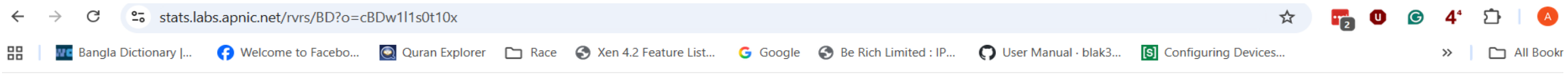
What Your DNS Logs Say About CDN Efficiency: A Practical Approach for ISP Visibility

Abu Sufian, Head of Technology (Udoy) ADN Telecom Limited | APNIC Community Trainer

DNS Reality in APNIC Stats for BD

60–70% of traffic resolved via Google/Cloudflare



Code	Region	googlepdns	sameas	cloudflare	samecc	diffcc	opendns	level3	quad9	dns4eu	quad101	Samples	Weight	Weighted Samples
XA	World	11.533%	63.577%	3.751%	16.732%	0.937%	0.480%	0.107%	0.113%	0.091%	0.000%	32,343,093	1	32,343,093
XD	Asia	7.363%	62.765%	1.855%	21.967%	0.971%	0.264%	0.056%	0.050%	0.045%	0.000%	18,539,433	1	18,272,624

Code	SubRegion	googlepdns	sameas	cloudflare	samecc	diffcc	opendns	level3	quad9	dns4eu	quad101	Samples	Weight	Weighted Samples
XT	Southern Asia, Asia	9.499%	57.306%	1.985%	29.009%	1.557%	0.351%	0.130%	0.042%	0.121%	0.000%	5,713,489	1	5,835,599

CC	Country	googlepdns	sameas	cloudflare	samecc	diffcc	opendns	level3	quad9	dns4eu	quad101	Samples	Weight	Weighted Samples
IN	India, Southern Asia, Asia	7.600%	60.142%	0.633%	31.051%	0.052%	0.367%	0.116%	0.038%	0.001%	0.000%	4,318,628	1.1	4,591,344
PK	Pakistan, Southern Asia, Asia	0.047%	57.870%	2.221%	25.125%	14.615%	0.054%	0.010%	0.057%	0.001%	0.000%	644,283	0.5	310,055
BD	Bangladesh, Southern Asia, Asia	51.186%	36.484%	5.900%	5.489%	0.377%	0.262%	0.194%	0.103%	0.006%	0.000%	432,807	0.6	266,933
NP	Nepal, Southern Asia, Asia	16.658%	74.468%	3.124%	5.518%	0.176%	0.011%	0.001%	0.022%	0.022%	0.000%	87,977	0.7	58,297
LK	Sri Lanka, Southern Asia, Asia	13.393%	85.869%	0.555%	0.015%	0.136%	0.023%	0.000%	0.000%	0.010%	0.000%	47,922	1.3	62,457
AF	Afghanistan, Southern Asia, Asia	11.108%	4.060%	14.307%	12.723%	39.374%	2.703%	1.441%	0.062%	14.222%	0.000%	22,416	2	45,440
IR	Iran (Islamic Republic of), Southern Asia, Asia	6.624%	0.231%	34.974%	44.188%	13.338%	0.147%	0.420%	0.070%	0.007%	0.000%	14,282	34.7	494,950
MV	Maldives, Southern Asia, Asia	36.074%	51.724%	10.331%	0.475%	0.447%	0.935%	0.000%	0.000%	0.014%	0.000%	7,163	0.4	2,741
BT	Bhutan, Southern Asia, Asia	44.825%	12.721%	3.577%	38.229%	0.201%	0.179%	0.246%	0.022%	0.000%	0.000%	4,473	0.8	3,378

