

AI Skills Primer for CCNP Candidates **cisco** Live !

Hank Preston

Distinguished Architect Learn with Cisco
ccie 38336 | devnet expert 20220001

Cisco Webex App

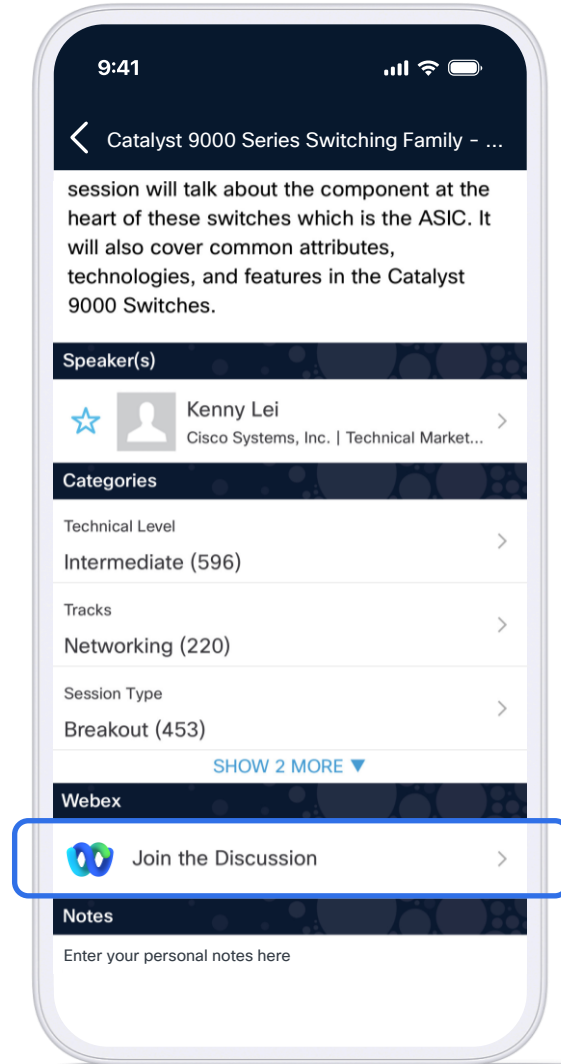
Questions?

Use Cisco Webex App to chat with the speaker after the session

How

- 1 Find this session in the Cisco Live Mobile App
- 2 Click “Join the Discussion”
- 3 Install the Webex App or go directly to the Webex space
- 4 Enter messages/questions in the Webex space

Webex spaces will be moderated by the speaker until 14 November 2025.



<https://ciscolive.ciscoevents.com/ciscolivebot/#BRKCRT-2301>

Agenda

- 01 Why is AI in Cisco certifications?**
- 02 AI basics and fundamentals**
- 03 7 Risks and security implications of AI**
- 04 AI's impact and requirements on infrastructure**
- 05 Developing Agentic AI solutions**



Why is AI in Cisco Certifications?

- 184% growth for AI related skills in job postings
- 18% - 47% higher salary for AI skilled professionals

ICT in Motion: The Next Wave of AI Integration (2025)

- **Pervasiveness of AI Skills in ICT**
78% of jobs include AI Skills
- **Explosive growth of AI Specific Roles**
7 of top 10 fastest growing roles AI related
- **Critical Skills Gaps in Technical AI Areas**
LLMs, Prompt Engineering, AI Ethics, etc
- **Surging Demand for AI Security Skills**
↑ 256% demand for “Responsible AI Implementation”
- **AI’s Impact on Infrastructure and Operations Skills**
↑ 256% growth for “LLMOps and Model Serving”

[Link to report](#)



Do we still need to learn networking?

- AI effectiveness depends on your expertise
- Real learning requires real effort
- Expertise takes practice, not shortcuts
- Innovation starts with humans
- AI is always catching up



Why the human network engineer still matters

- Innovates creative solutions to unprecedented network challenges
- Applies deep contextual understanding of business priorities and technical constraints
- Integrates new technologies thoughtfully into diverse, evolving environments
- Anticipates and addresses complex, multi-layered issues beyond AI's scope
- Translates business goals into resilient, adaptive network architectures
- Validates and interprets AI recommendations, preventing automation errors
- Provides accountability and trusted governance in critical situations



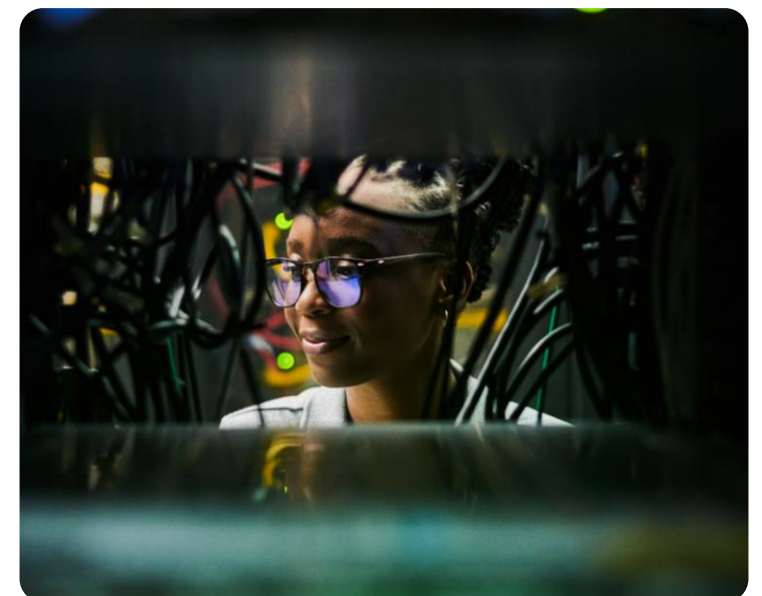
How we interact with AI today and tomorrow



User of AI Tools



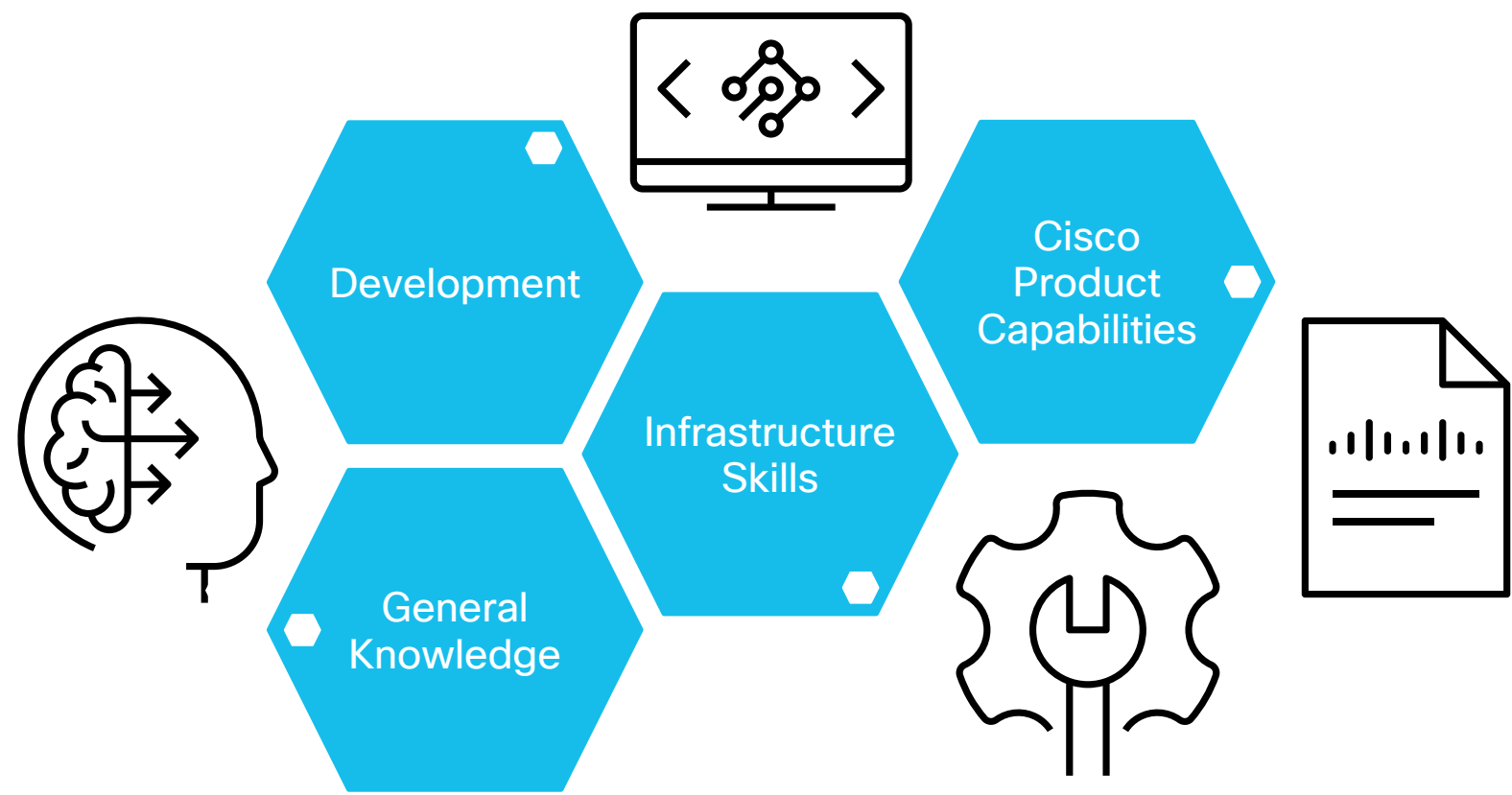
Supervisor of AI Tools



Builder of AI Infrastructure

**Where does AI show up in
Cisco certifications?**

Areas of AI Knowledge Covered in Blueprints



Where in Cisco Certifications will you find AI?

