使用 OpenSSL 制作 ECDH 密钥交换证书

2015-12-23 | 标签: centos, openssl, ecdh, rsa

前言

对于 ECDH，Wikipedia 如下描述:

Elliptic curve Diffie–Hellman (ECDH) is an anonymous key agreement protocol that allows two parties, each having an elliptic curve public–private key pair, to establish a shared secret over an insecure channel.

参见https://en.wikipedia.org/wiki/Elliptic_curve_Diffie-Hellman

ECDH 是基于 ECC（Elliptic Curve Cryptosystems，椭圆曲线密码体制）的 DH（ Diffie-Hellman）密钥交换算法。交换双方可以在不共享任何秘密的情况下协商出一个密钥。与 Diffie-Hellman 相比 ECDH 具有 ECC 的高强度、短密钥长度、计算速度快等优点。

由于 ECDH 每次用一个固定的 DH key, 导致不能向前保密（forward secrecy），安全性会降低，所以一般都是用 ECDHE（ECDH 的 ephemeral version）或其他版本的 ECDH 算法。

本文只对 ECDH 进行介绍，只为测试。

环境说明

CentOS 7.2(CentOS_7_x86_64_1151)

OpenSSL 1.0.2e

安装 OpenSSL

1.查看本机安装的版本

```
# openssl version
OpenSSL 1.0.1e-fips 11 Feb 2013
```

2.OpenSSL 应在1.0.2以上，这里使用 1.0.2e，去官网下载，源码安装一下，请根据实际情况更改下载地址。

```
# cd /usr/src
# wget https://www.openssl.org/source/openssl-1.0.2e.tar.gz
# tar -zxf openssl-1.0.2e.tar.gz
```

3.编译安装 OpenSSL

```
# cd openssl-1.0.2e
# ./config
# make
# make test
# make install
```

4.如果旧版本还在，可以先备份，并修改一个

```
# mv /usr/bin/openssl /root/
# ln -s /usr/local/ssl/bin/openssl /usr/bin/openssl
```

5.查看版本

```
# openssl version
OpenSSL 1.0.2e 3 Dec 2015
```

制作CA证书

ECDH 密钥交换算法，不能自签名，所以制作证书，需要一个 CA 进行颁发。

CA 要给别人颁发证书，首先自己得有一个作为根证书，我们得在一切工作之前修改好 CA 的配置文件、序列号、索引等等。这些参数都是在 openssl.cnf 里面配置的。

```
# vi /etc/pki/tls/openssl.cnf
```

openssl.cnf 配置文件中主要关注 `[CA_default]` 和 `[policy_match]` 规则

```
......
[ CA_default ]

dir             = /etc/pki/CA           # Where everything is kept
certs           = $dir/certs            # Where the issued certs are kept
crl_dir         = $dir/crl              # Where the issued crl are kept
database        = $dir/index.txt        # database index file.
#unique_subject = no                    # Set to 'no' to allow creation of
                                        # several ctificates with same subject.
new_certs_dir   = $dir/newcerts         # default place for new certs.

certificate     = $dir/cacert.pem       # The CA certificate
serial          = $dir/serial           # The current serial number
crlnumber       = $dir/crlnumber        # the current crl number
                                        # must be commented out to leave a V1 CRL
crl             = $dir/crl.pem          # The current CRL
private_key     = $dir/private/cakey.pem# The private key
RANDFILE        = $dir/private/.rand    # private random number file
......
......
# For the CA policy
```

```
[ policy_match ]
countryName             = match
stateOrProvinceName     = optional
organizationName        = optional
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional
......
```

1.先初始化 index.txt 和 serial 文件

```
# cd /etc/pki/CA/
# touch index.txt serial
# echo 01 > serial
```

2.生成 CA 的私钥（private key）

```
# cd /etc/pki/CA/
# openssl genrsa -out private/cakey.pem 2048
```

3.生成 CA 的证书(certificate)，使用 req 命令生成自签证书

```
# openssl req -new -x509 -key private/cakey.pem -out cacert.pem
```

会提示输入一些内容，请按提示输入即可。

制作 ECDH 密钥交换的证书

将根证书拷贝到 `$HOME` 目录，省去输入目录的麻烦，本文只为示例作用。

```
# cd /etc/pki/CA/private/
# cp cakey.pem ~
# cd /etc/pki/CA/
# cp cacert.pem ~
# cd ~
```

1.生成 private key 之前，先查看一下那种椭圆曲线可以使用

```
# openssl ecparam -list_curves
```

结果如下，OpenSSL 1.0.2e 支持很多。

```
......
secp521r1 : NIST/SECG curve over a 521 bit prime field
prime192v1: NIST/X9.62/SECG curve over a 192 bit prime field
prime192v2: X9.62 curve over a 192 bit prime field
prime192v3: X9.62 curve over a 192 bit prime field
prime239v1: X9.62 curve over a 239 bit prime field
prime239v2: X9.62 curve over a 239 bit prime field
prime239v3: X9.62 curve over a 239 bit prime field
prime256v1: X9.62/SECG curve over a 256 bit prime field
...... 本例使用 prime256v1
```

