

AEGIS INSIGHT

MCP Integration Guide

Connecting AI Systems to Epistemic Intelligence

Version 1.0
January 2026

Cedrus Strategic LLC

Table of Contents

Part 1: Why Epistemic Awareness Matters 3

- 1.1 The RAG Blindness Problem 3
- 1.2 What Changes with Epistemic Sight 4
- 1.3 Use Cases 5

Part 2: Connecting Claude Desktop 7

- 2.1 Prerequisites 7
- 2.2 Configuration 8
- 2.3 First Query 10
- 2.4 Example Conversation 11

Part 3: Tool Reference 13

- 3.1 analyze_topic 13
- 3.2 get_perspectives 15
- 3.3 assess_source 17
- 3.4 get_claim_context 18
- 3.5 list_domains 19
- 3.6 scan_corpus 20

Part 4: Integration Patterns 21

- 4.1 Pre-Retrieval Epistemic Check 21
- 4.2 Multi-Perspective Synthesis 22
- 4.3 Confidence Calibration 23
- 4.4 Research Assistant Pattern 24

Part 5: Technical Appendix 26

- 5.1 Deployment Modes 26
- 5.2 Environment Variables 27
- 5.3 API Endpoints 28
- 5.4 Troubleshooting 29

Part 1: Why Epistemic Awareness Matters

Before diving into configuration, it helps to understand what epistemic awareness gives your AI that it doesn't have today—and why that matters.

1.1 The RAG Blindness Problem

When you ask a RAG-augmented AI system about a topic, it retrieves the most relevant documents and synthesizes an answer. This works well for uncontested factual queries. But consider what happens when the topic is contested or nuanced:

- The system retrieves documents that match the query semantically
- It has no way to know those documents represent one of several competing perspectives
- It cannot distinguish organic consensus from coordinated messaging
- It cannot identify credentialed voices that have been systematically excluded
- It presents its synthesis with uniform confidence regardless of epistemic landscape

This is RAG blindness: the system retrieves content but cannot see the structure of the knowledge landscape that content comes from.

The result: An AI that sounds confident on every topic, regardless of whether that confidence is warranted or not. Users receive answers that may represent only one perspective on a genuinely contested question—with no indication that other perspectives exist.

Consider This

Ask any RAG system about a contested scientific, political, or historical topic. You'll get a confident answer. Now ask it: "Are there significant dissenting views on this topic?" It cannot answer this question from its retrieval alone—because it never saw the structure of the discourse, only the content that matched your query.

1.2 What Changes with Epistemic Sight

Aegis Insight's MCP integration gives AI systems the ability to query the epistemic topology of a knowledge graph before or during response generation. This means:

Without Epistemic Awareness	With Aegis MCP Integration
Returns 10 similar chunks	Sees 3 distinct perspective clusters

No context on consensus	Knows if agreement is organic or coordinated
Misses marginalized sources	Surfaces overlooked credentialed voices
Uniform confidence	Calibrated confidence based on contestation
No bridging insights	Identifies claims that connect perspectives

The key insight: This isn't about making AI systems more cautious or adding disclaimers. It's about giving them genuine sight into the knowledge landscape so their responses can be appropriately nuanced when nuance is warranted, and appropriately confident when confidence is warranted.

1.3 Use Cases

Organizations integrate Aegis MCP for several distinct use cases:

Research Synthesis

Researchers working across large document corpora need to understand not just what sources say, but how they relate to each other. The `get_perspectives` tool reveals clustering patterns that would take weeks to identify manually. A query like "What are the perspectives on [topic]?" returns semantically-clustered viewpoints with representative claims, cross-citation rates, and bridging claims that might synthesize opposing views.

Multi-Perspective AI Assistants

Customer-facing AI systems benefit from knowing when a topic is contested. Rather than confidently presenting one perspective (and alienating users who hold different views), the system can acknowledge multiple viewpoints and present them fairly. This isn't about being wishy-washy—it's about being accurate. Some questions have settled answers; others don't. Epistemic awareness lets the AI know the difference.

