Dell EMC servers are often used as components to build appliances by customers who then sell their solution to their end customers. In such a use case, the appliance builder (or OEM customer for the rest of this white paper) can greatly benefit from the OEM FRU capability which provides a dedicated FRU information storage area that they can populate with their own company-specific serial information for inventory and tracking purposes.

AUDIENCE

The OEM FRU feature is primarily targeted at OEM customers who use Dell EMC servers as components within their solution that then gets sold to their end customers.

This white paper is relevant to those interested in how this feature works and can be leveraged. It is also addressed at a technical audience who would need to actually build and deploy the OEM FRU information on a server during factory deployment.

Lastly, it will also be relevant to product developers who could use the FRU area for purposes beyond tracking, as well as service and support professionals who would use this information in troubleshooting situations.

INTRODUCTION

Dell EMC PowerEdge servers are built using a multitude of components sourced from various Dell EMC teams and vendors. Each component uses the common FRU format maintained by Intel to track various pieces of information such as manufacturer name, date, serial numbers, MAC addresses and other details that do not change once shipped from the factory. This helps the consumer of the component to identify who the manufacturer is, and allows the vendor of a component to track the exact history of a component if failures are reported.

The server itself, which is a conglomeration of such individual components, has its own FRU area which identifies the manufacturer and other details such as service tag and manufacture date. This helps Dell EMC as a manufacturer to track the server as a whole, beyond the individual components.

The OEM FRU storage feature of Dell EMC PowerEdge servers is an additional FRU area that allows OEM customers, who use Dell EMC servers as a component of their solution, to include their own tracking information in the FRU storage area. This can be loaded into the server during factory deployment and can be accessed when the information is required during troubleshooting or support.

This allows the OEM customers to store their own part numbers and track information within the server, which enables them to track their solutions in their internal management systems. This is similar to the way Dell EMC servers use the standard FRU areas to store Dell specific information such as service tags and manufacture date and use that information when having to identify and support those systems once in the field.

OBJECTIVE

The purpose of this white paper is to:

- Introduce the OEM FRU area to customers
- Provide recommendations on the structure and contents of this area

- Provide steps to build the OEM FRU data structure
- Provide commands to read and write the OEM FRU onto a server
- Provide commands to edit the OEM FRU on the fly
- Provide the recommended factory deployment workflow
- Discuss additional use cases for leveraging the OEM FRU

PREREQUISITES

The OEM FRU storage feature is a capability of the Integrated Dell Remote Access Controller or iDRAC service processor and is available on 13G+ servers shipping with an iDRAC firmware level of 2.40.40.40 or later.

The OEM FRU capability is a Premium tier feature of Dell EMC Identity Module (ID Module) and requires an OEM engagement to build and deploy. Details on the steps involved in enabling the OEM FRU storage feature within ID Module are described in a later section. Visit www.dellyourid.com for more details on ID Module.

Considering that the FRU area is a binary payload, it is not trivial to build the content structure by hand. In order to simplify the effort for OEM customers, a Python tool is provided to speed up the process of creating the payload. This tool can be downloaded from Dell Tech Center here. The FRU tool has been tested with Python version 2.7 and 3.5 which can be downloaded and installed from www.python.org.

In order to write, read or edit the OEM FRU storage area, the open source IPMItool utility is required. This utility can be installed on Linux distributions by using the built-in package manager such as yum or apt-get. Dell provides a Windows version which can be found in the *Driver and Downloads* section for any PowerEdge server on Dell Support under the *Systems Management* section. It is contained in the package named *Dell OpenManage BMC Utility* which can also be found on Google by searching for the package by name. For documentation on IPMItool, search for “man ipmitool” on Google.

OVERVIEW

The OEM FRU follows the standard structure for an FRU information area as defined by Intel. This specification documents the Platform Management FRU Information Storage Definition and was originally released in 1998. It was last updated in 2013 to v1.2 as linked above.

