MCP SERVER | NO CODE | CLOUD HOSTED

RSS Feed Parser MCP

Turns messy feeds into clean, usable data.

The RSS Feed Parser takes raw XML or Atom feed data from any blog, news site, or podcast and converts it into clean, structured JSON objects. It extracts all critical metadata—titles, links, publication dates, authors, categories, and full content snippets—without needing to scrape messy HTML. This lets your agent process reliable, machine-readable content feeds instantly.

A+ Quality Score 100/100

rss | atom | content-aggregation | web-scraping | json-parsing | news-monitoring

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/rss-feed-parser

SCAN TO VIEW

The connectivity layer between AI and the world's software.

Vinkius sits between AI and every application. All communication passes through Vinkius Cloud via the Model Context Protocol (MCP) — with governance, observability, and security at every layer.

Your AI Connections Run Through Vinkius Cloud

The world's largest managed MCP catalog

Vinkius is the connectivity layer where AI connects to the software your business already runs. We handle the hosting, the security, the credentials, the uptime — you get agents that actually do things.

We operate the world's largest managed MCP catalog. Major SaaS platforms, CRMs, databases, and cloud providers — running, monitored, production-ready. This MCP server is hosted and maintained by the Vinkius Cloud for AI Agents.

> *The agent doesn't manage credentials, doesn't manage uptime, doesn't manage security. Vinkius does.*
>
> — Architecture principle

Four Pillars of the Vinkius Runtime

01 — Security by design

Credentials stay encrypted at rest via AES-256. The AI agent never touches raw keys — they're injected into a sandboxed V8 isolate at runtime. Actions are logged, and connections have an emergency kill switch.

02 — Built on MCP Fusion

This MCP server was built with **MCP Fusion**, the open-source framework (Apache 2.0) that powers the entire Vinkius catalog. Schema-as-firewall strips undeclared fields, compiled PII redaction runs at zero overhead, and cryptographic lockfiles produce git-diffable audit trails.

03 — Deterministic observability

Eight immutable metrics per endpoint: request volume, p95 latency, error rate, active connections, cost attribution. A live payload feed logs every tool call with mutation detection.

04 — Autonomous operations

Servers are deployed, monitored, and patched autonomously. New capabilities and security patches ship weekly. Zero-downtime deployments ensure continuous availability across all managed MCP servers.

AES-256	Ed25519	24h TTL	V8 Isolate
Encryption at rest	PKI vault signatures	Ephemeral session keys	Sandboxed execution

One Token. Instant Access.

Every MCP server on Vinkius is accessed through a **Connection Token**. Tokens are generated in the cloud dashboard and produce a unique MCP endpoint URL. Paste this URL into any MCP-compatible client — no SDK required.

A single token can serve **multiple AI clients simultaneously**, or you can issue separate tokens per client for granular access control. Each token tracks its own request count, last activity timestamp, and can be individually enabled or revoked.

MCP ENDPOINT

```
https://edge.vinkius.com/{token}/mcp
```

Claude · Cursor · VS Code · Windsurf · Grok · Gemini

Security Is the Architecture

Security in Vinkius is not a feature — it's the foundation of the runtime. The gateway enforces multiple independent protection layers between AI agents and third-party APIs.

01 — Ed25519 PKI Vault

Every workspace has an Ed25519 Master Key. Session keys are generated ephemerally (24h TTL) and signed by the Master Key. Credentials never leave the vault boundary.

02 — V8 Isolate Sandboxing

Tool code runs inside isolated-vm V8 isolates with 64 MB memory caps and per-request timeouts. No filesystem access, no network access except through the SSRF-guarded fetch bridge.

03 — SSRF Guard

All outbound HTTP requests are DNS-resolved and validated before execution. Private IP ranges (10.x, 172.16-31.x, 192.168.x, AWS metadata 169.254.x) are blocked at the network layer.

04 — DLP & PII Redaction

A ResponseGuard pipeline intercepts every tool response. Configurable redaction patterns strip sensitive fields (emails, SSNs, card numbers) before data reaches the AI agent.

