The Windows Security Journey

Version 6.0
January-2025

By Dr. Shlomi Boutnaru

Table of Contents

Introduction

When starting to learn OS security I believe that there is a need for understanding multiple technologies and concepts. Because of that I have decided to write a series of short writeups aimed at providing the security vocabulary.

Overall, I wanted to create something that will improve the overall knowledge of Windows' different security mechanisms included with Windows in writeups that can be read in 1-3 mins. I hope you are going to enjoy the ride.

Lastly, you can follow me on twitter - @boutnaru (https://twitter.com/boutnaru). Also, you can read my other writeups on medium - https://medium.com/@boutnaru. Lastly, You can find my free eBooks at https://TheLearningJourneyEbooks.com.

Lets GO!!!!!!

SID (Security Identifier)

The goal of an SID is to uniquely identify a security principal/group. When talking about a security principal we mean any entity that can authenticate to the operating system like: user/computer account or a thread/process that runs with the security context of one of those. Every time a user is logged on the system creates an access token for that user (more on that in a future writeup). This access token holds the user's SID, privileges and the SIDs for any groups that the user is part of[1].

Moreover, there is a specific format for an SID. We can split it into three main parts: revision, identifier authority and sub authorities. Revision, which specifies the version of the SID structure. Identifier authority, which specifies the highest level of authority that can issue an SID for a security principal. Sub authorities, which hold the most important information (can identify a local computer/domain) and its last part is an RID (relative identifier) that identifies a specific user/group in a local computer/domain - as shown in the diagram below[2].

An example of some well-known SIDs are: "S-1-1-0" (group that includes all users), "S-1-0-0" (aka NULL SID, a group with no members). They are called "Universal well-known SIDs"[3]. Also, there are well-known RIDs such as: 500 (Administrator), 501 (Guest). Since Windows 2008/Vista most of the system files are owned by the "TrustedInstaller" SID, in order to prevent a process running with Administrator/Local System permissions to overwrite the OS files[4].

Lastly, there are also "Capability SIDs" which grant access to specific resources (like cameras, documents, location and more). Those type of SIDs that the system is aware of are stored in the registry value "AllCachedCapabilities" under "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\SecurityManager\CapabilityClasses"[5].

---
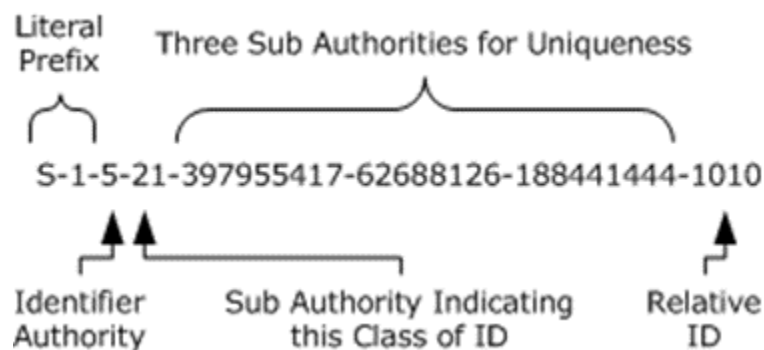
[1] https://learn.microsoft.com/en-us/windows-server/identity/ad-ds/manage/understand-security-identifiers
[2] https://learn.microsoft.com/en-us/openspecs/windows_protocols/ms-azod/ecc7dfba-77e1-4e03-ab99-114b349c7164
[3] https://learn.microsoft.com/en-us/windows/win32/secauthz/well-known-sids
[4] https://learn.microsoft.com/en-us/windows-server/identity/ad-ds/manage/understand-security-identifiers
[5] https://renenyffenegger.ch/notes/Windows/security/SID/index

SD (Security Descriptor)

The goal of a security descriptor (SD) is to hold the security information that is related with a specific securable object. Examples for securable objects are: file, folder, network share, printer, registry key, synchronization object, active directory objects and more. The structure which describes a SD is defined in "winnt.h" and is named "SECURITY_DESCRIPTOR"[6].

Overall, every object created by the "Object Manager" in Windows has a SD. Each objects has an header with different fields (like object name, reference count, object type and more) one of them is the security descriptor[7]. You can see an illustration of that in the diagram below[8].

Moreover, we can think about an SD as containing four main fields: an owner, group, DACL (Discretionary Access Control List) and SACL (System Access Control List) . DACL is used for allowing/denying permissions which SACL is used for auditing[9]. The description of each entity in the structure is stored in the form of an SID (Security Identifier). More on those in future writeups.

Lastly, the "SECURITY_DESCRIPTOR"   is a compact binary representation of the security associated with a specific object. Because it is not convenient to use it, there is a text-based form for representing it. This format is called SSDL (Security Descriptor Description Language). It has specific text tokens in order to describe: access rights, user accounts, user-mode drivers and more[10].

---
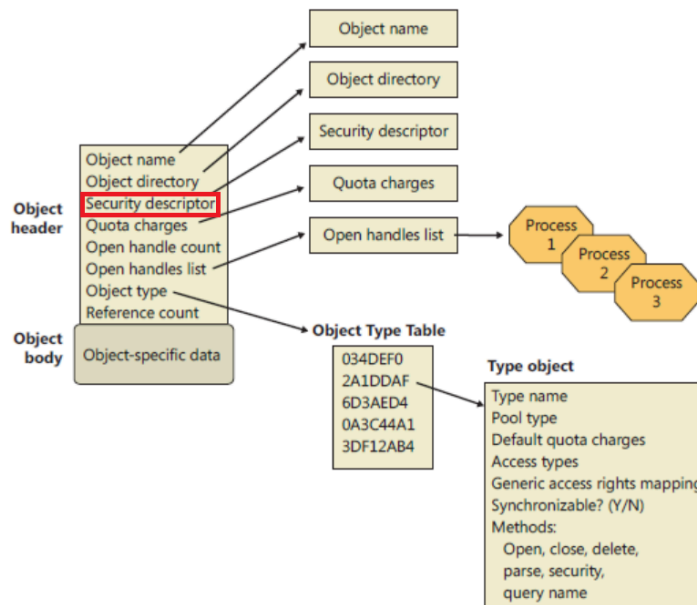
[6] https://learn.microsoft.com/en-us/windows/win32/api/winnt/ns-winnt-security_descriptor
[7] https://www.geoffchappell.com/studies/windows/km/ntoskrnl/inc/ntos/ob/object_header/index.htm
[8] https://www.tophertimzen.com/resources/cs407/slides/week02_01-KernelObjects.html#slide16
[9] https://learn.microsoft.com/en-us/openspecs/windows_protocols/ms-azod/ec52bde3-9c86-4484-9080-e72148a2d53b
[10] https://learn.microsoft.com/en-us/windows-hardware/drivers/kernel/sddl-for-device-objects?redirectedfrom=MSDN

Securable Objects

Overall, "securable objects" are Windows objects[11] that can have a "security descriptor"[12]. All named Windows objects are securable. There are also unnamed objects which are securable like processes and threads[13] - as shown in the screenshot below.

Moreover, each securable object has specific elements that are elaborated next. An owner's (user/group) SID[14]. A DCAL (Discretionary Access Control List) which contains a list of group/user SIDs and the access rights each of them has - as shown in the screenshot below in the permissions tab. A SACL (System Access Control List) which states what logging/auditing should be done when accessing the object. Also, there is a group associated with the object which is for POSIX compatibility only[15].

Lastly, examples of securable objects (but not limited to) are: files, directories, desktops, processes, threads, named pipes, mailslots, network shares, printers, private objects, events, semaphores, WMI namespaces and waitable timers and windows stations[16]..

---
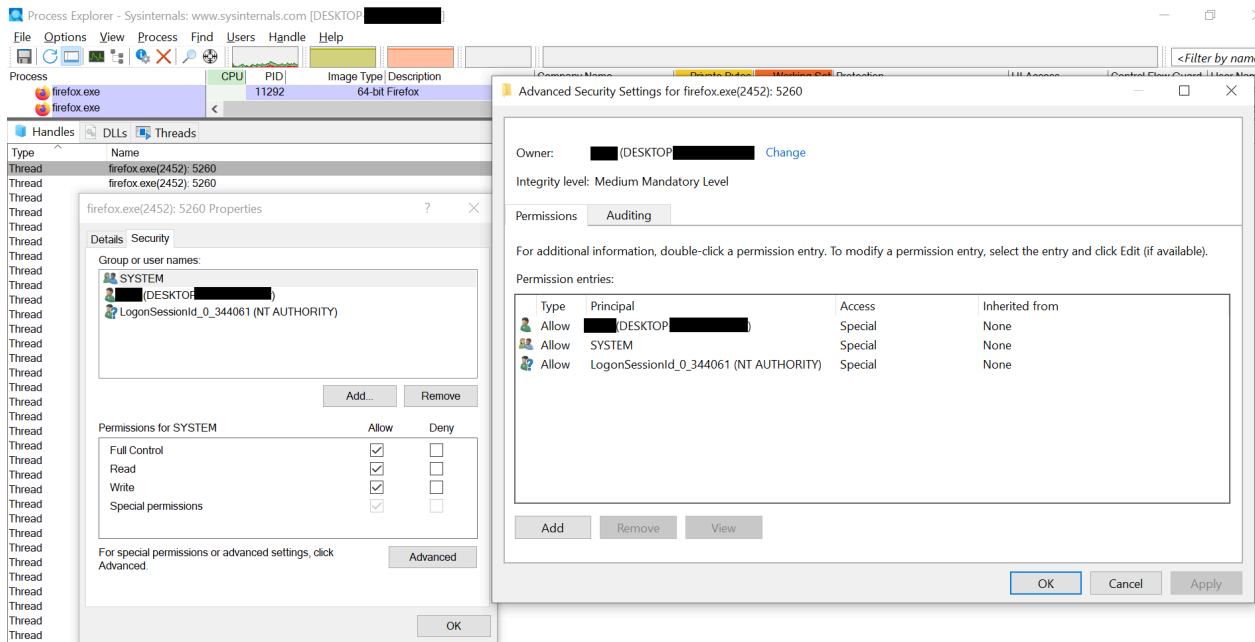
[11] https://medium.com/@boutnaru/windows-objects-2c289da600bf
[12] https://medium.com/@boutnaru/windows-security-security-descriptor-sd-ba95b8fa048a
[13] https://learn.microsoft.com/en-us/windows/win32/secauthz/securable-objects
[14] https://medium.com/@boutnaru/windows-security-sid-security-identifier-d5a27567d4e5
[15] https://renenyffenegger.ch/notes/Windows/development/objects/securable/index
[16] http://winapi.freetechsecrets.com/win32/WIN32Securable_Objects.htm

Privileges

Privileges are rights given for a specific account (user/group) which allows performing different system related operations on the local computer. Think about: changing the system time, loading a device driver, shutting down the system and more. There is a difference between access rights to privileges[17].

Thus, we can say that privileges control the access to system resources/system related tasks while access rights control access to securable objects (such as files, directories, registry keys and more). We assign privileges to user/group accounts whereas access rights are granted as part of DACLs.

Moreover, the operating system represents a privilege in a category of "User Rights Assignments". We can modify them using the "Local Group Policy" (or the "Group Policy") MMC snap-in[18] - as shown in the screenshot below.

Lastly, the privileges are defined using constants in the following pattern "SE_[DESCRIPTION]_NAME" and also has a text format which is in the pattern of "Se[DESCRIPTION]Privilege". A couple of examples are: "SE_CREATE_PAGEFILE_NAME"\"SeCreatePagefilePrivilege" which enables creating a new pagefile, "SE_DEBUG_NAME"\"SeDebugPrivilege" which is required for debugging/adjusting the memory of a processes owned by a different user account and "SE_LOAD_DRIVER_NAME"\"SeLoadDriverPrivilege" which is required to load/unload a device driver (it is also the one marked in the screenshot below).

---

[17] https://learn.microsoft.com/en-us/windows/win32/secauthz/privileges
[18] https://learn.microsoft.com/en-us/windows/win32/secauthz/privilege-constants
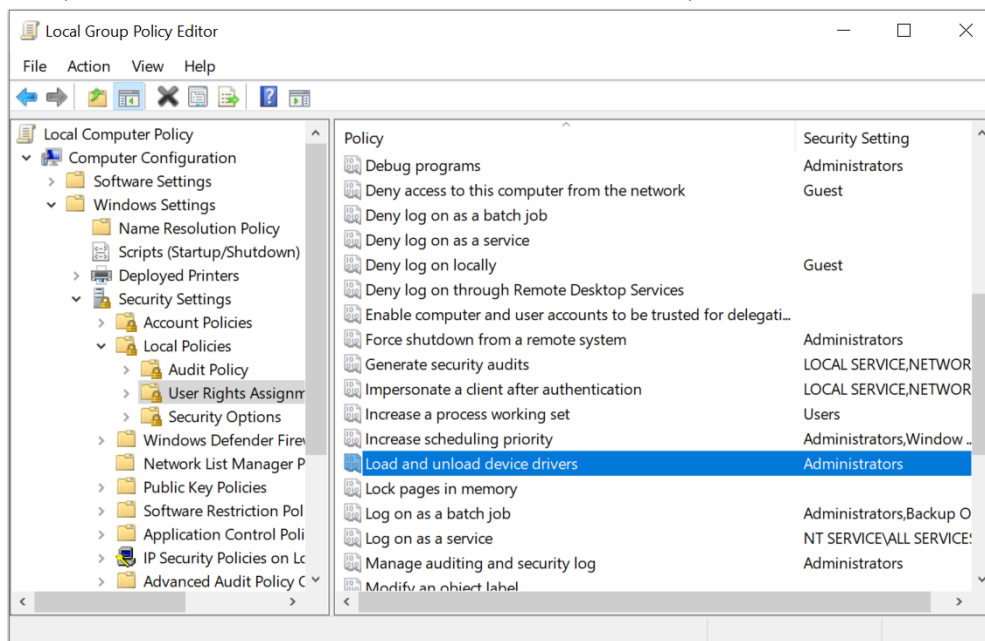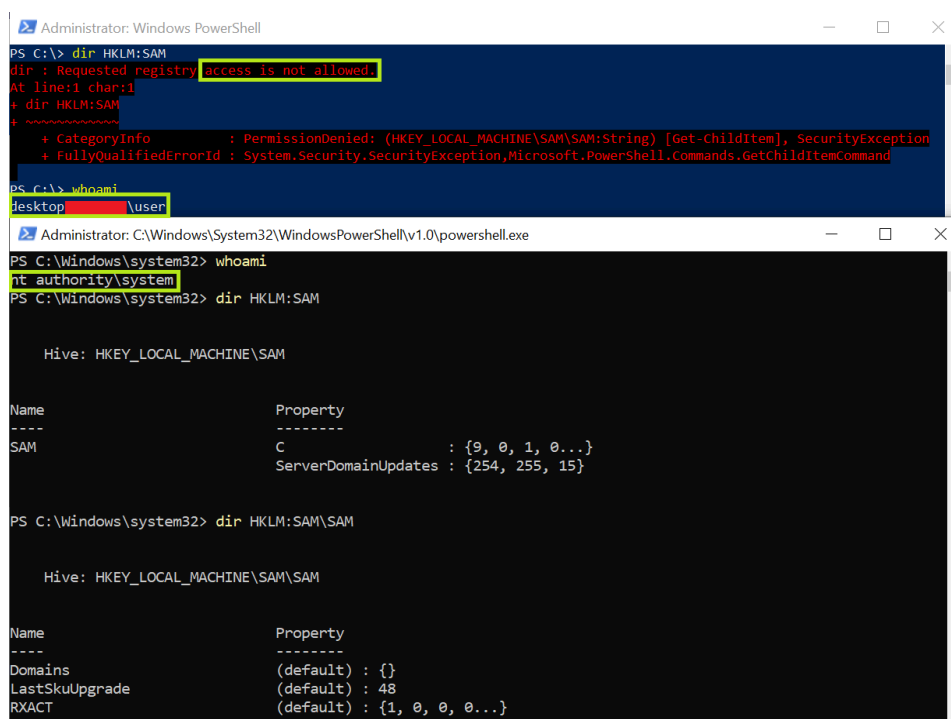
SAM (Security Account Manager)

SRM (Security Account Manager) is the DB in Windows that stores the user names/passwords of the local user defined on the system. By configuring SAM we allow users to authenticate to the local system[19].

Moreover, the SAM file is located at "%windir%\System32\config\SAM" which is mounted in the registry in the following "HKEY_LOCAL_MACHINE\SAM"[20]. In order to view its content we need to run as SYSTEM and Local Administrator is not enough - as shown in the screenshot below.

Thus, different hashes can be stored in SAM like LM hash and NTLM hash (more on those and others in future writeups). We can think about SAM as the equivalent of "/etc/passwd", "/etc/shadow" and "/etc/group" files under Linux.

Because Microsoft wanted to increase the security around the hashes stored in SAM they have created SYSKEY. It uses a system key for encrypting to protect the account password information stored in the SAM. The system key can be saved locally, on a floppy disk or stored locally but protected by an admin password[21]. Lastly, SYSKEY support was removed from Windows 10 version 1709[22].

[19] https://www.calcomsoftware.com/what-is-windows-security-accounts-manager/
[20] https://viperone.gitbook.io/pentest-everything/everything/everything-active-directory/credential-access/credential-dumping/security-account-manager-sam
[21] https://learn.microsoft.com/en-us/windows-server/security/kerberos/system-key-utility-technical-overview
[22] https://learn.microsoft.com/en-us/troubleshoot/windows-server/identity/syskey-exe-utility-is-no-longer-supported

Access Token

"Access Token" is an object which represents the access rights/privileges/identity for a specific process/thread. The operating system uses the access token in order to identify the user when a specific thread interacts with a securable object[23] or when it tries to perform a system task (that requires some kind of privilege)[24].

Thus, if a user authenticates to a system, the Local Security Authority (LSA) creates an access token (to be accurate it is the primary access token, as described in more detail later). It contains the SID of the user, the SIDs of all the groups the user belongs to, a list of privileges, the SID of the owner (user/group), the primary group (for POSIX subsystems), default DACL, source (process that caused the token to be created - RPC/LAN Manager/Session Manager/etc), type (primary/impersonation), impersonation level, restricting SIDs, Terminal service session ID (if relevant), session reference, SandBox inert, audit policy and origin - as shown below[25].

Moreover, using different Win32 API functions we can read/manipulate access tokens. As example we can use "OpenProcessToken"[26] or "OpenThreadToken"[27] to get a handle to the access token of the process/thread. Also, we can use "DuplicateTokenEx"[28] for duplicating the access token of the current process and "CreateProcessWithTokenW"[29] which allows creation of a process with a specified token. The access token is stored in kernel mode using "struct _TOKEN"[30].

User
Group 1 SID
Group n SID
Privilege 1
Privilege n
Default Owner
Primary Group
Default Discretionary Access Control List (DACL)
Source
Type
Impersonation Level
Statistics
Restricting SID 1
Restricting SID n
TS Session ID
Session Reference
SandBox Inert
Audit Policy
Origin

---
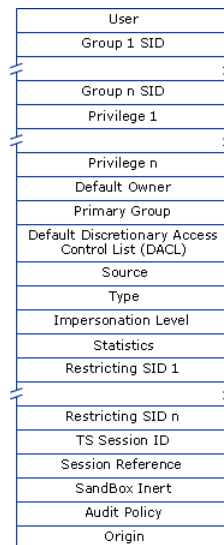
[23] https://medium.com/@boutnaru/windows-securable-objects-311a9d6c83ad
[24] https://learn.microsoft.com/en-us/windows/win32/secauthz/access-tokens
[25] https://learn.microsoft.com/pt-pt/previous-versions/windows/server/cc783557(v=ws.10)
[26] https://learn.microsoft.com/en-us/windows/win32/api/processthreadsapi/nf-processthreadsapi-openprocesstoken
[27] https://learn.microsoft.com/en-us/windows/win32/api/processthreadsapi/nf-processthreadsapi-openthreadtoken
[28] https://learn.microsoft.com/en-us/windows/win32/api/securitybaseapi/nf-securitybaseapi-duplicatetokenex
[29] https://learn.microsoft.com/en-us/windows/win32/api/winbase/nf-winbase-createprocesswithtokenw
[30]https://www.ired.team/miscellaneous-reversing-forensics/windows-kernel-internals/how-kernel-exploits-abuse-tokens-for-privilege-escalation

Primary Access Token

Overall, there are two types of access tokens[31] - as stated in the type field of the access token. Those are "Primary Token" and "Impersonation Token". In this writeup I am going to focus on the first one.

A primary token can only be associated with a process. Processes inherit a copy of the parent's process primary token[32]. Due to that, when a thread is attempting to access a securable object[33] by default this token is checked (threads can have an impersonation token but that is for a different writeup). Also, this token belongs to the user account that created the process[34].

Thus, every process has a primary token that it gets from its parent process - as shown in the diagram below[35]. Because primary tokens are associated with processes they are also known as "Process Tokens". We can also state that they are created while authenticating interactively[36].

[31] https://medium.com/@boutnaru/windows-security-access-token-81cd00000c64
[32] https://book.hacktricks.xyz/windows-hardening/windows-local-privilege-escalation/access-tokens
[33] https://medium.com/@boutnaru/windows-securable-objects-311a9d6c83ad
[34] https://jsecurity101.medium.com/better-know-a-data-source-access-tokens-and-why-theyre-hard-to-get-7bc951eae0b9
[35] https://i.blackhat.com/USA-20/Thursday/us-20-Burgess-Detecting-Access-Token-Manipulation.pdf
[36] https://sensepost.com/blog/2022/abusing-windows-tokens-to-compromise-active-directory-without-touching-lsass/

Impersonation Access Token

Overall, there are two types of access tokens[37] - as stated in the type field of the access token. Those are "Primary Token" and "Impersonation Token". In this writeup I am going to focus on the second one.

Basically, impersonation is a mechanism which allows server processes to run by using the credentials of some client. Meaning using creditails of another user than its primary token[38].

Thus, when can that impersonation allow a thread to switch to a different security context. By default, threads inherit the same security context as the primary token[39] of the process[40].

One of the main use-cases for impersonation is asking the a server to execute code on behalf of the user performing a network authentication - as shown in the diagram below[41]. It can also be used for cases where we want an application/process to have a thread running code with a different security context (than the other threads). However, we need to be careful because all the threads share the same memory space so one thread can hijack the execution flow of another.

[37] https://medium.com/@boutnaru/windows-security-access-token-81cd00000c64
[38] https://medium.com/@boutnaru/windows-security-primary-access-token-e295a35796a9
[39] https://medium.com/@boutnaru/windows-security-primary-access-token-e295a35796a9
[40] https://i.blackhat.com/USA-20/Thursday/us-20-Burgess-Detecting-Access-Token-Manipulation.pdf
[41] https://i.blackhat.com/USA-20/Thursday/us-20-Burgess-Detecting-Access-Token-Manipulation.pdf

SRM (Security Reference Monitor)

SRM (Security Reference Monitor) is a component that is part of the Windows executive (stored in %systemroot%\System32\ntoskrnl.exe). SRM is responsible for implementing the authorization system (together with LSA as shown in the diagram below). Also, SRM implements the access check algorithm[42].. This means it checks the access to different resources by getting the access token[43] of the subject and comparing it to the ACEs (Access Control Lists) in the security descriptor of the securable object[44].

Moreover, the routines that provide a direct interface with the SRM are those prefixed with "Se"[45]. An example of such function is: "SeAccessCheck" which determines if the requested access to an object can be granted[46]. If we want we can go over a reference implementation of "SeAccessCheck" as part of ReacOS[47].

Lastly, we can say that the "Object Manager" uses SRM to check if a specific process/thread has the proper rights to execute a certain action on an object. Also, it is part of the flow when implementing auditing functionality when objects are being accessed[48].
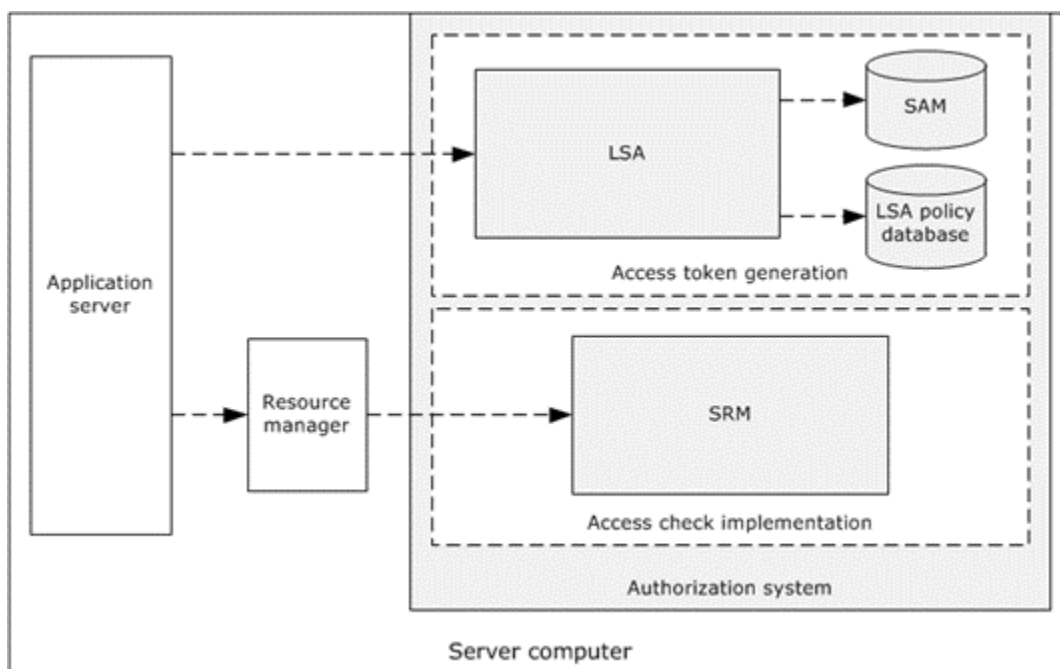
---

[42] https://learn.microsoft.com/en-us/openspecs/windows_protocols/ms-azod/d28d536d-3973-4c8d-b2c9-989e3a8ba3c5
[43] https://medium.com/@boutnaru/windows-security-access-token-81cd00000c64
[44] https://medium.com/@boutnaru/windows-securable-objects-311a9d6c83ad
[45] https://learn.microsoft.com/en-us/windows-hardware/drivers/kernel/windows-kernel-mode-security-reference-monitor
[46] https://learn.microsoft.com/en-us/windows-hardware/drivers/ddi/wdm/nf-wdm-seaccesscheck
[47] https://github.com/reactos/reactos/blob/master/ntoskrnl/se/accesschk.c#L1966
[48] https://cs.gmu.edu/~menasce/osbook/nt/sld034.html

Job Object

By using a job object we can manage a group of processes as one unit. Thus, an operation performed on a job object affects all the processes which are part of that job[49].

Moreover, in order to create a job object we can use the Win32 API call "CreateJobObjectA"[50]. When creating a job it has no processes associated with it, so we need to use the function "AssignProcessToJobObject"[51]. Until Windows 8/Windows Server 2012 a process could be associated with one job only. By the way, for getting an handle for an existing object we can use the "OpenJobObjectA" function[52].

Overall, by using jobs we can limit the usage of system resources by processes like: process priority, time limit, working set, number of child processes, desktop creation, writing data to the clipboard and more. For setting the limits we use the function "SetInformationJobObject"[53].

Lastly, job objects are being used in sandboxes like in web browsers (shown in the screenshot below) and are one of the building blocks of "Windows Containers". There are also named/unanmed, securable objects[54] and shareable objects - as shown in the screenshot below taken from Sysinternals' "Process Explorer"[55].

---

[49] https://learn.microsoft.com/en-us/windows/win32/procthread/job-objects
[50] https://learn.microsoft.com/en-us/windows/win32/api/winbase/nf-winbase-createjobobjecta
[51] https://learn.microsoft.com/en-us/windows/win32/api/jobapi2/nf-jobapi2-assignprocesstojobobject
[52] https://learn.microsoft.com/en-us/windows/win32/api/winbase/nf-winbase-openjobobjecta
[53] https://learn.microsoft.com/en-us/windows/win32/api/jobapi2/nf-jobapi2-setinformationjobobject
[54] https://medium.com/@boutnaru/windows-securable-objects-311a9d6c83ad
[55] https://learn.microsoft.com/en-us/sysinternals/downloads/sysinternals-suite

ACL (Access Control List)

ACL (Access Control List) is a list of ACEs (Access Control Entries). Every ACE identifies a trustee (user account/group/logon session) and the relevant allowed/denied/audited access for that trustee[56].

Overall, there are two types of ACLs which are in use in Windows systems: DACL aka as "Discretionary Access Control List" and SACL aka "System Access Control List"[57]. More information about those types in future writeups. Those types of ACLs are part of the security information stored as part of the "Security Descriptor"[58] related to securable objects[59] - as shown in the diagram below[60].

Moreover, every ACE has four main components. The first, the SID[61] to whom the access information in this ACE is relevant for. Second, a flag denoting the type of ACE (deny/allow/audit). Third, flags regarding the inheritance of the specific ACE. Forth, an access mask which is a 32 bit that describes the rights relevant for this ACE[62].

Lastly, due to the fact we have DACL and SACL, usually when saying ACE we talk about the first one and when saying System ACE we mean the second one.

---

[56] https://learn.microsoft.com/en-us/windows/win32/secauthz/access-control-lists
[57] https://www.securew2.com/blog/windows-access-control-acl-dacl-sacl-ace
[58] https://medium.com/@boutnaru/windows-security-security-descriptor-sd-ba95b8fa048a
[59] https://medium.com/@boutnaru/windows-securable-objects-311a9d6c83ad
[60] https://developer.aliyun.com/article/747446
[61] https://medium.com/@boutnaru/windows-security-sid-security-identifier-d5a27567d4e5
[62] https://book.hacktricks.xyz/windows-hardening/windows-local-privilege-escalation/acls-dacls-sacls-aces

DACL (Discretionary Access Control List)

In general DACL (Discretionary Access Control List) is an ACL[63] which identifies the trustees that allowed/denied access to a securable object[64].Thus, if the securable object does not have any DACL (Null) the SRM[65] allows everyone full access to it. If the list of ACL is empty no one has any access to the object[66].