DNS Comparison:
Measures how fast each resolver replies and which CDN IP it returns, showing how resolver choice impacts latency and CDN steering.

```
========================================
 DNS Comparison (Local vs 8.8.8.8 vs 1.1.1.1)
========================================

Domain: rr2---sn-5qouxaxjvhauqcq-q5jl.googlevideo.com
----------------------------------------
LOCAL      0ms      ,103.234.201.45
GOOGLE     24ms     ,103.234.201.45
CLOUDFLARE 1ms      ,103.234.201.45

Domain: www.google.com
----------------------------------------
LOCAL      1ms      ,142.250.67.36
GOOGLE     24ms     ,142.251.221.196
CLOUDFLARE 1ms      ,142.250.193.132

Domain: www.facebook.com
----------------------------------------
LOCAL      0ms      ,157.240.1.35
GOOGLE     24ms     ,57.144.214.1
CLOUDFLARE 1ms      ,31.13.64.35

Domain: scontent.fdac179-1.fna.fbcdn.net
----------------------------------------
LOCAL      1ms      ,103.234.201.209
GOOGLE     24ms     ,103.234.201.209
CLOUDFLARE 1ms      ,103.234.201.209

Domain: sf16-sg.tiktokcdn.com
----------------------------------------
LOCAL      0ms      ,23.199.69.130,23.199.69.106
GOOGLE     46ms     ,23.199.69.106,23.199.69.130
CLOUDFLARE 243ms    ,23.212.164.122,23.212.164.152

Domain: v9e.tiktokcdn.com
----------------------------------------
LOCAL      0ms      ,154.85.74.169,154.85.74.168
GOOGLE     45ms     ,154.85.74.169,154.85.74.168
CLOUDFLARE 1ms      ,154.85.74.168,154.85.74.169
```

cURL Load-Time:

Measures full webpage load time using each resolver's answer, proving how DNS affects real user experience and page performance.

Resolver	Domain	DNS(ms)	Connect(ms)	SSL(ms)	TTFB(ms)	Total(ms)	
LOCAL	www.google.com	1	23.75	80.30	437.77	484.63	
GOOGLE	www.google.com	24	23.63	95.83	453.30	500.54	
CLOUDFLARE	www.google.com	1	44.32	104.69	526.02	579.98	
LOCAL	www.facebook.com	0	10.22	60.10	460.19	460.30	
GOOGLE	www.facebook.com	24	43.79	96.17	343.15	343.25	
CLOUDFLARE	www.facebook.com	1	10.57	57.27	365.72	365.88	
LOCAL	www.tiktok.com	1	9.33	68.57	186.72	186.95	
GOOGLE	www.tiktok.com	48	9.23	77.63	176.56	176.74	
CLOUDFLARE	www.tiktok.com	1	9.27	65.25	150.43	150.53	
LOCAL	www.youtube.com	0	23.79	76.98	227.09	345.40	
GOOGLE	www.youtube.com	45	23.68	81.32	223.57	342.31	
CLOUDFLARE	www.youtube.com	1	40.82	97.77	295.74	501.08	
LOCAL	rr2---sn-5qouxaxjvhauqcq-q5jl.googlevideo.com	1	0.52	58.36	58.98	59.08	
GOOGLE	rr2---sn-5qouxaxjvhauqcq-q5jl.googlevideo.com	42	0.52	50.72	51.44	51.54	
CLOUDFLARE	rr2---sn-5qouxaxjvhauqcq-q5jl.googlevideo.com	1	0.49	59.23	59.87	59.98	
LOCAL	v9e.tiktokcdn.com	0	0.58	62.30	64.10	64.17	
GOOGLE	v9e.tiktokcdn.com	44	0.58	51.90	52.98	53.04	
CLOUDFLARE	v9e.tiktokcdn.com	1	0.56	51.19	54.50	54.56	

Why This Matters

Even if caches exist locally, resolver location decides which node is picked

Our network carries CDN traffic, but we don’t know which CDN answered the user.

