Netkiller Cryptography 手札

目录

范例清单

Netkiller Cryptography 手札

信息安全与加密

Mr. Neo Chan, 陈景峰(BG7NYT)

中国广东省深圳市宝安区龙华镇溪山美地
518109
+86 13113668890
+86 755 29812080
<netkiller@msn.com>

电子书最近一次更新于 2021-10-21 20:08:59

http://www.netkiller.cn
http://netkiller.github.io
http://netkiller.sourceforge.net
微信订阅号 netkiller-ebook
微信：13113668890 请注明“读者”
QQ：13721218 请注明“读者”
QQ群：128659835 请注明“读者”
知乎专栏 | 多维度架构

$Date: 2013-08-09 18:01:31 +0800 (Fri, 09 Aug 2013) $

内容摘要

Netkiller 系列手札 已经被 Github 收录，并备份保存在北极地下250米深的代码库中，备份会保留1000年。

Preserving open source software for future generations

The world is powered by open source software. It is a hidden cornerstone of modern civilization, and the shared heritage of all humanity.

The GitHub Arctic Code Vault is a data repository preserved in the Arctic World Archive (AWA), a very-long-term archival facility 250 meters deep in the permafrost of an Arctic mountain.

We are collaborating with the Bodleian Library in Oxford, the Bibliotheca Alexandrina in Egypt, and Stanford Libraries in California to store copies of 17,000 of GitHub's most popular and most-depended-upon projects—open source's "greatest hits"—in their archives, in museum-quality cases, to preserve them for future generations.

我们正在与牛津的博德利安图书馆、埃及的亚历山大图书馆和加利福尼亚州的斯坦福图书馆合作，将 GitHub 最受欢迎和最依赖的项目 17，000 份副本保存在他们的档案中，在博物馆质量的案例中，为子孙后代保存这些作品。

https://archiveprogram.github.com/arctic-vault/

我的系列文档:

Netkiller Linux 手札	Netkiller FreeBSD 手札	Netkiller Shell 手札	Netkiller Security 手札
Netkiller Web 手札	Netkiller Monitoring 手札	Netkiller Storage 手札	Netkiller Mail 手札
Netkiller Virtualization 手札	Netkiller Cryptography 手札		

以下文档停止更新合并到 《Netkiller Linux 手札》

Netkiller Debian 手札 Netkiller CentOS 手札 Netkiller Multimedia 手札

为什么写这篇文章

有很多想法,不能实现.工作中也用不到,所以想写出来,和大家分享.有一点写一点,写得也不好,就当学习笔记了.

我想到那写到那,你会发现文章没一个中心,今天这里写点,明天跳过本章写其它的.
文中例子决对多,对喜欢复制然后粘贴朋友很有用,不用动手写,也省时间.
理论的东西,网上大把,我这里就不写了,需要可以去网上查.
我爱写错别字,还有一些是打错的,如果发现请指正.
另外本文98%是我亲自编写，另有小部分来自引用网上，但作者不详.

致读者

Netkiller 系列手札 已经被 Github 收录，并备份保存在北极地下250米深的代码库中，备份会保留1000年。

Preserving open source software for future generations

The world is powered by open source software. It is a hidden cornerstone of modern civilization, and the shared heritage of all humanity.

The GitHub Arctic Code Vault is a data repository preserved in the Arctic World Archive (AWA), a very-long-term archival facility 250 meters deep in the permafrost of an Arctic mountain.

We are collaborating with the Bodleian Library in Oxford, the Bibliotheca Alexandrina in Egypt, and Stanford Libraries in California to store copies of 17,000 of GitHub’s most popular and most-depended-upon projects—open source’s “greatest hits”—in their archives, in museum-quality cases, to preserve them for future generations.

https://archiveprogram.github.com/arctic-vault/

自述

一直以来,我对数字证书,CA,PKI,非对称加密,非常感兴趣.很想写一篇这方面的文章,因为工作太忙的关系拖到至今.

对于CA,PKI我也是摸着石头过河,网上文章到是不少,全是理论性很强的文章,关于CA,PKI实现很少有文章提及.不要以为我是大拿.兴趣原因,我才写这篇文章.

我研究过一段时间Java.security.*,也写了一些例子,但工作中一直没用到.前几天我安装Mac OS X for x86,使用System Commander引导.System Commander破坏了我的硬盘分区表,我损失惨重,之前的例子也没了.

对数字证书兴趣也很浓,对于OpenSSL,OpenSSH,OpenPGP,GnuPG,SMIME...所用的证书和它们之间的关系一直搞不清楚,只知道他们可以结构很相似,所用的算法和标准,格式不同.

如果对我所谈的方面感兴趣可以去我的邮件列表讨论

1. 读者对象

本文档的读者对象:

文档面向有所有读者。您可以选读您所需要的章节,无需全篇阅读,因为有些章节不一定对你有用,用得着就翻来看看,暂时用不到的可以不看.

大体分来读者可以分为几类:

1. 系统管理员, 可以选读(Openssl,OpenSSH,OpenPGP...)
2. 程序开发人员, 可以选读(Java.security.*开发,SSL Socket开发)

3. 系统支持,部署工程师

不管是谁,做什么的,我希望通过阅读这篇文档都能对你有所帮助。

我的目的:

1. 通过阅读本文,你可以学会使用md5,sha,base64...技术

2. 懂得使用OpenSSL,OpenSSH,OpenPGP/GnuPG...

3. 使用Java.security.*来创建,修改删除,删除各种证书如:SMIME邮件签名与加密,Web服务器信认证书,SSL Socket开发...

4. 建立一个基本的PKI系统

2. 内容简介

本文档容简介:

文档全篇分为基础应用篇，管理篇，开发篇.

文档内容简介:

1. 基础应用篇，主要讲述一般日常用到的加密方法，如Office与PDF文档数字签名，一般权限设置与加密解密等，适合所有人阅读

2. 管理篇,主要是面向系统管理员，主要讲安装配置等

3. 开发篇,向要开发人员,讲述安全通信,和数据证书相关编程等.

3. 作者简介

陈景峯 (ㄔㄣ ㄐㄧㄥ ㄈㄥ)

Nickname： netkiller | English name: Neo chen | Nippon name: ちんけいほう (音訳) | Korean name: 천징봉 | Thailand name: ภูมิภาพภูเขา | Vietnam: Trần Cảnh Phong

Callsign: BG7NYT | QTH: ZONE CQ24 ITU44 ShenZhen, China

程序猿，攻城狮，挨踢民工, Full Stack Developer, UNIX like Evangelist, 业余无线电爱好者（呼号：BG7NYT）,户外运动，山地骑行以及摄影爱好者。

《Netkiller 系列 手札》 的作者

成长阶段

1981年1月19日(庚申年腊月十四)出生于黑龙江省青冈县建设乡双富大队第一小队

1989年9岁随父母迁居至黑龙江省伊春市，悲剧的天朝教育，不知道那门子归定，转学必须降一级，我本应该上一年级，但体制让我上学前班，那年多都10岁了

1995年小学毕业，体制规定借读要交3000两银子(我曾想过不升初中)，亲戚单位分楼告别平房，楼里没有地方放东西，把2麻袋书送给我，无意中发现一本电脑书BASIC语言，我竟然看懂了，对于电脑知识追求一发而不可收，后面顶零花钱，压岁钱主要用来买电脑书《MSDOS 6.22》《新编Unix实用大全》《跟我学Foxbase》。。。。。。

1996年第一次接触UNIX操作系统，BSD UNIX, Microsoft Xinux(盖茨亲自写的微软Unix，知道的人不多)

1997年自学Turbo C语言，苦于没有电脑，后来学校建了微机室才第一次使用QBASIC(DOS 6.22 自带命令)，那个年代只能通过软盘拷贝转播，Trubo C编译器始终没有搞到，

1997年第一次上Internet网速只有9600Bps, 当时全国兴起各种信息港域名格式是www.xxxx.info.net, 访问的第一个网站是NASA下载了很多火星探路者拍回的照片，还有“淞沪”sohu的前身

1998~2000年在哈尔滨学习计算机，充足的上机时间，但老师让我们练打字（明伦五笔/WT）打字不超过80个/每分钟还要强化训练，不过这个给我的键盘功夫打了好底。

1999年学校的电脑终于安装了光驱，在一张工具盘上终于找到了Turbo C, Borland C++与Quick Basic编译器，当时对VGA图形编程非常感兴趣，通过INT33中断控制鼠标，使用绘图函数模仿windows界面。还有操作 UCDOS 中文字库，绘制矢量与点阵字体。

2000年沉迷于Windows NT与Back Office各种技术，神马主域控制器，DHCP，WINS，IIS，域名服务器，Exchange邮件服务器，MS Proxy, NetMeeting...以及ASP+MS SQL开发；用56K猫下载了一张LINUX。ISO镜像，安装后我兴奋的24小时没有睡觉。

职业生涯

2001 年来深圳进城打工,成为一名外来务工者. 在一个4人公司做PHP开发，当时PHP的版本是2.0, 开始使用Linux Redhat 6.2.当时很多门户网站都是用FreeBSD,但很难搞到安装盘，在网易社区认识了一个网友,从广州给我寄了一张光盘，FreeBSD 3.2

2002 年我发现不能埋头苦干,还要学会"做人".后辗转广州工作了半年，考了一个Cisco CCNA认证。回到深圳重新开始，在车公庙找到一家工作做Java开发

2003 年这年最惨,公司拖欠工资16000元,打过两次官司2005才付清.

2004 年开始加入分布式计算团队,目前成绩，工作仍然是Java开发并且开始使用PostgreSQL数据库。

2004-10月开始玩户外和摄影

2005-6月成为中国无线电运动协会会员,呼号BG7NYT,进了一部Yaesu FT-60R手台。公司的需要转回PHP与MySQL，相隔几年发现PHP进步很大。在前台展现方面无人能敌，于是便前台使用PHP，后台采用Java开发。

2006 年单身生活了这么多年,终于找到归宿. 工作更多是研究PHP各种框架原理

2007 物价上涨,金融危机，休息了4个月（其实是找不到工作），关外很难上439.460中继，搞了一台Yaesu FT-7800.

2008 终于找到英文学习方法，《Netkiller Developer 手札》，《Netkiller Document 手札》

2008-8-8 08:08:08 结婚,后全家迁居湖南省常德市

2009《Netkiller Database 手札》,2009-6-13学车，年底拿到C1驾照

2010 对电子打击乐产生兴趣，计划学习爵士鼓。由于我对Linux热爱，我轻松的接管了公司的运维部，然后开发运维两把抓。我印象最深刻的是公司一次上架10个机柜，我们用买服务器纸箱的钱改善伙食。我将40多台服务器安装BOINC做压力测试，获得了中国第二的名次。

2011 平凡的一年，户外运动停止，电台很少开，中继很少上，摄影主要是拍女儿与家人，年末买了一辆山地车

2012 对油笔画产生了兴趣，活动基本是骑行银湖山绿道，

2013 开始学习民谣吉他，同时对电吉他也极有兴趣；最终都放弃了。这一年深圳开始推数字中继2013-7-6日入手Motorola

MOTOTRBO XIR P8668，Netkiller 系列手札从Sourceforge向Github迁移；年底对MYSQL UDF，Engine与PHP扩展开发产生很浓的兴趣，拾起遗忘10+年的C，写了几个mysql扩展（图片处理，fifo管道与ZeroMQ），10月份入Toyota Rezi 2.5V并写了一篇《攻城狮的苦逼选车经历》

2014-9-8 在淘宝上买了一架电钢琴 Casio Privia PX-5S pro 开始陪女儿学习钢琴，由于这家钢琴是合成器电钢，里面有打击乐，我有对键盘鼓产生了兴趣。

2014-10-2号罗浮山两日游，对中国道教文化与音乐产生了兴趣，10月5号用了半天时间学会了简谱。10月8号入Canon 5D Mark III + Canon Speedlite 600EX-RT香港过关被查。

2014-12-20号对乐谱制作产生兴趣（https://github.com/SheetMusic/Piano），给女儿做了几首钢琴伴奏曲，MuseScore制谱然后生成MIDI与WAV文件。

2015-09-01 晚饭后拿起爵士鼓基础教程尝试在Casio Privia PX-5S pro演练，经过反复琢磨加上之前学钢琴的乐理知识，终于在02号晚上，打出了简单的基本节奏，迈出了第一步。

2016 对弓箭（复合弓）产生兴趣，无奈天朝法律法规不让玩。每周游泳轻松1500米无压力，年底入 xbox one s 和 Yaesu FT-2DR, 同时开始关注功放音响这块

2017 7月9号入 Yamaha RX-V581 功放一台，连接Xbox打游戏爽翻了，入Kindle电子书，计划学习蝶泳，果断放弃运维和开发知识体系转攻区块链。

2018 从溪山美地搬到半岛城邦，丢弃了多年攒下的家底。11 月开始玩 MMDVM，使用 Yaesu FT-7800 发射，连接MMDVM中继板，树莓派，覆盖深圳湾，散步骑车通联两不误。

2019 卖了常德的房子，住了5次院，哮喘反复发作，决定停止电子书更新，兴趣转到知乎，B站

2020 准备找工作

职业生涯路上继续打怪升级

第 1 章 UUID (Universally Unique Identifier)

以前对UUID的了解很少，只知道是128位整数(16字节)的全局唯一标识符(Universally Unique Identifier)。

UUID 是指在一台机器上生成的数字，它保证对在同一时空中的所有机器都是唯一的。通常平台会提供生成UUID的API。UUID按照开放软件基金会(OSF)制定的标准计算，用到了以太网卡地址、纳秒级时间、芯片ID码和许多可能的数字。由以下几部分的组合：当前日期和时间(UUID的第一个部分与时间有关，如果你在生成一个UUID之后，过几秒又生成一个UUID，则第一个部分不同，其余相同)，时钟序列，全局唯一的IEEE机器识别号（如果有网卡，从网卡获得，没有网卡以其他方式获得），UUID的唯一缺陷在于生成的结果串会比较长。关于UUID这个标准使用最普遍的是微软的GUID(Globals Unique Identifiers)。

其格式为： xxxxxxxx-xxxx-xxxx-xxxxxxxxxxxxxxxx(8-4-4-16)，其中每个 x 是 0-9 或 a-f 范围内的一个十六进制的数字。而标准的UUID格式为：xxxxxxxx-xxxx-xxxx-xxxxxx-xxxxxxxxxx (8-4-4-4-12)

使用UUID的好处在分布式的软件系统中（比如：DCE/RPC, COM+,CORBA）就能体现出来，它能保证每个节点所生成的标识都不会重复，并且随着WEB服务等整合技术的发展，UUID的优势将更加明显。

http://en.wikipedia.org/wiki/UUID

RFC

1. GUID

GUID是UUID的windows实现，GUID也是一个128位长的数字，一般用16进制表示。算法的核心思想是结合机器的网卡、当地时间、一个随机数来生成GUID。从理论上讲，如果一台机器每秒产生10000000个GUID，则可以保证（概率意义上）3240年不重复。

到微软件网站下载GUIDGEN.EXE来生成GUID

点击"New GUID"生成新GUID

单击"Copy"复制到剪贴板

生成的GUID：{12466768-64A9-426a-A2E8-ABFDB016B248}

2. Subversion

svnlook uuid — 打印版本库的UUID。

```
svnlook uuid REPOS_PATH
```

打印版本库的UUID，UUID是版本库的universal unique IDentifier（全局唯一标示），Subversion客户端可以使用这个标示区分不同的版本库。

```
$ svnlook uuid /usr/local/svn/repos
e7fe1b91-8cd5-0310-98dd-2f12e793c5e8
```

请参考：http://www.subversion.org.cn/svnbook/nightly/index.html

3. PHP UUID

```
<?php

/* Copyright 2006 Maciej Strzelecki



   This program is free software; you can redistribute it and/o
r modify

   it under the terms of the GNU General Public License as publ
ished by

   the Free Software Foundation; either version 2 of the Licens
e, or

   (at your option) any later version.



   This program is distributed in the hope that it will be usef
ul,

   but WITHOUT ANY WARRANTY; without even the implied warranty 
of

   MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See th
e

   GNU General Public License for more details.



   You should have received a copy of the GNU General Public Li
```

```
cense
   along with this program; if not, write to the Free Software
   Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA  0
2110-1301  USA */


function uuid()
{
   // version 4 UUID
   return sprintf(
       '%08x-%04x-%04x-%02x%02x-%012x',
       mt_rand(),
       mt_rand(0, 65535),
       bindec(substr_replace(
           sprintf('%016b', mt_rand(0, 65535)), '0100', 11, 4)
       ),
       bindec(substr_replace(sprintf('%08b', mt_rand(0, 255)),
'01', 5, 2)),
       mt_rand(0, 255),
       mt_rand()
   );
}
?>
```

参考：http://cn.php.net/uniqid

4. JAVA UUID

```
import java.util.UUID;
public class Test {
        public static void main(String[] args) {
                UUID uuid = UUID.randomUUID();
                System.out.println (uuid);
        }
}
编译运行输出:
07ca3dec-b674-41d0-af9e-9c37583b08bb
```

参考：http://java.sun.com/j2se/1.5.0/docs/api/java/util/UUID.html

5. PERL UUID

```
#!/usr/bin/perl
use CGI::Carp qw(fatalsToBrowser);

my $uuid_str;
if (@ARGV) {
  $uuid_str = $ARGV[0];
} else {
  eval {
    require Data::UUID;
    my $ug = new Data::UUID;
    $uuid_str = $ug->create_str;
  };
  if ($@) {
    $uuid_str = `uuidgen`;
    $uuid_str =~ s/\r?\n?$//;
  }
}
my @stuff = split /-/, $uuid_str;

print "Content-type: text/html\n\n";
print "<html><head><title>GUID Generator</title></head><body>";
print '<h2><font face="verdana, arial">GUID Generator</font>
</h2>';
print '<font face="new courier, courier">';
print "{$uuid_str}</font><br>";
print '<h6><font face="verdana, arial"><a
href="http://extensions.roachfiend.com/cgi-bin/guid.pl">Get
another GUID</a></font></h6>';
print '<h6><font face="verdana, arial"><a
href="http://extensions.roachfiend.com/guid.txt">View the
source of this script</a></font></h6>';
print "</body></html>";
exit;
```

参考：http://extensions.roachfiend.com/cgi-bin/guid.pl

6. Python UUID

```
"""UUID (universally unique identifiers) as specified in RFC
4122.

This module provides the UUID class and the functions uuid1(),
uuid3(),
uuid4(), uuid5() for generating version 1, 3, 4, and 5 UUIDs
respectively.

This module works with Python 2.3 or higher."""

__author__ = 'Ka-Ping Yee <ping@zesty.ca>'
__date__ = '$Date: 2012-02-01 13:40:49 +0800 (Wed, 01 Feb 2012)
$'.split()[1].replace('/', '-')
__version__ = '$Revision: 333 $'

RESERVED_NCS, RFC_4122, RESERVED_MICROSOFT, RESERVED_FUTURE = [
    'reserved for NCS compatibility', 'specified in RFC 4122',
    'reserved for Microsoft compatibility', 'reserved for
future definition']

class UUID(object):
    """Instances of the UUID class represent UUIDs as specified
in RFC 4122.
    Converting a UUID to a string using str() produces a string
in the form
    "{12345678-1234-1234-1234-123456789abc}".  The UUID
constructor accepts
    a similar string (braces and hyphens optional), or six
integer arguments
    (with 32-bit, 16-bit, 16-bit, 8-bit, 8-bit, and 48-bit
values
    respectively).  UUID objects have the following attributes:

        bytes       gets or sets the UUID as a 16-byte string

        urn         gets the UUID as a URN as specified in RFC
4122

        variant     gets or sets the UUID variant as one of the
```

```
constants
                    RESERVED_NCS, RFC_4122, RESERVED_MICROSOFT,
RESERVED_FUTURE

        version     gets or sets the UUID version number (1
through 5)
    """

    def __init__(self, *args):
        """Create a UUID either from a string representation in
hexadecimal
        or from six integers (32-bit time_low, 16-bit time_mid,
16-bit
        time_hi_ver, 8-bit clock_hi_res, 8-bit clock_low, 48-
bit node)."""
        if len(args) == 1:
            digits = args[0].replace('urn:',
'').replace('uuid:', '')
            digits = digits.replace('{', '').replace('}',
'').replace('-', '')
            assert len(digits) == 32, ValueError('badly formed
UUID string')
            time_low = int(digits[:8], 16)
            time_mid = int(digits[8:12], 16)
            time_hi_ver = int(digits[12:16], 16)
            clock_hi_res = int(digits[16:18], 16)
            clock_low = int(digits[18:20], 16)
            node = int(digits[20:32], 16)
        else:
            (time_low, time_mid, time_hi_ver,
             clock_hi_res, clock_low, node) = args
        assert 0 <= time_low < 0x100000000,
ValueError('time_low out of range')
        assert 0 <= time_mid < 1<<16, ValueError('time_mid out
of range')
        assert 0 <= time_hi_ver < 1<<16,
ValueError('time_hi_ver out of range')
        assert 0 <= clock_hi_res < 1<<8,
ValueError('clock_hi_res out of range')
        assert 0 <= clock_low < 1<<8, ValueError('clock_low out
of range')
        assert 0 <= node < 0x1000000000000, ValueError('node
out of range')
        self.time_low = time_low
        self.time_mid = time_mid
        self.time_hi_ver = time_hi_ver
        self.clock_hi_res = clock_hi_res
        self.clock_low = clock_low
        self.node = node

    def __cmp__(self, other):
        return cmp(self.bytes, getattr(other, 'bytes', other))

    def __str__(self):
        return '{%08x-%04x-%04x-%02x%02x-%012x}' % (
            self.time_low, self.time_mid, self.time_hi_ver,
            self.clock_hi_res, self.clock_low, self.node)

    def __repr__(self):
        return 'UUID(%r)' % str(self)

    def get_bytes(self):
        def byte(n):
            return chr(n & 0xff)

        return (byte(self.time_low >> 24) + byte(self.time_low
>> 16) +
                byte(self.time_low >> 8) + byte(self.time_low)
+
                byte(self.time_mid >> 8) + byte(self.time_mid)
+
                byte(self.time_hi_ver >> 8) +
byte(self.time_hi_ver) +
                byte(self.clock_hi_res) + byte(self.clock_low)
+
                byte(self.node >> 40) + byte(self.node >> 32) +
                byte(self.node >> 24) + byte(self.node >> 16) +
                byte(self.node >> 8) + byte(self.node))

    def set_bytes(self, bytes):
        values = map(ord, bytes)
        self.time_low = ((values[0] << 24) + (values[1] << 16)
+
                         (values[2] << 8) + values[3])
        self.time_mid = (values[4] << 8) + values[5]
        self.time_hi_ver = (values[6] << 8) + values[7]
        self.clock_hi_res = values[8]
        self.clock_low = values[9]
        self.node = ((values[10] << 40) + (values[11] << 32) +
                     (values[12] << 24) + (values[13] << 16) +
                    (values[14] << 8) + values[15])

    bytes = property(get_bytes, set_bytes)

    def get_urn(self):
        return 'urn:uuid:%08x-%04x-%04x-%02x%02x-%012x' % (
            self.time_low, self.time_mid, self.time_hi_ver,
            self.clock_hi_res, self.clock_low, self.node)

    urn = property(get_urn)

    def get_variant(self):
        if not self.clock_hi_res & 0x80:
            return RESERVED_NCS
        elif not self.clock_hi_res & 0x40:
            return RFC_4122
        elif not self.clock_hi_res & 0x20:
            return RESERVED_MICROSOFT
        else:
            return RESERVED_FUTURE

    def set_variant(self, variant):
        if variant == RESERVED_NCS:
            self.clock_hi_res &= 0x7f
        elif variant == RFC_4122:
            self.clock_hi_res &= 0x3f
            self.clock_hi_res |= 0x80
        elif variant == RESERVED_MICROSOFT:
            self.clock_hi_res &= 0x1f
            self.clock_hi_res |= 0xc0
        elif variant == RESERVED_FUTURE:
            self.clock_hi_res &= 0x1f
            self.clock_hi_res |= 0xe0
        else:
            raise ValueError('illegal variant identifier')

    variant = property(get_variant, set_variant)

    def get_version(self):
        return self.time_hi_ver >> 12

    def set_version(self, version):
        assert 1 <= version <= 5, ValueError('illegal version
number')
        self.time_hi_ver &= 0x0fff
        self.time_hi_ver |= (version << 12)

    version = property(get_version, set_version)

def unixgetaddr(program):
    """Get the hardware address on a Unix machine."""
    from os import popen
    for line in popen(program):
        words = line.lower().split()
        if 'hwaddr' in words:
            addr = words[words.index('hwaddr') + 1]
            return int(addr.replace(':', ''), 16)
        if 'ether' in words:
            addr = words[words.index('ether') + 1]
            return int(addr.replace(':', ''), 16)

def wingetaddr(program):
    """Get the hardware address on a Windows machine."""
    from os import popen
    for line in popen(program + ' /all'):
        if line.strip().lower().startswith('physical address'):
            addr = line.split(':')[-1].strip()
            return int(addr.replace('-', ''), 16)

def getaddr():
    """Get the hardware address as a 48-bit integer."""
    from os.path import join, isfile
    for dir in ['/sbin', '/usr/sbin', r'c:\windows',
                r'c:\windows\system', r'c:\windows\system32']:
        if isfile(join(dir, 'ifconfig')):
            return unixgetaddr(join(dir, 'ifconfig'))
        if isfile(join(dir, 'ipconfig.exe')):
            return wingetaddr(join(dir, 'ipconfig.exe'))

def uuid1():
    """Generate a UUID based on the time and hardware
address."""
    from time import time
    from random import randrange
    nanoseconds = int(time() * 1e9)
    # 0x01b21dd213814000 is the number of 100-ns intervals
between the
    # UUID epoch 1582-10-15 00:00:00 and the Unix epoch 1970-
01-01 00:00:00.
    timestamp = int(nanoseconds/100) + 0x01b21dd213814000
    clock = randrange(1<<16) # don't use stable storage
    time_low = timestamp & (0x100000000 - 1)
    time_mid = (timestamp >> 32) & 0xffff
    time_hi_ver = (timestamp >> 48) & 0x0fff
    clock_low = clock & 0xff
    clock_hi_res = (clock >> 8) & 0x3f
    node = getaddr()
    uuid = UUID(time_low, time_mid, time_hi_ver, clock_low,
clock_hi_res, node)
    uuid.variant = RFC_4122
    uuid.version = 1
    return uuid

def uuid3(namespace, name):
    """Generate a UUID from the MD5 hash of a namespace UUID
and a name."""
    from md5 import md5
    uuid = UUID(0, 0, 0, 0, 0, 0)
    uuid.bytes = md5(namespace.bytes + name).digest()[:16]
    uuid.variant = RFC_4122
    uuid.version = 3
    return uuid

def uuid4():
    """Generate a random UUID."""
    try:
        from os import urandom
    except:
        from random import randrange
        uuid = UUID(randrange(1<<32), randrange(1<<16),
randrange(1<<16),
                    randrange(1<<8), randrange(1<<8),
randrange(1<<48))
    else:
        uuid = UUID(0, 0, 0, 0, 0, 0)
        uuid.bytes = urandom(16)
    uuid.variant = RFC_4122
    uuid.version = 4
    return uuid

def uuid5(namespace, name):
    """Generate a UUID from the SHA-1 hash of a namespace UUID
and a name."""
    from sha import sha
    uuid = UUID(0, 0, 0, 0, 0, 0)
    uuid.bytes = sha(namespace.bytes + name).digest()[:16]
    uuid.variant = RFC_4122
    uuid.version = 5
    return uuid

NAMESPACE_DNS = UUID('{6ba7b810-9dad-11d1-80b4-00c04fd430c8}')
NAMESPACE_URL = UUID('{6ba7b811-9dad-11d1-80b4-00c04fd430c8}')
NAMESPACE_OID = UUID('{6ba7b812-9dad-11d1-80b4-00c04fd430c8}')
NAMESPACE_X500 = UUID('{6ba7b814-9dad-11d1-80b4-00c04fd430c8}')
```

参考：http://zesty.ca/python/uuid.html

参考：https://svn.n-h.com/svn/exchange4linux/trunk/src/BILL-StorageServer/UUID.py

7. MySQL uuid()

```
mysql> select uuid();
+--------------------------------------+
| uuid()                               |
+--------------------------------------+
| 2f761256-8360-102c-b767-001cc07156cb |
+--------------------------------------+
1 row in set (0.02 sec)
```

8. linux command uuid

```
$ sudo apt-get install uuid

$ uuid
5ce08f58-21ac-11de-a16f-001cc07156cb
```

第 2 章 Encode & Decode

1. MIME (BASE64) 专题

什么是Base64？

按照RFC2045的定义，Base64被定义为：Base64内容传送编码被设计用来把任意序列的8位字节描述为一种不易被人直接识别的形式。

为什么要使用Base64？

在设计这个编码的时候，我想设计人员最主要考虑了3个问题：

1. 是否加密？
2. 加密算法复杂程度和效率？
3. 如何处理传输？

加密是肯定的，但是加密的目的不是让用户发送非常安全的Email。这种加密方式主要就是“防君子不防小人”。即达到一眼望去完全看不出内容即可。 基于这个目的加密算法的复杂程度和效率也就不能太大和太低。和上一个理由类似，MIME协议等用于发送Email的协议解决的是如何收发Email，而并不是如何安全的收发Email。因此算法的复杂程度要小，效率要高，否则因为发送Email而大量占用资源，路就有点走歪了。

但是，如果是基于以上两点，那么我们使用最简单的恺撒法即可，为什么Base64看起来要比恺撒法复杂呢？这是因为在Email的传送过程中，由于历史原因，Email只被允许传送ASCII字符，即一个8位字节的低7位。因此，如果您发送了一封带有非ASCII字符（即字节的最高位是1）的Email通过有“历史问题”的网关时就可能会出现问题。网关可能会把最高位置为0！很明显，问题就这样产生了！因此，为了能够正常的传送Email，这个问题就必须考虑！所以，单单靠改变字母的

