initMAX

ZABBIX PREMIUM PARTNER

ZABBIX CERTIFIED TRAINER

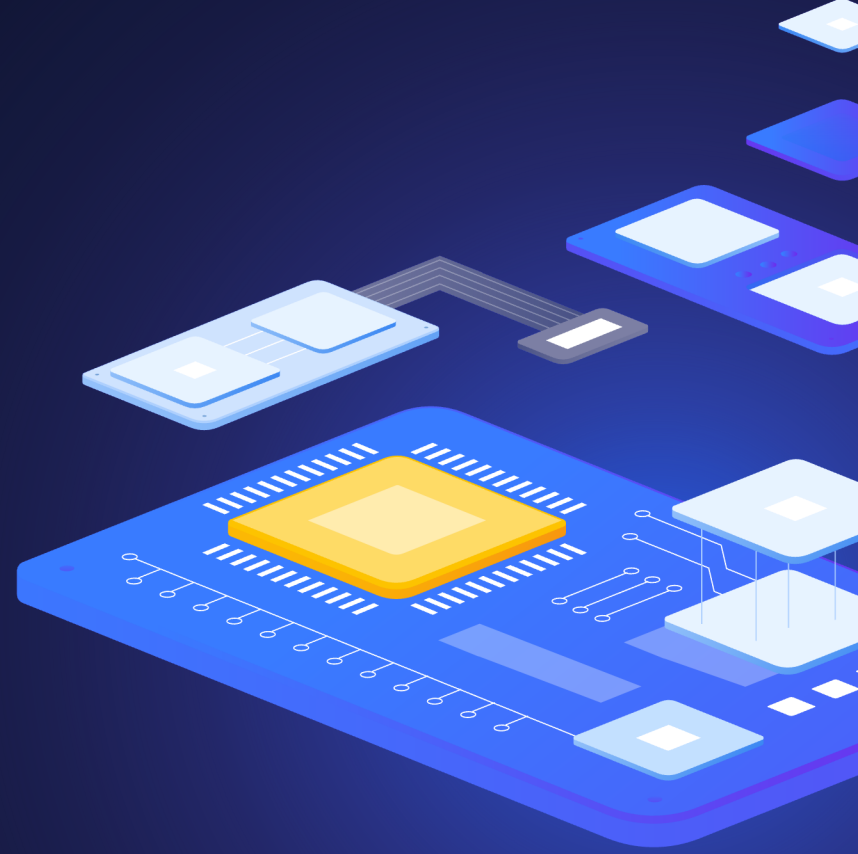
Webinar

AI Event Correlation in Zabbix

All microphones are muted

Please ask questions in Q&A, not in Chat

Use Chat for discussion, networking, or applause



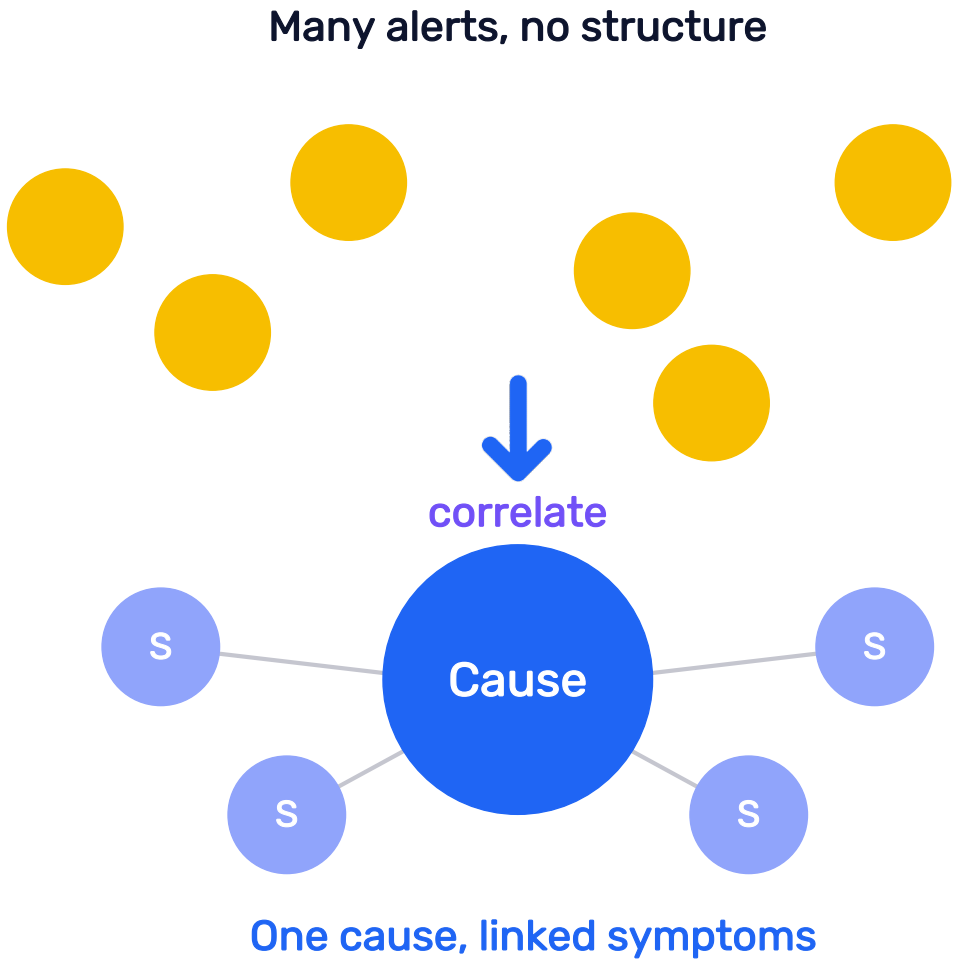
1

Event correlation - the basics

initMAX

What is event correlation

- **Analyze events together**, not in isolation
- **Identify relationships** between alerts across systems
- **Turn raw events into insights** — less noise, faster root cause
- **Group related alerts** — one incident, not dozens of tickets
- **Suppress duplicates (deduplication)** — silence repeating noise from the same source
- **Reveal cause-and-effect** — see which event triggered the rest

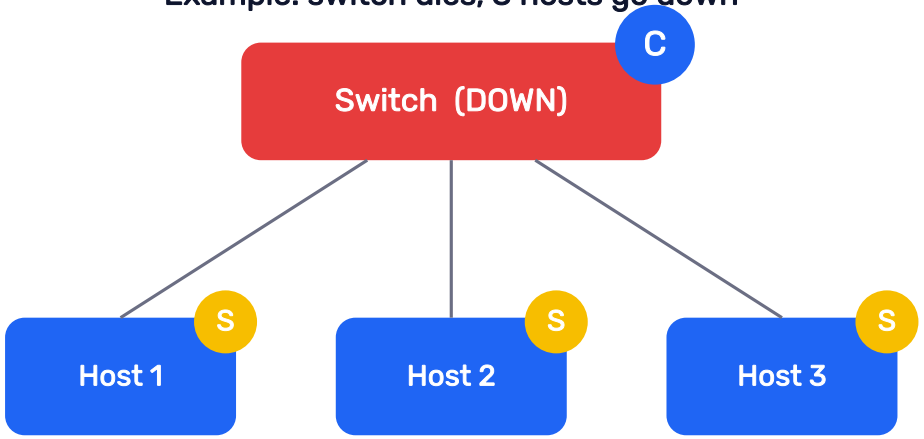


initMAX

Signal vs noise problem

- **100 alerts hit the operator at once** — most are symptoms, not causes
- **~80 are duplicates or follow-ons** of a few real failures
- **Without correlation** — each alert investigated separately
- **Example** — one DB node down triggers 40 service alerts, 20 timeouts, 15 queue backlogs
- **Outcome** — longer MTTR, missed root cause, alert fatigue

Example: switch dies, 3 hosts go down

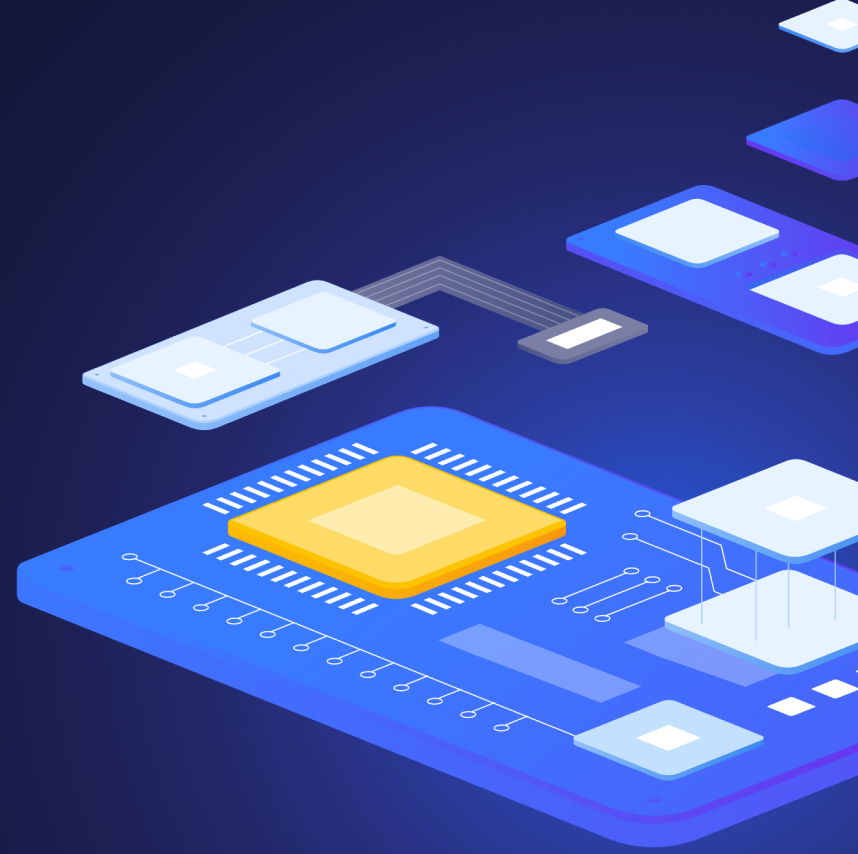


3 host alerts = symptoms, **1 switch alert = cause**

20:01:17	Average	SolarCache	Error Log Rate Increase	AI Assistant	1m 47s	Update	2	.app: nginx matrix: error.logs ~ticket: INC-9999
20:01:40	Average	ZenithPeak	Database Response Slow	AI Assistant	1m 24s	Update	1	.app: nginx matrix: slow.query.response ~ticket: INC-9999
20:01:37	Information	ZenithPeak	Memory Usage Critical	AI Assistant	1m 27s	Update	1	.app: nginx matrix: memory.usage ~ticket: INC-9999
20:01:35	High	ZenithPeak	High CPU Load Alert	AI Assistant	1m 29s	Update	2	.app: nginx matrix: CPU ~ticket: INC-9999
20:01:34	Information	VoyagerNet	Temperature High Alert	AI Assistant	1m 30s	Update	2	.app: nginx matrix: high.temperature ~ticket: INC-9999
20:01:28	Warning	VoyagerNet	Disk Space Low	AI Assistant	1m 36s	Update	2	.app: nginx matrix: disk.space ~ticket: INC-9999
20:01:24	Disaster	ThunderMail-02	Database Response Slow	AI Assistant	1m 40s	Update	3	.app: nginx matrix: slow.query.response ~ticket: INC-9999