Cisco Certified Network Associate



CCNP Data Center, and Collaboration



Cisco Certified Network Associate (CCNA)

6.4 Explain AI (generative and predictive) and machine learning in network operations

Data Center Core (DCCOR)

4.3 Describe high-performance network enabling technologies for AI in data center infrastructure

Collaboration Hybrid and Cloud (CLHCT)

4.2 Describe AI features in cloud collaboration solutions

Where in Cisco Certifications will you find AI?

CCNP Automation



Automation Core (AUTOCOR)

- 4.1 Describe the benefits and risks of AI-assisted code development for network automation such as data privacy, IP ownership, and code validation
- 4.2 Interpret the security risks in a given AI-based network automation solution
- 4.3 Construct an MCP server to provide network information to an AI-agent using Python FastMCP
- 4.4 Construct a conversational agent that leverages LLMs for network automation
- 4.5 Evaluate the accuracy of AI recommendations on a network automation solution

Enterprise Networking Automation (ENAUTO)

- 5.1 Describe AI in controller-based platforms
- 5.2 Describe AI-assisted code development for network automation
- 5.3 Describe the security risks in a given AI-based network automation solution
- 5.4 Construct an MCP server to provide network information to an AI-agent using Python FastMCP

Data Center Networking Automation (DCNAUTO)

- 5.1 Describe AI-assisted code development for network automation
- 5.2 Describe the security risks in a given AI-based network automation solution
- 5.3 Describe the integration of network devices, controllers, and management platforms with AI agents

Where in Cisco Certifications will you find AI?

Cisco Certified Design Expert



CCIE Data Center



CCDE Unified Exam Topics

1.4 AI/Machine Learning

- 1.4.a Business needs
- 1.4.b Data sovereignty (location and public/private/hybrid)
- 1.4.c Security
- 1.4.d Assurance
- 1.4.e Integrity
- 1.4.f Impacts (such as storage requirements and traffic patterns)
- 1.4.g Auto scalability
- 1.4.h Cost and ROI
- 1.4.i Governance

3.2 AI network design use cases (such as machine learning, large language models, and pattern recognition)

5.1.g Impacts of AI on corporate security policy (such as IP, PIL, proprietary information, quality, corporate credibility, and use of external AI services)

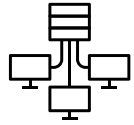
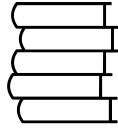
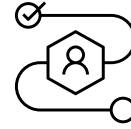
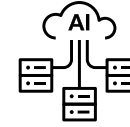
CCIE Data Center Lab Exam

5.2 RoCE v2 transport for high performance networks, such as for AI/ML workloads

- 5.2.a DCQCN congestion control
 - 5.2.a.i PFC
 - 5.2.a.ii ECN

AI basics and fundamentals

Brief History of Artificial Intelligence



1950s - 1980s

AI Foundation Laid

- Alan Turing's foundational concepts and Turing Test
- Early expert systems and rule-based AI
- First neural network experiments
- Multiple "AI winters" due to limited computing power and overhyped expectations

1990s - 2000s

Machine Learning Renaissance

- Support Vector Machines and improved statistical methods
- Early internet-scale data collection
- Introduction of practical machine learning algorithms
- Foundation for data mining and pattern recognition

2010 - 2017

Deep Learning Revolution

- GPU breakthrough (2012) - dramatically faster neural network training
- Mainstream AI applications emerge:
- Recommendation engines
- Voice assistants
- Image and facial recognition
- Language translation accuracy
- Convolutional Neural Networks (CNNs) for computer vision

2017 - 2022

Transformer Breakthrough

- Transformer model introduction (2017) - "Attention is All You Need"
- Language models become sophisticated (GPT series evolution)
- AI begins impacting enterprise and technical domains
- Advanced natural language processing capabilities

2022 - Now

Generative AI Explosion

- ChatGPT public release (November 2022) democratizes AI access
- Generative AI goes mainstream - creating content, code, and solutions
- Rapid integration into networking tools and platforms
- AI becomes essential skill for technical professionals

Six Types of AI – Different Types of Intelligence

Generative AI

- Creates new content, data, or solutions based on learned patterns from training data
- Produces human-like text, images, code, or other media that didn't exist before
- Examples: Writing network documentation, generating network configuration, creating troubleshooting scripts

Predictive AI

- Analyzes historical data and patterns to forecast future events or outcomes
- Uses statistical models and machine learning to identify trends and make predictions
- Examples: Network traffic forecasting, predicting equipment failures, anticipating security threats

Classification AI

- Categorizes and labels data into predefined groups or classes based on learned characteristics
- Identifies patterns to sort information into discrete categories
- Examples: Network traffic classification, malware detection, categorizing support tickets by priority

Recommendation AI

- Suggests optimal choices or actions based on user behavior, preferences, and similar patterns
- Combines collaborative filtering and content-based analysis to provide personalized suggestions
- Examples: Recommending network optimization strategies, suggesting configuration changes, proposing security policy updates

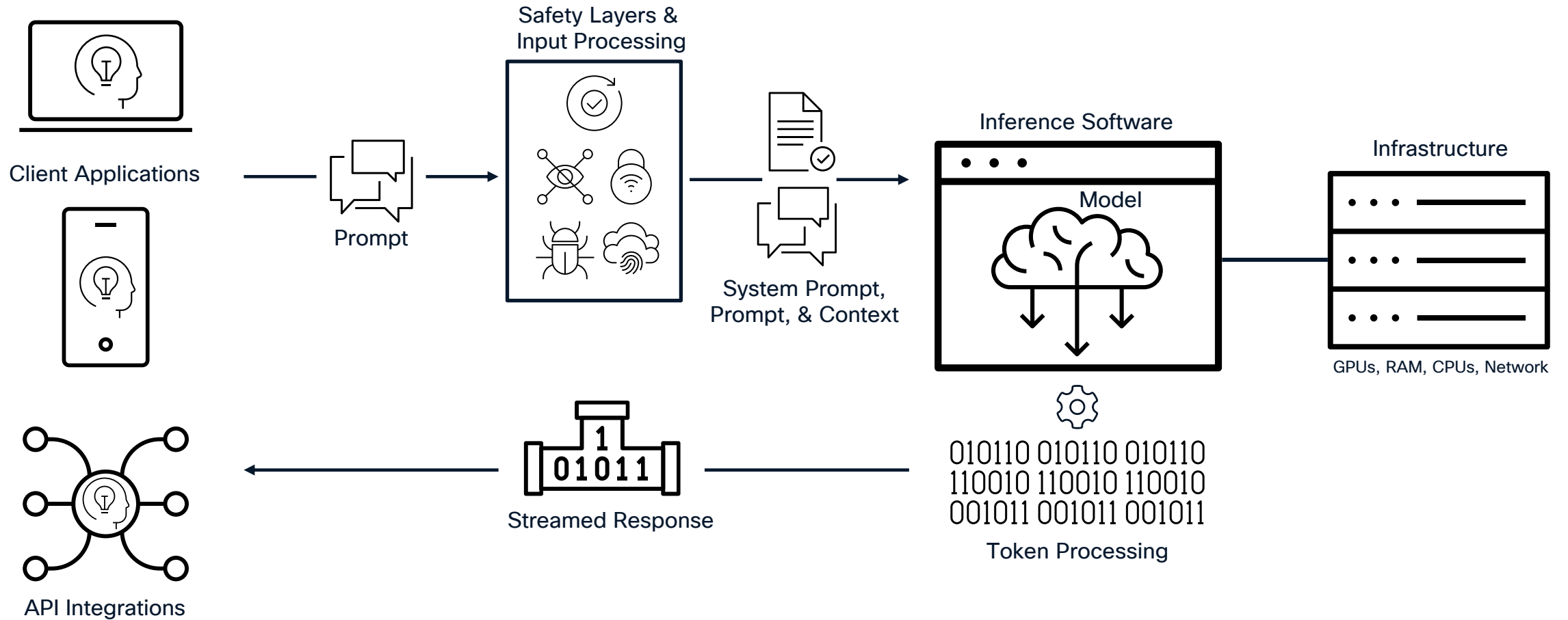
Computer Vision AI

- Processes and interprets visual information from images, videos, or real-time camera feeds
- Recognizes objects, patterns, text, and spatial relationships in visual data
- Examples: Network cable management through image analysis, datacenter monitoring via cameras, visual network topology mapping

Natural Language Processing AI

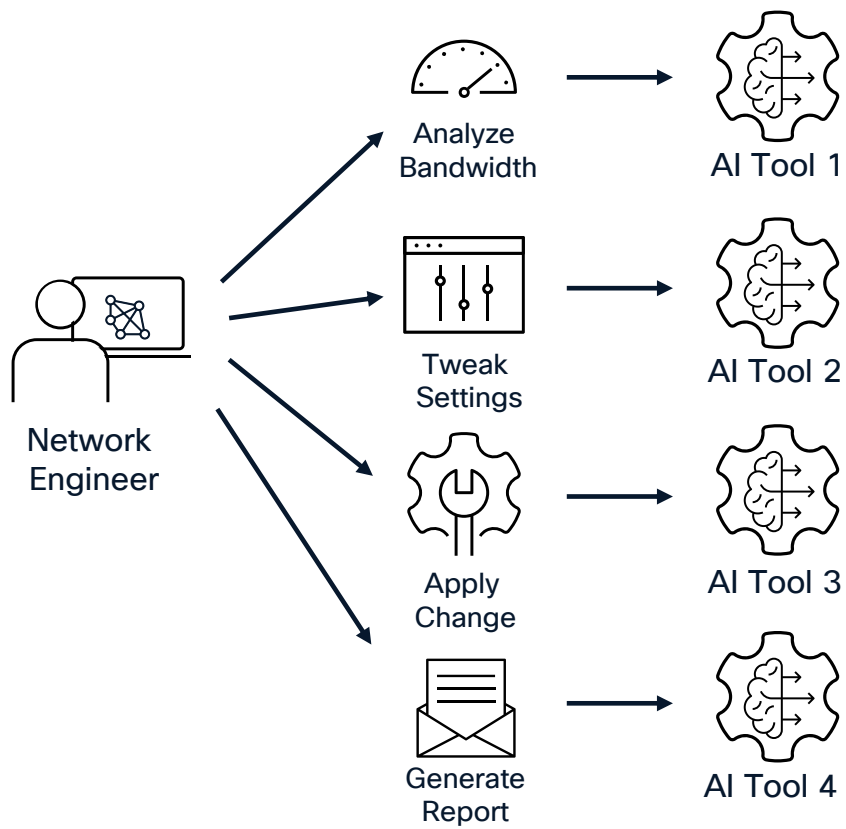
- Understands, interprets, and generates human language in text or speech form
- Processes unstructured text data to extract meaning, sentiment, and actionable insights
- Examples: Analyzing network logs for issues, processing support tickets, converting spoken commands to network configurations

