

How User Account Control Works

Updated: August 31, 2012

Applies To: Windows Server 2012, Windows 8

User Account Control (UAC) helps prevent malicious programs (also called malware) from damaging a computer and helps organizations deploy a better-managed desktop. With UAC, applications and tasks always run in the security context of a non-administrator account, unless an administrator specifically authorizes administrator-level access to the system. UAC can block the automatic installation of unauthorized applications and prevent inadvertent changes to system settings.

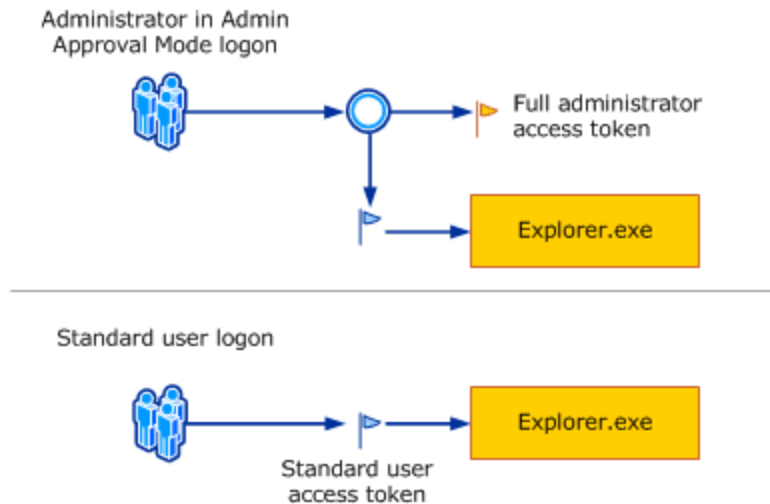
UAC Process and Interactions

Each application that requires the administrator access token must prompt the administrator for consent. The one exception is the relationship that exists between parent and child processes. Child processes inherit the user access token from the parent process. Both the parent and child processes, however, must have the same integrity level. Windows Server 2012 protects processes by marking their integrity levels. Integrity levels are measurements of trust. A "high" integrity application is one that performs tasks that modify system data, such as a disk partitioning application, while a "low" integrity application is one that performs tasks that could potentially compromise the operating system, such as a Web browser. Applications with lower integrity levels cannot modify data in applications with higher integrity levels. When a standard user attempts to run an application that requires an administrator access token, UAC requires that the user provide valid administrator credentials.

In order to better understand how this process happens it is important to review the details of the Windows Server 2012 logon process.

Windows Server® 2012 Logon Process

The following illustration demonstrates how the logon process for an administrator differs from the logon process for a standard user.



By default, standard users and administrators access resources and run applications in the security context of standard users. When a user logs on to a computer, the system creates an access token for that user. The access token contains information about the level of access that the user is granted, including specific security identifiers (SIDs) and Windows privileges.

When an administrator logs on, two separate access tokens are created for the user: a standard user access token and an administrator access token. The standard user access token contains the same user-specific information as the administrator access token, but the administrative Windows privileges and SIDs are removed. The standard user access token is used to start applications that do not perform administrative tasks (standard user applications). The standard user access token is then used to display the desktop (Explorer.exe). Explorer.exe is the parent process from which all other user-initiated processes inherit their access token. As a result, all applications run as a standard user unless a user provides consent or credentials to approve an application to use a full administrative access token.

A user that is a member of the Administrators group can log on, browse the Web, and read e-mail while using a standard user access token. When the administrator needs to perform a task that requires the administrator access token, Windows Server 2012 automatically prompts the user for approval. This prompt is called an elevation prompt, and its behavior can be configured by using the Local Security Policy snap-in (Secpol.msc) or Group Policy.

Note

The term "elevate" is used to refer to the process in Windows Server 2012 that prompts the user for consent or credentials to use a full administrator access token.

