CHAPTER 4

DESIGNING DATA-INTENSIVE APPLICATIONS

DATA SERIALIZATION

FORMATS FOR ENCODING DATA

SERIALIZATION

- JSON
- XML
- PROTOCOL BUFFER
- THRIFT
- AVRO

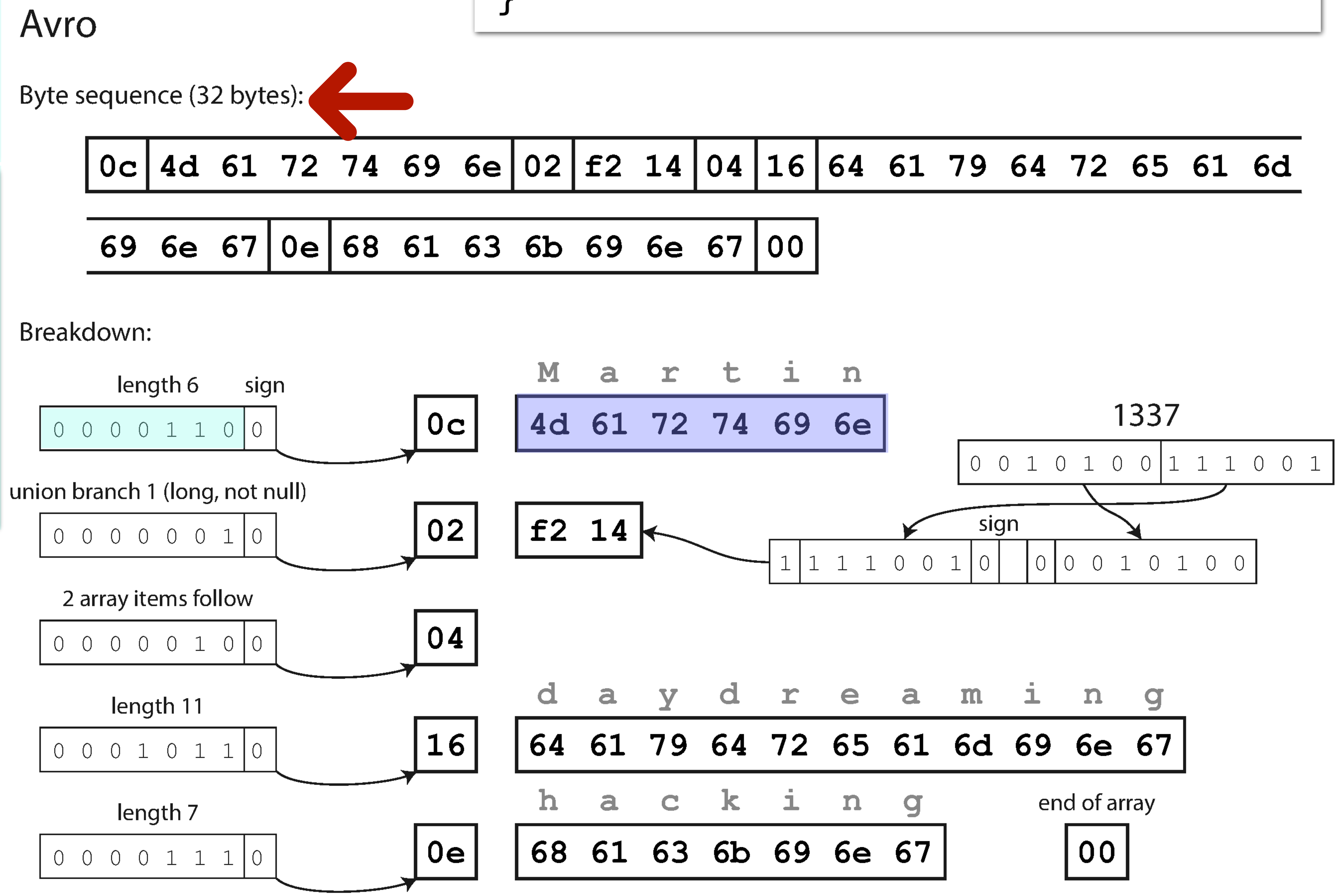
Emam

WHAT IS AVRO & HOW IT WORKS?

```
{
    "userName": "Martin",
    "favoriteNumber": 1337,
    "interests": ["daydreaming", "hacking"]
}
```

```
record Person {
    string                  userName;
    union { null, long } favoriteNumber = null;
    array<string>           interests;
}
```

```
{
    "type": "record",
    "name": "Person",
    "fields": [
        {"name": "userName",        "type": "string"},
        {"name": "favoriteNumber", "type": ["null", "long"],
"default": null},
        {"name": "interests",       "type": {"type": "array",
"items": "string"}}
    ]
}
```