No data to discuss with CDN vendors

Lost Telemetry

Objective

Build internal DNS telemetry without impacting production

Identify CDN source per query (Own, Vendor, Global)

Visualize cost impact using open-source stack (make an effort to build something free of cost and get some result)

Lab Architecture

Diagram:

PowerDNS → JSON log → Python loader → ClickHouse → Grafana

PowerDNS (collector)

Python (streamer)

ClickHouse (storage)

Grafana (visuals)

Why PowerDNS (not BIND)

Feature	BIND	PowerDNS
JSON or Lua logging	No	Yes
Extensible hooks	No	Yes
Lightweight resolver	Heavier	Faster
Live scripting	None	Lua supported

How Data Flows

PowerDNS writes JSON per query

Python tails /var/log/pdns_queries.json

ClickHouse stores into dns_queries

Classification joins IP map → categories (Udoy/Vendor/Global)

/etc/powerdns/log_answers.lua

```
local outfile = "/var/log/pdns_queries.json"
function postresolve(dq)
  local qname = dq.qname:toString()

  -- Only log Facebook/GGC domains
  if not string.find(qname, "fna.fbcdn.net", 1, true)
     and not string.find(qname, "googlevideo.com", 1, true) then
    return false
  end
  local answers = {}
  local records = dq:getRecords()
  if records then
    for _, r in ipairs(records) do
      local content = r.content or r:getContent() or tostring(r)
      if content and content ~= "" then
        -- Only capture IPv4 or IPv6 literals
        if string.match(content, "^%d+%.%d+%.%d+%.%d+$") or
           string.match(content, "^[%x:]+$") then
          table.insert(answers, content)
        end
      end
    end
  end

  -- Skip if there are no valid IP answers
  if #answers == 0 then return false end

  local f = io.open(outfile, "a")
  if f then
    f:write(string.format(
      '{"ts":"%s","client_ip":"%s","qname":"%s","answers":["%s"]}\n',
      os.date("!%Y-%m-%dT%H:%M:%SZ"),
      dq.remoteaddr:toString(),
      qname,
      table.concat(answers, '","')
    ))
    f:close()
  end
  return false
end
```

You can add new domain's here

Example Log Line

{"ts":"2025-10-20T10:07:13Z","client_ip":"103.190.x.x","qname":"googlevideo.com.","answers":["103.234.201.212"]}

{"ts":"2025-10-20T16:00:26Z","client_ip":"103.190.204.206","qname":"rr4---sn-npoe7nsr.googlevideo.com.","answers":["142.251.88.41"]}

{"ts":"2025-10-20T16:00:28Z","client_ip":"103.234.202.0","qname":"scontent.fdoh6-2.fna.fbcdn.net.","answers":["80.76.170.81"]}

{"ts":"2025-10-20T16:00:35Z","client_ip":"103.190.204.205","qname":"rr1.sn-5qouxaxjvhauqcq-q5jl.googlevideo.com.","answers":["2400:9220:0:32::c"]}

{"ts":"2025-10-20T16:00:44Z","client_ip":"103.234.202.1","qname":"rr1---sn-hxb5j5cax-nafe.googlevideo.com.","answers":["187.244.33.76"]}

{"ts":"2025-10-20T16:00:44Z","client_ip":"103.234.202.1","qname":"r10---sn-x5guiuxaxjvh-q5je7.googlevideo.com.","answers":["59.152.106.24"]}

{"ts":"2025-10-20T16:00:44Z","client_ip":"103.234.202.1","qname":"r10---sn-x5guiuxaxjvh-q5je7.googlevideo.com.","answers":["59.152.106.24"]}

{"ts":"2025-10-20T16:00:46Z","client_ip":"103.234.202.1","qname":"rr7---sn-x5guiuxaxjvh-q5jd.googlevideo.com.","answers":["103.21.42.210"]}

{"ts":"2025-10-20T16:01:00Z","client_ip":"103.234.202.0","qname":"r3---sn-qxaeenld.googlevideo.com.","answers":["172.217.139.104"]}

{"ts":"2025-10-20T16:01:02Z","client_ip":"103.234.202.1","qname":"z-p3-scontent.frjh7-1.fna.fbcdn.net.","answers":["37.111.248.33"]}

