Survey on the possibility of Windows 10 live login bypass

Inquérito sobre a possibilidade de desvio de login ao vivo no Windows 10

DOI:10.34117/bjdv8n3-159

Recebimento dos originais: 14/02/2022
Aceitação para publicação: 02/03/2022

João Vitor Assis Ribeiro
Bachelor in Computer Science
Instituto de Criminalística, Polícia Civil do Distrito Federal. Brasília/DF Brasil
Endereço: SPO Conjunto A, Lote 23, Complexo da PCDF Parque da Cidade Sudoeste Brasília DF
E-mail: joao.ribeiro@pcdf.df.gov.br

Daniel Mendes Caldas
M.Sc. Electrical Engineering, Computer Forensics and IT Security
Instituto de Criminalística, Polícia Civil do Distrito Federal. Brasília/DF Brasil
Endereço: SPO Conjunto A, Lote 23, Complexo da PCDF Parque da Cidade Sudoeste Brasília DF
E-mail: daniel.caldas@pcdf.df.gov.br

ABSTRACT
There are many scenarios where, during a law-enforcement or incident response situation, it is of interest to obtain live session data stored in volatile memory (RAM) in Windows 10 machines, which may be locked by a login screen prompt. This work attempts to survey the bibliography for methods and tools that could, in theory or in practice, bypass said security mechanism and possibly aid digital forensic investigators and law-enforcement get the full picture of a case.

Keywords: computer forensics, windows forensics, live analysis.

RESUMO
Há vários cenários em que, durante uma situação de aplicação da lei ou de resposta a incidente, é de interesse obter dados de sessão em tempo real armazenados em memória volátil (RAM) em computadores com Windows 10, que podem se encontrar bloqueados por tela de autenticação. Este trabalho tenta levantar a bibliografia em busca de métodos e ferramentas que poderiam, na teoria ou na prática, burlar tal mecanismo de segurança e possivelmente auxiliar investigadores de forense digital e agentes responsáveis pela aplicação da lei a obter um panora completo de um caso.

Palavras-chave: forense computacional, forense de windows, análise forense ao vivo

1INTRODUCTION

The conduction of forensics analysis of a "Live System" is an important step in the digital forensics examination process [1]. The analysis of a system that is up and

running can give light to valuable evidence, that otherwise would be lost if the system was turned off. Example of important information that could easily be acquired during a live system analysis are access to RAM, running processes, system settings, encrypted data, and access to cloud services. But many devices implements locks, such as passwords, to protect its users data.

An up-to-date, locked Windows 10 machine provides few options to law-enforcement and incident response teams in terms of ephemeral, volatile memory data acquisition and memory analysis. Specially if the target machine is not part of a domain or managed network or otherwise configured in very specific, non-default settings.

Even though some data is saved and committed to disk after a hard reboot or shutdown - such as internet browser session data, cookies, authentication tokens, and so on - it is vital for getting the full picture of a case that live forensics data acquisition is performed.

As the locked machine can not be put in a state that enables or facilitates volatile memory image acquisition (e.g hibernation), the investigator is usually advised - if lacking credentials - to perform a shutdown and, if the situation demands it, attempt a cold boot attack, by either cooling and removing the modules and transferring them to another machine, or by booting into a minimal OS and then performing volatile memory image acquisition.

The effectiveness of these types of attacks have been successfully demonstrated in the past for DDR1 and DDR2 ram modules in lab setups [17], but starting with DDR3 modules they have been increasingly difficult due to, to cite a few factors, a shorter period of RAM retention [18] and the introduction of memory scramblers [20], despite recent findings indicating their vulnerability to unscrambling and deobfuscation in some of its implementations [19].

Not only that, there is the increasing impracticability of performing these type of attacks in real-world scenarios, the challenges in forensic soundness and the time and effort needed before the actual processing of memory images obtained through cold-boot attacks.

That said, the possibility of a live login bypass has its appeal in a sense that not only it enables the investigator to potentially get the full picture of the machine's live user data, its context and avoid potentially undesirable legal implications, while also providing a more favorable scenario for volatile memory image acquisition in terms of forensic soundness.

