prpl Secure Manufacturing Data Standard

PRPL-SMD001

REVISION HISTORY

Version	Date	Authors	Description
0.1	August 2022	Vincent Harle (Author), Onur Zengin (Comments), Matthias Hofman (Comments)	Initial Version from Confluence
0.2	August 2022	Brendan Black (Editing)	Reformatting & general edits, Acronyms, Terms, References
0.3	October 2022	Vincent Harle (Editing)	Added more standard fields Additional edits for clarity following Onur Zengin comments
1.0 pre release	October 2022	Brendan Black	Initial Release Version for comment
1.0	January 2023	Wouter Cloetens (comments) Vincent Harle (editing) David Barr (editing)	Final Release Version
1.1	June 2023	Wouter Cloetens	Added XDSL ITU parameters

1 Contents

2 Important Notices, IPR Statement, Disclaimer, And Copyright

This chapter contains important information about PRPL and this document (hereinafter 'This PRPL Document').

2.1 About PRPL

The prpl Foundation (PRPL) is a not-for-profit organization which publishes documents including, but not limited to, Requirements, Specifications, Recommendations, API application programming interfaces, and Test Plans.

2.2 This may not be the latest version of this PRPL document

This PRPL Document is the output of the Working Groups of the PRPL and its members as of the date of publication. Readers of This PRPL Document should be aware that it can be revised, edited or have its status changed according to the PRPL working procedures.

2.3 There is no warranty provided with this PRPL document

The services, the content, and the information in This PRPL Document are provided on an "as is" basis. PRPL, to the fullest extent permitted by law, disclaims all warranties, whether express, implied, statutory or otherwise, including but not limited to the implied warranties of merchantability, non-infringement of third-parties rights and fitness for a particular purpose. PRPL, its affiliates and licensors make no representations or warranties about the accuracy, completeness, security or timeliness of the content or information provided in the PRPL Document. No information obtained via the PRPL Document shall create any warranty not expressly stated by PRPL in these terms and conditions.

2.4 Exclusion of liability

Any person holding a copyright in This PRPL Document, or any portion thereof, disclaims to the fullest extent permitted by law (a) any liability (including direct, indirect, special, or consequential damages under any legal theory) arising from or related to the use of or reliance upon This PRPL Document; and (b) any obligation to update or correct this technical report.

2.5 THIS PRPL DOCUMENT IS NOT BINDING ON PRPL NOR ITS MEMBER COMPANIES

This PRPL Document, though formally approved by the PRPL member companies, is not binding in any way upon the PRPL members.

2.6 INTELLECTUAL PROPERTY RIGHTS

Patents essential or potentially essential to the implementation of features described in This PRPL Document may have been declared in conformance to the PRPL IP Policy (available at the PRPL website: www.prplFoundation.org).

2.7 COPYRIGHT PROVISIONS

This PRPL Document is copyrighted by PRPL, and all rights are reserved. The contents of This PRPL Document are protected by the copyrights of PRPL or the copyrights of third-parties that are used by agreement. Trademarks and copyrights mentioned in This PRPL Document are the property of their respective owners. The content of This PRPL Document may only be reproduced, distributed, modified, framed, cached, adapted or linked to, or made available in any form by any means, or incorporated into or used in any information storage and retrieval system, **with the prior written permission of PRPL or the applicable third-party copyright owner**. Such written permission is **not** however required under the conditions specified in Section 2.7.1 and Section 2.7.2:

2.7.1 INCORPORATING PRPL DOCUMENTS IN WHOLE OR PART WITHIN DOCUMENTS RELATED TO COMMERCIAL TENDERS

Any or all section(s) of PRPL Documents may be incorporated into Commercial Tenders (RFP, RFT, RFQ, ITT, etc.) by PRPL and non-PRPL members under the following conditions:

(a) The PRPL Requirements numbers, where applicable, must not be changed from those within the PRPL Documents;

(b) A prominent acknowledgement of the PRPL must be provided within the Commercial document identifying any and all PRPL Documents referenced and giving the web address of the

PRPL;
(c) The Commercial Tender must identify which of its section(s) include material taken from PRPL Documents and must identify each PRPL Document used, and the relevant PRPL Section Numbers; and,
(d) The Commercial Tender must refer to the copyright provisions of PRPL Documents and must state that the sections taken from PRPL Documents are subject to copyright by PRPL and/or applicable third parties.

