

The State of Networking & the Need for Modern Network Visibility

Greg Villain



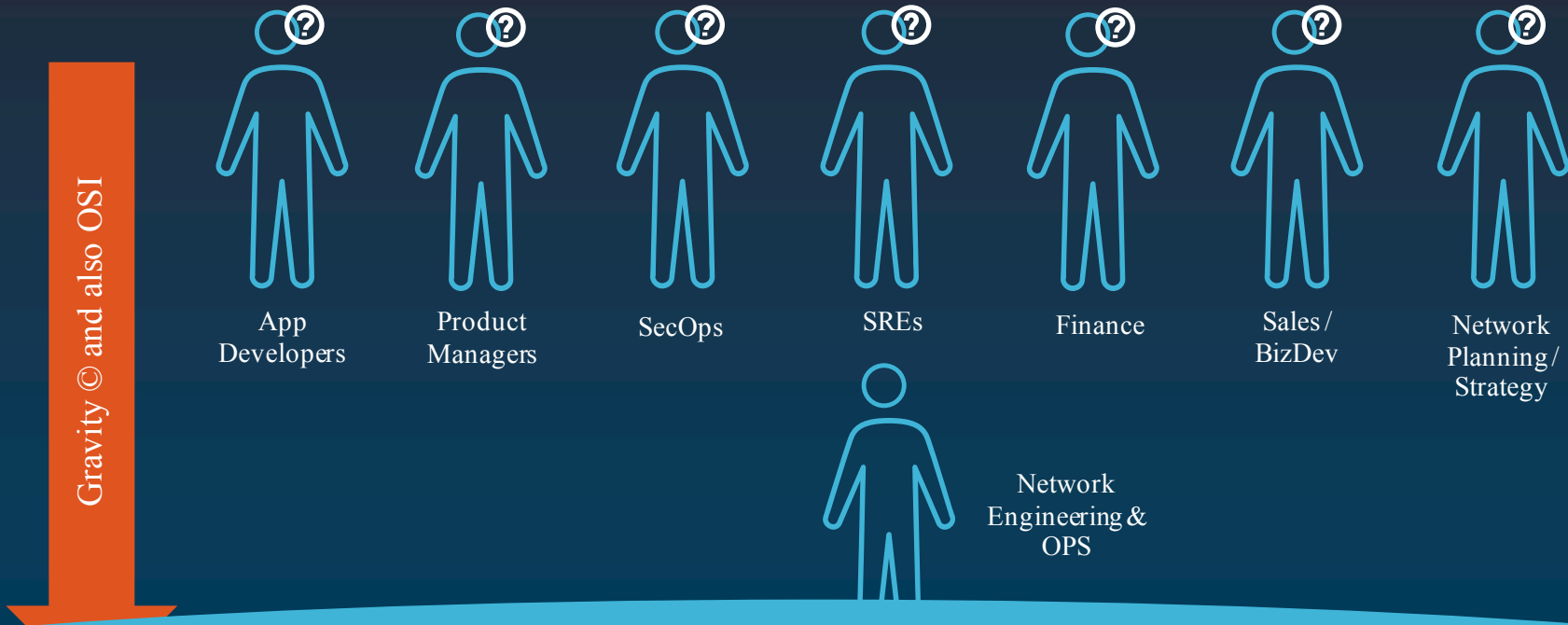
Easily, the world's most powerful network analytics.



Field notes.
Resulting requirements.
Design principles.



The network is the foundation of your value stack



NETWORK: Serves the business
Sees the business

Requirements for a modern flow platform?

- Flows can contain information that the **rest of the company** can/want to/should use
 - Flows should be **enriched with business logic** suitable for non-engineering user profiles
 - UI needs to drive content **producer/consumer** dynamics
- **Network engineers shouldn't be on the critical path** of other employees wanting/Needing that data
 - If others are going to leverage flow data, it better come with **a UI that works** for everyone.
 - **Flexible** dashboarding
 - Non NetEng users need **understandable viz** + canned **task-specific views**