位置的恺撒之类的方案也就不行了。关于这一点可以参考 RFC2046。基于以上的一些主要原因产生了Base64编码。

参考邮件正文 Content-Transfer-Encoding: base64

OpenSSL - Base64

1.1. Linux Command base64

```
$ cat file | base64
```

1.2. PHP Base64

base64_encode

base64_encode

(PHP 3, PHP 4, PHP 5)

base64_encode -- 使用 MIME base64 对数据进行编码

说明

string base64_encode (string data)

base64_encode() returns 使用 base64 对 data 进行编码。设计此种编码是为了使二进制数据可以通过非纯 8-bit 的传输层传输，例如电子邮件的主体。

Base64-encoded 数据要比原始数据多占用 33% 左右的空间。

例子 1. base64_encode() 示例

```

<?php
```

```
  $str = 'This is an encoded string';
  echo base64_encode($str);
?>
```

此示例将显示：

VGhpcyBpcyBhbiBlbmNvZGVkIHN0cmluZw==

例子 2. stream_filter_append() 示例

```
<?php
$fp = fopen('php://output', 'w');
stream_filter_append($fp, 'convert.base64-encode');
fwrite($fp, "This is a test.\n");
fclose($fp);
/* Outputs:  VGhpcyBpcyBhIHRlc3QuCg==  */
echo "\n=============================================\n";

$fp = fopen('php://output', 'w');
stream_filter_append($fp, 'convert.base64-decode');
fwrite($fp, "VGhpcyBpcyBhIHRlc3QuCg==");
fclose($fp);
/* Outputs:  This is a test.  */
echo "===========================================\n";

$param = array('line-length' => 8, 'line-break-chars' =>
"\r\n");
$fp = fopen('php://output', 'w');
stream_filter_append($fp, 'convert.base64-encode',
STREAM_FILTER_WRITE, $param);
fwrite($fp, "This is a test.\n");
fclose($fp);
/* Outputs:  VGhpcyBp
         :  cyBhIHRl
         :  c3QuCg==  */
?>
```

base64_decode

base64_decode

(PHP 3, PHP 4, PHP 5)

base64_decode -- 对使用 MIME base64 编码的数据进行解码

说明

string base64_decode (string encoded_data)

base64_decode() 对 encoded_data 进行解码，返回原始数据，失败则返回 FALSE。返回的数据可能是二进制的。

例子 1. base64_decode() 示例

```
<?php
$str = 'VGhpcyBpcyBhbiBlbmNvZGVkIHN0cmluZw==';
echo base64_decode($str);
?>
```

此示例将显示：

This is an encoded string

1.3. Python Base64

编码：b64encode

```
import base64
base64.b64encode('This is an encoded string')
```

此示例将显示:

'VGhpcyBpcyBhbiBlbmNvZGVkIHN0cmluZw=='

解码:

```
import base64
base64.b64decode('VGhpcyBpcyBhbiBlbmNvZGVkIHN0cmluZw==')
```

此示例将显示:

This is an encoded string

1.4. perl base64

```
perl -MMIME::Base64 -e 'print encode_base64("netkiller");'

perl -MMIME::Base64 -e 'print decode_base64("bmV0a2lsbGVy");'
```

1.5. Java Base64

Java 7

```
import java.io.*;
public class base64Test {

        public static void main(String[] args) {

            try {
                String text = "This is an encoded string";
                //Convert a string to base64 string
                byte[] buf = text.getBytes();
                String encode = new
sun.misc.BASE64Encoder().encode(buf);
```

```
                System.out.println(encode);

                // Convert base64 string to a string
                buf = new
sun.misc.BASE64Decoder().decodeBuffer(encode);
                String decode = new String(buf);
                System.out.println(decode);
            } catch (IOException e) {

            }
        }
}
```

Java 8

```
package cn.netkiller.test;

import java.nio.charset.StandardCharsets;
import java.util.Base64;

public class Base64Test {
        public static void main(String[] args) {
                final String text =
"http://www.netkiller.cn/index.html";

                final String encoded =
Base64.getEncoder().encodeToString(text.getBytes(StandardChar
sets.UTF_8));
                System.out.println(encoded);

                final String decoded = new
String(Base64.getDecoder().decode(encoded),
StandardCharsets.UTF_8);
                System.out.println(decoded);
        }
}
```

```
package cn.netkiller.security;

import java.io.UnsupportedEncodingException;
import java.util.Base64;

public class Base64Test {

        public Base64Test() {
                // TODO Auto-generated constructor stub
        }

        public static void main(String[] args) throws
UnsupportedEncodingException {
                // TODO Auto-generated method stub
                String asB64 =
Base64.getEncoder().encodeToString("some
string".getBytes("utf-8"));
                System.out.println(asB64); // 输出为:
c29tZSBzdHJpbmc=

                // 解码
                byte[] asBytes =
Base64.getDecoder().decode("c29tZSBzdHJpbmc=");
                System.out.println(new String(asBytes, "utf-
8")); // 输出为: some string

                // 但由于URL对反斜线“/”有特殊的意义，因此URL编码需要
替换掉它，使用下划线替换
                String basicEncoded =
Base64.getEncoder().encodeToString("subjects?
abcd".getBytes("utf-8"));
                System.out.println("Using Basic Alphabet: " +
basicEncoded);

                String urlEncoded =
Base64.getUrlEncoder().encodeToString("subjects?
abcd".getBytes("utf-8"));
                System.out.println("Using URL Alphabet: " +
urlEncoded);
        }

}
```

```

```

1.6. C/C++ Base64

2. Uuencode

注意

uuencode不是MIME标准

application/x-uuencode

Uuencode 是将二进制文件以文本文件方式进行编码表示、以利于基于文本传输环境中进行二进制文件的传输/交换的编码方法之一， 在邮件系统/二进制新闻组中使用频率比较高，经常用于附件二进制文件。

这种编码的特征是：每一行开头用“M”标志。

Uuencode的算法很简单，编码时它将3个字符顺序放入一个 24 位的缓冲区，缺字符的地方补零，然后将缓冲区截断成为 4 个部分，高位在先，每个部分 6 位，用下面的64个字符重新表示：

"`!"#$%&'()*+,-./0123456789:;<=>?
@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_"

解码时它将4个字符分别转换为4个6位字符后，截取有用的后六位放入一个 24 位的缓冲区，即得3个二进制代码。

2.1. PHP uuencode

编码：convert_uuencode()

```
<?php
$some_string = "test\ntext text\r\n";
echo convert_uuencode($some_string);
```

```
?>
```

解码：convert_uuencode()

```
<?php
        $some_string = "This is an encoded string";
        $encode = convert_uuencode($some_string);
        echo convert_uudecode($encode);
?>
```

3. Quoted-Printable

Quoted-Printable也是MIME邮件中常用的编码方式之一。同Base64一样，它也将输入的字符串或数据编码成全是ASCII码的可打印字符串。

Quoted-Printable编码的基本方法是：输入数据在33-60、62-126范围内的，直接输出；其它的需编码为“=”加两个字节的HEX码(大写)。为保证输出行不超过规定长度，可在行尾加“=\r\n”序列作为软回车。

3.1. C Quoted-Printable

```
int EncodeQuoted(const unsigned char* pSrc, char* pDst, int
nSrcLen, int nMaxLineLen)
{
    int nDstLen;        // 输出的字符计数
    int nLineLen;       // 输出的行长度计数

    nDstLen = 0;
    nLineLen = 0;

    for (int i = 0; i < nSrcLen; i++, pSrc++)
    {
        // ASCII 33-60, 62-126原样输出，其余的需编码
        if ((*pSrc >= '!') && (*pSrc <= '~') && (*pSrc != '='))
        {
            *pDst++ = (char)*pSrc;
            nDstLen++;
            nLineLen++;
        }
        else
        {
            sprintf(pDst, "=%02X", *pSrc);
            pDst += 3;
            nDstLen += 3;
            nLineLen += 3;
        }
        // 输出换行?
        if (nLineLen >= nMaxLineLen - 3)
        {
            sprintf(pDst, "=\r\n");
            pDst += 3;
            nDstLen += 3;
            nLineLen = 0;
        }
    }

    // 输出加个结束符
    *pDst = '\0';

    return nDstLen;
}
```

Quoted-Printable解码很简单，将编码过程反过来就行了。

```
int DecodeQuoted(const char* pSrc, unsigned char* pDst, int
nSrcLen)
{
    int nDstLen;        // 输出的字符计数
    int i;

    i = 0;
    nDstLen = 0;

    while (i < nSrcLen)
    {
        if (strncmp(pSrc, "=\r\n", 3) == 0)        // 软回车，跳
过
        {
            pSrc += 3;
            i += 3;
        }
        else
        {
            if (*pSrc == '=')        // 是编码字节
            {
                sscanf(pSrc, "=%02X", pDst);
                pDst++;
                pSrc += 3;
                i += 3;
            }
            else        // 非编码字节
            {
                *pDst++ = (unsigned char)*pSrc++;
                i++;
            }

            nDstLen++;
        }
    }

    // 输出加个结束符
    *pDst = '\0';

    return nDstLen;
}
```

参考:http://dev.csdn.net/develop/article/19/19205.shtm

3.2. Java Quoted-Printable

3.3. Python Quoted-Printable

4. Base58

什么是Base58?

4.1. php

```
<?php

$number = '12345678900987654321123456789 0';
$result = base58_encode($number);
echo('Encoded: ' . $result . '<br>');
echo('Decoded: ' . base58_decode($result) . '<br>');


function base58_encode($input)
{
    $alphabet =
'123456789abcdefghijkmnopqrstuvwxyzABCDEFGHJKLMNPQRSTUVWXYZ';
    $base_count = strval(strlen($alphabet));
    $encoded = '';
    while (floatval($input) >= floatval($base_count))
    {
        $div = bcdiv($input, $base_count);
        $mod = bcmod($input, $base_count);
        $encoded = substr($alphabet, intval($mod), 1) .
$encoded;
        $input = $div;
    }
    if (floatval($input) > 0)
    {
        $encoded = substr($alphabet, intval($input), 1) .
$encoded;
    }
    return($encoded);
}

function base58_decode($input)
{
    $alphabet =
'123456789abcdefghijkmnopqrstuvwxyzABCDEFGHJKLMNPQRSTUVWXYZ';
    $base_count = strval(strlen($alphabet));
    $decoded = strval(0);
    $multi = strval(1);
    while (strlen($input) > 0)
    {
        $digit = substr($input, strlen($input) - 1);
        $decoded = bcadd($decoded, bcmul($multi,
strval(strpos($alphabet, $digit))));
        $multi = bcmul($multi, $base_count);
        $input = substr($input, 0, strlen($input) - 1);
    }
    return($decoded);
}
```

4.2. Java Base58

Maven 文件添加下面代码

```
<repositories>
  <repository>
      <id>jitpack.io</id>
      <url>https://jitpack.io</url>
  </repository>
</repositories>

<dependencies>
  <dependency>
    <groupId>com.github.multiformats</groupId>
    <artifactId>java-multihash</artifactId>
    <version>$LATEST_VERSION</version>
  </dependency>
</dependencies>
```

```
import io.ipfs.multibase.Base58;
```

```
Base58.encode(string);

Base58.decode(encrypted)



Multihash m =
Multihash.fromBase58("QmatmE9msSfkKxoffpHwNLNKgwZG8eT9Bud6YoP
ab52vpy");
```

第 3 章 Message Digest (数字摘要)

1. MD5专题

1.1. md5sum

MD5 为当前常用的 hash function,一般用来计算资料的杂凑值,俾利资料正确性之验证；md5sum 则为用来检查计算hash function 的的工具程序，具体的参数用法可去man md5sum 的用法。

生成杂凑值,有些文章叫指纹

md5sum file.txt

```
C:\GnuWin32\neo>md5sum file.txt
7012acbb1d394b20567dffbf0992b677 *file.txt

C:\GnuWin32\neo>md5sum file.txt > file.txt.md5

C:\GnuWin32\neo>md5sum -c file.txt.md5
file.txt: OK
```

生成指纹并重订向到文件

md5sum file.txt > file.txt.md5

```
C:\GnuWin32\neo>md5sum file.txt
7012acbb1d394b20567dffbf0992b677 *file.txt

C:\GnuWin32\neo>md5sum file.txt > file.txt.md5

C:\GnuWin32\neo>md5sum -c file.txt.md5
file.txt: OK
```

生成一组文件

```
md5sum file0.txt > file.txt.md5
md5sum file1.txt >> file.txt.md5
md5sum file2.txt >> file.txt.md5
```

使用通配符

```
C:\GnuWin32\neo>md5sum *
7012acbb1d394b20567dffbf0992b677 *file.txt
d9226d4bd8779baa69db272f89a2e05c *message.txt

C:\GnuWin32\neo>md5sum * >file.txt.md5
```

验证文件是否被人更改过

md5sum -c file.txt.md5

```
C:\GnuWin32\neo>md5sum file.txt
7012acbb1d394b20567dffbf0992b677 *file.txt

C:\GnuWin32\neo>md5sum file.txt > file.txt.md5

C:\GnuWin32\neo>md5sum -c file.txt.md5
file.txt: OK
```

1.2. PHP md5()

```


# cat md5.php

<html>

<p>MD5 密码产生器</p>


```

```
<form method=post action=des.php>

<p>password:<input name=passwd type=text size=20></p>

<input type=submit value=submit>

</form>

<?

$enpw=md5($passwd);

echo "password is: $enpw";

?>
```

1.3. MySQL md5()

select md5('password');

```
[chen@linux chen]$ mysql

Welcome to the MySQL monitor.  Commands end with ; or \g.

Your MySQL connection id is 11947 to server version: 4.0.13-log


Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> select md5('chen');

+----------------------------------+
| md5('chen')                      |
+----------------------------------+
| a1a8887793acfc199182a649e905daab |
+----------------------------------+

1 row in set (0.00 sec)

mysql>
mysql> select md5('chen') as passwd;

+----------------------------------+
| passwd                           |
+----------------------------------+
| a1a8887793acfc199182a649e905daab |
+----------------------------------+

1 row in set (0.00 sec)

mysql>
```

1.4. Java MD5

JDK 1.2

1.2版之前的JDK没有实现md5;

```
/*************************************************
MD5 算法的Java Bean
@author:Topcat Tuppin
Last Modified:10,Mar,2001
*************************************************/
package netkiller.security;
import java.lang.reflect.*;
/*************************************************
md5 类实现了RSA Data Security, Inc.在提交给IETF
的RFC1321中的MD5 message-digest 算法。
*************************************************/

public class MD5 {
       /* 下面这些S11-S44实际上是一个4*4的矩阵，在原始的C实现中是用
#define 实现的，
       这里把它们实现成为static final是表示了只读，切能在同一个进程空间
内的多个
       Instance间共享*/
```

```
        static final int S11 = 7;
        static final int S12 = 12;
        static final int S13 = 17;
        static final int S14 = 22;

        static final int S21 = 5;
        static final int S22 = 9;
        static final int S23 = 14;
        static final int S24 = 20;

        static final int S31 = 4;
        static final int S32 = 11;
        static final int S33 = 16;
        static final int S34 = 23;

        static final int S41 = 6;
        static final int S42 = 10;
        static final int S43 = 15;
        static final int S44 = 21;



        static final byte[] PADDING = { -128, 0, 0, 0, 0, 0, 0,
0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
};
        /* 下面的三个成员是MD5计算过程中用到的3个核心数据，在原始的C实现
中
           被定义到MD5_CTX结构中
         */

        private long[] state = new long[4];  // state (ABCD)
        private long[] count = new long[2];  // number of bits,
modulo 2^64 (lsb first)
        private byte[] buffer = new byte[64]; // input buffer

       /* digestHexStr是MD5的唯一一个公共成员，是最新一次计算结果的
          16进制ASCII表示.
       */

        public String digestHexStr;

        /* digest,是最新一次计算结果的2进制内部表示，表示128bit的MD5
```

```
值. */

        private byte[] digest = new byte[16];

       /*
         getMD5ofStr是类MD5最主要的公共方法，入口参数是你想要进行MD5变
换的字符串
         返回的是变换完的结果，这个结果是从公共成员digestHexStr取得的.
       */

        public String getMD5ofStr(String inbuf) {

                md5Init();

                md5Update(inbuf.getBytes(), inbuf.length());

                md5Final();

                digestHexStr = "";

                for (int i = 0; i < 16; i++) {

                        digestHexStr += byteHEX(digest[i]);

                }

                return digestHexStr;



        }

        // 这是MD5这个类的标准构造函数，JavaBean要求有一个public的并且
没有参数的构造函数

        public MD5() {
                md5Init();
                return;
        }

        /* md5Init是一个初始化函数，初始化核心变量，装入标准的幻数 */

        private void md5Init() {

                count[0] = 0L;
                count[1] = 0L;
                ///* Load magic initialization constants.


                state[0] = 0x67452301L;
                state[1] = 0xefcdab89L;
                state[2] = 0x98badcfeL;
                state[3] = 0x10325476L;


                return;
        }
        /* F, G, H ,I 是4个基本的MD5函数，在原始的MD5的C实现中，由于它们是
        简单的位运算，可能出于效率的考虑把它们实现成了宏，在java中，我们把它们
         实现成了private方法，名字保持了原来C中的。 */


        private long F(long x, long y, long z) {
                return (x & y) | ((~x) & z);


        }
        private long G(long x, long y, long z) {
                return (x & z) | (y & (~z));
        }

        private long H(long x, long y, long z) {

                return x ^ y ^ z;

        }


        private long I(long x, long y, long z) {

                return y ^ (x | (~z));

        }


       /*

          FF,GG,HH和II将调用F,G,H,I进行近一步变换

          FF, GG, HH, and II transformations for rounds 1, 2,
3, and 4.

          Rotation is separate from addition to prevent
recomputation.

       */


        private long FF(long a, long b, long c, long d, long x,
long s,

                long ac) {

                a += F (b, c, d) + x + ac;

                a = ((int) a << s) | ((int) a >>> (32 - s));

                a += b;

                return a;
        }


        private long GG(long a, long b, long c, long d, long x,
long s,

                long ac) {

                a += G (b, c, d) + x + ac;

                a = ((int) a << s) | ((int) a >>> (32 - s));

                a += b;

                return a;

        }

        private long HH(long a, long b, long c, long d, long x,
long s,

                long ac) {

                a += H (b, c, d) + x + ac;

                a = ((int) a << s) | ((int) a >>> (32 - s));

                a += b;

                return a;

        }

        private long II(long a, long b, long c, long d, long x,
long s,

                long ac) {

                a += I (b, c, d) + x + ac;

                a = ((int) a << s) | ((int) a >>> (32 - s));

                a += b;
                return a;

        }

        /*
         md5Update是MD5的主计算过程，inbuf是要变换的字节串，inputlen
是长度，这个
         函数由getMD5ofStr调用，调用之前需要调用md5init，因此把它设计
成private的
        */

        private void md5Update(byte[] inbuf, int inputLen) {

                int i, index, partLen;

                byte[] block = new byte[64];

                index = (int)(count[0] >>> 3) & 0x3F;

                // /* Update number of bits */

                if ((count[0] += (inputLen << 3)) < (inputLen
<< 3))

                        count[1]++;

                count[1] += (inputLen >>> 29);

                partLen = 64 - index;

                // Transform as many times as possible.

                if (inputLen >= partLen) {

                        md5Memcpy(buffer, inbuf, index, 0,
partLen);

                        md5Transform(buffer);

                        for (i = partLen; i + 63 < inputLen; i
+= 64) {

                                md5Memcpy(block, inbuf, 0, i,
64);
```

```
                                md5Transform (block);

                        }

                        index = 0;

                } else

                        i = 0;


                ///* Buffer remaining input */

                md5Memcpy(buffer, inbuf, index, i, inputLen -
i);

        }

        /*
          md5Final整理和填写输出结果
        */

        private void md5Final () {

                byte[] bits = new byte[8];

                int index, padLen;

                ///* Save number of bits */

                Encode (bits, count, 8);


                ///* Pad out to 56 mod 64.

                index = (int)(count[0] >>> 3) & 0x3f;

                padLen = (index < 56) ? (56 - index) : (120 -
index);

                md5Update (PADDING, padLen);

                ///* Append length (before padding) */

                md5Update(bits, 8);
                ///* Store state in digest */

                Encode (digest, state, 16);



        }

        /* md5Memcpy是一个内部使用的byte数组的块拷贝函数，从input的
inpos开始把len长度的

          字节拷贝到output的outpos位置开始
        */

        private void md5Memcpy (byte[] output, byte[] input,

                int outpos, int inpos, int len)

        {

                int i;



                for (i = 0; i < len; i++)

                        output[outpos + i] = input[inpos + i];

        }

        /*
           md5Transform是MD5核心变换程序，有md5Update调用，block是分
块的原始字节
        */

        private void md5Transform (byte block[]) {

                long a = state[0], b = state[1], c = state[2],
d = state[3];

                long[] x = new long[16];
                Decode (x, block, 64);


                /* Round 1 */

                a = FF (a, b, c, d, x[0], S11, 0xd76aa478L); /*
1 */

                d = FF (d, a, b, c, x[1], S12, 0xe8c7b756L); /*
2 */

                c = FF (c, d, a, b, x[2], S13, 0x242070dbL); /*
3 */

                b = FF (b, c, d, a, x[3], S14, 0xc1bdceeeL); /*
4 */

                a = FF (a, b, c, d, x[4], S11, 0xf57c0fafL); /*
5 */

                d = FF (d, a, b, c, x[5], S12, 0x4787c62aL); /*
6 */

                c = FF (c, d, a, b, x[6], S13, 0xa8304613L); /*
7 */

                b = FF (b, c, d, a, x[7], S14, 0xfd469501L); /*
8 */

                a = FF (a, b, c, d, x[8], S11, 0x698098d8L); /*
9 */

                d = FF (d, a, b, c, x[9], S12, 0x8b44f7afL); /*
10 */

                c = FF (c, d, a, b, x[10], S13, 0xffff5bb1L);
/* 11 */

                b = FF (b, c, d, a, x[11], S14, 0x895cd7beL);
/* 12 */

                a = FF (a, b, c, d, x[12], S11, 0x6b901122L);
/* 13 */
```

```
                d = FF (d, a, b, c, x[13], S12, 0xfd987193L);
/* 14 */

                c = FF (c, d, a, b, x[14], S13, 0xa679438eL);
/* 15 */

                b = FF (b, c, d, a, x[15], S14, 0x49b40821L);
/* 16 */



                /* Round 2 */

                a = GG (a, b, c, d, x[1], S21, 0xf61e2562L); /*
17 */

                d = GG (d, a, b, c, x[6], S22, 0xc040b340L); /*
18 */

                c = GG (c, d, a, b, x[11], S23, 0x265e5a51L);
/* 19 */

                b = GG (b, c, d, a, x[0], S24, 0xe9b6c7aaL); /*
20 */

                a = GG (a, b, c, d, x[5], S21, 0xd62f105dL); /*
21 */

                d = GG (d, a, b, c, x[10], S22, 0x2441453L); /*
22 */

                c = GG (c, d, a, b, x[15], S23, 0xd8a1e681L);
/* 23 */

                b = GG (b, c, d, a, x[4], S24, 0xe7d3fbc8L); /*
24 */

                a = GG (a, b, c, d, x[9], S21, 0x21e1cde6L); /*
25 */

                d = GG (d, a, b, c, x[14], S22, 0xc33707d6L);
/* 26 */

                c = GG (c, d, a, b, x[3], S23, 0xf4d50d87L); /*
27 */
```

```
                b = GG (b, c, d, a, x[8], S24, 0x455a14edL); /*
28 */
                a = GG (a, b, c, d, x[13], S21, 0xa9e3e905L);
/* 29 */
                d = GG (d, a, b, c, x[2], S22, 0xfcefa3f8L); /*
30 */
                c = GG (c, d, a, b, x[7], S23, 0x676f02d9L); /*
31 */
                b = GG (b, c, d, a, x[12], S24, 0x8d2a4c8aL);
/* 32 */


                /* Round 3 */
                a = HH (a, b, c, d, x[5], S31, 0xfffa3942L); /*
33 */
                d = HH (d, a, b, c, x[8], S32, 0x8771f681L); /*
34 */
                c = HH (c, d, a, b, x[11], S33, 0x6d9d6122L);
/* 35 */
                b = HH (b, c, d, a, x[14], S34, 0xfde5380cL);
/* 36 */
                a = HH (a, b, c, d, x[1], S31, 0xa4beea44L); /*
37 */
                d = HH (d, a, b, c, x[4], S32, 0x4bdecfa9L); /*
38 */
                c = HH (c, d, a, b, x[7], S33, 0xf6bb4b60L); /*
39 */
                b = HH (b, c, d, a, x[10], S34, 0xbebfbc70L);
/* 40 */
                a = HH (a, b, c, d, x[13], S31, 0x289b7ec6L);
```

```
/* 41 */
                d = HH (d, a, b, c, x[0], S32, 0xeaa127faL); /*
42 */
                c = HH (c, d, a, b, x[3], S33, 0xd4ef3085L); /*
43 */
                b = HH (b, c, d, a, x[6], S34, 0x4881d05L); /*
44 */
                a = HH (a, b, c, d, x[9], S31, 0xd9d4d039L); /*
45 */
                d = HH (d, a, b, c, x[12], S32, 0xe6db99e5L);
/* 46 */
                c = HH (c, d, a, b, x[15], S33, 0x1fa27cf8L);
/* 47 */
                b = HH (b, c, d, a, x[2], S34, 0xc4ac5665L); /*
48 */


                /* Round 4 */
                a = II (a, b, c, d, x[0], S41, 0xf4292244L); /*
49 */
                d = II (d, a, b, c, x[7], S42, 0x432aff97L); /*
50 */
                c = II (c, d, a, b, x[14], S43, 0xab9423a7L);
/* 51 */
                b = II (b, c, d, a, x[5], S44, 0xfc93a039L); /*
52 */
                a = II (a, b, c, d, x[12], S41, 0x655b59c3L);
/* 53 */
                d = II (d, a, b, c, x[3], S42, 0x8f0ccc92L); /*
54 */
```

```
                c = II (c, d, a, b, x[10], S43, 0xffeff47dL);
/* 55 */

                b = II (b, c, d, a, x[1], S44, 0x85845dd1L); /*
56 */

                a = II (a, b, c, d, x[8], S41, 0x6fa87e4fL); /*
57 */

                d = II (d, a, b, c, x[15], S42, 0xfe2ce6e0L);
/* 58 */

                c = II (c, d, a, b, x[6], S43, 0xa3014314L); /*
59 */

                b = II (b, c, d, a, x[13], S44, 0x4e0811a1L);
/* 60 */

                a = II (a, b, c, d, x[4], S41, 0xf7537e82L); /*
61 */

                d = II (d, a, b, c, x[11], S42, 0xbd3af235L);
/* 62 */

                c = II (c, d, a, b, x[2], S43, 0x2ad7d2bbL); /*
63 */

                b = II (b, c, d, a, x[9], S44, 0xeb86d391L); /*
64 */


                state[0] += a;

                state[1] += b;

                state[2] += c;

                state[3] += d;


        }

        /*Encode把long数组按顺序拆成byte数组，因为java的long类型是
```

```
64bit的,
          只拆低32bit，以适应原始C实现的用途
        */
        private void Encode (byte[] output, long[] input, int
len) {
                int i, j;
                for (i = 0, j = 0; j < len; i++, j += 4) {
                        output[j] = (byte)(input[i] & 0xffL);
                        output[j + 1] = (byte)((input[i] >>> 8)
& 0xffL);
                        output[j + 2] = (byte)((input[i] >>>
16) & 0xffL);
                        output[j + 3] = (byte)((input[i] >>>
24) & 0xffL);
                }
        }


        /*Decode把byte数组按顺序合成成long数组，因为java的long类型是
64bit的,
          只合成低32bit，高32bit清零，以适应原始C实现的用途
        */
        private void Decode (long[] output, byte[] input, int
len) {
                int i, j;
                for (i = 0, j = 0; j < len; i++, j += 4)
                        output[i] = b2iu(input[j]) |
                                (b2iu(input[j + 1]) << 8) |

                                (b2iu(input[j + 2]) << 16) |

                                (b2iu(input[j + 3]) << 24);



                return;

        }

        /*
          b2iu是我写的一个把byte按照不考虑正负号的原则的＂升位＂程序，因
为java没有unsigned运算
        */

        public static long b2iu(byte b) {

                return b < 0 ? b & 0x7F + 128 : b;

        }

       /*byteHEX()，用来把一个byte类型的数转换成十六进制的ASCII表示，

         因为java中的byte的toString无法实现这一点，我们又没有C语言中的

         sprintf(outbuf,"%02X",ib)

       */

        public static String byteHEX(byte ib) {

                char[] Digit = {
'0','1','2','3','4','5','6','7','8','9',

                'A','B','C','D','E','F' };

                char [] ob = new char[2];

                ob[0] = Digit[(ib >>> 4) & 0X0F];

                ob[1] = Digit[ib & 0X0F];

                String s = new String(ob);
                return s;

        }



        public String getMD5String(String md5){
                      return getMD5ofStr(md5).toLowerCase();
        }

        public static void main(String args[]) {

                MD5 m = new MD5();

                if (Array.getLength(args) == 0) {   //如果没有参
数，执行标准的Test Suite



                    System.out.println("MD5 Test suite:");


System.out.println("MD5(\"\"):"+m.getMD5ofStr(""));


System.out.println("MD5(\"a\"):"+m.getMD5ofStr("a"));


System.out.println("MD5(\"abc\"):"+m.getMD5ofStr("abc"));

                       System.out.println("MD5(\"message
digest\"):"+m.getMD5ofStr("message digest"));


System.out.println("MD5(\"abcdefghijklmnopqrstuvwxyz\"):"+


m.getMD5ofStr("abcdefghijklmnopqrstuvwxyz"));


System.out.println("MD5(\"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijk
lmnopqrstuvwxyz0123456789\"):"+
m.getMD5ofStr("ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuv
wxyz0123456789"));

                }

                else

                          System.out.println("MD5(" + args[0] +
")=" + m.getMD5ofStr(args[0]));




        }



}
```