Moreover, when a thread tries to access a securable object, the system goes over the ACEs in the DACL until it finds one that allows/denies the access (think about it like firewall rules). The predefined order of ACEs are as follows: all explicit ACEs are before inherited ACEs and the inherited ones are placed in the order in which they are inherited. By the way, in every level access denied ACEs are placed before the access allowed ACEs ones[67].

Lastly, for configuring a DACL using the UI we just go to the properties of the object and select the "security tab", there we can edit the DACL of that specific object - as shown in the screenshot below. We can also use CLI tools like cacls.exe/icacls.exe (but that is for a different writeup).

---

[63] https://medium.com/@boutnaru/the-windows-security-journey-acl-access-control-list-b7d9a6fe4282
[64] https://medium.com/@boutnaru/windows-securable-objects-311a9d6c83ad
[65] https://medium.com/@boutnaru/windows-security-srm-security-reference-monitor-d715f96d9fd6
[66] https://learn.microsoft.com/en-us/windows/win32/secauthz/dacls-and-aces
[67] https://www.tenouk.com/ModuleH2.html
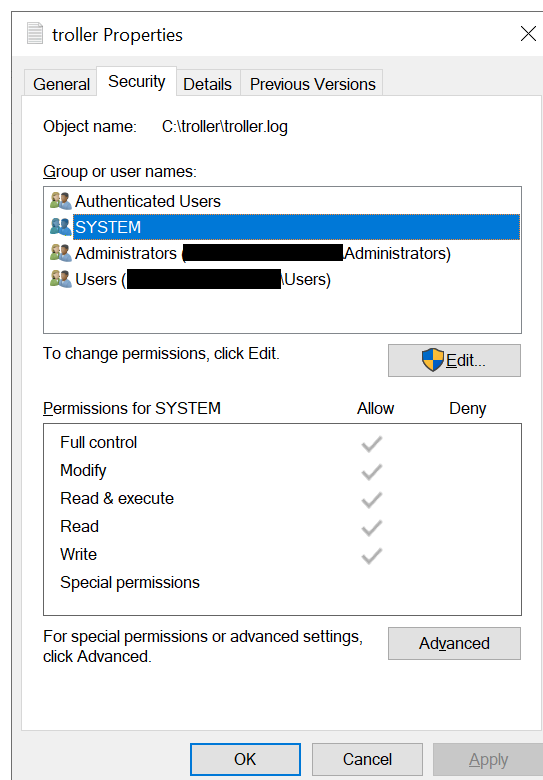
SACL (System Access Control List)

Overall, a SACL (System Access Control List) is an ACL[68] which enables the administrators of a system to audit attempts of accessing securable objects[69]. Every ACE (Access Control Entry) defines the type of access attempt that causes to generate an audit trail while performed by a trustee[70].

Thus, an ACE as part of an SACL can emit an audit record when an access attempt is failed/succeeds/both. The system writes audit messages to the security event log[71]. In order to read/write object's SACL the relevant thread/process should enable as part of its access token[72] the "SE_SECURITY_NAME" privilege[73].

Moreover, the "SE_SECURITY_NAME" privilege is defined as managing auditing and the security log[74]. We can use "SetNamedSecurityInfoA"/"SetNamedSecurityInfoW"[75] or "GetNamedSecurityInfoA"/"GetNamedSecurityInfoW" in order to access the SACL. Those functions enable the "SE_SECURITY_NAME" privilege.

Lastly, in order to configure an SACL on a securable object like a file/directory we go to its properties and then we go to the "security tab". In the "security tab" we need to press the "Advanced" button - as shown in the screenshot below. In the advanced security setting we can go to the "auditing tab" - also shown in the screenshot below.

[68] https://medium.com/@boutnaru/the-windows-security-journey-acl-access-control-list-b7d9a6fe4282
[69] https://medium.com/@boutnaru/windows-securable-objects-311a9d6c83ad
[70] https://learn.microsoft.com/en-us/windows/win32/secauthz/access-control-lists
[71] https://learn.microsoft.com/en-us/windows/win32/secauthz/audit-generation
[72] https://medium.com/@boutnaru/windows-security-access-token-81cd00000c64
[73] https://medium.com/@boutnaru/windows-security-privileges-b8fe18cf3d5a
[74] https://jeffpar.github.io/kbarchive/kb/188/Q188855/
[75] https://learn.microsoft.com/en-us/windows/win32/api/aclapi/nf-aclapi-setnamedsecurityinfow

Mandatory Integrity Control (MIC)

In general, "Mandatory Integrity Control" (MIC) has been added to Windows from Vista for adding support of MAC (Mandatory Access Control) to running processes[76]. This is done using a new attribute called "Integrity Level" (IL). MIC is designed to control access to securable objects[77]. The mechanism works in conjunction with DACL[78]. It is important to know that MIC evaluates access before the access check is made versus the object's DACL, and itself is implemented as ACEs (Access Control Entries) using special SIDs[79].

Moreover, each security principal[80] and any securable object is marked with an integrity level which is aimed at determining their level of access/protection. In Windows we have different integrity levels: "untrusted" (S-1-16-0), "low" (S-1-16-4096), "medium" (S-1-16-8192), "high" (S-1-16-12288) and "system" (S-1-16-16384). By default, standard users are given an integrity level of "medium" while elevated users get "high". Also, objects which lack an integrity level are treated as "medium"[81].

Lastly, the integrity level SIDs (as shown above) are stored in the SACL[82] of the secure object[83]. The Windows security policy states that a process can't interact with another process that has a higher integrity level[84], due to that it is also used by different sandbox implementations (like with Web Browsers). By the way, the integrity level is stored in the access token[85] of a process/thread - as shown in the screenshot below.

[76] https://learn.microsoft.com/en-us/windows/win32/secauthz/mandatory-integrity-control
[77] https://medium.com/@boutnaru/windows-securable-objects-311a9d6c83ad
[78] https://medium.com/@boutnaru/the-windows-security-journey-dacl-discretionary-access-control-list-c74545e472ec
[79] https://medium.com/@boutnaru/windows-security-sid-security-identifier-d5a27567d4e5
[80] https://medium.com/@boutnaru/windows-security-sid-security-identifier-d5a27567d4e5
[81] https://book.hacktricks.xyz/windows-hardening/windows-local-privilege-escalation/integrity-levels
[82] https://medium.com/@boutnaru/the-windows-security-journey-sacl-system-access-control-list-32488dcc80d7
[83] https://learn.microsoft.com/en-us/windows/win32/secauthz/mandatory-integrity-control
[84] https://en.wikipedia.org/wiki/Mandatory_Integrity_Control
[85] https://medium.com/@boutnaru/windows-security-access-token-81cd00000c64

UAC (User Account Control)

The goal of UAC (User Account Control) is to reduce the risk of malware by limiting the ability of malicious code from running with administrator permissions. When UAC is used an application the requests an access token[86] with administrator permissions must prompt the user for consent[87] - as shown in the screenshot below.

UAC (User Account Control) provides MAC (Mandatory Access Control) which was introduced as part of Windows Vista/Server 2008. Together with UIPI[88] UAC is used to isolate between applications with the same user on the same session. When a user tries to perform an operation that requires admin access it will trigger UAC (if it's enabled). Examples of such operations (but not limited to) are: executing an application as an administrator, changing system-wide settings, installing a device driver, changing UAC settings, configuring windows update, opening the registry editor, changing power setting and turning on/of Windows features[89].

Moreover, when UAC is enabled when an administrator logs on to a system two separate access tokens are created (standard access token and administrator access token). The difference between them is that the administrative privileges and SIDs are removed from the standard one[90].

Lastly, UAC is composed of several technologies in order to provide its capabilities, among them are: file and registry virtualization, same desktop elevation, filtered token, UIPI, protected internet explorer and installer detection[91].

---

[86] https://medium.com/@boutnaru/windows-security-access-token-81cd00000c64
[87] https://medium.com/@boutnaru/the-windows-process-journey-consent-exe-consent-ui-for-administrative-applications-d8e6976e8e40
[88] https://medium.com/@boutnaru/windows-security-user-interface-privilege-isolation-uipi-db790ad173eb
[89] https://en.wikipedia.org/wiki/User_Account_Control
[90] https://learn.microsoft.com/en-us/windows/security/application-security/application-control/user-account-control/how-it-works
[91] https://learn.microsoft.com/en-us/troubleshoot/windows-server/windows-security/disable-user-account-control

User Interface Privilege Isolation (UIPI)

User Interface Privilege Isolation (UIPI) was introduced in Windows 2008/Vista with the goal of mitigating "Shatter Attacks". Those types of attacks leverage the Windows's message passing system which can be used to inject arbitrary commands/code to any application/service running in the same session, those we are using a "message loop"[92].

UIPI allows isolating processes running as a full administrator from processes running as an account with lower permissions than an administrator on the same interactive desktop. UIPI is specific to the windowing/graphic subsystem (aka Windows USER). Thus, a process with lower privileges can't perform operations on a process with higher privileges like: DLL injection, thread hooks for attaching, journal hooks for attaching, use window messages API (SendMessage/PostMessage) and more[93].

However, there are still resources that are shared between processes at different privilege levels like: clipboard, global atom table, desktop window and the desktop heap read-only shared memory. Also, painting on a screen is not controlled using UIPI, so a lower privilege application can paint over the surface region of a higher privilege application window - the GDI model does not allow control over painting surfaces[94].

Lastly, we can control the configuration of UIPI using the "EnableUIPI" value under the "HKLM\Software\Microsoft\Windows\CurrentVersion\Policies\System\" registry path - as shown in the screenshot below[95]. A value of "0" disables UIPI, and if the value is not present by default it means UIPI is enabled[96].

[92] https://www.slideserve.com/milek/shoot-the-messenger-win32-shatter-attacks-by-brett-moore
[93] https://learn.microsoft.com/en-us/previous-versions/aa905330(v=msdn.10)
[94] https://learn.microsoft.com/en-us/windows/win32/gdi/painting-and-drawing
[95] https://www.tipandtrick.net/fix-third-party-input-language-method-editor-ime-issues-in-ie-and-windows-vista-by-disabling-uipi/
[96] http://pferrie.epizy.com/papers/antidebug.pdf

File Virtualization

Due to security considerations (UAC enabled, it is also known as "UAC File Virtualization") as of Windows Vista it does allow standard (non-administrator) users to access/manipulate folders (like "Program Files" and the "Windows" directory) or specific registry areas - as was allowed in previous Windows versions. However, because there are legacy applications which expect doing those operations Windows include "File and Registry Virtualization"[97].

Thus, if we have an application running with-out administrative permissions and it tries to write to "Program Files" it will be redirected to "C:\Users\%username%\AppData\Local\VirtualStore\Program Files\" and the operation will succeed[98]. Without the file virtualization the operation is going to fail - as shown in the screenshot below. By the way, I have enabled the virtualization on "cmd.exe" using "Task Manager"[99].

---

[97] https://www.c-sharpcorner.com/uploadfile/GemingLeader/windows-file-and-registry-virtualization/
[98] https://flylib.com/books/en/2.955.1.34/1/
[99] https://www.experts-exchange.com/questions/28943516/What-is-UAC-Virtualization-in-the-Process-TASK-Manager.html
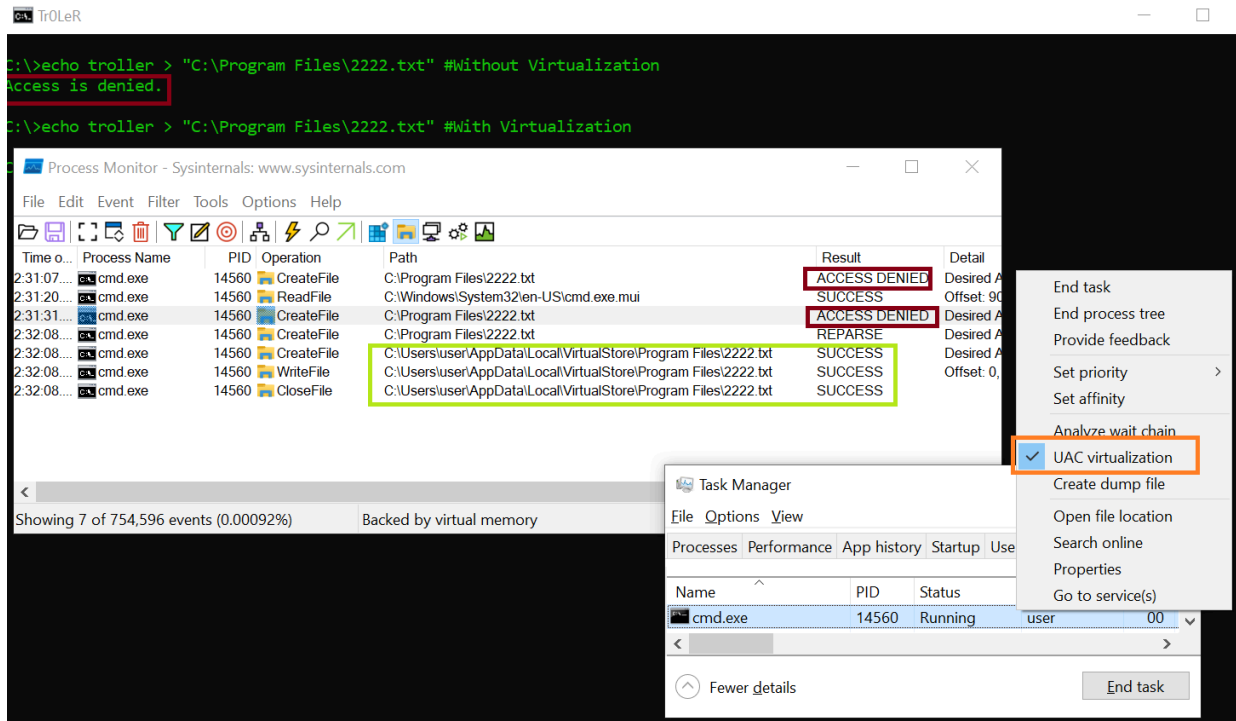
Registry Virtualization

Due to security considerations (like UAC enabled) as of Windows Vista it does allow standard (non administrator) users to access/manipulate specific registry areas - as was allowed in previous Windows versions. However, because there are legacy applications which expect doing those operations Windows include "File and Registry Virtualization"[100].

Thus, registry virtualization is an application compatibility technology. It helps registry write operations that can have global impact to be redirected to a per-user location. The redirection is transparent to the application writing/reading from the registry. There are specific cases in which it is enabled by default like for 32-bit processes which are interactive[101].

For example, if we have an application running with-out administrative permissions and it tries to write/read to "HKEY_LOCAL_MACHINE\SOFTWARE" it will be redirected to "HKEY_CURRENT_USER\Software\Classes\VirtualStore\MACHINE\SOFTWARE" and the operation will succeed. Without the registry virtualization the operation is going to fail - as shown in the screenshot below. By the way, I have enabled the virtualization on "powershell.exe" using "Task Manager"[102].

---

[100] https://www.c-sharpcorner.com/uploadfile/GemingLeader/windows-file-and-registry-virtualization/
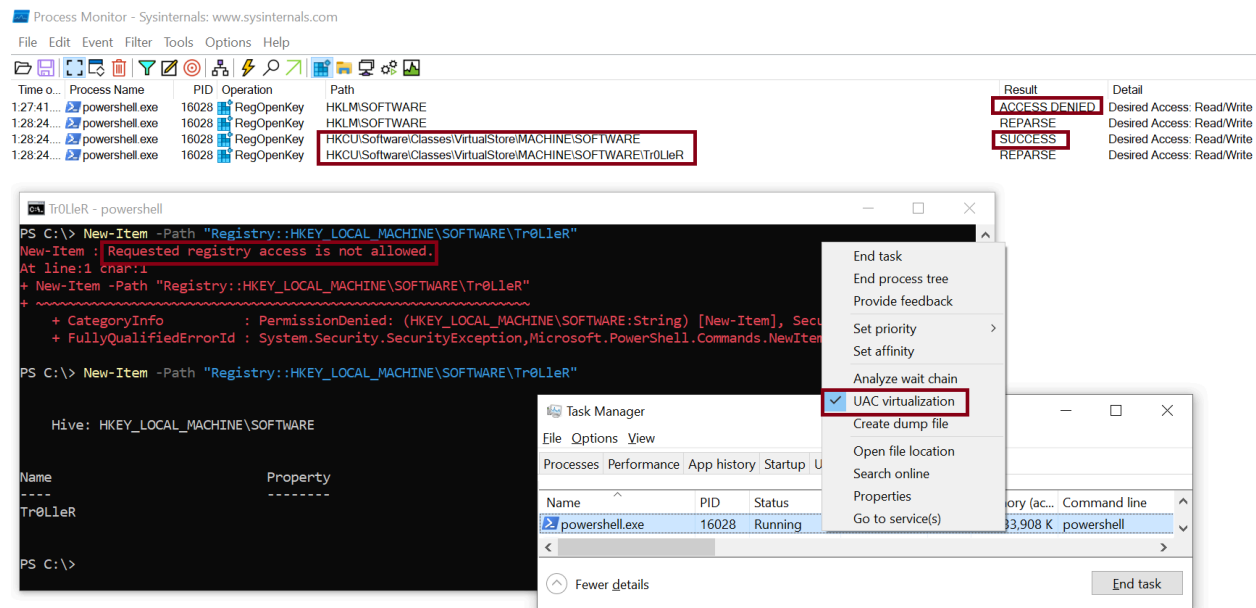[101] https://learn.microsoft.com/en-us/windows/win32/sysinfo/registry-virtualization
[102] https://www.experts-exchange.com/questions/28943516/What-is-UAC-Virtualization-in-the-Process-TASK-Manager.html

Share Permissions (Network Shares)

In general share permissions are used in order to control the access from Windows shares[103], which are folders exposed over the network. It is important to understand that share permissions apply to all the files/folders in the share. Also, share permissions are relevant for different file systems (NTFS/FAT/FAT32) as opposed to file/directory permissions which are only relevant for NTFS based filesystems[104].

Moreover, there are three types of share permissions: "Read", "Change" or "Full Control" - as shown in the screenshot below. When accessing a share that exposes a directory stored on an NTFS file system we use share permissions together with NTFS permissions. Thus, the most restrictive permission wins[105].

Lastly, share permissions are stored in the registry at the following location: "HKLM\SYSTEM\CurrentControlSet\Services\LanmanServer\Shares\Security"[106] - as shown in the screenshot below.   By the way, if we want we can also limit the number of users that can access the share simultaneously.

---

[103] https://medium.com/@boutnaru/the-windows-concepts-journey-windows-shares-8f9b60b8efd1
[104] https://blog.netwrix.com/2018/05/03/differences-between-share-and-ntfs-permissions/
[105] https://www.varonis.com/blog/ntfs-permissions-vs-share
[106] https://learn.microsoft.com/en-us/answers/questions/228360/share-folder-access-issue
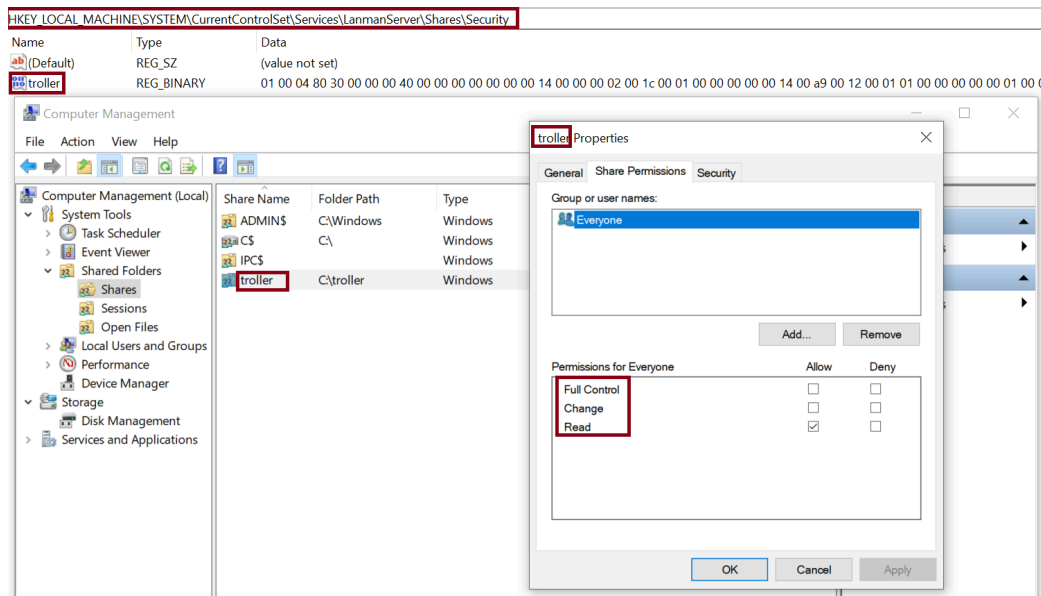
SysMon (System Monitor)

In general SysMon is a device driver and a Windows service which allows monitoring and logging System activities to the Windows event log[107] - as shown in the screenshot below[108]. It is part of the Sysinternals software package published by Microsoft[109]. After installing SysMon ("sysmon64/sysmon -i [<configfile>]") it remains resident across system reboots.

Moreover, SysMon has different capabilities such as logging process creation with full command line, record hash of process image files, logging loading of DLLs/Drivers with their signatures and hashes, logging the opening of disks/volumes in raw mode, logging network connections, generate events from early in the boot process (to capture activity by kernel mode malware), rule filtering, logging registry events, logging file deletion events and more[110]. Examples are shown also in the screenshot below.

Overall, the configurations of SysMon are stored in XML files, which allow event filtering based on type and their specific payload[111]. There are different templates for SysMon in Github which we can use like: SwiftOnSecurity SysMon config[112] and a repository of SysMon configuration modules by Olaf Hartong[113].

Lastly, it is recommended to send the events generated by SysMon to a SIEM (Security Information and Event Management). By the way, there is also a version of SysMon for the Linux operating system[114].

[107] https://medium.com/@boutnaru/the-windows-concept-journey-windows-event-logs-a9945bca421f
[108] https://www.syspanda.com/index.php/2017/05/19/sysmon-getting-started/
[109] https://www.blumira.com/glossary/system-monitor-sysmon/
[110] https://learn.microsoft.com/en-us/sysinternals/downloads/sysmon
[111] https://cqureacademy.com/blog/hacks/sysmon
[112] https://github.com/SwiftOnSecurity/sysmon-config
[113] https://github.com/olafhartong/sysmon-modular
[114] https://github.com/Sysinternals/SysmonForLinux

ELAM (Early Launch AntiMalware)

ELAM (Early Launch AntiMalware) is a feature that Microsoft provides which allows anti-malware software to start before other 3rd-party components. We can use it for developing drivers that are initialized before other boot start drivers for ensuring that subsequent drivers do not contain malicious code such as malware[115].
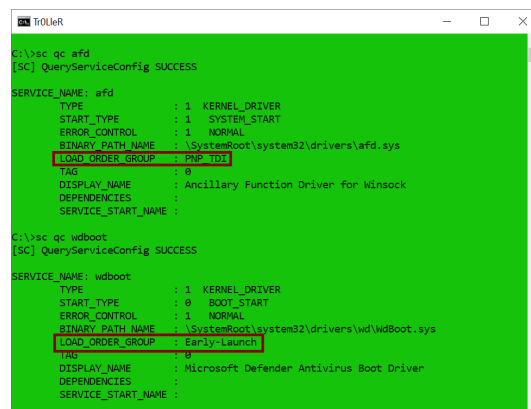
Overall, ELAM allows the registration of a kernel mode driver which is guaranteed to load early in the boot flow before other drivers. There is a signature data store for ELAM in the registry located at "HKLM\ELAM\[VENDOR_NAME]]" in which the vendor stores the allowlist/blocklist AV signatures for the ELAM driver to use. This hive is located at "%windir%\System32\config\ELAM" and loaded by the OS bootloader and unloaded after the ELAM has finished its job. Due to that, it does not appear after the boot process[116].

Moreover, as mentioned above ELAM allows antivirus software to scan boot drivers for malware before they are loaded. We can configure what boot start drivers we want to load based on the scan result of ELAM. The options are: "good only" (load only signed drivers which are not known as malware), "good and unknown" (loads only signed drivers or those not detected as malware) and "good, unknown, and bad but critical" (this is the defaults, drivers loaded if Windows does not start without it) and "all" (loads every driver). This can be set using "Local Computer Policy"/GPO in the following path "Computer Configuration\Administrative Templates\System\Early Launch Antimalware"[117].

Lastly, the above configuration is stored in the registry in the following location "HKLM\SYSTEM\CurrentControlSet\Policies\EarlyLaunch\DriverLoadPolicy"[118]. Also, ELAM vendors should be members of MVI which is the "Microsoft Virus Initiative" and their pass the WHQL which is the "Windows Hardware Quality Lab" for their driver[119].

---

[115] https://learn.microsoft.com/en-us/windows-hardware/drivers/install/early-launch-antimalware
[116] https://www.anoopcnair.com/understanding-windows-trusted-boot/
[117] https://www.bleepingcomputer.com/tutorials/configure-early-launch-antimalware-protection/
[118] https://learn.microsoft.com/en-us/windows-hardware/drivers/install/elam-driver-requirements
[119] https://learn.microsoft.com/en-us/windows-hardware/drivers/install/elam-prerequisites

Local System (NT AUTHORITY\SYSTEM)

Local System (NT AUTHORITY\SYSTEM) is a built-in user account (sometimes also called LocalSystem) which is used as a security context by different processes like Windows services[120] or scheduled tasks[121].

Overall, this user account has unlimited rights to a specific server/computer. By the way, when accessing a resource over the network (like a network share) with a thread/process holding a "SYSTEM" security context, the computer account is used. The name of the user in that case would be "[COUNTER_NAME]$"[122].

Moreover, The Local System user is more powerful than the builtin local administrator user. One example of that is that the local Administrator can't read the content of "KEY_LOCAL_MACHINE\SAM\SAM" while the System user can - as shown in the screenshot below. By the way, that subkey holds the db of the "Security Account Manager"[123].

Lasly, "NT AUTHORITY\SYSTEM" has a specific SID[124] which is relevant for all Windows systems in the world that is "S-1-5-18" - as shown below. Also, an access token[125] that belongs to a process with a security context of SYSTEM (for Windows VISTA+) has a mandatory level[126] of "system" and it contains the "Builtin\Administrators" group- as shown in the screenshot below. We can also see the list of privileges that are part of the token like: "SeDebugPrivilege" and "SeBackupPrivilege"[127].

---

[120] https://medium.com/@boutnaru/windows-services-part-1-5d6c2d25b31c
[121] https://medium.com/@boutnaru/windows-scheduler-tasks-84d14fe733c0
[122] https://www.libe.net/en-local-system
[123] https://medium.com/@boutnaru/windows-security-sam-security-account-manager-c93ddadf388a
[124] https://medium.com/@boutnaru/windows-security-sid-security-identifier-d5a27567d4e5
[125] https://medium.com/@boutnaru/windows-security-access-token-81cd00000c64
[126] https://medium.com/@boutnaru/the-windows-security-journey-mandatory-integrity-control-mic-f7963550c0e7
[127] https://learn.microsoft.com/en-us/windows/win32/services/localsystem-account

Network Service (NT AUTHORITY\NETWORK SERVICE)

Network Service (NT AUTHORITY\NETWORK SERVICE) is a builtin user account which is used as a security context by different services on Windows[128]. Examples of such services are: "RPC Endpoint Mapper", "Remote Desktop Services", "Network Location Awareness", "DNS Client" and "Workstation". By the way, the SID[129] of the "Network Service" account is "S-1-5-20".

Overall, "Network Service" has less permissions/privileges than the LocalSystem account[130], for example it is not part of the built-in Administrators group. However, the "Network Service" account can still interact over the network using the identity of the computer account[131].

Thus, we can say that the "Network Service" is a predefined local account used by the SCM (Service Control Manager). Also, this account is not recognized by the security subsystem in Windows, so we can't specify it's name while using "LookupAccountName" function[132].

Moreover, when using "net.exe"[133] to create a new user named "NetworkService" we get the following error "The account already exists". However, when trying to retrieve information about the "NetworkService" account we can "The user name could not be found." - as shown in the screenshot below. Lastly, we can use the "Network Service" account when calling the "CreateService"[134] function for creating a new service or when using "ChangeServiceConfig"[135] function to alter configuration parameters of a service.

