

网络程序设计系列丛书

Unix

4

网络编程

实用技术与实例分析

张 炯 编著
潇湘工作室 审校



清华大学出版社
<http://www.tup.tsinghua.edu.cn>



TP316.81

网络程序设计系列丛书

Unix网络编程

实用技术与实例分析

张 炯 编著
潇湘工作室 审校

清华大学出版社

(京)新登字 158 号

内 容 简 介

本书详细介绍了在 Unix 环境下网络编程的方法,全书分为四部分:第一部分“网络基础”主要讲述 TCP/IP 协议簇,尤其是与编程相关的部分,并说明了网络编程环境;第二部分“套接字”是网络编程的核心,在此通过讲解套接字库函数、TCP 套接字、UDP 套接字及相应的实例,使读者能够编写基本的网络程序;第三部分“Unix 网络编程实用技术”是本书的重点,讲述 Unix 网络开发过程中常用的技术,如并发服务器技术、名字和 IP 地址转换、同步及进程间通信技术、异常处理技术、实用套接字类库的创建,说明如何提高软件的性能、可靠性和可扩充性,并配有大量实例予以说明;第四部分“高级网络编程”主要涉及底层 IP 编程技术,可用于路由器、网络监视器及专用协议的开发,介绍了守护进程、原始套接字、数据链路访问、多接口捆绑及路由套接字技术。

本书涉及的内容包括 Unix 系统、网络协议及编程技术,并由浅入深地讲述了网络编程核心技术、实用技术和高级网络编程。本书既是从事网络开发人员的参考资料,也可以作为学习 Unix 网络编程知识的教材。

版权所有,翻印必究。

本书封面贴有清华大学出版社激光防伪标签,无标签者不得销售。

图书在版编目(CIP)数据

Unix 网络编程实用技术与实例分析/张炯编著. —北京:清华大学出版社, 2002.11

(网络程序设计系列丛书)

ISBN 7-302-05891-1

I. U... II. 张... III. Unix 操作系统 - 程序设计 IV. TP316.81

中国版本图书馆 CIP 数据核字(2002)第 070811 号

出版者: 清华大学出版社(北京清华大学学研大厦, 邮编: 100084)

<http://www.tup.tsinghua.edu.cn>

责任编辑: 龙啟铭

印刷者: 清华大学印刷厂

发行者: 新华书店总店北京发行所

开本: 787×1092 1/16 **印张:** 22.5 **字数:** 547 千字

版次: 2002 年 11 月第 1 版 2002 年 11 月第 1 次印刷

书号: ISBN 7-302-05891-1/TP·3497

印数: 0001~4000

定价: 32.00 元

前 言

网络编程是指编写网络通信程序，以完成网络上不同程序间的通信。网络程序分为服务器和客户端两部分。客户端发送请求给服务器，服务器处理客户请求并将结果发回客户端。网络编程是通过调用网络 API 实现的。目前有多种网络 API，其中套接字最为流行。本书讲述的网络编程就是以套接字 API 为基础的。

套接字的发展与 Unix 操作系统的发展紧密相关。Unix 是 20 世纪 70 年代由 AT&T 的 Bell 实验室开发的。从 Unix 的发展历史来看，主要有两大流派：AT&T 的 Unix 系统 V 版本和加州大学伯克利分校的 BSD 版本。套接字 API 最初是随 4.2 版的 BSD 系统于 1983 年发布的，最终在多数 Unix 系统上获得了支持。

网络编程涉及的范围极广，例如所支持的通信协议(TCP/IP、NetBEUI、IPX/SPX 等)、运行的操作系统(Unix、Windows 等)、网络 API(套接字 API、XTI 等)和所用的编程语言(C/C++、Java 等)。国际互联网无疑是网络应用的主体，其采用的通信协议 TCP/IP 协议簇被几乎所有的操作系统所支持并最为广泛地使用，因此，本书的网络协议主要是 TCP/IP 协议簇。对于操作系统，Unix 和 Windows 都非常流行，但由于 Unix 系统的可靠性、安全性及开放性，大量的网络应用是在 Unix 系统下开发的，尤其是服务器部分。例如，在浏览网页时，所用的浏览器多数是在 Windows 系统下运行的，而 Web 服务器则主要在 Unix 下运行。考虑到服务器是决定网络应用系统性能的关键，另外随着 Unix 系统微机版的推广，越来越多的程序员转向 Unix 系统开发，因而本书选择 Unix 系统。Windows 也支持套接字 API (Windows 套接字)，所以在 Unix 下开发的网络程序仅需作少量的修改便可在 Windows 下运行。Unix 操作系统是用 C 语言研制开发的。套接字 API 也是以 C 函数形式提供的，因此本书的实例都是用 C/C++ 编写的。