Generative AI Application Architecture



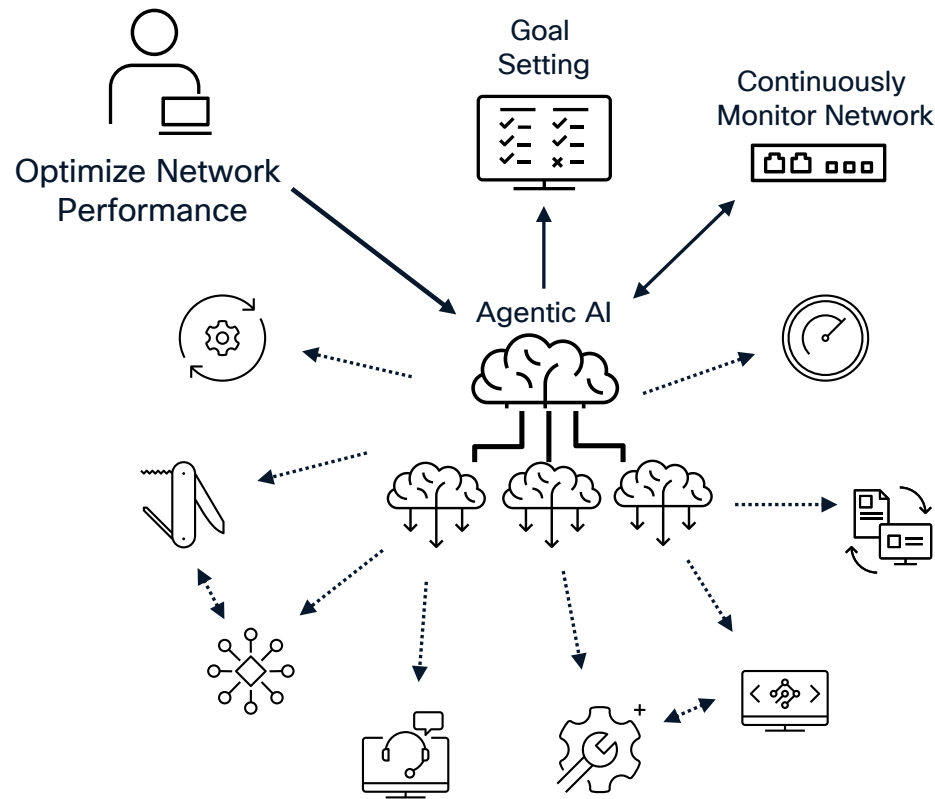
Agentic AI vs Traditional AI

“Traditional” Network AI



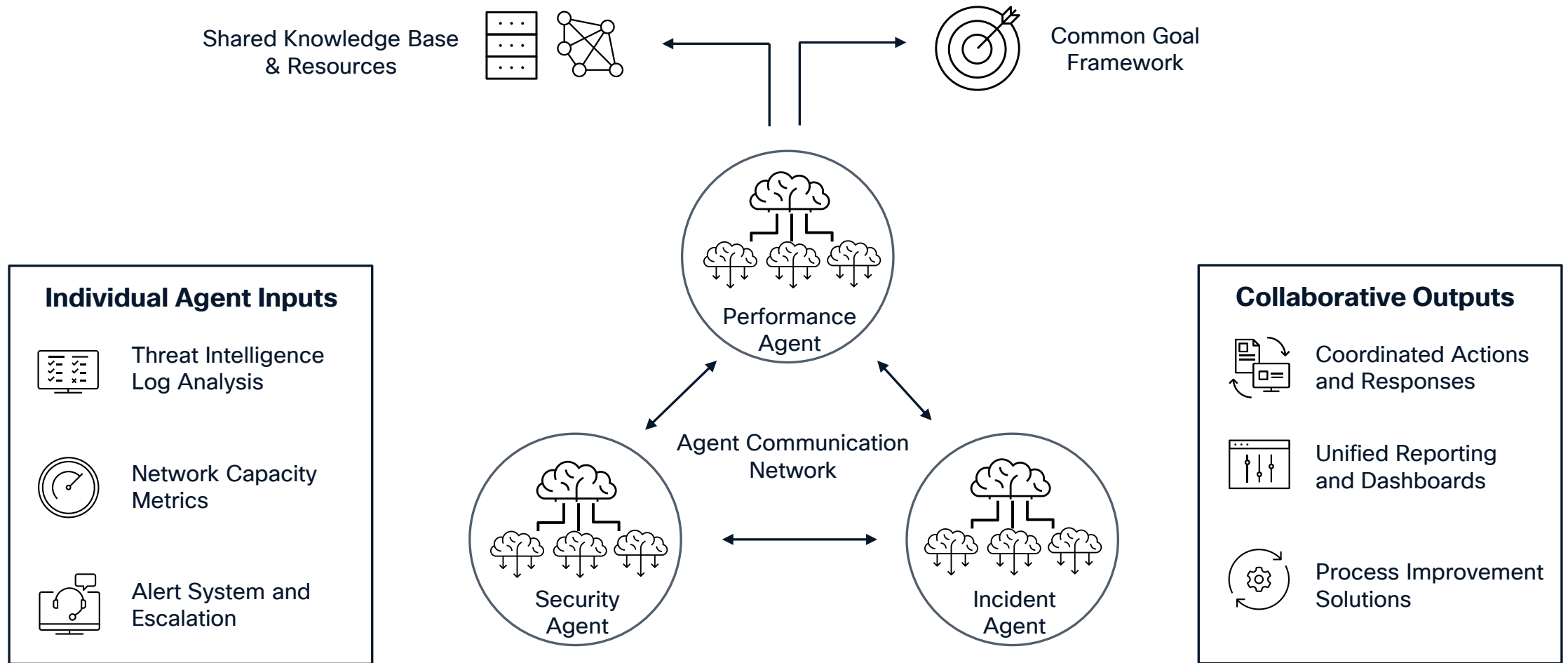
Reactive – Responds to specific asks

Future Agentic Network AI



Proactive – Works independently towards declared goals

Multi-Agent Systems



7 Risks and Security Implications of AI

Data Security and Confidentiality Breaches



Intellectual Property



Bias



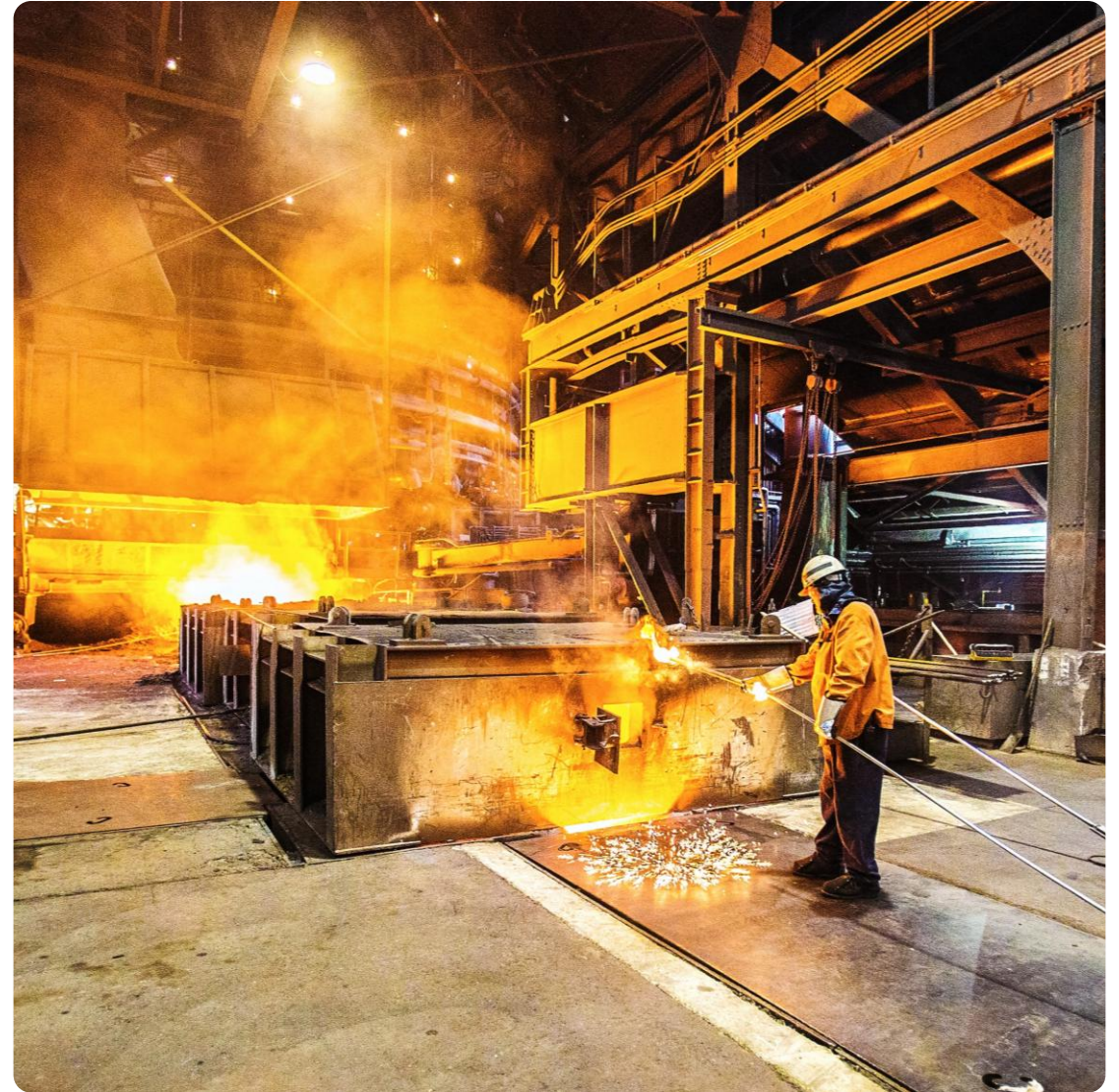
Accuracy and Hallucinations



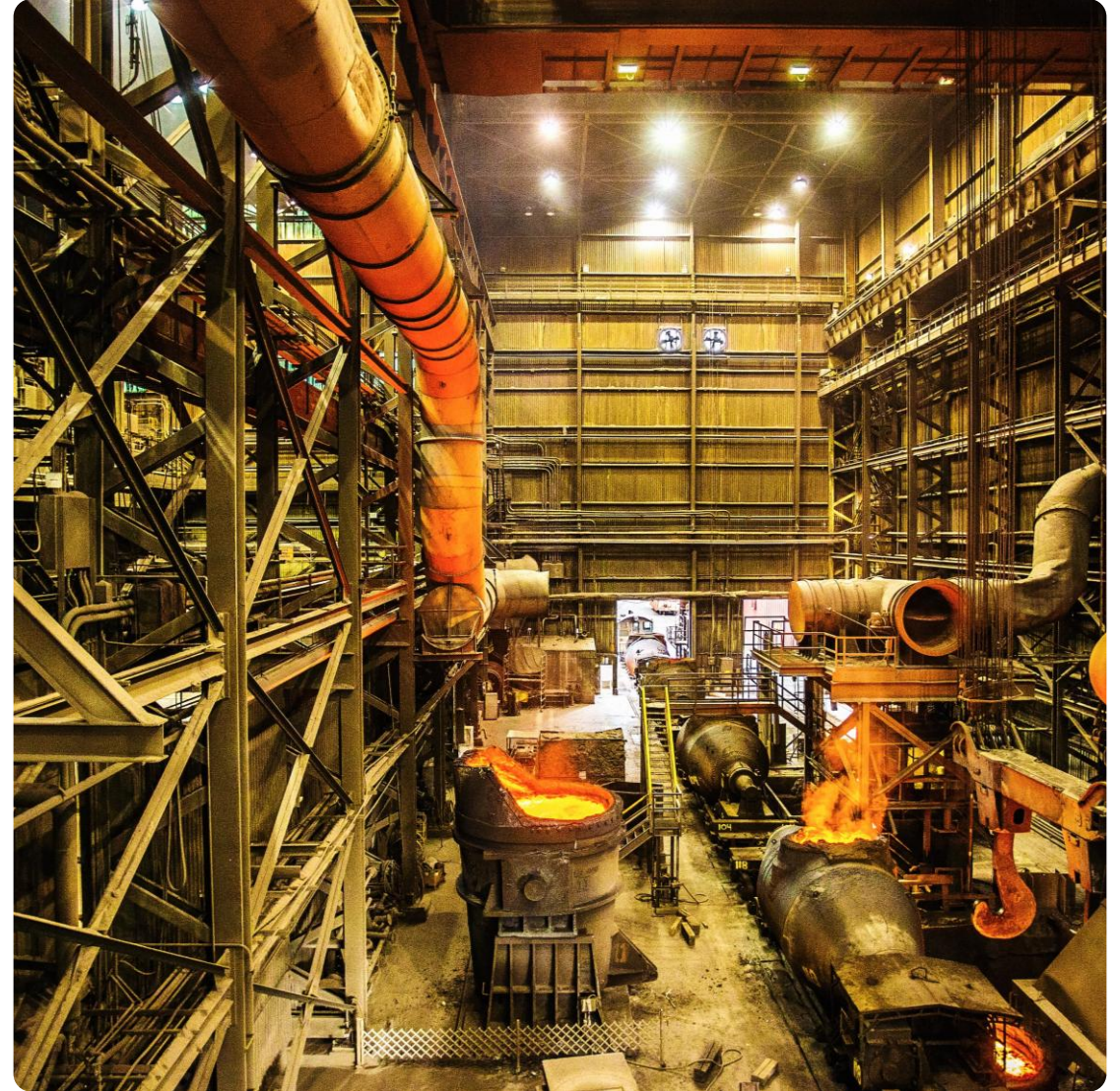
Autonomous AI Agents



Prompt Injection Attacks



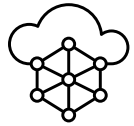
Model Poisoning



AI Risk Mitigation Best Practices

1. Implement Data Classification and AI Usage Policies
2. Establish Expert in the Loop Validation for Critical Operations
3. Use Multiple AI Sources and Cross Validation
4. Maintain Comprehensive Audit Trails and Version Control
5. Regular AI Security Awareness Training and Prompt Hygiene

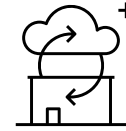
AI Deployment Models



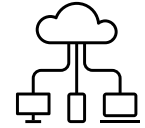
Public LLM



**Private
Cloud LLM**



**Private On-
Premises
LLM**



**Local / Edge
LLM**

Spot the Problem – AIOps Edition!

I'm having a BGP issue on our Cisco ASR 9000 routers. Sites in Dallas and Phoenix have lost connectivity to our MPLS backbone. Here's the current BGP config from our main router in Chicago:

```
router bgp 65001
  bgp router-id 10.50.1.1
  neighbor 10.50.2.1 remote-as 65002
  neighbor 10.50.3.1 remote-as 65003
  neighbor 172.16.100.5 remote-as 100
