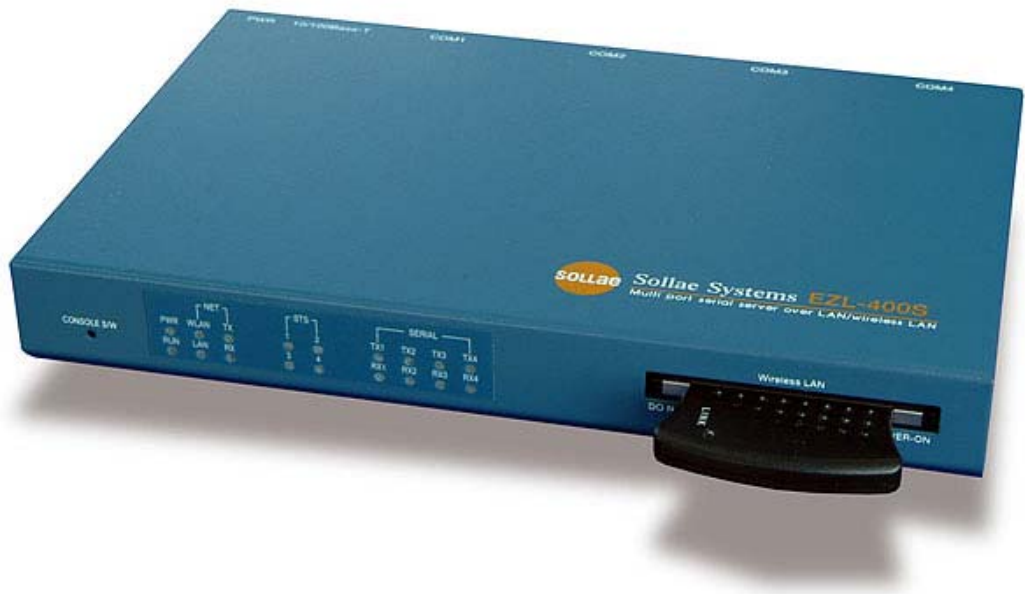


EZL-400S Application Note (002)

SSL(Secure Socket Layer)

Version 1.0



1. 클라이언트 모드에서 SSL 사용하기

클라이언트 모드에서 SSL 기능을 사용하려면 telnet으로 로그인하여 SSL 기능을 활성화시켜 주기만 하면 됩니다.

클라이언트 모드는 COD 모드 또는 ATC 모드에서 atd 명령으로 서버에 접속하는 것을 의미합니다.

SSL을 클라이언트로 사용하려면 통신 대상 서버도 SSL 을 사용해야 합니다.

1.1. SSL 기능 활성화 하기

telnet 클라이언트로 로그인 하여 SSL 기능을 활성화 할 수 있습니다. 다음은 SSL 기능을 활성화 하기 위한 예입니다.

1.1.1. IP 주소 설정

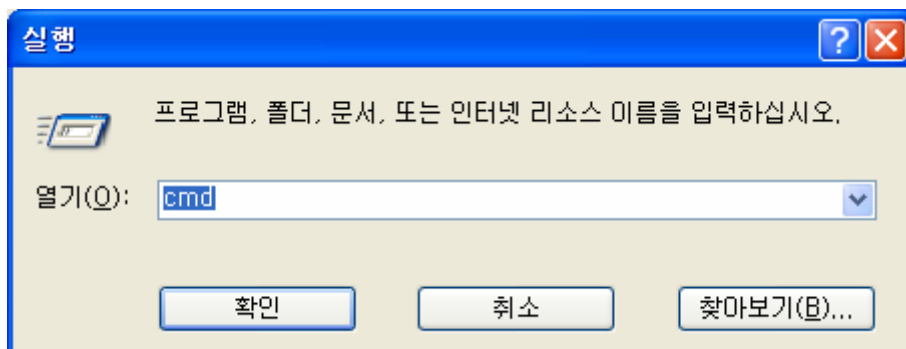
EZL-400S 가 설치된 환경에 맞게 IP 주소관련 항목을 설정해야 합니다. 설정해야 할 IP 주소 항목은 [LOCAL IP ADDRESS], [SUBNET MASK], [GATEWAY IP ADDRESS]입니다.

