Using Sources of Truth to Enrich and Understand Network Telemetry

Jac Kloots
Senior Solutions Engineer

Who am I?

Current

Senior Solutions Engineer - Kentik

Past

25 years in networking
Ran networks (including peering) before migrating to the vendor side

More details

/in/jackloots

Network observability

The difference between **more data** and **more answers**

Traditional monitoring lets you see **what's** happening on your network.

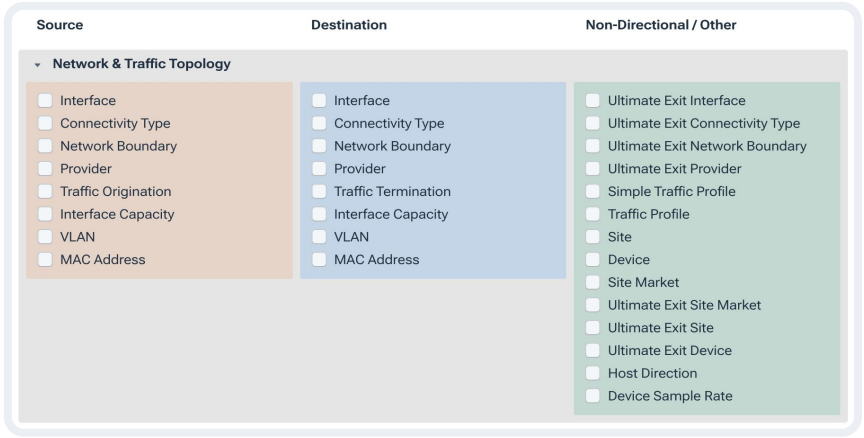
Network observability helps you understand **why** it's happening and automate a response.

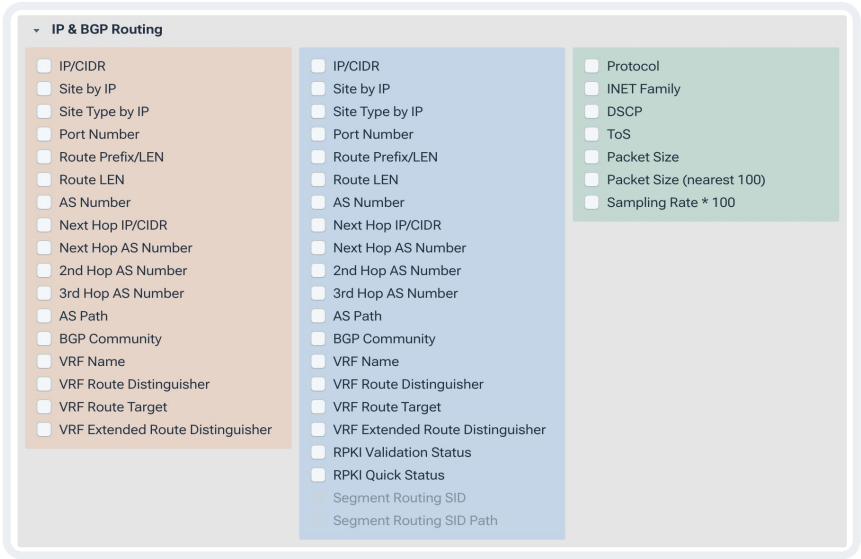
Network observability

What are the building blocks?