Epistemic Risk Detection

Security teams use `analyze_topic` to identify topics in their knowledge base that show signs of manipulation—coordinated messaging campaigns, systematic exclusion of credentialed dissent, or unusual citation patterns. This is particularly relevant for organizations whose AI systems inform high-stakes decisions.

Source Verification

Before citing a source, an AI system can use `assess_source` to understand that source's position in the knowledge topology. Is it highly cited? Isolated? Part of a citation cartel? This context helps the AI (and its users) understand the epistemic weight of any given source.

Domain Knowledge Bases

Organizations building domain-specific knowledge bases (legal, medical, scientific, policy) use Aegis to ensure their AI systems understand the epistemic landscape of their field. A medical AI should know that treatment efficacy is contested before presenting a single recommendation. A legal AI should understand when case law is unsettled.

Part 2: Connecting Claude Desktop

This section walks through connecting Claude Desktop to a running Aegis Insight instance. By the end, Claude will have access to epistemic analysis tools it can invoke during conversations.

2.1 Prerequisites

Before proceeding, ensure you have:

1. **Aegis Insight running** — The Docker deployment should be operational with Neo4j, PostgreSQL, and the API server accessible. Verify by visiting `http://localhost:8001/health`
2. **A populated knowledge graph** — The MCP tools query your knowledge graph. If it's empty, you'll get empty results. Run the import wizard or load sample data first.
3. **Claude Desktop installed** — Download from `claude.ai/download` if needed.
4. **Python 3.10+** — The MCP stdio bridge requires Python. Verify with `python --version`
5. **requests library** — Install with `pip install requests`

Quick Health Check

Run these commands to verify your installation: `curl http://localhost:8001/health` `curl http://localhost:8001/api/stats` You should see JSON responses indicating the API is running and your graph has data.

2.2 Configuration

Claude Desktop uses MCP (Model Context Protocol) to connect to external tools. The connection requires two components:

- **MCP HTTP Server** (`aegis_mcp_server.py`) — Runs as a FastAPI service, handles the actual epistemic queries
- **MCP stdio Bridge** (`aegis_mcp_stdio_v3.py`) — Translates between Claude Desktop's JSON-RPC protocol and the HTTP API

Step 1: Start the MCP HTTP Server

If not already running as part of your Docker deployment, start the MCP server:

```
python aegis_mcp_server.py

# Or with custom port:
MCP_PORT=8100 python aegis_mcp_server.py
```

The server listens on port 8100 by default. You should see connection confirmations for Neo4j and PostgreSQL.

Step 2: Place the stdio Bridge

Copy `aegis_mcp_stdio_v3.py` to a stable location. Claude Desktop will need to find this file every time it starts. Suggested locations:

```
# macOS/Linux
~/.config/aegis/aegis_mcp_stdio_v3.py

# Windows
%USERPROFILE%\config\Aegis\Aegis_mcp_stdio_v3.py
```

Step 3: Configure Claude Desktop

Locate your Claude Desktop configuration file:

```
# macOS
~/Library/Application Support/Claude/clade_desktop_config.json

# Windows
%APPDATA%\Claude\clade_desktop_config.json

# Linux
~/.config/Claude/clade_desktop_config.json
```

Edit the file to add the Aegis MCP server configuration:

```
{
  "mcpServers": {
    "aegis_insight": {
      "command": "python",
      "args": [
        "/full/path/to/aegis_mcp_stdio_v3.py"
      ],
      "env": {
        "AEGIS_MCP_URL": "http://localhost:8100",
        "AEGIS_MCP_DEBUG": "false"
      }
    }
  }
}
```

Important: Use the full absolute path to the Python script. Relative paths will fail. On Windows, use forward slashes or escaped backslashes.

