HACKERS DO THE HAKA

AN OPEN SOURCE SECURITY ORIENTED LANGUAGE

Fond national pour la Société Numérique

CONTEXT

- Sophisticated network security attacks
  - Have a fine-grained analysis of security attacks
  - React quickly to network security issues

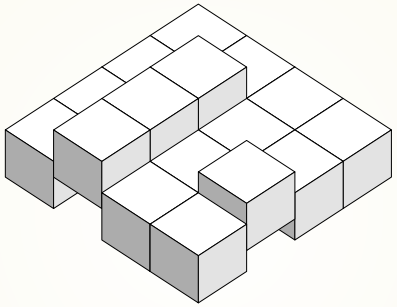
- New emerging application protocols: pseudo-level 8
  - Hard coding network protocol parsers is time-consuming, tedious, error-prone.

WHAT IS HAKA

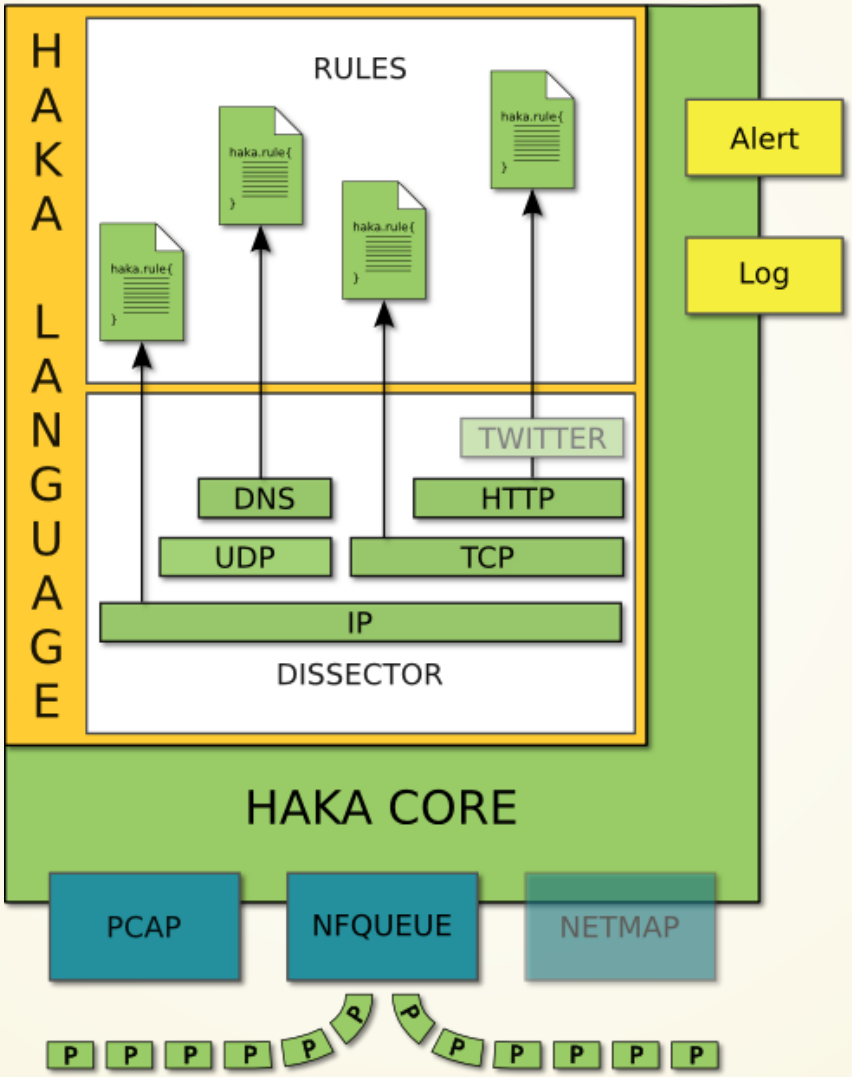
1. A **open source** oriented security language → Abstract low-level stuff to non developer experts
   - Specifiy network protocol: message syntax, state machine
   - Define security rules
   - Define attack countermeasures
2. A modular framework to apply security rules on **live**-captured trafic

"Haka **is not** an other alternative IPS"