Context is needed for network observability

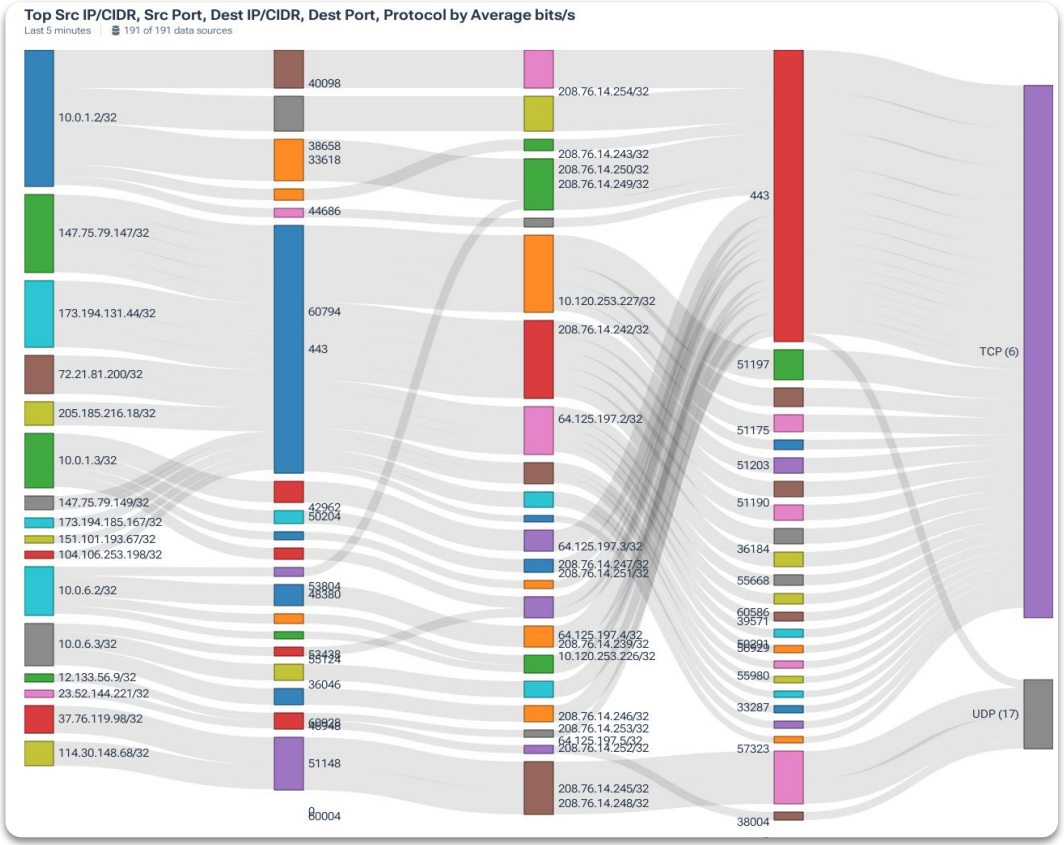
NetFlow is a good step



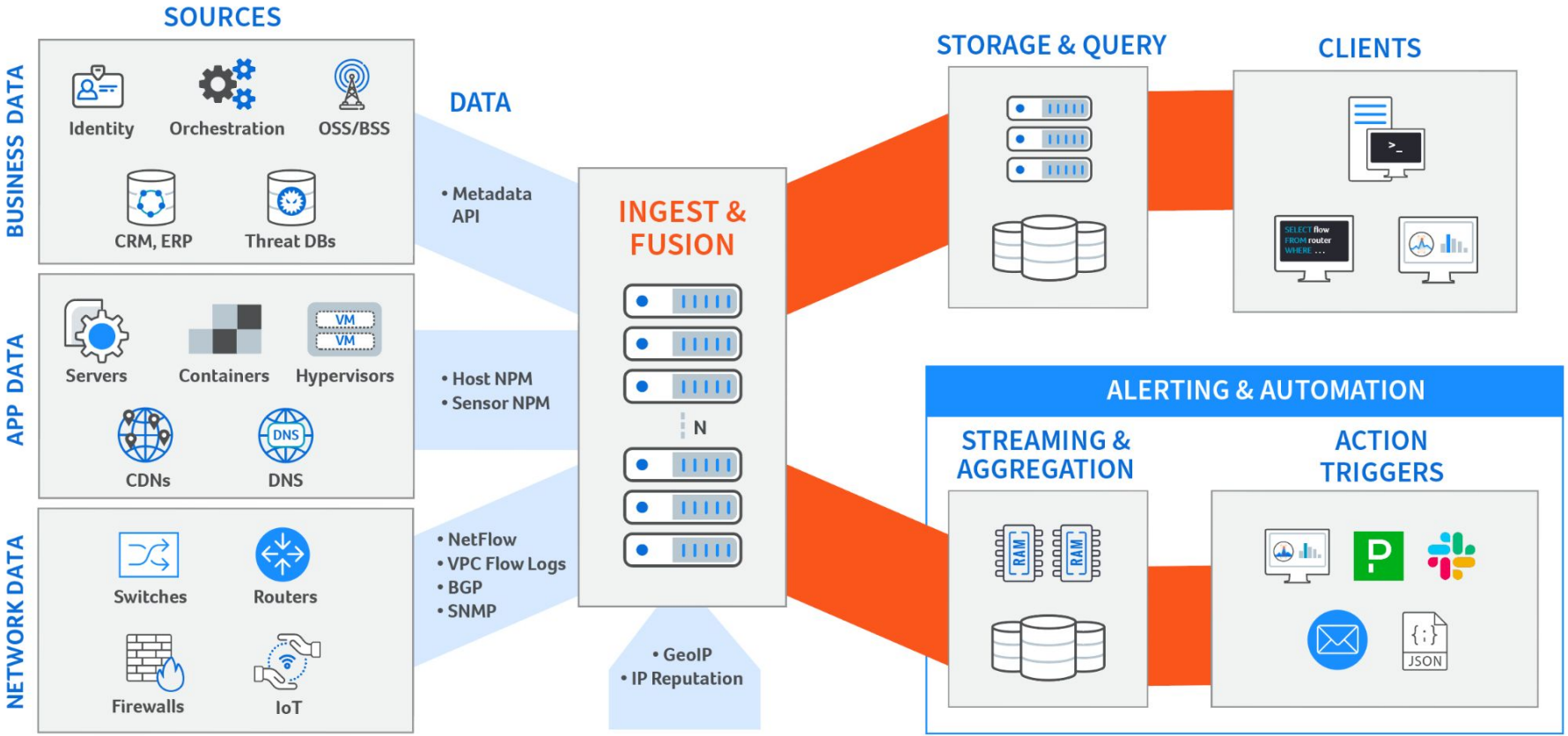


IP Addresses, Ports And Protocols Are Not Enough

- It is awesome to see what traffic is flowing on the network.
- But what does any of this mean in terms or users, content, or network costs?



Best Practices for Leveraging Contextual Data



Telemetry Enrichment

Enrichment with metadata provides context

Geo-location	Threat feeds	Transit costs
BGP table information	Pod names	Public BGP data
PeeringDB information	IPAM	OTT service name
Application identifiers	Process IDs	CDN names
DNS information	Synthetic test results	Customer names

But how?

- Controversial opinion → there is no single source of truth
- The data exists but is spread across numerous sources
- Automation can pull this data together