2.7.2 Copying this PRPL Document in its Entirety

This PRPL Document may be electronically copied, reproduced, distributed, linked to, or made available by other means, or incorporated into or used in any information storage and retrieval system, but **only in its original, unaltered PDF format**, and with its original PRPL title and file name unaltered. It may not be modified without the advanced written permission of the PRPL.

3 Acronyms

3.1 Acronyms

AES	Advanced Encryption Standard - a cryptographic block cipher
CBC	Cipher Block Chaining - an encryption mode where each block is XORed with the previous ciphertext block before being encrypted
DTB	Device Tree Blob - also known as Flattened Device Tree Format (FDT)
LL-API	Low Level Application Programming Interface
MAC	Media Access Control
OEM	Original Equipment Manufacturer
OUI	Organisational Unique Identifier - The first three sets of two hexadecimal numbers in a MAC Address which identifies the card manufacturer
PKCS	Public Key Cryptography Standards - a group of public-key cryptography standards devised and published by RSA Security
PSS	Probabilistic Signature Scheme - a cryptographic signature scheme formalised as a part of PKCS#1 v2.1

RSA	Rivest-Shamir-Adleman - a public key cryptography system
SHA	Secure Hash Algorithms - a family of cryptographic hash functions
SW	Software
TR-069	Broadband Forum Technical Report 069: CPE WAN Management Protocol. [12.2.a]
TR-181	Broadband Forum Technical Report 181: Device Data Model for TR-069 [12.2.b]
XOR	Exclusive or - a bitwise logical operation applied to data where the result is true if and only if its arguments differ

3.2 Terms

3.3 Definitions of requirements terms

The definitions of MUST and SHOULD in this document are as follows:

MUST: A functional requirement which is based on a clear consensus among PRPl Service Provider members and is the base level of required functionality.

MUST NOT: This function is prohibited by the specification.

SHOULD: Functionality which goes beyond the base requirements and can be used to provide vendor product differentiation.

Note: these definitions are specific to the PRPL and should not be confused with the same or similar terms used by other bodies.

4 Introduction

More and more sensitive data (e.g., Certificates and Keys) are embedded in devices during the manufacturing process.

This data needs to be protected against forgery (signature) and also sometimes from leaking out from simple flash analysis (ciphering).

Current state of the art currently relies on specific proprietary formats with closed source drivers required. A common Low-Level API to grant access to manufacturing data will reduce SW efforts when porting.

Note that this document focuses only on the format itself and the way it is handled. It assumes that various public keys and encryption keys are known to the overall embedded system. A global prpl key management system is to be described in the prpl secure bootloader documentation.

5 Use Cases

- Store specific devices information available at manufacturing time: MAC address, Serial Number, certificates...
- Prevent device spoofing: need to ensure authenticity of specific data, like MAC address or Serial Number
- Protect sensitive information by encryption. As an example, device private key should be protected from extraction by a hacker (not stored in clear on flash)
- Store keys and certificates to be used during runtime in a secure manner
- Access of manufacturing data from the bootloader stage

6 Requirements

ID	Item	Description
MD1	Open format	The format used should be public. Security MUST not rely on obfuscation.
MD2	Evolution	It SHOULD be possible to extend the format easily to add new field or even field types
MD3	Extensibility	Format SHOULD NOT restrain the number of parameters nor their sizes.
MD4	Support data authenticity	it MUST be possible to check the authenticity of a selection of fields
MD5	Support data ciphering	it MUST be possible to protect information through ciphering
MD6	Authenticity from different entity	Format SHOULD allow multiple key owners to authenticate their own field.
MD7	Not modifiable during runtime	The manufacturing data MUST NOT be modified during runtime, it MUST NOT be extended during runtime. Any change would only be taken into account on the next boot.
MD8	Refurbishment	it MUST be possible to update manufacturing data content from a refurbishment process
MD9	Support in bootloader a Linux environment	The format of the manufacturing data MUST be accessible in the prpl bootloader (u-boot) as well as in the Linux environment.

