FIELD NOTE / MALWARE ANALYSIS

Exploring Windows Sandbox for Malware Analysis

Petruknisme · 12 Aug, 2025 · 6 min to read

Malware analysis traditionally requires a **safe, isolated environment** where malicious files can be executed without risking the host system. Analysts often rely on **virtual machines** (VMware, VirtualBox, Hyper-V) or even **dedicated physical labs** to observe and dissect malicious behavior.

Spinning up a full VM for a single malware sample feels like using a cargo ship to cross a river. For quick behavioral checks, you need something lighter, faster, and disposable.

Windows Sandbox is a **secure, temporary instance of Windows** that runs in isolation from your host. It's powered by **Hyper-V-based virtualization** but is designed for speed and simplicity.

Core Security Features

- **Hardware-based isolation**: Utilizes Hyper-V hypervisor for strong isolation
- **Ephemeral environment**: All changes are discarded when the sandbox is closed
- **Kernel-level security**: Runs in a separate kernel space from the host OS
- **Memory isolation**: Complete separation of memory space from the host system
- **File system isolation**: Changes to files remain contained within the sandbox

Advantages for Malware Analysis

- **Zero setup time**: No need to maintain and restore VM snapshots
- **Resource efficiency**: Lighter than traditional VMs, faster startup times
- **Native Windows experience**: Full Windows 10/11 environment for accurate malware behavior observation
- **Built-in availability**: No additional software installation required on Windows Pro/Enterprise
- **"Quick reset"**: Faster than restoring VM snapshots.

> *"Ideal Use Case*: Quick behavioral triage of suspicious files without investing the time to set up and tear down a full lab VM.*"*

Requirements

Before enabling Windows Sandbox, ensure your system meets these requirements:

- **OS**: Windows 10 Pro/Enterprise (1903 or newer) or Windows 11 Pro/Enterprise
- **Hardware**:
- 4 GB RAM (8 GB+ recommended)
- 1 GB free disk space
- 2 CPU cores (4 recommended)
- Virtualization enabled in BIOS/UEFI

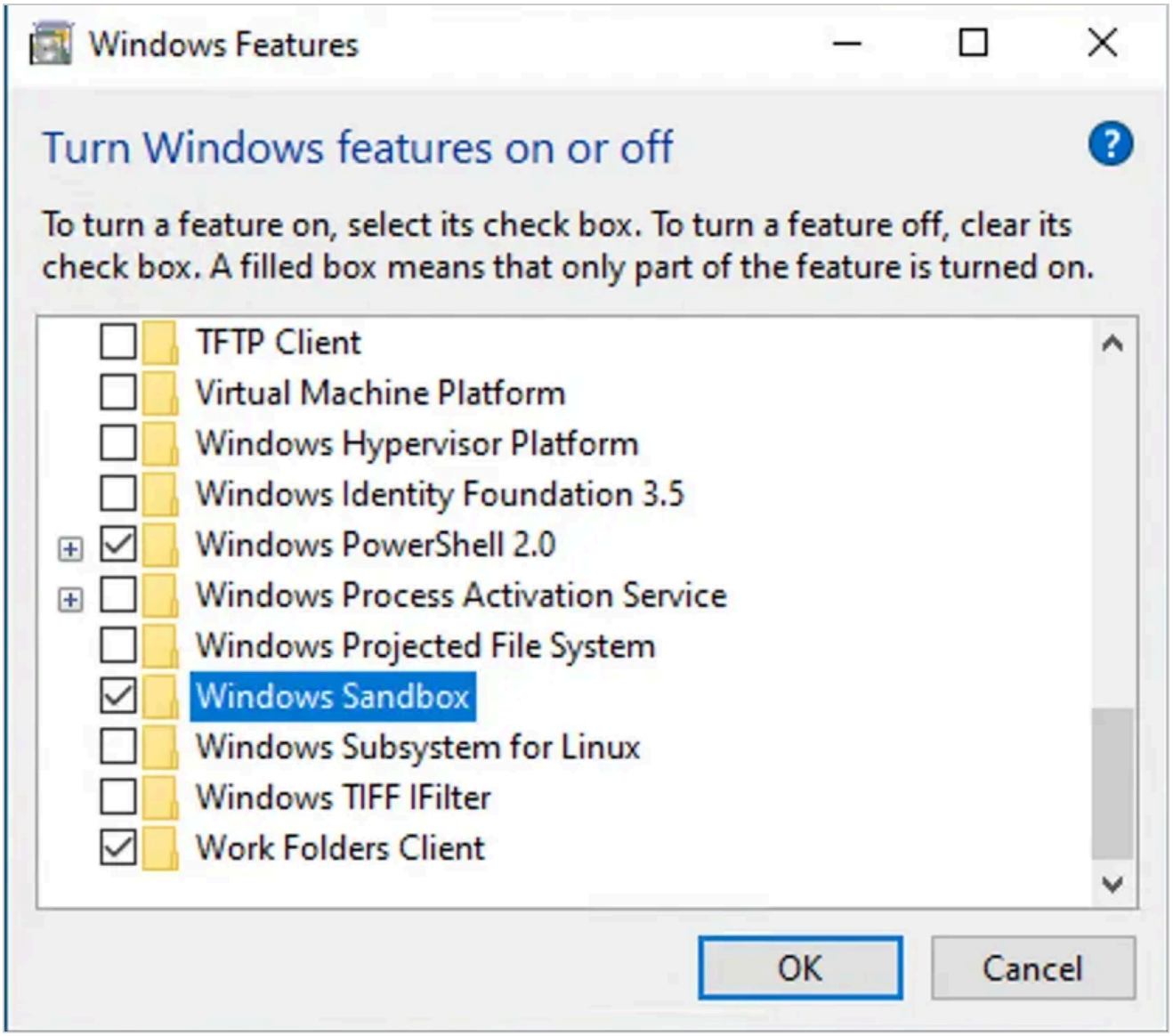
Host Features:

- Hyper-V
- Windows Hypervisor Platform

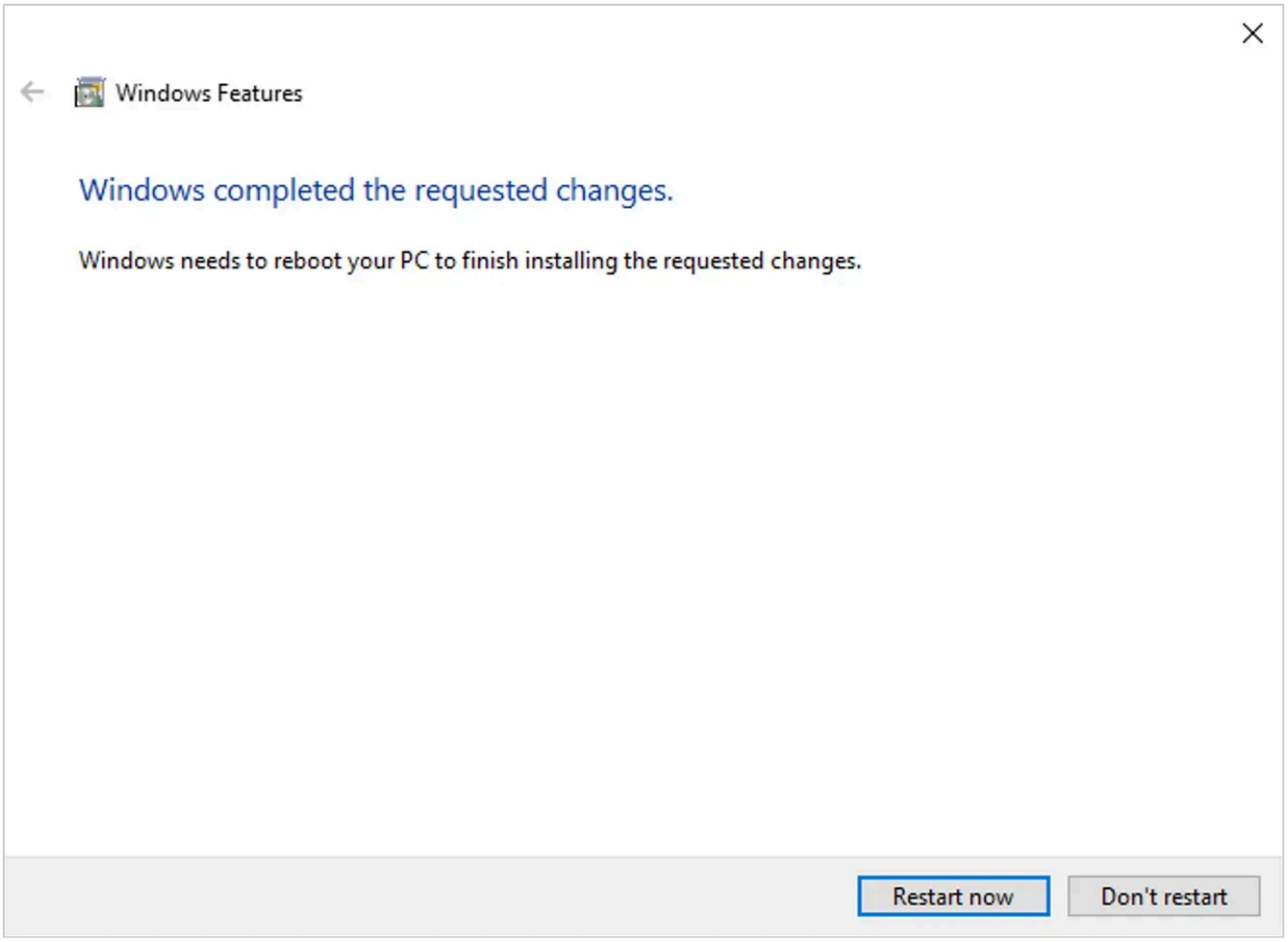
Enabling Windows Sandbox

Using GUI:

1. Open **Control Panel → Programs → Turn Windows features on or off**.
2. Check **Windows Sandbox**.
3. Click **OK** and restart your system.



Activating Windows Sandbox



Restart windows after activating windows sandbox

Using PowerShell:

```
# Enable through PowerShell (run as administrator)
Enable-WindowsOptionalFeature -Online -FeatureName "Containers-DisposableC

# Verify installation
Get-WindowsOptionalFeature -Online -FeatureName "Containers-DisposableClie
```

Manual Configuration with.wsb Files

By default, Windows Sandbox launches as a clean environment.
However, you can customize it using a **.wsb configuration file**.

Example: **sandbox-malware-analysis.wsb**

sandbox-malware-analysis.wsb

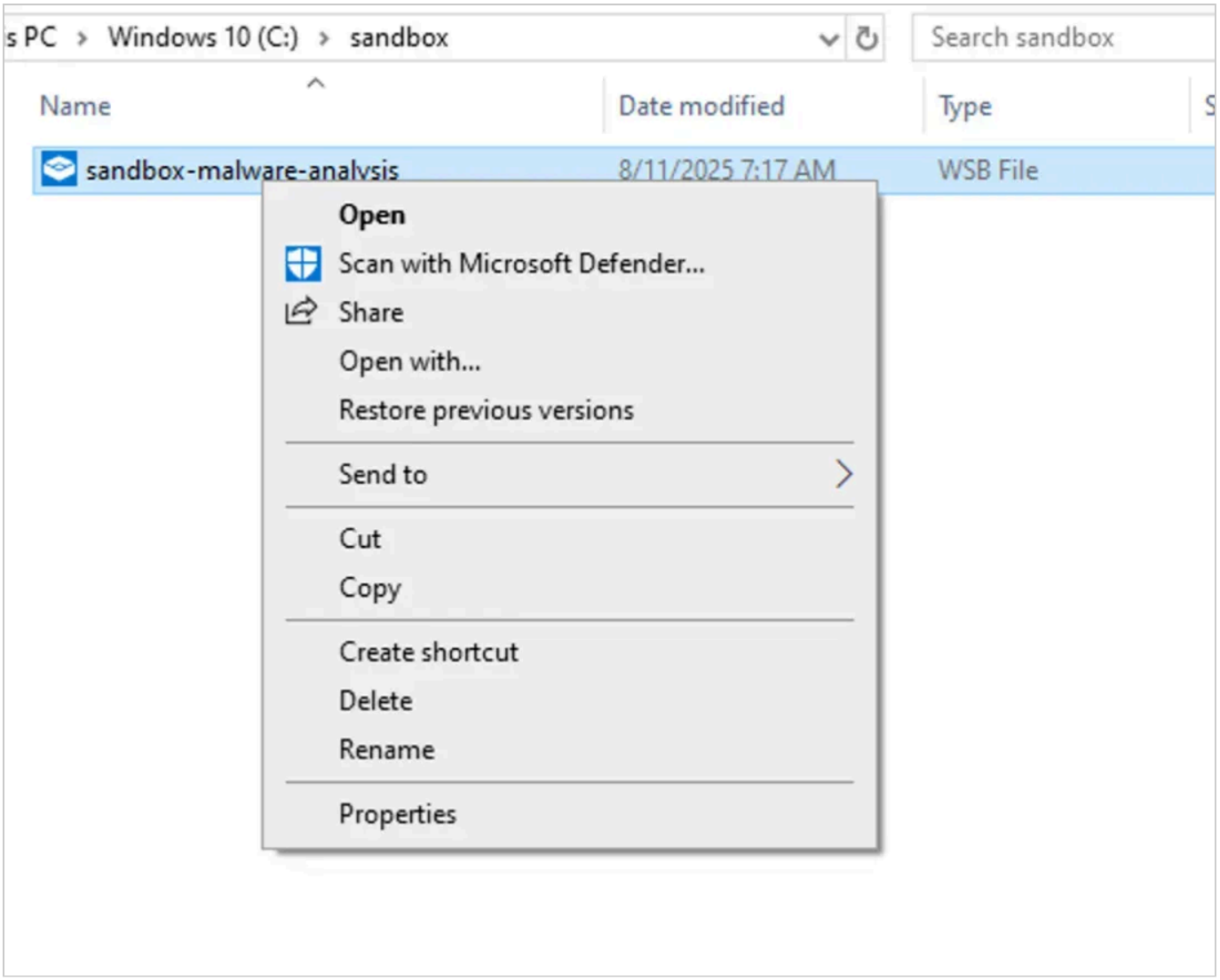
```
<Configuration>
    <Networking>Disable</Networking>
    <MappedFolders>
        <MappedFolder>
            <HostFolder>C:\MalwareSamples</HostFolder>
            <SandboxFolder>C:\Samples</SandboxFolder>
            <ReadOnly>true</ReadOnly>
        </MappedFolder>
    </MappedFolders>
    <LogonCommand>
        <Command>C:\Samples\analysis_tools.bat</Command>
    </LogonCommand>
</Configuration>
```

