RSA Charge 2016

Getting More Out of RSA NetWitness Logs and Packets with Lua Parsing

Chris Ahearn, RSA

#RSACharge

RSA Charge 2016

Because One Size Fits Many

…but Not All

Hello
my name is
Chris

What Do Parsers Do?

- Parsers originate meta
- They ask questions of the data?
  - Meta is the answer to those questions
- Examines the raw data as it comes into the decoder
- Can also examine meta created in the session
  - Can be used for data manipulation
- Help analysts get data they need to answer questions

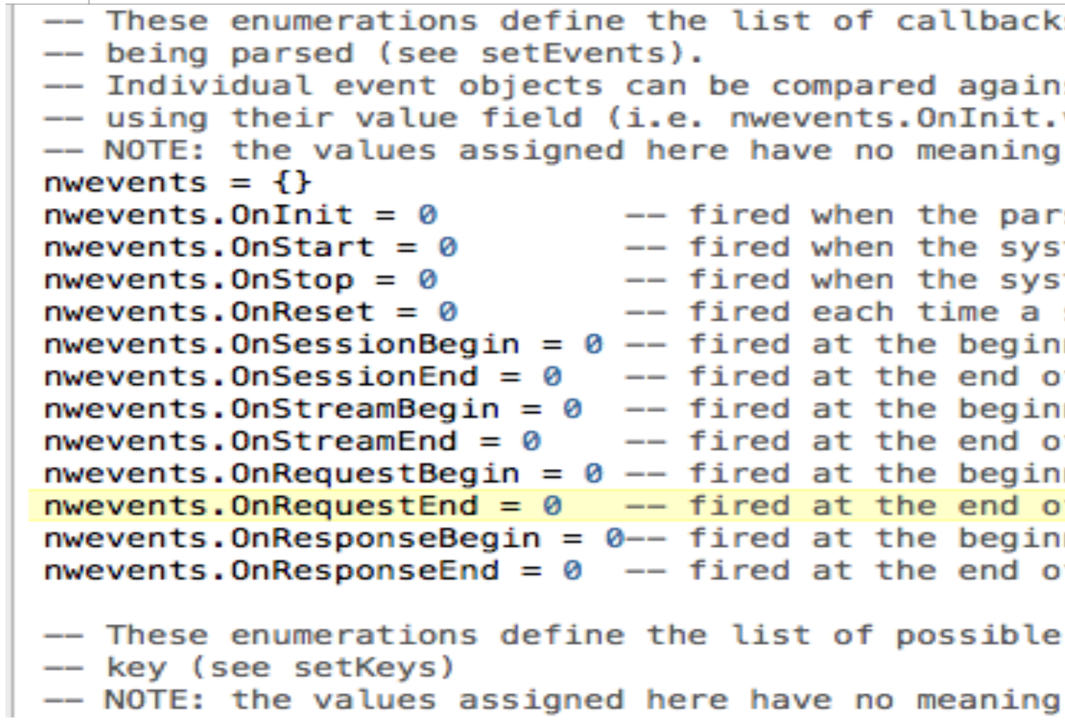
The Requirements

RSA

PARSERS

A Treatise on Writing Packet Parsers for Security Analytics

1st Edition, Revised 12/9/2014



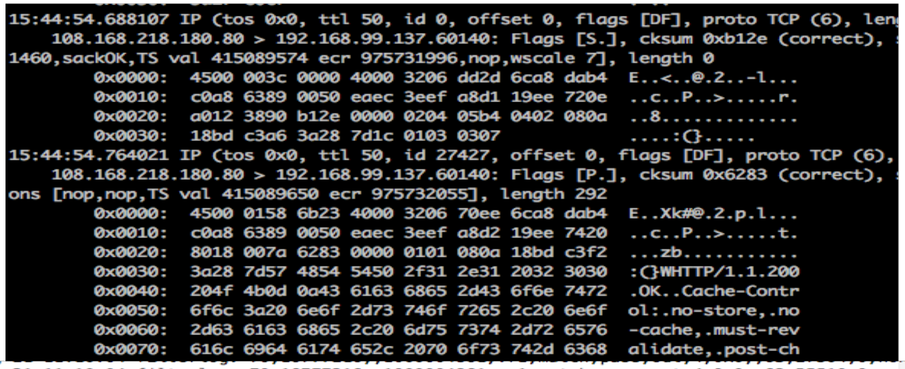
```
-- These enumerations define the list of callback
-- being parsed (see setEvents).
-- Individual event objects can be compared again
-- using their value field (i.e. nwevents.OnInit.
-- NOTE: the values assigned here have no meaning
nwevents = {}
nwevents.OnInit = 0          -- fired when the par
nwevents.OnStart = 0         -- fired when the sys
nwevents.OnStop = 0          -- fired when the sys
nwevents.OnReset = 0         -- fired each time a
nwevents.OnSessionBegin = 0 -- fired at the begin
nwevents.OnSessionEnd = 0    -- fired at the end o
nwevents.OnStreamBegin = 0  -- fired at the begin
nwevents.OnStreamEnd = 0     -- fired at the end o
nwevents.OnRequestBegin = 0 -- fired at the begin
nwevents.OnRequestEnd = 0    -- fired at the end o
nwevents.OnResponseBegin = 0-- fired at the begin
nwevents.OnResponseEnd = 0  -- fired at the end o

-- These enumerations define the list of possible
-- key (see setKeys)
-- NOTE: the values assigned here have no meaning
```

- Parsers Book
- nw-api.lua

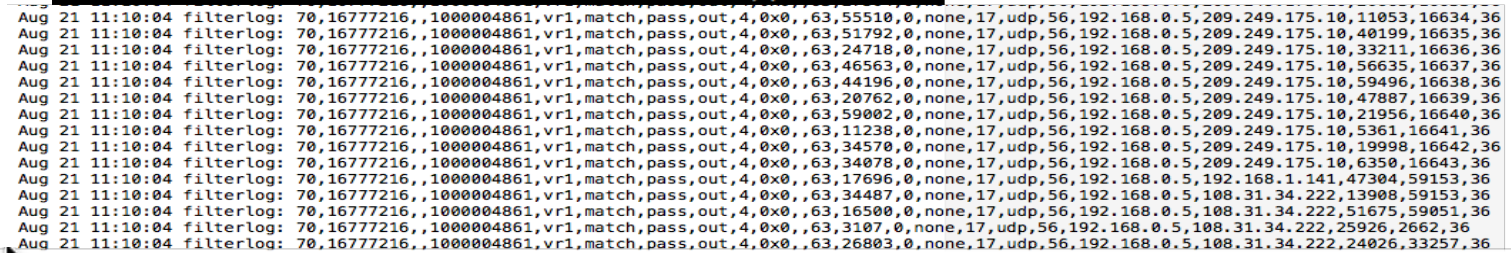
https://community.emc.com/docs/DOC-41108

