NAME

TLV - type-length-value data encoding scheme.

DESCRIPTION

TLV is the data encoding scheme for most of KSI data structures. The TLV scheme is used to encode both the KSI data structures and protocol data units (PDUs) for transferring them between the entities during the signature generation process.

ENCODING

The values are octet strings of given length that carry information to be interpreted as specified by the types. The value part of an encoded object may contain nested TLV objects:

```
[T][L][
        [T][L][
                [T][L][V]
                [T][L][V]
        ]
        [T][L][V]
        [T][L][V]
]
```

For space efficiency, two TLV encodings are used:

- A 16-bit TLV (TLV16) encodes a 13-bit type and 16-bit length (and can thus contain at most 65535 octets of data in the value part).
- An 8-bit TLV (TLV8) encodes a 5-bit type and 8-bit length (at most 255 octets of value data).

TLV8 and TLV16 are distinguished by the '16-Bit' flag in the first octet of the type field. Smaller objects are encoded as TLV8 for lower overhead. A TLV8 type has local significance and identifies the encapsulated structure in the context where it is used. A TLV16 type < 256 still has local significance, but may be used to encode data that needs 16-bit length. A TLV16 type >= 256 has global significance.

The bit-layout of 8-bit and 16-bit TLV, where the left-most bit, the '16-Bit' flag is the most significant bit:

[*0*][**N**][**F**][Type:*5*][Length:*8*][Value:*0-255*]
[*1*][**N**][**F**][Type:*13*][Length:*16*][Value:*0-65535*]

Here the '**N**' and '**F**' flags are used to indicate how to react to unknown TLV types. Unknown type is defined as a TLV object whose encoding size (length) can be extracted but whose internal structure cannot be decoded. A decoder encountering such an object has the ability to skip it but not to act on it in any other but a predetermined way. The two flags used to tell parsers how to handle unknown types are defined as:

- **N** - *Non-Critical flag*. If **N**=*0* in an object and the type is unknown to a parser, it is a critical error and the results of any further computations with the object must be considered undefined. If **N**=*1*, then a warning message may be generated, but computations with the object should continue and the value field should be treated as unstructured binary data of given length.
- **F** - *Forward Unknown flag*. Used to clarify permissible actions with unknown data in case it is non-critical. If **F**=*0*, then the unknown data object is skipped as if it was never encountered. If **F**=*1*, the data is used as an unstructured binary stream in the context of ongoing operation, defined for each object explicitly. Usually it is passed on to next step of computation as is. The **F**-flag has no meaning and defines no action if the data is critical (**N**=*0*).

EXAMPLES

In the following examples all values are interpreted in ascii ('T'), hex (*0x01*) or binary (*10110*) where the left-most bit is the most significant bit.

1 A string "KSI" encoded in TLV format where type=*0x01*, **N**=*1* and **F**=*0*:

[0][1][0][0x01][0x04]['K']['S']['I']['\0'] -->
[0100 0001][0000 0100][0100 1011][0101 0011][0100 1001][0000 0000]

2 A TLV from example No. 1 is nested inside a parent TLV where type=*0x100*, **N**=*0* and **F**=*0*:

[1][0][0][0x100][0x06]**[**[0][1][0][0x01][0x04]['K']['S']['I']['\0']**]** -->
[1000 0001][0000 0000][0000 0000][0000 0110][0100 0001][0000 0100][0100 1011][0101 0011][0100 1001][0000 0000]

AUTHOR

Guardtime AS, http://www.guardtime.com/

SEE ALSO

gttlvdump(1), **gttlvgrep**(1), **gttlvundump**(1), **gttlvwrap**(1) **tlv-desc**(5)