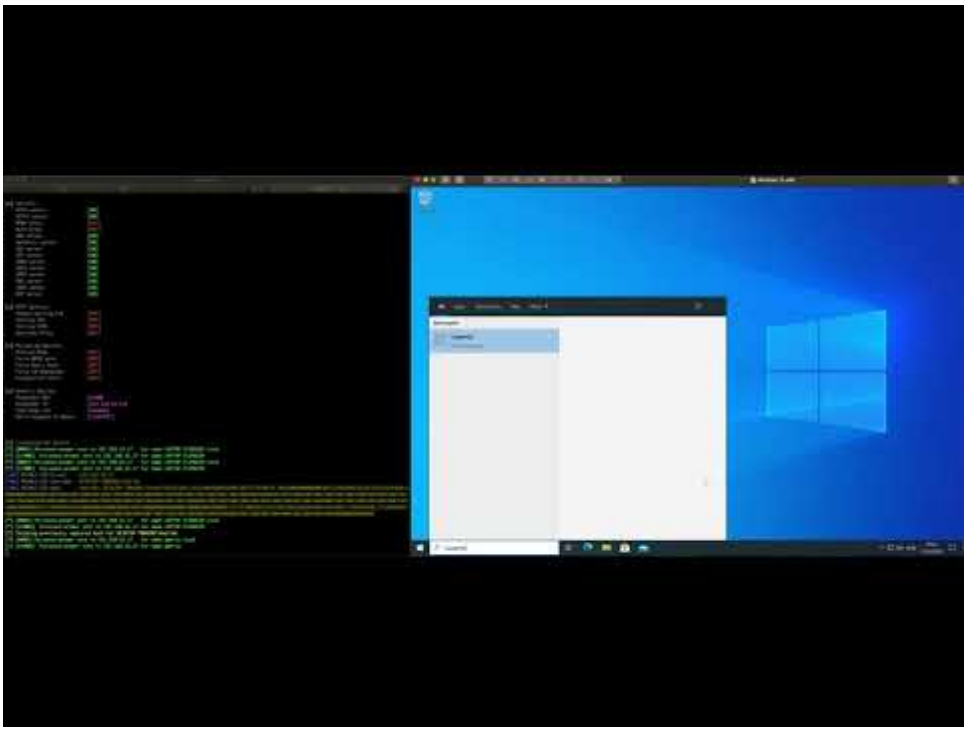
mDNS - The informal informer

```
[*] [MDNS] Poisoned answer sent to 192.168.16.158  for name m.local
[*] [MDNS] Poisoned answer sent to 192.168.16.158  for name mo.local
[*] [MDNS] Poisoned answer sent to 192.168.16.158  for name mof.local
[*] [MDNS] Poisoned answer sent to 192.168.16.158  for name mofo.local
[SMB] NTLMv2-SSP Client   : 192.168.16.158
[SMB] NTLMv2-SSP Username : AF11\andreas
[SMB] NTLMv2-SSP Hash     : andreas::AF11:ade42849c798f8de:3AB1E1112FAFE0374ADF2944466CCA5E:0101000000000000080EC7F896EE7D701554FF53FCA25C06D00
0000000200080033004D004600580001001E00570049004E002D0032004D004F0039004A004300530039004100370048000400340057004900 4E002D0032004D004F0039004A00
430053003900410037004800 2E0033004D00460058002E004C004F00430041004C00030014003300 4D00460058002E004C004F00430041004C00050014003300 4D00460058002E
004C004F00430041004C000700080080EC7F896EE7D7010600040002000000080030003000000000000000010000000200000C1E69D61F5C42CFC9D42E49B8146B6C538A4BBA2
7E3A9053F7CBCB2A3D1596890A0010000000000000000000000000000000000000000000009000C006300690066007300 2F006D000000000000000000
[*] Skipping previously captured hash for AF11\andreas
[*] Skipping previously captured hash for AF11\andreas
[*] Skipping previously captured hash for AF11\andreas
[*] [MDNS] Poisoned answer sent to 192.168.16.158  for name mofo.local
[*] Skipping previously captured hash for AF11\andreas
[*] [MDNS] Poisoned answer sent to 192.168.16.158  for name mofo.local
[*] Skipping previously captured hash for AF11\andreas
```

(Snitches Get Stitches)

I made a little POC video back in April, showing the dangers of leaving LLMNR enabled on your Windows computers.



