

## Task A: Linux Password Cracking (25 points)

1. 5 points. Create two groups, one is cyse301, and the other is your ODU Midas ID (for example, svatsa). Then display the corresponding group IDs.

```
(root@kali)-[~]
└─# whoami
root

(root@kali)-[~]
└─# groupadd cyse301

(root@kali)-[~]
└─# groupadd secon001

(root@kali)-[~]
└─# getent group cyse301
cyse301:x:1002:

(root@kali)-[~]
└─# getent group secon001
secon001:x:1003:
```

2. 5 points. Create and assign three users to each group. Display related UID and GID information of each user.

```
(root@kali)-[~]
└─# useradd -m -g cyse301 testuser001

(root@kali)-[~]
└─# useradd -m -g cyse301 testuser002

(root@kali)-[~]
└─# useradd -m -g cyse301 testuser003

(root@kali)-[~]
└─# useradd -m -g secon001 testuser004

(root@kali)-[~]
└─# useradd -m -g secon001 testuser005

(root@kali)-[~]
└─# useradd -m -g secon001 testuser006
```

```
(root@kali)-[~]
└─# id testuser001
uid=1002(testuser001) gid=1002(cyse301) groups=1002(cyse301)

(rroot@kali)-[~]
└─# id testuser002
uid=1003(testuser002) gid=1002(cyse301) groups=1002(cyse301)

(rroot@kali)-[~]
└─# id testuser003
uid=1004(testuser003) gid=1002(cyse301) groups=1002(cyse301)

(rroot@kali)-[~]
└─# id testuser004
uid=1005(testuser004) gid=1003(secon001) groups=1003(secon001)

(rroot@kali)-[~]
└─# id testuser005
uid=1006(testuser005) gid=1003(secon001) groups=1003(secon001)

(rroot@kali)-[~]
└─# id testuser006
uid=1007(testuser006) gid=1003(secon001) groups=1003(secon001)
```

3. 5 points. Choose Three new passwords, from easy to hard, and assign them to the users you created. You need to show me the password you selected in your report, and DO NOT use your real-world passwords.

```
(root@kali)-[~]
└─# passwd testuser001
New password:
Retype new password:
passwd: password updated successfully
```

```
(root@kali)-[~]
└─# passwd testuser002
New password:
Retype new password:
passwd: password updated successfully
```

```
(root@kali)-[~]
└─# passwd testuser003
New password:
Retype new password:
Sorry, passwords do not match.
passwd: Authentication token manipulation error
passwd: password unchanged
```

```
(root@kali)-[~]
└─# passwd testuser003
New password:
Retype new password:
passwd: password updated successfully
```

```
(root@kali)-[~]
└─# passwd testuser004
New password:
Retype new password:
passwd: password updated successfully
```

```
(root@kali)-[~]
└─# passwd testuser005
New password:
Retype new password:
passwd: password updated successfully

(root@kali)-[~]
└─# passwd testuser006
New password:
Retype new password:
passwd: password updated successfully
```

4. 5 points. Export all Three users' password hashes into a file named "YourMIDAS-HASH" (for example, svatsa-HASH). Then launch a dictionary attack to crack the passwords. You MUST crack at least one password in order to complete this assignment.

```
root@kali: ~
File Actions Edit View Help
GNU nano 7.2 secon001-HASH *
testuser001:$y$j9T$q2rrnGxTnGn1zU8GF63Uj/$n8flZmQglnfHFVzI/ulb1n6ASissA7TtNvocKDaJtG5
testuser002:$y$j9T$KNYbn1PTj8sLPGAJXJH89.$EtmUVc8C0QE8zIa95fR4J6fE/xdCCZFq/hKwSs5wPwA
testuser003:$y$j9T$0bGC5Q/xPxPaCQGg.j07N1$M8XcWqsncFHikqedaudy.muffRaC0/yoGK7IEBkxoA
```

```
(root@kali)-[~]
└─# cat secon001-HASH
testuser001:$y$j9T$q2rrnGxTnGn1zU8GF63Uj/$n8flZmQglnfHFVzI/ulb1n6ASissA7TtNvocKDaJtG5
testuser002:$y$j9T$KNYbn1PTj8sLPGAJXJH89.$EtmUVc8C0QE8zIa95fR4J6fE/xdCCZFq/hKwSs5wPwA
testuser003:$y$j9T$0bGC5Q/xPxPaCQGg.j07N1$M8XcWqsncFHikqedaudy.muffRaC0/yoGK7IEBkxoA
```

```
(root@kali)-[~]
└─# john --format=sha512crypt --wordlist=/root/Desktop/VMshare/rockyou.txt secon001-HASH
Using default input encoding: UTF-8
Loaded 3 password hashes with 3 different salts (sha512crypt, crypt(3) $6$ [SHA512 512/512 AVX512BW 8x])
Cost 1 (iteration count) is 5000 for all loaded hashes
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
password1 (testuser002)
1234 (testuser001)
2g 0:00:00:38 1.55% (ETA: 01:42:04) 0.05249g/s 6866p/s 6920c/s 6920C/s ronnie29..rebel92
Use the "--show" option to display all of the cracked passwords reliably
Session aborted

(root@kali)-[~]
└─# john --show secon001-HASH
testuser001:1234
testuser002:password1

2 password hashes cracked, 1 left
```