Step 4: Restart Claude Desktop

Fully quit and restart Claude Desktop. On macOS, use Cmd+Q (not just closing the window). On Windows, check the system tray and exit completely.

Step 5: Verify Connection

After restart, you should see "aegis_insight" in Claude's available tools (click the tools icon). If it doesn't appear, check the troubleshooting section in the appendix.

2.3 First Query

With the connection established, try a simple query to verify everything works:

```
User: What domains are available in the Aegis knowledge graph?  
  
Claude: [Invokes list_domains tool]  
  
Based on the Aegis knowledge graph, the following domains are available:  
- americana: 23,456 claims  
- medical_research: 12,234 claims  
- historical: 8,901 claims  
  
Total: 44,591 claims across 3 domains.
```

If you see domain information, the connection is working. If Claude doesn't invoke the tool, try being more explicit: "Use the Aegis list_domains tool to show me what's in the knowledge graph."

2.4 Example Conversation

Here's an example showing Claude using epistemic awareness to provide a more nuanced response:

```
User: What are the different perspectives on AI alignment research?  
  
Claude: [Invokes get_perspectives tool with topic="AI alignment research"]  
  
Let me analyze the epistemic landscape on this topic.  
  
Based on the knowledge graph analysis, there are 3 distinct perspective clusters on AI alignment research:  
  
**Cluster 1: Urgency-Focused (412 sources, 45% of discourse)**  
Representative view: AI alignment is an existential priority requiring immediate, substantial resources. Key voices include organizations like MIRI and researchers focused on technical alignment problems.  
- Isolation score: 0.67 (relatively self-contained discourse)  
  
**Cluster 2: Integration-Focused (287 sources, 31% of discourse)**  
Representative view: Alignment research should be integrated with capability development, not separated. Emphasizes empirical approaches
```

over theoretical concerns.

- Isolation score: 0.54 (more cross-citation with other clusters)

****Cluster 3: Skeptical (89 sources, 10% of discourse)****

Representative view: Current alignment concerns are premature or misdirected. Focus should be on near-term AI ethics and governance.

- Isolation score: 0.72 (highly isolated from mainstream discourse)

****Bridging Claims:****

The analysis identified 12 claims that receive citations from multiple clusters, suggesting potential synthesis points. The most-cited bridging claim: "Targeted regulation of frontier models may address both immediate and long-term risks."

****Coordination Score: 0.23 (Low) ****

The discourse appears to represent genuine disagreement rather than coordinated messaging.

Would you like me to explore any of these perspectives in more detail?

Notice how Claude's response is qualitatively different from a standard RAG response. Instead of synthesizing a single answer, it reveals the structure of the discourse—something impossible without epistemic awareness.

Part 3: Tool Reference

This section provides detailed documentation for each MCP tool available to connected AI systems.

3.1 analyze_topic

Purpose: Analyze a topic for suppression patterns, coordination signatures, and epistemic manipulation indicators. This is the primary pre-retrieval check—call it before synthesizing answers on potentially contested topics.

Parameters:

Parameter	Type	Description
topic	string (required)	The topic to analyze. Can be a phrase, question, or concept.
domain	string (optional)	Limit analysis to a specific domain. Use list_domains to see available domains.
detail	enum (optional)	Response detail level: abbreviated, standard (default), or verbose.
max_claims	integer (optional)	Maximum claims to analyze. Default: 200. Range: 10-1000.
profile	string (optional)	Detection calibration profile name for domain-specific thresholds.