RSA Charge 2016

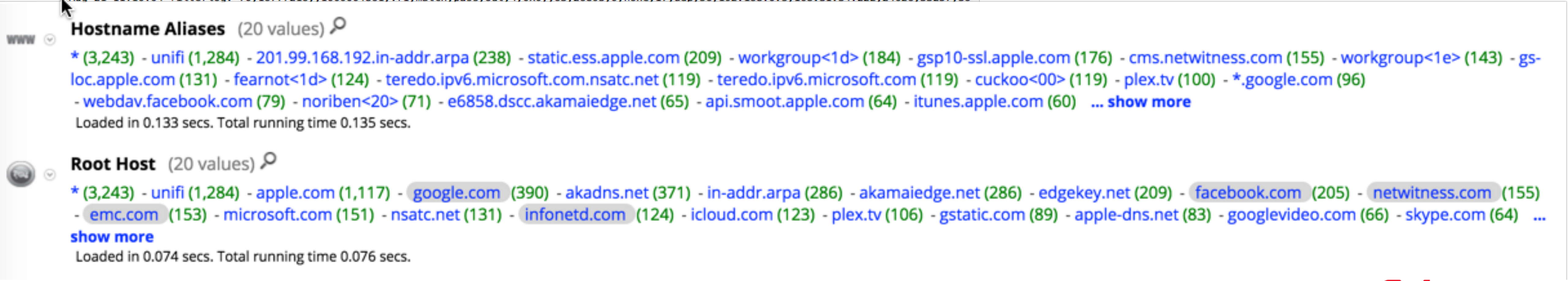
The Requirements



```
15:44:54.688107 IP (tos 0x0, ttl 50, id 0, offset 0, flags [DF], proto TCP (6), len
    108.168.218.180.80 > 192.168.99.137.60140: Flags [S.], cksum 0xb12e (correct),
1460,sackOK,TS val 415089574 ecr 975731996,nop,wscale 7], length 0
        0x0000:  4500 003c 0000 4000 3206 dd2d 6ca8 dab4  E..<..@.2..-l...
        0x0010:  c0a8 6389 0050 eaec 3eef a8d1 19ee 720e  ..c..P..>.....r.
        0x0020:  a012 3890 b12e 0000 0204 05b4 0402 080a  ..8.............
        0x0030:  18bd c3a6 3a28 7d1c 0103 0307            ....:(}.....
15:44:54.764021 IP (tos 0x0, ttl 50, id 27427, offset 0, flags [DF], proto TCP (6),
    108.168.218.180.80 > 192.168.99.137.60140: Flags [P.], cksum 0x6283 (correct),
ons [nop,nop,TS val 415089650 ecr 975732055], length 292
        0x0000:  4500 0158 6b23 4000 3206 70ee 6ca8 dab4  E..Xk#@.2.p.l...
        0x0010:  c0a8 6389 0050 eaec 3eef a8d2 19ee 7420  ..c..P..>.....t.
        0x0020:  8018 007a 6283 0000 0101 080a 18bd c3f2  ...zb...........
        0x0030:  3a28 7d57 4854 5450 2f31 2e31 2032 3030  :(}WHTTP/1.1.200
        0x0040:  204f 4b0d 0a43 6163 6865 2d43 6f6e 7472  .OK..Cache-Contr
        0x0050:  6f6c 3a20 6e6f 2d73 746f 7265 2c20 6e6f  ol:.no-store,.no
        0x0060:  2d63 6163 6865 2c20 6d75 7374 2d72 6576  -cache,.must-rev
        0x0070:  616c 6964 6174 652c 2070 6f73 742d 6368  alidate,.post-ch
```



```
Aug 21 11:10:04 filterlog: 70,16777216,,1000004861,vr1,match,pass,out,4,0x0,,63,55510,0,none,17,udp,56,192.168.0.5,209.249.175.10,11053,16634,36
Aug 21 11:10:04 filterlog: 70,16777216,,1000004861,vr1,match,pass,out,4,0x0,,63,51792,0,none,17,udp,56,192.168.0.5,209.249.175.10,40199,16635,36
Aug 21 11:10:04 filterlog: 70,16777216,,1000004861,vr1,match,pass,out,4,0x0,,63,24718,0,none,17,udp,56,192.168.0.5,209.249.175.10,33211,16636,36
Aug 21 11:10:04 filterlog: 70,16777216,,1000004861,vr1,match,pass,out,4,0x0,,63,46563,0,none,17,udp,56,192.168.0.5,209.249.175.10,56635,16637,36
Aug 21 11:10:04 filterlog: 70,16777216,,1000004861,vr1,match,pass,out,4,0x0,,63,44196,0,none,17,udp,56,192.168.0.5,209.249.175.10,59496,16638,36
Aug 21 11:10:04 filterlog: 70,16777216,,1000004861,vr1,match,pass,out,4,0x0,,63,20762,0,none,17,udp,56,192.168.0.5,209.249.175.10,47887,16639,36
Aug 21 11:10:04 filterlog: 70,16777216,,1000004861,vr1,match,pass,out,4,0x0,,63,59002,0,none,17,udp,56,192.168.0.5,209.249.175.10,21956,16640,36
Aug 21 11:10:04 filterlog: 70,16777216,,1000004861,vr1,match,pass,out,4,0x0,,63,11238,0,none,17,udp,56,192.168.0.5,209.249.175.10,5361,16641,36
Aug 21 11:10:04 filterlog: 70,16777216,,1000004861,vr1,match,pass,out,4,0x0,,63,34570,0,none,17,udp,56,192.168.0.5,209.249.175.10,19998,16642,36
Aug 21 11:10:04 filterlog: 70,16777216,,1000004861,vr1,match,pass,out,4,0x0,,63,34078,0,none,17,udp,56,192.168.0.5,209.249.175.10,6350,16643,36
Aug 21 11:10:04 filterlog: 70,16777216,,1000004861,vr1,match,pass,out,4,0x0,,63,17696,0,none,17,udp,56,192.168.0.5,192.168.1.141,47304,59153,36
Aug 21 11:10:04 filterlog: 70,16777216,,1000004861,vr1,match,pass,out,4,0x0,,63,34487,0,none,17,udp,56,192.168.0.5,108.31.34.222,13908,59153,36
Aug 21 11:10:04 filterlog: 70,16777216,,1000004861,vr1,match,pass,out,4,0x0,,63,16500,0,none,17,udp,56,192.168.0.5,108.31.34.222,51675,59051,36
Aug 21 11:10:04 filterlog: 70,16777216,,1000004861,vr1,match,pass,out,4,0x0,,63,3107,0,none,17,udp,56,192.168.0.5,108.31.34.222,25926,2662,36
Aug 21 11:10:04 filterlog: 70,16777216,,1000004861,vr1,match,pass,out,4,0x0,,63,26803,0,none,17,udp,56,192.168.0.5,108.31.34.222,24026,33257,36
```