The UAC User Experience

When UAC is enabled, the user experience for standard users is different from that of administrators in Admin Approval Mode. The recommended and more secure method of

running Windows Server 2012 is to make your primary user account a standard user account. Running as a standard user helps to maximize security for a managed environment. With the built-in UAC elevation component, standard users can easily perform an administrative task by entering valid credentials for a local administrator account. The default, built-in UAC elevation component for standard users is the credential prompt.

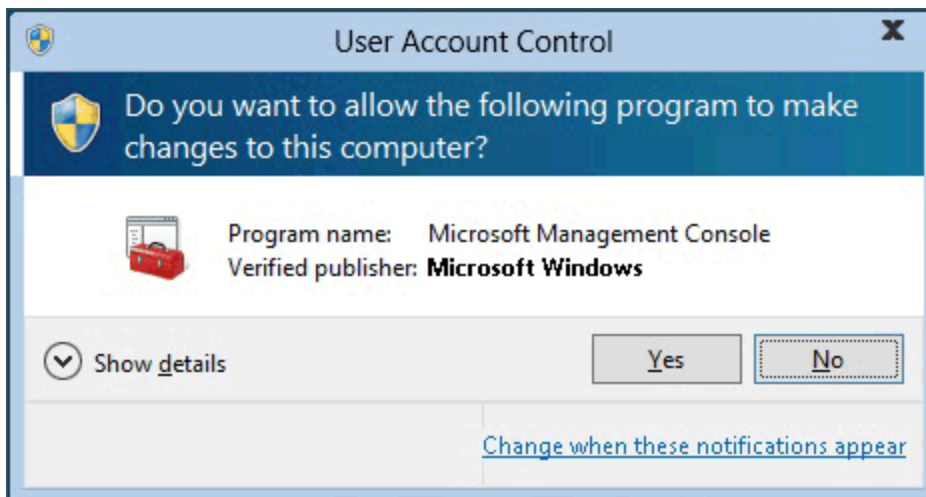
The alternative to running as a standard user is to run as an administrator in Admin Approval Mode. With the built-in UAC elevation component, members of the local Administrators group can easily perform an administrative task by providing approval. The default, built-in UAC elevation component for an administrator account in Admin Approval Mode is called the consent prompt. The UAC elevation prompting behavior can be configured by using the Local Security Policy snap-in (Secpol.msc) or Group Policy.

The consent and credential prompts

With UAC enabled, Windows Server 2012 prompts for consent or prompts for credentials of a valid local administrator account before starting a program or task that requires a full administrator access token. This prompt ensures that no malicious software can be silently installed.

The consent prompt

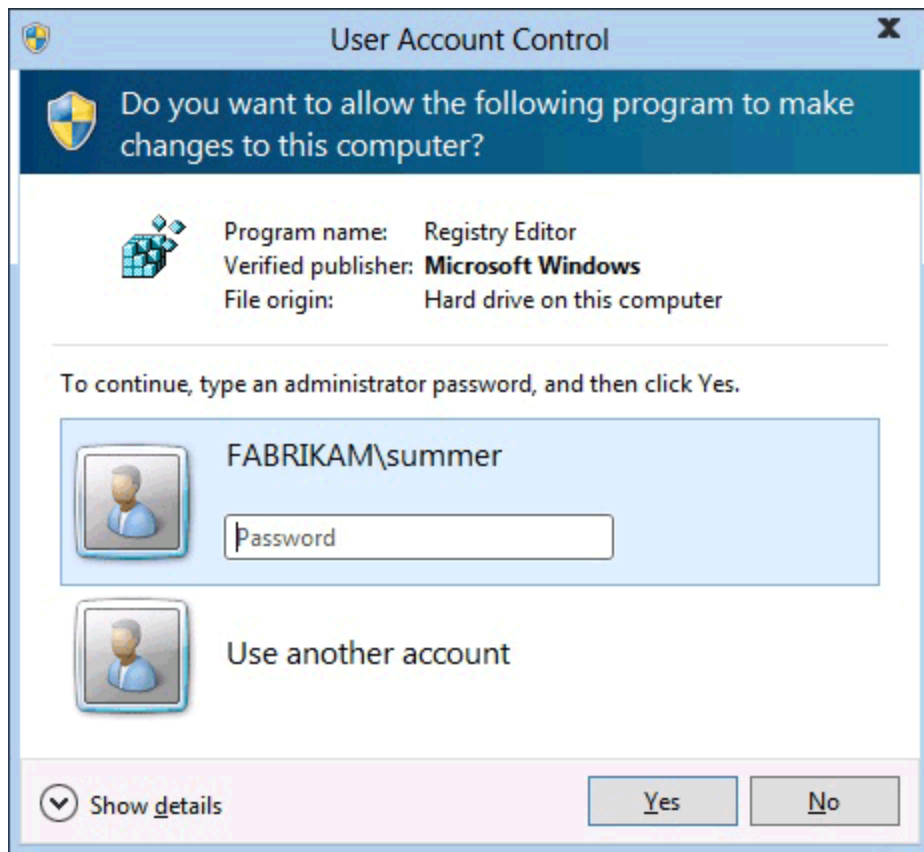
The consent prompt is presented when a user attempts to perform a task that requires a user's administrative access token. The following is a screen shot of the UAC consent prompt.



