

ntopng: an actionable event-driven network traffic analysis application

Luca Deri <deri@ntop.org>

Alfredo Cardigliano <cardigliano@ntop.org>

Welcome to ntopng

ntopng is a realtime network and cybersecurity traffic monitoring app.

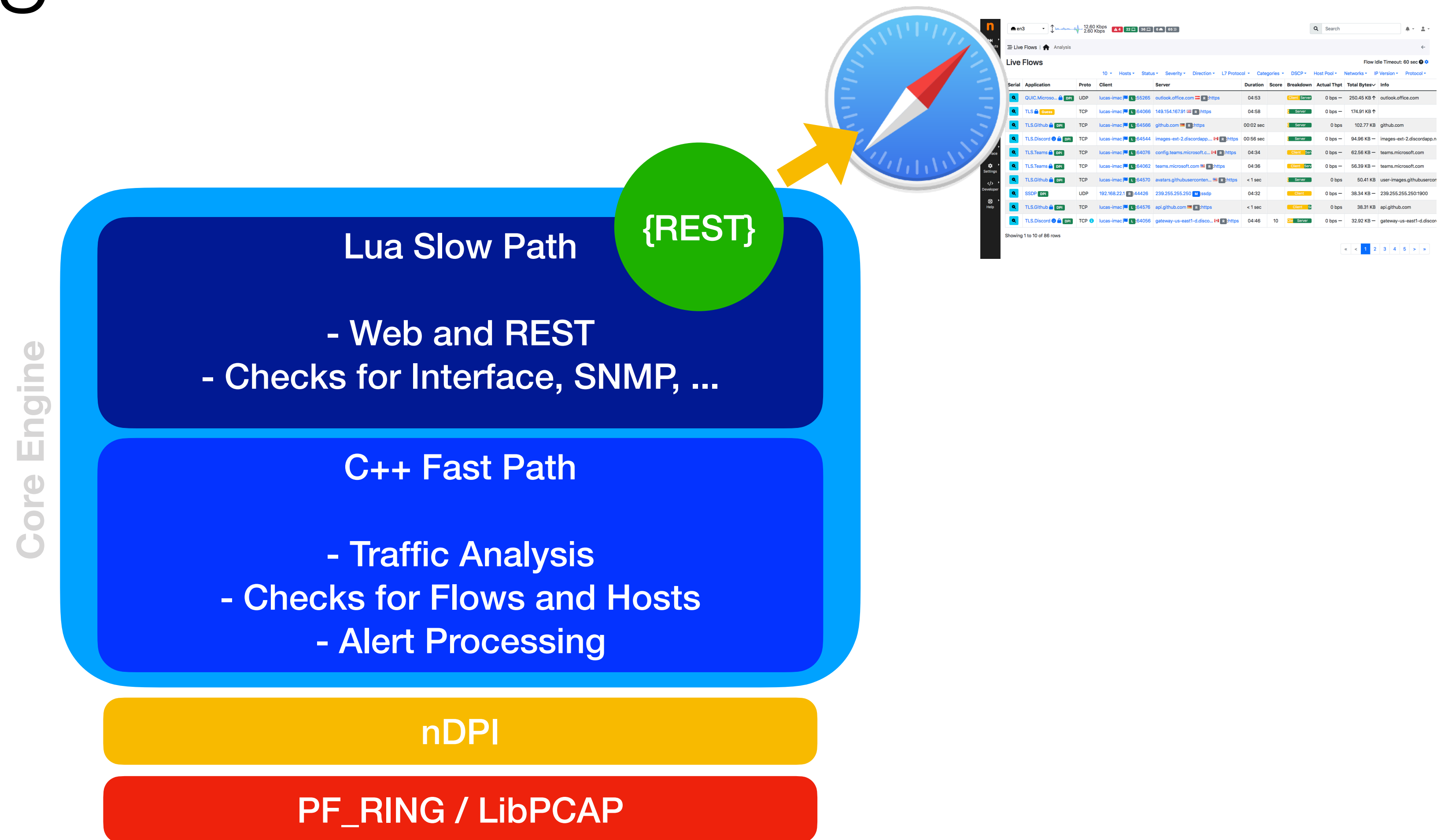
The screenshot displays the ntopng interface with a sidebar on the left containing navigation options: Shortcuts, Dashboard, Alerts, Flows (highlighted), Hosts, Maps, Interface, Settings, Developer, and Help. The main content area shows a 'Live Flows' table under the 'Analysis' tab. The table has a search bar and a 'Flow Idle Timeout: 60 sec' setting. The table columns are: Serial, Application, Proto, Client, Server, Duration, Score, Breakdown, Actual Thpt, Total Bytes, and Info. The table contains 10 rows of data, showing various network flows with their respective protocols, clients, servers, durations, and byte counts.

| Serial | Application | Proto | Client | Server | Duration | Score | Breakdown | Actual Thpt | Total Bytes | Info |
|--------|-----------------|-------|------------------------|--|-----------|-------|---------------|-------------|-------------|-------------------------------|
| | QUIC.Microso... | UDP | lucas-imac L:55265 | outlook.office.com R:https | 04:53 | | Client Server | 0 bps | 250.45 KB | ↑ outlook.office.com |
| | TLS | TCP | lucas-imac L:64066 | 149.154.167.91 R:https | 04:58 | | Server | 0 bps | 174.91 KB | ↑ |
| | TLS.Github | TCP | lucas-imac L:64566 | github.com R:https | 00:02 sec | | Server | 0 bps | 102.77 KB | github.com |
| | TLS.Discord | TCP | lucas-imac L:64544 | images-ext-2.discordapp... R:https | 00:56 sec | | Server | 0 bps | 94.96 KB | — images-ext-2.discordapp.n |
| | TLS.Teams | TCP | lucas-imac L:64076 | config.teams.microsoft.c... R:https | 04:34 | | Client Ser | 0 bps | 62.56 KB | — teams.microsoft.com |
| | TLS.Teams | TCP | lucas-imac L:64062 | teams.microsoft.com R:https | 04:36 | | Client Serv | 0 bps | 56.39 KB | — teams.microsoft.com |
| | TLS.Github | TCP | lucas-imac L:64570 | avatars.githubusercontent... R:https | < 1 sec | | Server | 0 bps | 50.41 KB | user-images.githubusercontent |
| | SSDP | UDP | 192.168.22.1 R:44426 | 239.255.255.250 M:ssdp | 04:32 | | Client | 0 bps | 38.34 KB | — 239.255.255.250:1900 |
| | TLS.Github | TCP | lucas-imac L:64576 | api.github.com R:https | < 1 sec | | Client Se | 0 bps | 38.31 KB | api.github.com |
| | TLS.Discord | TCP | lucas-imac L:64056 | gateway-us-east1-d.disco... R:https | 04:46 | 10 | Client Server | 0 bps | 32.92 KB | — gateway-us-east1-d.discon |