Hostname Aliases (20 values)

* (3,243) - unifi (1,284) - 201.99.168.192.in-addr.arpa (238) - static.ess.apple.com (209) - workgroup<1d> (184) - gsp10-ssl.apple.com (176) - cms.netwitness.com (155) - workgroup<1e> (143) - gs-loc.apple.com (131) - fearnot<1d> (124) - teredo.ipv6.microsoft.com.nsatc.net (119) - teredo.ipv6.microsoft.com (119) - cuckoo<00> (119) - plex.tv (100) - *.google.com (96) - webdav.facebook.com (79) - noriben<20> (71) - e6858.dscc.akamaiedge.net (65) - api.smoot.apple.com (64) - itunes.apple.com (60) **... show more**

Loaded in 0.133 secs. Total running time 0.135 secs.

Root Host (20 values)

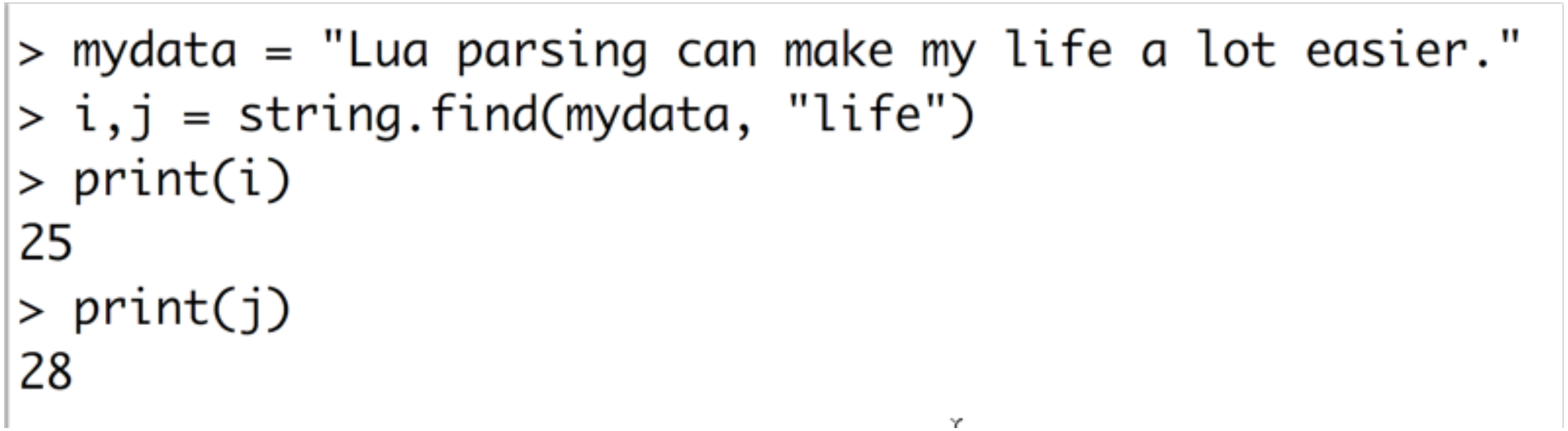
* (3,243) - unifi (1,284) - apple.com (1,117) - google.com (390) - akadns.net (371) - in-addr.arpa (286) - akamaiedge.net (286) - edgekey.net (209) - facebook.com (205) - netwitness.com (155) - emc.com (153) - microsoft.com (151) - nsatc.net (131) - infonetd.com (124) - icloud.com (123) - plex.tv (106) - gstatic.com (89) - apple-dns.net (83) - googlevideo.com (66) - skype.com (64) **... show more**

Loaded in 0.074 secs. Total running time 0.076 secs.

