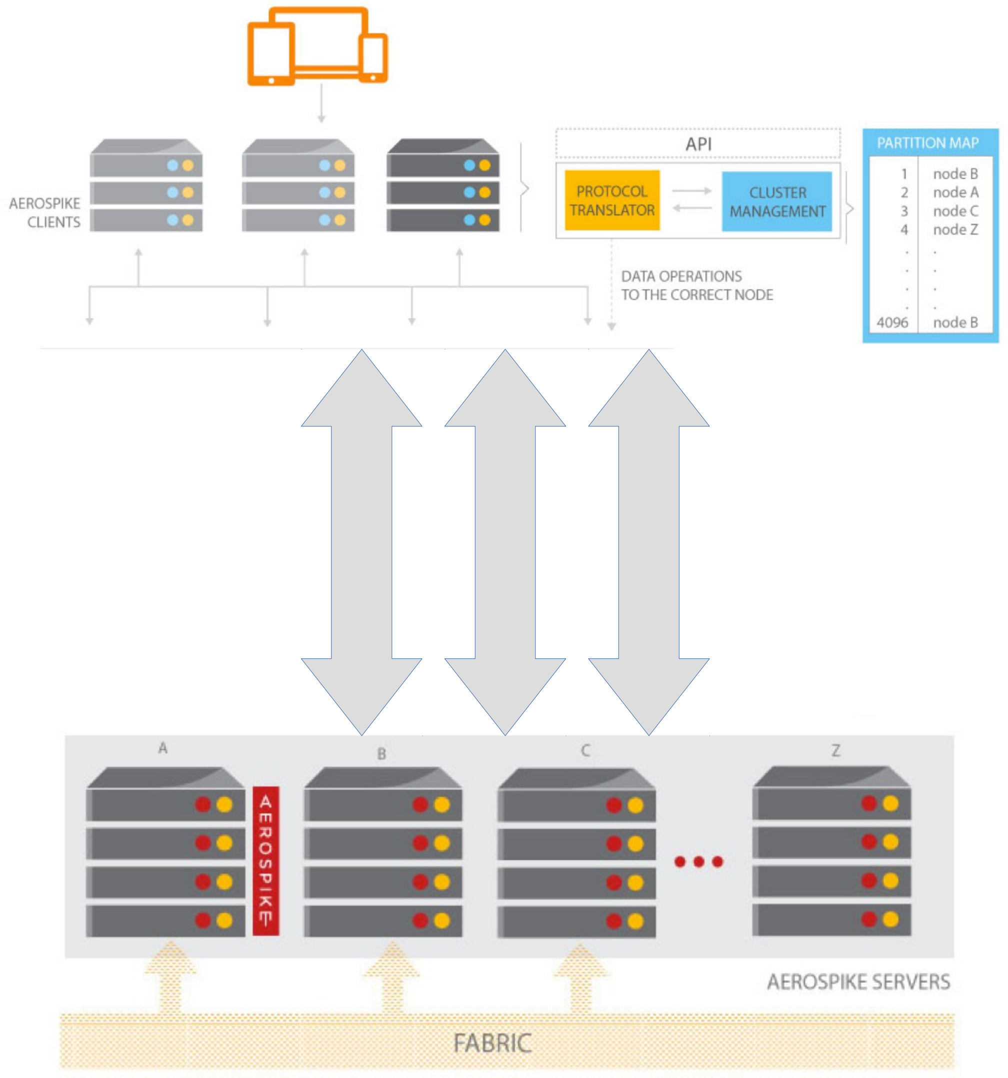
Shooting the trouble down to the Wireshark Lua Plugin

June 2019

Shakthi Kannan
Version 0.9
shakthi@aerospike.com

Motivation



Hello Production Support Engineer,
We are seeing client timeouts in our cluster,
Can you analyze the logs on the server,
And let us know how to proceed further?

"input from client unsupported proto version"
Is the message that we see for our operation,
What in the client can cause this assertion?
We need your help to provide us with a solution.

We are making batch read queries in production,
And receiving the values in a timely fashion,
Aerospike uses RIPEMD160 hash function,
But, can we also read the digests in conjunction?

AEROSPIKE

Wireshark Lua

- **Dissectors**

  Decode packet data.

- **Chained Dissectors**

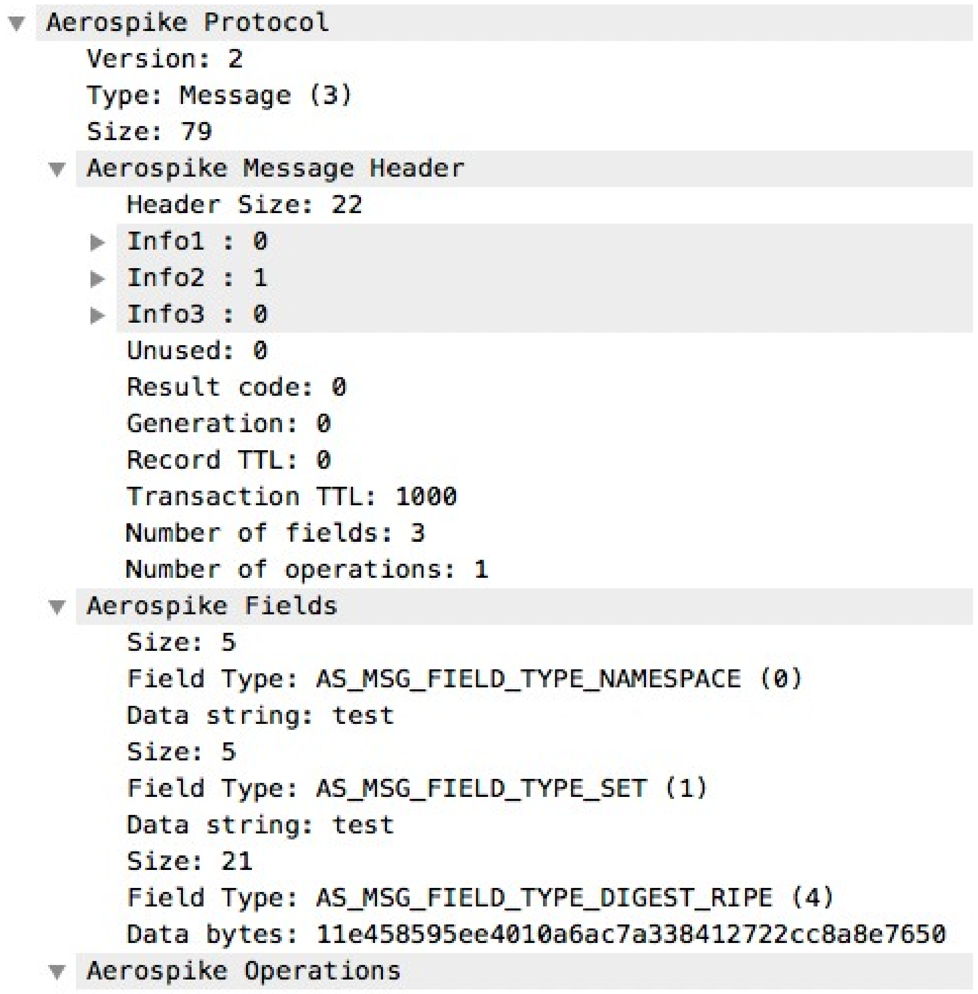
  Access to one dissector’s data.

- **Post-dissectors**

  Called after every other dissector has been called.

- **Listeners**

  Called for every packet that matches a filter or tap.



