



From Automated to Autonomous: Orchestrating the Network with Agentic AI

HPE Juniper Networking

Aditya Kaul – Principal Architect, Professional Services APJ

Can Agents replace Knowledge workers?



Wrong Question!

Augmenting, not Replacing the Knowledge worker

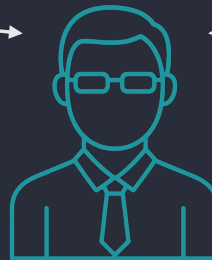
Guardian angel

Discovers and warns you about unanticipated, possibly catastrophic events that could impact your safety, security, and wellbeing



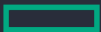
Cognitive amplifier

Anticipates what you want to do and helps you to do it with less effort

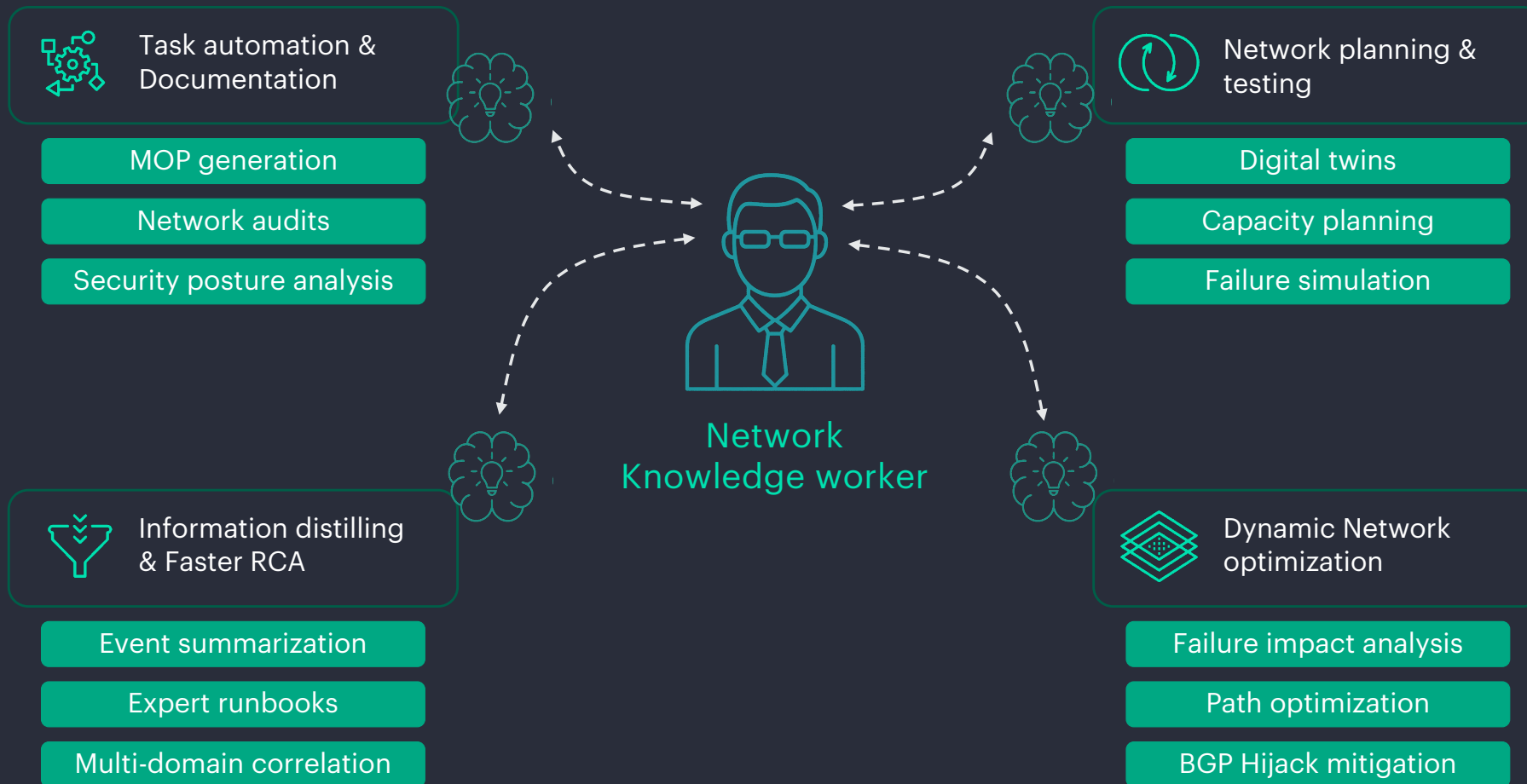


Knowledge worker

Future opportunities lie in creating tools for coping with 21st century world of "too much information and too little time". ... Dr. Raj Reddy, CMU, Turing Award winner