7 Description of Format on Device

7.1 Device tree structure

Device tree is the format of choice for storing the information:

- It is an open standard already widely used https://www.devicetree.org/
- easy to work with: many tools available. easy conversion between text/binary
- can be easily extended with nodes and fields
- supported by u-boot

See chapter [12.1] for references.

7.2 Manufacturing data structure

Here the device tree structure for secure manufacturing data:

```
/dts-v1/;
/{
	MFG_DATA {
		MFG_DATA_version = "<version>;
		MFG_DATA_date = "<unix_time_stamp>;
		ITEM_UNSECURE_ID_00 {
			type = "string";
			value = "SomeString";
		};
		<ITEM_SECURE_ID_01> {
			type = "raw";
			value = "<AA BB CC DD EE FF 00 11 22 33 44 55 66 77 88 99>";
			cipher {
				key_index = <0x01>;
				format = "key_format";
			};
			signature = {
				key_index = <0x10>;
			};

		};
...
		<ITEM_ID_XX> {
			type = "raw";
			value = "<AA BB CC DD EE FF 00 11 22 33 44 55 66 77 88 99>;
		};
		Signature {
			key_index = <0x10>;
```



```
                algo = "<sign-algo>";
                type = "raw";
                value = "<hexadecimal signature hash>";
          };
          Signature_1 {
                key_index = <0x11>;
                algo = "<sign-algo>";
                type = "raw";
                value = "<hexadecimal signature hash>";
          };

     };
};
```

7.3 Global nodes and fields

MFG_DATA: main root node

MFG_DATA.MFG_DATA_version: Version of the format.

MFG_DATA.MFG_DATA_date: timestamp of manufacturing data

7.4 Item nodes

Every data item would have its own node under MFG_DATA node.
The node name is the item identifier.
Mandatory properties are:

- type: ether string or raw type are currently supported
- value: actual value of the field

Optional child nodes:

- cipher: for encrypted data to point to proper decryption information
- signature: used for field requiring authentication

7.4.1 Item cipher sub node

This node is optional. When it is present, it means that the item value is ciphered and of raw type.

Properties are:

- key_index: index of the ciphering key in the system

- format: identify algorithm used for ciphering in openssl like notation.

Note that it is assumed that the system properly matches a known key from the key_index parameter.

7.4.2 Item signature sub node

This node is optional. When it is present, it means that the item is part of authenticated fields.

Properties are:
- key_index: identifier for the signing key to be used to check the authenticity of the item value. See signature nodes.

7.5 Signature nodes

Those nodes are used to verify the authenticity of a secured field. Secure field typically contains confidential/sensitive data like a private key for certificates or passwords, or integrity protected data like the serial number of the device.

As required multiple key owners may sign different sets of parameters. So multiple signature nodes can be present, each with a unique index linking it to the fields that it authenticates for. Each field that requires authentication has a signature sub-node with corresponding index.

Signature is computed by concatenating the fields for a specific signature index group and signing the concatenated data using a specified algorithm. In that sense you need to pay attention that tools used do not change the order of field nodes while processing.

Note that for ciphered fields, the ciphered value is used to allow checking authenticity without prior need to decipher data.

Properties are:

- key_index: identifier of the signature group.
- algo: signature algorithm to be used. For instance, "sha256-rsa-pss".
- type: type of signature for future evolution. only raw is supported.
- value: actual computed signature that is used to check the authenticity.

Note that it is assumed that the system properly matches a known key from the key_index parameter.

