GitHub - lzcapp/NotMe-BSOD: Codes that could trigger BSOD (Blue Screen of Death) on Windows.

By lzcapp

Archived: 2026-04-05 15:34:41 UTC

⚠️ SOLELY FOR SECURITY RESEARCH PURPOSES. DO NOT USE THIS REPO FOR ILLEGAL PURPOSES. ⚠️

The name "**NotMe**" is inspired by the Microsoft tool "**NotMyFault**", also a tool that can be used to cause BSOD (Blue Screen of Death) on Windows system.

BUILD NO STATUS 

Project Structure

NotMe_CPP

`C++` project that uses `CMake` and `CLion` to build.

- Toolchain: MinGW32 from MSYS2

NotMe_CSharp

`C#` and `.Net Framework 4`. Including a GUI app.

NtRaiseHardError & ZwRaiseHardError

> Ref: http://undocumented.ntinternals.net/index.html?page=UserMode%2FUndocumented%20Functions%2FError%2FNtSetDefaultHardErrorPort.html

Both functions are undocumented `Windows APIs (NTAPI)` included in `ntdll.dll`, which can cause `BlueScreen (BSOD, Blue Screen of Death)` with certain parameters.

```
NtRaiseHardError(
  IN  NTSTATUS                 ErrorStatus,
  IN  ULONG                    NumberOfParameters,
  IN  PUNICODE_STRING          UnicodeStringParameterMask OPTIONAL,
  IN  PVOID                    *Parameters,
  IN  HARDERROR_RESPONSE_OPTION ResponseOption,
  OUT PHARDERROR_RESPONSE      Response
);
```

```
ZwRaiseHardError(
  IN  NTSTATUS                 ErrorStatus,
  IN  ULONG                    NumberOfParameters,
  IN  PUNICODE_STRING          UnicodeStringParameterMask OPTIONAL,
  IN  PVOID                    *Parameters,
  IN  HARDERROR_RESPONSE_OPTION ResponseOption,
  OUT PHARDERROR_RESPONSE      Response
);
```

- `ErrorStatus` Error code.
- `NumberOfParameters` Number of optional parameters in Parameters array.
- `UnicodeStringParameterMask` Optional string parameter (can be only one per error code).
- `*Parameters` Array of **DWORD** parameters for use in error message string.
- `ResponseOption` See `HARDERROR_RESPONSE_OPTION` for possible values description.
- `Response` Pointer to `HARDERROR_RESPONSE` enumeration.

Parameters

PUNICODE_STRING

```
typedef struct _UNICODE_STRING {
 USHORT Length;
 USHORT MaximumLength;
 PWSTR Buffer;
} UNICODE_STRING, *PUNICODE_STRING;
```

HARDERROR_RESPONSE_OPTION

```
typedef enum _HARDERROR_RESPONSE_OPTION {
 OptionAbortRetryIgnore,
 OptionOk,
 OptionOkCancel,
 OptionRetryCancel,
 OptionYesNo,
 OptionYesNoCancel,
 OptionShutdownSystem
} HARDERROR_RESPONSE_OPTION, *PHARDERROR_RESPONSE_OPTION;
```

PHARDERROR_RESPONSE

```
typedef enum _HARDERROR_RESPONSE {
 ResponseReturnToCaller,
 ResponseNotHandled,
```

```
 ResponseAbort,
 ResponseCancel,
 ResponseIgnore,
 ResponseNo,
 ResponseOk,
 ResponseRetry,
 ResponseYes
} HARDERROR_RESPONSE, *PHARDERROR_RESPONSE;
```

SetProcessIsCritical

> Ref: http://www.netcore2k.net/projects/freeram

`RtlSetProcessIsCritical` is yet another undocumented function hidden in the Windows kernel. It is one of the few which do not have a kernel32 equivalent.

`RtlSetProcessIsCritical` sets a process to a system critical status. This means that the process is now "critical" to the running of Windows, which also means that on termination of your process, Windows itself terminates as well. When a system critical process ends/terminates, the stop code is CRITICAL_PROCESS_DIED (0xEF) for process exiting, and CRITICAL_OBJECT_TERMINATION (0xF4) if the process was abnormally terminated.

```
/**
 * @param NewValue the new critical setting: 1 for a critical process, 0 for a normal process
 * @param OldValue if not null, will receive the old setting for the process
 * @param bNeedScb specifics whether system critical breaks will be required (and already enabled) for
 */
NTSTATUS RtlSetProcessIsCritical (IN BOOLEAN bNew, OUT BOOLEAN *pbOld, IN BOOLEAN bNeedScb);
```

This means that calling `RtlSetProcessIsCritical(TRUE, NULL, FALSE)` would make a process critical, while another call to `RtlSetProcessIsCritical(FALSE, NULL, FALSE)` would return the process to normal. When critical status is set, termination or ending of the process in any way will usually cause either a BSOD (if BSOD-ing is enabled), or will cause the system to reboot itself.