Siloed tools, siloed teams

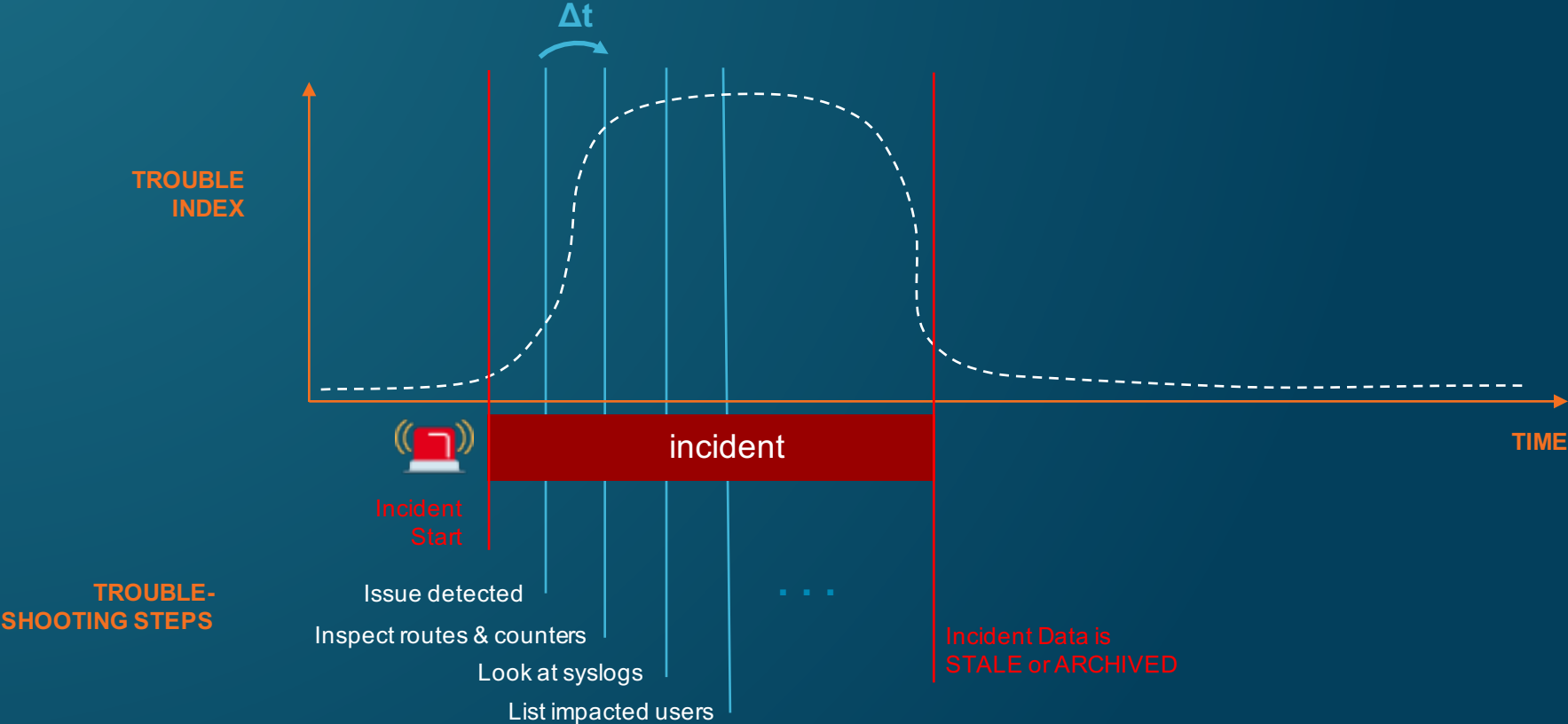
PLENTY OF TOOLS

- Flow collection subsystem
- SNMP collection subsystem
- Routing Tables
- On-Router interface counters
- Network Monitoring System
- NC Versioning
- Time Series / Metrics DB
- Incident management systems
- SIEMs
- Syslogs
- IM
- (Network) CRMs / Registries
- IPAMs, DNS Zone files

OWNED BY PLENTY OF TEAMS

- Network Operations
- Network Architects
- Software Engineers
- SREs
- Security Engineers
- (even sales engineers)
- NOC / Customer Support

Classical incident timeline

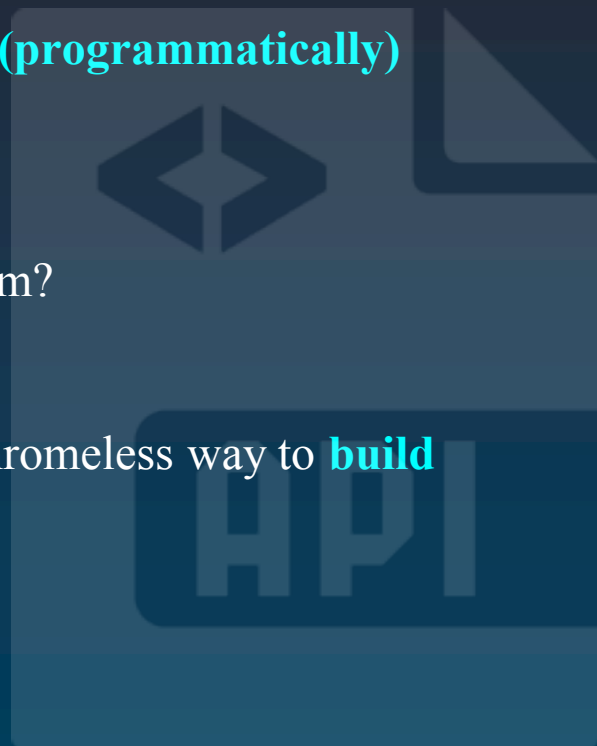


Requirements for a modern flow platform?

- Troubleshooting requires **fast data path IN**
 - Near real time ingest-to-query
- Troubleshooting requires **fast data path OUT**
 - <10s query responses
- We want **full-resolution flows**: aggregation dilutes precious information
 - We also want **history** under the same conditions
- Flow information needs to be **drill-down and drill-out friendly**
 - Ad-hoc explorations need to be fast and uninterrupted
- More importantly, outlines the need for an **automated Anomaly Detection engine** that leverages all of the above.

Automation & downstream usage of flow data

- If your data is useful to other teams **make it (programmatically) available**
- **Automate all the things**
 - provisioning devices in the flow platform?
 - Zero touch provisioning
- Let competent users leverage the data in a chromeless way to **build the features you don't offer yet**
- out-of-the-box **integrations**:
ticketing systems, chatops, custom triggers
- **Power real-world SDN** using Network Data as a signal



Requirements for a modern flow platform?

- **Data needs to be portable**
 - Programmatic formats
 - Formats that non-developers understand: pdf reports, CSV/XLS, visualization images
- **API-first** design principle
 - *“If the UI shows it, then you can CURL it”*

Scale gets in the way

Back of the envelope

Small Network

10 routers,
10Gbps peak traffic,
5,000 fps

Medium Network

50/100 routers,
100-500 Gbps peak traffic,
50-100k fps

Large Network

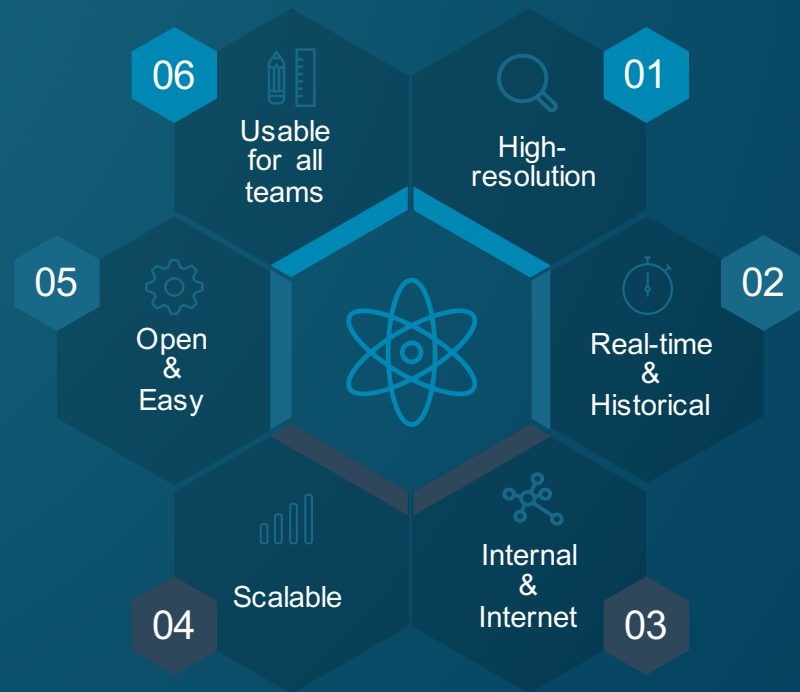
100+ routers
5-10 Tbps peak traffic
500k-1,000k fps

1 FPS = 1 row in store per second

- Query over 50 routers
 - Each router doing 1,000 fps
 - For a time-range of 1 week
- pull 30B rows / 15TB
→ reply in <10s

Ingest → **Enrich** → **Store** → **Query** functions all need to scale horizontally

In summary...



