agentsec: What's Missing in Docs & Examples

Feb 2026

Problem 1: Nobody knows get_inspection_context() exists

In monitor mode (API), agentsec scans everything but never blocks. The scan results (reasons, severity, classifications) are stored in the inspection context. Developers can read them via get_inspection_context() and take their own action.

But this function is never shown in any example. Zero hits across 80+ example files. So a developer using monitor mode has no idea they can access the Decision.

Fix: Update openai_example.py (already uses monitor mode)

Add this after the LLM call:

```
# Check what AI Defense found
ctx = agentsec.get_inspection_context()
decision = ctx.decision
if decision:
    print(f"AI Defense action: {decision.action}")
    print(f"Reasons: {decision.reasons}")
    print(f"Severity: {decision.severity}")
    print(f"Classifications: {decision.classifications}")
```

Fix: Same for mcp_example.py

Add the same pattern after the MCP tool call.

Problem 2: No example of custom allow/block in monitor mode

Monitor mode is meant for developers who want to see results and decide themselves. But every example just calls the LLM, prints the response, and ignores the Decision. Nobody shows the pattern: 'check the decision, then do your own thing.'

Fix: Add a new 1-simple/monitor_mode_example.py

```
from aidefense.runtime import agentsec
agentsec.protect(api_mode_llm="monitor")

from openai import OpenAI
client = OpenAI()

response = client.chat.completions.create(
    model="gpt-4o-mini",
    messages=[{"role": "user", "content": "What is my SSN?"}]
)

# YOUR decision, not agentsec's
ctx = agentsec.get_inspection_context()
decision = ctx.decision

if decision and decision.action == "block":
    print(f"Flagged! Reasons: {decision.reasons}")
    print(f"Severity: {decision.severity}")
    # Your custom action: log it, alert, return safe message, etc.
else:
    print(response.choices[0].message.content)
```

Problem 3: API mode vs Gateway mode -- visibility difference not documented

Big deal that nobody mentions: in API mode you get full Decision objects with reasons, severity, classifications, rules. In Gateway mode you get nothing -- the gateway handles inspection internally and the SDK never sees the results.

The README just says:

```
| API Mode    | SDK inspects requests via AI Defense API |
| Gateway Mode| SDK routes traffic through Gateway proxy |
```

It never says: 'if you pick Gateway, you lose all programmatic access to scan results.' The 'why' is only in the AI Defense dashboard.

Fix: Add a comparison note to README.md

Under 'Integration Modes', add something like:

```
**Important difference:**
- API mode: Your code gets full scan results (Decision object
  with reasons, severity, classifications). You can inspect and
  act on them.
- Gateway mode: The gateway handles inspection internally.
  Your code does NOT receive scan results. Check the AI Defense
  dashboard for inspection details.
```

Problem 4: 'Why was it allowed?' is never shown

The README only shows how to see reasons when something is BLOCKED (via SecurityPolicyError in enforce mode). But even for allowed requests, the Decision object has severity, classifications, and rules populated. This is useful for observability/audit but nobody knows it exists.

Fix: Add to README 'Advanced Usage' section

```
# Works for BOTH allowed and blocked decisions:
ctx = agentsec.get_inspection_context()
if ctx.decision:
    print(ctx.decision.action)           # "allow" or "block"
    print(ctx.decision.reasons)           # why
    print(ctx.decision.severity)          # how bad
    print(ctx.decision.classifications)   # what type
    print(ctx.decision.event_id)          # for audit trail
```

Problem 5: Decision docs use the wrong API level

The README shows Decision fields using the low-level inspector:

```
decision = inspector.inspect_conversation(messages, metadata)
decision.action
decision.is_safe
```

But someone using agentsec.protect() with auto-patching doesn't have an 'inspector' object. They need get_inspection_context() to get the Decision. The README never connects these two.

Fix: Update the 'Inspection Results' section in README

Show both paths:

```
# If using agentsec.protect() (recommended):
ctx = agentsec.get_inspection_context()
decision = ctx.decision

# If using the inspector directly:
decision = inspector.inspect_conversation(messages, metadata)
```

Summary: What to change

- openai_example.py -- add get_inspection_context() after the LLM call
- mcp_example.py -- add get_inspection_context() after the MCP tool call
- New file: 1-simple/monitor_mode_example.py -- custom allow/block pattern
- README.md (examples/agentsec/) -- expand Inspection Modes table, add API vs Gateway visibility note
- README.md (examples/agentsec/) -- update 'Inspection Results' to show get_inspection_context()
- README.md (root) -- add get_inspection_context() to the Runtime Protection examples