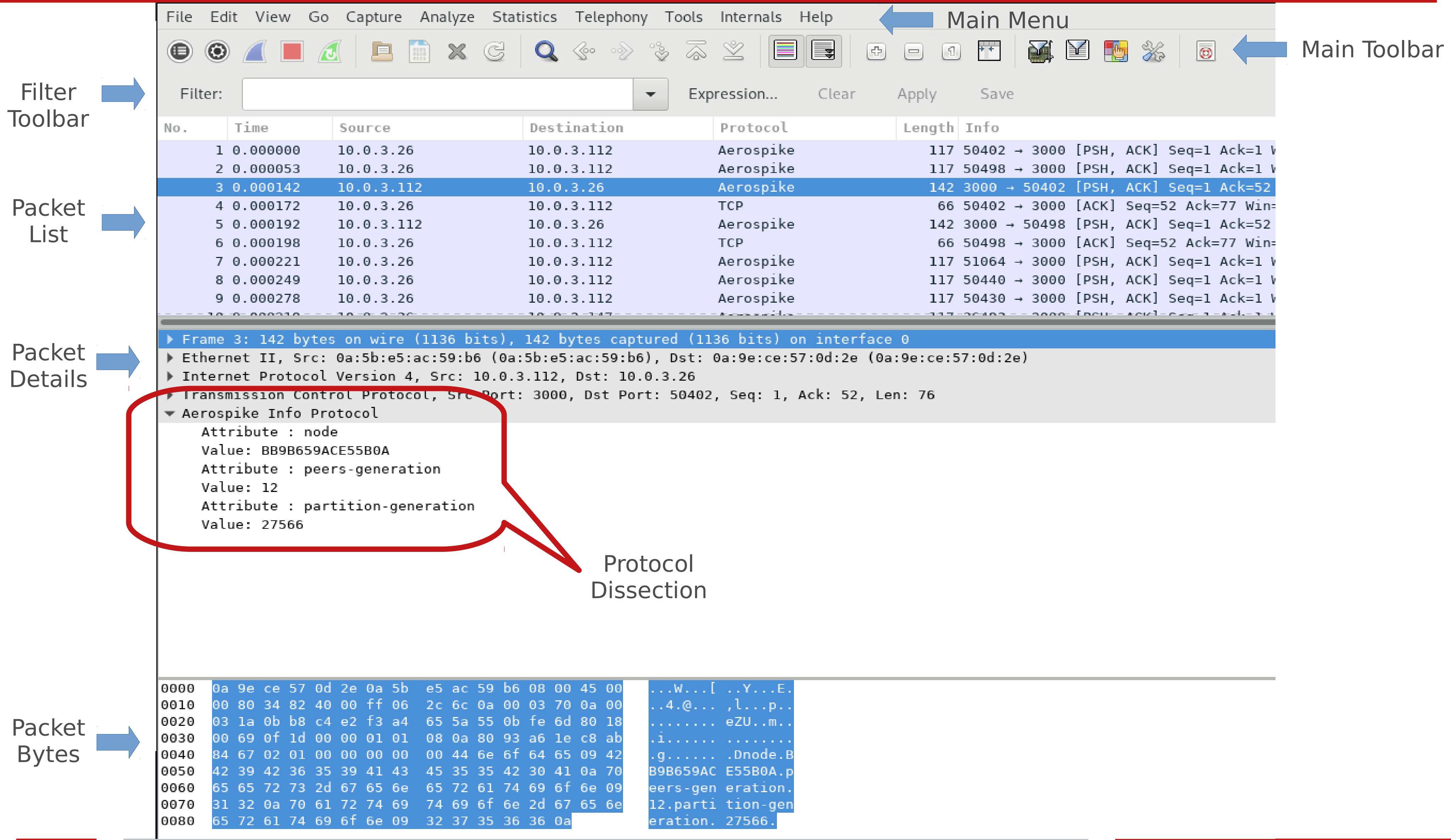
```
▼ Aerospike Protocol
    Version: 2
    Type: Message (3)
    Size: 79
  ▼ Aerospike Message Header
      Header Size: 22
    ▶ Info1 : 0
    ▶ Info2 : 1
    ▶ Info3 : 0
      Unused: 0
      Result code: 0
      Generation: 0
      Record TTL: 0
      Transaction TTL: 1000
      Number of fields: 3
      Number of operations: 1
  ▼ Aerospike Fields
      Size: 5
      Field Type: AS_MSG_FIELD_TYPE_NAMESPACE (0)
      Data string: test
      Size: 5
      Field Type: AS_MSG_FIELD_TYPE_SET (1)
      Data string: test
      Size: 21
      Field Type: AS_MSG_FIELD_TYPE_DIGEST_RIPE (4)
      Data bytes: 11e458595ee4010a6ac7a338412722cc8a8e7650
  ▼ Aerospike Operations
```

Source: **https://wiki.wireshark.org/Lua/Dissectors**

Tap Listener

```
-- simple_http.lua
-- implements a very simple tap in Lua

-- this is going to be our counter
http_packets = 0

-- this is going to be our tap
tap_http = nil

-- first we declare the tap called “http tap” with the filter it is going to use
tap_http = Listener.new(nil, “http”)

-- this function will get called at the end(3) of the capture to print the summary
function tap_http.draw()
   debug(“http packets:” .. http_packets)
end

-- this function is going to be called once each time the filter of the tap matches
function tap_http.packet()
   http_packets = http_packets + 1
end

-- this function will be called at the end of the capture run
function tap_http.reset()
   http_packets = 0
end
```

Wireshark User Interface



```
No.  Time      Source       Destination   Protocol   Length Info
  1 0.000000   10.0.3.26    10.0.3.112    Aerospike    117 50402 → 3000 [PSH, ACK] Seq=1 Ack=1 W
  2 0.000053   10.0.3.26    10.0.3.112    Aerospike    117 50498 → 3000 [PSH, ACK] Seq=1 Ack=1 W
  3 0.000142   10.0.3.112   10.0.3.26     Aerospike    142 3000 → 50402 [PSH, ACK] Seq=1 Ack=52
  4 0.000172   10.0.3.26    10.0.3.112    TCP           66 50402 → 3000 [ACK] Seq=52 Ack=77 Win=
  5 0.000192   10.0.3.112   10.0.3.26     Aerospike    142 3000 → 50498 [PSH, ACK] Seq=1 Ack=52
  6 0.000198   10.0.3.26    10.0.3.112    TCP           66 50498 → 3000 [ACK] Seq=52 Ack=77 Win=
  7 0.000221   10.0.3.26    10.0.3.112    Aerospike    117 51064 → 3000 [PSH, ACK] Seq=1 Ack=1 W
  8 0.000249   10.0.3.26    10.0.3.112    Aerospike    117 50440 → 3000 [PSH, ACK] Seq=1 Ack=1 W
  9 0.000278   10.0.3.26    10.0.3.112    Aerospike    117 50430 → 3000 [PSH, ACK] Seq=1 Ack=1 W
```

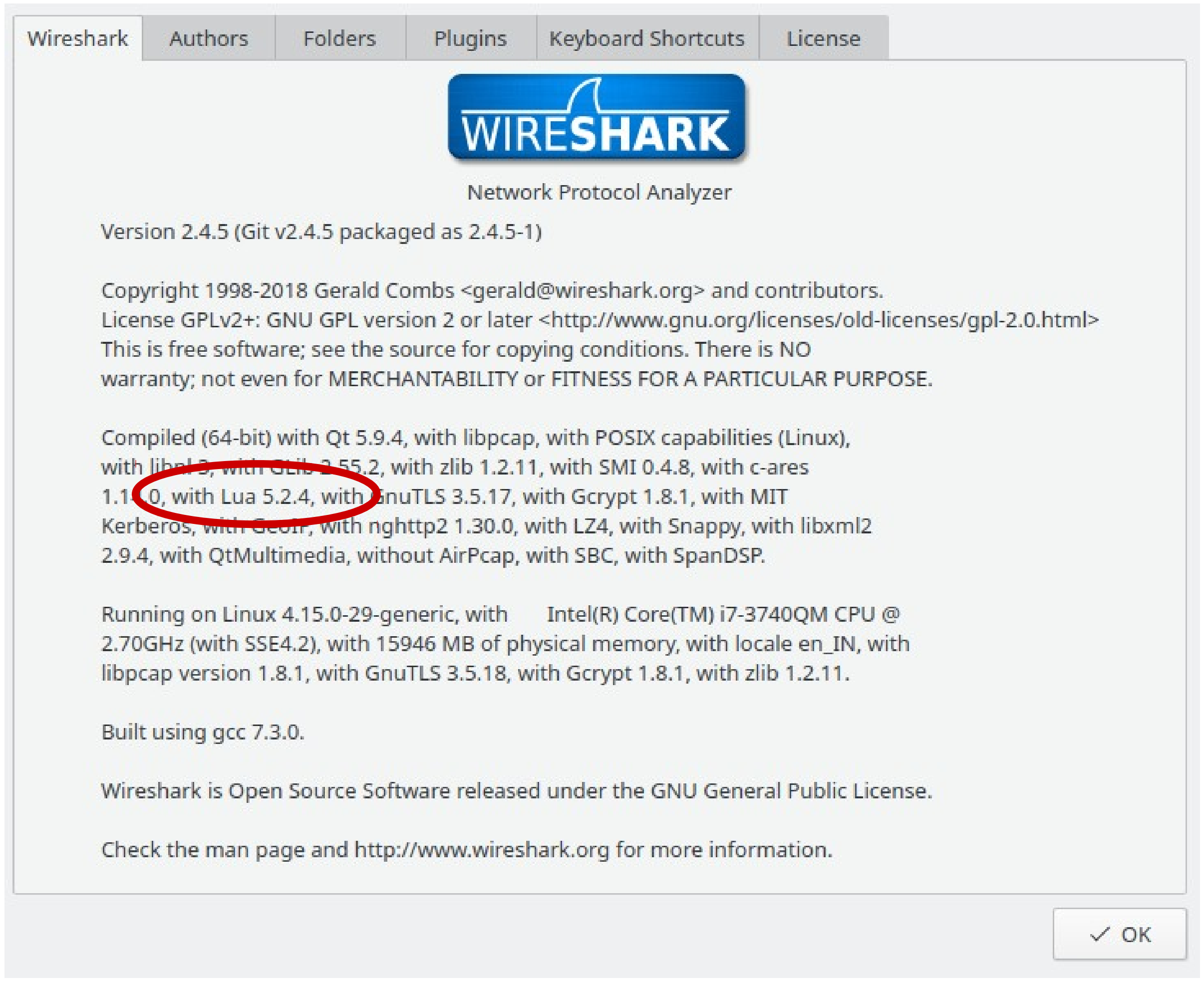
```
▸ Frame 3: 142 bytes on wire (1136 bits), 142 bytes captured (1136 bits) on interface 0
▸ Ethernet II, Src: 0a:5b:e5:ac:59:b6 (0a:5b:e5:ac:59:b6), Dst: 0a:9e:ce:57:0d:2e (0a:9e:ce:57:0d:2e)
▸ Internet Protocol Version 4, Src: 10.0.3.112, Dst: 10.0.3.26
▸ Transmission Control Protocol, Src Port: 3000, Dst Port: 50402, Seq: 1, Ack: 52, Len: 76
▾ Aerospike Info Protocol
    Attribute : node
    Value: BB9B659ACE55B0A
    Attribute : peers-generation
    Value: 12
    Attribute : partition-generation
    Value: 27566
```

```
0000  0a 9e ce 57 0d 2e 0a 5b  e5 ac 59 b6 08 00 45 00   ...W...[ ..Y...E.
0010  00 80 34 82 40 00 ff 06  2c 6c 0a 00 03 70 0a 00   ..4.@... ,l...p..
0020  03 1a 0b b8 c4 e2 f3 a4  65 5a 55 0b fe 6d 80 18   ........ eZU..m..
0030  00 69 0f 1d 00 00 01 01  08 0a 80 93 a6 1e c8 ab   .i...... ........
0040  84 67 02 01 00 00 00 00  00 44 6e 6f 64 65 09 42   .g...... .Dnode.B
0050  42 39 42 36 35 39 41 43  45 35 35 42 30 41 0a 70   B9B659AC E55B0A.p
0060  65 65 72 73 2d 67 65 6e  65 72 61 74 69 6f 6e 09   eers-gen eration.
0070  31 32 0a 70 61 72 74 69  74 69 6f 6e 2d 67 65 6e   12.parti tion-gen
0080  65 72 61 74 69 6f 6e 09  32 37 35 36 36 0a         eration. 27566.
```