Game of (bad) tradeoffs

- Aggregate to solve
 - for scale
 - Constrained storage (appliance)



- Loss of resolution, Miss signal inside aggregate
 - Keep changing your aggregates to match what you want to see

- Deferred enrichment batch jobs
 - Dodge complexity



- Useful data lags behind, loss of immediate visibility

- Not designed like a BI tool
 - UI is an afterthought



- No drill-downs/out
- Each new view needs be developed, inflexible
- Only NetEng can use

- Flow store using vanilla tech (MySQL, Elasticsearch, Hadoop, Druid...)

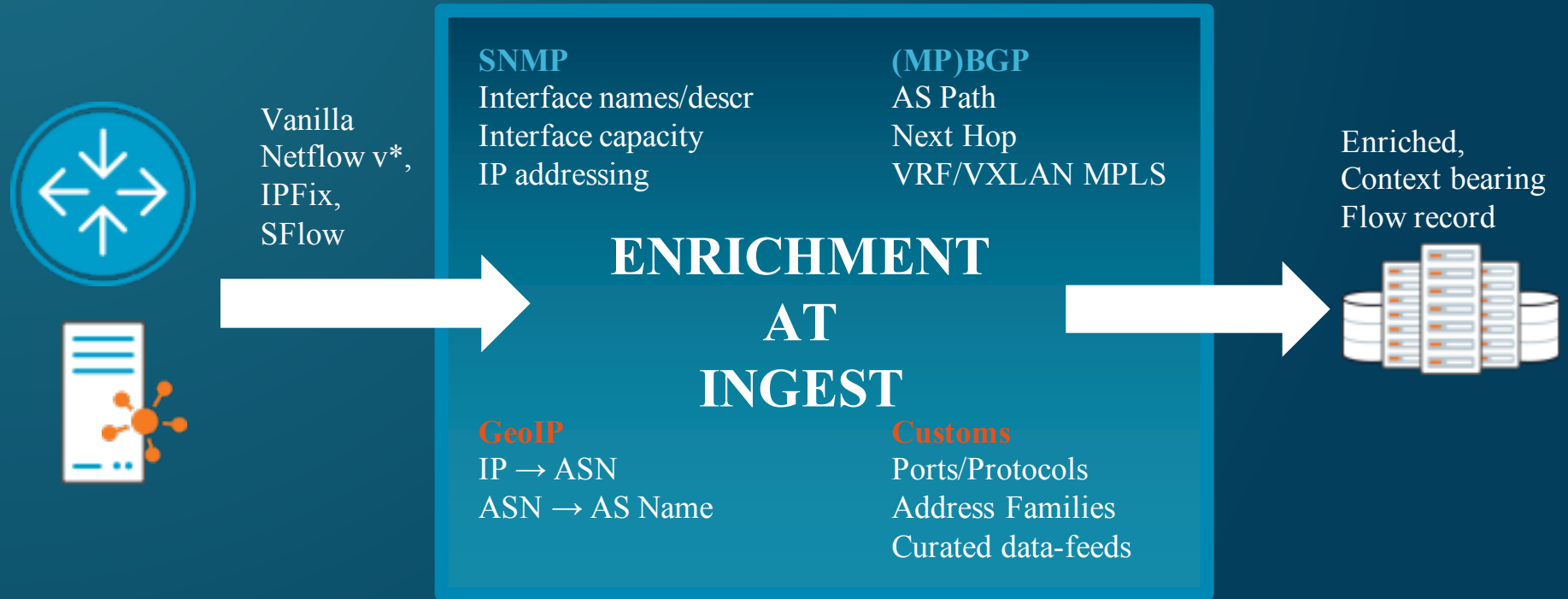


- Doesn't natively speak CIDR
- Fast but no history
- Full but Slow
- Sharding hell

Going beyond the basics.
Contextualizing flow-data.
aka **Enrichment.**



Making data useful: flow enrichment



Making data **super** useful: NEXT LEVEL CONTEXT

1st class citizen, UI supported config

Interface Classification

Inside/outside directionality

Connectivity type

Provider vs Customer

Network Classification

e2e directionality

Customer/Provider tagging

CRM meets flows

Custom AS Groups

Networks w/ multiple ASNs

Private ASNs

Custom Geo

Country groups/Markets

Sub country groups

Curated data feeds, auto-applied

Clouds

ISP Embedded + Self-hosted CDNs

Cloud providers

Threat feeds

Botnets

Infected hosts

Applications tagging

OTT services

Well known Apps

Full on BYOB

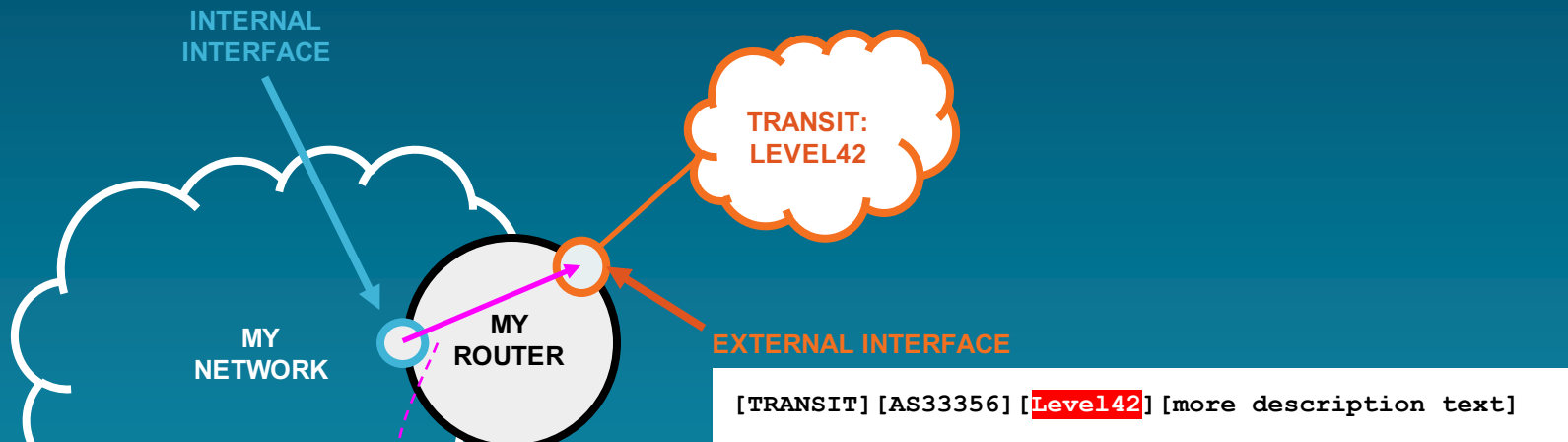
Rules matching flow attributes

Mark flows upon match

And remember...