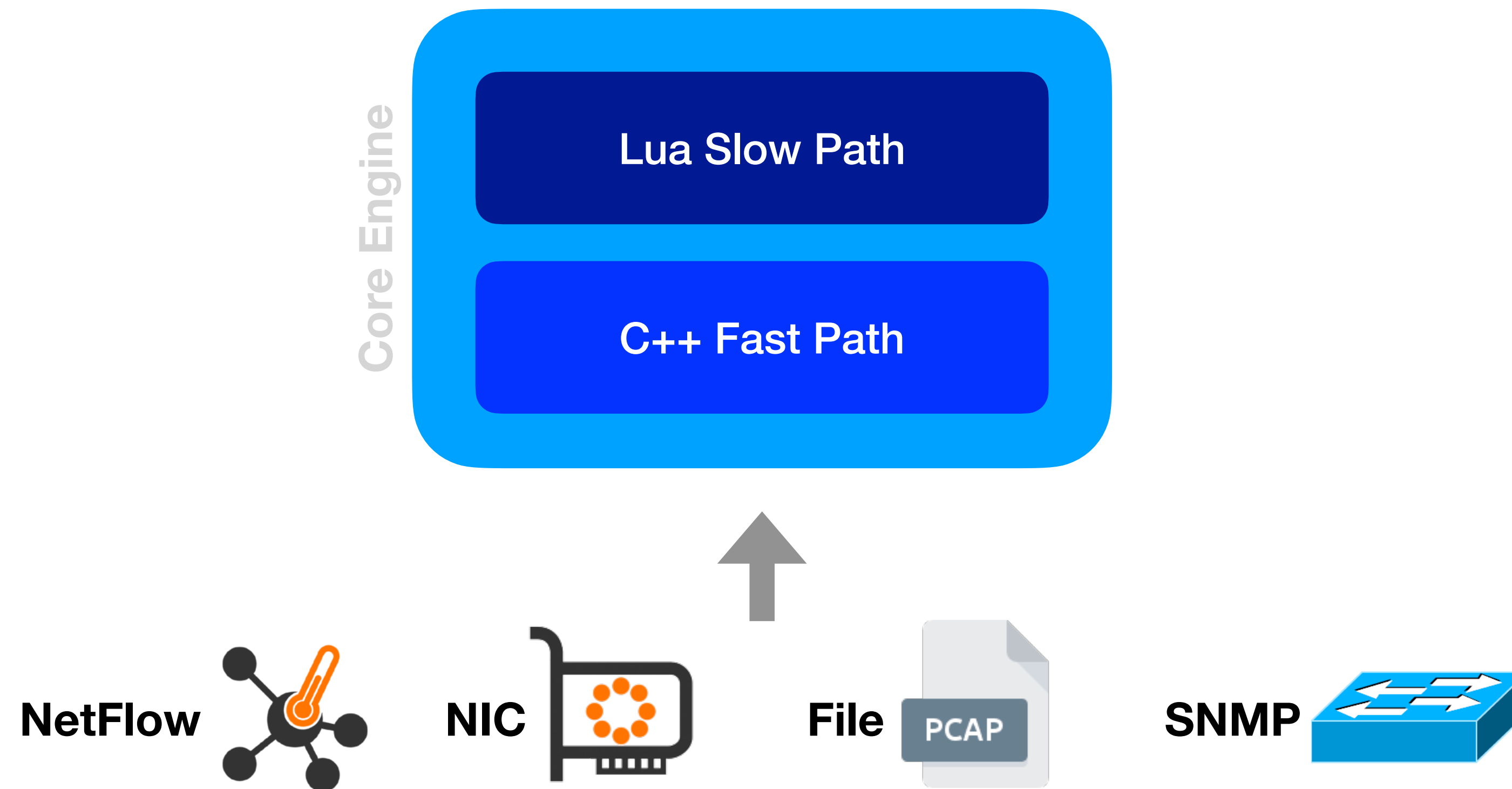
Showing 1 to 10 of 86 rows

« < 1 2 3 4 5 > »