- Pcaps
- Logs
- Meta
- Lua Interpreter
- Strings
- Documentation

Lua Tips

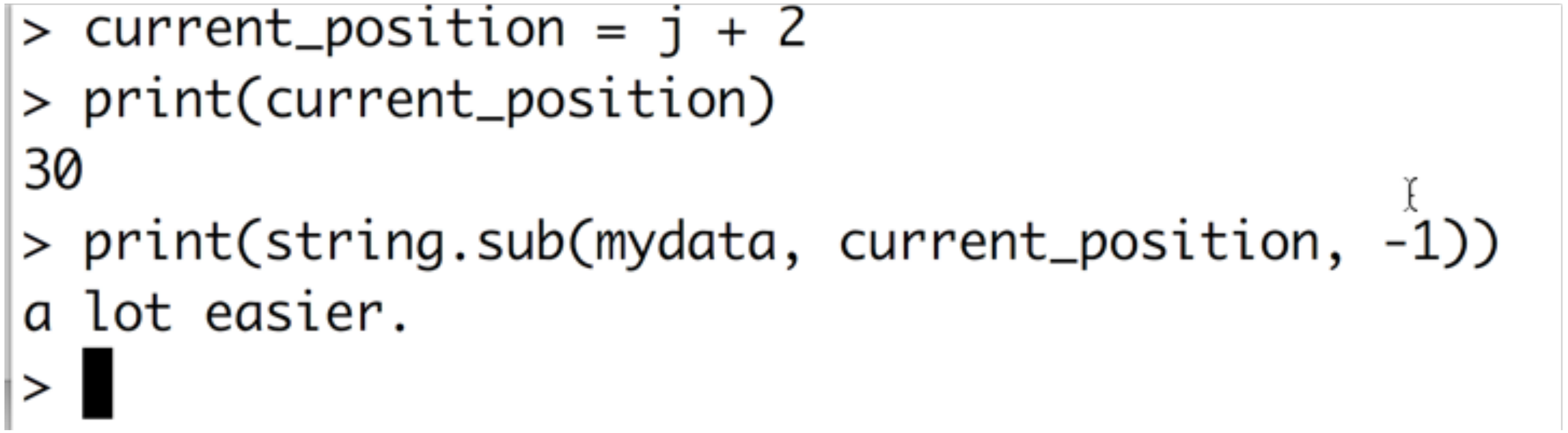
- You are always on a byte
- Counting starts at 1 not 0 (I know, I know)



```
> mydata = "Lua parsing can make my life a lot easier."
> i,j = string.find(mydata, "life")
> print(i)
25
> print(j)
28
```

Lua Tips

- The position number is relative to where you started from
- Always keep track of your position



```
> current_position = j + 2
> print(current_position)
30
> print(string.sub(mydata, current_position, -1))
a lot easier.
> 
```

Meta Keys

Where are you going to put your meta once you get it?

Meta Keys

txbytes

root.host

username

Where are you going to put your meta once you get it?

result.code

email.dst

email

alias.host

root.user

email.src

rxbytes

policy.name

referer.host

Meta Keys

How is the meta key formatted?

Meta Keys

How is the meta key formatted?

Some that I have used:

nwtypes.UInt8	-- An unsigned 8 bit number
nwtypes.UInt16	-- An unsigned 16 bit number
nwtypes.UInt32	-- An unsigned 32 bit number
nwtypes.Text	-- Free form text (256 character max)
nwtypes.IPv4	-- A IPv4 address
nwtypes.MAC	-- A MAC address

See the nw-api file for a complete listing.

Tokens

- Parsers run in memory but tokens trigger the parser to run
  - Be specific

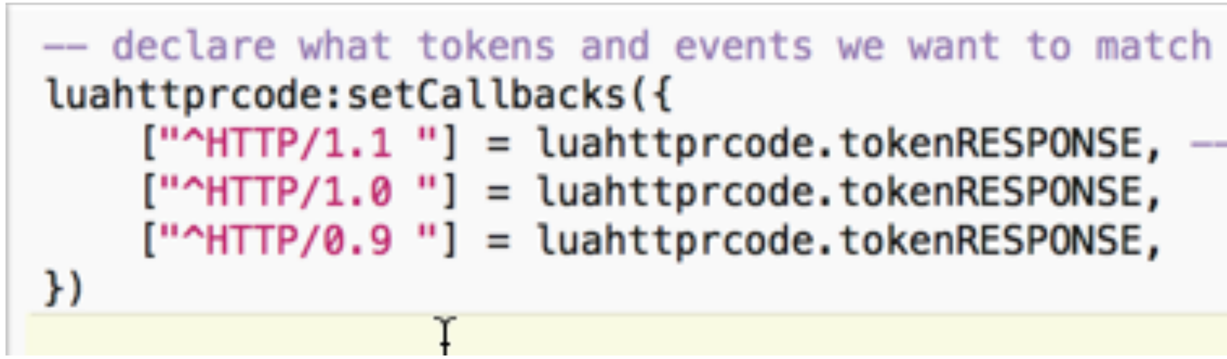
- Answer some questions first.
  - What is the question you are trying to answer?
  - What data do I need to answer the question
  - Where is the data located?
  - What tokens will I need to get me to the data consistently?

Tokens

- Tokens can be:
  1. Text strings
     - ^ for beginning of line
     - $ for end of line
  2. Decimal representation of HEX bytes
  3. Meta callbacks
     - Grab meta that was already generated in that session
  4. Events about that session

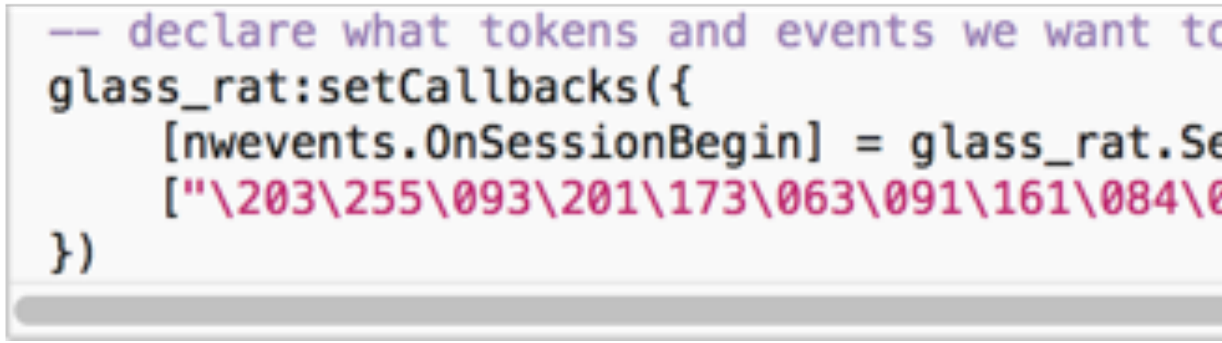
- Once your token matches, the parser functions are run