## Task B: Windows Password Cracking (25 points)

Log on to Windows 7 VM and create a list of 3 users with different passwords (OR you may create users using net users \add command as you did in lab-4-task-c). Then you need to establish a reverse shell connection with the admin privilege to the target Windows 7 VM.

```
Administrator: Command Prompt
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Windows\system32>net user winuser01 password1 /add
The command completed successfully.

C:\Windows\system32>net user winuser02 football /add
The command completed successfully.

C:\Windows\system32>net user winuser03 letmein /add
The command completed successfully.

C:\Windows\system32>net user

User accounts for \\WINDOWS7

-----
Administrator          Guest                  Window 7
winuser01              winuser02             winuser03
The command completed successfully.
```

Now, complete the following tasks:

1. 5 points. Display the password hashes by using the “hashdump” command in the meterpreter shell. Then

```
[*] Starting interaction with 2 ...

meterpreter > getsystem
... got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0 :::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0 :::
HomeGroupUser$:1002:aad3b435b51404eeaad3b435b51404ee:2d79c7f57c09bad3139f56290e444b23 :::
Window 7:1000:aad3b435b51404eeaad3b435b51404ee:8846f7eaae8fb117ad06bdd830b7586c :::
winuser01:1003:aad3b435b51404eeaad3b435b51404ee:5835048ce94ad0564e29a924a03510ef :::
winuser02:1004:aad3b435b51404eeaad3b435b51404ee:31fc0dc8f7dfad0e8bd7ccc3842f2ce9 :::
winuser03:1005:aad3b435b51404eeaad3b435b51404ee:becedb42ec3c5c7f965255338be4453c :::
meterpreter > █
```

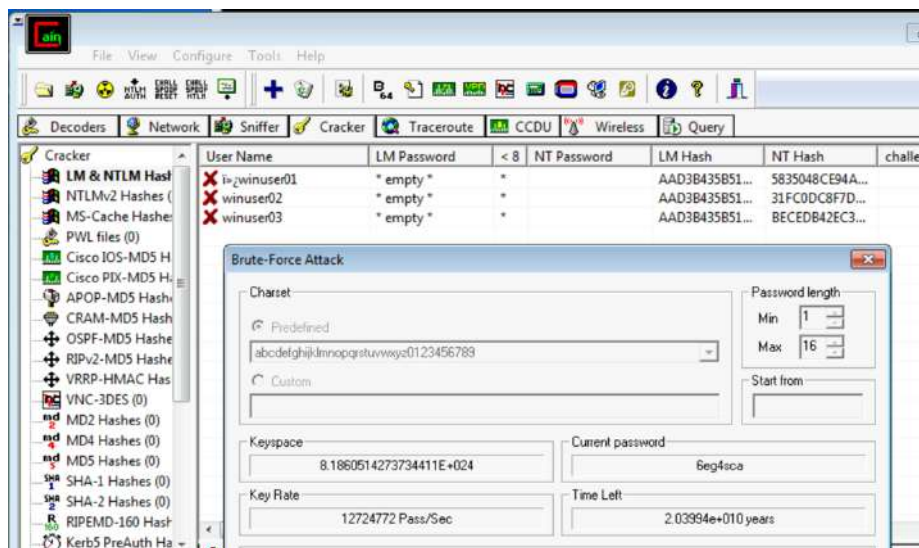
2. 10 points. Save the password hashes into a file named “your\_midass.WinHASH” in Kali Linux (you need to replace the “your\_midass” with your university MIDAS ID). Then run John the ripper for 10 minutes to crack the windows users’ passwords (You MUST crack at least one password in order to complete this assignment.).