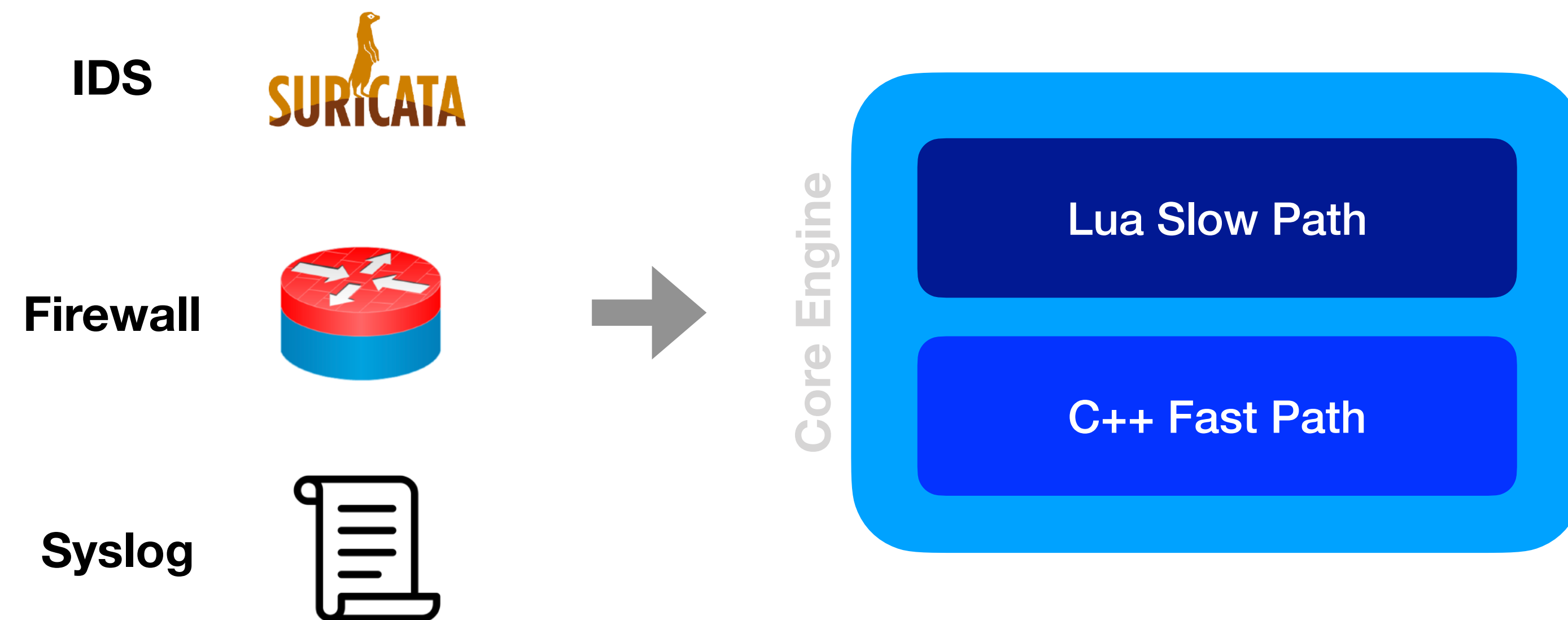
ntopng Architecture: Overview



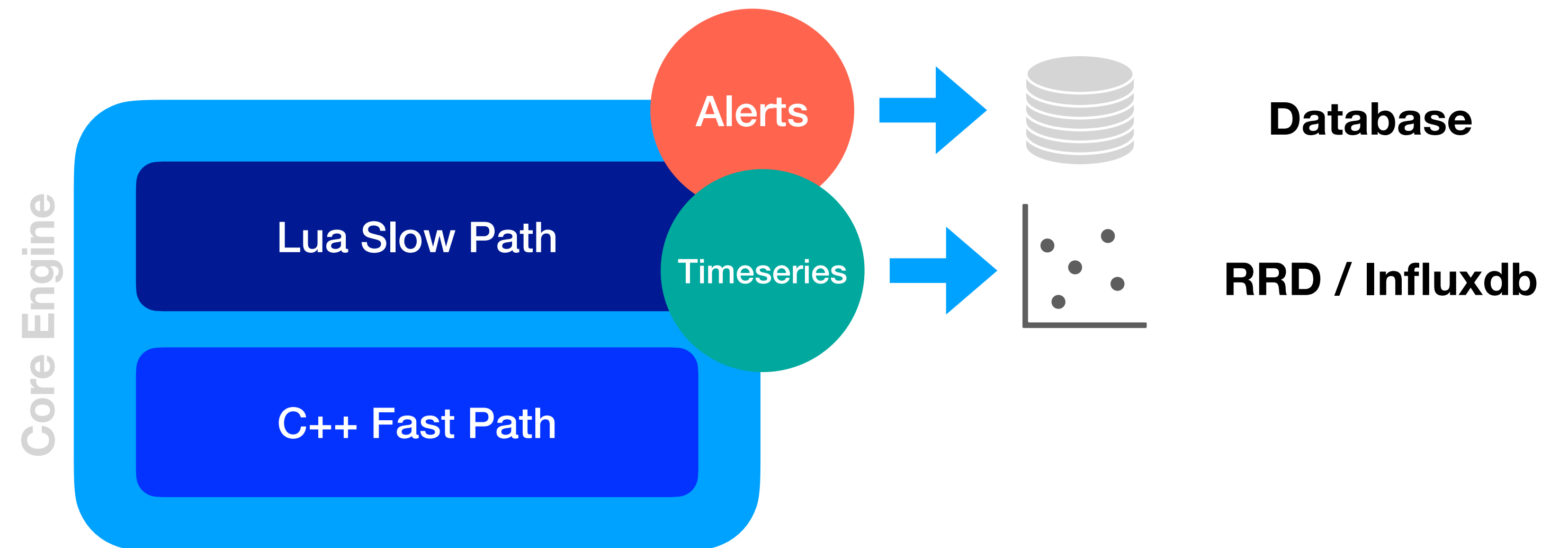
ntopng Architecture: Traffic Ingestion



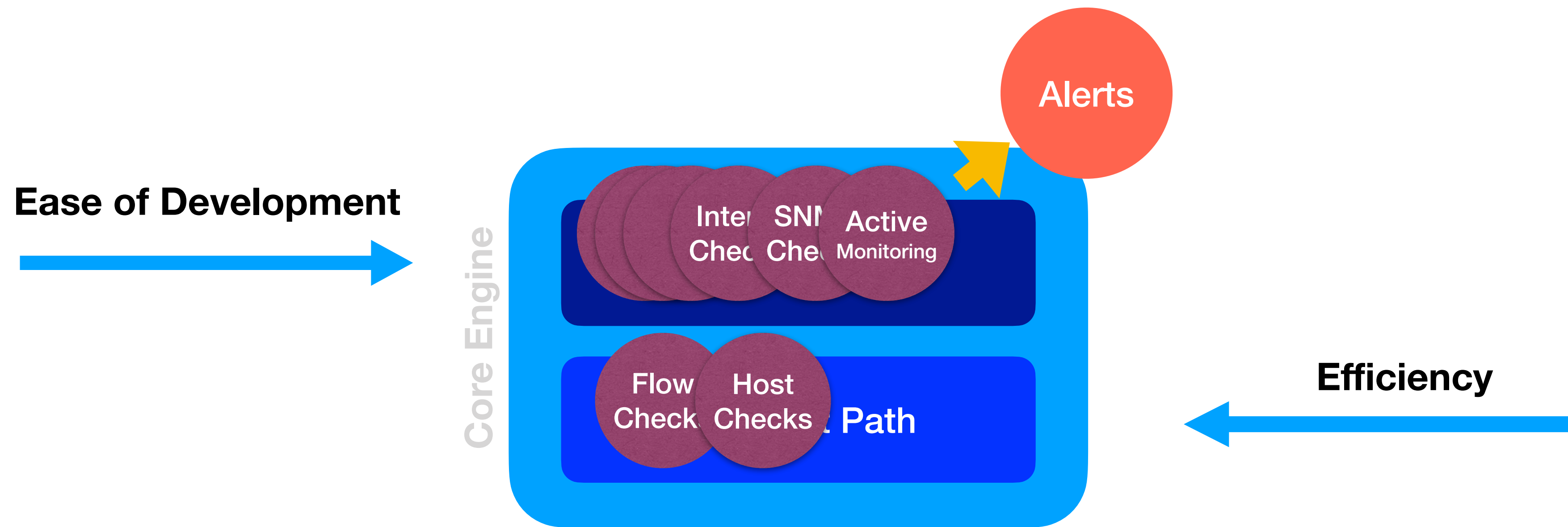
ntopng Architecture: Events Ingestion



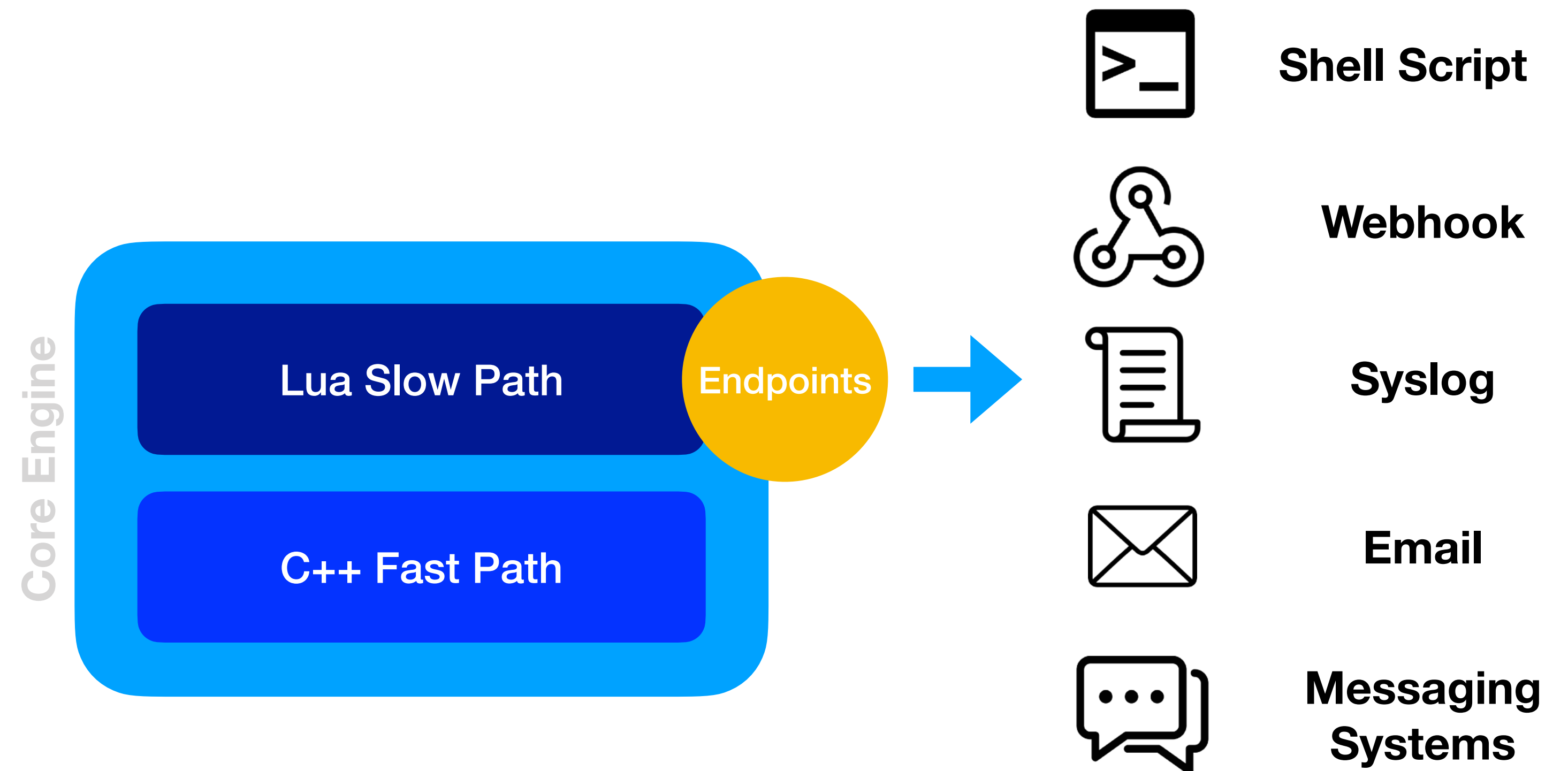
ntopng Architecture: Historical Data



ntopng Architecture: Checks

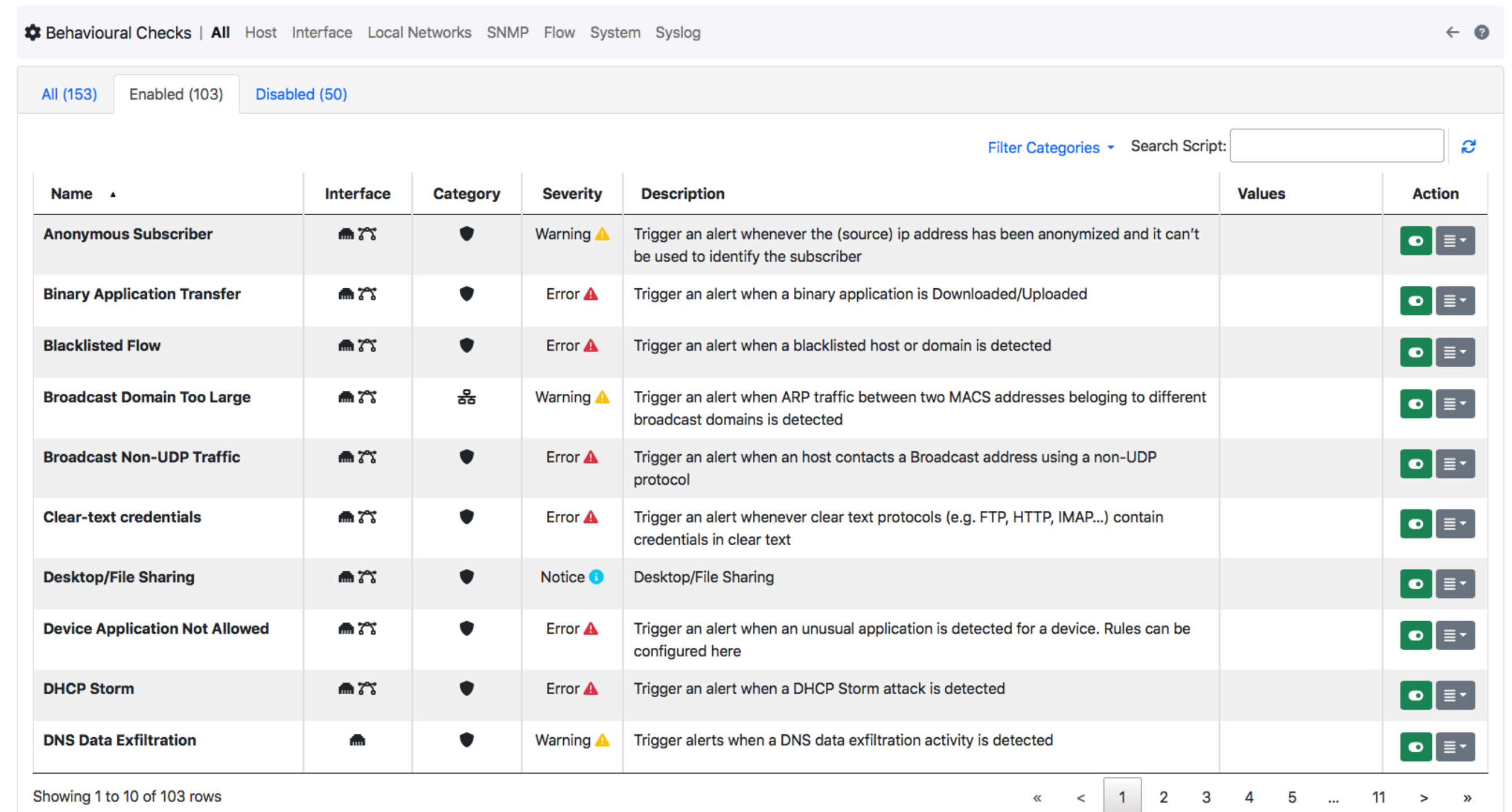


ntopng Architecture: Notifications



Problem Statement [1/2]

- How to extend ntopng's Behavioural Checks to meter something additional with respect to built-in checks?
- How can ntopng be used from languages such as Python to extract monitored data or traffic alerts?



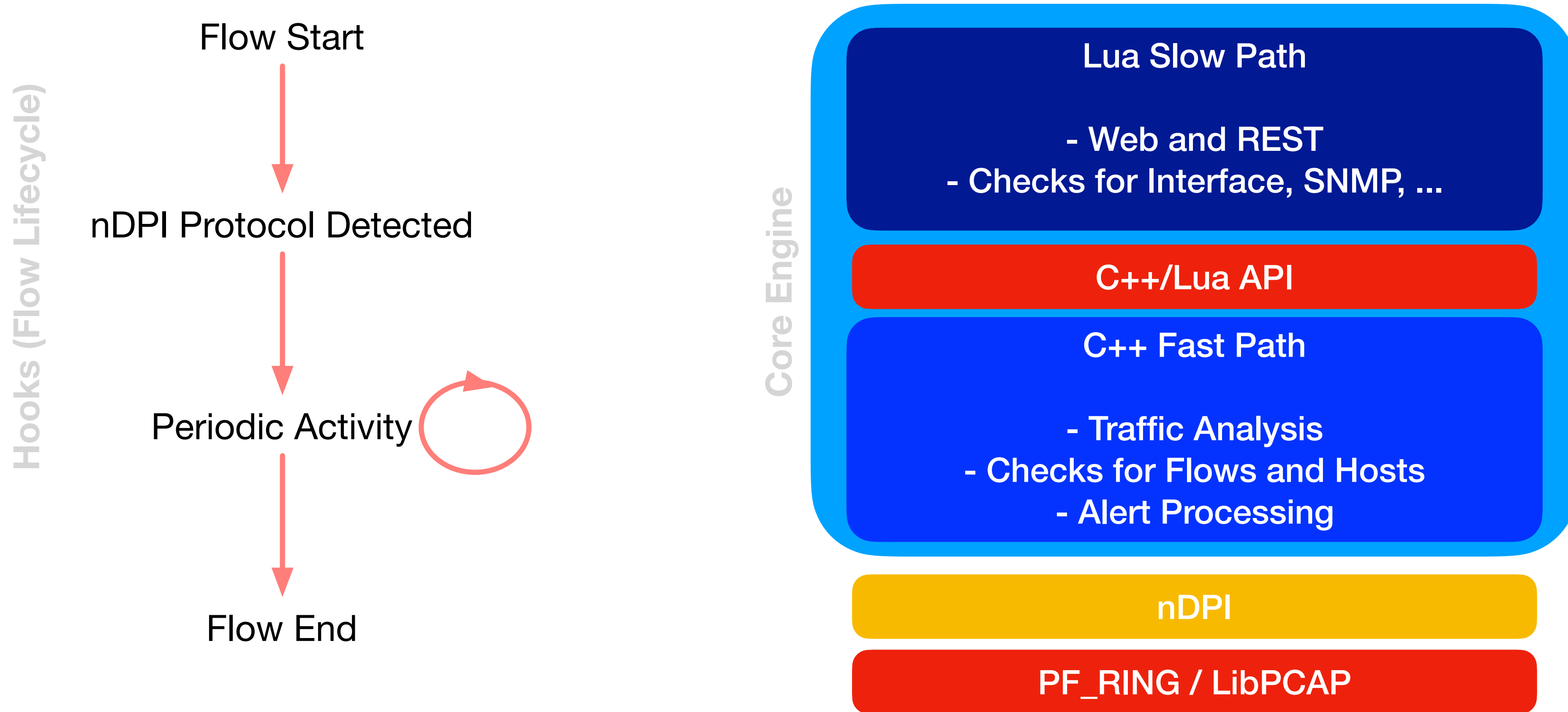
The screenshot shows the 'Behavioural Checks' interface in ntopng. It features a navigation bar with categories like 'All', 'Host', 'Interface', 'Local Networks', 'SNMP', 'Flow', 'System', and 'Syslog'. Below the navigation bar, there are tabs for 'All (153)', 'Enabled (103)', and 'Disabled (50)'. A search bar labeled 'Search Script:' is present. The main content is a table with columns: Name, Interface, Category, Severity, Description, Values, and Action. The table lists various checks such as 'Anonymous Subscriber', 'Binary Application Transfer', 'Blacklisted Flow', 'Broadcast Domain Too Large', 'Broadcast Non-UDP Traffic', 'Clear-text credentials', 'Desktop/File Sharing', 'Device Application Not Allowed', 'DHCP Storm', and 'DNS Data Exfiltration'. Each row includes a severity icon (Warning or Error) and an action menu (eye and list icons). At the bottom, it shows 'Showing 1 to 10 of 103 rows' and a pagination control with page numbers 1 through 11.