{"ts":"2025-10-20T16:01:02Z","client_ip":"103.234.202.1","qname":"z-p3-scontent.frjh7-2.fna.fbcdn.net.","answers":["37.111.248.96"]}

/opt/pdns_to_clickhouse.py

```
#!/usr/bin/env python3
import json
import subprocess
import time
from clickhouse_driver import Client
from datetime import datetime


def setup_clickhouse():
    return Client(host='localhost')


def process_log_line(client, line):
    try:
        data = json.loads(line.strip())
        answers = [a for a in data.get('answers', []) if a and a != '']

        if not answers:
            # Skip blank answer entries
            return False
        ts_str = data['ts'].replace('Z', '')
        ts_dt = datetime.strptime(ts_str, '%Y-%m-%dT%H:%M:%S')


        client.execute(
            "INSERT INTO dnslab.dns_queries (ts, client_ip, qname, answers) VALUES",
            [{
                'ts': ts_dt,
                'client_ip': data['client_ip'],
                'qname': data['qname'],
                'answers': answers
            }]
        )
        print(f"Inserted: {data['qname']} -> {answers}")
        return True

    except Exception as e:
        print(f"Error processing line: {e}")
        return False
def main():
    print("Starting PowerDNS → ClickHouse loader (dnslab.dns_queries)")
    client = setup_clickhouse()
    process = subprocess.Popen(
        ['tail', '-F', '/var/log/pdns_queries.json'],
        stdout=subprocess.PIPE,
        stderr=subprocess.PIPE,
        text=True
    )
    while True:
        line = process.stdout.readline()
        if line:
            process_log_line(client, line)
        else:
            time.sleep(0.1)

if __name__ == "__main__":
    main()
```

Schema Overview

- Tables:
  - dns_queries (raw)
  - cdn_ip_map (reference list)
  - dns_classified (derived)
- Each line of data = 1 resolved query with IP classification.

dns_queries (raw)—How it looks like

ts	client_ip	qname	answers	category
2025-11-12 00:27:55	103.190.204.204	v16-cla.tiktokcdn.com.	['23.212.164.49','23.212.164.24']	TikTok
2025-11-12 00:27:55	103.190.204.204	v16-cla.tiktokcdn.com.	['23.212.164.49','23.212.164.24']	TikTok
2025-11-12 00:27:54	103.234.202.2	scontent.fdac34-2.fna.fbcdn.net.	['42.0.5.83']	Facebook (FNA)
2025-11-12 00:27:54	103.234.202.2	scontent.fdac34-2.fna.fbcdn.net.	['42.0.5.83']	Facebook (FNA)
2025-11-12 00:27:53	103.190.204.207	z-p3-video.fjsr1-1.fna.fbcdn.net.	['123.108.245.33']	Facebook (FNA)
2025-11-12 00:27:53	103.190.204.207	z-p3-video.fjsr1-1.fna.fbcdn.net.	['123.108.245.33']	Facebook (FNA)
2025-11-12 00:27:50	103.190.204.207	rr4---sn-x5guiuxaxjvh-q5j6.googlevideo.com.	['45.113.133.79']	Google (GGC)
2025-11-12 00:27:50	103.190.204.207	rr4---sn-x5guiuxaxjvh-q5j6.googlevideo.com.	['45.113.133.79']	Google (GGC)
2025-11-12 00:27:48	103.190.204.205	pull-flv-f1-sg01.tiktokcdn.com.	['103.15.41.101','103.111.13.160']	TikTok

Logs are now exported to clickhouse databases. Which shows client_ip, qname and answers [IPv4, IPv6] of the queries

cdn_ip_map (reference list)

This tables has all the IP address of Own (udoy) & Vendor CDNs. If any ip that doesn’t matches these IP address are marked at Global.

