Jackson Warren

Jeremy Bergen

CSCI 370

9/17/21

Week 3 Assignment: Comparing Linux and Windows ACLs

While both Windows and Linux use ACLs (Access Control List) to support their security, there are some major differences between the two operating systems. Linux has ACLs in use at most levels, except when under use by admin/enterprise settings. Then, Linux uses UNIX permissions, deemed “less-powerful” (Puryear). When at the process-level, Linux has to use “sudo” commands to access many items at the admin level, which differs from Windows applications with users who commonly assume their permissions are at/near admin level to begin. However, Windows has come up with UAC (User Access Control) that serves a similar purpose as sudo, limiting the power of Windows permissions to resemble Linux permissions. The main differences between permissions rights between the two is that Windows uses file-level permissions, while Linux uses an owner-group-world permissions system.

Advantages are on both sides, but Linux tends to be more secure. This is due to the fact that Linux’s way of permissions tends to ere on the side of least permissions needed per file. This basically means that unless a file is specified to have more permissions, the file will have the absolute minimum permissions needed to operate the file on a basic level for certain users. Still, while this is a secure way of thinking, this can create issues with not being able to have the permissions you would like to operate your system. Windows has the opposite problem. Windows systems gives more permissions and have been trying to fight programs from accessing permissions that are too high.

Windows permissions and ACLs are dealt with in several ways. The first way this can be done is through simple inheritance, where child ACL resembles parents. The second way is through unique permissions, where the child has the same permissions as the parent’s, but the child has more/less permissions applied directly. This can be done by “icacls file1 /grant User1:(d,wdac)”, giving a specific user (User1) permissions for “file1”. The third way is to use protected permissions, where the changes from the parent and child are unrelated.

Files in Linux do not inherit the parent file’s permissions. Linux will create a file with permissions of its creators’ permissions. Changing permissions on Linux is somewhat simple, given you have the correct permissions to do so. Users can simply input “chmod 640 file1” to their terminal, changing permissions to said “file1”. Keep in mind, sudo permissions must be used to change a file’s permissions that are above a level that that user/group could grant. Files can also be commanded to inherit parents permissions/ACLs using the “file\_inherit” command, but will not do this normally.

From my knowledge and research, it appears that Linux and Windows both offer execution flags. I’m finding very scattered info about Windows have a buried flag that aren’t as prominent as Linux – but I can’t find much on it. Either way, both systems will send out an error message if a user tries to access something outside of their permissions. Windows will usually offer the ability to input a password the bypass this, while Linux will not, making users have to go in and change the permissions manually. These messages are very important when it comes to computer security because users must not be able to access things outside of their respective domains. In other words, rejecting users reaching outside their permissions should be flagged and not allowed to pass. In addition, being notified is also useful so that users that SHOULD have those permissions are now aware they need to be given higher permissions, and can proceed accordingly.

Macs use Linux and UNIX, and controlling permissions is the same tactic as mentioned earlier for Linux. They use the owner-group-world permissions ideology. However, they also use some of the Windows inheritance features mentioned earlier. This is very different from other operating systems, considering it is one of the few systems to have an odd mixture of these tactics.

References:

J. (2018, August 21). *icacls*. Microsoft Docs. https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/icacls

M. (2010, May 18). *Understanding Linux File Permissions*. Linux.Com. https://www.linux.com/training-tutorials/understanding-linux-file-permissions/

Petters, J. (2021, April 24). *Permission Propagation: Tips and Best Practices - Varonis*. Inside Out Security. https://www.varonis.com/blog/permission-propagation/

Puryear, D. (2010, June 8). *Comparing Access Control in Windows and Linux*. Network World. https://www.networkworld.com/article/2230977/comparing-access-control-in-windows-and-linux.html