JDK 1.5

以下使用JDK 1.5.x版实现;

```


import java.security.*;

public class md5Test {

        private static String dumpBytes(byte[] bytes) {
            int i;
            StringBuffer sb = new StringBuffer();
            for (i = 0; i < bytes.length; i++) {
              if (i % 32 == 0 && i != 0) {
                sb.append("\n");
              }
              String s = Integer.toHexString(bytes[i]);
              if (s.length() < 2) {
                s = "0" + s;
              }
              if (s.length() > 2) {
                s = s.substring(s.length() - 2);
              }
              sb.append(s);
            }
            return sb.toString();
        }
        public static void main(String[] args) {

                String passwd = "netkiller";
            MessageDigest md = null;
                try {
                        md = MessageDigest.getInstance("MD5");
                        md.update("chen".getBytes());
                } catch (NoSuchAlgorithmException e) {
                        e.printStackTrace();
                }

            System.out.println(dumpBytes(md.digest()));
        }

}
```

编译运行,输入字符串:a1a8887793acfc199182a649e905daab

JDK 1.8

JDK 1.8

```
String hash = 
DatatypeConverter.printHexBinary(MessageDigest.getInstance("M
D5").digest("Helloword!!!".getBytes("UTF-8")));
```

1.5. perl md5

```
# Functional style
use Digest::MD5 qw(md5 md5_hex md5_base64);

$digest = md5($data);
$digest = md5_hex($data);
$digest = md5_base64($data);

# OO style
use Digest::MD5;

$ctx = Digest::MD5->new;

$ctx->add($data);
$ctx->addfile(*FILE);

$digest = $ctx->digest;
$digest = $ctx->hexdigest;
$digest = $ctx->b64digest;
```

2. SHA 专题

2.1. sha1sum

```
$ sha1sum /etc/passwd
c144c5cc8d5d3b90ad74a1dcf6af9e6c935e2a2a  /etc/passwd

$ sha1sum about/*
905a26de0f2fd6fcb53bf8db75d76c538d094237  about/index.html
d0aeb4409808b6afded0522964bed6b795d30fc0  about/index.tpl

$ sha1sum about/* > about.sha1
$ cat about.sha1
905a26de0f2fd6fcb53bf8db75d76c538d094237  about/index.html
d0aeb4409808b6afded0522964bed6b795d30fc0  about/index.tpl

$ sha1sum -c about.sha1
about/index.html: OK
about/index.tpl: OK
```

2.2. PHP sha1()

string sha1 (string str [, bool raw_output])

```
<?php
        $str = 'netkiller';
        echo sha1($str);
?>
```

运行输出字符串:eb673aa189c814d2db9fb71f162da1c81b4eba1c

2.3. Java SHA

SHA

```
import java.security.*;

public class shaTest {

        private static String dumpBytes(byte[] bytes) {
            int i;
            StringBuffer sb = new StringBuffer();
            for (i = 0; i < bytes.length; i++) {
              if (i % 32 == 0 && i != 0) {
                sb.append("\n");
              }
              String s = Integer.toHexString(bytes[i]);
              if (s.length() < 2) {
                s = "0" + s;
              }
              if (s.length() > 2) {
                s = s.substring(s.length() - 2);
              }
              sb.append(s);
            }
            return sb.toString();
        }
        public static void main(String[] args) {

                String passwd = "netkiller";
            MessageDigest md = null;
                try {
                        md = MessageDigest.getInstance("SHA");
                        md.update("chen".getBytes());
                } catch (NoSuchAlgorithmException e) {
                        e.printStackTrace();
                }

            System.out.println(dumpBytes(md.digest()));
        }

}
```

```

```

编译运行,输入字符串:8a89798cf0878e37bb6589ae1c36b9d8a036275b

SHA-256

```


package cn.netkiller.security;

import java.math.BigInteger;
import java.nio.charset.StandardCharsets;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;

public class SHA256 {

        public SHA256() {
                // TODO Auto-generated constructor stub
        }

        public static void main(String[] args) throws 
NoSuchAlgorithmException {
                // TODO Auto-generated method stub
                MessageDigest md = 
MessageDigest.getInstance("SHA-256");
                String text = "Text to hash, 
cryptographically.";

                // Change this to UTF-16 if needed
                
md.update(text.getBytes(StandardCharsets.UTF_8));
                byte[] digest = md.digest();

                String hex = String.format("%064x", new 
BigInteger(1, digest));
                System.out.println(hex);

        }

}
```

2.4. Perl

```
# Functional style
use Digest::SHA1  qw(sha1 sha1_hex sha1_base64);

$digest = sha1($data);
$digest = sha1_hex($data);
$digest = sha1_base64($data);

# OO style
use Digest::SHA1;

$ctx = Digest::SHA1->new;

$ctx->add($data);
$ctx->addfile(*FILE);

$digest = $ctx->digest;
$digest = $ctx->hexdigest;
$digest = $ctx->b64digest;
```

3. CRC32

CRC校验实用程序库在数据存储和数据通讯领域，为了保证数据的正确，就不得不采用检错的手段。在诸多检错手段中，CRC是最著名的一种。CRC的全称是循环冗余校验，其特点是:检错能力极强，开销小，易于用编码器及检测电路实现。从其检错能力来看，它所不能发现的错误的几率仅为0.0047%以下。从性能上和开销上考虑，均远远优于奇偶校验及算术和校验等方式。因而，在数据存储和数据通讯领域，CRC无处不在:著名的通讯协议X.25的FCS(帧检错序列)采用的是CRC- CCITT，ARJ、LHA等压缩工具软件采用的是CRC32，磁盘驱动器的读写采用了CRC16，通用的图像存储格式GIF、TIFF等也都用CRC作为检错手段。

3.1. PHP CRC32

```
<?php
$checksum = crc32("The quick brown fox jumped over the lazy
dog.");
printf("%u\n", $checksum);
?>
```

3.2. Java CRC32

```
package cn.netkiller.security;

import java.nio.ByteBuffer;
import java.util.zip.CRC32;

public class CRC {

        public static void main(String[] args) {
                final CRC32 crc32 = new CRC32();
                ByteBuffer data = 
ByteBuffer.wrap("http://www.netkiller.cn".getBytes());
                crc32.update(data);
                System.out.println(crc32.getValue());

        }

}
```

4. 第三方工具

4.1. htpasswd

```
$ sudo apt-get install apache2-utils
```

CRYPT

```
neo@master:~$ htpasswd -d -n neo.chen
New password:
Re-type new password:
neo.chen:Tyr60pyBFo0ng
```

MD5

```
neo@master:~$ htpasswd -m -n neo.chen
New password:
Re-type new password:
neo.chen:$apr1$CbZkN...$QzT7LwjRpQCKr4IkryM3Z.
```

SHA

```
neo@master:~$ htpasswd -s -n neo.chen
New password:
Re-type new password:
neo.chen:{SHA}iol5jPCHjje7ZYmuHDa52KA2J1s=
```

4.2. htdigest

htdigest 與 htpasswd 不同的地方在於對密碼的加密方式，htdigest 是使用 md5 來加密而 htpasswd 則是使用 crypt 來加密

```
htdigest -c /home/neo/trac/conf/passwd.digest localhost
netkiller
htdigest /home/neo/trac/conf/passwd.digest localhost neo
```

4.3. md5sum

```
$ md5sum /etc/passwd
325b7229c82c90c8a1823f5d939156bc  /etc/passwd
```

4.4. sha1sum

```
$ sha1sum /etc/passwd
f7a5582dd42ce0411bc2c59e2f1d8e89adcf0f81  /etc/passwd
```

第 4 章 DES crypt() 专题

提示

CRYPT_MD5 是Unix like Shadow密码

1. C crypt()

crypt是个密码加密函数，它是基於Data Encryption Standard(DES)演算法。

crypt基本上是One way encryption，因此它只适用於密码的使用，不适合於资料加密。

char *crypt(const char *key, const char *salt);

key 是使用者的密码。salt是两个字，每个字可从[a-zA-Z0-9./]中选出来，因此同一密码增加了4096种可能性。透过使用key中每个字的低七位元，取得 56-bit关键字，这56-bit关键字被用来加密成一组字，这组字有13个可显示的 ASCII字，包含开头两个salt。

```
[root@linux root]# cat crypt.c
/*
Netkiller 2003-06-27 crypt.c
char *crypt(const char *key, const char *salt);
*/
#include <unistd.h>
main(){
    char key[256];
    char salt[64];
    char passwd[256];
    printf("key:");
    scanf("%s",&key);
    printf("salt:");
    scanf("%s",&salt);
    sprintf(passwd,"passwd:%s\n",crypt(key,salt));
    printf(passwd);
}
[root@linux root]# gcc -o crypt -s crypt.c –lcrypt
[root@linux root]# ./crypt
key:chen
salt:salt
passwd:sa0hRW/W3DLvQ
[root@linux root]#
```

2. PHP crypt()

将字符串用 DES 编码加密。

语法: string crypt(string str, string [salt]);

返回值: 字符串

函数种类: 编码处理

内容说明

本函数将字符串用 UNIX 的标准加密 DES 模块加密。这是单向的加密函数，无法解密。欲比对字符串，将已加密的字符串的头二个字符放在 salt 的参数中，再比对加密后的字符串。

更详细的资料请参考 UNIX Manual (man) 中的 crypt。

在一些较新的 UNIX 版本中，除了 DES 之外还提供了其它的加密模块，如 MD5。甚至有些系统还用 MD5 取代 DES。在 salt 参数还有一些变化，端看传给 salt 参数的字符串长度而定:

* CRYPT_STD_DES - 标准的 DES 编码，输入 2 字符的 salt。
* CRYPT_EXT_DES - 延伸的 DES 编码，输入 9 字符的 salt。
* CRYPT_MD5 - MD5 编码，输入 12 字符加上 $1$ 的 salt。
* CRYPT_BLOWFISH - 延伸的 DES 编码，输入 16 字符加上 $2$ 的 salt。

此外，若不使用 salt 参数，则程序会自动产生。

```
# cat crypt.php

<html>
```

```
<p>DES 密码</p>

<form method=post action=crypt.php>

<p>password:<input name=passwd type=text size=20></p>

<input type=submit value=submit>

</form>

<?

$enpw=crypt($passwd);

echo "password is: $enpw";

?>



[root@linux root]# wget
http://netkiller.hikz.com/linux/download/myphp/site-
2.1.0.tar.gz
[root@linux root]#tar zxvf site-2.1.0.tar.gz
[root@linux root]#cp –r site /usr/local/apache/htdocs
[root@linux root]#lynx http://localhost/site
```

3. perl crypt

```
perl -e 'print("userPassword: ".crypt("secret","salt")."\n");'
```

4. mysql crypt

```
select encrypt('password');


mysql> select encrypt('password');
+---------------------+
| encrypt('password') |
+---------------------+
| WXvvG0CWY7v5I       |
+---------------------+
1 row in set (0.00 sec)


mysql>
```

5. Java crypt

第一种方法:

Crypt.java

Import netkiller. Security;

Crypt pw = new Crypt();

String passwd = pw.crypt(“passwd”,”salt”);

System.out.println(passwd);

关于JAVA的Crypt包请与我联系

第二种方法:

使用PostgreSQL JDBC中提供的org.postgresql.util.UnixCrypt产生crypt。

Class   postgresql.util.UnixCrypt

java.lang.Object

|

+----postgresql.util.UnixCrypt

公共类　UnixCrypt　扩展　Object

这个类为我们提供了在通过网络流传输口令时的加密的功能

包含静态方法用于加密口令和与　Unix　加密的口令比较.

参阅　John　Dumas　的　Java　Crypt　(加密)页面获取原始代码.

http://www.zeh.com/local/jfd/crypt.html

方法

public　static　final　String　crypt(String　salt, String　original)

加密给出了明文口令和一个"种子"("salt")的口令.

参数:

salt　-　一个两字符字串代表的所用的种子，　用以向加密引擎说明加密的不同方式. 如果你要生成一个新的密文那么这个值应该是随机生成的.

original　-　待加密口令.

返回:

一个字串，先是　2　字符的种子，然后跟着密文口令.

方法:

1.　安装PostgreSQL JDBC，请到http://www.postgresql.org 下载

2.　　将JDBC的.jar文件加到JAVA 的CLASSPATH中

3.　　新建JAVA文件。

4.　　编译javac crypt.java

5.　　运行JAVA CLASS文件 java your-package.your-class
java crypt

```
import org.postgresql.util.UnixCrypt;

import java.io.InputStreamReader;

import java.io.BufferedReader;

import java.io.IOException;



public class crypt {

   public static void main(String[] args) throws IOException {

      String password;

      BufferedReader br=new BufferedReader(new InputStreamReader(Syst
em.in));

      System.out.println("Enter the password to encrypt. Your password"+

                    " will be echoed on the screen,");

      System.out.println("please ensure nobody is looking.");
```

```
        System.out.print("password :>");

        password=br.readLine();

        System.out.println(UnixCrypt.crypt(password));

    };

};
```

5.1. Java 8 DES

```


package cn.netkiller.security;

import java.nio.charset.StandardCharsets;
import java.security.SecureRandom;
import java.util.Base64;

import javax.crypto.Cipher;
import javax.crypto.SecretKey;
import javax.crypto.SecretKeyFactory;
import javax.crypto.spec.DESKeySpec;

public class DES {

        public DES() {
                // TODO Auto-generated constructor stub
        }

        public static String encrypt(String text, String
password) {
                try {
                        SecureRandom random = new
SecureRandom();
                        DESKeySpec desKey = new
DESKeySpec(password.getBytes());
                        // 创建一个密匙工厂，然后用它把DESKeySpec
转换成
```

```
                        SecretKeyFactory keyFactory =
SecretKeyFactory.getInstance("DES");
                        SecretKey securekey =
keyFactory.generateSecret(desKey);
                        // Cipher对象实际完成加密操作
                        Cipher cipher =
Cipher.getInstance("DES");
                        // 用密匙初始化Cipher对象
                        cipher.init(Cipher.ENCRYPT_MODE,
securekey, random);
                        // 现在，获取数据并加密
                        // 正式执行加密操作

                        return
Base64.getEncoder().encodeToString(cipher.doFinal(text.getByt
es(StandardCharsets.UTF_8)));
                } catch (Throwable e) {
                        e.printStackTrace();
                }
                return null;
        }

        private static String decrypt(String text, String
password) throws Exception {
                try {
                        // DES算法要求有一个可信任的随机数源
                        SecureRandom random = new
SecureRandom();
                        // 创建一个DESKeySpec对象
                        DESKeySpec desKey = new
DESKeySpec(password.getBytes());
                        // 创建一个密匙工厂
                        SecretKeyFactory keyFactory =
SecretKeyFactory.getInstance("DES");
                        // 将DESKeySpec对象转换成SecretKey对象
                        SecretKey securekey =
keyFactory.generateSecret(desKey);
                        // Cipher对象实际完成解密操作
                        Cipher cipher =
Cipher.getInstance("DES");
                        // 用密匙初始化Cipher对象
                        cipher.init(Cipher.DECRYPT_MODE,
securekey, random);
                        // 真正开始解密操作
                        return new
```

```
String(cipher.doFinal(Base64.getDecoder().decode(text)),
StandardCharsets.UTF_8);
                } catch (Exception e) {
                        e.printStackTrace();
                }
                return null;
        }

        public static void main(String[] args) throws
Exception {
                // TODO Auto-generated method stub
                String en = DES.encrypt("Helloworld!!!",
"www.netkiller.cn");
                String de = DES.decrypt(en,
"www.netkiller.cn");
                System.out.println(en);
                System.out.println(de);

        }

}
```

6. grub-md5-crypt - Encrypt a password in MD5 format.

```
# grub-md5-crypt
Password:
Retype password:
$1$ZlJ1u0$tdv/dr8pYuHh.eT47F6b70
```

第 5 章 AES

1. Java

1.1. AES/ECB/PKCS5Padding

```
package cn.netkiller.crypto;

import javax.crypto.Cipher;
import javax.crypto.spec.IvParameterSpec;
import javax.crypto.spec.SecretKeySpec;
import java.security.MessageDigest;
import java.security.SecureRandom;

public class TestAES {

        public static void main(String[] args) {
                // TODO Auto-generated method stub
                String key = 
"fm6I1D2HTFVVOWUKny76TThagNq5Czrv";
                String clean = "Helloworld!!!";

                try {
                        byte[] encrypted = encrypt(clean, key);
                        String decrypted = decrypt(encrypted, 
key);
                        System.out.println(decrypted);
                } catch (Exception e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }

        }

        public static byte[] encrypt(String plainText, String 
key) throws Exception {
                byte[] clean = plainText.getBytes();
                // Generating IV.
                int ivSize = 16;
                byte[] iv = new byte[ivSize];
                SecureRandom random = new SecureRandom();
                random.nextBytes(iv);
                IvParameterSpec ivParameterSpec = new
IvParameterSpec(iv);

                // Hashing key.
                MessageDigest digest =
MessageDigest.getInstance("SHA-256");
                digest.update(key.getBytes("UTF-8"));
                byte[] keyBytes = new byte[16];
                System.arraycopy(digest.digest(), 0, keyBytes,
0, keyBytes.length);
                SecretKeySpec secretKeySpec = new
SecretKeySpec(keyBytes, "AES");

                // Encrypt.
                Cipher cipher =
Cipher.getInstance("AES/CBC/PKCS5Padding");
                cipher.init(Cipher.ENCRYPT_MODE, secretKeySpec,
ivParameterSpec);
                byte[] encrypted = cipher.doFinal(clean);

                // Combine IV and encrypted part.
                byte[] encryptedIVAndText = new byte[ivSize +
encrypted.length];
                System.arraycopy(iv, 0, encryptedIVAndText, 0,
ivSize);
                System.arraycopy(encrypted, 0,
encryptedIVAndText, ivSize, encrypted.length);

                return encryptedIVAndText;
        }

        public static String decrypt(byte[]
encryptedIvTextBytes, String key) throws Exception {
                int ivSize = 16;
                int keySize = 16;

                // Extract IV.
                byte[] iv = new byte[ivSize];
                System.arraycopy(encryptedIvTextBytes, 0, iv,
0, iv.length);
```

```
                IvParameterSpec ivParameterSpec = new
IvParameterSpec(iv);

                // Extract encrypted part.
                int encryptedSize = encryptedIvTextBytes.length
- ivSize;
                byte[] encryptedBytes = new
byte[encryptedSize];
                System.arraycopy(encryptedIvTextBytes, ivSize,
encryptedBytes, 0, encryptedSize);

                // Hash key.
                byte[] keyBytes = new byte[keySize];
                MessageDigest md =
MessageDigest.getInstance("SHA-256");
                md.update(key.getBytes());
                System.arraycopy(md.digest(), 0, keyBytes, 0,
keyBytes.length);
                SecretKeySpec secretKeySpec = new
SecretKeySpec(keyBytes, "AES");

                // Decrypt.
                Cipher cipherDecrypt =
Cipher.getInstance("AES/CBC/PKCS5Padding");
                cipherDecrypt.init(Cipher.DECRYPT_MODE,
secretKeySpec, ivParameterSpec);
                byte[] decrypted =
cipherDecrypt.doFinal(encryptedBytes);

                return new String(decrypted);
        }

}
```

上面是 byte 类型使用中不是很方便，尤其是WEB中作为参数传递的情况，所以我们使用 BASE64编码

```


package cn.netkiller.crypto;

import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;
import java.util.Base64.*;

/**
 * @author netkiller
 *
 */
public class aes {

	public static String encrypt(String input, String key)
{
		byte[] crypted = null;
		try {

			SecretKeySpec skey = new
SecretKeySpec(key.getBytes(), "AES");

			Cipher cipher =
Cipher.getInstance("AES/ECB/PKCS5Padding");
			cipher.init(Cipher.ENCRYPT_MODE, skey);
			crypted =
cipher.doFinal(input.getBytes());
		} catch (Exception e) {
			System.out.println(e.toString());
		}
		java.util.Base64.Encoder encoder =
java.util.Base64.getEncoder();

		return new
String(encoder.encodeToString(crypted));
	}

	public static String decrypt(String input, String key)
{
		byte[] output = null;
		try {
			java.util.Base64.Decoder decoder =
java.util.Base64.getDecoder();
			SecretKeySpec skey = new
SecretKeySpec(key.getBytes(), "AES");
			Cipher cipher =
Cipher.getInstance("AES/ECB/PKCS5Padding");
			cipher.init(Cipher.DECRYPT_MODE, skey);
			output =
cipher.doFinal(decoder.decode(input));
```

```
                } catch (Exception e) {
                        System.out.println(e.toString());
                }
                return new String(output);
        }

        /**
         * @param args
         */
        public static void main(String[] args) {
                // TODO Auto-generated method stub

                String key = "mvLBiZsiTbGwrfJB";
                String data = "ABC";

                System.out.println(aes.encrypt(data, key));

System.out.println(aes.decrypt(aes.encrypt(data, key), key));
        }

}
```

1.2. AES/CBC/PKCS5PADDING

```
package cn.netkiller.security;

import java.util.Base64;

import javax.crypto.Cipher;
import javax.crypto.spec.IvParameterSpec;
import javax.crypto.spec.SecretKeySpec;

public class AES {
        private static final String initVector =
"encryptionIntVec";
        private String key;

        public AES(String key) {
                // TODO Auto-generated constructor stub
                this.key = key;
        }

        public String encrypt(String value) {
                return this.encrypt(value, this.key);
        }

        public String encrypt(String value, String key) {
                try {
                        IvParameterSpec ivParameterSpec = new
IvParameterSpec(initVector.getBytes("UTF-8"));
                        SecretKeySpec secretKeySpec = new
SecretKeySpec(key.getBytes("UTF-8"), "AES");

                        Cipher cipher =
Cipher.getInstance("AES/CBC/PKCS5PADDING");
                        cipher.init(Cipher.ENCRYPT_MODE,
secretKeySpec, ivParameterSpec);

                        byte[] encrypted =
cipher.doFinal(value.getBytes());
                        return
Base64.getEncoder().encodeToString(encrypted);
                } catch (Exception ex) {
                        ex.printStackTrace();
                }
                return null;
        }

        public String decrypt(String encrypted) {
                return this.decrypt(encrypted, this.key);
        }

        public String decrypt(String encrypted, String key) {
                try {
                        IvParameterSpec ivParameterSpec = new
IvParameterSpec(initVector.getBytes("UTF-8"));
                        SecretKeySpec secretKeySpec = new
SecretKeySpec(key.getBytes("UTF-8"), "AES");

                        Cipher cipher =
Cipher.getInstance("AES/CBC/PKCS5PADDING");
                        cipher.init(Cipher.DECRYPT_MODE,
secretKeySpec, ivParameterSpec);
                        byte[] original =
cipher.doFinal(Base64.getDecoder().decode(encrypted));
```

```
				return new String(original);
			} catch (Exception ex) {
				ex.printStackTrace();
			}

			return null;
		}

		public static void main(String[] args) {
			// key 长度16个字节
			String key = "www.netkiller.cn";
			System.out.println(key.length());
			AES aes = new AES(key);
			String en = aes.encrypt("Helloworld!!!");
			String de = aes.decrypt(en);
			System.out.println(en);
			System.out.println(de);

		}
}
```

2. PHP

2.1. AES/ECB/PKCS5Padding

```
<?php class CryptAES { protected $cipher = MCRYPT_RIJNDAEL_128;
protected $mode = MCRYPT_MODE_ECB; protected $pad_method =
NULL; protected $secret_key = ''; protected $iv = ''; public function
set_cipher($cipher) { $this->cipher = $cipher; } public function
set_mode($mode) { $this->mode = $mode; } public function set_iv($iv) {
$this->iv = $iv; } public function set_key($key) { $this->secret_key =
$key; } public function require_pkcs5() { $this->pad_method = 'pkcs5'; }
protected function pad_or_unpad($str, $ext) { if ( is_null($this-
>pad_method) ) { return $str; } else { $func_name = __CLASS__ . '::' .
$this->pad_method . '_' . $ext . 'pad'; if ( is_callable($func_name) ) { $size
= mcrypt_get_block_size($this->cipher, $this->mode); return
call_user_func($func_name, $str, $size); } } return $str; } protected
function pad($str) { return $this->pad_or_unpad($str, ''); } protected
function unpad($str) { return $this->pad_or_unpad($str, 'un'); } public
function encrypt($str) { $str = $this->pad($str); $td =
mcrypt_module_open($this->cipher, '', $this->mode, ''); if ( empty($this-
>iv) ) { $iv = @mcrypt_create_iv(mcrypt_enc_get_iv_size($td),
MCRYPT_RAND); } else { $iv = $this->iv; } mcrypt_generic_init($td,
$this->secret_key, $iv); $cyper_text = mcrypt_generic($td, $str); //$rt =
bin2hex($cyper_text); $rt = base64_encode($cyper_text);
mcrypt_generic_deinit($td); mcrypt_module_close($td); return $rt; } public
function decrypt($str){ $td = mcrypt_module_open($this->cipher, '', $this-
>mode, ''); if ( empty($this->iv) ) { $iv =
@mcrypt_create_iv(mcrypt_enc_get_iv_size($td), MCRYPT_RAND); }
else { $iv = $this->iv; } mcrypt_generic_init($td, $this->secret_key, $iv);
//$decrypted_text = mdecrypt_generic($td, self::hex2bin($str));
$decrypted_text = mdecrypt_generic($td, base64_decode($str)); $rt =
$decrypted_text; mcrypt_generic_deinit($td); mcrypt_module_close($td);
return $this->unpad($rt); } public static function hex2bin($hexdata) {
$bindata = ''; $length = strlen($hexdata); for ($i=0; $i < $length; $i += 2) {
```

```
$bindata .= chr(hexdec(substr($hexdata, $i, 2))); } return $bindata; } public
static function pkcs5_pad($text, $blocksize) { $pad = $blocksize -
(strlen($text) % $blocksize); return $text . str_repeat(chr($pad), $pad); }
public static function pkcs5_unpad($text) { $pad = ord($text{strlen($text) -
1}); if ($pad > strlen($text)) return false; if (strspn($text, chr($pad),
strlen($text) - $pad) != $pad) return false; return substr($text, 0, -1 * $pad);
} } $aes = new CryptAES(); $aes->set_key('NGjPs5cgNS497sdx'); $aes-
>require_pkcs5(); $rt = $aes->encrypt('ABC'); echo $rt . '<br/>'; echo $aes-
>decrypt($rt) . '<br/>';
```

第 6 章 GnuPG

1. 安装 GnuPG

http://www.gnupg.org

```
[root@netkiller ~]# dnf install -y gnupg
```

2. 创建密钥

```
[root@netkiller ~]# gpg --gen-key
gpg (GnuPG) 2.2.20; Copyright (C) 2020 Free Software Foundation,
Inc.
This is free software: you are free to change and redistribute
it.
There is NO WARRANTY, to the extent permitted by law.

Note: Use "gpg --full-generate-key" for a full featured key
generation dialog.

GnuPG needs to construct a user ID to identify your key.

Real name: Neo Chen
Email address: netkiller@msn.com
You selected this USER-ID:
    "Neo Chen <netkiller@msn.com>"

Change (N)ame, (E)mail, or (O)kay/(Q)uit? O
We need to generate a lot of random bytes. It is a good idea to
perform
some other action (type on the keyboard, move the mouse, utilize
the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
We need to generate a lot of random bytes. It is a good idea to
perform
some other action (type on the keyboard, move the mouse, utilize
the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
gpg: /root/.gnupg/trustdb.gpg: trustdb created
gpg: key F01C0CAEAAA458E6 marked as ultimately trusted
gpg: directory '/root/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/root/.gnupg/openpgp-
revocs.d/70CECE32E5D67D12B95ED1E7F01C0CAEAAA458E6.rev'
public and secret key created and signed.

pub   rsa2048 2021-10-08 [SC] [expires: 2023-10-08]
      70CECE32E5D67D12B95ED1E7F01C0CAEAAA458E6
```

```
uid                      Neo Chen <netkiller@msn.com>
sub   rsa2048 2021-10-08 [E] [expires: 2023-10-08]
```

```
[root@netkiller ~]# gpg --full-generate-key
gpg (GnuPG) 2.2.20; Copyright (C) 2020 Free Software Foundation,
Inc.
This is free software: you are free to change and redistribute
it.
There is NO WARRANTY, to the extent permitted by law.

Please select what kind of key you want:
   (1) RSA and RSA (default)
   (2) DSA and Elgamal
   (3) DSA (sign only)
   (4) RSA (sign only)
  (14) Existing key from card
Your selection?
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048)
Requested keysize is 2048 bits
Please specify how long the key should be valid.
         0 = key does not expire
      <n>  = key expires in n days
      <n>w = key expires in n weeks
      <n>m = key expires in n months
      <n>y = key expires in n years
Key is valid for? (0)
Key does not expire at all
Is this correct? (y/N) y

GnuPG needs to construct a user ID to identify your key.

Real name: Netkiller
Email address: netkiller@msn.com
Comment: Test Key
You selected this USER-ID:
    "Netkiller (Test Key) <netkiller@msn.com>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? O
We need to generate a lot of random bytes. It is a good idea to
perform
```

```
some other action (type on the keyboard, move the mouse, utilize
the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
We need to generate a lot of random bytes. It is a good idea to
perform
some other action (type on the keyboard, move the mouse, utilize
the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
gpg: key 5E27578A03F0B510 marked as ultimately trusted
gpg: revocation certificate stored as '/root/.gnupg/openpgp-
revocs.d/E1C21F034FC0ACBF1337EE905E27578A03F0B510.rev'
public and secret key created and signed.

pub   rsa2048 2021-10-08 [SC]
      E1C21F034FC0ACBF1337EE905E27578A03F0B510
uid                      Netkiller (Test Key)
<netkiller@msn.com>
sub   rsa2048 2021-10-08 [E]
```

3. 查看密钥

```
[root@netkiller ~]# gpg --list-keys
gpg: checking the trustdb
gpg: marginals needed: 3  completes needed: 1  trust model: pgp
gpg: depth: 0  valid:   2  signed:   0  trust: 0-, 0q, 0n, 0m,
0f, 2u
gpg: next trustdb check due at 2023-10-08
/root/.gnupg/pubring.kbx
------------------------
pub   rsa2048 2021-10-08 [SC] [expires: 2023-10-08]
      70CECE32E5D67D12B95ED1E7F01C0CAEAAA458E6
uid           [ultimate] Neo Chen <netkiller@msn.com>
sub   rsa2048 2021-10-08 [E] [expires: 2023-10-08]

pub   rsa2048 2021-10-08 [SC]
      E1C21F034FC0ACBF1337EE905E27578A03F0B510
uid           [ultimate] Netkiller (Test Key)
<netkiller@msn.com>
sub   rsa2048 2021-10-08 [E]
```