CloseWindowStation

```
HWINSTA CreateWindowStationA(
  [in, optional] LPCSTR               lpwinsta,
                 DWORD                dwFlags,
  [in]           ACCESS_MASK          dwDesiredAccess,
  [in, optional] LPSECURITY_ATTRIBUTES lpsa
);
```

```
BOOL CloseWindowStation(
  [in] HWINSTA hWinSta
```

```
);
```

NTSD_Winlogon

```
cmd /c start /min ntsd -c q -pn winlogon.exe 1>nul 2>nul
```

PowerShell_Wininit

TaskKill_Wininit

```
taskkill /f /im wininit.exe
```

Compatibility

NtRaiseHardError & ZwRaiseHardError

Works on all Windows systems with `Windows NT kernel` (`ntdll.dll`). Does not triger the `UAC (User Account Control)` prompt.

SetProcessIsCritical

Require `ntdll.dll` versions 5.1 (`Windows XP`) and higher.

Needs `Administrator privilege` / trigers `UAC (User Account Control)` on `Winodws 10` and `Windows 11` .

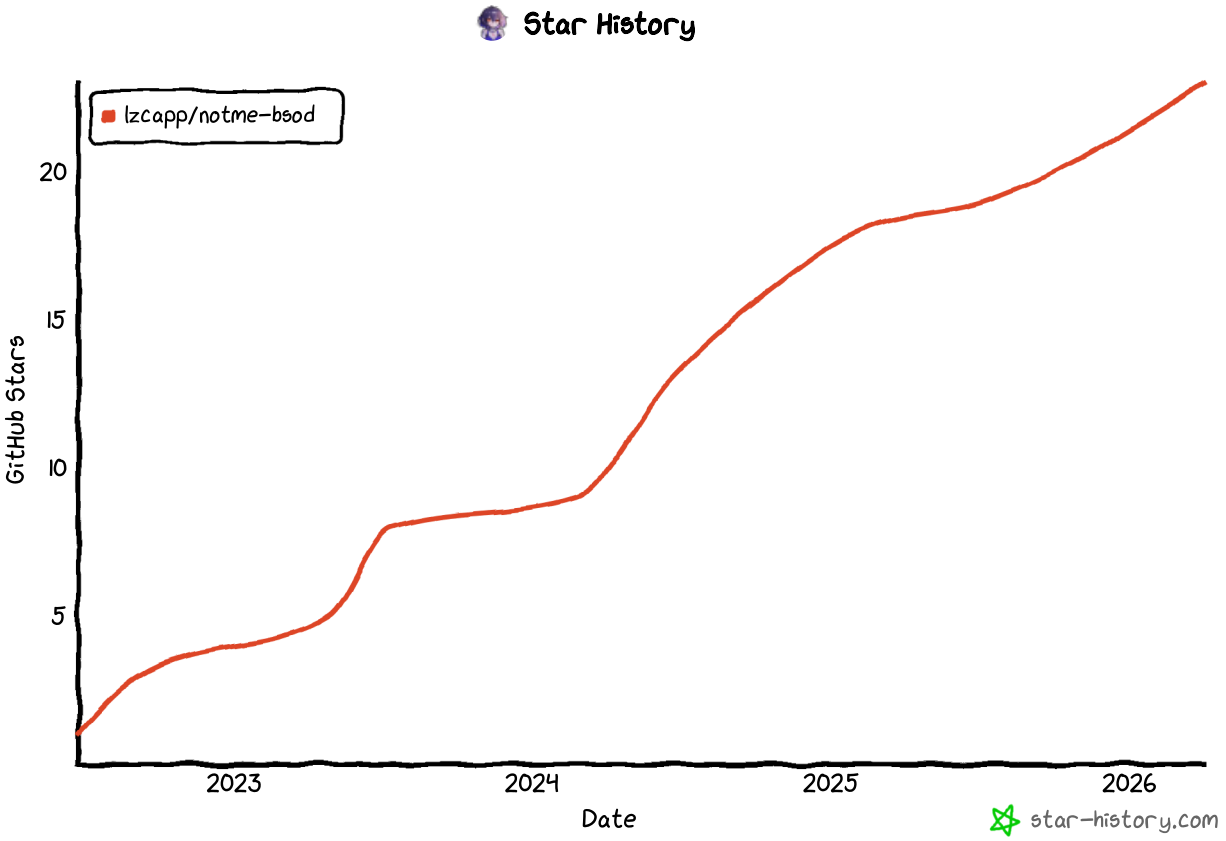
Compatibility Table

	ReactOS 0.4.14	Windows 2000	Windows XP	Windows Vista	Windows 7	Windows 10	Windows 11
NtRaiseHardError / ZwRaiseHardError	❌	✔️	✔️	✔️	✔️	✔️	✔️
SetProcessIsCritical	✔️	❌	✔️	⭕	⭕	⭕	⭕
CloseWindowStation	❌	❌	✔️	❌	❌	❌	❌
NTSD_Winlogon	❌	✔️	✔️	❌	❌	❌	❌
PowerShell_Wininit	❌	❌	❌	❌	⭕	❌	❌
TaskKill_Wininit	❌	❌	❌	⭕	⭕	❌	❌

> ✔️: Works Well
>
> ⭕: Requires `Administrator Privilege` / `UAC (User Account Control)`

> ❌: Not Working

Star History



Source: https://github.com/lzcapp/NotMe-BSOD