---

[128] https://medium.com/@boutnaru/windows-services-part-1-5d6c2d25b31c
[129] https://medium.com/@boutnaru/windows-security-sid-security-identifier-d5a27567d4e5
[130] https://medium.com/@boutnaru/the-windows-security-journey-local-system-nt-authority-system-f087dc530588
[131] https://stackoverflow.com/questions/1301080/confused-over-localsystem-and-localservice-accounts
[132] https://learn.microsoft.com/en-us/windows/win32/services/networkservice-account
[133] https://medium.com/@boutnaru/the-windows-process-journey-net-exe-net-command-91e4964f20b8
[134] https://learn.microsoft.com/en-us/windows/win32/api/winsvc/nf-winsvc-createservicew
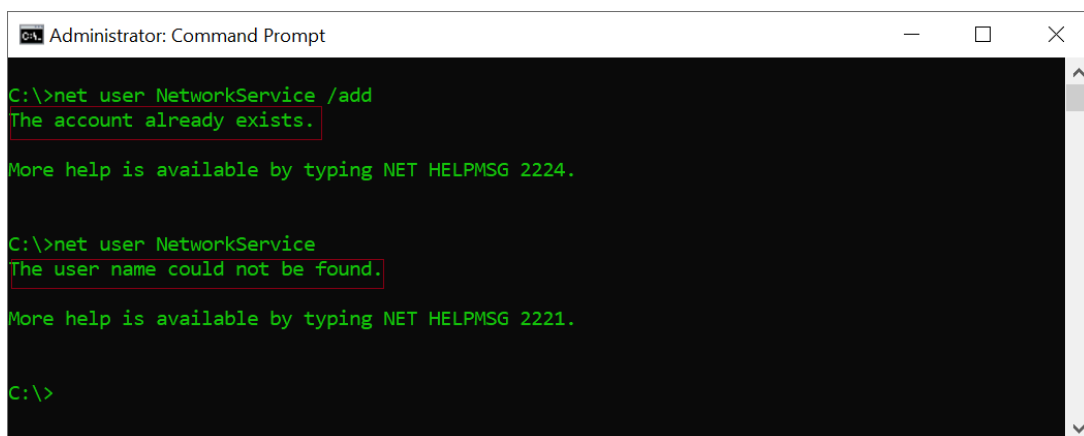[135] https://learn.microsoft.com/en-us/windows/win32/api/winsvc/nf-winsvc-changeserviceconfigw

Local Service (NT AUTHORITY\LOCAL SERVICE)

Local Service (NT AUTHORITY\LOCAL SERVICE) is a builtin user account which is used as a security context by different Windows services[136]. Examples of such services are: "Windows Time", "SSDP Discovery", "DHCP Client", "Windows Audio", "Windows Defender Firewall", and "Windows Image Acquisition". By the way, the SID[137] of the "Local Service" account is "S-1-5-19".

Overall, "Local Service" is a built-in user account like "Network Service" built-in user account[138] which is used to run least-privileges services. However, the user is limited to the local computer (as opposed to the "Network Service" account). Thus, this user should be used anytime the worker process/thread does not need access outside the specific workstation/server[139], because it accesses the network as an anonymous user.

Lastly, we can use the "Network Service" account when calling the "CreateService"[140] function for creating a new service or when using "ChangeServiceConfig"[141] function to alter configuration parameters of a service. As with "LocalSystem" and "NetworkService" the "LocalService" has its own profile in "HKEY_USERS\S-1-5-19"[142].

---

[136] https://medium.com/@boutnaru/windows-services-part-1-5d6c2d25b31c
[137] https://medium.com/@boutnaru/windows-security-sid-security-identifier-d5a27567d4e5
[138] https://medium.com/@boutnaru/the-windows-security-jorueny-network-service-nt-authority-network-service-e8706688e383
[139] https://stackoverflow.com/questions/1301080/confused-over-localsystem-and-localservice-accounts
[140] https://learn.microsoft.com/en-us/windows/win32/api/winsvc/nf-winsvc-createservicew
[141] https://learn.microsoft.com/en-us/windows/win32/api/winsvc/nf-winsvc-changeserviceconfigw
[142] https://www.cnblogs.com/talentzemin/p/16845794.html

Windows Defender Firewall

The "Windows Defender Firewall" (aka "Windows Firewall") is available as part of the Windows operating system is used in helping to prevent malicious software/hackers from gaining access to a workstation/server through the Internet\other networks to which the system is connected[143]. The firewall component was first included with Windows XP SP2\Windows 2003 Server SP1. Also, before Windows XP it was called "Internet Connection Firewall". We can manage the firewall using the local/domain group policy - as shown in the screenshot below. We can also leverage the GUI (firewall.cpl), "netsh.exe" or specific COM objects.

Moreover, each network interface has one of three profiles activated automatically: "Public", "Private" and "Domain". The "Public" profile is the most restrictive one and it assumes the interface has been connected to a shared network. The "Private " profile assumes the interface has been connected to a network isolated from the Internet, by the way only an administrator can set a network to this profile. The "Domain" is the least restrictive and is set when an interface is connected to a network with a domain trusted by the local computer[144]. By the way, we can also enable the firewall to log dropped packets and/or successful connections[145].

Lastly, it managed using the "Windows Defender Firewall" Windows service (MpsSvc) which is executed with the permissions/privileges of the "Local Service" user[146]. The service is dependent both on the "Base Filtering Engine" (%windir%\System32\BFE.DLL) service and the "Windows Defender Firewall Authorization" driver (%windir%\system32\drivers\mpsdrv.sys).

---
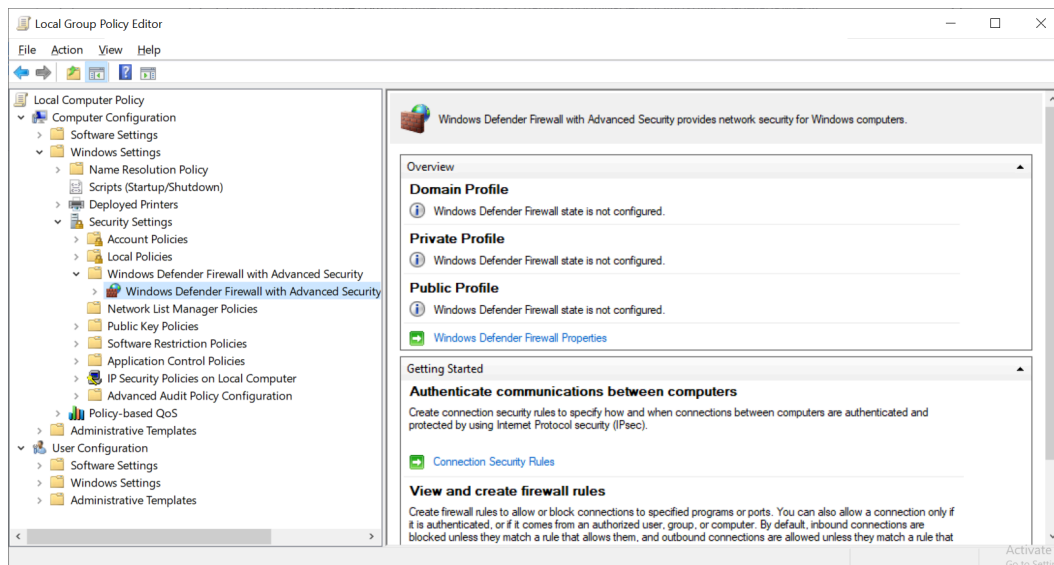
[143] https://learn.microsoft.com/en-us/mem/intune/user-help/you-need-to-enable-defender-firewall-windows
[144] https://en.wikipedia.org/wiki/Windows_Firewall
[145]https://learn.microsoft.com/en-us/windows/security/operating-system-security/network-security/windows-firewall/configure-logging?tabs=intune
[146] https://medium.com/@boutnaru/the-windows-security-journey-local-service-nt-authority-local-service-b1a624472931

TrustedInstaller

Since Windows 2008/Windows 7 most of the operating system files are owned by the "Trusted Installer" entity and thus in the ACL we can see the SID of that security principal[147] - as shown in the screenshot below. The reason of doing that is to prevent a process running under the "Administrator"/"Local System" account form replacing/altering operating system files[148].

Overall, the "TrustedInstaller" is a hidden built-in user of the operating system. It is used in the process of installing/removing updates and other Windows components[149]. By the way, the SID of the "NT Service\TrustedInstaller" user is "S-1-5-80-956008885-3418522649-1831038044-1853292631-2271478464"[150].

Lastly, in some cases "Trusted Installer" can be abbreviated as "TI"[151]. Also, there is a binary with the same name "%windir%\servicing\TrustedInstaller.exe", the description of the binary is "Windows Modules Installer" - more on that in a future writeup.

---

[147] https://medium.com/@boutnaru/windows-security-sid-security-identifier-d5a27567d4e5
[148] https://learn.microsoft.com/en-us/windows-server/identity/ad-ds/manage/understand-security-identifiers
[149] https://softwarekeep.com/blogs/what-is/what-is-trustedinstaller-and-should-i-remove-it-from-windows-10
[150] https://learn.microsoft.com/en-us/archive/msdn-magazine/2008/november/access-control-understanding-windows-file-and-registry-permissions
[151] https://renenyffenegger.ch/notes/Windows/security/SID/S-1-5/80/956008885-3418522649-1831038044-1853292631-2271478464

Windows Sessions

Every process started on a system belongs to a specific user maintained by an access token[152] and every process also belongs to a specific session. Each Windows session hosts processes, windows, Window Stations, Desktops and other resources[153]. A diagram that demonstrates that is shown below[154].

Also, we can create a process in another session using the Win32 API function "CreateProcessAsUser"[155]. We use the API call "SetTokenInformation"[156] while setting the "TokenInformationClass" with "TokenSessionId"[157]. The privilege[158] named SeTcbPrivilege (Act as part of the operating system) is needed for launching a process in a different session[159].

Lastly, its session has its own ID aka "Session ID". The first user that performs login is connected to session 1, the next is session 2 and so on[160]. By the way, session 0 is saved for services but I am going to elaborate about it in a different writeup. As part of the creation flow of a session an instance of "smss.exe"[161] is started.

[152] https://medium.com/@boutnaru/windows-security-access-token-81cd00000c64
[153] https://brianbondy.com/blog/100/understanding-windows-at-a-deeper-level-sessions-window-stations-and-desktops
[154] https://www.slideserve.com/ellery/windows-7-training
[155] https://learn.microsoft.com/en-us/windows/win32/api/processthreadsapi/nf-processthreadsapi-createprocessasusera
[156] https://learn.microsoft.com/en-us/windows/win32/api/securitybaseapi/nf-securitybaseapi-settokeninformation
[157] https://learn.microsoft.com/en-us/windows/win32/api/winnt/ne-winnt-token_information_class
[158] https://medium.com/@boutnaru/windows-security-privileges-b8fe18cf3d5a
[159] https://stackoverflow.com/questions/3128017/launching-a-process-in-user-s-session-from-a-service
[160] https://renenyffenegger.ch/notes/Windows/development/session/index
[161] https://medium.com/@boutnaru/the-windows-process-journey-smss-exe-session-manager-subsystem-bca2cf748d33

Secure Desktop

The "Secure Desktop" is used for displaying the sign-in UI and thus restricts the access and functionality until the sign-in is performed. The main difference between the "Secure Desktop" and the user's desktop is that only trusted processes running as SYSTEM are allowed to execute in the "Secure Desktop". It is can also be used when prompting for evaluation[162].

Overall, by using the "Secure Desktop" we can protect against input/output spoofing by presenting a credentials dialog or sending keys to it, which is required by UAC[163] to reduce the risk of malware by limiting the ability of malicious code from running with administrator permissions[164].

Thus, the consent prompt[165] and the credentials prompt by default are displayed on the "Secure Desktop". The "Secure Desktop" dims the user desktop and shows the elevation prompt which must be responded before continuing. After the user selects Yes/No the desktops is switched blacked to the user's desktop from the "Secure Desktop"[166]. Lastly, since Windows 2019 it is not possible to paste the content of the clipboard[167] on the "Secure Desktop". The "Secure Desktop" can be disabled/enabled using domain/local group policy[168].

[162]https://learn.microsoft.com/en-us/windows/security/threat-protection/security-policy-settings/user-account-control-switch-to-the-secure-desktop-when-prompting-for-elevation
[163] https://medium.com/@boutnaru/the-windows-security-journey-uac-user-account-control-ce395df5c784
[164] https://security.stackexchange.com/questions/3759/how-does-the-windows-secure-desktop-mode-work
[165]https://medium.com/@boutnaru/the-windows-process-journey-consent-exe-consent-ui-for-administrative-applications-d8e6976e8e40
[166]https://learn.microsoft.com/en-us/windows/security/application-security/application-control/user-account-control/how-it-works
[167] https://medium.com/@boutnaru/windows-clipboard-c63e369d35d9
[168]https://answers.microsoft.com/en-us/windows/forum/all/disable-secure-desktop-without-changing-uac-level/7c2522df-7beb-49d7-8793-e762facbd252

MSRC (Microsoft Security Response Center)

MSRC (Microsoft Security Response Center) is part of the defender community. For more than 20 years MSRC has engaged with security research in order to protect Microsoft's customers[169]. Thus, MSRC's mission is defined as protecting customers and Microsoft from current and emerging threats related to security and privacy[170].

Overall, MSRC also provides a bug bounty program. Among the bounty program we can find: cloud programs ("Microsoft Azure", "Microsoft Identity", "Xbox", "M365", "Microsoft Azure DevOps Services", "Microsoft Dynamics 365 and Power Platform", Microsoft AI"and "Microsoft Defender"), platform programs ("Microsoft Hyper-V", "Microsoft Windows Insider Preview", "Microsoft Applications and On-Premises Servers", "Windows Defender Application Guard", "Microsoft Edge (Chromium-based)" and "Microsoft 365 Insider") and defense & grant programs ("Mitigation Bypass and Bounty for Defense", "Grant: Microsoft Identity" and "SIKE Cryptographic Challenge"). Each program has its own scope, however they share the same high level requirements - as shown below[171].

Lastly, MSRC investigates all reports of security vulnerabilities affecting Microsoft products and services. Also, MSRC provides the information here as part of the ongoing effort to help manage security risks and help keep systems protected[172]. By the way, there is a GitHub repository which stores security research conducted by MSRC[173].

We want to award you for your research	Submissions that contain steps to reproduce your proof of concept code along with a detailed analysis are eligible for higher awards because they help us quickly assess the risk posed by a vulnerability.	**We are looking for new and novel vulnerabilities**	Your contributions help us address vulnerabilities we may have missed in the development process. If you are the first external researcher to identify a vulnerability we already know about and are working to fix you may still be eligible for a bounty award.
Avoid harm to customer data, privacy, and service availability	Some security research may occur on production services that our customers use and depend on. Do your best to avoid research that violates customer privacy, destroys data, or interrupts service. If you discover customer data while researching, or are unclear if it is safe to proceed, please stop immediately and contact us so we can take immediate action to resolve the issue and protect our customers.	**Follow coordinated vulnerability disclosure**	Our customer's security is important to us. If you find a vulnerability in our products, services, or devices, report it to us privately and give us the opportunity to correct the vulnerability and protect our customers before disclosing it publicly. We will work on each report diligently and to address it in a reasonable time. In recognition of your partnership we offer bounty awards and will acknowledge your contributions to customer security when the vulnerability is fixed.

---

[169] https://www.microsoft.com/en-us/msrc
[170] https://www.microsoft.com/en-us/msrc/mission?rtc=1
[171] https://www.microsoft.com/en-us/msrc/bounty
[172] https://msrc.microsoft.com/update-guide/en-us
[173] https://github.com/microsoft/MSRC-Security-Research

PEL (Protected Event Logging)

"Protected Event Logging" is a new security feature added in Windows 10. Its goal is to use encryption in order to protect sensitive data written to event logs. The data is encrypted using CMS (IETF Cryptographic Message Syntax) while it is written to the logs - as shown in the screenshot below[174]. The data can be decrypted when the records are moved to a central server using "Windows Event Forwarding"[175].

Moreover, "Protected Event Logging" is not enabled by default on Windows. As of Windows 10 the only powershell[176] uses PEL as part of its advanced logging[177].

Lastly, the main goal of PEL is to allow performing script block logging while preserving security. This is done by encrypting the content of scripts while writing it to the event logs.

---
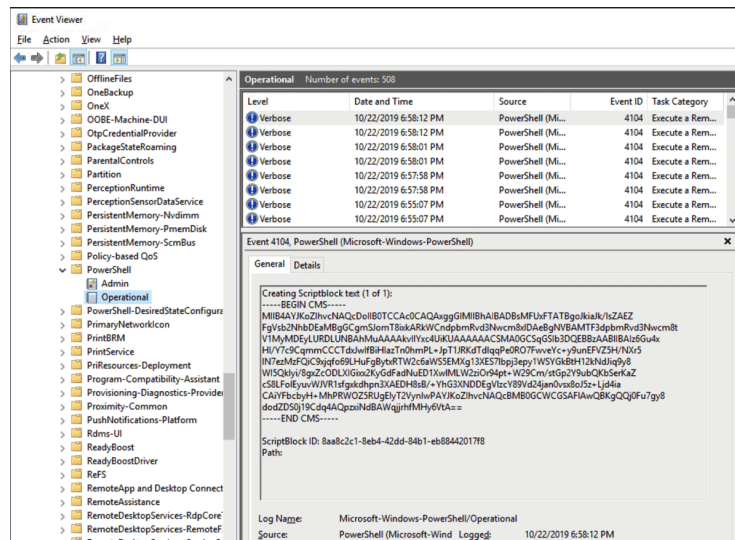
[174] https://4sysops.com/archives/encrypt-event-logs-and-files-with-powershell-and-group-policies/
[175] https://petri.com/windows-10-protected-event-logging/
[176] https://medium.com/@boutnaru/the-windows-process-journey-powershell-exe-windows-powershell-36daabaa74c4
[177] https://learn.microsoft.com/en-us/powershell/module/microsoft.powershell.core/about/about_logging_windows?view=powershell-7.4

DSE (Driver Signature Enforcement)

"DSE" (Driver Signature Enforcement) is a security feature that was introduced in Windows Vista 64 bit. It ensures that only signed drivers can be loaded by the operating system kernel[178]. It is important to know that starting with Windows 10/Windows Server 2016 kernel-mode drivers must be signed by the "Windows Hardware Dev Center Dashboard". In order to do that we need an EV certificate - this is called the "Driver Signing Policy"[179].

Moreover, one of the ways of bypassing this feature is by loading a vulnerable signed driver and overwriting the variable and thus disabling DSE[180]. By the way, from Windows 10 (1507) all drivers are signed by the "Hardware Dev Center" by leveraging the SHA2 hash function[181].

Lastly, in order to disable DSE we have a couple of choices. We can use the "bcdedit /set TESTSIGNING OFF" (using Windows test mode). Also, we can use the "Local Group Policy"/"Group Policy" for that, this can be done **"**User Configuration > Administrative Templates > System > Driver Installation" - as shown in the screenshot below. Finally, we can also reboot the system and go to the troubleshooting menu (pressing F7) and selecting "Disable driver signature enforcement"[182].

---
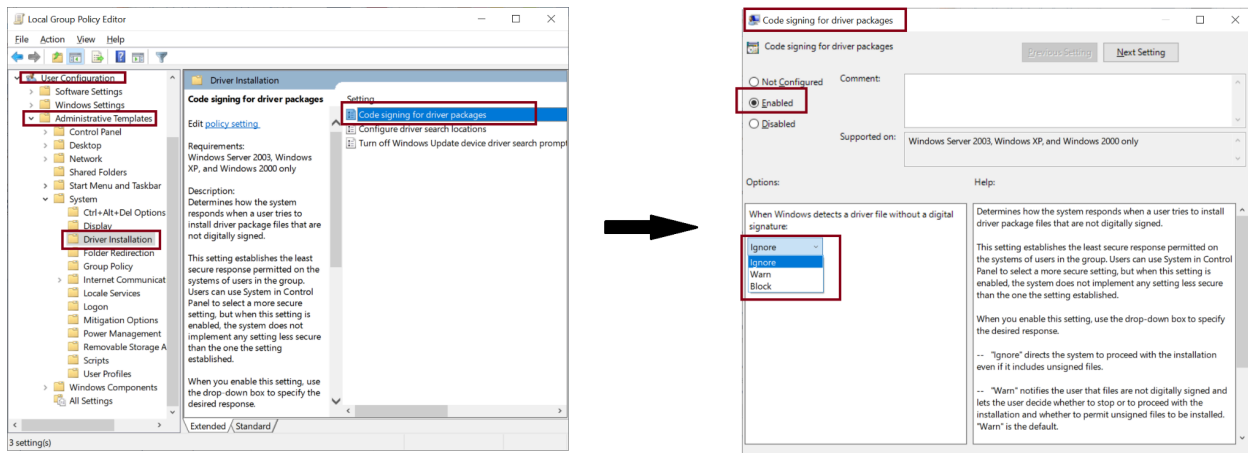
[178] https://www.codeproject.com/Articles/5348168/Disable-Driver-Signature-Enforcement-with-DSE-Patc
[179] https://learn.microsoft.com/en-us/windows-hardware/drivers/install/kernel-mode-code-signing-policy--windows-vista-and-later-
[180] https://github.com/hfiref0x/DSEFix
[181] https://learn.microsoft.com/en-us/windows-hardware/drivers/install/driver-signing
[182] https://www.majorgeeks.com/content/page/how_to_disable_driver_signature_enforcement.html

EMET (Enhanced Mitigation Experience Toolkit)

EMET (Enhanced Mitigation Experience Toolkit) is a utility created by Microsoft for helping in preventing vulnerabilities in software from being successfully exploited. This is done by leveraging mitigation techniques[183]. EMET was released in 2009 as a standalone tool from Windows 10 Microsoft has implemented many features and mitigations that can make EMET unnecessary[184].

Overall, by using EMET an administrator can enable different security mitigations such as: DEP[185] (Data Execution Prevention), ASLR[186] (Address Space Layout Randomization), Certificate Trust (Pinning), SEHOP (Structured Exception Handler Overwrite Protection) - as shown in the screenshot below. We can configure each mitigation as relevant for a specific application or system wide. The second which is called "Always On" is more secure[187].

Lastly, the biggest benefit of EMET is the ability to enable different security mitigations without the need of recompiling the binary. It is important to know that EMET is implemented using the Windows shim infrastructure called the "Application Compatibility Framework"[188].

---

[183] https://support.microsoft.com/en-us/topic/emet-mitigations-guidelines-b529d543-2a81-7b5a-d529-84b30e1ecee0
[184] https://msrc.microsoft.com/blog/2016/02/enhanced-mitigation-experience-toolkit-emet-version-5-5-is-now-available/
[185] https://medium.com/@boutnaru/security-nx-bit-non-executable-18759fd2802e
[186] https://medium.com/@boutnaru/security-aslr-address-space-layout-randomization-part-1-overview-3aec7fec01e0
[187] https://insights.sei.cmu.edu/blog/life-beyond-microsoft-emet/
[188] https://techcommunity.microsoft.com/t5/configuration-manager-archive/deploying-and-configuring-the-enhanced-mitigation-experience/ba-p/273099

Windows Defender Antivirus

"Windows Defender Antivirus" is available for Windows 10/11/Server. It uses advanced techniques for protecting from known/unknown malware. It does that by monitoring suspicious activities and blocking threats based on behavior. This is achieved through machine learning/anomaly detection (until 2015 it used only statically signatures), cloud-based intelligence, and real-time updates. "Windows Defender Antivirus" works both online and offline[189] - as shown in the screenshot below. By the way, it is also called MDA (Microsoft Defender AntiVirus).

Overall, we can think about "Windows Defender Antivirus" as a next-generation protection solution that comes with Windows, Microsoft Defender Antivirus is real-time, always-on antivirus protection[190]. By the way, there is also a version of "Microsoft Defender: Antivirus" for Android[191], however it is out of the scope of the current writeup.

Lastly, since 2021 "Windows Defender Antivirus" is part of a larger Microsoft Defender brand. Among that brand are included: "Microsoft Defender Vulnerability Management", "Microsoft Defender for Cloud Apps", "Microsoft Defender for Identity", "Microsoft Defender Endpoint" and more[192].

---
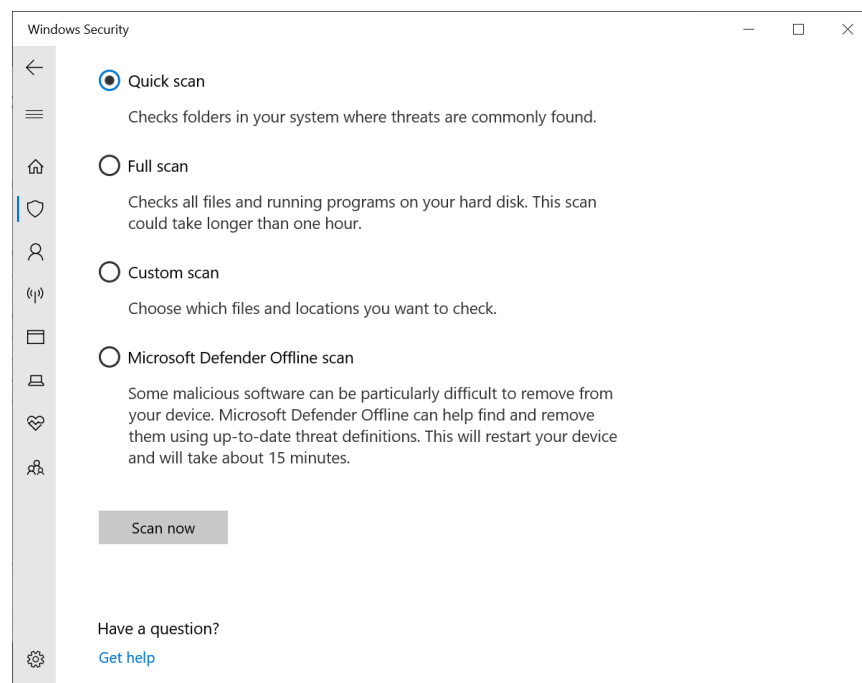
[189]https://learn.microsoft.com/en-us/microsoft-365/security/defender-endpoint/microsoft-defender-antivirus-windows?view=o365-worldwide
[190] https://www.microsoft.com/en-us/windows/comprehensive-security
[191] https://play.google.com/store/apps/details?id=com.microsoft.scmx&hl=en&gl=US
[192] https://en.wikipedia.org/wiki/Microsoft_Defender_Antivirus

Windows Defender SmartScreen

Windows SmartScreen is a cloud-based anti malware/phishing solution included in different Microsoft products (Windows/Internet Explorer/Microsoft Edge). The SmartScreen intelligence is also in use as part of Microsoft's online services backend (like Outlook.com/Bing search engine).

Since Windows 10 the setting of SmartScreen is part of "Windows Defender Security Center"[193] - as shown in the screenshot below

Thus, Windows Defender SmartScreen is a filtering mechanism which verifies files/websites in order to prevent malicious activities on the system. The different features which are provided by SmartScreen are: protection from unauthorized applications (checks the publisher's signature), keeping the user safe from phishing/malware attacks (based on a DB of malicious websites), automatic blocking of malicious URLs, reputation checks and safe downloading[194].

Moreover, SmartScreen has different benefits such as the operating system integration, improved heuristics and diagnostic data and management using Microsoft Intune and group policy. It is important to understand it does not protect against malicious files on internal locations or network shares such as shared folders with UNC paths or SMB/CIFS[195].

Lastly, in case we think there is a false positive/negative we can submit the specific file to Microsoft for further analysis[196].

---
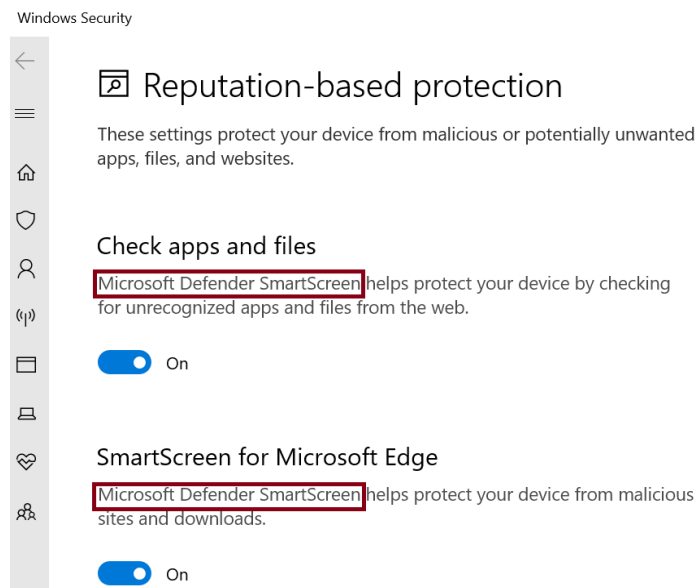
[193] https://en.wikipedia.org/wiki/Microsoft_SmartScreen
[194] https://signmycode.com/blog/what-is-windows-defender-smartscreen
[195] https://learn.microsoft.com/en-us/windows/security/operating-system-security/virus-and-threat-protection/microsoft-defender-smartscreen/
[196] https://www.microsoft.com/wdsi/filesubmission/

MSDE (Microsoft Defender for Endpoint)

MSDE (Microsoft Defender for Endpoint) is an enterprise endpoint security solution\platform. It is used for helping enterprises prevent\detect\investigate\respond to advanced threats. "Defender for Endpoint" leverages technologies which are built into Windows 10+ and/or Microsoft's cloud services. Among those technologies are: threat intelligence, cloud security analytics and endpoint behavioral sensors. Those provide users: attack surface reduction, next generation protection and more[197].

Overall, "Defender for Endpoint" has the following key capabilities: auto-deployed deception (automatically generate and disperse deception techniques at scale to expose cyberattackers), copilot for security (security-specific generative AI to rapidly investigate and respond, prioritize alerts, and learn new skills - as shown in the screenshot below), network detection and response, simplified endpoint management and more[198].

Lastly, "Microsoft Defender for Endpoint" supports not only Windows as it also supports Linux[199], Android[200], macOS[201] and iOS[202].