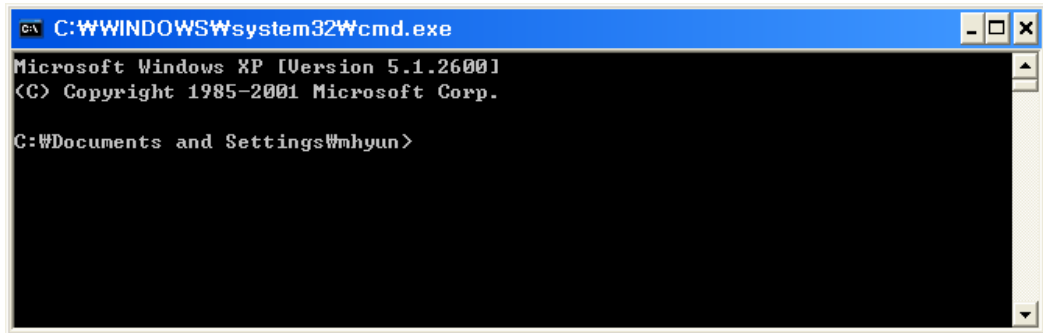
PC와 EZL-400S의 IP 주소가 다음과 같다는 가정하에 설명하겠습니다.

	PC	EZL-400S
Local IP Address	10.1.0.2	10.1.0.1
Subnet Mask	255.0.0.0	255.0.0.0
Gateway IP Address	10.1.0.254	10.1.0.254

1.1.2. telnet으로 로그인

윈도우즈 [시작]메뉴에서 [실행] 누르고 ‘cmd’를 입력하면 도스창이 나타납니다.

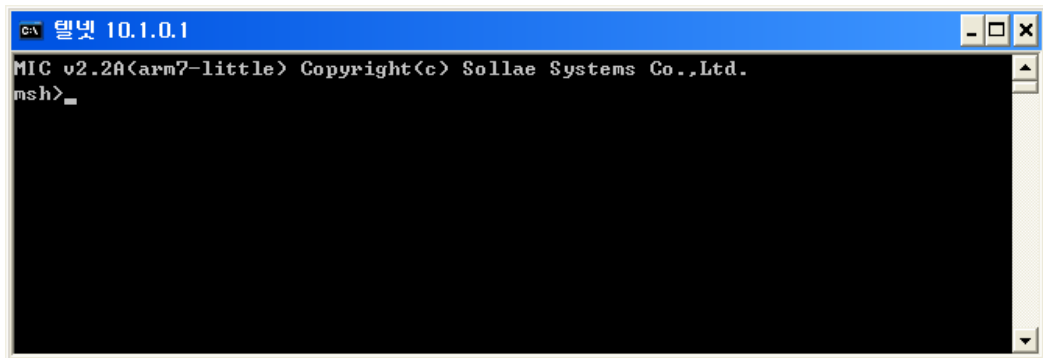




도스창이 나타나면 다음과 같이 입력하여 telnet으로 로그인 합니다.
'telnet [LOCAL IP ADDRESS]'
[LOCAL IP ADDRESS]는 1.1.1에서 설정한 EZL-400S 의 IP 주소입니다.

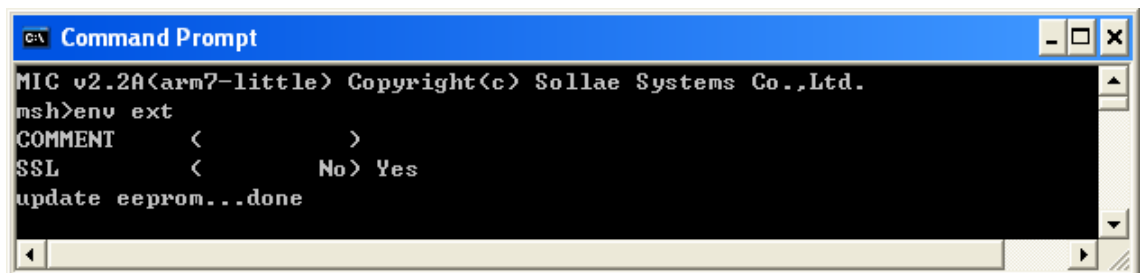
예) telnet 10.1.0.1

아래 그림은 로그인한 그림입니다.



1.1.3. SSL 기능 활성화

'env ext' 명령을 입력한 후 SSL 기능 설정 항목인 'SSL'에 'y'를 입력합니다.



1.1.4. Rebooting

모든 항목들이 지나가면 자동으로 리부팅됩니다.

