

openssl 摘要和签名验证指令dgst使用详解

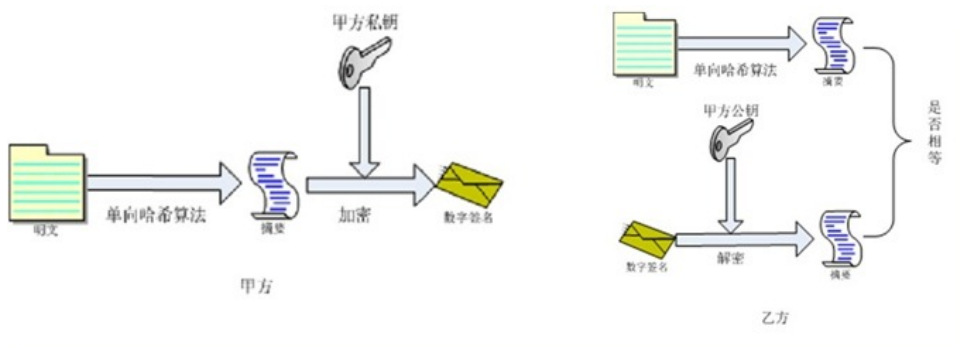
1、信息摘要和数字签名概述

信息摘要：对数据进行处理，得到一段固定长度的结果，其特点输入：

- 1、输出长度固定。即输出长度和输入长度无关。
- 2、不可逆。即由输出数据理论上不能推导出输入数据
- 4、对输入数据敏感。当输入数据变化极小时，输出数据也会发生明显的变化
- 5、防碰撞。即不同的数据数据得到相同输出数据的可能性极低。

由于信息摘要有上述特点，一般保证数据的完整性，对一个大文件进行摘要运算，得到其摘要值。通过网络或者其他渠道传输后，通过验证其摘要值，确定大文件本身有没有发生变化。

数字签名：数字签名其实分成两步，首先对原始文件进行摘要运算，得到摘要值，然后使用公开密钥算法中的私钥对摘要值进行加密。其签名和验证过程如下图所示



有数字签名的过程可以知道，对发送信息进行数字签名，可以保证数字签名的完整性、真实性、不可抵赖性。即接收者可以确认消息的来源、消息的真实，发送者不可以抵赖自己发送的消息，与现实生活中签名的作用大致相同。

2、摘要算法和数字签名相关指令及用法

目前openssl提供的摘要算法有md4、md5、ripemd160、sha、sha1、sha224、sha256、sha512、sha384、wirlpool。可以通过openssl dgst -命令查看。

上面我们已经提到了，数字签名分为摘要和加密两部分。在openssl提供的指令中，并没有区分两者。而是在摘要算法指令中包含了签名和校验参数。例如我们适用openssl md5 -命令可以看到它提供的选项有签名和验证等参数。

在openssl中单独使用摘要算法指令完成摘要或者签名操作，也可以通过dgst完成相同的操作。在签名的时候多数使用RSA私钥或者DSA私钥，当使用RSA私钥的时候，我们可以使用单独的摘要算法指令指定摘要算法进行签名，但当使用DSA使用签名的时候，就必须使用dgst指令，因为使用DSA签名的时候必须使用DSA自身的摘要算法，而openssl没有为它提供相应的指令。

```
/*有明文文件file.txt和RSA密钥RSA.pem*/
xlzh@cmos:~/test$ ls
file.txt  RSA.pem
/*使用md5指令指定sha1算法，对file.txt进行签名，生成签名文件sign1.txt*/
xlzh@cmos:~/test$ openssl md5 -sha1 -sign RSA.pem -out sign1.txt file.txt
/*使用md5指令指定sha1算法，对file.txt进行签名，生成签名文件sign1.txt*/
xlzh@cmos:~/test$ openssl dgst -sha1 -sign RSA.pem -out sign2.txt file.txt
/*两个签名文件一样，说明两个指令完成相同的功能*/
xlzh@cmos:~/test$ diff sign1.txt sign2.txt
```

可以看到md5和dgst完成相同的功能。不过让人纠结的是使用md5进行签名的时候可以指定其他摘要算法，笔者觉得太别扭了。所以建议做摘要和签名验证时使用dgst指令，忘记其他.....