Watch Video At: https://youtu.be/GUr4U8irXZ0

In this video, we can see there is a "feature" in Windows, making the computer sending LLMNR broadcast on every character, if you start typing a UNC path. If you place a computer running Responder on the local network, it will response to every single one of that characters, pretending to be a computer asking for authentication.

Your Windows computer gladly hands out the netNTLM-hash of the logged on user. Those hashes could easily ble cracked by an attacker if the password is easy or/and short, but it can also be relayed to other services on your network, giving the attacker access to them, without even bothering trying to crack the hash.

There have been written tons about this on blogs/articles around on the Internet. If you're not familiar with it, you can read more about Responder and relaying in this excellent article.

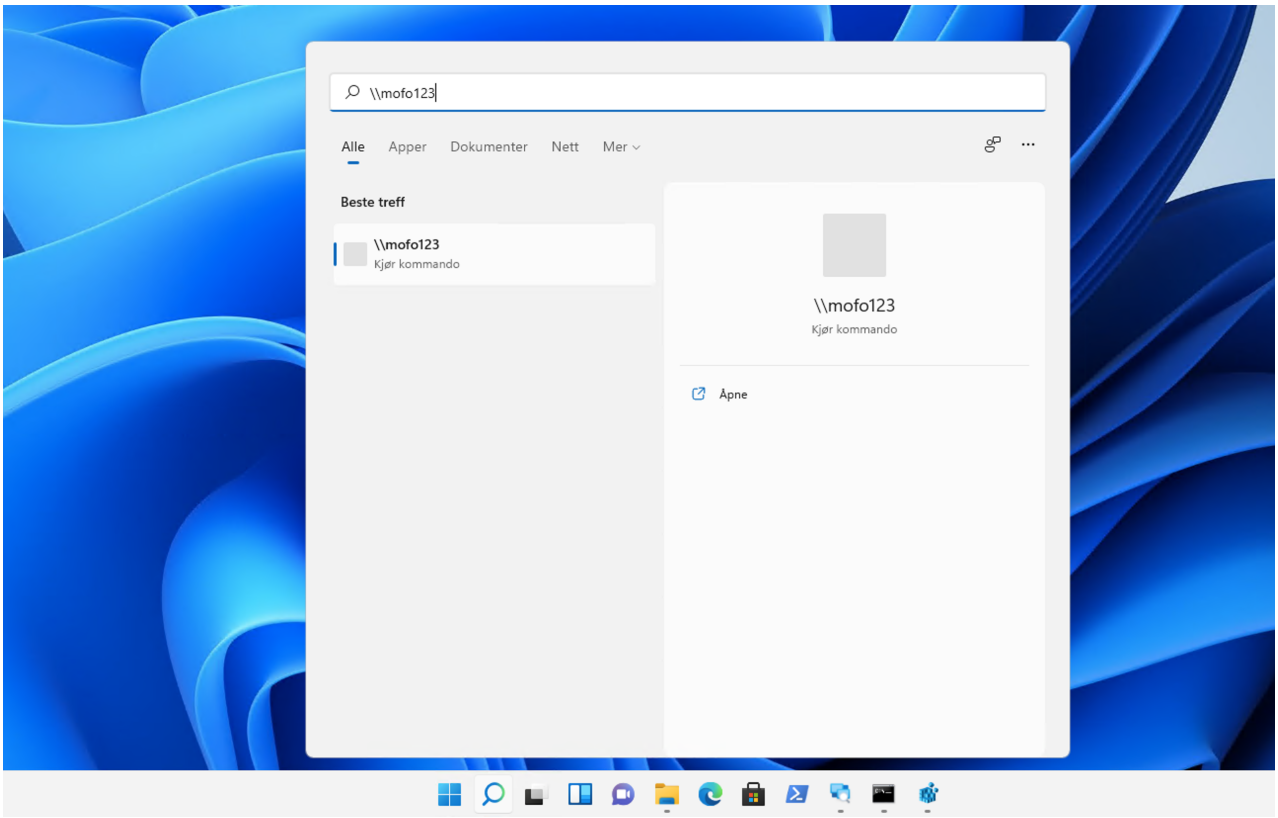
The same problem exists if your computer har NBT-NS (Netbios Name-Service) enabled. So, one should absolutely disable those two protocols, as they are really not needed in a corporate network.