The explanation of the above configuration is:

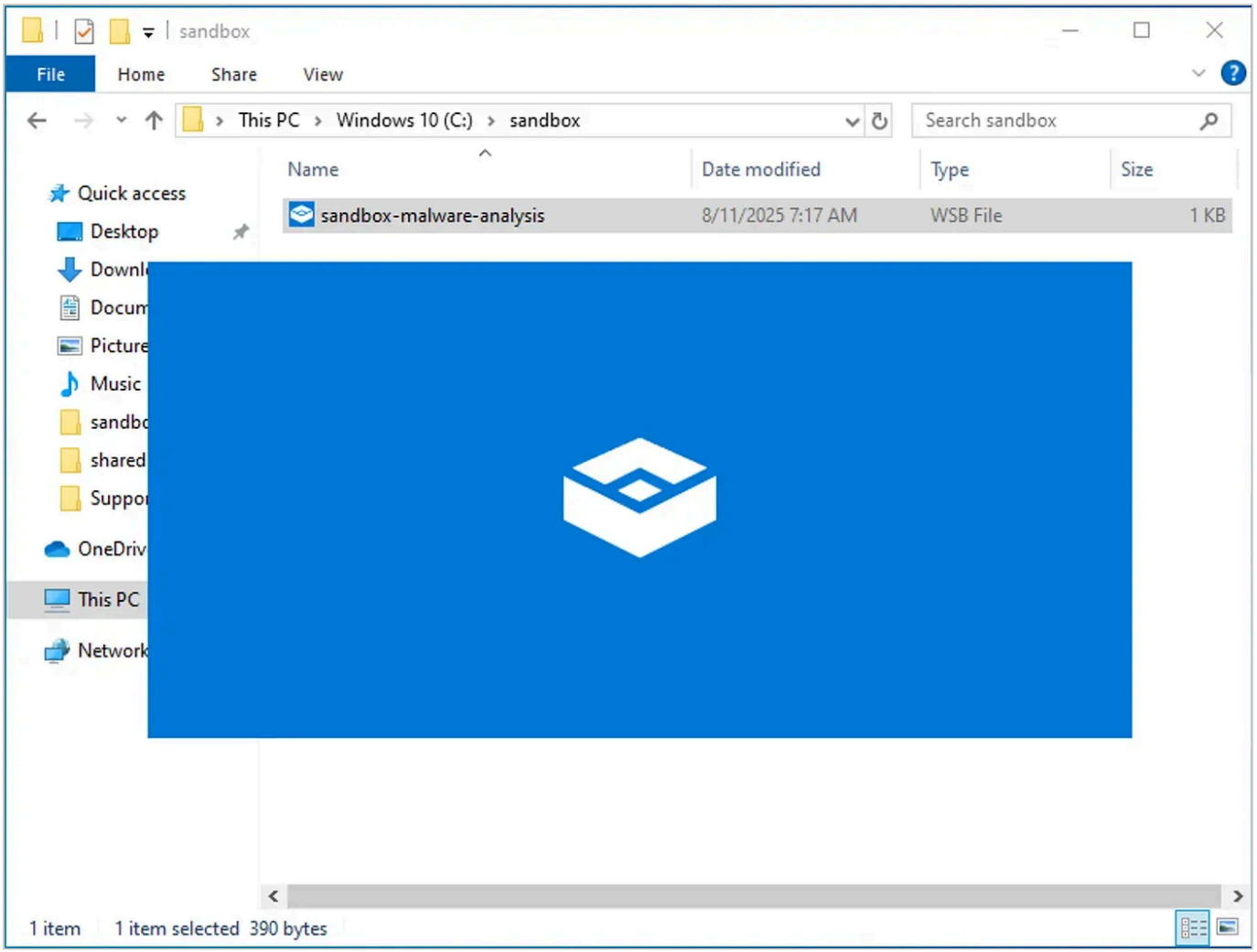
- **Networking Disabled**
  Prevents internet/LAN access, isolating the sandbox.
- **Mapped Folders (Read-Only)**
  Maps `C:\MalwareSamples` on the host to `C:\Samples` in the sandbox, allowing read-only access to prevent host modification.
- **Auto-Run Command**
  Executes `analysis_tools.bat` from `C:\Samples` on sandbox startup, automating tool launch.

Save with `.wsb` extension and double-click to launch.