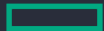
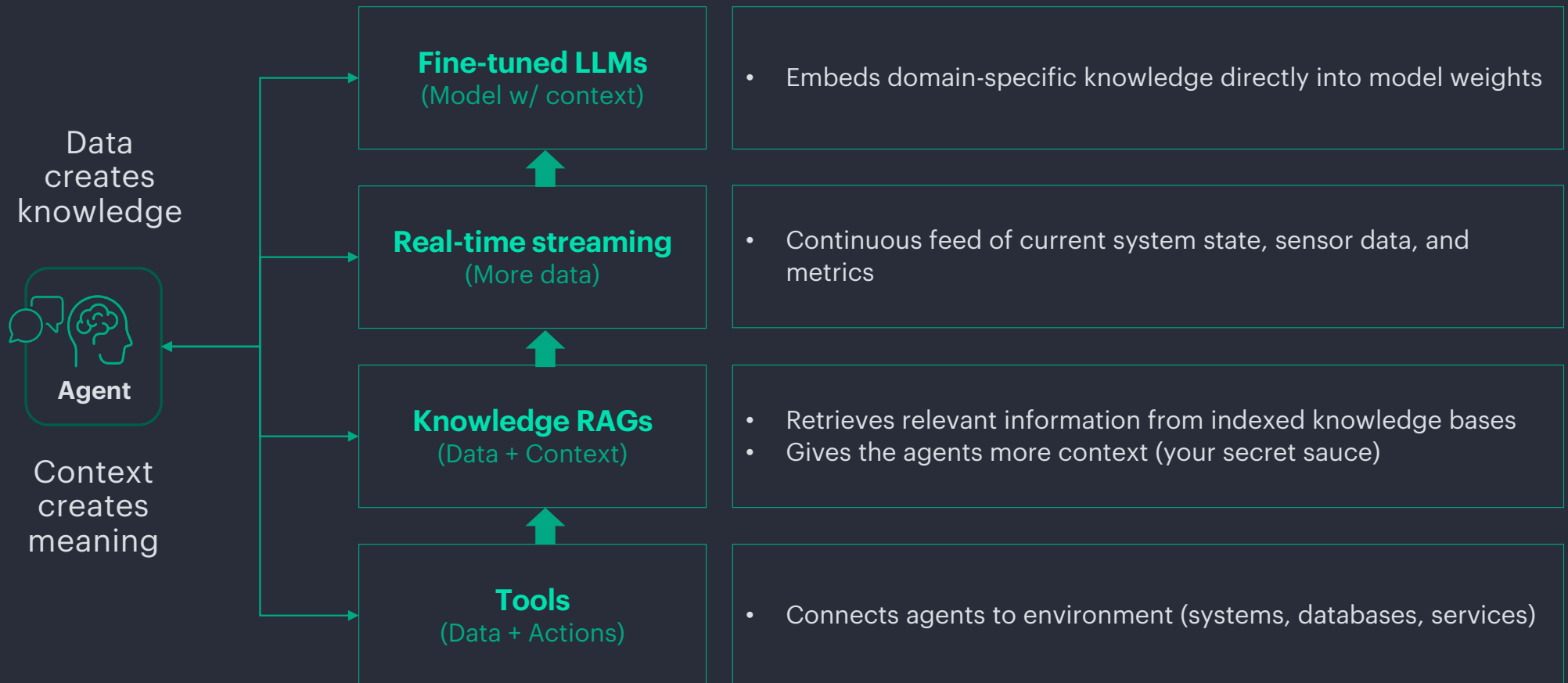