- **Enrichment needs to happen at ingest time**
 - Enrichment mappings will sit in memory
 - ...where compute is costly
 - ...you will need to correlate w/ other live streams
- **Cardinality kills !**
 - Enrichment data-feeds can/should map **millions of IPs to large number of values**
 - **Don't drop at ingest, don't make queries slower**
- **Change frequency kills !**
 - When large #values mapped to IPs keep changing

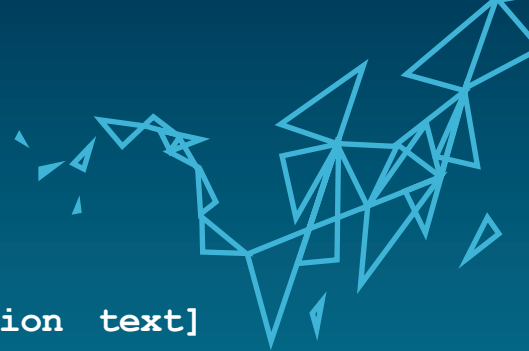
Useful enrichment: Interface Classification



Enriched flow record

src_int: {INTERNAL, BACKBONE, n/a}
dst_int: {EXTERNAL, TRANSIT, LEVEL42}

Useful enrichment: Interface Classification



INTERFACE DESCRIPTION (SNMP)

```
[TRANSIT] [AS33356] [Level42] [more description text]
```

DESCRIPTION MATCH REGEX (Enrichment engine)

```
^\[TRANSIT\]\[.*\]\[(.*)\].*$
```

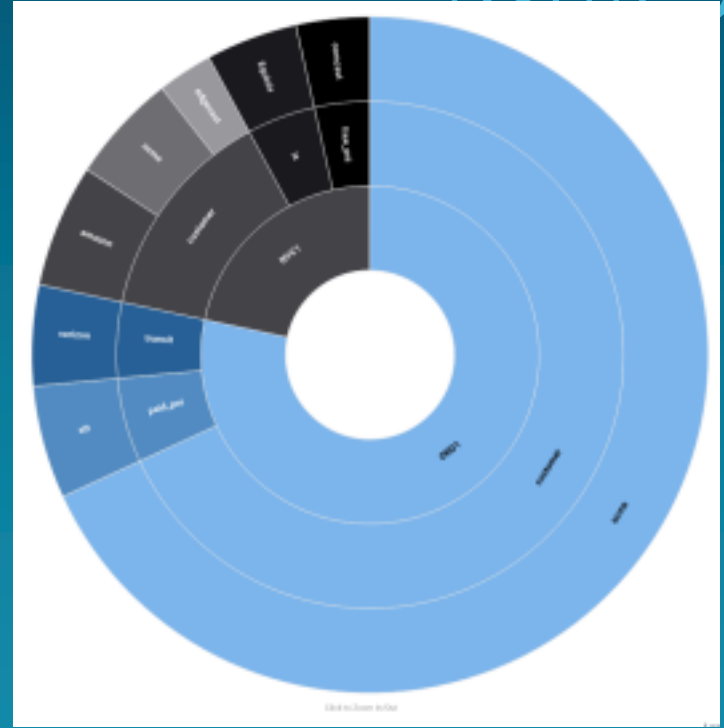
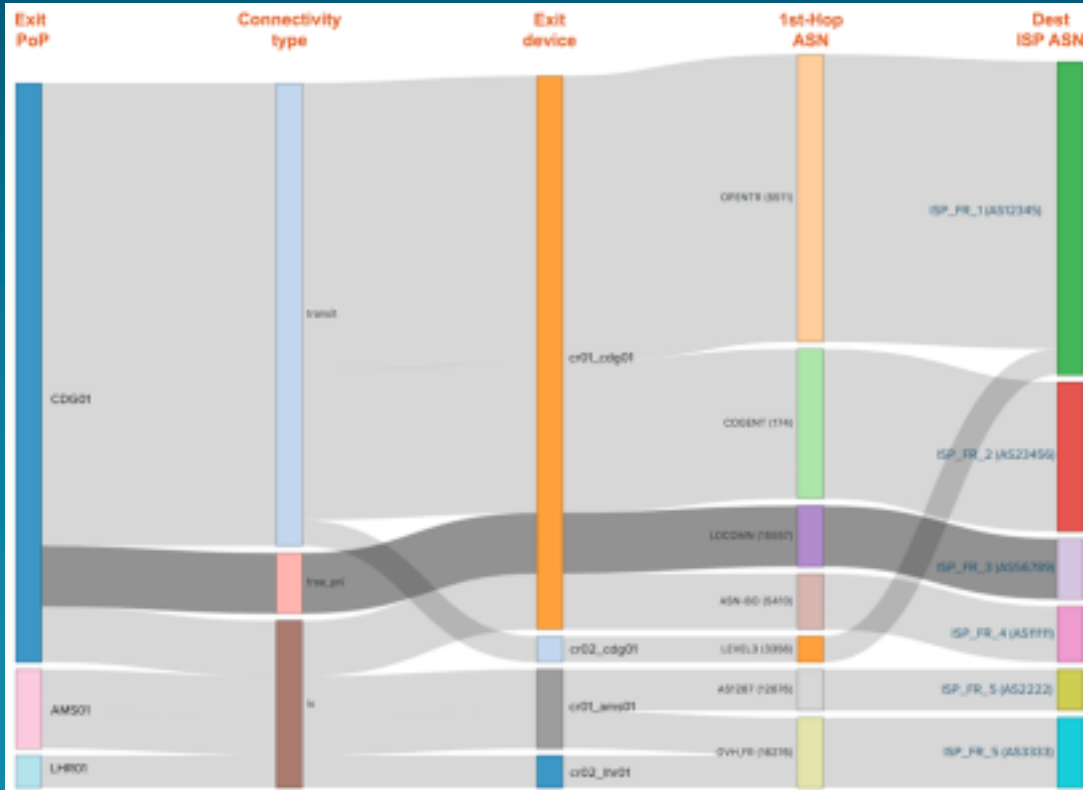
INTERFACE CLASSIFIERS

```
SET INTERFACE NETWORK BOUNDARY: EXTERNAL  
SET INTERFACE CONNECTIVITY TYPE: TRANSIT  
SET INTERFACE PROVIDER: $1  
(LEVEL42)
```