  address-family ipv4 unicast
    neighbor 10.50.2.1 activate
    neighbor 10.50.2.1 route-map CUSTOMER_IN in
    neighbor 10.50.3.1 activate
    neighbor 172.16.100.5 activate
    neighbor 172.16.100.5 send-community
```

The show bgp summary shows neighbors are up but we're not receiving expected routes from AS 65002. What could be wrong and how do I fix this quickly?



Spot the Problem – AIOps Edition!

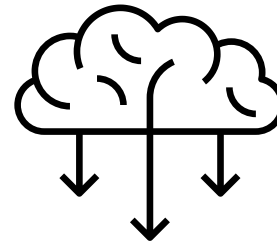
Based on your BGP configuration, I can see several potential issues:

1. **Route-map filtering:** Your neighbor 10.50.2.1 has an inbound route-map 'CUSTOMER_IN' that might be filtering routes. Check: ``show route-map CUSTOMER_IN``
2. **Missing address-family activation:** I notice neighbor 172.16.100.5 is activated but neighbors 10.50.2.1 and 10.50.3.1 might have issues with their address-family configuration.
3. **BGP synchronization issue:** Try disabling BGP synchronization with the command ``no synchronization`` under the BGP process. This is often the cause of route advertisement problems.
4. **Route reflection:** Since you're using AS 65001, you might need to configure route reflection. Use ``bgp cluster-id 1`` and mark your Chicago router as a route reflector with ``neighbor 10.50.2.1 route-reflector-client``.

Quick fix commands to try:

```
router bgp 65001
  no synchronization
  bgp cluster-id 1
  neighbor 10.50.2.1 route-reflector-client
  neighbor 10.50.3.1 route-reflector-client
```

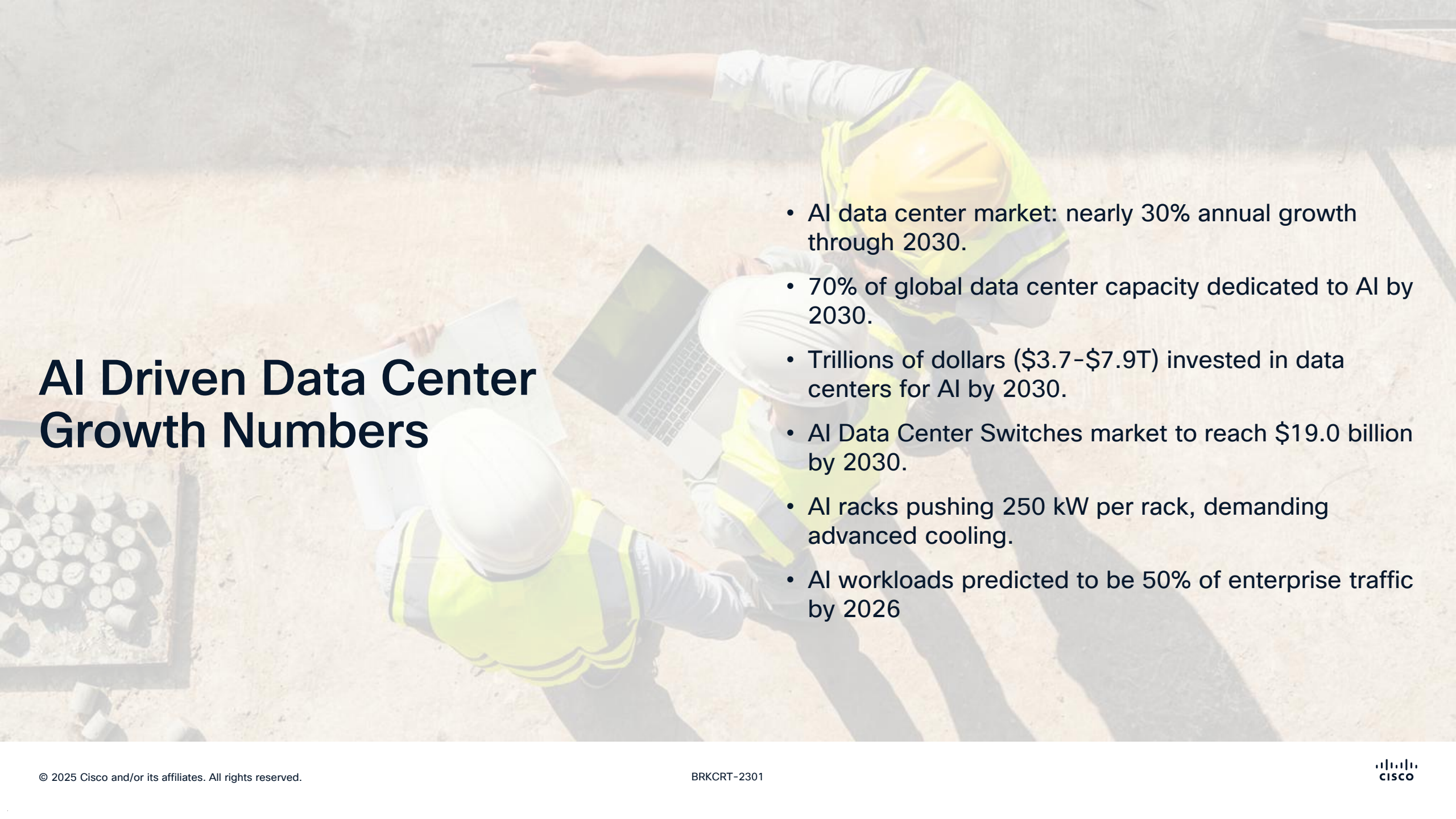
This should restore your routing within 2-3 minutes."