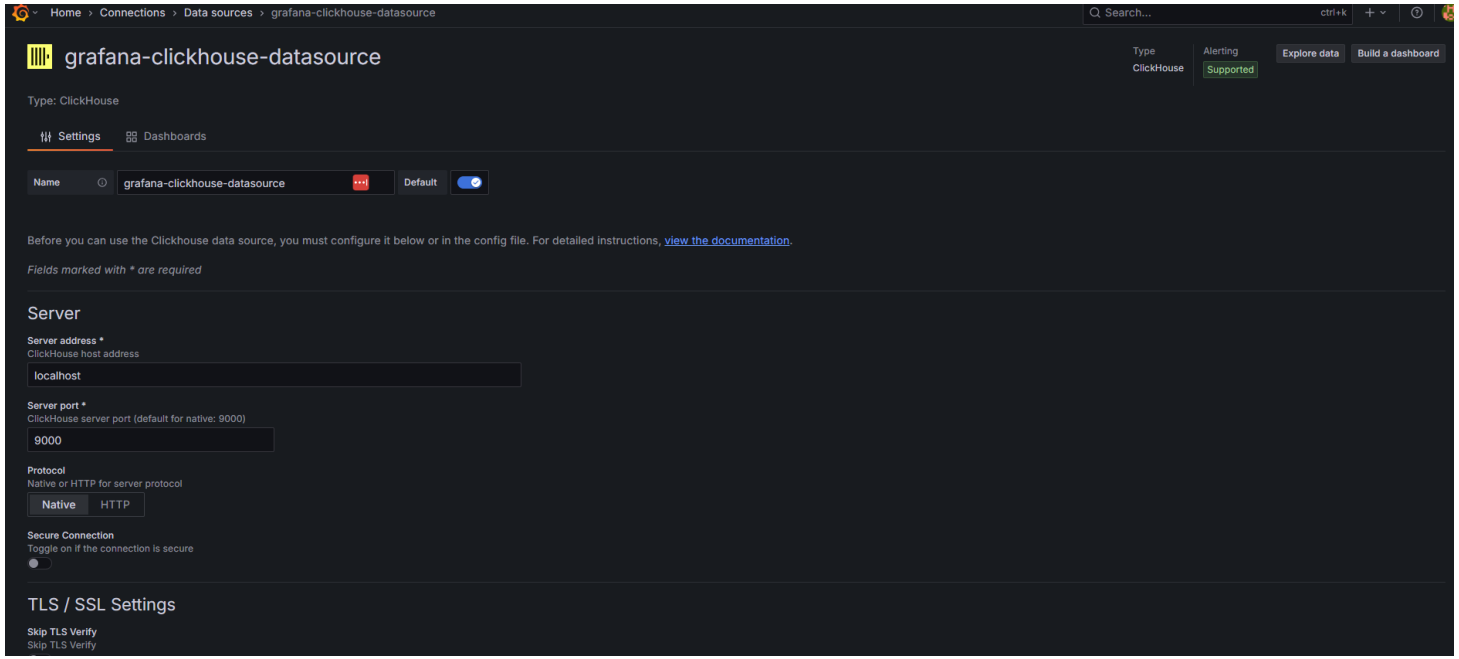
ip	label
103.234.201.193	udoy
103.234.201.194	udoy
103.234.201.195	udoy
103.234.201.196	udoy
103.234.201.197	udoy
103.234.201.198	udoy
103.234.201.199	udoy
103.234.201.200	udoy
103.234.201.201	udoy
103.234.201.202	udoy
103.105.87.1	vendor
103.105.87.10	vendor
103.105.87.100	vendor
103.105.87.101	vendor
103.105.87.102	vendor
103.105.87.103	vendor
103.105.87.104	vendor
103.105.87.105	vendor
103.105.87.106	vendor
103.105.87.107	vendor

dns_classified (derived)