查看私钥

```
[root@netkiller ~]# gpg --list-secret-keys
/root/.gnupg/pubring.kbx
------------------------
sec   rsa2048 2021-10-08 [SC] [expires: 2023-10-08]
      70CECE32E5D67D12B95ED1E7F01C0CAEAAA458E6
uid           [ultimate] Neo Chen <netkiller@msn.com>
ssb   rsa2048 2021-10-08 [E] [expires: 2023-10-08]
```

SC（"sign","certify"，代表可以签名和认证其它密钥）

E（"encrypt"，加密）

S（"sign"，签名）

```
sec => 'SECret key'
ssb => 'Secret SuBkey'
pub => 'PUBlic key'
sub => 'public SUBkey'
```

格式化

```
[root@gitlab ~]# gpg --list-secret-keys --keyid-format LONG
/root/.gnupg/pubring.kbx
------------------------
sec   rsa2048/F01C0CAEAAA458E6 2021-10-08 [SC] [expires: 2023-
10-08]
      70CECE32E5D67D12B95ED1E7F01C0CAEAAA458E6
uid                 [ultimate] Neo Chen <netkiller@msn.com>
ssb   rsa2048/EAA2F7FD813D2A2E 2021-10-08 [E] [expires: 2023-10-
08]

sec   rsa2048/4113D1268C351687 2021-10-09 [SC] [expires: 2023-
10-09]
      DBF998A60B206C9297ABC57A4113D1268C351687
uid                 [ultimate] Tests <test@test.com>
ssb   rsa2048/EA6FAF428416D577 2021-10-09 [E] [expires: 2023-10-
09]

sec   rsa2048/0C835D03507C8536 2021-10-09 [SC] [expires: 2023-
10-09]
      18235CBA04497C42EFAC78210C835D03507C8536
uid                 [ultimate] Backup <backup@netkiller.cn>
ssb   rsa2048/339634D92F842BE7 2021-10-09 [E] [expires: 2023-10-
09]
```

4. 吊销密钥

```
[root@netkiller ~]# gpg --gen-revoke
E1C21F034FC0ACBF1337EE905E27578A03F0B510

sec  rsa2048/5E27578A03F0B510 2021-10-08 Netkiller (Test Key)
<netkiller@msn.com>

Create a revocation certificate for this key? (y/N) y
Please select the reason for the revocation:
  0 = No reason specified
  1 = Key has been compromised
  2 = Key is superseded
  3 = Key is no longer used
  Q = Cancel
(Probably you want to select 1 here)
Your decision? 0
Enter an optional description; end it with an empty line:
> revoke key
>
Reason for revocation: No reason specified
revoke key
Is this okay? (y/N) y
ASCII armored output forced.
-----BEGIN PGP PUBLIC KEY BLOCK-----
Comment: This is a revocation certificate

iQFABCABCAAqFiEE4cIfA0/ArL8TN+6QXidXigPwtRAFAmFgHfAMHQByZXZva2U
g
a2V5AAoJEF4nV4oD8LUQHTQH/iZdX2mAq2jOT/QwvKxMT6ePLJb1yQ+2aFUkvV2
m
tgGfWh7k94rKpukXBk3Ay9HgOQRx450SFqo7dA9sGZFoVGxAd8iC2c0ofwpdm4W
5
UWy6eeiQI2Huq+HuvEYebuz/ZLDsKMq53ZFTfu8GndTQfLcXvu/jk7ACzPgtvyV
8
4eIqg9Lnlbs6GDomMmcaLlG2kF1lHCYeFgxJlMCJgpwJAqDetAwB/6q7xFPGhfb
t
mLeR7dwCffoVivdBgiFVjlSDL8PJX0fDvsm0uY7gZmGyMzEQUbrBD2s9QIqtDKa
K
KamXegbf3pM1EIWdgK9xtbQV2SRYAJmUwaKz6Sfab623jYc=
=pW/X
```

```
-----END PGP PUBLIC KEY BLOCK-----
Revocation certificate created.

Please move it to a medium which you can hide away; if Mallory
gets
access to this certificate he can use it to make your key
unusable.
It is smart to print this certificate and store it away, just
in case
your media become unreadable.  But have some caution:  The
print system of
your machine might store the data and make it available to
others!
```

5. 删除密钥

首先，删除私钥

```
[root@netkiller ~]# gpg --delete-secret-keys
E1C21F034FC0ACBF1337EE905E27578A03F0B510
gpg (GnuPG) 2.2.20; Copyright (C) 2020 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.


sec  rsa2048/5E27578A03F0B510 2021-10-08 Netkiller (Test Key)
<netkiller@msn.com>

Delete this key from the keyring? (y/N) y
This is a secret key! - really delete? (y/N) y
```

然后，删除公钥

```
[root@netkiller ~]# gpg --delete-key
E1C21F034FC0ACBF1337EE905E27578A03F0B510
gpg (GnuPG) 2.2.20; Copyright (C) 2020 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.


pub  rsa2048/5E27578A03F0B510 2021-10-08 Netkiller (Test Key)
<netkiller@msn.com>

Delete this key from the keyring? (y/N) y
```

最后，查看密钥，并确认删除

```
[root@netkiller ~]# gpg --list-keys
gpg: checking the trustdb
gpg: marginals needed: 3  completes needed: 1  trust model: pgp
gpg: depth: 0  valid:   1  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 1u
gpg: next trustdb check due at 2023-10-08
```

```
/root/.gnupg/pubring.kbx
------------------------
pub   rsa2048 2021-10-08 [SC] [expires: 2023-10-08]
      70CECE32E5D67D12B95ED1E7F01C0CAEAAA458E6
uid           [ultimate] Neo Chen <netkiller@msn.com>
sub   rsa2048 2021-10-08 [E] [expires: 2023-10-08]
```

6. 密钥倒入/导出

6.1. 导出密钥

导出所有公钥

--export export keys

```
[root@netkiller ~]# gpg --export -a
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQENBGFgEfYBCACXIT6K61G3uwwFPxwKaKirZyhSnhh22CwTPEGkeviyXCCfpr2X
d8bjibOCwO8bigXFjaKuTikHmpppy7B/CKJ4OlsLXnoMnnSmynntudJ+jcGmC3/0
QE1nvDzqbe8L5KJ3TMgAuDUSp3QWXqIAXxQfEABLl49wJ11envwTXJVPG/ks2U3m
b/QAFZqd3AxUpEzASIKbtiB5JE/rxnhyZH7fHkt3vU2N3qAcUQ67cJN+thkMEsOo
wnp9eGvDv1qBieQKK5DzxC+a04p4cWv5z0rV4IEE3bRR2wKW45HI9Lmgz8zZyFcO
gTV1HshRYnDBVgzcnyombQfzbd76g5tBQC2vABEBAAG0HE5lbyBDaGVuIDxuZXRr
aWxsZXJAbXNuLmNvbT6JAVQEEwEIAD4WIQRwzs4y5dZ9Erle0efwHAyuqqRY5gUC
YWAR9gIbAwUJA8JnAAULCQgHAgYVCgkICwIEFgIDAQIeAQIXgAAKCRDwHAyuqqRY
5v8UB/9GuKFO86BprJUfPBOE4sqUPH44kLupVMuvM+XaBkuOQIT5q37MPoUpb3Uj
g7tV3Nc+6/VLTCDTERKEfV7PRke2UjjjdYf4EYA2PMVVtHEnWngKhVcMkD2iEvR2
ViCQQ6sCve5lefMQcPyLVMX1ynMOQCNiVcOZjfv+vW2H4BynZC6kG472a3TjoTKz
TlbrsiK/n7CSMLsevQh9UrG2n24rKfxQiWCo9tVxyWjcYLEO6yRzOxC+KnEBVr30
O86qn8A/soKY3PEWWUWCcve9g7Km3OVMQf3kJo+xy3hDafDhuBTvNUH3Bz9lwXa3
Sune2h5J77AbgUCHZSw4MZEWdknxuQENBGFgEfYBCACVjr3QGs1b2cei5sHyBO59
hC8VgehGs42jiItaNQSLpBO8g8Z2UbwcB9y3QWrbBITXfj1Jmy+XJInbc3FYYoZE
9bVHb+KjIR4JLqWrieGCWaKzl78ByRRKfQWO0di5OVMQBWg3yzd2dRJnvpa8+W60
ksHoyL0wcXLDbCxYxTNmpHacbvEJYe4zxYJxMyD3V8BEF/r6HtA8ZrhPHrI23AF6
iqSK7PIKAFBLIbU9jinncy/Vbv1DgXZrh72cxhl0n7hTgX8tI2gFRpz+p10iKX2B
zab3F4Ac1YNBy/F9tqIeCPBGK4CmFTtZkzpokevrIfzLThWuqRGIRtnwqlvMKHxz
ABEBAAGJATwEGAEIACYWIQRwzs4y5dZ9Erle0efwHAyuqqRY5gUCYWAR9gIbDAUJ
A8JnAAAKCRDwHAyuqqRY5hpyB/4hh3qMpSOtjOFS5nWGrYNb/o//YRKDwORjJUdI
t0A1RvQkIZEQ9MYR67xpQ8OO2JrsznB7yF0D/Wrmleuu9lY9IVgdaNdNYRRzAdam
MuU5hYe6cUkNudjekhWb2J77EIaL70g9tboEHlQEdVe/FesLg1iZVlPZaaN6UjN6
81AcVw3nloBgIHQUWWsdsSW5sTfymnMhtUfJVlPfeEagLIioTvTzUqy0LjjeIOhR
B1EXkjs/4g/20c/X9JH8z+QwnZ0lmHy9HzUl+g3zLQ7Vu2xaTwHgBWl5sGdkDkJX
RiSdzxKOlGfxNN0e5r7fUYvlCkqOvAFvdpZANcVYkWurjWt2
=W+8i
-----END PGP PUBLIC KEY BLOCK-----
```

导出公钥到指定文件

-o, --output use as output file

```
$ gpg --export -a -o test.asc
$ gpg --output yourname.asc --export -a
```

```
[root@netkiller ~]# gpg --list-keys
/root/.gnupg/pubring.kbx
------------------------
pub   rsa2048 2021-10-08 [SC] [expires: 2023-10-08]
      70CECE32E5D67D12B95ED1E7F01C0CAEAAA458E6
uid           [ultimate] Neo Chen <netkiller@msn.com>
sub   rsa2048 2021-10-08 [E] [expires: 2023-10-08]

[root@netkiller ~]# gpg --output neo.gpg --export
70CECE32E5D67D12B95ED1E7F01C0CAEAAA458E6

[root@netkiller ~]# ls neo.gpg
neo.gpg
```

导出私钥

```
gpg --armor --output private-key.gpg --export-secret-keys
```

6.2. 导入密钥

--import import/merge keys

导入公钥

```
[root@testing ~]# gpg --import /home/www/backup.gpg
gpg: /root/.gnupg/trustdb.gpg: trustdb created
gpg: key 0C835D03507C8536: public key "Backup <backup@netkiller.cn>" imported
gpg: Total number processed: 1
gpg:               imported: 1
```

查看公钥

```
[root@testing ~]# gpg -k
/root/.gnupg/pubring.kbx
```

```
------------------------
pub   rsa2048 2021-10-09 [SC] [expires: 2023-10-09]
      18235CBA04497C42EFAC78210C835D03507C8536
uid           [ unknown] Backup <backup@netkiller.cn>
sub   rsa2048 2021-10-09 [E] [expires: 2023-10-09]
```

6.3. 密钥迁移

从一台机器，迁移到另一台机器

原电脑

```
[root@gitlab ~]# gpg --list-keys
/root/.gnupg/pubring.kbx
------------------------
pub   rsa2048 2021-10-08 [SC] [expires: 2023-10-08]
      70CECE32E5D67D12B95ED1E7F01C0CAEAAA458E6
uid           [ultimate] Neo Chen <netkiller@msn.com>
sub   rsa2048 2021-10-08 [E] [expires: 2023-10-08]

[root@gitlab ~]# gpg --armor --export-secret-keys --output private_key.asc
netkiller@msn.com
[root@gitlab ~]# gpg --armor --export --output public_key.asc netkiller@msn.com
[root@gitlab ~]# scp private_key.asc public_key.asc root@other:/home/gitlab-
runner/
```

目标电脑或另一个账号

```
[root@localhost ~]# gpg --import public_key.asc
gpg: /root/.gnupg/trustdb.gpg: trustdb created
gpg: key F01C0CAEAAA458E6: public key "Neo Chen <netkiller@msn.com>" imported
gpg: Total number processed: 1
gpg:               imported: 1

[root@localhost ~]# gpg --import private_key.asc
gpg: key F01C0CAEAAA458E6: "Neo Chen <netkiller@msn.com>" not changed
gpg: key F01C0CAEAAA458E6: secret key imported
gpg: Total number processed: 1
gpg:              unchanged: 1
gpg:       secret keys read: 1
gpg:   secret keys imported: 1

[root@localhost ~]# gpg --list-keys
/root/.gnupg/pubring.kbx
------------------------
pub   rsa2048 2021-10-08 [SC] [expires: 2023-10-08]
      70CECE32E5D67D12B95ED1E7F01C0CAEAAA458E6
```

```
uid           [ unknown] Neo Chen <netkiller@msn.com>
sub   rsa2048 2021-10-08 [E] [expires: 2023-10-08]

[root@localhost ~]# gpg --list-secret-keys --keyid-format LONG
/root/.gnupg/pubring.kbx
------------------------
sec   rsa2048/F01C0CAEAAA458E6 2021-10-08 [SC] [expires: 2023-10-08]
      70CECE32E5D67D12B95ED1E7F01C0CAEAAA458E6
uid                 [ unknown] Neo Chen <netkiller@msn.com>
ssb   rsa2048/EAA2F7FD813D2A2E 2021-10-08 [E] [expires: 2023-10-08]
```

7. 签名

二进制格式 *.gpg 签名文件

```
[root@netkiller ~]# gpg --passphrase "123456" --sign compose.yml
```

BASE64 格式的 *.asc 签名文件

```
[root@netkiller ~]# gpg --clear-sign stdin.log
[root@netkiller ~]# cat stdin.log.asc
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA256


-----BEGIN PGP SIGNATURE-----

iQEzBAEBCAAdFiEEcM7OMuXWfRK5XtHn8BwMrqqkWOYFAmFhDW8ACgkQ8BwMrqqk
WOY//ggAloOa20OfQUrVMKeHNijAhUCvG6wvEAz/cQRMjmkzinsdtVjoDo2gTtiS
lsfB6s4+5PkPq2kCW1v3edWiW33ghb4eK/1GDfTiFvE8ly0goAlD4N5Ruk3ROXJy
9InT5LtsuKTNWO3pKaJTjQ/dVdjcBUdEEZWMxJX5T/+JtJUg1tUOWmG7t+6gon+/
wMAzScqEIa2aaTEyX1tvUVIJWZUakZiqHY2mb3rnKmaUFe7Ny7vbqmkgvkDgzvbV
iIphoFRuTAtu7CbB5BV1oZ0IhiIztH7hlm5M5XlwTPo7fb8hyPo3NkvGsxCEM3mr
H770N2NHUVCumIw1jSOxqIv0cARcyg==
=J8o1
-----END PGP SIGNATURE-----
```

生成 *.sig 格式的签名文件

```
[root@netkiller ~]# gpg --detach-sign stdin.log

[root@netkiller ~]# ll stdin.log.*
-rw-r--r-- 1 root root 538 Oct  9 11:33 stdin.log.asc
```

```
-rw-r--r-- 1 root root 310 Oct  9 11:34 stdin.log.sig
```

生成BASE64 *.sig 格式的签名文件

```
[root@netkiller ~]# gpg --armor --detach-sign stdin.log
File 'stdin.log.asc' exists. Overwrite? (y/N) y
[root@netkiller ~]# cat stdin.log.asc
-----BEGIN PGP SIGNATURE-----

iQEzBAABCAAdFiEEcM7OMuXWfRK5XtHn8BwMrqqkWOYFAmFhDnkACgkQ8BwMrqqk
WOadjQf7BlOH6iIjylXIN3ziSrwtaM6AgQweBv3+PKV4VCFKLsKJOwecSTzy4sRu
JyzPoPoR/sHbAwe3VqbLTri1HVqtOy5mMuiK9KE3BjbmC8seo97tJ1O4gSd6OjyX
Q3eK/sQfRmo802qAFOj7iKldPGbJyBxhIZ+pirMNbMxrX2tfCq3o7qy1SjAyOhgO
ECorVAcHWl00xBj5vbgJJSIwo1kIx23+e/6lzKEwoseAj0sEPALqmi1r9HzocDoO
B0f5GgJVj74RC6svVQQMDwMjRgf27285CzxLpoTTioURtJawNI2NlUk59U11fYVH
4j8vOaHxTvP17LxAJyA9ndAsxz8y5w==
=+wjo
-----END PGP SIGNATURE-----
```

8. 加密/解密文件

8.1. 加密文件

加密

```
# echo hello > file.txt
# gpg -c file.txt

       lqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqk
       x Enter passphrase                                    x
       x                                                     x
       x                                                     x
       x Passphrase ****____________________________________ x
       x                                                     x
       x       <OK>                             <Cancel>     x
       mqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqj

       lqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqk
       x Please re-enter this passphrase                     x
       x                                                     x
       x Passphrase ****____________________________________ x
       x                                                     x
       x       <OK>                             <Cancel>     x
       mqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqj
```

```
# ls file.txt*
file.txt  file.txt.gpg
```

8.2. 解密

解密

```
# gpg -o myfile -d file.txt.gpg

       lqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqk
       x Enter passphrase                                     x
       x                                                      x
       x                                                      x
       x Passphrase ________________________________________ x
       x                                                      x
       x       <OK>                             <Cancel>      x
       mqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqj
```

8.3. 指定用户ID

-r, --recipient USER-ID encrypt for USER-ID

```
gpg --recipient [用户ID] --output netkiller.epub.gpg --encrypt
netkiller.epub
```

```
gpg --decrypt netkiller.epub.gpg --output netkiller.epub
```

8.4. 签名+加密

```
gpg --local-user [发信者ID] --recipient [接收者ID] --armor --sign
--encrypt netkiller.epub
```

```
gpg --verify netkiller.epub.asc netkiller.epub
```

9. 修改密钥

```
[root@netkiller ~]# gpg --edit-key
70CECE32E5D67D12B95ED1E7F01C0CAEAAA458E6
gpg (GnuPG) 2.2.20; Copyright (C) 2020 Free Software
Foundation, Inc.
This is free software: you are free to change and redistribute
it.
There is NO WARRANTY, to the extent permitted by law.

Secret key is available.

sec  rsa2048/F01C0CAEAAA458E6
     created: 2021-10-08  expires: 2023-10-08  usage: SC
     trust: ultimate      validity: ultimate
ssb  rsa2048/EAA2F7FD813D2A2E
     created: 2021-10-08  expires: 2023-10-08  usage: E
[ultimate] (1). Neo Chen <netkiller@msn.com>

gpg>
```

9.1. 显示帮助信息

使用 "?" 显示帮助信息

```
gpg> ?
quit        quit this menu
save        save and quit
help        show this help
fpr         show key fingerprint
grip        show the keygrip
list        list key and user IDs
uid         select user ID N
key         select subkey N
check       check signatures
```

```
sign        sign selected user IDs [* see below for related
commands]
lsign       sign selected user IDs locally
tsign       sign selected user IDs with a trust signature
nrsign      sign selected user IDs with a non-revocable
signature
adduid      add a user ID
addphoto    add a photo ID
deluid      delete selected user IDs
addkey      add a subkey
addcardkey  add a key to a smartcard
keytocard   move a key to a smartcard
bkuptocard  move a backup key to a smartcard
delkey      delete selected subkeys
addrevoker  add a revocation key
delsig      delete signatures from the selected user IDs
expire      change the expiration date for the key or selected
subkeys
primary     flag the selected user ID as primary
pref        list preferences (expert)
showpref    list preferences (verbose)
setpref     set preference list for the selected user IDs
keyserver   set the preferred keyserver URL for the selected
user IDs
notation    set a notation for the selected user IDs
passwd      change the passphrase
trust       change the ownertrust
revsig      revoke signatures on the selected user IDs
revuid      revoke selected user IDs
revkey      revoke key or selected subkeys
enable      enable key
disable     disable key
showphoto   show selected photo IDs
clean       compact unusable user IDs and remove unusable
signatures from key
minimize    compact unusable user IDs and remove all signatures
from key

* The 'sign' command may be prefixed with an 'l' for local
signatures (lsign),
  a 't' for trust signatures (tsign), an 'nr' for non-revocable
signatures
  (nrsign), or any combination thereof (ltsign, tnrsign, etc.).
```

```
```

9.2. 签名

```
gpg> sign
"Neo Chen <netkiller@msn.com>" was already signed by key
F01C0CAEAAA458E6
Nothing to sign with key F01C0CAEAAA458E6

gpg> save
```

9.3. 公钥信任配置

当我们使用 GPG 加密文件的时候会提示如下。

```
gpg: checking the trustdb
gpg: no ultimately trusted keys found
gpg: EAA2F7FD813D2A2E: There is no assurance this key belongs
to the named user

sub  rsa2048/EAA2F7FD813D2A2E 2021-10-08 Neo Chen
<netkiller@msn.com>
 Primary key fingerprint: 70CE CE32 E5D6 7D12 B95E  D1E7 F01C
0CAE AAA4 58E6
      Subkey fingerprint: CEFB 98EA 8508 45F8 338B  3898 EAA2
F7FD 813D 2A2E

It is NOT certain that the key belongs to the person named
in the user ID.  If you *really* know what you are doing,
you may answer the next question with yes.

Use this key anyway? (y/N)
```

信任公钥

```
[gitlab-runner@gitlab ~]$ gpg --edit-key netkiller@msn.com
gpg (GnuPG) 2.2.20; Copyright (C) 2020 Free Software
Foundation, Inc.
This is free software: you are free to change and redistribute
it.
There is NO WARRANTY, to the extent permitted by law.


pub  rsa2048/F01C0CAEAAA458E6
     created: 2021-10-08  expires: 2023-10-08  usage: SC
     trust: undefined     validity: unknown
sub  rsa2048/EAA2F7FD813D2A2E
     created: 2021-10-08  expires: 2023-10-08  usage: E
[ unknown] (1). Neo Chen <netkiller@msn.com>

gpg> trust
pub  rsa2048/F01C0CAEAAA458E6
     created: 2021-10-08  expires: 2023-10-08  usage: SC
     trust: undefined     validity: unknown
sub  rsa2048/EAA2F7FD813D2A2E
     created: 2021-10-08  expires: 2023-10-08  usage: E
[ unknown] (1). Neo Chen <netkiller@msn.com>

Please decide how far you trust this user to correctly verify
other users' keys
(by looking at passports, checking fingerprints from different
sources, etc.)

  1 = I don't know or won't say
  2 = I do NOT trust
  3 = I trust marginally
  4 = I trust fully
  5 = I trust ultimately
  m = back to the main menu

Your decision? 5
Do you really want to set this key to ultimate trust? (y/N) y

pub  rsa2048/F01C0CAEAAA458E6
     created: 2021-10-08  expires: 2023-10-08  usage: SC
     trust: ultimate      validity: unknown
sub  rsa2048/EAA2F7FD813D2A2E
     created: 2021-10-08  expires: 2023-10-08  usage: E
```

```
[ unknown] (1). Neo Chen <netkiller@msn.com>
Please note that the shown key validity is not necessarily
correct
unless you restart the program.

gpg> save
Key not changed so no update needed.
```

10. 加密备份 MySQL

准备环境:

数据库服务器一台，备份服务器一台。

我们将在备份服务器上创建密钥，然后将公钥导出并在数据库服务器上导入。

数据库服务器运行定时备份脚本，加密备份文件，同时每日将加密后的备份文件同步到本地。

备份内容只能在备份服务器上解密和查看

10.1. 创建密钥对

过程 6.1. 密钥管理

1. 创建密钥

```
[root@netkiller ~]# gpg --generate-key
gpg (GnuPG) 2.2.20; Copyright (C) 2020 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Note: Use "gpg --full-generate-key" for a full featured key generation dialog.

GnuPG needs to construct a user ID to identify your key.

Real name: Backup
Email address: backup@netkiller.cn
You selected this USER-ID:
    "Backup <backup@netkiller.cn>"

Change (N)ame, (E)mail, or (O)kay/(Q)uit? O
```

数据备份不需要 Passphrase 直接回车

```
        ┌.....................................................┐
        │ Please enter the passphrase to                      │
        │ protect your new key                                │
        │                                                     │
        │ Passphrase: ________________________________________│
        │                                                     │
        │       <OK>                              <Cancel>    │
        └─                                                   ─┘
```

选择 “Yes, protection is not needed” 直接回车。

```
                                        ┌
┐
                                        │ You have not entered a passphrase -
this is in general a bad idea!                            │
                                        │ Please confirm that you do not want to
have any protection on your key.                       │
                                        │
│
                                        │ <Yes, protection is not needed>
<Enter new passphrase>     │
                                        └
┘
```

系统会重复上面👆步骤两次。然后创建密钥

```
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
gpg: key 0C835D03507C8536 marked as ultimately trusted
gpg: revocation certificate stored as '/root/.gnupg/openpgp-
revocs.d/18235CBA04497C42EFAC78210C835D03507C8536.rev'
public and secret key created and signed.

pub   rsa2048 2021-10-09 [SC] [expires: 2023-10-09]
      18235CBA04497C42EFAC78210C835D03507C8536
uid                      Backup <backup@netkiller.cn>
sub   rsa2048 2021-10-09 [E] [expires: 2023-10-09]
```

2. 导出公钥

   查看用户ID

```
[root@netkiller ~]# gpg --list-keys backup@netkiller.cn
pub   rsa2048 2021-10-09 [SC] [expires: 2023-10-09]
      18235CBA04497C42EFAC78210C835D03507C8536
uid           [ultimate] Backup <backup@netkiller.cn>
sub   rsa2048 2021-10-09 [E] [expires: 2023-10-09]
```

导出 Backup 用户公钥

```
[root@netkiller ~]# gpg --armor --output backup.gpg --export
18235CBA04497C42EFAC78210C835D03507C8536
```

把公钥发送给数据库服务器

```
[root@netkiller ~]# scp backup.gpg www@192.168.30.10:/home/www
Warning: Permanently added '192.168.30.10' (ECDSA) to the list of known hosts.
www@192.168.30.10's password:
backup.gpg
```

10.2. 数据库备份

过程 6.2. 数据库备份

1. 导入公钥

```
[www@testing ~]$ gpg --import backup.gpg
gpg: directory '/home/www/.gnupg' created
gpg: keybox '/home/www/.gnupg/pubring.kbx' created
gpg: /home/www/.gnupg/trustdb.gpg: trustdb created
gpg: key 0C835D03507C8536: public key "Backup <backup@netkiller.cn>" imported
gpg: Total number processed: 1
gpg:               imported: 1
```

```
[www@testing ~]$ gpg -k
/home/www/.gnupg/pubring.kbx
----------------------------
pub   rsa2048 2021-10-09 [SC] [expires: 2023-10-09]
      18235CBA04497C42EFAC78210C835D03507C8536
uid           [ unknown] Backup <backup@netkiller.cn>
sub   rsa2048 2021-10-09 [E] [expires: 2023-10-09]
```

测试

```
[www@testing ~]$ gpg -r 18235CBA04497C42EFAC78210C835D03507C8536 -e
netkiller.sql.gz
gpg: 339634D92F842BE7: There is no assurance this key belongs to the named user
```

```
sub  rsa2048/339634D92F842BE7 2021-10-09 Backup <backup@netkiller.cn>
 Primary key fingerprint: 1823 5CBA 0449 7C42 EFAC  7821 0C83 5D03 507C 8536
      Subkey fingerprint: BA6F 7A53 C82B 9945 C1B4  AB09 3396 34D9 2F84 2BE7

It is NOT certain that the key belongs to the person named
in the user ID.  If you *really* know what you are doing,
you may answer the next question with yes.

Use this key anyway? (y/N) y
[www@testing ~]$ ls netkiller.sql.gz*
netkiller.sql.gz  netkiller.sql.gz.gpg
```

信任密钥

```
[www@testing ~]$ gpg --edit-key 18235CBA04497C42EFAC78210C835D03507C8536
gpg (GnuPG) 2.2.20; Copyright (C) 2020 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.


pub  rsa2048/0C835D03507C8536
     created: 2021-10-09  expires: 2023-10-09  usage: SC
     trust: unknown       validity: unknown
sub  rsa2048/339634D92F842BE7
     created: 2021-10-09  expires: 2023-10-09  usage: E
[ unknown] (1). Backup <backup@netkiller.cn>

gpg> trust
pub  rsa2048/0C835D03507C8536
     created: 2021-10-09  expires: 2023-10-09  usage: SC
     trust: unknown       validity: unknown
sub  rsa2048/339634D92F842BE7
     created: 2021-10-09  expires: 2023-10-09  usage: E
[ unknown] (1). Backup <backup@netkiller.cn>

Please decide how far you trust this user to correctly verify other users' keys
(by looking at passports, checking fingerprints from different sources, etc.)

  1 = I don't know or won't say
  2 = I do NOT trust
  3 = I trust marginally
  4 = I trust fully
  5 = I trust ultimately
  m = back to the main menu

Your decision? 5
Do you really want to set this key to ultimate trust? (y/N) y

pub  rsa2048/0C835D03507C8536
     created: 2021-10-09  expires: 2023-10-09  usage: SC
     trust: ultimate      validity: unknown
sub  rsa2048/339634D92F842BE7
     created: 2021-10-09  expires: 2023-10-09  usage: E
[ unknown] (1). Backup <backup@netkiller.cn>
```

```
Please note that the shown key validity is not necessarily correct
unless you restart the program.

gpg> quit
```