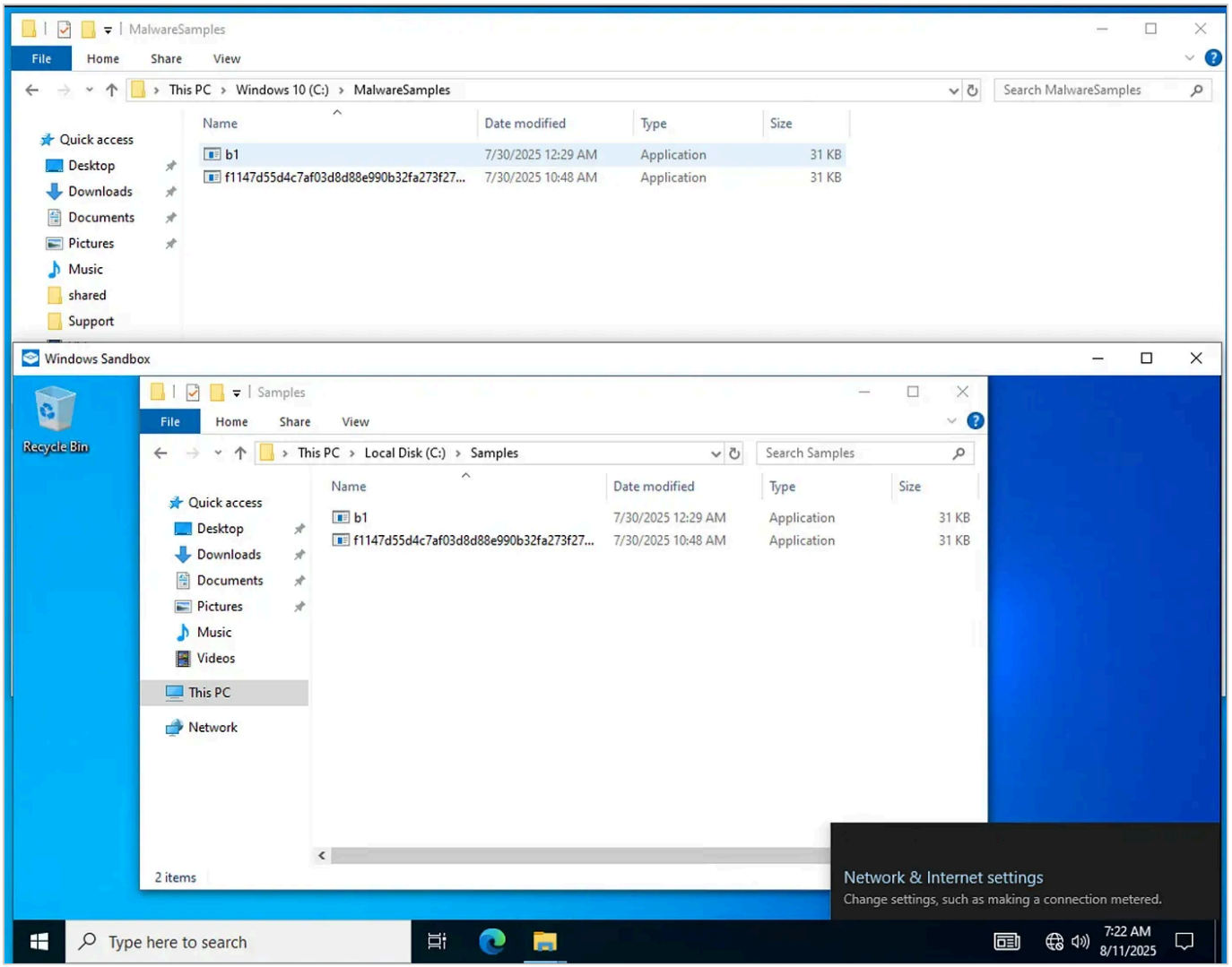
> *"Note: For more info about windows sandbox configuration, you can read here https://learn.microsoft.com/en-us/windows/security/application-security/application-isolation/windows-sandbox/windows-sandbox-configure-using-wsb-file (https://learn.microsoft.com/en-us/windows/security/application-security/application-isolation/windows-sandbox/windows-sandbox-configure-using-wsb-file)"*