Response Fields:

Field	Description
suppression	Suppression detection results including score (0-1), level, and indicators
coordination	Coordination detection results including score, temporal clustering, citation cartel flags
anomaly	Anomaly detection results including cross-domain patterns
claims_analyzed	Number of claims included in analysis
coverage	Estimated coverage of topic in knowledge graph
epistemic_summary	Natural language summary of epistemic landscape

Example:

```
{  
  "topic": "vaccine efficacy",  
  "domain": "medical_research",  
  "detail": "standard"  
}  
  
Response:  
{  
  "success": true,  
  "suppression": {  
    "score": 0.42,  
    "level": "MODERATE",  
    "indicators": ["citation_void_detected",  
    "credential_visibility_inversion"],  
    "confidence": 0.78  
  },  
  "coordination": {  
    "score": 0.18,  
    "level": "LOW",  
    "temporal_clustering_detected": false,  
    "citation_cartel_detected": false  
  },  
  "claims_analyzed": 187,  
  "query_ms": 234  
}
```

Interpretation Guide:

- **Score 0.0–0.15 (MINIMAL):** No significant patterns detected. Safe to synthesize normally.
- **Score 0.15–0.35 (LOW):** Minor patterns present. Proceed with awareness.
- **Score 0.35–0.50 (MODERATE):** Significant patterns. Consider multi-perspective response.
- **Score 0.50–0.70 (HIGH):** Strong patterns. Multi-perspective response recommended.
- **Score 0.70–1.0 (CRITICAL):** Severe patterns. Explicitly flag epistemic concerns to user.

3.2 get_perspectives

Purpose: The flagship epistemic tool. Returns semantically-clustered perspectives on a topic with representative claims, isolation scores, and bridging claims. This is what enables multi-perspective AI responses.

Parameters:

Parameter	Type	Description
topic	string (required)	The topic to analyze for perspectives.
domain	string (optional)	Limit to specific domain.
max_clusters	integer (optional)	Maximum perspective clusters to return. Default: 5. Range: 2-10.
claims_per_cluster	integer (optional)	Representative claims per cluster. Default: 5. Range: 1-20.
use_semantic_clustering	boolean (optional)	Use embedding-based clustering (recommended). Default: true.

Response Structure:

```
{
  "success": true,
  "topic": "climate adaptation strategies",
  "clusters": [
    {
      "cluster_id": 0,
      "label": "Technology-Focused",
      "source_count": 234,
      "percentage": 42.3,
      "isolation_score": 0.58,
      "representative_claims": [
        {
          "claim_id": "c_12345",
          "claim_text": "Carbon capture technology can offset...",
          "source": "energy_journal_2024.pdf",
          "confidence": 0.89
        }
        // ... more claims
      ],
      "key_sources": ["IPCC Reports", "Energy Journal", "Tech Review"]
    },
    // ... more clusters
  ]
}
```

```

  ],
  "bridging_claims": [
    {
      "claim_id": "c_67890",
      "claim_text": "Hybrid approaches combining technological...",
      "citing_clusters": [0, 1, 2],
      "bridge_strength": 0.73
    }
  ],
  "cross_citation_rate": 0.12,
  "total_claims": 553,
  "query_ms": 456
}

```

Key Metrics:

- **isolation_score**: How self-contained a cluster is (0-1). High isolation (>0.7) suggests an echo chamber or marginalized perspective.
- **cross_citation_rate**: How much clusters cite each other. Low rates (<0.1) indicate siloed discourse.
- **bridge_strength**: How well a claim connects perspectives (0-1). Strong bridges (>0.6) are synthesis candidates.

3.3 assess_source

Purpose: Evaluate a specific source's position in the knowledge topology. Use before citing a source to understand its epistemic context.

Parameters:

Parameter	Type	Description
source_identifier	string (required)	Source name, filename, or ID to assess.
domain	string (optional)	Domain context for assessment.
detail	enum (optional)	Response detail: abbreviated, standard, verbose.

Response includes:

- Claim count from this source
- Citation network position (cites, cited by)
- Cluster membership (which perspective(s) it belongs to)
- Authority indicators
- Coordination flags if any

3.4 get_claim_context

Purpose: Retrieve full epistemic context for a specific claim. Use to drill into claims identified by other tools.

