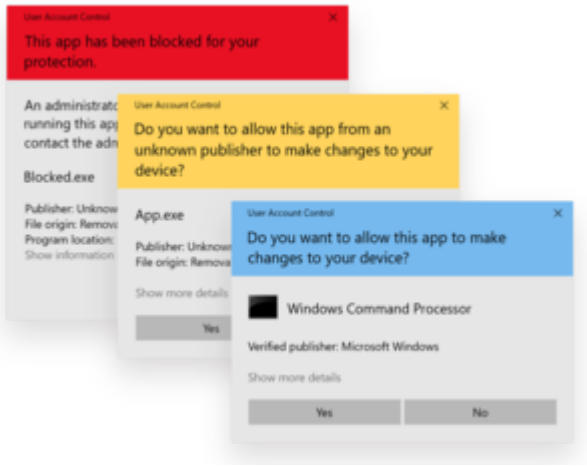
WIKIPEDIA

User Account Control

User Account Control (**UAC**) is a mandatory access control enforcement facility introduced with Microsoft's Windows Vista[1] and Windows Server 2008 operating systems, with a more relaxed[2] version also present in Windows 7, Windows Server 2008 R2, Windows 8, Windows Server 2012 and Windows 10. It aims to improve the security of Microsoft Windows by limiting application software to standard user privileges until an administrator authorizes an increase or elevation. In this way, only applications trusted by the user may receive administrative privileges, and malware should be kept from compromising the operating system. In other words, a user account may have administrator privileges assigned to it, but applications that the user runs do not inherit those privileges unless they are approved beforehand or the user explicitly authorizes it.



User Account Control "Windows Security" alerts in Windows 10 in light mode. From top to bottom: blocked app, app with unknown publisher, app with a known/trusted publisher.

UAC uses Mandatory Integrity Control to isolate running processes with different privileges. To reduce the possibility of lower-privilege applications communicating with higher-privilege ones, another new technology, User Interface Privilege Isolation, is used in conjunction with User Account Control to isolate these processes from each other.[3] One prominent use of this is Internet Explorer 7's "Protected Mode".[4]

Contents

History

Operating systems on mainframes and on servers have differentiated between superusers and userland for decades. This had an obvious security component, but also an administrative component, in that it prevented users from accidentally changing system settings.

Early Microsoft home operating-systems (such as MS-DOS, Windows 95, Windows 98 and Windows Me) did not have a concept of different user-accounts on the same machine. Subsequent versions of Windows and Microsoft applications encouraged the use of non-administrator user-logons, yet some applications continued to require administrator rights. Microsoft does not certify applications as Windows-compliant if they require administrator privileges; such applications may not use the Windows-compliant logo with their packaging.

- **MS-DOS** and Windows versions **1.0** to **3.11:** all applications had privileges equivalent to the operating system;
- **Windows 9x and Windows Me:** all applications enjoyed system-wide privileges rivaling those of the operating system itself;
- **All versions of Windows NT up to Windows XP:** introduced multiple user-accounts, but in practice most users continued to function as an administrator for their normal operations. Further, some applications would require that the user be an administrator for some or all of their functions to work.[5]
- **Windows Vista:** Microsoft developed Vista security firstly from the *Limited User Account* (LUA), then renamed the concept to *User Account Protection* (UAP) before finally shipping User Account Control (UAC).[6] Introduced in Windows Vista, User Account Control (UAC) offers an approach to encourage "super-user when necessary". The key to UAC lies in its ability to elevate privileges without changing the user context (user "Bob" is still user "Bob"). As always, it is difficult to introduce new security features without breaking compatibility with existing applications.
  - When someone logs into Vista as a standard user, the system sets up a logon session and assigns a token containing only the most basic privileges. In this way, the new logon session cannot make changes that would affect the entire system.
  - When a person logs in as a user with membership in the Administrators group, the system assigns two separate tokens: the first token contains all privileges typically awarded to an administrator, and the second is a restricted token similar to what a standard user would receive.
    - User applications, including the Windows Shell, then start with the restricted token, resulting in a reduced-privilege environment – even when running under an Administrator account.
    - When an application requests higher privileges or when a user selects a "Run as administrator" option, UAC will prompt standard users to enter the credentials of an Administrator account and prompt Administrators for confirmation and, if consent is given, continue or start the process using an unrestricted token.[7]
- **Windows 7:** Microsoft included a user interface to change User Account Control settings, and introduced one new notification mode: the *default* setting. By default, UAC does not prompt for consent when users make changes to Windows settings that require elevated permission through programs stored in %SystemRoot% and digitally signed by Microsoft. Programs that require permission to run still trigger a prompt. Other User Account Control settings that can be changed through the new UI could have been accessed through the registry in Windows Vista.[8]
- **Windows 8** and **8.1:** add a design change. When UAC is triggered, all applications and the taskbar are hidden when the desktop is dimmed.
- **Windows 10:** copies the same layout as Windows 8 and 8.1, but the Anniversary Update has a more modern look. Also, Windows 10 adds support for Windows Hello in the User Account Control dialog box.

