

KALKITECH

KALKITECH

Member



DLMS User Association

Tamper Handling in DLMS

Prepared By

**Vinoo S Warriar and Balagopal
Kalki Communication Technologies Limited
#147 2nd Floor 5th Main Road, Sector 7 HSR Layout
Bangalore 560102
<http://www.kalkitech.com>**

Table of Contents

1	Introduction.....	3
2	References.....	3
3	Object modeling in DLMS.....	3
3.1	Objects.....	3
3.2	Interface classes	4
3.2.1	Data(class_id: 1).....	4
3.2.2	Register(class_id: 3).....	5
3.2.3	Extended Register(class_id: 4).....	6
3.2.4	Profile generic(class_id: 7).....	6
4	Event related objects.....	7
4.1	Event code.....	7
4.2	Event logs.....	8
4.3	Error register.....	10
4.4	Error profile	10
4.5	Alarm register.....	10
4.6	Alarm profile.....	10
4.7	Alarm register filter.....	11
4.8	Objects for monitoring cover openings.....	11
5	DLMS model for storing/notification of tampering information in Indian context.....	11
5.1	Reference table	11
5.2	Event code object.....	14
5.3	Defining new tamper information.....	14
5.4	Error and alarm register.....	15
5.5	Alarm filter.....	15
6	Application layer services	16
6.1	Event notification.....	16
6.2	Trigger event notification.....	16

1 Introduction

This document describes handling of tamper information in DLMS protocol. Sections 3 and 4 in this document throws light on modeling DLMS objects to store instantaneous/historic tamper information. The last section briefly explains the role of Application layer in notifying the captured events.

2 References

Reference	Title
DLMS UA 1000-1:2007, Eighth Edition	DLMS UA Blue book : COSEM identification system and Interface classes
DLMS UA 1000-2:2007, Sixth Edition	DLMS UA Green book : DLMS Architecture and Protocols
BlueBook 9th_V01_GK080804	Pre-release v0.1 DLMS UA Blue book : COSEM identification system and Interface classes
dlms_014_1.3_smart_GK081117	Proposal to add new interface class

3 Object modeling in DLMS

DLMS models all meter data as objects , which contributes to its salient features such as interoperability, usability for various energy types etc. This section explains some of the object modeling basics which will be referred in later sections.

3.1 Objects

An object is a collection of attributes and methods. The information of an object is organized in attributes. They represent the characteristics of an object by means of attribute values. The value of an attribute may affect the behavior of an object. The first attribute in any object is the "logical_name". It is one part of the identification of the object. An object may offer a number of methods to either examine or modify the values of the attributes

<i>Attributes</i>		
<i>Attribute ID</i>	<i>Name</i>	
1	Logical Name	
2		
n		
<i>Methods</i>		
<i>Method ID</i>	<i>Name</i>	

3.2 Interface classes

Objects that share common characteristics are generalized as an interface class with a `class_id`. Within a specific class, the common characteristics (attributes and methods) are described once for all objects. Instantiations of an interface class are called COSEM interface objects.

The set of different interface classes form a standardized library from which the manufacturer can assemble (model) its individual products. The elements are designed so that with them the entire range of products (from residential to commercial and industrial applications) can be covered. The choice of the subset of interface classes used to build a meter, their instantiation, and their implementation are part of the product design and therefore left to the manufacturer. The concept of the standardized metering interface class library provides the different users and manufacturers with a maximum of diversity without having to sacrifice interoperability.

Four interface classes which will be used in event capturing/profiling context are explained in detail

3.2.1 Data(class_id: 1)

This IC allows to model various data, such as configuration data and parameters. The data are identified by the attribute *logical_name*.

NOTE: The first attribute of all interface class will be Logical Name, which stores the OBIS(Object Identification System)code – 6 field value for uniquely identifying each object.

