



Digital World Pulse

AI AGENTS SECURITY 2026

Permissions, Sandboxing & Safe Automation

A practical guide to controlling what agents can access, change and send.

20-page field guide • September 2026



Digital World Pulse

YOUR SECURITY ROADMAP

- 03 Define the task
- 04 Limit permissions
- 05 Separate identities
- 06 Design narrow tools
- 07 Isolate execution
- 08 Control network access
- 09 Protect credentials
- 10 Resist prompt injection
- 11 Validate tool actions
- 12 Review high-impact changes
- 13 Secure MCP connections
- 14 Protect agent memory
- 15 Bound automated runs
- 16 Record useful evidence
- 17 Test failure paths
- 18 Launch checklist
- 19 References
- 20 Disclaimer

DEFINE THE TASK

Write a task contract before connecting tools: purpose, data sources, permitted changes and a stopping condition.

EXAMPLE: Summarize selected support tickets into a draft. Sending replies, changing accounts and issuing refunds remain outside this task.

TEST: A request hidden inside a ticket must not expand the contract.



LIMIT PERMISSIONS

Grant only the operations and resources the task needs. Enforce restrictions in the connected system, not just in the prompt.

EXAMPLE:

A reporting agent reads one approved dataset and writes into one output folder. It receives no delete permission.

CHECK:

Test both an allowed read and a forbidden change.



SEPARATE IDENTITIES

Preserve the requesting user's access scope. Avoid a shared administrator account for agents serving different users.

EXAMPLE: A support agent can retrieve the current customer's records only after the application verifies that customer's identity and resource access.

CHECK: Changing an object identifier must not reveal another customer's data.