本书的特点

大量的实例和代码分析

本书以实用为出发点介绍网络编程，并配有大量的实例和代码分析。这些代码均为作者长期工作经验的积累，非常具有实用价值。

为了方便读者，在介绍各种技术的同时提供了大量实例并进行了详细的分析，不仅分析源代码，还针对程序的运行结果进行说明。

体贴读者，详细讲解网络开发难点

本书涉及的技术都是在网络开发过程中常用的技术，由于网络应用的性能及可靠性是网络开发中的主要难点，本书重点对这方面的技术进行了深入的讲解说明。

为了增强实用性，本书介绍了大量实用性强的实例，并且部分实例采用面向对象方法进行封装，便于读者使用、扩充和移植。

在叙述方面，对于仅需了解的知识讲解力求简明，而对于关键技术则深入透彻。尤其是通过实例对比的方式，使读者能体会到每种技术所起到的作用和产生的问题。另外，本

书介绍了所涉及技术的应用背景,使得读者在实际网络编程中能够知道采用哪种技术解决相应问题。

作者的体会与经验之谈

网络编程是非常复杂的,需要长期的学习和积累,关于这一点本人深有体会。在开始学习网络编程时,觉得非常简单,找个例子简单修改一下就可以运行了,甚至连网络方面的书籍也不必看。

然而,在开始开发实际项目时,才发现问题的层出不穷,每当费了九牛二虎之力解决一个问题后,又产生了新的问题,更糟糕的是经常出现“无故故障”。这时才开始较系统地学习网络协议及相关编程技术,利用相关的网络知识找出这些“无故故障”的原因,并通过采用相应技术提高网络应用系统的性能和可靠性。当看到自己开发的系统高效可靠地运行时,真是有苦尽甘来的感觉。相信读者在学习本书的过程中也会有相同的体会。

在学完本书第二部分时,读者就可以编写网络程序了。然而,由于实际网络应用的高并发性,这些程序会变得毫无用处。当采用并发技术后,又带来同步等新问题。当学完全书后,相信读者将在 Unix 网络编程方面游刃有余。

本书结构

在结构安排上,本书遵循由浅入深的原则,非常适于没有网络开发经验的人员学习。本书共分四部分:

- 第一部分“网络基础”主要讲述 TCP/IP 协议簇,尤其是与编程相关的部分,并且对网络编程环境进行了介绍。这部分是网络编程所应具备的基本知识。
- 第二部分“套接字”是网络编程的核心。本部分通过讲解套接字库函数、TCP 套接字、UDP 套接字及相应实例,使读者能够编写基本的网络程序。
- 第三部分“Unix 网络编程实用技术”是本书的重点。讲述 Unix 网络开发过程常用的技术,以提高软件的性能、可靠性和可扩充性,并配有大量实例予以说明。该部分共分 5 章:并发服务器技术、名字和 IP 地址转换、同步及进程间通信技术、异常处理技术、实用套接字类库的创建。
- 第四部分“高级网络编程”主要涉及底层 IP 编程技术,可用于路由器,网络监视器及专用协议的开发,包括守护进程、原始套接字、数据链路访问、多接口捆绑及路由套接字技术。

读者对象

本书的目标是使读者学完本书后能够开发实用的网络程序,本书有大量的实例,因此读者应具备 C/C++ 编程及 Unix 系统的使用经验。另外,本书系统地讲述了 Unix 网络编程的主要实用技术,也适用于从事网络开发人员参考之用。

联系我们

本书由潇湘工作室策划和组织编写,张炯主笔。该作者自 1991 年计算机系毕业后,一直从事网络开发及研究工作,对 Unix 系统下的网络应用级及系统级的开发具有丰富的经验,并对网络协议和网络互连有深入的研究。

在本书的编写过程中,得到了许多人的热情指导及多方面的帮助,在此一并表示诚挚

的感谢!