再次测试，密钥已信任

```
[www@testing ~]$ rm netkiller.sql.gz.gpg
[www@testing ~]$ gpg -r 18235CBA04497C42EFAC78210C835D03507C8536 -e
netkiller.sql.gz
```

2. 数据库备份

   在 /etc/cron.daily/ 目录下创建 mysql 脚本，然后赋予执行权限

```
root@production:~# cat /etc/cron.daily/mysql
#!/bin/bash
###################################
# $Id: backup 379 2012-04-02 08:43:42Z netkiller $
# Author: netkiller@msn.com
# Home: http://netkiller.github.com
###################################
# SELECT `user`, `host`, `password` FROM `mysql`.`user`;
# CREATE USER 'backup'@'localhost' IDENTIFIED BY
'SaJePoM6BAPOmOFOd7Xo3e1A52vEPE';
# GRANT SELECT, LOCK TABLES  ON *.* TO 'backup'@'localhost';
# FLUSH PRIVILEGES;
# SHOW GRANTS FOR 'backup'@'localhost';
###################################
BACKUP_HOST="172.188.122.155"
BACKUP_USER="dba"
BACKUP_PASS=""
BACKUP_DIR=/opt/database/mysql
BACKUP_DBNAME="netkiller neo test"
#TIMEPOINT=$(date -u +%Y-%m-%d)
TIMEPOINT=$(date +%Y-%m-%d.%H:%M:%S)
#Number of copies
COPIES=30
####################################
MYSQLDUMP="/usr/bin/mysqldump"
MYSQLDUMP_OPTS="-h $BACKUP_HOST -u$BACKUP_USER -p$BACKUP_PASS --compress --
events --triggers --routines --set-gtid-purged=OFF"
# --skip-lock-tables
####################################
umask 0077
test ! -d "$BACKUP_DIR" && mkdir -p "$BACKUP_DIR"
test ! -w $BACKUP_DIR && echo "Error: $BACKUP_DIR is un-writeable." && exit 0

for dbname in $BACKUP_DBNAME
do
```

```
        test ! -d "$BACKUP_DIR/$dbname" && mkdir -p "$BACKUP_DIR/$dbname"
        LOGFILE=$BACKUP_DIR/$dbname/error.log
        $MYSQLDUMP $MYSQLDUMP_OPTS --log-error=$LOGFILE $dbname | gpg -r
backup@netkiller.cn -e -o $BACKUP_DIR/$dbname/$dbname.$TIMEPOINT.sql.gpg
done
find $BACKUP_DIR -type f -mtime +$COPIES -delete
```

提示

gpg 自带压缩，所以备份数据无需使用 gzip 压缩

```
[www@testing ~]$ gpg -r backup@netkiller.cn -e netkiller.2021-8-28.sql
[www@testing ~]$ ll
-rw-r--r-- 1 www www 588143144 2021-08-28 10:31 netkiller.2021-8-28.sql
-rw-r--r-- 1 www www  41395738 2021-10-09 12:01 netkiller.2021-8-28.sql.gpg
```

源文件大小是 588143144，经过 gpg 压缩后 41395738

使用 -z 参数可以设置压缩级别，这里设置为最高级别9，压缩后大小是 39847904，但是通常我不建议设置，这会影响数据被备份时常，数据备份过程需要锁表，会影响用户访问，所以要尽快完成备份。

```
[www@testing ~]$ gpg -r backup@netkiller.cn -z 9 -e netkiller.2021-8-28.sql
File 'netkiller.2021-8-28.sql.gpg' exists. Overwrite? (y/N) y

[www@testing ~]$ ll netkiller.2021-8-28.sql*
-rw-r--r-- 1 www www 588143144 2021-08-28 10:31 netkiller.2021-8-28.sql
-rw-r--r-- 1 www www  39847904 2021-10-09 12:17 netkiller.2021-8-28.sql.gpg
```

10.3. 数据库还原

过程 6.3. 数据库还原

1. 定时同步

```
[root@netkiller ~]# cat /etc/cron.daily/mysql
rsync -auzv www@db.netkiller.cn:/opt/database/mysql /opt/backup/database/
```

2. 解密数据库备份文件

```
[root@netkiller ~]# gpg netkiller.2021-8-28.sql.gpg
```

--output 指定文件名

```
[root@netkiller ~]# gpg --output netkiller.2021-8-28.sql --decrypt
netkiller.2021-8-28.sql.gpg
gpg: encrypted with 2048-bit RSA key, ID 339634D92F842BE7, created 2021-10-09
      "Backup <backup@netkiller.cn>"
```

直接恢复数据库

```
[root@netkiller ~]# gpg --decrypt netkiller.2021-8-28.sql.gpg | mysql netkiller
```

11. 旧版本 1.4.11

11.1. GnuPG

GnuPG

```
~/.gnupg/gpg.conf        – 配置文件
~/.gnupg/trustdb.gpg – 信任库
~/.gnupg/pubring.gpg – 公钥库
~/.gnupg/secring.gpg – 私钥库
```

11.2. Creating a key (创建key)

--gen-key generate a new key pair

```
neo@neo-laptop:~$ gpg --gen-key
gpg (GnuPG) 1.4.11; Copyright (C) 2010 Free Software
Foundation, Inc.
This is free software: you are free to change and redistribute
it.
There is NO WARRANTY, to the extent permitted by law.

Please select what kind of key you want:
   (1) RSA and RSA (default)
   (2) DSA and Elgamal
   (3) DSA (sign only)
   (4) RSA (sign only)
Your selection?
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048)
Requested keysize is 2048 bits
Please specify how long the key should be valid.
         0 = key does not expire
      <n>  = key expires in n days
      <n>w = key expires in n weeks
      <n>m = key expires in n months
      <n>y = key expires in n years
Key is valid for? (0) 1y
Key expires at Tuesday, October 22, 2013 PM03:25:35 HKT
Is this correct? (y/N) y

You need a user ID to identify your key; the software
constructs the user ID
from the Real Name, Comment and Email Address in this form:
    "Heinrich Heine (Der Dichter) <heinrichh@duesseldorf.de>"

Real name: Neo Chan
Email address: netkiller@msn.com
Comment: Office Email
You selected this USER-ID:
    "Neo Chan (Office Email) <netkiller@msn.com>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? O
You need a Passphrase to protect your secret key.

We need to generate a lot of random bytes. It is a good idea to
perform
some other action (type on the keyboard, move the mouse,
utilize the
disks) during the prime generation; this gives the random
number
generator a better chance to gain enough entropy.

Not enough random bytes available.  Please do some other work
to give
the OS a chance to collect more entropy! (Need 282 more bytes)
.......+++++
.......+++++
We need to generate a lot of random bytes. It is a good idea to
perform
some other action (type on the keyboard, move the mouse,
utilize the
disks) during the prime generation; this gives the random
number
generator a better chance to gain enough entropy.
..+++++
..+++++
gpg: key CEF09301 marked as ultimately trusted
public and secret key created and signed.

gpg: checking the trustdb
```

```
gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust
model
gpg: depth: 0  valid:   1  signed:   0  trust: 0-, 0q, 0n, 0m,
0f, 1u
gpg: next trustdb check due at 2013-10-22
pub   2048R/CEF09301 2012-10-22 [expires: 2013-10-22]
      Key fingerprint = C5B7 50EF 762F 3F7F 1AE5  8AEC AF10
D50D CEF0 9301
uid                  Neo Chan (Office Email)
<netkiller@msn.com>
sub   2048R/0BF4D523 2012-10-22 [expires: 2013-10-22]
```

11.3. --list-keys 列出已存在的证书

```
$ gpg --list-keys
/home/neo/.gnupg/pubring.gpg
----------------------------
pub   1024R/63268A35 2013-09-11
uid                  Neo Chen (netkiller) <netkiller@msn.com>
sub   1024R/F4F946F9 2013-09-11
```

11.4. Exporting keys (导出key)

--export export keys

```
gpg --export -a
```

-o, --output use as output file

```
```

```
$ gpg --export -a -o test.asc
$ gpg --output yourname.asc --export -a
```

11.5. Importing keys (导入key)

--import import/merge keys

```
neo@neo-laptop:~$ gpg --import 409.asc
gpg: key 4D3A0803: public key "Phoenix.L <409@example.com>"
imported
gpg: key CEF09301: "Neo Chan (Office Email)
<netkiller@msn.com>" not changed
gpg: Total number processed: 2
gpg:               imported: 1  (RSA: 1)
gpg:              unchanged: 1
```

11.6. Revoke a key (吊销key)

```
gpg --gen-revoke
```

12. GnuPG For Windows

下载OpenGPG: http://www.gnupg.org/

注意

GnuPG (OpenGPG)安装时可以选择语言，支持简体中文. 但对中文支持不是很好，如真实姓名输入：王老五,系统提示"姓名至少要有五个字符长"

12.1. 生成密钥对

使用 gpg --gen-key 生成密钥对

```
C:\GNU>gpg --gen-key
gpg (GnuPG) 1.4.3; Copyright (C) 2006 Free Software Foundation,
Inc.
This program comes with ABSOLUTELY NO WARRANTY.
This is free software, and you are welcome to redistribute it
under certain conditions. See the file COPYING for details.

请选择您要使用的密钥种类：
   (1) DSA 和 ElGamal (默认)
   (2) DSA (仅用于签字)
   (5) RSA (仅用于签字)
您的选择?
DSA 密钥对会有 1024 位。
ELG-E 密钥长度应在 1024 位与 4096 位之间。
您想要用多大的密钥尺寸?(2048)
您所要求的密钥尺寸是 2048 位
请设定这把密钥的有效期限。
         0 = 密钥永不过期
      <n>  = 密钥在 n 天后过期
      <n>w = 密钥在 n 周后过期
      <n>m = 密钥在 n 月后过期
      <n>y = 密钥在 n 年后过期
密钥的有效期限是?(0)
密钥永远不会过期
以上正确吗?(y/n)y
```

```
您需要一个用户标识来辨识您的密钥；本软件会用真实姓名、注释和电子邮件地址组合
成用户标识，如下所示：
    “Heinrich Heine (Der Dichter) <heinrichh@duesseldorf.de>”

真实姓名：neo chen
电子邮件地址：openunix@163.com
注释：netkiller
您选定了这个用户标识：
    “neo chen (netkiller) <openunix@163.com>”

更改姓名(N)、注释(C)、电子邮件地址(E)或确定(O)/退出(Q)？O
您需要一个密码来保护您的私钥。

我们需要生成大量的随机字节。这个时候您可以多做些琐事(像是敲打键盘、移动
鼠标、读写硬盘之类的)，这会让随机数字发生器有更好的机会获得足够的熵数。
+++++++++++++++++++++++++++++++++++.++++++++++++++++++++++++++
++++++++.++++++++
++++++++++++.++++++++++++++++++++.+++++++++++++++++++++++++...>
+++++............
..+++++
我们需要生成大量的随机字节。这个时候您可以多做些琐事(像是敲打键盘、移动
鼠标、读写硬盘之类的)，这会让随机数字发生器有更好的机会获得足够的熵数。
+++++.++++++++++++++++++++..++++++++++.+++++.++++++++++.+++++++
+++.+++++.+++++++
++++++++.+++++....+++++..++++++++++.+++++++++++++++++++..+++++
+++++++++++++++.+
++++>++++++++++...>.+++++............................>+++++
<+++++..............
..............+++++^^^^^
gpg: 密钥 C9441A1A 被标记为绝对信任
公钥和私钥已经生成并经签字。

gpg: 正在检查信任度数据库
gpg: 需要 3 份勉强信任和 1 份完全信任，PGP 信任模型
gpg: 深度：0 有效性：  2 已签字：  0 信任度：0-，0q，0n，0m，0f，2u
pub   1024D/C9441A1A 2006-06-02
密钥指纹 = EFDC A97C C711 E9C9 FAC1  3EA9 33C1 1FB2 C944 1A1A
uid                  neo chen (netkiller) <openunix@163.com>
sub   2048g/B713326C 2006-06-02

C:\GNU>
```

12.2. 列出密钥

列出密钥使用 gpg --list-keys

```
C:\GNU>gpg --list-keys
C:/Documents and Settings/neo.chen/Application
Data/gnupg\pubring.gpg
---------------------------------------------------------------
------
pub   1024D/C9441A1A 2006-06-02
uid                  neo chen (netkiller) <openunix@163.com>
sub   2048g/B713326C 2006-06-02
```

列出密钥和签字使用 gpg --list-keys

```
C:\GNU>gpg --list-sigs
C:/Documents and Settings/neo.chen/Application
Data/gnupg\pubring.gpg
---------------------------------------------------------------
------
pub   1024D/C9441A1A 2006-06-02
uid                  neo chen (netkiller) <openunix@163.com>
sig 3        C9441A1A 2006-06-02  neo chen (netkiller)
<openunix@163.com>
sub   2048g/B713326C 2006-06-02
sig          C9441A1A 2006-06-02  neo chen (netkiller)
<openunix@163.com>
```

列出并检查密钥签字 gpg --check-sigs

```
C:\GNU>gpg --check-sigs
C:/Documents and Settings/neo.chen/Application
Data/gnupg\pubring.gpg
```

```
------------------------------------------------------------------
------
pub   1024D/C9441A1A 2006-06-02
uid                  neo chen (netkiller) <openunix@163.com>
sig!3        C9441A1A 2006-06-02  neo chen (netkiller)
<openunix@163.com>
sub   2048g/B713326C 2006-06-02
sig!         C9441A1A 2006-06-02  neo chen (netkiller)
<openunix@163.com>
```

12.3. 验证签字

检查 PGP 签名 与 md5sum 作用类似:

```
bash$ gpg --verify gnupg-x.x.x.tar.gz.sig gnupg-x.x.x.tar.gz
bash$ md5sum gnupg-x.x.x.tar.gz
```

12.4. EMail-Security

EMail-Security using GnuPG for Windows

gpg4win

13. Smart Card

http://www.gnupg.org/howtos/card-howto/en/smartcard-howto-single.html

14. PGP

下载PGP: http://www.pgp.com/

15. OpenPGP

https://www.openpgp.org

下载OpenPGP: http://www.pgpi.org/

第 7 章 OpenSSL

不多说了。

1. openssl 命令参数

1.1. version

```
[root@netkiller nginx]# openssl version
OpenSSL 1.0.1e-fips 11 Feb 2013
```

1.2. 测试加密算法的速度

```
$ openssl speed
```

```
$ openssl speed rsa
$ openssl speed aes
```

1.3. req

```
openssl req -new -x509 -days 7300 -key ca.key -out ca.crt
```

1.4. x509

```
openssl x509 -req -in client-req.csr -out client.crt -signkey client-
key.pem -CA ca.crt -CAkey ca.key -days 365 -CAserial serial
```

验证一下我们生成的文件。

```
openssl x509 -in cacert.pem -text -noout
```

-extfile

```
openssl x509 -req -in careq.pem -extfile openssl.cnf -extensions v3_ca -
signkey key.pem -out cacert.pem
```

1.5. ca

```
# 生成CRL列表
$ openssl ca -gencrl -out exampleca.crl
```

1.6. crl

```
# 查看CRL列表信息
$ openssl crl -in exampleca.crl -text -noout

# 验证CRL列表签名信息
$ openssl crl -in exampleca.crl -noout -CAfile cacert.pem
```

1.7. pkcs12

-clcerts 表示仅导出客户证书。

```
openssl pkcs12 -export -clcerts -in 324.cer -inkey ca.pem -out 324.p12 -
name "Email SMIME"
```

转换PEM证书文件和私钥到PKCS#12文件

```
openssl pkcs12 -export -out certificate.pfx -inkey privateKey.key -in
certificate.crt -certfile CACert.crt
```

1.8. passwd

MD5-based password algorithm

```
# openssl passwd -1 -salt 'random-phrase-here' 'your-password-here'
```

```
$1$random-p$AOw9RDIWQm6tfUo9Ediu/0
```

-crypt standard Unix password algorithm (default)

```
# openssl passwd -crypt -salt 'sa' 'password'
sa3tHJ3/KuYvI
```

1.9. digest

如何创建一个文件的 MD5 或 SHA1 摘要?

摘要创建使用 dgst 选项.

list-message-digest-commands

列出可用摘要

```
$ openssl list-message-digest-commands
md2
md4
md5
mdc2
rmd160
sha
sha1
```

md5

```
# MD5 digest
openssl dgst -md5 filename
```

注意

MD5 信息摘要也同样可以使用md5sum创建

```
$ echo "Hello World!" > message.txt
$ openssl dgst -md5 message.txt
MD5(message.txt)= d9226d4bd8779baa69db272f89a2e05c
```

```
```

sha1

```
# SHA1 digest
openssl dgst -sha1 filename
```

```
$ openssl dgst -sha1 /etc/passwd
SHA1(/etc/passwd)= 9d883a9d35fd9a6dc81e6a1717a8e2ecfc49cdd8
```

1.10. enc

使用方法:

$ openssl enc 加密算法 -k 密码 -in 输入明文文件 -out 输出密文文件

$ openssl enc 加密算法 -k 密码 -in 输出密文文件 -out 输入明文文件

list-cipher-commands

可用的编码/解码方案

```
# or get a long list, one cipher per line
openssl list-cipher-commands

# openssl list-cipher-commands
aes-128-cbc
aes-128-ecb
aes-192-cbc
aes-192-ecb
aes-256-cbc
aes-256-ecb
base64
bf
bf-cbc
bf-cfb
bf-ecb
bf-ofb
cast
```

```
cast-cbc
cast5-cbc
cast5-cfb
cast5-ecb
cast5-ofb
des
des-cbc
des-cfb
des-ecb
des-ede
des-ede-cbc
des-ede-cfb
des-ede-ofb
des-ede3
des-ede3-cbc
des-ede3-cfb
des-ede3-ofb
des-ofb
des3
desx
idea
idea-cbc
idea-cfb
idea-ecb
idea-ofb
rc2
rc2-40-cbc
rc2-64-cbc
rc2-cbc
rc2-cfb
rc2-ecb
rc2-ofb
rc4
rc4-40
rc5
rc5-cbc
rc5-cfb
rc5-ecb
rc5-ofb
```

base64

使用 base64-encode 编码/解码?

使用 enc -base64 选项

```
# send encoded contents of file.txt to stdout
```

```
openssl enc -base64 -in file.txt

# same, but write contents to file.txt.enc
openssl enc -base64 -in file.txt -out file.txt.enc
```

命令行

```
C:\GnuWin32\neo>openssl enc -base64 -in file.txt
SGVsbG8gV29ybGQhDQo=

C:\GnuWin32\neo>openssl enc -base64 -in file.txt -out file.txt.enc

C:\GnuWin32\neo>type file.txt.enc
SGVsbG8gV29ybGQhDQo=

C:\GnuWin32\neo>
```

通过管道操作

```
C:\GnuWin32\neo>echo "encode me" | openssl enc -base64
ImVuY29kZSBtZSIgDQo=

C:\GnuWin32\neo>echo -n "encode me" | openssl enc -base64
LW4gImVuY29kZSBtZSIgDQo=

C:\GnuWin32\neo>
```

使用 -d (解码) 选项来反转操作.

```
C:\GnuWin32\neo>openssl enc -base64 -d -in file.txt.enc
Hello World!

C:\GnuWin32\neo>openssl enc -base64 -d -in file.txt.enc -out file.txt
```

快速命令行

```
C:\GnuWin32\neo>type file.txt.enc | openssl enc -base64 -d
Hello World!

C:\GnuWin32\neo>type file.txt.enc
SGVsbG8gV29ybGQhDQo=
C:\GnuWin32\neo>echo SGVsbG8gV29ybGQhDQo= | openssl enc -base64 -d
Hello World!
```

des

对称加密与解密

加密

```
# openssl enc -des -e -a -in file.txt -out file.txt.des
enter des-cbc encryption password:
Verifying - enter des-cbc encryption password:
```

解密

```
# openssl enc -des -d -a -in file.txt.des -out file.txt.tmp
enter des-cbc decryption password:
```

```
% echo abc | openssl des-cbc -k 123 -base64
U2FsdGVkX1+atYQyhz7I1ktb5XtYasGk
```

aes

加密

```
openssl enc -aes-128-cbc -in filename -out filename.out
```

解密

```
openssl enc -d -aes-128-cbc -in filename.out -out filename
```

```
echo abc | openssl aes-128-cbc -k 123 -base64
```

1.11. rsa

产生密钥对

生成私钥

```
openssl genrsa -out private.key 1024
```

根据私钥产生公钥

```
openssl rsa -in private.key -pubout > public.key
```

用公钥加密明文

```
$ openssl rsautl -encrypt -pubin -inkey public.key -in filename -out
filename.out
```

用私钥解密

```
$ openssl rsautl -decrypt -inkey private.key -in filename.out -out
filename
```

1.12. dsa

例 7.1. dsaparam & gendsa

```
# create parameters in dsaparam.pem
openssl dsaparam -out dsaparam.pem 1024

# create first key
openssl gendsa -out key1.pem dsaparam.pem

# and second ...
```

```
openssl gendsa -out key2.pem dsaparam.pem
```

生成私钥

```
openssl dsaparam -out dsaparam.pem 1024
openssl gendsa -out private.key dsaparam.pem
```

根据私钥产生公钥

```
openssl dsa -in private.key -pubout -out public.key
```

```
$ ls
dsaparam.pem  private.key  public.key

$ cat *
-----BEGIN DSA PARAMETERS-----
MIIBHgKBgQCAkvuZmbK7zgTv3WnYayypdghcNKA+jP7/fdwy82JfqkJeF38FOOu8
4cbrQjzs6XdANeZk3c6BVQfqNfFnUomKARm0gdqeelsmyHMV+0jy7fuX1HHIUZyJ
Rqravmh+o9iYX1aA3jsP5sDoosEEEYKQBAUEi6vwzCnjCra3TBuvmQIVAPYqwKI3
v6nkKAfn+lqPvmHqVDv5AoGAb7vilZ7EtuYpJbpURZtTPOtLpMmpfwXq+g7cKQ7Z
mC+TCwzVUkBv8s/gxwr7r92bCmGTGJGuBVGqI0yEbrkMRGieJwOrS885NNg+AiTW
DB0Xo2klaTg5rFydGxPvWI72cpyds69Ptm4z9Th0xrtDUNIYPdDIR+rVUao5XBS9
U4w=
-----END DSA PARAMETERS-----
-----BEGIN DSA PRIVATE KEY-----
MIIBugIBAAKBgQCAkvuZmbK7zgTv3WnYayypdghcNKA+jP7/fdwy82JfqkJeF38F
OOu84cbrQjzs6XdANeZk3c6BVQfqNfFnUomKARm0gdqeelsmyHMV+0jy7fuX1HHI
UZyJRqravmh+o9iYX1aA3jsP5sDoosEEEYKQBAUEi6vwzCnjCra3TBuvmQIVAPYq
wKI3v6nkKAfn+lqPvmHqVDv5AoGAb7vilZ7EtuYpJbpURZtTPOtLpMmpfwXq+g7c
KQ7ZmC+TCwzVUkBv8s/gxwr7r92bCmGTGJGuBVGqI0yEbrkMRGieJwOrS885NNg+
AiTWDB0Xo2klaTg5rFydGxPvWI72cpyds69Ptm4z9Th0xrtDUNIYPdDIR+rVUao5
XBS9U4wCgYBISbp4/z5JY2OqXVttS6G4GQT0PMAiJZi9pty4H0rKoSmbrgjev/wp
7BW8NqaJnlSjNCzF4SH+DXxZeuktJPNftHYi8BPIrHxR6CG1h7VPDr/IwSoff0Kx
Lhc6vqxcCRpcQoqbhXGG5RxMsczD4nRmdmhXbelPRu10T4qxEiVG7gIUc1KsK+hA
+EzXl80Eyj2Si7UH/wI=
-----END DSA PRIVATE KEY-----
-----BEGIN PUBLIC KEY-----
MIIBtjCCASsGByqGSM44BAEwggEeAoGBAICS+5mZsrvOBO/dadhrLKl2CFw0oD6M
/v993DLzYl+qQl4XfwU467zhxutCPOzpd0A15mTdzoFVB+o18WdSiYoBGbSB2p56
WybIcxX7SPLt+5fUcchRnIlGqtq+aH6j2JhfVoDeOw/mwOiiwQQRgpAEBQSLq/DM
KeMKtrdMG6+ZAhUA9irAoje/qeQoB+f6Wo++YepUO/kCgYBvu+KVnsS25iklulRF
m1M860ukyal/Ber6DtwpDtmYL5MLDNVSQG/yz+DHCvuv3ZsKYZMYka4FUaojTIRu
uQxEaJ4nA6tLzzk02D4CJNYMHRejaSVpODmsXJ0bE+9YjvZynJ2zr0+2bjP1OHTG
u0NQ0hg90MhH6tVRqjlcFL1TjAOBhAACgYBISbp4/z5JY2OqXVttS6G4GQT0PMAi
```

```
JZi9pty4H0rKoSmbrgjev/wp7BW8NqaJnlSjNCzF4SH+DXxZeuktJPNftHYi8BPI
rHxR6CG1h7VPDr/IwSoff0KxLhc6vqxcCRpcQoqbhXGG5RxMsczD4nRmdmhXbelP
Ru10T4qxEiVG7g==
-----END PUBLIC KEY-----
```

1.13. rc4

加密文件

```
# openssl enc -e -rc4 -in in.txt -out out.txt
enter rc4 encryption password:
Verifying - enter rc4 encryption password:
```

解密文件

```
# openssl enc -d -rc4 -in out.txt -out test.txt
enter rc4 decryption password:
```

使用 -k 指定密钥

```
openssl enc -e -rc4 -k passwd -in in.txt -out out.txt
openssl enc -d -rc4 -k passwd -in out.txt -out test.txt
```

1.14. -config 指定配置文件

```
# openssl req -new -newkey rsa:2048 -config openssl.cfg -keyout
server.key -nodes -out certreq.csr
```

1.15. -subj 指定参数

```
# openssl req -new -newkey rsa:2048 -keyout server.key -nodes -subj
/C=CN/O=example.com/OU=IT/CN=Neo/ST=GD/L=Shenzhen -out certreq.csr

C:\> openssl req -new -newkey rsa:2048 -config openssl.cfg -keyout
server.key -nodes -subj
/C=CN/O="%OrganizationName%"/OU="%OrganizationUnit%"/CN="%CommonName%"/S
T="%StateName%"/L="%LocalityName%" -out certreq.csr
```

```
openssl req -x509 -nodes -days 365 -newkey rsa:4096 -keyout
/etc/nginx/ssl/www.netkiller.cn.key -out
/etc/nginx/ssl/www.netkiller.cn.crt -subj
"/C=CN/ST=Guangdong/L=Shenzhen/O=Global Security/OU=IT
Department/CN=www.netkiller.cn/emailAddress=netkiller@msn.com"

openssl req -x509 -nodes -days 365 -newkey rsa:4096 -keyout
/etc/nginx/ssl/www.netkiller.cn.key -out
/etc/nginx/ssl/www.netkiller.cn.crt -subj
"/C=CN/ST=Guangdong/L=Shenzhen/O=Global Security/OU=IT
Department/CN=*netkiller.cn/emailAddress=netkiller@msn.com"
```

1.16. rand

生成随机数

```
openssl rand 12 -base64
```

```
# openssl rand -base64 24
rgphwqZFFA2tY1QfuBrmw3aN62i6ctFy
```

1.17. 去除私钥的密码

```
$ openssl rsa -in neo.key -out nopassword.key
Enter pass phrase for neo.key:
writing RSA key
```