ts	client_ip	qname	answers	categories	cdn_category
2025-11-12 00:33:57	103.190.204.206	pull-cmaf-f77-va01.tiktokcdn.com.	['79.127.213.229','79.127.213.226']	['global','global']	TikTok
2025-11-12 00:33:57	103.190.204.206	pull-cmaf-f77-va01.tiktokcdn.com.	['79.127.213.229','79.127.213.226']	['global','global']	TikTok
2025-11-12 00:33:55	103.234.202.2	v9e.tiktokcdn.com.	['154.85.74.168','154.85.74.169']	['udoy','udoy']	TikTok
2025-11-12 00:33:55	103.234.202.2	v9e.tiktokcdn.com.	['154.85.74.168','154.85.74.169']	['udoy','udoy']	TikTok
2025-11-12 00:33:54	103.190.204.205	redirector.googlevideo.com.	['142.251.222.142']	['global']	Google (GGC)
2025-11-12 00:33:54	103.190.204.205	redirector.googlevideo.com.	['142.251.222.142']	['global']	Google (GGC)
2025-11-12 00:33:50	103.190.204.206	pull-flv-l1-va01.tiktokcdn.com.	['103.111.13.160','103.15.41.101']	['global','global']	TikTok
2025-11-12 00:33:50	103.190.204.206	pull-flv-l1-va01.tiktokcdn.com.	['103.111.13.160','103.15.41.101']	['global','global']	TikTok
2025-11-12 00:33:50	103.190.204.207	v19-cla.tiktokcdn.com.	['151.101.38.73']	['global']	TikTok

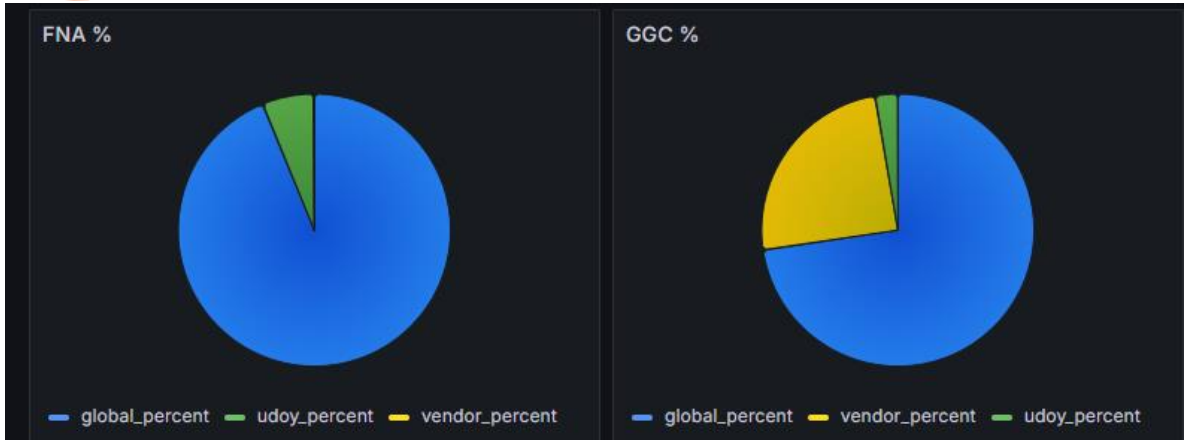
This is what the tables looks like after consolidating both query data & ip mapping data. From here we can see and store the logs.