Security Risks

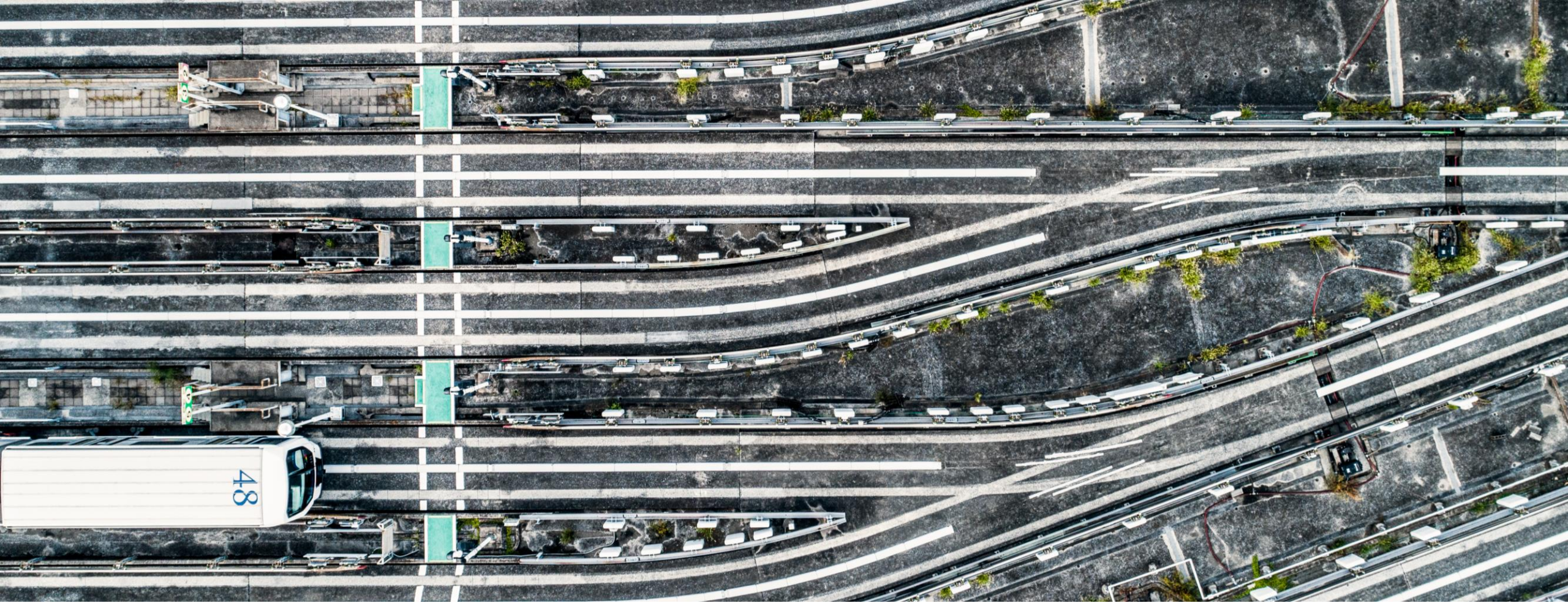
1. Data Exposure
2. Hallucination Risk
3. Incorrect Technical Advice
4. Operational Risk
5. Compliance Violation
6. Future AI Training

AI's impact and requirements on infrastructure

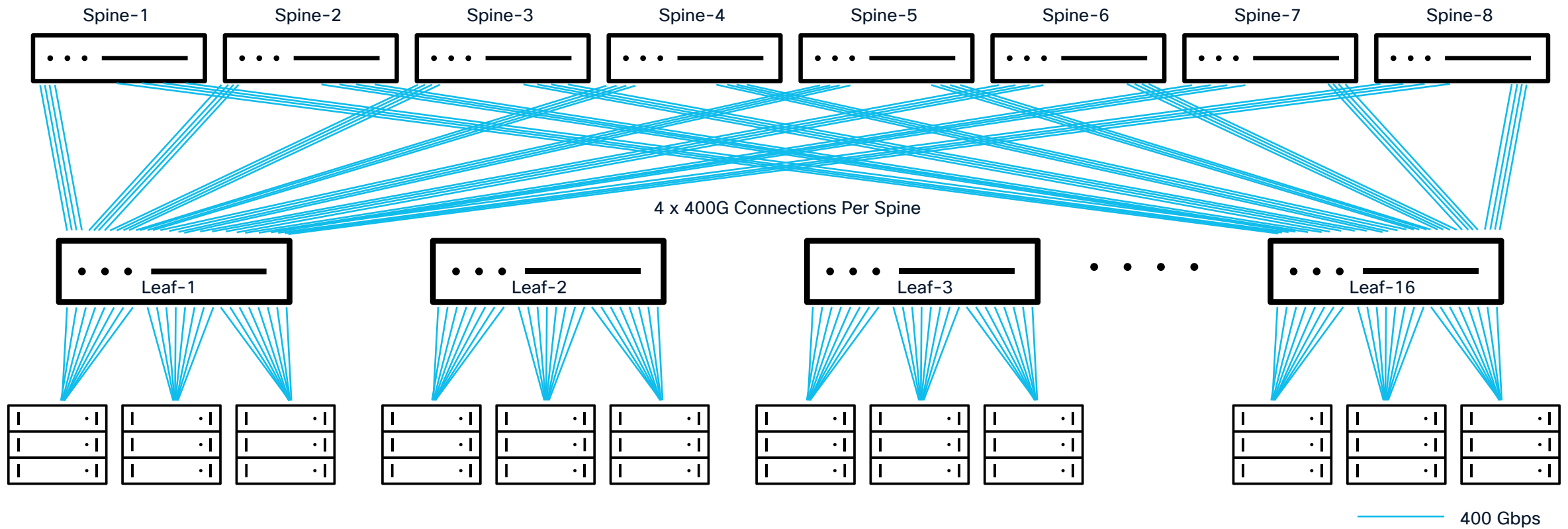


AI Driven Data Center Growth Numbers

- AI data center market: nearly 30% annual growth through 2030.
- 70% of global data center capacity dedicated to AI by 2030.
- Trillions of dollars (\$3.7–\$7.9T) invested in data centers for AI by 2030.
- AI Data Center Switches market to reach \$19.0 billion by 2030.
- AI racks pushing 250 kW per rack, demanding advanced cooling.
- AI workloads predicted to be 50% of enterprise traffic by 2026