Usage

Help -> About Wireshark

Free (Libre) Open Source



Network Protocol Analyzer

Version 2.4.5 (Git v2.4.5 packaged as 2.4.5-1)

Copyright 1998-2018 Gerald Combs <gerald@wireshark.org> and contributors.
License GPLv2+: GNU GPL version 2 or later <http://www.gnu.org/licenses/old-licenses/gpl-2.0.html>
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

Compiled (64-bit) with Qt 5.9.4, with libpcap, with POSIX capabilities (Linux),
with libnl 3, with GLib 2.55.2, with zlib 1.2.11, with SMI 0.4.8, with c-ares
1.14.0, with Lua 5.2.4, with GnuTLS 3.5.17, with Gcrypt 1.8.1, with MIT
Kerberos, with GeoIP, with nghttp2 1.30.0, with LZ4, with Snappy, with libxml2
2.9.4, with QtMultimedia, without AirPcap, with SBC, with SpanDSP.

Running on Linux 4.15.0-29-generic, with Intel(R) Core(TM) i7-3740QM CPU @
2.70GHz (with SSE4.2), with 15946 MB of physical memory, with locale en_IN, with
libpcap version 1.8.1, with GnuTLS 3.5.18, with Gcrypt 1.8.1, with zlib 1.2.11.

Built using gcc 7.3.0.

Wireshark is Open Source Software released under the GNU General Public License.

Check the man page and http://www.wireshark.org for more information.

```
$ wireshark -X lua_script:aerospike.lua capture.pcapng
$ tshark    -X lua_script:aerospike.lua capture.pcapng
```

You can also place plugins in ~/.wireshark/plugins folder.

Hello World Lua!

```
$ lua -v
Lua 5.3.4  Copyright (C) 1994-2017 Lua.org, PUC-Rio


$ lua
Lua 5.3.4  Copyright (C) 1994-2017 Lua.org, PUC-Rio
> print(“Hello, World!”)
Hello World

$ cat hello_world.lua
#!/usr/bin/lua

print(“Hello, World!”)

$ lua hello_world.lua
Hello, World!
```

Lua: Assignment and Operations

```
$ lua
Lua 5.3.4  Copyright (C) 1994-2017 Lua.org, PUC-Rio
> i, j = 1, 2
> -3
-3
> k = i + j
> k
3
> j ^ k
8.0
> k % j
1
> not i
false
```

Category	Operator	Associativity
Unary	not # -	Right to left
Concatenation	..	Right to left
Multiplicative	* / %	Left to right
Additive	+ -	Left to right
Relational	< > <= >= == ~=	Left to right
Equality	== ~=	Left to right
Logical AND	and	Left to right
Logical OR	or	Left to right

Lua: Strings

```
> name = “Lua”
> type(name)
string
> print(name .. “ Language”)
Lua Language
> print(“99” + 1)
100.0
> print(“Value of k”, k)
Value of k 3
> [[ This is also a string ]]
This is also a string
> print(“Lua’s \”syntax\” is simple!”)
Lua’s “syntax” is simple!
```

Escape Sequence	Use
\a	Bell
\b	Backspace
\f	Form feed
\n	New line
\r	Carriage return
\t	Tab
\v	Vertical tab
\\	Backslash
\”	Double quotes
\[	Left square bracket
\]	Right square bracket

Lua: Tables

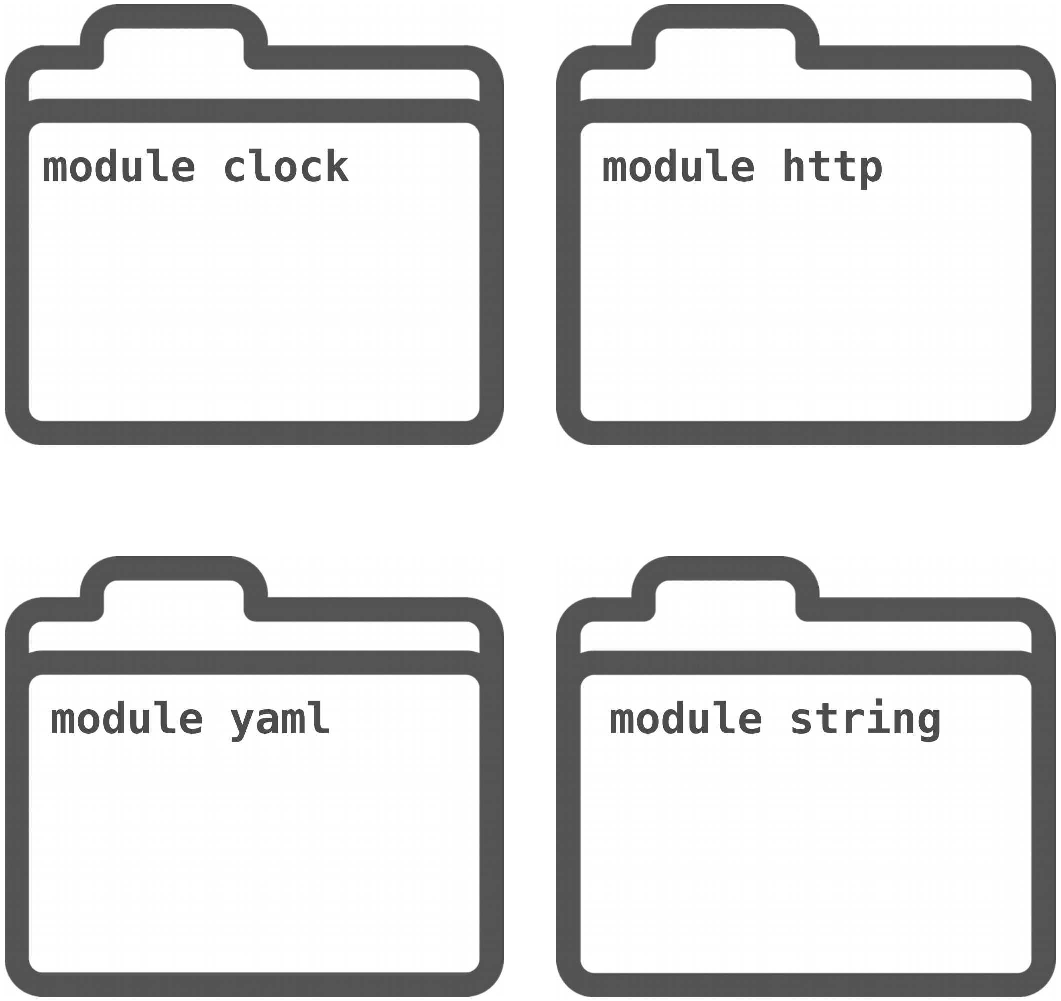
```
> work_days = {“Mon”, “Tue”, “Wed”, “Thu”, “Fri”}
> work_days
table: 0x9f10b0
> work_days[0]
nil
> work_days[1]
Mon
> work_days[6] = “Sat”
> table.insert(work_days, “Sun”)
> work_days[7]
Sun
```

table: 0x9f10b0

1	Mon
2	Tue
3	Wed
4	Thu
5	Fri
6	Sat
7	Sun

Lua: Functions

```
-- fact.lua
function fact (n)
   local f = 1
   for i = 2, n do
     f = f * i
   end
   return f
end

print(fact(5))

$ lua fact.lua
120
```



Lua: Conditions and Loops

```lua
if number < 10 then
   print(“Less than 10”)
else
   print(“Greater than 10”)
end
```

```lua
while number < 10 do
   print(number)
   number = number + 1
end
```

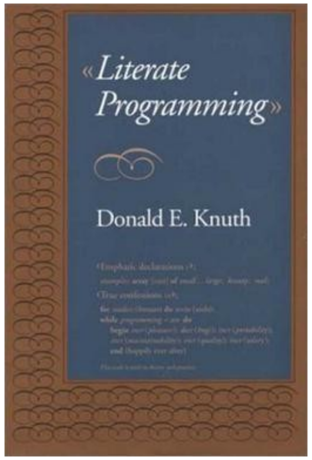
```lua
repeat
   print(number)
   number = number + 1
until number >= 10
```

```lua
for number = 0, 9, 1 do
   print(number)
end
```

Literate Programming

"I believe that the time is ripe for significantly better documentation of programs, and that we can best achieve this by considering programs to be works of literature. Hence, my title: "Literate Programming."

Let us change our traditional attitude to the construction of programs: Instead of imagining that our main task is to instruct a computer what to do, let us concentrate rather on explaining to human beings what we want a computer to do."