Cognitive Amplifiers for the Network Knowledge Worker



Agentic systems: The building blocks

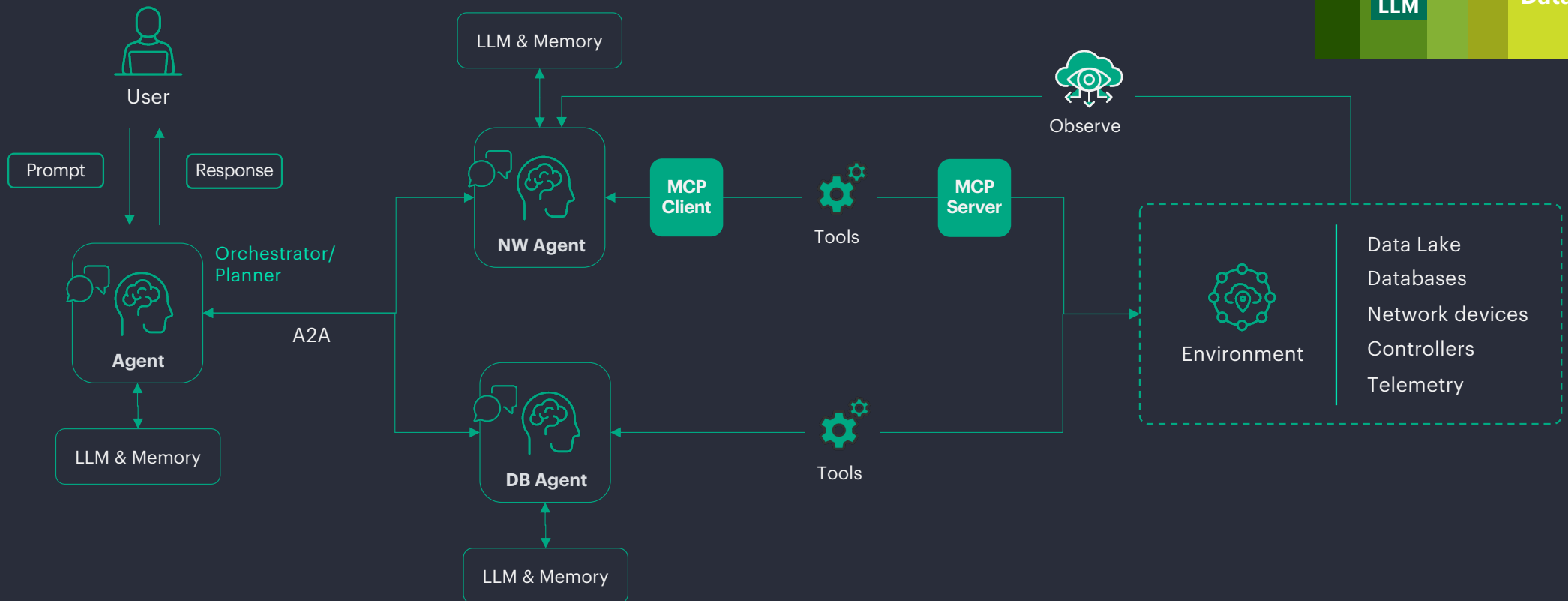
Tuning the system to its environment



Agents & Tools

Beyond traditional AI:

- Agentic AI doesn't just respond—it **plans, executes, and adapts** to achieve complex objectives.
- **A2A & MCP** provide standardized interfaces to integrate custom environments into the LLM's context.
- Agents don't always need A2A & MCP for integration. They can leverage traditional interfaces.



Agentic system

Agents

LLM MCP

LLM Tools

LLM Data



UC 1: Network infrastructure Assessment, Config Audit (BGP, SRv6)

Anatomy of an Effective Network Assessment Prompt

Provide Structured Instructions: Guide the AI with a clear, step-by-step process rather than a single, broad command.

Decompose the Objective into Granular Tasks: Break down the high-level goal into specific areas of analysis. For example:

- ISIS Topology & Adjacency Review
- BGP Peering & Policy Audit
- SRv6 Path Validation
- VPN Service Configuration Check
- Summary of Findings & Recommendations

Define Negative Constraints (The "Don'ts"): Explicitly state what the agent must *not* do or include in its output. This is a critical guardrail to mitigate model hallucination and keep the analysis focused.

```
Analyse the entire network by checking the Juniper vMX logical systems in the POP1(192.168.1.221) and POP2(192.168.1.222)
Create an extensive report with graphics, html but tabulation and illustrations , i want to download this report as html.
```

```
First chat/prompt should define the Part 1 and 2 subsequently other prompt should finish parts 3 & 4.
```

```
Requirements:
```

1. Detailed graphical topologies showing all nodes and how they're connected
 2. OSPF Database and ISIS database details with ISO addressing with realtime examples.
 3. Loopback addressing information
 4. IBGP and eBGP peering diagrams (including Internet peering)
 5. Route Reflector connection and BGP topology visualisation
- ```
5. PDF-ready format
```

```
Part 1 will include:
```

- \* Executive summary and overall network health
- \* Detailed ISIS topology with ISO addresses
- \* ISIS database visualisation
- \* SRv6 infrastructure deep dive with locator topology illustration

```
Part 2 will include:
```

```
PART 2: Routing Protocol Analysis (Next Chat)
```

- | — BGP peering topology & route reflector mesh
- | — IBGP/eBGP relationships & external connectivity
- | — Performance metrics & convergence analysis
- | — Traffic distribution & ECMP analysis
- | please add more details on the internet part, as-path visualisation, the analysis of various prefixes

```
Also show the entire BGP community structure with policies and mention the community and local-preference.
```

```
Part 3 will include:
```

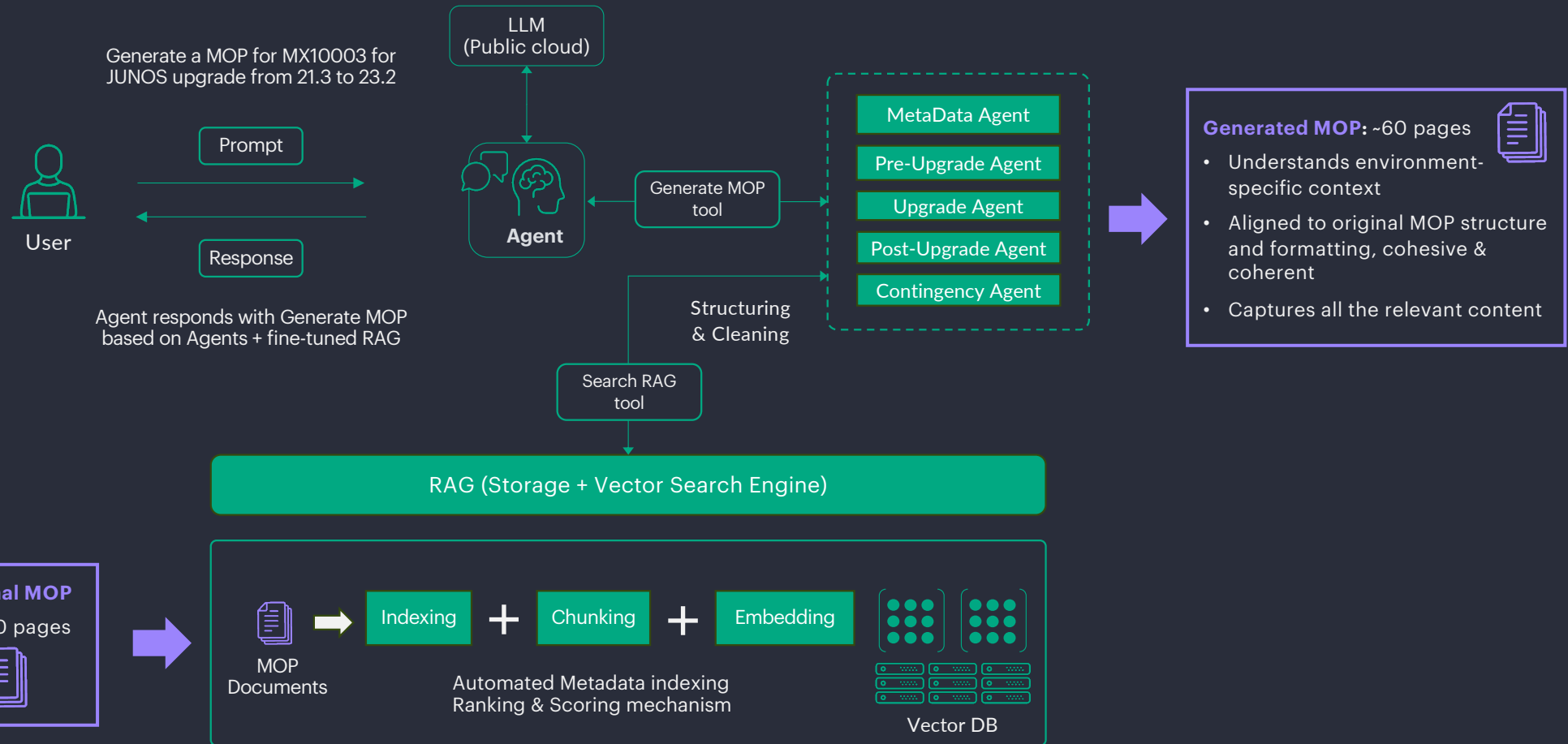
- \* VRF services detailed analysis
- \* Flex-Algorithm 128 deployment details

```
Part 4 will include:
```