2

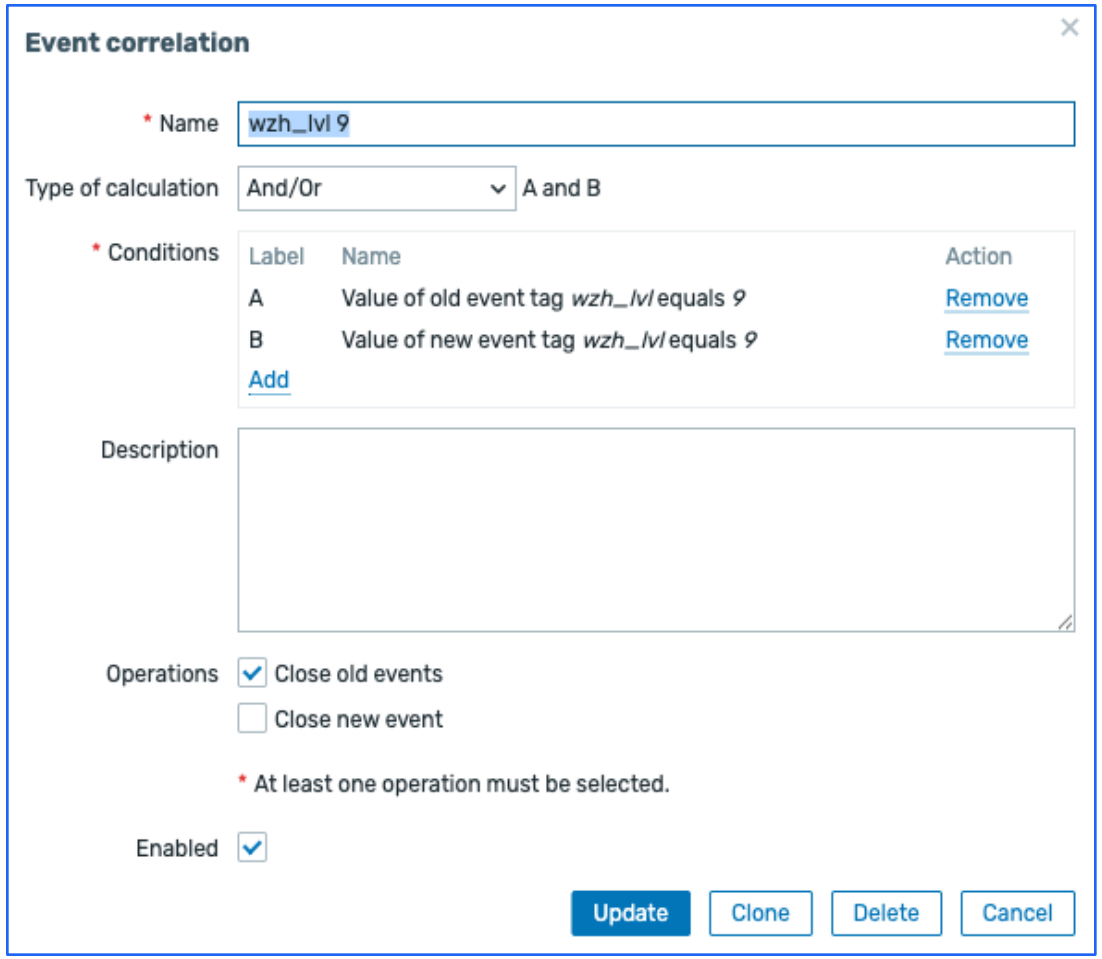
Approaches to correlation



initMAX

Rule-based correlation

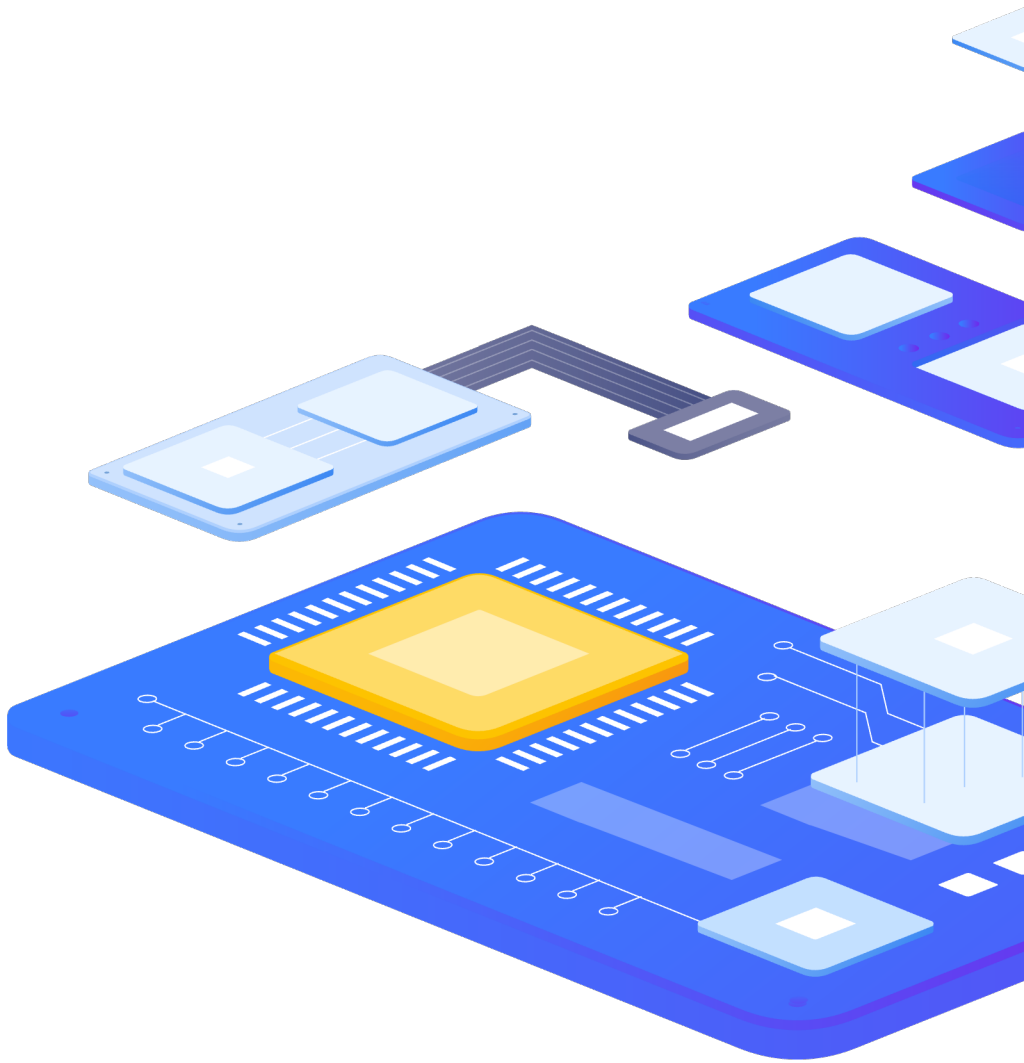
- **Engineers write explicit rules** — if A and B within 5 min then group
- **Predictable, auditable** — easy to debug and reason about
- **Hard to scale** — rules grow with every new pattern
- **Cannot handle unknown** or novel failure scenarios
- **Examples** — Zabbix correlation rules, Prometheus Alertmanager grouping



initMAX

Topology-based correlation

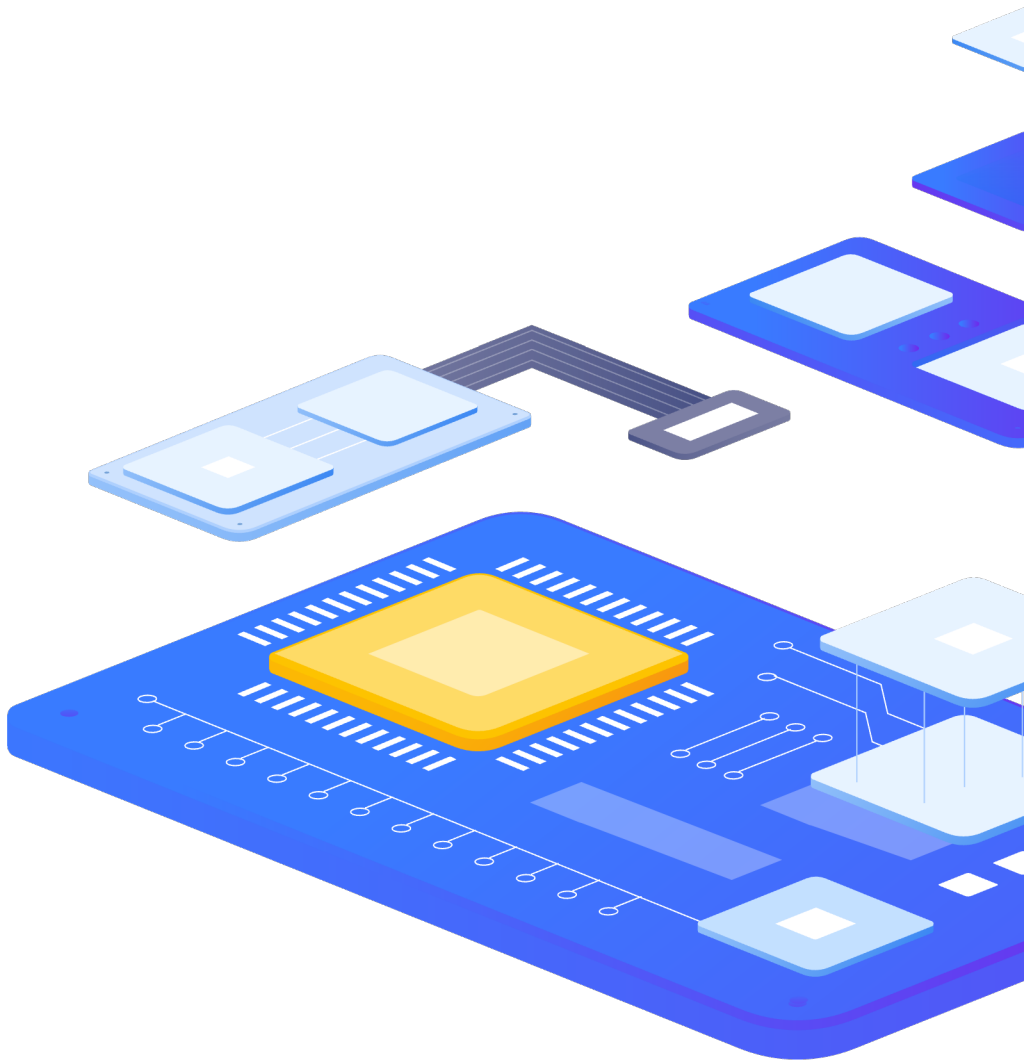
- **Uses CMDB or service map** to understand dependencies
- **Parent down → child alerts suppressed** as symptoms
- **Strong** when topology is accurate and up to date
- **Weak** when CMDB drifts or relationships are missing
- **Examples** — NetBox, IP Fabric, ServiceNow CMDB



initMAX

ML and statistical correlation

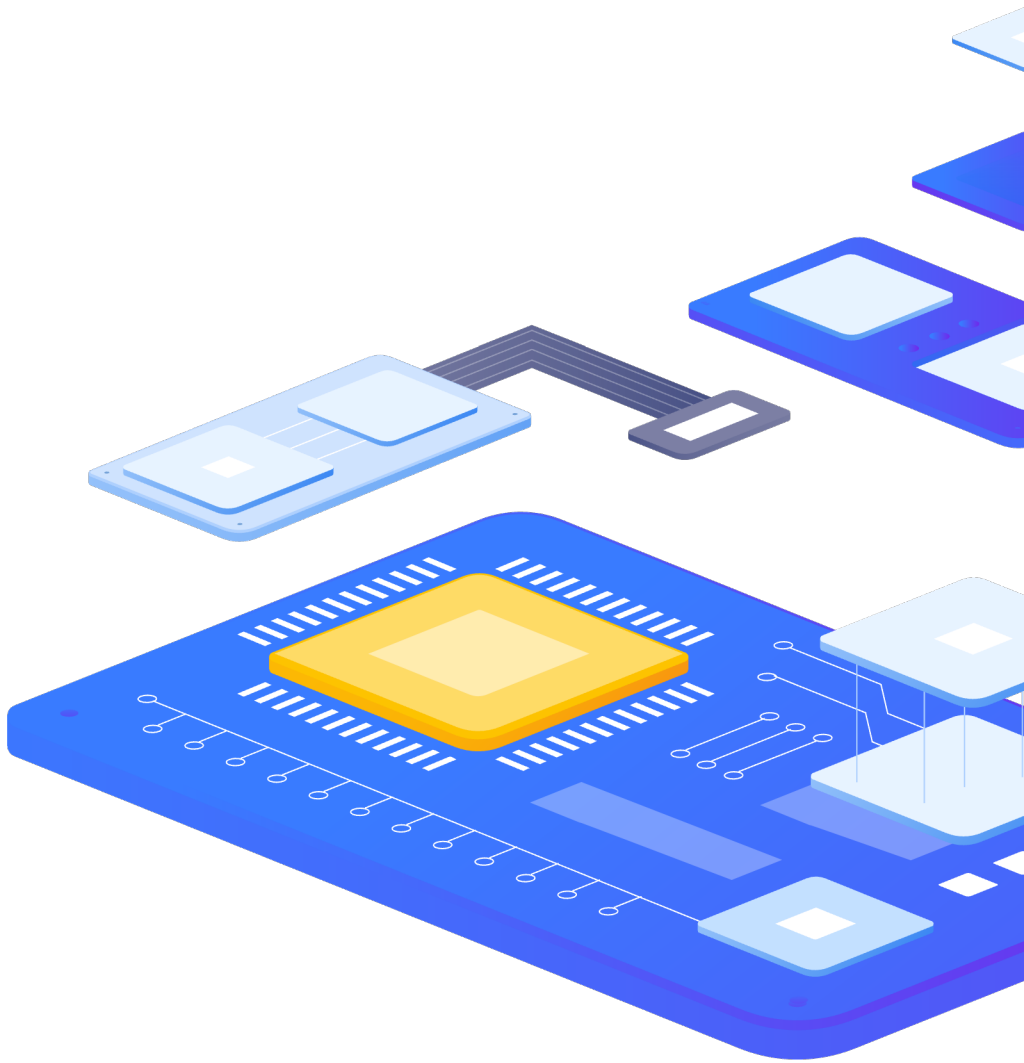
- **Clusters events** by time, host, tags, text similarity
- **Models learn** typical incident shapes from history
- **Finds patterns** humans miss in large volumes
- **Needs training data**, retraining and tuning
- **Examples** — Moogsoft, BigPanda, ServiceNow AIOps