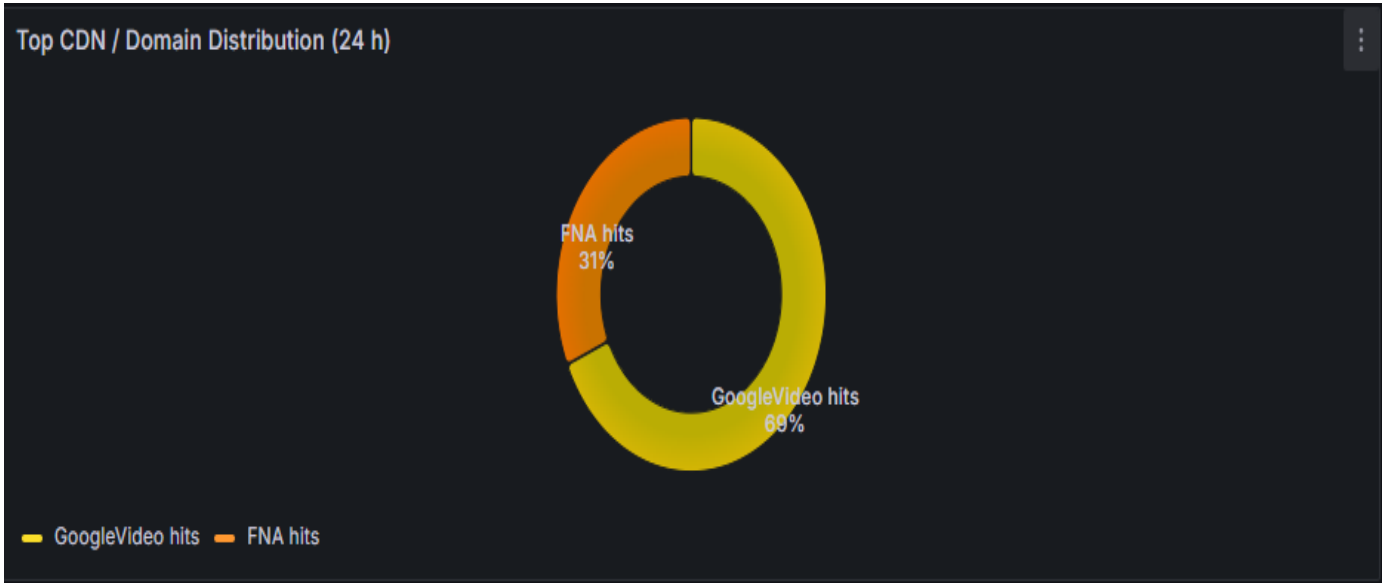
Grafana Datasource setup



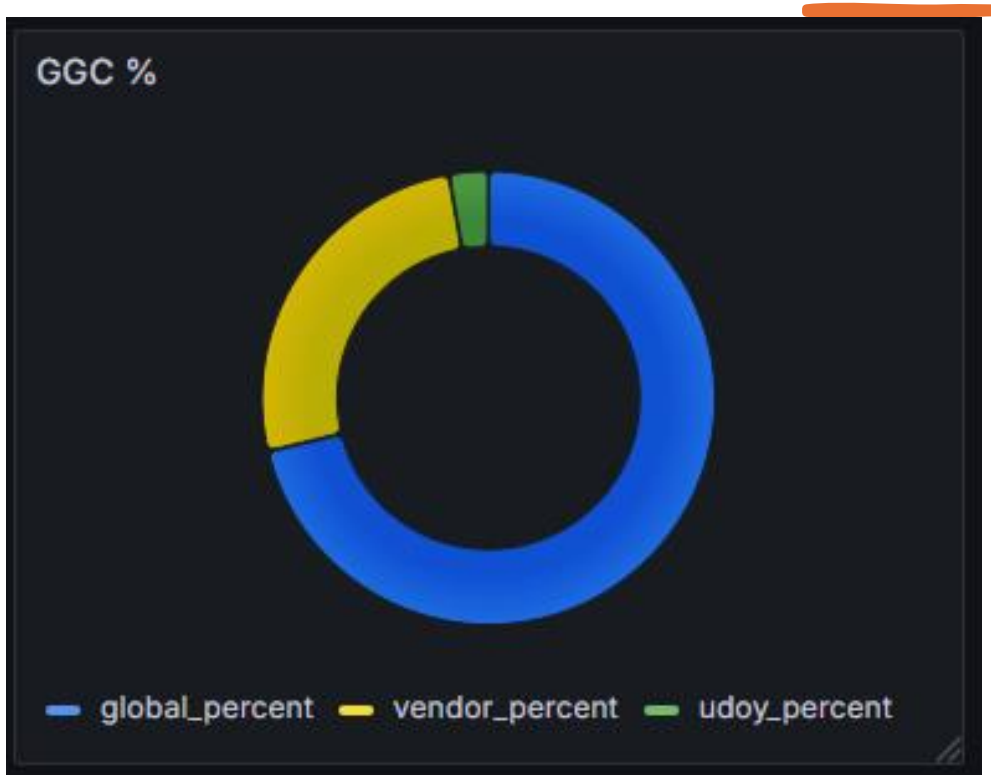
Once we have data in clickhouse tables. We can visualize it using Grafana.

