Subject: strange behaviour of Vector serialization
Posted by michael79 on Sun, 31 May 2020 12:19:31 GMT
View Forum Message <> Reply to Message

Here is the serialization of struct data which contains Vector(byte):

```
#pragma pack(push, 1)
struct OttMessage
{
    uint16 descriptor;
    uint16 msgNum;
    int32 sec;
    int32 nsec;

    Vector<byte> data;

    void Serialize(Stream& s)
    {
     s % descriptor % msgNum % sec % nsec % data;
    }
};
#pragma pack(pop)
```

DUMP of buffer show the next:
(int)buf[0] = 102
(int)buf[1] = 0
(int)buf[2] = 111
(int)buf[3] = 0
(int)buf[4] = 123
(int)buf[5] = 0
(int)buf[6] = 0
(int)buf[7] = 0
(int)buf[8] = 222
(int)buf[9] = 0
(int)buf[10] = 0
(int)buf[11] = 0
(int)buf[12] = 7
(int)buf[13] = 10
(int)buf[14] = 144
(int)buf[15] = 56
(int)buf[16] = 233
(int)buf[17] = 77

Data from Vector begins since buf[12], but the value 7 unexpectable, it was not added to Vector.

When I change Serialize() as the following, then all data correct:

```
void Serialize(Stream& s)
{
    s % descriptor % msgNum % sec % nsec;

    for(int i = 0; i < data.size(); i++)
    {
 s % data[i];
    }
}
```

```
(int)buf[0] = 102
(int)buf[1] = 0
(int)buf[2] = 111
(int)buf[3] = 0
(int)buf[4] = 123
(int)buf[5] = 0
(int)buf[6] = 0
(int)buf[7] = 0
(int)buf[8] = 222
(int)buf[9] = 0
(int)buf[10] = 0
(int)buf[11] = 0
(int)buf[12] = 10
(int)buf[13] = 144
(int)buf[14] = 56
(int)buf[15] = 233
(int)buf[16] = 77
(int)buf[17] = 188
```

Can anyone explane such strange behaviour?
Why I need to add Vector data one by one?