initMAX

LLM-based correlation

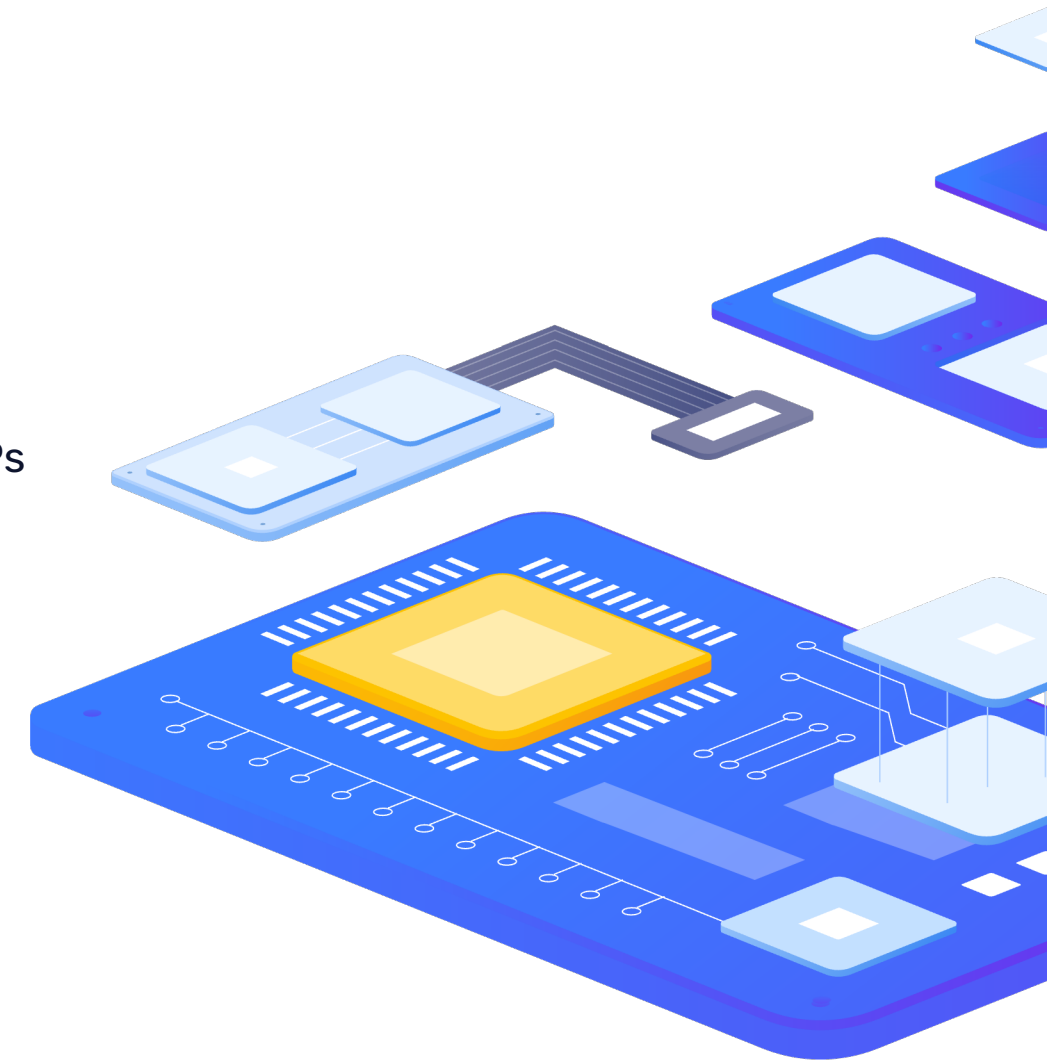
- **LLM decides** if events belong to one incident
- **Reasons over** event text, tags, host, severity, time
- **No training data needed** — uses general world knowledge
- **Produces explanation** — short, human-readable summary
- **Risks** — latency, cost, hallucinations, non-determinism



initMAX

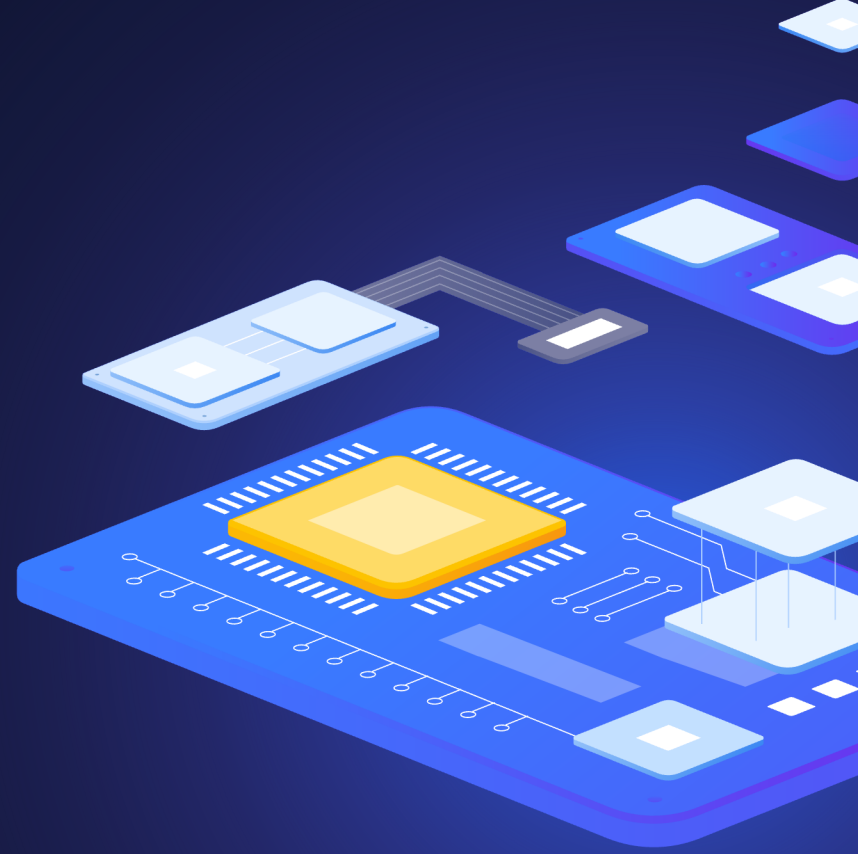
Comparison of approaches

- **Rule-based** — high precision, low recall, hard to maintain
- **Topology** — strong with good CMDB, blind to unknown patterns
- **ML / statistical** — scalable, requires data and tuning
- **LLM** — flexible, explainable, slower and more expensive
- **Hybrid wins in practice** — combine deterministic and LLM with MCPs



3

Hybrid approach and architectures



initMAX

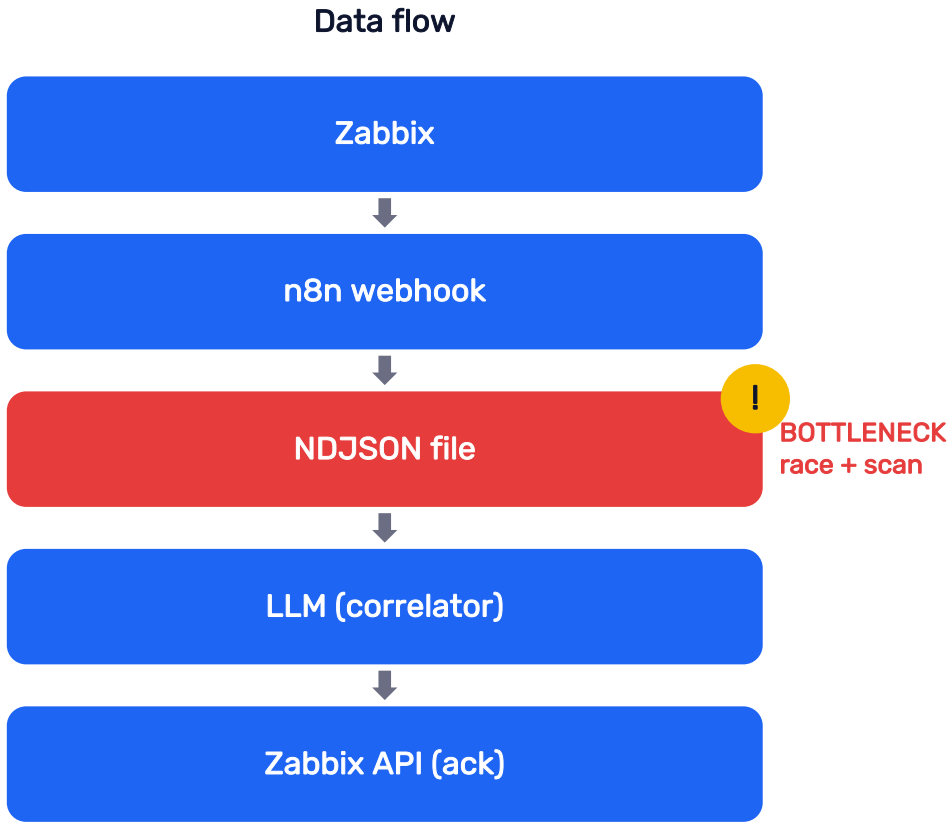
Hybrid approach - deterministic plus LLM

- **Step 1:** deterministic pre-filter narrows candidates
  - same host, shared tag, recent time window (60 min)
- **Step 2:** LLM decides cause / symptom only on the short list
- **Step 3:** deterministic recovery handler detaches symptoms
- **Result:** fewer LLM calls, lower cost, predictable behavior
- **LLM** is used where it shines — judgment under ambiguity

initMAX

Architecture 1 - file-based prototype

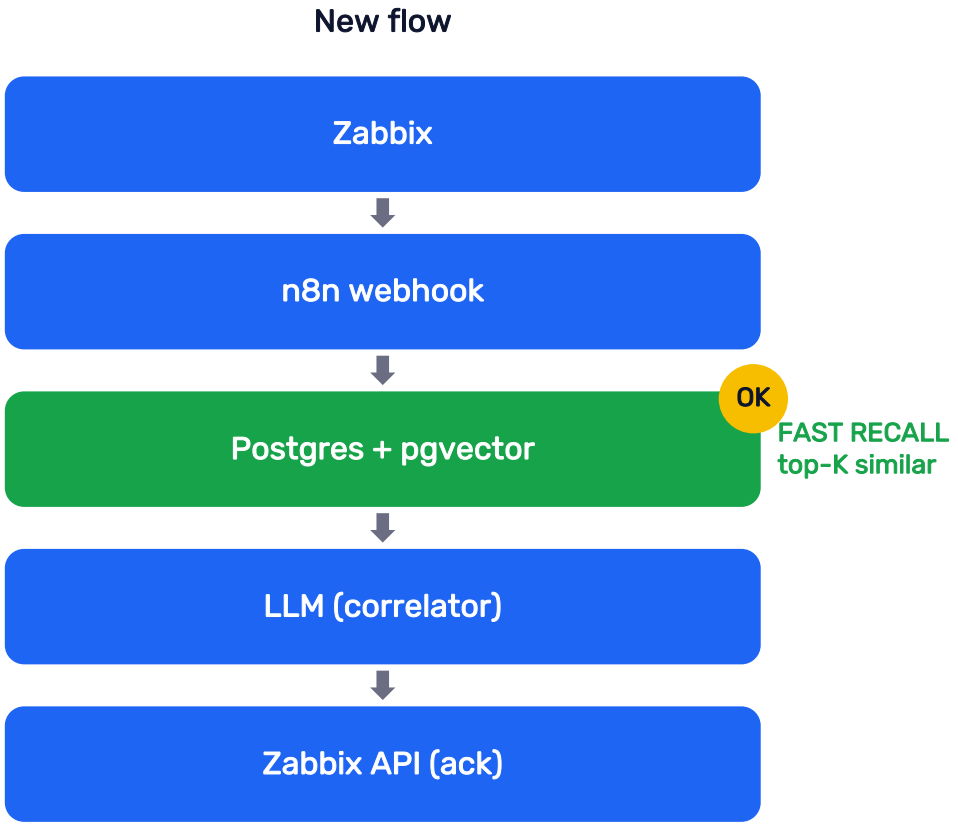
- **NDJSON file** stores recent problem events
- **Webhook appends,** prompt template is sent to LLM
- **Pros:** simplest possible setup, fast to demo
- **Cons:** race conditions on parallel webhooks
- **Cons:** file lock issues, no proper indexing
- **Verdict:** antipattern for production, only for first PoC