| Name | Interface | Category | Severity | Description | Values | Action |
|--------------------------------|-----------|----------|----------|---|--------|--------|
| Anonymous Subscriber | | | Warning | Trigger an alert whenever the (source) ip address has been anonymized and it can't be used to identify the subscriber | | |
| Binary Application Transfer | | | Error | Trigger an alert when a binary application is Downloaded/Uploaded | | |
| Blacklisted Flow | | | Error | Trigger an alert when a blacklisted host or domain is detected | | |
| Broadcast Domain Too Large | | | Warning | Trigger an alert when ARP traffic between two MACS addresses belonging to different broadcast domains is detected | | |
| Broadcast Non-UDP Traffic | | | Error | Trigger an alert when an host contacts a Broadcast address using a non-UDP protocol | | |
| Clear-text credentials | | | Error | Trigger an alert whenever clear text protocols (e.g. FTP, HTTP, IMAP...) contain credentials in clear text | | |
| Desktop/File Sharing | | | Notice | Desktop/File Sharing | | |
| Device Application Not Allowed | | | Error | Trigger an alert when an unusual application is detected for a device. Rules can be configured here | | |
| DHCP Storm | | | Error | Trigger an alert when a DHCP Storm attack is detected | | |
| DNS Data Exfiltration | | | Warning | Trigger alerts when a DNS data exfiltration activity is detected | | |

Problem Statement [2/2]

- How can we give users the ability to write **efficient** Lua-based checks for flows and hosts without requiring C++ coding skills ?
- How can we allow people to create custom checks and alerts for **specific use cases** that they only have?
 - Trigger an alert if you see TLS traffic between host A and B.
 - Send me a Telegram notification if host A receives a TLS request with certificate issued by “O=GoDaddy.com”
 - Alert me of BitTorrent connections whose throughput exceeds 1 Gbit.
 - Report me about Zoom calls with bad quality.

Lua Scripting in ntopng [1/3]



Lua Scripting in ntopng [2/3]

C++

Lua

```
static luaL_Reg _ntop_host_reg[] = {  
  { "ip",          ntop_host_get_ip          },  
  { "mac",         ntop_host_get_mac         },  
  { "name",        ntop_host_get_name        },  
  { "vlan_id",     ntop_host_get_vlan_id     },  
  
  { "is_unicast",  ntop_host_is_unicast    },  
  { "is_multicast", ntop_host_is_multicast },  
  { "is_broadcast", ntop_host_is_broadcast },  
  { "is_blacklisted", ntop_host_is_blacklisted },  
  
  { "bytes_sent",  ntop_host_get_bytes_sent },  
  { "bytes_rcvd", ntop_host_get_bytes_rcvd },  
  { "bytes",       ntop_host_get_bytes_total },  
  { "l7",          ntop_host_get_l7_stats   },  
  
  { "skipVisitedHost", ntop_skip_visited_host },  
  { "triggerAlert",    ntop_trigger_host_alert },  
}
```

```
function dump_host()  
  local rsp = {}  
  
  rsp.ip          = host.ip()  
  rsp.mac         = host.mac()  
  rsp.name        = host.name()  
  rsp.vlan_id     = host.vlan_id()  
  rsp.is_unicast  = host.is_unicast()  
  rsp.is_multicast = host.is_multicast()  
  rsp.is_broadcast = host.is_broadcast()  
  rsp.is_blacklisted = host.is_blacklisted()  
  rsp.bytes_sent  = host.bytes_sent()  
  rsp.bytes_rcvd  = host.bytes_rcvd()  
  rsp.bytes       = host.bytes()  
  rsp.l7          = host.l7()  
  
  tprint(rsp)  
end  
  
dump_host()
```




Lua Methods



Function Callback

Lua Scripting in ntopng [3/3]

```
void CustomFlowLuaScript::checkFlow(Flow *f, LuaEngine *lua) {  
    lua->setFlow(f);  
    lua->run_loaded_script(); /* Run script */  Call LuaVM  
  
    if(f->isCustomFlowAlertTriggered()) {  
        FlowAlertType alert_type = CustomFlowLuaScriptAlert::getClassType();  
        u_int8_t c_score, s_score;  
        risk_percentage cli_score_pctg = CLIENT_FAIR_RISK_PERCENTAGE;  
  
        computeCliSrvScore(alert_type, cli_score_pctg, &c_score, &s_score);  
  
        f->triggerAlertAsync(alert_type, c_score, s_score);  
    }  
}
```

Simple Flow Check (User Script)

Current Flow



```
local t = flow.tls_quic()

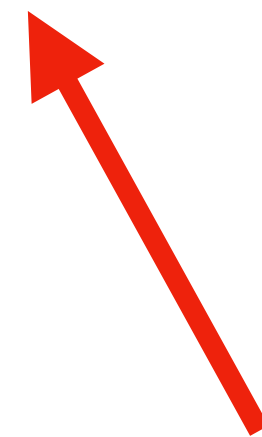
if((flow.cli() == "192.168.1.178") and (flow.srv() == "192.168.1.1") and (t ~= nil)) then

    if(t["protos.tls.issuerDN"] == "CN=AGCOMBO, O=Technicolor, OU=1827SAZCH") then
        local score = 111
        local message = "Found unexpected TLS/QUIC flow 192.168.1.178 -> 192.168.1.1 (invalid certificate)"

        flow.triggerAlert(score, message)

        dump_flow()
    end
end

return(0)
```



Check Overhead (Intel i3, 2010)

- Lua: ~30 usec
- C++: ~1 usec

Simple Flow Check (Alert)

The screenshot shows the ntopng Alerts Explorer interface. At the top, there's a status bar with 'en3' and network statistics: 55.70 Kbps (up) and 416.60 Kbps (down). Below that, the 'Alerts Explorer' header shows filters for 'All', 'Host', 'Flow', 'MAC Address', 'System', and 'User'. The main area displays a timeline of alerts from 03/02/2023 14:36 to 15:06. A table below shows a single alert entry:

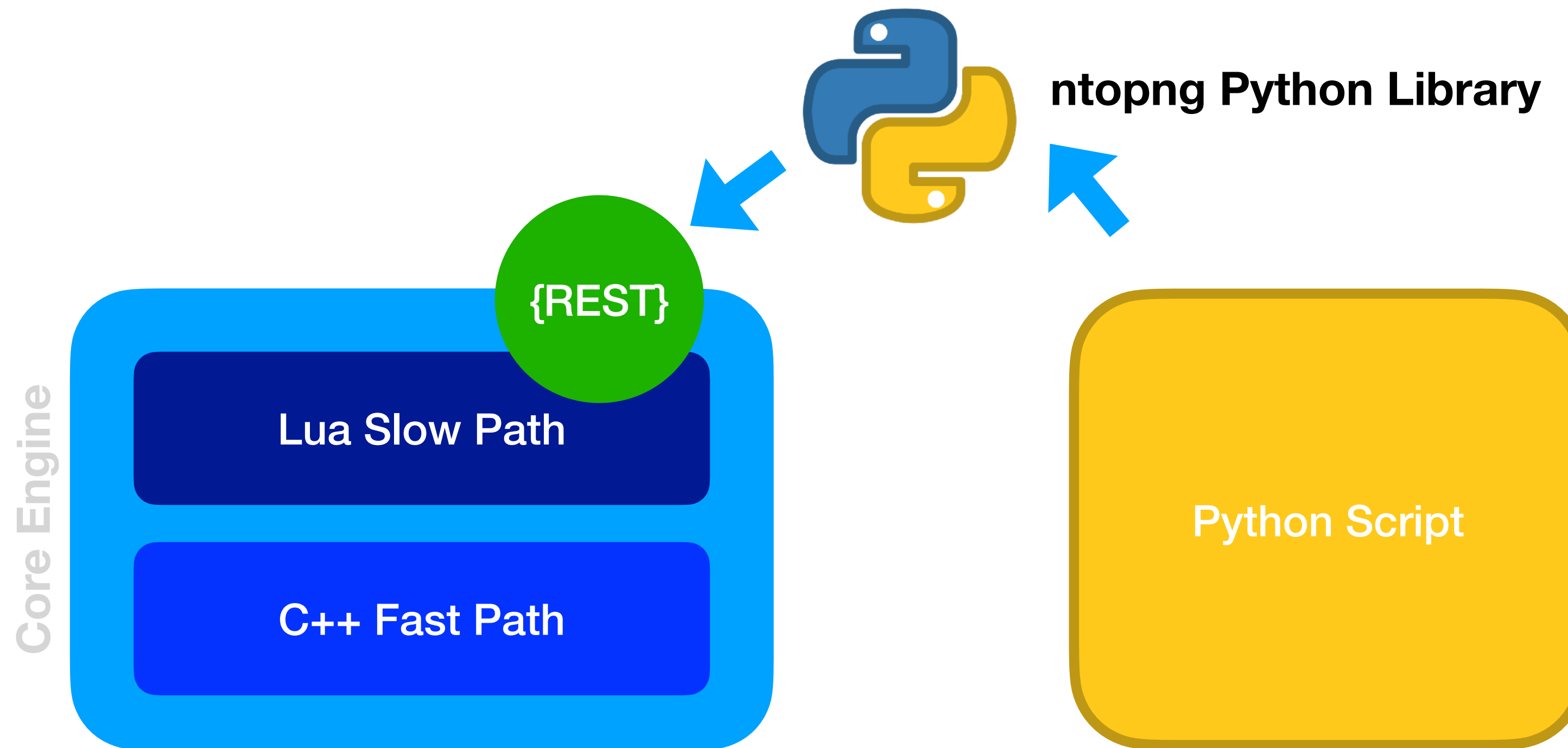
| Date/Time | Score | Application | Alert | Flow | Actions |
|-----------|-------|--------------------------|---------------|----------------------------------|---------|
| 15:06:41 | 350 | TCP:TLS DPI | Custom Script | 192.168.1.178:54810 ↔ router:443 | [Menu] |

The alert details are expanded, showing:

- Information Found unexpected TLS/QUIC flow 192.168.1.178 -> 192.168.1.1 (invalid certificate)
- Other Issues TLS Certificate Self-signed
- Unsafe TLS Ciphers
- Missing TLS SNI
- Info
- Flow Related Info [TLS Certificate Validity: 01/01/2005 01:00:00 - 31/12/2024 01:00:00] [Cipher State: **weak**] [Server to Client Traffic: 1.61 KB | Client to Server Traffic: 1.61 KB]
- Client Pool [Default](#)
- Server Pool [Default](#)
- Client Network [192.168.1.0/24](#)
- Server Network [192.168.1.0/24](#)
- Flow Exporter

A red arrow points from the text 'Triggered Alert' to the first bullet point in the alert details.

Python 3 API



```
$ pip3 install ntopng
```


Script Example: Live Data

```
#!/usr/bin/env python3
```

```
from ntopng.ntopng import Ntopng
```

```
auth_token = "532b8dbe1092e591435c7a13f2"
```

Connect to ntopng

```
ntopng_handle = Ntopng(auth_token=token)
```

Get a live interface instance

```
interface = ntopng_handle.get_interface(iface_id)
```

```
hosts = interface.get_active_hosts()
```

Print all active hosts

```
for host in hosts:
```

```
    print(host['ip'] + " => ", end='')
```

```
    print(host)
```

Script Example: Historical Data

```
#!/usr/bin/env python3
```

```
from ntopng.ntopng import Ntopng
```

```
auth_token = "532b8dbe1092e591435c7a13f2"
```

Connect to ntopng



```
ntopng_handle = Ntopng(auth_token=token)
```

Get an historical interface instance



```
interface = ntopng_handle.get_interface(iface_id)
```

```
historical = interface.get_historical()
```

```
epoch_end = int(time.time())
```

```
epoch_begin = epoch_end - 24*60*60
```

```
all_stats = historical.get_alerts_stats(epoch_begin, epoch_end)
```