Data	0...n	class_id = 1, version = 0			
Attribute(s)	Data type	Min.	Max.	Def.	Short name
1. logical_name (static)	octet-string				x
2. value	CHOICE				x + 0x08
Specific methods	m/o				

CHOICE

```
{
  --simple data types
  null-data          [0],
  boolean            [3],
  bit-string         [4],
  double-long        [5],
  double-long-unsigned [6],
  octet-string       [9],
  visible-string     [10],
  bcd                [13],
  integer            [15],
  long               [16],
  unsigned           [17],
  long-unsigned      [18],
  long64             [20],
```

```

long64-unsigned    [21],
enum               [22],
float32            [23],
float64            [24],
date-time          [25],
date               [26],
time              [27],
--complex data types
array              [1],
structure          [2],
compact-array     [19]
}

```

3.2.2 Register(class_id: 3)

This IC allows to model a process value or a status value with its associated unit. "Register" objects know the nature of the process value or status value. It is identified by the attribute *logical_name*.

Register		0...n	class_id = 3, version = 0			
Attribute(s)		Data type	Min.	Max.	Def.	Short name
1. logical_name	(static)	octet-string				x
2. value	(dyn.)	CHOICE				x + 0x08
3. scaler_unit	(static)	scal_unit_type				x + 0x10
Specific methods		m/o				
1. reset (data)		o				x + 0x28

CHOICE

```

{
--simple data types
null-data         [0],
bit-string        [4],
double-long       [5],
double-long-unsigned [6],
octet-string      [9],
visible-string    [10],
integer           [15],
long              [16],
unsigned          [17],
long-unsigned     [18],
long64            [20]
}

```

```

    long64-unsigned    [21],
    float32            [23],
    float64            [24]
}

```

3.2.3 Extended Register(class_id: 4)

This IC allows to model a process value with its associated scaler, unit status and time information. “Extended register” objects know the nature of the process value. It is described by the attribute *logical_name*.

Extended register		0...n	class_id = 4, version = 0			
Attribute(s)		Data type	Min.	Max.	Def.	Short name
1. logical_name	(static)	octet-string				x
2. value	(dyn.)	CHOICE				x + 0x08
3. scaler_unit	(static)	scal_unit_type				x + 0x10
4. status	(dyn.)	CHOICE				x + 0x18
5. capture_time	(dyn.)	octet-string				x + 0x20
Specific methods		m/o				
1. reset (data)		o				x + 0x38

CHOICE

```

{
    --simple data types
    null-data            [0],
    bit-string           [4],
    double-long-unsigned [6],
    octet-string         [9],
    visible-string       [10],
    unsigned             [17],
    long-unsigned        [18],
    long64-unsigned      [21]
}

```

3.2.4 Profile generic(class_id: 7)

Profile generic object can be used to record historic values, where as the interface classes discussed above can store only one(most recent) value at a time.

Profile generic		0...n	class_id = 7, version = 1			
Attribute(s)		Data type	Min.	Max.	Def.	Short name

1. logical_name	(static)	octet-string				x
2. buffer	(dyn.)	compact-array or array				x + 0x08
3. capture_objects	(static)	array				x + 0x10
4. capture_period	(static)	double-long-unsigned				x+ 0x18
5. sort_method	(static)	enum				x + 0x20
6. sort_object	(static)	capture_object_definition				x + 0x28
7. entries_in_use	(dyn.)	double-long-unsigned	0		0	x + 0x30
8. profile_entries	(static)	double-long-unsigned	1		1	x + 0x38
Specific methods		m/o				
1. reset (data)		o				x + 0x58
2. capture (data)		o				x+ 0x60
3. reserved from previous versions		o				
4. reserved from previous versions		o				

One or more instantaneous value(object attributes) can be defined as Capture object of a profile. If the Capture period(in seconds) is greater than one, then automatic capture is assumed; else capture should be triggered externally. Every capture result in adding new set of values(entry) to the profile buffer.

4 Event related objects

4.1 Event code

An event code object is used to hold the identifier corresponding to most recent event. Data, Register or Extended Register classes can be used to model this object. DLMS allows to define country specific reference table defining all possible events with corresponding identifier.

Event code	IC	OBIS code					
		A	B	C	D	E	F
Event code	1, Data, 3, Register, 4, Extended register	0	b	96	11	e	255

Value group E allows to classify events as needed such as power related, vectors related, hardware related, fraud related etc. Currently DLMS allows 10 values(0...9) for value group E enabling user to define up to 10 event categories.