访问人数：

UV

访问总量：

PV

昵称： Gordon0918

园龄： 6年8个月

粉丝： 21

关注： 1

+加关注

< 2020年4月 >						
日	一	二	三	四	五	六
29	30	31	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	1	2
3	4	5	6	7	8	9

搜索

常用链接

[我的随笔](#)
[我的评论](#)
[我的参与](#)
[最新评论](#)
[我的标签](#)


我的标签

[android\(6\)](#)
[逆向\(4\)](#)
[Hook\(2\)](#)
[ida\(2\)](#)
[genrsa\(2\)](#)
[RSA\(2\)](#)
[smali\(2\)](#)
[so\(1\)](#)
[sphinx\(1\)](#)
[substrate\(1\)](#)
[更多](#)


随笔分类

[android\(6\)](#)
[android安全\(11\)](#)
[C/C++\(2\)](#)
[git\(1\)](#)
[Linux\(3\)](#)
[openssl\(8\)](#)
[Scrapy\(2\)](#)
[Windows\(1\)](#)
[渗透测试](#)
[协议\(2\)](#)

dgst指令用法介绍如下



```
xlzh@cmos:~/test$ openssl dgst -
unknown option '-'
options are
-c                to output the digest with separating colons      //输出的摘要信息以分号隔
离, 和-hex同时使用
-r                to output the digest in coreutils format          //指定输出的格式
-d                to output debug info                             //输出BIO调试信息
-hex              output as hex dump                               //以16进制打印输出结果
-binary           output in binary form                            //输出二进制结果
-hmac arg         set the HMAC key to arg                          //指定hmac的key
-non-fips-allow   allow use of non FIPS digest                    //允许使用不符合fips标准的
摘要算法
-sign file        sign digest using private key in file            //执行签名操作, 后面指定私
钥文件
-verify file       verify a signature using public key in file      //执行验证操作, 后面指定公
钥文件, 与prverify不能同时使用
-prverify file     verify a signature using private key in file     //执行验证操作, 后面指定密
钥文件, 与verify不能同时使用
-keyform arg       key file format (PEM or ENGINE)                 //指定密钥文件格式, pem或
者engine
-out filename      output to filename rather than stdout           //指定输出文件, 默认标准输
出
-signature file    signature to verify                             //指定签名文件, 在验证签名
时使用
-sigopt nm:v       signature parameter                             //签名参数
-hmac key           create hashed MAC with key                     //制作一个hmac 使用key
-mac algorithm      create MAC (not necessarily HMAC)              //制作一个mac
-macopt nm:v        MAC algorithm parameters or key                //mac算法参数或者key
-engine e           use engine e, possibly a hardware device.      //使用硬件或者三方加密库
-md4                to use the md4 message digest algorithm        //摘要算法使用md4
-md5                to use the md5 message digest algorithm        //摘要算法使用md5
-ripemd160          to use the ripemd160 message digest algorithm   //摘要算法使用ripemd160
-sha                to use the sha message digest algorithm        //摘要算法使用sha
-sha1               to use the sha1 message digest algorithm       //摘要算法使用sha1
-sha224             to use the sha224 message digest algorithm      //摘要算法使用sha223
-sha256             to use the sha256 message digest algorithm      //摘要算法使用sha256
-sha384             to use the sha384 message digest algorithm      //摘要算法使用sha384
-sha512             to use the sha512 message digest algorithm      //摘要算法使用sha512
-whirlpool          to use the whirlpool message digest algorithm   //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:ae:c2: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*/
```

随笔档案

- 2017年12月(1)
- 2017年4月(6)
- 2017年3月(4)
- 2016年6月(4)
- 2016年5月(1)
- 2016年4月(5)
- 2016年3月(6)
- 2016年1月(5)
- 2015年7月(1)
- 2015年1月(3)
- 2014年7月(1)