Each OEM can choose to customize field lengths and extend Internal Area, for example, but is generally expected to leverage the standard structure. In order to describe the structure, some portions of Platform Management FRU Information Storage Definition are captured here for convenience.

The FRU has the following defined areas:

- Common Header
- Internal Area
- Chassis Info Area
- Board Info Area
- Product Info Area
- MultiRecord Area

The most relevant area for a typical OEM customer would be the Product Info Area.

The Dell OEM FRU implementation provides 1024 bytes of storage space. While most of the fields in the information areas below allow variable string lengths, the overall size of the FRU would need to be kept within this 1024 byte limit. The Intel specification provides some guidance in section 6 and 7 regarding the storage organization.

COMMON HEADER

This header is used to define which sections of the FRU are populated and is the starting point for populating and parsing the information. It does not contain any string fields that need to be selected, but it needs to be set up correctly to reflect which areas are being used.

The FRU tool described later in this white paper automatically generates the common header definitions based on the INI file definitions.

INTERNAL AREA

Internal Area is typically used to hold private data for the FRU device or a Management Controller and does not follow any prescribed format. The OEM is free to choose any format. Within the scope of the OEM FRU, it can be used to store any appliance specific details such as licensing information.

CHASSIS INFO AREA

Chassis Info Area is reserved to describe the overall chassis that contains all components within a product.

The relevant fields within the Chassis Info Area that could be populated by an OEM customer are as follows:-

- Chassis Type (Possible values are defined in the SMBIOS specification v3.0.0 under table 17. A typical value for Dell EMC PowerEdge rack mount servers is 17h = Rack Mount Chassis.)
- Chassis Part Number
- Chassis Serial Number
- Custom Chassis Info Fields (These are additional fields that can be populated with any information pertaining to the chassis that does not fall into the above three fields.)

BOARD INFO AREA

The Board Info Area is most commonly populated for FRU devices and contains the typical details that pertain to the manufacture of the FRU device. This includes:-

- Date of Manufacture
- Manufacturer’s Name
- Product Name
- Serial Number
- Part Number
- FRU File ID
- Custom Board Info Fields

PRODUCT INFO AREA

The Product Info Area contains very similar fields to the Board Info Area, but are more relevant for an OEM customer. The fields are:-

- Manufacturer’s Name

- Product Name
- Product Part/Model Number
- Product Version
- Product Serial Number
- Asset Tag
- FRU File ID
- Custom Product Info Fields

MULTIRECORD AREA

MultiRecord Area provides an extension area that can go beyond the standard areas defined in the specification. OEM customers are free to leverage this area if it meets their requirements, but the structure and use cases of this area are out of scope of this document. The FRU tool also does not support generating this content.

DEVELOPMENT

The following section provides details on using the Python FRU tool made available to OEM customers along with this white paper to speed up the process of creating the OEM FRU binary payload.

INI FILE STRUCTURE

The FRU tool requires an INI file as an input in order to generate the binary FRU structure. Sections of this INI file are captured below with notes explaining how the fields are to be used. It follows the same general structure of the FRU as defined earlier.

COMMON HEADER

```
[common]

; Version of FRU spec being adhered to – always set to 1
version = 1

; Size of FRU – the OEM FRU space is 1KB in size
size = 1024

; Is Internal Use Area section to be populated
;   Tool only parses the [internal] section if this bit is set
internal = 0

; Is Chassis Info Area to be populated
;   Tool only parses the [chassis] section if this bit is set
chassis = 0

; Is Board Info Area to be populated
;   Tool only parses the [board] section if this bit is set
board = 0

; Is Product Info Area to be populated
;   Tool only parses the [product] section if this bit is set
product = 0
```

```
; Is Multi-Record Info Area to be populated
;   Not implemented in Python tool as of today
multirecord = 0
```