The credential prompt

The credential prompt is presented when a standard user attempts to perform a task that requires a user's administrative access token. This standard user default prompt behavior can be configured by using the Local Security Policy snap-in (Secpol.msc) or Group Policy. Administrators can also be required to provide their credentials by setting the User Account Control: Behavior of the elevation prompt for administrators in Admin Approval Mode policy setting value to Prompt for credentials.

The following screen shot is an example of the UAC credential prompt.



UAC elevation prompts

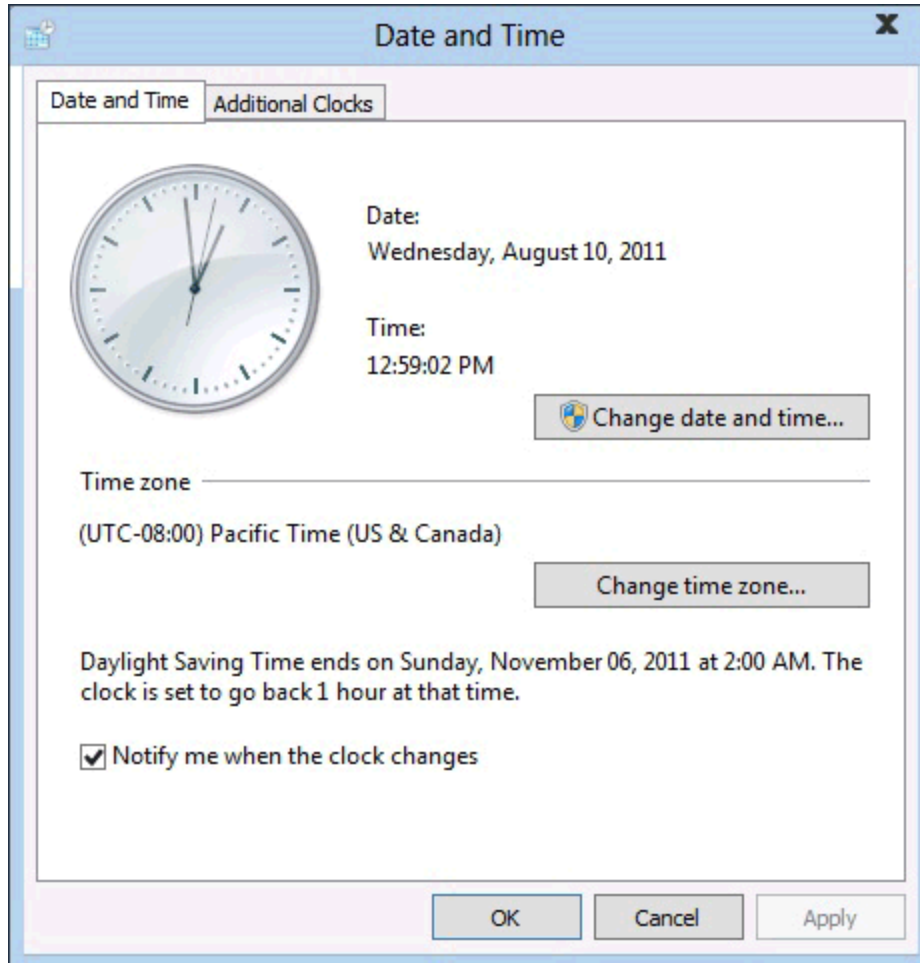
The UAC elevation prompts are color-coded to be application-specific, enabling for immediate identification of an application's potential security risk. When an application attempts to run with an administrator's full access token, Windows Server 2012 first analyzes the executable file to determine its publisher. Applications are first separated into three categories based on the executable file's publisher: Windows Server 2012, publisher verified (signed), and publisher not verified (unsigned). The following diagram illustrates how Windows Server 2012 determines which color elevation prompt to present to the user.