1.18. ciphers

```


neo@MacBook-Pro-Neo ~ % openssl ciphers -v
ECDHE-RSA-AES256-GCM-SHA384 TLSv1.2 Kx=ECDH     Au=RSA  Enc=AESGCM(256)
Mac=AEAD
ECDHE-ECDSA-AES256-GCM-SHA384 TLSv1.2 Kx=ECDH     Au=ECDSA
Enc=AESGCM(256) Mac=AEAD
ECDHE-RSA-AES256-SHA384 TLSv1.2 Kx=ECDH     Au=RSA  Enc=AES(256)
Mac=SHA384
ECDHE-ECDSA-AES256-SHA384 TLSv1.2 Kx=ECDH     Au=ECDSA Enc=AES(256)
Mac=SHA384
```

```
ECDHE-RSA-AES256-SHA    SSLv3 Kx=ECDH     Au=RSA  Enc=AES(256)  Mac=SHA1
ECDHE-ECDSA-AES256-SHA  SSLv3 Kx=ECDH     Au=ECDSA Enc=AES(256)
Mac=SHA1
DHE-RSA-AES256-GCM-SHA384 TLSv1.2 Kx=DH       Au=RSA  Enc=AESGCM(256)
Mac=AEAD
DHE-RSA-AES256-SHA256   TLSv1.2 Kx=DH       Au=RSA  Enc=AES(256)
Mac=SHA256
DHE-RSA-AES256-SHA      SSLv3 Kx=DH       Au=RSA  Enc=AES(256)  Mac=SHA1
ECDHE-ECDSA-CHACHA20-POLY1305 TLSv1.2 Kx=ECDH     Au=ECDSA Enc=ChaCha20-
Poly1305 Mac=AEAD
ECDHE-RSA-CHACHA20-POLY1305 TLSv1.2 Kx=ECDH     Au=RSA  Enc=ChaCha20-
Poly1305 Mac=AEAD
DHE-RSA-CHACHA20-POLY1305 TLSv1.2 Kx=DH       Au=RSA  Enc=ChaCha20-
Poly1305 Mac=AEAD
GOST2012256-GOST89-GOST89 SSLv3 Kx=GOST     Au=GOST01 Enc=GOST-28178-89-
CNT Mac=GOST89IMIT
DHE-RSA-CAMELLIA256-SHA256 TLSv1.2 Kx=DH       Au=RSA  Enc=Camellia(256)
Mac=SHA256
DHE-RSA-CAMELLIA256-SHA SSLv3 Kx=DH       Au=RSA  Enc=Camellia(256)
Mac=SHA1
GOST2001-GOST89-GOST89  SSLv3 Kx=GOST     Au=GOST01 Enc=GOST-28178-89-
CNT Mac=GOST89IMIT
AES256-GCM-SHA384       TLSv1.2 Kx=RSA      Au=RSA  Enc=AESGCM(256)
Mac=AEAD
AES256-SHA256           TLSv1.2 Kx=RSA      Au=RSA  Enc=AES(256)
Mac=SHA256
AES256-SHA              SSLv3 Kx=RSA      Au=RSA  Enc=AES(256)  Mac=SHA1
CAMELLIA256-SHA256      TLSv1.2 Kx=RSA      Au=RSA  Enc=Camellia(256)
Mac=SHA256
CAMELLIA256-SHA         SSLv3 Kx=RSA      Au=RSA  Enc=Camellia(256)
Mac=SHA1
ECDHE-RSA-AES128-GCM-SHA256 TLSv1.2 Kx=ECDH     Au=RSA  Enc=AESGCM(128)
Mac=AEAD
ECDHE-ECDSA-AES128-GCM-SHA256 TLSv1.2 Kx=ECDH     Au=ECDSA
Enc=AESGCM(128) Mac=AEAD
ECDHE-RSA-AES128-SHA256 TLSv1.2 Kx=ECDH     Au=RSA  Enc=AES(128)
Mac=SHA256
ECDHE-ECDSA-AES128-SHA256 TLSv1.2 Kx=ECDH     Au=ECDSA Enc=AES(128)
Mac=SHA256
ECDHE-RSA-AES128-SHA    SSLv3 Kx=ECDH     Au=RSA  Enc=AES(128)  Mac=SHA1
ECDHE-ECDSA-AES128-SHA  SSLv3 Kx=ECDH     Au=ECDSA Enc=AES(128)
Mac=SHA1
DHE-RSA-AES128-GCM-SHA256 TLSv1.2 Kx=DH       Au=RSA  Enc=AESGCM(128)
Mac=AEAD
DHE-RSA-AES128-SHA256   TLSv1.2 Kx=DH       Au=RSA  Enc=AES(128)
Mac=SHA256
DHE-RSA-AES128-SHA      SSLv3 Kx=DH       Au=RSA  Enc=AES(128)  Mac=SHA1
DHE-RSA-CAMELLIA128-SHA256 TLSv1.2 Kx=DH       Au=RSA  Enc=Camellia(128)
Mac=SHA256
DHE-RSA-CAMELLIA128-SHA SSLv3 Kx=DH       Au=RSA  Enc=Camellia(128)
Mac=SHA1
```

```
AES128-GCM-SHA256        TLSv1.2 Kx=RSA      Au=RSA  Enc=AESGCM(128)
Mac=AEAD
AES128-SHA256            TLSv1.2 Kx=RSA      Au=RSA  Enc=AES(128)
Mac=SHA256
AES128-SHA               SSLv3 Kx=RSA      Au=RSA  Enc=AES(128)  Mac=SHA1
CAMELLIA128-SHA256       TLSv1.2 Kx=RSA      Au=RSA  Enc=Camellia(128)
Mac=SHA256
CAMELLIA128-SHA          SSLv3 Kx=RSA      Au=RSA  Enc=Camellia(128)
Mac=SHA1
ECDHE-RSA-RC4-SHA        SSLv3 Kx=ECDH     Au=RSA  Enc=RC4(128)  Mac=SHA1
ECDHE-ECDSA-RC4-SHA      SSLv3 Kx=ECDH     Au=ECDSA Enc=RC4(128)
Mac=SHA1
RC4-SHA                  SSLv3 Kx=RSA      Au=RSA  Enc=RC4(128)  Mac=SHA1
RC4-MD5                  SSLv3 Kx=RSA      Au=RSA  Enc=RC4(128)  Mac=MD5
ECDHE-RSA-DES-CBC3-SHA   SSLv3 Kx=ECDH     Au=RSA  Enc=3DES(168) Mac=SHA1
ECDHE-ECDSA-DES-CBC3-SHA SSLv3 Kx=ECDH     Au=ECDSA Enc=3DES(168)
Mac=SHA1
EDH-RSA-DES-CBC3-SHA     SSLv3 Kx=DH       Au=RSA  Enc=3DES(168) Mac=SHA1
DES-CBC3-SHA             SSLv3 Kx=RSA      Au=RSA  Enc=3DES(168) Mac=SHA1
```

2. web 服务器 ssl 证书

2.1. Nginx

```
$ sudo openssl req -new -x509 -keyout server.pem -out
server.pem -days 365 -nodes
```

指定证书位数为4096

```
# openssl req -x509 -nodes -days 1825 -newkey rsa:4096 -keyout
/etc/nginx/ssl/api.netkiller.cn.key -out
/etc/nginx/ssl/api.netkiller.cn.crt
```

Nginx + Tomcat (HTTP2)

```
upstream api.netkiller.cn {
    server 127.0.0.1:7000;
    server api2.netkiller.cn backup;
}

server {
    listen       80;
    listen 443 ssl http2;
    server_name api.cfd88.com api.netkiller.cn;

    ssl_protocols       TLSv1 TLSv1.1 TLSv1.2;
    ssl_ciphers         AES128-SHA:AES256-SHA:RC4-SHA:DES-CBC3-
SHA:RC4-MD5;
    ssl_certificate     ssl/api.netkiller.cn.crt;
    ssl_certificate_key ssl/api.netkiller.cn.key;
    ssl_session_cache   shared:SSL:30m;
    ssl_session_timeout 60m;

    charset utf-8;
    access_log  /var/log/nginx/api.netkiller.cn.access.log;
    error_log  /var/log/nginx/api.netkiller.cn.error.log;

    location / {
        proxy_pass http://api.netkiller.cn;
        proxy_http_version 1.1;
        proxy_set_header    Host    $host;
        proxy_set_header    X-Real-IP   $remote_addr;
        proxy_set_header    X-Forwarded-For
$proxy_add_x_forwarded_for;
        proxy_ignore_client_abort  on;
    }
}
```

3. s_server / s_client

3.1. SSL POP3 / SMTP / IMAP

SSL POP3 / SMTP / IMAP 端口号

```
POP3 995
SMTP 465
IMAP 993
```

```
openssl s_client -connect localhost:110 -starttls pop3
```

如果提示 CONNECTED(00000003) 侧省去 -starttls pop3 选项

```
openssl s_client -connect pop.163.com:995
```

```
openssl s_client -connect smtp.163.com:465
```

```
openssl s_client -connect imap.163.com:993
```

```
neo@MacBook-Pro-Neo ~ % openssl s_client -starttls smtp -
connect smtp.qq.com:587
CONNECTED(00000005)
depth=2 C = BE, O = GlobalSign nv-sa, OU = Root CA, CN =
GlobalSign Root CA
verify return:1
depth=1 C = BE, O = GlobalSign nv-sa, CN = GlobalSign
Organization Validation CA - SHA256 - G2
verify return:1
depth=0 C = CN, ST = guangdong, L = shenzhen, O = Tencent
```

```
Technology (Shenzhen) Company Limited, CN = *.mail.qq.com
verify return:1
---
Certificate chain
 0 s:/C=CN/ST=guangdong/L=shenzhen/O=Tencent Technology
(Shenzhen) Company Limited/CN=*.mail.qq.com
   i:/C=BE/O=GlobalSign nv-sa/CN=GlobalSign Organization
Validation CA - SHA256 - G2
 1 s:/C=BE/O=GlobalSign nv-sa/CN=GlobalSign Organization
Validation CA - SHA256 - G2
   i:/C=BE/O=GlobalSign nv-sa/OU=Root CA/CN=GlobalSign Root CA
 2 s:/C=BE/O=GlobalSign nv-sa/OU=Root CA/CN=GlobalSign Root CA
   i:/C=BE/O=GlobalSign nv-sa/OU=Root CA/CN=GlobalSign Root CA
---
Server certificate
-----BEGIN CERTIFICATE-----
```

```
MIIHBjCCBe6gAwIBAgIMQRECNeI6N/Pq0txeMA0GCSqGSIb3DQEBCwUAMGYxCzA
J
BgNVBAYTAkJFMRkwFwYDVQQKExBHbG9iYWxTaWduIG52LXNhMTwwOgYDVQQDEzN
H
bG9iYWxTaWduIE9yZ2FuaXphdGlvbiBWYWxpZGF0aW9uIENBIC0gU0hBMjU2IC0
g
RzIwHhcNMTkxMTExMTAzMjE2WhcNMjAwNjAzMDQwMDMzWjCBhDELMAkGA1UEBhM
C
Q04xEjAQBgNVBAgTCWd1YW5nZG9uZzERMA8GA1UEBxMIc2hlbnpoZW4xNjA0BgN
V
BAoTLVRlbmNlbnQgVGVjaG5vbG9neSAoU2hlbnpoZW4pIENvbXBhbnkgTGltaXR
l
ZDEWMBQGA1UEAwwNKi5tYWlsLnFxLmNvbTCCASIwDQYJKoZIhvcNAQEBBQADggE
P
ADCCAQoCggEBAMjn7wo/fZVfzKi9q7VOPZrSjTFFymgzS/TyJonILailwQMvL3n
e
R52n9NMVl9VbaIiJvdkzSunnrZrTOViqLdIODNsbiHCNeeskYV3bPgKIWU1LuNT
/
5LYcoR6qxX1X58sQttpxLE0TIVrcqKJBaCVXhoRnR5aRY1bXuaUkYCw0m3Jq1hT
3
em0iF5gTos4TAR3BMI/Z3sjACkB55WW/qDXx9uiG9P1HWIu8drq1SH4yrx9h2zY
A
yV6/s2CbNELwPUYHgSrbca3Sr9y+XCZocpECVLml5ZPO+ShbJHzWvztDz+ETZXZ
g
AD09mUOfrHgxXZDKvC47lawMT4+DQgc9DXECAwEAAaOCA5MwggOPMA4GA1UdDwE
B
/wQEAwIFoDCBoAYIKwYBBQUHAQEEgZMwgZAwTQYIKwYBBQUHMAKGQWh0dHA6Ly9
z
ZWN1cmUuZ2xvYmFsc2lnbi5jb20vY2FjZXJ0L2dzb3JnYW5pemF0aW9udmFsc2h
```

```
h
MmcycjEuY3J0MD8GCCsGAQUFBzABhjNodHRwOi8vb2NzcDIuZ2xvYmFsc2lnbi5
j
b20vZ3Nvcmdhbml6YXRpb252YWxzaGEyZzIwVgYDVR0gBE8wTTBBBgkrBgEEAaA
y
ARQwNDAyBggrBgEFBQcCARYmaHR0cHM6Ly93d3cuZ2xvYmFsc2lnbi5jb20vcmV
w
b3NpdG9yeS8wCAYGZ4EMAQICMAkGA1UdEwQCMAAwSQYDVR0fBEIwQDA+oDygOoY
4
aHR0cDovL2NybC5nbG9iYWxzaWduLmNvbS9ncy9nc29yZ2FuaXphdGlvbnZhbHN
o
YTJnMi5jcmwwgcMGA1UdEQSBuzCBuIINKi5tYWlsLnFxLmNvbYIOOTkzLmRhdi5
x
cS5jb22CDjk5My5lYXMucXEuY29tgg85OTMuaW1hcC5xcS5jb22CDjk5My5wb3A
u
cXEuY29tgg85OTMuc210cC5xcS5jb22CC2ltYXAucXEuY29tggpteDEucXEuY29
t
ggpteDIucXEuY29tggpteDMucXEuY29tggpwb3AucXEuY29tggtzbXRwLnFxLmN
v
bYILbWFpbC5xcS5jb20wHQYDVR0lBBYwFAYIKwYBBQUHAwEGCCsGAQUFBwMCMB8
G
A1UdIwQYMBaAFJbeYfG9HBYpUxzAzH07gwBA5hp8MB0GA1UdDgQWBBRL6XBdL20
t
FXnea3SBT+kdMMfp8TCCAQUGCisGAQQB1nkCBAIEgfYEgfMA8QB2AKS5CZC0GFg
U
h7sTosxncAo8NZgE+RvfuON3zQ7IDdwQAAABbloFfggAAAQDAEcwRQIgGxSwA4f
J
0EjOBQCIqJEZY44CB42NTjj+dTXyFrj+1FQCIQCMpCiQvkTxI4XdBrhT4U7tCQG
b
BC6xAUVP1TrDPVNCbAB3AMZSoOxIzrP8qxcJksQ6h0EzCegAZaJiUkAbozYqF8V
l
AAABbloFfi4AAAQDAEgwRgIhAOk6QzHNQHo9bTh5ALgZ05BcSpdoGdUzjDSG6KW
/
eejUAiEA2XMy8m3iQQyBw1oYj4GVKNGA1SsPrfKwTc+V8Wk0J1UwDQYJKoZIhvc
N
AQELBQADggEBACJ3IP+kzCWJTbsxo6wr0209CUPPDHAK2749OvYc59/xVNsOKwM
R
K+JLiiCr3V6WWjSouoZoGXRxcMZI/MFsjN2v0cIkLQSOzQNjYv3Gpm21M8dfMuc
M
WySQfzm0+iFmsBt91rGBVMJe+vrKk9bRFAU0X7v6ScpsbEKKZ9eM+xcqBy2LzMp
M
6sbPqmskfKUDy/2Ow46ivKiFjfRbaJDDnClisFFEtX50yJQpSGmNwBBw04gcarA
J
+tQxtx93Q9MjrRpO6z8c8JxyvMzq9k1gTwVs8K6Xpz0NKKPqs8K7uu2mQDcZptD
D
```

```
SB4IP+p0vlCJ8WzwoTP9WGEA9wvqNMwtPJo=
-----END CERTIFICATE-----
subject=/C=CN/ST=guangdong/L=shenzhen/O=Tencent Technology
(Shenzhen) Company Limited/CN=*.mail.qq.com
issuer=/C=BE/O=GlobalSign nv-sa/CN=GlobalSign Organization
Validation CA - SHA256 - G2
---
No client certificate CA names sent
Server Temp Key: ECDH, P-256, 256 bits
---
SSL handshake has read 4633 bytes and written 357 bytes
---
New, TLSv1/SSLv3, Cipher is ECDHE-RSA-AES128-GCM-SHA256
Server public key is 2048 bit
Secure Renegotiation IS supported
Compression: NONE
Expansion: NONE
No ALPN negotiated
SSL-Session:
    Protocol  : TLSv1.2
    Cipher    : ECDHE-RSA-AES128-GCM-SHA256
    Session-ID:
FABAC96B719F190C64B7CD0F6A140FD57E2D9917239370F813F1BD9547A91AA
5
    Session-ID-ctx:
    Master-Key:
FEF86566E6A588239A3779F721E7A22A7406611A4F419246F1695E435C4BBB6
D560F25CB18FC684FE15AD546798EC9BC
    Start Time: 1589379176
    Timeout   : 7200 (sec)
    Verify return code: 0 (ok)
---
250 8BITMIME
```

3.2. server / client 文件传输

生成证书

```
$ openssl req -new -x509 -keyout server.pem -out server.pem -
days 365 -nodes
```

在一个终端运行以下命令

```
openssl s_server -accept 2009 -key server.pem -cert server.pem
```

在另外一个终端运行命令如下

```
openssl s_client -connect localhost:2009
```

例 7.2. 加密传输文件

现在我们来尝试使用使用 openssl 加密传输文件

传输 /etc/passwd 文件

```
$ cat /etc/passwd | openssl s_server -accept 2009 -key
server.pem -cert server.pem
```

输出类似

```
$ cat /etc/passwd | openssl s_server -accept 2009 -key
server.pem -cert server.pem
Using default temp DH parameters
Using default temp ECDH parameters
ACCEPT
bad gethostbyaddr
DONE
shutdown accept socket
shutting down SSL
CONNECTION CLOSED
   0 items in the session cache
   0 client connects (SSL_connect())
   0 client renegotiates (SSL_connect())
   0 client connects that finished
   1 server accepts (SSL_accept())
   0 server renegotiates (SSL_accept())
   1 server accepts that finished
   0 session cache hits
   0 session cache misses
   0 session cache timeouts
   0 callback cache hits
   0 cache full overflows (128 allowed)
```

另一个服务器上运行

```
openssl s_client -connect 192.168.6.2:2009
```

输出类似

```
# openssl s_client -connect 192.168.6.2:2009
CONNECTED(00000003)
depth=0 C = AU, ST = Some-State, O = Internet Widgits Pty Ltd
verify error:num=18:self signed certificate
verify return:1
depth=0 C = AU, ST = Some-State, O = Internet Widgits Pty Ltd
verify error:num=9:certificate is not yet valid
notBefore=Sep  2 06:59:06 2013 GMT
verify return:1
depth=0 C = AU, ST = Some-State, O = Internet Widgits Pty Ltd
notBefore=Sep  2 06:59:06 2013 GMT
verify return:1
---
Certificate chain
 0 s:/C=AU/ST=Some-State/O=Internet Widgits Pty Ltd
   i:/C=AU/ST=Some-State/O=Internet Widgits Pty Ltd
---
Server certificate
-----BEGIN CERTIFICATE-----
MIIDXTCCAkWgAwIBAgIJAM1t1q1Hl5eUMA0GCSqGSIb3DQEBBQUAMEUxCzAJBgN
V
BAYTAkFVMRMwEQYDVQQIDApTb21lLVN0YXRlMSEwHwYDVQQKDBhJbnRlcm5ldCB
X
aWRnaXRzIFB0eSBMdGQwHhcNMTMwOTAyMDY1OTA2WhcNMTQwOTAyMDY1OTA2WjB
F
MQswCQYDVQQGEwJBVTETMBEGA1UECAwKU29tZS1TdGF0ZTEhMB8GA1UECgwYSW5
0
```

```
ZXJuZXQgV2lkZ2l0cyBQdHkgTHRkMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMII
B
CgKCAQEAvGWRExTsfte2ys8LYELMpznAEsc11CwPBgE81DgQNxswCyIY2EzhlvX
6
gnv4x+JttexdU1hXTSBY+eZwQmAP9RpJnX+dIxTOPdpgsJQd4SYn2uI1OWWhs0H
O
108DPsxx7WvlCIsLY6sJCGkJYnX0P4DIGNYU0KZSPY9dSSa6QPB2TKLaWwiRXWJ
q
m++1N4DF+LAbQb7gPwwacbBKMv8U4ZY4bmLxgQdPa2WahlSTMnwrntQv7+gkLL7
R
snILrXhoEalP1EaOr5awM0CdxT5SaIQwgKGv+5Vssw8KgnzNAtKaHw6uc/jgPGt
9
j6Qpo8+io+yMjypyi7FwEje4Rzl3SQIDAQABo1AwTjAdBgNVHQ4EFgQUFRScMNS
C
tHb8KbDilgijJ2mz2BAwHwYDVR0jBBgwFoAUFRScMNSCtHb8KbDilgijJ2mz2BA
w
DAYDVR0TBAUwAwEB/zANBgkqhkiG9w0BAQUFAAOCAQEAQANVwx4rMFPBtlHiWSO
U
wBt2XZvnSfarBpb/A2hWexzXQey9urKH8/8egKgxOCFhI42E2fH6RFhtI7x3CU6
i
1QQwKis9ZIiEEcn9inM0ZJOnaOx2gr/fcXnzKPWZFibAQP6gyGV/EQBCJ0j395c
Q
rHEfpfdKBPb5YN+NxXK1wHIIFV01lcZH2GDwDNDPtRNas/JNbS8X1iA8ti1VZnD
p
pSm8eZrzdJWsIQ/YFRNI/1mklSJr44NuvrbE7ivulBFpeIitc9ppkVa3xzhxM0x
l
cWz6l/jr3Dil5qWcCKsEZ0Hd0sZHuXm5eNJwwTO0XXT+vxJDM8Gf5fMqwx5VdUW
Z
uA==
-----END CERTIFICATE-----
subject=/C=AU/ST=Some-State/O=Internet Widgits Pty Ltd
issuer=/C=AU/ST=Some-State/O=Internet Widgits Pty Ltd
---
No client certificate CA names sent
---
SSL handshake has read 1583 bytes and written 246 bytes
---
New, TLSv1/SSLv3, Cipher is DHE-RSA-AES256-SHA
Server public key is 2048 bit
Secure Renegotiation IS supported
Compression: NONE
Expansion: NONE
SSL-Session:
    Protocol  : TLSv1
    Cipher    : DHE-RSA-AES256-SHA
    Session-ID:
7CA47FFBFC896FC90F7E9E5F3147BC9621C07E10882A7C7831BFA7D61AD24EE
F
    Session-ID-ctx:
    Master-Key:
5CB630D741EA2D209E0DC882A2E5C16E2009138A7DB7920ABEFD1E9CC5D6973
F7DC7228295B5AC75F5E7CD1726DC3E5F
    Key-Arg   : None
    Krb5 Principal: None
    PSK identity: None
    PSK identity hint: None
    TLS session ticket lifetime hint: 300 (seconds)
    TLS session ticket:
    0000 - 7d 76 b1 eb bb 9d 63 49-fe 9f 18 c0 78 82 66 bd
}v....cI....x.f.
    0010 - 65 69 ac 27 11 63 05 8a-57 8d 13 23 d8 85 3c fa
ei.'.c..W..#..<.
    0020 - 6b 54 4c 39 92 c4 53 22-16 e3 73 98 a0 fe 15 67
kTL9..S"..s....g
    0030 - c1 5f 47 66 f9 42 50 f5-67 be 91 a8 70 fa ef eb
._Gf.BP.g...p...
    0040 - 1c 51 c2 94 62 ff b0 97-1b 7b de ac 3a c8 39 52
.Q..b....{..:.9R
    0050 - 85 d6 51 02 33 48 2c 39-fc db f8 55 87 c5 1b 58
..Q.3H,9...U...X
    0060 - 81 e7 00 0b 9d ae e3 fd-04 dc 0d dd 26 20 3c b2
............& <.
    0070 - b2 0f 56 e1 7c be d2 89-2a 64 42 b4 9f eb b3 e2
..V.|...*dB.....
    0080 - ee 3d 51 ac 3f 9e 14 49-52 f4 b6 d7 9f 59 0b c8
.=Q.?..IR....Y..
    0090 - fa f2 74 38 e0 c8 12 1a-b3 81 e8 2f 13 cf 44 44
..t8......./..DD

    Start Time: 1378104227
    Timeout   : 300 (sec)
    Verify return code: 9 (certificate is not yet valid)
---
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
```

```
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System
(admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
syslog:x:101:103::/home/syslog:/bin/false
messagebus:x:102:105::/var/run/dbus:/bin/false
whoopsie:x:103:106::/nonexistent:/bin/false
landscape:x:104:109::/var/lib/landscape:/bin/false
sshd:x:105:65534::/var/run/sshd:/usr/sbin/nologin
neo:x:1000:1000:neo,,,:/home/neo:/bin/bash
ntop:x:106:114::/var/lib/ntop:/bin/false
redis:x:107:116:redis server,,,:/var/lib/redis:/bin/false
postgres:x:108:117:PostgreSQL
administrator,,,:/var/lib/postgresql:/bin/bash
colord:x:109:120:colord colour management
daemon,,,:/var/lib/colord:/bin/false
mysql:x:110:121:MySQL Server,,,:/nonexistent:/bin/false
zookeeper:x:111:122:ZooKeeper,,,:/var/lib/zookeeper:/bin/false
read:errno=0
```

3.3. HTTP SSL 证书

证书链

```
[www@netkiller ~]$ openssl s_client -connect www.google.com:443
-state
CONNECTED(00000003)
SSL_connect:before/connect initialization
SSL_connect:SSLv2/v3 write client hello A
SSL_connect:SSLv3 read server hello A
depth=3 C = US, O = Equifax, OU = Equifax Secure Certificate
```

```
Authority
verify return:1
depth=2 C = US, O = GeoTrust Inc., CN = GeoTrust Global CA
verify return:1
depth=1 C = US, O = Google Inc, CN = Google Internet Authority
G2
verify return:1
depth=0 C = US, ST = California, L = Mountain View, O = Google
Inc, CN = www.google.com
verify return:1
SSL_connect:SSLv3 read server certificate A
SSL_connect:SSLv3 read server key exchange A
SSL_connect:SSLv3 read server done A
SSL_connect:SSLv3 write client key exchange A
SSL_connect:SSLv3 write change cipher spec A
SSL_connect:SSLv3 write finished A
SSL_connect:SSLv3 flush data
SSL_connect:SSLv3 read server session ticket A
SSL_connect:SSLv3 read finished A
---
Certificate chain
 0 s:/C=US/ST=California/L=Mountain View/O=Google
Inc/CN=www.google.com
   i:/C=US/O=Google Inc/CN=Google Internet Authority G2
 1 s:/C=US/O=Google Inc/CN=Google Internet Authority G2
   i:/C=US/O=GeoTrust Inc./CN=GeoTrust Global CA
 2 s:/C=US/O=GeoTrust Inc./CN=GeoTrust Global CA
   i:/C=US/O=Equifax/OU=Equifax Secure Certificate Authority
---
Server certificate
-----BEGIN CERTIFICATE-----
MIIEgDCCA2igAwIBAgIISCr6QCbz5rowDQYJKoZIhvcNAQELBQAwSTELMAkGA1U
E
BhMCVVMxEzARBgNVBAoTCkdvb2dsZSBJbmMxJTAjBgNVBAMTHEdvb2dsZSBJbnR
l
cm5ldCBBdXRob3JpdHkgRzIwHhcNMTYxMjE1MTQwNzU2WhcNMTcwMzA5MTMzNTA
w
WjBoMQswCQYDVQQGEwJVUzETMBEGA1UECAwKQ2FsaWZvcm5pYTEWMBQGA1UEBww
N
TW91bnRhaW4gVmlldzETMBEGA1UECgwKR29vZ2xlIEluYzEXMBUGA1UEAwwOd3d
3
Lmdvb2dsZS5jb20wggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQC8vLL
z
GhY7xvadKOHvjpKbE7Kue1CP8LTgNo0JAOSEUVd/bDll8KEgyTc2ZOEGZPJ2biu
X
```

```
SvtOWqg1+Q1zxev8/5Ym0OS7xqLZH+6wVY+trJlka2VZ3oGkF8jmNW4hofJK0tn
D
v4gyG0d9AOjXCzCY/HSzGYA6oR6hdxfjnHkbwspPWfvyvQ1fxuMAzS6mTl2x6Dd
A
JUo1I+BVS54gAze3/kHoamovRHZyOn4dp2wkCv3eXRu4Eh8ZT3XWTie25jcnNhQ
R
tDvBqtlPtsFPUUhfonRGkUNojGIiFL6UdkfOIo/mlv5BQYWdqRCaCW78vUP6Tca
j
VZqeB4v5sR7O0SJJAgMBAAGjggFLMIIBRzAdBgNVHSUEFjAUBggrBgEFBQcDAQY
I
KwYBBQUHAwIwGQYDVR0RBBIwEIIOd3d3Lmdvb2dsZS5jb20waAYIKwYBBQUHAQE
E
XDBaMCsGCCsGAQUFBzAChh9odHRwOi8vcGtpLmdvb2dsZS5jb20vR0lBRzIuY3J
0
MCsGCCsGAQUFBzABhh9odHRwOi8vY2xpZW50czEuZ29vZ2xlLmNvbS9vY3NwMB0
G
A1UdDgQWBBS7Scfe9bno5yvK3NosrZJ6/SZVvTAMBgNVHRMBAf8EAjAAMB8GA1U
d
IwQYMBaAFErdBhYbvPZotXb1gba7Yhq6WoEvMCEGA1UdIAQaMBgwDAYKKwYBBAH
W
eQIFATAIBgZngQwBAgIwMAYDVR0fBCkwJzAloCOgIYYfaHR0cDovL3BraS5nb29
n
bGUuY29tL0dJQUcyLmNybDANBgkqhkiG9w0BAQsFAAOCAQEAlM1mVYPxFn1G2GY
h
BuzGnXwcK8H2T7c+zQGtab2hgWp8lvWcJ/O0PPb7XfXVIx+umAQUJ9Vx/3gUHLN
H
hN0k+ElUSSAIagKgx/tg+S9GizsWM926tqXdq6JpBLJr9nE5zg9/TE9kI7Ycplx
9
rAqYyqJG13a6xzde+Y2Ua8bvqgtPvte9cvqU4HULBptsHLAhMDe/ln5CsI6EK3U
C
cb9reU8in8yCaH8dtzrFyUracpMureWnBeajOYXRPTdCFccejAh/xyH5SKDOOZ4
v
3TP9GBtClAH1mSXoPhX73dp7jipZqgbY4kiEDNx+hformTUFBDHD0eO/s2nqwuW
L
pBH6XQ==
-----END CERTIFICATE-----
subject=/C=US/ST=California/L=Mountain View/O=Google
Inc/CN=www.google.com
issuer=/C=US/O=Google Inc/CN=Google Internet Authority G2
---
No client certificate CA names sent
Server Temp Key: ECDH, prime256v1, 256 bits
---
SSL handshake has read 3727 bytes and written 373 bytes
---
```

```
New, TLSv1/SSLv3, Cipher is ECDHE-RSA-AES128-GCM-SHA256
Server public key is 2048 bit
Secure Renegotiation IS supported
Compression: NONE
Expansion: NONE
SSL-Session:
    Protocol  : TLSv1.2
    Cipher    : ECDHE-RSA-AES128-GCM-SHA256
    Session-ID:
E90DBF6A7E78AAA9499388799l3996225FE815F91B34A65BA9C84CDFD222EB6
C
    Session-ID-ctx:
    Master-Key:
ED751A4B1BCC2EB08AF01A69F5474960E289EC77065C84FEB6E93C0923834DC
03265F8B1CFD3AED0454EDB6CE7855AB6
    Key-Arg   : None
    Krb5 Principal: None
    PSK identity: None
    PSK identity hint: None
    TLS session ticket lifetime hint: 100800 (seconds)
    TLS session ticket:
    0000 - 60 81 b9 6b 8a 3b 30 0f-50 bc 0b 16 de 4b b2 e3
`..k.;0.P....K..
    0010 - df b1 67 c1 28 2a 9c 2d-fc 64 76 f8 3f f0 a3 b1
..g.(*.-.dv.?...
    0020 - e0 70 5a 7a b8 2b 08 80-77 0d 21 e8 b8 82 fc 66
.pZz.+..w.!....f
    0030 - df c4 c0 da a5 6a 8f f8-66 05 0c 22 07 5c a4 3b
.....j..f..".\.;
    0040 - d8 af 31 37 28 6f 8c 2f-24 2d c0 40 f5 0d 6c da
..17(o./$-.@..l.
    0050 - c6 10 6e bf 16 55 8e 98-14 c8 ff 6a b6 22 51 f7
..n..U.....j."Q.
    0060 - 5b c0 11 ed 04 d0 62 40-e2 ad a5 9f 93 69 2b 72
[.....b@.....i+r
    0070 - e0 ff 8f 34 5f 78 0c 58-e4 a6 6a 08 11 f9 da d4
...4_x.X..j.....
    0080 - f4 1a 6e 1f b6 ff 2b 60-3b de 7e 57 fb 9a 79 33
..n...+`;.~W..y3
    0090 - 1f bd 92 d8 ae df 1d 0a-53 20 cd 9c 37 a9 e3 83
........S ..7...
    00a0 - 1c 72 84 30
.r.0

    Start Time: 1482905312
    Timeout   : 300 (sec)
    Verify return code: 0 (ok)
---
```