Get the alerts summary for a time interval

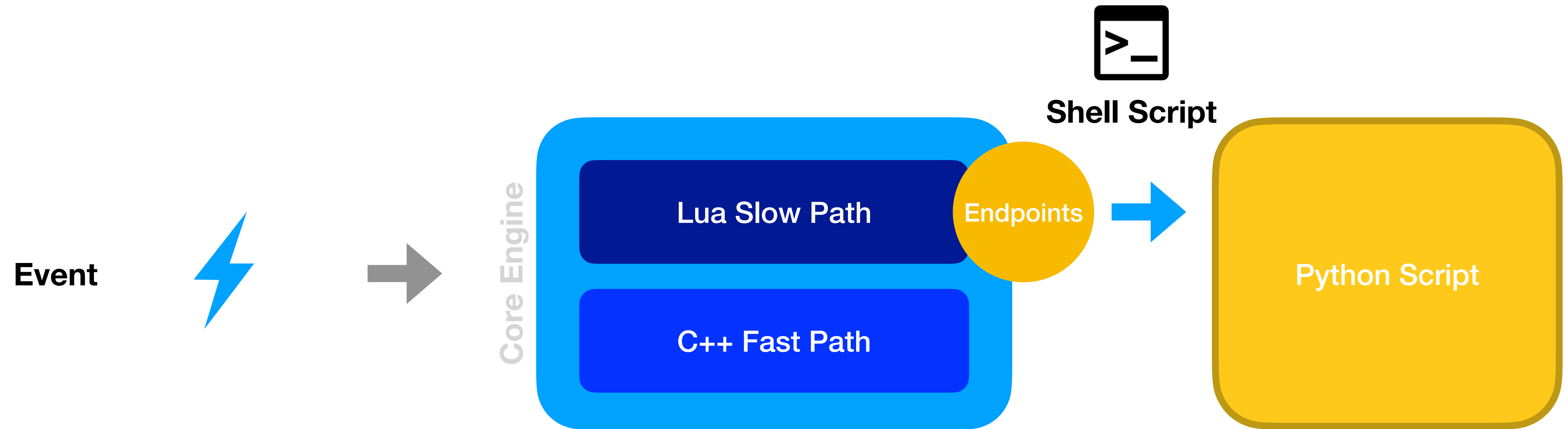


```
for stats in all_stats:
```

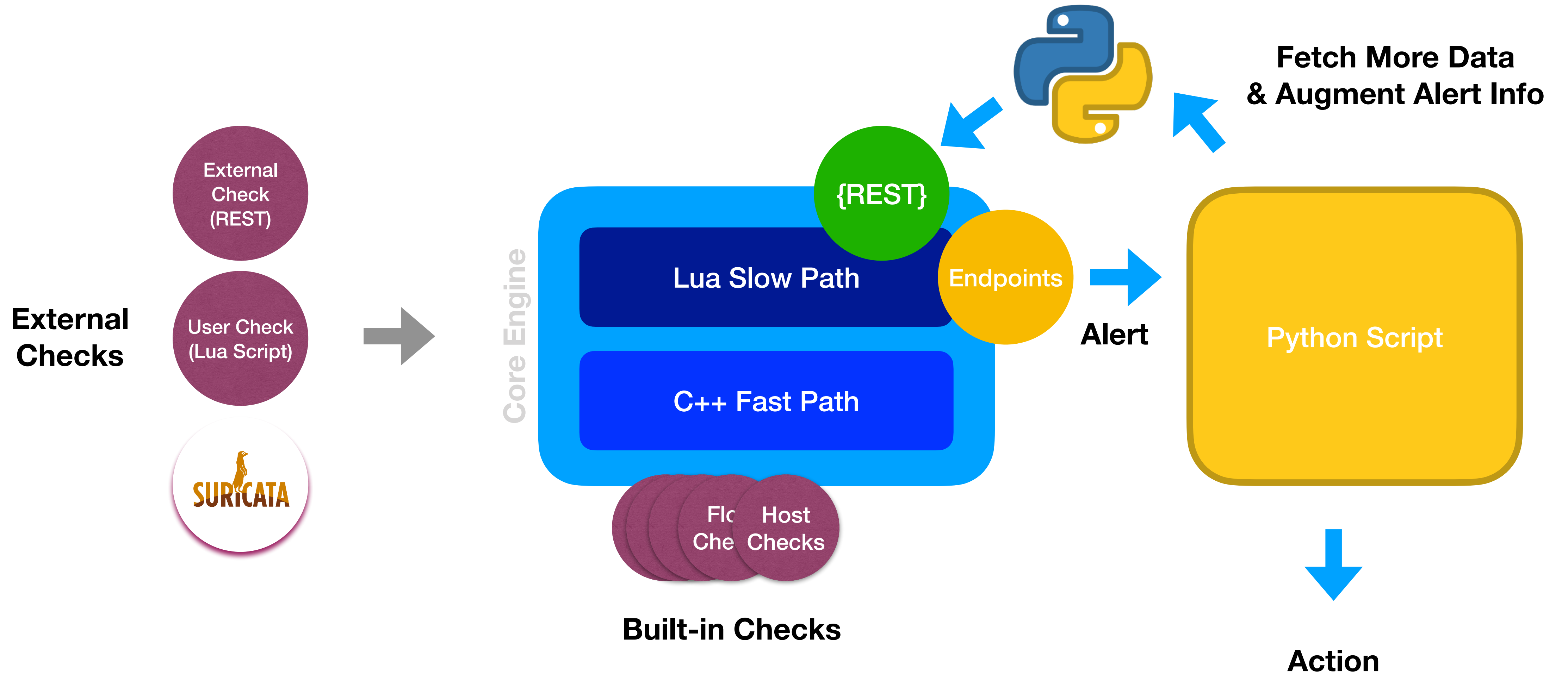
```
    print(stats['label'])
```

```
    print(stats['value'])
```

Python Script as Alert Handler



Let's Put Pieces Together



Step 1: Enable The Check

The screenshot shows the ntopng interface for configuring behavioural checks. The top navigation bar includes a search bar and a notification bell with 3 alerts. The main content area is titled 'Behavioural Checks' and has tabs for 'All (25)', 'Enabled (20)', and 'Disabled (5)'. A table lists various checks with columns for Name, Interface, Category, Severity, Description, Values, and Action. The 'Host User Check Script' check is highlighted with a red circle around its action buttons, which include a red 'off' button and a grey 'on' button.

| Name | Interface | Category | Severity | Description | Values | Action |
|---------------------------|-----------|----------|----------|---|-----------------------------|--------|
| Host User Check Script | | | Error | Trigger a host alert based on a custom Lua user script | | |
| ICMP Flood Alert | | | Error | Trigger an alert when the number of sent/received ICMP Flows/sec exceeds the threshold | > 1 ICMP Flows/sec (Minute) | |
| NTP Server Contacts Alert | | | Notice | Trigger an alert when the number of different NTP servers contacted exceeds the threshold | > 5 Contacts (Minute) | |
| NTP Traffic Alert | | | Error | Trigger an alert when the Layer 2 bytes delta (sent + received) for NTP traffic exceeds the threshold | > (1 Byte) | |
| P2P Traffic Alert | | | Error | Trigger an alert when the Layer 2 bytes delta (sent + received) for P2P traffic exceeds the threshold | > (1 Byte) | |
| Packets Alert | | | Error | Trigger an alert when the packet delta (sent + received) exceeds the threshold | > 1 Packets (Minute) | |
| Remote Connection | | | Notice | Trigger an alert whenever an host has at least one active flow using a remote access protocol | | |
| RST Scan Alert | | | Error | Trigger an alert when the number of sent/received RSTs/min (with no response) exceeds the threshold | > 256 RSTs/min (Minute) | |
| Scan Detection Alert | | | Error | Trigger an alert when a scan (host/port) is detected when the number of incomplete TCP/UDP flows exceeds the specified t... | > 1 Flows (Minute) | |
| Score Anomaly | | | Error | Detects anomalies in host score | | |

Step 2: Write The Alert Handler

```
$ cat /usr/share/ntopng/scripts/shell/event-handler.py
```

```
#!/usr/bin/env python3
```

```
import sys
import json
```

```
input = sys.stdin.readlines()
alert = json.loads(input[0])
```

```
if alert["alert_id"] == host_user_check_script:
```

```
    user_info = json.loads(alert["json"])
```

```
    log(alert["ip"])
```

```
    log(user_info["message"])
```

Step 3: Set The Handler As Recipient

The image displays two screenshots from the ntopng web interface. The left screenshot shows the 'Add New Endpoint' dialog box. The 'Name' field is 'MyEventHandler', the 'Type' is 'Shell Script', and the 'Script PATH' field has a dropdown menu with 'event-handler.py' selected and circled in red. Below the field, there is a note: 'The script must be stored in "/usr/share/ntopng/scripts/shell/" and 'Alert information are provided to the script through the standard input in JSON format.'. An 'Add' button is at the bottom right.

The right screenshot shows the 'Add New Recipient' dialog box. The 'Name' field is 'MyEventHandlerRecipient', the 'Endpoint' dropdown is set to 'MyEventHandler', and the 'Options' field is empty. Below this, there are instructions: 'Insert here the options with which the script is going to be executed (e.g. '-i eno1 -p 2220')'. Further down, there are several dropdown menus: 'Minimum Severity' (Critical), 'Check Categories' (Cybersecurity), 'Check Entities' (Host), 'Host Pools' (Default), and 'Active Monitoring' (Nothing selected). A 'Check' button is at the bottom left and an 'Add' button is at the bottom right.

Step 4: Extend The Alert Handler

```
from pathlib import Path
from redmail import EmailSender
from ntopng.ntopng import Ntopng
from ntopng.report import Report
```

```
[...]
```

```
auth_token = "532b8dbe1092e591435c7a13d561db73"
report_path = "/tmp/report.pdf"
```

Connect to ntopng to fetch more data



```
ntopng_handle = Ntopng(auth_token=token)
```

Build a PDF report



```
generator = Report(ntopng_handle, alert["ifid"], alert["ip"])
generator.build(report_path)
```