Tasks that trigger a UAC prompt

Tasks that require administrator privileges will trigger a UAC prompt (if UAC is enabled); they are typically marked by a security shield icon with the 4 colors of the Windows logo (in Vista and Windows Server 2008) or with two panels yellow and two blue (Windows 7, Windows Server 2008 R2 and later). In the case of executable files, the icon will have a security shield overlay. The following tasks require administrator privileges:[9][10]

- Running an Application as an Administrator
- Changes to system-wide settings
- Changes to files in folders that standard users don't have permissions for (such as %SystemRoot% or %ProgramFiles% in most cases)
- Changes to an access control list (ACL), commonly referred to as file or folder permissions
- Installing and uninstalling applications outside of:
  - The %USERPROFILE% (e.g. C:\Users\{logged in user}) folder and its sub-folders.
    - Most of the time this is in %APPDATA%. (e.g. C:\Users\{logged in user}\AppData), by default, this is a hidden folder.
      - Chrome's and Firefox's installer ask for admin rights during install, if given, Chrome will install in the Program Files folder and be usable for all users, if denied, Chrome will install in the %APPDATA% folder instead and only be usable by the current user.
  - The Microsoft Store.
  - The folder of the installer and its sub-folders.
    - Steam installs its games in the /steamapps/ sub-folder, thus not prompting UAC. Some games require prerequisites to be installed, which may prompt UAC.
- Installing device drivers
- Installing ActiveX controls
- Changing settings for Windows Firewall
- Changing UAC settings
- Configuring Windows Update
- Adding or removing user accounts
- Changing a user's account name or type
- Creating a new account or deleting a user account
- Turning on Guest account (Windows 7 and 8.1)
- Turning on network discovery, file and printer sharing, Public folder sharing, turning off password protected sharing or turning on media streaming
- Configuring Parental Controls (in Windows 7) or Family Safety (Windows 8.1)
- Running Task Scheduler
- Backing up and restoring folders and files
- Merging and deleting network locations
- Turning on or cleaning logging in Remote Access Preferences
- Running Color Calibration
- Changing remote, system protection or advanced system settings
- Restoring backed-up system files
- Viewing or changing another user's folders and files

- Running Disk Defragmenter, System Restore or Windows Easy Transfer (Windows 7 and 8.1)
- Running Registry Editor
- Running the Windows Experience Index assessment
- Troubleshoot audio recording and playing, hardware / devices and power use
- Change power settings, turning off Windows features, uninstall, change or repair a program
- Change date and time and synchronizing with an Internet time server
- Installing and uninstalling display languages
- Change Ease of Access administrative settings

Common tasks, such as changing the time zone, do not require administrator privileges[11] (although changing the system time itself does, since the system time is commonly used in security protocols such as Kerberos). A number of tasks that required administrator privileges in earlier versions of Windows, such as installing critical Windows updates, no longer require administrator privileges in Vista.[12] Any program can be run as administrator by right-clicking its icon and clicking "Run as administrator", except MSI or MSU packages as, due to their nature, if administrator rights will be required a prompt will usually be shown. Should this fail, the only workaround is to run a Command Prompt as an administrator and launch the MSI or MSP package from there.