With that motivation, a survey was performed to see if there was anything new in the wild that could help law enforcement, DFIR personnel, pentesters or interested IT professionals to perform a live login bypass on locked Windows 10 machines without a reboot or shutdown, as well as to provide a discussion of the challenges that this endeavor faces. For the scope of this work, a live login bypass on Windows 10 machines is (not exhaustively) defined with the following restrictions and constraints exposed as follows:

2 ATTACK SURFACE CONSTRAINTS AND RESTRICTIONS

In the scope of this work, the security premises are the following:

1. The attacker has physical access to a login-screen locked target machine running Windows 10 OS;

2. The attacker may not shutdown the machine, reboot it, interrupt or alter its power supply in any way that would cause its volatile memory (RAM) data to be lost in any significant investigative or forensic manner.

3. The attacker may not remove any attached storage or peripheral device that would cause the machine to halt, render it inoperable (crashed) or make its volatile (RAM) memory contents irretrievable;

4. The attacker is free to insert and remove any USB storage devices and HID devices, as well as to manipulate network devices and interfaces, inasmuch as it does not violates 2 and 3;

5. The attacker has no previous knowledge of any of the target machine's users and their passwords (either local or in a remote domain);

6. The target machine is assumed to **not** have volume or drive encryption technology (e.g. BitLocker). The attack may bypass volume or drive encryption as collateral, but for the scope of this work, it is not a requisite and its implications will not be covered.

7. The target machine operating system and its files are assumed to **not** have been previously (i.e. before the attack) modified or tempered in a way that would alter the normal, intended execution flow of Windows 10 authentication mechanism.

8. Knowledge, by the attacker, of the target machine's services and applications - obtained either through monitoring, intelligence or other means - is assumed to be not relevant to achieve login bypass either directly or indirectly.

The first condition aims to focus the survey to local, physical attacks, although it does not exclude remote attack scenarios; The second and third ones mean to prevent the potential loss of session and user data stored in volatile memory (RAM), thus excluding attacks that involve rebooting the machine in order to change or remove (blank) credential information stored in registry hives and then log in as a target user. As these type of techniques (boot or re-boot attacks) do not fit the established criteria, they will only be briefly touched upon.

The fourth one, although the risk of modification to the contents of permanent as well as volatile memory introduced by the manipulation of plugable USB devices and network interfaces (e.g. Ethernet cables) is not negligible, this kind of compromise is believed to be acceptable in most scenarios, within few exceptions. In any case, in real-word scenarios the reasonableness of this (and any) compromise is left for the attacker discretion.

The fifth one is the premise of live login bypass; the sixth one is, in a first moment, for limiting the scope of the research; the seventh is for excluding oddities that involve hypothesis not feasible to be assumed in real world as expected. Finally, the eight one is intended to limit the scope of the research to core Windows 10 authentication ecosystem, which is, leaving out user-specific and case-specific assumptions that would make the survey too broad.

3 WINDOWS INTERACTIVE LOGON - A BRIEF OVERVIEW

In most Windows 10 machines configured with the default login-at-boot settings, the authentication and access management is handled by a chain of processes, the first layer being the credential and login layer [3], which is handled by the process **winlogon.exe**, which in turn launches **LogonUI.exe**, a process, as the name suggests, associated with providing the typical user interface with user and password fields.

