Stress Testing and Reliability Validation of Process Monitoring Module

This section documents stress testing and reliability evaluation performed on the Windows process monitoring module to assess its stability, correctness, and operational resilience under realistic and adversarial conditions.

▼ Test 1: Single Process Creation

Description:

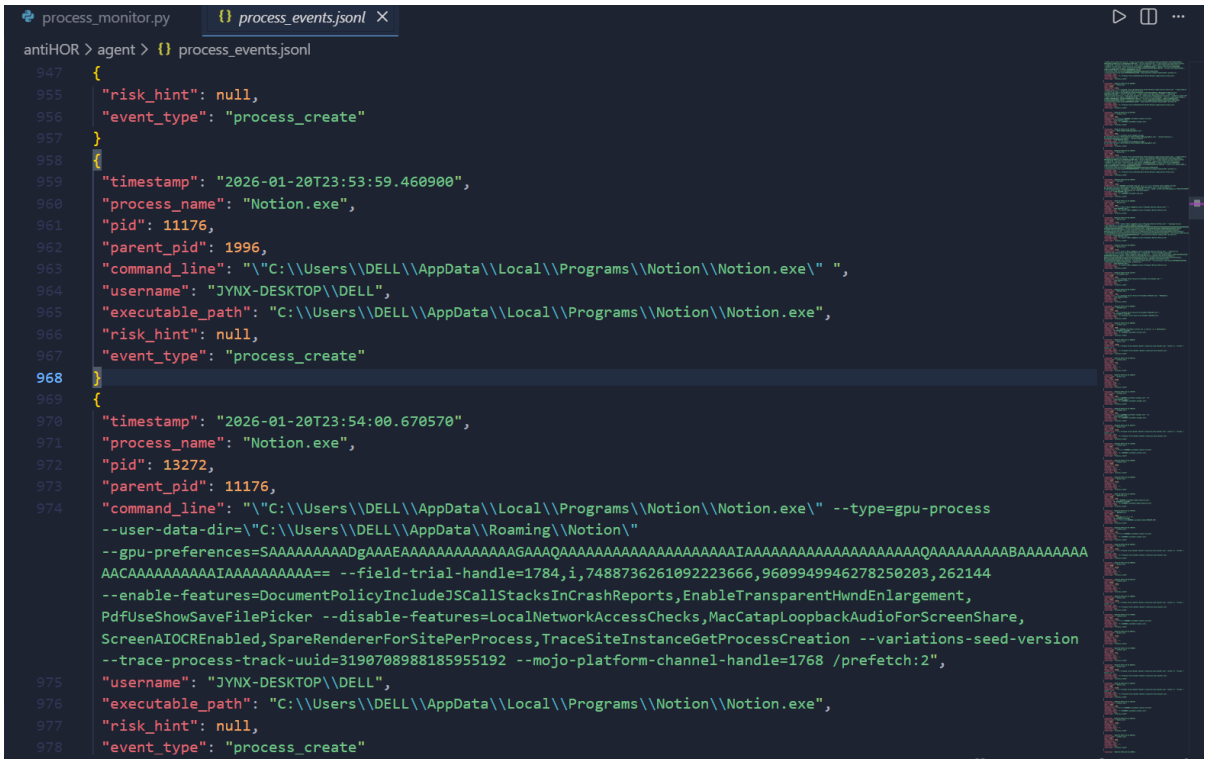
Multiple applications and desktop apps were launched as a basic prerequisite to test the operational feasibility of the script.

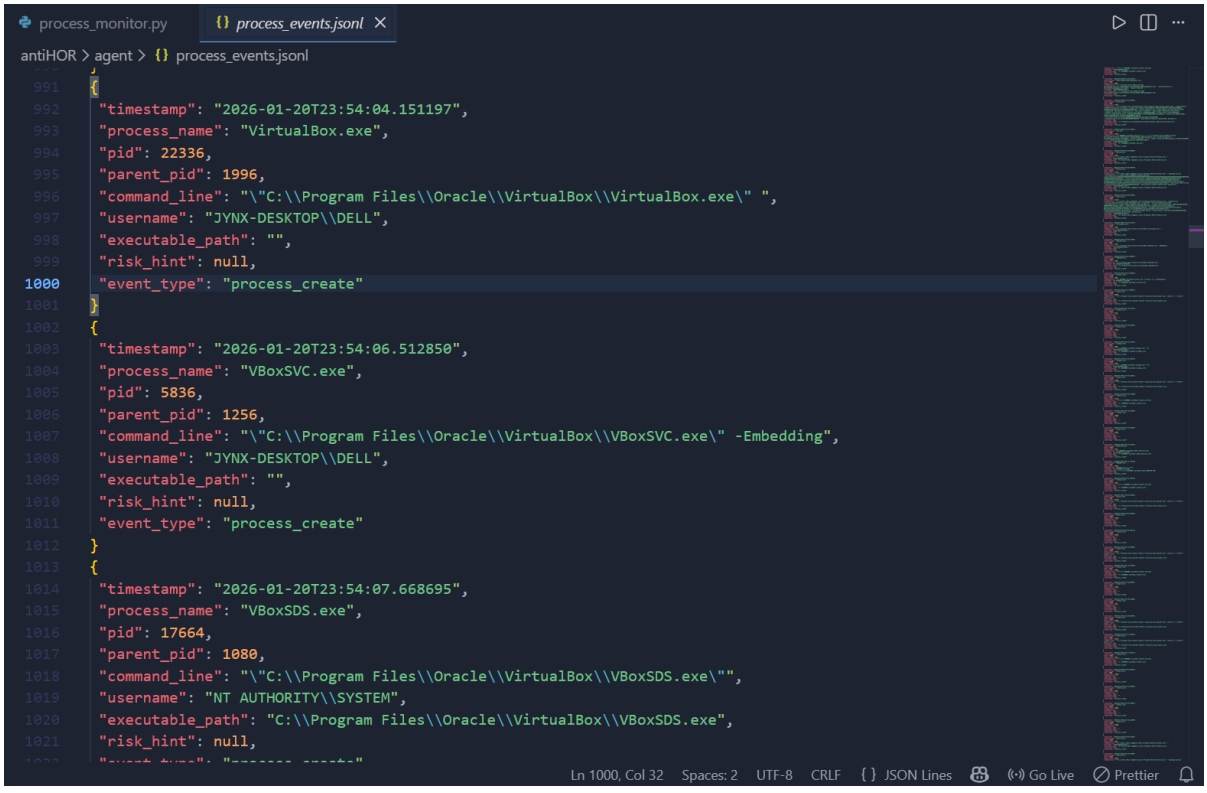
```
Executing:
    brave.exe
    notion.exe
    virtualbox.exe
```