Console 에서 어떤 명령을 입력할 때 한 항목이라도 변경이 되면 자동으로 리부팅합니다.

1.2. 클라이언트로서 SSL 통신시 주의점

1.2.1. SSL 클라이언트 통신이 가능한 모드

SSL은 TCP 프로토콜 상위에서 동작하는 프로토콜이므로 TCP가 적용되는 통신 모드에서만 통신을 할 수 있습니다.

아래는 SSL 클라이언트 통신을 지원하는 통신모드입니다.

- COD 모드
- ATC 모드에서 'atd' 명령을 사용할 때

2. 서버 모드에서 SSL 사용하기

2.1. SSL 기능 활성화 하기

telnet 클라이언트로 로그인 하여 SSL 기능을 활성화 할 수 있습니다. 다음은 SSL 기능을 활성화 하기 위한 예입니다.

2.1.1. IP 주소 설정

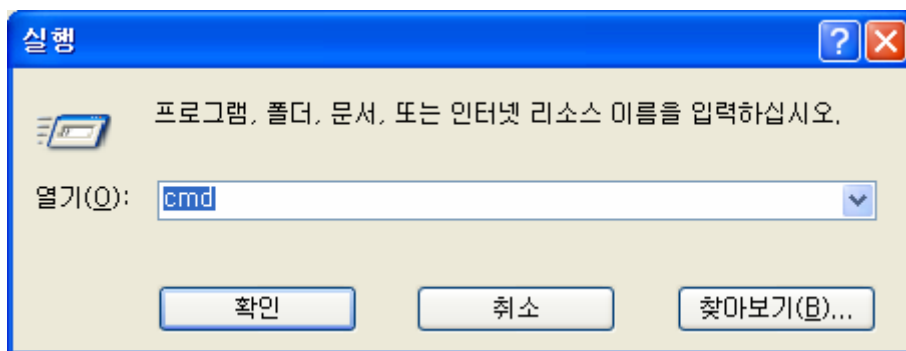
EZL-400S 가 설치된 환경에 맞게 IP 주소관련 항목을 설정해야 합니다. 설정해야 할 IP 주소 항목은 [LOCAL IP ADDRESS], [SUBNET MASK], [GATEWAY IP ADDRESS]입니다.

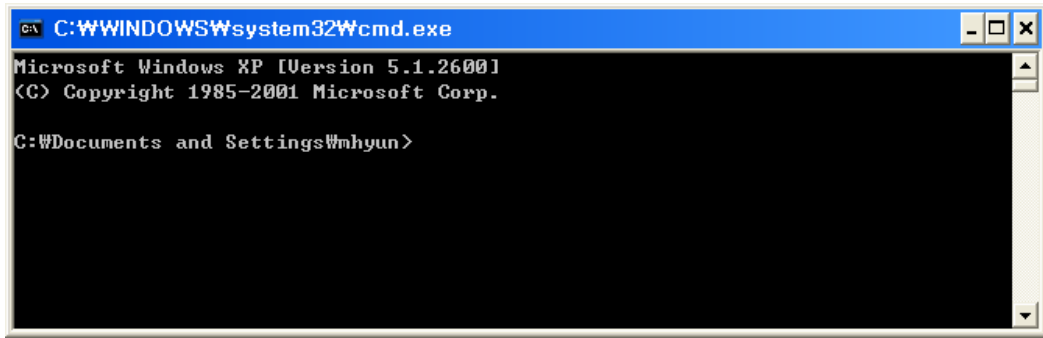
PC와 EZL-400S의 IP 주소가 다음과 같다는 가정하에 설명하겠습니다.

	PC	EZL-400S
Local IP Address	10.1.0.2	10.1.0.1
Subnet Mask	255.0.0.0	255.0.0.0
Gateway IP Address	10.1.0.254	10.1.0.254

2.1.2. telnet으로 로그인

윈도우즈 [시작]메뉴에서 [실행] 누르시고 'cmd'를 입력하면 도스창이 나타납니다.