This great article, will tell you how to go around to get that task done.

The other day I read this post on LinkedIn, showing you could use a tool named mitm6, to set up a rogue IPv6 DHCP server, and then be able to catch the hashes over IPv6 instead.

That sounded really scary and was news for me.

I had to see this for myself, so I put up Responder on a computer in my lab network. I first had to verify everything was locked down as it should, and did a testrun from a freshly installed Windows 11 with LLMNR and NBT-NS disabled.



I was in for a surprise...

I didn't even get around to start mitm6! The hashes just started to furiously hit my Responder. Something had clearly changed since the last time I visited the topic. This was really strange and unexpected behaviour.

```
[+] Listening for events ...

[*] [MDNS] Poisoned answer sent to 192.168.200.163 for name mofo1.local
[*] Skipping previously captured hash for WINDCORP\Administrator
[*] [MDNS] Poisoned answer sent to 192.168.200.163 for name mofo1.local
[*] [MDNS] Poisoned answer sent to 192.168.200.163 for name mofo12.local
[*] Skipping previously captured hash for WINDCORP\Administrator
[*] [MDNS] Poisoned answer sent to 192.168.200.163 for name mofo1.local
[*] Skipping previously captured hash for WINDCORP\Administrator
[*] [MDNS] Poisoned answer sent to 192.168.200.163 for name mofo12.local
[*] [MDNS] Poisoned answer sent to 192.168.200.163 for name mofo123.local
[*] Skipping previously captured hash for WINDCORP\Administrator
[*] Skipping previously captured hash for WINDCORP\Administrator
[*] [MDNS] Poisoned answer sent to 192.168.200.163 for name mofo12.local
[*] Skipping previously captured hash for WINDCORP\Administrator
[*] [MDNS] Poisoned answer sent to 192.168.200.163 for name mofo123.local
[*] Skipping previously captured hash for WINDCORP\Administrator
[*] Skipping previously captured hash for WINDCORP\Administrator
[*] [MDNS] Poisoned answer sent to 192.168.200.163 for name mofo123.local
[*] Skipping previously captured hash for WINDCORP\Administrator
```

This time, it looked like mDNS was the culprit!? I wasn't even aware Microsoft had implemented native mDNS support, but apparently they did at some point in Windows 10 and of course Windows 11 and Windows server 2022 has it enabled by default too!

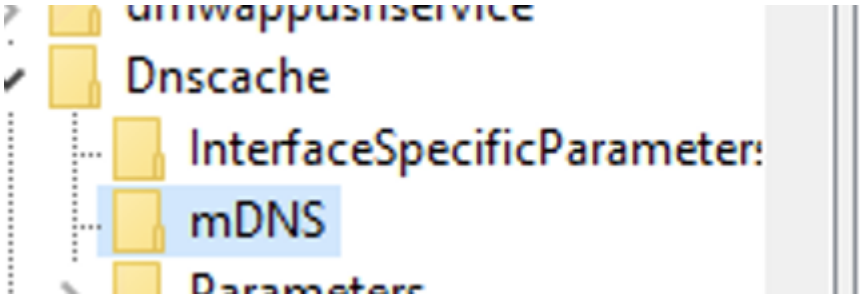
I testet it out on a Windows 2022 Domain controller and a Windows 10 computer too. Hashes all over! mDNS had no qualms about help telling anyone interested, what my netNTLM-hash looked like...