ARCHITECTURE



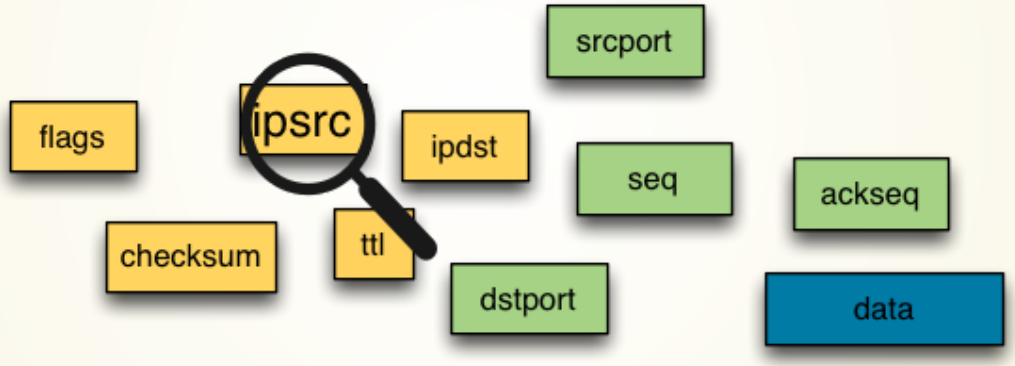
THE BIG PICTURE



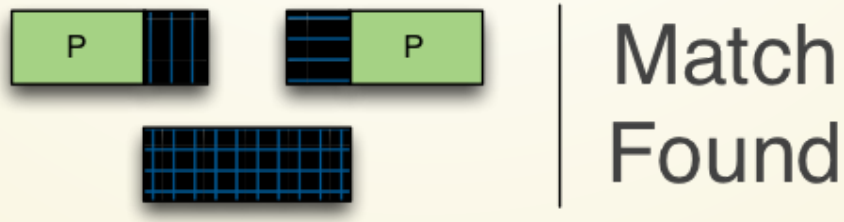
SECURITY RULES

CHECKING SECURITY PROPERTIES

- Check packet fields and data content



- Match malicious pattern across multiple packets



SECURITY RULES

LOGGING AND ALERTING

- Log suspicious network activities in syslog format

- Generate alerts following in IDMEF-like format

SECURITY RULES

ACTIVE REACTION

- Create and inject new packets

- Drop/Reset packets/connections

- Alter on the fly packet content

BASIC IP FILTERING

```
local ipv4 = require('protocol/ipv4')

haka.rule{
    hook = ipv4.events.receive_packet,
    eval = function (pkt)
        local bad_net = ipv4.network('192.168.10.0/24')
        if bad_net:contains(pkt.src) then
            -- Raise an alert
            haka.alert{
                description = string.format("Filtering IP %s", pkt.src),
                severity = 'low'
            }
            -- and drop the packet
            pkt:drop()
        end
    end
}
```

PACKET ALTERING

```
local http = require('protocol/http')

haka.rule{
    hook = http.events.request,
    eval = function (http, request)
        haka.log("http", "%s %s", request.method, request.uri)
        request.headers["User-Agent"] = "Modified by Haka"
    end
}
```

STREAM FILTERING

```
local rem = require('regexp/pcre')
local re = rem.re:compile('<script.*</script>')

haka.rule{
    hook = http.events.response_data,
    options = { streamed = true },
    eval = function (http, iter)
        local match
        repeat
            match = re:match(iter)
            if match then
                match:erase()
            end
        until not match
    end
}
```