最新评论

- 1. Re:openssl 对称加密算法enc命令详解
fedora 29 x86 workstation OpenSSL
1.1.1d FIPS 10 Sep 2019 没有 aes-25
6-gcm. openssl enc -ciphers 何解...
--NickD
- 2. Re:openssl 对称加密算法enc命令详解
-pass env:passwd 的passwd的前面不需
要加\$?
--crazyloser
- 3. Re:Android AccessibilityService(辅
助服务) 使用示例
他是返回的整个activity 的view , 所以会
包含三个Fragment 的
--伍歌歌
- 4. Re:PPTP协议握手流程分析
大佬 自己能软件模拟vpn 并建立通道 进
行数据传输吗
--54辉哥
- 5. Re:Https协议简析及中间人攻击原理
写的不错
--aqu415

阅读排行榜

- 1. openssl 对称加密算法enc命令详解(25
870)
- 2. openssl 证书请求和自签名命令req详解
(23148)
- 3. 如何把java代码转换成smali代码(2090
6)
- 4. openssl 非对称加密算法RSA命令详解
(18578)
- 5. openssl 摘要和签名验证指令dgst使用
详解(18019)

评论排行榜

- 1. 如何把java代码转换成smali代码(4)
- 2. android调试系列--使用ida pro调试原
生程序(3)
- 3. openssl 对称加密算法enc命令详解(2)
- 4. openssl 非对称加密算法RSA命令详解
(1)
- 5. openssl 非对称加密算法DSA命令详解
(1)

推荐排行榜

- 1. openssl 证书请求和自签名命令req详解
(5)
- 2. Android调试系列一使用android studi
o调试smali代码(3)
- 3. openssl 非对称加密算法RSA命令详解
(2)
- 4. openssl CA服务器模拟指令CA详解(1)
- 5. Https协议简析及中间人攻击原理(1)

```
xlzh@cmos:~/test$ openssl dgst -sha1 -binary file.txt
0000000000000000000000000000000000000000000000000000000000000000
`G@xlzh@cmos:~/test$
```

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.....

分类: [openssl](#)

标签: [dgst](#), [hmac](#), [sha1](#), [sha256](#), [dss1](#), [签名](#), [验证](#), [摘要](#)

好文要顶

关注我

收藏该文



[Gordon0918](#)

[关注 - 1](#)

[粉丝 - 21](#)

[+加关注](#)

0

0

« 上一篇: [openssl 非对称加密算法DSA命令详解](#)

» 下一篇: [openssl 证书请求和自签名命令req详解](#)

posted @ 2016-04-12 18:23 [Gordon0918](#) 阅读(18019) 评论(0) [编辑](#) [收藏](#)

[刷新评论](#) [刷新页面](#) [返回顶部](#)

注册用户登录后才能发表评论, 请 [登录](#) 或 [注册](#), [访问](#) 网站首页。

【推荐】超50万行VC++源码: 大型组态工控、电力仿真CAD与GIS源码库

【推荐】腾讯云产品限时秒杀, 爆款1核2G云服务器99元/年!

相关博文:

- [数字摘要、数字签名和加密算法](#)
- [C++ openssl ECDSA签名](#)
- [openssl 非对称加密算法RSA命令详解](#)
- [加解密算法、消息摘要、消息认证技术、数字签名与公钥证书](#)
- [Jar 包签名](#)
- » [更多推荐...](#)

最新 IT 新闻:

- [SpaceX公布Starship手册](#) 介绍它如何取代航天飞机并提供舒适旅途
- [冠状病毒爆发期间](#) 研究人员在Zoom当中发现更多安全问题
- [SpaceX载人龙飞船首次正式运营](#) 增加了NASA和日本JAXA宇航员
- [Mojang新作《我的世界: 地下城》5月28日发行](#)
- [开放源代码的项目Frontline Foods问世](#) 向医院工作人员提供餐食
- » [更多新闻...](#)