~ Prof. Donald E. Knuth, 1984

Markdown Structure

```
# Requires...
# Configuration...
## Common...
# Helper Functions...
# Statistics...
## Hot Key...
# GUI...
# Protocols...
## Info...
## Batch...
## Message...
### Aerospike Message: Header Section...
### Aerospike Message: Fields...
### Aerospike Message: Operations...
### Functions...
## Heartbeat...
# The Main...
```

lit2lua

```
## Heartbeat

Heartbeat protocol

>  +---------------------------+-----------------------------+
>  |    Message Header         |      Message Fields         |
>  +---------------------------+-----------------------------+

Message Header

>  +-------------+--------------+
>  |   size      |    type      |
>  +-------------+--------------+
>  0             4              6

Constants

    local HB_HEADER_SZ_START    = 0
    local HB_HEADER_SZ_LENGTH   = 4
    local HB_HEADER_TYPE_START  = 4
    local HB_HEADER_TYPE_LENGTH = 2
```

lit2lua …

Op

Value	Name	Description
1	AS_MSG_OP_READ	Read the value
2	AS_MSG_OP_WRITE	Write the value
3	AS_MSG_OP_CDT_READ	Prospective CDT top-level ops
4	AS_MSG_OP_CDT_MODIFY	Prospective CDT top-level ops
5	AS_MSG_OP_INCR	Add a value to an existing value (only on integers)
6	Unused	Reserved
7	Unused	Reserved
8	Unused	Reserved
9	AS_MSG_OP_APPEND	Append a value to an existing value (on strings and blobs)
10	AS_MSG_OP_PREPEND	Prepend a value to an existing value (on strings and blobs)
11	AS_MSG_OP_TOUCH	Touch a value (will only increment the generation)
129	AS_MSG_OP_MC_INCR	Memcache-compatible version of the increment command
130	AS_MSG_OP_MC_APPEND	Append value to existing value (only on strings)
131	AS_MSG_OP_MC_PREPEND	Prepend a value to an existing value (only on strings)
132	AS_MSG_OP_MC_TOUCH	Memcache-compatible touch (does not change generation)

To generate the Wireshark Lua plugin from documentation, use:

```
$ less -f “docs/aerospike.lua.md” | lit2lua > lua/aerospike.lua
```

lit2lua …

Table Definition

```
local TYPES_OPS = {
       [1] = “AS_MSG_OP_READ”,
       [2] = “AS_MSG_OP_WRITE”,
       [3] = “AS_MSG_AP_CDT_READ”,
       [4] = “AS_MSG_OP_CDT_MODIFY”,
       [5] = “AS_MSG_OP_INCR”,
       [6] = “Unused”,
       [7] = “Unused”,
       [8] = “Unused”,
       [9] = “AS_MSG_OP_APPEND”,
      [10] = “AS_MSG_OP_PREPEND”,
      [11] = “AS_MSG_OP_TOUCH”,
     [129] = “AS_MSG_OP_MC_INCR”,
     [130] = “AS_MSG_OP_MC_APPEND”,
     [131] = “AS_MSG_OP_MC_PREPEND”,
     [132] = “AS_MSG_OP_MC_TOUCH”,
}
```

Protocol Dissection Pattern

Constants

```
local PROTO_VERSION_START  = 0
local PROTO_VERSION_LENGTH = 1

local PROTO_TYPE_START  = 1
local PROTO_TYPE_LENGTH = 1

local PROTO_TYPE_INFO = 1
local PROTO_TYPE_MSG  = 3

local INFO_SIZE_START  = 2
local INFO_SIZE_LENGTH = 6
local INFO_DATA_START  = 8
```

```
>  +---------+-------------+-----------------------------------+
>  | version |    type     |               size                |
>  +---------+-------------+-----------------------------------+
>  0         1             2                                   8
```

Protocol Dissection Pattern …

Create Proto objects

```
   local aerospike_info_proto      = Proto(“Aerospike”,               “Aerospike Info Protocol”)
   local aerospike_attribute       = Proto(“AerospikeAttribute”,      “Aerospike Attributes”)
   local aerospike_attribute_value = Proto(“AerospikeAttributeValue”, “Aerospike Attribute Value pairs”)
```

Proto header fields

```
   local header_fields = {
      version = ProtoField.uint8  (“header.version”, “Version”, base.DEC),
      type    = ProtoField.uint8  (“header.type”,    “Type”,    base.DEC, TYPES_PROTO),
      size    = ProtoField.uint64 (“header.size”,    “Size”,    base.DEC),
   }

   local header_attributes = {
      attribute = ProtoField.string(“header.attribute”, “Attribute”),
   }

   local header_attribute_values = {
      attribute = ProtoField.string(“header_attribute_values.attribute”,  “Attribute “),
      value     = ProtoField.string(“header_attribute_values.value”,      “Value”),
   }
```

Register the protocol fields

```
   aeropike_info_proto.fields       = header_fields
   aerospike_attribute.fields       = header_attributes
   aerospike_attribute_value.fields = header_attribute_values
```

Protocol Dissection Pattern ...

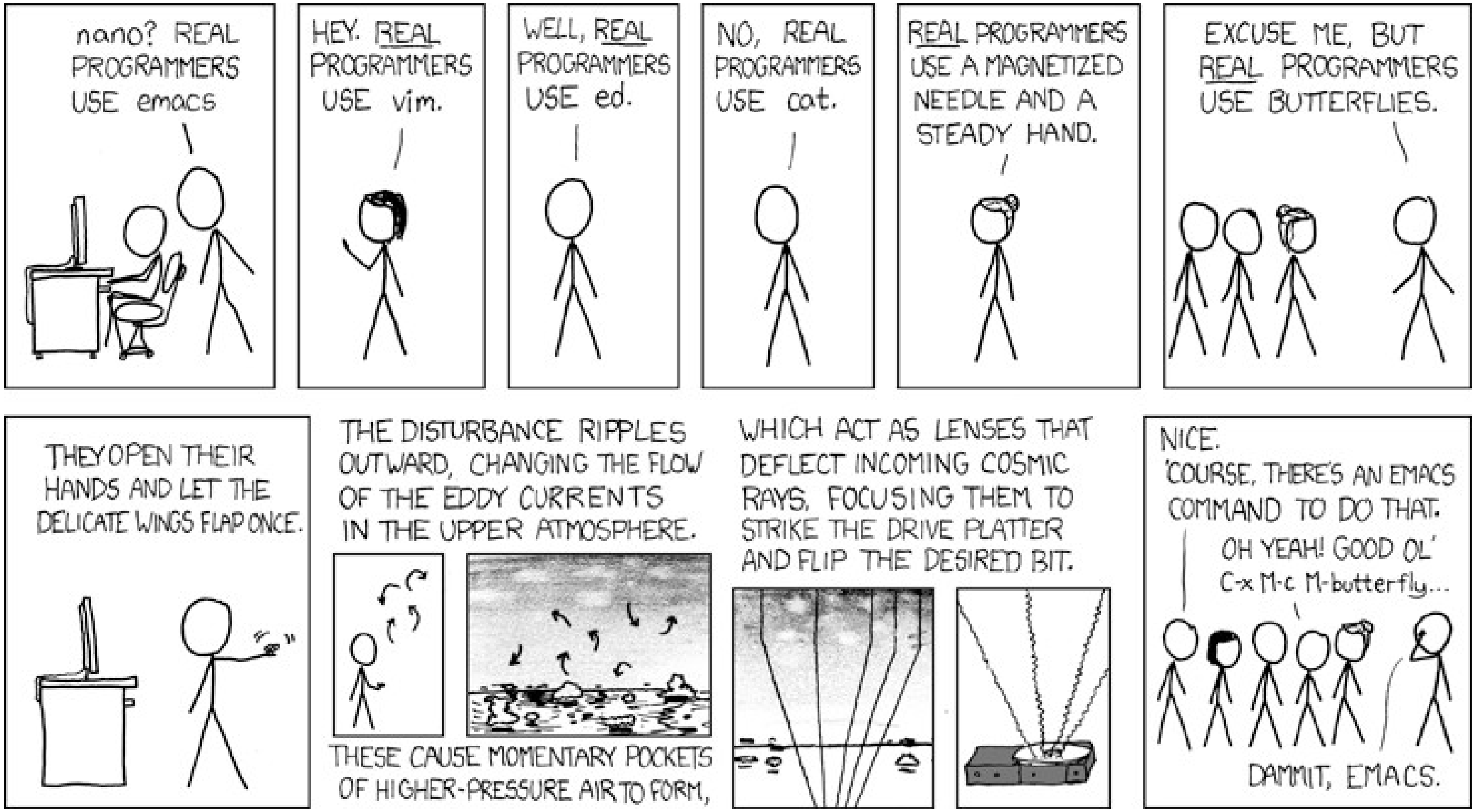
Functions

```
local function dissect_aerospike_info (tvbuf, tree, size)
   -- Separate the data by newline
   local data_tvbr = tvbuf:range(INFO_DATA_START, tonumber(size))
   local data_string = data_tvbr:string()

   local data_start = INFO_DATA_START
   for line in string.gmatch(data_string, "[^\n]+") do
      local d = tvbuf:range(data_start, string.len(line))
      local d_string = d:string()

      -- if contains attribute-values
      if string.find(d_string, "\t") then
         local parts = split_tab(d_string)
         ...
      end
      data_start = data_start + string.len(line) + 1 -- for \n
  end
end
```