8 prpl Standard Fields Description

Some fields are common to all devices, and used to expose TR-181 DeviceInfo data model or set basic

Here are the standard fields in prpl secure manufacturing data structure:

- Mandatory fields (**M**) must be present to ensure proper operation.
- Recommended fields (**R**) should be present if target feature is available, unless another custom way of provisioning is defined (for instance by USP controller or by custom algorithm)
- Optional fields (**O**) are for data that should be part of manufacturing data but requires a custom handling (e.g. calibration data).

Also note that, format being open, it remains possible to add any custom non standard field to the manufacturing data.

Item identifier	**Type**	**M/R/O**	**Usage**
BASE_MAC_ADDRESS	Raw (6 Bytes)	M	Base MAC address of the device to be used by bootloader and user space as primary MAC address for LAN
SERIAL_NUMBER	string	M	Serial number of the device. Exposed in TR-181 DeviceInfo.SerialNumber (see [12.2.b]).
PRODUCT_CLASS	string	M	Class of product. Exposed in TR-181 DeviceInfo.ProductClass (see [12.2.b]). It is recommended not to use space in this field as some tooling won't handle it properly.
MODEL_NAME	string	M	Hardware device model. Exposed in TR-181 DeviceInfo.ModelName (see [12.2.b]).
MANUFACTURER_OUI	Raw (3 Bytes)	M	Manufacturer OUI. Exposed in TR-181 DeviceInfo.ManufacturerOUI (see [12.2.b]).
MANUFACTURER	string	M	Manufacturer identifier. Exposed in TR-181 DeviceInfo.Manufacturer (see [12.2.b]).
HARDWARE_VERSION	string	R	Hardware revision. Exposed in TR-181 DeviceInfo.HardwareVersion (see [12.2.b]).

Item identifier	Type	M/R/O	Usage
			It is highly recommended this string is incremental and allows ASCII string comparison to sort versions in order. (i.e use 01.02 instead of 1.2 for it to be inferior to 01.10)
WLAN_SSID	string	R	Default SSID for Wi-Fi Access Point. Initial value for TR-181 WiFi.SSID.*.SSID (see [12.2.b]).
WLAN_PASSPHRASE	string	R	Default password for Wi-Fi Access Point. Initial value for TR-181 WiFi.AccessPoint.*.Security.KeyPassPhrase (see [12.2.b]).
WLAN_REGDOMAIN	string	O	Wi-Fi regulatory domain to use if tied to a specific country. Initial value for TR-181 WiFi.Radio.1.RegulatoryDomain (see [12.2.b]).
DECT_RFPI	Raw (5 Bytes)	R	DECT Radio Fixed Part Identity (RFPI)
PON_SERIAL	Raw (8 bytes)	R	ONUSerial Number presented to the OLT, as defined in G.984.3 with 4 bytes ASCII vendor ID, followed by 4 character hexadecimal the serial number
DEVICE_CERT	string	R	Device unique certificate in ASCII PEM format [12.3.a]. Newlines must be encoded as ASCII line feed (without carriage return). In case a PKI is used, intermediate CA certificates in ASCII PEM format [12.3.a] can be concatenated in this field after the device unique certificate to form a certificate chain.
DEVICE_CERT_PRIVATE	string	R	Private key matching DEVICE_CERT in PEM format
DEVICE_SECRET	string	O	A device factory provisioned secret that can be used as source of entropy for other operations (e.g. password computation)
CALIBRATION_xxx	raw	O	Calibration data with xxx presenting the device/domain it applies to. This is optional as it is eventually tied to the driver implementation.
DSL_ANNEX	string	O	DSL annex hardware configuration "P" = Annex A (POTS/PSTN) "I" = Annex B/J (ISDN)

Item identifier	Type	M/R/O	Usage
DSL_XTUR_VENDOR	raw (8 bytes)	O	ITU G.997.1 7.4.2 xTU-R vendor ID in G.944.1 vendor ID format: 2 bytes T.35 country code 4 bytes provider code 2 bytes vendor-specific information
DSL_XTUR_SYSTEM_VENDOR	raw (8 bytes)	O	ITU G.997.1 7.4.4 xTU-R system vendor ID in G.944.1 vendor ID format: 2 bytes T.35 country code 4 bytes provider code 2 bytes vendor-specific information
DSL_XTUR_SERIAL	string	O	ITU G.997.1 7.4.8 xTU-R serial number Up to 32 ASCII characters.