---

[197] https://learn.microsoft.com/en-us/defender-endpoint/microsoft-defender-endpoint
[198] https://www.microsoft.com/en-us/security/business/endpoint-security/microsoft-defender-endpoint
[199] https://learn.microsoft.com/en-us/defender-endpoint/linux-whatsnew
[200] https://learn.microsoft.com/en-us/defender-endpoint/android-whatsnew
[201] https://learn.microsoft.com/en-us/defender-endpoint/mac-whatsnew
[202] https://learn.microsoft.com/en-us/defender-endpoint/ios-whatsnew

SRP (Software Restriction Policies)

SRP ("Software Restrictions Policies") is a security feature available in Windows that we can configure using "Group Policy"/"Local Security Policy". The main goal of SRP is to allow administrators to control which applications/programs 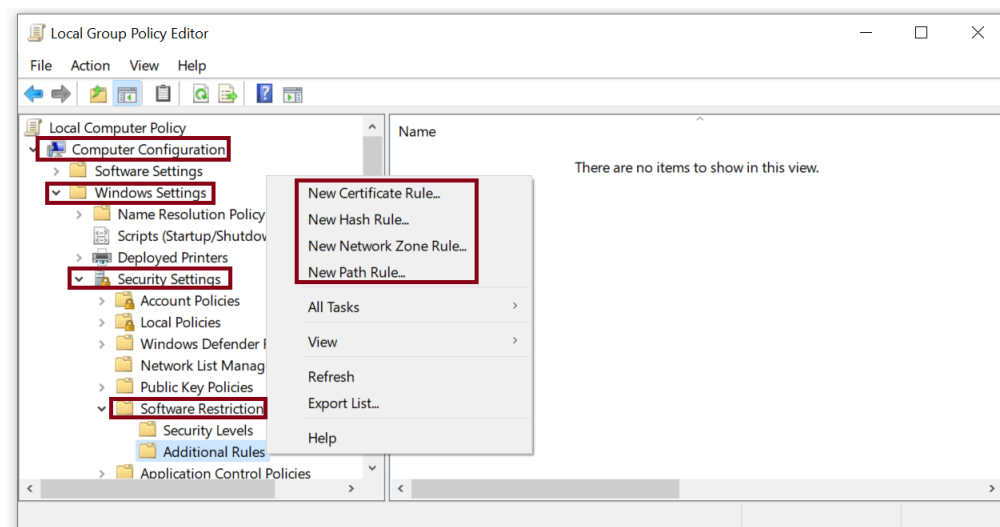are allowed to run on a Windows based device. We can set different restriction policies using SRP like: which programs (executables) can run, who can add trusted publishers, configure if the policies affect all users or just specific ones, preventing specific executable files from being executed based on OU, domain and more[203].

Moreover, SRP has been supported since Windows XP\Windows Server 2003. We can think about SRP as "Trust Policies" that can restrict the execution of scripts and other code (think about macros, ActiveX controls and more) from running. Starting from Windows 7/Windows 2008 R2 we should use "AppLocker" instead of SRP. From Windows 10 we can also use WDAC[204].

Lastly, each SRP (Software Restriction Policy) rule can have one of three security levels[205]: "Disallowed" (software will not run), "Basic User" (programs can execute as a user without Administrator access rights) and "Unrestricted" (the access rights are determined by the access rights of the user). Overall, we have four types of SRP rules that we can define: "Certificate Rule", "Hash Rule", "Network Zone Rule" and "Path Rule" - as shown in the screenshot below. For each of them one security level is defined as an action.

---

[203] https://learn.microsoft.com/en-us/windows-server/identity/software-restriction-policies/software-restriction-policies
[204]https://learn.microsoft.com/en-us/windows-server/identity/software-restriction-policies/software-restriction-policies-technical-overview
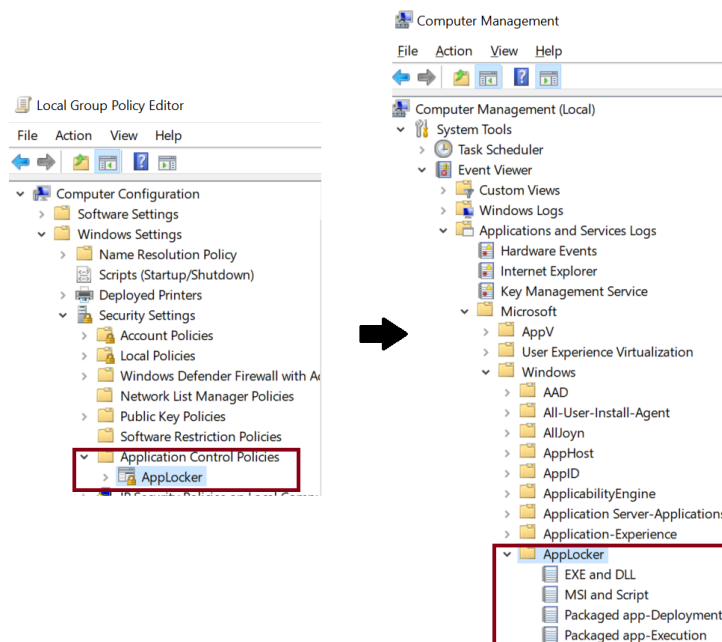[205] https://www.adamcouch.co.uk/software-restriction-policies/

AppLocker (Application Locking)

"AppLocker" is a security feature introduced in Windows 7. The goal of "AppLocker" is to allow a system administrator to configure (control) which applications are allowed to be executed on a Windows system. Thus, preventing users from executing unwanted/unapproved applications. AppLocker's settings can apply for all users of the computer or just specific users/groups[206]. The configuration can be done using "GPO"\"Local Group Policy"->"Computer Configuration"->"Security Settings"->"Application Control Policies"->"AppLocker" - as shown in the screenshot below.

Moreover, AppLocker's rules can be configured based on: the path from which the app/script is launched, attributes of the code signing certificate used for signing the binary, metadata of the file like the original file name, the hash of the file and more. The rules can target different file formats such as: scripts ("*.ps1", "*.bat", *".cmd", "*.vbs" and "*.js"), executables ("*.exe" and "*.com"), libraries ("*.dll" and "*.ocx"), Windows installer files ("*.msi", "*.msp" and "*.mst") and packaged apps\packaged app installers ("*.appx")[207].

Lastly, it is important to understand that the AppLocker rules can apply for any PE file[208] regardless of the extension. Also, in case a DLL rule is configured we need to create an allow rule that covers every DLL used by all allowed apps. In case of a violation relevant entries are logged to the Windows eventlog[209] - as shown in the screenshot below.

---

[206]https://learn.microsoft.com/en-us/windows/security/application-security/application-control/windows-defender-application-control/wdac-and-applocker-overview

[207]https://learn.microsoft.com/en-us/windows/security/application-security/application-control/windows-defender-application-control/applocker/working-with-applocker-rules
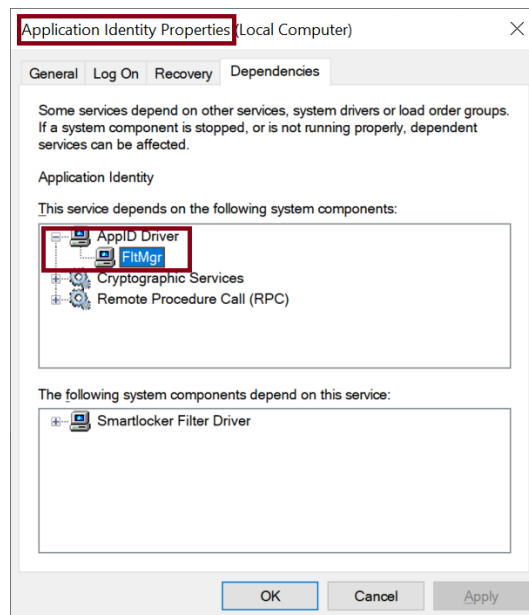
[208] https://medium.com/@boutnaru/the-portable-executable-journey-file-header-struct-image-file-header-00360271f147

[209] https://medium.com/@boutnaru/the-windows-concept-journey-windows-event-logs-a9945bca421f

AppIDSvc (Application Identity Service)

AppIDSvc (Application Identity Service) is a Windows service hosted by "svchost.exe"[210]. The description of the service states it is used for determining and verifying the identity of an application. Thus, if we disable this service it will prevent AppLocker[211] from being enforced.

Moreover, the service is launched using the access permissions of the "Local Service" user[212]. Also, the process of the service is defined with the protection level of "PsProtectedSignerWindows-Light", this is also known as "Protected Process Light" (PPL). It is important to understand that the actual protection level is a combination of "Type" and "Signer"[213].

Lastly, the "Application Identity Service" is dependent on: RPC ("Remote Procedure Call), "Cryptographic Services" (think about verifying digital signatures of executables) and the "AppID Driver" (%windir%\system32\drivers\appid.sys), which itself is dependent on "FltMgr" (%windir%\system32\drivers\fltmgr.sys) - as shown in the screenshot below. The user-mode part of the service is implemented as part of "%windir%\System32\appidsvc.dll", which is digitally signed by Microsoft.

[210] https://medium.com/@boutnaru/the-windows-process-journey-svchost-exe-host-process-for-windows-services-b18c65f7073f
[211] https://medium.com/@boutnaru/the-windows-security-journey-applocker-application-locking-b9547fb9cbbd
[212] https://medium.com/@boutnaru/the-windows-security-journey-local-service-nt-authority-local-service-b1a624472931
[213] https://uberagent.com/blog/uberagent-7-preview-edr-antivirus-windows-defender-protected-process-performance-monitoring/

Differences between AppLocker and SRP

In general, both SRP[214] and Applocker[215] are built in security features of the Windows operating system used for application control/whitelisting in order to increase the security posture of a Windows based device. There are some differences between the two, part of those differences are documented in this writeup - an example of a table comparing the both is shown below[216].

Overall, SRP is relevant since Windows XP/Windows Server 2003 while AppLocker is relevant since Windows 7/Window Server 2008 R2. It is important to know that AppLocker has an audit more (which allows testing the configuration in production) and rule exceptions while SRP does not. Both SRP and AppLocker "Windows Installer" files, however only AppLocker package apps and installers. SRP has the ability to set different security levels for approved applications (like as limited/basic user) this is not supported by AppLocker[217].

Lastly, SRP's enforcement is done in user-mode while AppLocker's enforcement occurs in kernel-mode. Moreover, AppLocker supports exporting importing policies, rule collections, PowerShell support and the ability to define custom error messages. The SRP policies are relevant for all users (per GPO) while in AppLocker we can specify specific users/groups per rule[218].

Feature	SRP (Windows XP & W2K3)	AppLocker
Rule scope	Specific user or group (per GPO)	Specific user or group (per rule)
Rule conditions provided	File hash, path, certificate, registry path, and Internet zone rules	File hash, path, and publisher rules
Rule types provided	Allow and deny	Allow and deny
Default rule action	Allow and deny	Deny
Audit-only mode	No	Yes
Wizard to create multiple rules at one time	No	Yes
Policy import or export	No	Yes
Rule collection	No	Yes
PowerShell support	No	Yes
Custom error messages	No	Yes

---

[214] https://medium.com/@boutnaru/the-windows-security-journey-srp-software-restriction-policies-9f658a4ed648
[215] https://medium.com/@boutnaru/the-windows-security-journey-applocker-application-locking-b9547fb9cbbd
[216] https://www.cievo.sk/2012/11/30/quickie-software-restricion-policy-vs-applocker/
[217]https://learn.microsoft.com/en-us/windows-server/identity/software-restriction-policies/software-restriction-policies-technical-overview
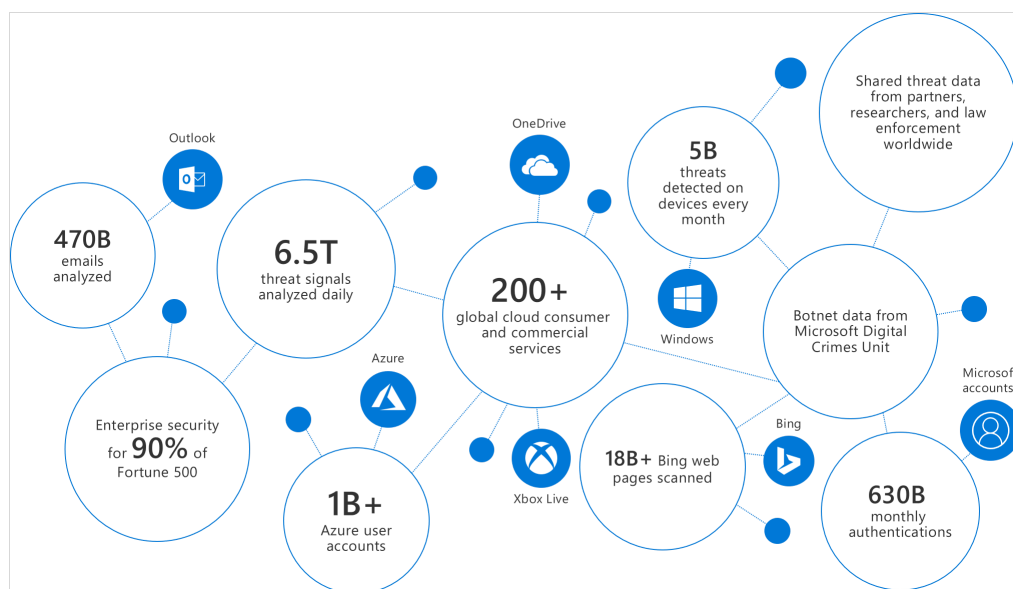[218] https://slideplayer.com/slide/11890654/

Microsoft ISG (Microsoft Intelligent Security Graph)

Microsoft ISG (Microsoft   Intelligent Security Graph) is a collection of data (like security intelligence), people, and machine learning used to efficiently and effectively protect and strengthen Microsoft products and services[219].

Overall, as developers we can consume data from ISG using a REST API (Microsoft Security Graph API). Using the API we can retrieve different events in various categories such as: "Anonymous IP Risks", "Leaked Credentials Risks", "Suspicious IP Risks", "Malware Risks", "Impossible Travel Risks" and "Unfamiliar Location Risks"[220].

Thus, we can use the "Microsoft Graph Security API" in order to build security applications that can: provide visibility and enable proactive risk management, respond to new security threats, automate security actions, provide contextual data, enrichment information and more[221].

Lastly, ISG pulls data from Microsoft endpoints (like Windows/Office 365), services and from companies part of the Microsoft Intelligent Security Association. Examples are the founding members like "Anomoli", "PwC" and "Palo Alto Networks" which add signals to the graph[222]. For better understanding the data included in the graph you can check out the diagram below, which is relevant for 2019[223].

---

[219] https://www.linkedin.com/pulse/microsoft-intelligent-security-graph-peter-ginnegar
[220] https://www.sharepointeurope.com/consuming-microsoft-intelligent-security-graph-custom-developed-solutions/
[221]https://github.com/microsoftgraph/microsoft-graph-docs-contrib/blob/main/api-reference/beta/resources/security-api-overview.md
[222]https://www.forbes.com/sites/patrickmoorhead/2018/04/17/microsoft-announces-iot-security-silicon-and-platform-in-its-march-to-deliver-end-to-end-security/
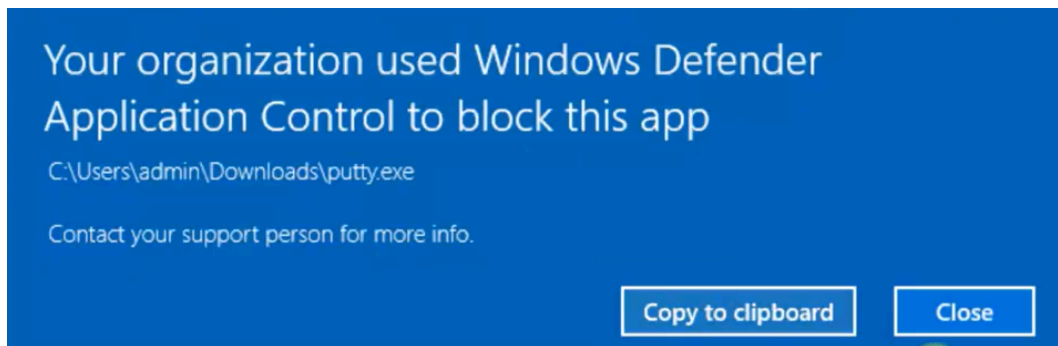[223]https://web.archive.org/web/20190529062503/https://www.microsoft.com/security/blog/2019/05/14/executing-vision-microsoft-threat-protection/

WDAC (Windows Defender Application Control)

WDAC (Windows Defender Application Control) is a security feature which was introduced as part of Windows 10. We can use WDAC for controlling which applications/drivers are allowed (or blocked) to be executed/loaded on a Windows base device - as shown in the screenshot below[224]. We can think about it as a replacement/enhancement of "Software Restriction Policy"[225] and/or AppLocker[226].

Overall, the configuration of WDAC is relevant and affects all users connected to the system. Also, when it was released it was part of "Device Guard" under the name of "Configurable Code Integrity". WDAC supports defining rules based on different attributes such as: the reputation of the application (based on Microsoft's Intelligent security graph), the process which launched the binary/application, the path from which the application was launched, signed metadata of the binary (like the original file name), the identity of the process which started the installation flow of the application, attributes of the code-signing certificates used to signed the application[227].

Lastly, WDAC can be configured using different mechanisms/tools like: "Group Policy", "Script", "Microsoft Configuration Manager" or "Microsoft Intune"[228]. IT professionals can create powerful WDAC policies for deployment using "The WDAC Policy Wizard" tool created by Microsoft[229].

---

[224] https://blog.ciaops.com/category/windows/
[225] https://medium.com/@boutnaru/the-windows-security-journey-srp-software-restriction-policies-9f658a4ed648
[226] https://medium.com/@boutnaru/the-windows-security-journey-applocker-application-locking-b9547fb9cbbd
[227] https://learn.microsoft.com/en-us/windows/security/application-security/application-control/windows-defender-application-control/wdac-and-applocker-overview
[228] https://learn.microsoft.com/en-us/windows/security/application-security/application-control/windows-defender-application-control/feature-availability
[229] https://github.com/MicrosoftDocs/WDAC-Toolkit

Differences between WDAC and AppLocker

In general, both AppLocker[230] and WDAC[231] are built in security features of the Windows operating system used for application control/whitelisting in order to increase the security posture of a Windows based device. There are some differences between the two, part of those differences are documented in this writeup.

Overall, AppLocker is supported since Windows 8 while WDAC is available for Windows 10/Windows Server 2016. There are a couple of features which are only supported by WDAC and not AppLocker such as: kernel mode policies, per app rules, reputation based intelligence, COM object whitelisting, application ID tagging, packaged app rules[232] - more on those in future writeups. In the case of WDAC it is recommended to start with a template policy and remove/add rules on top of it. WDAC wizard provides three basic policy templates - as shown in the screenshot below[233].

Lastly, WDAC is suitable for a highly secured environment. As opposed to AppLocker in WDAC, administrators can be excluded by rules for executing specific applications. Think about a case in which we have not allows the execution of an installer, even an admin can't uninstall the application[234]. Thus, if we want to enforce different policies for users/groups on a shared device or we don't want to set application control rules on DLLs/drivers we should used AppLocker and not WDAC[235].

---

[230] https://medium.com/@boutnaru/the-windows-security-journey-applocker-application-locking-b9547fb9cbbd
[231] https://medium.com/@boutnaru/the-windows-security-journey-wdac-windows-defender-application-control-26955abe4c01
[232] https://learn.microsoft.com/en-us/windows/security/application-security/application-control/windows-defender-application-control/feature-availability
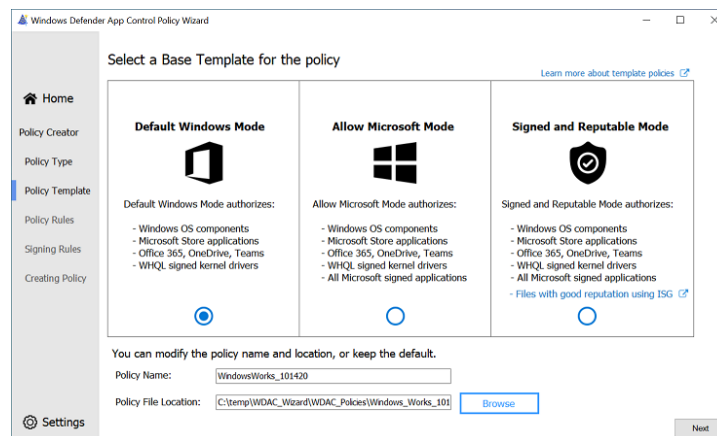[233] https://learn.microsoft.com/en-us/windows/security/application-security/application-control/windows-defender-application-control/design/wdac-wizard-create-base-policy
[234] https://www.reddit.com/r/Intune/comments/1apqpjp/applocker_vs_wdac/
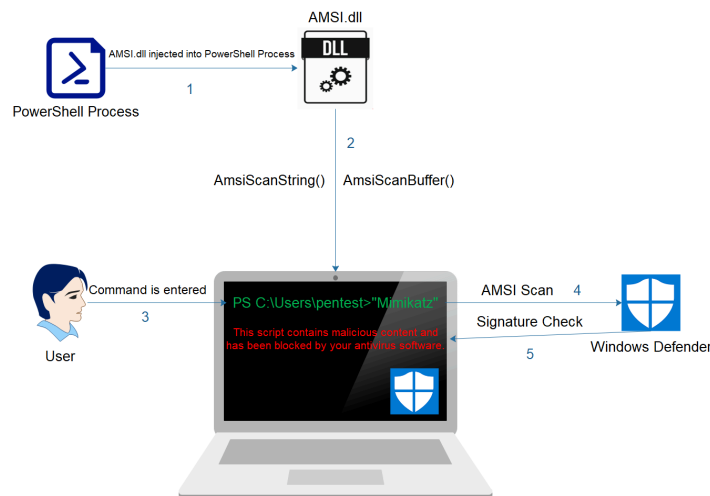[235] https://learn.microsoft.com/en-us/windows/security/application-security/application-control/windows-defender-application-control/wdac-and-applocker-overview

AMSI (Anti-Malware Scan Interface)

"AMSI" (Anti-Malware Scan Interface) is used by "Microsoft Defender for Endpoint" in order to enhance protection against fileless malware, dynamic script-based attacks, and other nontraditional cyber threats. AMSI supports different scripting languages such as: VBScript, JScript, PowerShell, WMI (Windows Management Instrumentation), .NET 4.8 assemblies scanning and more[236].

Moreover, AMSI is used by major antivirus engines (like BitDefender, ESET and Kaspersky) for evaluating the content for malware and detecting possible threats. Also, 3-rd party AVs can replace the default AMSI engine, thus every AMSI call ends up invoking the third-party engine. However, in case there is not a 3-party aware AMSI engine the security evaluation requested over AMSI is provided by Defender[237].

Lastly, when a script is executed (like using PowerShell) the "%windir%\System32\amsi.dll" (or "%windir%\SysWOW64\amsi.dll" in case of a 32-bit process) is loaded into the memory address space of the process (powershell.exe in our example). Before the script is executed the two APIs are called by the relevant AV engine to scan the buffer and string (AmsiScanBuffer() and AmsiScanString()). In case a known signature is triggered the execution doesn't initiate and a message appears that the script has been blocked by the antivirus software - as shown in the diagram below[238].

---

[236] https://learn.microsoft.com/en-us/defender-endpoint/amsi-on-mdav
[237] https://github.com/MirekVales/MVsDotNetAMSIClient
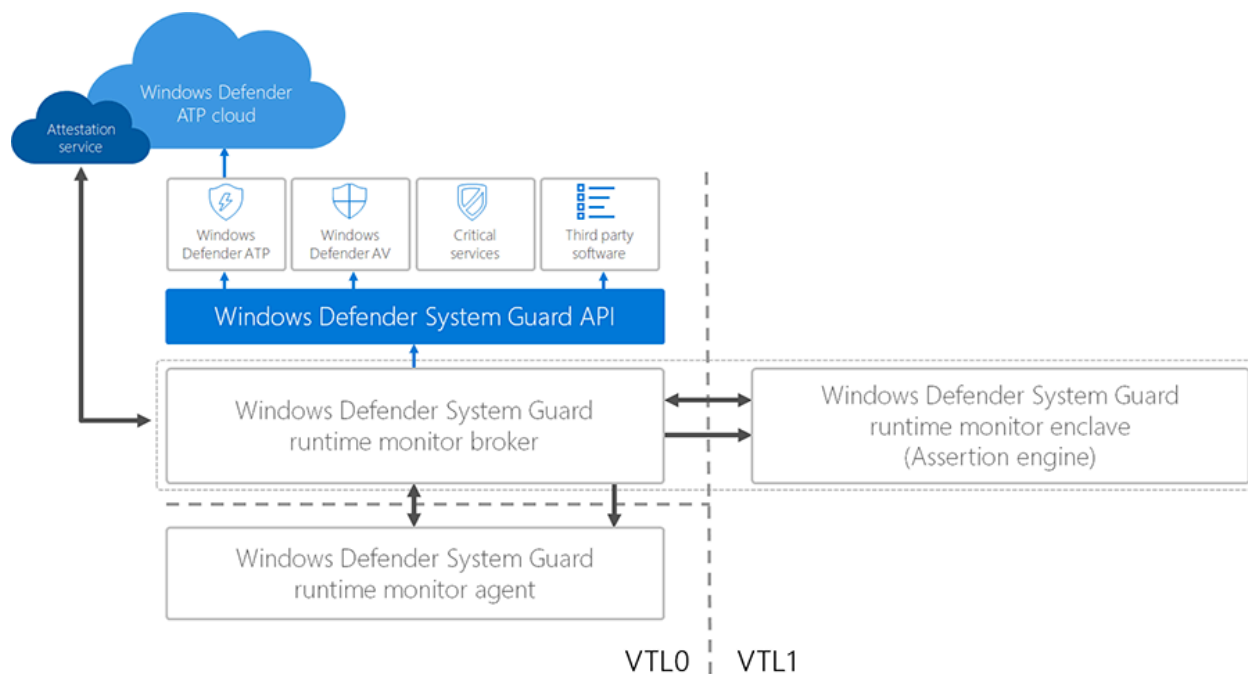[238] https://pentestlaboratories.com/2021/05/17/amsi-bypass-methods/

SGRM (System Guard Runtime Monitor)

SGRM (System Guard Runtime Monitor) is part of Windows Defender, which is used to ensure the operating system integrity. It was introduced as part of Windows 10 1709 update. By the way, SGRM has a Microsoft internal project name which is "Octagon"[239].

Overall, the main goal of SGRM is ensuring the security of the operating system. The runtime attestation of SGRM can assist with use cases such as: detecting artifacts of exploits/rootkits/kernel tampering, providing conditional access based on device posture, helping with game anti-cheating capabilities and more - a block diagram of SGRM is shown below[240].

Lastly, as part of Windows 10 Fall Creators Update all system integrity features were merged into "Windows Defender System Guard". It provides an API that can be used for attesting the device in a point of time (based on different measurements). For the attestation to be trustable it must be: isolated from an attacker, signed and the isolation should be attestable. In order to provide that VBS and enclaves are leveraged[241].

---

[239] https://blog.syscall.party/2022/08/02/inside-windows-defender-system-guard-runtime-monitor.html
[240] https://www.microsoft.com/en-us/security/blog/2018/04/19/introducing-windows-defender-system-guard-runtime-attestation/
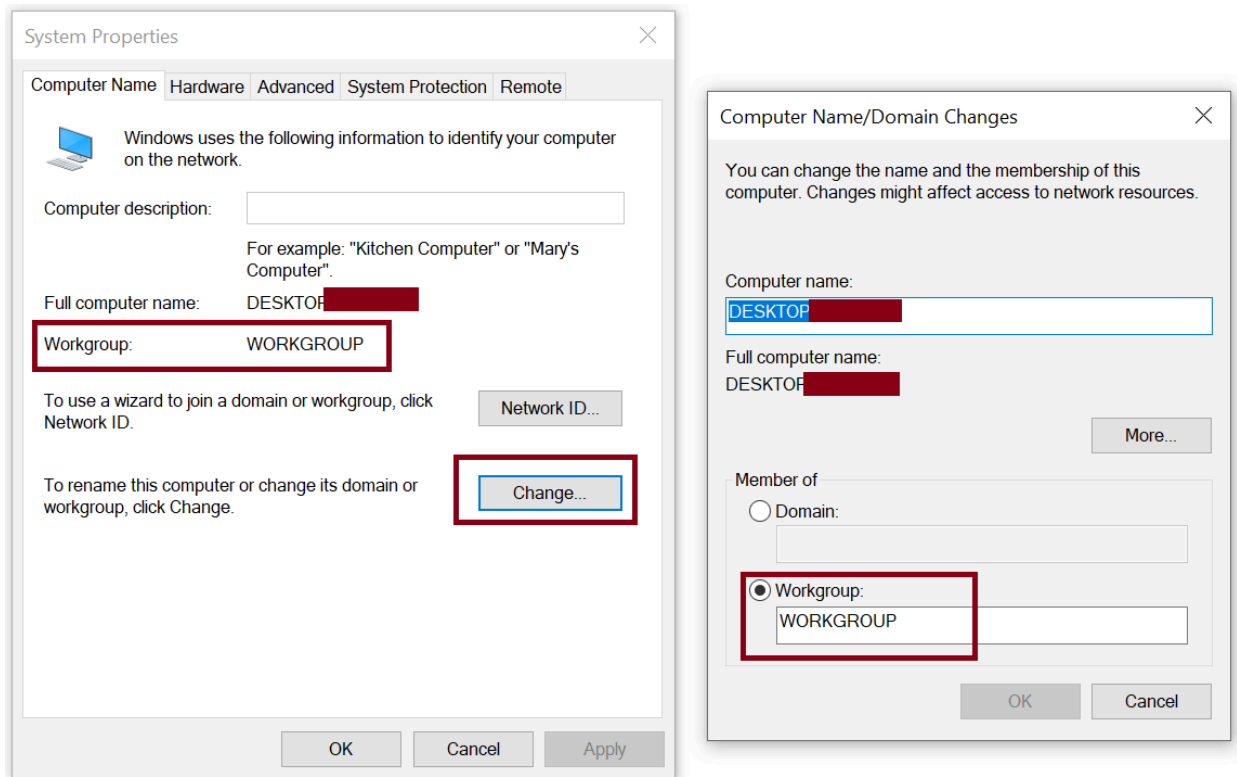[241] (https://www.slideshare.net/IgorKorkin/protected-process-light-will-be-protected-memoryranger-fills-the-gap-again

Windows Workgroup

Overall, a workgroup is a peer-to-peer network based on nodes running the Microsoft Windows operating system. We can think about it as a group of computers/servers on the LAN (Local Area Network), which don't have to go through a specific server in order to share resources[242]. Examples of such resources are: printers, files and internet connections.

