nccgroup
freedom from doubt

An NCC Group Publication

Local network compromise despite good patching:

The dangers of NBNS/LLMNR spoofing attacks and how to prevent them

Prepared by:

Jon Macfarlane

Contents

1 Introduction

A common misconception by Windows system administrators is that keeping operating systems fully updated is sufficient to keep them secure. However, even on a network which is fully patched and using the latest Windows operating systems, it is often trivial for an internal attacker to obtain user credentials, and in many cases privileged credentials, that can be leveraged to gain control over the entire domain.

One of the most popular methods used by penetration test teams in an internal engagement, is to listen for Windows broadcast traffic on the local network segment. By intercepting and manipulating name resolution traffic, it is possible to redirect authentication traffic to the attacker's machine in a Man-in-the-Middle (MitM) attack. With the authentication handshake captured by the attacker, it is possible to perform a brute force offline password cracking attack in order to obtain the clear-text credentials. Even if the passwords in use are sufficiently complex to prevent passwords being cracked within a realistic timeframe, the authentication attempt can be relayed to another host in order to grant the attacker access to resources.

These attacks are well known to penetration testers and malicious attackers, having been around for many years, and the necessary tools are freely available. They represent a valuable opportunity for an attacker who is able to gain physical access to the target network, as well as one who has already gained authenticated access to an internal host (for example via a phishing attack) and is seeking to escalate privileges.

However, the measures required to prevent these attacks are less well known. Some do not feature in most server hardening guides, and some are rarely implemented in corporate networks due to a fear of compatibility or performance issues. This paper aims to raise awareness of the dangers of these attacks, and particularly the steps required to prevent them.

2 The attacks

2.1 NBNS and LLMNR spoofing

The attacks in question are known as NBNS (sometimes called NBT-NS) and LLMNR spoofing (or poisoning). Essentially they involve exploiting the broadcast nature of protocols that are by default turned on in all versions of Windows.

This section aims to provide a high-level overview of how the attacks work rather than a guide for how to perform them. The necessary tools are readily available and the steps required to perform them are well documented elsewhere.

In order to understand how these attacks work it is first necessary to look at the protocols involved.

NetBIOS Name Service

The NetBIOS Name Service (NBNS, or sometimes NBT-NS) is part of the NetBIOS over TCP protocol suite. NetBIOS is a relic from the past; originally developed in 1983 to allow computers to communicate within a local area network (LAN), it was adopted by Microsoft in the late 1980s for the LAN Manager operating system, found its way into early versions of Windows and has been present ever since. NBNS is the name resolution component of the suite, and essentially works by sending broadcast messages on the local network segment in order to resolve the names of other hosts. Sometimes used in conjunction with a WINS server, it was the primary method of name resolution in Windows until the advent of Windows 2000 when dynamic DNS was introduced.

Link Local Multicast Name Resolution (LLMNR)

LLMNR is the successor protocol to NBNS which was introduced with Windows Vista and supports IPv6 which its predecessor does not. It uses a multicast address which shouldn't be routed beyond the local subnet, so in effect it is a local subnet broadcast like NBNS.

NetBIOS over TCP/IP is enabled by default on all network interfaces on Windows machines (or if DHCP is used, set by the DHCP server for which the default setting is enabled). LLMNR is enabled by default on all network interfaces on machines running Windows Vista and later (7, Server 2008, 8, Server 2012, 10).