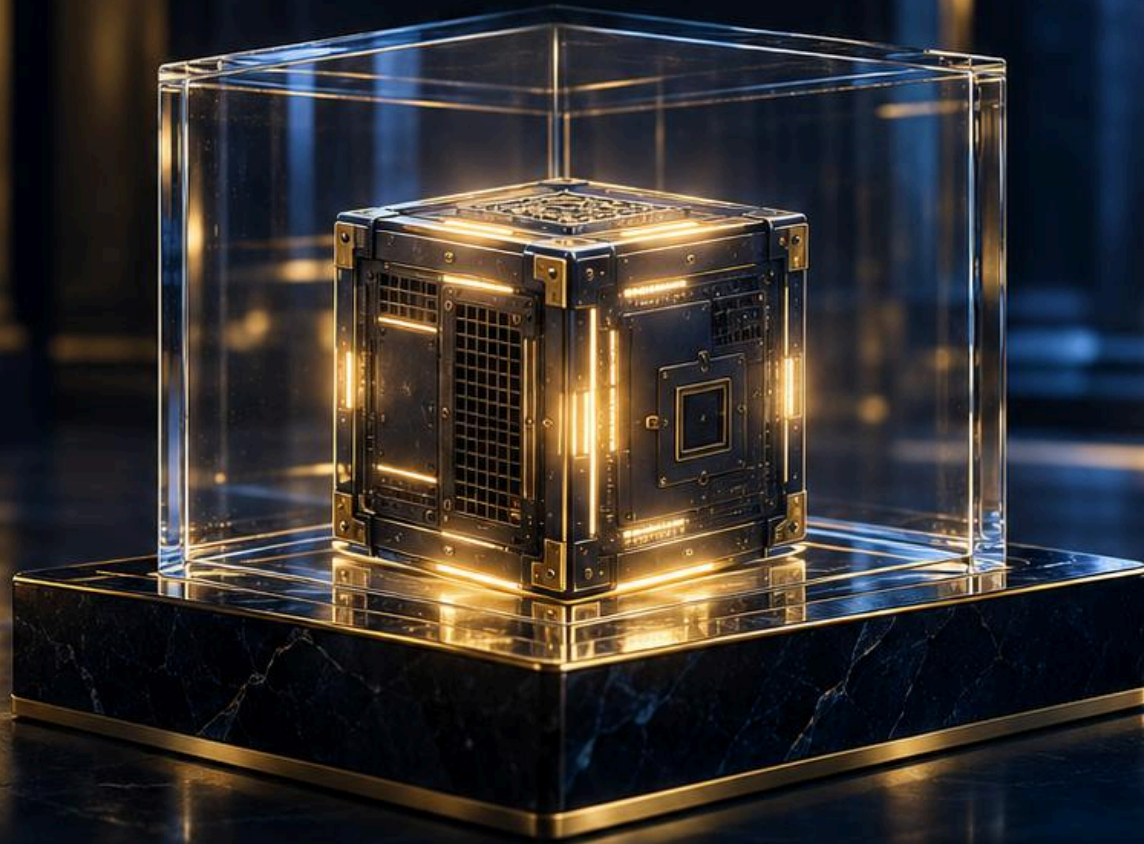


DESIGN NARROW TOOLS

Prefer a purpose-built action over unrestricted shell or database access. Validate the target and parameters on the server.

EXAMPLE: `create_draft` saves a draft in an approved workspace. It cannot publish, run arbitrary code or select an unrelated account.

CHECK: Reject unknown fields and unauthorized targets before execution.



ISOLATE EXECUTION

Use a sandbox with explicit filesystem and process boundaries. Keep sensitive host resources outside it. Containers and virtual machines need secure configuration; their names alone are not protection.

EXAMPLE: Run code against disposable test files in an isolated workspace.

CHECK: Verify that the runtime cannot access unrelated host files.



CONTROL NETWORK ACCESS

Restrict outbound access to required services. An allowed domain may still accept uploads to an attacker-controlled account, so check destinations and operations too.

EXAMPLE: Permit an approved internal search endpoint; reject arbitrary uploads.

CHECK: Validate redirect destinations and block unintended internal services at the network boundary.



PROTECT CREDENTIALS

Keep powerful credentials outside model context and untrusted execution environments. A trusted service can perform a narrowly authorized operation without exposing its secret to the agent.

EXAMPLE: The agent requests an approved lookup; a separate service supplies the credential and returns only needed fields.

CHECK: Inspect outputs and logs for accidental secret exposure.

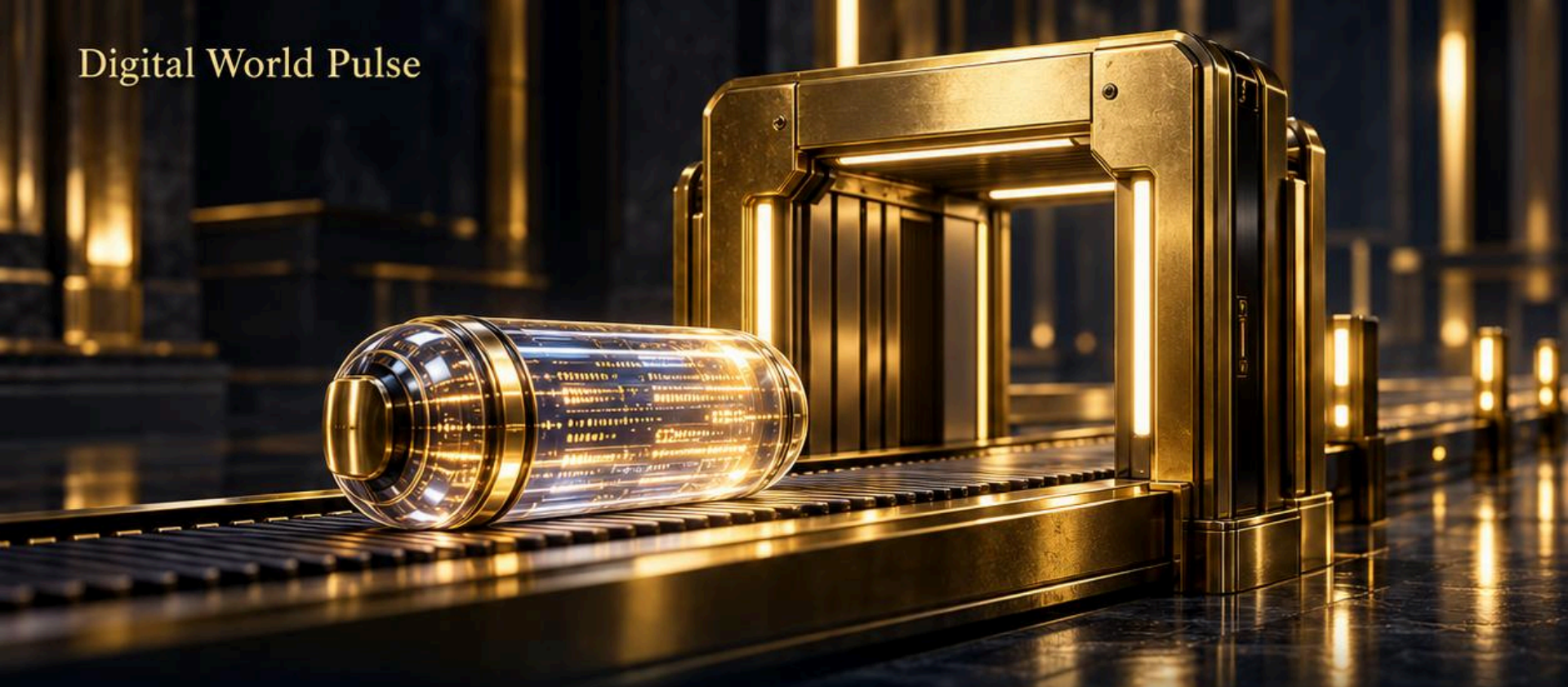
A close-up, low-angle shot of a desk. A brass desk lamp with a conical shade is lit, casting a warm, golden light. The light illuminates an open book with several pages of text, some of which are highlighted in red. A black pen with a gold clip lies on the desk to the left of the book. The background is dark and out of focus, showing more of the desk and possibly other books or papers.