Dissector Table

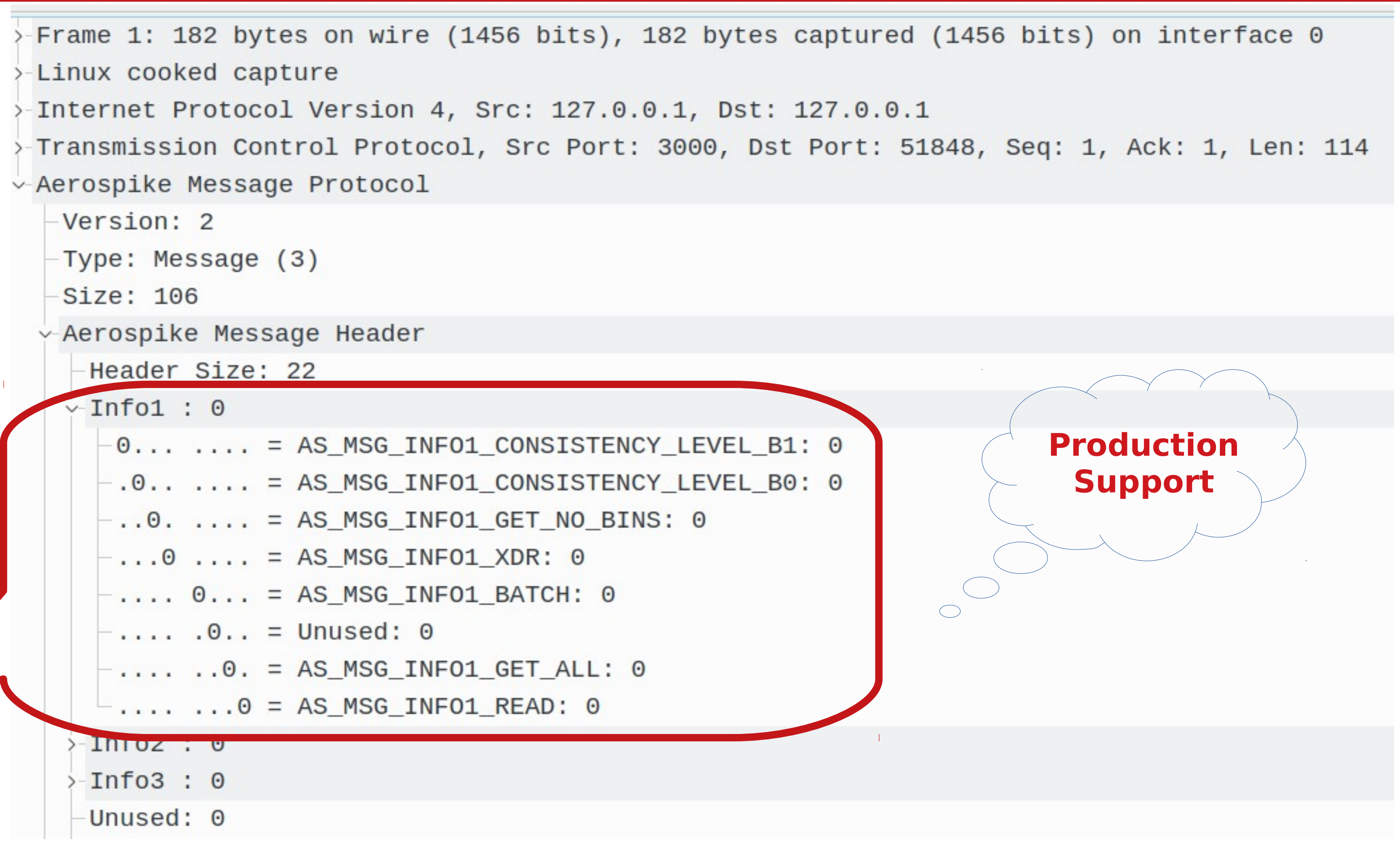
```
# Configuration
local default_settings = {
   aerospike_port           = 3000,
   heartbeat_multicast_port = 9918,
   heartbeat_mesh_port      = 3002,
}
# Create Proto objects
local aerospike_proto = Proto(“AerospikeProtocol”,  “Aerospike Protocol”)
local heartbeat_proto = Proto(“AerospikeHeartbeat”, “Aerospike Heartbeat”)


# The Main
local function enable_dissector()
   DissectorTable.get(“tcp.port”):add(
      default_settings.aerospike_port, aerospike_proto)
   DissectorTable.get(“tcp.port”):add(
      default_settings.heartbeat_mesh_port, heartbeat_proto)
   DissectorTable.get(“udp.port”):add(
      default_settings.heartbeat_multicast_port, heartbeat_proto)
end

enable_dissector()
```

Live Coding!


nano? REAL PROGRAMMERS USE emacs
HEY. REAL PROGRAMMERS USE vim.
WELL, REAL PROGRAMMERS USE ed.
NO, REAL PROGRAMMERS USE cat.
REAL PROGRAMMERS USE A MAGNETIZED NEEDLE AND A STEADY HAND.
EXCUSE ME, BUT REAL PROGRAMMERS USE BUTTERFLIES.
THEY OPEN THEIR HANDS AND LET THE DELICATE WINGS FLAP ONCE.
THE DISTURBANCE RIPPLES OUTWARD, CHANGING THE FLOW OF THE EDDY CURRENTS IN THE UPPER ATMOSPHERE.
THESE CAUSE MOMENTARY POCKETS OF HIGHER-PRESSURE AIR TO FORM,
WHICH ACT AS LENSES THAT DEFLECT INCOMING COSMIC RAYS, FOCUSING THEM TO STRIKE THE DRIVE PLATTER AND FLIP THE DESIRED BIT.
NICE.
'COURSE, THERE'S AN EMACS COMMAND TO DO THAT.
OH YEAH! GOOD OL' C-x M-c M-butterfly...
DAMMIT, EMACS.

Source: https://www.xkcd.com/378/

Message Protocol

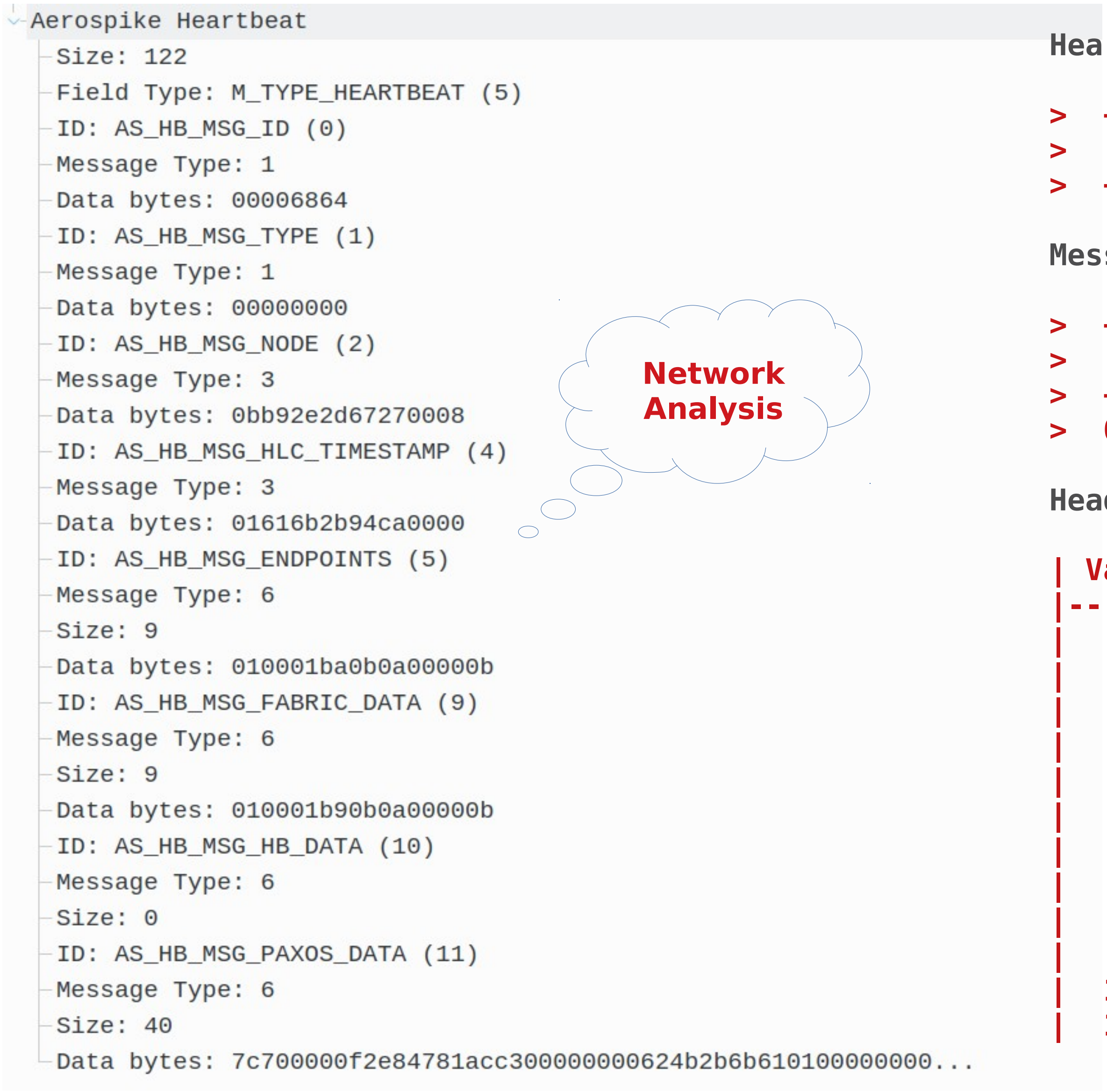


```
> Frame 1: 182 bytes on wire (1456 bits), 182 bytes captured (1456 bits) on interface 0
> Linux cooked capture
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
> Transmission Control Protocol, Src Port: 3000, Dst Port: 51848, Seq: 1, Ack: 1, Len: 114
v Aerospike Message Protocol
    Version: 2
    Type: Message (3)
    Size: 106
  v Aerospike Message Header
      Header Size: 22
    v Info1 : 0
        0... .... = AS_MSG_INFO1_CONSISTENCY_LEVEL_B1: 0
        .0.. .... = AS_MSG_INFO1_CONSISTENCY_LEVEL_B0: 0
        ..0. .... = AS_MSG_INFO1_GET_NO_BINS: 0
        ...0 .... = AS_MSG_INFO1_XDR: 0
        .... 0... = AS_MSG_INFO1_BATCH: 0
        .... .0.. = Unused: 0
        .... ..0. = AS_MSG_INFO1_GET_ALL: 0
        .... ...0 = AS_MSG_INFO1_READ: 0
    > Info2 : 0
    > Info3 : 0
      Unused: 0
```