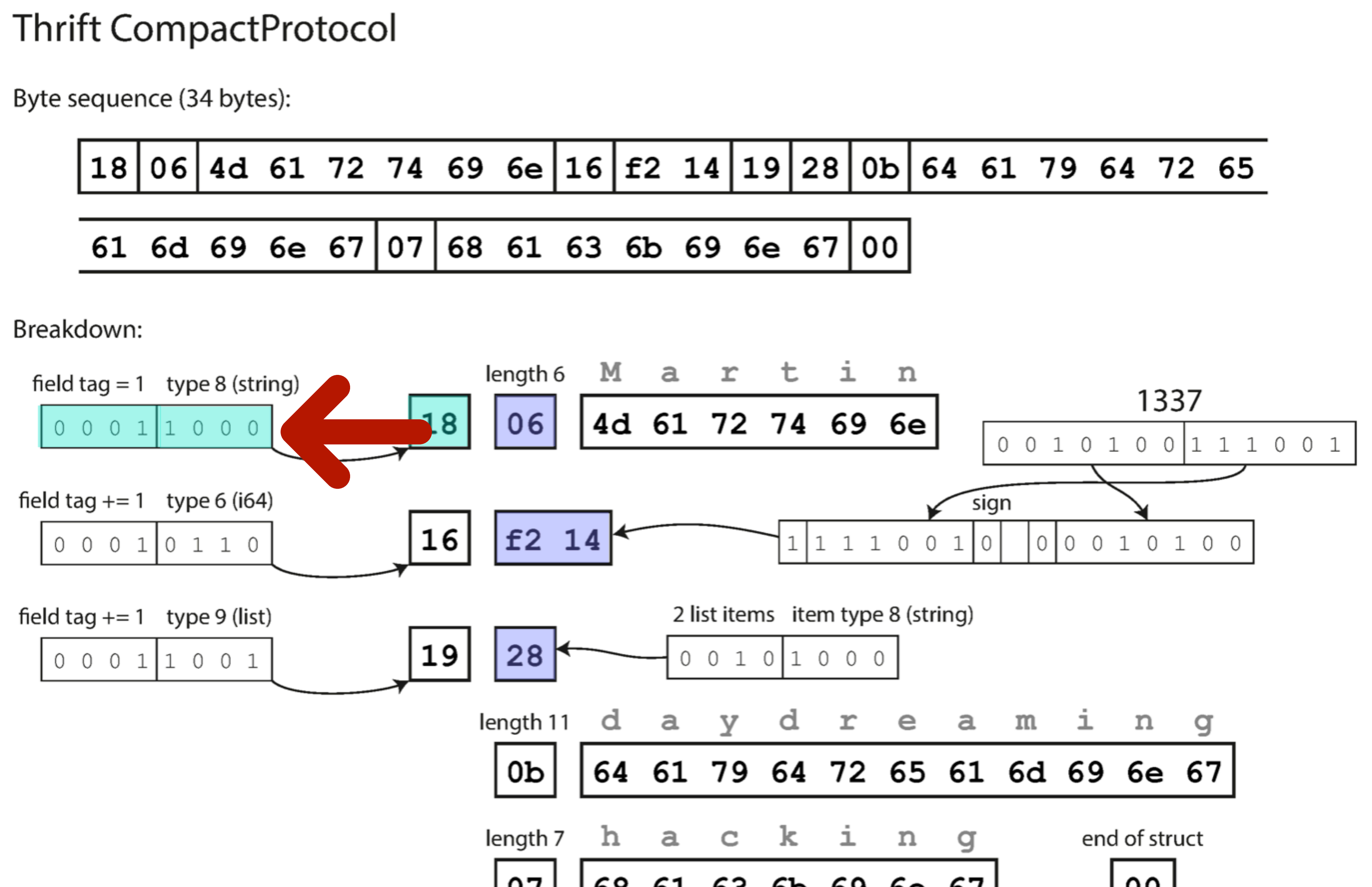


Avro

Byte sequence (32 bytes):

```
0c | 4d 61 72 74 69 6e | 02 | f2 14 | 04 | 16 | 64 61 79 64 72 65 61 6d
69 6e 67 | 0e | 68 61 63 6b 69 6e 67 | 00
```

Breakdown:



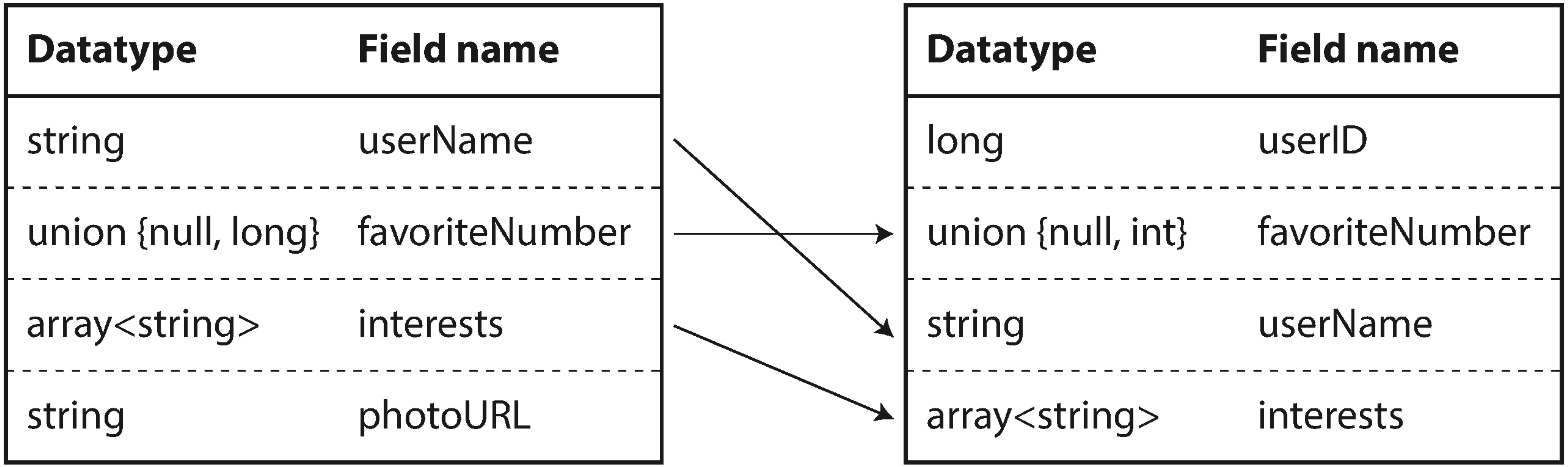
Thrift CompactProtocol

Byte sequence (34 bytes):

```
18 | 06 | 4d 61 72 74 69 6e | 16 | f2 14 | 19 | 28 | 0b | 64 61 79 64 72 65
61 6d 69 6e 67 | 07 | 68 61 63 6b 69 6e 67 | 00
```

Breakdown:

WHAT IS SCHEMA EVOLUTION?



Writer's schema for Person record

Datatype	Field name
string	userName
union {null, long}	favoriteNumber
array<string>	interests
string	photoURL

Reader's schema for Person record

Datatype	Field name
long	userID
union {null, int}	favoriteNumber
string	userName
array<string>	interests