Moreover, in the case of a workgroup there is no central management of users and groups. Both of them are stored locally on every system as part of the local SAM[243]. Thus, an administrator must create an account on each system if we want users to access them[244].

Lastly, each workgroup is identified by a name that we can change if we want - as shown in the screenshot below. Also, it is dependent on the SMB (Server Message Block) protocol over NetBIOS (Network Basic Input Output System) - more on them in future writeups.

---

[242] https://www.windows-active-directory.com/what-is-a-workgroup-and-how-is-it-set-up.html
[243] https://medium.com/@boutnaru/windows-security-sam-security-account-manager-c93ddadf388a
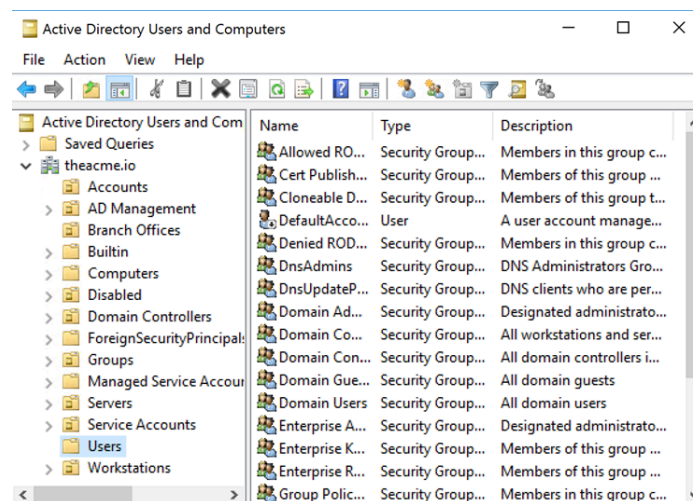[244] https://networkencyclopedia.com/windows-workgroup/

AD (Active Directory)

AD (Active Directory) is an hierarchical database which is used for storing information about different objects on the network (users, computer, printers, shares and more). AD is composed of "schema" (defines the classes of objects and attributes stored by them), "global catalog" (contains a subset of attributes of every object in the directory), "query and index mechanism" and a "replication service" for distributing the data across the network[245]. Probably the most command UI for managing AD is the "Active Directory Users and Computers" snap-in (das.msc) - as shown in the screenshot below[246].

Overall, AD provides a security boundary and is used to federate authentication (aka "domain authentication" vs local SAM based) and policy enforcement (GPO). The AD database is stored on a DC (Domain Controller) inside the "ntds.dit" file - more on that in future writeups. By the way , there are multiple best practices (patching, monitoring, recovery plans and more) for security AD as part of Microsoft's documentation[247].

Lastly, due to the fact the AD database is stored on DCs we can access it remotely using the LDAP (Lightweight Directory Access Protocol) over TCP port 389 or 636 in case of LDAPS (LDAP over SSL). LDAP supports an interface of queries in order to ask the directory service for data[248] - more on that in future writeups.

---

[245]https://learn.microsoft.com/en-us/windows-server/identity/ad-ds/get-started/virtual-dc/active-directory-domain-services-overview
[246] https://www.torchsec.org/what-is-azure-active-directory-active-directory-security/
[247]https://learn.microsoft.com/en-us/windows-server/identity/ad-ds/plan/security-best-practices/best-practices-for-securing-active-directory
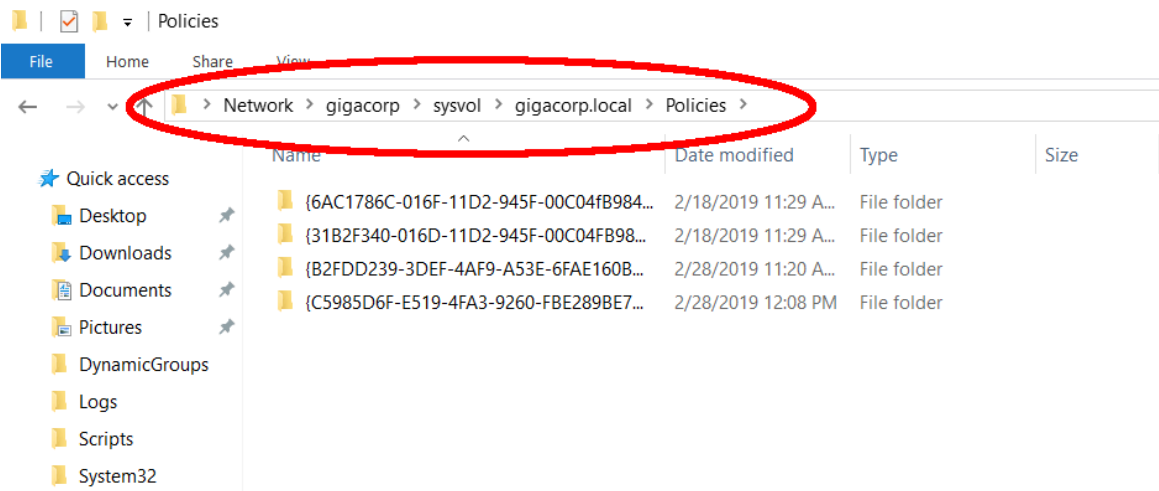[248] https://www.varonis.com/blog/the-difference-between-active-directory-and-ldap

GPO (Group Policy Object)

GPO (Group Policy Object) is a collection of policy settings (aka "Group Policy Settings"), which are represented in the file system and in the Active Directory. Each GPO has a unique name and a GUID[249]. GPO can be used for customizing settings in different areas such as: registry based policy, security options and folder redirection[250]. Policies can be assigned to a computer (like startup/shutdown scripts) or to a user (logon/logoff scripts).

Moreover, in order to configure a GPO we can use the "Group Policy Editor" which is a tool that hosts MMC extension snap-ins that manage policy settings[251]. We can link (associate) a GPO with one or more containers (site/domain/organizational unit). Thus, multiple containers can be linked to the same GPO, and a single container can have more than one GPO linked to it[252].

Lastly, a GPO is stored both in a "Group Policy Container" (GPC) and a "Group Policy Template" (GPT). GPC is an Active Directory container (located at [Domain]\System\Policies) that contains GPO properties (like version information and GPO status). GPT is a file system folder that includes policy data specified by .adm files, security settings, script files, and information about applications that are available for installation. The GPT is located in the system volume folder (sysvol) in the domain \Policies subfolder. By the way, sysvol is a network share exposed by domain controllers[253] - as shown in the screenshot below[254] - the pattern of the location is \\[Domain]\Sysvol\[Domain]\Policies.

---

[249] https://learn.microsoft.com/en-us/previous-versions/windows/desktop/policy/group-policy-objects
[250] https://www.varonis.com/blog/group-policy-objects
[251] https://learn.microsoft.com/en-us/previous-versions/windows/desktop/policy/group-policy-object-editor
[252] https://learn.microsoft.com/en-us/previous-versions/windows/desktop/policy/linking-gpos-to-active-directory-containers
[253] https://learn.microsoft.com/en-us/previous-versions/windows/desktop/policy/group-policy-storage
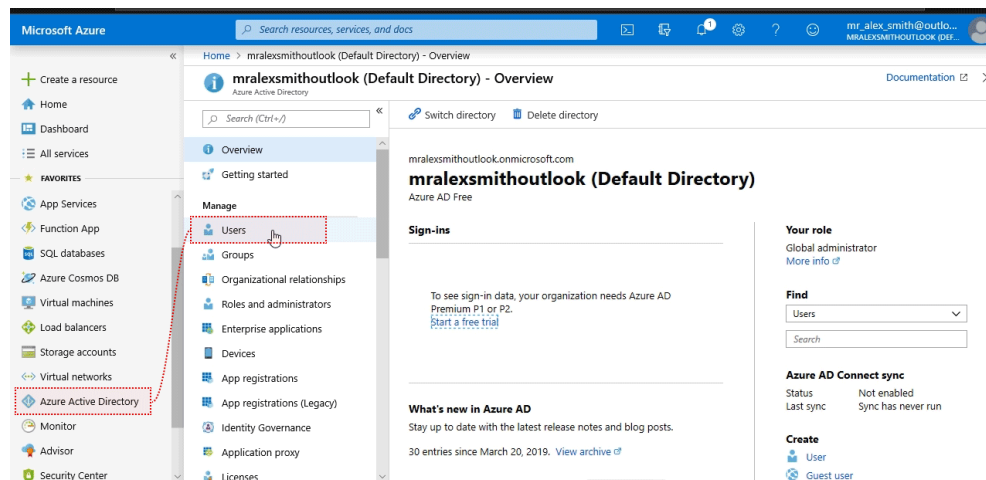[254] https://www.easy365manager.com/where-are-gpos-stored/

AAD (Azure Active Directory)

AAD (Azure Active Directory) is a cloud directory and authentication service. It is leveraged by "Office 365" and "Azure" for account/groups/role management. AAZ can act as an IdP (Identity Provider) and supports federation (like SAML and more). As of 2023 AAD is deployed in over 30 datacenters around the world leveraging "Azure Availability Zones" present[255].

Overall, AAZ is not a cloud version of AD[256]. For example AAZ does not support NTLM and/or kerberos authentication methods and uses cloud based authentication mechanisms (like OAuth2, WS-Security and SAML). Also, LDAP is not used and REST (Representational State Transfer) API is used instead for communicating with web services. AAZ can provide mobile device management with Microsoft Intune while AD does not support mobile devices[257].

Lastly, for managing AAD we can use the "Microsoft Azure Portal" - as shown in the screenshot below[258]. By the way, on 2023 the name of "Azure Active Directory" was changed to "Microsoft Entra ID"[259].

---

[255] https://www.torchsec.org/what-is-azure-active-directory-active-directory-security/

[256] https://medium.com/@boutnaru/the-windows-concept-journey-ad-active-directory-af9e795f86b4

[257] https://www.varonis.com/blog/azure-active-directory

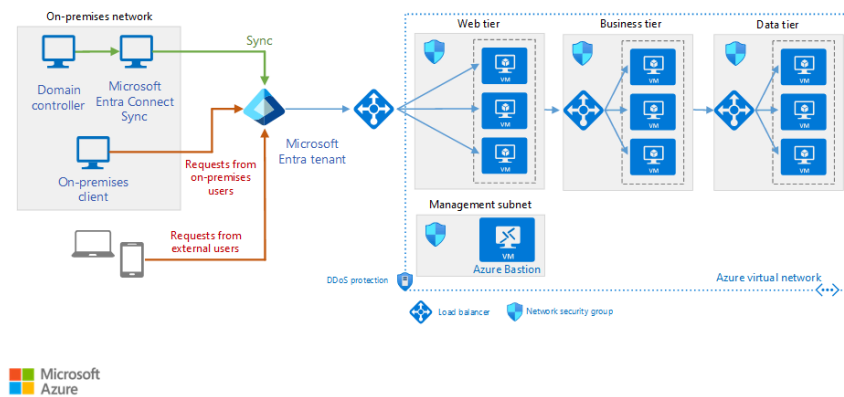[258] https://stealthmail.com/help/deployment-guide/add-user-to-azure-active-directory

[259] https://www.microsoft.com/en-us/security/blog/2023/07/11/microsoft-entra-expands-into-security-service-edge-and-azure-ad-becomes-microsoft-entra-id/

Microsoft Entra ID

"Microsoft Entra ID" is a cloud identity and access management solution which provides secure adaptive access, unified identity management and seamless user experience. Among "Microsoft Entra ID" capabilities we can find: applications integration and SSO (single sign on), unified admin center, passwordless and MFA (multi-factor authentication), conditional access, identity protection, privileged identity management and end-user self service[260].

Overall, "Microsoft Entra ID" is the new name of "Azure Active Directory"[261]. It can be used for providing access to external resources like "Azure Portal", "Office 365" and many other SaaS (Software as a Service) applications[262]. For guidelines of how to create a new tenant in "Microsoft Entra ID" we should check out Microsoft's documentation[263].

Lastly, we can sync between an on-premises "Active Directory" to "Microsoft Entra ID" (Microsoft Entra Connect Sync service). By doing so we ensure that identity information stored in the cloud is consistent with the identity information stored on-premises. The sync service is installed using "Microsoft Entra Connect" software on a virtual machine/physical host as part of the on-premises environment - as shown in the diagram below[264].

---

[260] https://www.microsoft.com/en-us/security/business/identity-access/microsoft-entra-id
[261] https://medium.com/@boutnaru/the-windows-security-journey-aad-azure-active-directory-b7e33d2fb16b
[262] https://learn.microsoft.com/en-us/entra/fundamentals/whatis
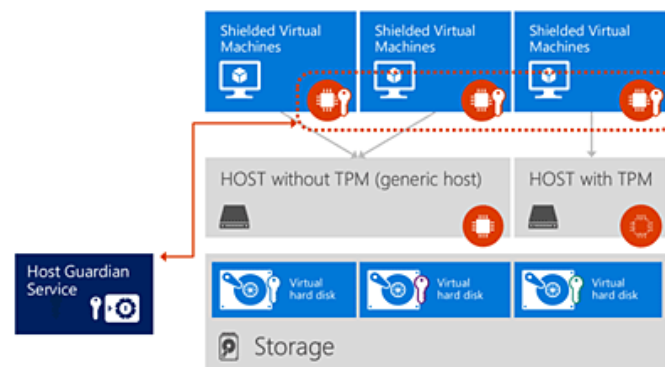[263] https://learn.microsoft.com/en-us/entra/fundamentals/create-new-tenant
[264] https://learn.microsoft.com/en-us/azure/architecture/reference-architectures/identity/azure-ad

HGS (Host Guardian Service)

HGS (Host Guardian Service) is a server role supported by the Windows operating system. It is used for configuring guarded hosts and running shielded VMs (which are VMs leveraging BitLocker for protecting their disks and state). This role is included in Windows from "Windows Server 2016". HGS provides mainly attestation and key protection services[265] - as shown in the diagram below.

Overall, the attestation service is used for validating a Hyper-V host as a "guarded host". In case of a validation which the key protection services enabled. It is used to provide the transport key required to unlock and subsequently run shielded virtual machines[266].

Lastly, to install HGS we can use the dashboard of the "Server Manager" and by selecting in the "Server Roles" menu the "Host Guardian Service", we can also use PowerShell for doing that. By the way, the security best practice (by Microsoft) states the domain admins and HGS admins should be different persons with different user accounts[267].

---

[265] https://www.webopedia.com/definitions/host-guardian-service/
[266] https://argonsys.com/microsoft-cloud/library/step-by-step-configuring-the-host-guardian-service-in-windows-server-2016/
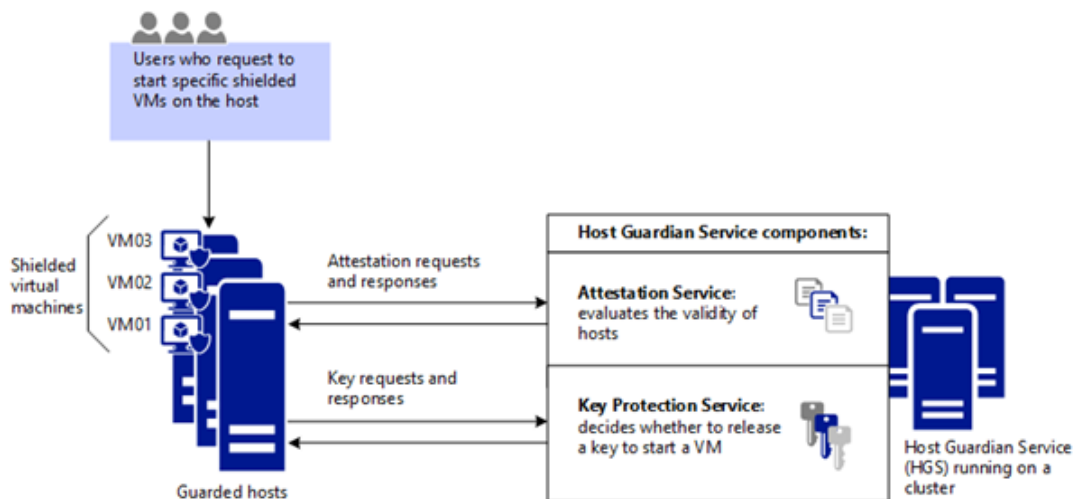[267] https://www.vladan.fr/what-is-host-guardian-service/

Guarded Host

The goal of a "Guarded Host" is to provide a trusted environment/server on which we can create and run "Shielded VMs"[268]. The trustworthiness of a "Guarded Host" is verified/attested by the "Host Guardian Service" (HGS) in order to provide a protection key for decrypting the virtual machine[269].

Overall, the "Guarded Host" is the host server on which Hyper-V is installed. Thus, virtual machine on that host can become "Shielded VMs"[270]. By the way, together with HGS and shielded virtual machines it composes a "Guarded Fabric" - as shown in the diagram below[271].

Lastly, for example a guarded host is running a "Windows 2016 Datacenter" edition of the operating system with Hyper-V installed[272]. Due to the fact "Azure Confidential Computing" and "Azure Security Center" are becoming a significant focus, Microsoft has deprecated the "Guarded Fabric" and "Shielded VMs" features as part of "Windows 2022"[273].

[268] https://medium.com/@boutnaru/the-windows-security-journey-microsoft-hyper-v-shielded-vm-2f7c86dbdf56
[269] https://medium.com/@boutnaru/the-windows-security-journey-hgs-host-guardian-service-e641ce2f8dde
[270]https://www.entrust.com/sites/default/files/documentation/integration-guides/microsoft-host-guardian-server-shielded-vm-nshield-hsm-ig.pdf
[271]https://github.com/MicrosoftDocs/windowsserverdocs/blob/main/WindowsServerDocs/security/guarded-fabric-shielded-vm/guarded-fabric-and-shielded-vms.md
[272]https://github.com/MicrosoftDocs/windowsserverdocs/blob/main/WindowsServerDocs/security/guarded-fabric-shielded-vm/guarded-fabric-deployment-overview.md
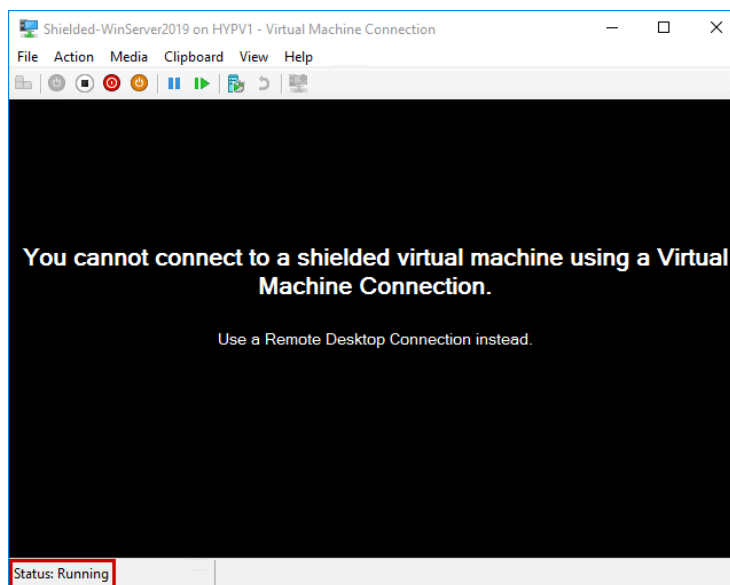[273] https://www.parallels.com/blogs/ras/windows-server-2019-vs-2022/

Microsoft Hyper-V Shielded VM

A "Hyper-V Shielded VM" is a security feature for protecting Hyper-V second generation virtual machines. This is done by leveraging different technologies such as: secure boot, TPM (Trusted Platform Module), BitLocker and "Host Guardian Service"[274].

Overall, a shielded virtual machine protects the state/data by leveraging a vTPM (Virtual Platform Module) device that enables BitLocker to encrypt the virtual machine disk/s. By the way, the vTPM device is itself encrypted with a transport key[275]. In order to protect the transport key, HGS is used.

Lasly, the goal of "Shielded VMs" is to provide protection from Hyper-V administrators and malware attacks. Shielded virtual machines are based on: code integrity, virtual secure mode (VSM), TPMv2 (Trusted Platform Module V2) and HGS (Host Guardian Service). Due to the encryption of a shielded VM VHD (Virtual Hard Disk) an administrator is not able to attach it[276]. Also, we can't connect to the video stream of a shielded VM using the "Hyper-V Manager" - as shown in the screenshot below[277]. If we want we can disable that feature (while still encrypting the disk/s). By the way, we can still use RDP (Remote Desktop Protocol) to connect to the shielded VM.

---

[274] https://www.techtarget.com/searchwindowsserver/definition/Microsoft-Hyper-V-Shielded-VM
[275] https://argonsys.com/microsoft-cloud/library/step-by-step-configuring-the-host-guardian-service-in-windows-server-2016/
[276] https://techcommunity.microsoft.com/discussions/windowsserver/introduction-to-shielded-virtual-machines-in-windows-server-2016/12753
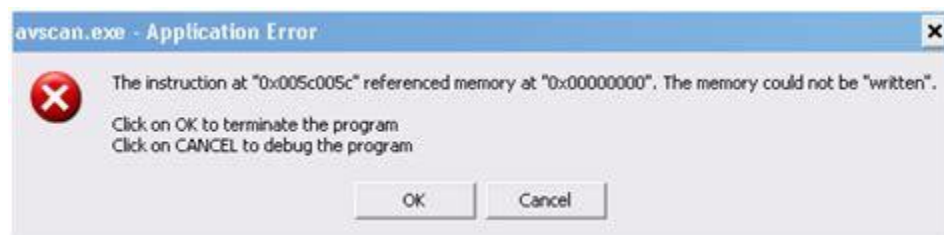[277] https://www.red-gate.com/simple-talk/sysadmin/powershell/hyper-v-and-powershell-shielded-virtual-machines/)

Null Page Mitigation

Since Windows 8, Microsoft blocks the ability to map/allocate the "Null Page" (this corresponds to the virtual address "0x00000000"). This is done for mitigating "NULL Pointer dereference" vulnerabilities. It does not matter where the bug/vulnerability exists this mitigation reduces the possibility for arbitrary code execution to a DoS (Denial of Service) at best[278]. By the way, this feature was also backported to Vista+ for x64 only[279].

Overall, a "NULL Pointer dereference" vulnerability occurs when a pointer has a NULL value and it is used as a valid pointer (for example in case of function pointers) to memory[280] - as shown in the screenshot below[281]. This feature was also supported in EMET (Enhanced Mitigation Experience Toolkit) for user-mode only. In that case the mitigation worked by allocating the first page of memory before a program starts, which of course is not[282].

Lastly, the only exception to the mitigation is by enabling NTVDM which is disabled by default[283]. Thus, enabling NTVDM decreases protection against Null dereference vulnerabilities[284].

---

[278] https://securityintelligence.com/exploitation-vulnerabilities-allow-diverting-kernel-execution-flow-windows/
[279] https://seclists.org/fulldisclosure/2014/May/112
[280] https://www.mista.nu/research/nullpage.pdf
[281] https://iso26262fs.wordpress.com/2015/06/18/null-pointer-dereference-%EB%9E%80/
[282] https://msrc.microsoft.com/blog/2009/10/announcing-the-release-of-the-enhanced-mitigation-evaluation-toolkit/
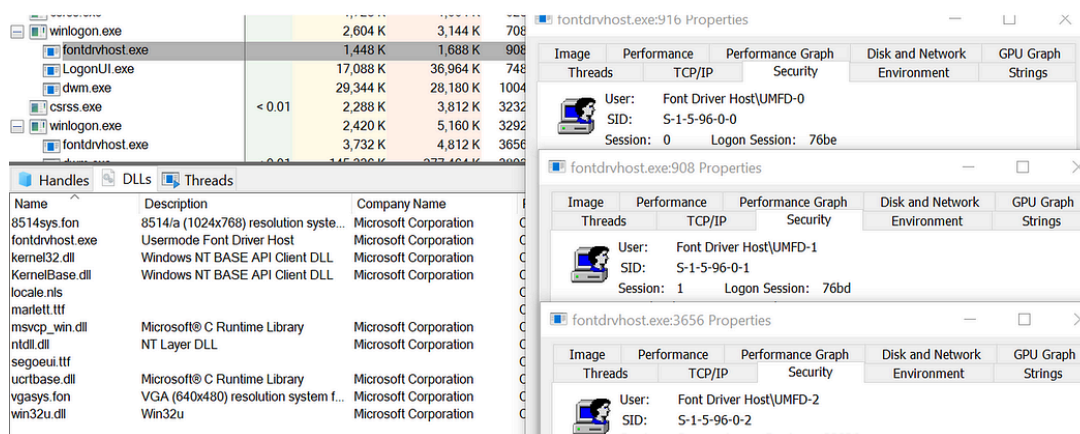[283] https://www.abatchy.com/2018/01/kernel-exploitation-6
[284] https://learn.microsoft.com/en-us/windows/security/threat-protection/overview-of-threat-mitigations-in-windows-10

UMFD (User Mode Font Driver)

The UMFD (User Mode Font Driver) security mitigation has been released as part of Windows 10. It moves font processing into user-mode as part of an "App Container"[285]. By the way, we can also block untrusted fonts from being processed[286].

Overall, the user-mode process which is responsible for the font parsing is "fontdrvhost.exe". It is executed with the permissions of a user in the following pattern: "Font Driver Host\UMFD[SessionID]". Also, the SID of the user is in the pattern of "S-1–5–96-[SessionID]"[287] - as shown in the screenshot below.

Lastly, UMFD is used for preventing local escalation of privileges and/or sandbox escapes using untrusted fonts. Also, a memory corruption in font processing won't influence the kernel directly[288].

---

[285] https://www.ghacks.net/2017/09/02/windows-10-threat-mitigation-advancements/
[286] https://learn.microsoft.com/en-us/windows/security/operating-system-security/device-management/block-untrusted-fonts-in-enterprise
[287] https://medium.com/@boutnaru/the-windows-process-journey-fontdrvhost-exe-usermode-font-driver-host-a5dd4f6159ba
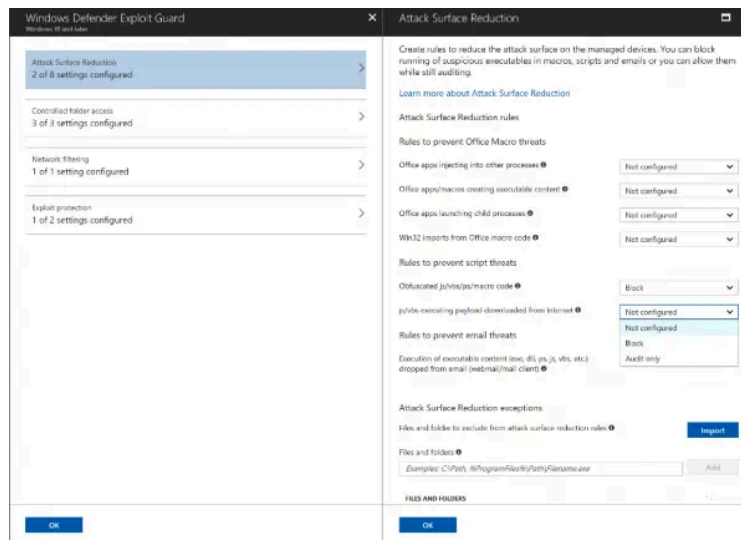[288] https://www.slideserve.com/tait/10

WDEG (Windows Defender Exploit Guard)

WDEG (Windows Defender Exploit Guard) is based on four different components: "Attack Surface Reduction" (ASR) , "Controlled Folder Access", "Exploit Protection" and "Network Protection" - as explained later in detail. It is basically an IPS (Intrusion Detection Prevention) capability part of Windows[289].   Security personnel can view events of WDEG in Windows Defender Advanced Threat Protection (WD ATP) - as shown in the screenshot below[290].

Overall, "Attack Surface Reduction" (ASR) allows enterprises blocking Office/scripts/email based threats for preventing malware infecting the device. "Controlled Folder Access" is leveraged for protecting sensitive data from ransomware by blocking untrusted apps for accessing protected folders[291]. "Exploit Mitigations" is a replacement of EMET providing a set of exploit mitigations[292].

Lastly, "Network Protection" is used for protecting the endpoint against web-based threats, this is done by blocking any outbound processes while leveraging "Windows Defender SmartScreen"[293] .By the way, WDEG leverages the "Microsoft Intelligent Security Graph (ISG)"[294].

