

Realtime
publishers

"Leading the Conversation"

The Essentials Series:
Eliminating Administrator Rights

Limitations in Native Solutions for Privilege Management

sponsored by



by Greg Shields



Limitations in Native Solutions for Privilege Management1
Runas.....2
User Account Control3
Capabilities to Resolve Least Privilege5

Copyright Statement

© 2008 Realtime Publishers, Inc. All rights reserved. This site contains materials that have been created, developed, or commissioned by, and published with the permission of, Realtime Publishers, Inc. (the “Materials”) and this site and any such Materials are protected by international copyright and trademark laws.

THE MATERIALS ARE PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. The Materials are subject to change without notice and do not represent a commitment on the part of Realtime Publishers, Inc or its web site sponsors. In no event shall Realtime Publishers, Inc. or its web site sponsors be held liable for technical or editorial errors or omissions contained in the Materials, including without limitation, for any direct, indirect, incidental, special, exemplary or consequential damages whatsoever resulting from the use of any information contained in the Materials.

The Materials (including but not limited to the text, images, audio, and/or video) may not be copied, reproduced, republished, uploaded, posted, transmitted, or distributed in any way, in whole or in part, except that one copy may be downloaded for your personal, non-commercial use on a single computer. In connection with such use, you may not modify or obscure any copyright or other proprietary notice.

The Materials may contain trademarks, services marks and logos that are the property of third parties. You are not permitted to use these trademarks, services marks or logos without prior written consent of such third parties.

Realtime Publishers and the Realtime Publishers logo are registered in the US Patent & Trademark Office. All other product or service names are the property of their respective owners.

If you have any questions about these terms, or if you would like information about licensing materials from Realtime Publishers, please contact us via e-mail at info@realtimepublishers.com.

Limitations in Native Solutions for Privilege Management

The first two articles in this series have taken a high-level approach to privilege management and the need for a reduction in the assignment of administrative privileges. However, a technical explanation is necessary to fully understand how Least Privilege impacts the operation of the IT environment. The explanation in this article will relate to the Microsoft Windows environment and focus on existing solutions that are natively available. Each solution attempts to manifest a resolution to the problem of Least Privilege. As you'll find, each also has limitations in its efficacy and scalability as a holistic solution.

Before discussing each solution in-depth, a look at the limitations of the Windows operating system (OS) itself is first necessary. When a user logs into a computer, the user's interaction with the computer is typically comprised of system configuration activities such as changing networking information or connecting to printers as well as making use of applications that are installed to the computer. Applications are typically installed to the location *C:\Program Files* or *C:\Program Files (x86)* (in the case of 32-bit applications installed on x64 systems) with computer-specific information stored in *HKEY_LOCAL_MACHINE*. These locations are considered secure installation locations by the Windows OS. Standard users are prevented from making changes to these locations. File-based user-specific information is typically stored in the user's profile with registry-based user information stored in *HKEY_CURRENT_USER*.

This separation between areas that are considered secure and available for users ensures that application installations can be completed only by an assigned administrator. It also ensures that for the protection of the application, its core installation cannot be modified by standard users. Some applications, however, do not recognize the separation between these two areas. Others may require access to drivers and other system-level components that are typically reserved for administrative users only. When applications such as these are installed to a computer, their use effectively requires that its user have Administrator-level access.

As discussed in the first article of this series, the problem is that the common solution for providing this level of access is to add the user to the computer's Administrators group. Doing so results in every action by the user being completed with administrative privileges when the desired result is merely to make one or more "bad" applications run.

Early attempts in the Microsoft Windows environment to limit the use of Administrator privileges suggested that users log in using a non-administrative account. That account would include the privileges necessary to accomplish the "regular" tasks of the day. In the case in which a task requires the use of administrative privileges, early guidance suggested that the user log out and log back in with a separate Administrator account.

This process incurs major administrative overhead to the user. If a user regularly needs to use the "bad" application, the user will find themselves wasting time logging out and back in repeatedly. With no enforcement mechanism in place to ensure proper behavior, many users ultimately chose to log in with their Administrator privileges to the neglect of the lesser-privileged and more secure "primary" account.

Runas


An early alternative solution to resolve this problem was the use of the Runas tool. This command-line tool and shell extension allowed users to log into their computers with standard credentials. When the use of an application required administrative elevation, the user would use Runas to launch the application with alternate administrative credentials. Leveraging the use of the Runas tool along with its Windows shell extensions eliminated the need to log out and back in to enable credential elevations. Yet this solution suffered from many of the same limitations as the double-login concept:

- *Additional steps are required.* In order to launch the process under the alternate credentials, extra steps are still required. For non-technical users, these extra steps can be challenging, forcing them to remember the extra steps each and every time the alternate credentials are required to launch an application. Users must also be aware of which applications require administrative credentials and ensure that they elevate only the applications that need them.
- *The user must manage separate credentials.* There is an additional administrative overhead on the part of the user as well as IT in managing two separate sets of credentials. As examples, dual credentials include the typical processes of managing password changes and remembering passwords. IT must also manage the storage of double credentials for each user on the network, which adds a burden on their operational workflow.
- *No enforcement.* Leveraging the Runas tool is a purely client-based approach. Thus, no policy-based enforcement is present to ensure that users are correctly elevating approved applications and not using credentials for unapproved actions. The elevated credentials are also equivalent in capability to standard credentials. So there is no technical control in place to ensure that users don't simply log in to the computer with the elevated credentials.
- *Launched processes were created as a separate user.* When processes and applications are launched using the alternate set of credentials, they are launched under a different user account entirely. This alternate account method does not work with some applications and often causes problems with the correct logging of user actions.