- \* Issues and remediation plans
- \* Recommendations and conclusions

# UC 2: MOP generation

## Inference + Multi agent + Fine-tuned RAG



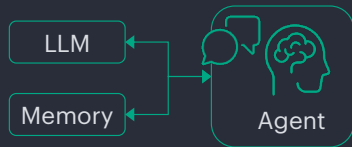
# Why RAG & Multi-agent?

Original MOP: ~60-70 pages



## Inference only

(no RAG, LLM fine-tuning or multi-agent)



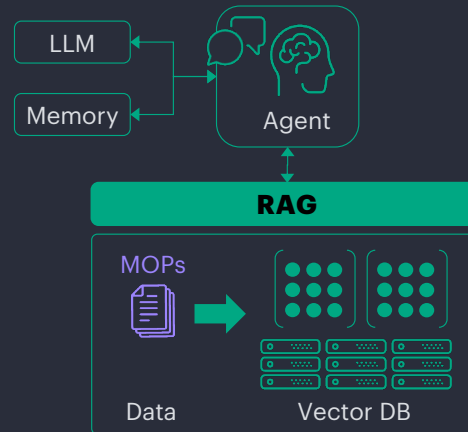
### Generated MOP:



- 11 pages
- Generic MOP based on LLM's previously learnt knowledge
- No environment-specific context

## Inference + RAG

(no LLM fine-tuning or multi-agent)



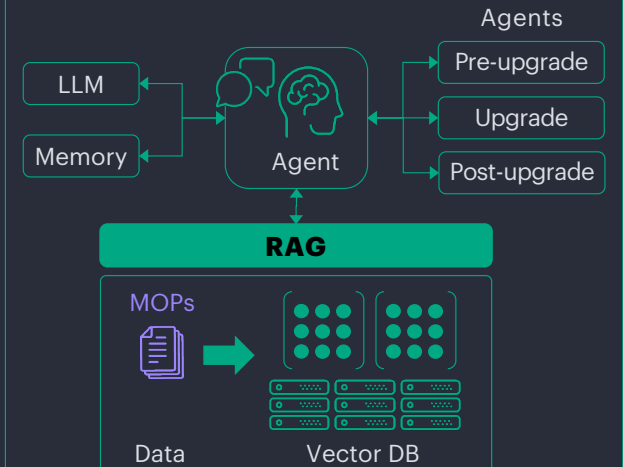
### Generated MOP:



- 278 pages
- Understands environment-specific context
- Poorly structured, not coherent & cohesive

## Inference + RAG + Multi-agent

(no LLM fine-tuning)