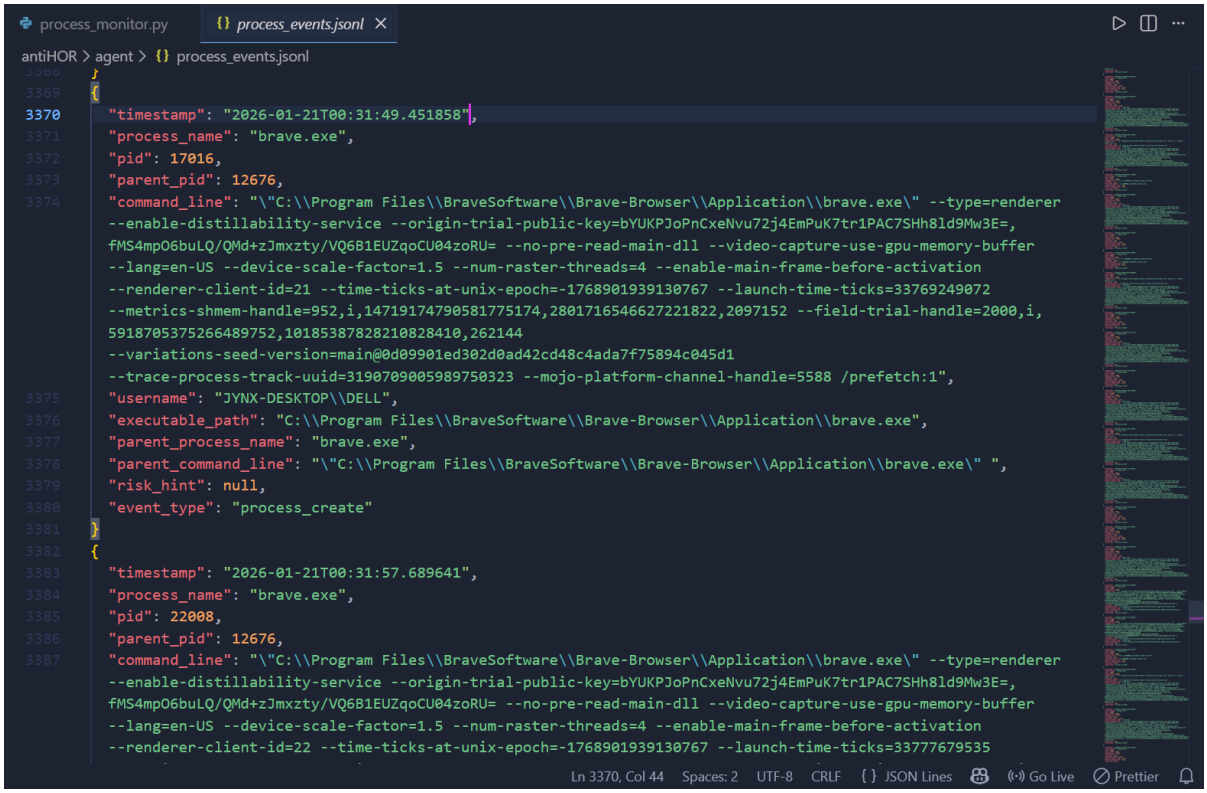
Expected Behavior:

The monitoring agent should capture all process creation events i.e. applications as and when launched without missing entries, crashing, or blocking execution.

Result:


process_monitor.py
process_events.jsonl
antiHOR > agent > process_events.jsonl
{
"risk_hint": null,
"event_type": "process_create"
}
{
"timestamp": "2026-01-20T23:53:59.460900",
"process_name": "Notion.exe",
"pid": 11176,
"parent_pid": 1996,
"command_line": "\"C:\\Users\\DELL\\AppData\\Local\\Programs\\Notion\\Notion.exe\" ",
"username": "JYNX-DESKTOP\\DELL",
"executable_path": "C:\\Users\\DELL\\AppData\\Local\\Programs\\Notion\\Notion.exe",
"risk_hint": null,
"event_type": "process_create"
}
{
"timestamp": "2026-01-20T23:54:00.670570",
"process_name": "Notion.exe",
"pid": 13272,
"parent_pid": 11176,
"command_line": "\"C:\\Users\\DELL\\AppData\\Local\\Programs\\Notion\\Notion.exe\" --type=gpu-process
--user-data-dir=\"C:\\Users\\DELL\\AppData\\Roaming\\Notion\"
--gpu-preferences=SAAAAAAAAADgAAAEAAAAAAAAAAAAAGAAAQAAAAAAAAAAAAAAAAAAAIAAAAAAAAAAAAAAAAAAAQAAAAAAAABAAAAAAAA
AACAAAAAAAAAAIAAAAAAAAAA== --field-trial-handle=1784,i,7488736261092223666,9609949946278250203,262144
--enable-features=DocumentPolicyIncludeJSCallStacksInCrashReports,EnableTransparentHwndEnlargement,
PdfUseShowSaveFilePicker --disable-features=LocalNetworkAccessChecks,MacCatapLoopbackAudioForScreenShare,
ScreenAIOCREnabled,SpareRendererForSitePerProcess,TraceSiteInstanceGetProcessCreation --variations-seed-version
--trace-process-track-uuid=3190708988185955192 --mojo-platform-channel-handle=1768 /prefetch:2",
"username": "JYNX-DESKTOP\\DELL",
"executable_path": "C:\\Users\\DELL\\AppData\\Local\\Programs\\Notion\\Notion.exe",
"risk_hint": null,
"event_type": "process_create"


process_monitor.py
process_events.jsonl
antiHOR > agent > process_events.jsonl
{
"timestamp": "2026-01-20T23:54:04.151197",
"process_name": "VirtualBox.exe",
"pid": 22336,
"parent_pid": 1996,
"command_line": "\"C:\\Program Files\\Oracle\\VirtualBox\\VirtualBox.exe\" ",
"username": "JYNX-DESKTOP\\DELL",
"executable_path": "",
"risk_hint": null,
"event_type": "process_create"
}
{
"timestamp": "2026-01-20T23:54:06.512850",
"process_name": "VBoxSVC.exe",
"pid": 5836,
"parent_pid": 1256,
"command_line": "\"C:\\Program Files\\Oracle\\VirtualBox\\VBoxSVC.exe\" -Embedding",
"username": "JYNX-DESKTOP\\DELL",
"executable_path": "",
"risk_hint": null,
"event_type": "process_create"
}
{
"timestamp": "2026-01-20T23:54:07.668695",
"process_name": "VBoxSDS.exe",
"pid": 17664,
"parent_pid": 1080,
"command_line": "\"C:\\Program Files\\Oracle\\VirtualBox\\VBoxSDS.exe\"",
"username": "NT AUTHORITY\\SYSTEM",
"executable_path": "C:\\Program Files\\Oracle\\VirtualBox\\VBoxSDS.exe",
"risk_hint": null,
Ln 1000, Col 32
Spaces: 2
UTF-8
CRLF
{ } JSON Lines
Go Live
Prettier



Observed Behavior:

All process creation events were logged correctly.
No crashes, deadlocks, or delays were observed.