Enriched flow record

```
src_int: {INTERNAL, BACKBONE, n/a}  
dst_int: {EXTERNAL, TRANSIT, LEVEL42}
```

Useful enrichment: Interface Classification



Next:

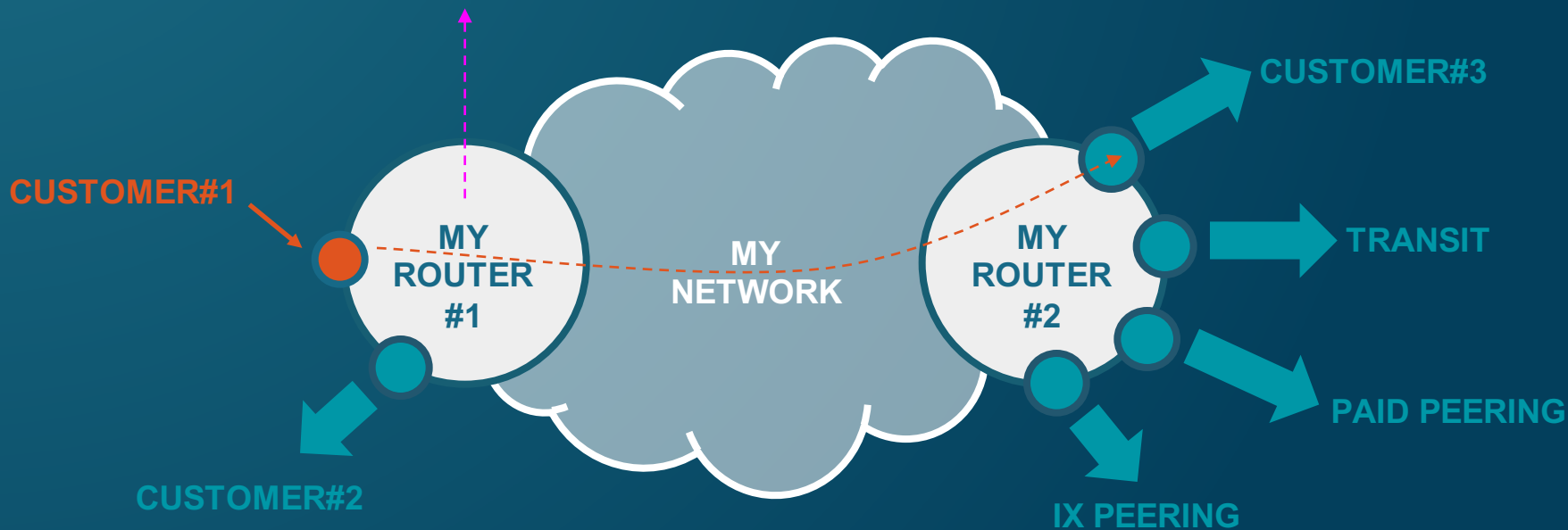
Future-Proof Network Visibility.



Ultimate Exit Discrimination

FLOW RECORD:

- Ultimate Exit {country, site, device, interface,}: {country,site, MyRouter#2, customer#3}
- Ultimate Exit Connectivity Type: customer
- Ultimate Exit Connectivity Provider/Customer: CUSTOMER#3