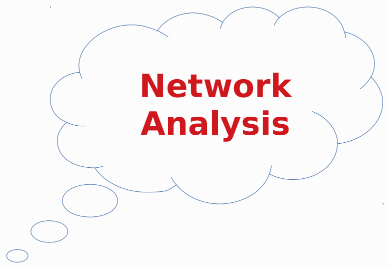
Bit-level Dissection

Value	Name	Description
1	AS_MSG_INFO1_READ	Contains a read operation
2	AS_MSG_INFO1_GET_ALL	Get all bins data
4	Unused	Unused
8	AS_MSG_INFO1_BATCH	New batch protocol
16	AS_MSG_INFO1_XDR	Operation is performed by XDR
32	AS_MSG_INFO1_GET_NO_BINS	Do not read the bin information
64	AS_MSG_INFO1_CONSISTENCY_LEVEL_B0	Read consistency level – bit 0
128	AS_MSG_INFO1_CONSISTENCY_LEVEL_B1	Read consistency level – bit 1

Heartbeat Protocol



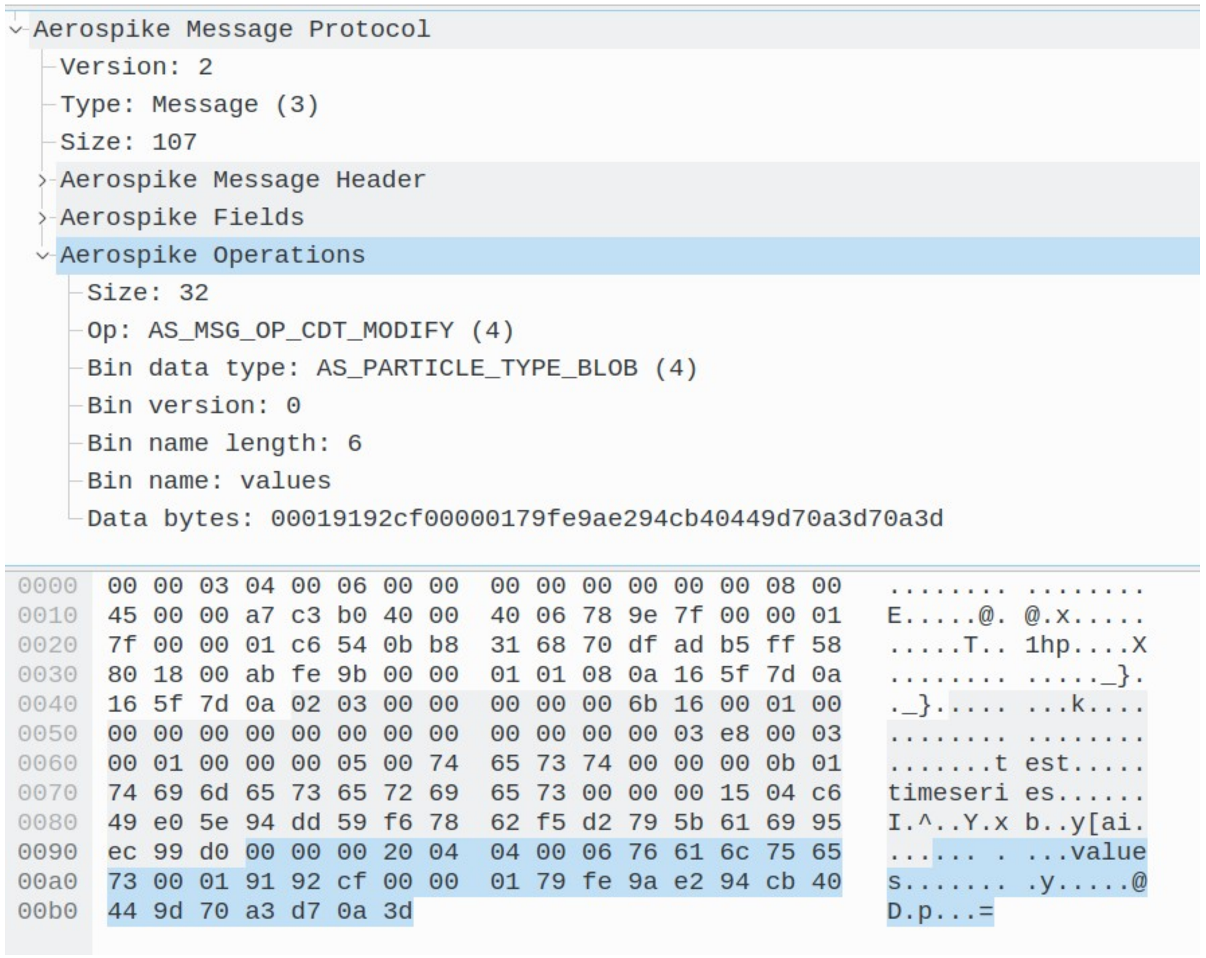
```
Aerospike Heartbeat
  Size: 122
  Field Type: M_TYPE_HEARTBEAT (5)
  ID: AS_HB_MSG_ID (0)
  Message Type: 1
  Data bytes: 00006864
  ID: AS_HB_MSG_TYPE (1)
  Message Type: 1
  Data bytes: 00000000
  ID: AS_HB_MSG_NODE (2)
  Message Type: 3
  Data bytes: 0bb92e2d67270008
  ID: AS_HB_MSG_HLC_TIMESTAMP (4)
  Message Type: 3
  Data bytes: 01616b2b94ca0000
  ID: AS_HB_MSG_ENDPOINTS (5)
  Message Type: 6
  Size: 9
  Data bytes: 010001ba0b0a00000b
  ID: AS_HB_MSG_FABRIC_DATA (9)
  Message Type: 6
  Size: 9
  Data bytes: 010001b90b0a00000b
  ID: AS_HB_MSG_HB_DATA (10)
  Message Type: 6
  Size: 0
  ID: AS_HB_MSG_PAXOS_DATA (11)
  Message Type: 6
  Size: 40
  Data bytes: 7c700000f2e84781acc300000000624b2b6b610100000000...
```



```
Heartbeat protocol

>  +------------------+----------------+
>  |  Message Header  | Message Fields |
>  +------------------+----------------+

Message Header

>  +-------------+-------------+
>  |   size      |     type    |
>  +-------------+-------------+
>  0             4             6

Header Type
```

Value	Name
0	M_TYPE_FABRIC
1	M_TYPE_HEARTBEAT_V2
2	M_TYPE_PAXOS
3	M_TYPE_MIGRATE
4	M_TYPE_PROXY
5	M_TYPE_HEARTBEAT
6	M_TYPE_CLUSTERING
7	M_TYPE_RW
8	M_TYPE_INFO
9	M_TYPE_EXCHANGE
11	M_TYPE_XDR
15	M_TYPE_SMD

CDT List Operations



```
Aerospike Message Protocol
  Version: 2
  Type: Message (3)
  Size: 107
  Aerospike Message Header
  Aerospike Fields
  Aerospike Operations
    Size: 32
    Op: AS_MSG_OP_CDT_MODIFY (4)
    Bin data type: AS_PARTICLE_TYPE_BLOB (4)
    Bin version: 0
    Bin name length: 6
    Bin name: values
    Data bytes: 00019192cf00000179fe9ae294cb40449d70a3d70a3d
```

```
0000  00 00 03 04 00 06 00 00  00 00 00 00 00 00 08 00   ........ ........
0010  45 00 00 a7 c3 b0 40 00  40 06 78 9e 7f 00 00 01   E.....@. @.x.....
0020  7f 00 00 01 c6 54 0b b8  31 68 70 df ad b5 ff 58   .....T.. 1hp....X
0030  80 18 00 ab fe 9b 00 00  01 01 08 0a 16 5f 7d 0a   ........ ....._}.
0040  16 5f 7d 0a 02 03 00 00  00 00 00 6b 16 00 01 00   ._}..... ...k....
0050  00 00 00 00 00 00 00 00  00 00 00 00 03 e8 00 03   ........ ........
0060  00 01 00 00 00 05 00 74  65 73 74 00 00 00 0b 01   .......t est.....
0070  74 69 6d 65 73 65 72 69  65 73 00 00 00 15 04 c6   timeseri es......
0080  49 e0 5e 94 dd 59 f6 78  62 f5 d2 79 5b 61 69 95   I.^..Y.x b..y[ai.
0090  ec 99 d0 00 00 00 20 04  04 00 06 76 61 6c 75 65   ...... . ...value
00a0  73 00 01 91 92 cf 00 00  01 79 fe 9a e2 94 cb 40   s....... .y.....@
00b0  44 9d 70 a3 d7 0a 3d                               D.p...=
```