Windows sandbox properties

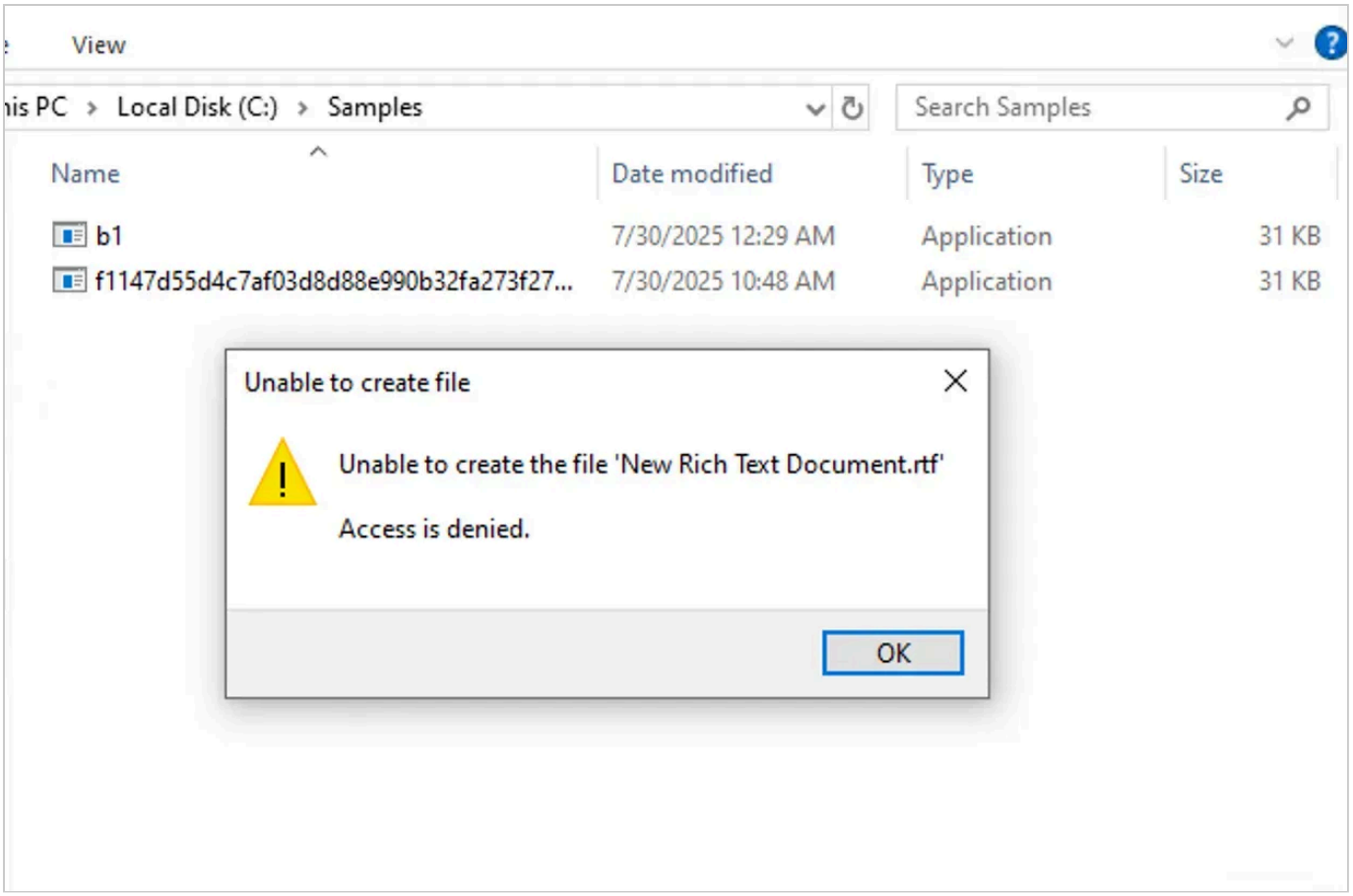


Windows sandbox splash screen



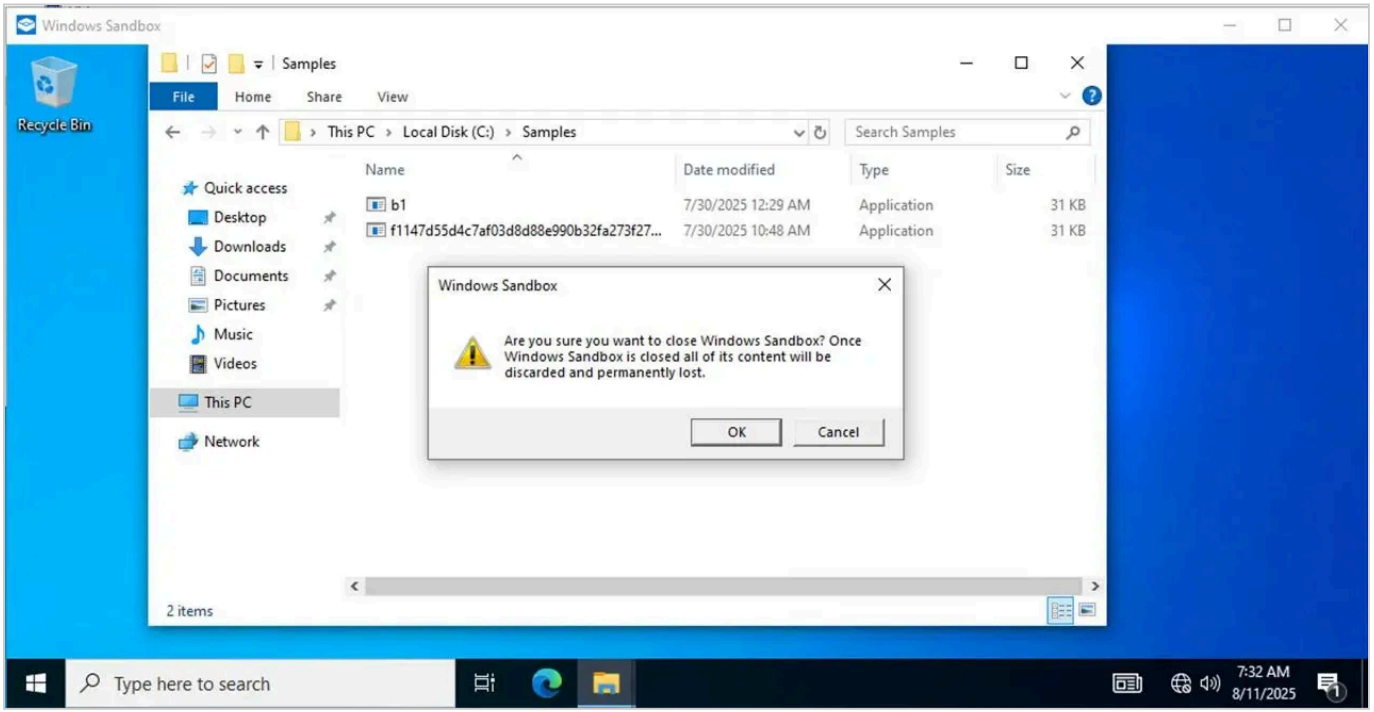
Running windows sandbox

As shown above, we successfully created a Windows Sandbox with an isolated network and mapped `C:\MalwareSamples` on the host to `C:\Samples` inside the sandbox.
Any attempt to create or modify files in the mapped folder from within the sandbox will fail, as it is mounted in read-only mode.



Access Denied when creating file

When closing Windows Sandbox, a confirmation popup will appear, informing you that all data and changes inside the sandbox will be permanently deleted.