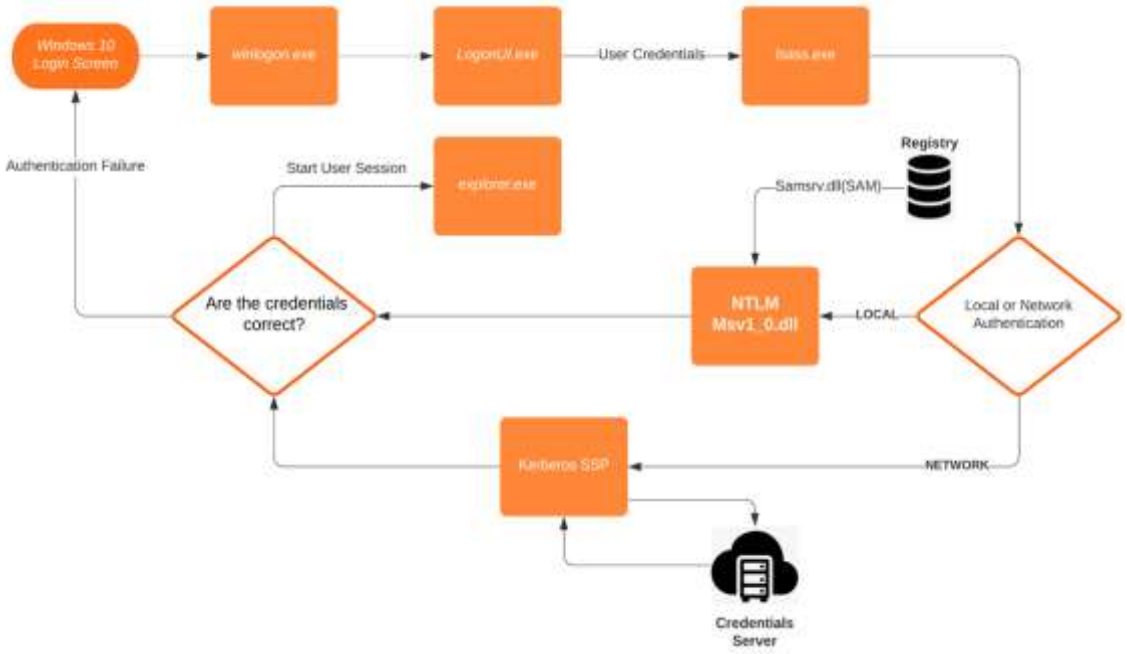
The provided credentials then are passed into the Local Security Authority - LSA, more specifically to the Local Security Authority Subsystem Server - LSASS, represented by the process **lsass.exe**, which then selects the appropriate Security Service Provider for the handling of the authentication procedure. For local accounts, this would be MSV1_0 Authentication Package. For Domain accounts, this could also be MSV1_0 (with pass-through authentication) or Kerberos SSP/AP.

MSV1_0 Authentication Package and NLTM authentication: This protocol is used for the default authentication of local accounts [4]. In short, it confronts the provided credentials passed by the LSA against the SAM - Security Accounts Manager and pass the appropriate response back.

Kerberos SSP / Authentication Package: Kerberos is the preferred Security Service Provider of authenticating a machine in a network domain [5]. The protocol is, in summary, based on the premise of ticket authentication in which the protocol performs an exchange of tickets (Ticket Granting Tickets) between a Ticket Granting Server - in the form of a KDC - Key Distribution Center - and a client. The client then finally receives a Client-to-Server ticket and provides it to the desired server.

After **lsass.exe** validates the provided input with the chosen SSP/AP and returns the response to **winlogon.exe**, the latter generates an appropriate access token for the authenticated user, containing access privileges and constraints, loads the users information into the registry hive and launches an **explorer.exe** session.

Fig. 1: Simplified Flowchart of the Windows 10 logon procedure



4 POSSIBLE ATTACKS

For the scope of this work, the attacks will be divided in two main types. Software-driven and hardware-driven. This classification is intended merely as a guideline for research and not as a strict model of vulnerability and risk assessment.

1. Software-based: Exploitation of vulnerabilities in the instantiation, calling and management of Windows processes, in the processes themselves (e.g. use-after-free, buffer overflow), or in Inter-Process Communication (IPC) of binaries, libraries (e.g. Dynamic-link Libraries) and files involved in the login process and/or in the authentication protocol (MSV1_0, Kerberos);
2. Hardware-based: Circumvent restrictions and constraints that govern the access (read/write) of data structures holding credential information kept in non-persistent memory, mainly by accessing volatile memory directly, but also including changing the expected flow of data (that is, by the employment of devices that intercept normal data flow).

This classification is intended merely as a way to distinguish and highlight the role (and requirement) of hardware in successfully achieving the attack, as any type of attack would require some form of software interfacing or manipulation.