9 Example Secure Manufacturing Data

Here is an example of a manufacturing data device tree converted to text format.

In this example only one party is signing sensitive data with key referenced as 0x10. There are 2 keys used to cipher data referenced with 0x00 and 0x01.

Template example

```
/dts-v1/;
/{
	MFG_DATA {
		MFG_DATA_version = "1.0";
		MFG_DATA_date = "1638787082";
		SERIAL_NUMBER {
			type = "string";
			value = "PRPL1234567890";
		};
		BASE_MAC_ADDRESS {
			type = "raw";
			value = [12 34 56 78 90 AB];
		};
		MANUFACTURER {
			type = "string";
			value = "prplManufacturer";
		};
		MANUFACTURER_OUI {
			type = "raw";
			value = [12 34 56];
		};
		MANUFACTURER_URL {
			type = "string";
			value = "http://prplfoundation.org/";
		};
		MODEL_NAME {
			type = "string";
			value = "prpl_gateway_01";
		};
		PRODUCT_CLASS {
			type = "string";
			value = "PRPLDEVICE";
		};
		HARDWARE_VERSION {
			type = "string";
			value = "PRPLDEVICE_0_0_1";
		};
```

```
          DEVICE_SPECIFIC_SECRET {
                type = "string";
                value = "------BEGIN RSA PRIVATE KEY-----
MIIFDjCCAvagAwIBAgIVALKkgBjO3RzX6EM2umdI2axaWPjOMA0GCSqGSIb3DQEB
CwUAMGExCzAJBgNVBAYTAkZSMQ8wDQYDVQQKDAZPcmFuZ2UxGTAXBgNVBAsMEEZP
...
bR8N7fnGW1rZ9bf+Pu6N3LQWRKVRPtIKGGWyzYwdaupYVb8fXgMmBUSElmLpscyx
sxY=
-----END RSA PRIVATE KEY-----";
                signature = {
                        key_index = <0x10>;
                };
                cipher {
                        key_index = <0x00>;
                        format = "aes-256-cbc-base64";
                };
          };
          WLAN_PASSPHRASE{
                type = "string";
                value = "V0ZVNkc3YkpEYnBIYXZqMmo3";
                signature = {
                        key_index = <0x10>;
                };
                cipher {
                        key_index = <0x01>;
                        format = "aes-256-cbc-base64";
                };
          };
          USERFS_KEY {
                type = "raw";
                value = [F3 6C CC 85 2D 62 75 EC 32 6F A6 3E 88 B0 DF 1B 66 36 23
5E E5 7D 43 6D 67 90 32 1A C0 BC 9E 5E];
                cipher {
                        key_index = <0x01>;
                        format = "aes-256-cbc-base64";
                };
            };
          Signature {
            key_index = <0x10>
            algo = "sha256-pss";
            type = "raw";
            value = <0x52777954 0x31713054 0x59585670 0x41344772 0x6b366670
0x5a36345a 0x476a6645 ... (cut out) ... 0x30773d3d>;
          };
     };
};
```

10 Security Notes

The format doesn't by itself contain any key used for ciphering or authentication: only indexes are used to point to the proper key to be used by the manufacturing data driver.

Those keys should be properly secured and externally provided either to kernel ring, secure enclave, or any secure storage.

Also, the integrity/authenticity of the global device tree binary itself MUST be assured by the running system to avoid simple replacement, for instance by a global signature or having it stored in a secure area.