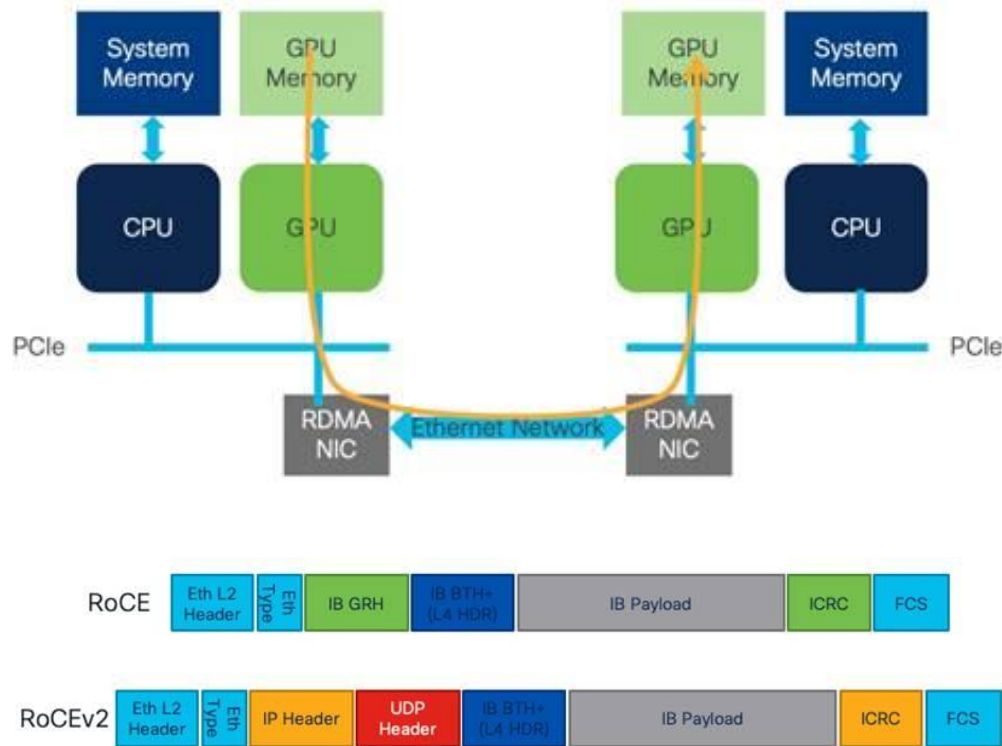


Network Requirements for AI Workloads

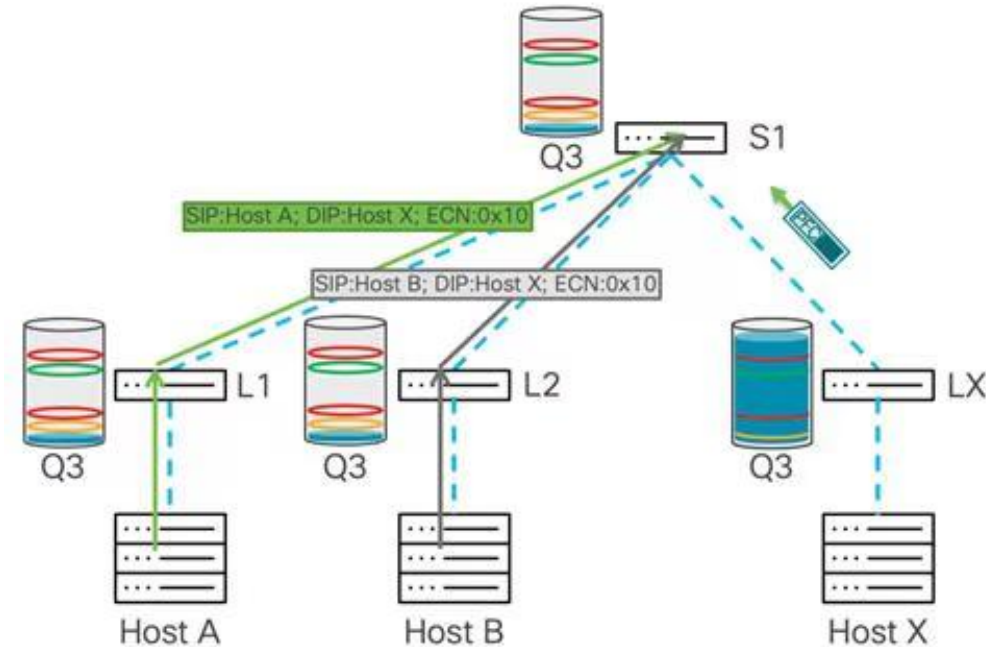


Bandwidth and Latency

RDMA / RoCE



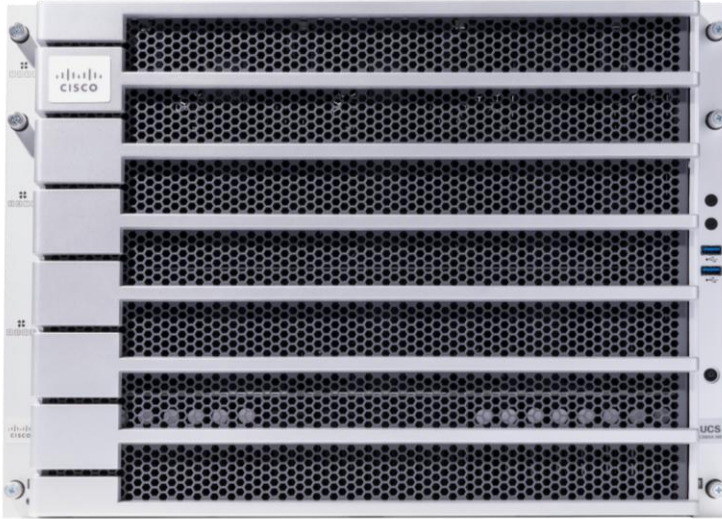
PFC / ECN



RDMA, RoCE, PFC, and ECN

Other AI Networking Considerations

- East-West Traffic Dominance
- Burstiness
- High-Density Cabling
- Advanced Telemetry



Cisco UCS C885A M8

- High density 8U rack server designed for AI.
- Support up to 8 NVIDIA HGX H100/200, or AMD Instinct MI300X/MI350X.
- NVIDIA ConnectX-7 or BlueField-3 SuperNICs and DPUs.
- Dual 4th or 5th Gen AMD EPYC CPUs

What about compute requirements?

Developing Agentic AI solutions

AI Assisted Coding

Using AI to help developers write, understand, and debug code

The screenshot displays a code editor with a file named `device_info_collector.py`. The code is a Python script designed to collect software version and uptime from Cisco network devices using the `pyATS` library and the `rich` library for tabular output. The script includes a `collect_device_info` function that takes a `pyATS` device object as input and returns a dictionary containing the device's name, IP address, and version.

```
1  #!/usr/bin/env python3
2  """
3  Script to collect software version and uptime from Cisco network devices.
4  Uses pyATS for device connectivity and Rich for tabular output.
5
6  Usage:
7  ... python device_info_collector.py <testbed_file>
8  """
9
10 import sys
11 import argparse
12 from pyats.topology import loader
13 from rich.console import Console
14 from rich.table import Table
15
16 def collect_device_info(device):
17     """
18     Connect to a device and collect software version and uptime.
19
20     Args:
21         device: pyATS device object
22
23     Returns:
24         dict: Dictionary containing device information
25     """
26     device_info = {
27         'name': device.name,
28         'ip': device.connections.cli.ip,
29         'version': 'N/A',
30     }
```

On the right side of the editor, a chat interface is open, providing instructions and suggestions for the script. The chat messages include:

- "I need a simple ad-hoc script that will connect to my Cisco network devices using SSH and CLI to collect the current software versions running and uptime. I'd like to then print the data out in a nice tabular format."
- "Please use the pyATS library to connect to devices, and collect the data. The pyATS testbed file that contains the inventory will be provided as a command line argument."
- "Use the Rich library to print out the data. Include the following columns in the table:"

 - Device Name
 - IP Address
 - Software Version
 - Uptime

- "I'll help you create a script that connects to Cisco devices using pyATS and displays the information in a nice table"

The chat interface also shows a summary of changes: "> 1 file changed +133 -1" with buttons for "Keep" and "Undo". Below this, there is a section for adding context, including file names, extensions, and commands, and a dropdown menu for the AI agent, currently set to "Claude Sonnet 4.5".

AI Assisted Coding

```
1 #!/usr/bin/env python3
2 →|
3 This is a simple Python script that will ask users
4
This is a simple Python script that will ask users for their name and age,
and then greet them with their name and age.

name = input("What is your name? ")
age = input("What is your age? ")

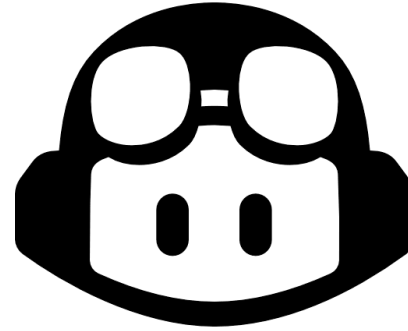
print(f"Hello {name}, you are {age} years old!")
```

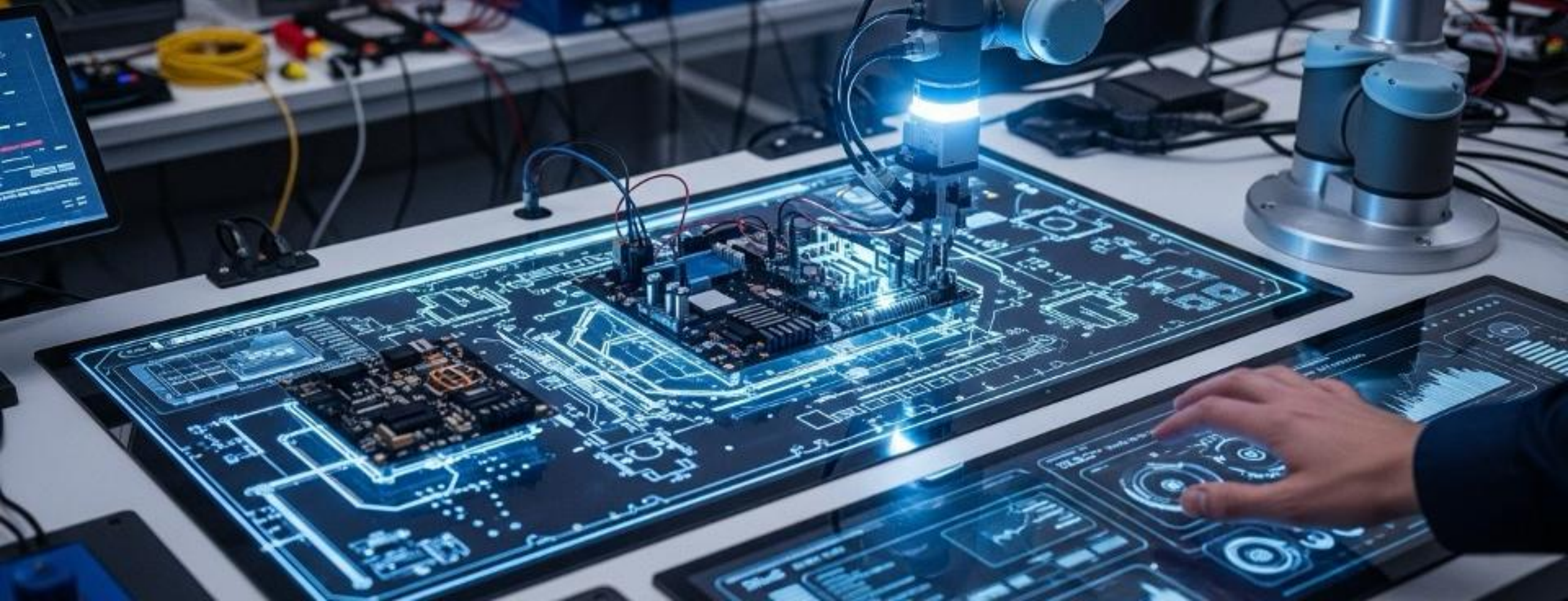
Using AI to help developers write, understand, and debug code

