# GitHub - hfiref0x/UACME: Defeating Windows User Account Control

By hfiref0x

Archived: 2026-04-05 17:23:58 UTC


Defeating Windows User Account Control by abusing built-in Windows AutoElevate backdoor. This project demonstrates various UAC bypass techniques and serves as an educational resource for understanding Windows security mechanisms.

> ⚠️ **Warning**: This tool demonstrates security vulnerabilities that could be exploited maliciously. Use responsibly and only in controlled environments.

## System Requirements

- **Operating Systems**: Windows 7/8/8.1/10/11 (x86-32/x64, client, some methods however works on server version too)
- **User Account**: Administrator account with UAC set on default settings

## Usage

Run the executable from command line using the following syntax:

```
akagi32.exe [Method_Number] [Optional_Command]
```

or

```
akagi64.exe [Method_Number] [Optional_Command]
```

### Parameters:

- **Method_Number**: Number corresponding to the UAC bypass method (see Methods List below)
- **Optional_Command**: Full path to an executable file to run with elevated privileges
    - If omitted, the program will launch an elevated command prompt (%systemroot%\system32\cmd.exe)

### Examples:

```
 akagi32.exe 23
akagi64.exe 61
akagi32.exe 23 c:\windows\system32\calc.exe
akagi64.exe 61 c:\windows\system32\charmap.exe
```

> **Note**: Since version 3.5.0, all previously "fixed" methods are considered obsolete and have been removed. If you need them, use v3.2.x branch.

▶ Keys (click to expand/collapse)

**Important Notes:**

- Method 30, 63 and later are implemented only in x64 version
- Method 30 requires x64 because it exploits WOW64 subsystem feature
- Method 55 is included primarily for educational purposes and may not be reliable
- Method 78 requires that the current user account password is not blank

## Warning

⚠️ **Important Security and Usage Information**:

- This tool demonstrates **only publicly known UAC bypass methods** used by malware. It reimplements some techniques in different ways to improve upon original concepts.
- **Not intended for antivirus testing** and not guaranteed to work in environments with aggressive security software. Use with active antivirus at your own risk.
- Many antivirus solutions may flag this tool as a "HackTool" - this is expected behavior due to its capabilities.
- **Clean up after usage**: If running on a production system, ensure you remove all program artifacts afterward. See source code for details about files dropped to system folders.
- Most methods were developed primarily for x64 systems. While many can work on x86-32 with minor adjustments, 32-bit support is not a focus of this project.
- For an official Microsoft explanation on why UAC bypasses still exist, see: Microsoft's stance on UAC

## Windows 10 support and testing policy

- UACMe is tested only with LSTB/LTSC variants (1607/1809) and the current RTM-1 versions
- For example: if the current version is 2004, it will be tested on 2004 (19041) and the previous 1909 (18363)
- Insider builds are not supported as methods may be fixed in preview releases

## Protection Measures

The most effective protection against UAC bypass techniques is using an account without administrative privileges.

## Build instructions

UACMe is written in C and requires Microsoft Visual Studio 2019 or later to build from source.

### Prerequisites

- **IDE**: Microsoft Visual Studio 2019 or 2022
- **SDK Requirements**:
  - Windows 8.1 or Windows 10 SDK (tested with 19041 version)
  - NET Framework SDK (tested with 4.8 version)

### Build Steps

1. **Configure Platform ToolSet** (Project->Properties->General):
   - For Visual Studio 2019: Select v142
   - For Visual Studio 2022: Select v143
2. **Set Target Platform Version** (Project->Properties->General):
   - For v140: Select 8.1 (Windows 8.1 SDK must be installed)
   - For v141 and above: Select 10
3. **Build Process**:
   - Compile payload units
   - Compile Naka module
   - Encrypt all payload units using Naka module
   - Generate secret blobs for these units using Naka module
   - Move compiled units and secret blobs to the Akagi\Bin directory
   - Rebuild Akagi

> **Note**: Compiled binaries are not provided and will never be provided. This serves as a barrier against malicious usage and helps maintain the educational purpose of this project.

## Legal Disclaimer

- This tool is provided for **educational and research purposes only**
- We do not take any responsibility for this tool being used in malicious activities
- We have no affiliation with any "security company" using this code for commercial activities
- This GitHub repository (hfiref0x/UACME) is the only genuine source for UACMe code