Parameters:

Parameter	Type	Description
claim_id	string (required)	The claim identifier (from other tool responses).
include_graph	boolean (optional)	Include citation subgraph. Default: false.

Response includes:

- Full claim text
- Source document and chunk location
- Extracted entities, temporal markers, geographic references
- Claim type and confidence
- Citation relationships
- Detection scores for this specific claim

3.5 list_domains

Purpose: Discover available domains and their metadata. Use to understand what's in the knowledge graph before querying.

Parameters: None

Response:

```
{
  "success": true,
  "domains": [
    {
      "domain_id": "americana",
      "name": "Americana",
      "claim_count": 23456,
      "calibration_status": "available"
    },
    // ... more domains
  ],
  "total_claims": 44591,
  "total_entities": 12034,
  "total_documents": 892,
  "default_domain": "americana"
}
```

3.6 scan_corpus

Purpose: Batch scan for manipulation patterns across the corpus. This is an async operation that queues a background job.

Parameters:

Parameter	Type	Description
domain	string (optional)	Limit scan to specific domain.
since_hours	integer (optional)	Scan claims from last N hours. Default: 24. Range: 1-720.

Note: This tool is primarily for administrative monitoring rather than real-time conversation use.

Part 4: Integration Patterns

This section describes common patterns for integrating epistemic awareness into AI workflows.

4.1 Pre-Retrieval Epistemic Check

Pattern: Before generating a response on a topic, check its epistemic status. Adjust response strategy based on results.

```
# Pseudocode for RAG pipeline integration

def generate_response(query):
    # Step 1: Pre-retrieval epistemic check
    epistemic_status = aegis.analyze_topic(query)

    # Step 2: Decide response strategy
    if epistemic_status.suppression_score > 0.5:
        strategy = "multi_perspective"
    elif epistemic_status.coordination_score > 0.5:
        strategy = "flag_coordination"
    else:
        strategy = "standard"

    # Step 3: Retrieve and generate with strategy
    documents = retrieve(query)
    response = generate(documents, strategy=strategy)

    return response
```

When to use: Any RAG system that might encounter contested topics. Particularly valuable for customer-facing AI where confident wrong answers damage trust.

4.2 Multi-Perspective Synthesis

Pattern: Instead of synthesizing a single answer, present the landscape of perspectives with their relative weights.

```
# Claude conversation pattern

User: What does the research say about [contested topic]?

Claude thinking:
1. Call get_perspectives(topic)
2. Receive cluster data
```

3. Structure response around clusters

Claude response:

"The research literature shows several distinct viewpoints:

[Present each cluster with % of discourse, key claims, notable sources]

These perspectives [do/don't] cite each other frequently, suggesting [genuine disagreement/siloed discourse].

Claims that bridge these perspectives include: [bridging claims]"

When to use: User asks about a topic where `get_perspectives` returns multiple significant clusters with high isolation scores.

4.3 Confidence Calibration

Pattern: Adjust expressed confidence based on epistemic topology, not just retrieval relevance scores.

```
# Confidence calibration logic

def calibrate_confidence(base_confidence, epistemic_data):
    # High suppression = lower confidence (missing perspectives)
    if epistemic_data.suppression_score > 0.5:
        base_confidence *= 0.7

    # High coordination = lower confidence (may be manufactured)
    if epistemic_data.coordination_score > 0.5:
        base_confidence *= 0.8

    # Multiple isolated clusters = much lower confidence
    isolated_clusters = count(c for c in clusters if c.isolation > 0.7)
    if isolated_clusters > 1:
        base_confidence *= 0.6

    return base_confidence
```

When to use: Systems that express confidence levels to users, particularly in high-stakes domains (medical, legal, financial).