Note: Value group E refer only to the broad categorization of event and there is no limit to the number of events(event code) defined in reference table.

Table 1: Event code object definitions

OBIS code	Meaning	Example
0.b.96.11.0.255	Event code #0	Power related
0.b.96.11.1.255	Event code #1	Vectors related
0.b.96.11.2.255	Event code #2	Hardware related
0.b.96.11.3.255	Event code #3	Fraud related
0.b.96.11.4.255	Event code #4	
0.b.96.11.5.255	Event code #5	
0.b.96.11.6.255	Event code #6	
0.b.96.11.7.255	Event code #7	
0.b.96.11.8.255	Event code #8	
0.b.96.11.9.255	Event code #9	

Illustration

Table 2: Event reference table

Event identifier	Event description
1	
2	
37	HV , 35kV abnormal ESD disturbance tamper Status bit = 18
38	Unauthorized time setting

With reference to table-1(Event code object definitions), object to store fraud related events is 0.0.96.11.3.255(Interface Class: Data). When ever an abnormal ESD disturbance tamper occur, the corresponding event identifier (37) will be stored in the "value" attribute of 0.0.96.11.3.255 object.

The event code object stores instantaneous values only; which means a new event will over write a previously captured event code. Event logs allows to historically record all events occurred.

4.2 Event logs

These are profile generic objects to store historic values in its buffer attribute. The capture object

contains object attribute definitions of interested data. Interested data includes event code and other relevant information such as timestamp, instantaneous electricity related information (such as current/voltage/energy register contents).

Event logs	IC	OBIS code					
		A	B	C	D	E	F
Event log	7, Profile generic	0	b	99	98	e	255 _a

Value group E allows to classify event logs as needed such as power related, vectors related, hardware related, fraud related etc. Currently DLMS allows 10 values (0...9) for value group E allowing to define up to 10 event log categories.

OBIS code	Meaning	Example
0.b.99.98.0.255	Event log #0	Power related
0.b.99.98.1.255	Event log #1	Vectors related
0.b.99.98.2.255	Event log #2	Hardware related
0.b.99.98.3.255	Event log #3	Fraud related
0.b.99.98.4.255	Event log #4	
0.b.99.98.5.255	Event log #5	
0.b.99.98.6.255	Event log #6	
0.b.99.98.7.255	Event log #7	
0.b.99.98.8.255	Event log #8	
0.b.99.98.9.255	Event log #9	

Illustration

An event log object for recording historical fraud related events is modeled using 0.b.99.98.3.255. The capture object contains event code object for fraud related tamper, timestamp of the occurrence of event and the value of an energy register at the time of tamper. Thus every time a fraud related tamper occurs, a new entry is added in profile buffer.

Table 3: Event log

	Event Identifier	Timestamp	Energy register value
Entry #1			
Entry #8	37	1-Jan-2008 10:10:00	XX volts
Entry #9	38	3-Jan-2008 06:00:00	YY volts

4.3 Error register

Error register objects are used to communicate error indications of the device. The different error registers are held by the *value* attribute of “Data” objects, with data type or *bit-string*, *octet-string*, *unsigned*, *long-unsigned*, *double-long-unsigned* or *long64-unsigned*.

The individual bits of the error register may be set and cleared by a pre-defined selection of events. Depending on the type of the error, some errors may clear themselves when the reason setting the error flag disappears.

Error register	IC	OBIS code					
		A	B	C	D	E	F
Error register 1...10 object	1, Data	0	b	97	97	0...9	255

4.4 Error profile

If more than one instance of Error register is used, it is also allowed to combine them into one instance of the IC "Profile generic". In this case, the captured objects are the “Data” objects, the capture period is 1 to have just actual values, the sort method is FIFO, the profile entries are limited to 1.

Error profile	IC	OBIS code					
		A	B	C	D	E	F
Error profile object	7, Profile generic	0	b	97	97	255	255

4.5 Alarm register

A number of objects are available to hold alarm flags. The different alarm registers are held by the *value* attribute of “Data” objects, with data type or *bit-string*, *octet-string*, *unsigned*, *long-unsigned*, *double-long-unsigned* or *long64-unsigned*. When selected events occur, they set the corresponding flag and the device raises an alarm. Alarm flags do not re-set themselves; they can be reset by writing the *value* attribute only.