- Using APIs, push it into a network observability platform that can enrich the network traffic

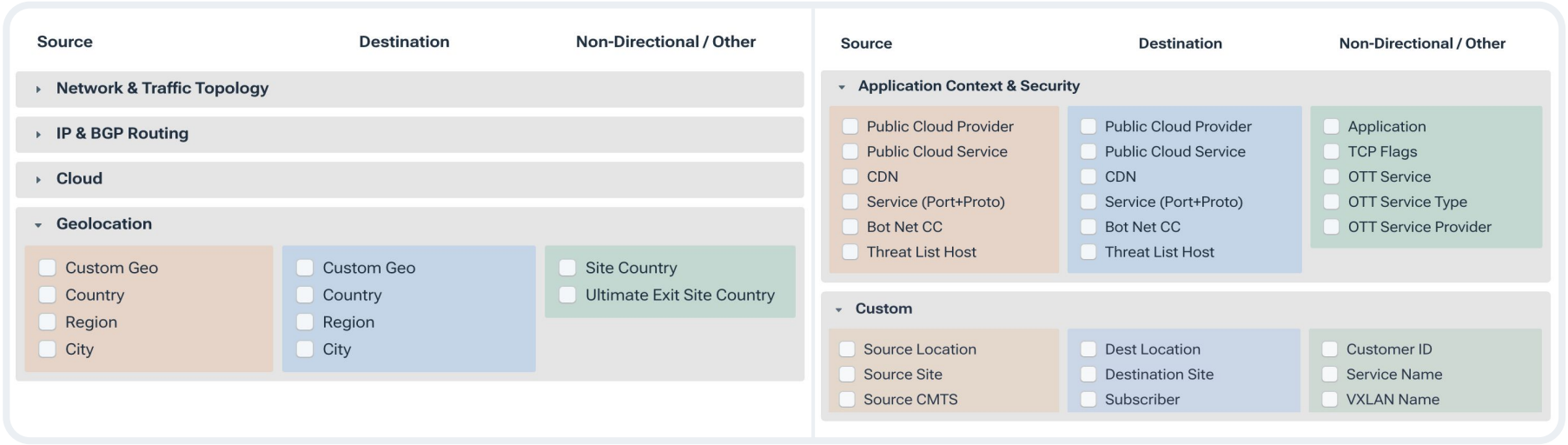
Automation for Enrichment

```
def pull_clients():
    print("Grabbing client list from controller...")
    client = UnifiClient(host=UNIFI_HOST, port=8443, username=UNIFI_USER, password=UNIFI_PASSWORD, site=UNIFI_SITE)
    client_list = client.list_clients()

    print("Writing client list to a file...")
    outfile = open(csvfile, "w")
    # write a header row for the CSV file
    outfile.write("mac_addr,hostname\n")
    for client in client_list:
        mac_addr = client['mac']
        if 'name' in client:
            hostname = client['name']
        elif 'hostname' in client:
            hostname = client['hostname']
        else:
            hostname = client['oui']
        outfile.write(mac_addr + ',' + hostname + '\n')
    outfile.close()
```

