What is LSASS?

LSASS stands for Local Security Authority Subsystem Service. Think of it as a security guard for your computer. It makes sure that only the right people can access the computer and its files.

How LSASS Works

1. **Starting Up**:
    - When you turn on your computer, LSASS starts running automatically. It's like the security guard waking up and getting ready for duty.
2. **User Login**:
    - When you enter your username and password to log in to your computer, LSASS checks if the information is correct. It's like the guard checking if you have the right key to enter the building.
    - If the username and password are correct, LSASS lets you in. If they're wrong, it stops you.
3. **Access Control**:
    - LSASS keeps track of who is allowed to do what on the computer. For example, it knows which files you can open, which programs you can run, and which settings you can change.
    - This is like the guard knowing which rooms you have permission to enter in the building.
4. **Password Management**:
    - LSASS helps manage passwords. If you need to change your password, LSASS makes sure the new password follows the rules (like having enough characters or using special symbols).
5. **Creating Access Tokens**:
    - When you log in, LSASS creates an "access token" for you. This is like a special ID card that you carry while using the computer. It tells the system what you are allowed to do.
    - Every time you try to do something (like open a file or run a program), the computer checks your access token to see if you have permission.

Why LSASS is Important

LSASS is very important for keeping your computer safe. It makes sure that only the right people can use the computer and access sensitive information. Without LSASS, anyone could get into your computer and mess with your files or steal your information.

Simple Example

Imagine your computer is a big library:

- LSASS is the librarian.

- You are a person who wants to enter the library and read some books.
- When you arrive at the library, the librarian (LSASS) checks if you have a library card (your username and password).
- If your card is valid, the librarian lets you in and tells you which sections of the library you can access (your permissions).
- If you need a new library card (changing your password), the librarian makes sure it meets all the rules.

By understanding this, you can see how LSASS helps keep your computer secure and ensures that only authorized users some common attacks that can occur on LSASS.exe:

1. Credential Dumping

Description: Attackers use tools like Mimikatz to extract usernames, passwords, and hashes from LSASS.exe memory.

How it works:

- The attacker gains administrative access to the system.
- They run a tool (e.g., Mimikatz) that reads the memory of LSASS.exe to retrieve credential information.
- The extracted credentials can then be used for lateral movement within the network.

Protection Measures:

- Use Credential Guard to protect LSASS memory.
- Regularly update and patch the system.
- Monitor for suspicious activity related to LSASS.exe.

2. Pass-the-Hash Attack

Description: Attackers use stolen password hashes to authenticate to other systems without knowing the actual password.

How it works:

- The attacker dumps the password hashes from LSASS.exe.
- They use these hashes to authenticate to other systems by passing the hash directly to the authentication process.

Protection Measures:

- Implement strong password policies and regular password changes.
- Use multi-factor authentication (MFA).

- Limit administrative privileges and monitor for unusual login patterns.

3. Process Injection

Description: Attackers inject malicious code into LSASS.exe to manipulate its behavior or escalate privileges.

How it works:

- The attacker gains access to the system and injects code into the LSASS.exe process.
- This code can be used to escalate privileges, bypass security controls, or extract sensitive information.

Protection Measures:

- Use endpoint protection solutions to detect and prevent process injection.
- Regularly update and patch the system to fix vulnerabilities.
- Limit administrative access and use application whitelisting.

4. Pass-the-Ticket Attack

Description: Attackers use Kerberos tickets extracted from LSASS.exe to authenticate as a user without needing their password.

How it works:

- The attacker extracts Kerberos tickets from the LSASS.exe process.
- They use these tickets to authenticate to other systems within the network.

Protection Measures:

- Use Kerberos with short ticket lifetimes and regular ticket renewals.
- Monitor for unusual activity involving Kerberos tickets.
- Implement network segmentation and access controls.

5. Remote Code Execution (RCE)

Description: Attackers exploit vulnerabilities in LSASS.exe to execute malicious code remotely.

How it works:

- The attacker identifies and exploits a vulnerability in LSASS.exe.
- This allows them to run arbitrary code with the same privileges as LSASS.exe, often SYSTEM-level.

Protection Measures:

- Regularly update and patch the system to fix vulnerabilities.
- Use network security measures to prevent unauthorized access.

- Implement intrusion detection and prevention systems.