2.生成 ECDH 的私钥（private key）

```
# openssl ecparam -out ecparam.pem -name prime256v1
# openssl genpkey -paramfile ecparam.pem -out ecdhkey.pem
```

3.生成 ECDH 的公钥（public key）

```
# openssl pkey -in ecdhkey.pem -pubout -out ecdhpubkey.pem
```

4.生成 CSR（Certificate Request）文件，CSR 是需要自签名的，不能使用 ECDH 算法，因为 ECDH 不是签名算法，本例使用RSA算法生成。

```
# openssl genrsa -out rsakey.pem 1024
# openssl req -new -key rsakey.pem -out ecdhrsacsr.pem
```

5.最后，使用 ECDH 的公钥和 RSA 的 CSR 制作 ECDH 证书，由于 ECDH 不是自签名算法，不能自签名生成。本例使用刚才制作的 CA 证书生成。

```
#openssl x509 -req -in ecdhrsacsr.pem -CAkey cakey.pem -CA cacert.pem -force_pubkey ecdhpubkey.pem -out
ecdhcert.pem -CAcreateserial
```

本例后来使用的 RSA 算法生成的 CSR 文件，所以生成的 `ecdhcert.pem` 支持 ECDH_RSA 的密码套件。

目前生成的证书列表如下：

```
cakey.pem  # CA private key(RSA算法的)
cacert.pem  # CA certificate
ecparam.pem  # EC Parameters
ecdhkey.pem  # ECDH private key
ecdhpubkey.pem # ECDH public key
rsakey.pem  # RSA private key(用于请求证书的)
ecdhrsacsr.pem # RSA 的 CSR文件
ecdhcert.pem # ECDH certificate(RSA算法的)
```

验证 ECDH

使用 OpenSSL 测试

1.服务端

```
# openssl s_server -cert ecdhcert.pem -key ecdhkey.pem -port 8888
```

2.客户端 (需要打开一个新的Terminal进行)

```
# cd ~
# vi test_ciphers
```

输入如下内容

```
#!/usr/bin/env bash

# OpenSSL requires the port number.
SERVER=127.0.0.1:8888
DELAY=1
ciphers=$(openssl ciphers 'ECDH:eNULL' | sed -e 's/:/ /g')

echo Obtaining cipher list from $(openssl version).

for cipher in ${ciphers[@]}
do
echo -n Testing $cipher...
result=$(echo -n | openssl s_client -cipher "$cipher" -connect $SERVER 2>&1)
if [[ "$result" =~ ":error:" ]] ; then
  error=$(echo -n $result | cut -d':' -f6)
  echo NO \($error\)
else
  if [[ "$result" =~ "Cipher is ${cipher}" || "$result" =~ "Cipher    :" ]] ; then
    echo YES
  else
    echo UNKNOWN RESPONSE
    echo $result
  fi
fi
sleep $DELAY
done
```

此脚本会验证包含 ECDH 密钥交换算法的密码套件的支持程度，可以修改

```
ciphers=$(openssl ciphers 'ECDH:eNULL' | sed -e 's/:/ /g')
```

测试其他算法的支持。 具体可以参考 https://www.openssl.org/docs/manmaster/apps/ciphers.html

保存文件后，更改文件为可执行

```
# chmod +x test_ciphers
```

执行测试

```
# ./test_ciphers
```

结果如下

```
Obtaining cipher list from OpenSSL 1.0.2e 3 Dec 2015.
Testing ECDHE-RSA-AES256-GCM-SHA384...NO (sslv3 alert handshake failure)
Testing ECDHE-ECDSA-AES256-GCM-SHA384...YES
Testing ECDHE-RSA-AES256-SHA384...NO (sslv3 alert handshake failure)
Testing ECDHE-ECDSA-AES256-SHA384...YES
Testing ECDHE-RSA-AES256-SHA...NO (sslv3 alert handshake failure)
Testing ECDHE-ECDSA-AES256-SHA...YES
Testing AECDH-AES256-SHA...NO (sslv3 alert handshake failure)
Testing ECDH-RSA-AES256-GCM-SHA384...YES
Testing ECDH-ECDSA-AES256-GCM-SHA384...NO (sslv3 alert handshake failure)
Testing ECDH-RSA-AES256-SHA384...YES
Testing ECDH-ECDSA-AES256-SHA384...NO (sslv3 alert handshake failure)
Testing ECDH-RSA-AES256-SHA...YES
......
```

可以发现，包含 ECDH-RSA 的密码套件的，都是通过的。

参考资料

HOW TO INSTALL AND UPDATE OPENSSL ON CENTOS 6 / CENTOS 7

基于 OpenSSL 自建CA和颁发 SSL 证书

OpenSSL generate different types of self signed certificat

How do I list the SSL/TLS cipher suites a particular website offers?