Trying to get to the core of the problem, I inspected services running on the computer and figured out mDNS is a part of the DNSCache service, now.

svchost.exe	1056	UDP	0.0.0.0	3389	*	03.12.2021 17:28:40 TermService
svchost.exe	2788	UDP	0.0.0.0	4500	*	03.12.2021 17:28:44 IKEEXT
svchost.exe	5384	UDP	0.0.0.0	5050	*	03.12.2021 17:30:45 CDPSvc
svchost.exe	1260	UDP	0.0.0.0	5353	*	03.12.2021 17:28:39 Dnscache
svchost.exe	1300	UDP	127.0.0.1	53868	*	03.12.2021 17:28:40 netprofm
svchost.exe	2164	UDP	127.0.0.1	58767	*	03.12.2021 17:28:42 LanmanWorkstation

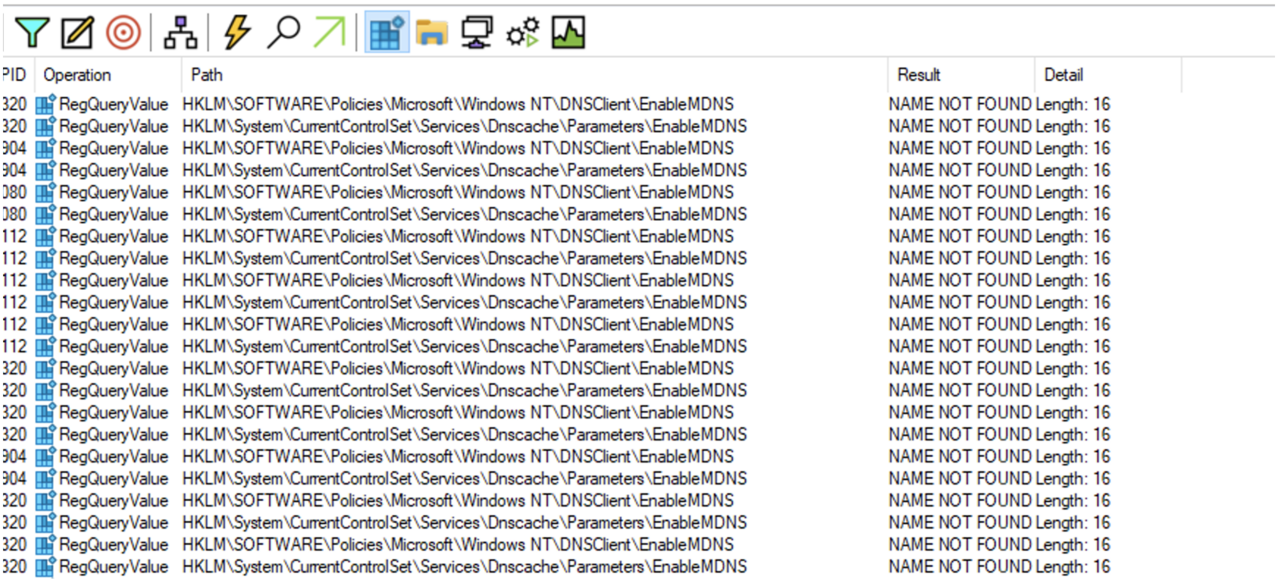
Disabling DNSCache is not an option. (tried it) It stops mDNS, but it also breaks a lot of other stuff.

Checking the registry, it turns out there is only a key named mDNS under the DNSCache service, nothing else. No parameters to be set/changed.