INTERNAL AREA

```
[internal]

; Data entered here will directly go into Internal Use Area
data =

; Either/or - not both data and file
; Contents of file mentioned will go into Internal Use Area
file =
```

CHASSIS INFO AREA

```
[chassis]

; Hexadecimal values defined in the SMBIOS specification v3.0.0 under table 17
type = 17

; Chassis part number
part =

; Chassis serial number
serial =

; Custom fields pertaining to the chassis
extra1 =
extra2 =
extra3 =
```

BOARD INFO AREA

```
[board]

; Hexadecimal values defined in table 15-1 of the Intel FRU specification
language = 0

; Date of manufacture of the board
; 3 bytes - #minutes since Jan 1, 1996 00:00:00
date =

; Board manufacturer name
manufacturer =

; Board product name
product =

; Board serial number
serial =

; Board part number
part =

; File that was used during manufacture or field update to load the FRU information
field =
; Custom fields pertaining to the board
extra1 =
extra2 =
extra3 =
```

PRODUCT INFO AREA

```
[product]

; Hexadecimal values defined in table 15-1 of the Intel FRU specification
language = 0

; Product manufacturer name
manufacturer =

; Name of product
product =

; Product part number
part =

; Product version
version =

; Product serial number
serial =

; Product asset tag
asset =

; File that was used during manufacture or field update to load the FRU information
field =

; Custom fields pertaining to the product
extra1 =
extra2 =
extra3 =
```

BUILDING THE FRU BINARY

Once the INI file has been filled out as required for the solution, the FRU tool can be used to convert it into BIN format which can then be deployed onto a server for test purposes.

The FRU tool has a simple syntax:-

```
> python fru.py INIFILE BINFILE [--force] [--cmd]

  INIFILE = input INI file as described above
  BINFILE = output BIN file that can be programmed into server
  --force = overwrite BIN file if it already exists
  --cmd   = print sample IPMItool commands as a reference
```

For example, let’s create an INI file with `common.product = 1` and `product.manufacturer = Widgets Inc.` and generate the BIN file with the `--cmd` flag to show IPMItool command examples. It is important to enable the product area in the `[common]` section before the tool includes it in the BIN file. If this step is omitted, the tool assumes that the section is to be omitted.

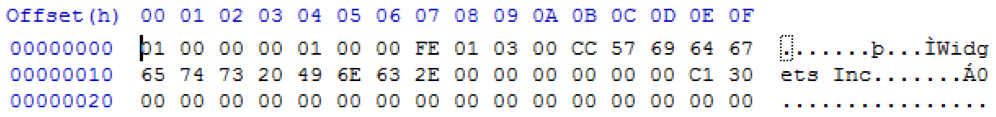
```
> python fru.py inifile.ini binfile.bin --cmd
```

```
[Product]
; ipmitool -I lanplus -H %IP% -U root -P password fru edit 17 field p 0 123456789012
0: manufacturer = b'Widgets Inc.' (12)
```

The tool generates the BIN file and displays what fields were added to it, along with the field length. It also displays the IPMItool syntax (due to `--cmd`) that can be used to edit each field once the BIN file is programmed on a server. This is useful since the typical use case is to create a template BIN which can be created once using the FRU tool and then programmed on each server, followed by editing all server unique fields as a second step. The IPMItool fru edit commands displayed by the FRU tool show how to edit these unique fields during deployment

Note: Once the fields are programmed onto a server, their lengths cannot be changed. In the above case, the `Widgets Inc.` can only be replaced by a string of 12 characters. This will need to be kept in mind when creating a BIN file template. Blank spaces can be used to pad strings to the required lengths within the INI file, enclosed in quotes.

