



Interceptor-NG
sniffing since 2006...

SMB Hijacking. Kerberos is defeated.

© Ares, June 2013

interceptor.mail@gmail.com
<http://sniff.su>



Intro

The last thing i wanted to do was to talk about SMBRelay (or actually NTLM-Relay) one more time. It was discussed many times before, even by me.

Just to remind, it is about manipulating user's credentials that can be redirected to third-party resource. This kind of manipulation (called relaying) gives us an ability to authenticate and get access to that resource. Usually an attacker is able to execute some code remotely.

Development of SMBRelay has stopped for a while. Right, it seems that everything is done and known already. A year ago Interceptor-NG learned to perform SMBRelay with NTLM version 2. After a few months NTLMv2-relay appeared in Metasploit. Well, this is the logical end. The SMBRelay itself is outdated and almost unusable. It is much easier to perform NTLM-relay against another protocols such as HTTP. Furthermore, using SMBRelay in the networks with Domain is complicated by usage of "undefeatable" Kerberos, that is usually called "the cure against SMBRelay". In this research i want to break this myth and introduce a completely new way of attack against SMB protocol. It is necessary to note that this kind of attack is not actually SMBRelay, let's call it SMB Hijacking.



Getting started

Working on the new version of Interceptor i created NTLM-response grabber in WPAD MiTM. To quickly check if the NTLM-response (and other types of hashes) were easy to crack i added an option to invoke John the Ripper right from the program. Looking through Wireshark's archive of packet captures i found one with Kerberos data inside. Cain was able to get some kind of hash from it and recover password by means of bruteforce attack. The bruteforce is possible because the AS-REQ request to authentication server contains a timestamp that is encrypted by user's password, and some part of the timestamp is known beforehand. The Kerberos dissector was added to Interceptor and tested on the live Domain Controller (Windows 2008 R2). Unfortunately nothing was intercepted.

But why? The deal is that new encryption algorithms were added to Windows 2008 and old rc4-hmac was replaced by new aes256-cts-hmac-sha1-96. After little modifications new hash type was intercepted, but a new problem occurred. There are no suitable tools to recover password from aes-encrypted timestamp. On the openwall's maillist i found that someone coded a patch for JTR (john the ripper), but it still seems kinda raw. Moreover, bruteforcing aes is hundred times slower than rc4.

That's how Kerberos Downgrade feature appeared in Interceptor-NG. Old DES encryption was completely disabled starting from Windows 7\2008 R2, hence there was no reason to downgrade aes down to des. The lowest possible algorithm is rc4 now.

It became my starting point to dig into Kerberos a little bit deeper.

Realtek PCIe FE Family Controller: \Device\NPF_{03F0C0EF-E1C2-4501-8F08-7FEC8AD0C5AD} [Wireshark...]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: **kerberos** Expression... Clear

Length	Info
269	AS-REQ
269	AS-REQ
276	KRB Error: KRB5KDC_ERR_PREAUTH_REQUIRED
276	KRB Error: KRB5KDC_ERR_PREAUTH_REQUIRED
345	AS-REQ
345	AS-REQ
1423	AS-REP
1423	AS-REP
1369	TGS-REQ

Encryption Types: des-cbc-md5 des-cbc-md5 rc4-hmac rc4-hmac-exp rc4-hmac-old-exp

- Encryption type: des-cbc-md5 (3)
- Encryption type: des-cbc-md5 (3)
- Encryption type: rc4-hmac (23)
- Encryption type: rc4-hmac-exp (24)
- Encryption type: rc4-hmac-old-exp (-135)
- Encryption type: des-cbc-md5 (3)

HostAddresses: ADMIN<20>

0090	a0 03 02 01 02 a1 12 30 10 1b 06 6b 72 62 74 670 ...krbtg
00a0	74 1b 06 54 45 53 54 44 43 a5 11 18 0f 32 30 33	t..TESTD C....203
00b0	37 30 39 31 33 30 32 34 38 30 35 5a a6 11 18 0f	70913024 805Z....
00c0	32 30 33 37 30 39 31 33 30 32 34 38 30 35 5a a7	20370913 024805Z.
00d0	06 02 04 2e 7b 31 be a8 15 30 13 02 01 03 02 01{1... .0.....
00e0	03 02 01 17 02 01 18 02 02 ff 79 02 01 03 a9 1dY.....
00f0	30 1b 30 19 a0 03 02 01 14 a1 12 04 10 41 44 4d	0.0.....ADM
0100	49 4e 20 20 20 20 20 20 20 20 20 20 20 20 20	IN

This is a list of Kerberos encryption types (kerberos.etypes), 19 bytes

The downgrade is easy: just replace available encryption types in outgoing packet.