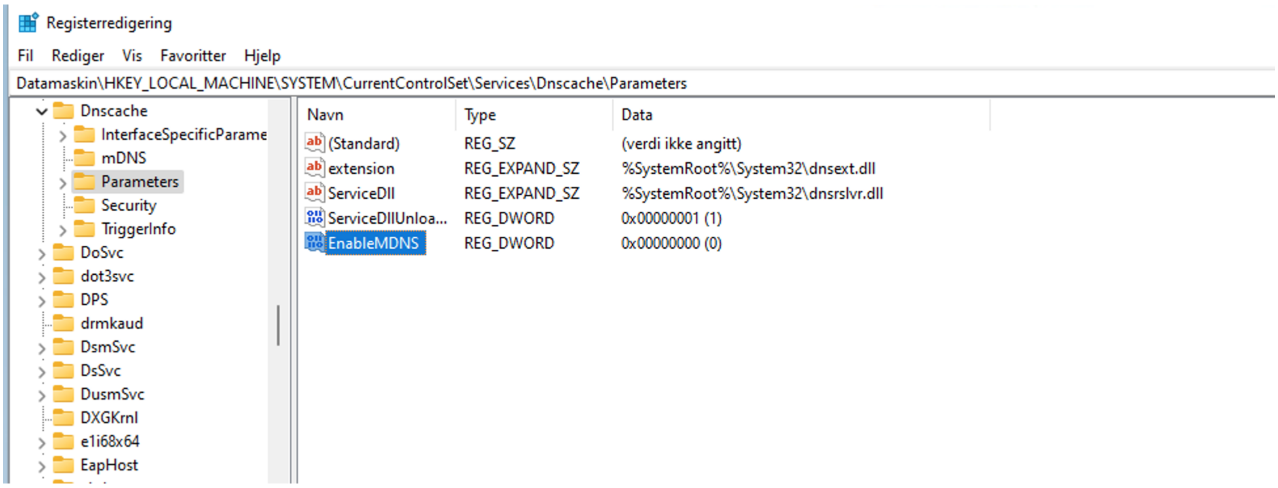


Spent a whole day and evening Googling for parameters, but couldn't find anything documented anywhere about the "new" mDNS feature.

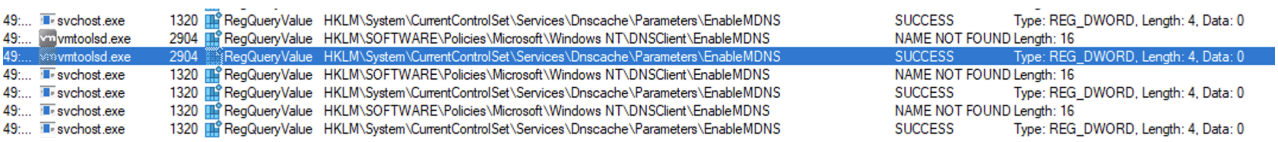
Using Procmon from Sysinternals, I saw the DNSCache process queriyng the registry for some nonexisting entries.