▼ Test 2: PowerShell Variants

Description:
PowerShell execution patterns were tested,

Executing:

```
powershell -Command "Get-Process | Select-Object -First 5"
```

Expected Behavior:

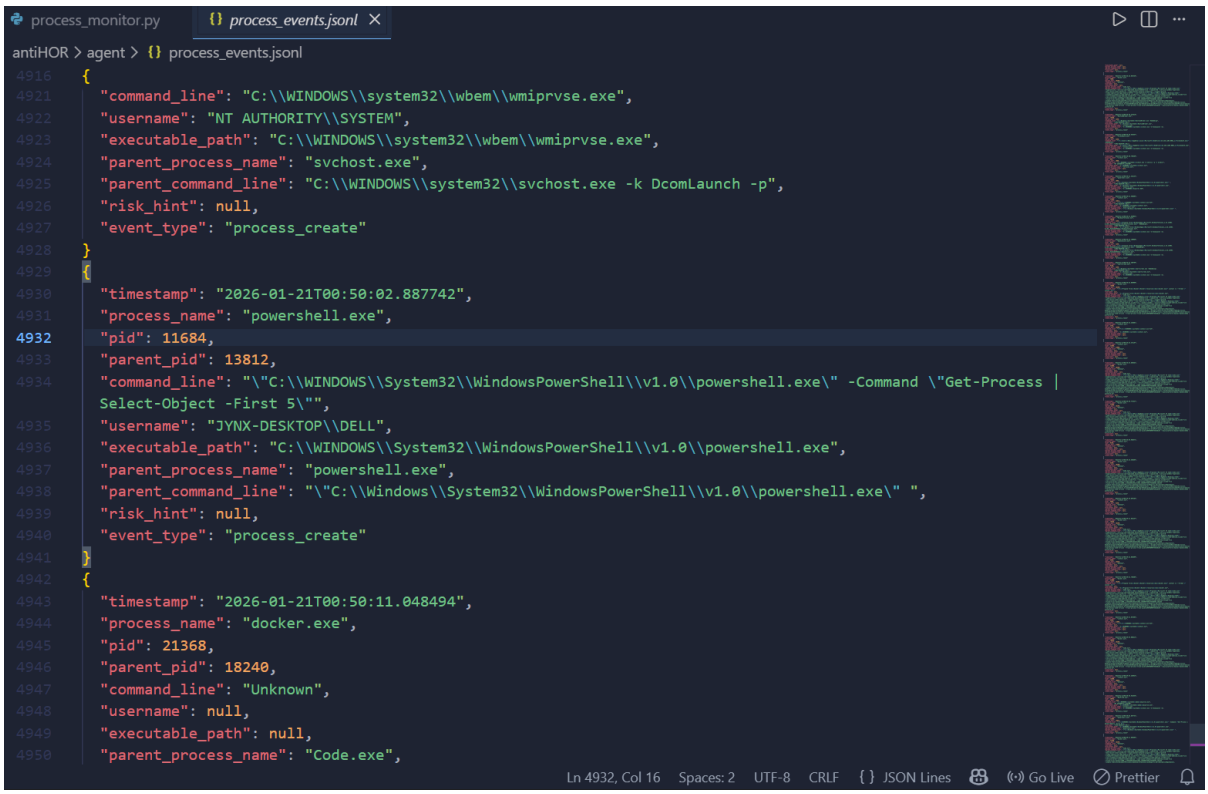
The monitoring agent should reliably capture PowerShell process creation events along with relevant metadata such as parent process, command-line arguments, and execution context.

Result:

```
PS C:\Users\DELL> powershell -Command "Get-Process | Select-Object -First 5"

Handles  NPM(K)    PM(K)      WS(K)     CPU(s)      Id  SI ProcessName
-------  ------    -----      -----     ------      --  -- -----------
    242      14     4896      16240                8244   0 AggregatorHost
    141      10     1796       4852                4884   0 armsvc
    334      13     8104      21892       1.06   21632   0 audiodg
    454      20    22088      50656       1.09   10496   5 backgroundTaskHost
    336      28    37892      75936       9.27    1424   5 brave


PS C:\Users\DELL>
```



Observed Behavior:

PowerShell process creation events were logged successfully. Variations in command-line visibility were observed depending on invocation method, but the agent remained stable and continued logging without interruption.

Rationale:

Human-operated ransomware heavily relies on PowerShell for reconnaissance, lateral movement, payload staging, and defense evasion,

making robust observation of PowerShell execution patterns essential for early detection.

▼ Test 3: Burst Process Creation

Description:

Multiple short-lived processes were spawned in rapid succession using batch scripts and command loops to simulate burst execution scenarios.

Expected Behavior:

The monitoring agent should capture all process creation events without missing entries, crashing, or blocking execution.