Summary

Key Attacks on LSASS.exe:

- Credential Dumping (e.g., with Mimikatz)
- Pass-the-Hash Attacks
- Process Injection
- Pass-the-Ticket Attacks
- Remote Code Execution (RCE)

General Protection Measures:

- Apply least privilege principles.
- Regularly update and patch the system.
- Use strong password policies and multi-factor authentication.
- Monitor for suspicious activities.
- Use endpoint protection and network security measures.

Understanding these attacks and implementing the corresponding protection measures can significantly enhance the security of systems against threats targeting LSASS.exe.

Explain Code Injection Techniques:

1. **Process Injection:** This involves injecting code into a running process. The injected code then executes within the memory space of that process, inheriting its privileges. Techniques include:
    - **DLL Injection:** Loading a malicious DLL (Dynamic Link Library) into an existing process.
    - **API Hooking:** Intercepting system API calls and injecting custom code before or after the original call is executed.
2. **Buffer Overflow:** Exploiting vulnerabilities in software that allows writing malicious code beyond the intended buffer size, potentially overwriting program instructions and injecting new code

Explain Pass-the-Ticket Attack

A Pass-the-Ticket (PtT) attack exploits the Kerberos authentication protocol used in Windows environments. Here's a simplified overview:

1. **Attacker Gains Initial Access:** The attacker first needs to gain initial access to a compromised system within the network. This could be achieved through various methods like phishing, malware, or exploiting vulnerabilities.

2. **Extracting the Ticket Granting Ticket (TGT):** Once inside the compromised system, the attacker uses tools (e.g., Mimikatz) to extract the Kerberos ticket-granting ticket (TGT) from memory. The TGT is a crucial credential that allows the system to request other service tickets from the Kerberos Key Distribution Center (KDC).
3. **Passing the TGT:** The attacker copies the stolen TGT (often in the form of a hash) and injects it into another system. This can be done through various methods, depending on the attacker's access and privileges.
4. **Impersonation:** The new system, unaware of the stolen credentials, uses the TGT to request service tickets from the KDC. These service tickets grant the attacker temporary access to resources on the network as if they were the legitimate user associated with the stolen TGT.

Practical Example

Imagine a scenario (in a controlled test environment) where an attacker compromises a user's computer (PC-A) within a network domain.

1. **Gaining Initial Access:** The attacker might have tricked the user on PC-A into clicking a malicious link or opening an infected attachment, compromising the system.
2. **Extracting the TGT:** Using tools like Mimikatz (hypothetically in this scenario), the attacker extracts the user's TGT from PC-A's memory.
3. **Passing the TGT:** The attacker copies the stolen TGT and injects it into another user's computer (PC-B) on the network. This could involve exploiting vulnerabilities or tricking the user on PC-B.
4. **Impersonation:** PC-B, unaware of the stolen credentials, uses the TGT to request service tickets, allowing the attacker to impersonate the legitimate user and potentially access resources they shouldn't have access to (like file servers, applications) on PC-B or other systems within their permission range.

What is NTLM?

NTLM is a suite of security protocols used for authentication in Windows networks. It ensures that users are who they claim to be by verifying their credentials, such as username and password.

How NTLM Works with LSASS

1. **User Login**:
    - When you log in to a Windows computer, you enter your username and password.
    - LSASS, which is running on the computer, takes these credentials and starts the authentication process.
2. **Creating a Hash**:
    - Instead of sending your actual password over the network, LSASS creates a cryptographic hash of your password. A hash is a scrambled version of your password that can't be easily converted back to the original password.

- This hash is created using an algorithm defined by NTLM.

3. **Challenge-Response Mechanism**:
    - The NTLM authentication process uses a challenge-response mechanism to verify the user's identity without sending the password over the network.

Steps of NTLM Authentication

Step 1: Negotiate:

- The client (your computer) sends a message to the server (the computer you're trying to access) to establish communication and indicate that it wants to use NTLM for authentication.

Step 2: Challenge:

- The server responds with a "challenge," which is a random number or nonce.
- This challenge is sent back to the client.

Step 3: Response:

- The client takes the challenge and the hash of your password, and combines them using a specific algorithm to create a response.
- This response is then sent back to the server.

Step 4: Verification:

- The server receives the response and uses the same algorithm (and the stored hash of the user's password) to verify if the response matches what it expects.
- If the response is correct, it means the user is authenticated, and access is granted.
- If the response is incorrect, access is denied.