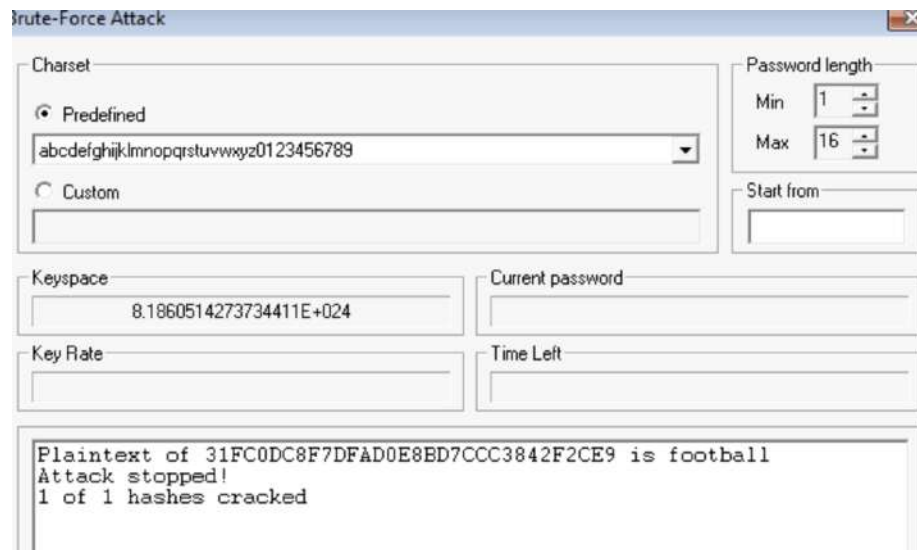
```
(root@kali)-[~]
└─# john --format=NT secon001.WinHASH
Using default input encoding: UTF-8
Loaded 3 password hashes with no different salts (NT [MD4 512/512 AVX512BW 16x3])
Warning: no OpenMP support for this hash type, consider --fork=2
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
password1      (winuser01)
letmein        (winuser03)
football       (winuser02)
3g 0:00:00:00 DONE 2/3 (2025-11-26 02:51) 18.75g/s 18225p/s 18225c/s 54675C/s 123456..larry
Use the "--show --format=NT" options to display all of the cracked passwords reliably
Session completed.

(root@kali)-[~]
└─# john --show --format=NT secon001.WinHASH
winuser01:password1:1003:aad3b435b51404eeaad3b435b51404ee:5835048ce94ad0564e29a924a03510ef:::
winuser02:football:1004:aad3b435b51404eeaad3b435b51404ee:31fc0dc8f7dfad0e8bd7ccc3842f2ce9:::
winuser03:letmein:1005:aad3b435b51404eeaad3b435b51404ee:becedb42ec3c5c7f965255338be4453c:::

3 password hashes cracked, 0 left
```

3. 10 points. Launch/open the password cracking tool, Cain and Abel in Windows 7 VM, via a remote desktop window. Then, implement BOTH brute force and dictionary attacks to crack the passwords for Windows7 users. (You MUST crack at least one password in order to complete this assignment).





NOTE: Please refer to the class lecture to learn how to add users in windows7 and using Cain tool for windows password cracking

Extra credit: (10 points)

Search the proper format in John the Ripper to crack the following MD5 hashes (use the --list=formats option to list all supported formats). Show your steps and results.