Features

User Account Control asks for credentials in a *Secure Desktop* mode, where the entire screen is temporarily dimmed, Windows Aero disabled, and only the authorization window at full brightness, to present only the elevation user interface (UI). Normal applications cannot interact with the Secure Desktop. This helps prevent spoofing, such as overlaying different text or graphics on top of the elevation request, or tweaking the mouse pointer to click the confirmation button when that's not what the user intended.[13] If an administrative activity comes from a minimized application, the secure desktop request will also be minimized so as to prevent the focus from being lost. It is possible to disable *Secure Desktop*, though this is inadvisable from a security perspective.[14]

In earlier versions of Windows, Applications written with the assumption that the user will be running with administrator privileges experienced problems when run from limited user accounts, often because they attempted to write to machine-wide or system directories (such as *Program Files*) or registry keys (notably HKLM).[5] UAC attempts to alleviate this using *File and Registry Virtualization*, which redirects writes (and subsequent reads) to a per-user location within the user's profile. For example, if an application attempts to write to a directory such as "C:\Program Files\appname\settings.ini" to which the user does not have write permission, the write will be redirected to "C:\Users\username\AppData\Local\VirtualStore\Program Files\appname\settings.ini". The redirection feature is only provided for non-elevated 32-bit applications, and only if they do not include a manifest that requests specific privileges.[15]

There are a number of configurable UAC settings. It is possible to:[16]

- Require administrators to re-enter their password for heightened security,
- Require the user to press Ctrl+Alt+Del as part of the authentication process for heightened security;
- Disable only file and registry virtualization[17]
- Disable *Admin Approval Mode* (UAC prompts for administrators) entirely; note that, while this disables the UAC confirmation dialogs, it does not disable Windows' built-in LUA feature, which means that users, even those marked as administrators, are still limited users with no true administrative access.

Command Prompt windows that are running elevated will prefix the title of the window with the word "Administrator", so that a user can discern which instances are running with elevated privileges.[18]

A distinction is made between elevation requests from a signed executable and an unsigned executable; and if the former, whether the publisher is 'Windows Vista'. The color, icon, and wording of the prompts are different in each case; for example, attempting to convey a greater sense of warning if the executable is unsigned than if not.[19]

Internet Explorer 7's "Protected Mode" feature uses UAC to run with a 'low' integrity level (a Standard user token has an integrity level of 'medium'; an elevated (Administrator) token has an integrity level of 'high'). As such, it effectively runs in a sandbox, unable to write to most of the system (apart from the Temporary Internet Files folder) without elevating via UAC.[7][20] Since toolbars and ActiveX controls run within the Internet Explorer process, they will run with low privileges as well, and will be severely limited in what damage they can do to the system.[21]

Requesting elevation

A program can request elevation in a number of different ways. One way for program developers is to add a requestedPrivileges section to an XML document, known as the manifest, that is then embedded into the application. A manifest can specify dependencies, visual styles, and now the appropriate security context:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<assembly xmlns="urn:schemas-microsoft-com:asm.v1" manifestVersion="1.0">
  <v3:trustInfo xmlns:v3="urn:schemas-microsoft-com:asm.v3">
    <v3:security>
      <v3:requestedPrivileges>
        <v3:requestedExecutionLevel level="highestAvailable"/>
      </v3:requestedPrivileges>
    </v3:security>
  </v3:trustInfo>
</assembly>
```

Setting the level attribute for requestedExecutionLevel to "asInvoker" will make the application run with the token that started it, "highestAvailable" will present a UAC prompt for administrators and run with the usual reduced privileges for standard users, and "requireAdministrator" will require elevation.[22] In both highestAvailable and requireAdministrator modes, failure to provide confirmation results in the program not being launched.

An executable that is marked as "`requireAdministrator`" in its manifest cannot be started from a non-elevated process using `CreateProcess()`. Instead, `ERROR_ELEVATION_REQUIRED` will be returned. `ShellExecute()` or `ShellExecuteEx()` must be used instead. If an `HWND` is not supplied, then the dialog will show up as a blinking item in the taskbar.

Inspecting an executable's manifest to determine if it requires elevation is not recommended, as elevation may be required for other reasons (setup executables, application compatibility). However, it is possible to programmatically detect if an executable will require elevation by using `CreateProcess()` and setting the `dwCreationFlags` parameter to `CREATE_SUSPENDED`. If elevation is required, then `ERROR_ELEVATION_REQUIRED` will be returned.[23] If elevation is not required, a success return code will be returned at which point one can use `TerminateProcess()` on the newly created, suspended process. This will not allow one to detect that an executable requires elevation if one is already executing in an elevated process, however.

A new process with elevated privileges can be spawned from within a .NET application using the "runas" verb. An example using C#:

```
System.Diagnostics.Process proc = new System.Diagnostics.Process();
proc.StartInfo.FileName = "C:\\Windows\\system32\\notepad.exe";
proc.StartInfo.Verb = "runas"; // Elevate the application
proc.StartInfo.UseShellExecute = true;
proc.Start();
```

In a native Win32 application the same "runas" verb can be added to a `ShellExecute()` or `ShellExecuteEx()` call:[7]

```
ShellExecute(hwnd, "runas", "C:\\Windows\\Notepad.exe", 0, 0, SW_SHOWNORMAL);
```

In the absence of a specific directive stating what privileges the application requests, UAC will apply heuristics, to determine whether or not the application needs administrator privileges. For example, if UAC detects that the application is a setup program, from clues such as the filename, versioning fields, or the presence of certain sequences of bytes within the executable, in the absence of a manifest it will assume that the application needs administrator privileges.[24]

Security

UAC is a convenience feature; it neither introduces a security boundary nor prevents execution of malware.[25][26][27][28]

Leo Davidson discovered that Microsoft weakened UAC in Windows 7 through exemption of about 70 Windows programs from displaying a UAC prompt and presented a proof of concept for a privilege escalation.[29]

Stefan Kanthak presented a proof of concept for a privilege escalation via UAC's installer detection and IExpress installers.[30]

Stefan Kanthak presented another proof of concept for arbitrary code execution as well as privilege escalation via UAC's auto-elevation and binary planting.[31]

Criticism

There have been complaints that UAC notifications slow down various tasks on the computer such as the initial installation of software onto Windows Vista.[32] It is possible to turn off UAC while installing software, and re-enable it at a later time.[33] However, this is not recommended since, as File & Registry Virtualization is only active when UAC is turned on, user settings and configuration files may be installed to a different place (a system directory rather than a user-specific directory) if UAC is switched off than they would be otherwise.[14] Also Internet Explorer 7's "Protected Mode", whereby the browser runs in a sandbox with lower privileges than the standard user, relies on UAC; and will not function if UAC is disabled.[20]

Yankee Group analyst Andrew Jaquith said, six months before Vista was released, that "while the new security system shows promise, it is far too chatty and annoying."[34] By the time Windows Vista was released in November 2006, Microsoft had drastically reduced the number of operating system tasks that triggered UAC prompts, and added file and registry virtualization to reduce the number of legacy

applications that triggered UAC prompts.[5] However, David Cross, a product unit manager at Microsoft, stated during the RSA Conference 2008 that UAC was in fact designed to "annoy users," and force independent software vendors to make their programs more secure so that UAC prompts would not be triggered.[35] Software written for Windows XP, and many peripherals, would no longer work in Windows Vista or 7 due to the extensive changes made in the introduction of UAC. The compatibility options were also insufficient. In response to these criticisms, Microsoft altered UAC activity in Windows 7. For example, by default users are not prompted to confirm many actions initiated with the mouse and keyboard alone such as operating Control Panel applets.

In a controversial article, New York Times Gadgetwise writer Paul Boutin said "Turn off Vista's overly protective User Account Control. Those pop-ups are like having your mother hover over your shoulder while you work."[36] Computerworld journalist Preston Gralla described the NYT article as "...one of the worst pieces of technical advice ever issued."[37]

See also

- Comparison of privilege authorization features
- Features new to Windows Vista
- Polkit
- runas
- Secure attention key (SAK)
- Security and safety features new to Windows Vista
- sudo – A similar feature in UNIX-like operating systems

References