05 — Cryptographic Audit Trail

Every request is signed into a SHA-256 hash chain with Ed25519 signatures. Events form a tamper-proof, SIEM-exportable forensic record.

06 — Honeytoken Trap System

Phantom credentials are injected into isolated environments. If a honeytoken is used outside Vinkius infrastructure, the server is quarantined instantly.

Emergency Kill Switch

EU AI Act Art. 14(1) Compliant

The kill switch is an **emergency halt** mechanism — not a simple toggle. When triggered, it executes three actions atomically:

01 — Server deactivated

The MCP server is immediately taken offline across the entire cluster.

02 — All tokens revoked

Every connection token is invalidated. Total lockout — reconnection blocked until new tokens are issued.

03 — WebSocket connections killed

Active connections terminated via Redis pubsub broadcast. Propagates to every runtime node in the cluster.

Full Visibility. Zero Guesswork.

The Vinkius cloud dashboard includes a full MCP Governance suite — real-time analytics and security controls for production AI operations.

Control Plane

KPI dashboard with request volume, latency, success rate, token consumption, and AI-generated operational briefings.

Agent Activity

Which AI clients are connecting, how often, and what they're doing — real-time session tracking.

FinOps

Cost tracking per tool, payload compression savings, budget optimization signals, and consumption trends.

Tool Health

Slowest and most error-prone tools, with actionable root-cause insights and performance baselines.

Firewall & DLP

PII redaction activity, sensitive data protection counters, and security event timeline.

Incident Log

Error trends, failure rates, status-code breakdowns, and forensic audit trail access.

> Get started at **cloud.vinkius.com** — connect your AI agent in under 60 seconds.

RSS Feed Parser MCP

1 tools available
Cloud-hosted on Vinkius

Content teams spend way too much time manually copying article data from multiple sources. You're supposed to monitor twenty competitor blogs and five industry news sites, but you end up staring at fragmented HTML that breaks every other day.

This MCP changes that. It reads native web feeds—the XML or Atom format—and turns the entire stream into clean JSON objects. This means your agent gets predictable data: titles, links, publication dates, authors, and full content snippets, all packaged neatly for summarization or automated distribution. You don't have to worry about messy DOM traversal; this tool handles the tough parsing work so you can focus on what matters.

Whether it’s an entire blog feed or a podcast channel with enclosures, the data is structured and ready to go. Accessing reliable content feeds like this used to require custom scrapers for every single site. Now, through Vinkius, your agent gets access to one unified parser that handles both RSS 2.0 and Atom formats identically.

Core Capabilities

01 — Extract structured article data

Pass a raw XML feed string and receive a clean JSON array containing titles, links, dates, and content for multiple items.

02 — Handle multiple feed standards

Processes both RSS 2.0 and Atom formats using the same unified structure.

03 — Extract podcast metadata

Identifies audio or video enclosures within a feed, providing URLs, durations, and file sizes.

One Click on Vinkius — From Prompt to Execution

Available at **vinkius.com/mcp/rss-feed-parser** — connect your AI agent in three steps.

01 You supply the raw XML string from the RSS or Atom feed.

02 The MCP processes this stream, extracting all metadata (titles, links, dates, content) into a structured JSON format.

03 Your agent receives the clean JSON object, which is ready for immediate use in workflows like summarization or data storage.

The bottom line is that you trade unpredictable website code for predictable, standardized data objects.

Built For

Content strategists, SEO analysts, and technical product managers use this MCP. They struggle with the manual labor of aggregating content from dozens of external sources daily. The pain point is time—spending hours cleaning up scraped HTML instead of analyzing trends.

Content Strategist

Uses it to automatically collect and analyze articles from competitor blogs, ensuring the agent always has the most current content streams for trend spotting.

SEO Analyst

Connects feeds to pull publication dates and category tags across multiple sites, helping map out comprehensive topic coverage gaps in the market.