With DNS being the primary means of name resolution for Windows machines, these protocols are used as a fallback when DNS fails. When a Windows machine from Vista onwards tries to resolve a hostname to an IP address, the machine attempts to resolve the name in the following order:

- Domain Name System (DNS)
  - DNS resolver cache (including hosts file contents)
  - DNS servers
- LLMNR
  - LLMNR cache
  - Multicast (IPv6 FF02::1:3) (IPv4 224.0.0.252)
- NetBIOS
  - WINS servers (if configured)
  - LMHosts file
  - NBNS Broadcast

You might expect that on a well-configured network where every host has an entry in DNS, it would be rare for the secondary methods to be required. However, grabbing a packet capture on the typical Windows network usually reveals that they are awash with both LLMNR and NBNS broadcasts; usually failing. Some of the reasons for the DNS requests that precede these broadcasts failing are:

- Mistyped addresses
- Software configured to contact hosts that no longer exist or do not reside on the current network
- Queries for 'wpad' - The Web Proxy Auto-Discovery Protocol (WPAD) is enabled in by default on Windows machines and will attempt to resolve this name
- Random looking names caused by Google Chrome detecting DNS redirection

An attacker on the local subnet can listen for these broadcasts and respond to them, claiming that the requested hostname resolves to their own IP address. This results in the requesting client machine connecting to the attacker's machine, and, depending on the protocol, may attempt to authenticate. In the case of SMB or HTTP traffic, the attacker can negotiate NTLM authentication with the client. The primary goal of the attacker in this scenario is to obtain the encrypted NTLM challenge response, which can then be brute forced offline with password cracking tools to retrieve the clear text password of the currently logged-in user.

The diagram below illustrates the steps taking place in a NBNS or LLMNR spoofing attack:

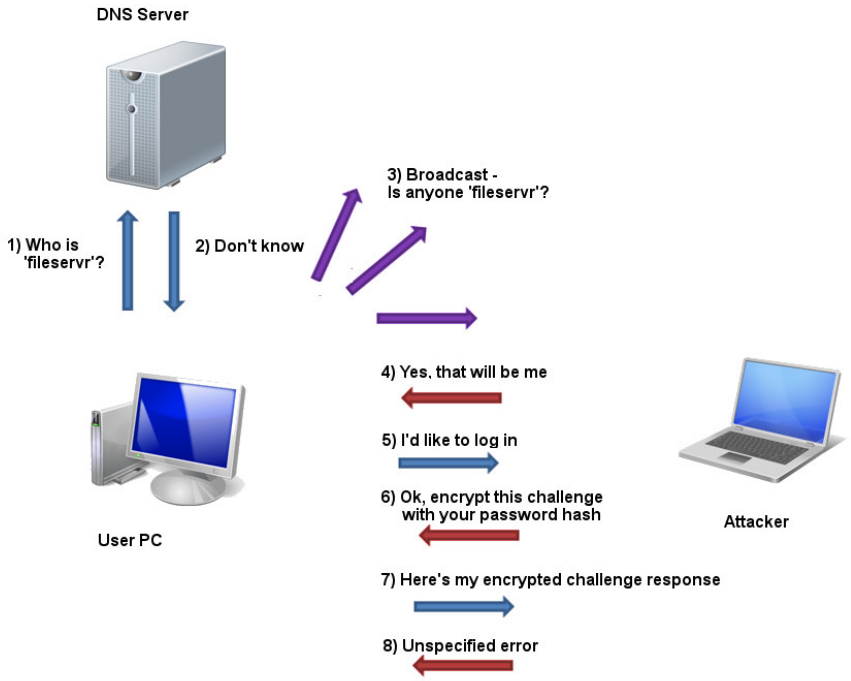

Figure 1 - NBNS/LLMNR spoofing and capture of NTLM challenge response

The diagram illustrates a failed DNS lookup for the mistyped 'fileservr', and the subsequent broadcast (either NBNS or LLMNR) which the attacker is able to respond to and accept a login attempt, in order to capture the NTLM challenge response, before returning an unspecified error to the client.

These attacks are intrinsically stealthy and difficult to detect, especially because in most cases the original name looked up was either incorrect or automatically requested in the background.

This issue is exacerbated by the fact that Windows machines by default use the Web Proxy Auto-Discovery Protocol (WPAD) to attempt to automatically find a web proxy on the local network. In doing so they attempt to resolve the name 'wpad', and if this name is not in DNS, an attacker may spoof this name and set up a rogue proxy server, directing all the victim's web traffic through their machine.