MODES OF DATAFLOW

https://example.com/users/:id

DATABASES

SERVICE CALLS

ASYNC MSG PASSING

ESPRESSO 1.0

REST

SOAP

RPC

SOA

MICRO-SERVICES

RabbitMQ

APACHE kafka®

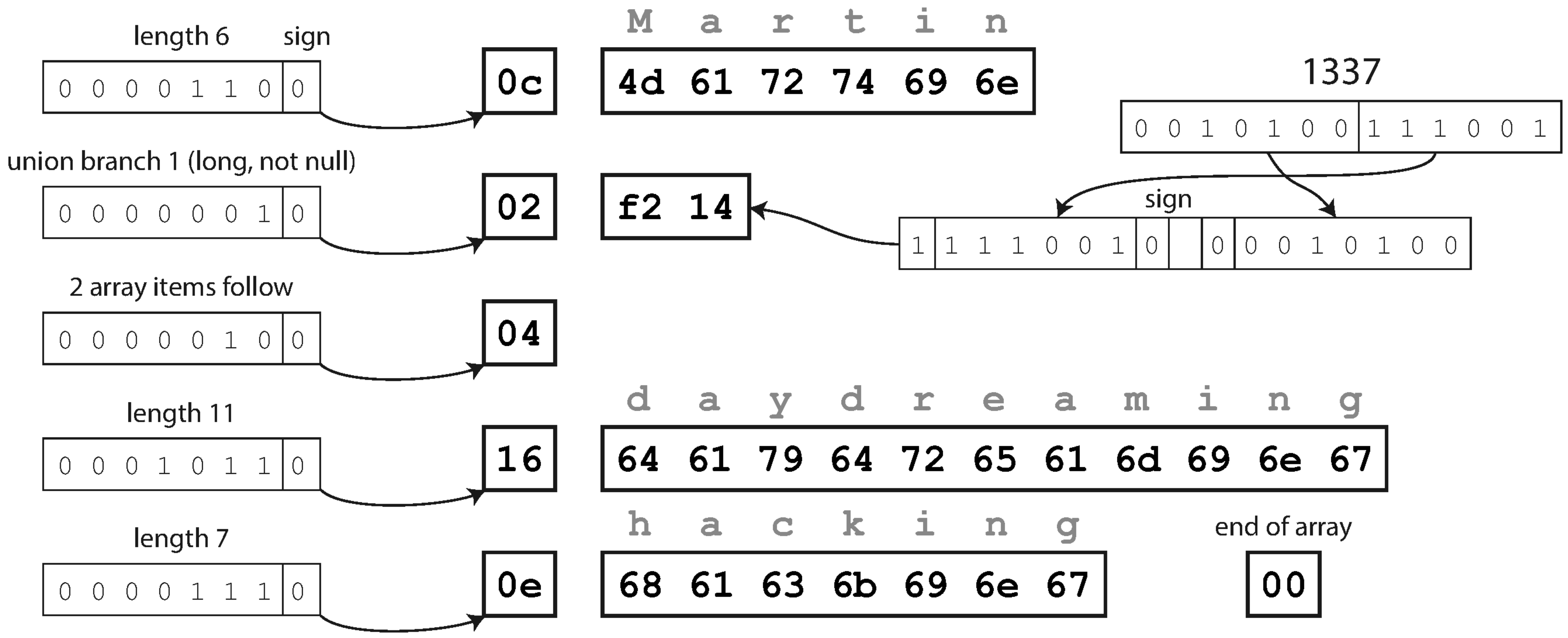
Emam

Avro

Byte sequence (32 bytes):

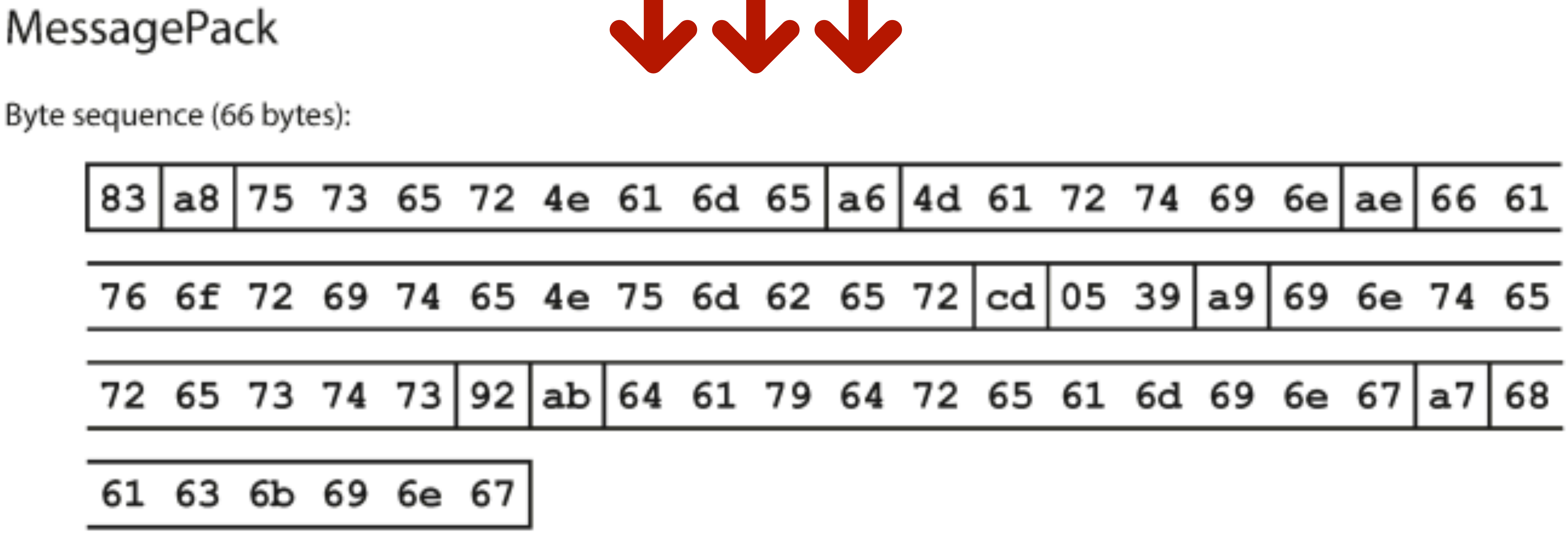
```
0c 4d 61 72 74 69 6e 02 f2 14 04 16 64 61 79 64 72 65 61 6d
69 6e 67 0e 68 61 63 6b 69 6e 67 00
```

