



```
ptions are
-c          to output the digest with separating colons
-r          to output the digest in coreutils format
-d          to output debug info
-hex        output as hex dump
-binary     output in binary form
-hmac arg   set the HMAC key to arg
-non-fips-allow allow use of non FIPS digest
-sign file  sign digest using private key in file
-verify file verify a signature using public key in file
-prverify file verify a signature using private key in file
-keyform arg key file format (PEM or ENGINE)
-out filename output to filename rather than stdout
-signature file signature to verify
-sigopt nm:v signature parameter
-hmac key   create hashed MAC with key
-mac algorithm create MAC (not neccessarily HMAC)
-macopt nm:v MAC algorithm parameters or key
-engine e   use engine e, possibly a hardware device.
-md4        to use the md4 message digest algorithm
-md5        to use the md5 message digest algorithm
-ripemd160  to use the ripemd160 message digest algorithm
-sha        to use the sha message digest algorithm
-sha1       to use the sha1 message digest algorithm
-sha224     to use the sha224 message digest algorithm
-sha256     to use the sha256 message digest algorithm
-sha384     to use the sha384 message digest algorithm
-sha512     to use the sha512 message digest algorithm
-whirlpool  to use the whirlpool message digest algorithm
```

```
//输出的摘要信息以分号隔离, 和-hex同时使用
//指定输出的格式
//输出BIO调试信息
//以16进制打印输出结果
//输出二进制结果
//指定hmac的key
//允许使用不符合fips标准的摘要算法
//执行签名操作, 后面指定私钥文件
//执行验证操作, 后面指定公钥文件, 与prverify不能同时使用
//执行验证操作, 后面指定密钥文件, 与verify不能同时使用
//指定密钥文件格式, pem或者engine
//指定输出文件, 默认标准输出
//指定签名文件, 在验证签名时使用
//签名参数
//制作一个hmac 使用key
//制作一个mac
//mac算法参数或者key
//使用硬件或者三方加密库
//摘要算法使用md4
//摘要算法使用md5
//摘要算法使用ripemd160
//摘要算法使用sha
//摘要算法使用sha1
//摘要算法使用sha223
//摘要算法使用sha256
//摘要算法使用sha384
//摘要算法使用sha512
//摘要算法使用whirlpool
```

### 3、dgst使用示例

#### 1、仅做摘要运算而不做签名操作

```
/*对file.txt文件使用sha1算法进行hash运算*/
xlzh@cmos:~/test$ openssl dgst -sha1 file.txt
SHA1(file.txt)= c994aec2a9007221a9b9113b8ab60a60144740c9
/*指定-non-fips-allow参数, 与fips标准有关, 尚待研究*/
xlzh@cmos:~/test$ openssl dgst -sha1 -non-fips-allow file.txt
SHA1(file.txt)= c994aec2a9007221a9b9113b8ab60a60144740c9
/*指定-d参数, 打印调试消息*/
xlzh@cmos:~/test$ openssl dgst -sha1 -d file.txt
BIO[02469910]:ctrl(6) - FILE pointer
BIO[02469910]:ctrl return 0
BIO[02469910]:ctrl(108) - FILE pointer
BIO[02469910]:ctrl return 1
BIO[02469910]:read(0,8192) - FILE pointer
BIO[02469910]:read return 37
BIO[02469910]:read(0,8192) - FILE pointer
BIO[02469910]:read return 0
SHA1(file.txt)= c994aec2a9007221a9b9113b8ab60a60144740c9
BIO[02469910]:ctrl(1) - FILE pointer
BIO[02469910]:ctrl return 0
BIO[02469910]:Free - FILE pointer
/*指定-c -hex参数, 以16进制打印结果*/
xlzh@cmos:~/test$ openssl dgst -sha1 -c -hex file.txt
SHA1(file.txt)= c9:94:aec2:a9:00:72:21:a9:b9:11:3b:8a:b6:0a:60:14:47:40:c9
/*指定-r参数, 输出结果如下所示, 然并卵.....*/
xlzh@cmos:~/test$ openssl dgst -sha1 -r file.txt
c994aec2a9007221a9b9113b8ab60a60144740c9 *file.txt
/*指定-binary参数, 输入结果为二进制*/
xlzh@cmos:~/test$ openssl dgst -sha1 -binary file.txt
o@r!;
`G@xlzh@cmos:~/test$
```

libevent

#### 阅读排行榜

1. openssl 摘要和签名验证指令dgst使用详解(2435)
2. windows openssl-1.1.1 编译静态库和动态库(1381)
3. ECC算法软件保护中的应用(636)
4. JAVA同步锁机制 wait() notify() notifyAll()(468)
5. MFC窗口风格 WS\_style/WS\_EX\_style(263)

#### 评论排行榜

1. windows openssl-1.1.1 编译静态库和动态库(4)
2. PhpStorm激活方式(1)

#### 最新评论

1. Re:windows openssl-1.1.1 编译静态库和动态库  
dmake下载链接  
--哈哈稻草人
2. Re:windows openssl-1.1.1 编译静态库和动态库  
4、安装dmake , ppm install dmake  
很可能会失败, 可以手动安装dmake  
解压dmake, 将解压后大目录放在Perl64\site\lib\auto中, auto目录不存在就自己建立...  
--哈哈稻草人
3. Re:windows openssl-1.1.1 编译静态库和动态库  
如果不使用汇编就这样: perl configure no-asm VC-WIN32 --prefix="e:\mylib" perl configure no-asm no-shared VC-WIN32...  
--哈哈稻草人
4. Re:windows openssl-1.1.1 编译静态库和动态库  
编译debug库 perl configure VC-WIN32 --debug --prefix="e:\mylib"  
--哈哈稻草人
5. Re:PhpStorm激活方式  
以上方式同时可以激活 webstorm  
2018.2版本, 但是有闪退的现象解决办法: 链接: 密码: rblr替换新的破解包...  
--大地之光