注意下面证书链，通常有三级，根证书，中级证书，服务器证书

```
---
Certificate chain
 0 s:/C=US/ST=California/L=Mountain View/O=Google
Inc/CN=www.google.com
   i:/C=US/O=Google Inc/CN=Google Internet Authority G2
 1 s:/C=US/O=Google Inc/CN=Google Internet Authority G2
   i:/C=US/O=GeoTrust Inc./CN=GeoTrust Global CA
 2 s:/C=US/O=GeoTrust Inc./CN=GeoTrust Global CA
   i:/C=US/O=Equifax/OU=Equifax Secure Certificate Authority
---
```

GeoTrust Global CA 是根证书上

Google Internet Authority G2 中级证书

www.google.com 是服务器证书

提示

没有根证书WEB浏览器通常是可以正常访问的，因为证书厂商已经根微软签了协议，根证书已经安装到了Windows中。

开发中会遇到一些问题例如JDK他又自己的根证书管里，很多厂商的根证书没有根Oracle签协议并放到java/jre/lib/security/cacerts中，这是代码访问https服务器就不信任这些厂商的证书。

显示证书

```
$ openssl s_client -connect www.google.com:443 -showcerts
```

指定 servername

默认s_client使用IP地址链接并不会推送HTTP的HOST头，如果链接的是虚拟机就会有麻烦。

```
$ openssl s_client -servername api.netkiller.com -connect
api.netkiller.com:443
```

4. smime

5. Outlook smime x509 证书

5.1. 快速创建自签名证书

以下适合个人使用

```
openssl genrsa -out ca.pem 1024
openssl req -new -out neo.csr -key ca.pem
openssl x509 -req -in neo.csr -out neo.cer -signkey ca.pem -
days 365
openssl pkcs12 -export -clcerts -in neo.cer -inkey ca.pem -out
neo.p12
```

安装cer与p12两个证书，然后打开outlook测试

例 7.3. 快速创建自签名证书

```
                                <![CDATA[
[root@localhost smime]# openssl genrsa -out ca/ca.pem 1024
Generating RSA private key, 1024 bit long modulus
...............++++++
...................++++++
e is 65537 (0x10001)

[root@localhost smime]# openssl req -new -out ca/ca.csr -key
ca/ca.pem
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) [XX]:CN
State or Province Name (full name) []:GD
Locality Name (eg, city) [Default City]:SZ
```

```
Organization Name (eg, company) [Default Company Ltd]:XXX Ltd
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []:neo
Email Address []:neo.chan@live.com

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

[root@localhost smime]# openssl x509 -req -in ca/ca.csr -out
ca/ca-cert.cer -signkey ca/ca.pem -days 365
Signature ok
subject=/C=CN/ST=GD/L=SZ/O=XXX
Ltd/CN=neo/emailAddress=neo.chan@live.com
Getting Private key

[root@localhost smime]# openssl pkcs12 -export -clcerts -in
ca/ca-cert.cer -inkey ca/ca.pem -out ca/ca.p12
Enter Export Password:
Verifying - Enter Export Password:
```

更便捷的方法

```
openssl genrsa -out ca.pem 1024
openssl req -new -out neo.csr -key ca.pem -subj
"/C=CN/ST=GD/L=SZ/O=Internet Widgits Pty
Ltd/OU=IT/CN=neo/emailAddress=neo@668x.net"
openssl x509 -req -in neo.csr -out neo.cer -signkey ca.pem -
days 365
openssl pkcs12 -export -in neo.cer -inkey ca.pem -out neo.p12 -
name "neo"
```

5.2. 企业或集团方案

证书环境

```
% mkdir keys
% cd keys/
```

建立空文件 index.txt 用来保存以后的证书信息，这是OpenSSL的证书数据库：

```
touch  index.txt
```

建立一个文件 serial 在文件中输入一个数字，做为以后颁发证书的序列号，颁发证书序列号就从你输入的数字开始递增：

```
echo 01 > serial
```

颁发CA证书

首先创建CA根证书私钥文件，使用RSA格式，1024位：

```
% openssl genrsa -des3 -out ca.key 1024
```

例 7.4. 创建CA根证书

```
% openssl genrsa -des3 -out ca.key 1024
Generating RSA private key, 1024 bit long modulus
...........................++++++
...........................................++++++
e is 65537 (0x10001)
Enter pass phrase for ca.key:
Verifying - Enter pass phrase for ca.key:
```

私钥在建立时需要输入一个密码用来保护私钥文件，私钥文件使用3DES加密；也可以不进行加密，这样不安全，因为一旦ca证书遗失，别人就可以随意颁发用户证书：

```
openssl genrsa -out ca.key 1024
```

利用建立RSA私钥，为CA自己建立一个自签名的证书文件：

```
openssl req -new -x509 -days 365 -key ca.key -out ca.crt
```

生成证书的过程中需要输入证书的信息，

例 7.5. 创建自签名的证书

```
% openssl req -new -x509 -days 365 -key ca.key -out ca.crt
Enter pass phrase for ca.key:
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]:CN
State or Province Name (full name) [Some-State]:GD
Locality Name (eg, city) []:Shenzhen
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Your
Company Ltd
Organizational Unit Name (eg, section) []:IT
Common Name (e.g. server FQDN or YOUR name) []:Neo Chan
Email Address []:neo.chan@live.com
```

颁发客户证书

生成客户证书的私钥文件，与生成CA根证书文件的方法一样，

```
openssl genrsa -des3 -out client.key 1024
```

OpenSSL生成客户端证书的时候，不能直接生成证书，而是必须通过证书请求文件来生成，因此现在我们来建立客户端的证书请求文件，生成的过程中一样要输入客户端的信息：

```
openssl req -new -key client.key -out client.csr
```

有了证书请求文件之后，就可以使用CA的根证书、根私钥来对请求文件进行签名，生成客户端证书 client.pem 了：

```
openssl x509 -req -in client.csr -out client.crt -signkey
client.key -CA ca.crt -CAkey ca.key -days 365 -CAserial serial
```

```
openssl pkcs12 -export -clcerts -in client.crt -inkey
client.key -out client.p12
```

注意

到这里为止，根CA为客户端签发证书的过程就结束了。

吊销已签发的证书

使用ca中的 -revoke 命令：

```
openssl ca -revoke client.pem -keyfile ca.key -cert ca.crt
```

证书被吊销之后，还需要发布新的CRL文件：

```
openssl ca -gencrl  -out ca.crl -keyfile ca.key -cert ca.crt
```

6. 证书转换

PKCS 全称是 Public-Key Cryptography Standards ，是由 RSA 实验室与其它安全系统开发商为促进公钥密码的发展而制订的一系列标准，PKCS 目前共发布过 15 个标准。常用的有:
PKCS#7 Cryptographic Message Syntax Standard
PKCS#10 Certification Request Standard
PKCS#12 Personal Information Exchange Syntax Standard
X.509是常见通用的证书格式。所有的证书都符合为 Public Key Infrastructure (PKI) 制定的 ITU-T X509 国际标准。
PKCS#7 常用的后缀是： .P7B .P7C .SPC
PKCS#12 常用的后缀有： .P12 .PFX
X.509 DER 编码(ASCII)的后缀是： .DER .CER .CRT
X.509 PAM 编码(Base64)的后缀是： .PEM .CER .CRT
.cer/.crt是用于存放证书，它是2进制形式存放的，不含私钥。
.pem跟crt/cer的区别是它以Ascii来表示。
pfx/p12用于存放个人证书/私钥，他通常包含保护密码，2进制方式
p10是证书请求
p7r是CA对证书请求的回复，只用于导入
p7b以树状展示证书链(certificate chain)，同时也支持单个证书，不含私钥。

6.1. CA证书

用openssl创建CA证书的RSA密钥(PEM格式)：

```
openssl genrsa -des3 -out ca.key 1024
```

6.2. 创建CA证书有效期为一年

用openssl创建CA证书(PEM格式,假如有效期为一年)：

```
openssl req -new -x509 -days 365 -key ca.key -out ca.crt -
config openssl.cnf
```

openssl是可以生成DER格式的CA证书的，最好用IE将PEM格式的CA证书转换成DER格式的CA证书。

6.3. x509转换为pfx

```
openssl pkcs12 -export -out server.pfx -inkey server.key -in
server.crt
```

6.4. PEM格式的ca.key转换为Microsoft可以识别的pvk格式

```
pvk -in ca.key -out ca.pvk -nocrypt -topvk
```

6.5. PKCS#12 到 PEM 的转换

```
openssl pkcs12 -nocerts -nodes -in cert.p12 -out private.pem
验证
openssl pkcs12 -clcerts -nokeys -in cert.p12 -out cert.pem
```

6.6. 从 PFX 格式文件中提取私钥格式文件 (.key)

```
openssl pkcs12 -in mycert.pfx -nocerts -nodes -out mycert.key
```

6.7. 转换 pem 到到 spc

```
openssl crl2pkcs7 -nocrl -certfile venus.pem  -outform DER -out
```

```
venus.spc
```

用 -outform -inform 指定 DER 还是 PAM 格式。例如：

```
openssl x509 -in Cert.pem -inform PEM -out cert.der -outform
DER
```

6.8. PEM 到 PKCS#12 的转换

```
openssl pkcs12 -export -in Cert.pem -out Cert.p12 -inkey
key.pem
```

IIS 证书

```
cd c:\openssl
set OPENSSL_CONF=openssl.cnf
openssl pkcs12 -export -out server.pfx -inkey server.key -in
server.crt
```

server.key和server.crt文件是Apache的证书文件，生成的server.pfx用于导入IIS

6.9. How to Convert PFX Certificate to PEM Format for SOAP

```
$ openssl pkcs12 -in test.pfx -out client.pem
Enter Import Password:
MAC verified OK
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
```

6.10. DER文件（.crt .cer .der）转为PEM格式文件

```
转换DER文件(一般后缀名是.crt .cer .der的文件)到PEM文件
openssl x509 -inform der -in certificate.cer -out
certificate.pem
转换PEM文件到DER文件
openssl x509 -outform der -in certificate.pem -out
certificate.der
```

6.11. JKS 转 X509

```
keytool -list -rfc --keystore netkiller.jks | openssl x509 -
inform pem -pubkey
```

6.12. jks to pem

转换流程 jks - p12 - pem

```
keytool -keystore foo.jks -genkeypair -alias foo \
    -dname 'CN=foo.example.com,L=Melbourne,ST=Victoria,C=AU'

keytool -importkeystore -srckeystore foo.jks \
   -destkeystore foo.p12 \
   -srcalias foo \
   -srcstoretype jks \
   -deststoretype pkcs12

openssl pkcs12 -in foo.p12 -out foo.pem
```

```
neo@MacBook-Pro ~/test/test % ls
foo.jks foo.p12 foo.pem
```

查看证书

```
keytool -keystore foo.jks -exportcert -alias foo | openssl x509
```

```
-inform der -text

openssl x509 -text -in foo.pem

openssl dsa -text -in foo.pem
```

7. 其他证书工具

```
sudo apt-get install sslscan
```

```
                   _
           ___ ___| |___  ___ __ _ _ __
          / __/ __| / __|/ __/ _` | '_ \
          \__ \__ \ \__ \ (_| (_| | | | |
          |___/___/_|___/\___\__,_|_| |_|

                  Version 1.8.2
             http://www.titania.co.uk
        Copyright Ian Ventura-Whiting 2009
```

8. OpenSSL 开发库

8.1. DES encryption with OpenSSL

例 7.6. DES encryption example in C

```
#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <openssl/des.h>


char *
Encrypt( char *Key, char *Msg, int size)
{

        static char*    Res;
        int             n=0;
        DES_cblock      Key2;
        DES_key_schedule schedule;

        Res = ( char * ) malloc( size );

        /* Prepare the key for use with DES_cfb64_encrypt */
        memcpy( Key2, Key,8);
        DES_set_odd_parity( &Key2 );
        DES_set_key_checked( &Key2, &schedule );

        /* Encryption occurs here */
        DES_cfb64_encrypt( ( unsigned char * ) Msg, ( unsigned
char * ) Res,
                           size, &schedule, &Key2, &n,
DES_ENCRYPT );

         return (Res);
}


char *
Decrypt( char *Key, char *Msg, int size)
```

```
{

        static char*    Res;
        int             n=0;

        DES_cblock      Key2;
        DES_key_schedule schedule;

        Res = ( char * ) malloc( size );

        /* Prepare the key for use with DES_cfb64_encrypt */
        memcpy( Key2, Key,8);
        DES_set_odd_parity( &Key2 );
        DES_set_key_checked( &Key2, &schedule );

        /* Decryption occurs here */
        DES_cfb64_encrypt( ( unsigned char * ) Msg, ( unsigned
char * ) Res,
                           size, &schedule, &Key2, &n,
DES_DECRYPT );

        return (Res);

}

int main() {

char key[]="password";
char clear[]="This is a secret message";
char *decrypted;
char *encrypted;

encrypted=malloc(sizeof(clear));
decrypted=malloc(sizeof(clear));

printf("Clear text\t : %s \n",clear);
memcpy(encrypted,Encrypt(key,clear,sizeof(clear)),
sizeof(clear));
printf("Encrypted text\t : %s \n",encrypted);
memcpy(decrypted,Decrypt(key,encrypted,sizeof(clear)),
sizeof(clear));
printf("Decrypted text\t : %s \n",decrypted);
return (0);
}
```

编译运行

```
$ gcc des.c -o des -lssl -lcrypto
$ ./des
```

第 8 章 数据库与加密

1. MySQL 加密函数

1.1. AES_ENCRYPT / AES_DECRYPT

简单用法

```
mysql> select AES_ENCRYPT('helloworld','key');
+----------------------------------+
| AES_ENCRYPT('helloworld','key') |
+----------------------------------+
|                                  |
+----------------------------------+
1 row in set (0.00 sec)

mysql> select
AES_DECRYPT(AES_ENCRYPT('helloworld','key'),'key');
+-----------------------------------------------------+
| AES_DECRYPT(AES_ENCRYPT('helloworld','key'),'key') |
+-----------------------------------------------------+
| helloworld                                          |
+-----------------------------------------------------+
1 row in set (0.00 sec)

mysql>
```

加密数据入库

```
CREATE TABLE `encryption` (
        `mobile` VARBINARY(16) NOT NULL,
        `key` VARCHAR(32) NOT NULL
)
```

```
ENGINE=InnoDB;

INSERT INTO encryption(`mobile`,`key`)VALUES(
AES_ENCRYPT('13691851789',md5('13691851789')),
md5('13691851789'))
select AES_DECRYPT(mobile,`key`), length(mobile) from
encryption;
```

1.2. 通过PHP mcrypt 函数加密解密MySQL数据库

由于AES_DECRYPT()与AES_ENCRYPT()会耗费一部们数据库资源，于是我想出在外部实现AES_DECRYPT/AES_ENCRYPT同时完全兼容mysql。

MySQL AES_ENCRYPT() 加密,通过 PHP mcrypt_decrypt() 解密

PHP mcrypt_encrypt 加密,通过MySQL AES_DECRYPT() 解密

```
<?php
$dbh = new PDO ( 'mysql:host=192.168.6.1;dbname=test', 'www',
'passw0rd' );
$dbh->setAttribute ( PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION
);
$dbh->exec ( 'set names utf8' );
$data = 'Helloworld!!!';
//$data = '123456789';
$key = 'neo';

$sql = "SELECT AES_ENCRYPT(:data, :key) as string";
$stmt = $dbh->prepare ( $sql );
$stmt->execute ( array (
                ':data' => $data,
                ':key' => $key
) );
$row = $stmt->fetch ( PDO::FETCH_ASSOC );
$encrypt = $row ['string'];
printf ( "MySQL AES Encrypt: %s \n", $encrypt );

$sql = "SELECT AES_DECRYPT(:data, :key) as string";
```

```
$stmt = $dbh->prepare ( $sql );
$stmt->execute ( array (
                ':data' => $encrypt,
                ':key' => $key
) );
$row = $stmt->fetch ( PDO::FETCH_ASSOC );

$decrypt = $row ['string'];
printf ( "MySQL AES Decrypt: %s \n", $decrypt );
printf ( "----------------------------------\n" );
function aes_decrypt($encrypted, $key) {
        return rtrim ( mcrypt_decrypt ( MCRYPT_RIJNDAEL_128,
$key, $encrypted, MCRYPT_MODE_ECB, '' ), "\x00..\x1F" );
}
function aes_encrypt($decrypted, $key) {
        return mcrypt_encrypt ( MCRYPT_RIJNDAEL_128, $key,
$decrypted, MCRYPT_MODE_ECB, '' );
}

printf ( "MySQL AES_ENCRYPT => PHP AES_Decrypt: %s => %s \n",
$encrypt, aes_decrypt ( $encrypt, $key ) );

$str = 'Test by neo';

$sql = "SELECT AES_DECRYPT(:data, :key) as string";
$stmt = $dbh->prepare ( $sql );
$stmt->execute ( array (
                ':data' => aes_encrypt ( $str, $key ),
                ':key' => $key
) );
$row = $stmt->fetch ( PDO::FETCH_ASSOC );

$decrypt = $row ['string'];
printf ( "PHP encrypt => MySQL Decrypt: %s => %s \n", $str,
$decrypt );

printf ( "PHP enctypt => PHP  Decrypt: %s => %s \n", $str,
aes_decrypt ( aes_encrypt ( $str, $key ), $key ) );
?>
```

第 9 章 Java - keytool

```
Keytool介绍
Keytool 是一个Java数据证书的管理工具 ,Keytool将密钥（key）和证书
（certificates）存在一个称为keystore的文件中在keystore里，包含两种数
据:密钥实体（Key entity）-密钥（secret key）或者是私钥和配对公钥（采用
非对称加密）可信任的证书实体（trusted certificate entries）-只包含公
钥.

JDK中keytool常用参数说明（不同版本有差异，详细可参见【附录】中的官方文档链
接）:

-genkey 在用户主目录
-genkey 在用户主目录中创建一个默认文件".keystore",还会产生一个mykey的别
名，mykey中包含用户的公钥、私钥和证书(在没有指定生成位置的情况下,keystore
会存在用户系统默认目录)
-alias 产生别名 每个keystore都关联这一个独一无二的alias，这个alias通常
不区分大小写
-keystore 指定密钥库的名称(产生的各类信息将不在.keystore文件中)
-keyalg 指定密钥的算法 (如 RSA DSA，默认值为：DSA)
-validity 指定创建的证书有效期多少天(默认 90)
-keysize 指定密钥长度 （默认 1024）
-storepass 指定密钥库的密码(获取keystore信息所需的密码)
-keypass 指定别名条目的密码(私钥的密码)
-dname 指定证书发行者信息 其中： "CN=名字与姓氏,OU=组织单位名称,O=组织名
称,L=城市或区域名 称,ST=州或省份名称,C=单位的两字母国家代码"
-list 显示密钥库中的证书信息 keytool -list -v -keystore 指定
keystore -storepass 密码
-v 显示密钥库中的证书详细信息
-export 将别名指定的证书导出到文件 keytool -export -alias 需要导出的
别名 -keystore 指定keystore -file 指定导出的证书位置及证书名称 -
storepass 密码
-file 参数指定导出到文件的文件名
-delete 删除密钥库中某条目 keytool -delete -alias 指定需删除的别 -
keystore 指定keystore – storepass 密码
-printcert 查看导出的证书信息 keytool -printcert -file
g:\sso\michael.crt
-keypasswd 修改密钥库中指定条目口令 keytool -keypasswd -alias 需修改
的别名 -keypass 旧密码 -new 新密码 -storepass keystore密码 -
keystore sage
```

```
-storepasswd 修改keystore口令 keytool -storepasswd -keystore
g:\sso\michael.keystore(需修改口令的keystore) -storepass
pwdold(原始密码) -new pwdnew(新密码)
-import 将已签名数字证书导入密钥库 keytool -import -alias 指定导入条
目的别名 -keystore 指定keystore -file 需导入的证书
中创建一个默认文件”.keystore”,还会产生一个mykey的别名，mykey中包含用户
的公钥、私钥和证书(在没有指定生成位置的情况下,keystore会存在用户系统默认目
录)
-alias 产生别名 每个keystore都关联这一个独一无二的alias，这个alias通常
不区分大小写
-keystore 指定密钥库的名称(产生的各类信息将不在.keystore文件中)
-keyalg 指定密钥的算法 (如 RSA DSA，默认值为：DSA)
-validity 指定创建的证书有效期多少天(默认 90)
-keysize 指定密钥长度 （默认 1024）
-storepass 指定密钥库的密码(获取keystore信息所需的密码)
-keypass 指定别名条目的密码(私钥的密码)
-dname 指定证书发行者信息 其中： “CN=名字与姓氏,OU=组织单位名称,O=组织名
称,L=城市或区域名 称,ST=州或省份名称,C=单位的两字母国家代码”
-list 显示密钥库中的证书信息 keytool -list -v -keystore 指定
keystore -storepass 密码
-v 显示密钥库中的证书详细信息
-export 将别名指定的证书导出到文件 keytool -export -alias 需要导出的
别名 -keystore 指定keystore -file 指定导出的证书位置及证书名称 -
storepass 密码
-file 参数指定导出到文件的文件名
-delete 删除密钥库中某条目 keytool -delete -alias 指定需删除的别 -
keystore 指定keystore – storepass 密码
-printcert 查看导出的证书信息 keytool -printcert -file
g:\sso\michael.crt
-keypasswd 修改密钥库中指定条目口令 keytool -keypasswd -alias 需修改
的别名 -keypass 旧密码 -new 新密码 -storepass keystore密码 -
keystore sage
-storepasswd 修改keystore口令 keytool -storepasswd -keystore
g:\sso\michael.keystore(需修改口令的keystore) -storepass
pwdold(原始密码) -new pwdnew(新密码)
-import 将已签名数字证书导入密钥库 keytool -import -alias 指定导入条
目的别名 -keystore 指定keystore -file 需导入的证书
```

1. 创建证书

```
keytool -genkey -keyalg RSA -keystore keys/server.keystore
Enter keystore password:  changeit
What is your first and last name?
  [Unknown]:  www.caucho.com
What is the name of your organizational unit?
  [Unknown]:  Resin Engineering
What is the name of your organization?
  [Unknown]:  Caucho Technology, Inc.
What is the name of your City or Locality?
  [Unknown]:  San Francisco
What is the name of your State or Province?
  [Unknown]:  California
What is the two-letter country code for this unit?
  [Unknown]:  US
Is <CN=www.caucho.com, OU=Resin Engineering,
  O="Caucho Technology, Inc.", L=San Francisco, ST=California,
C=US> correct?
  [no]:  yes

Enter key password for <mykey>
        (RETURN if same as keystore password):  changeit
```

2. Private key generation

```
keytool -genkey -keyalg RSA -alias myserverkeypair \
                -storepass YourPasswordHere -keystore
private.keystore
What is your first and last name?
  [Unknown]:  www.myserver.com
What is the name of your organizational unit?
  [Unknown]:  Foo Dept
What is the name of your organization?
  [Unknown]:  Bar
What is the name of your City or Locality?
  [Unknown]:  Paris
What is the name of your State or Province?
  [Unknown]:  France
What is the two-letter country code for this unit?
  [Unknown]:  FR
Is <CN=www.myserver.com, OU=Foo Dept, O=Bar, L=Paris,
                ST=France, C=FR> correct?
  [no]:  yes

Enter key password for <myserverkeypair>
        (RETURN if same as keystore password):
```

3. Public Key Certificate (optional)

```
>keytool -certreq -alias myserverkeypair -storepass
YourPasswordHere \
                -keystore private.keystore
-----BEGIN NEW CERTIFICATE REQUEST-----
MIIBqjCCARMCAQAwajELMAkGA1UEBhMCRlIxDzANBgNVBAgTBkZyYW5jZTEOMAw
GA1UEBxMFUGFy
... cut ...
KDYZTklbg1NOiXTdXIhPHb3+YOgZ+HoeDTxOx/rRhA==
-----END NEW CERTIFICATE REQUEST-----
```

4. import your signed certificate

```
keytool -import -alias servertest -storepass YourPasswordHere \
                -keystore private.keystore -file servertest.crt
```

5. Import the certificate and attach it to your server key pair

Import the certificate and attach it to your server key pair by typing the command

```
keytool -import -alias myserverkeypair -storepass
YourPasswordHere \
                -keystore private.keystore -file myserver.cer
Certificate reply was installed in keystore
```

6. Key pair verification

```
keytool -list -v -alias myserverkeypair -storepass
YourPasswordHere \
                -keystore private.keystore
```

第 10 章 .Net makecert

1. 访问X.509证书

Java访问X.509证书

第 11 章 Secure Tunnel

1. OpenSSH Tunnel

mysql tunnel

```
$ ssh -L 3306:127.0.0.1:3306 user@example.org
```

testing

```
$ mysql -h 127.0.0.1 -uroot -p test
```

1.1. SOCKS v5 Tunnel

```

ssh -D 1080 <远程主机地址>

```

Firefox 配置

为了防止所访问网站的DNS被窥探，可以在Firefox的地址栏中输入about:config 把network.proxy.socks_remote_dns 改为true

2. SSL Tunnel

http://www.stunnel.org/

2.1. 通过SSL访问POP、IMAP、SMTP

例 11.1. stunnel.conf

```
# Sample stunnel configuration file
# Copyright by Michal Trojnara 2002

# Comment it out on Win32
cert = /etc/stunnel/stunnel.pem
# chroot = /usr/var/run/stunnel/
# PID is created inside chroot jail
pid = /stunnel.pid
#setuid = nobody
#setgid = nogroup

setuid = root
setgid = root

# Workaround for Eudora bug
#options = DONT_INSERT_EMPTY_FRAGMENTS

# Authentication stuff
#verify = 2
# don't forget about c_rehash CApath
# it is located inside chroot jail:
#CApath = /certs
# or simply use CAfile instead:
#CAfile = /usr/etc/stunnel/certs.pem

# Some debugging stuff
debug = 7
output = stunnel.log

# Use it for client mode
#client = yes
```

```
# Service-level configuration

[pop3s]
accept  = 995
connect = 110

[imaps]
accept  = 993
connect = 143

[ssmtp]
accept  = 465
connect = 25

#[https]
#accept  = 443
#connect = 80
#TIMEOUTclose = 0

[nntps]
accept  = 563
connect = 119
```

```
# SMTP
/sbin/iptables -A INPUT -p tcp --dport 25 -j ACCEPT
# SMTPS
/sbin/iptables -A INPUT -p tcp --dport 465 -j ACCEPT
# POP3
/sbin/iptables -A INPUT -p tcp --dport 110 -j ACCEPT
# POP3S
/sbin/iptables -A INPUT -p tcp --dport 995 -j ACCEPT
# IMAP
/sbin/iptables -A INPUT -p tcp --dport 143 -j ACCEPT
# IMAPS
/sbin/iptables -A INPUT -p tcp --dport 993 -j ACCEPT
```

```
[root@linuxas3 stunnel]# nmap localhost

Starting nmap V. 3.00 ( www.insecure.org/nmap/ )
Interesting ports on linuxas3.9812.net (127.0.0.1):
```

```
(The 1582 ports scanned but not shown below are in state:
closed)
Port       State       Service
22/tcp     open        ssh
25/tcp     open        smtp
80/tcp     open        http
110/tcp    open        pop-3
111/tcp    open        sunrpc
119/tcp    open        nntp
143/tcp    open        imap2
443/tcp    open        https
465/tcp    open        smtps
563/tcp    open        snews
631/tcp    open        ipp
783/tcp    open        hp-alarm-mgr
993/tcp    open        imaps
995/tcp    open        pop3s
3306/tcp   open        mysql
5000/tcp   open        UPnP
5001/tcp   open        commplex-link
8009/tcp   open        ajp13
8080/tcp   open        http-proxy

Nmap run completed -- 1 IP address (1 host up) scanned in 3
seconds
[root@linuxas3 stunnel]#
```

3. DeleGate

http://www.delegate.org/delegate/

第 12 章 硬盘分区与文件系统加密

Harddisk Partition & File System

1. Microsoft 文件系统加密

1.1. Microsoft Encrypting File System (EFS)

http://www.microsoft.com/china/technet/security/sgk/protect_data_EFS.mspx

1.2. BitLocker

BitLocker 与 Encrypting File System (EFS) 的比较 http://windows.microsoft.com/zh-cn/windows7/whats-the-difference-between-bitlocker-drive-encryption-and-encrypting-file-system

第 13 章 Office

1. 给文档加密码

以Word为例,启动Word -> 工具 -> 选项 -> 安全性

常规操作,在这里输入:

打开密码

修改密码

这样的安全性对一般用户有效,对于电脑高手,他们可以下载一些破解工具,可以在几秒钟内破解你认为安全的密码.

我们现在让密码更安全一些,方法是点击"打开时的文件密码"后面的"高级"按钮

现在你可以看到默认的加密方式是"Offices 97/2000 兼容",这是一种不太安全加密方法

请在列表中选择RAS/DSS算法的加密类型,同时"选择密钥长度"输入大于128的倍数如:256,1024,2048...