The elevation prompt color-coding is as follows:

- Red background with a red shield icon: The application is blocked by Group Policy or is from a publisher that is blocked.
- Blue background with a blue and gold shield icon: The application is a Windows Server 2012 administrative application, such as a Control Panel item.
- Blue background with a blue shield icon: The application is signed by using Authenticode and is trusted by the local computer.
- Yellow background with a yellow shield icon: The application is unsigned or signed but is not yet trusted by the local computer.

Shield icon

Some Control Panel items, such as **Date and Time Properties**, contain a combination of administrator and standard user operations. Standard users can view the clock and change the time zone, but a full administrator access token is required to change the local system time. The following is a screen shot of the **Date and Time Properties** Control Panel item.



The shield icon on the **Change date and time** button indicates that the process requires a full administrator access token and will display a UAC elevation prompt.

Securing the elevation prompt

The elevation process is further secured by directing the prompt to the secure desktop. The consent and credential prompts are displayed on the secure desktop by default in Windows Server 2012. Only Windows processes can access the secure desktop. For higher levels of security, we recommend keeping the **User Account Control: Switch to the secure desktop when prompting for elevation** policy setting enabled.

When an executable file requests elevation, the interactive desktop, also called the user desktop, is switched to the secure desktop. The secure desktop dims the user desktop and displays an elevation prompt that must be responded to before continuing. When the user