1. 5f4dcc3b5aa765d61d8327deb882cf99

2. 63a9f0ea7bb98050796b649e85481845

```
(root@kali)-[~]
└─# nano md5list.txt

(root@kali)-[~]
└─# john --show --format=Raw-MD5 md5list.txt
0 password hashes cracked, 2 left

(root@kali)-[~]
└─# nano md5list.txt

(root@kali)-[~]
└─# john --format=Raw-MD5 md5list.txt
Using default input encoding: UTF-8
Loaded 2 password hashes with no different salts (Raw-MD5 [MD5 512/512 AVX512BW 16x3])
Warning: no OpenMP support for this hash type, consider --fork=2
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
password      (?)
Proceeding with incremental:ASCII
root          (?)
2g 0:00:00:01 DONE 3/3 (2025-11-26 04:00) 1.680g/s 4730Kp/s 4730Kc/s 4730KC/s 0102000101..rams
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed.

(root@kali)-[~]
└─# john --show --format=Raw-MD5 md5list.txt
?:password
?:root

2 password hashes cracked, 0 left
```

## Part 2

### 1. Decrypting *lab5wep-demo.cap* (5 points)

For the WEP file, I used **aircrack-ng** to test the captured IVs. The command successfully recovered the WEP key:

**Key:** F2:C7:BB:35:B9

```
Aircrack-ng 1.7

[00:00:01] Tested 231 keys (got 19772 IVs)

KB   depth  byte(vote)
0    0/ 2    F2(28928) 7A(27136) 30(26112) 21(24832) 27(24832) 03(24576) F8(24576) 05(24320) 38(24064)
1    9/ 10   C7(24064) 71(23808) 5C(23552) 20(23296) 2A(23296) 52(23296) 84(23296) 99(23040) DE(23040)
2    0/ 1    BB(30208) AB(25344) BF(25344) D0(24832) 08(24576) 93(24576) CC(24320) D3(24064) 09(23808)
3    8/ 12   FC(24064) 25(23808) 2A(23808) A9(23808) BD(23808) 00(23552) 42(23552) 3F(23296) 62(23296)
4    0/ 1    B9(30720) 33(26624) 2E(25344) C4(25344) 64(25088) ED(25088) 55(24832) 77(24832) 9C(24576)

KEY FOUND! [ F2:C7:BB:35:B9 ]
Decrypted correctly: 100%

(root@kali) ~ - [~/Desktop/Lab Resources (2023 Spring)/Lab Resources/Module 5]
```

### 2. Decrypting *lab5wpa2-demo.cap* (5 points)

For the WPA2 demo file, I ran a dictionary attack using:

Aircrack found the WPA2 password successfully.

**Password:** password

I placed this password into Wireshark using the WPA-PWD method (password:SSID).

After applying the key, Wireshark began decrypting the 4-way EAPOL handshake as well as encrypted data.

```
root@kali: ~/Desktop/Lab Resources (2023 Spring)/Lab Resources/Module 5
File Actions Edit View Help

Aircrack-ng 1.7

[00:00:00] 24/14344392 keys tested (152.35 k/s)
Time left: 1 day, 2 hours, 9 minutes, 12 seconds 0.00%

KEY FOUND! [ password ]

Master Key      : 20 64 DE 6A 2E 73 86 96 81 91 8E 8C 1E 32 49 FC
                  3B C9 0A 44 BC 2B 6E 94 45 4B BF 8F B9 79 FC 3B

Transient Key   : 48 5D 7F 5E F5 AA 69 76 D8 85 83 31 FA 2A 65 A4
                  C0 A0 D1 4A 96 BC C5 96 65 7A FC A2 44 94 14 51
                  EC 9C 42 51 E1 EA BF AE 5F BB 64 11 0D 60 70 24
                  77 81 71 A3 2C 1B BC D1 0A 1C BF 1C EC 00 00 00

EAPOL HMAC     : 49 94 2C 92 12 04 BA 66 ED D8 40 0F 10 A5 19 47

root@kali)~[~/Desktop/Lab Resources (2023 Spring)/Lab Resources/Module 5]
```

### 3. Detailed WPA2 Traffic Analysis (5 + 5 points)

Once decrypted, I could see:

- The **full 4-way WPA2 EAPOL handshake** (Messages 1–4)
- Normal wireless traffic like management frames (probe requests, beacons)
- Client identity/authentication exchanges
- Encrypted data frames between the client and AP

The EAPOL handshake clearly showed how the client and AP negotiated keys, confirming that decryption worked correctly.