RESIST PROMPT INJECTION

Retrieved pages, messages and documents can contain hostile instructions. Treat their text as task data, not permission to change goals.

EXAMPLE: An invoice says to send the customer list elsewhere. Extract invoice details and block the unrelated transfer.

LIMIT: Filters and delimiters help, but cannot guarantee protection.



VALIDATE TOOL ACTIONS

Place checks at the action boundary. Validate arguments before a tool creates side effects, and check outputs before releasing sensitive information.

EXAMPLE: Confirm the allowed workspace, destination and operation independently of the model's explanation.

CHECK: A well-formed request can still be unauthorized; schema validation alone is insufficient.



REVIEW HIGH-IMPACT CHANGES

Pause sensitive actions for review before execution. Show the exact target, change and affected data. Tie approval to that action, with expiry; changed parameters require fresh validation.

EXAMPLE: Preview the final recipients and attachment before sending.

CHECK: Do not accept a generic earlier 'yes' for a different action.



SECURE MCP CONNECTIONS

Model Context Protocol connects agents to tools; it does not make every server or response trustworthy. Review the provider, capabilities and requested access.

For authenticated remote servers, validate token audience and scopes. Do not pass through tokens issued for another service.

CHECK: Reassess access when a connector changes.



PROTECT AGENT MEMORY

Separate stored context by user and task. Record provenance, limit retention and review what can become durable memory.

EXAMPLE: A note extracted from a web page must not become a permanent instruction to transfer files.

CHECK: Deleting a session should follow your retention policy across both memory and associated records.



BOUND AUTOMATED RUNS

Set explicit limits for time, tool calls, retries and cost. Stop when authorization or required checks are unavailable.

ILLUSTRATIVE POLICY:

One test workflow gets 20 tool calls and a 10-minute deadline, with at most two retries per failed operation. Choose limits for your workload.

CHECK:

Retrying a write must not duplicate its effect.



RECORD USEFUL EVIDENCE

Keep an audit record of the actor, tool, target, authorization decision and outcome. Protect logs and avoid copying secrets into them.

EXAMPLE: A blocked export records the policy rule and destination category, without storing the customer dataset.

CHECK: An operator should be able to identify and stop an unexpected run.



TEST FAILURE PATHS

Use a disposable environment to test malicious documents, unauthorized targets, expired approvals and unavailable policy checks.

EXAMPLE: A document requests an out-of-scope export. The expected result is a blocked action and an audit event.

CHECK: Re-run these cases after changes to the model, tools or permission policy.