5 SOFTWARE-BASED

Vulnerability survey: A survey of vulnerabilities with generic keywords such as 'Windows 10 login' or 'Windows 10 login bypass' yielded no results related or relevant to Windows 10 conforming to the constraints presented in the introduction. Most of the documented vulnerabilities are related to Kernel Information Disclosure [21] [22] and a successful exploitation requires an already authenticated attacker.

Other vulnerabilities were found related to binaries such as **lsass.exe** [23], but they were dated and not applicable to Windows 10 and modern Windows-variants OSes. Most of the vulnerabilities found that were related to the login and authentication processes either required an already authenticated user (and allowed privilege escalation) or were related to other services (and enabled denial-of-service or remote code execution).

Although it is acknowledged that an attacker may leverage remote code execution to gain access to a target Windows 10 machine - the most prominent attack in the past being known as CVE-2017-0144 [24] (EternalBlue) - these type of attacks do not conform to the **#8** premise, as it involves the exploitation of a service outside the scope of core,

essential Windows 10 authentication (as in described in section 2.3), and may require knowledge and/or intelligence that may not be available to the attacker in reasonable time.

Login brute force: An naive brute-force approach to bypass Windows 10 machines (that is, an attacker manually typing the password on the login screen) is typically not feasible, as the most recent versions of Windows 10 employ a timer between failed attempts, with the default value generally set to one minute between each attempt after five failed login attempts [2].

The probability of success of a dictionary attack is higher if the attacker has the knowledge of previously used passwords (by OSINT research, by collecting on-site information or with the cooperation of the machine's owner or suspect of investigation), but, in the vast majority of cases, negligible.

Boot attacks: These attacks involve rebooting or shutting down the target machine, altering system files and/or tampering the initialization process, and for such, don't meet criteria **#2**. Amongst the commercial tools is Konboot, a tool sold as a product to bypass the login screen of Windows and Mac OS/OSX operating systems [16], claiming to not change or remove (blank) a user's passwords and even working with UEFI systems and providing automated execution of powershell scripts, but, as mentioned, requiring the machine to be rebooted.

Another one is mimikatz [25], which is a freely distributed framework for manipulating and obtaining credential information from Windows-related artifacts (i.e. process and memory dumps), perform pass-the-hash and pass-the-ticker attacks, among other functionalities. Although the relevance of the tool to post-exploitation and pentesting scenarios is undeniable, an application of mimikatz or other similar tools to achieve live, local Windows 10 login bypass on a locked machine (and not previously accessed) was not found.

6 HARDWARE-BASED

DMA - Direct Memory Access Attacks: The premise of DMA attacks is, as the name implies, directly accessing regions of memory, its data structures and/or runtime code that would otherwise be protected by standard runtime and memory-model restrictions. A few attacks have been demonstrated in the past in previous Windows installments [14][15].

These attacks worked by leveraging DMA (either by using physical addresses or guessing the physical-virtual translation) to inject code and/or alter the execution flow of

binaries and functions used involved in authentication and then changing the workflow of functions used in password validation (MSV1_0 protocol).

For mitigating this, some chip manufactures have simply stopped shipping hardware with DMA access capabilities or disabling it by default while also adhering to the UEFI model which generally comes with a more strict policy regarding hardware and bus changes.

On the OS side, Microsoft has introduced Kernel DMA Protection, which comes enabled by default in the most recent Windows installments, blocking not only external, pluggable PCI devices but also internal PCI-Express ports [6]. While the possibility of it being disabled is still there, the probability of encountering it in real-world scenario is narrow as a policy change is needed for DMA to be enabled at login screen [13].

Other protections involves the use of Virtualization-based Security (VBS), which encapsulates part of the process-memory model, preventing the load and execution of unsigned binaries [7], and further difficulting runtime binary injection.

Rogue devices: Rogue devices are reprogrammable USB devices that act as plug--and-play peripheral devices (HID - Human Interface Devices), that can perform automated tasks, data exfiltration, man-in-the-middle attacks and intelligence gathering. They work by exploiting the always-trust policy of most modern machines and IoT devices, that, for convenience and logistic reasons, cannot afford to validate the firmware of each and every USB device.

For devices that emulate HIDs (e.g. keyboard, mouse, headset), they are usually automatically trusted by Windows machines, even if the screen is locked (for the legitimate, local user obviously need a way to input their credentials).

