**Finding 1: Sensitive Information Stored in Windows Registry**

**Description**:  
During the assessment, sensitive information (the first flag) was discovered within the Windows registry. The search was conducted using the reg query command to locate specific registry entries.

A screenshot of a computer screen

Description automatically generatedCommand used:  
reg query HKLM /f flag /t REG\_SZ /s

This command successfully located a string-based registry entry containing the first flag.

**Remediation Recommendation**:  
To mitigate this, ensure that sensitive information like flags or credentials is not stored in the Windows registry. If storing sensitive data is necessary, use encryption or other mechanisms to secure the data.

**Finding 2: User Account Description Revealing Credentials**

**Description**:  
A credential was found within the description field of a user account. During enumeration of Active Directory users, the following account description was identified:

* **Name**: THM Victim
* **Description**: Change the password: Passw0rd!@#

A screenshot of a computer

Description automatically generatedThe description contained a default password that could potentially be exploited by an attacker to gain unauthorized access.

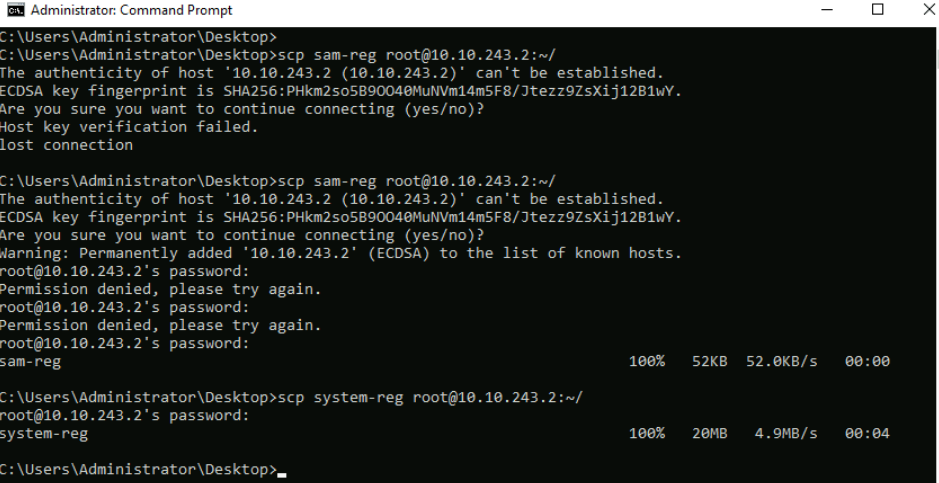
Attackers who gain access to the Active Directory could easily leverage such information to compromise user accounts.

**Remediation Recommendation**:  
Implement stricter policies to ensure that sensitive information such as passwords are not stored in user account descriptions or other metadata. Regular audits should be conducted to verify that such information is not inadvertently exposed.

**Finding 3: Extracting Administrator NTLM Hash via SAM and SYSTEM Registry Files**

**Description**:  
The Administrator's NTLM hash was retrieved by leveraging the SAM and SYSTEM registry hives. These registry hives contain sensitive information about local user accounts, including password hashes. By extracting these files and decrypting them using the secretsdump.py tool from Impacket, the NTLM hash of the Administrator account was obtained.

1. A screenshot of a computer

   Description automatically generated**Saving the SAM and SYSTEM Registry Hives**: The following commands were used to copy the SAM and SYSTEM registry hives to the victim machine’s Desktop:
2. **Transferring the Registry Files to the Attack Box**: The registry files were transferred to the attack box using scp:
3. **Decrypting the SAM File Using secretsdump.py**: The SAM and SYSTEM files were decrypted using the secretsdump.py tool from Impacket to reveal the NTLM hash of the Administrator account:

A computer screen shot of a computer code

Description automatically generated

**The NTLM hash for the Administrator account was successfully extracted, which could be used in further attacks (e.g., pass-the-hash).**

**Remediation Recommendation**:  
To prevent unauthorized access through SAM and SYSTEM registry files, implement the following measures:

* Ensure that only authorized users have administrative privileges on critical systems.
* Regularly audit access to sensitive files and enforce strict access controls.
* Use strong, regularly updated passwords and enforce multi-factor authentication (MFA) wherever possible.
* Consider encrypting sensitive system files and using tools like LAPS (Local Administrator Password Solution) to manage local admin passwords securely.