Because of these limitations, the use of Runas as a tool for selective elevation is based on little more than the “honor system.” Although this might have been moderately effective for use by small groups of highly trusted administrators within IT, the spread of its use through the regular user population was not controllable and did not meet the requirements of a technical control as defined by many compliance regulations.


User Account Control

To combat the inadequacies of Runas, the Windows Vista and Windows Server 2008 OSs were both updated to include a new service called User Account Control (UAC). This change affects how Administrators interact with the OS. In short, when an Administrator logs into a UAC-enabled system, they initially use a set of credentials that do not have administrative privileges. When the administrator attempts to accomplish a task that requires administrative credentials, the system then switches between the initial “regular” and the alternate “elevated” credentials for the needed activity. To prevent an unwanted action or malware infection from automatically elevating a user’s credentials, any occurrence of credential switching involves graying out the desktop and prompting the user to approve the elevation.

 Due to this added level of security, administrators that operate under UAC’s controls are referred to as *Protected Administrators*.

UAC affects standard users as well. When a standard user attempts to perform a task that requires elevation, a prompt appears for them as well. This prompt, called an *Over the Shoulder (OTS) Elevation Prompt*, provides a place to enter credentials for an alternate administrative account. Entering the username and password for an Administrator into the OTS prompt allows the standard user to accomplish the task.

Yet there’s a problem with this implementation. The only way to complete an OTS elevation is by handing over Administrator privileges to a user, an action that ultimately gives them the username and password that could be used for future elevations. This implementation results in the same type of person-based credentials assignment that Least Privilege specifically desires to avoid.

 With UAC, for a standard user to accomplish a task or run an application that requires elevated privileges, the user must have access to Administrator account credentials. With these credentials, a standard user has what amounts to full control over the entire computer and is no longer operating in a Least Privilege environment.

The implementation of UAC resolves one of the biggest problems with Runas: Namely, the requirement for an Administrator to manually identify and elevate when administrative privileges are necessary. The UAC service in many—though not all—cases automatically recognizes when privilege elevation is required. UAC also resolves some of the issues associated with Runas' use of completely separated credentials for standard and administrative access. When an Administrator elevates their credentials, they are effectively using the same user account. However, UAC still suffers from significant limitations that make it challenging to use in an enterprise environment:

- *Complex.* Much of UAC's functionality lies "under the covers" of the desktop itself, making it difficult to truly understand. Although UAC in consumer environments works well for protecting individual users and their computers, understanding how to best use it in a managed, corporate environment requires specialized skills.
- *Annoying prompts.* Without a core understanding of UAC's protections, non-technical users and IT generalists often see UAC and its prompting as an intrusion into their daily workflow. With many actions requiring administrative access, especially considering the reconfiguration of the Windows Vista and Windows Server 2008 file system, UAC prompts are seen relatively frequently. The ever-present barrage of elevation requests slows the ability of the privileged user to get their job done.
- *Responsibility for elevation remains with the user.* Because the decision whether to accept or reject an elevation request remains with the individual user, that user retains the ability to make the wrong decision. When users are not appropriately trained on how to use UAC and when to approve or reject elevation requests, they may out of frustration always choose to accept. This behavior ultimately reduces UAC's effectiveness in protecting the computer against malicious code.
- *Little or no centralized mechanisms for control, management, or logging.* UAC is a purely client-based solution that includes no mechanism for the centralized logging of elevation requests. It also includes only a coarse capability for centralized management and tuning, having only ten configurations that can be controlled via local or Group Policy.
- *Person-based credentials assignment.* Last, and possibly most important, UAC's implementation retains the person-based credentials assignment architecture discussed in the first article of this series. Administrative credentials are assigned to an individual and, once assigned, grant the individual what amounts to full control over the entire computer. Credentials with UAC cannot be granularly assigned to enable Administrator access to a specified activity or application. This limits its utility as a solution for enabling a Least Privilege environment.

Capabilities to Resolve Least Privilege

To this point, this series has discussed the definition of Least Privilege, the benefits it portends to provide for the IT environment, and how native solutions have yet to truly align with its goals. In order to achieve an environment that supports Least Privilege, there are a set of desired capabilities that are necessary. These capabilities at present are not available through the native Windows OS but in many cases are available through third-party add-on products. Although not a comprehensive list, consider the following needed capabilities when looking for solutions that enable Least Privilege:

- *Granular privilege assignment.* Considering the three separate elements that must aggregate to identify the individual, the element, and the period of use, solutions that support a more granular level of control are necessary. Solutions that support this granularity will be able to elevate administrator-assigned actions and applications based on a user and his or her role. Individual users and applications should have the ability to be enabled or disabled as needed by IT administrators. This granular level of control eliminates the need to assign entire-computer privileges for the resolution of point issues.
- *Centralized control and policy-based assignment.* With the move to granular control of elevation comes the need for centralized control over which applications, actions, and users can and should be elevated. Centralized control also includes centralized logging support that fulfills security and compliance needs. Policy-based assignment ensures that configured elevation settings at each client computer are enforced based on the policy applied to that computer. It also enables entire classes of users, computers, and applications to be managed as single units for enhanced management.
- *Movement of elevation responsibility from individual users to IT.* A chosen solution should remove the onus of responsibility for elevation completely from individual users. This movement of responsibility ensures that educated and trained IT personnel are making decisions about elevations based on corporate policy and security experience. It also eliminates the possibility that an uninformed user will automatically choose to elevate when any prompt appears.
- *Right-sizing of notifications.* Although UAC's excess of prompting has led many administrators to desire a complete elimination of prompting, there may be situations where users require a minimal level of notification. This level of notification may be unobtrusive unless necessary for the protection of the OS.

Third-party applications are available today that augment the native OS as well as UAC by enabling elevations while protecting the OS from attack. This additional software assists IT with maintaining a secured and controlled desktop that supports the needs of the user while providing the protections required by IT.