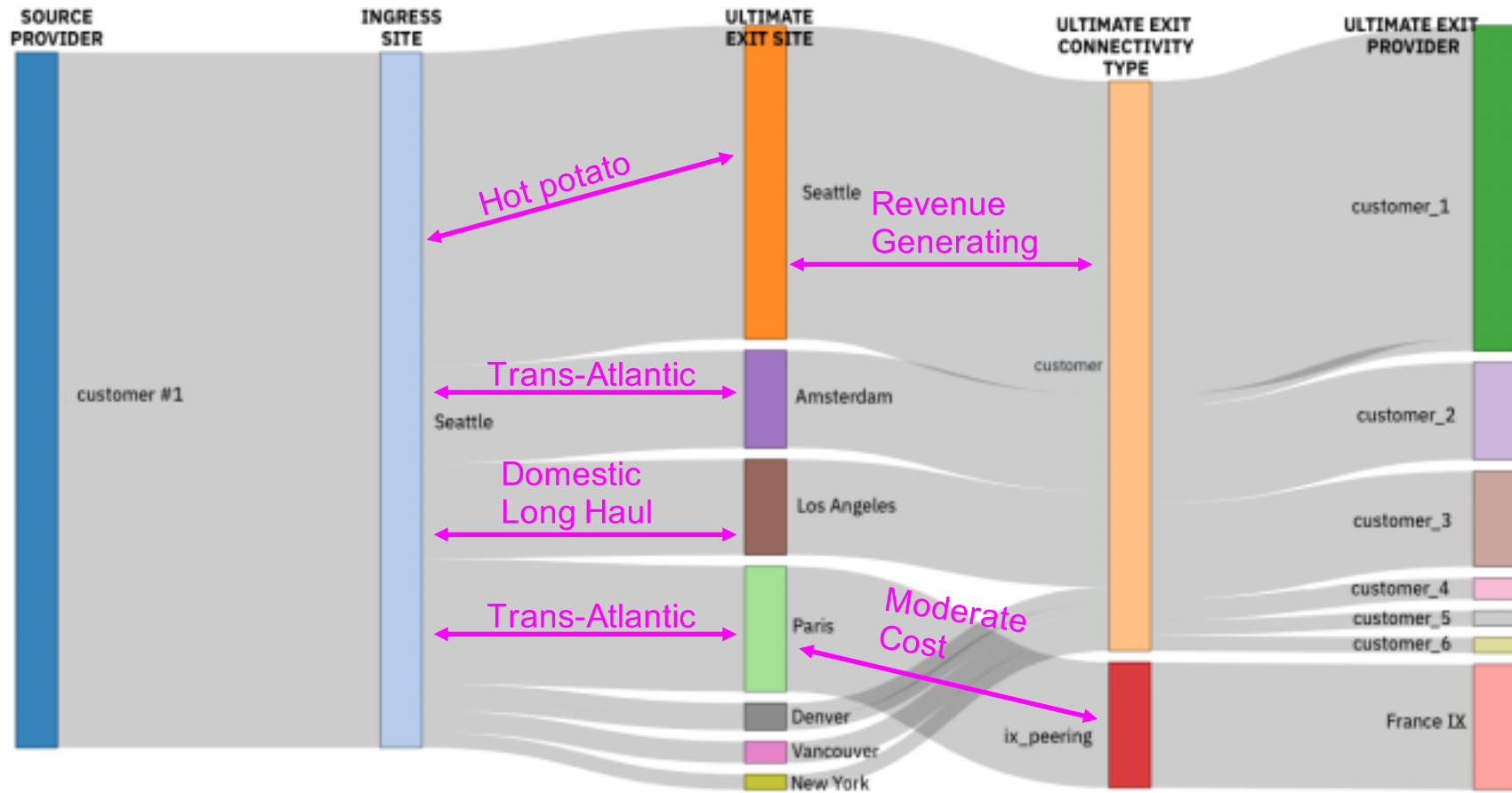


Ultimate Exit Discrimination

- **Hard, because at ingest** - you can't take flow info from the egress and attach it to the ingress ! (time travel...)
- **Ultimate Exit + Interface Classification**
 - Foundation to any **Cost Modelling** activity
- **BizDev / Salesforce freed** from collecting/wrangling spreadsheet data and interrupting NetEng work.



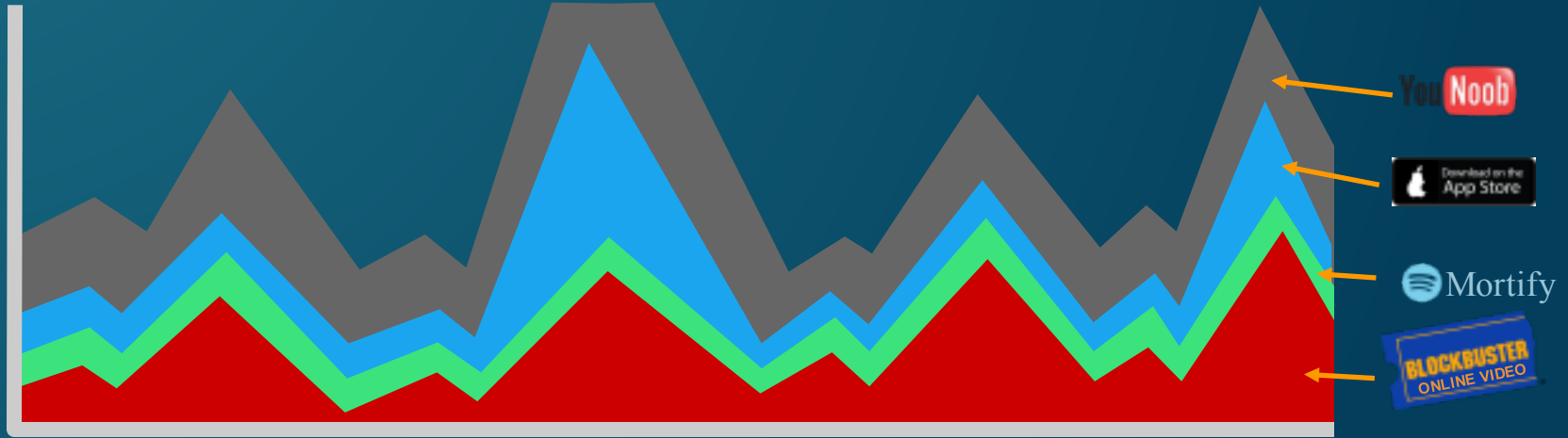
Ultimate Exit discrimination



Ultimate Exit discrimination

$$\text{\$COST} = \sum_{\text{connectivity}} (\text{\$Mbps} \times \text{Mbps}) + \sum_{\text{transport}} (\text{\$Mbps} \times \text{Mbps})$$