AI Assisted Coding

```
1  #!/usr/bin/env python3
2
3  # This function configures OSPF on an IOS XE device
4  |
```

AI Assisted Coding Tools and Options





Rapid Prototyping

Development Example: Conversational Agent

Development Example: Conversational Agent

[Code Available on GitHub](#)

The Basic Info

- Written in Python, using the LangChain, LangGraph, and Rich libraries
- Uses ollama as inference software
- Uses gpt-oss model (configurable)
- System Prompt create persona as “everyone’s friend”

```
🎉 AI Chat Assistant

Welcome to the Conversational Agent!

🤖 I'm here to help answer your questions and have a conversation.
💡 Just type your message and press Enter.
📝 Type 'exit', 'quit', or 'bye' to end the conversation.

You: Which is better, chocolate or strawberry milk?
🤖 Agent Response
Oh, the age-old milk showdown! 🥛🌟

Chocolate milk: It's the "smooth operator" of the dairy world. Think of it
as the latte of the liquid kingdom—rich, comforting, and perfect for
post-workout refuel (the protein + carbs combo is a win). It's got that
deep, velvety vibe and can even double as a dessert when you dunk a cookie
in it. 🍪

Strawberry milk: Picture a bright, sunny smoothie that's all about that
sweet, fruity punch. It's lighter, a bit tangy, and the color alone can turn
a gloomy day into a picnic in your glass. Plus, it's a great way to sneak in
some antioxidants—berries for the win! 🍓

So, who's better? It depends on your mood and what you're craving:

• Need a cozy, energy-boosting sip? Go chocolate.
• Want a fresh, fruit-filled pick-me-up? Go strawberry.

If you're feeling adventurous, why not blend the two? A "choco-berry"
milkshake could be the ultimate party trick—just don't forget the whipped
cream! 🎉

You:
```

Development Example: Conversational Agent

[Code Available on GitHub](#)

4 Steps to Creating a Basic Agent

1. Read in system prompt
2. Create model object to send chats for responses
3. Establish “memory” to support ongoing conversations
4. Create agent using the three components

```
# Read system prompt from file
with open("system_prompt.md", "r") as f:
    SYSTEM_PROMPT = f.read()

# Configure model details
from langchain.chat_models import init_chat_model

model = init_chat_model(
    model= "gpt-oss:20b",
    model_provider="ollama",
    temperature=0.7
)

# Setup memory for conversation
from langgraph.checkpoint.memory import InMemorySaver

checkpointner = InMemorySaver()
config = {"configurable": {"thread_id": "conversation_1"}}

# Create agent
from langchain.agents import create_agent
agent = create_agent(
    system_prompt=SYSTEM_PROMPT,
    model=model,
    checkpointner=checkpointner,
)
```

Development Example: Conversational Agent

Code Available on GitHub

5 Steps to a Conversation

1. Create thread/conversation identifier
2. Request user prompt
3. Send prompt to agent
4. Render response to user
5. Repeat

```
# Unique identifier for the conversation thread
config = {"configurable": {"thread_id": "conversation_1"}}

# Conversational loop
while True:
    # Get user input with Rich prompt.
    console.print() # Add some spacing
    user_input = Prompt.ask("[bold green]You[/bold green]")

    # Send message to the agent and get response
    response = agent.invoke(
        {"messages": [{"role": "user", "content": user_input}]},
        config=config,)

    # Create a Markdown object to properly render markdown formatting
    markdown_response = Markdown(response["messages"][-1].content)

    # Create a panel for the agent's response with markdown
    response_panel = Panel(
        markdown_response,
        title="🤖 Agent Response", title_align="left",
        border_style="cyan",
        padding=(0, 1), width=None,)

    console.print(response_panel)
```

Note: Code for Rich Console setup removed from slide, see full code for complete example

Development Example: Conversational Agent

Code Available on GitHub

Final Thoughts

- Frameworks, patterns, and tooling for agent development changes rapidly
- Extend Agents by giving them “tools”
- Experiment with different model’s and system prompts
- Checkout LangChain QuickStart and Docs
- Explore ollama.com and on GitHub

You: Which milk would you drink?



Agent Response

🌟 If I Could Sip, I’d Pick...

Chocolate Milk - because it’s the smooth operator that can turn a boring day into a “choco-lutely” delicious adventure! 🍫🦄

Why?

- 1 Rich, creamy flavor that feels like a hug from a marshmallow-topped blanket.
- 2 Protein & calcium - perfect for “post-AI-workout” refuel (yes, even virtual muscles need a boost!).
- 3 Classic combo - it’s the secret ingredient in the best chocolate-chip cookie dough (and the best cookie-dough-milkshake too!).
- 4 Pun-tastic: “I’m choco-lutely in love with this milk!” 🍫❤️

But hey, if you’re feeling adventurous, a strawberry-chocolate fusion (aka a “choco-straw” smoothie) would be the ultimate double-whammy: sweet, fruity, and a little bit of chocolatey swagger. 🍓🍫

So, if I could drink, I’d go for chocolate, maybe with a splash of strawberry for that extra “berry” good vibe. What about you? Do you lean toward the classic chocolate, the bright strawberry, or the daring mix-up? Let’s chat about your favorite milk-based adventures! 🌈🥛💬

You:

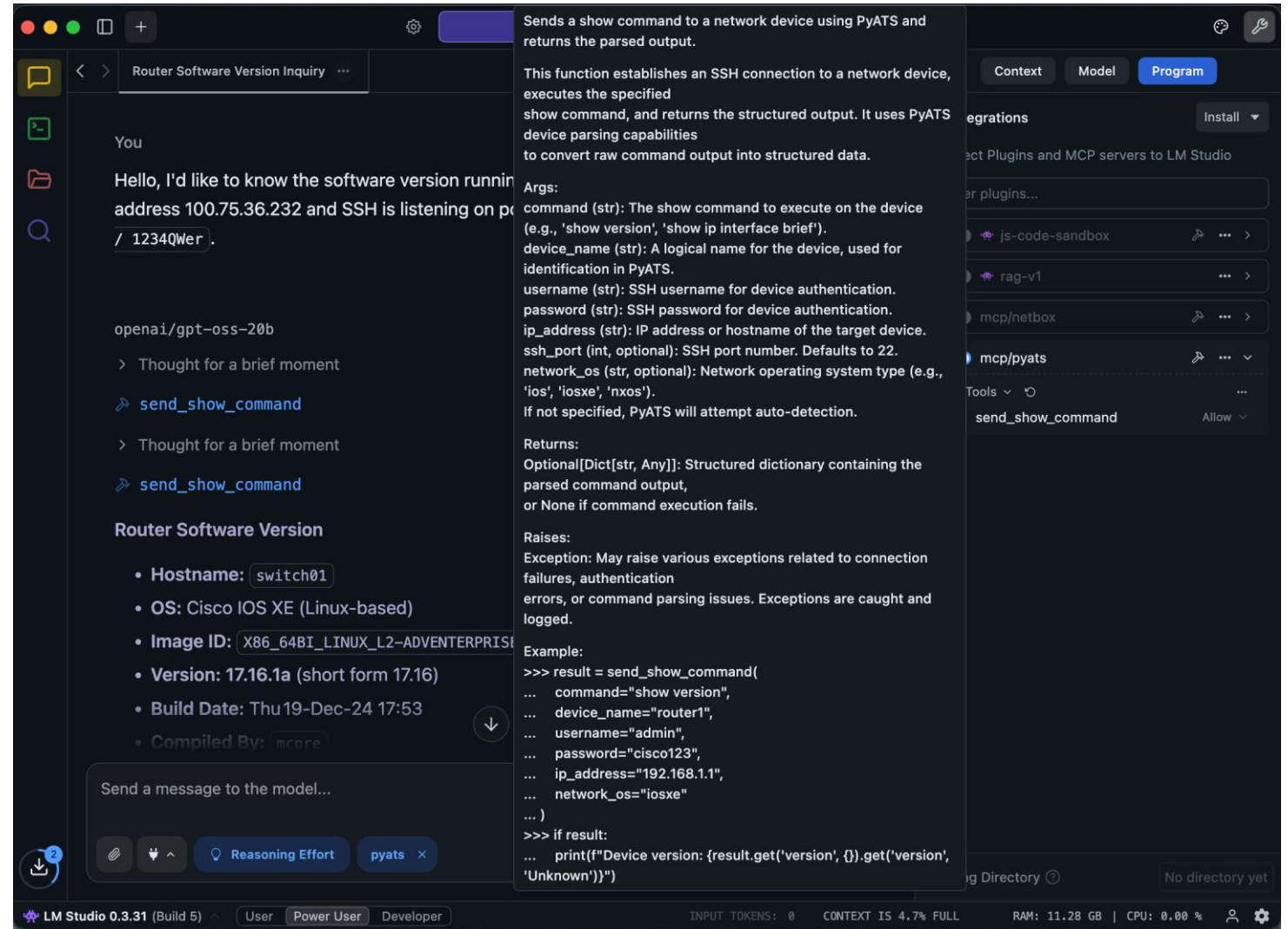
Development Example: MCP Server

Development Example: MCP Server

[Code Available on GitHub](#)

The Basic Info

- Allows AI Agents to send show commands to network devices
- Written in Python, using the FastMCP 2.0 and pyATS libraries
- Uses http transport, so could be deployed as remote MCP server
- Should work with any MCP client, but tested with LM Studio
- Single tool “`send_show_command()`”



Development Example: MCP Server

[Code Available on GitHub](#)

4 Steps to Developing an MCP Server/Tool

1. Initialize the FastMCP object
2. Write a decorated function that performs the intended action
3. Provide detailed doc string for how to use the tool, inputs, outputs, errors, etc
4. Leverage type hints on arguments and function return