Interaction Between NTLM and LSASS

- **Hash Storage**:
    - LSASS is responsible for securely storing the password hashes. When a user logs in, LSASS uses these stored hashes to create the response needed for the NTLM challenge-response mechanism.
- **Session Management**:
    - Once authenticated, LSASS manages the user's session, including creating access tokens that define the user's permissions on the system.

Technical Example

1. **Login**:
    - You enter your username "Alice" and password "password123".

2. **Hashing**:
    - LSASS hashes the password "password123" using the NTLM algorithm, resulting in a hash like "ABCD1234".
3. **Challenge-Response**:
    - The server sends a challenge, say "XYZ789".
    - LSASS combines "XYZ789" and "ABCD1234" to create a response, say "LMNOP456".
4. **Verification**:
    - The server uses the stored hash for "Alice" (which is also "ABCD1234") and combines it with "XYZ789" to see if it gets "LMNOP456".
    - If it matches, authentication is successful, and LSASS grants access.

Key Points

- **Security**: The actual password is never sent over the network, reducing the risk of interception.
- **Trust**: The challenge-response mechanism ensures that both the client and the server can trust the authentication process.
- **Efficiency**: LSASS handles all these operations quickly and securely, ensuring seamless login experiences.

Detail Of NTLM

In Windows, NTLM (NT LAN Manager) information isn't directly stored in a single directory. NTLM is an authentication protocol used for logging in to resources on a network. Here's a breakdown of how credentials and NTLM information are handled:

- **Credentials:** Usernames and passwords are not stored in plain text. They are hashed (converted into a one-way mathematical value) using a secure algorithm before being saved. This makes it extremely difficult to recover the original password even if someone gains access to the stored data.
- **NTLM Hashes:** When you log in using NTLM, your password is hashed on your machine and then sent to the server. The server performs its own hashing and compares the two values. If they match, authentication is successful. These NTLM hashes are typically stored in the **Security Account Manager (SAM)** ***(C:\Windows\System32\config\SAM)*** database on a local machine or in the Active Directory database on a domain-joined system.
- **Windows Registry:** Some NTLM-related settings might be stored in the Windows Registry, which is a database that stores configuration information for the operating system. However, this wouldn't include actual user credentials.

Important Security Notes:

- NTLM is considered a less secure authentication protocol compared to newer methods like Kerberos. It's recommended to use more secure options whenever possible.
- If you're concerned about someone accessing your credentials, it's crucial to:
    - Use strong, unique passwords for all your accounts.
    - Enable two-factor authentication (2FA) whenever available.
    - Keep your operating system and software applications updated with the latest security patches.

However, it's important to understand a few key things about the SAM file:

- **Inaccessibility:** The SAM file is a critical system file and cannot be accessed directly through normal means in Windows. It's protected by the operating system and requires administrative privileges even to view its properties.
- **Security:** The SAM file itself doesn't store user passwords in plain text. It stores one-way mathematical hashes of passwords, making it very difficult to crack them.
- **Special Tools:** If you have a legitimate reason to access the SAM file, such as for password recovery purposes, specialized tools and techniques are required. These tools are typically used by system administrators and data recovery professionals.

Here's what you should avoid doing:

- **Trying to access or modify the SAM file yourself:** This can lead to system instability or data corruption if not done correctly.
- **Using untrusted third-party tools:** There might be tools that claim to access the SAM file, but they could be malicious or ineffective.

NTDS → C:\Windows\NTDS\NTDS.DIT

NTDS stands for **New Technology Directory Services**. It's a core component of Microsoft's Active Directory (AD) service, which is used for managing user accounts, groups, computers, and other resources in a Windows domain network.

what NTDS does:

Centralized Directory: AD uses NTDS to maintain a centralized database that stores information about all the objects in a domain, such as:

- User accounts and their passwords (hashed for security)
- Groups of users and their permissions
- Computers and their configurations
- Printers and other network resources

Accessibility and Management: IT administrators can access and manage this information through tools like the Active Directory Users and Computers console. This allows them to:

- Create new user accounts and assign permissions
- Manage groups and group memberships
- Control access to network resources
- Implement security policies

NTDS.DIT File: The core data of NTDS is stored in a file called **NTDS.DIT**. This file resides on each domain controller (a server that runs AD) and is replicated across all domain controllers to ensure consistency.