## Support

If you find this project interesting, you can buy me a coffee

BTC (Bitcoin): bc1qzkvtpa0053cagf35dqmpvv9k8hyrwl7krwdz84q39mcpy68y6tmqsju0g4

## References


- Windows 7 UAC whitelist, http://www.pretentiousname.com/misc/win7_uac_whitelist2.html
- Malicious Application Compatibility Shims, https://www.blackhat.com/docs/eu-15/materials/eu-15-Pierce-Defending-Against-Malicious-Application-Compatibility-Shims-wp.pdf
- Junfeng Zhang from WinSxS dev team blog, https://blogs.msdn.microsoft.com/junfeng/
- Beyond good ol' Run key, series of articles, http://www.hexacorn.com/blog
- KernelMode.Info UACMe thread, https://www.kernelmode.info/forum/viewtopicf985.html?f=11&t=3643
- Command Injection/Elevation - Environment Variables Revisited, https://breakingmalware.com/vulnerabilities/command-injection-and-elevation-environment-variables-revisited
- "Fileless" UAC Bypass Using eventvwr.exe and Registry Hijacking, https://enigma0x3.net/2016/08/15/fileless-uac-bypass-using-eventvwr-exe-and-registry-hijacking/
- Bypassing UAC on Windows 10 using Disk Cleanup, https://enigma0x3.net/2016/07/22/bypassing-uac-on-windows-10-using-disk-cleanup/
- Using IARPUninstallStringLauncher COM interface to bypass UAC, http://www.freebuf.com/articles/system/116611.html
- Bypassing UAC using App Paths, https://enigma0x3.net/2017/03/14/bypassing-uac-using-app-paths/
- "Fileless" UAC Bypass using sdclt.exe, https://enigma0x3.net/2017/03/17/fileless-uac-bypass-using-sdclt-exe/
- UAC Bypass or story about three escalations, https://habrahabr.ru/company/pm/blog/328008/
- Exploiting Environment Variables in Scheduled Tasks for UAC Bypass, https://tyranidslair.blogspot.ru/2017/05/exploiting-environment-variables-in.html
- First entry: Welcome and fileless UAC bypass, https://winscripting.blog/2017/05/12/first-entry-welcome-and-uac-bypass/
- Reading Your Way Around UAC in 3 parts:
    1. https://tyranidslair.blogspot.ru/2017/05/reading-your-way-around-uac-part-1.html
    2. https://tyranidslair.blogspot.ru/2017/05/reading-your-way-around-uac-part-2.html
    3. https://tyranidslair.blogspot.ru/2017/05/reading-your-way-around-uac-part-3.html
- Research on CMSTP.exe, https://msitpros.com/?p=3960
- UAC bypass via elevated .NET applications, https://offsec.provadys.com/UAC-bypass-dotnet.html
- UAC Bypass by Mocking Trusted Directories, https://medium.com/tenable-techblog/uac-bypass-by-mocking-trusted-directories-24a96675f6e
- Yet another sdclt UAC bypass, http://blog.sevagas.com/?Yet-another-sdclt-UAC-bypass
- UAC Bypass via SystemPropertiesAdvanced.exe and DLL Hijacking, https://egre55.github.io/system-properties-uac-bypass/
- Accessing Access Tokens for UIAccess, https://tyranidslair.blogspot.com/2019/02/accessing-access-tokens-for-uiaccess.html
- Fileless UAC Bypass in Windows Store Binary, https://www.activecyber.us/1/post/2019/03/windows-uac-bypass.html
- Calling Local Windows RPC Servers from .NET, https://googleprojectzero.blogspot.com/2019/12/calling-local-windows-rpc-servers-from.html

- Microsoft Windows 10 UAC bypass local privilege escalation exploit, https://packetstormsecurity.com/files/155927/Microsoft-Windows-10-Local-Privilege-Escalation.html
- UACMe 3.5, WD and the ways of mitigation, https://swapcontext.blogspot.com/2020/10/uacme-35-wd-and-ways-of-mitigation.html
- UAC bypasses from COMAutoApprovalList, https://swapcontext.blogspot.com/2020/11/uac-bypasses-from-comautoapprovallist.html
- Utilizing Programmatic Identifiers (ProgIDs) for UAC Bypasses, https://v3ded.github.io/redteam/utilizing-programmatic-identifiers-progids-for-uac-bypasses
- MSDT DLL Hijack UAC bypass, https://blog.sevagas.com/?MSDT-DLL-Hijack-UAC-bypass
- UAC bypass through .Net Deserialization vulnerability in eventvwr.exe, https://twitter.com/orange_8361/status/1518970259868626944
- Advanced Windows Task Scheduler Playbook - Part.2 from COM to UAC bypass and get SYSTEM directly, http://www.zcgonvh.com/post/Advanced_Windows_Task_Scheduler_Playbook-Part.2_from_COM_to_UAC_bypass_and_get_SYSTEM_dirtectly.html
- Bypassing UAC with SSPI Datagram Contexts, https://splintercod3.blogspot.com/p/bypassing-uac-with-sspi-datagram.html
- Mitigate some Exploits for Windows'® UAC, https://skanthak.hier-im-netz.de/uacamole.html

## Authors



---