Breakdown:



BJSON

```
{
    "userName": "Martin",
    "favoriteNumber": 1337,
    "interests": ["daydreaming", "hacking"]
}
```



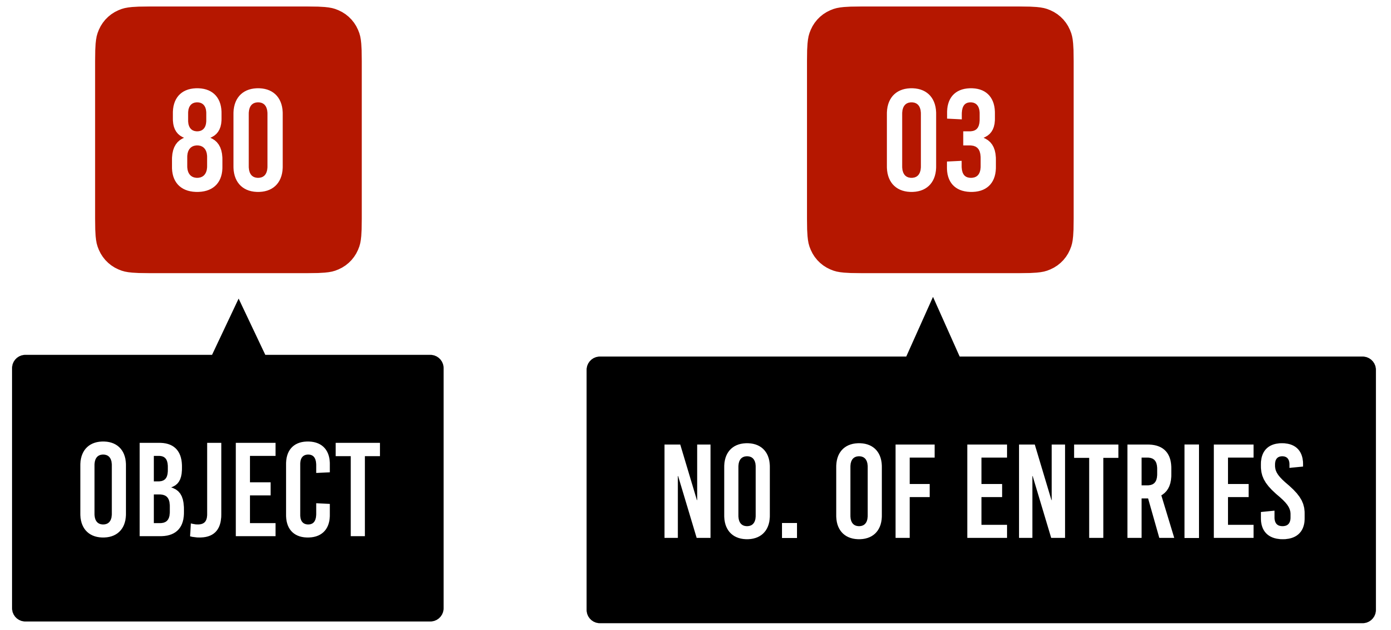
MessagePack

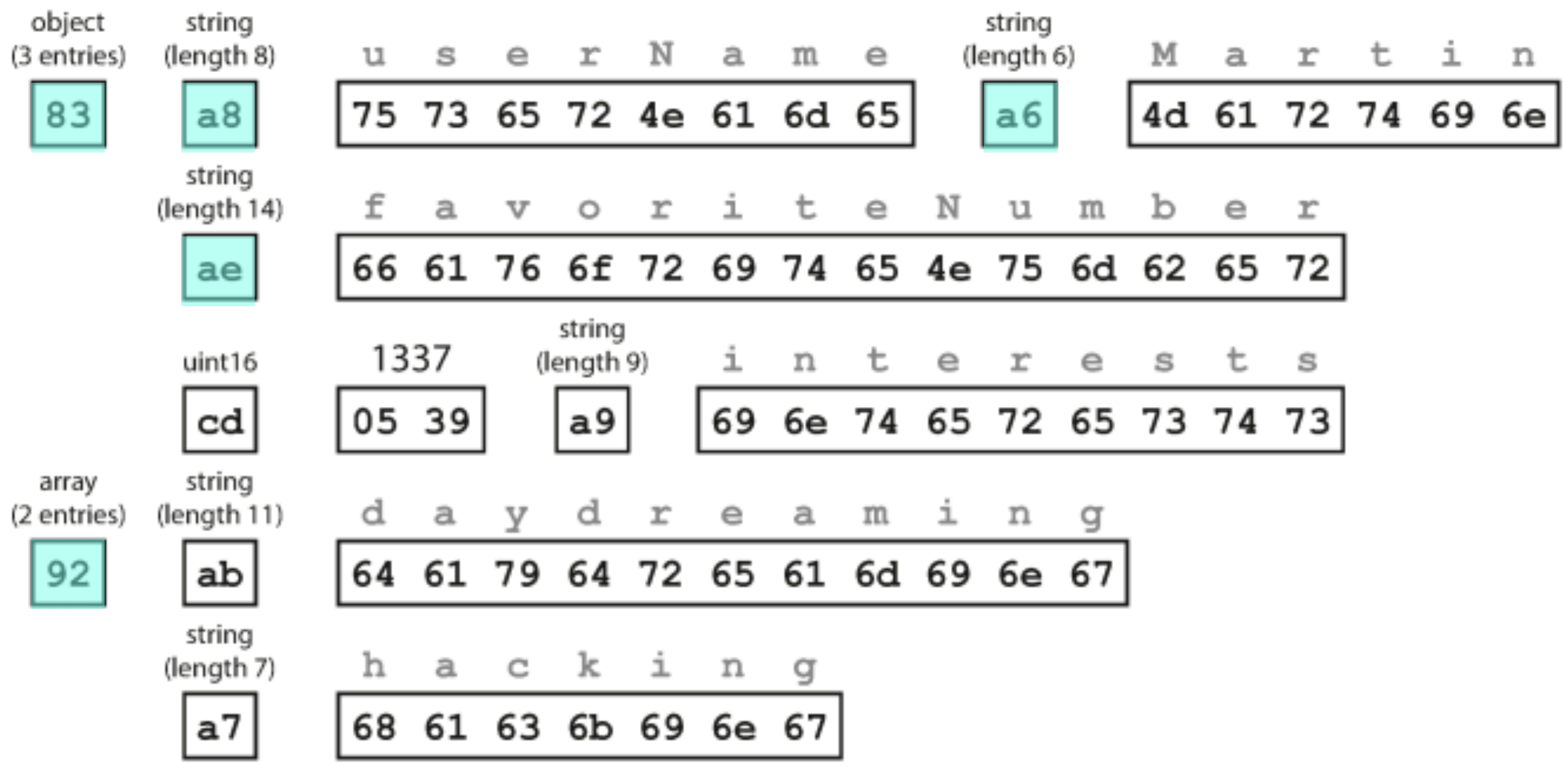
Byte sequence (66 bytes):

```
83 a8 75 73 65 72 4e 61 6d 65 a6 4d 61 72 74 69 6e ae 66 61
76 6f 72 69 74 65 4e 75 6d 62 65 72 cd 05 39 a9 69 6e 74 65
72 65 73 74 73 92 ab 64 61 79 64 72 65 61 6d 69 6e 67 a7 68
61 63 6b 69 6e 67
```

```
{
    "userName": "Martin",
    "favoriteNumber": 1337,
    "interests": ["daydreaming", "hacking"]
}
```

128	64	32	16	8	4	2	1
0	1	0	1	0	0	1	1





APACHE THRIFT

PROTOCOL BUFFERS

AVRO

HADOOP

APACHE THRIFT

```
struct Person {
  1: required string       userName,
  2: optional i64          favoriteNumber,
  3: optional list<string> interests
}
```

BINARY PROTOCOL

COMPACT PROTOCOL

JSON PROTOCOL

PROTOCOL BUFFERS

```
message Person {
    required string user_name       = 1;
    optional int64  favorite_number = 2;
    repeated string interests       = 3;
}
```

COMPACT PROTOCOL

Thrift BinaryProtocol

Byte sequence (59 bytes):

```
0b | 00 01 | 00 00 00 06 | 4d 61 72 74 69 6e | 0a | 00 02 | 00 00 00 00
00 00 05 39 | 0f | 00 03 | 0b | 00 00 00 02 | 00 00 00 0b | 64 61 79 64
72 65 61 6d 69 6e 67 | 00 00 00 07 | 68 61 63 6b 69 6e 67 | 00
```