**Finding 4: Extracting Web Application Credentials via PowerShell Script**

**Description**:  
Using a PowerShell script called Get-WebCredentials.ps1, credentials for the user **THMuser** were extracted from the internal web application at **internal-app.thm.red**. This allowed the identification of sensitive credentials stored on the target system.

1. **Bypass PowerShell Execution Policy**: The following command was used to bypass the PowerShell execution policy: powershell -ex bypass
2. **Importing the PowerShell Module**: The Get-WebCredentials.ps1 script was imported to extract the web credentials: Import-Module C:\Tools\Get-WebCredentials.ps1
3. A screenshot of a computer screen

   Description automatically generated**Extracting Credentials**: The credentials for **THMuser** were extracted using: Get-WebCredentials

This process successfully retrieved the credentials for THMuser.

**Remediation Recommendation:**

Ensure that sensitive credentials are securely stored and managed. Avoid storing plaintext credentials and use secure credential vaults. Regular audits of credential storage should also be conducted.

**Finding 5: Credential Dump via Mimikatz for SMB Share**

**Description:**

Using Mimikatz, credentials for the 10.10.237.226 SMB share, stored in the Windows Credential vault, were retrieved. Mimikatz allows attackers to dump credentials from memory, which could be leveraged for lateral movement and privilege escalation.

1. **Elevating Privileges:** Mimikatz was run with elevated privileges using the following command to enable debugging: privilege::debug
2. **A screenshot of a computer

   Description automatically generatedDumping Credentials:** The credentials were dumped from the Windows Credential vault with: sekurlsa::credman

Remediation Recommendation:

To mitigate this, ensure that privileged accounts are protected with stronger access controls, and sensitive credentials in memory are minimized. Use Credential Guard and other advanced protection mechanisms to secure the Windows Credential vault.

**Finding 6: Gaining Access to THM-Local User Account via runas**

**Description:**

To escalate privileges and access the flag located in C:\Users\thm-local\Saved Games\flag.txt, the runas command was used to run cmd.exe under the context of the thm-local user.

1. **Running cmd.exe as thm-local User:** The following command was used to execute cmd.exe under the thm-local user: runas /savecred /user:THM.red\thm-local cmd.exe
2. **Reading the Flag:** Once the command prompt was elevated to thm-local, the flag was located and read using: type C:\Users\thm-local\Saved Games\flag.txt

A computer screen with white text

Description automatically generated

A screenshot of a computer

Description automatically generated

**Finding 7: Dumping NTDS File and Extracting Hashes via PowerShell**

**Description:**

The NTDS (NT Directory Services) file was dumped using the ntdsutil tool through a one-liner PowerShell command. This dump allows the extraction of user password hashes and sensitive information from the Active Directory database, along with obtaining the system's bootkey.

1. **Dumping the NTDS File**: The following PowerShell one-liner was used to dump the NTDS file using the Ntdsutil tool: powershell "ntdsutil.exe 'ac i ntds' 'ifm' 'create full c:\temp' q q"

A black screen with a black background

Description automatically generatedA screenshot of a computer program

Description automatically generated

1. I transferred the SECURITY, SYSTEM and ntds.dit to my attacking machine

1. **Retrieving the Hashes**: The following command was used to retrieve the hashes from the NTDS dump which extracted the password hashes from the NTDS.dit file, along with other sensitive information from the **SECURITY**, **SYSTEM** and **ntds.dit** files.

A computer screen shot of a computer screen

Description automatically generated

1. A screenshot of a computer

   Description automatically generated**Analyzing the Extracted Hashes**: The extracted hashes can then be stored and analyzed for potential attacks
2. **Crack the hash:** either online or using tools like hashcat and john the ripper

**A screenshot of a computer

Description automatically generated**