Important Detail: The doc string and type hints are what tell the LLM how to use the tool!

```
mcp = FastMCP()

@mcp.tool()
def send_show_command(command: str, device_name: str,
username: str, password: str, ip_address: str,
ssh_port: int = 22, network_os: Optional[str] = "ios",
) -> Optional[Dict[str, Any]]:
    """
    Sends a show command to a network device using PyATS and returns...

    This function establishes an SSH connection to a network device, ...
    show command, and returns the structured output. It uses PyATS ...
    to convert raw command output into structured data.

    Args:
    command (str): The show command to execute on the device
    device_name (str): A logical name for the device
    .
    .

    Returns:
    Optional[Dict[str, Any]]: dictionary containing the parsed output

    Raises:
    Exception: May raise various exceptions related to failures

    Example:
    >>> result = send_show_command(command="show version", ...
    ... username="admin", password="cisco123", ip_address="192.168.1.1",
    ... network_os="iosxe")
    """
```

Note: Code edited for slide and presentation. See GitHub link for full and complete code

Development Example: MCP Server

[Code Available on GitHub](#)

Running the MCP Server

- Servers run local (stdio) or remote (http*)
- Local agents are started by the MCP Client
- Remote agents are run as independent software

```
fastmcp run --transport http --port 8002 mcp_pyats_01.py
```

FASTMCP

FastMCP 2.13.0.2

 Server name: FastMCP-081e

 Transport: HTTP

 Server URL: <http://127.0.0.1:8002/mcp>

 Docs: <https://gofastmcp.com>

 Hosting: <https://fastmcp.cloud>

```
[10/29/25 08:54:35] INFO      Starting MCP server 'FastMCP-081e' with transport
                        'http' on server.py:2050 http://127.0.0.1:8002/mcp
INFO:      Started server process [5012]
INFO:      Waiting for application startup.
INFO:      Application startup complete.
```

Code Available on GitHub

Connecting MCP Server from Client

- The screenshot shows the LM Studio application window. The top bar displays the model name "openai/gpt-oss-20b". Below it, the "Router Software Version Inquiry" tab is active, showing the "mcp.json" file. A message at the top states: "Saving this file will reload your MCP servers." with "Save" and "Discard" buttons. The main editor area contains the following JSON code:

```
1 {  
2   "mcpServers": {  
3     "netbox": {  
4       "command": "uv",  
5       "args": [  
6         "--directory",  
7         "/Users/hapresto/code/netbox-mcp-server",  
8         "run",  
9         "server.py"  
10      ],  
11      "env": {  
12        "NETBOX_URL": "http://100.75.36.232:4080/",  
13        "NETBOX_TOKEN": "0123456789012345678901234567"  
14      }  
15    },  
16    "pyats": {  
17      "url": "http://localhost:8002/mcp"  
18    }  
19  }  
20 }
```

On the right side, the "Integrations" panel is visible under the "Program" tab. It lists several plugins with toggle switches and icons:

 - js-code-sandbox
 - rag-v1
 - mcp/netbox
 - mcp/pyats (selected)

Below the plugin list, there are sections for "Tools" and "Commands". Under "Commands", the "send_show_command" command is listed with an "Allow" button.

Development Example: MCP Server

[Code Available on GitHub](#)

Using the MCP Server

- Write a prompt that should trigger the tool use
- Check the tool call arguments and results
- Troubleshoot and tune the doc_string and system prompts to best use the tools

Hello, I'd like to know the software version running on `router01`. It is reachable at IP address 100.75.36.232 and SSH is listening on port 4002. The credentials are `netadmin / 1234QWer`.

openai/gpt-oss-20b

> Thought for a brief moment

🔗 `send_show_command`

mcp/pyats >

> Thought for a brief moment

🔗 `send_show_command`

mcp/pyats ▾

▾ Arguments

command: `show version`

device_name: `router01`

username: `netadmin`

password: `1234QWer`

ip_address: `100.75.36.232`

ssh_port: `4002`

network_os: `ios`

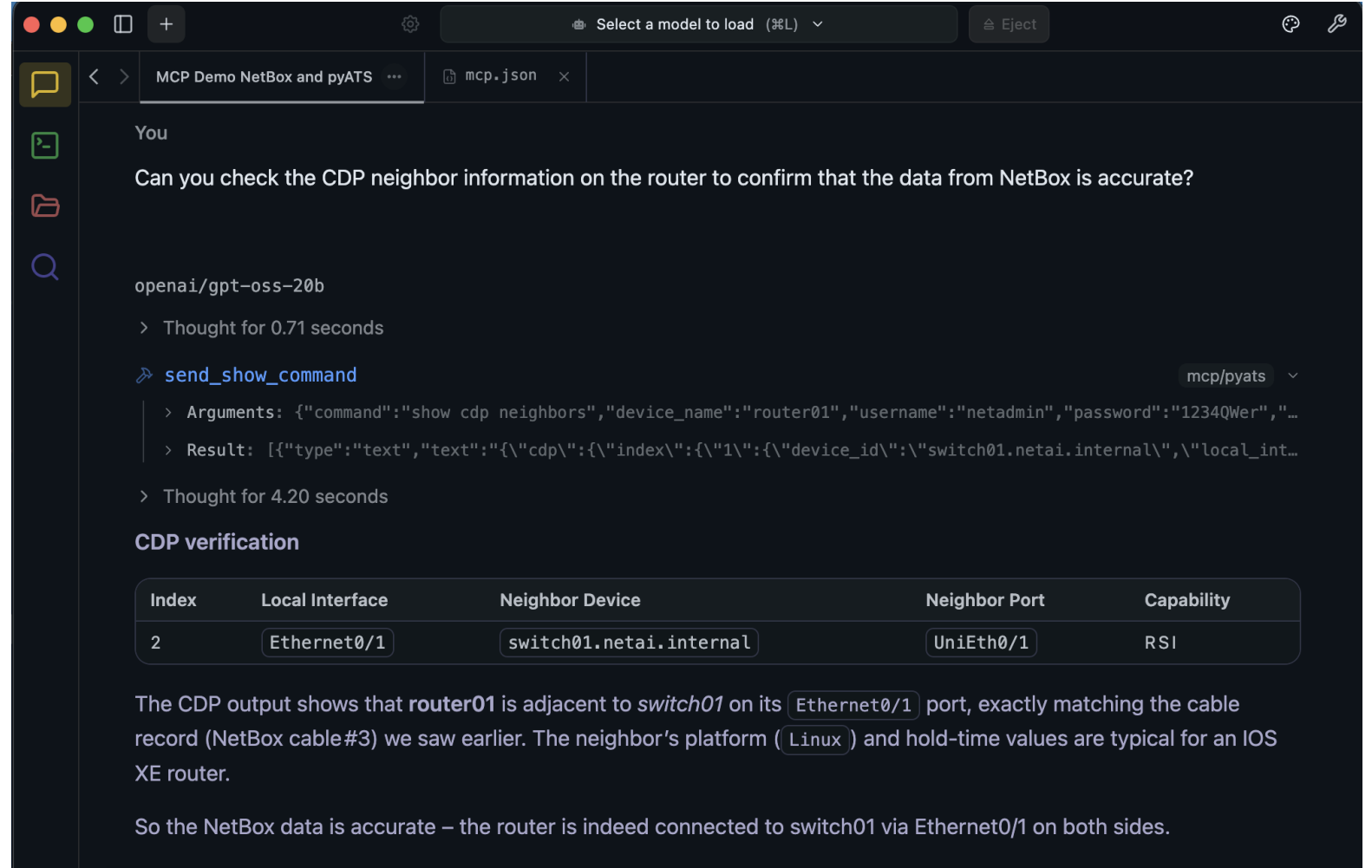
> Result: [{"type": "text", "text": "{\"version\": \"17.16\", \"platform\": \"Linux\"}"}]

Development Example: MCP Server

Code Available on GitHub

Final Thoughts

- MCP Server development is not that different from any other network automation
- Good code doc strings and typing will make your tools work better
 - Consider context size of the LLM and Clients in use. Easy to fill a context window just with tool info
- Connect your NetDevOps scripts and tools to AI with FastMCP



You

Can you check the CDP neighbor information on the router to confirm that the data from NetBox is accurate?

openai/gpt-oss-20b

> Thought for 0.71 seconds

`send_show_command` mcp/pyats

> Arguments: {"command": "show cdp neighbors", "device_name": "router01", "username": "netadmin", "password": "1234QWer", "...}

> Result: [{"type": "text", "text": "{\n \"cdp\": {\n \"index\": {\n \"1\": {\n \"device_id\": \"switch01.netai.internal\", \"local_int...\"}

> Thought for 4.20 seconds

CDP verification

Index	Local Interface	Neighbor Device	Neighbor Port	Capability
2	Ethernet0/1	switch01.netai.internal	UniEth0/1	RSI

The CDP output shows that **router01** is adjacent to **switch01** on its **Ethernet0/1** port, exactly matching the cable record (NetBox cable#3) we saw earlier. The neighbor's platform (**Linux**) and hold-time values are typical for an IOS XE router.

So the NetBox data is accurate – the router is indeed connected to switch01 via Ethernet0/1 on both sides.

Closing

Important AI Vocabulary Review

- Model
- Token
- Context
- Hallucination
- Training
- Inference
- Prompt
- Prompt Engineering
- Temperature

Agenda Review

- 01 Why is AI in Cisco certifications?
- 02 AI basics and fundamentals
- 03 Risks and security implications of AI
- 04 AI's impact and requirements on infrastructure
- 05 Developing Agentic AI solutions

Hank Preston

Stay in touch:

Webex/Email: hapresto@cisco.com

Blogs: <https://blogs.cisco.com/author/hankpreston>



Complete your session evaluations



Complete a minimum of 4 session surveys and the Overall Event Survey to claim a Cisco Live T-Shirt.



Earn up to 800 points by completing all surveys and climb the Cisco Live Challenge leaderboard.



Level up and earn exclusive prizes!



Complete your surveys in the Cisco Live Events app.

Continue your education



Visit the Cisco Stand for related demos



Book your one-on-one Meet the Expert meeting



Attend the interactive education with Capture the Flag, and Walk-in Labs



Visit the On-Demand Library for more sessions at www.CiscoLive.com/on-demand

Thank You