## 2、使用RSA密钥进行签名验证操作

```
/*摘要算法选取sha256, 密钥RSA密钥, 对file.txt进行签名*/
xlzh@cmos:~/test$ openssl dgst -sign RSA.pem -sha256 -out sign.txt file.txt
/*使用RSA密钥验证签名 (prverify参数), 验证成功*/
xlzh@cmos:~/test$ openssl dgst -prverify RSA.pem -sha256 -signature sign.txt file.txt
Verified OKt
/*从密钥中提取公钥*/
xlzh@cmos:~/test$ openssl rsa -in RSA.pem -out pub.pem -pubout
writing RSA key
/*使用RSA公钥验证签名 (verify参数), 验证成功*/
xlzh@cmos:~/test$ openssl dgst -verify pub.pem -sha256 -signature sign.txt file.txt
Verified OK
```

## 3、使用DSA密钥进行签名验证操作

```
/*使用DSA算法, 摘要算法sha256, 对file.txt进行签名*/
xlzh@cmos:~/test$ openssl dgst -sign DSA.pem -sha256 -out sign.txt file.txt
/*使用DSA密钥验证签名*/
xlzh@cmos:~/test$ openssl dgst -prverify DSA.pem -sha256 -signature sign.txt file.txt
Verified OK
/*使用DSA算法, 摘要算法dss1, 对file.txt进行签名*/
xlzh@cmos:~/test$ openssl dgst -sign DSA.pem -dss1 -out sign1.txt file.txt
/*使用DSA密钥验证签名*/
xlzh@cmos:~/test$ openssl dgst -prverify DSA.pem -dss1 -signature sign1.txt file.txt
Verified OK
/*提取公钥*/
xlzh@cmos:~/test$ openssl dsa -in DSA.pem -out pub.pem -pubout
read DSA key
writing DSA key
/*使用DSA公钥验证签名*/
xlzh@cmos:~/test$ openssl dgst -verify pub.pem -dss1 -signature sign1.txt file.txt
Verified OK
/*使用DSA公钥验证签名*/
xlzh@cmos:~/test$ openssl dgst -verify pub.pem -sha256 -signature sign.txt file.txt
Verified OK
xlzh@cmos:~/test$
```

根据dgst man手册的定义, 如果使用DSA算法进行签名验证, 必须使用dss1摘要算法, 但是本实验证明使用其他摘要算法也可以签名验证。此处不明白, 希望大牛指点.....

## 4、HMAC的使用

MAC 消息认证码, 构造方法可以基于hash, 也可以基于对称加密算法, HMAC是基于hash的消息认证码。数据和密钥作为输入, 摘要信息作为输出, 常用于认证。

```
xlzh@cmos:~/test$ openssl dgst -sha256 -hmac 123456 file.txt
HMAC-SHA256(file.txt)= b8e92990b9fc2ac9b58fde06f4738dceb4fb1fc47b4d2234a9c3f152907b333a
```

例如用户登录服务器

- 1、服务器给客户端发送一个随机数
- 2、客户端使用随机数作为密钥和用户密码做HMAC, 结果发送给服务器
- 3、服务器去除存储的用户密码, 也是用随机数与用户密码做HMAC, 根据HMAC结果是否一样确认用户身份。

## 4、遗留问题

dgst中sigopt、mac、macopt参数的含义即使用方法, 因为doc都没给出具体例子, 待研究openssl源码后进行补充

为什么使用DSA签名的时候可以选择其他hash算法(man 手册说只能使用dss1)

还有dgst的hmac和hmac参数, 没错, 你没看错, 它的确提供了两个完全一样的参数, 给出的解释还不一样, 还是研究源码去吧。

可恶的openssl.....

转自:https://www.cnblogs.com/gordon0918/p/5382541.html

好文要顶

关注我

收藏该文

哈哈稻草人

关注 - 1

粉丝 - 0

+加关注

« 上一篇：[PhpStorm激活方式](#)

» 下一篇：[windows openssl-1.1.1 编译静态库和动态库](#)

posted on 2018-09-14 14:49 [哈哈稻草人](#) (阅读(2435) 评论(0) 编辑 收藏 举报)

登录后才能查看或发表评论，立即 [登录](#) 或者 [逛逛](#) [博客园首页](#)

穿山甲

App开发者高效成长

 增长变现闭环

 收入提升 **28%**



立即注册

编辑推荐：

- [聊聊我在微软外服的工作经历及一些个人见解](#)
- [死磕 NIO — Reactor 模式就一定意味着高性能吗？](#)
- [消息队列那么多，为什么建议深入了解下RabbitMQ？](#)
- [技术管理进阶——管人还是管事？](#)
- [以终为始：如何让你的开发符合预期](#)

最新新闻：

- [LSTM—败涂地！男生发表4页最高谱论文，用时序模型预测女友情绪 \( 2021-10-25 17:09 \)](#)
- [AI杀手终成现实？美国陆军「杀人机器狗」引发恐慌 \( 2021-10-25 17:02 \)](#)
- [AI学会灌水和造假！Google新研究揭露了AI现实应用的陷阱 \( 2021-10-25 16:50 \)](#)
- [可口可乐宣布推出由100%植物性塑料制成的瓶子 \( 2021-10-25 16:40 \)](#)
- [抢先Win11！华为移动应用引擎第二批众测开启：在PC上玩安卓App \( 2021-10-25 16:30 \)](#)

» [更多新闻...](#)

0

推荐

0

反对

[刷新评论](#) [刷新页面](#) [返回顶部](#)