initMAX

Architecture 1 - the right way

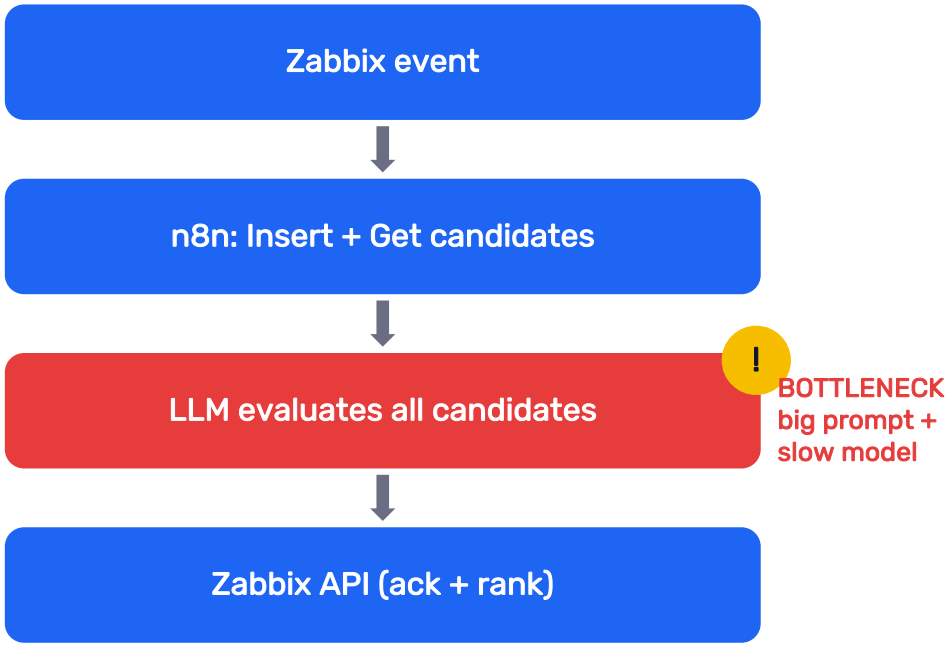
- **Storage:** Postgres — indexed, concurrent-safe, queryable
- **Add pgvector** extension — same DB, no new infra
- **Embed event name + host + tags** via embed model (LiteLLM)
- **LLM gets top-K similar past events,** not a full file scan
- **Bonus:** pattern discovery (clustering), RAG context
- **Future:** anomaly detection, few-shot examples



initMAX

Architecture 2 - n8n Data Tables

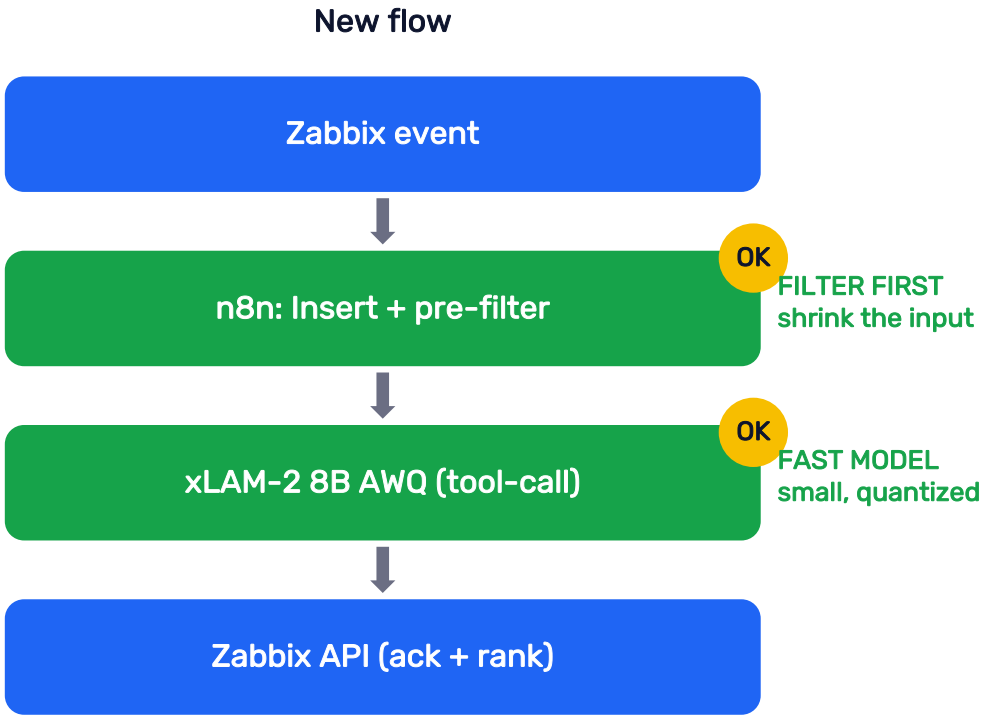
- **n8n Data Tables** act as a small managed event store
- **Normalize step:** eventid, name, host, severity, clock, tags
- **Insert on PROBLEM,** Get within 60 min for candidates
- **LLM prompt** receives current event plus candidates
- **AI returns** cause_eventid, confidence, short reason
- **Atomic Insert + Get** removes race conditions of file approach



initMAX

Architecture 2 - the right way

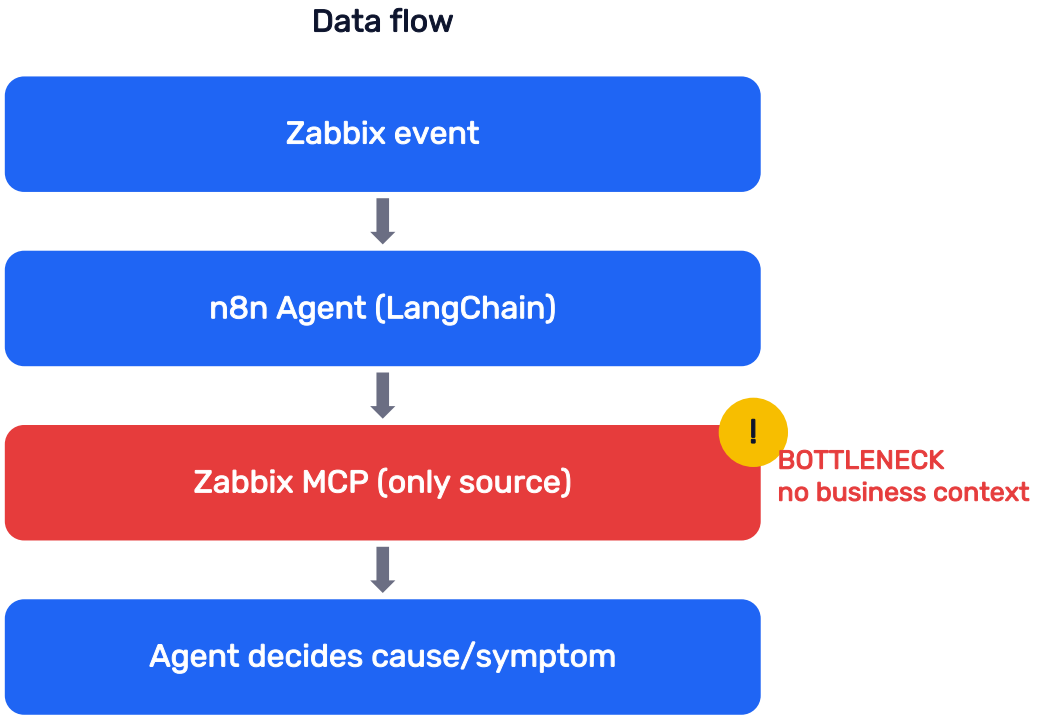
- **Pre-filter candidates BEFORE the LLM**
  - same host, same tag prefix, narrow time window
  - if 0 candidates → skip the model, save money
- **Pick a small, fast tool-call model** (xLAM-2 8B) or 4o-mini
- **Quantize to AWQ Int4** — ~3× smaller, ~2× faster
- **LLM gets a tight prompt:** only top relevant events



initMAX

Architecture 3 - n8n Agent plus MCP

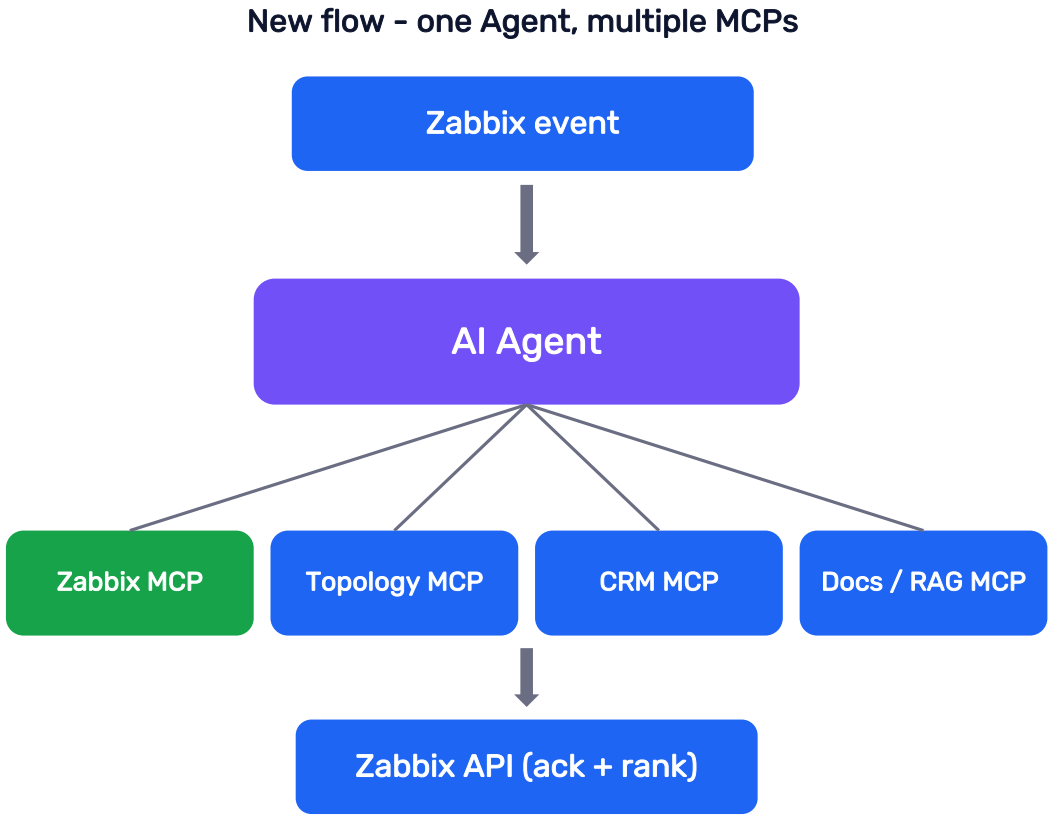
- **LangChain agent** inside n8n with tool calling
- **Model:** qwen3-vl on-prem (or gpt-4o-mini, claude-haiku in cloud)
- **Tool:** Zabbix MCP server exposes problem.get, event.acknowledge
- **Agent reasons,** calls tools, decides cause / symptom
- **Recovery path** is deterministic (action 128 detach), no LLM needed
- **Most flexible** — new tools added without prompt rewrite

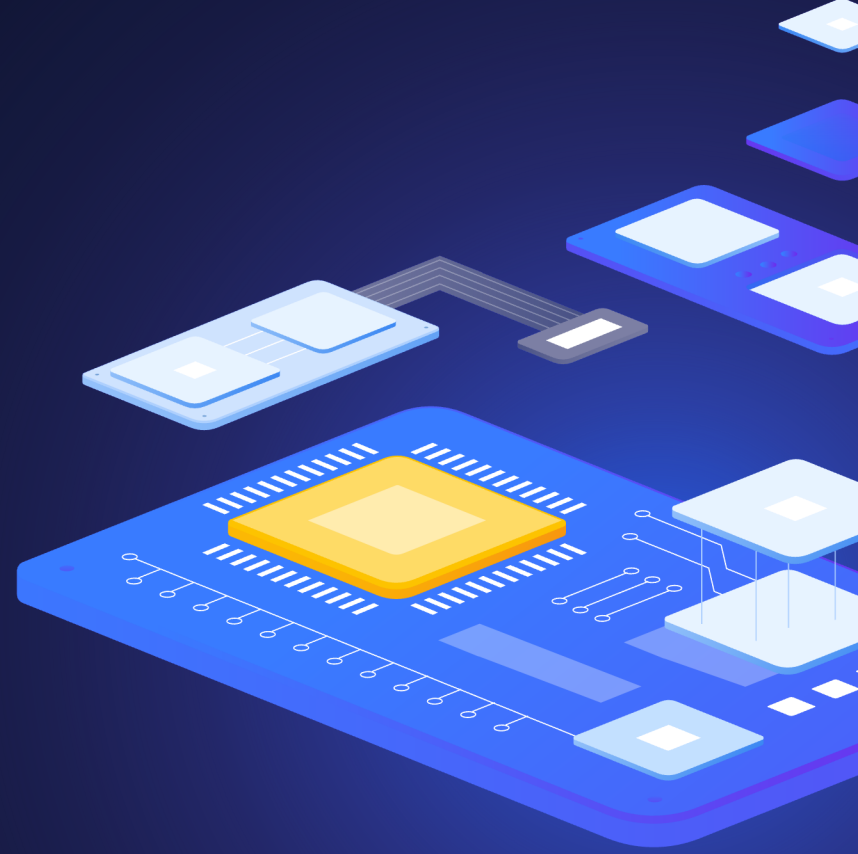


initMAX

Architecture 3 - the right way

- **One agent, many MCPs** — layered context
- **Zabbix MCP** — read events, ack, set rank
- **Topology MCP** — host dependencies, links
- **CRM MCP** — customer impact, SLA tier
- **Docs MCP** — runbooks, past resolutions
- **History MCP** — similar past incidents (RAG)
- **Agent reasons** over the full picture, not just metrics