The resulting BIN file looks as follows:-



```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000   01 00 00 00 01 00 00 FE 01 03 00 CC 57 69 64 67  .......þ...ÌWidg
00000010   65 74 73 20 49 6E 63 2E 00 00 00 00 00 00 C1 30  ets Inc.......Á0
00000020   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

This BIN can be programmed onto each server using the IPMItool fru write command or can be injected using Dell EMC ID Module. Server unique fields will still require additional IPMItool fru edit commands in the factory process.

DEPLOYING THE FRU

Once the BIN file has been generated, it can be programmed onto a server as follows:-

```
> ipmitool -I lanplus -H %IP% -U root -P password fru write 17 binfile.bin

Fru Size         : 1024 bytes
Size to Write    : 1024 bytes
```

Note: The OEM FRU has a FRU ID = 17. The OEM FRU has a maximum size of 1024 bytes and so the FRU tool creates a BIN file of size 1024 bytes, padded with blank data. This size of 1024 is defined in the INI file so the FRU tool can be reused for FRU development beyond the OEM FRU on PowerEdge servers if needed.

The portion highlighted in red is to run IPMItool on a remote server. The IP address is the address of the iDRAC. The -U and -P are the login and password for the iDRAC user with admin access.

If running locally on the target server, it can be replaced with `-I wmi` on a Windows OS for example. Use `IPMItool -I` to identify what interfaces are available.

In order to run IPMI commands remotely, the IPMI over LAN capability needs to be enabled on the iDRAC. This can be done via the BIOS F2 settings, the iDRAC GUI under iDRAC Network Settings or via the RACADM CLI as below:-

```
> racadm -r %IP% -u root -p password set iDRAC.IPMILan.Enable Enabled
```

The portion in red can be omitted if Racadm is being run locally on the target server.

After the OEM FRU data has been programmed, it can be printed out using the IPMItool print command:-

```
> ipmitool -I lanplus -H %IP% -U root -P password fru print 17

Product Manufacturer  : Widgets Inc.
```

This OEM FRU BIN was configured with only the Product Manufacturer field and it is displayed accordingly. If the FRU ID 17 is omitted in the command line, all the available FRUs are printed.

In order to read back the entire OEM FRU content, including data that isn’t printable such as any data in the Internal Area, use the IPMItool fru read command. It creates a file with the same binary structure as was generated by the FRU tool.

```
> ipmitool -I lanplus -H %IP% -U root -P password fru read 17 readback.bin

Fru Size         : 1024 bytes
Done
```

Writing, printing and reading back the OEM FRU can be done on any PowerEdge server by using IPMItool. The contents do not persist through an iDRAC powercycle unless the feature is enabled by using ID Module.

EDITING FRU FIELDS

After a BIN file has been programmed onto a server, programmed string fields can be modified at any time by using the IPMItool fru edit command. The FRU tool displays sample commands when the `--cmd` flag is used while building the BIN payload.

```
> ipmitool -I lanplus -H %IP% -U root -P password fru edit 17 field p 0 123456789012

Updating Field 'Widgets Inc.' with '123456789012' ...
```

On printing back the OEM FRU, the change is reflected:

```
> ipmitool -I lanplus -H %IP% -U root -P password fru print 17

Product Manufacturer  : 123456789012
```

By using the IPMItool fru edit command, any unique field such as serial numbers can be programmed as required on a per server basis. This needs to be done as part of the factory process.

PRODUCTION READY

ID MODULE ENGAGEMENT

ID Module is an iDRAC capability that enables customization of various aspects of PowerEdge servers such as branding (logos and SMBIOS strings), Microsoft OS activation, BIOS and iDRAC custom defaults and other advanced tweaks as required by an OEM configuration. The www.dellyourid.com website describes these features and enables an OEM to configure the Express and Professional tier customizations by using a web browser.

The OEM FRU storage feature is a premium tier customization capability of ID Module. The OEM FRU content can be developed, generated and tested on any 13G+ PowerEdge server by following the procedure described in this white paper. In order to enable the OEM FRU capability in production servers, the feature needs to be enabled by using ID Module.

The OEM FRU capability is designed to persist across iDRAC resets and resetting to default settings. This ensures that any content programmed in the OEM factory process is retained across any field deployment scenarios. While OEM FRU continues to remain writable at all times, it is only altered when it is directly written to or edited, not as a side effect of other actions. The exception to this is the installation of a new ID Module. This causes any existing content to be replaced by the default OEM FRU contents of the installed ID Module or deleted altogether if the feature is disabled.

The Premium tier requires additional engagements and is best handled by working with your Dell EMC Sales representative through the ID Module development process.

FACTORY RECOMMENDATIONS

As part of enabling the OEM FRU capability, the ID Module payload can be configured to include either a default blank OEM FRU or the OEM FRU BIN file specifically developed for your solution. Depending on the FRU content and factory process, multiple approaches are possible.

As discussed earlier, the OEM FRU area can be used to store generic values such as Manufacturer Name and Part Number, which are fixed and don't change on a per server basis. Serial Numbers and similar fields, however, need to be unique per system. If the OEM FRU contents for your solution have no server specific content, it might be easiest to create the BIN file and have it included within ID Module.

If there are instances of unique content in the FRU or you want to allow easy change of the FRU structure over time, Dell EMC recommends that you have a default blank OEM FRU defined in the ID Module and use one of the two approaches within the factory to deploy the actual FRU content on each shipping system:

1. Build and deploy a unique BIN per system during factory process

OR

1. Build a generic BIN template for the product with placeholders for unique fields
2. Deploy a generic BIN template on each system during factory process
3. Change the placeholder values to system specific values during factory process

Both approaches are available, depending on the capabilities of the factory process. However, the latter method reduces the tool dependencies in the factory.

The first method requires a process to generate the INI file (since OEM FRU content is system specific) and Python to create the BIN with the FRU tool. In addition, IPMItool is required to write the BIN to the server.

The second method requires only IPMItool to write the BIN and edit the system specific fields. The INI file can be hand authored. The BIN file can be generated separately during development and it removes the dependency of running Python in the factory.

BEYOND INVENTORY AND TRACKING

We have seen how the OEM FRU storage area enables OEM customers to effectively inventory and track their products built on Dell EMC PowerEdge servers. However, this storage space built into the hardware also allows for other interesting use cases.

Given the OEM FRU has 1024 bytes of space available, it is possible to use the Internal Area section to store extended information beyond what is defined within the FRU specification. One example would be to store configuration data that is application specific. An OEM application running on the host could load this content and change its behavior based on the configuration contained within that file.

Another example is to use this space to hardware activate an application running on the server similar to how Microsoft enables OEM activation for its Windows operating systems. This would allow an OEM to license the software that runs on the server without having to distribute and manage license keys to their end customers.

The application would use the IPMItool fru read command to pull the FRU contents, extract the Internal Area section and use it to change its behavior or limit the functionality. The content could be signed by using Public Key cryptography to verify the authenticity of the content and be tied to the server by using unique fields such as Service Tag, serial numbers or MAC addresses to ensure that the license cannot be moved from one server to another.

Such a hardware activation mechanism could be achieved as follows:

FACTORY PROCESS

1. Create an application license file based on what the customer purchased, such as Bit mask, INI file, JSON, and XML. The contents of the file should instruct the application to enable/disable specific features or feature tiers.

2. Add server unique identity fields into the license file.