SMB Hijacking

So, reading different documents about Kerberos i've learned that it is vulnerable to the some kind of replay attacks. However i found only two real tools that perform evil tricks with Kerberos tickets: an old one kdcspooof, by Dug Song and a tool by Emmanuel Bouillon. It is no surprise, dealing with Kerberos is really hard when theory comes to the practice.

After deep thoughts and some traffic analysis i asked myself a question: **why do we need to replay anything at all? Actually, we dont!**

See what the typical SMB session is:

- (1). Host_A connects to Host_B
- (2). Session protocol negotiation
- (3). Authentication protocol negotiation
- (4). SMB commands...

We don't have to redirect user's credentials to another resource because the user can authenticate himself !

All we have to do is to stay in the middle and proxify connection between Host_A and Host_B and then take over control of the session into our own hands.

At the step (4) we already authenticated and can inject our own commands. Bingo! It absolutely doesn't matter how the user authenticated to the system, using NTLM or Kerberos. The packets of SMB session are not encrypted and SMB Signing is not used between common computers (only DC). In addition to greater abilities, this technique is more elegant than SMBRelay. There is no need to code a pretty complex SMB authentication algorithm. Very little amount of SMB commands is necessary to be coded, only those to perform file uploading and service execution.

The functionality described above was added to the new version of Interceptor-NG and was tested on latest versions of Windows 2008 and Windows 7\8.

Let's see what's been done to accomplish this task...

Starting from Windows Vista a new SMB version 2 appeared. It was meant to simplify the command set and improve performance. The packet structure is also different from the previous SMB implementation. Two new fields are especially important for us: the Session ID and the Command Sequence Number. To inject our own commands we have to track the ID and increment command counter every new step.

Existing SMBRelay code in Interceptor is based on smbrelay3 by Tarasco Security and it uses an old SMB format. That's why i decided to create a completely new code for SMBv2.

The logic of the injection process is the following:

1. Get in the middle between the target and a third-party host
2. Wait for successful SessionSetup Response
3. Get Session ID and current command's number
4. Upload a file to administrative share (admin\$)
5. Create a service that will execute it
6. Run the service
7. Profit!

The 5 and 6 steps are possible because Microsoft implemented a transfer of RPC calls over SMB protocol.

That's all. This technique is perfect for the common domain based networks. Usually, there is at least one centralized software that connects to the user's shares with administrative privileges. You can also choose administrator as a target.

We start injection after SessionSetup command.

The image shows a Wireshark network traffic capture window. The title bar indicates the interface is 'Realtek PCIe FE Family Controller: \Device\NPF_{03F0C0EF-E1C2-4501-8F08-7FEC8AD0C5AD}' with a filter set to 'smb2'. The packet list pane shows several SMB2 messages, including 'SessionSetup Response', 'TreeConnect Request Tree: \\192.16.1.35\admin\$', 'TreeConnect Response', 'Create Request File: smrs.exe', 'Create Response File: smrs.exe', 'write Request Len:2048 Off:0 File: smrs.exe', 'write Response', and 'TreeConnect Request Tree: \\192.16.1.35\IPC\$'. The packet details pane for the selected 'krb5_blob' (138 bytes) shows the following structure:

- Length: 184
- GSS-API Generic Security Service Application Program Interface
 - Simple Protected Negotiation
 - negTokenTarg
 - negResult: accept-completed (0)
 - supportedMech: 1.2.840.48018.1.2.2 (MS KRB5 - Microsoft Kerberos 5)
 - responseToken: 60819706092a864886f71201020202006f8187308184a003...
 - krb5_blob: 60819706092a864886f71201020202006f8187308184a003...

The packet bytes pane shows the raw data in hexadecimal and ASCII. The ASCII column contains the following text:

```
.....*.H .....  
o..0.... .....  
.x0v.... ..o.m.04  
.2.a|.. [ .f... ("
```

The status bar at the bottom indicates 'krb5_blob (spnego.krb5.blob), 138 bytes' and 'Packets:...'.



Conclusion

Thus the Kerberos is "defeated" or in fact avoided. By the way, Kerberos is used only if the SMB connection established by computer name. In case of IP address, NTLM negotiation will occur.

SMB Hijacking is also suitable for a workgroups, just make sure that your target has access to the third-party host. Earlier i discussed a "passive" way of attack, however Interceptor speeds up the process by means of injecting UNC link of the third-party host into target's pc web traffic.

Whether your attack was successful or not you have to break hijacked connection, because it's no longer usable for target user - command sequence is broken. To avoid deadlock situation of unstoppable hijacking and let the user access remote resource without problems and suspicions Interceptor marks the connection and lets it pass through.

The only effective way of preventing SMB Hijacking is using SMB Signing: an attacker will not be able to create a valid signature of injected commands, thus the server will reject them.