11 Example of Manufacturing Data Handling Between Multiple Parties:

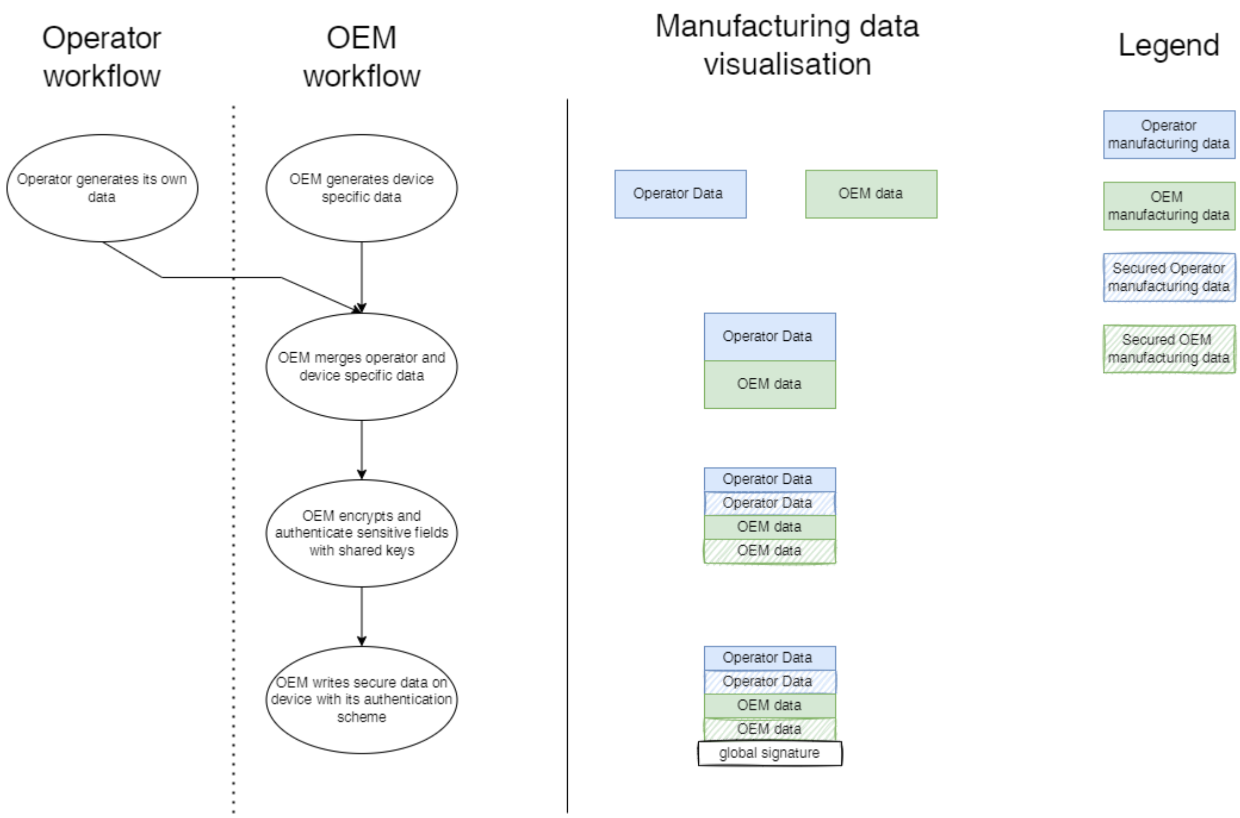
This section showcases possible workflows to handle secure manufacturing data between various parties.

11.1 Simple Use Case: OEM Trusted by Operator

In this example the operator trusts the OEM enough so he would provide him with the private keys and encryption keys to secure its own operator data section.

In that scheme of trusted OEM, it would have access to all the private keys to sign Operator Data, OEM data and perform the global device tree signature.

That way the manufacturer can merge and secure the manufacturing data directly on site.