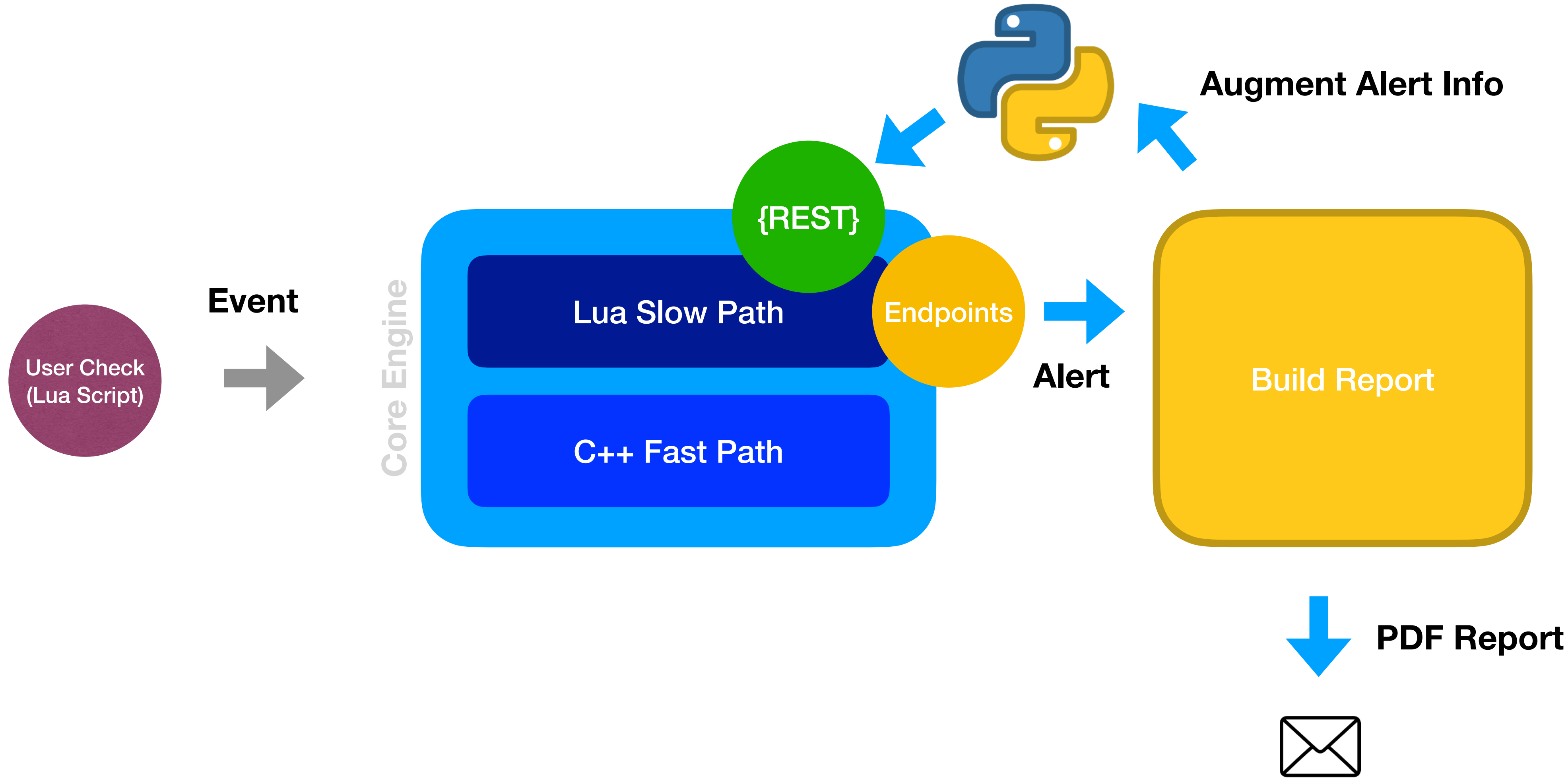
Send the report by email



```
email = EmailSender(host=e_host, port=e_port, username=e_username, password=e_password)
```

```
email.send(sender = e_sender, receivers = [e_recipient], subject = e_subject,
attachments = { "report.pdf": Path(report_path) } )
```


Example: Alert Report





17:11

< 5 ^ v

Host 192.168.2.134 Alert

Unexpected traffic detected for host 192.168.2.134

Host 192.168.2.134 Report (24h)



ntopng URL: http://localhost:3000
Date: 03-02-2023 16:53:31
Interface: 0 (of 1 interfaces)

Top Alerts

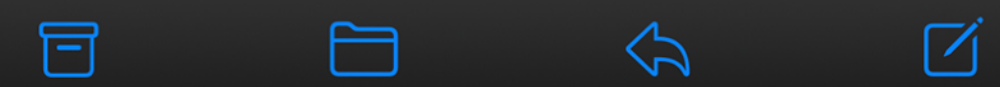
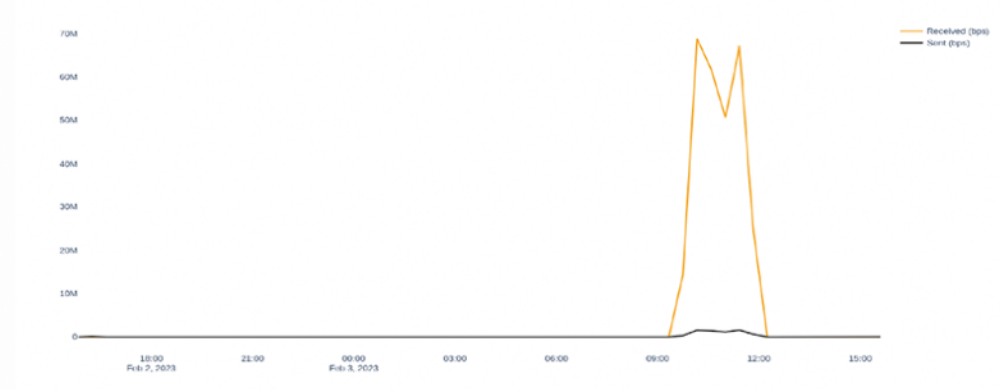
| Alert type | Severity | Alert type | Occurrence % |
|-----------------------------|----------|-----------------------------|--------------|
| Pkt Threshold | Error | Periodic Flow | 75.8 % |
| NTP Traffic | Error | Endpoint Shell Script Ex... | 7.7 % |
| HTTP Suspicious User-Age... | Error | Pkt Threshold | 4.4 % |
| Binary Application Trans... | Error | Unidirectional Traffic | 4.1 % |
| DNS Traffic | Error | App. on Non-Sid Port | 2.8 % |
| TCP Packets Issues | Error | Low Goodput Ratio | 1.4 % |
| Blacklisted Country | Error | TLS not carrying HTTPS | 1.1 % |
| Scan Detected | Error | NTP Traffic | 0.4 % |
| Custom Script | Error | Remote to Remote | 0.4 % |
| Clear-tx credentials | Error | Error Code | 0.4 % |

Top Contacts

| Client Name | Client IP | Server Name | Server IP | Traffic |
|--------------|---------------|---------------------------|-----------------|-----------|
| devele | 192.168.2.134 | 151.11.48.122 | | 35.19 GB |
| raspberrypi3 | 192.168.2.153 | devele | 192.168.2.134 | 81.37 MB |
| devele | 192.168.2.134 | files.pythonhosted.org | 151.101.65.63 | 1.11 MB |
| devele | 192.168.2.134 | urthaus.abuse.ch | 151.101.2.49 | 918.72 KB |
| devele | 192.168.2.134 | cdn.hexpm.org | 151.101.194.49 | 829.67 KB |
| devele | 192.168.2.134 | grafana.com | 34.120.177.193 | 346.67 KB |
| devele | 192.168.2.134 | raw.githubusercontent.com | 185.199.109.133 | 135.31 KB |
| devele | 192.168.2.134 | pypi.org | 151.101.192.223 | 134.72 KB |
| devele | 192.168.2.134 | feed.ntop.org | 178.62.197.130 | 116.8 KB |
| devele | 192.168.2.134 | raw.githubusercontent.com | 185.199.108.133 | 95.15 KB |

Traffic Summary

Total Sent/Received: 1290.11 / 51429.26 MB



Final Remarks

- ntopng is now a versatile flow collector and traffic analysis application.
- It combines traditional traffic visibility with cybersecurity analysis.
- Lua-based engine scripting enables the creation of behavioural checks that can be implemented in seconds by non-skilled developers.
- They are fast enough to be used in Gbit networks, but converting them to C++ is required for scaling up.
- Python bindings enable programmers to use ntopng as both a live and historical data source, and also extend ntopng in areas such as PDF reporting not currently available as native feature.
- All code is available at <https://github.com/ntop/ntopng>