### Generated MOP:



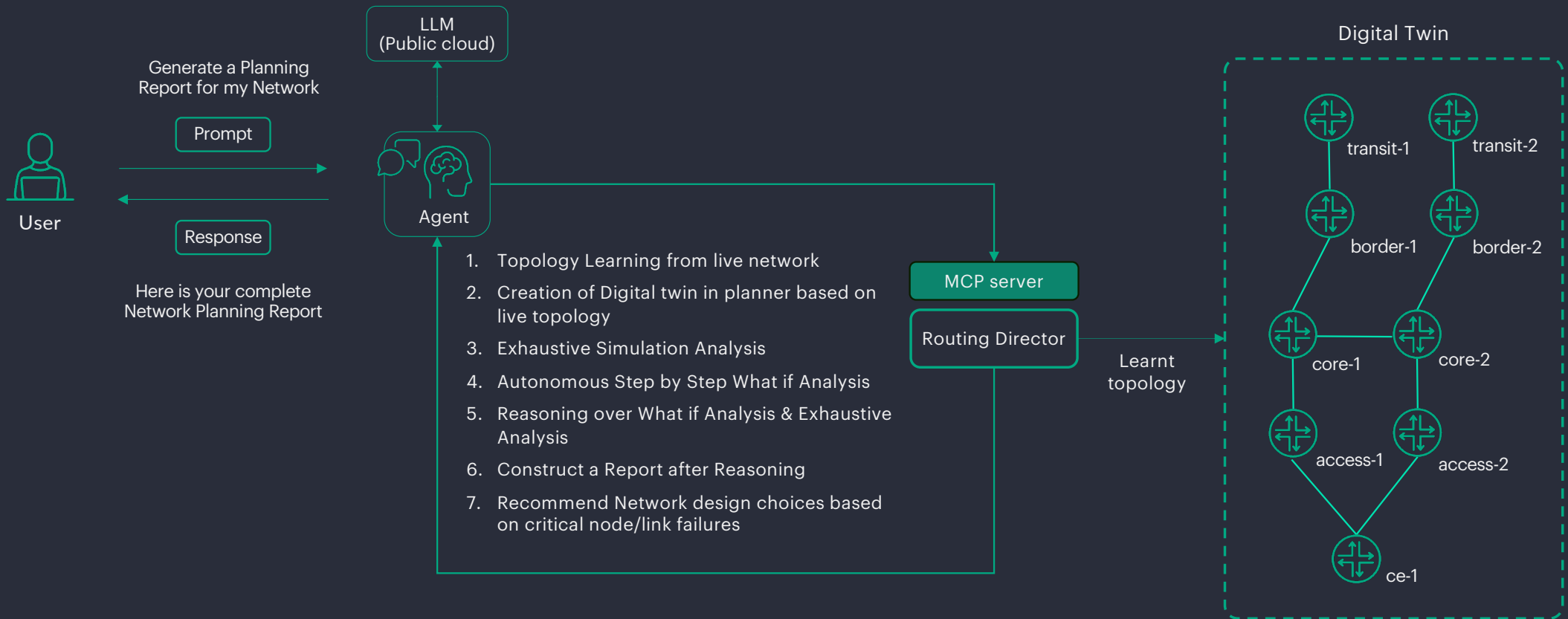
- ~60 pages
- Understands environment-specific context
- Aligned to original MOP structure and formatting, cohesive & coherent
- Captures all the relevant content

Setup

Result

# UC 3: Network capacity planning

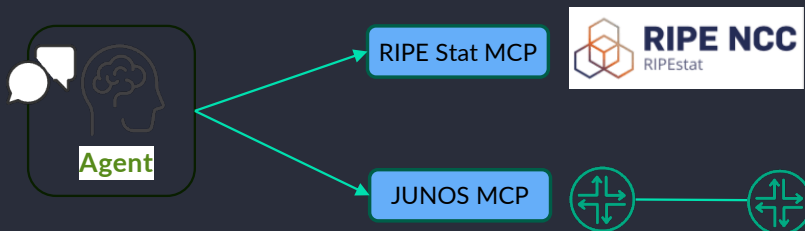
## Failure simulation and What-If analysis



# UC 4: BGP Route hijacking

## Leveraging External Intelligence for BGP Analysis to mitigate issues like Route Hijacking

- **Integrates with Authoritative Sources:** The agent autonomously queries public data platforms like Cloudflare Radar and RIPE Stats.
- **Provides External Visibility:** Gains an essential "outside-in" view of BGP routing to validate internal observations.
- **Enables Event Correlation:** Correlates global routing data with internal network events for faster, more accurate root cause analysis.