1. "What is User Account Control?" (http://windows.microsoft.com/en-id/windows/what-is-user-account-control#1TC=windows-vista). Microsoft. January 2015. Retrieved 2015-07-28.
2. Windows 7 Feature Focus: User Account Control (http://winsupersite.com/article/windows-7/windows-7-feature-focus-user-account-control), An overview of UAC in Windows 7 by Paul Thurott
3. "The Windows Vista and Windows Server 2008 Developer Story: Windows Vista Application Development Requirements for User Account Control (UAC)" (https://msdn.microsoft.com/en-us/library/aa905330.aspx). *The Windows Vista and Windows Server 2008 Developer Story Series*. Microsoft. April 2007. Retrieved 2007-10-08.
4. Marc Silbey, Peter Brundrett (January 2006). "Understanding and Working in Protected Mode Internet Explorer" (https://msdn.microsoft.com/en-us/library/bb250462.aspx). Microsoft. Retrieved 2007-12-08.
5. Torre, Charles (March 5, 2007). "UAC – What. How. Why" (http://channel9.msdn.com/ShowPost.aspx?PostID=288259) (video). Retrieved 2007-12-08.
6. Howard, Michael; LeBlanc, David (2010). *Writing Secure Code for Windows Vista* (https://books.google.com/books?id=bRpSUetqwc8C). O'Reilly Media, Inc. ISBN 9780735649316. Retrieved 2013-08-06. "UAC started life as the Limited User Account (LUA), then was renamed to User Account Protection (UAP), and finally we got UAC."
7. Kerr, Kenny (September 29, 2006). "Windows Vista for Developers – Part 4 – User Account Control" (https://weblogs.asp.net/kennykerr/archive/2006/09/29/Windows-Vista-for-Developers-_1320_-Part-4-_1320_-User-Account-Control.aspx). Retrieved 2007-03-15.
8. "Registry Tweaks to Customize User Account Control (UAC) Options in Windows Vista and Later - AskVG" (https://www.askvg.com/how-to-tweak-user-account-control-uac-options-in-windows-vista-home-basic-home-premium/).

9. Bott, Ed (2007-02-02). "What triggers User Account Control prompts?" (https://web.archive.org/web/20150927215218/http://www.edbott.com/weblog/2007/02/what-triggers-user-account-control-prompts/). Archived from the original (http://www.edbott.com/weblog/2007/02/what-triggers-user-account-control-prompts/) on 2015-09-27.
10. "Living with and benefiting from User Account Control" (http://windows.microsoft.com/en-us/windows-vista/living-with-and-benefiting-from-user-account-control-from-windows-vista-inside-out). Microsoft. 2014-12-09.
11. Allchin, Jim (2007-01-23). "Security Features vs. Convenience" (http://windowsvistablog.com/blogs/windowsvista/archive/2007/01/23/security-features-vs-convenience.aspx). *Windows Vista Team Blog*. Microsoft.
12. "User Account Control Overview" (https://technet.microsoft.com/en-us/library/aa906021.aspx). *TechNet*. Microsoft.
13. "User Account Control Prompts on the Secure Desktop" (http://blogs.msdn.com/b/uac/archive/2006/05/03/589561.aspx). *UACBlog*. Microsoft. 4 May 2006.
14. Bott, Ed (2 February 2007). "Why you need to be discriminating with those Vista tips" (http://www.edbott.com/weblog/2007/02/what-happens-when-those-tips-dont-work/). *Ed Bott's Windows Expertise*.
15. "Determine How to Fix Applications That Are Not Windows 7 Compliant" (https://technet.microsoft.com/en-us/library/ee732424.aspx). *TechNet*. Microsoft. Retrieved 2013-09-09.
16. "Chapter 2: Defend Against Malware" (https://technet.microsoft.com/en-us/library/bb629436.aspx). *Windows Vista Security Guide*. Microsoft. November 8, 2006.
17. User Account Control: Virtualize file and registry write failures to per-user locations (https://technet.microsoft.com/en-us/library/dd851895.aspx)
18. "Administrator Marking for Command Prompt" (http://blogs.msdn.com/uac/archive/2006/08/01/685645.aspx). *UACBlog*. Microsoft. 1 August 2006.
19. "Accessible UAC Prompts" (https://web.archive.org/web/20080127133403/http://windowsvistablog.com/blogs/windowsvista/archive/2007/01/25/accessible-uac-prompts.aspx). *Windows Vista Blog*. Microsoft. Archived from the original (http://windowsvistablog.com/blogs/windowsvista/archive/2007/01/25/accessible-uac-prompts.aspx) on 2008-01-27. Retrieved 2008-02-13.