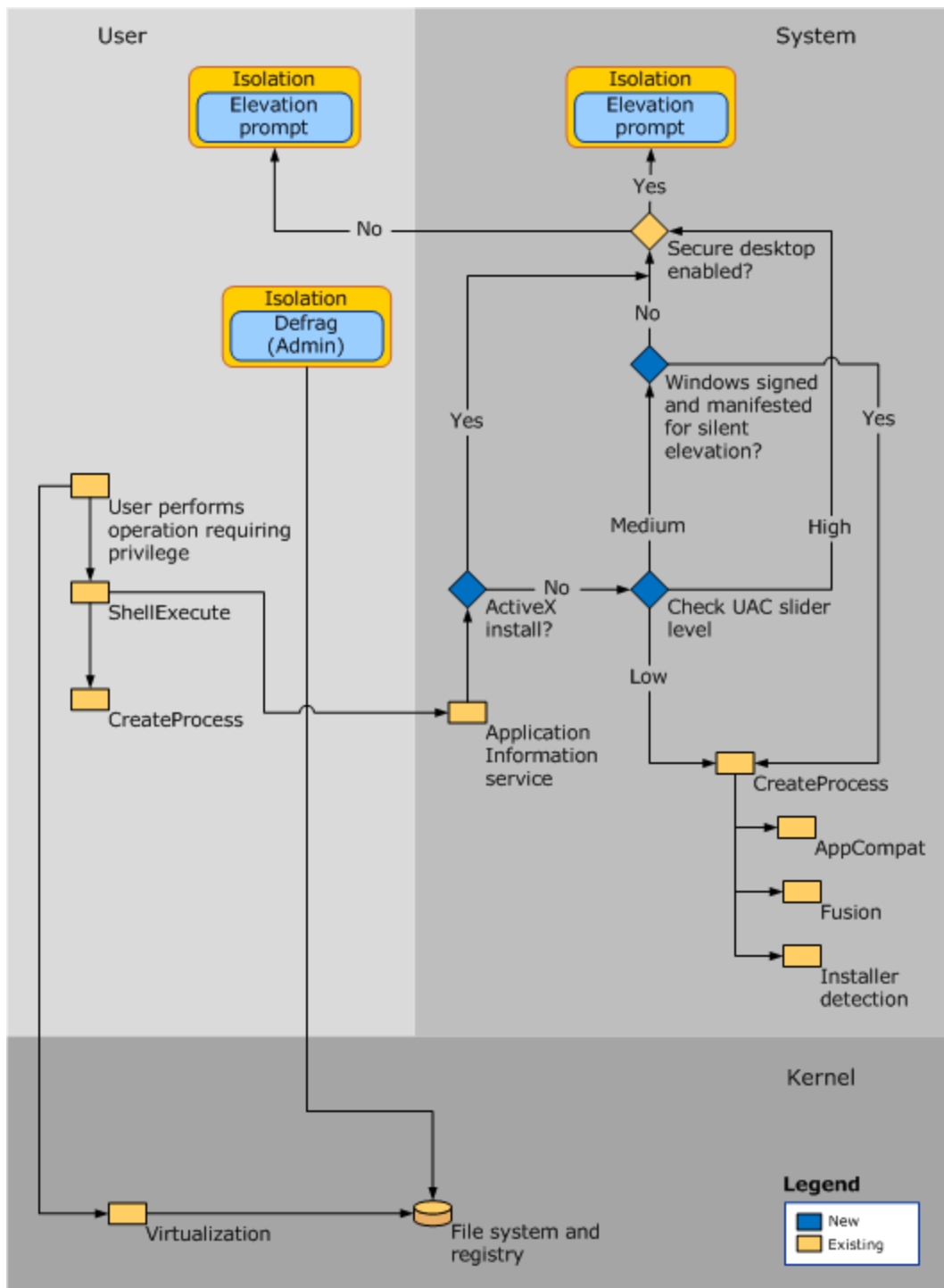
clicks Yes or No, the desktop switches back to the user desktop.

Malware can present an imitation of the secure desktop, but when the User Account Control: Behavior of the elevation prompt for administrators in Admin Approval Mode policy setting is set to Prompt for consent, the malware does not gain elevation if the user clicks Yes on the imitation. If the policy setting is set to Prompt for credentials, malware imitating the credential prompt may be able to gather the credentials from the user. However, the malware does not gain elevated privilege and the system has other protections that mitigate malware from taking control of the user interface even with a harvested password.

While malware could present an imitation of the secure desktop, this issue cannot occur unless a user previously installed the malware on the computer. Because processes requiring an administrator access token cannot silently install when UAC is enabled, the user must explicitly provide consent by clicking **Yes** or by providing administrator credentials. The specific behavior of the UAC elevation prompt is dependent upon Group Policy.

UAC Architecture

The following diagram details the UAC architecture.