---

[289] https://learn.microsoft.com/en-us/mem/configmgr/protect/deploy-use/create-deploy-exploit-guard-policy
[290] https://www.thewindowsclub.com/windows-defender-exploit-guard
[291] https://medium.com/@boutnaru/the-windows-security-journey-controlled-folder-access-d9972de1b9ea
[292]https://www.microsoft.com/en-us/security/blog/2017/10/23/windows-defender-exploit-guard-reduce-the-attack-surface-against-next-generation-malware/
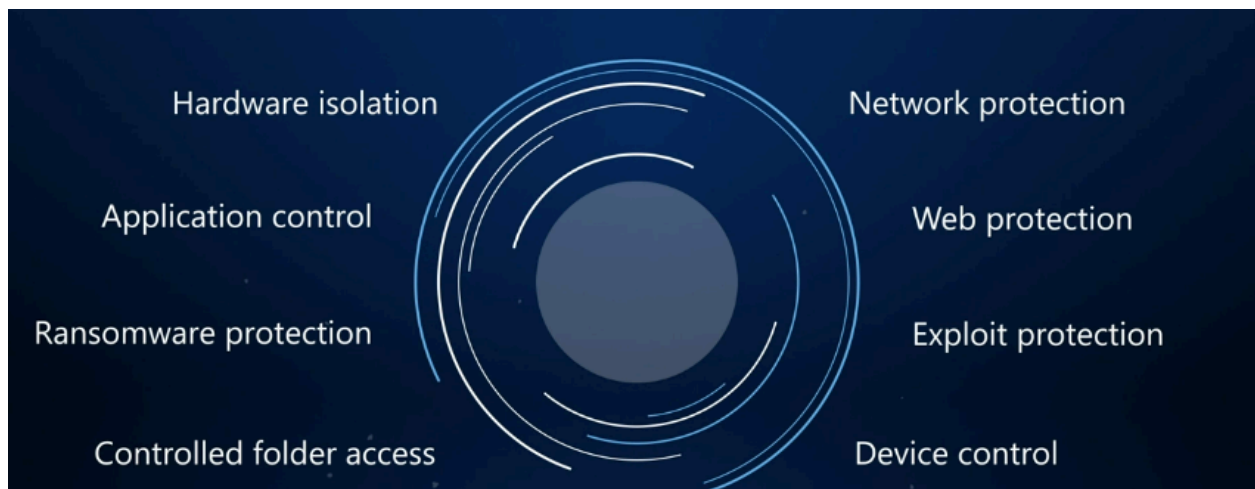[293] https://medium.com/@boutnaru/the-windows-security-journey-windows-defender-smartscreen-afdda6fb63e1
[294]https://medium.com/@boutnaru/the-windows-security-journey-microsoft-isg-microsoft-intelligent-security-graph-a32d787122b9

ASR (Attack Surface Reduction)

ASR (Attack Surface Reduction) are security features that are included as part of the Windows operating system. The goal of those features is to block risky actions on Windows based systems. Thus, help in preventing different type of ransomware attacks and malware infection[295] - as shown in the diagram below[296].

Overall, we can configure the following as part of ASR: enabling hardware-base isolation for Microsoft Edge (Microsoft Defender Application Guard), enabling attack surface reduction, enabling application control (WDAC), enabling controlled folder access[297], enabling device control, enabling exploit protection, setting up the network firewall[298], enabling Web protection and turning on network protection - more information about those features in future writeups.

Lastly, we can configure the attack surface reduction capabilities in different methods like: "Microsoft Intune", "Group Policy", "PowerShell cmdlets" and "Microsoft Configuration Manager". Also, we can set part of the features (attack surface reduction rules, exploit protection, controlled folder access and device control) to audit mode. The audited entries can be found in "Applications and Services > Microsoft > Windows > Windows Defender > Operational"[299].

---

[295] https://www.csoonline.com/article/570103/how-to-use-windows-defender-attack-surface-reduction-rules.html
[296] https://learn.microsoft.com/en-us/defender-endpoint/overview-attack-surface-reduction
[297] https://medium.com/@boutnaru/the-windows-security-journey-controlled-folder-access-d9972de1b9ea
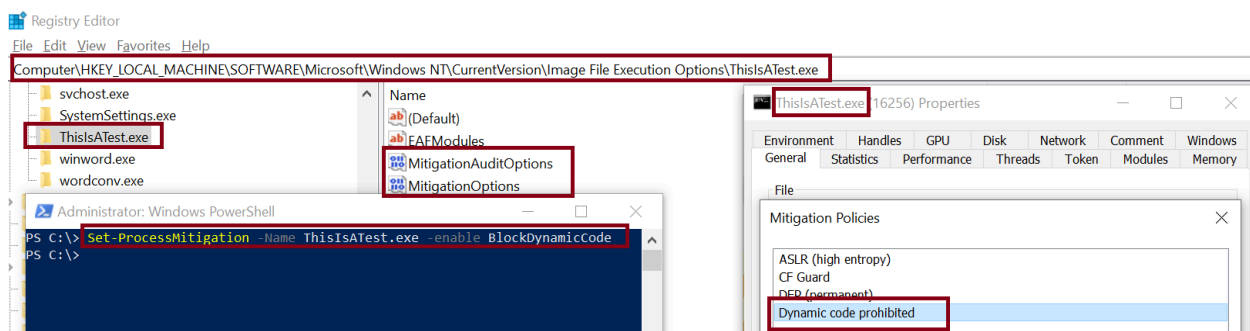[298] https://medium.com/@boutnaru/the-windows-security-journey-windows-defender-firewall-7e596b369661
[299] https://learn.microsoft.com/en-us/defender-endpoint/overview-attack-surface-reduction

ACG (Arbitrary Code Guard)

ACG (Arbitrary Code Guard) is a security mitigation included as part of the Windows operating system. It is designed for preventing/detecting that: existing code can't be modified (code pages can't become writable) and writing code to data segments and executing theme (data cannot biome code). This is done by enforcing memory from having both W (write) and X (execute) bits which is also referenced as "W^X"[300].

Overall, we can set ACG on the current process using the "SetProcessMitigationPolicy" API[301]. ACG is also called "ProcessDynamicCodePolicy". When enforcing it, code pages of the process are immutable, thus calling "VirtualProtect" with "PAGE_EXECUTE_READ_WRITE" will fail. Also, new unsigned code can't be created which means that calling "VirtualAlloc" with "PAGE_EXECUTE_READWRITE"is going to fail[302].

Lastly, we can also use the "Set-ProcessMitigation" PowerShell cmdlet[303] in order to enable ACG - as shown below. By the way, the ACG configuration is saved in the registry per process in the following location: "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options" with the "MitigationOptions" under the relevant process name[304] - as shown in the screenshot below (the right side is from SystemInformer[305]).

---

[300] https://www.countercraftsec.com/blog/arbitrary-vs-kernel/
[301] https://learn.microsoft.com/en-us/windows/win32/api/processthreadsapi/nf-processthreadsapi-setprocessmitigationpolicy
[302] https://www.ired.team/offensive-security/defense-evasion/acg-arbitrary-code-guard-processdynamiccodepolicy
[303] https://learn.microsoft.com/en-us/powershell/module/processmitigations/set-processmitigation?view=windowsserver2022-ps
[304] https://learn.microsoft.com/en-us/defender-endpoint/exploit-protection-reference
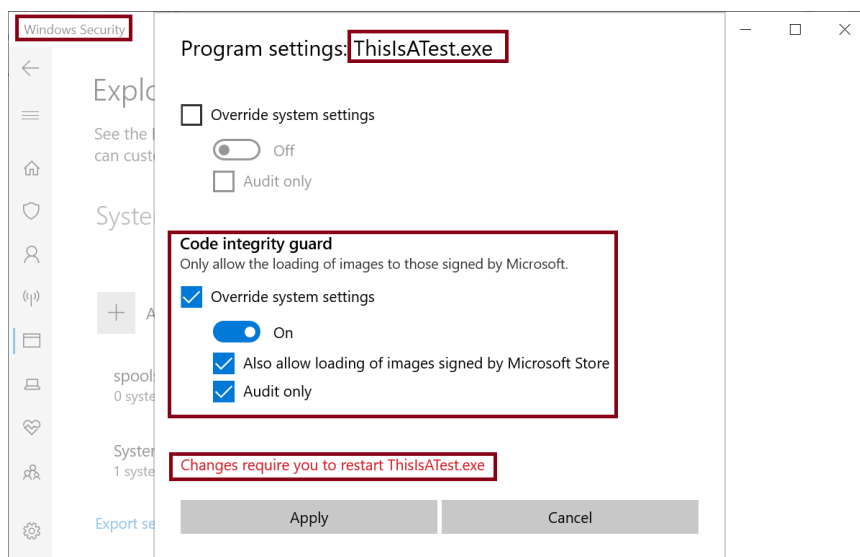[305] https://github.com/winsiderss/systeminformer

CIG (Code Integrity Guard)

CIG (Code Integrity Guard) is a security feature of the Windows operating system (that was launched in 2015 with the launch of Windows 10). It first started as part of "Device Guard" used for preventing the tamparing of OS drivers loaded by Windows 10. Later Microsoft expanded CIG to "Microsoft Edge" and after that to any application (based on the developer decision)[306].

Overall, an application with CIG enabled can only load binaries/DLLs that are signed by Microsoft. By the way, CIG is implemented as part of the memory manager of Windows. Thus, CIG is relevant for binaries loaded by the process or those injected to the process. The integrity check is done while trying to map the binary into memory. In case of a non signed binary by Microsoft, the memory manager returns the error "STATUS_INVALID_IMAGE_HASH"[307].

Lastly, we can configure (audit/blocking mode) CIG per process as part of the "Exploit Protection" menu of "System Settings" (Windows Security->App & Browser Control->Exploit Protection->Programs Settings). The settings are relevant from the next time the executable is launched - as shown in the screenshot below. Like with ACG[308] the configuration is saved in the registry per process in the following location: "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options".

---

[306] https://www.bleepingcomputer.com/news/security/cigslip-attack-bypasses-windows-code-integrity-guard-cig/
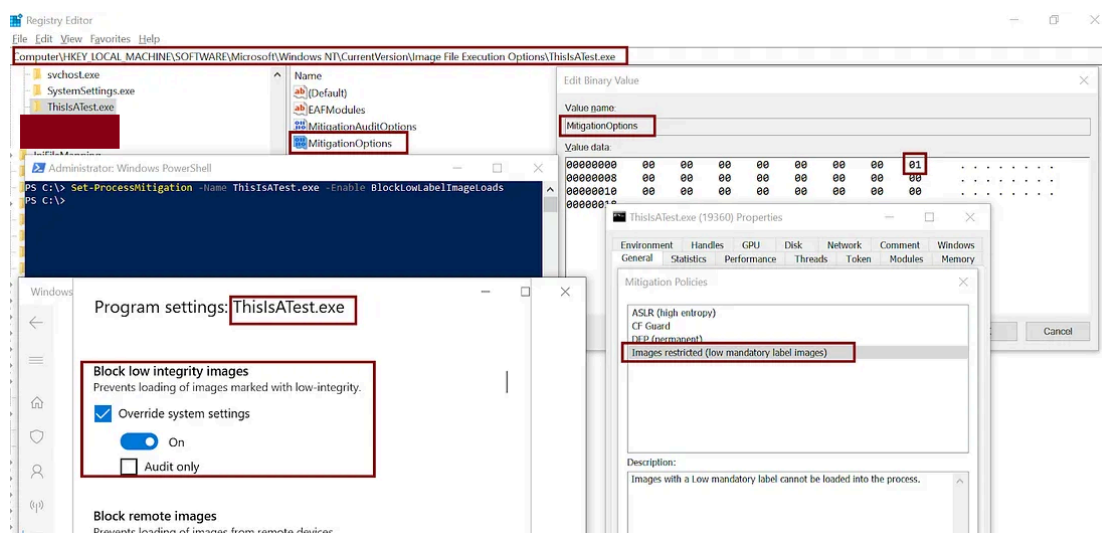[307] https://learn.microsoft.com/en-us/defender-endpoint/exploit-protection-reference
[308] https://medium.com/@boutnaru/the-windows-security-journey-acg-arbitrary-code-guard-74b08a8bd1e5

Block Low Integrity Images

"Block Low Integrity Images" is a security feature of the Windows operating system that is used in order to prevent applications from loading files that are untrusted. Think for example about files downloaded from the Internet by a sandboxed browser[309].

Overall, this security mitigation is used for blocking the loading of images that are marked with a "Low Integrity Level" label[310]. Like with CIG[311] also "Block Low Integrity Images" is handled by the memory manager in the flow of mapping a file into the memory of the system. We can configure this mitigation (audit/block) per process as part of the "Exploit Protection" menu of "System Settings" (Windows Security->App & Browser Control->Exploit Protection->Programs Settings) - as shown in the screenshot below.

Lastly, we can also use the "Set-ProcessMitigation" PowerShell cmdlet[312] in order to enable "Block Low Integrity Images"- as shown below. By the way, the as with other mitigations (like ACG) the configuration is saved in the registry per process in the following location: "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options" with the "MitigationOptions" under the relevant process name[313] - as shown in the screenshot below (the right side is from SystemInformer[314]).

---

[309] https://learn.microsoft.com/en-us/defender-endpoint/exploit-protection-reference#block-low-integrity-images
[310] https://medium.com/@boutnaru/the-windows-security-journey-mandatory-integrity-control-mic-f7963550c0e7
[311] https://medium.com/@boutnaru/the-windows-security-journey-cig-code-integrity-guard-7e410c8d2304
[312] https://learn.microsoft.com/en-us/powershell/module/processmitigations/set-processmitigation
[313] https://medium.com/@boutnaru/the-windows-security-journey-acg-arbitrary-code-guard-74b08a8bd1e5
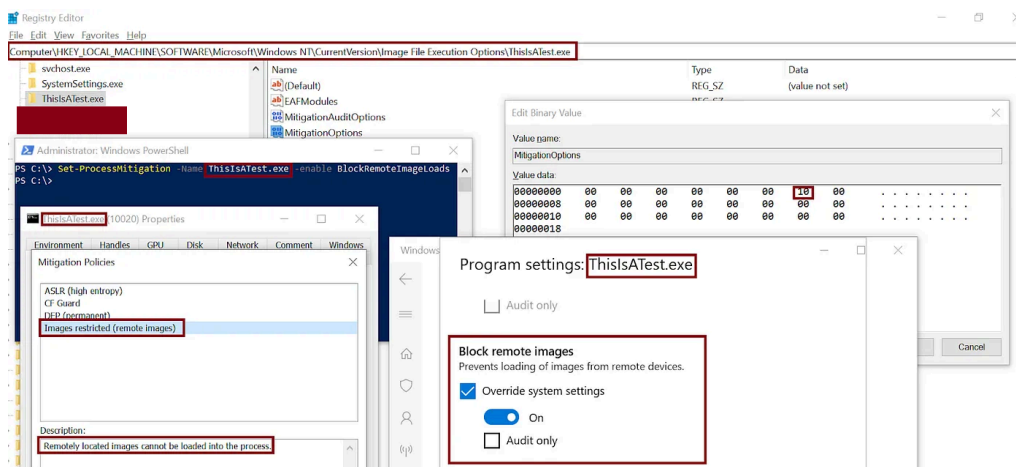[314] https://github.com/winsiderss/systeminformer

Block Remote Images

"Block Remote Images" is a security feature of the Windows operating system that is used for preventing applications from loading files stored on remote systems (think about UNC shares). By doing so it protects against loading binaries into the memory address space of processes from external devices which are controlled by attackers. In case an application attempts to map a remote file, a "STATUS_ACCESS_DENIED" error is returned[315].

Overall, like with "Block Low Integrity Level"[316] also "Block Remote Images" is handled by the memory manager in the flow of mapping a file into the memory of the system. We can configure this mitigation (audit/block) per process as part of the "Exploit Protection" menu of "System Settings" (Windows Security->App & Browser Control->Exploit Protection->Programs Settings) - as shown in the screenshot below.

Lastly, we can also use the "Set-ProcessMitigation" PowerShell cmdlet[317] in order to enable "Block Remote Images"- as shown below (Set-ProcessMitigation -Name <ProcessName> -enable BlockRemoteImageLoads). By the way, the as with other mitigations (like ACG) the configuration is saved in the registry per process in the following location: "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options" with the "MitigationOptions" under the relevant process name[318] - as shown in the screenshot below (the left side is from SystemInformer[319]).

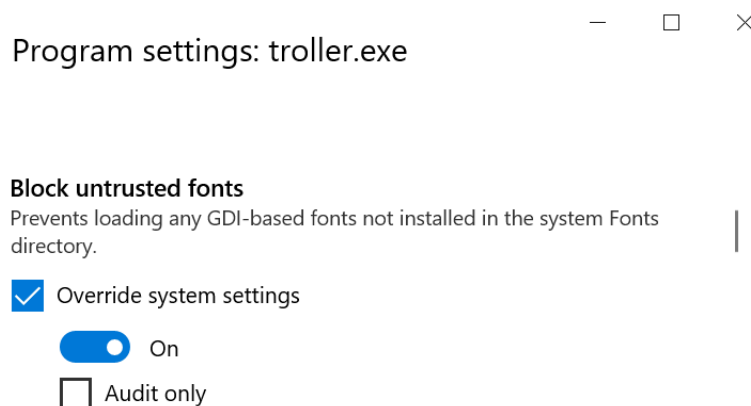[315] https://learn.microsoft.com/en-us/defender-endpoint/exploit-protection-reference#block-remote-images
[316] https://medium.com/@boutnaru/the-windows-security-journey-block-low-integrity-images-cd1fabe641cd
[317] https://learn.microsoft.com/en-us/powershell/module/processmitigations/set-processmitigation
[318] https://medium.com/@boutnaru/the-windows-security-journey-acg-arbitrary-code-guard-74b08a8bd1e5
[319] https://github.com/winsiderss/systeminformer

Block Untrusted Fonts

"Block Untrusted Fonts" is a security feature of the Windows operating system that is used for preventing applications from loading arbitrary font files. By default this feature is not turned on. There are three options of configuration: "On" (stops loading fonts outside the %windir%\Fonts directory and turns on event logging), "Audit" (turns on event logging, without blocking fonts from loading) and "Exclude apps to load untrusted fonts"[320].

Overall, this security mitigation is used for blocking the load of GDI based fonts not installed in the system fonts directory. Thus, using it can reduce functionality in different use cases such as: applications using memory based fonts, documents which have embedded fonts, using "Internet Explorer" to view websites which embedded fonts, sending documents for printing with fonts not stored on the printing server and more[321].

Lastly, we can configure "Block Untrusted Fonts" per process as part of the "Exploit Protection" menu of "System Settings" (Windows Security->App & Browser Control->Exploit Protection->Programs Settings). The settings are relevant from the next time the executable is launched - as shown in the screenshot below. Like with ACG[322] the configuration is saved in the registry per process in the following location: "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options".

---

[320]https://learn.microsoft.com/en-us/windows/security/operating-system-security/device-management/block-untrusted-fonts-in-enterprise
[321] https://learn.microsoft.com/en-us/troubleshoot/windows-client/shell-experience/feature-to-block-untrusted-fonts
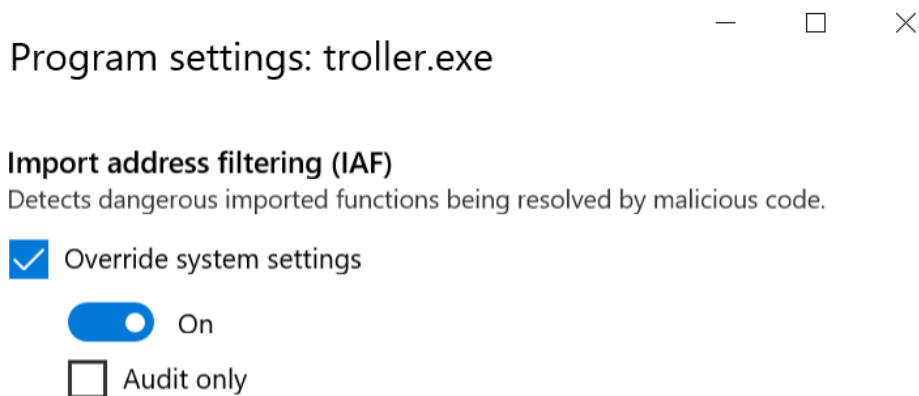[322] https://medium.com/@boutnaru/the-windows-security-journey-acg-arbitrary-code-guard-74b08a8bd1e5

IAF (Import Address Filtering)

IAF (Import Address Filtering) is a security mitigation used for preventing code not backed by an image on disk to lookup addresses from the IAT (Import Address Table). Due to that, IAF is also referred to as "Import Address Table Access Filtering"[323]. It is implemented by marking the relevant pages as "PAGE_GUARD" and validating the above condition as part of the exception handler. We can configure this mitigation (audit/block) per process as part of the "Exploit Protection" menu of "System Settings" (Windows Security->App & Browser Control->Exploit Protection->Programs Settings) - as shown in the screenshot below.

Overall, as with other mitigations IAF had originally implemented as part of EMET and later added to "Windows Defender"[324]. Among the APIs protected by IAF we can find: "GetProcAddress", "GetProcAddressForCaller", "LoadLibraryA", "LoadLibraryExA", "LoadLibraryW", "LoadLibraryExW", "VirtualProtect", "VirtualProtectEx", "WinExec", "CreateProcessAsUserA", "CreateProcessAsUserW", "CreateProcessA", "CreateProcessW" and more[325].

Lastly, it is important to know that .NET 2.0 is not compatible with "Import Address Filtering". Thus, if we want to execute .NET 2.0 code we can't leverage "Import Address Filtering"[326]. Also, we can use the "Set-Processmitigation" PowerShell cmdlet for setting the IAF (using the "EnableImportAddressFilter" mitigation parameter keyword) mitigation[327]. By the way, the as with other mitigations (like ACG) the configuration is saved in the registry per process in the following location: "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options" under the relevant process name[328].

Program settings: troller.exe

Import address filtering (IAF)
Detects dangerous imported functions being resolved by malicious code.

☑ Override system settings

🔵 On

☐ Audit only

[323] https://i.blackhat.com/USA21/Wednesday-Handouts/us-21-Rope-Bypassing-Behavioral-Detection-Of-Malware-With-Distributed-ROP-Driven-Execution-wp.pdf
[324] https://github.com/yardenshafir/MitigationFlagsCliTool
[325] https://learn.microsoft.com/en-us/defender-endpoint/exploit-protection-reference)
[326] https://learn.microsoft.com/en-us/defender-endpoint/enable-exploit-protection
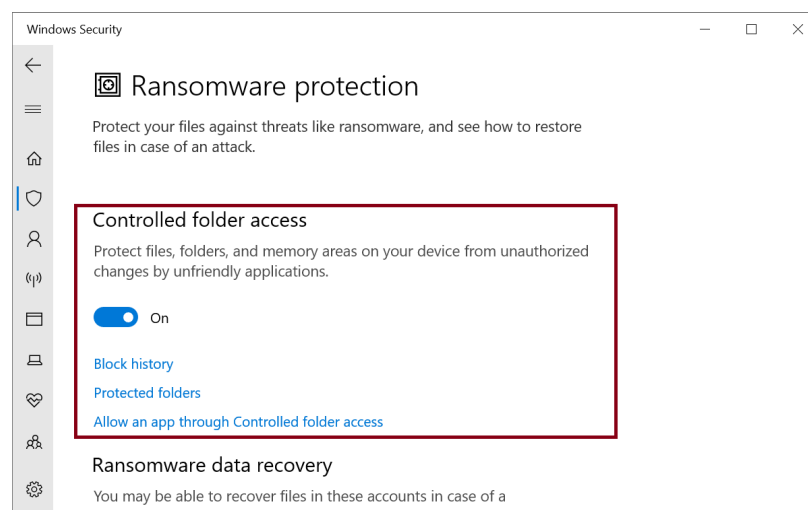[327] https://learn.microsoft.com/en-us/powershell/module/processmitigations/set-processmitigation
[328] https://medium.com/@boutnaru/the-windows-security-journey-acg-arbitrary-code-guard-74b08a8bd1e5

Controlled Folder Access

"Controlled Folder Access" is a ransomware protection feature included as part of the Windows operating system (since Windows 10). It is used for blocking suspicious programs from writing to protected folders. In order for it to work it requires "Windows Defender AV" but it works best with "Windows Defender ATP". Example for enabling it are (but not limited to): "System Settings->Windows Security->Ransomware Protection" (as shown in the screenshot below), GPO ("Windows components->Windows Defender Antivirus->Windows Defender Exploit Guard->Controlled Folder Access") or by leveraging PowerShell with the command "Set-MpPreference -EnableControlledFolderAccess Enabled"[329].

Overall, "Controlled Folder Access" operates based on a list of trusted applications. Applications which are part of the list work as expected while applications that are not part of the list are blocked/prevented from making any changes to files located at protected folders. Applications are added to the list based on their prevalence and/or reputation. If an application is prevalent and has never displayed any behavior deemed malicious it is added automatically to the list. Of course we can also add applications manually to the list[330].

Lastly, the list of the protected folders by default contains the following: "%userprofile%\Documents", "c:\Users\Public\Documents", "%userprofile%\Pictures", "c:\Users\Public\Pictures", "c:\Users\Public\Videos", "%userprofile%\Videos", "%userprofile%\Music", "c:\Users\Public\Music" and "%userprofile%\Favorites"[331].

---

[329] https://slideplayer.com/slide/13416611/
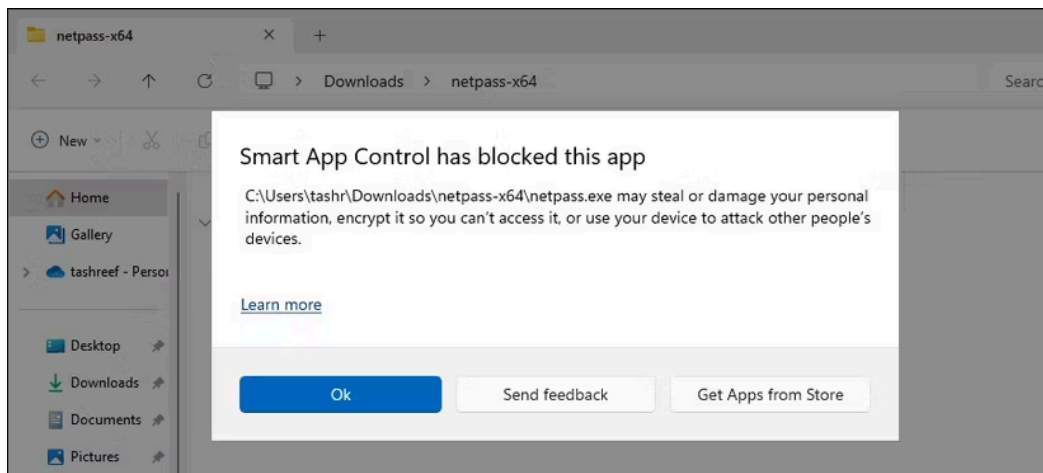[330] https://learn.microsoft.com/en-us/defender-endpoint/controlled-folders
[331] https://wiki.nexusmods.com/index.php/Controlled_Folder_Access

SAC (Smart App Control)

SAC (Smart App Control) is a security feature part of the Windows operating system which is used for protecting against untrustworthy applications. Thus, when trying to execute an application on Windows its credibility is being analyzed by SCA. This is done while leveraging the Microsoft's Intelligent Security Graph[332].

Overall, if SCA is not able to make a confident prediction, it then checks the application for a valid signature (that identifies where the file originates from and if it's a trusted source). In case the application is from a known publisher or with a valid signature it will execute (if not it is blocked - as shown in the screenshot below). By the way, there's no manual override option, nor can you add an app to an exemption list[333].

Lastly, smart app control is only available on clean installations of Windows 11. It is started in evaluation mode, which can determine that the specific device is not a good candidate for SAC. It is important to understand that we have to enable "optional diagnostic data" in Windows for SCA to work. The settings of SCA can be found in the "App & Browser Security" panel of the Windows security app[334].

---

[332]https://medium.com/@boutnaru/the-windows-security-journey-microsoft-isg-microsoft-intelligent-security-graph-a32d787122b9

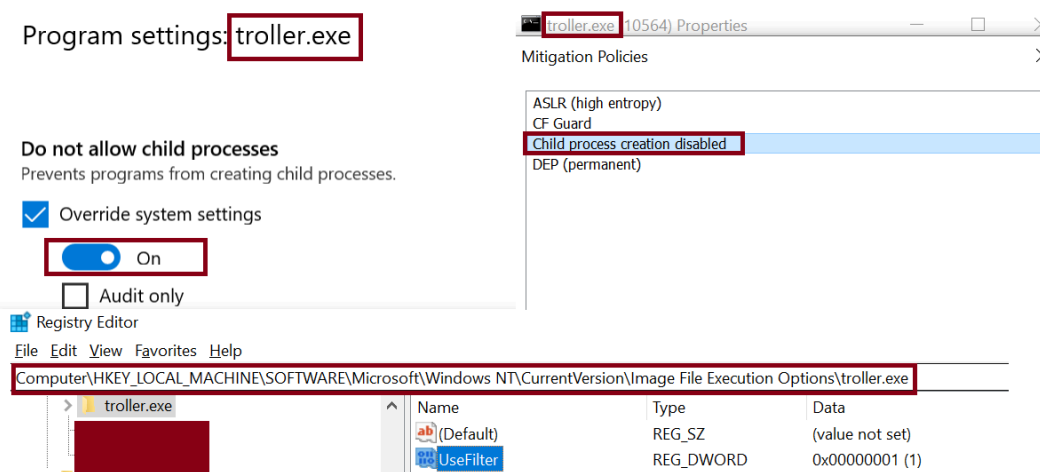[333] https://www.howtogeek.com/smart-app-control-windows-11-explained/

[334] https://support.microsoft.com/en-gb/topic/what-is-smart-app-control-285ea03d-fa88-4d56-882e-6698afdb7003

No Child Process Mitigation

The "No Child Process" security mitigation has been designed to block code execution by preventing the launching of child processes[335]. The mitigation is enforced as a flag in the token of the process, we can check for the flag using the "SeTokenIsNoChildProcessRestrictionEnforced" kernel function[336]. The mitigation blocks the creation of a token for a child process (and thus a creation of a process) while returning "STATUS_CHILD_PROCESS_BLOCKED"[337].

Overall, one of the common techniques leveraged by attacks is to initiate a trusted process with malicious parameters/input (aka "living off the land"). This causes the launching/starting of another executable /application by using the "No Child Process" mitigation to prevent (we can also use "audit mode" only if we want) this attack vector[338]. By the way, the as with other mitigations (like ACG) the configuration is saved in the registry per process in the following location: "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options" with the "UseFilter" under the relevant process name[339] - as shown in the screenshot below (the left side is from SystemInformer[340]).

Lastly, this mitigation is used for example as part of "Microsoft Edge". It is set for the content processes which ensures that a child process cannot be created. By having the mitigation set as part of the token it ensures both direct (think about calling WinExec) and indirect (for example out-of-process COM server) process launches are blocked[341].