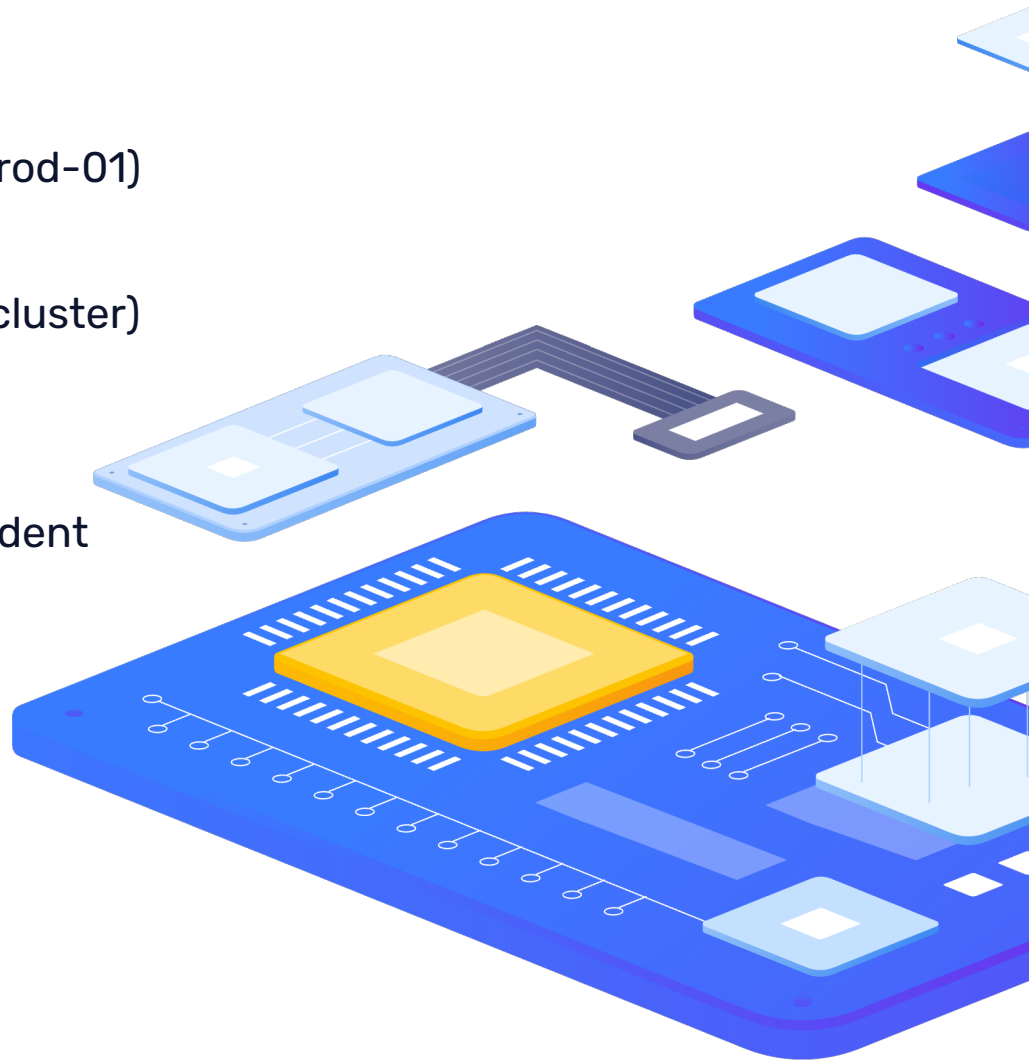
4

LLM decision and Zabbix actions

initMAX

AI weights in the decision

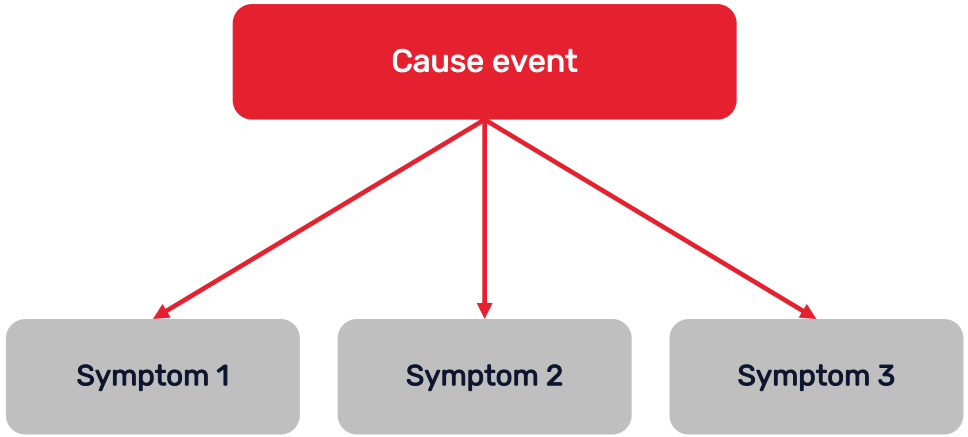
- **Same host** — strongest signal (e.g. CPU + disk + memory all on db-prod-01)
- **Shared tags** — service=payments, env=prod across hosts
- **Alert type / template** — same trigger family (e.g. ICMP unreachable cluster)
- **Time proximity** — 12 alerts within 90 s = likely one root cause
- **Severity relation** — High disaster outranks Warning symptoms
- **Confidence ≥ 0.7** — below threshold the agent leaves alerts independent



initMAX

Cause / Symptom in Zabbix 7+

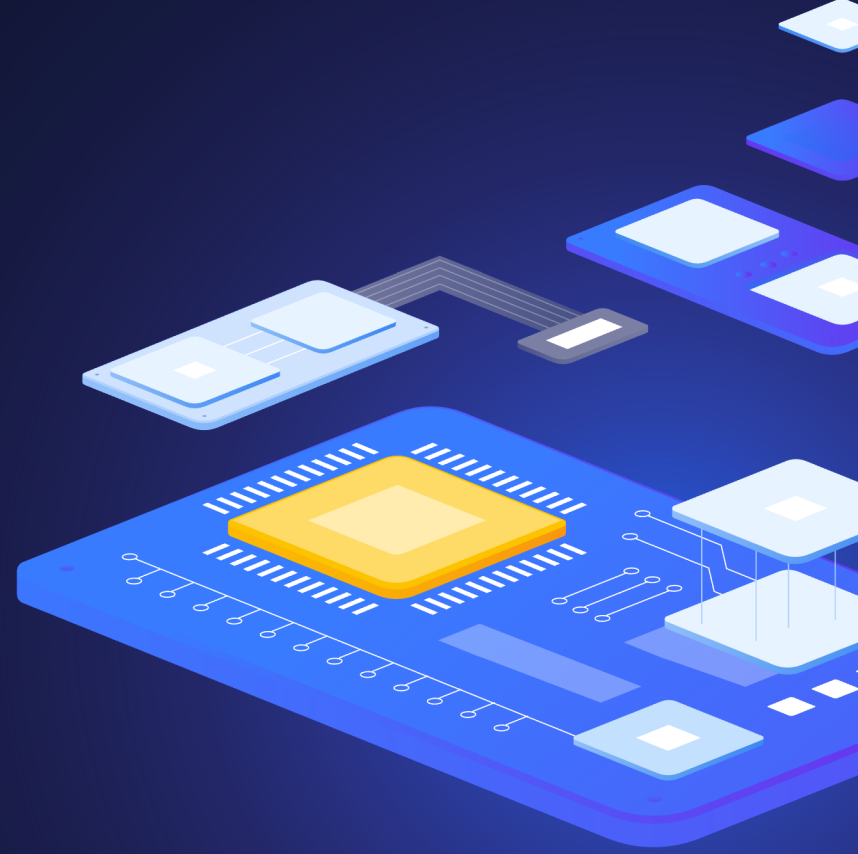
- **Native concept** since Zabbix 7.0 — exposed via event.acknowledge API
- **Link a symptom:** API call with action=256 + cause_eventid (256 = "rank as symptom" bit)
- **One cause event** groups many symptoms in the UI
- **Detach when symptom resolves:** API call with action=128 ("rank as cause") promotes it back to standalone
- **Detach is rule-based, not AI** — fixed steps: problem.get → if resolved → event.acknowledge(128).
  No judgment, no LLM cost.
- **LLM is used only once** — at the start, to decide which event is cause and which are symptoms



rank-as-symptom (action bit 256)