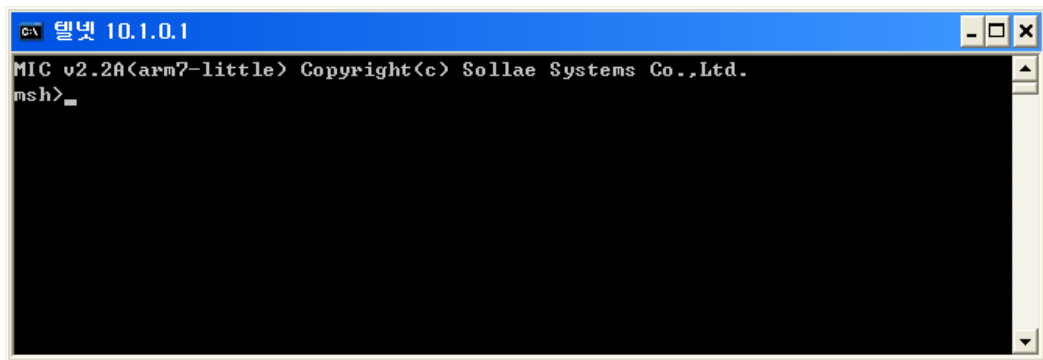




도스창이 나타나면 다음과 같이 입력하여 telnet으로 로그인 합니다.
'telnet [LOCAL IP ADDRESS]'
[LOCAL IP ADDRESS]는 1.1.1에서 설정한 EZL-400S 의 IP 주소입니다.

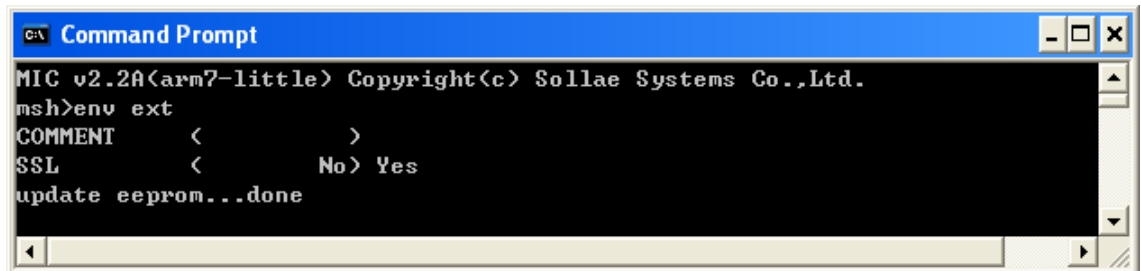
예) telnet 10.1.0.1

아래 그림은 로그인한 그림입니다.



2.1.3. SSL 기능 활성화

'env ext' 명령을 입력한 후 SSL 기능 설정 항목인 'SSL'에 'y'를 입력합니다.



2.1.4. Rebooting

모든 항목들이 지나가면 자동으로 리부팅됩니다.

Console 에서 어떤 명령을 입력할 때 한 항목이라도 변경이 되면 자동으로 리부팅합니다.

2.2. 키 생성

SSL 서버로서 동작하려면 SSL 클라이언트에게 전달할 public key와 SSL 서버 자신이 사용할 private key를 생성해야 합니다.

2.2.1. telnet login

telnet 로그인 상태가 아니라면 2.1 과정으로 telnet 로그인합니다.

2.2.2. 키 생성

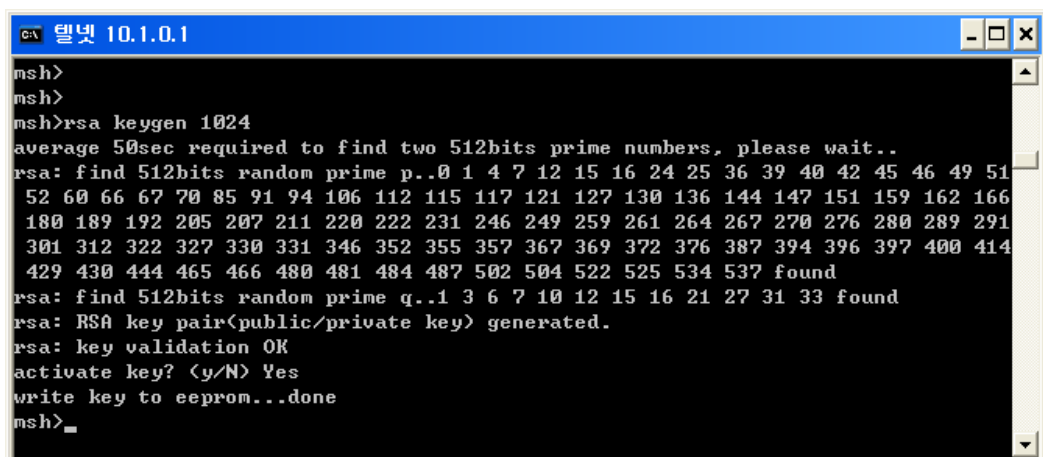
다음의 명령어 형식으로 RSA 키를 생성합니다.