Breakdown:

```
type 11 (string)   field tag = 1   length 6                  M  a  r  t  i  n
0b                 00 01           00 00 00 06               4d 61 72 74 69 6e
                                   4 BYTES

type 10 (i64)      field tag = 2   1337
0a                 00 02           00 00 00 00 00 00 05 39   8 BYTES

type 15 (list)     field tag = 3   item type 11 (string)     2 list items
0f                 00 03           0b                        00 00 00 02

                                   length 11                 d  a  y  d  r  e  a  m  i  n  g
                                   00 00 00 0b               64 61 79 64 72 65 61 6d 69 6e 67

                                   length 7                  h  a  c  k  i  n  g                end of struct
                                   00 00 00 07               68 61 63 6b 69 6e 67               00
```

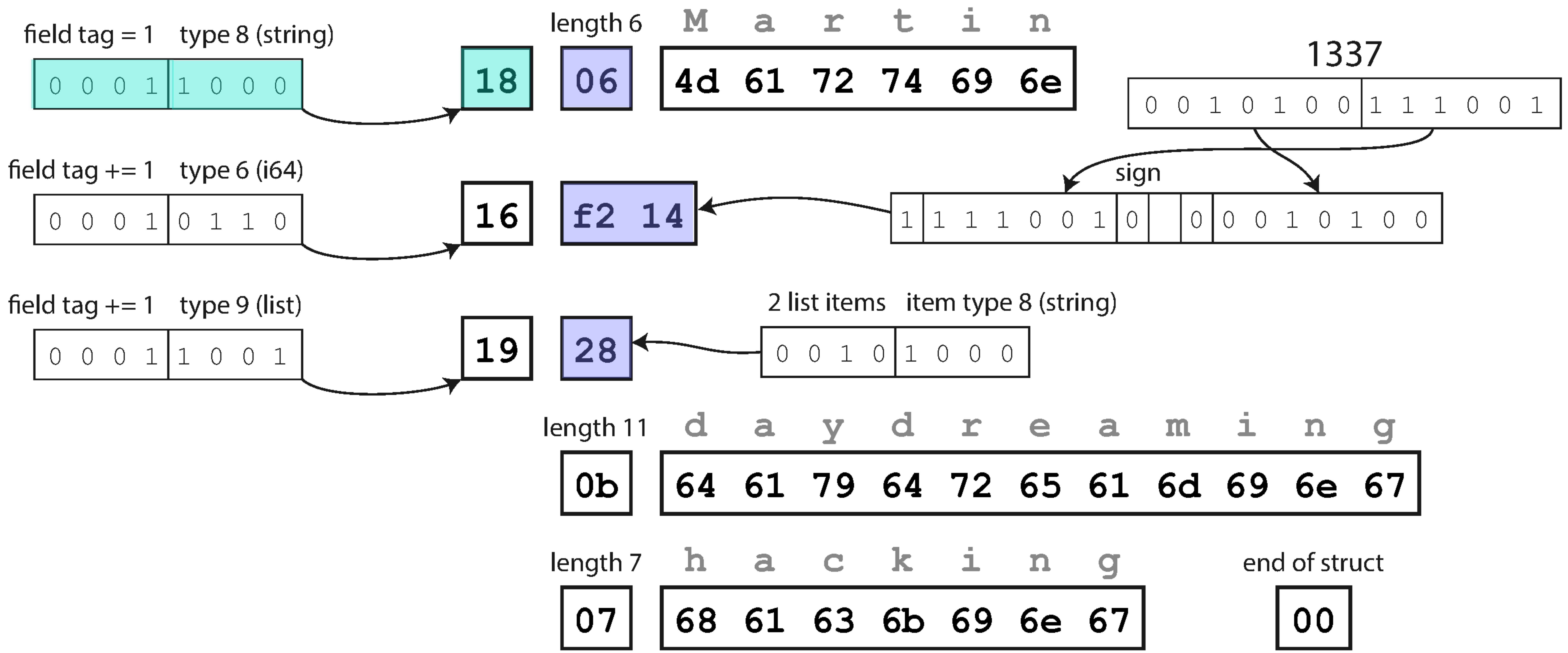
```json
{
    "userName": "Martin",
    "favoriteNumber": 1337,
    "interests": ["daydreaming", "hacking"]
}
```

Thrift CompactProtocol

Byte sequence (34 bytes):