Modifying list of lists

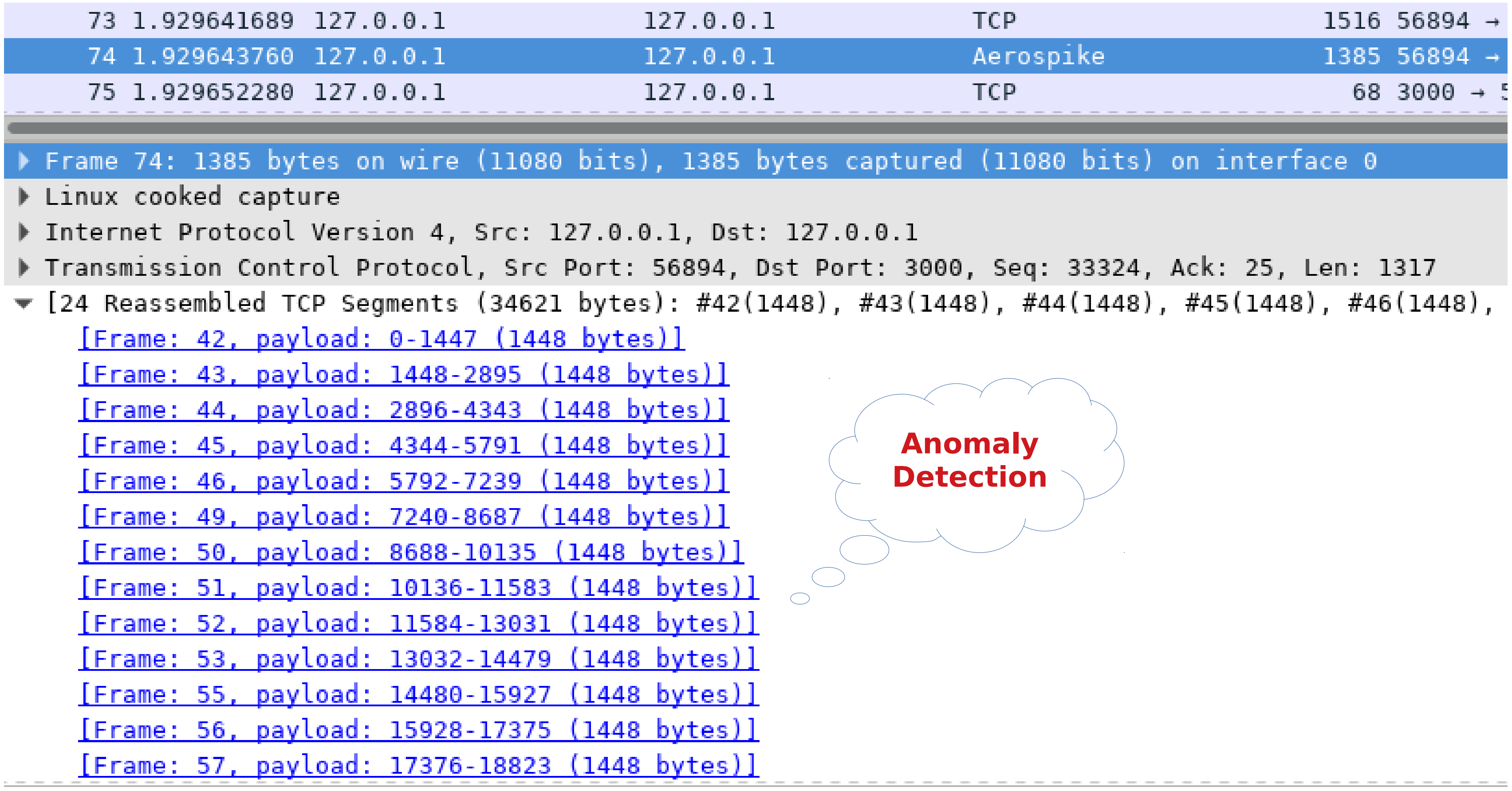
‘values’ bin:
[[1523474230000, 39.04],
[1523474231001, 39.78],
[1523474236006, 40.07],
[1523474235005, 41.18],
[1523474233003, 40.89],
[1523474234004, 40.93]]

```
client.list_append(
        key,
        ‘values’,
        [1623474234004, 41.23])
```

'values' bin:
[[1523474230000, 39.04],
[1523474231001, 39.78],
[1523474236006, 40.07],
[1523474235005, 41.18],
[1523474233003, 40.89],
[1523474234004, 40.93],
[1623474234004, 41.23]]

Source: **https://www.aerospike.com/docs/guide/cdt-list-ops.html**

Reassembled TCP Segments

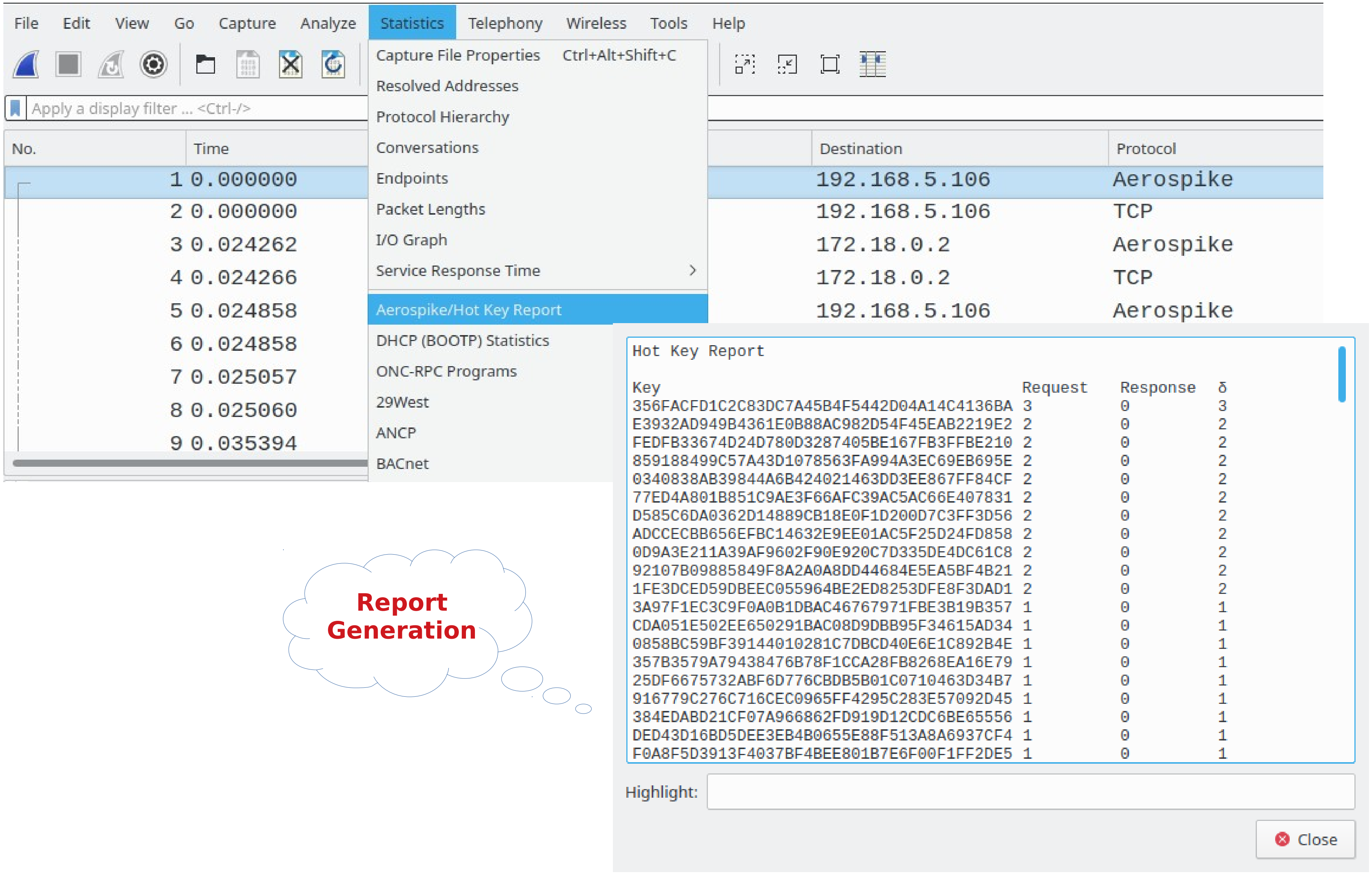


```
73 1.929641689  127.0.0.1           127.0.0.1           TCP            1516 56894 →
74 1.929643760  127.0.0.1           127.0.0.1           Aerospike      1385 56894 →
75 1.929652280  127.0.0.1           127.0.0.1           TCP              68 3000 → 5

▸ Frame 74: 1385 bytes on wire (11080 bits), 1385 bytes captured (11080 bits) on interface 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
▸ Transmission Control Protocol, Src Port: 56894, Dst Port: 3000, Seq: 33324, Ack: 25, Len: 1317
▾ [24 Reassembled TCP Segments (34621 bytes): #42(1448), #43(1448), #44(1448), #45(1448), #46(1448),
    [Frame: 42, payload: 0-1447 (1448 bytes)]
    [Frame: 43, payload: 1448-2895 (1448 bytes)]
    [Frame: 44, payload: 2896-4343 (1448 bytes)]
    [Frame: 45, payload: 4344-5791 (1448 bytes)]
    [Frame: 46, payload: 5792-7239 (1448 bytes)]
    [Frame: 49, payload: 7240-8687 (1448 bytes)]
    [Frame: 50, payload: 8688-10135 (1448 bytes)]
    [Frame: 51, payload: 10136-11583 (1448 bytes)]
    [Frame: 52, payload: 11584-13031 (1448 bytes)]
    [Frame: 53, payload: 13032-14479 (1448 bytes)]
    [Frame: 55, payload: 14480-15927 (1448 bytes)]
    [Frame: 56, payload: 15928-17375 (1448 bytes)]
    [Frame: 57, payload: 17376-18823 (1448 bytes)]
```