Grafana Dashboard





GGC (Googlevideo) Analysis



Top GGC (Googlevideo) Answer IPs – Last 24 Hours

answer_ip	total_hits	percent_share	category
142.251.222.142	539	1.18	global
142.251.43.46	466	1.02	global
142.251.220.110	398	0.870	global
103.234.201.45	337	0.740	udoy
103.234.201.44	321	0.700	udoy
2404:6800:4007:834::2	310	0.680	global
2400:9220:0:32::d	256	0.560	global
2400:9220:0:32::c	251	0.550	global

‹ 1 … 3 ›

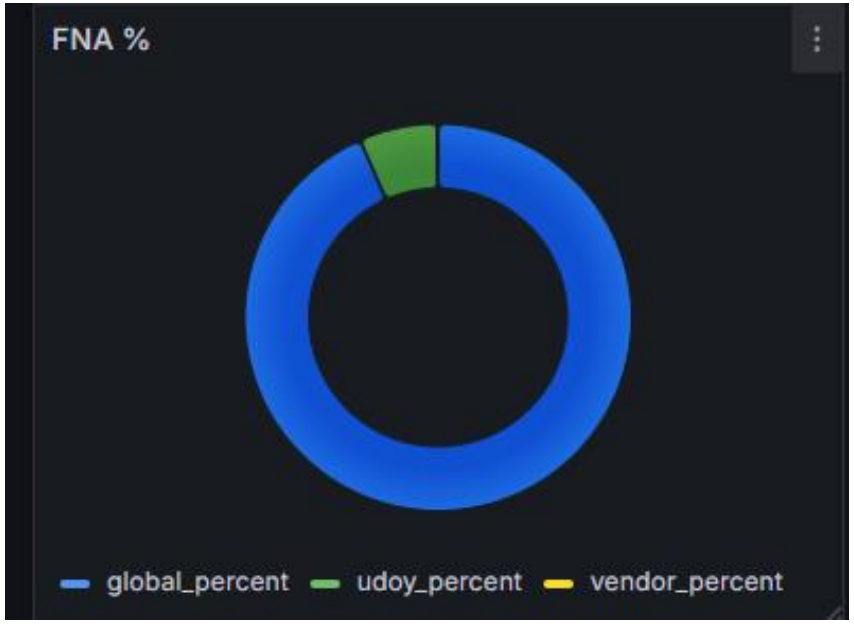
This help us identify Top googlevideo url resolved IP address

FNA (Facebook) Analysis

Top FNA (Facebook) Answer IPs – Last 24 Hours

answer_ip	total_hits	percent_share	category
103.234.201.209	275	1.62	udoy
103.234.201.227	275	1.62	udoy
103.234.201.223	202	1.19	udoy
2400:9220:0:30:face:b0	116	0.680	global
103.230.105.160	111	0.660	global
37.111.248.33	107	0.630	global
123.108.245.99	104	0.610	global
37.111.248.96	103	0.610	global

‹ 1 ··· 3 ›



This help us identify Top facebook video url resolved IP address

Lessons for ISPs

- Run your own resolver
- Tag CDN answers (Own/Vendor/Global)
- Measure before optimizing
- Use DNS data for routing + CDN negotiation

Future Work

- Add ASN and BGP tie-in
- Merge with NetFlow for QoE tracking

GitHub link

https://github.com/sufian-bd/apricot-2026

Thank You

DNS logs tell you where your traffic really lives