To better understand each component, review the table below:

Component	Description
User	

User performs operation requiring privilege	If the operation changes the file system or registry, Virtualization is called. All other operations call ShellExecute.
ShellExecute	ShellExecute calls CreateProcess. ShellExecute looks for the ERROR_ELEVATION_REQUIRED error from CreateProcess. If it receives the error, ShellExecute calls the Application Information service to attempt to perform the requested task with the elevated prompt.
CreateProcess	If the application requires elevation, CreateProcess rejects the call with ERROR_ELEVATION_REQUIRED.
System	
Application Information service	A system service that helps start applications that require one or more elevated privileges or user rights to run, such as local administrative tasks, and applications that require higher integrity levels. The Application Information service helps start such applications by creating a new process for the application with an administrative user's full access token when elevation is required and (depending on Group Policy) consent is given by the user to do so.
Elevating an ActiveX install	If ActiveX is not installed, the system checks the UAC slider level. If ActiveX is installed, the User Account Control: Switch to the secure desktop when prompting for elevation Group Policy setting is checked.
Check UAC slider level	<p>UAC now has four levels of notification to choose from and a slider to use to select the notification level:</p> <ul style="list-style-type: none"> • High <p>If the slider is set to Always notify, the system checks whether the secure desktop is enabled.</p> • Medium <p>If the slider is set to Default-Notify me only when programs try to make changes to my computer, the User Account Control: Only elevate executable files that are signed and validated policy setting is checked:</p> <ul style="list-style-type: none"> ◦ If the policy setting is enabled, the public key infrastructure (PKI) certification path validation is enforced for a given executable file before it is permitted to run. ◦ If the policy setting is not enabled (default), the PKI certification path validation is not enforced before a given executable file is

	<p>permitted to run. The User Account Control: Switch to the secure desktop when prompting for elevation Group Policy setting is checked.</p> <ul style="list-style-type: none"> • Low <p>If the slider is set to Notify me only when programs try to make changes to my computer (do not dim by desktop), the CreateProcess is called.</p> <ul style="list-style-type: none"> • Never Notify <p>If the slider is set to Never notify me when, UAC prompt will never notify when a program is trying to install or trying to make any change on the computer.</p> <p>Important: This setting is not recommended. This setting is the same as setting the User Account Control: Behavior of the elevation prompt for administrators in Admin Approval Mode policy setting to Elevate without prompting.</p>
Secure desktop enabled	<p>The User Account Control: Switch to the secure desktop when prompting for elevation policy setting is checked:</p> <ul style="list-style-type: none"> • If the secure desktop is enabled, all elevation requests go to the secure desktop regardless of prompt behavior policy settings for administrators and standard users. • If the secure desktop is not enabled, all elevation requests go to the interactive user's desktop, and the per-user settings for administrators and standard users are used.
CreateProcess	<p>CreateProcess calls AppCompat, Fusion, and Installer detection to assess if the application requires elevation. The executable file is then inspected to determine its requested execution level, which is stored in the application manifest for the executable file. CreateProcess fails if the requested execution level specified in the manifest does not match the access token and returns an error (ERROR_ELEVATION_REQUIRED) to ShellExecute. For more information, see Requested execution levels later in this section.</p>
AppCompat	<p>The AppCompat database stores information in the application compatibility fix entries for an application.</p>
Fusion	<p>The Fusion database stores information from application manifests that describe the applications. The manifest schema is updated to add a new requested execution level field.</p>

Installer detection	Installer detection detects setup executable files, which helps prevent installations from being run without the user's knowledge and consent. For more information, see Installer detection technology later in this section.
Kernel	
Virtualization	Virtualization technology ensures that non-compliant applications do not silently fail to run or fail in a way that the cause cannot be determined. UAC also provides file and registry virtualization and logging for applications that write to protected areas. For more information, see Virtualization later in this section.
File system and registry	The per-user file and registry virtualization redirects per-computer registry and file write requests to equivalent per-user locations. Read requests are redirected to the virtualized per-user location first and to the per-computer location second.

There is a change on Windows Server 2012 UAC from previous Windows versions. The new slider will never turn UAC completely off. The new setting will:

- Keep the UAC service running.
- Cause all elevation request initiated by administrators to be auto-approved without showing a UAC prompt.
- Automatically deny all elevation requests for standard users.

Important

In order to fully disable UAC you must disable the policy **User Account Control: Run all administrators in Admin Approval Mode**.