### **Critical Finding: This is a Major Issue!**

#### **Who Owns 1.1.1.0/24:**

**Owner:** Cloudflare (AS13335) / APNIC Labs

**Purpose:** Cloudflare's famous public DNS resolver service (1.1.1.1)

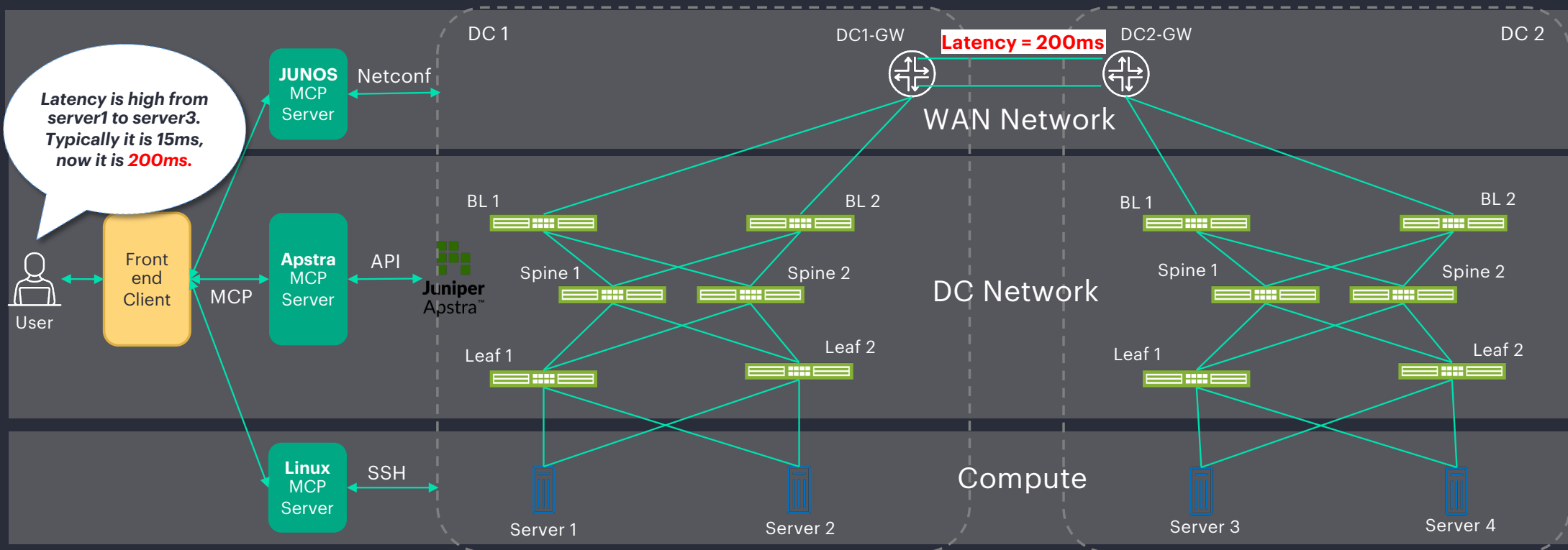
This is **one of the most well-known IP addresses on the internet** - Cloudflare's public DNS service!

#### **⚠️ What This Means for Your Network:**

##### **This is a SERIOUS Configuration Problem:**

- 1. IP Hijacking/Route Leak:** Your router B4 is incorrectly advertising Cloudflare's prefix (1.1.1.0/24) as if AS7473 owns it
- 2. BGP Hijack Scenario:** When you advertise 1.1.1.0/24:
  - Some networks may route traffic destined for Cloudflare's DNS (1.1.1.1) to YOUR network instead
  - This creates a "blackhole" where DNS queries never reach Cloudflare
  - Users trying to use 1.1.1.1 as their DNS server would fail
- 3. Why It's Not Globally Visible in RIPE:**
  - RIPE collectors may be filtering it as an invalid announcement
  - Most networks likely reject this route as invalid (AS13335 is the legitimate origin)
  - Your peer AS 2004 may be accepting it but others are not
- 4. Reputation Risk:**
  - This looks like BGP hijacking (even if unintentional)
  - Could result in your network being filtered or blacklisted
  - Violates internet routing best practices ↓

# UC 5: Multi-domain Root Cause Analysis



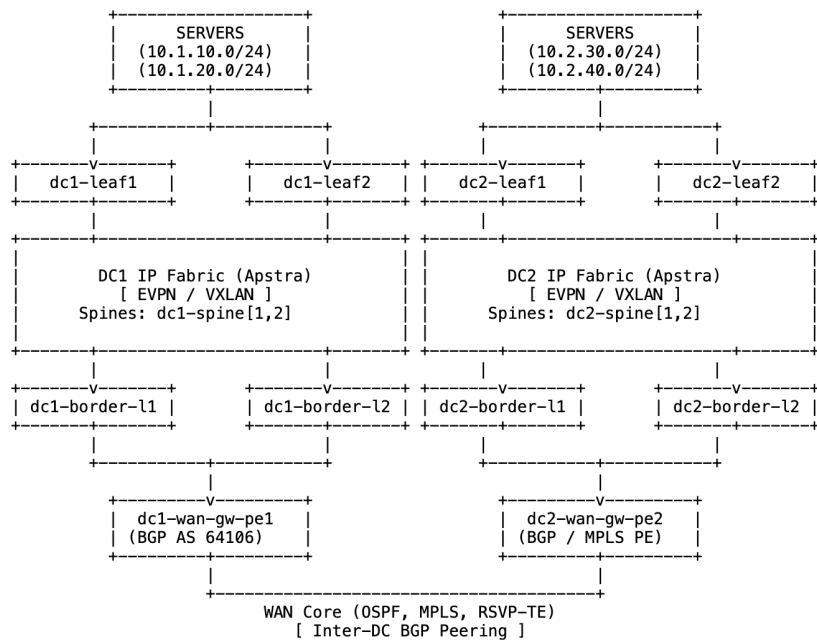
## Demo scenario:

- User complains that latency is high from server1 to server3
- Typical latency = 5ms, Current latency = 200ms
- Agent troubleshoots the problem to find the root cause



# Agent understands the underlay and overlay topology

give an overview of my datacenter. how are the underlay and overlay network built, how are the DCs connected over WAN, which are the servers in each DC and which networks do they belong to? give me a report including an ascii diagram of the network.



## 1. Data Center Architecture (Underlay & Overlay)

Both datacenters (JCL-DC1 and JCL-DC2) are managed as independent Juniper Apstra blueprints using a standard IP Fabric Spine-Leaf architecture.

- **Underlay Network:** Built using eBGP to provide robust, loop-free IPv4 connectivity between all leaf and spine switches.
- **Overlay Network:** Built using EVPN-VXLAN. This provides the Layer 2/Layer 3 multi-tenancy abstraction over the IP fabric, allowing virtual networks to span across the respective datacenters.