키를 생성할 때 key의 크기에 따라 수분이 걸릴 수도 있습니다.

(명령어형식) `rsa keygen [keylength]`

[keylength]는 사용하는 키 길이에 따라 512, 768, 1024, 또는 2048를 입력하실 수 있습니다.

키가 길면 길수록 시간이 길어집니다. 예를 들어, 키 길이가 2048 비트인 경우에는 평균적으로 5분정도 소요됩니다.

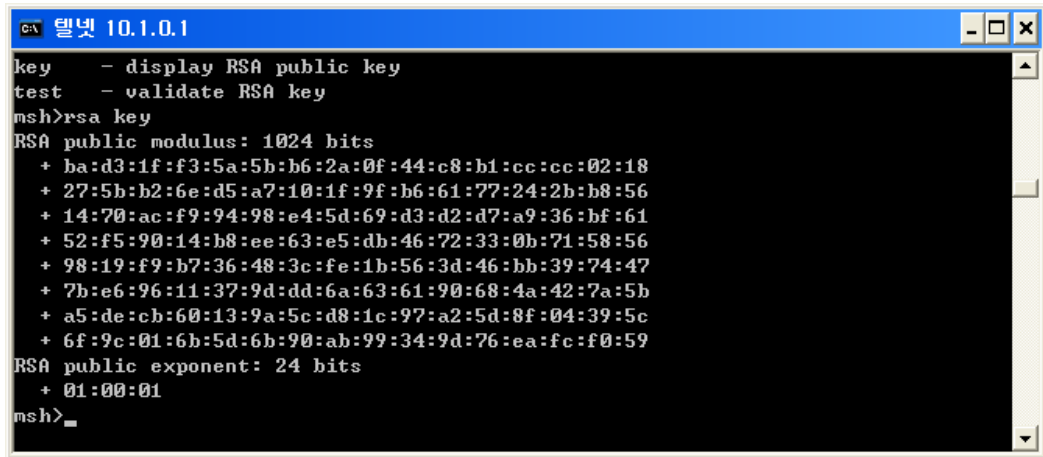
A terminal window titled 'CA 텔넷 10.1.0.1' showing the execution of the 'rsa keygen 1024' command. The output indicates that it takes an average of 50 seconds to find two 512-bit prime numbers. It lists the prime numbers p and q, and confirms that the RSA key pair (public/private key) has been generated and validated. The user is prompted to activate the key and write it to EEPROM, both of which are confirmed with 'Yes' and 'done' respectively.

```
CA 텔넷 10.1.0.1
msh>
msh>
msh>rsa keygen 1024
average 50sec required to find two 512bits prime numbers, please wait..
rsa: find 512bits random prime p..0 1 4 7 12 15 16 24 25 36 39 40 42 45 46 49 51
52 60 66 67 70 85 91 94 106 112 115 117 121 127 130 136 144 147 151 159 162 166
180 189 192 205 207 211 220 222 231 246 249 259 261 264 267 270 276 280 289 291
301 312 322 327 330 331 346 352 355 357 367 369 372 376 387 394 396 397 400 414
429 430 444 465 466 480 481 484 487 502 504 522 525 534 537 found
rsa: find 512bits random prime q..1 3 6 7 10 12 15 16 21 27 31 33 found
rsa: RSA key pair(public/private key) generated.
rsa: key validation OK
activate key? (y/N) Yes
write key to eeprom...done
msh>
```

키를 생성한 후 생성된 키를 비휘발성 메모리에 저장여부를 질의합니다. 이 때 'y'를 눌러 생성된 키를 비휘발성 메모리에 저장합니다.

2.2.3. RSA public key 확인

'rsa key' 명령으로 RSA public key를 확인할 수 있습니다.