DEMO

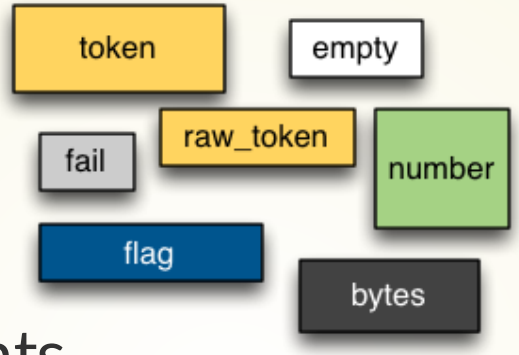
DISSECTION

GRAMMAR

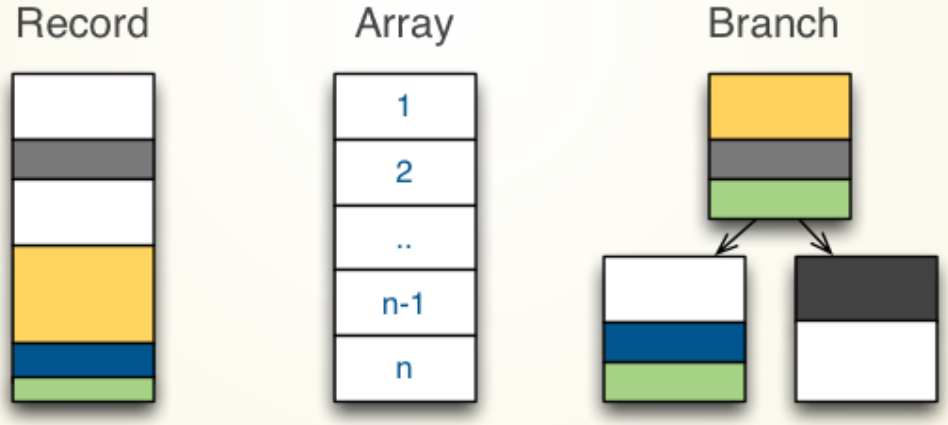
- Specify text-based and binary-based protocols
- Check protocol compliance
- Provide a read/write access to all protocol fields

GRAMMAR - BUILDING BLOCKS

- Final elements



- Compounds elements



- Mutliple options: validate, convert, apply, untilcond, count.

GRAMMAR - PARSING HTTP REQUEST

```
request_line = method WS uri WS version CRLF                    (rfc 2616)
```

```
GET /foo/bar/index.html HTTP/1.1                                  example
```

```
WS = token('[[:blank:]]+')
CRLF = token('[%r]?%n')

version = record{
    token('HTTP/'),
    field('version', token('[0-9]+%.[0-9]+'))
}

request_line = record{
    field('method', token('[[:alnum:]]+')),
    WS,
    field('uri', token('[[:alnum:][:punct:]]+')),
    WS,
    version,
    CRLF
}
```

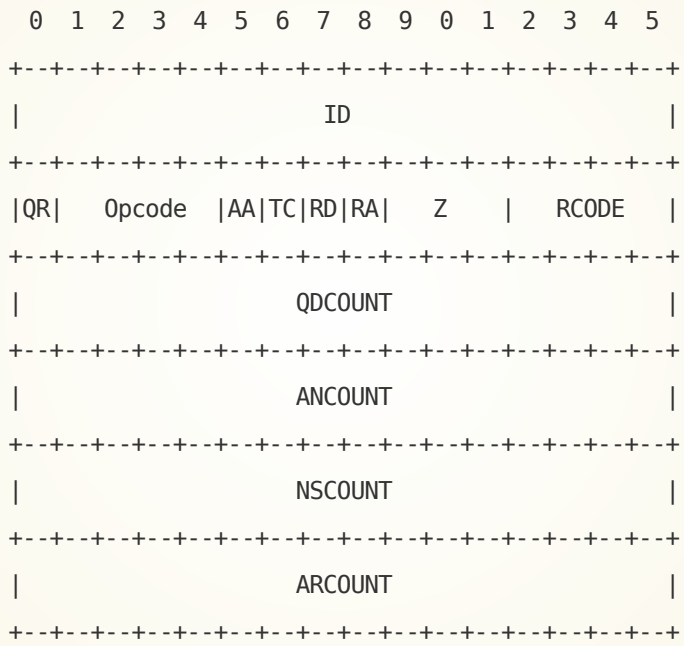
GRAMMAR - PARSING HTTP HEADERS

```
header = record{
    field('name', token('[^:[:blank:]]+')),
    token(':'),
    WS,
    field('value', token('[^%r%n]+')),
    CRLF
}

headers = record{
    field('headers', array(header)
        :untilcond(function (elem, ctx)
            local la = ctx:lookahead()
            return la == 0xa or la == 0xd
        end),
    CRLF
}
```

GRAMMAR - DNS SPEC (1)