1



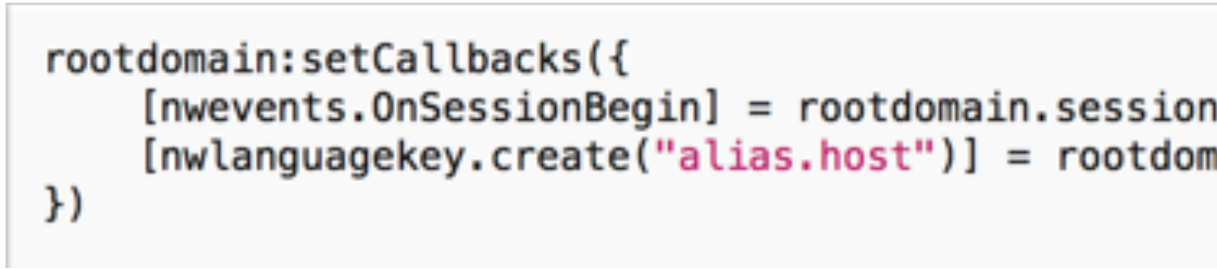
```
-- declare what tokens and events we want to match
luahttprcode:setCallbacks({
    ["^HTTP/1.1 "] = luahttprcode.tokenRESPONSE, --
    ["^HTTP/1.0 "] = luahttprcode.tokenRESPONSE,
    ["^HTTP/0.9 "] = luahttprcode.tokenRESPONSE,
})
```

2



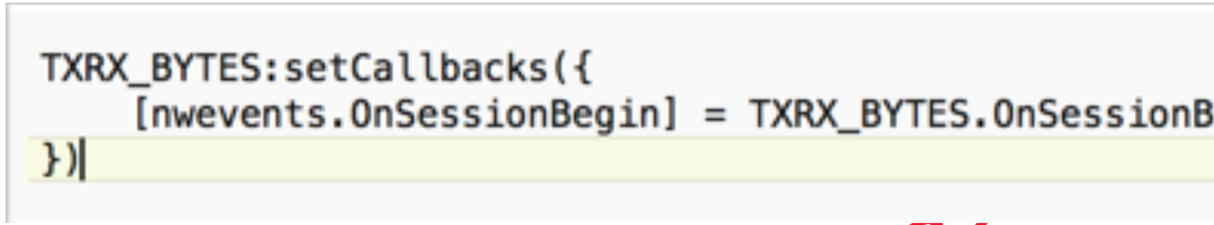
```
-- declare what tokens and events we want to
glass_rat:setCallbacks({
    [nwevents.OnSessionBegin] = glass_rat.Se
    ["\203\255\093\201\173\063\091\161\084\
})
```

3



```
rootdomain:setCallbacks({
    [nwevents.OnSessionBegin] = rootdomain.session
    [nwlanguagekey.create("alias.host")] = rootdom
})
```

4



```
TXRX_BYTES:setCallbacks({
    [nwevents.OnSessionBegin] = TXRX_BYTES.OnSessionB
})
```

RSA Charge 2016

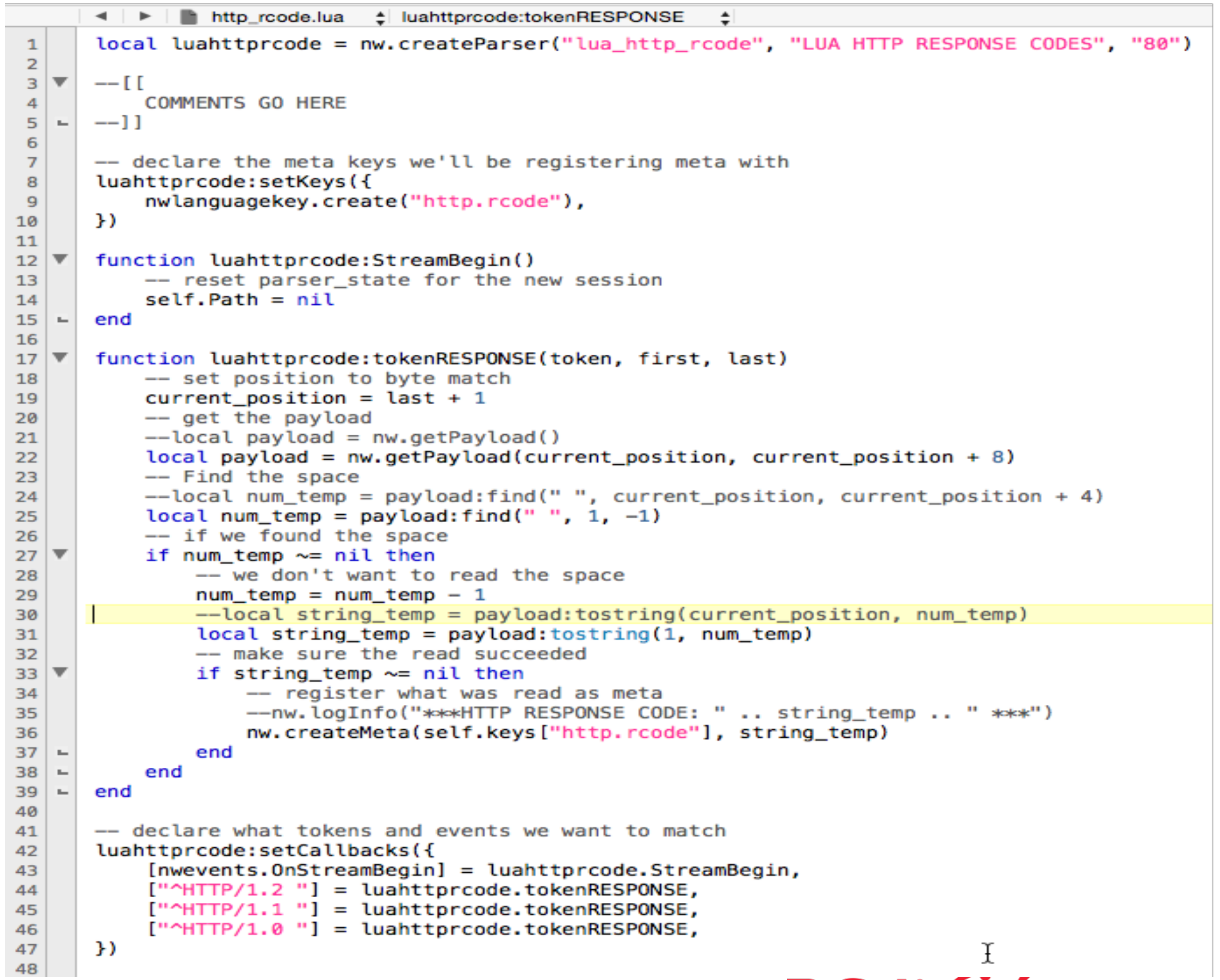
Real World Examples

Use Case – HTTP Response Codes

Is there a way to detect all the HTTP response codes?