Technical Product Manager

Uses it to validate if external data sources (like news aggregators) provide consistent metadata fields before building a new internal reporting dashboard.

What Changes When You Connect

01 Stop dealing with inconsistent HTML. This MCP reads native feed XML/Atom and delivers structured JSON, so your agent never has to worry about broken tags or malformed code.

02 Process entire content streams, not just single articles. You can pass a raw feed string and get up to twenty items returned in one clean batch of data.

03 Capture rich podcast details automatically. The tool extracts enclosure URLs, durations, and file sizes from episode feeds, which is critical for media monitoring.

04 Centralize multiple formats. It treats RSS 2.0 and Atom feeds the same way, unifying disparate content sources into a single, predictable JSON structure.

05 Eliminate manual data entry. Instead of copying titles, links, and dates across dozens of tabs, your agent gets all that metadata in one go.

Real-World Applications

Monitoring Competitor News

A content strategist wants to track five competitors' latest posts. They use the MCP to feed the raw XML for each site, allowing their agent to consolidate titles and publication dates into a single JSON list for weekly reporting.

Building a Podcast Content Index

A marketing team needs all recent episode details (URL, duration, description) from a podcast. The MCP ingests the feed and gives them an array of clean objects containing the necessary enclosure data for their newsletter.

Curating Industry News Briefs

A researcher needs to summarize key takeaways from three industry news aggregators. They run the `parse_rss_feed` tool on all three feeds, receiving a consistent JSON output that can be fed into a summarization model.

Validating Content Source Integrity

A product team needs to ensure an external data partner's feed is reliable. They use the MCP to test various raw XML strings, verifying that all expected metadata fields—like author and category—are consistently present.

Patterns to Avoid

Using generic web scrapers

✗ AVOID

Trying to scrape content directly using basic HTTP requests or a simple HTML parser. This fails when the target site changes its internal structure.

✓ INSTEAD

Pass the raw XML feed string and use `parse_rss_feed`. The MCP is designed specifically for standardized feed formats, making it immune to common website layout breaks.

Handling multiple feeds separately

✗ AVOID

Running separate scripts or workflows for RSS 2.0 feeds versus Atom feeds, leading to inconsistent output handling.

✓ INSTEAD

The MCP unifies both RSS 2.0 and Atom formats into the same JSON structure. You just feed it the source; you get one standardized result.

The Right Fit

Use this MCP if your primary need is to gather structured metadata (titles, links, dates) from multiple content sources that publish via a standard feed format. If you're building a system that consumes information from blogs, news aggregators, or podcasts, this is exactly what you want. You must have the raw XML or Atom URL ready for input.

Don't use this if you are dealing with proprietary data endpoints that don't offer an RSS feed, or if your content sources require login credentials to access. For those cases, a specialized API connector would be better. This MCP focuses purely on transforming standard, publicly available feed XML.

The Daily Pain of Content Aggregation

Think about what it takes today: you have to open one browser tab for the tech news, another for the industry blog, and a third for the competitor's site.

With this MCP, that entire process vanishes. You feed the raw XML string—the source material itself—into your agent using `parse_rss_feed`. What

Then, for every single article, you manually copy the title, paste the link into your spreadsheet, and then click through to get the publication date. If you need to do this across twenty sources, it takes hours of repetitive clicking and copy-pasting.

comes back isn't a spreadsheet; it’s clean JSON containing every title, link, date, author, and content snippet you need, ready to be indexed or summarized automatically.

Parsing Feeds with the RSS Feed Parser

The manual steps that go away are: opening multiple browser tabs, navigating to different article pages, and manually extracting metadata one item at a time. You no longer have to worry about whether the source site changed its HTML structure or if you missed an enclosure URL.

Now, feeding data is a single step. You send the feed XML, and you get reliable, structured JSON back. It’s predictable, standardized content for your agent.

RSS Feed Parser: 1 Tool Available

Use the single available tool to convert raw XML feed streams into standardized JSON data objects ready for immediate AI processing.