Alarm register	IC	OBIS code					
		A	B	C	D	E	F
Alarm register objects 1...10	1, Data	0	b	97	98	0...9	255

4.6 Alarm profile

If more than one instance of Alarm register is used, it is also allowed to combine them into one instance of the IC "Profile generic". In this case, the captured objects are the “Data” objects, the capture period is 1 to have just actual values, the sort method is FIFO, the profile entries are limited to 1.

Alarm profile	IC	OBIS code					
		A	B	C	D	E	F
Alarm profile object	7, Profile generic	0	b	97	98	255	255

4.7 Alarm register filter

The alarm filters define if an event is to be handled as an alarm when it appears. The different alarm filters are held by the *value* attribute of “Data” objects, with data type or *bit-string*, *octet-string*, *unsigned*, *long-unsigned*, *double-long-unsigned* or *long64-unsigned*. The bit mask has to same structure as the alarm register object. If a bit in the alarm filter is set, then the corresponding alarm is enabled, otherwise it is disabled.

Alarm register filter	IC	OBIS code					
		A	B	C	D	E	F
Alarm filter objects 1...10	1, Data	0	b	97	98	10...19	255

4.8 Objects for monitoring cover openings

Object for monitoring cover openings	IC	OBIS code					
		A	B	C	D	E	F
Event log	4, Extended register	0	b	96	15	0	255

Value group B is used to identify the cover.

OBIS code	Meaning
0.0.96.15.0.255	Main cover
0.1.96.15.0.255	Terminal cover
0.2.96.15.0.255	Battery compartment

5 DLMS model for storing/notification of tampering information in Indian context

5.1 Reference table

The country specific authority is responsible for creating and maintaining event reference table. A reference table defining possible Indian event/tamper identifier and description is as below

Tamper identifier	Name	Status_bit	Remarks
1.	Phase 1 PT Link miss tamper	0	
2.	Phase 1 PT Link miss tamper restore		
3.	Phase 2 PT Link miss tamper	1	
4.	Phase 2 PT Link miss tamper restore		
5.	Phase 3 PT Link miss tamper	2	
6.	Phase 3 PT Link miss tamper restore		
7.	Phase 4(neutral) PT Link miss tamper, also neutral miss in 1 phase	3	
8.	Phase 4(Neutral) PT Link miss tamper(also neutral miss in 1 phase) restore		
9.	Phase 1 CT reverse tamper	4	
10.	Phase 1 CT reverse tamper restore		
11.	Phase 2 CT reverse tamper	5	
12.	Phase 2 CT reverse tamper restore		
13.	Phase 3 CT reverse tamper	6	
14.	Phase 3 CT reverse tamper restore		
15.	Any CT bypass tamper (poly or 1ph meters including earthed load)	7	phasor sum of I_phase minus I_neutral not equal to zero
16.	Any CT bypass tamper (poly or 1ph meters) restore		
17.	Phase 1 CT open tamper	8	
18.	Phase 1 CT open tamper restore		
19.	Phase 2 CT open tamper	9	
20.	Phase 2 CT open tamper restore		
21.	Phase 3 CT open tamper	10	
22.	Phase 3 CT open tamper restore		
23.	Current unbalance (in poly-phase only)	11	
24.	Current unbalance (in poly-phase only) restore		

25.	Voltage unbalance (in poly-phase only)	12	
26.	Voltage unbalance (in poly-phase only) restore		
27.	1 phase miss (split phase condition to run 3ph motor with capacitor using the 2 available phases)	13	1 phase miss and Very low PF
28.	1 phase miss (split phase condition to run 3ph motor with capacitor) restore		
29.	Any one phase and neutral miss (in poly-phase only)	14	in this condition meter gets $415V/2=207.5V$ in 2 phases but the load is given all the 3 phases and earth
30.	Any one phase and neutral miss (in poly-phase only) restore		
31.	Magnetic tamper	15	
32.	Magnetic tamper restore		
33.	Top cover open tamper	16	
34.	Top cover open tamper restore		
35.	Terminal cover open tamper	17	
36.	Terminal cover open tamper restore		
37.	HV , 35kV abnormal ESD disturbance tamper	18	
38.	HV , 35kV abnormal ESD disturbance tamper restore		
39.	Phase sequence reversal tamper (in poly-phase only)	19	
40.	Phase sequence reversal tamper (in poly-phase only) restore		
41.	Neutral disturbance	20	ac or dc or chopped wave superimposed on neutral
42.	Neutral disturbance restore		
43.	kW Overload	21	kW Demand exceeding sanctioned load or a predefined threshold
44.	kW Overload restore		
45.	kVA overload	22	kVA Demand exceeding over sanctioned load or a predefined