Over-the-Top (OTT) traffic enrichment

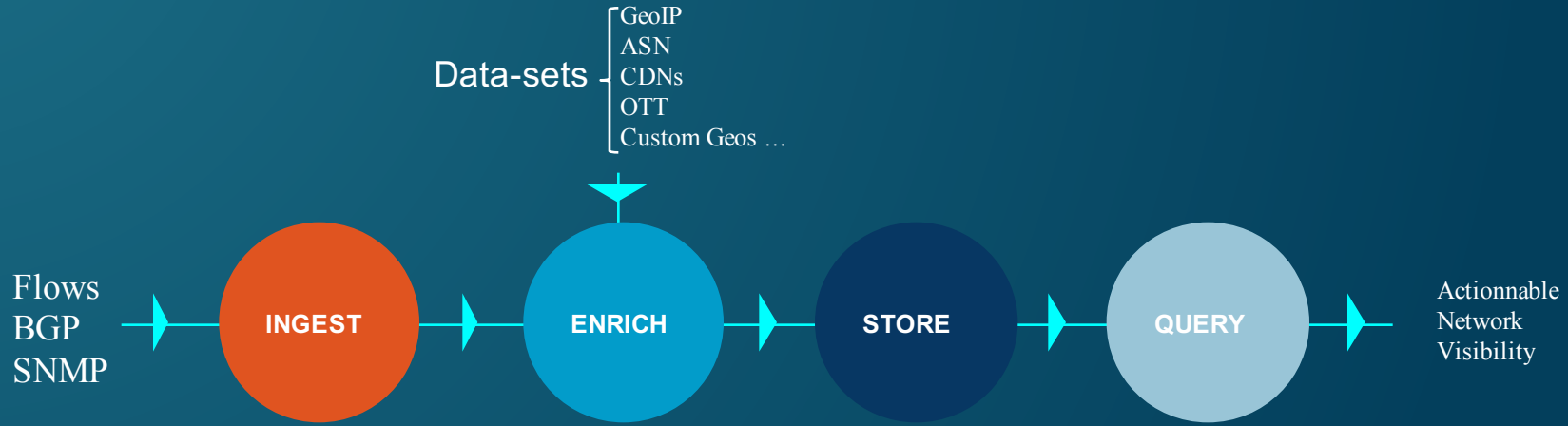


Over-the-Top (OTT) traffic enrichment

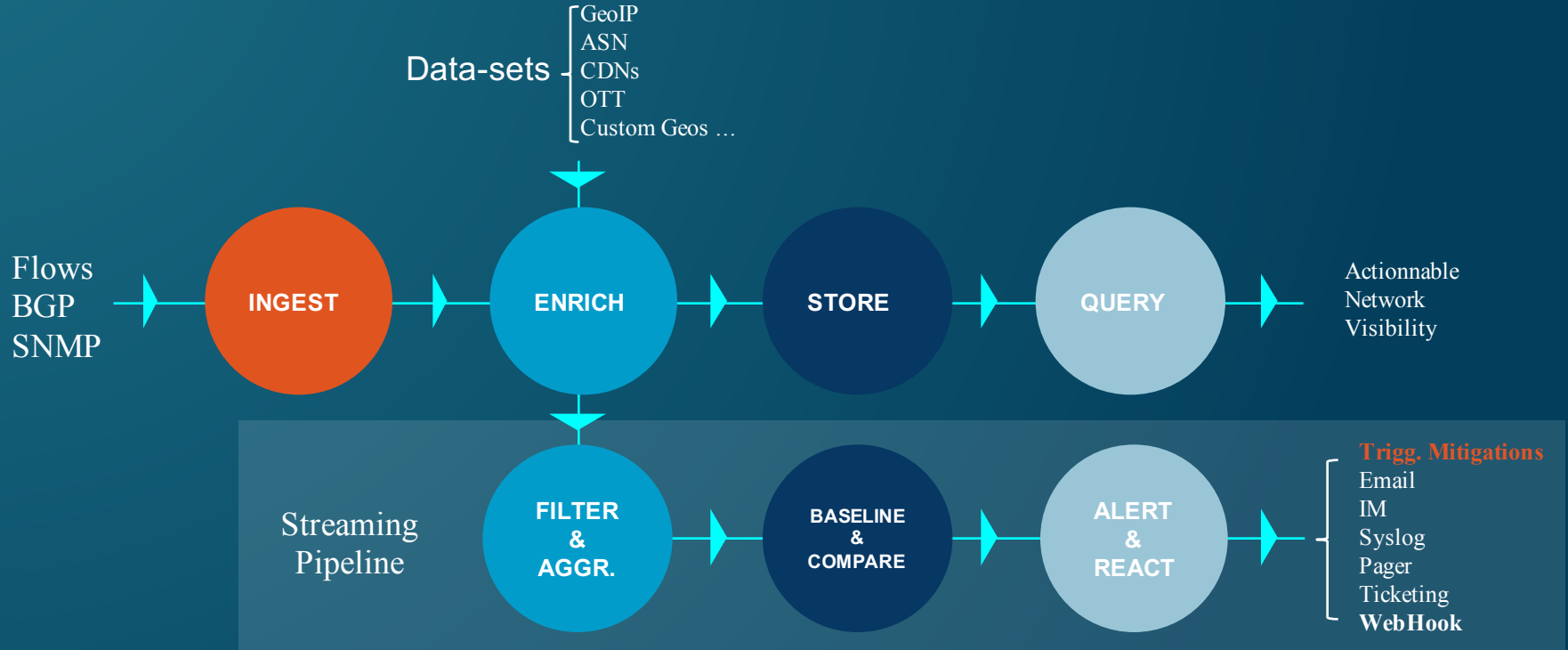
- **Hard, but feasible**
 - OTT providers rely on owned infrastructure and CDNs
 - Combine Flows + DNS query data + Curated host patterns
 - Still done near real time at ingest.
 - A high cardinality / frequency flow tagging backend is required
- **Business impact**
 - **Identify** traffic or cache embedding opportunities
 - Additional, **end-to-end end-user support tool**



Anomaly detection

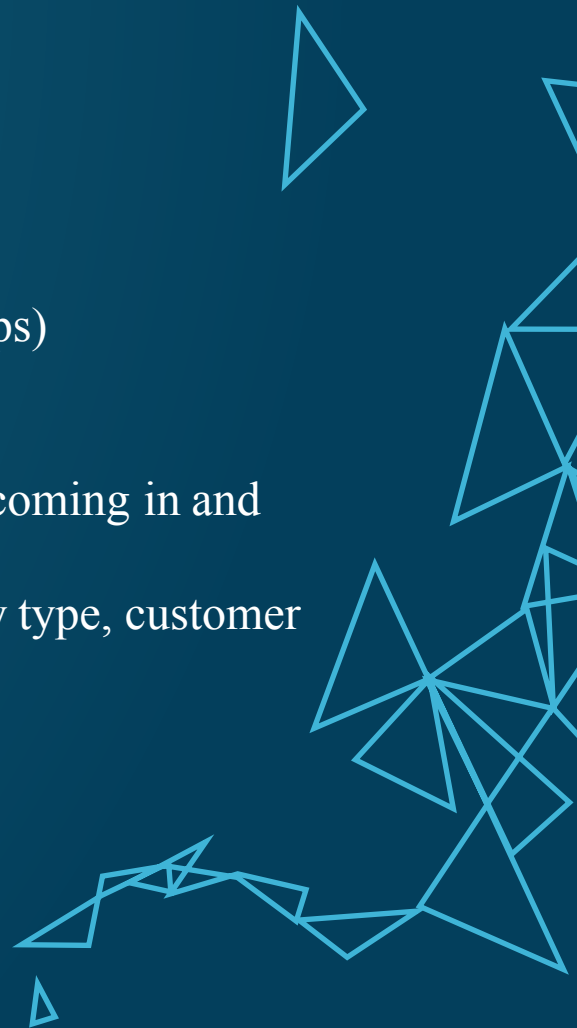


Anomaly detection



Anomaly detection

- **Leverage your enriched flow data**
 - Baseline flow derived metrics (bps, pps, #unique_ips)
- **Generic Anomaly Detection**
 - Not only DDoS: capacity, {Service, Geo, Subnet} coming in and out of TopN
 - Through the lens of this enriched data (connectivity type, customer identification, any custom dimension...)
- **Tee the ingested data** into a real-time system