Use Case – HTTP Response Codes

- Meta Key – result.code
  - No format defaults to TEXT formatted Meta key
- Multiple Token Matches
  - Kicks off same function
- Specifically calls payload within a certain range
- Looks for a space within that payload from beginning (1) to end (-1)
- Backs up 1 byte if it has data
- Converts that Payload to a string
- Writes that string as meta



http_rcode.lua   luahttprcode:tokenRESPONSE

```lua
local luahttprcode = nw.createParser("lua_http_rcode", "LUA HTTP RESPONSE CODES", "80")

--[[
    COMMENTS GO HERE
--]]

-- declare the meta keys we'll be registering meta with
luahttprcode:setKeys({
    nwlanguagekey.create("http.rcode"),
})

function luahttprcode:StreamBegin()
    -- reset parser_state for the new session
    self.Path = nil
end

function luahttprcode:tokenRESPONSE(token, first, last)
    -- set position to byte match
    current_position = last + 1
    -- get the payload
    --local payload = nw.getPayload()
    local payload = nw.getPayload(current_position, current_position + 8)
    -- Find the space
    --local num_temp = payload:find(" ", current_position, current_position + 4)
    local num_temp = payload:find(" ", 1, -1)
    -- if we found the space
    if num_temp ~= nil then
        -- we don't want to read the space
        num_temp = num_temp - 1
        --local string_temp = payload:tostring(current_position, num_temp)
        local string_temp = payload:tostring(1, num_temp)
        -- make sure the read succeeded
        if string_temp ~= nil then
            -- register what was read as meta
            --nw.logInfo("***HTTP RESPONSE CODE: " .. string_temp .. " ***")
            nw.createMeta(self.keys["http.rcode"], string_temp)
        end
    end
end

-- declare what tokens and events we want to match
luahttprcode:setCallbacks({
    [nwevents.OnStreamBegin] = luahttprcode.StreamBegin,
    ["^HTTP/1.2 "] = luahttprcode.tokenRESPONSE,
    ["^HTTP/1.1 "] = luahttprcode.tokenRESPONSE,
    ["^HTTP/1.0 "] = luahttprcode.tokenRESPONSE,
})
```

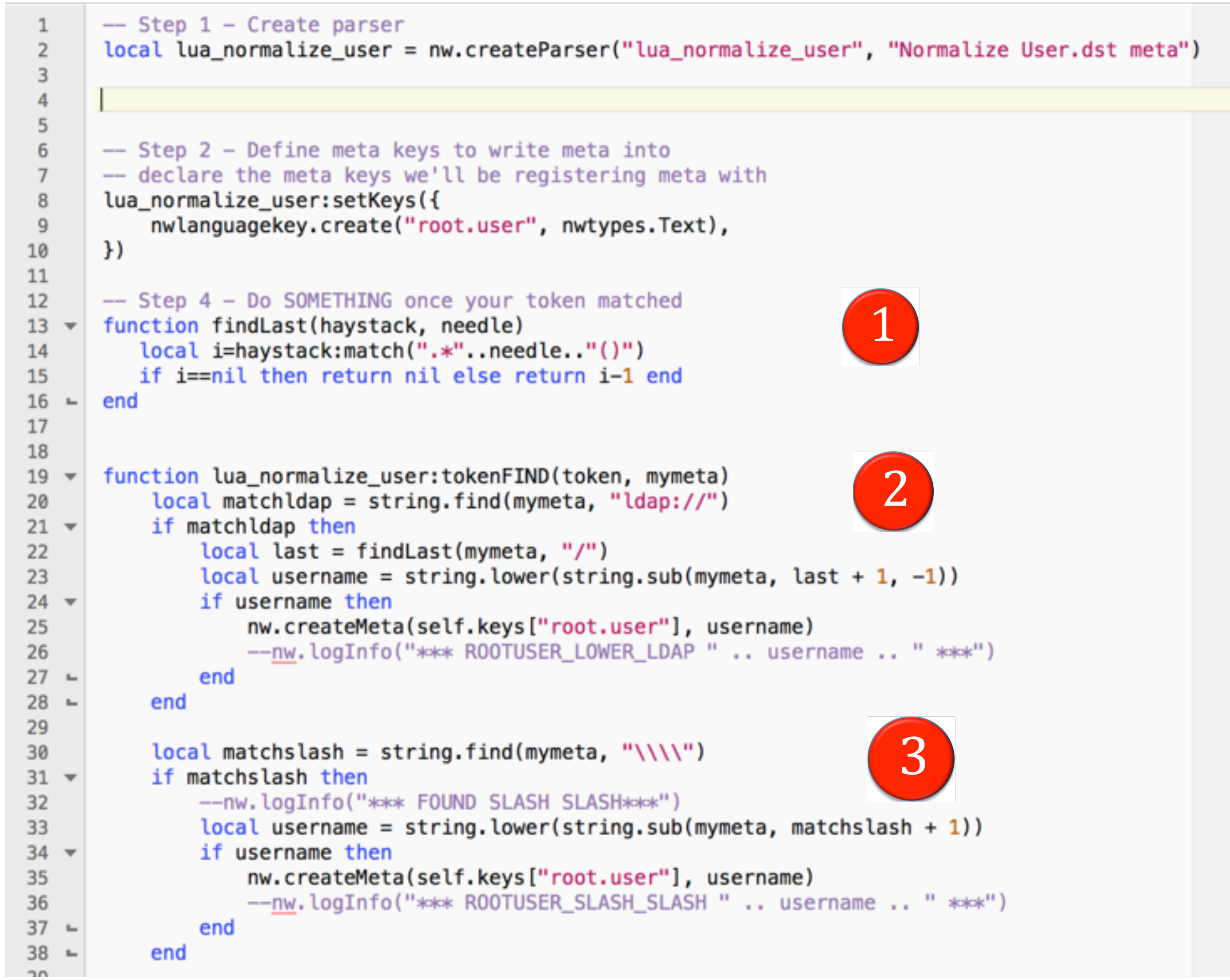
Use Case – Normalize User Accounts

Is there a way to normalize the meta I am seeing in user.dst?