Importance of NTDS:

- **Centralized Management:** NTDS simplifies the management of user accounts and resources in a domain network, especially for large organizations with many users and devices.
- **Security:** By using a central directory for authentication and authorization, NTDS helps enforce security policies and control access to resources.
- **Scalability:** AD and NTDS can scale to accommodate a large number of users and devices in a domain.

In summary, NTDS is the foundation of Active Directory, providing a centralized and secure way to manage user accounts, groups, and other resources in a Windows domain network.

- **C:\Windows\NTDS\NTDS.DIT**

However, it's important to remember a few things about this path:

- **Domain Controllers Only:** This path applies specifically to domain controllers in a Windows domain network. If your computer is not a domain controller, it won't have an NTDS.DIT file.
- **Modification During Setup:** While the default path is C:\Windows\NTDS, it's possible to modify this location during the initial Active Directory setup process. So, in rare cases, the NTDS.DIT file might be stored in a different location on a domain controller.

Here's how to check the exact location of the NTDS.DIT file on a domain controller:

Using Registry (Command Prompt or PowerShell):

1. Open Command Prompt or PowerShell as administrator.
2. Run the following command:
3. reg query "HKLM\SYSTEM\CurrentControlSet\Services\NTDS\Parameters" /v "DSA Database file"

This command will display the registry value for the "DSA Database file" which points to the location of the NTDS.DIT file.

Additional Notes:

- The NTDS.DIT file is a critical system file and should not be modified directly unless you're a qualified IT professional. Modifying it can lead to data corruption and problems with Active Directory.
- The NTDS.DIT file is protected by the operating system and requires administrative privileges to access.

Feature/Aspect	SAM (Security Account Manager)	NTDS (NT Directory Services)
Purpose	Manages local user accounts and security information	Manages Active Directory (AD) database for domain environments
Location	Located on individual Windows machines (standalone or workgroup)	Located on domain controllers (DCs) in a Windows domain
Database File	SAM file located in C:\Windows\System32\config	NTDS.dit file located in C:\Windows\NTDS on domain controllers
Scope	Local machine	Entire domain
User Management	Handles local user and group accounts	Handles domain user and group accounts
Authentication	Used for local logins	Used for domain logins and network resource access
Replication	No replication (local only)	Replicated across all domain controllers in the domain
Backup and Restore	Managed by local machine backup tools	Managed by domain backup tools (e.g., Active Directory Backup)
Security Policies	Local security policies (Local Security Policy)	Domain security policies (Group Policy Objects)
Typical Use Case	Workgroup environments, standalone servers, personal computers	Enterprise environments with multiple computers and centralized management

Summary

- **SAM**: Primarily for local accounts and security management on individual machines.

- **NTDS**: Primarily for managing user accounts and security information within an Active Directory domain environment, with centralized control and replication across domain controllers.

LM, NTLM, Net-NTLMv2

LM

LM-hashes is the oldest password storage used by Windows, dating back to OS/2 in the 1980's. Due to the limited charset allowed, they are fairly easy to crack. You can obtain them, if still available, from the SAM database on a Windows system, or the NTDS database on the Domain Controller.

LM was turned off by default starting in Windows Vista/Server 2008, but might still linger in a network if there older systems are still used. It is possible to enable it in later versions through a GPO setting (even Windows 2016/10).

When dumping the SAM/NTDS database, they are shown together with the NTHash, before the colon.

Example

299BD128C1101FD6

The algorithm

1. Convert all lower case to upper case
2. Pad password to 14 characters with NULL characters
3. Split the password to two 7 character chunks
4. Create two DES keys from each 7 character chunk
5. DES encrypt the string "KGS!@#$%" with these two chunks
6. Concatenate the two DES encrypted strings. This is the LM hash.

Cracking it

- john --format=lm hash.txt
- hashcat -m 3000 -a 3 hash.txt

NTHash (A.K.A. NTLM)

This is the way passwords are stored on modern Windows systems, and can be obtained by dumping the SAM database, or using Mimikatz. They are also stored on domain controllers in the NTDS file. These are the hashes you can use to pass-the-hash.

Usually people call this the NTLM hash (or just NTLM), which is misleading, as Microsoft refers to this as the NTHash (at least in some places). I personally recommend to call it the NTHash, to try to avoid confusion.