    - The IPMItool print command lists all FRUs on the system

    - Relevant serial numbers could be included in the license file

3. Sign the license file with OEM private key and append to the license file.

4. Ensure that the total file size is under 1k.

5. Create an INI file setting `common.internal = 1` and `internal.file = filename`

6. Include definitions for any other FRU areas or fields if required.

7. Generate the BIN file by using the FRU tool.

8. Inject the BIN file into the server by using the IPMItool fru write command.

9. Edit any server unique fields in OEM FRU by using the IPMItool fru edit command if required.

APPLICATION VERIFICATION PROCESS

Every time the OEM application starts up, it will:

- Read all server FRU information by using the IPMItool fru print command.

- Read OEM FRU by using IPMItool fru read command.

- Extract the license file within the Internal Area from the BIN file.

- Verify the signature with the public key embedded in the OEM application.

- Verify that all serial numbers read from server match contents of license file.

- Load the license level from the file and adjust the OEM application features accordingly.

If any of the steps fail, the application can assume that the server is not licensed to run and can fall through to a base license or refuse to run altogether as required.

Note: If any hardware component is changed due to a bad part or server upgrade, the activation will fail. This can be managed by limiting the type of components that are relied on or by allowing minor changes before causing activation to fail altogether (e.g. RAID controller was changed but everything else still matches on the server so allow application to still run at license level).

CHANGE MANAGEMENT

As part of storing hardware activation information in the OEM FRU, the OEM might need to enable some of the features below to provide a smooth customer experience. Failure conditions such as Re-activation or Missing OEM FRU described below will likely be needed for customer satisfaction.

RE-ACTIVATION

If for some reason, activation fails, re-activation could be accomplished as below:

- Technician on site or end customer to:

    1. Run IPMItool fru print to get server information

    2. Run IPMItool fru read to get existing OEM FRU BIN

    3. Send both results to OEM for verification

- OEM to:

    1. Verify the validity of re-activation scenario:

        i. Compare IPMItool fru print contents with license file contents

        ii. Correlate with sales database to verify the license sold on that server

    2. Extract the license file within Internal Area from FRU BIN

    3. Update the server specific info within the license file with IPMItool fru print results

    4. Sign the license file with the OEM private key and append it to the license file

    5. Create a new INI file by using IPMItool fru print results and the new license file created above

    6. Generate a new BIN file by using the FRU tool

    7. Send the new FRU BIN to the technician or customer on site

- Technician or customer to:

    1. Write new FRU BIN onto the server by using IPMItool fru write

    2. Verify that the application is now activated

BAD OEM FRU

The steps for handling this scenario will be very similar to the re-activation scenario. If the OEM FRU is lost or corrupted altogether, the OEM application should be able to distinguish various failure scenarios depending on the contents of the OEM FRU:

- OEM FRU is blank
- OEM FRU contains mismatched server serial numbers
- License file is not signed or has a bad signature
- OEM FRU is completely corrupt

If any of these failures occur, the technician would need to go through additional steps to verify that the server is actually licensed within the sales database. This is because the contents of the OEM FRU cannot be trusted and other information such as platform model or service tag might have to be used to correlate the license to the server.

If the content is blank, it is also important to verify that the system actually has the ID Module enabling the OEM FRU capability without which the content will not be persistent and be restored to default. This can be done by checking the system inventory for “Identity Module”. For example, using the Racadm CLI:

```
> racadm -r %IP% -u root -p password swinventory | grep "Identity Module"

ElementName = Identity Module
```

MIGRATION

If the license needs to be migrated to another server altogether, the re-activation steps above can still be used. However, the IPMItool fru print data would have to come from the new server, whereas the OEM FRU BIN file would come from the original server.

In order to make the verification more robust, the technician or end customer could be asked to send the IPMItool fru print contents of the original server as well. This will allow the OEM to verify that the license file within the OEM FRU BIN actually matches the server it was running on before approving the migration.

UPGRADES

If a customer is allowed to upgrade or downgrade their OEM application license, the OEM FRU content would have to be updated. It would be yet another variation of the re-activation scenario, except the IPMItool fru print contents will be for the same server, but it is the license file contents that need to be changed depending on what the customer purchased.

CONCLUSION

This white paper aims at introducing the OEM FRU feature and how it can be utilized by OEM customers. For more information, please feel free to contact a Dell EMC OEM representative who can help you on how Dell EMC PowerEdge servers can be leveraged and further customized.