#	TOOL	DESCRIPTION
01	`parse_rss_feed`	Converts raw RSS 2.0 or Atom XML strings into structured JSON, extracting titles, links, dates, categories, and full content snippets.

See It in Action

Real prompts you can use once this MCP is connected to your AI agent through Vinkius Cloud.

U Monitor TechCrunch's RSS feed and give me the latest 5 articles with titles and publish dates.

5 articles extracted: title, link, pubDate, author, and content snippet for each.

U Parse our company blog feed and extract all articles tagged 'product-update' from the last 30 days.

8 articles found with 'product-update' category in the last 30 days.

U Get the latest episode URLs from this podcast RSS feed for our newsletter.

3 episodes with audio URLs, durations, and descriptions extracted.

Frequently Asked Questions

01 **How do I use the RSS Feed Parser with podcast feeds?**

The MCP handles podcast data by extracting enclosure metadata. You simply pass the feed XML; it returns a structured JSON object that includes URLs, durations, and descriptions for every episode.

02 **Does the RSS Feed Parser support Atom feeds or only RSS 2.0?**

No, the MCP is built to handle both. It parses both RSS 2.0 and Atom formats identically, unifying them into one consistent JSON structure for your agent.

03 **What kind of data does parse_rss_feed extract?**

It extracts comprehensive metadata including the title, full link, publication date, author, categories, content snippet, and even podcast enclosure details into a single structured JSON object.

04 **Can I use the RSS Feed Parser for news sites that require logins?**

No. This MCP requires access to the public feed XML/Atom URL. If the site content is paywalled or behind a login, you won't be able to parse it.

05 **Is this better than simple web scraping?**

Yes. Traditional scraping parses messy HTML; this MCP understands standardized feed XML and delivers clean JSON directly. It’s reliable because the input format is controlled by the web standard, not the site's design.

Go Live in 60 Seconds

Get your connection token from **cloud.vinkius.com**, then paste the endpoint URL into any MCP-compatible client.

YOUR MCP ENDPOINT

```
https://edge.vinkius.com/[TOKEN]/mcp
```

CLIENT	WHERE TO CONFIGURE
Claude AI	Profile → Customize → Connectors → "+" → Add custom connector → Paste endpoint
Cursor	Settings → Features → MCP Servers → "+ Add New MCP Server" → Type: SSE → Paste endpoint
VS Code	Ctrl/Cmd+Shift+P → "MCP: Add Server" → add `"rss-feed-parser": { "url": "…" }`
Windsurf	MCP Settings → `mcp_settings.json` → Add endpoint URL
ChatGPT	Settings → Tools & plugins → Add MCP server → Paste endpoint
Gemini	Extensions → Add MCP Server → Paste endpoint URL

ASK AN AI ABOUT THIS

Let your preferred AI explain this MCP server

- Ask ChatGPT
- Ask Claude
- Ask Perplexity
- Ask Gemini
- Ask Grok

READY TO CONNECT

RSS Feed Parser is live on Vinkius Cloud.

Get your connection token, paste it into your AI agent, and start building. No SDK. No deployment. Just results.

Start at cloud.vinkius.com →

vinkius.com · support@vinkius.com

INDEPENDENT PLATFORM DISCLAIMER

Vinkius is an independent platform and is not affiliated with, endorsed by, sponsored by, verified by, or otherwise authorized by RSS Feed Parser. All third-party trademarks, logos, and brand names are the property of their respective owners. Their use in this document is strictly for informational purposes to identify service compatibility and interoperability.

DOCUMENT INFORMATION

Generated	June 2026
MCP Server	RSS Feed Parser MCP
Server ID	`019e38e6-a2af-72af-915b-c6180315afff`
Platform	Vinkius Cloud for AI Agents
Endpoint	`https://edge.vinkius.com/{token}/mcp`

LICENSE & USAGE

This document is generated automatically by the Vinkius PDF Engine. Content reflects the MCP server configuration at the time of generation and may change as updates are deployed. For the most current information, visit vinkius.com/mcp/rss-feed-parser.