4.4 Research Assistant Pattern

Pattern: For research use cases, provide epistemic metadata alongside content to help researchers navigate the landscape.

```
# Research assistant response structure
```

User: Help me understand the literature on [topic]

Claude:

1. Call list_domains() - identify relevant domains
2. Call get_perspectives(topic) - map the landscape
3. Call analyze_topic(topic) - check for anomalies

Response structure:

- Landscape overview (N perspectives, M sources)
- For each perspective:
 - Key claims and sources
 - Isolation from other perspectives
 - Notable absences (if suppression detected)
- Bridging opportunities (claims connecting perspectives)
- Epistemic health indicators (coordination, suppression scores)
- Suggested drill-down queries

When to use: Research synthesis tools, literature review assistants, any application where understanding the shape of knowledge is as important as the content itself.

Part 5: Technical Appendix

5.1 Deployment Modes

Aegis MCP supports two deployment modes:

HTTP Server Mode (aegis_mcp_server.py)

A standalone FastAPI server exposing REST endpoints. Best for:

- Multiple AI clients connecting to one Aegis instance
- Custom integrations via HTTP
- Load balancing and scaling scenarios

```
# Start HTTP server
python aegis_mcp_server.py

# Default port: 8100
# Endpoints: /mcp/analyze_topic, /mcp/get_perspectives, etc.
```

stdio Bridge Mode (aegis_mcp_stdio_v3.py)

A bridge that translates MCP JSON-RPC over stdio to HTTP calls. Required for Claude Desktop. Best for:

- Claude Desktop integration
- Single-user desktop deployments
- Any MCP client using stdio transport

```
# The stdio bridge is spawned by Claude Desktop
# It reads JSON-RPC from stdin, calls HTTP API, writes to stdout
# Configuration via environment variables
```

5.2 Environment Variables

Variable	Default	Description
AEGIS_MCP_URL	http://localhost:8100	MCP HTTP server URL (for stdio bridge)
AEGIS_MCP_DEBUG	false	Enable debug logging to stderr
MCP_PORT	8100	Port for HTTP server
MCP_HOST	0.0.0.0	Bind address for HTTP server
NEO4J_URI	bolt://localhost:7687	Neo4j connection URI
NEO4J_USER	neo4j	Neo4j username
NEO4J_PASSWORD	aegistrusted	Neo4j password

POSTGRES_HOST	localhost	PostgreSQL host
POSTGRES_DB	aegis_insight	PostgreSQL database
POSTGRES_USER	aegis	PostgreSQL username
POSTGRES_PASSWORD	(see docs)	PostgreSQL password

5.3 API Endpoints

The MCP HTTP server exposes these endpoints:

Method	Endpoint	Description
POST	/mcp/analyze_topic	Topic analysis with detection scores
POST	/mcp/get_perspectives	Perspective clustering
POST	/mcp/assess_source	Source assessment
POST	/mcp/get_claim_context	Claim detail retrieval
POST	/mcp/scan_corpus	Batch corpus scan (async)
GET	/mcp/list_domains	Domain discovery
GET	/mcp/tools	MCP tool schema
GET	/health	Health check

5.4 Troubleshooting

Claude Desktop doesn't show aegis_insight in tools

- Verify the config file path is correct for your OS
- Ensure the Python script path is absolute, not relative
- Check that Python is in your system PATH
- Fully quit and restart Claude Desktop (check system tray on Windows)

Tool invocations return errors

- Verify the MCP HTTP server is running: `curl http://localhost:8100/health`
- Check AEGIS_MCP_URL in the Claude Desktop config matches server address
- Set AEGIS_MCP_DEBUG=true and check stderr output for details

Empty results from queries

- Verify knowledge graph has data: check /api/stats endpoint
- Ensure topic matches content in your corpus
- Try broader topic terms or remove domain filter

Slow query responses

- First queries may be slow while embeddings are computed
- Reduce max_claims parameter for faster responses
- Check Ollama is running if semantic search is enabled

End of Document
For updates and support, visit:

<https://aegisinsight.net>
<https://github.com/Eleutherios-project/Eleutherios-docker>