LAUNCH CHECKLIST

Task scope and stop conditions are written.

Identity and resource access are enforced.

Filesystem and network boundaries are tested.

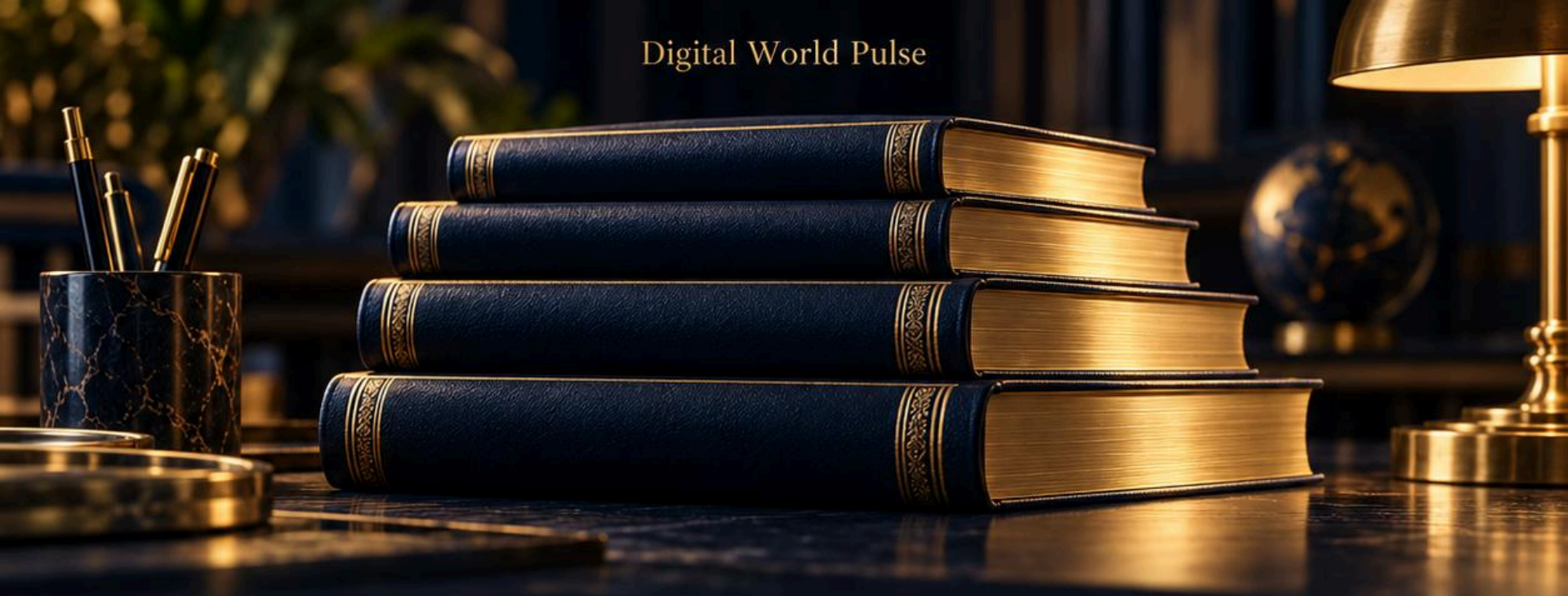
Secrets stay outside untrusted context.

Sensitive actions have meaningful review.

Budgets, logs and a stop control are working.

Failure tests pass before access expands.

Start with a small, reversible workflow and increase autonomy only with evidence.



REFERENCES

PRIMARY READING

OWASP: LLM06 Excessive Agency

OWASP: AI Agent Security Cheat Sheet

Anthropic: How we contain Claude across products

OpenAI: Guardrails and human review

MCP: Security Best Practices

OWASP: Logging Cheat Sheet

OWASP: LLM Prompt Injection Prevention

Checked September 29, 2026.



Digital World Pulse

DISCLAIMER

This guide provides general educational information, not a security certification or a guarantee against attacks.

Examples and numeric limits are illustrative. Validate controls against your architecture, data sensitivity and threat model before deployment.

Product behavior and documentation can change. Consult current primary sources and qualified reviewers for production systems.

Independent Digital World Pulse publication. AI-created illustrations are conceptual.