As for rogue network USB devices, their use is predicated upon the metric used by most Windows machines to choose the network device that provide the fastest network connection, prioritizing it over other network connections [11], which is achieved by emulation, effectively "tricking" the OS to "think" the USB device is not only legitimate but the preferred gateway to a LAN or to the internet.

LAN Turtle: One of the most used and known network-based rogue device is LAN Turtle [9]. Sold as a pentesting tool, it is an USB Ethernet adapter, which can, with proper configuration, provide an attacker with a remote, interactive graphical shell. The device functionality has been allegedly demonstrated [12] [10] and it makes use of the newly connected device's network capabilities to trigger and then intercept sensible data about the target machine's users and passwords.

Concerning the proposed scenario described in Attack surface restriction and constraints, a possible attack using LAN Turtle is described as follows:

1. The attacker previously sets up and configures the LAN Turtle device to enable the necessary modules needed to deploy a persistent remote shell (e.g. by using a VPS);
2. The attacker removes the network connection cable of the target machine (Ethernet), plugs the LAN Turtle and insert the cable on the device's network port
3. The LAN Turtle acts as a DHCP server and spoof netBIOs and SMB traffic, tricking the target machine to send its user's account and hash information (SMBv2 NTLM hashes);
4. The attacker logs into the persistent shell and has access to credential information hashes of the logged in users;
5. The attacker uses a password cracking rig to crack the obtained hashes and log in the target machine.

The success of the attack depends upon the target machine having a few prerequisites. First, it needs to be connected to a network via an Ethernet cable. Secondly, it needs to be part of a domain that is regulated by netBIOS or other protocol that broadcasts authentication info on the network.

There is, of course, the drawback of the interruption of the target machine's incoming and outgoing network traffic, as well as the noise introduced by the insertion (and later removal) of the rogue device. As it was not possible to get hands on the device for testing purposes, the possible implications of the deployment of the rogue device concerning forensic soundness and integrity could not be evaluated.

IO tampering: Another idea, in theory, would be to use rogue devices to deploy attacks that could leverage the unrestricted speed and parallelism of a programmable HID to detect and/or exploit vulnerabilities or flaws in input processing (e.g. by feeding a locked machine millions of keystrokes per second with little to no delay apart from the physical medium), possibly triggering wild OS exceptions and/or interrupt states (due to the slower nature of interrupt code handling) that would possibly lead to arbitrary code execution. While reviewing the bibliography, no such works could be found.

7 CONCLUSION

The research by law enforcement and digital forensics investigators of live, local login bypass of Windows machines is driven by the need to get the full picture of a case

by logging in the investigated locked target machine and amongst other things, performing volatile memory acquisition in a way that maximizes forensic soundness and case integrity.

As the result of the survey, no generic, software-based solutions for achieving live, local login bypass of Windows 10 machines with the established conditions and restrictions were found. As for hardware-based solutions, rogue devices, especially LAN Turtle, were found to be conformant, but real-word scenarios might prove to be more challenging.

As the industry is geared towards securing more and more the hardware, which could be observed by the implementation and implantation of technologies such as UEFI, the use of TPM (Trusted Platform Module) modules, and the overall advances in trusted computing, the scenario is expected to change in the future, and for achieving security the industry might need to shift the usability-convenience versus security axis towards a compromise in the former. But for that to happen, also a shift in the way pluggable devices are thought might need to occur, one with considerable logistic challenges.

More so, it is believed that a generic approach to a live login bypass in up-to-date versions of Windows 10 is not feasible, and an attacker might need to balance pros and cons of the techniques and tools available in the context of law enforcement and digital forensics investigations, resorting to a more thorough intelligence work and relying on the specificities of the case to achieve the desired results, applying the necessarily legal means to do so.