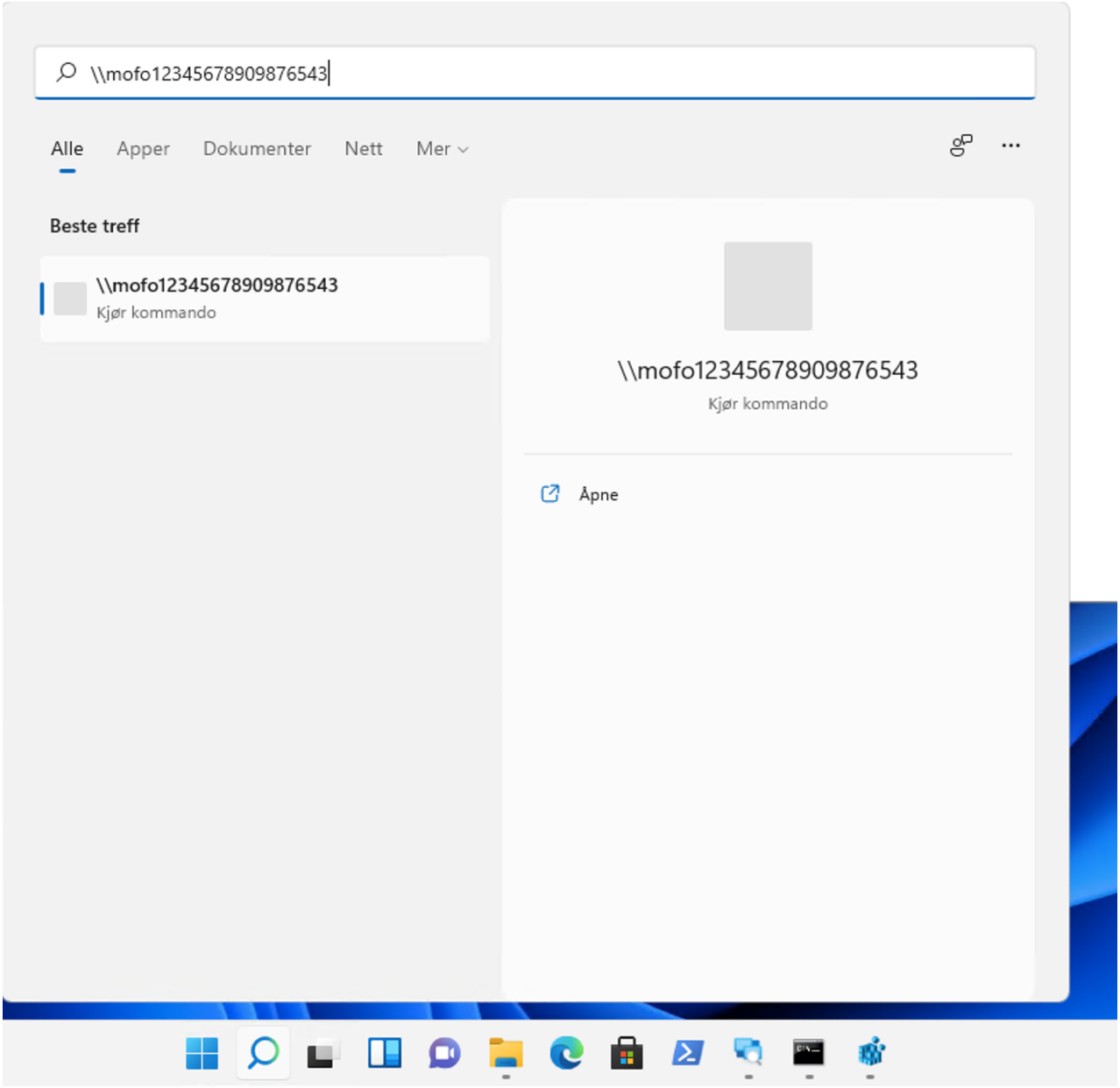
EnableMDNS sounded exactly like what i was looking for. I added the entry, with a value of 0.



Confirming it being found by the DNSCaching service



Then testing again from my Windows 11 computer.



And....

Nothing...

mDNS stopped its promiscous behaviour and everything was again good in the LANd.

My two cents: mDNS doesn't serve any purpose at all in a corporate network and should be disabled the instance you join the domain.

Don't think Microsoft have implemented control over the feature in an GPO yet.

How to disable mDNS using Powershell

```
set-ItemProperty "HKLM:\SYSTEM\CurrentControlSet\Services\Dnscache\Parameters\" -
Name EnableMDNS -Value 0 -Type DWord
```

How to disable mDNS using the reg command

```
REG ADD  "HKLM\SYSTEM\CurrentControlSet\Services\Dnscache\Parameters" /v "
EnableMDNS" /t REG_DWORD /d "0" /f
```

It is worth to mention; This mDNS behaviour is seen only in Windows domain environments. Not if you are running it in a Workgroup. This is really strange and I don't know why. If anyone does, I would be happy to be enlightened.