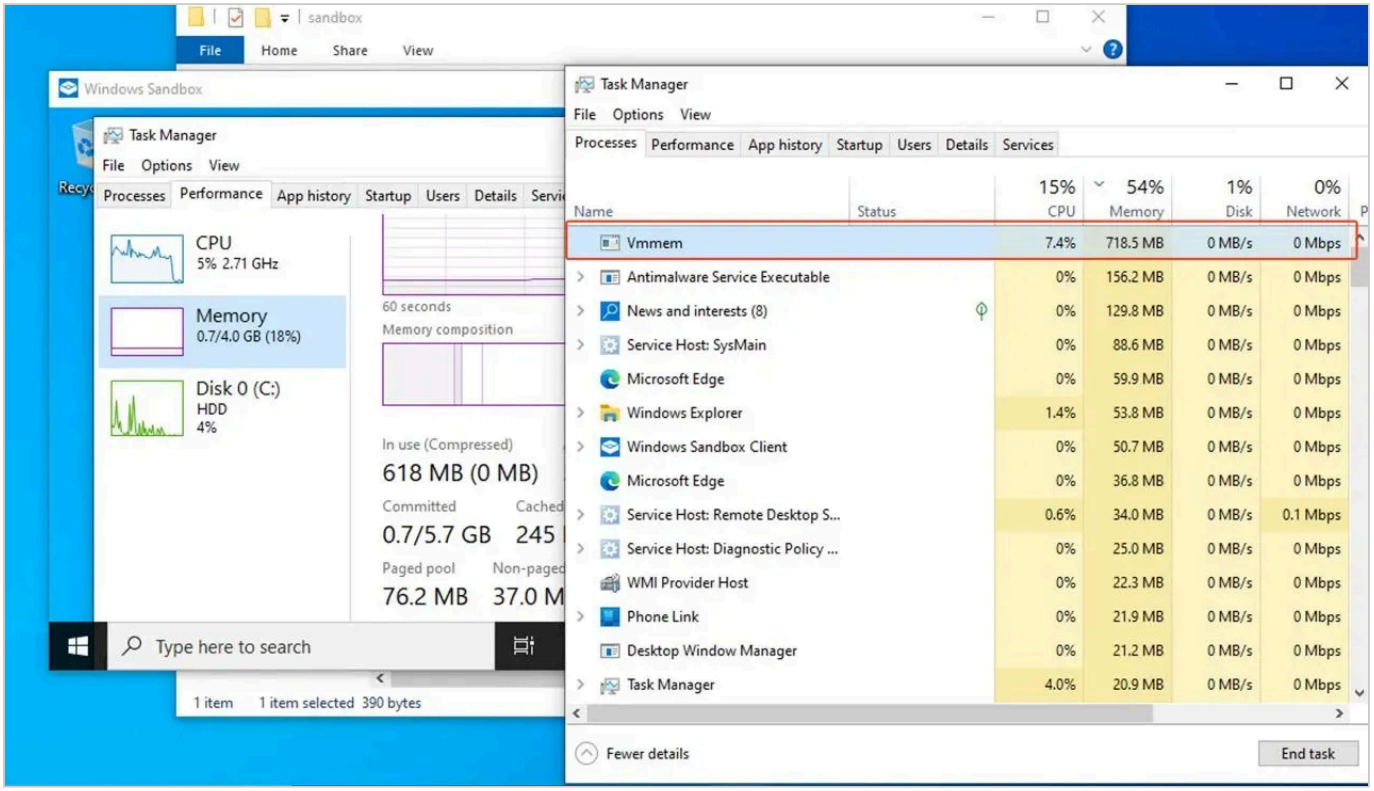


Closing windows sandbox

Windows Sandbox Resource Usage

One of the advantages of Windows Sandbox is its efficiency compared to traditional virtual machines. When freshly launched, the `Vmmem` process (responsible for managing the virtualized environment) typically consumes **around 600–720 MB of RAM** to run a clean Windows session.

In the example below, Task Manager shows `Vmmem` using **718.5 MB** of memory, with total system memory usage for the sandbox instance around **0.7 GB**. This low overhead allows for quick startup and smooth operation, even on systems with limited resources.



GUI Context Menu

For users who prefer a GUI, no-script approach, **Run-in-Sandbox(https://github.com/damienvanrobaeys/Run-in-Sandbox** (https://github.com/damienvanrobaeys/Run-in-Sandbox)) offers a convenient PowerShell-based tool that integrates directly into the Windows context menu.

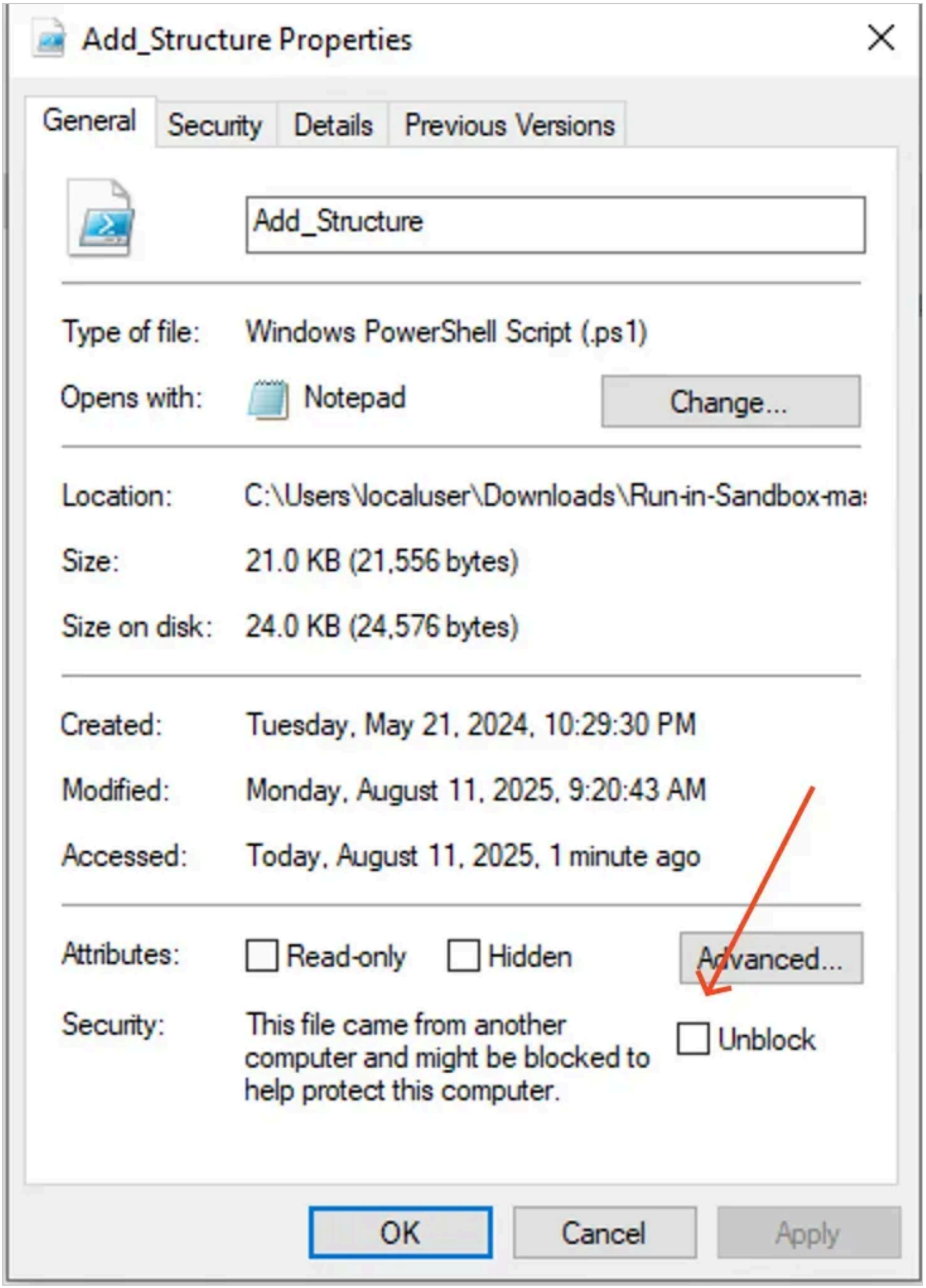
Once installed, you can right-click on various file types including **PS1**, **VBS**, **CMD**, **EXE**, **MSI**, **MSIX**, **Intunewin**, **PDF/HTML/URL**, **ZIP**, **ISO**, and more, and choose to run or extract them directly inside Windows Sandbox. It also supports sharing folders and launching multiple apps in a single session.

Installation Steps for Run-in-Sandbox

- **Download** the full ZIP of the **Run-in-Sandbox** project.