https://github.com/jryburn/unifi-to-kentik-cd/blob/main/update_custom_dimension.py

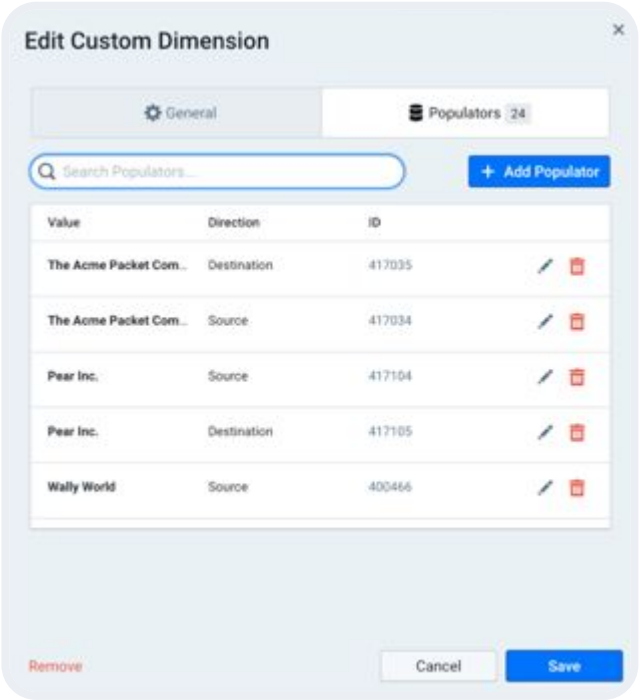
Context for the win!