5

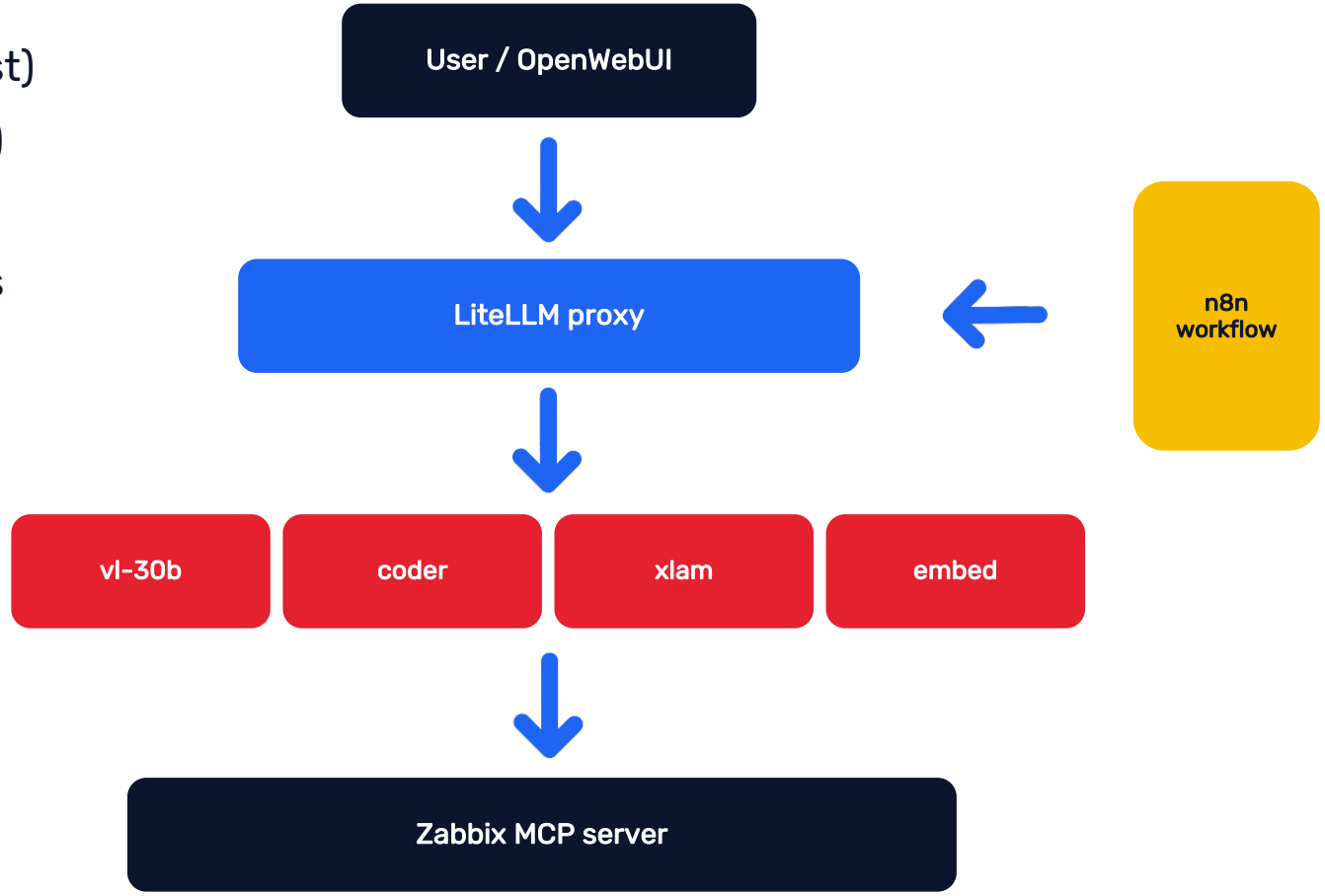
On-prem vs Cloud



initMAX

On-prem architecture

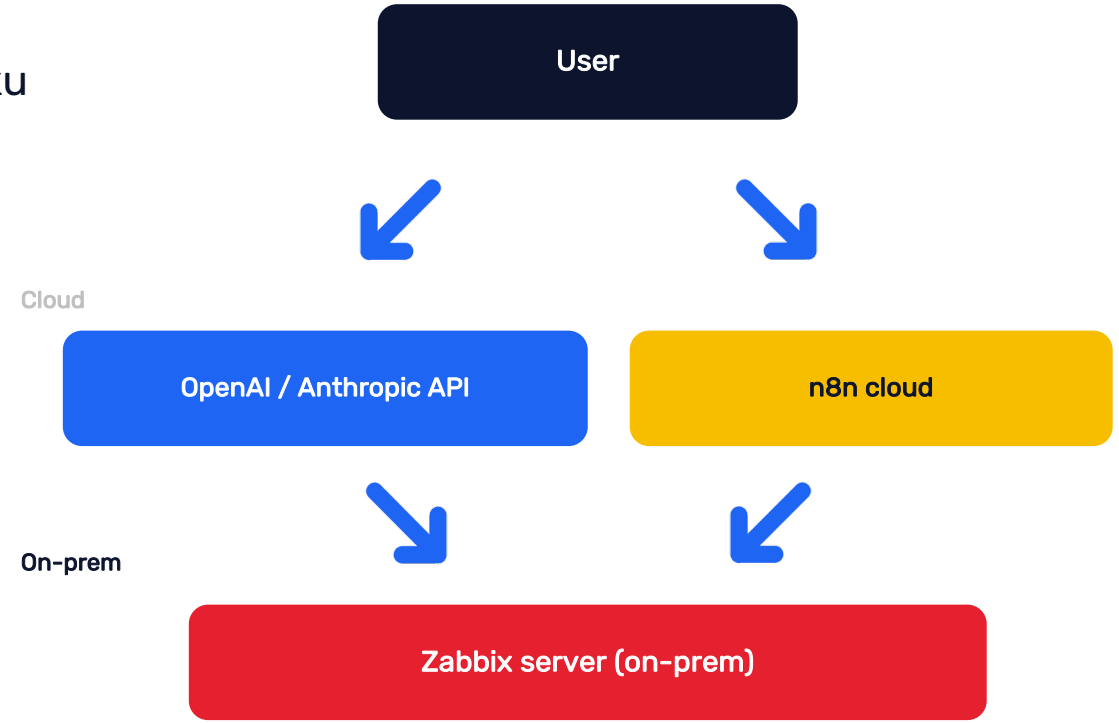
- **NVIDIA DGX Spark** or similar GPU box (one-time cost)
- **vLLM** serves the model (qwen3-vl-30b, xlam-2-8b)
- **LiteLLM proxy** unifies the OpenAI-compatible API
- **n8n self-hosted** runs the workflow and Data Tables
- **Zabbix MCP server** bridges API to the agent
- **All data stays on premises** — no external calls



initMAX

Cloud architecture

- **Provider API:** OpenAI gpt-4o-mini or Anthropic claude-haiku
- **n8n cloud or self-hosted** runs the workflow
- **Zabbix MCP server** reachable from n8n
- **Pay per token,** no GPU to manage
- **Quick to start** — no infra investment
- **Data leaves the perimeter** — check compliance



initMAX

On-prem vs Cloud - comparison

- **Privacy:** on-prem wins, data never leaves
- **Compliance:** GDPR, NIS2, HIPAA — cloud needs DPA & data residency
- **Cost (CAPEX vs OPEX):** on-prem = high CAPEX, cloud = OPEX scaling with volume
- **Latency:** on-prem can be lower, depends on hardware
- **Scaling:** cloud scales instantly, on-prem needs more GPUs
- **Vendor lock-in:** on-prem = swap weights freely; cloud = retune prompts on API change
- **Reproducibility:** cloud silently updates models; on-prem stable output
- **Network dependency:** cloud breaks on WAN outage; on-prem runs isolated
- **Time to value:** cloud = minutes; on-prem = weeks (HW + setup)
- **Skill required:** on-prem = GPU/vLLM ops; cloud = HTTP API only
- **Energy:** on-prem GPU runs 24/7; cloud bills only on call
- **Observability:** on-prem = full token/latency view; cloud = provider logs only

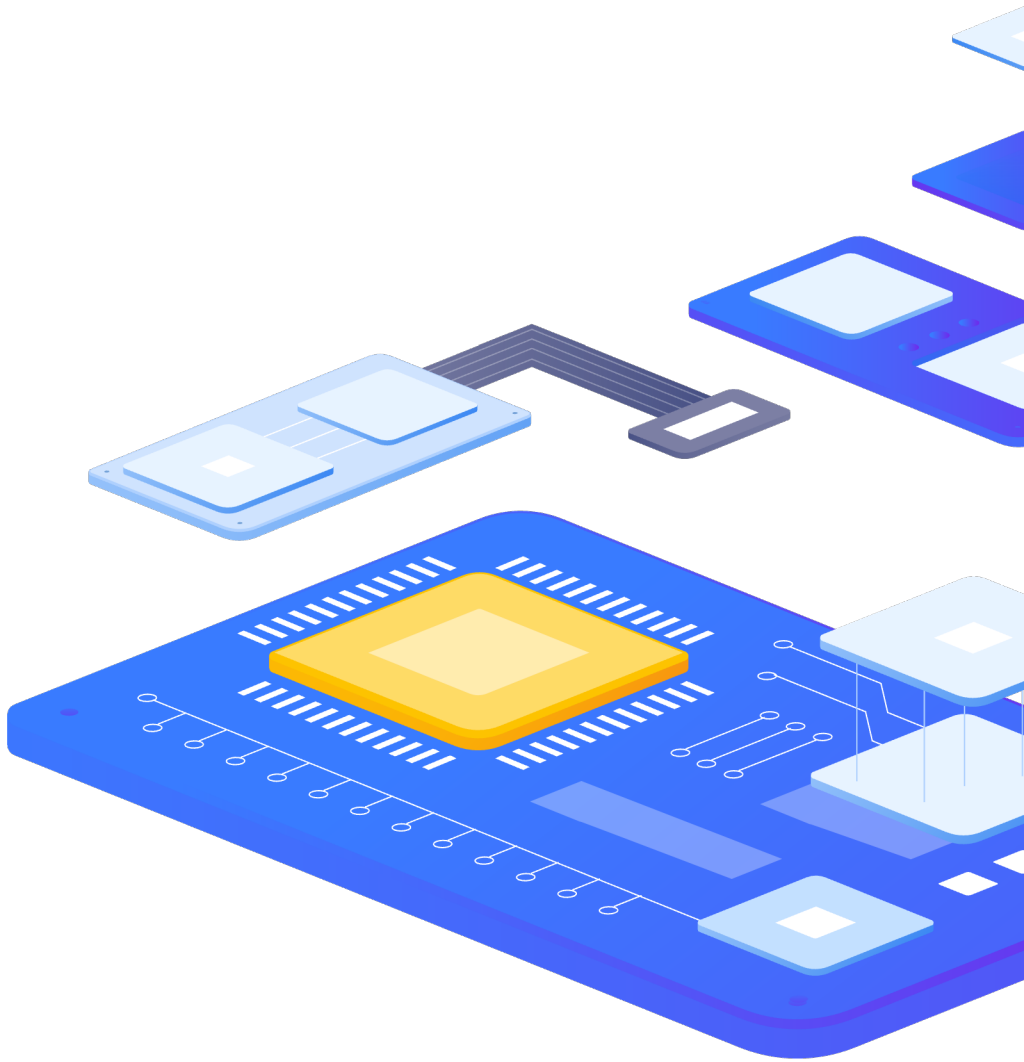
initMAX

Pricing - rough estimate

- **On-prem DGX Spark:** about 4000 USD one-time, 0 USD per token
- **Electricity and ops** not included — typically minor at this scale
- **Cloud gpt-4o-mini:** 0.15 USD / 1M input, 0.60 USD / 1M output
- **Typical event prompt:** ~1k tokens, ~0.0001 USD per event
- **1M events per month:** ~100 USD on cloud

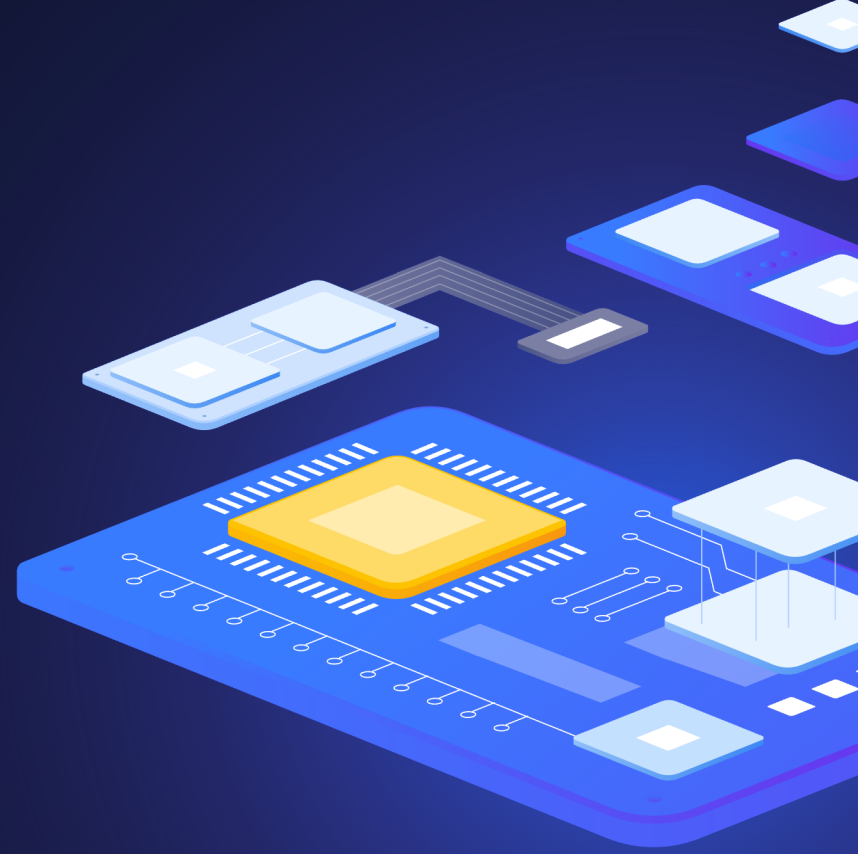
⚠ Caveat — cloud pricing will rise

- **Market leaders are loss-making:** OpenAI projects ~$74B operating loss in 2028, profit only by 2030; Anthropic break-even by 2028
- **Price hikes are baked in** — providers must close massive losses to honor profitability promises
- **gpt-4o-mini will be deprecated** — successor gpt-5.4-mini: $0.75 input / $4.50 output → 5× input, 7.5× output
- **1M events on gpt-5.4-mini:** jumps from ~$100 to ~$600/month at same usage



6

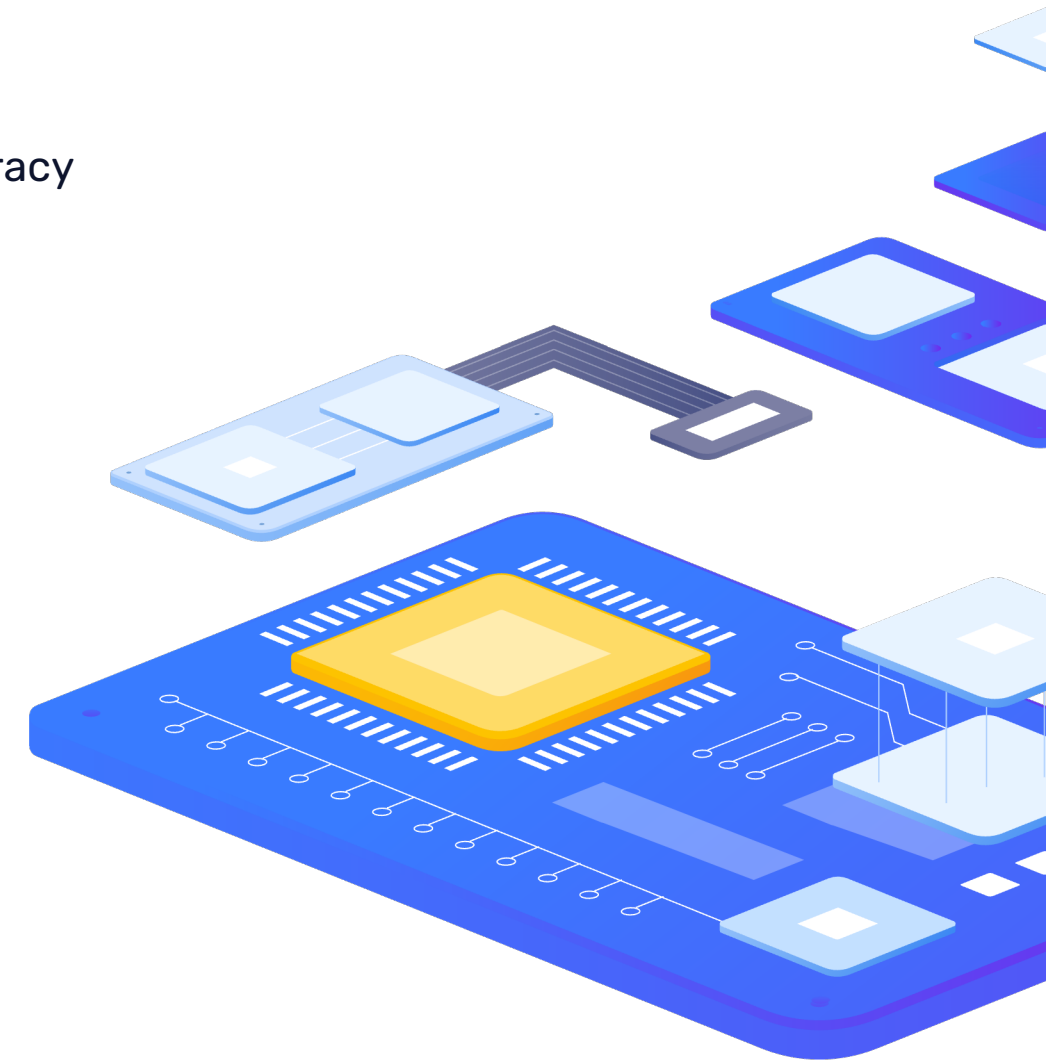
Benchmark and tips



initMAX

Benchmark - models tested

- **qwen3-vl-30b on-prem:** ~5–15 s per event, ~1.2k tokens, 88% accuracy
- **xlam-2-8b on-prem:** ~2 s per event, ~1.0k tokens, 82% accuracy
- **gpt-4o-mini cloud:** ~1 s per event, ~1.0k tokens, 90% accuracy
- **claude-haiku cloud:** ~2 s per event, ~1.1k tokens, 91% accuracy
- **Success rate (no parse error):** 99% cloud, 96% on-prem
- **Numbers are indicative** — measure on your own data