A terminal window titled 'c:\ 텔넷 10.1.0.1' with standard window controls. The terminal text is as follows:

```
key - display RSA public key
test - validate RSA key
msh>rsa key
RSA public modulus: 1024 bits
+ ba:d3:1f:f3:5a:5b:b6:2a:0f:44:c8:b1:cc:cc:02:18
+ 27:5b:b2:6e:d5:a7:10:1f:9f:b6:61:77:24:2b:b8:56
+ 14:70:ac:f9:94:98:e4:5d:69:d3:d2:d7:a9:36:bf:61
+ 52:f5:90:14:b8:ee:63:e5:db:46:72:33:0b:71:58:56
+ 98:19:f9:b7:36:48:3c:fe:1b:56:3d:46:bb:39:74:47
+ 7b:e6:96:11:37:9d:dd:6a:63:61:90:68:4a:42:7a:5b
+ a5:de:cb:60:13:9a:5c:d8:1c:97:a2:5d:8f:04:39:5c
+ 6f:9c:01:6b:5d:6b:90:ab:99:34:9d:76:ea:fc:f0:59
RSA public exponent: 24 bits
+ 01:00:01
msh>
```

2.2.4. RSA 키 테스트

생성된 RSA private key와 public key를 테스트합니다. 테스트는 public key로 암호화한 임의의 텍스트를 private key로 암호를 푼 다음 원래의 텍스트와 같은지 확인하고, private key로 암호화한 임의의 텍스트를 public key로 암호를 푼 다음 원래의 텍스트와 같은지 확인합니다.


```
CA> 텔넷 10.1.0.1
msh>rsa test
rsa: key validation OK
public key encryption >> private key decryption
* plain text
45f1331825c821161e5ecb620555ed0341764735b266f74f5d14107dc361874e4768716c21ab0352
c348704368ac7a79ccadb701e9e11b4c9103414486051234926d3e3436a9070f590d85410aed5116
a82ba71460005e6c13d9090103c2da1129f9b531abcbf02b85a0ad323f60802c1b0f2a541c93de12
* encrypted text
8b9dc6fab939c164ff6c097c16b1682ae8358b0d538478a61dabb6257437d7dcc6cdd292c4e100a9
4cef521f65f74e5636dce79be2bbb64af36aa02568f344fcbbf87fd431d139f71534f6f4abb476d3
037e288c51f9d50d1821da17fa8f829e78b3b1fe9545aea7a345aa60785aaf5d2fbeb5e2055fb2
c2cba2d6a6d8ddb1
* decrypted text
45f1331825c821161e5ecb620555ed0341764735b266f74f5d14107dc361874e4768716c21ab0352
c348704368ac7a79ccadb701e9e11b4c9103414486051234926d3e3436a9070f590d85410aed5116
ad82cf00aeb7723a13d9090103c2da1129f9b531abcbf02b85a0ad323f60802c1b0f2a541c93de12
a82ba71460005e6c
verify ok

private key encryption >> public key decryption
* plain text
415b0029c689727766554b7083d1475e78f06947c3695b6d4633cf2cbf58db33e4145f3f097c3f70
2705562db0c21641f25d5b3cb808977137c8287584cb9970eeb19e0090d5ad368fb8eb069c346e01
3e8d2071a291f5079ff648136786d6224d5de6332497f645a7e6564f686c1008412ad5584f12fe63
c96c6e748285d501
* encrypted text
3aa4646753e7242cc716ecc4fd61e787382e397e3e676638dd667b41fcb23eeb323ae4832182138d
b25f5f292bce828a0e919ec1fac8b8490f7fe4f53d9773ab1af982a7ad64def8641faf7c1bb729ad
5405525ce43952dabdbc640ff3ea5fe8bc438f34510fedbcf79e708e43cb9c186301714872fa7981
2308e179b7d7c5b3
* decrypted text
415b0029c689727766554b7083d1475e78f06947c3695b6d4633cf2cbf58db33e4145f3f097c3f70
2705562db0c21641f25d5b3cb808977137c8287584cb9970eeb19e0090d5ad368fb8eb069c346e01
3e8d2071a291f5079ff648136786d6224d5de6332497f645a7e6564f686c1008412ad5584f12fe63
c96c6e748285d501
verify ok
msh>
```

2.3. 인증서 생성

SSL 서버는 인증서를 가지고 있어야 합니다. 공인 인증서를 받지 않고 자체적인 인증서를 생성해서 사용할 수 있습니다.

2.3.1. 인증서 생성

‘cert new’ 명령으로 인증서를 생성할 수 있습니다. ‘cert new’ 명령에 EZL-400S 는 인증서를 생성하고, 인증서에 자체적으로 싸인을 합니다. 인증서 생성이 끝나면 비휘발성 메모리에 저장 여부를 묻습니다. ‘y’라고 입력하면 자체 생성된 인증서가 비휘발성 메모리에 저장됩니다.