单击"确定"按钮完成操作

上面的操作也不能保证万无一失,通过穷举算法加黑客字典,只要有足够的时间还可以猜到你的密码,这取决你密码长度和复杂度

2. 数字签名

开始菜单 -> 程序 -> Microsoft Office -> Microsoft Office 工具 -> VBA 项目的数字证书

位置:D:\Program Files\Microsoft Office\OFFICE11\SELFCERT.EXE

"您的证书名称" 中输入如:netkiller

单击"确定"按钮

以Word为例,启动Word 新键文档并输入一些内容或打开一个有内容的文档

Word -> 工具 -> 选项 -> 安全性 -> 数字签名

单击"添加"按钮

单击"是"按钮,同时保存文档

在列表内选择刚才创建是数字证书"netkiller"

单击"OK"按钮

单击"确定"按钮

选项对话框内单击"确定"按钮

关闭Word 完成操作

验证数字签名是否有效

打开刚刚签名的文档

Word窗口标题栏上(已签名,未验证) 表示签名成功

如果有人修改你的文档,当他再次保存时提示

再打开文档时Word窗口标题栏上"(已签名,未验证)"消失,表示你的文档被其它人撰改

提示

你同样可以使用第三方数字证书来签名你的文档

3. Email Security using OpenPGP and S/MIME

3.1. Gpg4win

http://www.gpg4win.org/

3.2. S/MIME

数字签名、加密与证书颁发机构

第 14 章 OpenStego - 图像文件水印加密

http://openstego.sourceforge.net/

第 15 章 数字证书开发

1. Java (java.security.*)

1.1. 访问X.509证书

Java访问X.509证书

```
/*
 * Created on 2005-7-1
 *
 * Author: neo chen <openunix@163.com>
 * Nickname: netkiller
 */
import java.io.*;
import java.security.cert.*;
import java.security.cert.CertificateFactory;

public class CertInfo {
        static String issue,after,before,subject;
        static String serialno,signalg;
        static int version;
        public void Init() throws Exception{
        CertificateFactory certFactory =
CertificateFactory.getInstance("X.509");
        FileInputStream fis=new
FileInputStream("e:/Java/chen.cer");
        X509Certificate cert =
(X509Certificate)certFactory.generateCertificate(fis);

        fis.close();
        issue=cert.getIssuerDN().toString();
        subject=cert.getSubjectDN().getName();
        after=cert.getNotAfter().toString();
        before=cert.getNotBefore().toString();
        version=cert.getVersion();
        serialno=cert.getSerialNumber().toString();
        signalg=cert.getSigAlgName();
        }
        public String getIssue(){
        return issue;
        }

        public String getAfter(){
        return after;
        }

        public String getBefore(){
        return before;
        }

        public String getSerial(){
        return serialno;
        }

        public String getsignalg(){
        return signalg;
        }

        public String getsubject(){
        return subject;
        }

        public String getversion(){
        return ("ver:"+version);
        }


        public static void main(String[] args) throws Exception
        {
        CertInfo c=new CertInfo();
        c.Init();
        System.out.println(c.getBefore());
        System.out.println(version);
        System.out.println(c.getversion());
        System.out.println(issue);
        System.out.println(c.getsubject());
        System.out.println(c.getsignalg());
        }
}
```

1.2. 创建证书

2. SSL Socket

2.1. Java Socket HTTPS

```
package netkiller;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.MalformedURLException;
import java.net.URL;

import javax.net.ssl.HttpsURLConnection;
import javax.net.ssl.SSLContext;
import javax.net.ssl.TrustManager;
import javax.net.ssl.X509TrustManager;

public class HTTPS {

        public static void main(String[] args) {
            // Create a trust manager that does not validate
certificate chains
            TrustManager[] trustAllCerts = new TrustManager[]{
                new X509TrustManager() {
                    public java.security.cert.X509Certificate[]
getAcceptedIssuers() {
                        return null;
                    }
                    public void checkClientTrusted(
                        java.security.cert.X509Certificate[]
certs, String authType) {
                    }
                    public void checkServerTrusted(
                        java.security.cert.X509Certificate[]
certs, String authType) {
                    }
                }
            };

            // Install the all-trusting trust manager
            try {
                SSLContext sc = SSLContext.getInstance("SSL");
                sc.init(null, trustAllCerts, new
java.security.SecureRandom());

HttpsURLConnection.setDefaultSSLSocketFactory(sc.getSocketFacto
ry());
            } catch (Exception e) {
            }

            // Now you can access an https URL without having
the certificate in the truststore
            try {
                //Create a URL for the desired page
                URL url = new URL("https://java.sun.com/");

                // Read all the text returned by the server
                BufferedReader in = new BufferedReader(new
InputStreamReader(url.openStream()));
                String html;
                while ((html = in.readLine()) != null) {
                    // str is one line of text; readLine()
strips the newline character(s)
                        System.out.println(html);
                }
                in.close();

            } catch (MalformedURLException mue) {
            } catch (IOException ioe) {
            }

        }

}
```

2.2. Java SSL Socket Client

```
package netkiller;
```

```
import java.io.*;
import java.net.*;
import javax.net.SocketFactory;
import javax.net.ssl.*;

public class SSLClientSocket {

        public static void main(String[] args) {
            try {
                int port = 443;
                String hostname = "java.sun.com";

                SocketFactory socketFactory =
SSLSocketFactory.getDefault();
                Socket socket =
socketFactory.createSocket(hostname, port);

                // Create streams to securely send and receive
data to the server
                InputStream in = socket.getInputStream();
                OutputStream out = socket.getOutputStream();

                BufferedReader socketReader = new
BufferedReader(new InputStreamReader(in));
                PrintWriter socketWriter = new
PrintWriter(out);

                socketWriter.println("GET /");
                socketWriter.flush();
                String line=null;
                StringBuffer html = new StringBuffer();
                while((line=socketReader.readLine())!=null){
                        html.append(line+"\n");
                }
                // Read from in and write to out...
                System.out.println(html.toString());

                // Close the socket
                socketReader.close();
                socketWriter.close();
                in.close();
                out.close();
            } catch(IOException e) {
            }
        }

}
```

2.3. Java SSL Socket Server

这里实现一个简单的SSL Echo服务器

创建证书

keytool -genkey -keyalg RSA -alias mycert -keystore mySrvKeystore

```
C:\workspace\test>keytool -genkey -keyalg RSA -alias mycert -
keystore mySrvKeystore
输入keystore密码：  13721218
您的名字与姓氏是什么?
  [Unknown]：  陈景峰
您的组织单位名称是什么?
  [Unknown]：  中国无线电运动协会
您的组织名称是什么?
  [Unknown]：  无线电运动协会
您所在的城市或区域名称是什么?
  [Unknown]：  深圳
您所在的州或省份名称是什么?
  [Unknown]：  广东省
该单位的两字母国家代码是什么
  [Unknown]：  CN
CN=陈景峰, OU=中国无线电运动协会, O=无线电运动协会, L=深圳, ST=广东省,
C=CN 正确
吗?
  [否]：  Y

输入<mycert>的主密码
        (如果和 keystore 密码相同，按回车)：  13721218
```

```
C:\workspace\neo>javac netkiller\SSLServerSocket.java
```

java -Djavax.net.ssl.keyStore=mySrvKeystore -
Djavax.net.ssl.keyStorePassword=13721218 netkiller.SSLServerSocket

Client

```
C:\workspace\neo>javac netkiller\SSLClientSocket.java java -
Djavax.net.ssl.trustStore=truststore -
Djavax.net.ssl.trustStorePassword=13721218
netkiller.SSLClientSocket
```

第 16 章 Smart card(智能卡)

1. OpenSC - tools and libraries for smart cards

https://github.com/OpenSC

1.1. 安装 OpenSC

```
yum install -y autoconf automake libtool gcc
yum install -y readline-devel openssl-devel pcsc-lite-devel

tar zxvf opensc-0.13.0.tar.gz
cd opensc-0.13.0
./bootstrap
./configure --prefix=/srv/opensc --sysconfdir=/etc/opensc
make
make install
```

```
# /etc/init.d/pcscd start
Starting PC/SC smart card daemon (pcscd):                       [
OK  ]
```

2. openct-tool - OpenCT smart card utility

```
yum install openct

cp /etc/openct.conf /etc/openct.conf.old
```

```
# /etc/init.d/openct start
Initializing OpenCT smart card terminals:                  [
OK  ]
```

3. ccid - Generic USB CCID smart card reader driver

```
# yum install ccid
```

4. usbutils: Linux USB utilities

```
# yum install usbutils
```

```
# lsusb
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 003 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 004 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 005 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 005 Device 006: ID 096e:0302 Feitian Technologies, Inc.
```

```
# usb-devices

T:  Bus=01 Lev=00 Prnt=00 Port=00 Cnt=00 Dev#=  1 Spd=480 MxCh=
8
D:  Ver= 2.00 Cls=09(hub  ) Sub=00 Prot=00 MxPS=64 #Cfgs=  1
P:  Vendor=1d6b ProdID=0002 Rev=02.06
S:  Manufacturer=Linux 2.6.32-431.1.2.0.1.el6.x86_64 ehci_hcd
S:  Product=EHCI Host Controller
S:  SerialNumber=0000:00:1d.7
C:  #Ifs= 1 Cfg#= 1 Atr=e0 MxPwr=0mA
I:  If#= 0 Alt= 0 #EPs= 1 Cls=09(hub  ) Sub=00 Prot=00
Driver=hub

T:  Bus=02 Lev=00 Prnt=00 Port=00 Cnt=00 Dev#=  1 Spd=12  MxCh=
2
D:  Ver= 1.10 Cls=09(hub  ) Sub=00 Prot=00 MxPS=64 #Cfgs=  1
P:  Vendor=1d6b ProdID=0001 Rev=02.06
S:  Manufacturer=Linux 2.6.32-431.1.2.0.1.el6.x86_64 uhci_hcd
S:  Product=UHCI Host Controller
S:  SerialNumber=0000:00:1d.0
C:  #Ifs= 1 Cfg#= 1 Atr=e0 MxPwr=0mA
I:  If#= 0 Alt= 0 #EPs= 1 Cls=09(hub  ) Sub=00 Prot=00
Driver=hub

T:  Bus=03 Lev=00 Prnt=00 Port=00 Cnt=00 Dev#=  1 Spd=12  MxCh=
2
```

```
D:  Ver= 1.10 Cls=09(hub  ) Sub=00 Prot=00 MxPS=64 #Cfgs=  1
P:  Vendor=1d6b ProdID=0001 Rev=02.06
S:  Manufacturer=Linux 2.6.32-431.1.2.0.1.el6.x86_64 uhci_hcd
S:  Product=UHCI Host Controller
S:  SerialNumber=0000:00:1d.1
C:  #Ifs= 1 Cfg#= 1 Atr=e0 MxPwr=0mA
I:  If#= 0 Alt= 0 #EPs= 1 Cls=09(hub  ) Sub=00 Prot=00
Driver=hub

T:  Bus=04 Lev=00 Prnt=00 Port=00 Cnt=00 Dev#=  1 Spd=12  MxCh=
2
D:  Ver= 1.10 Cls=09(hub  ) Sub=00 Prot=00 MxPS=64 #Cfgs=  1
P:  Vendor=1d6b ProdID=0001 Rev=02.06
S:  Manufacturer=Linux 2.6.32-431.1.2.0.1.el6.x86_64 uhci_hcd
S:  Product=UHCI Host Controller
S:  SerialNumber=0000:00:1d.2
C:  #Ifs= 1 Cfg#= 1 Atr=e0 MxPwr=0mA
I:  If#= 0 Alt= 0 #EPs= 1 Cls=09(hub  ) Sub=00 Prot=00
Driver=hub

T:  Bus=05 Lev=00 Prnt=00 Port=00 Cnt=00 Dev#=  1 Spd=12  MxCh=
2
D:  Ver= 1.10 Cls=09(hub  ) Sub=00 Prot=00 MxPS=64 #Cfgs=  1
P:  Vendor=1d6b ProdID=0001 Rev=02.06
S:  Manufacturer=Linux 2.6.32-431.1.2.0.1.el6.x86_64 uhci_hcd
S:  Product=UHCI Host Controller
S:  SerialNumber=0000:00:1d.3
C:  #Ifs= 1 Cfg#= 1 Atr=e0 MxPwr=0mA
I:  If#= 0 Alt= 0 #EPs= 1 Cls=09(hub  ) Sub=00 Prot=00
Driver=hub

T:  Bus=05 Lev=01 Prnt=01 Port=00 Cnt=01 Dev#=  6 Spd=1.5 MxCh=
0
D:  Ver= 1.10 Cls=00(>ifc ) Sub=00 Prot=00 MxPS= 8 #Cfgs=  1
P:  Vendor=096e ProdID=0302 Rev=01.07
S:  Manufacturer=OEM
S:  Product=HID Token M8
C:  #Ifs= 1 Cfg#= 1 Atr=10 MxPwr=50mA
I:  If#= 0 Alt= 0 #EPs= 0 Cls=03(HID  ) Sub=00 Prot=00 Driver=
(none)
```

5. USB Token

USB Key

以 ePass1000ND 为例:

```
Feitian ePass1000ND Editor v 1.0.7.412
Create at Jun 19 2009, 16:23:26

Library version 1.1.0
Create Context:Success!


Main Menu:
-------------------------------------------------------------
  Open[F]irst  LED[O]n   [P]in     [L]ist       File[M]enu
  Open[N]ext   LE[D]Off  [S]oPin   [G]enRandom  Cr[y]ptMenu
  [R]eOpen     Rese[t]   [A]ccess               Set[u]pMenu
  [C]lose      E[x]it

Input selection:
```

5.1. Open[F]irst

```
Input selection:f
Open device:Success!

=>> Firmware Version: 1.07
=>> Product Code: 10
=>> Capabilities: 3
=>> Total memory size: 8192 bytes
=>> Free memory space: 7592 bytes
=>> Max directory levels: 2
=>> File system type: 1
=>> Friendly token name: ePass1000ND
=>> Hardware serial number: 0x42F81BC1B4B3CDD3

Main Menu:
-------------------------------------------------------------
  Open[F]irst  LED[O]n   [P]in     [L]ist       File[M]enu
  Open[N]ext   LE[D]Off  [S]oPin   [G]enRandom  Cr[y]ptMenu
  [R]eOpen     Rese[t]   [A]ccess               Set[u]pMenu
  [C]lose      E[x]it
```

5.2. [S]oPin 验证管理员

```
Input selection:s
Input SO-Pin:
```

5.3. LED 灯控制

开灯

```
Input selection:o

Turn LED on:Success!


Main Menu:
-------------------------------------------------------------
  Open[F]irst  LED[O]n   [P]in     [L]ist       File[M]enu
  Open[N]ext   LE[D]Off  [S]oPin   [G]enRandom  Cr[y]ptMenu
  [R]eOpen     Rese[t]   [A]ccess               Set[u]pMenu
  [C]lose      E[x]it
```

关灯

```
Input selection:d

Turn LED off:Success!


Main Menu:
-------------------------------------------------------------
  Open[F]irst  LED[O]n   [P]in     [L]ist       File[M]enu
  Open[N]ext   LE[D]Off  [S]oPin   [G]enRandom  Cr[y]ptMenu
  [R]eOpen     Rese[t]   [A]ccess               Set[u]pMenu
  [C]lose      E[x]it
```

5.4. [L]ist

```
Input selection:l

DirID    FileID  Type   Read   Write   Delete  Crypt  Size
-----    -----    ----   -----  -----   ------   -----  -----
0000      FFFF    DATA  ANYONE  ANYONE  ANYONE ANYONE  512
0000      FFFE    DATA  ANYONE  ANYONE  ANYONE ANYONE  32
0000      0000    FREE  ANYONE  ANYONE  ANYONE ANYONE  7592

Main Menu:
-------------------------------------------------------------
  Open[F]irst  LED[O]n   [P]in     [L]ist       File[M]enu
  Open[N]ext   LE[D]Off  [S]oPin   [G]enRandom  Cr[y]ptMenu
  [R]eOpen     Rese[t]   [A]ccess               Set[u]pMenu
  [C]lose      E[x]it
```

5.5. File[M]enu 文件菜单

```
Input selection:m


File Menu:
-------------------------------------------------------------
  C[h]Dir    [L]ist   Create[D]ir   Create[F]ile   Create[A]pp
  DelD[i]r   D[e]leteFile
  [O]pen     [R]ead   [W]rite      [C]lose
  Cr[y]ptMenu         Set[u]pMenu   E[x]it
```

[L]ist 列目录与文件

```
Input selection:l

DirID    FileID  Type   Read   Write   Delete  Crypt  Size
-----    -----    ----   -----  -----  ------   -----  -----
0000      FFFF    DATA  ANYONE  ANYONE  ANYONE ANYONE  512
0000      FFFE    DATA  ANYONE  ANYONE  ANYONE ANYONE  32
0000      0000    FREE  ANYONE  ANYONE  ANYONE ANYONE  7592

Main Menu:
-------------------------------------------------------------
  Open[F]irst  LED[O]n   [P]in     [L]ist       File[M]enu
  Open[N]ext   LE[D]Off  [S]oPin   [G]enRandom  Cr[y]ptMenu
  [R]eOpen     Rese[t]   [A]ccess               Set[u]pMenu
  [C]lose      E[x]it
```

目录管理

创建目录

Create[D]ir

```
Input selection:d
Input ID of the Dir to be created(0-FFFF):1122
Success!


File Menu:
-------------------------------------------------------------
  C[h]Dir    [L]ist   Create[D]ir   Create[F]ile   Create[A]pp
  DelD[i]r   D[e]leteFile
  [O]pen     [R]ead   [W]rite      [C]lose
  Cr[y]ptMenu         Set[u]pMenu   E[x]it
```

查看是否成功

```
Input selection:l

DirID    FileID  Type   Read   Write   Delete  Crypt  Size
-----    -----    ----   -----  -----  ------   -----  -----
0000      FFFF    DATA  ANYONE  ANYONE  ANYONE ANYONE  512
0000      FFFE    DATA  ANYONE  ANYONE  ANYONE ANYONE  32
0000      1122    DIR   NONE    NONE    ANYONE NONE    0
0000      0000    FREE  ANYONE  ANYONE  ANYONE ANYONE  7584

File Menu:
-------------------------------------------------------------
  C[h]Dir    [L]ist   Create[D]ir   Create[F]ile   Create[A]pp
  DelD[i]r   D[e]leteFile
  [O]pen     [R]ead   [W]rite      [C]lose
  Cr[y]ptMenu         Set[u]pMenu   E[x]it
```

如果正确我们可以看到 0000 1122 DIR NONE NONE ANYONE NONE 0

C[h]Dir 进入目录

```
Input selection:h
Input ID of the dir to change in(0-FFFF):1122
Success!


File Menu:
-------------------------------------------------------------
  C[h]Dir    [L]ist   Create[D]ir   Create[F]ile   Create[A]pp
  DelD[i]r   D[e]leteFile
  [O]pen     [R]ead   [W]rite      [C]lose
  Cr[y]ptMenu         Set[u]pMenu   E[x]it
```

删除目录 DelD[i]r

```
Input selection:i
Input ID of the Dir to delete(0-FFFF):1122
Success!


File Menu:
-------------------------------------------------------------
  C[h]Dir    [L]ist   Create[D]ir   Create[F]ile   Create[A]pp
  DelD[i]r   D[e]leteFile
  [O]pen     [R]ead   [W]rite      [C]lose
  Cr[y]ptMenu         Set[u]pMenu   E[x]it
```

确认是否删除成功

```
Input selection:l

DirID    FileID  Type   Read   Write   Delete  Crypt  Size
-----    -----    ----   -----  -----  ------   -----  -----
0000      FFFF    DATA  ANYONE  ANYONE  ANYONE ANYONE  512
0000      FFFE    DATA  ANYONE  ANYONE  ANYONE ANYONE  32
0000      0000    FREE  ANYONE  ANYONE  ANYONE ANYONE  7592

File Menu:
-------------------------------------------------------------
  C[h]Dir    [L]ist   Create[D]ir   Create[F]ile   Create[A]pp
  DelD[i]r   D[e]leteFile
  [O]pen     [R]ead   [W]rite      [C]lose
  Cr[y]ptMenu         Set[u]pMenu   E[x]it
```

文件管理

Create[F]ile 创建文件

创建 MD5 文件

```
Input selection:f
Input ID of the file to be created(0-FFFF): 0011
Input file type(2=DATA,4=MD5,8=TEA): 4
Input write access (0=Anyone 1=User 2=SO 7=None): 0
Input crypt access (0=Anyone 1=User 2=SO 7=None): 0
Success!
```

[L]ist 查看文件是否创建

```
Input selection:l

DirID    FileID  Type   Read   Write   Delete  Crypt  Size
-----    -----    ----   -----  -----  ------   -----  -----
0000      FFFF    DATA  ANYONE  ANYONE  ANYONE ANYONE  512
0000      FFFE    DATA  ANYONE  ANYONE  ANYONE ANYONE  32
0000      0011    MD5   NONE    ANYONE  ANYONE ANYONE  16
0000      0000    FREE  ANYONE  ANYONE  ANYONE ANYONE  7568
```

创建 TEA 文件

```
Input selection:f
Input ID of the file to be created(0-FFFF): 0022
Input file type(2=DATA,4=MD5,8=TEA): 8
Input write access (0=Anyone 1=User 2=SO 7=None): 0
Input crypt access (0=Anyone 1=User 2=SO 7=None): 0
Success!
```

查看文件是否正确创建

```
Input selection:l

DirID    FileID  Type   Read   Write   Delete  Crypt  Size
-----    -----    ----   -----  -----  ------   -----  -----
0000      FFFF    DATA  ANYONE  ANYONE  ANYONE ANYONE  512
0000      FFFE    DATA  ANYONE  ANYONE  ANYONE ANYONE  32
0000      0011    MD5   NONE    ANYONE  ANYONE ANYONE  16
0000      0022    TEA   NONE    ANYONE  ANYONE ANYONE  16
0000      0000    FREE  ANYONE  ANYONE  ANYONE ANYONE  7544
```

[O]pen 打开文件

```
Input selection:o
Input ID of the file to open(0-FFFF):0011
Success!

 File size: 16
 File type: MD5
 File Crypt Access: ANYONE
 File Read Access: NONE
 File Write Access: ANYONE
 File Delete Access: ANYONE
```

D[e]leteFile 删除文件

```
Input selection:e
Input ID of the file to delete(0-FFFF):0022
Success!
```

查看删除结果

```
Input selection:l

DirID    FileID  Type   Read   Write   Delete  Crypt  Size
-----    -----    ----   -----  -----  ------   -----  -----
0000      FFFF    DATA  ANYONE  ANYONE  ANYONE ANYONE  512
0000      FFFE    DATA  ANYONE  ANYONE  ANYONE ANYONE  32
0000      0011    MD5   NONE    ANYONE  ANYONE ANYONE  16
0000      F000    DIR   NONE    NONE    ANYONE NONE    0
                  Name: "a360d5826bdee7e6bd04a428eefc0bba"
0000      0000    FREE  ANYONE  ANYONE  ANYONE ANYONE  7560
```

Create[A]pp 创建GUID

```
Input selection:a
Input App Name of the Dir(0-32chars):a360d5826bdee7e6bd04a428eefc0bba
Input GUID of the Dir{xxxx-xxxx...}:5744e586-cc37-11e3-945f-00259035906c
Success!
```

查看创建情况

```
DirID    FileID  Type   Read   Write   Delete  Crypt  Size
-----    -----    ----   -----  -----  ------   -----  -----
0000      FFFF    DATA  ANYONE  ANYONE  ANYONE ANYONE  512
0000      FFFE    DATA  ANYONE  ANYONE  ANYONE ANYONE  32
0000      0011    MD5   NONE    ANYONE  ANYONE ANYONE  16
0000      0022    TEA   NONE    ANYONE  ANYONE ANYONE  16
0000      F000    DIR   NONE    NONE    ANYONE NONE    0
                  Name: "a360d5826bdee7e6bd04a428eefc0bba"
0000      0000    FREE  ANYONE  ANYONE  ANYONE ANYONE  7536
```

5.6. Set[u]pMenu 设置菜单

修改pin

[P]IN 修改用户pin

```
Input selection:p

Change User Pin
Input Old User Pin:12345678
Input new User Pin:87654321
```

[S]oPIN 修改管理员pin

[T]okenName 修改Token名字

```
Input selection:t
Input new Token Name:Netkiller
Success!
```

E[x]it 推出然后使用Open[F]irst查看

```
Input selection:f
Open device:Success!

=>> Firmware Version: 1.07
=>> Product Code: 10
=>> Capabilities: 3
=>> Total memory size: 8192 bytes
=>> Free memory space: 7512 bytes
=>> Max directory levels: 2
=>> File system type: 1
=>> Friendly token name: Netkiller
=>> Hardware serial number: 0x42F81BC1B4B3CDD3
```

[C]leanup

清空文件，目录等等

```

```

[U]lockPIN

[A]ccessSettings

[I]nitToken 初始化

首次使用，请初始化

5.7. Linux ePass

```
# dmesg | grep usb
usb 5-1: USB disconnect, device number 5
usb 5-1: new low speed USB device number 6 using uhci_hcd
usb 5-1: New USB device found, idVendor=096e, idProduct=0302
usb 5-1: New USB device strings: Mfr=1, Product=2, SerialNumber=0
usb 5-1: Product: HID Token M8
usb 5-1: Manufacturer: OEM
usb 5-1: configuration #1 chosen from 1 choice
usbhid 5-1:1.0: couldn't find an input interrupt endpoint
```

第 17 章 Credentials Organization

1. VeriSign

http://www.verisign.com/

http://www.verisign.com/cn/

VeriSign (Nasdaq: VRSN) is the trusted provider of Internet infrastructure services for the networked world. Billions of times each day, VeriSign helps companies and consumers all over the world engage in communications and commerce with confidence. VeriSign offerings include SSL, SSL Certificates, Extended Validation (EV SSL), VeriSign Trust Seal, two-factor authentication, identity protection, malware scan, public key infrastructure (PKI), DDoS mitigation and Domain Name Services.

1.1. iTrusChina

http://verisign.itrus.com.cn/

1.2. Thawte

http://www.thawte.com/

Thawte is a leading global Certification Authority. Our SSL and code signing digital certificates are used globally to secure servers, provide data encryption, authenticate users, protect privacy and assure online identifies through stringent authentication and verification processes. Our SSL certificates include Wildcard SSL Certificates, SGC SuperCerts and Extended Validation SSL Certificates.

thawte 是全球领先的认证机构。我们的 SSL 和代码签名数字证书在全球范围内提供服务器的安全保护，可以进行数据加密、可以验证用户，通过严格的验证和认证程序保护个人隐私，确保在线识别过程

的安全。我们的 SSL 证书包括通配符 SSL 证书、SGC SuperCerts 和扩展验证 SSL 证书。

1.3. Geotrust

http://geotrust.itrus.com.cn/

2. UserTrust

http://www.usertrust.com/

Comodo offers essential infrastructure to enable e-merchants, other Internet-connected companies, software providers, and individual consumers to interact and conduct business via the Internet safely and securely. Our PKI solutions including, SSL Certificates, Extended Validation Certificates, Code Signing Certificates as well as Secure E-Mail Certificates, increase consumer trust in transacting business online, secure information through strong encryption, and satisfy many industry best practices or security compliance requirements with SSL.

3. 境内其他CA机构

3.1. WoSign®、I'm Verified®、WoTrust®、沃通®

http://www.wosign.com/

上级是 UserTrust

4. SSL FOR FREE

https://www.sslforfree.com

免费SSL证书

nginx

```
cat certificate.crt ca_bundle.crt >> netkiller.cn.crt
openssl rsa -in private.key -out netkiller.cn.key
```

5. Let’s Encrypt

```
git clone https://github.com/letsencrypt/letsencrypt
cd letsencrypt
./letsencrypt-auto
```

```
[root@netkiller letsencrypt]# ./letsencrypt-auto
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Failed to find executable apachectl in PATH:
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/root/bin
Plugins selected: Authenticator nginx, Installer nginx

Which names would you like to activate HTTPS for?
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - -
1: netkiller.cn
2: www.netkiller.cn
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - -
Select the appropriate numbers separated by commas and/or
spaces, or leave input
blank to select all options shown (Enter 'c' to cancel): 2
Obtaining a new certificate
Performing the following challenges:
http-01 challenge for www.netkiller.cn
Waiting for verification...
Cleaning up challenges
Deploying Certificate to VirtualHost
/etc/nginx/conf.d/www.netkiller.cn.conf

Please choose whether or not to redirect HTTP traffic to HTTPS,
removing HTTP access.
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - -
1: No redirect - Make no further changes to the webserver
configuration.
2: Redirect - Make all requests redirect to secure HTTPS
```

```
access. Choose this for
new sites, or if you're confident your site works on HTTPS. You
can undo this
change by editing your web server's configuration.
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - -
Select the appropriate number [1-2] then [enter] (press 'c' to
cancel): 2
Redirecting all traffic on port 80 to ssl in
/etc/nginx/conf.d/www.netkiller.cn.conf

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - -
Congratulations! You have successfully enabled
https://www.netkiller.cn

You should test your configuration at:
https://www.ssllabs.com/ssltest/analyze.html?d=www.netkiller.cn
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - -

IMPORTANT NOTES:
 - Congratulations! Your certificate and chain have been saved
at:
   /etc/letsencrypt/live/www.netkiller.cn/fullchain.pem
   Your key file has been saved at:
   /etc/letsencrypt/live/www.netkiller.cn/privkey.pem
   Your cert will expire on 2018-11-12. To obtain a new or
tweaked
   version of this certificate in the future, simply run
   letsencrypt-auto again with the "certonly" option. To
   non-interactively renew *all* of your certificates, run
   "letsencrypt-auto renew"
 - If you like Certbot, please consider supporting our work by:

   Donating to ISRG / Let's Encrypt:
https://letsencrypt.org/donate
   Donating to EFF:                    https://eff.org/donate-
le
```

查看证书 https://www.ssllabs.com/ssltest/analyze.html?d=www.netkiller.cn