

MCP went stateless. The control point moved. The record did not follow it.

The 2026-07-28 specification removed the session handshake and mirrored the method and object name into HTTP headers. A gateway can now route, rate-limit and audit agent traffic without opening the request body. That is a better home for a control. It is not, on its own, a record.

What actually changed

Removed or deprecated

`initialize / initialized`
`Mcp-Session-Id`
`HTTP GET stream endpoint`
`Last-Event-ID` resumability
`ping, logging/setLevel`
`Roots, Sampling, Logging (dep.)`

Suggested logging migration is `stderr` or `OpenTelemetry`. A twelve-month minimum deprecation window is now policy.

Now required, and where

`MCP-Protocol-Version`
every POST to the endpoint
`Mcp-Method`
all requests
`Mcp-Name`
`tools/call, resources/read, prompts/get`
`Mcp-Param-*` server opt-in

The body stays the source of truth. Headers are written by the client from that body, so they are asserted, not independently observed.

What the edge can and cannot do

Cheap at the gateway now

- › Deny or allow by method and by tool name
- › Rate-limit and route by tenant
- › Audit the call without parsing the body
- › Refuse regardless of what the agent believes
- › Enforce on parameters a server chose to expose

Still not available up there

- › The query, path, diff or message body
- › Any parameter the server did not annotate
- › Guaranteed client identity: `clientInfo` is a SHOULD
- › Assurance the header matches the body
- › What the call actually did downstream

Two sources of truth in one request

Servers that process the body MUST reject any request whose headers disagree with it, returning 400 and JSON-RPC error -32020 `HeaderMismatch`. The stated reason is to prevent a load balancer routing on the header value while the server executes on the body value. The matching intermediary control is weaker: a note says gateways enforcing on mirrored headers SHOULD verify that the declared protocol version requires that validation. Nothing will fail loudly if yours does not.

Six moves, in the order to do them

- 01 Establish which revision you are actually on.
Every compliant request POST carries MCP-Protocol-Version. Header requirements for notification POSTs are undefined in this revision, so build the audit around requests. If nothing in your estate can answer the question, that is the first finding.
- 02 Write down what the gateway is for, before you build it.
It enforces on method, object name and whichever parameters a server chose to expose. Record that boundary so nobody later assumes it covers the payload.
- 03 Test mismatch handling on both sides.
Server MUST reject header-body disagreement with -32020. Intermediaries are only advised to refuse enforcement on unvalidated protocol versions, so nothing will remind you. Test both.
- 04 Mint your own correlation identifier and carry it end to end.
Use the OpenTelemetry trace context conventions the spec documents in _meta: traceparent, tracestate, baggage. Do not build the story on clientInfo, which clients are only advised to send.
- 05 Assume retries and make effects idempotent.
The core protocol defines no idempotency key, and a broken stream now loses the in-flight request. Anything that moves money, provisions infrastructure, sends a message or deletes something needs a key held by the tool.
- 06 Ask what the record is for.
Traffic logs answer what was called. Accountability answers who authorised it, against which policy, and what resulted. The second set does not appear because you enabled the first.

The signal you are looking for

IF THIS IS TRUE	THEN THIS IS THE GAP
The gateway logs tool names and call counts	You have traffic telemetry, not an accountability record. No authorising party, no policy, no outcome.
Correlation relied on the server-minted session ID	That spine is withdrawn. Anything reconstructable by accident before is now deliberate or absent.
Policy lives inside the agent runtime or the MCP server	The control sits inside the thing it governs. It holds only while that thing cooperates.
Nobody owns idempotency for agent-triggered effects	The protocol supplies no key, and redelivery is gone. A dropped connection no longer resolves itself.
Controls depend on protocol features	They inherit the protocol's release cadence. A twelve-month deprecation window is policy now, but it is still a clock.

Where the legal obligation actually sits

Article 12 of the EU AI Act requires automatic recording of events over the lifetime of the system, at a level of traceability appropriate to the intended purpose. Two qualifications before quoting it at a delivery team: it bites only on systems classified high-risk under Article 6 and Annex III, and the duty falls on providers (Articles 16 and 19) and deployers (Article 26(6)), not on a protocol. Where it applies, nothing MCP stopped keeping makes it smaller. Outside that perimeter the same question arrives through contracts and incident reviews instead.

SOURCES

MCP specification 2026-07-28: Key Changes, and Streamable HTTP transport, 28 July 2026 · Soria Parra and Delimarsky, The 2026-07-28 Specification Release Candidate, MCP blog, 21 May 2026 · Van Gent and Blount, Google Developers Blog, 5 August 2026 · Gorgiev and Dalla Piazza, Stateless MCP, Equixly, 5 August 2026 · Regulation (EU) 2024/1689, Articles 12, 16, 19 and 26 · AWS, Bedrock AgentCore Runtime, 9 April 2026. All verified 11 August 2026.