Publicly available tools to perform these attacks include Responder [1], Inveigh [2] and Metasploit [3], amongst others.

2.2 SMB Relay

While NTLM hashes captured with the technique described above can often be cracked, this is usually only possible when the password in question is weak or predictable in some way. From Windows Vista onwards the default for NTLM authentication is NTLMv2, a considerable improvement from v1, and as such the times for password cracking are greatly increased. When truly complex passwords are in use, it is highly unlikely that an NTLMv2 hash will be cracked in a realistic timescale.

Enter SMB relay – a technique that allows the intercepted authentication attempt to be forwarded to a third host, allowing the execution of code as the authenticating user and avoiding the need to crack the password. SMB relay attacks date back to 2001, at which point is was possible to reflect the authentication attempt back to the authenticating host. This issue was not patched by Microsoft until 2008 [4] and the patch only fixed the vulnerability when the SMB connection is reflected back to the client. If it is forwarded to another host, the vulnerability can be still exploited.

The diagram below illustrates how an SMB relay attack works, in conjunction with NBNS/LLMNR spoofing:

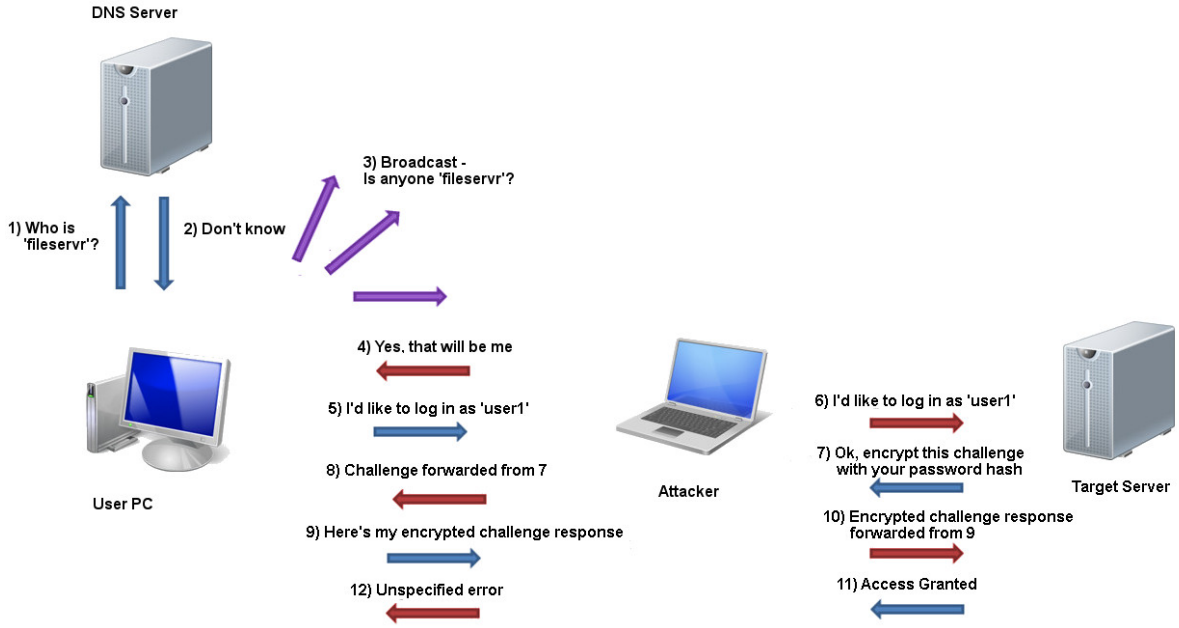

Figure 2 - NBNS/LLMNR spoofing and SMB Relay

The example in the diagram extends the previous example in that the authentication attempt is relayed to a third host in a Man-in-the-Middle attack. The attacker forwards authentication traffic between the client and the server, resulting in the server successfully authenticating the attacker as the original user.