> *"⚠ Do **not** download only* ***`Add_Structure.ps1`****."*

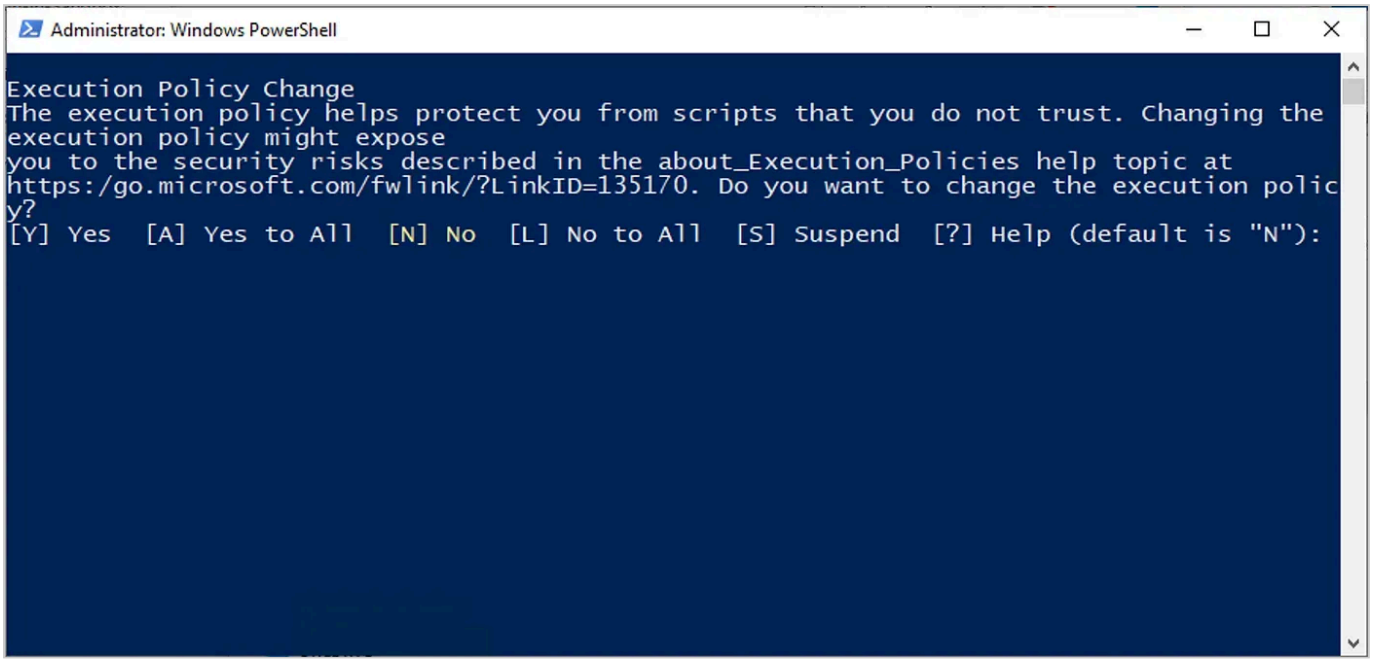
- **Extract** the ZIP.
  You should now have a folder named **`Run-in-Sandbox-master`** containing:
  - **`Add_Structure.ps1`**
  - A **`Sources`** folder with a subfolder **`Run_in_Sandbox`** containing ~58 files.
- **Check for blocked files** after download:
  Right-click **`Add_Structure.ps1`** → **Properties** → if you see an **Unblock** checkbox, check it and apply.



Unblock the script

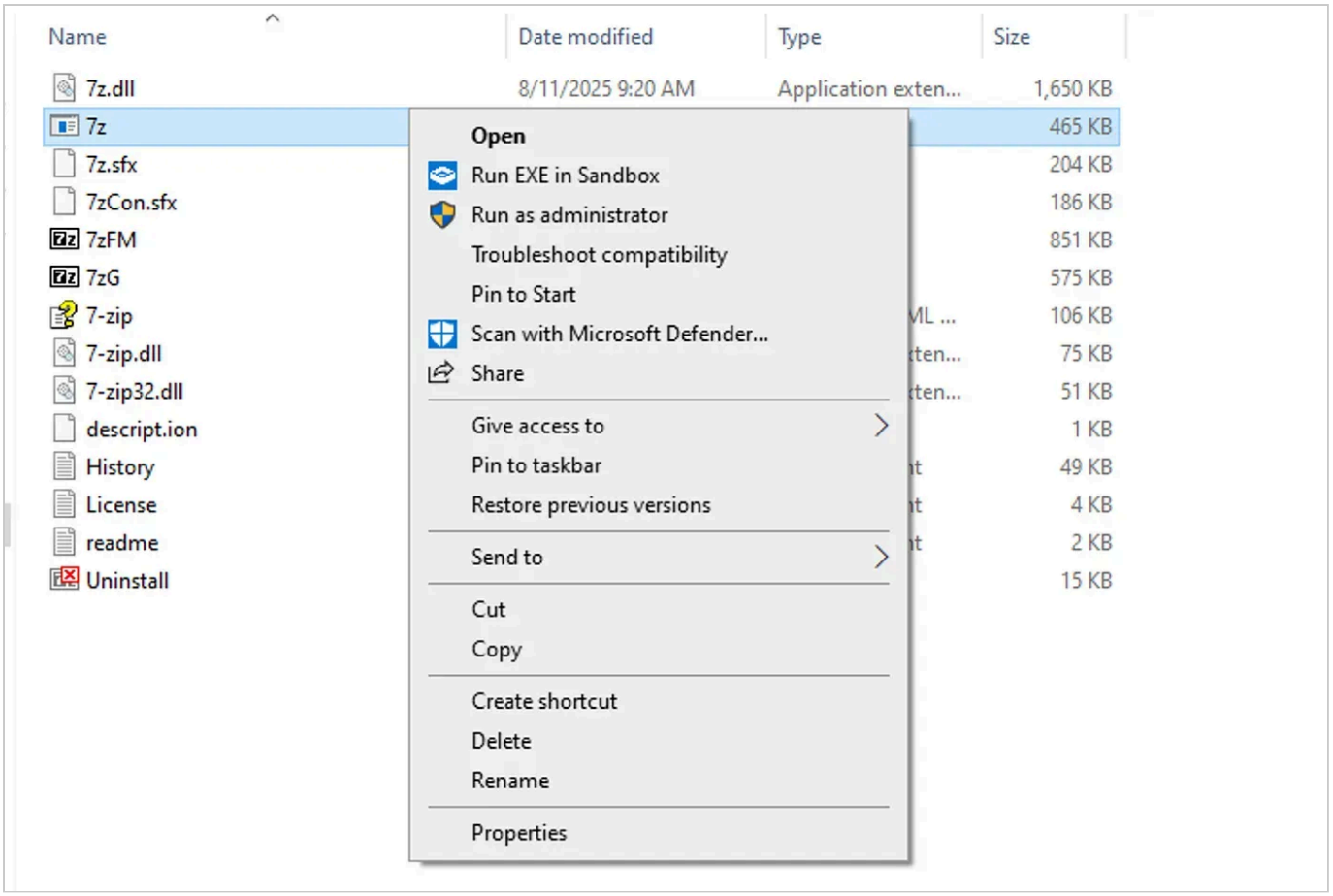
- **Install** the context menu integration:
  Run `Add_Structure.ps1` **as Administrator**.