PARSING DNS HEADER



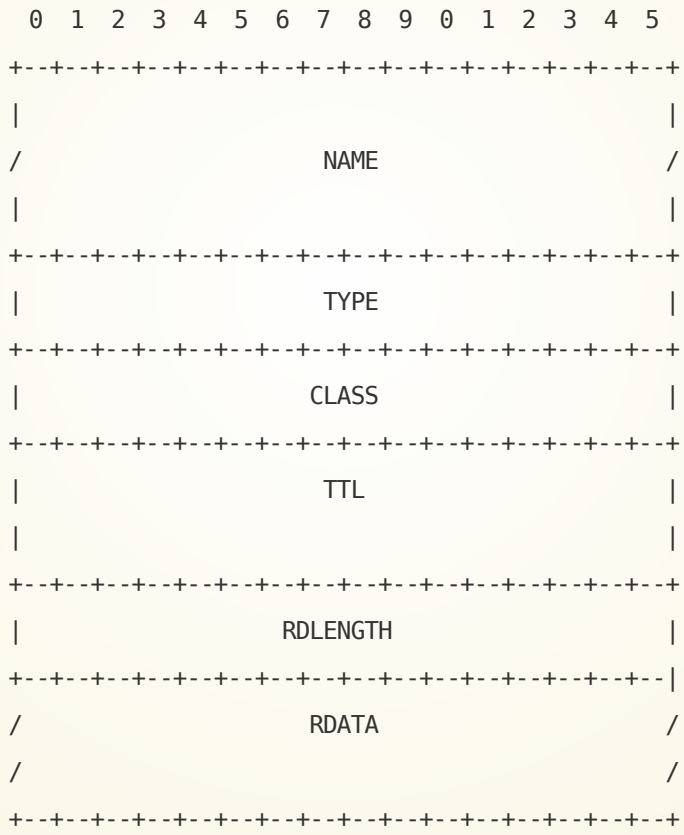
GRAMMAR - DNS SPEC (1)

PARSING DNS HEADER

```
header = record{
    field('id',      number(16)),
    field('qr',      flag),
    field('opcode',  number(4)),
    field('aa',      flag),
    field('tc',      flag),
    field("rd",      flag),
    field("ra",      flag),
    number(3),
    field("rcode",   number(4)),
    field("qdcount", number(16)),
    field("ancount", number(16)),
    field("nscount", number(16)),
    field("arcount", number(16)),
}
```

GRAMMAR - DNS SPEC (2)

PARSING DNS RESSOURCE RECORD HEADER

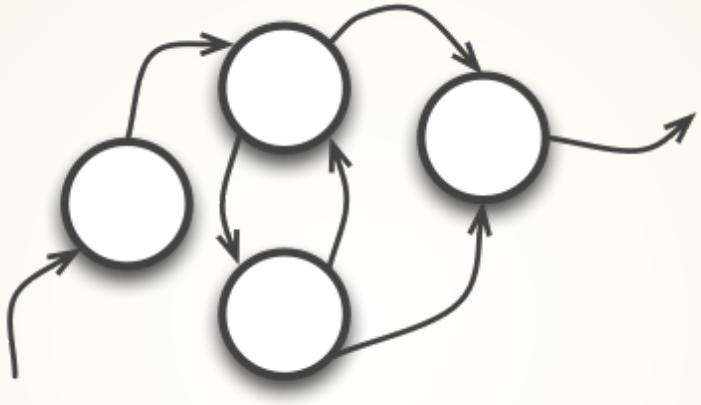


GRAMMAR - DNS SPEC (2)

PARSING DNS RESSOURCE RECORD HEADER

```
resourcerecord = record{
    field('name',    dn),
    field('type',    dns_dissector.type),
    field('class',   number(16)),
    field('ttl',     number(32)),
    field('length',  number(16)),
    branch({
            A =       field('ip', number(32)
                :convert(ipv4_addr_convert, true)),
            NS =      field('name', dn),
            ...
        },
        function (self, ctx)
            return self.type
        end
    ),
}
```

STATE MACHINE



STATE MACHINE EXAMPLE

```
state_type(BidirectionnalState)

request = state(request_grammar, nil)
response = state(nil, response_grammar)

request:on{
    event = events.down,
    action = function (self, res)
        self.request = res
    end,
    jump = connect
}

request:on{
    event = events.parse_error,
    action = function (self, res) ... end,
    jump = fail
}

initial(request)
```

AUXILIARY FEATURES

- Debugger
  - Inspect an existing Haka script file
  - list source code, set breakpoints, follow code execution or dump the content of variables
- Interactive packet filtering mode
- Intergrated console to get info and live stats

CONCLUSIONS

- Open source security oriented language:
  - Specify security rules (0.1 version)
  - Specify protocol message syntax + state machine (0.2 version)
- Modular architecture to apply user's specified policy on **live**-captured trafic
  - Multiple packet capture module: nfqueue, pcap
  - Logging and alerting module

FUTURE WORKS

PERFORMANCES

- **Packet capture**: add efficient packet capture modules (netmap, dpdk)
- **Security rules**: optimize the way they are evaluated
- **Grammar**: rework grammar compilation
- **Multi-thread support**: optimize their use

SEEMS INTERESTING ?

DON'T MISS OUR TUTORIAL ON THURSDAY 10 (9H30 A.M)

QUESTIONS ?


haka-security.org


@hakasecurity


haka-security/haka