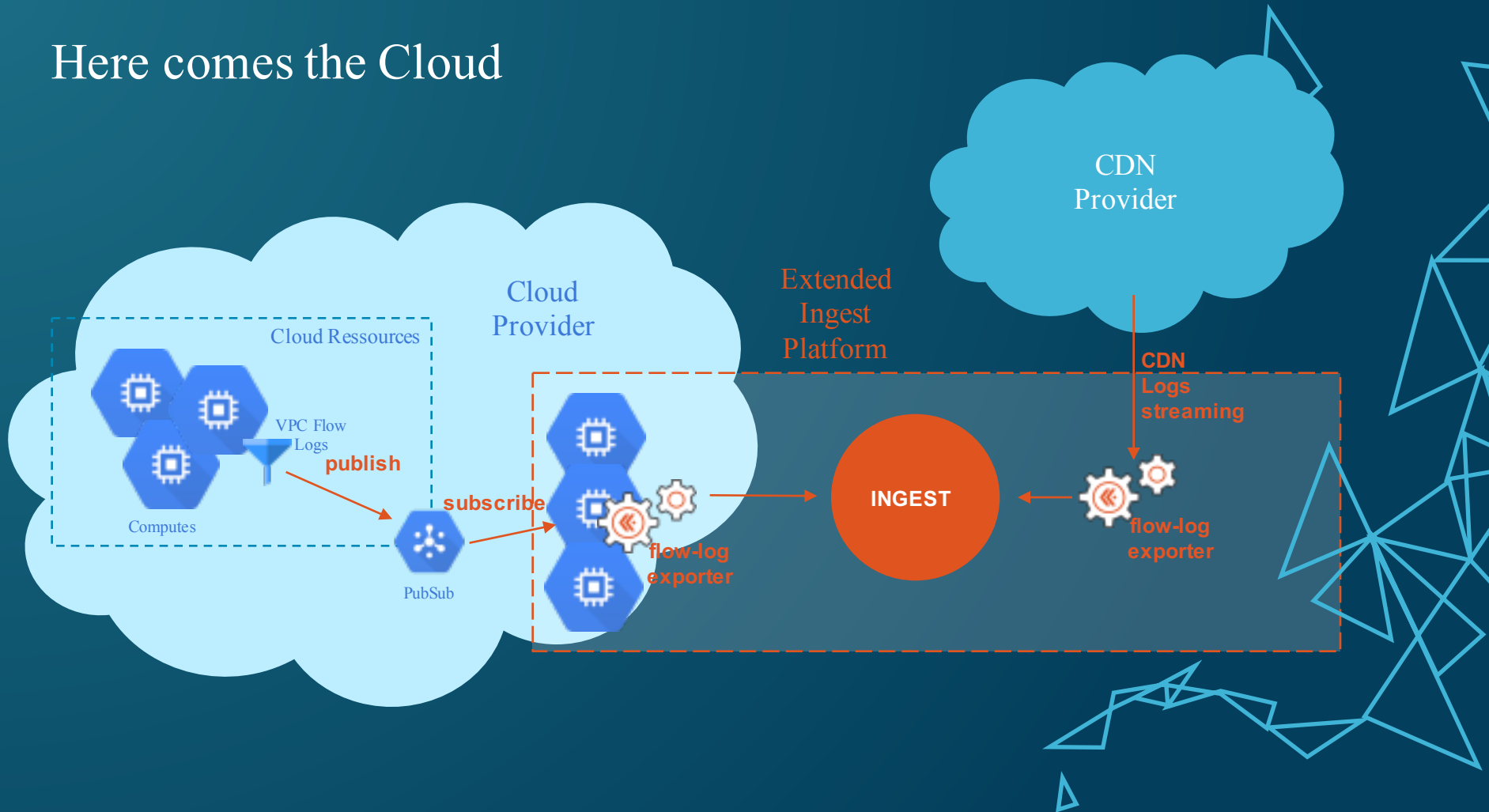


Here comes the Cloud

- **Hybridization of production environments**
 - Network Infrastructure
 - Native Apps
 - Building Apps in the cloud
 - Consuming Cloud Apps
 - Also using CDNs
- **New dependencies**
 - Internally hosted platforms break because of external components, vice-versa
- **Loss of global context**
 - No end to end visibility
 - Inability to baseline and track performance for migrating apps to the cloud
 - We're re-silo'ing visibility !!!



Here comes the Cloud



Once this is all built

- An increasing number of users rely on it
- Your Network Automation relies on it
- Tenants rely on it
- It drives some of your business
- Your monitoring/alerting depends on it

Congrats.

Your Modern Network Visibility Platform is now **business and ops critical.**



Should we do it ourselves?

Horizontal scalability
Distributed Enrichment Ingest
Custom Flow Datastore
Query & Tagging Engine
Streaming pipelines

Field experience of
Production Networks to fuel
the other groups involved

**Network
Engineers**

**Distributed Systems
Software
Engineers**

**Network Protocol
Developers**

Speak/Code *flow
Sampling
Templates
(MP)-BGP daemons
SNMP collectors
Programmable mitigations

Build a specialized BI tool
Make the UX composable
Enable producer/consumer users
Data-visualization

**UX Designers
Frontend
Software
Engineers**

**Site
Reliability
Engineers**

Support fast/iterative
Build
Deploy
Make all of the above
work reliably

Should we do it ourselves?



Field experience of
Production Networks to fuel
the other groups involved

**Network
Engineers**

**Distributed Systems
Software
Engineers**

**Network Protocol
Developers**

**UX Designers
Frontend
Software
Engineers**

**Site
Reliability
Engineers**

Horizontal scalability
Distributed Enrichment Ingest
Custom Flow Datastore
Query & Tagging Engine
Streaming pipelines

Speak/Code *flow
Sampling
Templates
(MP)-BGP daemons
SNMP collectors
Programmable mitigations

Support fast/iterative
Build
Deploy
Make all of the above
work reliably

Build a specialized BI tool
Make the UX composable
Enable producer/consumer users
Data-visualization



THANK YOU. QUESTIONS?

Greg Villain

greg@kentik.com

@kentakinc



Easily, the world's most powerful network analytics.