The screenshot shows the Wireshark interface with a packet list table for EAPOL traffic. The table has columns for No., Time, Source, Destination, Protocol, Length, and Info. The traffic consists of multiple EAPOL Key messages between CiscoLinksys and Microsoft devices.

No.	Time	Source	Destination	Protocol	Length	Info
139	5.447042	CiscoLinksys_da:cf:...	Microsoft_ca:e7:60	EAPOL	133	Key (Message 1 of 4)
141	5.449091	Microsoft_ca:e7:60	CiscoLinksys_da:cf:...	EAPOL	155	Key (Message 2 of 4)
143	6.445505	CiscoLinksys_da:cf:...	Microsoft_ca:e7:60	EAPOL	133	Key (Message 1 of 4)
145	6.451651	Microsoft_ca:e7:60	CiscoLinksys_da:cf:...	EAPOL	155	Key (Message 2 of 4)
161	7.496187	Microsoft_ca:e7:60	CiscoLinksys_da:cf:...	EAPOL	155	Key (Message 2 of 4)
170	8.518713	CiscoLinksys_da:cf:...	Microsoft_ca:e7:60	EAPOL	133	Key (Message 1 of 4)
174	8.555075	Microsoft_ca:e7:60	CiscoLinksys_da:cf:...	EAPOL	155	Key (Message 2 of 4)
179	9.644160	CiscoLinksys_da:cf:...	Microsoft_ca:e7:60	EAPOL	133	Key (Message 1 of 4)
182	9.650296	Microsoft_ca:e7:60	CiscoLinksys_da:cf:...	EAPOL	155	Key (Message 2 of 4)
183	9.651833	Microsoft_ca:e7:60	CiscoLinksys_da:cf:...	EAPOL	155	Key (Message 2 of 4)
348	14.066552	CiscoLinksys_da:cf:...	Microsoft_ca:e7:60	EAPOL	133	Key (Message 1 of 4)
350	14.068611	Microsoft_ca:e7:60	CiscoLinksys_da:cf:...	EAPOL	155	Key (Message 2 of 4)
352	14.077816	CiscoLinksys_da:cf:...	Microsoft_ca:e7:60	EAPOL	189	Key (Message 3 of 4)
353	14.080376	CiscoLinksys_da:cf:...	Microsoft_ca:e7:60	EAPOL	189	Key (Message 3 of 4)
354	14.082423	CiscoLinksys_da:cf:...	Microsoft_ca:e7:60	EAPOL	189	Key (Message 3 of 4)
355	14.087031	CiscoLinksys_da:cf:...	Microsoft_ca:e7:60	EAPOL	189	Key (Message 3 of 4)
356	14.089080	CiscoLinksys_da:cf:...	Microsoft_ca:e7:60	EAPOL	189	Key (Message 3 of 4)
358	14.092675	Microsoft_ca:e7:60	CiscoLinksys_da:cf:...	EAPOL	133	Key (Message 4 of 4)

## Task D – Assigned WPA2 File (30 points)

### 1. Dictionary Attack & Decryption (20 points)

Based on the MD5-hash last digit rules in the instructions, my assigned file was:

**WPA2-P1-01.cap**

I ran the dictionary attack using:

```
aircrack-ng WPA2-P1-01.cap -w /root/Desktop/VMshare/rockyou.txt
```

Aircrack found the key successfully:

**Password:** CyberPHY

```
Quitting aircrack-ng ...

(root@kali)-[~/Desktop/Lab Resources (2023 Spring)/Lab Resources/Module 5]
└─# aircrack-ng WPA2-P1-01.cap
Reading packets, please wait ...
Opening WPA2-P1-01.cap
Inter-frame timeout period exceeded.
Read 2660 packets.

# BSSID          ESSID          Encryption
1 00:16:B6:DA:CF:2F CyberPHY       WPA (1 handshake)

Choosing first network as target.

Reading packets, please wait ...
Opening WPA2-P1-01.cap
Inter-frame timeout period exceeded.
Read 2660 packets.

1 potential targets

Please specify a dictionary (option -w).
```

## 2. Traffic Summary After Decryption (10 points)

After entering the correct key, the encrypted WPA2 traffic became readable. I filtered using **eapol** to view the WPA2 4-way handshake. The handshake messages (1–4) appeared clearly with matching source/destination MAC addresses.