Use Case – Normalize User Accounts

- Custom Meta Key – root.user
  - No format defaults to TEXT formatted Meta key
- Meta callback of ‘user.dst’ meta key
- Performs multiple string finds based on defined criteria

1. Function that finds the last occurrence of a string.
2. If found, may look for the last occurrence of a particular delimiter
   - Then moves forward 1 byte and reads to the end. Then converts to lower case
3. Finds the \\ and then moves 1 byte forward and reads to the end. Then converts to lower case.

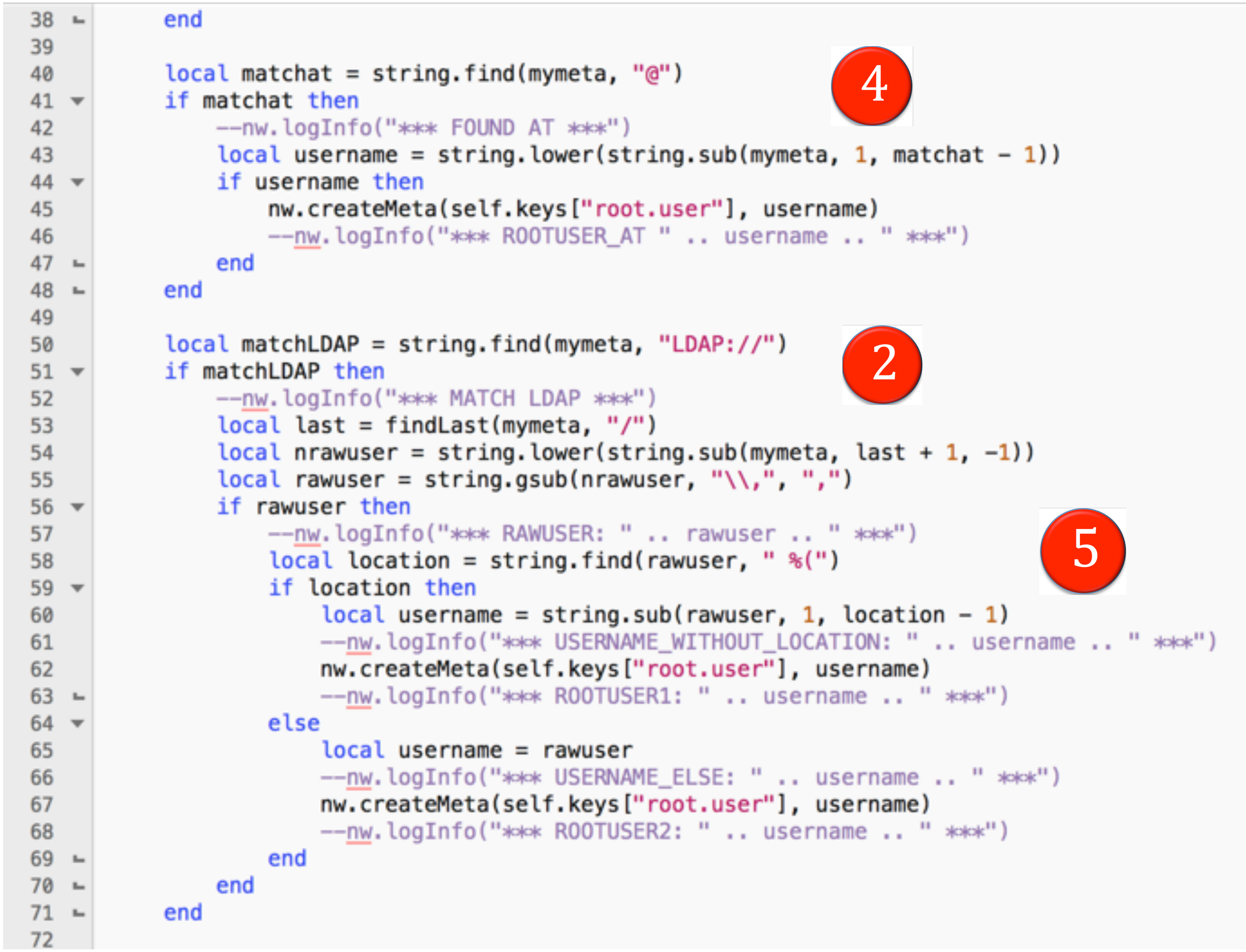


```lua
-- Step 1 - Create parser
local lua_normalize_user = nw.createParser("lua_normalize_user", "Normalize User.dst meta")



-- Step 2 - Define meta keys to write meta into
-- declare the meta keys we'll be registering meta with
lua_normalize_user:setKeys({
  nwlanguagekey.create("root.user", nwtypes.Text),
})

-- Step 4 - Do SOMETHING once your token matched
function findLast(haystack, needle)
  local i=haystack:match(".*"..needle.."()")
  if i==nil then return nil else return i-1 end
end


function lua_normalize_user:tokenFIND(token, mymeta)
  local matchldap = string.find(mymeta, "ldap://")
  if matchldap then
    local last = findLast(mymeta, "/")
    local username = string.lower(string.sub(mymeta, last + 1, -1))
    if username then
      nw.createMeta(self.keys["root.user"], username)
      --nw.logInfo("*** ROOTUSER_LOWER_LDAP " .. username .. " ***")
    end
  end

  local matchslash = string.find(mymeta, "\\\\")
  if matchslash then
    --nw.logInfo("*** FOUND SLASH SLASH***")
    local username = string.lower(string.sub(mymeta, matchslash + 1))
    if username then
      nw.createMeta(self.keys["root.user"], username)
      --nw.logInfo("*** ROOTUSER_SLASH_SLASH " .. username .. " ***")
    end
  end
```

Use Case – Normalize User Accounts

4. Finds the @ and then reads up to, but not including the @. Then converts to lower case.

5. Also found last occurrence of "/" but then continues to replace a "\," with just a comma "," (line 55), look for the open parenthesis " %(" (% is an escape) and then read up to that minus one space.