Hardware Footprint:

- **DC1 Nodes:** dc1-spine1, dc1-spine2, dc1-leaf1, dc1-leaf2 (compute leafs), dc1-border-leaf1, dc1-border-leaf2 (exit leafs).
- **DC2 Nodes:** dc2-spine1, dc2-spine2, dc2-leaf1, dc2-leaf2 (compute leafs), dc2-border-leaf1, dc2-border-leaf2 (exit leafs).

## 2. WAN Interconnect (Data Center Interconnect - DCI)

The two datacenters are connected over a WAN via Provider Edge (PE) Gateway Routers (dc1-wan-gw-pe1 and dc2-wan-gw-pe2).

- **Routing & Transport:** The WAN core operates on an OSPF / MPLS underlay.
- **Traffic Engineering:** The interconnect uses RSVP-TE (Resource Reservation Protocol with Traffic Engineering). Dedicated Label Switched Paths (LSPs) such as LSP-T0-DC2-PRIMARY and backup LSPs are established to strictly steer traffic over primary and backup links across the WAN.
- **DCI Overlay:** Inter-DC overlay connectivity relies on external BGP (WAN-INTERDC-L1) peered over the MPLS transport to exchange routes between the Border Leafs of DC1 and DC2.

## 3. Servers and Virtual Networks

The physical servers (Generic Systems) are connected to the compute leaf switches in their respective datacenters and are mapped into the EVPN-VXLAN overlay networks.

DC1 Networks (Mapped to dc1-leaf1/2):

- v10: Subnet 10.1.10.0/24 (VXLAN VNID: 10003)
- v20: Subnet 10.1.20.0/24 (VXLAN VNID: 10002)

DC2 Networks (Mapped to dc2-leaf1/2):

- v30: Subnet 10.2.30.0/24 (VXLAN VNID: 10005)
- v40: Subnet 10.2.40.0/24 (VXLAN VNID: 10004)

# Agent runs server-to-server latency test

give me a table with the server mapping to the virtual networks and routing zones

| Datacenter | Server Name | IP Address   | Subnet       | Virtual Network | Routing Zone         |
|------------|-------------|--------------|--------------|-----------------|----------------------|
| JCL-DC1    | DC1-server1 | 10.1.10.100  | 10.1.10.0/24 | v10 (VNI 10003) | RZ_10-20 (VNI 10001) |
| JCL-DC1    | DC1-server2 | 10.1.20.100  | 10.1.20.0/24 | v20 (VNI 10002) | RZ_10-20 (VNI 10001) |
| JCL-DC2    | DC2-server1 | 10.2.30.100* | 10.2.30.0/24 | v30 (VNI 10005) | RZ_30-40 (VNI 10000) |
| JCL-DC2    | DC2-server2 | 10.2.40.100  | 10.2.40.0/24 | v40 (VNI 10004) | RZ_30-40 (VNI 10000) |

check ping between servers within DC1 and from DC1 to DC2. ignore DC2-server1, its down. give a table with the latency

| Source Server | Destination Server | Destination IP | Ping Status         | Avg Latency (RTT) | Scope                |
|---------------|--------------------|----------------|---------------------|-------------------|----------------------|
| DC1-server1   | DC1-server2        | 10.1.20.100    | ✔ Success (0% loss) | 2.74 ms           | Intra-DC (DC1)       |
| DC1-server1   | DC2-server2        | 10.2.40.100    | ✔ Success (0% loss) | 218.13 ms         | Inter-DC (DC1 → DC2) |
| DC1-server2   | DC2-server2        | 10.2.40.100    | ✔ Success (0% loss) | 215.17 ms         | Inter-DC (DC1 → DC2) |

As expected, traffic within DC1 is very fast (~2.7ms routing between VNIs v10 and v20), while the WAN transit over to DC2 incurs a higher latency of ~215-218ms.

# Agent diagnoses the cause of the high-latency

why is the latency so high between the DCs? Investigate

## 1. Two WAN Links with Drastically Different Profiles

The datacenters are connected via two distinct physical WAN links. Tests executed directly from the WAN gateway (`dc1-wan-gw-pe1`) show their raw performance:

- LINK1 (`172.16.252.1` via `ge-0/0/2.0`): Highly degraded/high latency link. Pings average >210 ms.
- LINK2 (`172.16.252.3` via `ge-0/0/4.0`): Low latency link. Pings average ~4 - 8 ms.

## 2. Forwarding Table Override

Despite having an active, low-latency backup path (LINK2), there is an explicit forwarding-table export policy (`RSVP-STEER-DCI-LINK1`) deployed on the routers that forces all DCI subnets onto the high-latency LINK1 path.

On `dc1-wan-gw-pe1`, the EVPN overlay transit subnets (`192.168.2.0/24`) are explicitly matched and steered over the `LSP-T0-DC2-PRIMARY` label-switched path:

**Conclusion:** All server-to-server traffic between DC1 and DC2 relies on EVPN/VXLAN encapsulated within these IPs. Because the WAN edge routers are strictly forcing this encapsulation traffic over the high-latency "LINK1", the overlay application traffic inherits the ~215ms latency.

To fix this, the forwarding-table export policy on both WAN routers needs to be updated to steer traffic to `LSP-T0-DC2-PRIMARY-LINK2` (and its DC1 equivalent) which maps to the fast `172.16.252.3` path. Would you like me to make these configuration changes to restore optimal latency?