If the authenticating user has local administrator rights, it is possible for the attacker to execute code on the target server in order to take full control of it. Even if the user is not a local administrator, the attacker can still gain access to any resources (e.g. file shares) that the authenticating user has access to.

Because NTLM authentication is used for both the SMB and HTTP protocols, it is possible to perform a cross-protocol attack. For example, an HTTP request for an unknown resource can be responded to by the attacker and the authentication relayed to an SMB server.

Tools available to perform these attacks include smbrelayx.py (from the impacket [5] library), Inveigh [2], Snarf [6] and ZackAttack [7]. More recently, scripts that automate the process have begun to appear, such as Chuckle [8].

Note that to perform an SMB relay attack, the attacker just needs to cause the victim to initiate an HTTP or SMB connection back to the attacker; NBNS/LLMNR spoofing is only one means of accomplishing this. Another is to persuade the victim via a phishing email to visit a website that contains a link to an image on the attacker's machine. Another effective method for an internal attacker is to create a shortcut (.lnk file) with an icon reference pointing to the attacker's machine, and place this on a network share. When a user views the share, an SMB connection to the attacker is automatically made [9].

2.3 Local privilege escalation via NBNS spoofing

A more recent technique uses NBNS spoofing for another end; escalating privileges locally.

Originally disclosed by Google Project Zero [10] and later expanded on by FoxGlove Security in their 'Hot Potato' tool [11], this attack effectively performs an HTTP to SMB relay attack on the local machine, with a low-privileged user relaying requests sent by a high-privileged user on the machine to the local SMB server in order to execute code as the high-privileged user.

This is very similar in concept to the NBNS spoofing/SMB Relay attack just described, with some differences to allow it to work on a single host. The privilege escalation attack works by flooding NBNS responses locally in anticipation of an expected request for 'wpad'. A local proxy is started and when Windows update sends a web request this is directed through the local proxy, where NTLM authentication is requested and relayed to local SMB server in order to execute code as SYSTEM.

The issue affects Windows 7, 8, 10, Server 2008 and Server 2012 and at the time of writing there is no Microsoft patch for the issue. Indeed Microsoft have indicated that they will not patch the issue [12]. Although mitigations are available, they require changing configuration settings from the defaults, so a fully patched Windows machine that has not had additional hardening applied is likely vulnerable to this issue.

Update: This privilege escalation vulnerability was patched by Microsoft in June 2016, with MS16-075 preventing the local HTTP to SMB relay.

3 Prevention

How can these attacks be prevented? Simply installing all security patches is not sufficient; all of the above attacks are possible against all Windows versions (up to and including Windows 10) unless additional hardening measures are applied. These measures are discussed below.

3.1 Disable Broadcast Protocols

The most effective way to stop NBNS and LLMNR spoofing is simply to disable the protocols concerned, eliminating the name resolution broadcasts that the attack tools respond to.

NetBIOS over TCP/IP is enabled by default on all network interfaces on Windows machines, and LLMNR is enabled by default on all network interfaces on machines running Windows Vista and later. However, in a corporate environment they are not normally required because any networked host should have an entry in DNS. If, for whatever reason, DNS entries are not an option, entries in the hosts file can be used.

Although notably absent from popular Windows hardening guides [13], disabling both NBNS and LLMNR provides a significant increase in security. It is important to note that both protocols need to be disabled to prevent spoofing attacks.

Disabling NetBIOS over TCP/IP

Preventing NBNS broadcasts from being sent requires disabling NetBIOS over TCP/IP. A common misconception is that NetBIOS is required in order for Windows file and printer sharing to work, but this has not been the case for many years. Before Windows 2000, the Server Message Block (SMB) protocol which supports file sharing did require NetBIOS (using TCP port 139). From Windows 2000 onwards, SMB is also directly hosted on TCP (using port 445) [14].

In most cases NetBIOS over TCP/IP can be safely disabled. Thorough testing should be performed when disabling NetBIOS in a corporate environment, however. Although they are increasingly rare, NetBIOS-dependent applications still exist.