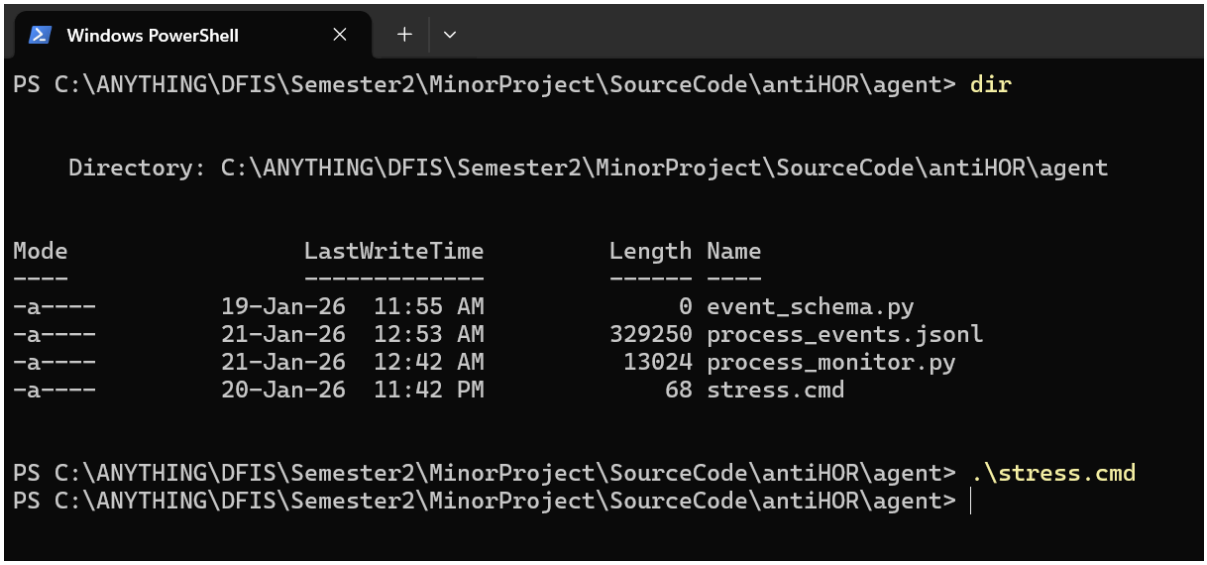
Action: Creating a script

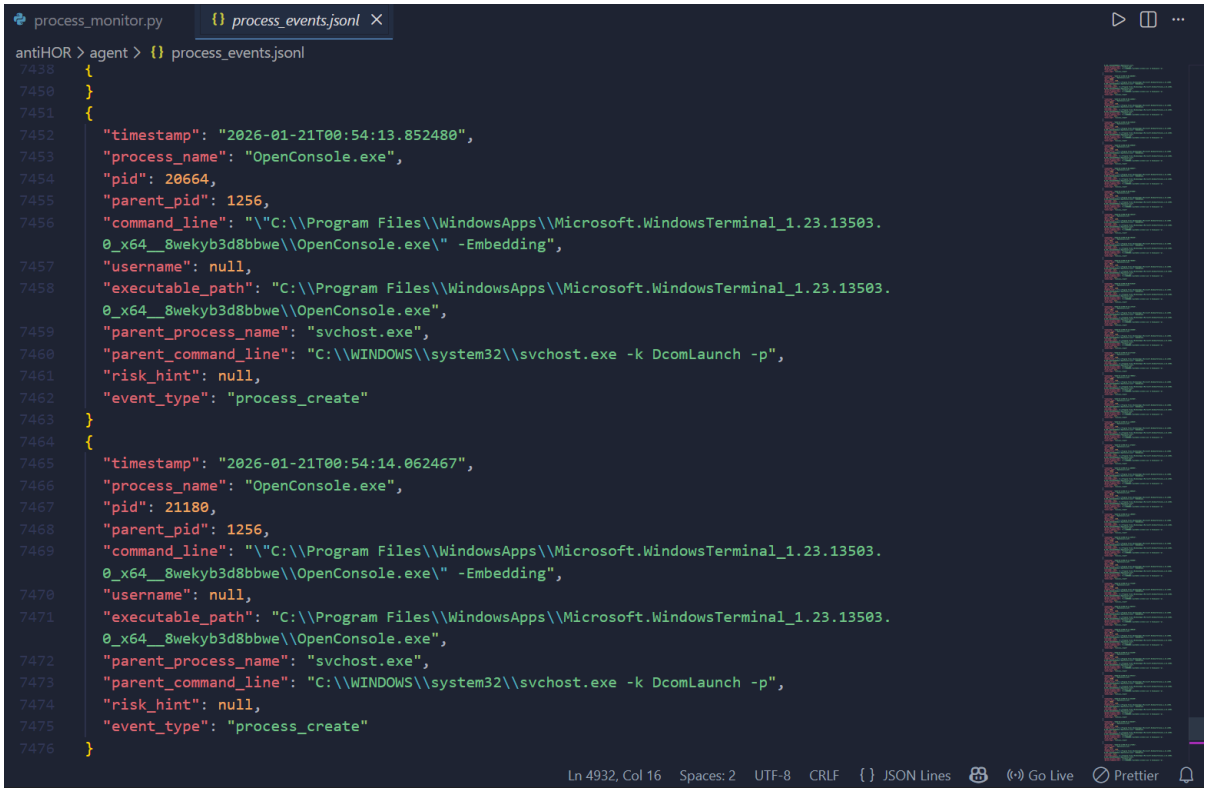
```
@echo off
for /L %%i in (1,1,50) do (
  start cmd /c echo %%i
)
```

Expected behavior:

- No crash
- No missed events
- JSON remains valid

Result:





Observed Behavior:

All process creation events were logged correctly.
No crashes, deadlocks, or delays were observed.

Rationale:

Human-operated ransomware frequently deploys tools in bursts during initial access, lateral movement, and staging phases, making burst process creation a realistic adversarial pattern.

▼ Test 4: Sustained Activity

Description:

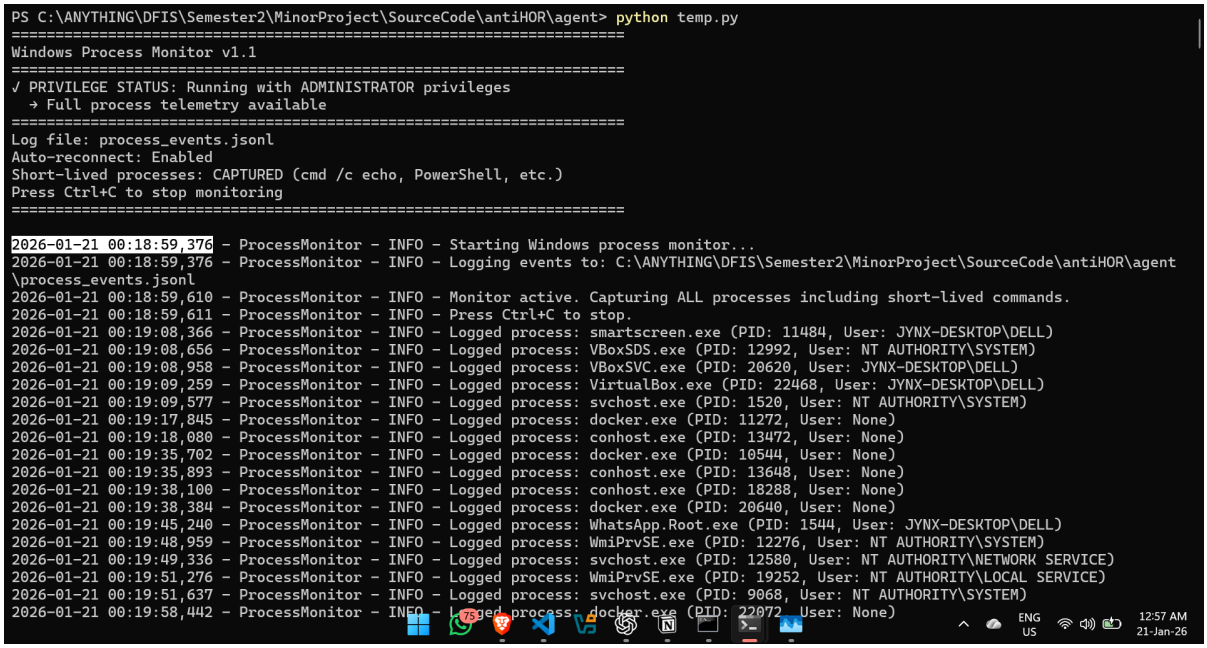
The monitoring agent was allowed to run continuously for an extended period while normal user activity occurred in the background.
Leaving the agent running for 30–60 mins while:

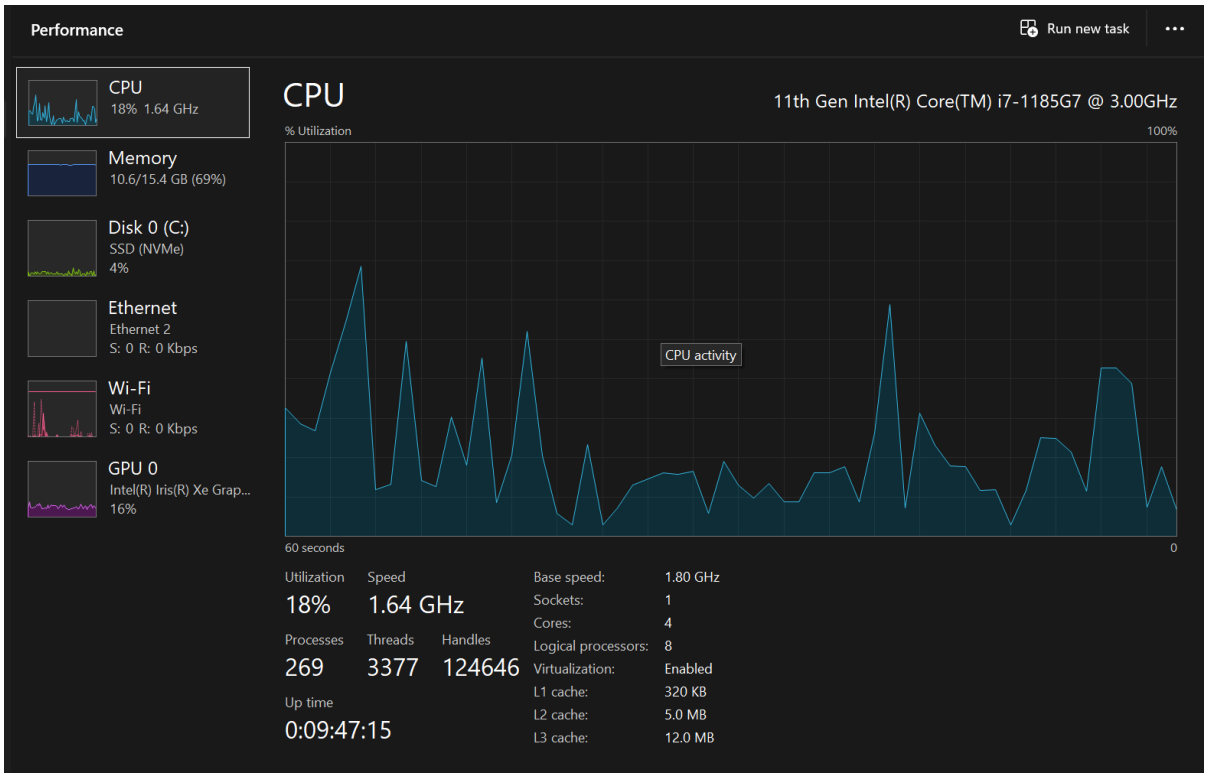
- Browsing
- Using VS Code
- Open/close apps

Expected Behavior:

The agent should remain stable, consume predictable resources, and continue logging events reliably.

Result:





Observed Behavior:

The agent remained stable with consistent logging behavior and no memory leaks or performance degradation.

Rationale:

Human-operated ransomware campaigns can persist for days or weeks; long-lived defensive agents must remain reliable throughout prolonged dwell times.

▼ Test 5: WMI Failure Simulation

Description:

Simulated failure scenarios were introduced where WMI queries intermittently failed or returned incomplete results.

Manually stopping WMI service (briefly):

```
net stop winmgmt
```

Then restart:

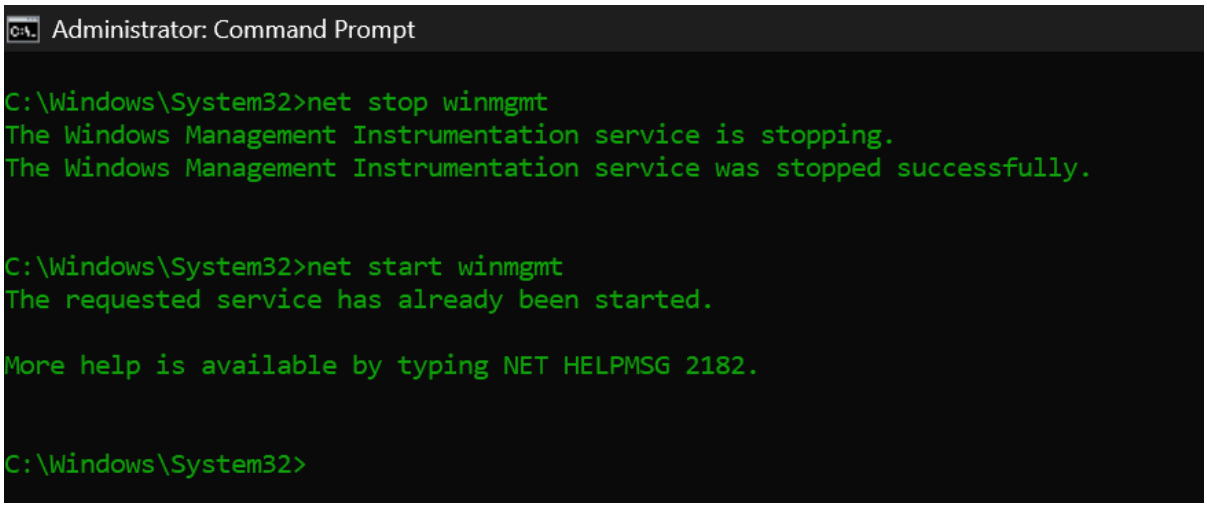
```
net start winmgmt
```

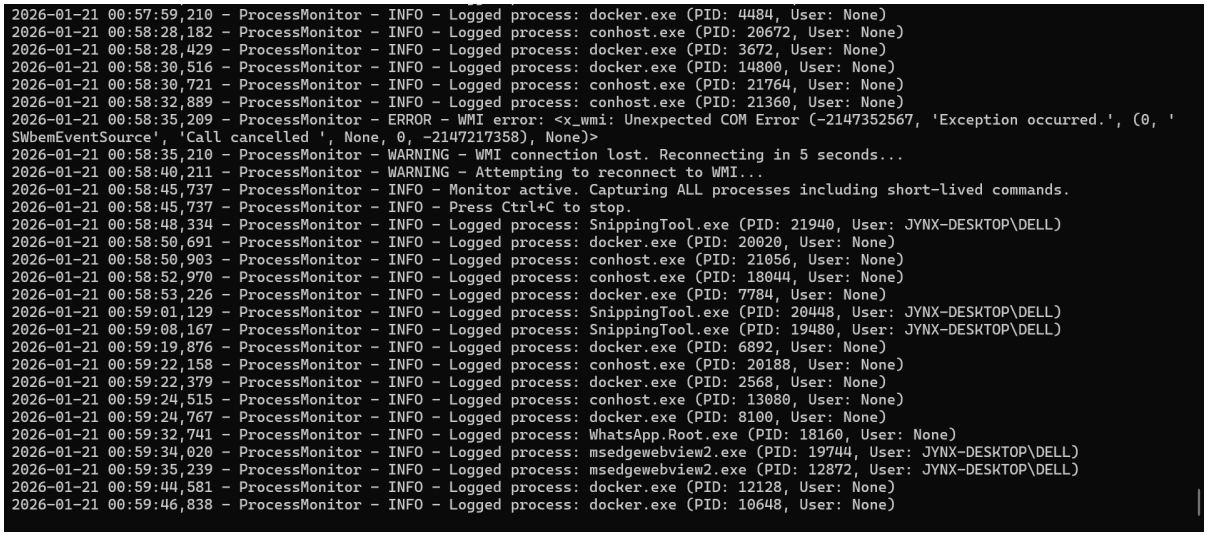
Expected Behavior:

The agent should handle exceptions cleanly and continue monitoring without crashing.