initMAX

LLM quantization - shrinking weights

- **Original LLM weights are FP16 / BF16** — 30B params = ~60 GB just for weights
- **Quantize to lower precision** → 3–4× smaller, ~2× faster inference, 1–2% accuracy loss
- **AWQ Int4** = sweet spot for GPU inference; FP8 KV cache halves cache memory
- **Our stack:** Qwen3-30B AWQ Int4 (~16 GB), xLAM-2-8B BF16 (~15 GB), KV cache fp8

Format	Bits/wt	Use	Notes	Size for 30B
BF16 / FP16	16	Training, full quality	Baseline	~60 GB
FP8 (E4M3)	8	KV cache compression	vLLM --kv-cache-dtype fp8	Halves cache
AWQ Int4	4	Inference (GPU)	Activation-aware, vLLM awq_marlin	~16 GB
GPTQ Int4	4	Inference (GPU)	Older, vLLM gptq_marlin	~16 GB
GGUF Q4_K_M	~4-5	llama.cpp / Ollama	CPU+GPU, NOT vLLM	~18 GB

Format = quantization scheme | Bits/wt = bits stored per weight (lower = smaller, less precise) | Use = typical deployment | Notes = trade-offs | Size for 30B = on-disk size of a 30-billion-parameter model

initMAX

vLLM GPU memory layout

- **vLLM** is the inference server that hosts the LLM and serves requests over an OpenAI-compatible API
- **--gpu-memory-utilization 0.X** = how much VRAM vLLM is allowed to grab (e.g. 0.8 = 80% of the card)
- **That budget is split into 3 buckets:**
  - **Model weights** = parameters × bits/wt ÷ 8 (e.g. 16B × 8 ÷ 8 = 16 GB in FP8; 30B × 4 ÷ 8 = 15 GB in Q4)
  - **Overhead (~1–2 GB)** — CUDA runtime, kernels, activation buffers; you cannot shrink this
  - **KV cache** — whatever is left; remembers attention state for in-flight requests
- **Example below — 16B FP8 model on a 32 GB GPU:** 32 × 0.78 ≈ 25 GB budget – 16 GB weights – 2 GB overhead = **7 GB KV cache**
- **Bigger KV cache** = more parallel requests or longer contexts; single-user latency does NOT improve
- **Tuning lever:** raise --gpu-memory-utilization → bigger KV cache → server handles more concurrent users without OOM

Larger batches and longer contexts shrink the KV cache budget.

initMAX

KV cache sizing - real example

- **Formula:** 2 × num_hidden_layers × num_key_value_heads × head_dim × bytes_per_element
- **Llama-xLAM-2-8B** (GQA 8 KV heads, fp8 cache):
  - **num_hidden_layers** = 32, **num_key_value_heads** = 8, **head_dim** = 128
  - 2 × 32 × 8 × 128 × 1 byte = ~64 KB / token (with vLLM block overhead ~80 KB)
- **Rule of thumb:** total VRAM = weights + 1.5 GB overhead + (concurrent × avg_tokens × KV_per_token)
- **Comparison Qwen3-VL-30B AWQ** (60 layers): ~640 KB / token, ~10× more than xLAM 8B
  - 20 parallel × 4k tokens with Qwen3-VL = 51 GB KV — not feasible alone, needs batching
- **Takeaway:** size util to expected peak load + small headroom; default 0.9 is overkill when stacking models

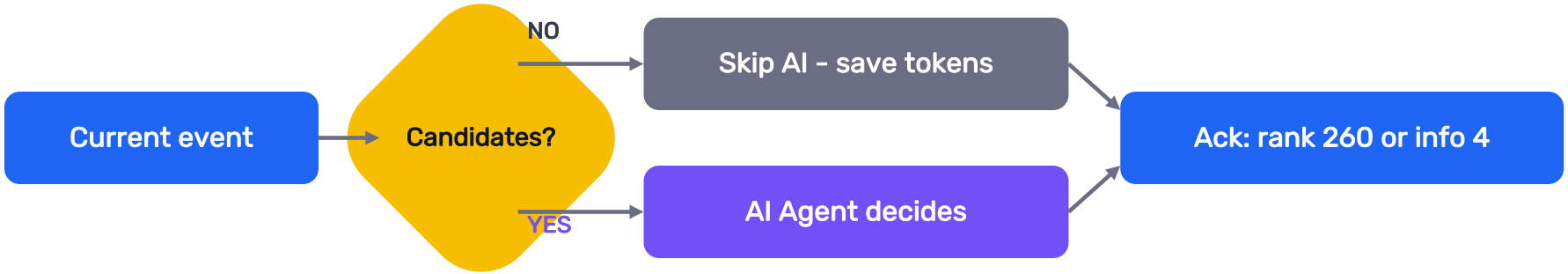
```
{
  "architectures": ["LlamaForCausalLM"],
  "hidden_size": 4096,
  "num_hidden_layers": 32,
  "num_attention_heads": 32,
  "num_key_value_heads": 8,
  "head_dim": 128,
  "torch_dtype": "bfloat16",
  "vocab_size": 128256,
  ...
}
```

Concurrent	Avg context	KV needed	Total VRAM	--gpu-mem-util
20	4,000 tok	6.4 GB	~24 GB	0.20
20	7,000 tok (peak)	11.2 GB	~29 GB	0.24
50	4,000 tok	16 GB	~33.5 GB	0.28

initMAX

Tips and tricks

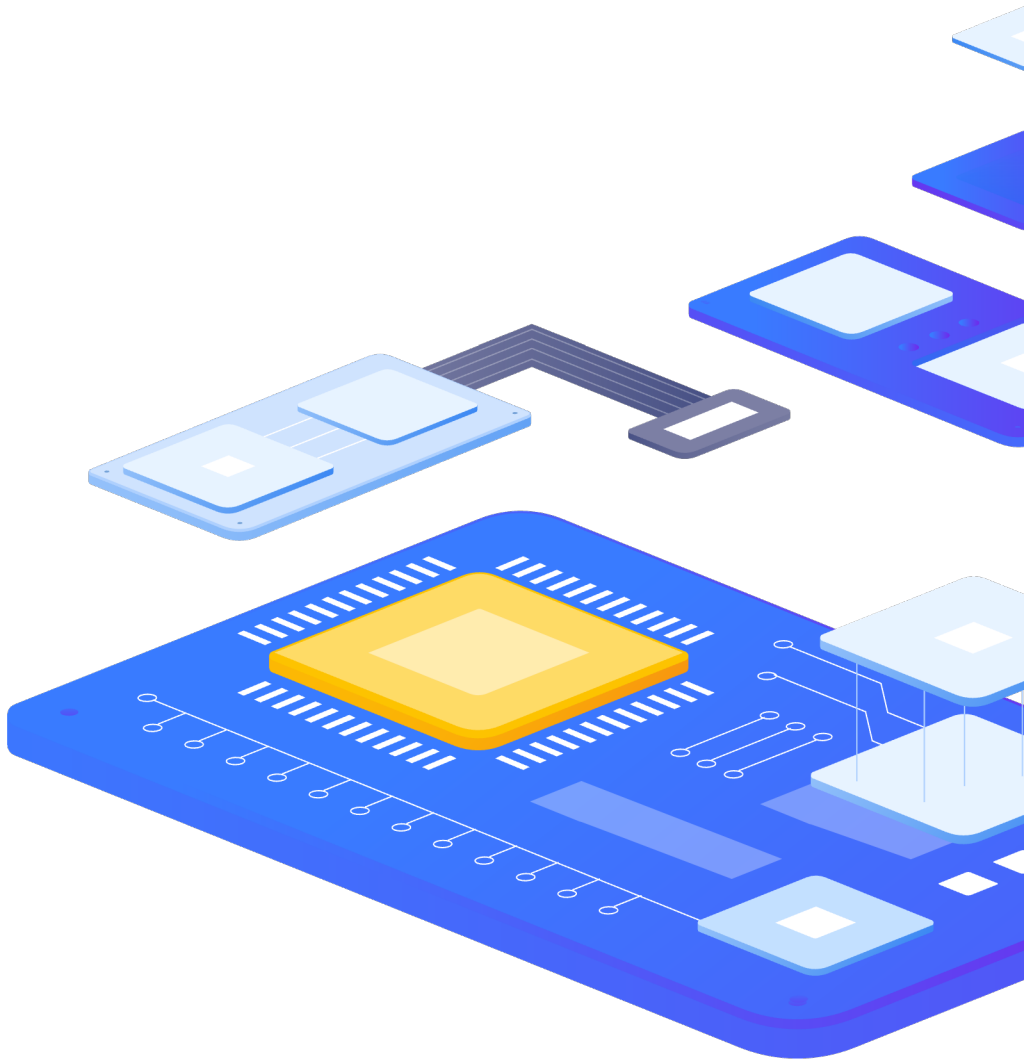
- **Pre-filter candidates** by same host or shared tag
- **Pre-filter alone** reduces LLM calls by 50% or more
- **Compact event shape:** id, name, host, severity, clock, tags
- **Avoid sending raw webhook payload** to the model
- **Time window 60 min** is a sensible default
- **Confidence threshold 0.7** — tune up if precision matters



initMAX

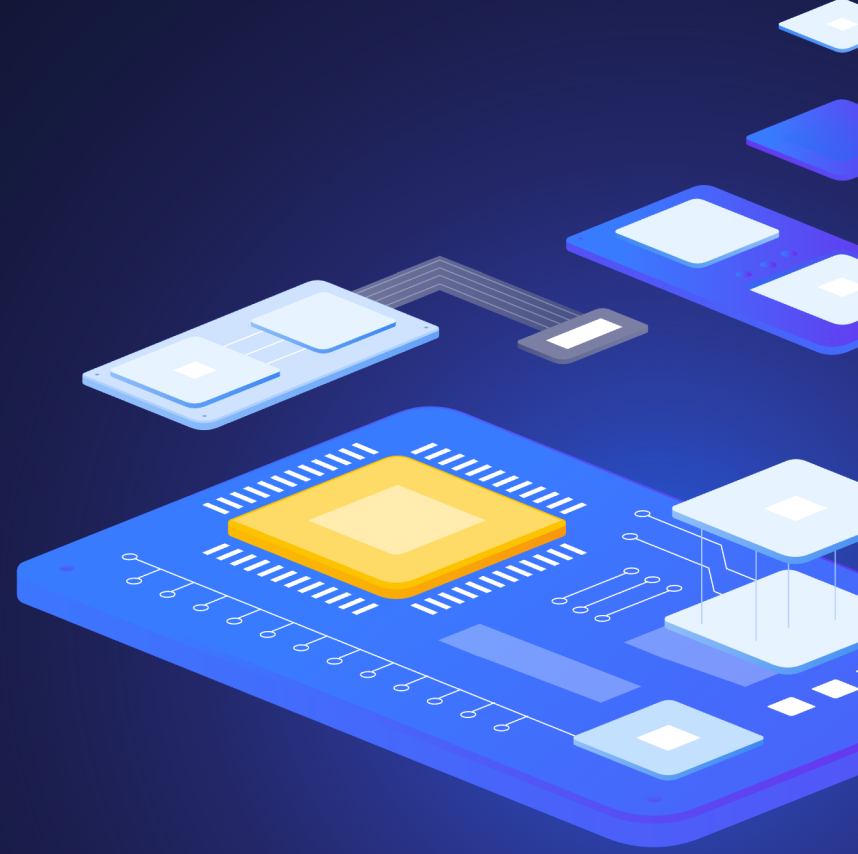
Pitfalls to watch

- **Race conditions** when many webhooks fire in parallel
- **Hallucinated related_eventid** — validate against Data Table
- **Validate** before calling event.acknowledge with cause_eventid
- **Recovery should be rule-based**, do not ask the LLM
- **Token bloat** from raw payloads — compact and cap fields
- **Silent prompt drift** — version your prompts