If PowerShell's execution policy prompts you, type **Y** (Yes) to temporarily allow the script to run. The script will then execute normally.

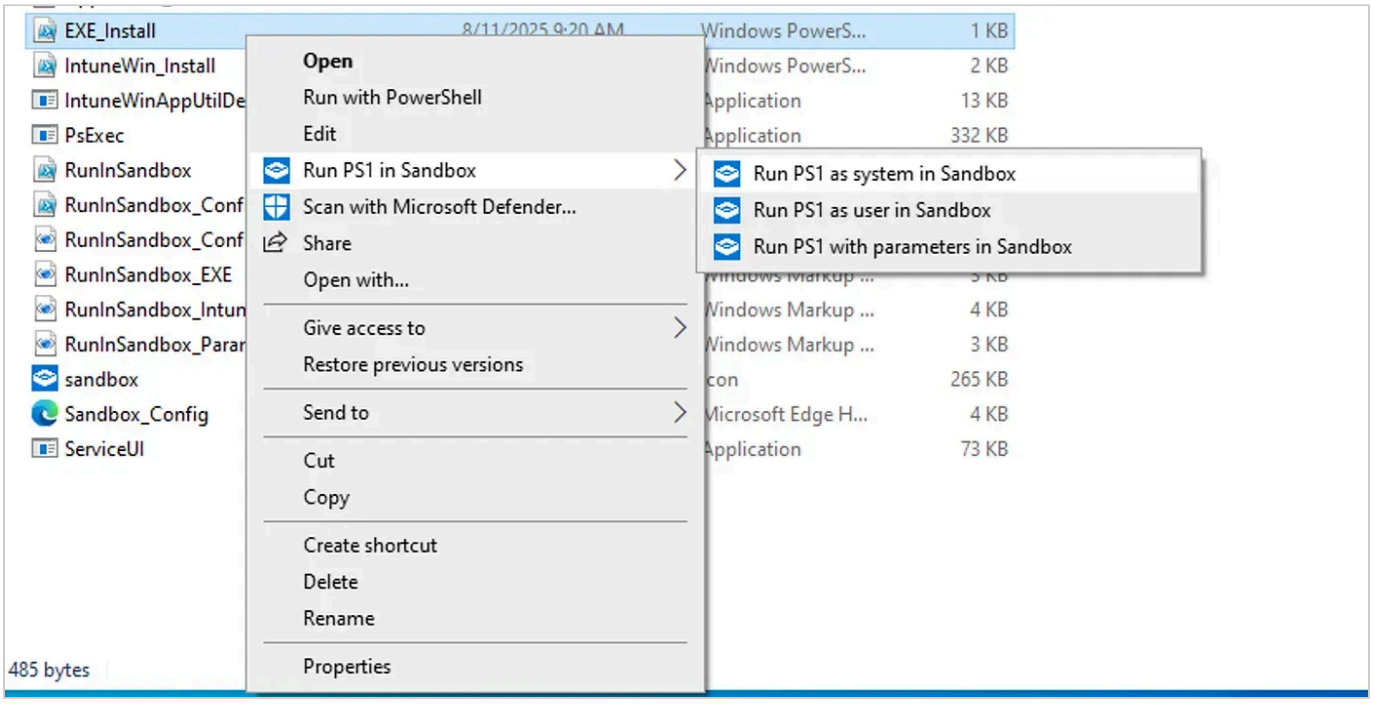


Execution Policy Popup

If the installation is successful, a **"Run <file> in Sandbox"** option will appear in the context menu when you right-click any supported file.



When launched from the context menu on a **PowerShell (.ps1)** file, *Run-in-Sandbox* supports executing the script under either the **System** or **User** context, and also allows passing **parameters** to the script.



Right-clicking powershell script in run windows sandbox context menu

Example Malware Analysis Workflow

Use Case: Testing a Suspicious Executable

1. **Obtain the sample** from a trusted malware research source (e.g., MalwareBazaar) in a password-protected ZIP file.
2. **Right-click** the sample → **Run `**<file>**` in Sandbox** *(or launch via a prepared `*.wsb*` configuration)*.
3. Run the executable and observe its behavior using monitoring tools:
   - **Process Explorer**: watch for new or unexpected processes.
   - **TCPView** or **Wireshark:** monitor any network connections the sample tries to make.
   - **Process Monitor (Procmon)**: track file operations, registry changes, and system modifications.
4. Close Windows Sandbox

Conclusion

Windows Sandbox provides a fast, disposable, and isolated environment that makes it ideal for safely running suspicious files and observing their behavior without risking your host system. By combining manual `.wsb` configurations with the **Run-in-Sandbox** context menu tool, you can launch and test malware samples in just a few clicks, while still maintaining control over permissions, folder mounts, and network access.

While it cannot fully replace a dedicated virtual machine for deep reverse engineering, Windows Sandbox is a powerful first-line tool for initial malware triage, behavioral observation, and quick security testing.
With the right setup and monitoring tools like Process Explorer, TCPView, and Procmon, you can gather valuable insights in minutes and with a single click, reset the environment back to a clean state.

> *"⚠ **Important Warning:** No sandbox or virtualized environment is completely safe. Advanced malware can detect virtualized environments, escape containment, or exploit vulnerabilities in the host operating system (such as past sandbox escape CVEs).*
>
> *Even with every safeguard, there is still a small chance something could go wrong, treat the environment as potentially unsafe.*
>
> *For safer analysis, use an isolated or offline machine, avoid mapping unnecessary host folders, and never run live malware on a system containing sensitive data, credentials, or production workloads."*

References:

- https://learn.microsoft.com/en-us/windows/security/application-security/application-isolation/windows-sandbox/ (https://learn.microsoft.com/en-us/windows/security/application-security/application-

isolation/windows-sandbox/)



(https://buymeacoffee.com/petruknisme)

Like the article and want to support me? you can support me **here** (https://buymeacoffee.com/petruknisme)

Originally published on **Medium** (https://medium.com/@petruknisme/exploring-windows-sandbox-for-malware-analysis-c0f50eb0fa6c).

Malware Analysis # Windows # Tutorial