- Agent does NOT crash
- Logs error
- Recovers automatically

Result:





Observed Behavior:

WMI failures were logged, and monitoring resumed automatically without manual intervention.

Rationale:

WMI instability is common in compromised or heavily loaded systems, especially during attacker activity; resilience to telemetry degradation is critical.

▼ Test 6: Permission Degradation

Description:

The process monitor was executed under non-administrative privileges to evaluate behavior under restricted access conditions.
Running the agent:

- Once as admin
- Once as standard user

Expected Behavior:

The agent should continue functioning gracefully, logging available data while handling permission errors without crashing.

- Reduced fields when non-admin
- No crashes
- Clear logs

Result:

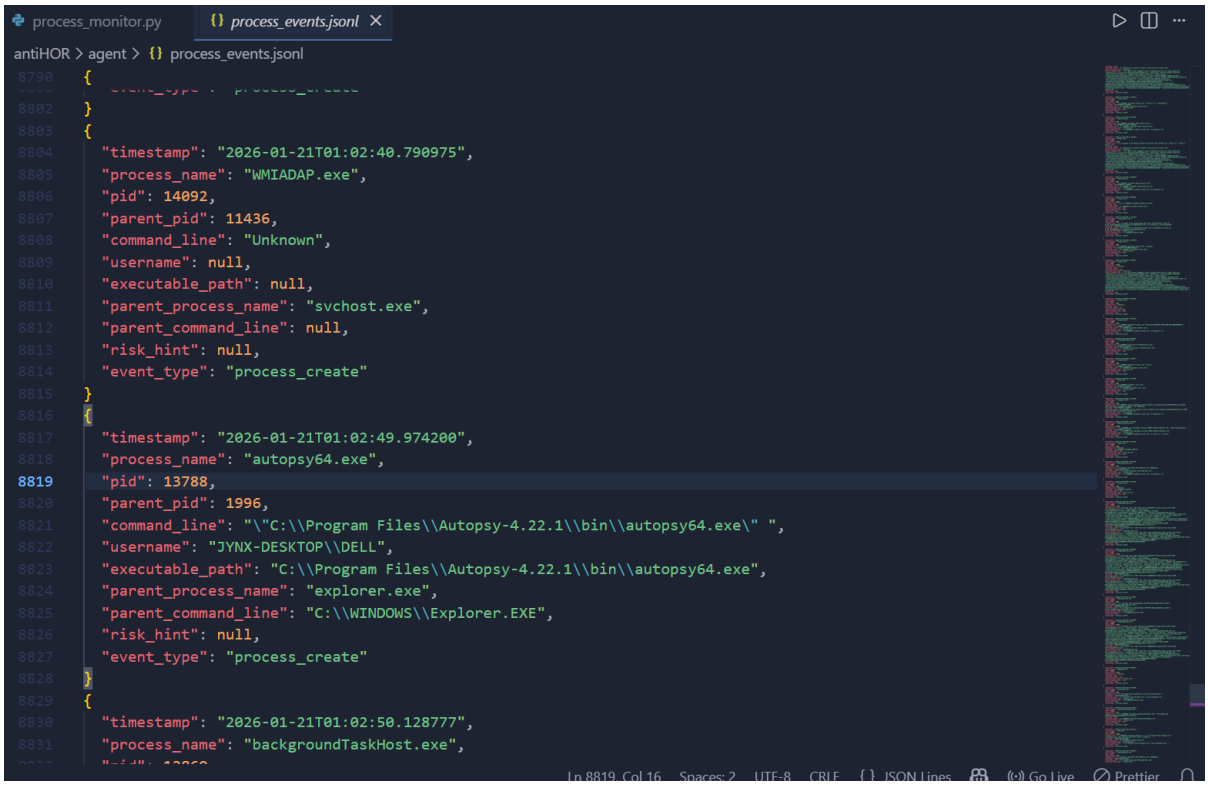
```
PS C:\ANYTHING\DFIS\Semester2\MinorProject\SourceCode\antiHOR\agent> python temp.py
======================================================================
Windows Process Monitor v1.1
======================================================================
✓ PRIVILEGE STATUS: Running with ADMINISTRATOR privileges
  → Full process telemetry available
======================================================================
Log file: process_events.jsonl
Auto-reconnect: Enabled
Short-lived processes: CAPTURED (cmd /c echo, PowerShell, etc.)
Press Ctrl+C to stop monitoring
======================================================================

2026-01-21 00:18:59,376 - ProcessMonitor - INFO - Starting Windows process monitor...
2026-01-21 00:18:59,376 - ProcessMonitor - INFO - Logging events to: C:\ANYTHING\DFIS\Semester2\MinorProject\SourceCode\antiHOR\agent
\process_events.jsonl
2026-01-21 00:18:59,610 - ProcessMonitor - INFO - Monitor active. Capturing ALL processes including short-lived commands.
2026-01-21 00:18:59,611 - ProcessMonitor - INFO - Press Ctrl+C to stop.
2026-01-21 00:19:08,366 - ProcessMonitor - INFO - Logged process: smartscreen.exe (PID: 11484, User: JYNX-DESKTOP\DELL)
2026-01-21 00:19:08,656 - ProcessMonitor - INFO - Logged process: VBoxSDS.exe (PID: 12992, User: NT AUTHORITY\SYSTEM)
2026-01-21 00:19:08,958 - ProcessMonitor - INFO - Logged process: VBoxSVC.exe (PID: 20620, User: JYNX-DESKTOP\DELL)
2026-01-21 00:19:09,259 - ProcessMonitor - INFO - Logged process: VirtualBox.exe (PID: 22468, User: JYNX-DESKTOP\DELL)
2026-01-21 00:19:09,577 - ProcessMonitor - INFO - Logged process: svchost.exe (PID: 1520, User: NT AUTHORITY\SYSTEM)
2026-01-21 00:19:17,845 - ProcessMonitor - INFO - Logged process: docker.exe (PID: 11272, User: None)
2026-01-21 00:19:18,080 - ProcessMonitor - INFO - Logged process: conhost.exe (PID: 13472, User: None)
2026-01-21 00:19:35,702 - ProcessMonitor - INFO - Logged process: docker.exe (PID: 10544, User: None)
2026-01-21 00:19:35,893 - ProcessMonitor - INFO - Logged process: conhost.exe (PID: 13648, User: None)
2026-01-21 00:19:38,100 - ProcessMonitor - INFO - Logged process: conhost.exe (PID: 18288, User: None)
2026-01-21 00:19:38,384 - ProcessMonitor - INFO - Logged process: docker.exe (PID: 20640, User: None)
2026-01-21 00:19:45,240 - ProcessMonitor - INFO - Logged process: WhatsApp.Root.exe (PID: 1544, User: JYNX-DESKTOP\DELL)
2026-01-21 00:19:48,959 - ProcessMonitor - INFO - Logged process: WmiPrvSE.exe (PID: 12276, User: NT AUTHORITY\SYSTEM)