			threshold
46.	kVA overload restore		
47.	Over voltage	23	Phase to phase voltage
48.	Over voltage restore		
49.	Low voltage in any phase	24	$V < -40\% V_n$ but $v > 10\% V_n$, I present in that phase
50.	Low voltage in any phase restore		
51.	RTC bad	25	
52.	Battery bad	26	
53.	Memory error	27	
54.	No of interruption(power ups) exceeding a predefined threshold	28	The frequency (hourly/daily/monthly) and threshold to be defined by the utility.
55.	POH less than a predefined threshold.	29	The threshold to be defined by the utility.

5.2 Event code object

An instance of Interface Class Data can be used to model event code object; the value attribute of this object will hold the identifier corresponding to most recent event. The data type of “value” attribute should be defined such that it must be capable of holding the biggest event identifier value. Data type “unsigned” can hold value up to event identifier = 255 and will suffice reference table requirements(unless the number of events exceed 255).

OBIS code = 0.0.96.11.3.255, IC = 1(Data)		
Attributes		
Attribute ID	Name	Data type
1	Logical name	Octet string
2	Value	unsigned
Methods		

5.3 Defining new tamper information

The country specific authority can add new entries into reference table(section 5.1) as need arises. To cope with the latest reference table, all that meter manufacturer need to do is to define a big enough data type to the “value” attribute of event code object.

5.4 Error and alarm register

0.0.97.97.3.255 and 0.0.97.98.3.255 (IC: Data) can be used to model error register and alarm register respectively. When an event defined in reference table occur, the corresponding bit(specified as status bit) is set in error register and/or alarm register.

Error register		
OBIS code = 0.0.97.97.3.255, IC = 1(Data)		
Attributes		
Attribute ID	Name	Data type
1	Logical name	Octet string
2	Value	Bit-string / octet-string / unsigned / long-unsigned / double-long-unsigned / long64-unsigned
Methods		
Alarm register		
OBIS code = 0.0.97.98.3.255, IC = 1(Data)		
Attributes		
Attribute ID	Name	Data type
1	Logical name	Octet string
2	Value	Bit-string / octet-string / unsigned / long-unsigned / double-long-unsigned / long64-unsigned
Methods		

5.5 Alarm filter

An alarm register filter object holds a bit mask to (temporarily)enable/disable an alarm irrespective of the occurrence of the corresponding event.

Alarm register filter		
OBIS code = 0.0.97.98.13.255, IC = 1(Data)		
Attributes		
Attribute ID	Name	Data type
1	Logical name	Octet string
2	Value	Bit-string / octet-string / unsigned / long-unsigned / double-long-unsigned / long64-unsigned
Methods		

6 Application layer services

6.1 Event notification

Data exchange between data collection systems and metering equipment using the COSEM interface object model is based on the client/server paradigm. Metering equipment play the role of the server. Event notification is the only non client server service supported by DLMS application layer as this need to be initiated from the server.

Event notification is an unconfirmed service requested by the server, upon the occurrence of an event, in order to inform the client of the value of an attribute, as though it had been requested by the COSEM. Upon the reception of the .request primitive, the Server Application Layer builds the EVENT-NOTIFICATION-Request APDU(Application Protocol Data Unit).

6.2 Trigger event notification

In some cases, the supporting lower layer protocol(s) do (does) not allow sending a protocol data unit in a real, unsolicited manner. In these cases, the client has to explicitly solicit sending an Event notification frame, by invoking the Trigger EventNotification service primitive.