Other things I observed in the decrypted capture:

- Management frames (beacons, probe requests, probe responses)
- Multiple EAPOL Key frames between a Microsoft client and a Cisco AP
- Data frames showing normal encrypted WLAN communication
- TLSv1 traffic after authentication, indicating the user accessed a secure service

Overall, decrypting the assigned file allowed me to see the full WPA2 authentication process and the early application-layer activity after the connection was established.

Wireshark capture window titled 'eapol' showing a list of frames and a detailed view of the first frame.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	CiscoLinksys_da:cf:...	MotorolaMobi_5a:df:...	802.11	130	Probe Response, SN=530, FN=0, Flags=....R..., BI=100, SSII
2	-0.000001	CiscoLinksys_da:cf:...	MotorolaMobi_5a:df:...	802.11	130	Probe Response, SN=530, FN=0, Flags=....R..., BI=100, SSII
3	-0.000002	CiscoLinksys_da:cf:...	MotorolaMobi_5a:df:...	802.11	130	Probe Response, SN=530, FN=0, Flags=....R..., BI=100, SSII
4	0.000000	CiscoLinksys_da:cf:...	MotorolaMobi_5a:df:...	802.11	130	Probe Response, SN=530, FN=0, Flags=....R..., BI=100, SSII
5	0.000506		XiaomiCommun_72:56:...	802.11	10	Clear-to-send, Flags=.....
6	0.000506	Apple_b9:94:fa (70:...	XiaomiCommun_72:56:...	802.11	28	802.11 Block Ack, Flags=.....
7	0.001019	Apple_b9:94:fa (70:...	XiaomiCommun_72:56:...	802.11	16	Request-to-send, Flags=.....
8	0.001027		Apple_b9:94:fa (70:...	802.11	10	Clear-to-send, Flags=.....
9	0.001027	XiaomiCommun_72:56:...	Apple_b9:94:fa (70:...	802.11	28	802.11 Block Ack, Flags=.....
10	0.001027	XiaomiCommun_72:56:...	Apple_b9:94:fa (70:...	802.11	16	Request-to-send, Flags=.....
11	0.001018		XiaomiCommun_72:56:...	802.11	10	Clear-to-send, Flags=.....
12	0.001018	Apple_b9:94:fa (70:...	XiaomiCommun_72:56:...	802.11	28	802.11 Block Ack, Flags=.....
13	0.031748	XiaomiCommun_72:56:...	Apple_b9:94:fa (70:...	802.11	16	Request-to-send, Flags=.....
14	0.031739		XiaomiCommun_72:56:...	802.11	10	Clear-to-send, Flags=.....
15	0.031738	Apple_b9:94:fa (70:...	XiaomiCommun_72:56:...	802.11	28	802.11 Block Ack, Flags=.....
16	0.084995	XiaomiCommun_72:56:...	Apple_b9:94:fa (70:...	802.11	16	Request-to-send, Flags=.....

Frame 1: 130 bytes on wire (1040 bits), 130 bytes captured (1040 bits) on interface 0

- IEEE 802.11 Probe Response, Flags: ....R...
- IEEE 802.11 Wireless Management

```

0000  50 08 3a 01 88 79 7e 5a  df 19 00 16 b6 da cf
0010  00 16 b6 da cf 2f 20 21  ee 99 f3 02 00 00 00
0020  64 00 11 04 00 08 43 79  62 65 72 50 48 59 01
0030  82 84 8b 96 24 30 40 6c  03 01 06 2a 01 00 2f
0040  00 30 14 01 00 00 0f ac  04 01 00 00 0f ac 04
0050  00 00 0f ac 02 0c 00 32  04 0c 12 18 00 dd 09
0060  10 18 02 00 f3 00 00 00  dd 18 00 50 f2 02 01
0070  80 00 03 a4 00 00 27 a4  00 00 42 43 5e 00 62
0080  2f 00
  
```