2026-01-21 00:19:49,336 - ProcessMonitor - INFO - Logged process: svchost.exe (PID: 12580, User: NT AUTHORITY\NETWORK SERVICE)
2026-01-21 00:19:51,276 - ProcessMonitor - INFO - Logged process: WmiPrvSE.exe (PID: 19252, User: NT AUTHORITY\LOCAL SERVICE)
2026-01-21 00:19:51,637 - ProcessMonitor - INFO - Logged process: svchost.exe (PID: 9068, User: NT AUTHORITY\SYSTEM)
2026-01-21 00:19:58,442 - ProcessMonitor - INFO - Logged process: docker.exe (PID: 22072, User: None)
```

```
Command Prompt - python p

C:\ANYTHING\DFIS\Semester2\MinorProject\SourceCode\antiHOR\agent>python process_monitor.py
======================================================================
Windows Process Monitor v1.1
======================================================================
✗ PRIVILEGE STATUS: Running WITHOUT administrator privileges
  → Limited telemetry (some processes may be inaccessible)
  → Recommendation: Run as Administrator for complete monitoring
======================================================================
Log file: process_events.jsonl
Auto-reconnect: Enabled
Short-lived processes: CAPTURED (cmd /c echo, PowerShell, etc.)
Press Ctrl+C to stop monitoring
======================================================================

2026-01-21 01:02:24,188 - ProcessMonitor - INFO - Starting Windows process monitor...
2026-01-21 01:02:24,189 - ProcessMonitor - INFO - Logging events to: C:\ANYTHING\DFIS\Semester2\MinorProject\SourceCode\antiHOR\agent
\process_events.jsonl
2026-01-21 01:02:24,362 - ProcessMonitor - INFO - Monitor active. Capturing ALL processes including short-lived commands.
2026-01-21 01:02:24,363 - ProcessMonitor - INFO - Press Ctrl+C to stop.
2026-01-21 01:02:33,778 - ProcessMonitor - INFO - Logged process: WhatsApp.Root.exe (PID: 10564, User: JYNX-DESKTOP\DELL)
2026-01-21 01:02:36,104 - ProcessMonitor - INFO - Logged process: msedgewebview2.exe (PID: 19004, User: JYNX-DESKTOP\DELL)
2026-01-21 01:02:40,791 - ProcessMonitor - INFO - Logged process: WMIADAP.exe (PID: 14092, User: None)
2026-01-21 01:02:49,975 - ProcessMonitor - INFO - Logged process: autopsy64.exe (PID: 13788, User: JYNX-DESKTOP\DELL)
2026-01-21 01:02:50,129 - ProcessMonitor - INFO - Logged process: backgroundTaskHost.exe (PID: 13860, User: None)
2026-01-21 01:02:56,838 - ProcessMonitor - INFO - Logged process: cmd.exe (PID: 20360, User: JYNX-DESKTOP\DELL)
2026-01-21 01:02:57,173 - ProcessMonitor - INFO - Logged process: RuntimeBroker.exe (PID: 7612, User: JYNX-DESKTOP\DELL)
2026-01-21 01:02:58,036 - ProcessMonitor - INFO - Logged process: conhost.exe (PID: 1036, User: None)
2026-01-21 01:02:58,421 - ProcessMonitor - INFO - Logged process: docker.exe (PID: 15448, User: None)
2026-01-21 01:02:58,659 - ProcessMonitor - INFO - Logged process: java.exe (PID: 22244, User: None)
2026-01-21 01:02:59,039 - ProcessMonitor - INFO - Logged process: conhost.exe (PID: 7064, User: JYNX-DESKTOP\DELL)
2026-01-21 01:02:59,553 - ProcessMonitor - INFO - Logged process: java.exe (PID: 22088, User: JYNX-DESKTOP\DELL)
2026-01-21 01:02:59,919 - ProcessMonitor - INFO - Logged process: java.exe (PID: 22164, User: JYNX-DESKTOP\DELL)
2026-01-21 01:03:00,546 - ProcessMonitor - INFO - Logged process: conhost.exe (PID: 17168, User: JYNX-DESKTOP\DELL)
2026-01-21 01:03:00,988 - ProcessMonitor - INFO - Logged process: docker.exe (PID: 21020, User: None)
```