Warning

Tailored Applications will not work on Windows Server 2012 when UAC is disabled.

Virtualization

Because system administrators in enterprise environments attempt to secure systems, many line-of-business (LOB) applications are designed to use only a standard user access token. As a result, IT administrators do not need to replace the majority of applications when running

Windows Server 2012 with UAC enabled.

Windows Server 2012 includes file and registry virtualization technology for applications that are not UAC compliant and that require an administrator's access token to run correctly. Virtualization ensures that even applications that are not UAC compliant are compatible with Windows Server 2012. When an administrative application that is not UAC compliant attempts to write to a protected directory, such as Program Files, UAC gives the application its own virtualized view of the resource it is attempting to change. The virtualized copy is maintained in the user's profile. This strategy creates a separate copy of the virtualized file for each user that runs the non-compliant application.

Most application tasks operate properly by using virtualization features. Although virtualization allows a majority of applications to run, it is a short-term fix and not a long-term solution. Application developers should modify their applications to be compliant with the Windows Server 2012 logo program as soon as possible, rather than relying on file, folder, and registry virtualization.

Virtualization is not in option in the following scenarios:

1. Virtualization does not apply to applications that are elevated and run with a full administrative access token.
2. Virtualization supports only 32-bit applications. Non-elevated 64-bit applications simply receive an access denied message when they attempt to acquire a handle (a unique identifier) to a Windows object. Native Windows 64-bit applications are required to be compatible with UAC and to write data into the correct locations.
3. Virtualization is disabled for an application if the application includes an application manifest with a requested execution level attribute.

Request Execution Levels

An application manifest is an XML file that describes and identifies the shared and private side-by-side assemblies that an application should bind to at run time. In Windows Server 2012, the application manifest includes entries for UAC application compatibility purposes. Administrative applications that include an entry in the application manifest prompt the user for permission to access the user's access token. Although they lack an entry in the application manifest, most administrative applications can run without modification by using application compatibility fixes. Application compatibility fixes are database entries that enable applications that are not UAC compliant to work properly with Windows Server 2012.

All UAC-compliant applications should have a requested execution level added to the application manifest. If the application requires administrative access to the system, then marking the application with a requested execution level of "require administrator" ensures that the system identifies this program as an administrative application and performs the necessary elevation steps. Requested execution levels specify the privileges required for an

application.

Installer Detection Technology

Installation programs are applications designed to deploy software. Most installation programs write to system directories and registry keys. These protected system locations are typically writeable only by an administrator in Installer detection technology, which means that standard users do not have sufficient access to install programs. Windows Server 2012 heuristically detects installation programs and requests administrator credentials or approval from the administrator user in order to run with access privileges. Windows Server 2012 also heuristically detects updates and programs that uninstall applications. One of the design goals of UAC is to prevent installations from being run without the user's knowledge and consent because installation programs write to protected areas of the file system and registry.

Installer detection only applies to:

- 32-bit executable files.
- Applications without a requested execution level attribute.
- Interactive processes running as a standard user with UAC enabled.

Before a 32-bit process is created, the following attributes are checked to determine whether it is an installer:

- The file name includes keywords such as "install," "setup," or "update."
- Versioning Resource fields contain the following keywords: Vendor, Company Name, Product Name, File Description, Original Filename, Internal Name, and Export Name.
- Keywords in the side-by-side manifest are embedded in the executable file.
- Keywords in specific StringTable entries are linked in the executable file.
- Key attributes in the resource script data are linked in the executable file.
- There are targeted sequences of bytes within the executable file.

Note

The keywords and sequences of bytes were derived from common characteristics observed from various installer technologies.

 **Note**

The User Account Control: Detect application installations and prompt for elevation policy setting must be enabled for installer detection to detect installation programs. This setting is enabled by default and can be configured locally by using the Local Security Policy snap-in (Secpol.msc) or configured for the domain, OU, or specific groups by Group Policy (Gpedit.msc).