One potentially visible impact of disabling NetBIOS over TCP/IP is that the 'Computer Browser' service depends on it. This is the service that produces the browse lists seen under the Network icon in Windows Explorer, as well as the 'net view' command. When NetBIOS is enabled in a network, the master browser collects information about all the computers in the network and that information is propagated periodically to all workstations to populate these lists.

Seeing machines disappear from these lists may cause alarm; however, the machines can still be reached by either name (via DNS) or IP address. Network browse lists can also be populated by SSDP, if Network Discovery is turned on.

Unfortunately there is no global setting for disabling NetBIOS over TCP/IP, and changes must be made for *each network interface*. The following methods are available:

Manually:

Control Panel > Network and Internet > {Network adapter name} > Properties > Network Connections > Internet Protocol Version 4 > Properties >Advanced >WINS tab > Disable NetBIOS over TCP/IP

Via the registry (set the following value to 2):

HKLM\SYSTEM\CurrentControlSet\services\NetBT\Parameters\Interfaces\Tcpip_{GUID}\NetbiosOptions

Where *GUID* is the globally unique identifier (GUID) of the network interface adapter.

By command line (the below will configure all interfaces):

```
wmic nicconfig where TcpipNetbiosOptions=0 call SetTcpipNetbios 2
wmic nicconfig where TcpipNetbiosOptions=1 call SetTcpipNetbios 2
```

Note that where DHCP is used, the default setting for NetBIOS over TCP/IP is to accept the setting from the DHCP server. While a quick fix can be to disable NetBIOS over TCP/IP via options on the DHCP server, this is not the best solution, as laptops may be connected to other networks and potentially leak corporate account credentials to attackers on those networks.

Disabling LLMNR

Disabling LLMNR is less problematic, with the sole purpose of the protocol being name resolution. It can be disabled on a system-wide basis via the following Group Policy setting:

Location	
Computer Configuration\Policies\Administrative Templates\Network\DNS Client\	
Policy	**Setting**
Turn off Multicast Name Resolution	Enabled (default is Disabled)

Note that counterintuitively, setting the above policy to 'Enabled' will *disable* LLMNR.

3.2 Require SMB signing

SMB signing is supported by all versions of Windows as far back as Windows 98 SE. It places a digital signature into each server message block in order to prevent man-in-the-middle attacks. However, while it is *enabled* by default for both outgoing and incoming SMB sessions, it is not *required* by default (except for incoming connections to domain controllers). In the SMB relay example given above, the attacker machine claims not to support signing and therefore it is not used. The attack would have failed if the target server required SMB signing.

Despite this, not requiring SMB signing is one of the most commonly detected and reported issues on internal penetration tests. In most environments there is no reason not to require SMB signing on all Windows system. As it is enabled by default in Windows both clients and servers, all SMB traffic between Windows hosts will typically already be signed. Non-Microsoft SMB clients and servers

(including SANS) may need reconfiguring to support signing, however. Again there is normally no good reason not to do this. While a performance impact is often cited as a negative effect, this is probably negligible with modern systems. The performance degradation shows as increased CPU usage on the client and server, but the amount of network traffic does not change.

The following Group Policy settings will require that both incoming and outgoing SMB connections are signed:

Location	
Computer Configuration\Windows Settings\Security Settings\Local Policies\Security Options\	
Policy	**Setting**
Microsoft network client: Digitally sign communications (always)	Enabled
Microsoft network server: Digitally sign communications (always)	Enabled

Note that while requiring SMB signing can prevent the relaying of NTLM authentication to an SMB server, it does nothing to prevent either SMB or HTTP NTLM authentication from being relayed to an HTTP server. SMB signing also does not prevent the capture and offline cracking of the NTLM challenge response.

3.3 Extended Protection for Authentication

A lesser known mitigation for SMB relay attacks is Extended Protection for Authentication (EPA).