REFERENCES

[1] Interpol (2019) Global Guidelines for Digital Forensics Laboratories https://www.interpol.int/content/download/13501/file/INTERPOL_DFL_ GlobalGuidelinesDigitalForensicsLaboratory.pdf
[2] Microsoft Corporation (2021) Account lockout threshold https://docs.microsoft.com/en-us/windows/security/threat-protection/security-policy-settings/
account-lockout-threshold
[3] Microsoft Corporation (2021) Windows Logon Scenarios https://docs.microsoft.com/en-us/windows-server/security/windows-authentication/ windows-logon-scenarios
[4] Microsoft Corporation (2021) MSV1_0 Authentication Package https://docs.microsoft.com/en-us/windows/win32/secauthn/msv1-0-authentication-package
[5] Microsoft Corporation (2021) Kerberos SSP/AP https://docs.microsoft.com/en-us/windows/win32/secauthn/kerberos-ssp-ap
[6] Microsoft Corporation (2021) Kernel DMA Protection https://docs.microsoft.com/en-us/windows/security/information-protection/kernel-dma-protection-for-thunderbolt
[7] Microsoft Corporation (2021) Virtualization-based Security (VBS)https://docs.microsoft.
com/en-us/windows-hardware/design/device-experiences/oem-vbs
[8] Microsoft Corporation (2021) Active Directory Accounts https://docs.microsoft.
com/en-us/windows/security/identity-protection/access-control/
active-directory-accounts#sec-krbtgt
[9] Hak5 (2021) Lan Turtle https://hak5.org/products/lan-turtle
[10] DemmSec (YouTube) Hack a locked computer using the LAN Turtle https://www.youtube.
com/watch?v=M0qZXLPOlvM
[11] IJntema, Tim and Walet, Christian and Smit, Koen and van der Velde, Dave. (2018) Analysing
the Rogue Device Phenomena: An Explorative View on Threats and Potential Solutions. Euro-
pean Conference on Management, Leadership & Governance pgs 98-104. Academic Conferences
International Limited
[12] Hack5 (2021) Stealing Credentials from a locked PC with
the LAN Turtle https://docs.hak5.org/hc/en-us/articles/
360010471814-Stealing-Credentials-from-a-locked-PC-with-the-LAN-Turtle
[13] Thunderbolt Docking Stations Not Initializing at the Login Screen When on a Domain https://www.dell.com/support/kbdoc/en-us/000134656/
thunderbolt-docking-stations-not-initializing-at-the-login-screen-when-on-a-domain?lwp=rt
[14] Aumaitre, Damien, and Christophe Devine. ”Subverting windows 7 x64 kernel with dma attacks.” HITBSecConf Amsterdam (2010). https://conference.hitb.
org/hitbsecconf2010ams/materials/D2T2%20-%20Devine%20&%20Aumaitre%20-%20Subverting%20Windows%207%20x64%20Kernel%20with%20DMA%20Attacks.pdf
[15] Boileau, Adam. ”Hit by a Bus: Physical Access Attacks with Firewire” https://papers.put.

as/papers/macosx/2006/ab_firewire_rux2k6-final.pdf
[16] PiotrBani.Kon-BootforWindowsOfficialGUIDEhttps://kon-boot.com/docs/windows_
guide/
[17] Gruhn, M., & Müller, T. (2013, September). On the practicability of cold boot attacks. In 2013
International Conference on Availability, Reliability and Security (pp. 390-397). IEEE
[18] Bauer, Johannes, Michael Gruhn, and Felix C. Freiling. "Lest we forget: Cold-boot attacks on
scrambled DDR3 memory." Digital Investigation 16 (2016): S65-S74.
[19] Yitbarek, Salessawi Ferede, et al. "Cold boot attacks are still hot: Security analysis of memory
scramblers in modern processors." 2017 IEEE International Symposium on High Performance
Computer Architecture (HPCA). IEEE, 2017.
[20] Yitbarek, S. F. (2018). Hardware Mechanisms for Efficient Memory System Security (Doctoral
dissertation).
[21] The MITRE Corporation. CVE-2017-11842 (2017) https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-11842
[22] The MITRE Corporation. CVE-2017-11850 (2017) https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-11850
[23] CVE Details. CVE-2003-0533 https://www.cvedetails.com/cve/CVE-2003-0533/
[24] National Vulnerability Database. NVD CVE-2017-0144 https://nvd.nist.gov/vuln/detail/CVE-2017-0144