7

What is next

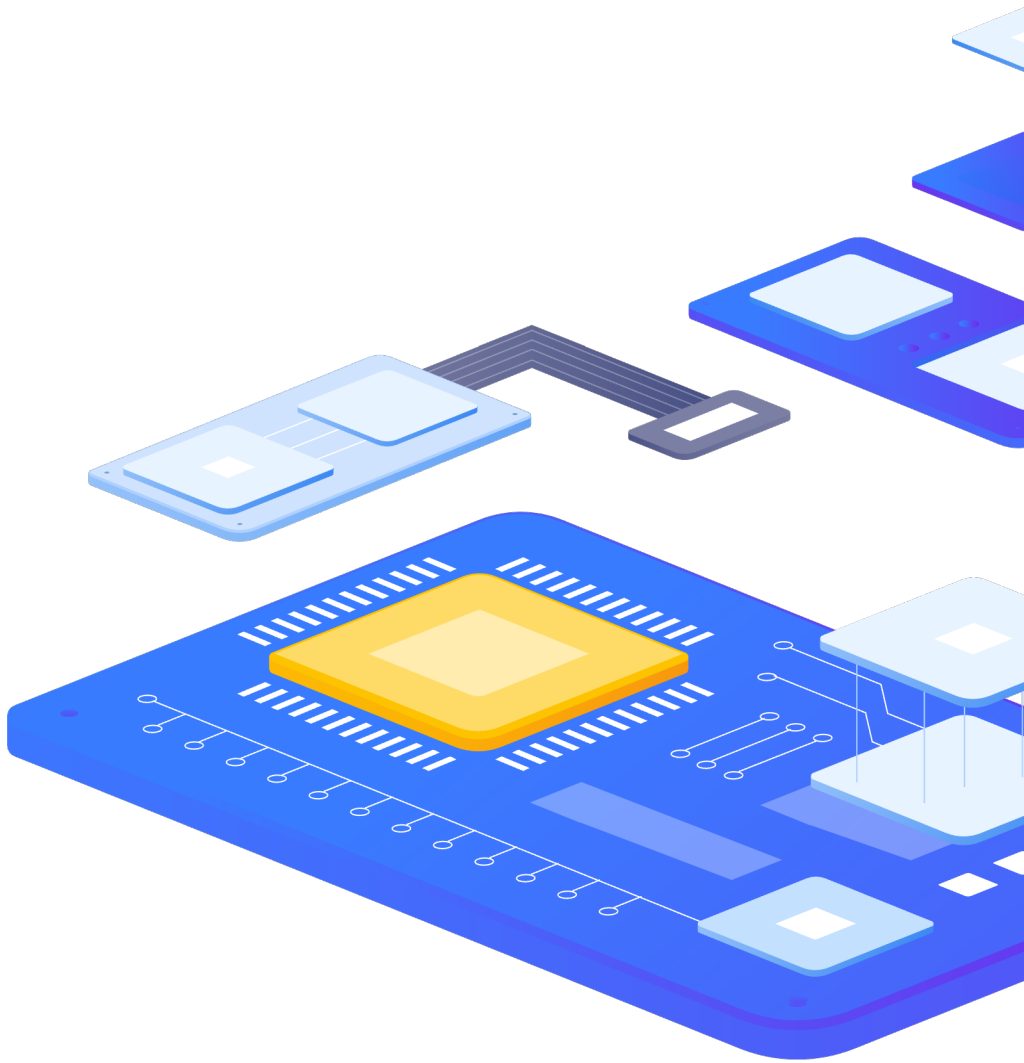


initMAX

AI Event Correlation in Zabbix

Coming in Zabbix 8.0

- **Native correlation engine** in the core platform
- **Rule-based, not AI** — predictable and auditable
- **Closes the gap** for simple grouping use cases
- **AI correlation** stays valuable for ambiguous and novel cases
- **Both can coexist:** rules first, AI as a fallback
- **Set expectations** — 8.0 is not a full AIOps replacement

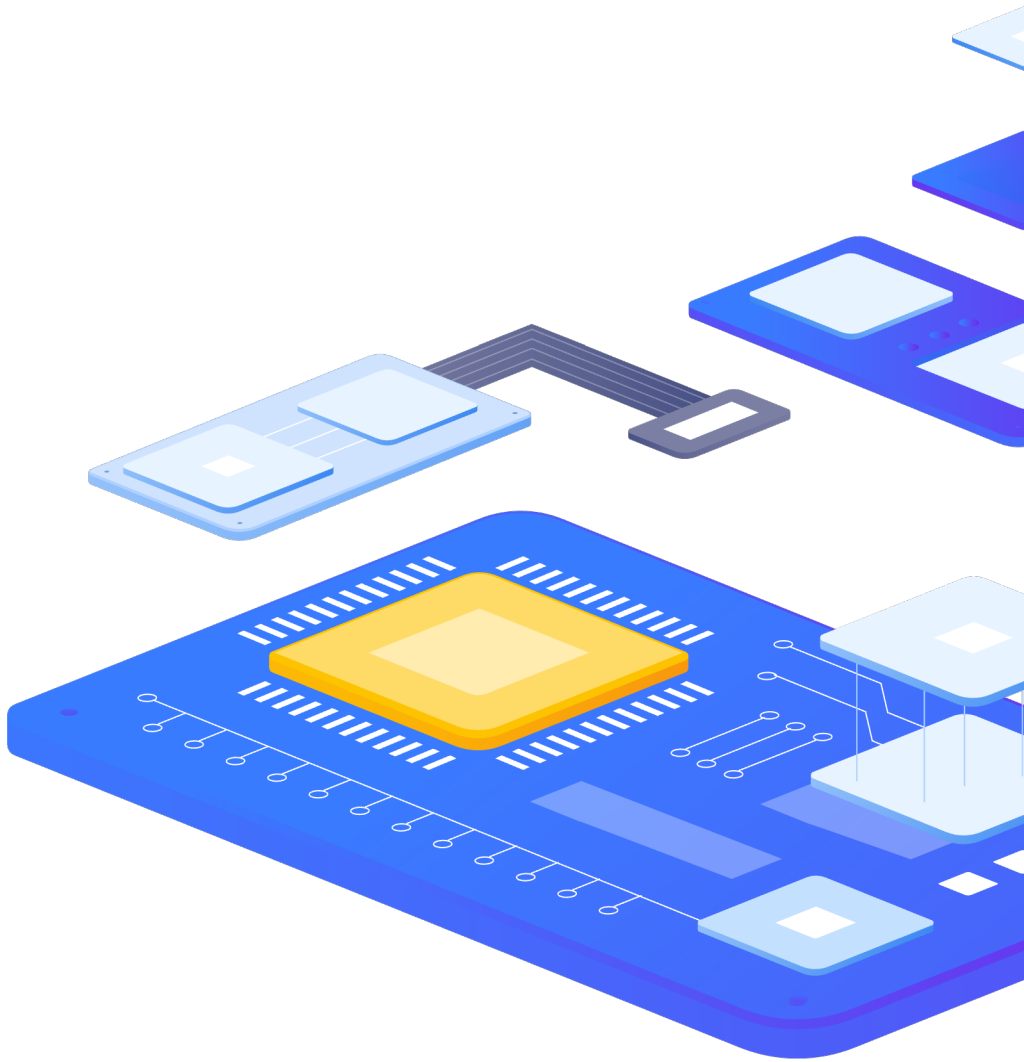


initMAX

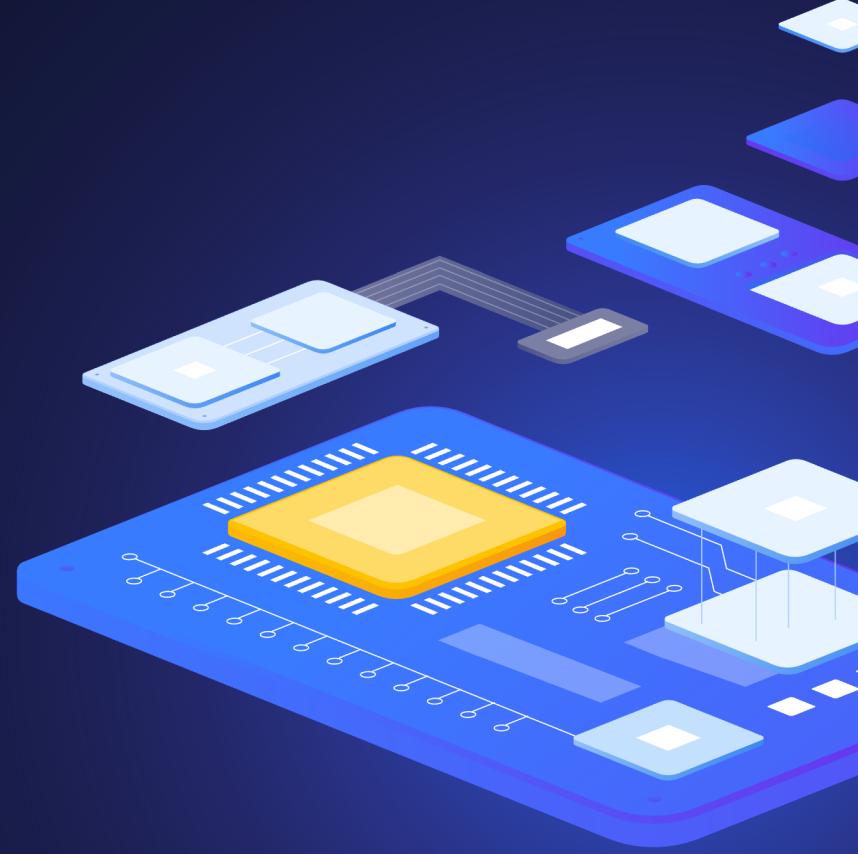
Pros and Cons - when to use AI

- **Use AI** when alert volume is high and rules cannot keep up
- **Use AI** for novel patterns and multi-source events
- **Use AI** when explanation in plain language is valuable
- …

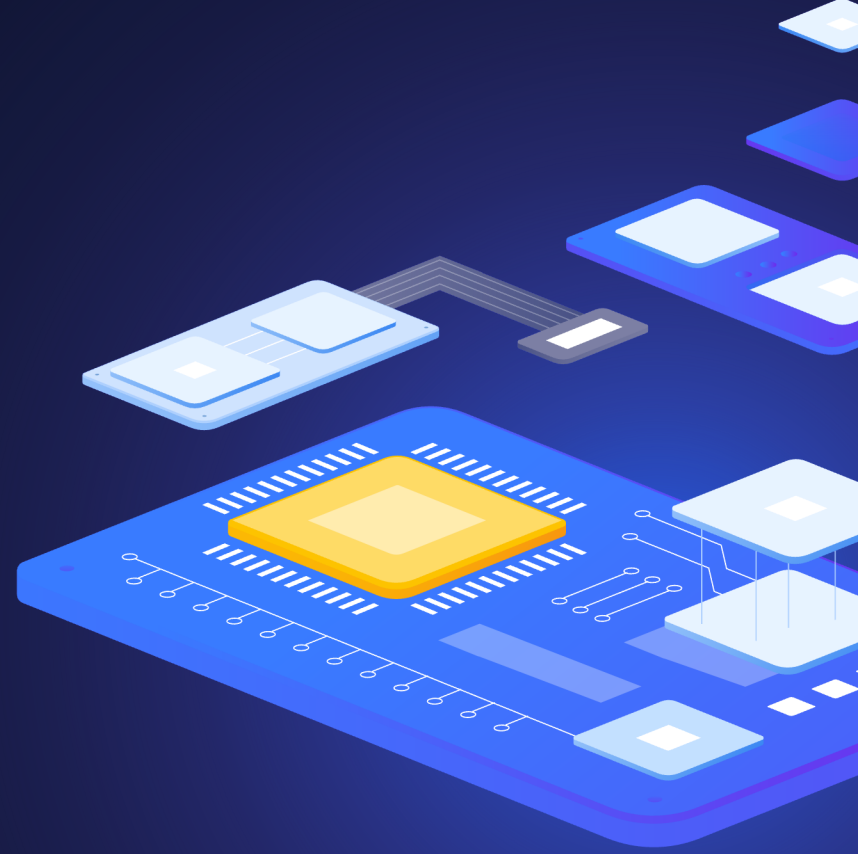
- **Skip AI** when rules already solve it cleanly
- **Skip AI** for latency-critical alerts (sub-second response)
- **Skip AI** when budget for tokens or GPU is not available
- …



DEMO



Questions?



initMAX

Contact us:

	+420 800 244 442
Web:	https://www.initmax.cz
Email:	tomas.hermanek@initmax.cz
LinkedIn:	https://www.linkedin.com/company/initmax
Twitter:	https://twitter.com/initmax
Tomáš Heřmánek:	+420 732 447 184