# Agent fixes the problem

yes do it

The configuration has been successfully applied to both WAN gateways (`dc1-wan-gw-pe1` and `dc2-wan-gw-pe2`). I swapped the forwarding-table export policy from `RSVP-STEER-DCI-LINK1` over to `RSVP-STEER-DCI-LINK2`, effectively moving the DCI overlay traffic onto the low-latency transport path.

I ran a quick verification ping from `DC1-server1` to `DC2-server2` (`10.2.40.100`), and the latency has successfully dropped from ~218 ms down to ~15 ms.

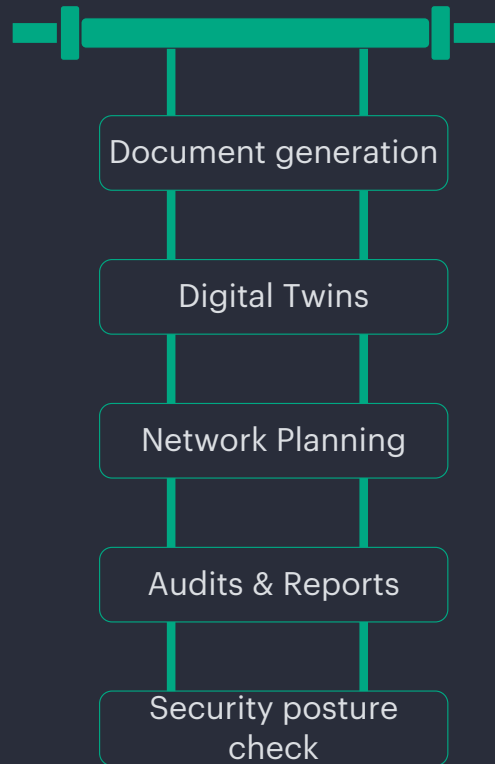
give me the same latency table again

Here is the updated latency table reflecting the changes after routing traffic over the low-latency DCI link:

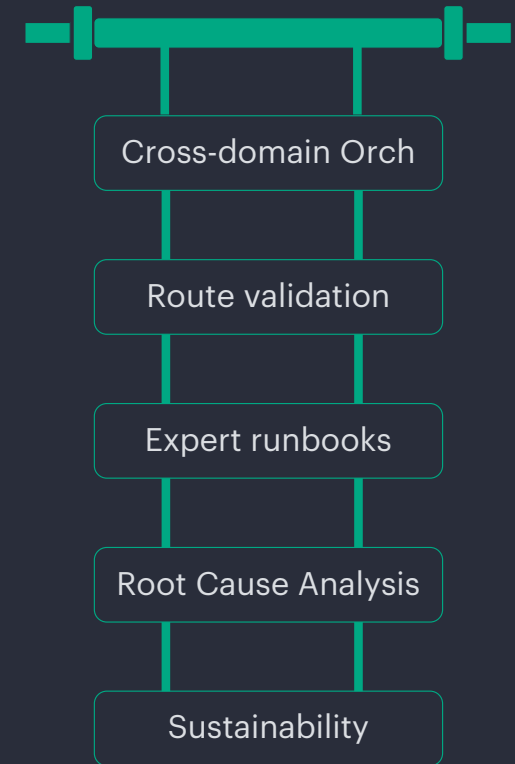
| Source Server | Destination Server | Destination IP | Ping Status         | Avg Latency (RTT) | Scope                |
|---------------|--------------------|----------------|---------------------|-------------------|----------------------|
| DC1-server1   | DC1-server2        | 10.1.20.100    | ✓ Success (0% loss) | 2.62 ms           | Intra-DC (DC1)       |
| DC1-server1   | DC2-server2        | 10.2.40.100    | ✓ Success (0% loss) | 17.75 ms          | Inter-DC (DC1 → DC2) |
| DC1-server2   | DC2-server2        | 10.2.40.100    | ✓ Success (0% loss) | 13.73 ms          | Inter-DC (DC1 → DC2) |

The Inter-DC latency has successfully dropped from ~215-218 ms down to ~13-17 ms.

# The Augmented Network Knowledge Worker



The Augmented  
Network Knowledge Worker



# Learnings

## Model selection

**Choose wisely:** LLM vs SLM (*Bigger is not always better!*)

**Find your sweet spot:** 20-30B models in our experiments (GPT-OSS-20B, Qwen2.5-coder:32B, Gemini2.5-flash-lite)

**Consider:** Context window, Training data, Agentic capabilities, Open-source vs Open-weights vs Closed

## Agents

**Be pedantic:** Use Pydantic classes for data validation, parsing, and serialization

**Design your execution patterns:** Sequential vs Parallel, Orchestrator & Planner agents, Observer & Critic agents

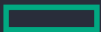
**Manage context windows:** Provide only the required tools

## Agent-to-agent

**Frameworks:** Choose your favorite, connect them through A2A

**Design your callbacks:** How and when agents handoff, How they use short-term and persistent memory

**Keep them in check:** Prevent scope creep across agents (define the boundaries)



# Thank You

Nitin Vig ([nitin.vig@hpe.com](mailto:nitin.vig@hpe.com))

Aditya Kaul ([aditya.kaul@hpe.com](mailto:aditya.kaul@hpe.com))

Credits: Thank you **co-pilot** for augmenting this knowledge worker