---

[335] https://www.ghacks.net/2017/09/02/windows-10-threat-mitigation-advancements/
[336] https://learn.microsoft.com/en-us/windows-hardware/drivers/ddi/ntifs/nf-ntifs-setokenisnochildprocessrestrictionenforced
[337] https://codemachine.com/downloads/win10.1709/ntstatus.h
[338] https://learn.microsoft.com/en-us/defender-endpoint/exploit-protection-reference
[339] https://medium.com/@boutnaru/the-windows-security-journey-acg-arbitrary-code-guard-74b08a8bd1e5
[340] https://github.com/winsiderss/systeminformer
[341] https://blogs.windows.com/msedgedev/2017/02/23/mitigating-arbitrary-native-code-execution/

Windows Hello

In general, "Windows Hello" is a more secure and personal way to access Windows devices. It provides the ability to use PIN code, face recognition or fingerprint authentication. When setting up a fingerprint/facial recognition we will need to set up a PIN also. After that we can sign in with just the PIN code. This is safer because the PIN code is only associated with one device and is not back up for recovery using a Microsoft account[342].

Overall, by using Windows it eliminates the need for passwords. Thus, it is a new way to sign into devices, apps, online services and networks. Even if our Windows 10/11 device supports Windows Hello we are not required to use it. For facial recognition it requires a specialized illuminated infrared camera or a fingerprint reader which supports the Window-Biometric Framework[343].

Lastly, "Windows Hello" provides the ability to go passwordless on Windows based devices. We just need to configure "Require Windows Hello sign-in for Microsoft Accounts". To do so go to "WinKey->Settings->Accounts->Sign-in Options" - as shown in the screenshot below[344].

---
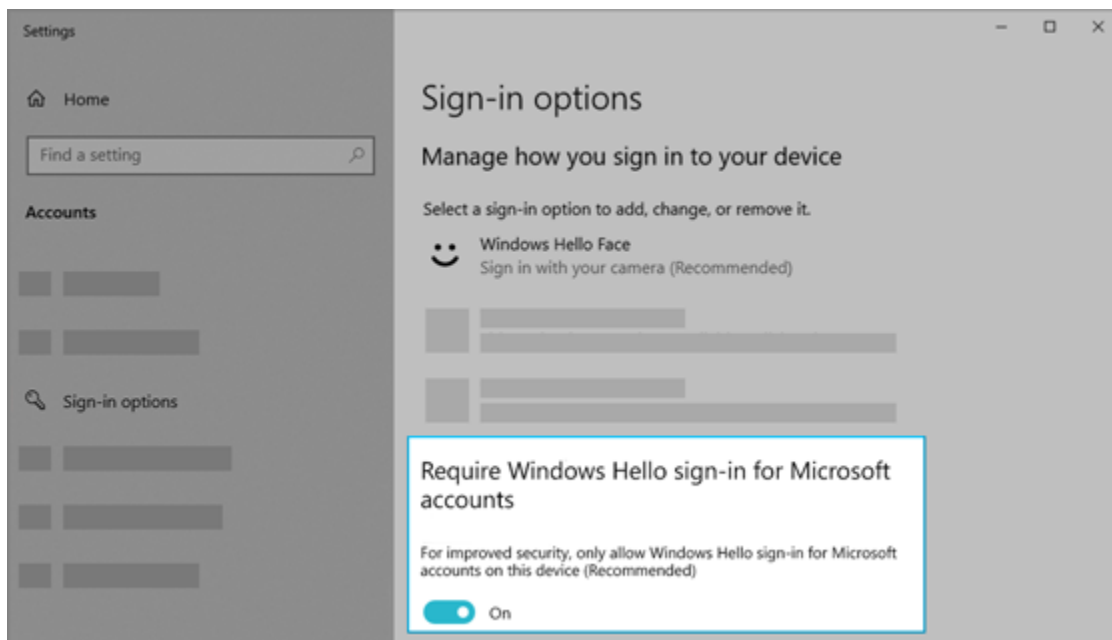
[342]https://support.microsoft.com/en-us/windows/learn-about-windows-hello-and-set-it-up-dae28983-8242-bb2a-d3d1-87c9d265a5f0#WindowsVersion=Windows_11
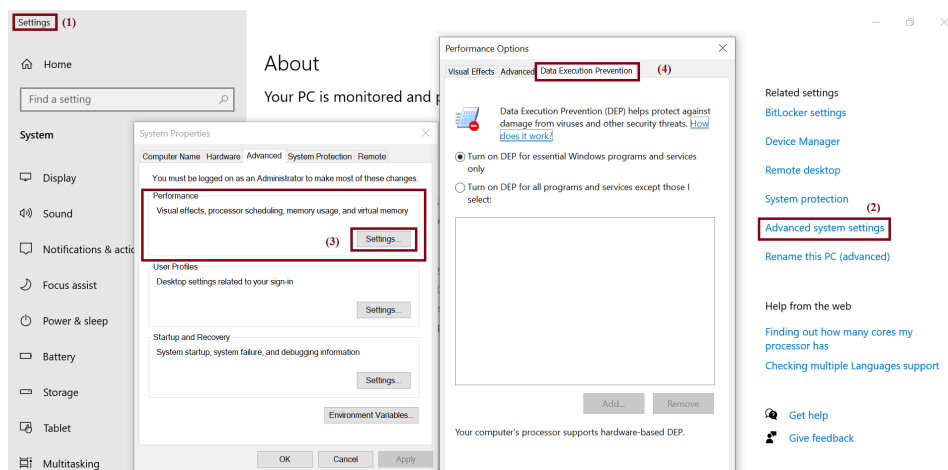
[343] https://www.dell.com/support/kbdoc/en-us/000148929/what-is-windows-hello

[344] https://support.microsoft.com/en-us/windows/go-passwordless-on-your-device-5035693d-8f87-c5d5-d16c-b2f75d37bb37

DEP (Data Execution Prevention)

DEP (Data Execution Prevention) is a security feature built into the Windows operating system (since Windows XP/2003). It allows marking memory pages as non-executable, this means the code cannot be run from those regions of memory (for example the stack and the heap). Thus, making it harder to exploit security bugs such as buffer overflow (of course this is not the only feature for that). DEP is basically the implementation of the OS side required for completing the support for "NX bit" in the CPU[345]. For configuring DEP on Windows we can go to "Settings"-> "Advanced System Settings" -> "Performance" (Settings)->"Data Execution Prevention" - as shown in the screenshot below.

Overall, if an application tries to jump/call an address which resides in a protected page an exception of "STATUS_ACCESS_VIOLATION" is raised. To avoid that we need to allocate memory using one of the following (using the VirtualAlloc/VirtualAllocEx API function) flags: "PAGE_EXECUTE", "PAGE_EXECUTE_READ", "PAGE_EXECUTE_READWRITE" or "PAGE_EXECUTE_WRITECOPY"[346].

Moreover, a process can get its current DEP policy using the "GetSystemDEPPolicy" function call the return value can be: "0" (always off), "1" (always on), "2" (opt in) or "3" (opt out)[347]. In order to enable/disable DEP for a specific process we can call the "SetProcessDEPPolicy" API function[348]. Lastly, we can also change the protection of a the current process using "VirtualProtect"[349] or even any other process (based on permissions) using "VirtualProtectEx"[350]. By the way there is also a software implementation of DEP which we have not focused on[351].

---

[345] https://medium.com/@boutnaru/security-nx-bit-non-executable-18759fd2802e
[346] https://learn.microsoft.com/en-us/windows/win32/Memory/memory-protection-constants
[347] https://learn.microsoft.com/en-us/windows/win32/api/WinBase/nf-winbase-getsystemdeppolicy
[348] https://learn.microsoft.com/en-us/windows/win32/api/WinBase/nf-winbase-setprocessdeppolicy
[349] https://learn.microsoft.com/en-us/windows/win32/api/memoryapi/nf-memoryapi-virtualprotect
[350] https://learn.microsoft.com/en-us/windows/win32/api/memoryapi/nf-memoryapi-virtualprotectex
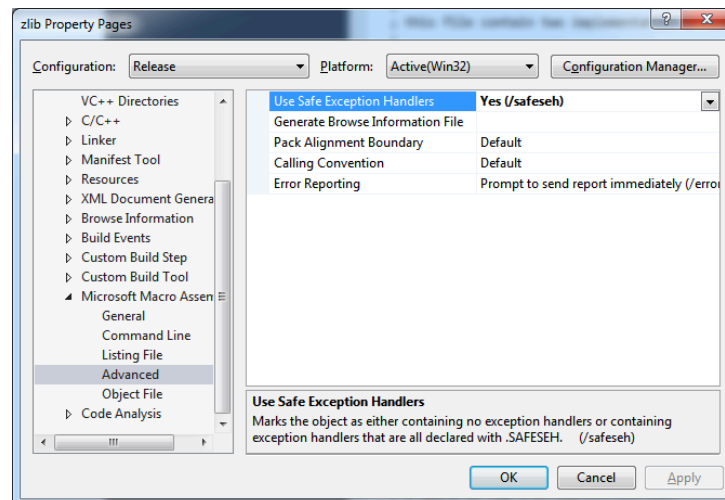[351] https://learn.microsoft.com/en-us/windows/win32/memory/data-execution-prevention

SafeSEH (Safe Structured Exception Handling)

The goal of the SafeSEH (Safe Structured Exception Handling) mechanism is to protect stack-based SEH[352] chains from overwriting. It is important to know that in case of x86/Itanium/ARM architectures the exception handlers are already stored in a table as one of the PE sections (i.e. PDATA). Thus, it can't be overwritten directly as part of a stack overflow[353].

Overall, for x86 code projects in case we use "Visual Studio" we can provide the "/SAFESEH" flag. This causes a creation of a PE image with a table of safe exception handlers[354]. We can configure it as part of the "Microsoft Macro Assembler" settings as shown in the screenshot below[355].

Lastly, this means that if the flow of an application is changed within the exception handler chain and the address of the execution handler is not within such a table the application is terminated. By doing so, SafeSEH prevents the code execution through an exploited exception handler chain. As with other security mechanisms also SafeSEH can be paypassed using techniques like: finding a module which is not protected by SafeSEH and/or using addresses outside the range of the loaded module/the application image which are consider safe by SafeSEH[356].

---

[352] https://medium.com/@boutnaru/the-windows-concept-journey-seh-structured-exception-handling-10458086118a
[353] https://security.stackexchange.com/questions/23315/safeseh-and-x64
[354] https://learn.microsoft.com/en-us/cpp/build/reference/safeseh-image-has-safe-exception-handlers?view=msvc-170
[355] https://www.gangofcoders.net/solution/error-lnk2026-module-unsafe-for-safeseh-image/
[356] https://www.rcesecurity.com/2012/11/bypassing-safeseh-memory-protection-in-zoner-photo-studio-v15/
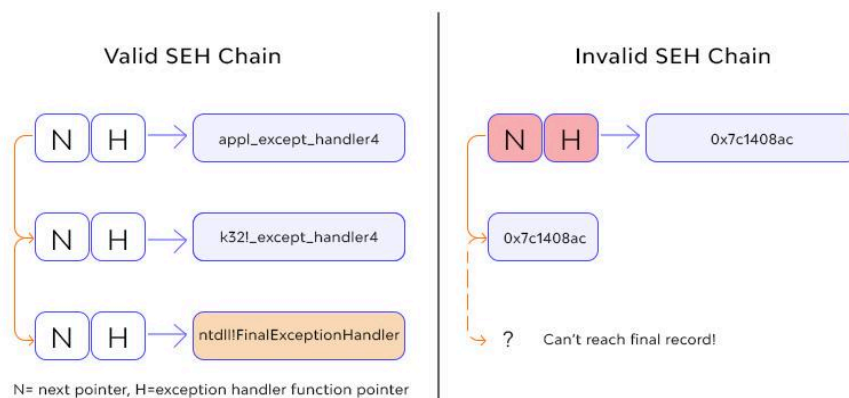
SEHOP (Structured Exception Handling Overwrite Protection)

The goal of the SEHOP (Structured Exception Handling Overwrite Protection) mechanism is to protect stack-based exception chains from overwriting. It is important to know that in case of x64/Itanium architectures the exception handlers are stored in a different PE section[357] (by default it is called "PDATA").

Overall, SEHOP verifies the integrity of SEH[358] chains before executing exception handling code. This mechanism works at runtime regardless if we use SafeSEH[359] or not. SafeSEH is a build time solution (the support is added by the linker).

Moreover, SEHOP is based on the assumption that an SEH overwrite has side effects. When used (in most of stack-based buffer overflows) the next pointer of the exception registration is overwritten. Since the next pointer is corrupted the it invalidates the integrity exception chain[360] - as shown in the diagram below[361].

Lastly, SEOP adds a dummy record at the top of the SEH list. In case an exception exception is triggered the dispatcher walks up the list and verifies that the dummy record is valid (was not changed), if not the process is terminated[362].

Valid SEH Chain / Invalid SEH Chain. N= next pointer, H=exception handler function pointer

[357] https://security.stackexchange.com/questions/23315/safeseh-and-x64
[358] https://medium.com/@boutnaru/the-windows-concept-journey-seh-structured-exception-handling-10458086118a
[359] https://medium.com/@boutnaru/the-windows-security-journey-safeseh-safe-structured-exception-handling-835b6b938468
[360] https://msrc.microsoft.com/blog/2009/02/preventing-the-exploitation-of-structured-exception-handler-seh-overwrites-with-sehop
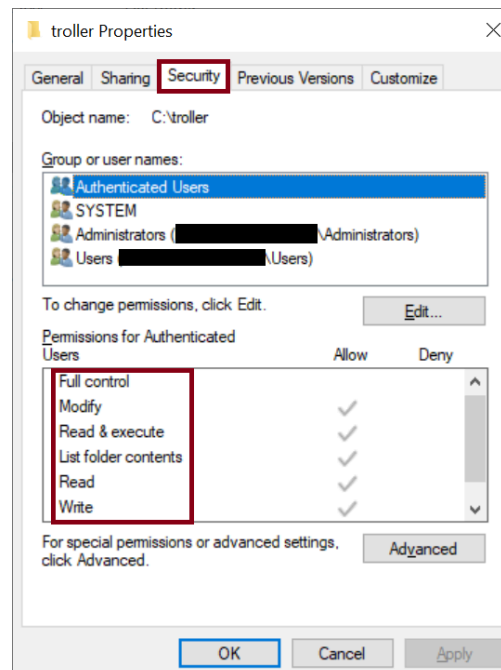[361] https://www.wallarm.com/what/buffer-overflow-attack-preventing-and-mitigation-methods-part-2
[362] https://www.slideserve.com/christyrose/control-hijacking-defenses-powerpoint-ppt-presentation

NTFS (New Technology File System) Permissions

NTFS (New Technology File System) is the default file system used on Windows based devices[363]. NTFS supports different features like: compression, quotas, ads, journaling and more. Here we are going to focus on the ability of setting permissions to files/folders.

Overall, we can group NTFS permissions to the following categories: "Full Control", "Modify", "Read & Execute", "List folder contents", "Read" and "Write" - as shown in the screenshot below. All of those are composed of a specific set of advanced (special) permissions[364].

Lastly, for each file/folder we have a security descriptor[365] which holds the access control list relevant for permissions aka DACL[366] - as seen in the screenshot below. We can also see that information using CLI tools like "cacls.exe"[367] and "icacls.exe"[368].

---

[363] https://medium.com/@boutnaru/the-windows-concept-journey-ntfs-new-technology-file-system-433e27a2256a
[364] https://www.permissionsreporter.com/ntfs-permissions
[365] https://medium.com/@boutnaru/windows-security-security-descriptor-sd-ba95b8fa048a
[366] https://medium.com/@boutnaru/the-windows-security-journey-dacl-discretionary-access-control-list-c74545e472ec
[367] https://medium.com/@boutnaru/the-windows-process-journey-cacls-exe-control-acls-program-296ba9e7761c
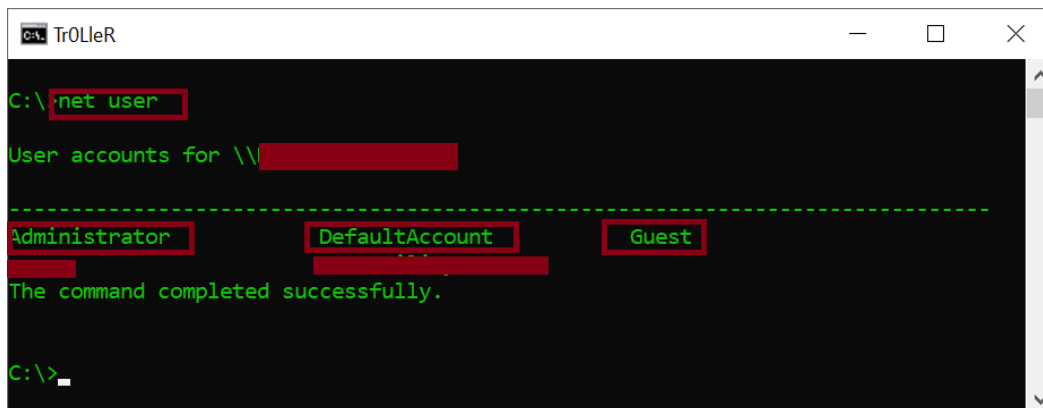[368] https://ss64.com/nt/icacls.html

Local User Account

In general, a "Local User Account" is a Windows account which was created on the local device. This type of account can only logon to a specific device (the local machine) and due to that is used for managing/securing resources on a specific device or as service users[369].

Moreover, the information regarding local user accounts on Windows is stored as part of the SAM[370] (Security Account Manager) which itself is part of the registry. Also, Windows has default local accounts that are created when the operating system is installed. Examples of such users are: Administrator, Guest, DefaultAccount, Local System, Network Service, Local Service[371].

Lastly, we can also list the local users accounts of a specific computer using the "user" argument of the "net.exe" command line utility - as shown in the screenshot below[372]. It is important to note that there are also local groups that can be created.

[369] https://learn.microsoft.com/en-us/windows/security/identity-protection/access-control/local-accounts
[370] https://medium.com/@boutnaru/windows-security-sam-security-account-manager-c93ddadf388a
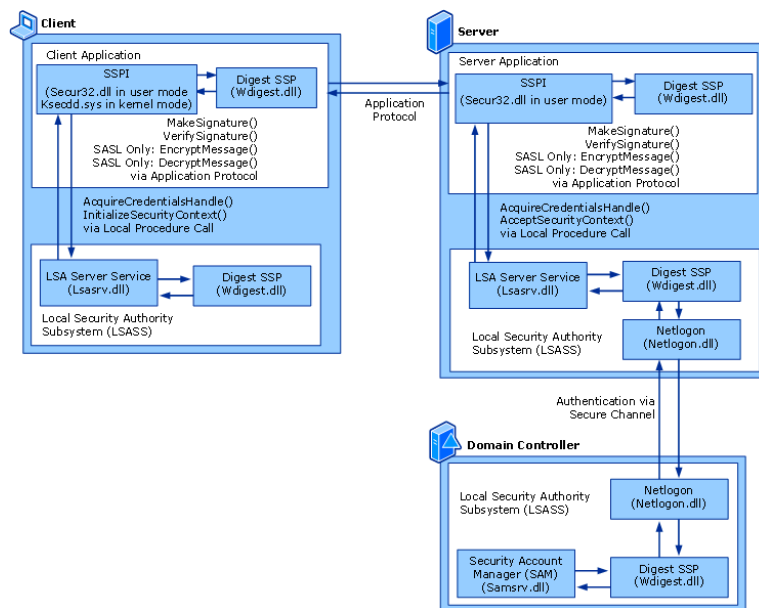[371] https://learn.microsoft.com/en-us/windows/security/identity-protection/access-control/local-accounts
[372] https://medium.com/@boutnaru/the-windows-process-journey-net-exe-net-command-91e4964f20b8

WDigest (Windows Digest)

WDigest is an authentication protocol which was introduced as part of Windows XP. It is designed for HTTP (Hypertext Transfer Protocol) authentication or SASL (Simple Authentication Security Layer) based on RFC 2617 and 2831[373]. WDigest is enabled by default from Windows XP to Windows 8 in case of clients and from Windows 2003 to Windows 2012 in case of servers[374].

Moreover, WDigest is implemented as part of "%windir%\System32\wdigest.dll" (which is digitally signed by Microsoft). By the way, in the case of a 64-bit version of Windows there is also a 32-bit implementation located at "%windir%\SysWOW64\wdigest.dll" - the digest authentication architecture is shown below[375]. In order to enable WDigest we need to set the "UseLogonCredential" value located at the registry location "HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest" to "1"[376].

Lastly, using WDigest causes "lsass.exe"[377] to store the passwords of those which logged on in cleartext (which can be extracted when reading/dumping memory like with Mimikatz). Thus, using the credentials stored in cleartext is used by the OS to authenticate without requesting users to enter their passwords again[378].

---

[373] https://www.elastic.co/guide/en/security/current/modification-of-wdigest-security-provider.html
[374] https://book.hacktricks.xyz/windows-hardening/stealing-credentials/credentials-protections
[375] https://learn.microsoft.com/pt-pt/previous-versions/windows/server/cc778868(v=ws.10)
[376] https://learn.microsoft.com/en-us/answers/questions/463368/disabling-credentials-caching-in-wdigest
[377] https://medium.com/@boutnaru/the-windows-process-journey-lsass-exe-local-security-authority-process-24166cb0358f
[378] https://www.triskelelabs.com/blog/wdigest-extracting-passwords-in-cleartext

Protected Processes

"Protected Processes" is a security feature introduced as part of Windows Vista in order to enhance support for DRM (Digital Right Management) functionality on Windows. Those processes are running in parallel to normal processes[379].

Overall, the difference between a normal process and a protected process is the level of access that other processes in the system can obtain to protected processes. Thus, example of limitations of operations performed by a normal process on protected process are: inability to debug an active protected process, can't duplicate an handle from a protected process, accessing the virtual memory of a protected process and injecting a thread into a protected process[380].

Thus, in case of protected processes we can't debug them in user mode even if we are running as an administrator, "Local System"[381] or "TrustedInstaller"[382] - as shown in the screenshot below[383]. However, we can use a kernel debugger/driver in order to disable the protection on a specific process or alter the protection level of our user-mode debugger[384].

Lastly, from Vista only a bit field (ProtectedProcess) as part "struct _EPROCESS"[385] was used to mark a process as protected Since Windows 8.1, due to signer types and PPL (Protected Processes Light) we have a "Protection" field from type "struct _PS_PROTECTION"[386].

---
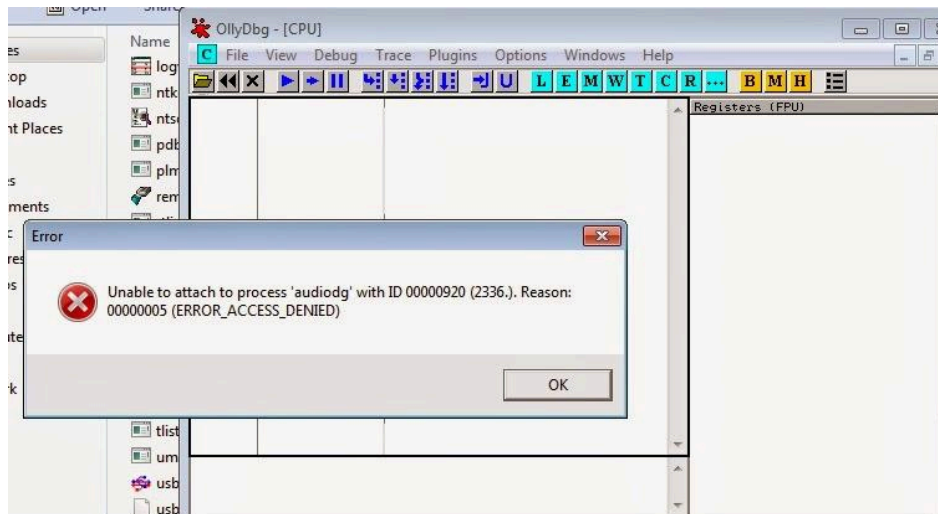
[379] https://web.archive.org/web/20070515091802/http://www.microsoft.com/whdc/system/vista/process_Vista.mspx
[380] https://web.archive.org/web/20090124094639/http://download.microsoft.com/download/a/f/7/af7777e5-7dcd-4800-8a0a-b18336565f5b/process_Vista.doc
[381] https://medium.com/@boutnaru/the-windows-security-journey-local-system-nt-authority-system-f087dc530588
[382] https://medium.com/@boutnaru/the-windows-security-journey-trusted-installer-dc44c73ef04c
[383] http://rce4fun.blogspot.com/2014/01/a-quick-dive-into-kernel-debugging.html
[384] https://itm4n.github.io/debugging-protected-processes/
[385] https://medium.com/@boutnaru/the-windows-kernel-data-structures-journey-struct-eprocess-executive-process-d43e93c51996
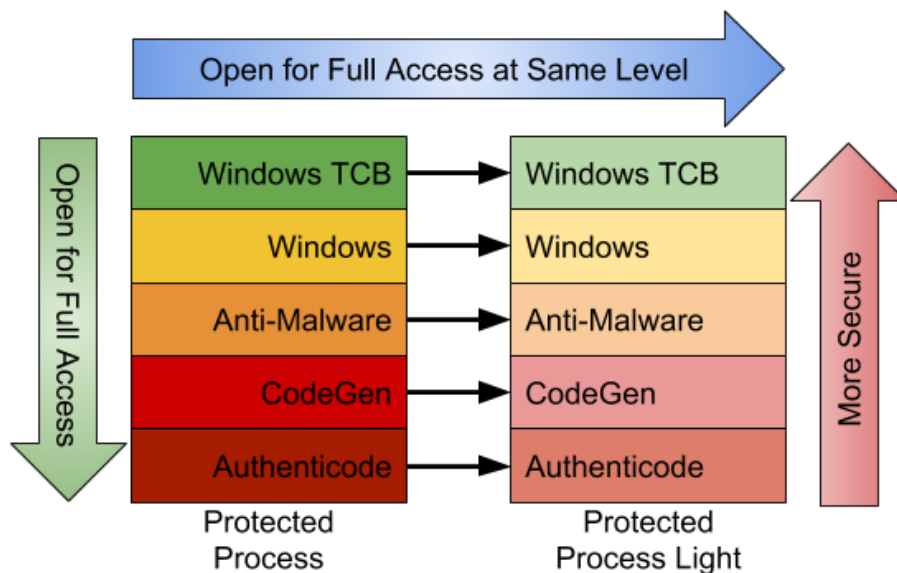[386] https://www.crowdstrike.com/blog/evolution-protected-processes-part-1-pass-hash-mitigations-windows-81/

PPL (Protected Processes Light)

PPL (Protected Processes Light) was introduced as a security feature in Windows 8.1. We can think about them as an extension to the concept of protected processes[387]. PPL allows binaries signed by specific keys to execute in such a way that they are immune from tampering\termination even from users with administrative permissions. Also, PPL supports levels of a hierarchy which means higher-privilege applications can't be tampered by lower-privilege applications, but not vice-versa[388].

Overall, the extension of protected processes to PPL was in order to protect system processes which are not related to DRM (Digital Rights Management). This is done by leveraging digital signatures and code signing, which resembles the entitlement model of iOS[389].

Lastly, the signing levels were also added to protected processes. Thus, a protected process from one level can open all protected processes and PPLs at the same signing level or below. However, a PPL at any signing level can't open a protected process with full access[390] - as shown in the diagram below.

[387] https://medium.com/@boutnaru/the-windows-security-journey-protected-processes-6451a5fff277
[388] https://www.elastic.co/blog/protecting-windows-protected-processes
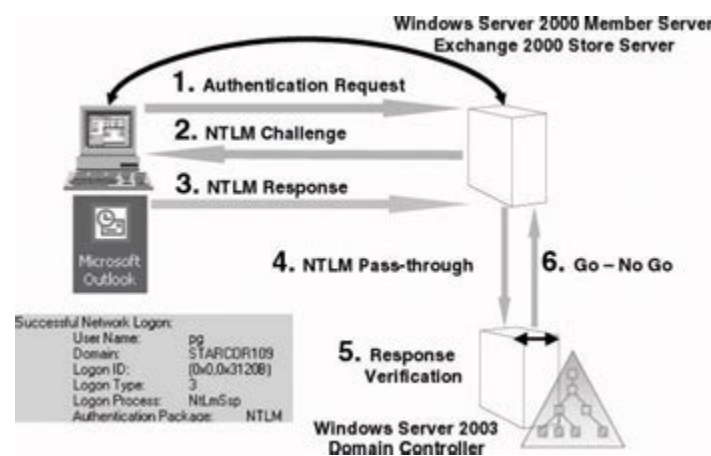[389] https://nosuchcon.org/talks/2014/D3_05_Alex_ionescu_Breaking_protected_processes.pdf
[390] https://googleprojectzero.blogspot.com/2018/10/injecting-code-into-windows-protected.html

Windows Authentication

The goal of an authentication flow is to validate the login information of a specific user. For example comparing a username and password to an authorized list. Microsoft provides different authentication technologies which can be used by developers for authenticating clients/users[391].

Overall, among the Windows authentication technologies we can find the following: credentials management (obtaining and managing credential data), LSA authentication (handles the logon process to a system), network provider API (interacting with other types of networks), smart card authentication, SSPI, Winlogon and credentials providers[392].

Lastly, Windows supports different authentication protocols such as: LM, NTLM and Kerberos[393] - more on them in future writeups. Example is shown in the diagram below[394] .Those authentication methods can be used with other protocols like: LDAP, SMB and HTTP[395].