`dissect_tcp_pdus(tvb, tree, min_header_size, get_len_func, dissect_func)`

Source: https://www.wireshark.org/docs/wsdg_html_chunked/lua_module_Proto.html

Hot Key Report



No.	Time	Destination	Protocol
1	0.000000	192.168.5.106	Aerospike
2	0.000000	192.168.5.106	TCP
3	0.024262	172.18.0.2	Aerospike
4	0.024266	172.18.0.2	TCP
5	0.024858	192.168.5.106	Aerospike
6	0.024858		
7	0.025057		
8	0.025060		
9	0.035394		

```
Hot Key Report

Key                                       Request    Response   δ
356FACFD1C2C83DC7A45B4F5442D04A14C4136BA  3          0          3
E3932AD949B4361E0B88AC982D54F45EAB2219E2  2          0          2
FEDFB33674D24D780D3287405BE167FB3FFBE210  2          0          2
859188499C57A43D1078563FA994A3EC69EB695E  2          0          2
0340838AB39844A6B424021463DD3EE867FF84CF  2          0          2
77ED4A801B851C9AE3F66AFC39AC5AC66E407831  2          0          2
D585C6DA0362D14889CB18E0F1D200D7C3FF3D56  2          0          2
ADCCECBB656EFBC14632E9EE01AC5F25D24FD858  2          0          2
0D9A3E211A39AF9602F90E920C7D335DE4DC61C8  2          0          2
92107B09885849F8A2A0A8DD44684E5EA5BF4B21  2          0          2
1FE3DCED59DBEEC055964BE2ED8253DFE8F3DAD1  2          0          2
3A97F1EC3C9F0A0B1DBAC46767971FBE3B19B357  1          0          1
CDA051E502EE650291BAC08D9DBB95F34615AD34  1          0          1
0858BC59BF39144010281C7DBCD40E6E1C892B4E  1          0          1
357B3579A79438476B78F1CCA28FB8268EA16E79  1          0          1
25DF6675732ABF6D776CBDB5B01C0710463D34B7  1          0          1
916779C276C716CEC0965FF4295C283E57092D45  1          0          1
384EDABD21CF07A966862FD919D12CDC6BE65556  1          0          1
DED43D16BD5DEE3EB4B0655E88F513A8A6937CF4  1          0          1
F0A8F5D3913F4037BF4BEE801B7E6F00F1FF2DE5  1          0          1
```

Tests: Execution

```
aeropike-wireshark-plugin/tests $ make clean; make

Testing test-msg-write-response.pcapng.pdml ... [OK]
Testing test-msg-write-request.pcapng.pdml ... [OK]
Testing test-batch.pcapng.pdml ... [OK]
Testing test-payload-response-greater-than-1500.pcapng.pdml ... [OK]
Testing test-heartbeat-mesh.pcapng.pdml ... [OK]
Testing test-payload-request-greater-than-1500.pcapng.pdml ... [OK]
Testing test-msg-read-response.pcapng.pdml ... [OK]
Testing test-msg-read-request.pcapng.pdml ... [OK]
Testing test-heartbeat-multicast.pcapng.pdml ... [OK]
Testing test-info-response.pcapng.pdml ... [OK]
Testing test-info-request.pcapng.pdml ... [OK]

Generate report using luacov-console ...

luacov-console ../lua
luacov-console -s # --no-colored
==============================================================================
Summary
==============================================================================

File                 Hits Missed Coverage
-----------------------------------------
../lua/aerospike.lua 634  48     92.96%
-----------------------------------------
Total                634  48     92.96%
```

Debugging

Function	Description
critical(text)	Critical severity
warn(text)	Warning
message(text)	Normal
info(text)	Informational
debug(text)	Debugging
report_failure(text)	Message box with error icon

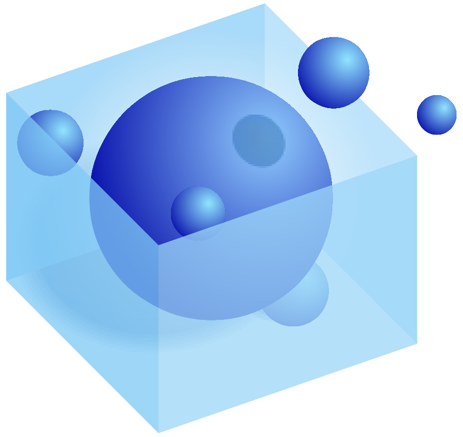
```
local d = require ‘debug’

print (d.traceback())

d.debug()
```

Utility Functions
https://wiki.wireshark.org/LuaAPI/Utils

Development Tips
https://wiki.wireshark.org/Development/Tips

Luacheck

- Accessing undefined variable
- Line contains only whitespace
- Setting non-standard global variable
- Unused variable
- Unused argument
- Unused loop variable i
- Unused function
- Line is too long
- Trailing whitespace in a comment



```
$ luarocks install luacheck

$ luacheck lua/aerospike.lua
Total: 0 warnings / 0 errors in 1 file
```

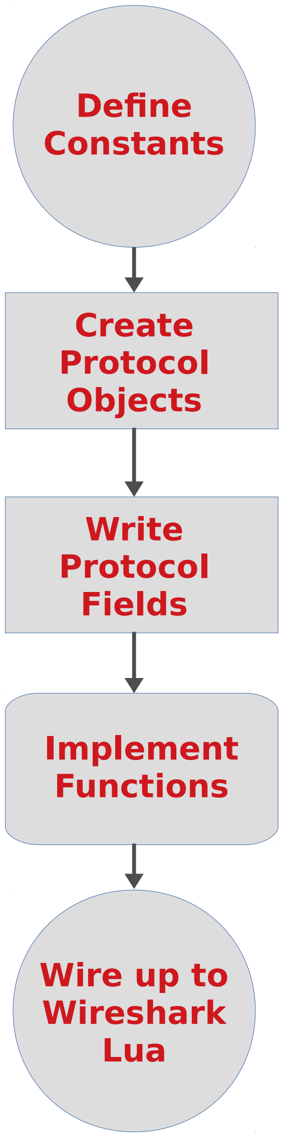
Performance

- Lua performs slower than implementing a plugin in C
- Wireshark becomes slow for capture files greater than 100 MB
- Display filter and save filtered results
- Use TCP/Allow sub-dissectors to reassemble TCP streams
- Use faster CPU and more physical RAM
- Stop other programs on machine to reduce system load
- Split/merge packet captures to analyze critical time intervals

```
$ editcap -r source.pcap target.pcap 0-15000      # 0-15000 packets
$ editcap -i 20 source.pcap 20starget.pcap        # 20s
$ editcap -c 10000 source.pcap 10000target.pcap   # 10000 packets
$ editcap -s 128 source.pcap 128btarget.pcap      # 128 bytes of packet
$ mergecap -w output.pcap client.pcap server.pcap
```

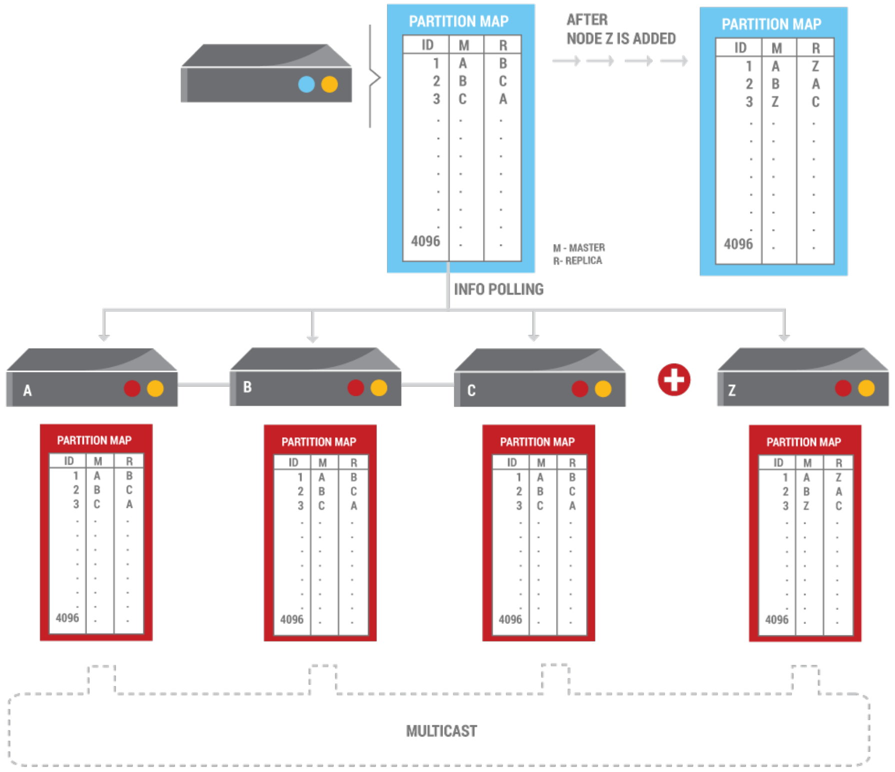
Source: https://www.wireshark.org/docs/man-pages/editcap.html
https://www.wireshark.org/docs/man-pages/mergecap.html

Summary


Define Constants
Create Protocol Objects
Write Protocol Fields
Implement Functions
Wire up to Wireshark Lua

Future Work

- Migration
- Clustering
- Proxy
- RW (Replication)
- Fabric
- Info
- Exchange
- System Metadata
- Security
- Cross Datacentre Replication (XDR)



References

- Lua: https://www.lua.org
- Wireshark Lua API: https://wiki.wireshark.org/LuaAPI
- Aerospike Wireshark Lua Plugin: https://github.com/aerospike/aerospike-wireshark-plugin
- Lua Examples: https://www.wireshark.org/Lua/Examples
- “Changing Wireshark with Lua: Writing a Lua Plug-in to Create a Custom Decoder” (~ 1h 20m) https://www.youtube.com/watch?v=HTtVHxIh6ww
- Lua style guide: http://lua-users.org/wiki/LuaStyleGuide
- Lua Performance: https://wiki.wireshark.org/Performance
- Peter Wu (“Lekensteyn” at #wireshark irc.freenode.net) Wireshark notes: https://git.lekensteyn.nl/peter/wireshark-notes
- Lua scripting in Wireshark: https://sharkfestus.wireshark.org/sharkfest.09/DT06_Bjorlykke_Lua%20Scripting%20in%20Wireshark.pdf

Thank You

@shakthimaan