Observed Behavior:

Some metadata fields (e.g., system-level processes, executable paths) were unavailable, but monitoring continued without failure.

Rationale:

Attackers may intentionally degrade system visibility or operate in restricted environments; defensive tooling must fail gracefully rather than terminate.

▼ Test 7: Log Corruption Test

Description:

The event log file was manually modified, truncated, and temporarily locked during runtime to test logging robustness.
While the agent is running:

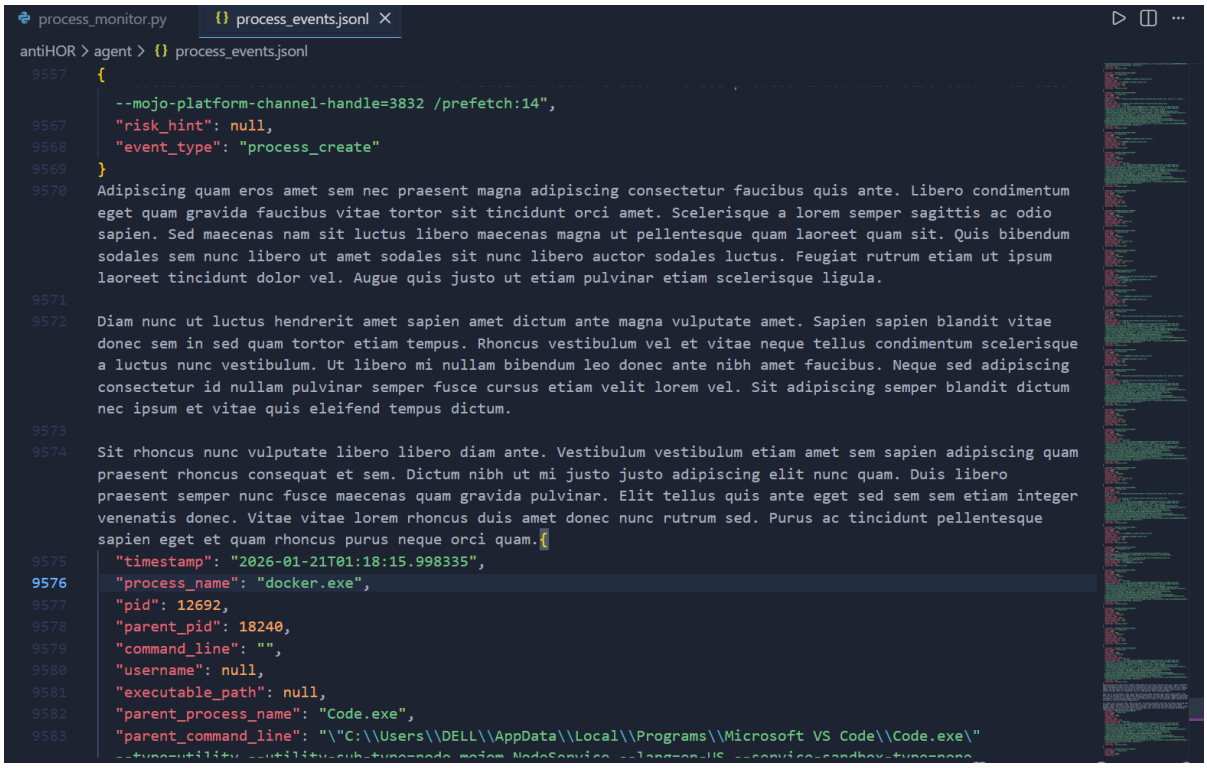
- Open `process_events.jsonl`
- Force-write junk
- Save

Expected Behavior:

The agent should either continue appending safely or recover without data corruption.
Agent continues logging

- Does not crash
- Appends clean JSON

Result:



Observed Behavior:

The agent continued operation without crashing, and subsequent events were logged correctly.

Rationale:

Human-operated ransomware frequently targets logs to erase traces; defensive telemetry systems must tolerate partial log tampering.

▼ Test 8: Disk Stress

Description:

High-volume disk write activity was generated concurrently with process monitoring to simulate heavy I/O conditions, including large file creation, modification, and deletion operations.

Let logs grow:

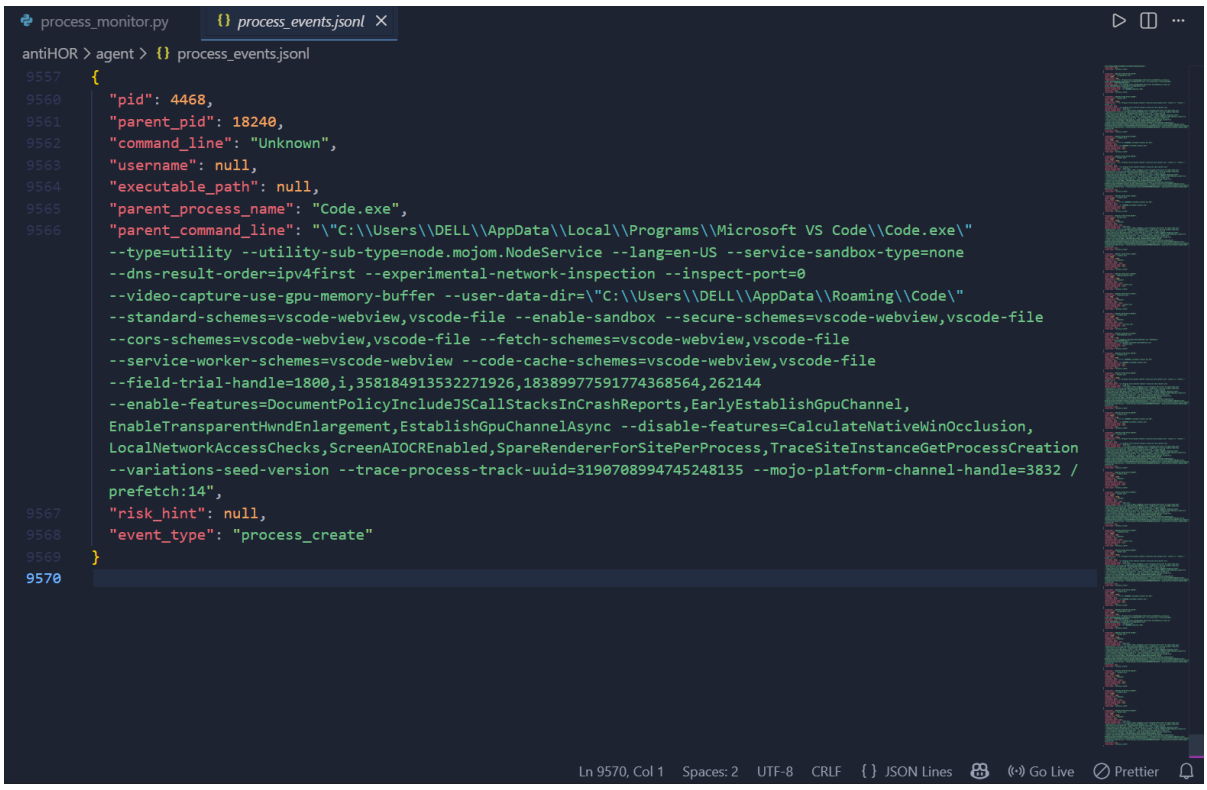
- 8k+ Events

Expected Behavior:

The monitoring agent should continue capturing process events accurately without performance degradation, crashes, or log corruption despite increased disk I/O pressure.

- No slowdown
- No blocking
- No memory growth

Result:



Observed Behavior:

The agent maintained stable operation, continued logging process creation events correctly, and exhibited no observable impact from concurrent disk stress.

Rationale:

Ransomware encryption phases generate intense disk write activity; defensive agents must remain operational during such conditions to capture late-stage attacker behaviors and preserve forensic evidence.

Quantitative Observation:

During burst testing, the agent reliably handled **dozens of process creation events per second** without observable delays or missed entries, indicating suitability for real-world operational workloads.

Out of Scope for This Phase:

The following aspects are intentionally excluded from this phase of testing and development:

- Kernel-level process injection and rootkit activity
- Command-level tracing within interactive shells (e.g., `cmd.exe`, PowerShell)
- Memory-only or fileless execution techniques
- Correlation with ETW, Sysmon, or EDR telemetry

These areas are identified as **future expansion points** rather than limitations.

Summary

The stress testing demonstrates that the process monitoring module is operationally stable, resilient under adverse conditions, and suitable as a foundational telemetry layer for higher-level behavioral detection of human-operated ransomware.