EPA allows a client's authentication to be tied to an outer security channel, and checks that the outer channel that arrives at the server-side matches the outer channel opened on the client-side. The server verifies that the outer channel at the client-side matches based on information supplied by the client in an inner channel exchange. If the two channel ends do not match, the server rejects the client's request.

To enable EPA for an SMB client, the following registry entry should be created and set to 0 (default is 1):

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\LSA\SuppressExtendedProtection

To enable EPA for an SMB server, the following group policy setting should be used:

Location	
Computer Configuration\Windows Settings\Local Policies\Security Options	
Policy	**Setting**
Microsoft network server: Server SPN target name validation level	Accept if provided by client **or** Required from client (default is Off)

The setting **Required from client** is the most secure option although it requires any connecting clients to have **SuppressExtendedProtection** set to **0**; clients with the default setting will be rejected. Therefore changes will likely have to be made across the entire environment. EPA is also supported by other Microsoft products including IIS and SQL Server.

The setting **Accept if provided by client** will still allow clients with the default setting to connect, and should be considered if making environment-wide changes is not possible or desirable. This setting

does still prevent SMB relay attacks using popular tools, when **SuppressExtendedProtection** is set to 0 on the client. It will also prevent privilege escalation via Hot Potato.

3.4 Restrict NTLM

Kerberos, the successor protocol to NTLM, solves many of the problems of its predecessor. Domain-joined machines typically will use Kerberos for authentication, however, some situations still require NTLM, for example if the client is authenticating using the server's IP address, or if the server does not belong to a domain. It is difficult and may be impractical to completely eliminate NTLM [15]. However, it is possible to control where NTLM authentication is allowed via Group Policy. This can go a long way to defending against hash capture and relay attacks. This can be achieved with the following group policy settings:

Location	
Computer Configuration\Windows Settings\Local Policies\Security Options	
Policy	**Setting**
Network Security: Restrict NTLM: Outgoing NTLM traffic to remote servers	Deny all outgoing NTLM traffic to remote servers
Restrict NTLM: Add remote server exceptions	{Add a list of servers or ranges for which NTLM traffic is required}

Configuring address ranges for the latter setting which include an organisation's servers but not desktops is a reasonably simple way of preventing one compromised desktop from capturing NTLM hashes from other desktops.

3.5 Ensure only NTLMv2 is used

Ensuring that only NTLMv2 is in use has been recommended in the past as a means of protecting against SMB relay, although this way only really any help because the freely-available tools at the time did not support NTLMv2 [16]. From the point of view of an attacker who has captured a hash however, the password cracking times for NTLMv2 are many times higher than for LM and NTLM.

Since Windows Vista onwards the default settings are to use only NTLMv2, so unless older operating systems are in use, no change is required.

The following Group Policy setting will ensure only NTLMv2 is used:

Location	
Computer Configuration\Windows Settings\Local Policies\Security Options	
Policy	**Setting**
Network security: LAN Manager authentication level	Send NTLMv2 response only\refuse LM & NTLM

3.6 Network segregation

The importance of segregating internal networks cannot be overstated when discussing prevention of internal attacks. A large 'flat' or unfiltered internal network can be a huge gift to an internal attacker as it can allow them to move around laterally unimpeded.

With regards to NBNS/LLMNR spoofing, the number of hosts in each subnet or broadcast domain will determine the quantity of broadcasts that might be visible to any host. An attacker in a large subnet with hundreds of hosts will have many opportunities for poisoning and therefore will intercept a wide range of credentials. By limiting the size of each subnet and the number of hosts within them, administrators can reduce these opportunities. Of particular importance is ensuring that users with access to privileged accounts (e.g. IT staff) use a separate subnet to regular end users, reducing the chance that a compromised end user machine will be able to intercept privileged credentials.

VLANs should also be separated by Access Control Lists (ACLs). Restricting TCP ports 139 and 445 between VLANs and allowing them only where strictly necessary will greatly limit opportunities for an attacker attempting to relay intercepted credentials.