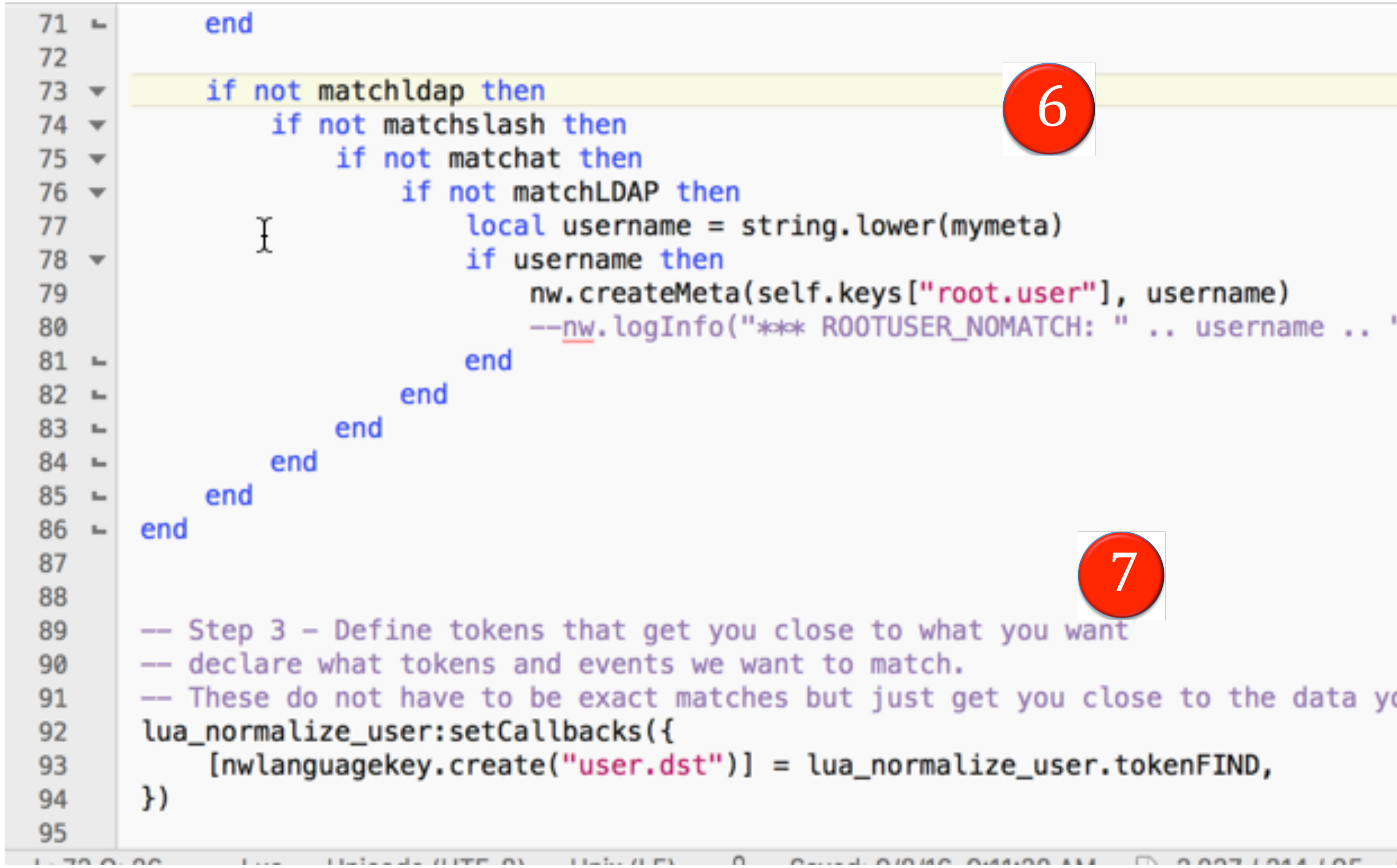
```
38      end
39
40      local matchat = string.find(mymeta, "@")
41      if matchat then
42          --nw.logInfo("*** FOUND AT ***")
43          local username = string.lower(string.sub(mymeta, 1, matchat - 1))
44          if username then
45              nw.createMeta(self.keys["root.user"], username)
46              --nw.logInfo("*** ROOTUSER_AT " .. username .. " ***")
47          end
48      end
49
50      local matchLDAP = string.find(mymeta, "LDAP://")
51      if matchLDAP then
52          --nw.logInfo("*** MATCH LDAP ***")
53          local last = findLast(mymeta, "/")
54          local nrawuser = string.lower(string.sub(mymeta, last + 1, -1))
55          local rawuser = string.gsub(nrawuser, "\\,", ",")
56          if rawuser then
57              --nw.logInfo("*** RAWUSER: " .. rawuser .. " ***")
58              local location = string.find(rawuser, " %(")
59              if location then
60                  local username = string.sub(rawuser, 1, location - 1)
61                  --nw.logInfo("*** USERNAME_WITHOUT_LOCATION: " .. username .. " ***")
62                  nw.createMeta(self.keys["root.user"], username)
63                  --nw.logInfo("*** ROOTUSER1: " .. username .. " ***")
64              else
65                  local username = rawuser
66                  --nw.logInfo("*** USERNAME_ELSE: " .. username .. " ***")
67                  nw.createMeta(self.keys["root.user"], username)
68                  --nw.logInfo("*** ROOTUSER2: " .. username .. " ***")
69              end
70          end
71      end
72
```

RSA Charge 2016

Use Case – Normalize User Accounts

6. Catchall

6. Our meta callback key



```
    end

    if not matchldap then
        if not matchslash then
            if not matchat then
                if not matchLDAP then
                    local username = string.lower(mymeta)
                    if username then
                        nw.createMeta(self.keys["root.user"], username)
                        --nw.logInfo("*** ROOTUSER_NOMATCH: " .. username .. '
                    end
                end
            end
        end
    end
end


-- Step 3 - Define tokens that get you close to what you want
-- declare what tokens and events we want to match.
-- These do not have to be exact matches but just get you close to the data yo
lua_normalize_user:setCallbacks({
    [nwlanguagekey.create("user.dst")] = lua_normalize_user.tokenFIND,
})
```

Please Complete Session Evaluation

#RSACharge

#RSACharge