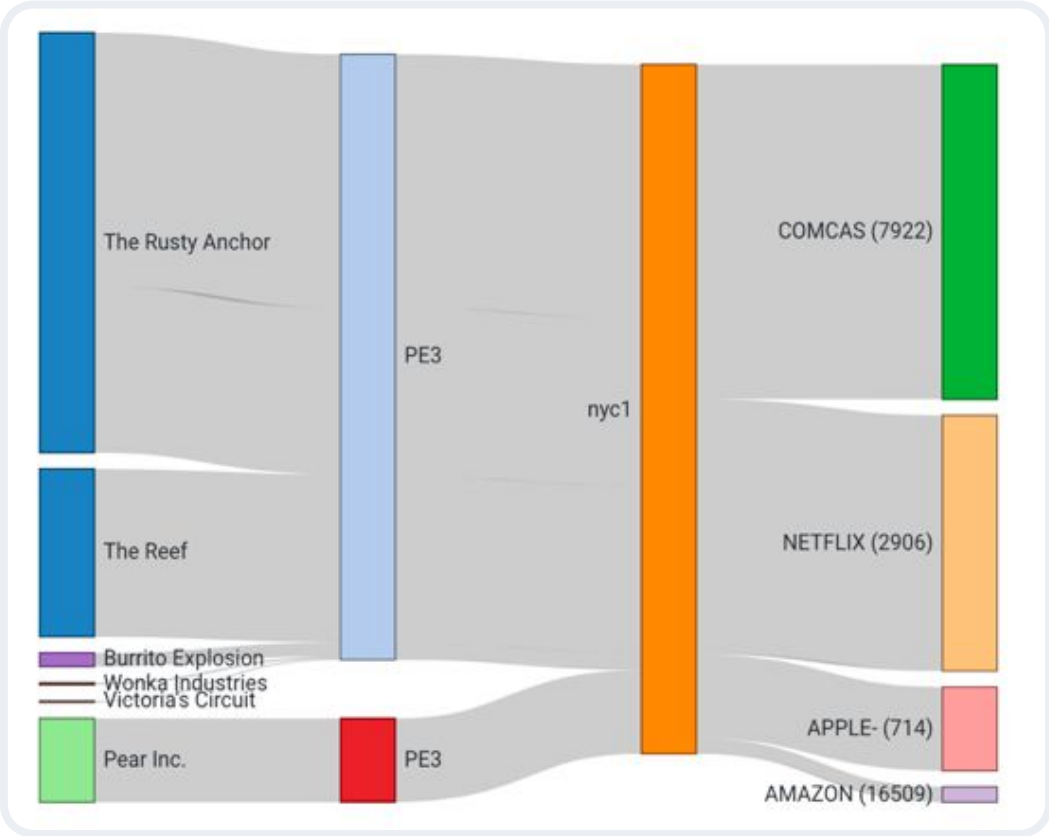


Open Source or Commercial: No Right Answer

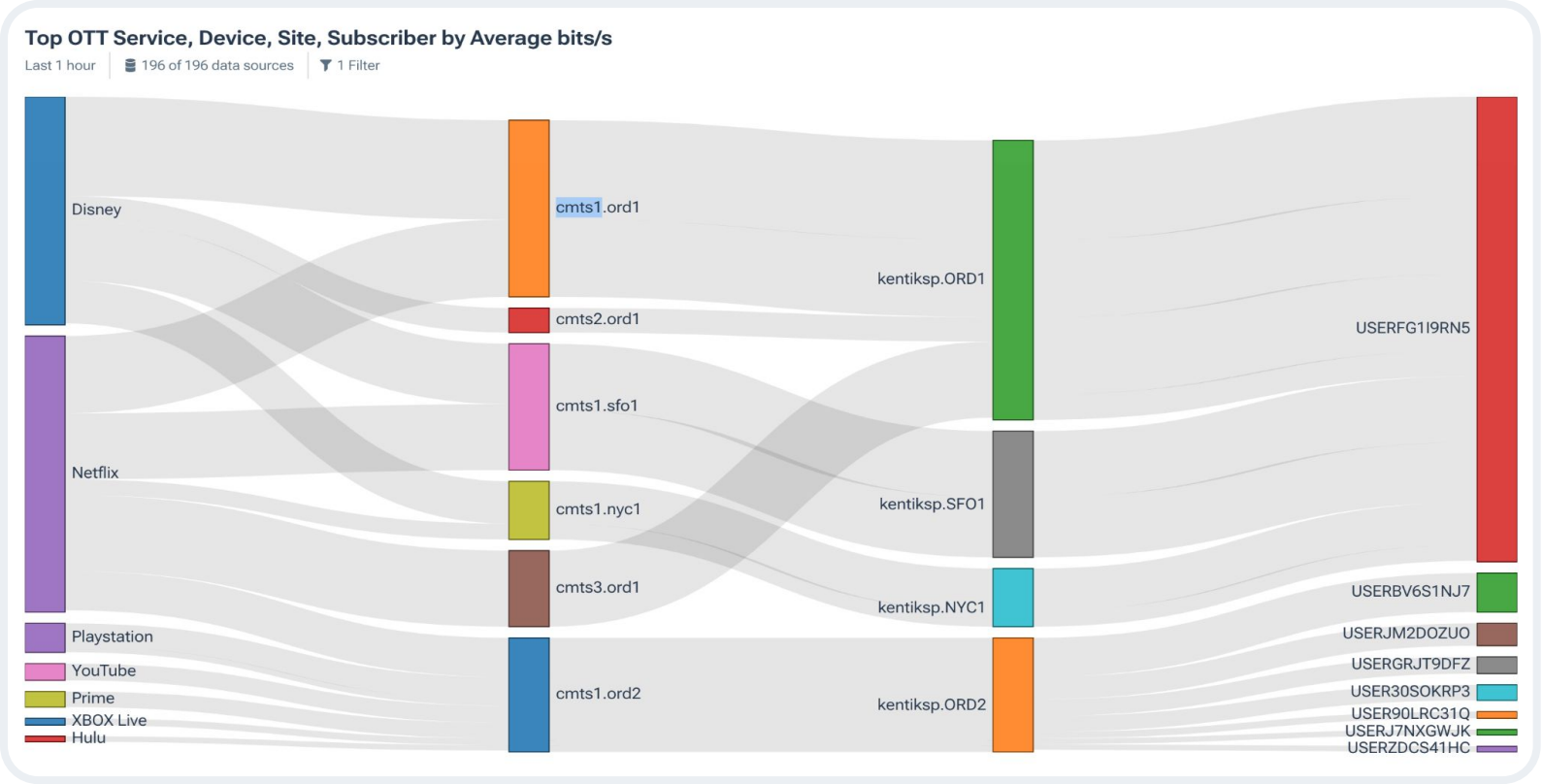
- Open Source tools can be used for flow collection and enrichment:
  - Elastiflow
  - Prometheus
- Make sure you do the enrichment at data ingest time or you lose the context and query performance (useability) will suffer
- Like anything Open Source, each organization has to weigh the resource commitment against the cost of commercial offerings
- **Call to Action:** In 2025, you cannot operate a large scale network without a network observability platform that can collect and enrich this data to help your teams make good decisions