3.7 Apply the principle of least privilege

Care should be taken to avoid privileged account credentials traversing networks wherever possible. End user accounts should not have local admin rights on servers unless they really require them. IT staff should have separate standard and domain admin accounts, and use the latter only for tasks that require them. Jump boxes can be used where necessary so that low-privileged credentials are used across less trusted networks and admin privileges only used between the jump box and target server.

4 Conclusion

The attacks described represent a serious security threat that is presently overlooked in many corporate Windows networks. There is no forthcoming patch to fix the issues and tools to perform these attacks have become increasing available, more powerful and easier to use. In many cases they represent the 'path of least resistance' to an attacker.

Microsoft have backwards compatibility and functionality out-of-the-box to contend with which can impact on full security out-of-the-box. Legacy protocols and features are allowed to persist and be enabled by default for years after they have been identified as security risks. For example, the LAN Manager hash for password storage, another relic last required by Windows 95, was not removed as a default until Windows Vista and Server 2008. NetBIOS has lasted considerably longer. The onus is very much on the system administrator to make changes to default configurations if security is considered desirable.

Serious consideration should be given to disabling NetBIOS over TCP/IP and LLMNR in Windows environments, across the board if possible. This is the single most complete mitigation to the attacks described above. In most environments, they are unlikely to be missed, and if some problems do occur as a result, there are likely to be workarounds that are far preferable to the risks. SMB signing also should be required across the board, NTLM traffic should be restricted and at least a partial implementation of EPA should be considered. Fears of compatibility issues are often overstated; the best way to find out the impact is to make changes to a sample of hosts and any problems are likely to soon be apparent.

A remote attacker could be only one phishing email away from user-level access to your internal network. With NBNS/LLMNR spoofing offering internal attackers opportunities for lateral movement, all data held on the network is at risk and denying them these opportunities is vital.

5 References & further reading

1. Github - SpiderLabs Responder https://github.com/SpiderLabs/Responder
2. Github – Inveigh https://github.com/Kevin-Robertson/Inveigh
3. Metasploit - NetBIOS Name Service Spoofer - https://www.rapid7.com/db/modules/auxiliary/spoof/nbns/nbns_response
4. Microsoft Security Bulletin MS08-068 - https://technet.microsoft.com/en-us/library/security/ms08-068.aspx
5. Github – Impacket https://github.com/CoreSecurity/impacket
6. Github - Snarf https://github.com/purpleteam/snarf
7. Github - ZackAttack https://github.com/urbanesec/ZackAttack
8. Github – Chuckle https://github.com/nccgroup/chuckle
9. MS08_068 + MS10_046 = FUN UNTIL 2018 https://room362.com/post/2012/2012-02-11-ms08_068-ms10_046-fun-until-2018/
10. Windows: Local WebDAV NTLM Reflection Elevation of Privilege https://code.google.com/p/google-security-research/issues/detail?id=222)
11. Hot Potato – Windows Privilege Escalation https://foxglovesecurity.com/2016/01/16/hot-potato/
12. Full Disclosure - Windows Local WebDAV NTLM Reflection Elevation of Privilege http://seclists.org/fulldisclosure/2015/Mar/149
13. Direct hosting of SMB over TCP/IP https://support.microsoft.com/en-us/kb/204279
14. CIS Benchmarks for Windows 10 https://benchmarks.cisecurity.org/tools2/windows/CIS_Microsoft_Windows_10_Enterprise_RTM_Release_1507_Benchmark_v1.0.0.pdf
15. Auditing and restricting NTLM usage guide https://technet.microsoft.com/en-us/library/jj865674(v=ws.10).aspx
16. SMB Relay Demystified and NTLMv2 Pwnage with Python http://pen-testing.sans.org/blog/pen-testing/2013/04/25/smb-relay-demystified-and-ntlmv2-pwnage-with-python