Example

B4B9B02E6F09A9BD760F388B67351E2B

The algorithm

MD4(UTF-16-LE(password))

UTF-16-LE is the little endian UTF-16. Windows used this instead of the standard big endian, because *Microsoft*.

Cracking it

- john --format=nt hash.txt
- hashcat -m 1000 -a 3 hash.txt

NTLMv1 (A.K.A. Net-NTLMv1)

The NTLM protocol uses the NTHash in a challenge/response between a server and a client. The v1 of the protocol uses both the NT and LM hash, depending on configuration and what is available. The Wikipedia page on NT Lan Manager has a good explanation.

A way of obtaining a response to crack from a client, Responder is a great tool. The value to crack would be the K1 | K2 | K3 from the algorithm below. Version 1 is deprecated, but might still be used in some old systems on the network.

Example

u4-
netntlm::kNS:338d08f8e26de93300000000000000000000000000000000:9526fb8c23a90751cdd619b6cea564742e1e4bf33006ba41:cb8086049ec4736c

The algorithm

C = 8-byte server challenge, random
K1 | K2 | K3 = LM/NT-hash | 5-bytes-0
response = DES(K1,C) | DES(K2,C) | DES(K3,C)

Cracking it

- john --format=netntlm hash.txt
- hashcat -m 5500 -a 3 hash.txt

NTLMv2 (A.K.A. Net-NTLMv2)

This is the new and improved version of the NTLM protocol, which makes it a bit harder to crack. The concept is the same as NTLMv1, only different algorithm and responses sent to the server. Also captured through Responder or similar. Default in Windows since Windows 2000.

Example

admin::N46iSNekpT:08ca45b7d7ea58ee:88dcbe4446168966a153a0064958dac6:5c7830315c7830310000000000000b45c67103d07d7b95acd12ffa11230e0000000052920b85f78d013c31cdb3b92f5d765c783030

The algorithm

SC = 8-byte server challenge, random
CC = 8-byte client challenge, random
CC* = (X, time, CC2, domain name)
v2-Hash = HMAC-MD5(NT-Hash, user name, domain name)
LMv2 = HMAC-MD5(v2-Hash, SC, CC)
NTv2 = HMAC-MD5(v2-Hash, SC, CC*)
response = LMv2 | CC | NTv2 | CC*

Cracking it

- john --format=netntlmv2 hash.txt
- hashcat -m 5600 -a 3 hash.txt

IN SUMMARY

LM- and NT-hashes are ways Windows stores passwords. NT is confusingly also known as NTLM. Can be cracked to gain password, or used to pass-the-hash.

NTLMv1/v2 are challenge response protocols used for authentication in Windows environments. These use the NT-hash in the algorithm, which means it can be used to recover the password through Brute Force/Dictionary attacks. They can also be used in a relay attack, see byt3bl33d3r's article [1].

Sources

[1] https://byt3bl33d3r.github.io/practical-guide-to-ntlm-relaying-in-2017-aka-getting-a-foothold-in-under-5-minutes.html

[2] https://technet.microsoft.com/en-us/library/dd277300.aspx#ECAA

[3] https://en.wikipedia.org/wiki/LAN_Manager

[4] https://en.wikipedia.org/wiki/NT_LAN_Manager

[5] https://en.wikipedia.org/wiki/Security_Account_Manager

[6] https://hashcat.net/wiki/doku.php?id=example_hashes

NTLMv1 vs NTLMv2.

NTLM is the original version of the protocol, providing authentication and security features. It is an outdated protocol and has been replaced by NTLMv1 and NTLMv2.

NTLMv1

NTLMv1 was an enhancement over the original NTLM protocol. It introduced improved security features but still has several vulnerabilities that make it less secure compared to NTLMv2. NTLMv1 is supported in the following Windows versions:

- **Windows NT 4.0**
- **Windows 2000**

- **Windows XP**
- **Windows Server 2003**
- **Windows Vista**
- **Windows Server 2008**
- **Windows 7**
- **Windows Server 2008 R2**

NTLMv2

NTLMv2 is a more secure version of the NTLM protocol, providing stronger authentication and security features. It is recommended to use NTLMv2 over NTLMv1 due to its enhanced security measures. NTLMv2 is supported in the following Windows versions:

- **Windows NT 4.0 SP4 and later**
- **Windows 2000**
- **Windows XP**
- **Windows Server 2003**
- **Windows Vista**
- **Windows Server 2008**
- **Windows 7**
- **Windows Server 2008 R2**
- **Windows 8**
- **Windows Server 2012**
- **Windows 8.1**
- **Windows Server 2012 R2**
- **Windows 10**
- **Windows Server 2016**
- **Windows 11**
- **Windows Server 2019**
- **Windows Server 2022**

Net-NTLMv1 vs Net-NTLMv2

Net-NTLM (sometimes referred to as NTLM pass-through) refers to the NTLM challenge-response authentication mechanism used over a network, such as in SMB (Server Message Block) protocol or HTTP.

Net-NTLMv1

Net-NTLMv1 is the network implementation of NTLMv1. It is supported by the same Windows versions that support NTLMv1 but is highly discouraged due to known security weaknesses.

Net-NTLMv2

Net-NTLMv2 is the network implementation of NTLMv2, providing better security for network authentication. It is supported by the same Windows versions that support NTLMv2 and is the recommended protocol for network authentication due to its enhanced security features.

Summary

In summary, NTLMv2 (and Net-NTLMv2) is supported in all modern versions of Windows starting from Windows NT 4.0 SP4 and later. NTLMv1 (and Net-NTLMv1) is supported in older versions but should be avoided due to security vulnerabilities. Modern Windows systems default to NTLMv2 for better security.

NTLMv2

- **Purpose**: Used for local authentication.
- **Usage**: Provides secure authentication for users logging into their local Windows machine or domain.
- **Function**: Enhances security with stronger encryption and better protection against attacks compared to NTLMv1.

Net-NTLMv2

- **Purpose**: Used for network authentication.
- **Usage**: Provides secure authentication for users accessing services over a network, such as file shares (SMB) or web services (HTTP).
- **Function**: Uses the same enhanced security features of NTLMv2 but is specifically designed for network communication.

Key Difference

- **Context**: NTLMv2 is for local authentication, while Net-NTLMv2 is for authenticating over a network.
- **Application**: NTLMv2 is used when logging into a computer; Net-NTLMv2 is used when accessing network resources.

NTLMv2 Attacks

1. **Pass-the-Hash (PtH) Attack**:
    - **Description**: Attackers capture the NTLM hash of a user's password and use it to authenticate as that user without needing the plaintext password.
    - **Impact**: Allows attackers to gain access to systems and resources using stolen hashes.
2. **Relay Attacks**:
    - **Description**: Attackers capture and relay NTLM authentication messages between a client and server, allowing them to authenticate as the user.
    - **Impact**: Can lead to unauthorized access to network services and resources.

Net-NTLMv2 Attacks

1. **Relay Attacks**:
    - **Description**: Similar to NTLMv2, attackers capture Net-NTLMv2 challenge-response messages and relay them to another service to authenticate as the user.
    - **Impact**: Allows attackers to access network services by relaying authentication messages.
2. **Downgrade Attacks**:
    - **Description**: Attackers force the use of weaker protocols (like NTLMv1) by interfering with the negotiation process, making it easier to break the authentication.
    - **Impact**: Weaker security and increased vulnerability to other attacks, such as brute force.
3. **Credential Forwarding**:
    - **Description**: Attackers trick users into authenticating to a malicious server, which then forwards the credentials to a legitimate server.
    - **Impact**: Unauthorized access to network resources and services using forwarded credentials.

Mitigation Strategies

1. **Use Kerberos Instead of NTLM**:
    - Kerberos is a more secure authentication protocol available in modern Windows environments.
2. **Enable SMB Signing**:
    - Helps prevent relay attacks by ensuring the integrity and authenticity of SMB messages.
3. **Implement Multi-Factor Authentication (MFA)**:

- Adds an extra layer of security beyond just the NTLM credentials.

4. **Restrict NTLM Usage**:
    - Limit or disable NTLM authentication in your network where possible.
5. **Regularly Update and Patch Systems**:
    - Ensure all systems are up-to-date with the latest security patches and updates.
6. **Monitor and Audit**:
    - Keep track of authentication attempts and detect unusual or suspicious activity.

These strategies can significantly reduce the risk of attacks on both NTLMv2 and Net-NTLMv2.