---

[391] https://learn.microsoft.com/en-us/windows/win32/secauthn/authentication-portal
[392] https://learn.microsoft.com/en-us/windows/win32/secauthn/about-authentication
[393] https://blog.certcube.com/windows-authentication-protocols/
[394] https://flylib.com/books/en/2.244.1.26/1/
[395] https://www.blumira.com/blog/authentication-protocols-101

LSA (Local Security Authority)

LSA (Local Security Authority) is the security subsystem part of the Windows operating system[396]. LSA is used in order to authenticate and log users to the local system. LSA also maintains information regarding all aspects of the local security of the operating system[397].

Overall, it is implemented as part of "%windir%\System32\lsasrv.dll" (LSA Server DLL). Also, it is hosted by the "lsass.exe" (Local Security Authority SubSystem) process[398]. It is one of the basic security components of the Windows operating system together with SAM (Security Account Manager)[399] and SRM (Security Reference Monitor)[400].

Lastly, LSA is used in different use cases such as: the logon process, user authentication, auditing, security policy enforcement and more - as demonstrated in the flows diagram below[401]. We can check the reference implementation of "lsasrv.dll" as part of ReactOS[402].
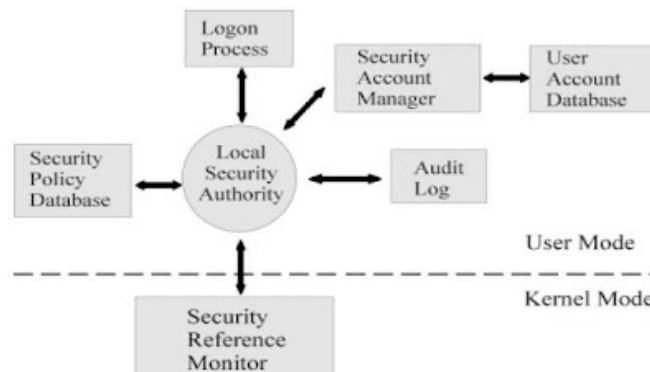
[396] https://learn.microsoft.com/en-us/openspecs/windows_protocols/ms-azod/d28d536d-3973-4c8d-b2c9-989e3a8ba3c5
[397] https://learn.microsoft.com/en-us/windows/win32/secgloss/l-gly
[398] https://medium.com/@boutnaru/the-windows-process-journey-lsass-exe-local-security-authority-process-24166cb0358f
[399] https://medium.com/@boutnaru/windows-security-sam-security-account-manager-c93ddadf388a
[400] https://medium.com/@boutnaru/windows-security-srm-security-reference-monitor-d715f96d9fd6
[401] https://youthtalentauzzar.blogspot.com/2012/01/its-windows-security-hackers.html
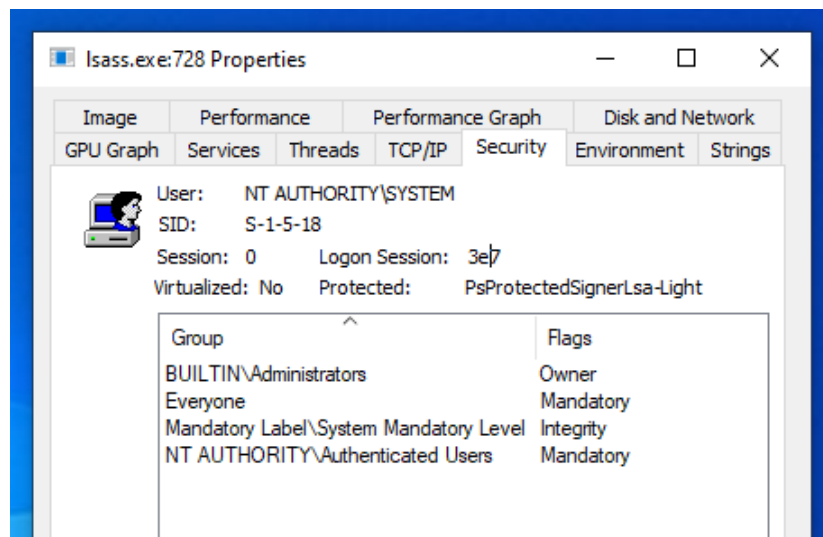[402] https://github.com/reactos/reactos/tree/master/dll/win32/lsasrv

LSA Protection (Local Security Authority Protection)

"LSA Protection" (Local Security Authority Protection) is a security feature of the Windows operating system which is used to disallow memory reads/code injection targeting the "lsass.exe"[403] system process. This feature is supported since Windows 8.1 however just since Windows 11 (22H2) it is enabled by default. The last is in case of a newly installed device or if the device is enterprise-joined to a AD and/or Azure domain[404].

Overall, this feature is needed due the fact the memory address spaces of "lsass.exe" contains very sensitive data such as: NT hashes (which can be used for pass the hash attacks), kerberos tickets (which can be used for pass the ticket attacks), credentials in clear text[405] and more. This security feature can be bypassed in case of kernel level access (arbitrary write/drive). This is also supported by "Mimikatz" which is based on the "mimidrv.sys" driver[406].

Lastly, in order to enable LSA protection we need to set "1"/"2" to the value "RunAsPPL" in the following registry location "HKLM\SYSTEM\CurrentControlSet\Control\Lsa"[407]. In this case the LSA protection is based on the PPL protected processes (PsProtectedSignerLsa-Light) mechanism[408] - as shown in the screenshot below[409]. In case of Windows 11 "lsass.exe" could be protected by "PsProtectedSignerNone" which is based on the protected processes mechanism[410].

---

[403] https://medium.com/@boutnaru/the-windows-process-journey-lsass-exe-local-security-authority-process-24166cb0358f
[404] https://www.mdmandgpanswers.com/blogs/view-blog/3-ways-to-enabledisable-lsa-on-windows-10-and-11
[405] https://medium.com/@boutnaru/the-windows-security-journey-wdigest-windows-digest-3be028607f62
[406] https://book.hacktricks.xyz/windows-hardening/stealing-credentials/credentials-protections
[407] https://learn.microsoft.com/en-us/windows-server/security/credentials-protection-and-management/configuring-additional-lsa-protection#BKMK_HowToConfigure
[408] https://medium.com/@boutnaru/the-windows-security-journey-ppl-protected-processes-light-831d5f371004
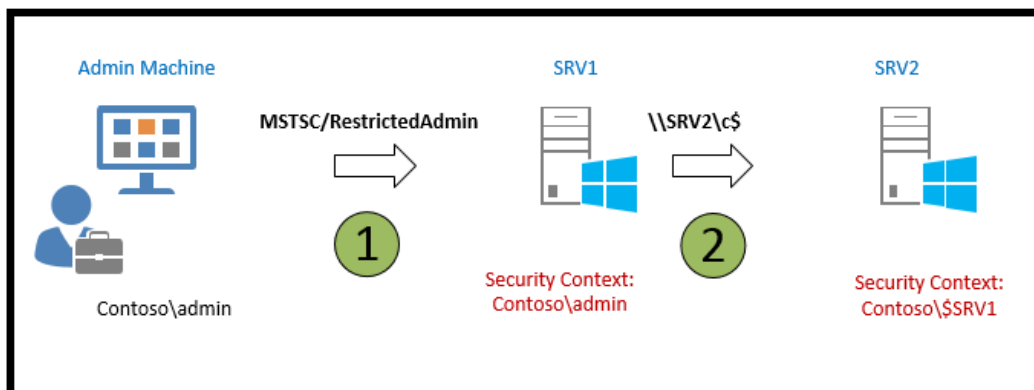[409] https://www.bordergate.co.uk/bypassing-lsa-protections/
[410] https://medium.com/@boutnaru/the-windows-security-journey-protected-processes-6451a5fff277

RestrictedAdminMode for RDP (Remote Desktop Protocol Restricted Admin Mode)

RestrictedAdminMode for RDP (Remote Desktop Protocol Restricted Mode) is a security feature that was introduced as part of Windows 8.1/Windows Server 2012 R2 (and later ported to Windows 7). The goal of the feature is to prevent credentials harvesting by avoiding the transmission of reusable credentials for the connecting admin user[411]. By default, when connecting to a remote device using RDP our credentials are also stored there. Due to that, it limits the access of the user to other servers on the network[412].

Thus, when passing the "/RestrictedAdmin" command line argument to "mstsc.exe"[413] we are authenticating to a but our credentials are not stored on the remote device. In that case any access over the network would be done using the computer account of the RDP server - as shown in the diagram below[414].

Lastly, we can enable the support of RestrictedAdminMode by setting the value of "DisableRestrictedAdmin" to "0" in the following registry location: "HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Lsa"[415]. We can also enable that using local policy/GPO by setting the " Restrict delegation of credentials to remote servers" which can be found in "Computer Configuration > Policies > Administrative Templates > System > Delegation of Credentials"[416].

---

[411] https://learn.microsoft.com/en-us/archive/technet-wiki/32905.remote-desktop-services-enable-restricted-admin-mode
[412] https://blog.ahasayen.com/restricted-admin-mode-for-rdp/
[413] https://medium.com/@boutnaru/the-windows-process-journey-mstsc-exe-remote-desktop-connection-981bae774bac
[414] https://www.anyviewer.com/how-to/restricted-admin-mode-for-remote-desktop-connection-2578.html
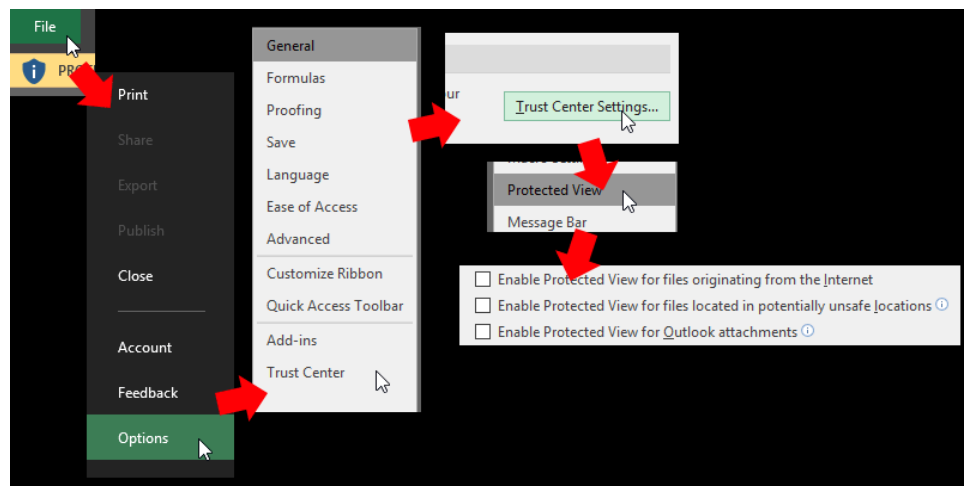[415] https://github.com/GhostPack/RestrictedAdmin
[416] https://4sysops.com/archives/secure-rdp-connections-using-remote-credential-guard/

Protected View

Due to the fact that opening files downloaded from the Internet (browser/outlook attachment/etc) are unsafe (can contain malware). Because of that files from these potentially unsafe locations are opened as read only or in "Protected View". Using "Protected View" we can read a file and see its contents and enable editing while reducing the risks. This security feature is relevant for docs opened by: "Excel for Microsoft 365", "Word for Microsoft 365", "PowerPoint for Microsoft 365", "Excel 2024", "Word 2024", "PowerPoint 2024", "Excel 2021", "Word 2021", "PowerPoint 2021", "Excel 2019", "Word 2019", "PowerPoint 2019", "Excel 2016", "Word 2016" and "PowerPoint 2016"[417].

Moreover, there was a time that if our system supported "Application Guard for Microsoft 365"[418] files which previously opened using "Protected View" were opened in "Application Guard for Microsoft 365" However, "Microsoft Defender Application Guard for Office" is being deprecated and is no longer being updated[419] - more information about that in future writeups.

Lastly, "Protected View" acts like a sandbox while disabling macros[420]. Alos, while opening a document in "Protected View" a yellow dialog box pops up with an "Enable Editing" option. We can also modify the "Protected View" configuration as part of the "Trust Center Settings"- as shown below[421].

---

[417] https://support.microsoft.com/en-us/office/what-is-protected-view-d6f09ac7-e6b9-4495-8e43-2bbcdbcb6653
[418] https://support.microsoft.com/en-us/office/application-guard-for-office-9e0fb9c2-ffad-43bf-8ba3-78f785fdba46
[419] https://support.microsoft.com/en-us/office/application-guard-for-office-9e0fb9c2-ffad-43bf-8ba3-78f785fdba46
[420] https://superuser.com/questions/1771627/why-does-microsoft-words-protected-view-prevent-editing
[421] https://www.digitalwelt.org/en/tips/software/disable-protected-view

Microsoft LAPS (Microsoft Local Administrator Password Solution)

"Microsoft LAPS" (Microsoft Local Administrator Password Solution) is a Windows feature that is deprecated as of Windows 11 23 H2 (and later). Thus, the installation of the MSI package (LAPS.x64.msi\LAPS.ARM64.msi\LAPS.x86.msi) is blocked by a new version of the operating system. LAPS provides management of local account passwords for domain joined devices. Passwords are stored in AD (Active Directory) and protected by ACL, so only eligible users can read it or request its reset. It is replaced by "Windows LAPS"[422]..

Overall, by using LAPS, passwords for the local administrator are unique for each managed device, the passwords are randomly generated and securely stored as part of the AD infrastructure. The feature is based on GPO CSE (Client Side Extension). The CSE checkers if the password expired or not, generates a new password if needed, changes the password of the administrator account and notifies AD regarding the password and expiration date[423].

Lastly, LAPS helps in providing the ability to avoid reusable passwords of the local administrator account (which also helps in coping with PtH attacks). We can configure LAPS using different methods like: PowerShell and GPO. Among the settings are the name of the administrator account to manage (in case it was renamed), password settings and more - as shown in the screenshot below[424].

---
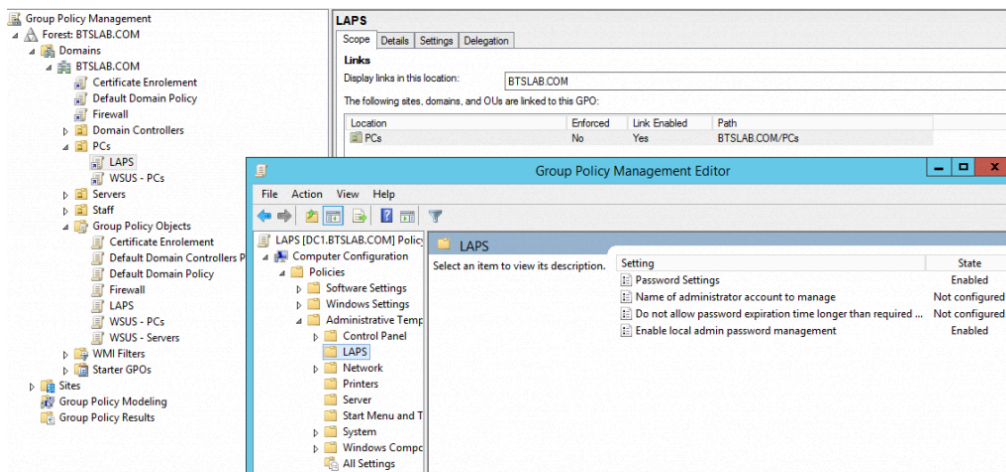
[422] https://www.microsoft.com/en-us/download/details.aspx?id=46899
[423] https://download.microsoft.com/download/C/7/A/C7AAD914-A8A6-4904-88A1-29E657445D03/LAPS_Datasheet.docx
[424] https://www.adamcouch.co.uk/microsoft-laps/
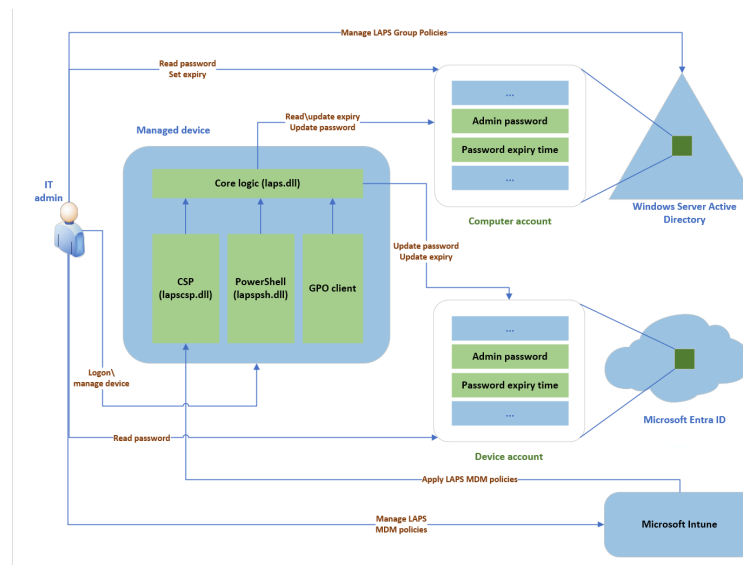
Windows LAPS (Windows Local Administrator Password Solution)

LAPS (Local Administrator Password Solution) is a Windows feature which is used to manage/backup passwords of local user accounts for "Active Directory"[425] or "Microsoft Entra ID"[426] joined devices. It is relevant both for server and workstation editions of the Windows operating system.

Overall, if a device is joined to an active directory domain we can backup the password of the local administrator account only to a Windows server AD. Also, if the device is only joined to Microsoft Entra ID we can backup the passwords only there[427]. The core logic of LAPS is implemented as part of "%windir%\System32\laps.dll" (LAPS Server DLL) while "%windir%\System32\lapscsp.dll" contains the configuration service provider (CSP) logic[428] - as shown in the diagram below.

Lastly, due to the fact LAPS is used for rotating the password of a local Windows account. We can configure the complexity of the password (such as if we want to use large letters+small letters+numbers and special characters) and setting the level for generating random passphrases[429]. By the way, "Windows LAPS" is the replacement of "Microsoft LAPS"[430].

---

[425] https://medium.com/@boutnaru/the-windows-concept-journey-ad-active-directory-af9e795f86b4
[426] https://medium.com/@boutnaru/the-windows-security-journey-microsoft-entra-id-617489ca3d27
[427] https://learn.microsoft.com/en-us/windows-server/identity/laps/laps-overview
[428] https://learn.microsoft.com/en-us/windows-server/identity/laps/laps-concepts-overview
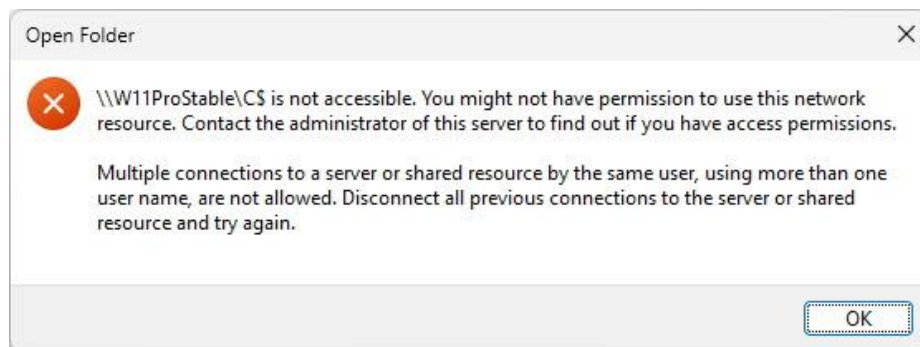[429] https://learn.microsoft.com/en-us/windows-server/identity/laps/laps-concepts-passwords-passphrases
[430]https://medium.com/@boutnaru/the-windows-security-journey-microsoft-laps-microsoft-local-administrator-password-solution-7e08c9d19fda

LocalAccountTokenFilterPolicy (User Account Control Remote Restrictions)

UAC (USer Account Control) is a security feature of the Windows operating system (since Windows Vista) that, if enabled, gives a filtered token to the user (in case the user belongs to the local administrators group). If the user wants the privileged token he needs to consent to in order to get the unfiltered token with the administrator rights[431]. To better protect administrator users there are UAC restrictions for network connections. The goal of those is to prevent "loopback attacks" and malicious software for executing code remotely using the administrative rights

Overall, the "LocalAccountTokenFilterPolicy" is a registry[432] value (DWORD) stored as part of "HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System". In case it has a "0" value (which is the default one) the administrator rights are removed for network connections. In case of "1" the eveneted token is used and not the filtered one[433]. For example, let's say we try to access a remote machine ("\\[RemoteDevice]\[ShareName]") using "net use" with a local account (SAM defined user account) which has administrator rights we will get an "Access is Denied" error - as shown in the screenshot below[434].

Lastly, it is important to know that in case of a domain user the restriction is not relevant. In case a domain user (which is a member of the local administrators group) connects remotely it will get the full administrator access token of the remote device without any UAC restrictions. If we want to administer a remote system with a local user we need to logon interactively like by using "Remote Desktop" (RDP) and/or "Remote Assistant"[435].

---

[431] https://medium.com/@boutnaru/the-windows-security-journey-uac-user-account-control-ce395df5c784
[432] https://medium.com/@boutnaru/the-windows-concept-journey-registry-0767e79387a9
[433] https://kb.cybertecsecurity.com/knowledge/localaccounttokenfilterpolicy
[434] https://www.itechtics.com/localaccounttokenfilterpolicy
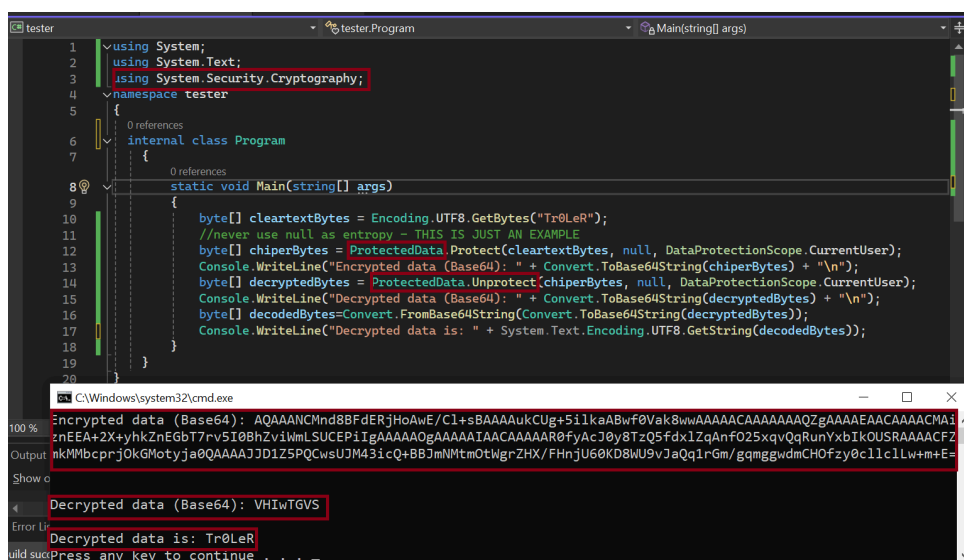[435] https://learn.microsoft.com/en-us/troubleshoot/windows-server/windows-security/user-account-control-and-remote-restriction

DPAPI (Data Protection API)

The goal of DPAI (Data Protection API) is to provide the ability for encrypting data using information related to the current user/computer account. By using DPAPI we avoid the need for explicitly generating and storing cryptographic keys[436]. We can use the "ProtectedData" in .NET for encrypting/decrypting data using DAPI[437] - as shown in the screenshot below (taken from Visual Studio).

Overall, it is used for symmetric encryption of asymmetric private keys. This is done by leveraging user/system secrets as a source of entropy. The keys are derived from the user's logon secrets or the system's domain logon secrets[438]. We select each of those based on whom we want to be able to decrypt the data (every user on the computer or only a specific user).

Lastly, among the data which is protected by DPAPI we can find the following (examples only): "Google Chrome"/"Internet Explorer" password and auto-compilation, passwords of remote desktop connections, email/FTP accounts used by Outlook, passwords of shared folders, wireless networks, .NET passport and personal data in applications such as Skype\MSN messenger[439].

---

[436] https://learn.microsoft.com/en-us/dotnet/standard/security/how-to-use-data-protection
[437] https://learn.microsoft.com/en-us/dotnet/api/system.security.cryptography.protecteddata
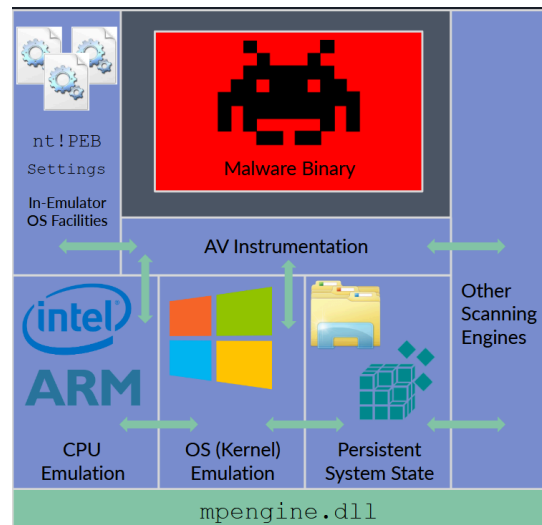[438] https://www.thehacker.recipes/ad/movement/credentials/dumping/dpapi-protected-secrets
[439] https://book.hacktricks.xyz/windows-hardening/windows-local-privilege-escalation/dpapi-extracting-passwords

Windows Defender Antivirus Emulator

Due to the fact AV signatures are easily bypassed (like by using packers) antimalware solutions have added emulation capabilities. The goal of an emulation engine is to execute an unknown binary/script in an emulated environment and check at runtime for malicious behavior/scan for signatures in memory/files dropped to disk/etc[440].

Overall, an analysis conducted on "Windows Defender Antivirus Emulator" showed that some emulated stubs had hardcoded returns like: "GetComputerName" returning "HAL9TH" and "GetUserName" returning "JohnDoe". Also, the emulator has specific files like "C:\INTERNAL__empty". Moreover, there is also a concept of VDLLs which are modified versions of Windows' DLLs which export extra functions. All of those artifacts (and more) can be used by binaries/scripts to identify they are being emulated[441].

Lastly, because we can execute the binary/script forever the emulation can stop due one of the following reasons: termination of the process, limitation of time, max number of CPU instructions, number of API calls, amount of memory used and more. We can summarize that the emulation environment should provide user-mode and kernel-mode emulation, CPU emulation, file-systems and registry emulation and more - as shown in the diagram below[442]

---

[440] https://www.kaspersky.com/enterprise-security/wiki-section/products/emulator
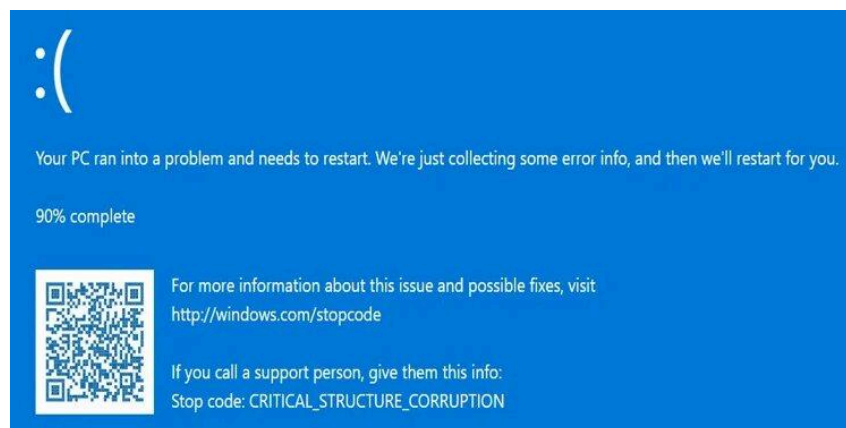[441] https://harfanglab.io/insidethelab/raspberry-robin-and-its-new-anti-emulation-trick/
[442]https://i.blackhat.com/us-18/Thu-August-9/us-18-Bulazel-Windows-Offender-Reverse-Engineering-Windows-Defenders-Antivirus-Emulator.pdf

PatchGuard (Kernel Patch Protection)

"PatchGuard" (aka "Kernel Patch Protection" or KPP) is a security feature as part of 64-bit versions of the Windows operating system. The goal of "PatchGuard" is preserving the integrity of the operating system (mostly hooking, patching and direct object kernel manipulation attacks). Kernel code and data structures are verified to block inconsistent modifications or code injections into the kernel by developers and/or attackers[443].

Overall, "PatchGuard" periodically checks different kernel code areas/data structures. Among those data structures are: SSDT, IDT/GDT, HAL, NDIS, debug routines, loaded module list and more. "PatchGuard" can compute a checksum of sensitive memory areas regularly during the execution time of the system and compare it with the one obtained at boot time (before any user driverload). In case of a mismatch a BSOD (Blue Screen Of Death) is triggered with a BugCheck code 0x109 which is also known as "CRITICAL_STRUCTURE_CORRUPTION"[444] - as shown in the screenshot below[445].

Lastly, "PatchGuard"\"KPP" was introduced in 2005 with the launch of 64-bit versions of "Windows XP" and "Windows 20023 Server". Also, there is no public publication by Microsoft about the specific internal in which "PatchGuard" performs it checks but it is believed to be randomized and probably varies between seconds to minutes[446]. For more information about the causes of " BugCheck code 0x109" I suggest going over the Microsoft documentation[447]. It is important to know that along the years different techniques for bypassing "PatchGuard" were introduced like: "GhostHook", "InfinityHook" and "ByePg"[448].

[443] https://web.archive.org/web/20230427020624/https://zerocondition.com/posts/demystifying-patchguard/
[444] https://blog.tetrane.com/downloads/Tetrane_PatchGuard_Analysis_RS4_v1.01.pdf
[445] https://tech4gamers.com/critical-structure-corruption-windows-11/
[446] https://blog.nviso.eu/2021/10/21/kernel-karnage-part-1/
[447] https://learn.microsoft.com/en-us/windows-hardware/drivers/debugger/bug-check-0x109---critical-structure-corruption
[448] https://therecord.media/poc-published-for-new-microsoft-patchguard-kpp-bypass