20. Russinovich, Mark (June 2007). "Inside Windows Vista User Account Control" (http://www.microsoft.com/technet/technetmag/issues/2007/06/UAC/). *TechNet Magazine*. Microsoft.
21. Friedman, Mike (10 February 2006). "Protected Mode in Vista IE7" (http://blogs.msdn.com/ie/archive/2006/02/09/528963.aspx). *IEBlog*. Microsoft.
22. Carlisle, Mike (10 March 2007). "Making Your Application UAC Aware" (http://www.codeproject.com/KB/vista-security/MakingAppsUACAware.aspx). *The Code Project*.
23. Zhang, Junfeng (18 October 2006). "Programmatically determine if an application requires elevation in Windows Vista" (http://blogs.msdn.com/b/junfeng/archive/2006/10/18/programmatically-determine-if-an-application-requires-elevation-in-windows-vista.aspx). *Junfeng Zhang's Windows Programming Notes*. Microsoft.
24. "Understanding and Configuring User Account Control in Windows Vista" (https://technet.microsoft.com/en-us/library/cc709628.aspx). *TechNet*. Microsoft. Retrieved 2007-07-05.
25. "Disabling User Account Control (UAC) on Windows Server" (https://support.microsoft.com/en-us/kb/2526083). *Microsoft Support Knowledge Base*. Microsoft. Retrieved 2015-08-17.
26. Russinovich, Mark. "Inside Windows 7 User Account Control" (https://technet.microsoft.com/en-us/magazine/2009.07.uac.aspx). *Microsoft*. Retrieved 2015-08-25.
27. Johansson, Jesper. "The Long-Term Impact of User Account Control" (https://technet.microsoft.com/en-us/magazine/2007.09.securitywatch.aspx). *Microsoft*. Retrieved 2015-08-25.
28. Russinovich, Mark. "Inside Windows Vista User Account Control" (https://technet.microsoft.com/en-us/magazine/2007.06.uac.aspx). *Microsoft*. Retrieved 2015-08-25.

29. Davidson, Leo. "Windows 7 UAC whitelist: – Code-injection Issue – Anti-Competitive API – Security Theatre" (http://www.pretentiousname.com/misc/win7_uac_whitelist2.html). Retrieved 2015-08-25.
30. Kanthak, Stefan. "Defense in depth – the Microsoft way (part 11): privilege escalation for dummies" (http://seclists.org/fulldisclosure/2013/Oct/5). *Full disclosure (mailing list)*. Retrieved 2015-08-17.
31. Kanthak, Stefan. "Defense in depth – the Microsoft way (part 31): UAC is for binary planting" (http://seclists.org/fulldisclosure/2015/Mar/92). *Full disclosure (mailing list)*. Retrieved 2015-08-25.
32. Trapani, Gina (31 January 2007). "Geek to Live: Windows Vista upgrade power tips" (https://lifehacker.com/software/vista/geek-to-live-windows-vista-upgrade-power-tips-231922.php). *Lifehacker*.
33. "Disable UAC in Vista" (https://www.youtube.com/watch?v=M7Uwx_yaxUM).
34. Evers, Joris (2006-05-07). "Report: Vista to hit anti-spyware, firewall markets" (https://web.archive.org/web/20061210153354/http://news.zdnet.com/2100-1009_22-6069464.html). *ZDNet*. CBS Interactive. Archived from the original (http://news.zdnet.com/2100-1009_22-6069464.html) on 2006-12-10. Retrieved 2007-01-21.
35. Espiner, Tom (11 April 2008). "Microsoft: Vista feature designed to 'annoy users' " (http://news.cnet.com/Microsoft-Vista-feature-designed-to-annoy-users/2100-1016_3-6237191.html). *CNET*. CBS Interactive.
36. Boutin, Paul (14 May 2009). "How to Wring a Bit More Speed From Vista" (https://query.nytimes.com/gst/fullpage.html?res=9907E5DC1131F937A25756C0A96F9C8B63). *New York Times* – *Gadgetwise*. Retrieved 2015-01-04.
37. Gralla, Preston (14 May 2009). "NYT Offers Bad Tech Advice" (http://www.pcworld.com/article/164911/nyt_offers_bad_tech_advice.html). *PCworld.com*. Retrieved 2015-01-04.

External links

- Turning UAC On or Off (https://windows.microsoft.com/en-US/windows7/turn-user-account-control-on-or-off) in Windows 7
- Documentation about UAC for Windows 7, Windows Server 2008, Windows Server 2008 R2, Windows Vista (https://technet.microsoft.com/en-us/library/cc731416.aspx)
- UAC Understanding and Configuring (https://technet.microsoft.com/WindowsVista/en/library/00d04415-2b2f-422c-b70e-b18ff918c2811033.mspx?mfr=true) More Information at Microsoft Technet
- Development Requirements for User Account Control Compatibility (https://msdn.microsoft.com/en-us/library/bb530410.aspx) More information at Microsoft Developer Network
- UAC Team Blog (http://blogs.msdn.com/uac/)