Business Context: The Customer

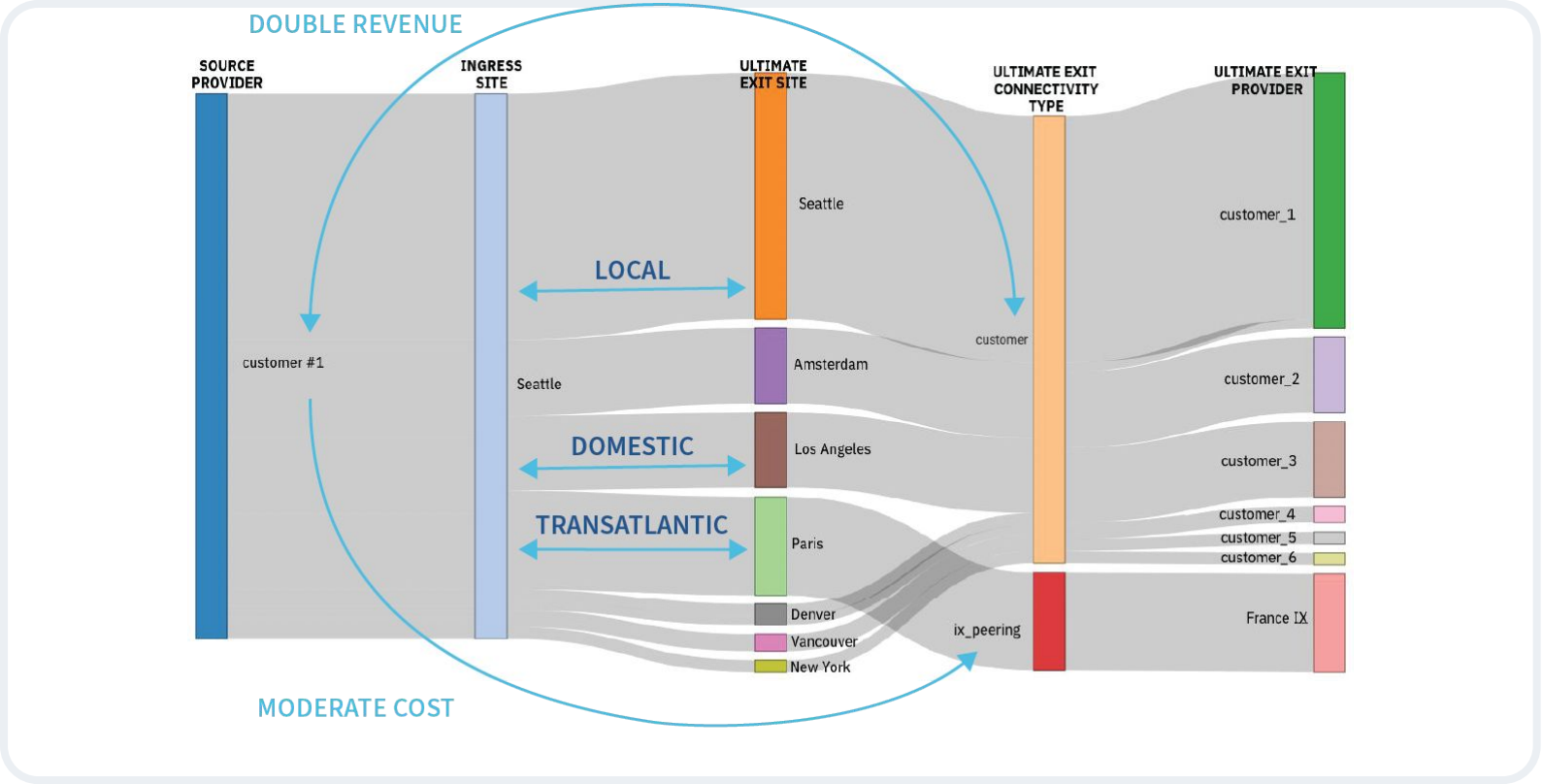




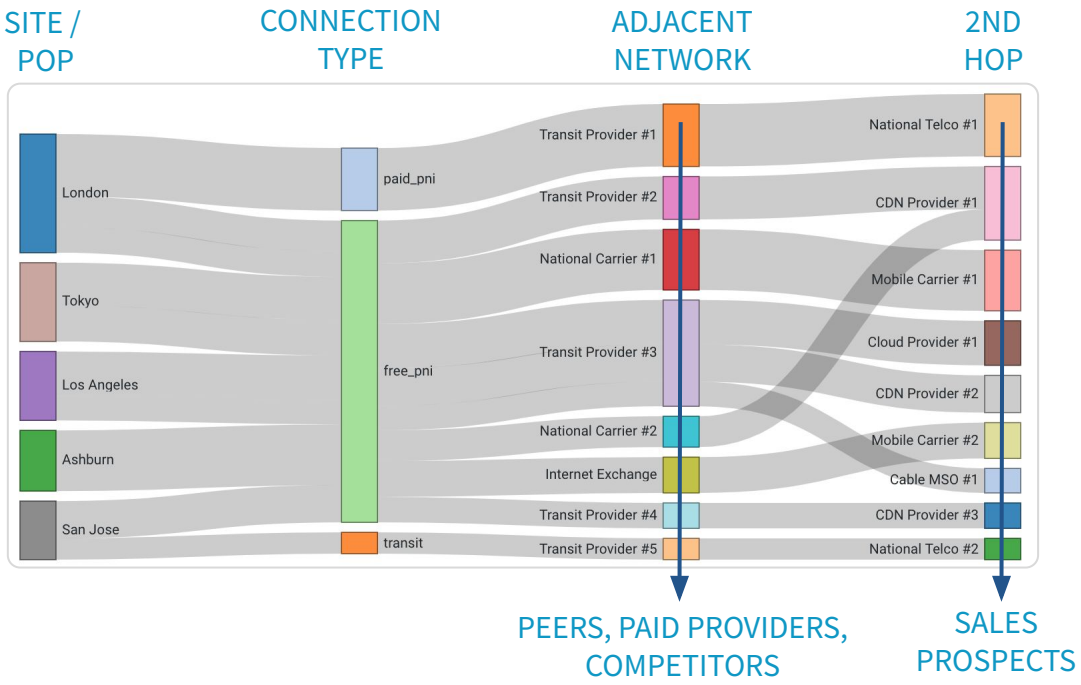
Business Context: OTT Service Usage by CMTS, Site, & User



Business Context: Customer Contract Negotiations



Business Context: Discover Sales and Upsell Opportunities



But now what?

- Alerting with context
- Outbound API webhooks can trigger network automation workflows
- Examples:
  - Customized DDoS mitigation
  - Updating ACLs or Firewalls based on dropped traffic
  - Changing routing policies due to congested ports

Summary

Modern networks are both critical and complex

Network observability is no longer a nice to have

Organizing data by network layer constructs limits the usefulness

Contextual data is needed to fully utilize the power of network data

Network automation can tie the sources of truth into the observability systems to provide this context

Better action can be taken by automation platforms with contextualized observability

Questions?

Thank you!

Jac Kloots
jac@kentik.com

@jkloots

in/jackloots

Join Kentik on Slack