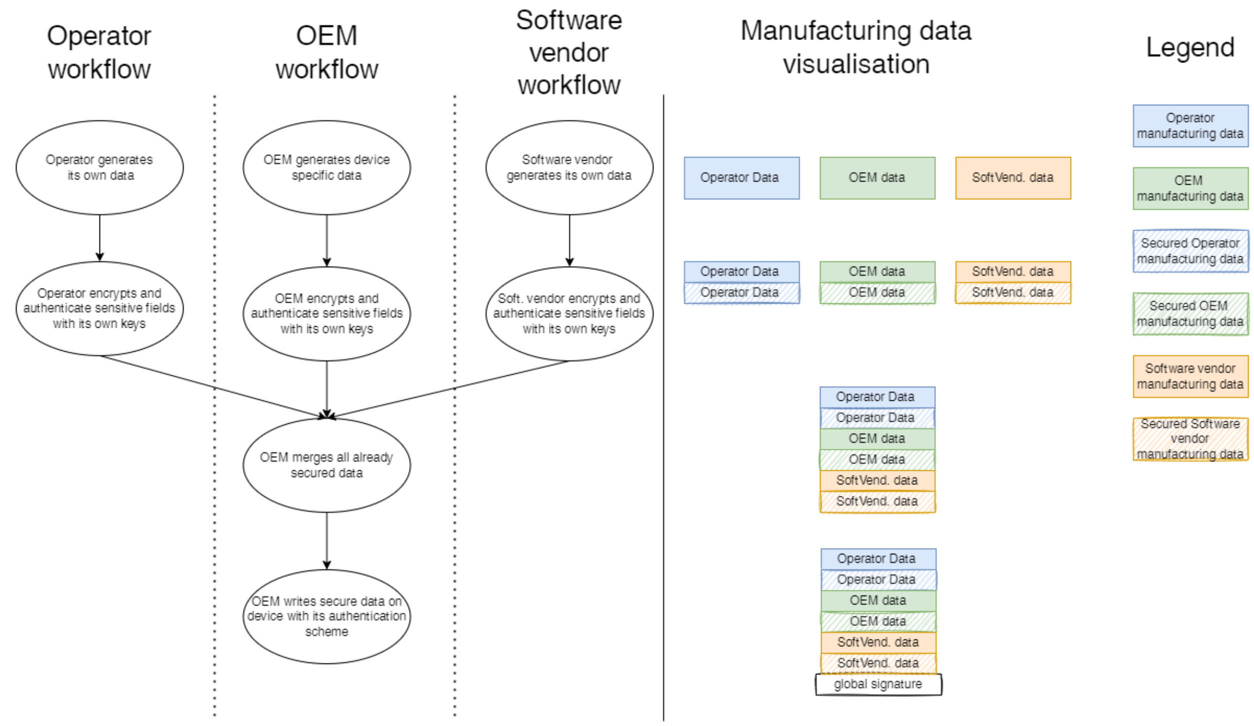
11.2 COMPLEX USE CASE: MULTITRUST SCHEME

In this example each party doesn't trust each other enough and wants to remain the owner of its own data set to avoid unauthorised modification.

This need can arise if for instance operators have specific device private keys they don't want to disclose.

For a software vendor this can also allow embedding a specific usage licence directly in the manufacturing data.

Note that the OEM still needs to have access to the private key to perform the global signature of the DTB once it has aggregated all the various DTB parts.



12 REFERENCES

1) Device Tree:
   a) https://www.devicetree.org/
   b) https://elinux.org/Device_Tree_Reference
2) BBF Technical Reports:
   a) TR-069: https://www.broadband-forum.org/technical/download/TR-069_Amendment-6_Corrigendum-1.pdf
   b) TR-181: https://www.broadband-forum.org/technical/download/TR-181_Issue-2_Amendment-15.pdf
   c) TR-369: https://usp.technology/
3) RFC7468 - Textual Encodings of PKIX, PKCS, and CMS Structures
   a) Section 5.1 - Textual encoding of certificates: https://www.rfc-editor.org/rfc/rfc7468.html#section-5
   b) Section 10 - One Asymmetric Key and the Textual Encoding of PKCS #8 Private Key Info: https://www.rfc-editor.org/rfc/rfc7468#page-12