

Shooting the trouble down to the Wire...Shark Lua Plugin

March 2018

Shakthi Kannan

Version 0.1

shakthi@aerospike.com

Objectives

- Network analysis
 - Client-Server
 - Server-Server
- Production support
- Report generation
- Anomaly detection
- Protocol Development

Why Lua?

- Light-weight
- Dynamic
- Prototyping dissectors
- Ease of deployment
- Scripting
- Portability
- C interface

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.

Usage



Network Protocol Analyzer

Version 2.5.1

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 GTK+ 3.22.28, with Cairo 1.15.10, with Pango 1.40.13,
with libpcap, with POSIX capabilities (Linux), with libnl 3, with GLib 2.54.3,
with zlib 1.2.11, without SMI, with c-ares 1.13.0, with Lua 5.2.4, with GnuTLS
3.5.18, with Gcrypt 1.8.2, with MIT Kerberos, without MaxMind DB resolver, with
nghttp2 1.30.0, with LZ4, without Snappy, with libxml2 2.9.8, with PortAudio
V19.6.0-devel, revision 396fe4b6699ae929d3a685b3ef8a7e97396139a4, without SBC,
without SpanDSP, without bcg729.

```
$ 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.

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
```



Debugging

- `critical(text)`
- `warn(text)`
- `info(text)`
- `debug(text)`
- `print(text)`

```
local d = require 'debug'  
  
print(d.traceback())  
  
d.debug()
```

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

Luacheck

- accessing undefined variable
- line contains only whitespace
- setting non-standard global variable
- line is too long
- trailing whitespace in a comment

```
$ luarocks install luacheck
```

```
$ luacheck lua/aerospike.lua
```

```
Total: 0 warnings / 0 errors in 1 file
```


Markdown Structure

```
# Configuration
# Statistics
# Helper Functions
# Protocols
## Common
## Info
## 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 a 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)

Source: <https://github.com/citrusleaf/aerospike-server/blob/master/as/include/base/proto.h#L239>

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



lit2lua ...

```
local TYPES_OPS = {
    [1] = "AS_MSG_OP_READ",
    [2] = "AS_MSG_OP_WRITE",
    [3] = "AS_MSG_OP_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_HEADER_START = 1
local PROTO_HEADER_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
```

Protocol Dissection Pattern ...

Proto Objects, Header Fields

Create Proto objects

```
local aerospike_attribute      = Proto("AerospikeAttribute", "Aerospike Attributes")
local aerospike_attribute_value = Proto("AerospikeAttributeValue", "Aerospike Attribute Value pairs")
```

Proto header fields

```
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

```
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 subtree = tree:add(aerospike_attribute, data_tvbr)
  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, "\n") then
      local parts = split_tab(d_string)
      ...
      ...
      ...
    end
    data_start = data_start + string.len(line) + 1 -- for \n
  end
end
```

Dissector Table

```
local default_settings =
    aerospike_port          = 3000,
    heartbeat_multicast_port = 9918,
    heartbeat_mesh_port     = 3002,

DissectorTable.get("tcp.port"):add(
    default_settings.heartbeat_mesh_port,      heartbeat_proto)
DissectorTable.get("udp.port"):add(
    default_settings.heartbeat_multicast_port, heartbeat_proto)
```


User Interface

The screenshot displays the Wireshark interface with a network capture loaded. The top menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. The 'Tools' menu is open, showing 'Firewall ACL Rules', 'Compare', and 'Hex Key'. The main display area shows a list of captured packets with the following columns: No., Time, Source, Destination, Protocol, Stream index, Length, and Info.

No.	Time	Source	Destination	Protocol	Stream index	Length	Info
1	0.000000000	192.168.5.14	239.1.99.222	Aerospike		164	9918 → 9918 Len=120
2	0.150309833	192.168.5.14	239.1.99.222	Aerospike		164	9918 → 9918 Len=120
3	0.189847990	127.0.0.1	127.0.0.1	Aerospike	0	119	51846 → 3000 [PSH, ACK] Seq=1 Ack=1 Win=687 Len=
4	0.189979399	127.0.0.1	127.0.0.1	Aerospike	0	139	3000 → 51846 [PSH, ACK] Seq=1 Ack=52 Win=171 Le
5	0.190004576	127.0.0.1	127.0.0.1	TCP	0	68	51846 → 3000 [ACK] Seq=52 Ack=72 Win=687 Len=0
6	0.281141787	127.0.0.1	127.0.0.1	Aerospike	1	119	51840 → 3000 [PSH, ACK] Seq=1 Ack=1 Win=1819 Le
7	0.281262245	127.0.0.1	127.0.0.1	Aerospike	1	139	3000 → 51840 [PSH, ACK] Seq=1 Ack=52 Win=1819 Le

Below the packet list, the details pane shows the expanded view of the first packet:

- +Frame 1: 164 bytes on wire (1312 bits), 164 bytes captured (1312 bits) on interface 0
- +Linux cooked capture
- +Internet Protocol Version 4, Src: 192.168.5.14, Dst: 239.1.99.222
- +User Datagram Protocol, Src Port: 9918, Dst Port: 9918
- +Aerospike Heartbeat



Info protocol

```
-Aerospike Protocol
Version: 2
Type: Info (1)
Size: 43
-Aerospike Attributes
Attribute: node
Attribute: peers-generation
Attribute: partition-generation

0000  00 00 03 04 00 06 00 00 00 00 00 00 00 08 00  .....
0010  45 00 00 67 cb bf 40 00 40 06 70 cf 7f 00 00 01  E..g..@. @.p.....
0020  7f 00 00 01 ca 86 0b b8 8d ba 71 bb 7f c1 70 7f  .....[. ....;..
0030  80 18 02 af fe 5b 00 00 01 01 08 0a 00 3b ab 16  .;..... ..+node
0040  00 3b aa 1c 02 01 00 00 00 00 00 2b 6e 6f 64 65  .peers-g eneratio
0050  0a 70 65 65 72 73 2d 67 65 6e 65 72 61 74 69 6f  n.partit ion-gene
0060  6e 0a 70 61 72 74 69 74 69 6f 6e 2d 67 65 6e 65  ration.
0070  72 61 74 69 6f 6e 0a
```

Message protocol

```
-Aerospike Message Header
  Header Size: 22
+Info1 : 0
-Info2 : 1
  0... .... = AS_MSG_INFO2_RESPOND_ALL_OPS: 0
  .0.. .... = Unused: 0
  ..0. .... = AS_MSG_INFO2_CREATE_ONLY: 0
  ...0 .... = AS_MSG_INFO2_DURABLE_DELETE: 0
  .... 0... = AS_MSG_INFO2_GENERATION_GT: 0
  .... .0.. = AS_MSG_INFO2_GENERATION: 0
  .... ..0. = AS_MSG_INFO2_DELETE: 0
  .... ...1 = AS_MSG_INFO2_WRITE: 1
+Info3 : 0
  Unused: 0
  Result code: 0
  Generation: 0
  Record TTL: 0
  Transaction TTL: 1000
  Number of fields: 3
  Number of operations: 2
+Aerospike Fields
+Aerospike Operations
```

New List Operations

▼ Aerospike Operations

Size: 10

Op: AS_MSG_OP_CDT_MODIFY (4)

Bin data type: AS_PARTICLE_TYPE_BLOB (4)

Bin version: 0

Bin name length: 1

Bin name: l

Data bytes: 000c920006

0000	08 00 27 21 3c 79 0a 00	27 00 00 02 08 00 45 02	..!<y.. '.....E.
0010	00 8b 00 00 40 00 40 06	77 0f c0 a8 21 01 c0 a8@.@. w...!...
0020	21 0a e1 b0 0b b8 9b 07	65 aa 0c da 18 75 80 18	!..... e.....u..
0030	10 13 db 94 00 00 01 01	08 0a 1e 86 ac 8d 00 ff
0040	5a 09 02 03 00 00 00 00	00 4f 16 00 01 00 00 00	Z..... .0.....
0050	00 00 00 00 00 00 00 00	00 00 03 e8 00 03 00 01
0060	00 00 00 05 00 74 65 73	74 00 00 00 05 01 74 65tes t.....te
0070	73 74 00 00 00 15 04 11	e4 58 59 5e e4 01 0a 6a	st..... .XY^...j
0080	c7 a3 38 41 27 22 cc 8a	8e 76 50 00 00 00 0a 04	..8A'"... vP.....
0090	04 00 01 6c 00 0c 92 00	06	...l.... .

Heartbeat protocol

```
+User Datagram Protocol, Src Port: 9918, Dst Port: 9918
-Aerospike Heartbeat
  Size: 114
  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: 0bb9b11300000000
  ID: AS_HB_MSG_HLC_TIMESTAMP (4)
  Message Type: 3
  Data bytes: 01600c6828f50000
```



Demo



Source: <https://xkcd.com/1323/>



Future

- Migration
- Clustering
- Proxy
- RW (Replication)
- Fabric
- System Meta Data
- Cross Data Centre Replication (XDR)
- Security

References

- Lua: <https://www.lua.org/>
- Wireshark Lua API: <https://wiki.wireshark.org/LuaAPI>
- Dissectors: <https://wiki.wireshark.org/Lua/Dissectors>
- Lua Examples: <https://wiki.wireshark.org/Lua/Examples>
- Lua scripting in Wireshark:
https://sharkfest.wireshark.org/sharkfest.09/DT06_Bjorlykke_Lua%20Scripting%20in%20Wireshark.pdf
- Lua style guide: <http://lua-users.org/wiki/LuaStyleGuide>
- “Changing Wireshark with Lua: Writing a Lua Plug-in to Create a Custom Decoder” (~ 1h 20m)
<https://www.youtube.com/watch?v=HTtVHxIh6ww>