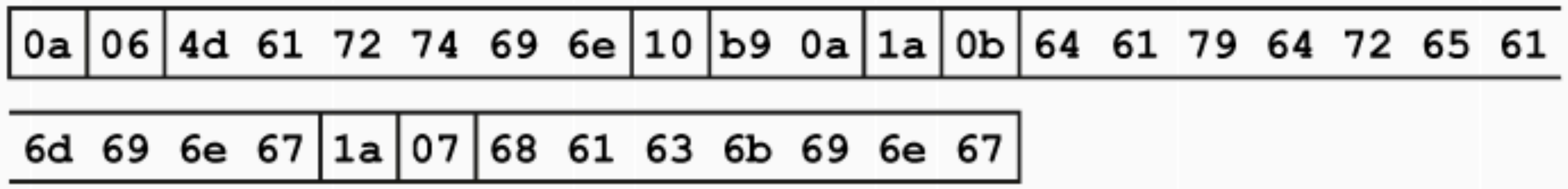
```
18 06 4d 61 72 74 69 6e 16 f2 14 19 28 0b 64 61 79 64 72 65
61 6d 69 6e 67 07 68 61 63 6b 69 6e 67 00
```

Breakdown:

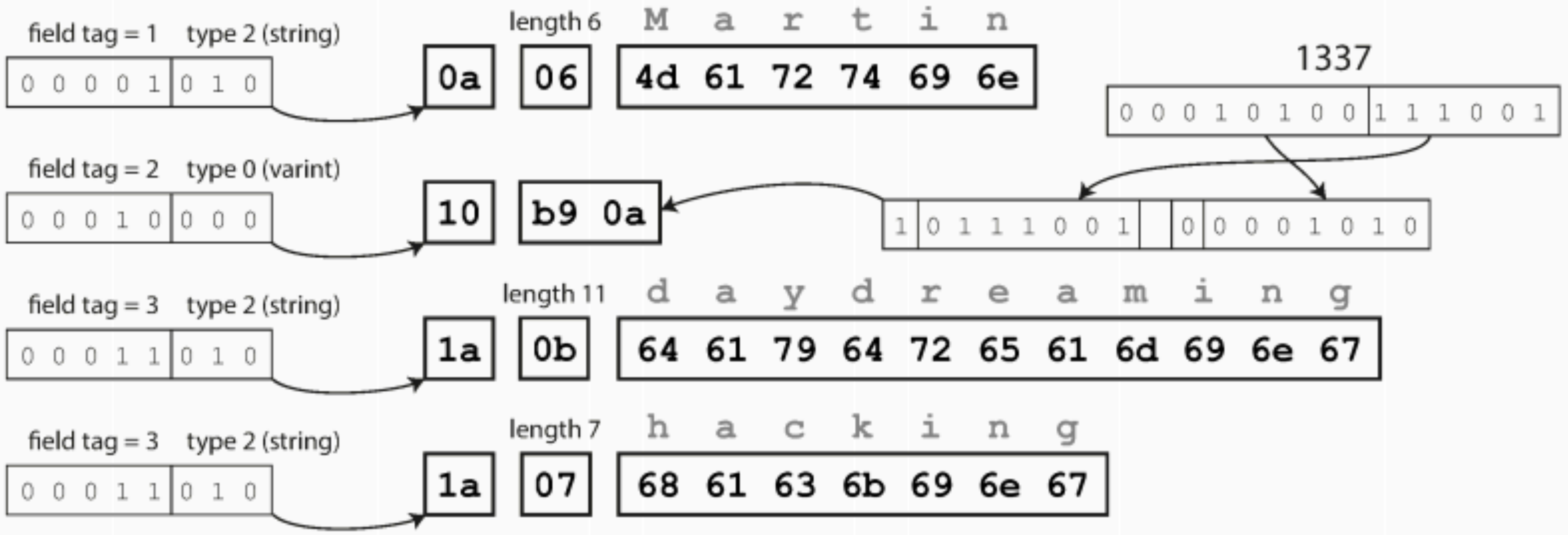


Protocol Buffers

Byte sequence (33 bytes):


0a 06 4d 61 72 74 69 6e 10 b9 0a 1a 0b 64 61 79 64 72 65 61
6d 69 6e 67 1a 07 68 61 63 6b 69 6e 67

Breakdown:


field tag = 1
type 2 (string)
0 0 0 0 1 0 1 0
0a
length 6
06
M a r t i n
4d 61 72 74 69 6e
1337
0 0 0 1 0 1 0 0 1 1 1 0 0 1
field tag = 2
type 0 (varint)
0 0 0 1 0 0 0 0
10
b9 0a
1 0 1 1 1 0 0 1 0 0 0 0 1 0 1 0
field tag = 3
type 2 (string)
0 0 0 1 1 0 1 0
1a
length 11
0b
d a y d r e a m i n g
64 61 79 64 72 65 61 6d 69 6e 67
field tag = 3
type 2 (string)
0 0 0 1 1 0 1 0
1a
length 7
07
h a c k i n g
68 61 63 6b 69 6e 67