读者在学习本书的过程中,若发现本书有疏漏之处和错误,或者,如果您有好的想法和建议,亦或是您需要本书程序的源代码,均请与我们联系,我们将竭尽所能提供帮助,并不断改进工作,为读者奉献高品质的好书。

我们的电子邮件地址为: xiaoxiang-007@sohu.com

目 录

第一部分 网络基础

第 1 章 Unix 系统基础.....	1
1.1 Unix 系统概述.....	1
1.1.1 Unix 系统的历史.....	1
1.1.2 Unix 系统的特点.....	1
1.1.3 Unix 系统的体系结构.....	1
1.1.4 Unix 系统的地址空间.....	2
1.1.5 POSIX 标准.....	2
1.2 常用 Unix 网络命令.....	2
1.2.1 ping.....	2
1.2.2 netstat.....	3
1.2.3 ifconfig.....	3
1.2.4 route.....	4
1.2.5 tcpdump.....	4
1.3 网络基本配置文件.....	5
1.4 软件开发环境.....	5
1.4.1 vi 编辑器.....	5
1.4.2 gcc 编译器.....	7
1.4.3 gdb 调试器.....	7
1.5 简单实例.....	8
1.5.1 源程序分析.....	8
1.5.2 实现过程.....	10
1.6 小结.....	12
第 2 章 TCP/IP.....	13
2.1 TCP/IP 体系.....	13
2.2 IP 协议.....	14
2.2.1 IP 包的结构.....	14
2.2.2 IP 地址组成.....	15
2.2.3 IP 地址表示.....	15
2.2.4 IP 地址类型.....	15
2.2.5 子网掩码.....	16
2.3 TCP 协议.....	16

2.3.1	建立 TCP 连接	16
2.3.2	关闭 TCP 连接	16
2.3.3	TCP 数据包结构	17
2.4	UDP 协议	17
2.5	ICMP 协议	18
2.6	端口号分配	19
2.6.1	端口分类	19
2.6.2	常用端口号	19
2.7	IP 路由	20
2.7.1	路由表分类	20
2.7.2	IP 路由过程	20
2.8	小结	21

第二部分 套 接 字

第 3 章	套接字基础	22
3.1	套接字概述	22
3.2	套接字类型	23
3.3	套接字地址结构	23
3.3.1	INET 协议簇地址结构 <code>sockaddr_in</code>	23
3.3.2	存储地址和端口信息的 <code>sockaddr</code>	24
3.3.3	32 位 IPv4 地址结构 <code>in_addr</code>	24
3.4	端口	25
3.5	带外数据	26
3.6	连接类型	26
3.7	小结	27
第 4 章	TCP 套接字	28
4.1	基本方法	28
4.1.1	TCP 套接字实现过程	28
4.1.2	TCP 服务器模板	29
4.1.3	TCP 客户模板	30
4.2	实现 TCP 套接字	31
4.2.1	产生 TCP 套接字	31
4.2.2	绑定	32
4.2.3	监听	34
4.2.4	接受请求	35
4.2.5	连接建立	36
4.2.6	数据传输	38

4.2.7	终止连接	39
4.3	TCP 套接字编程实例	40
4.3.1	实例说明	40
4.3.2	TCP 服务器	40
4.3.3	TCP 客户	42
4.3.4	运行程序	44
4.4	小结	45
第 5 章	UDP 套接字	46
5.1	基本方法	46
5.1.1	UDP 套接字实现过程	46
5.1.2	UDP 服务器模板	47
5.1.3	UDP 客户模板	48
5.2	函数说明	49
5.2.1	UDP 套接字的数据发送——sendto()函数	49
5.2.2	UDP 套接字的数据接收——recvfrom()函数	49
5.3	UDP 套接字编程实例	51
5.3.1	UDP 服务器	51
5.3.2	UDP 客户	53
5.3.3	运行程序	55
5.4	小结	55
第三部分 Unix 网络编程实用技术		
第 6 章	并发服务器	57
6.1	并发服务器基础	57
6.1.1	服务器分类	57
6.1.2	重复性服务器实例	58
6.1.3	并发技术	64
6.1.4	并发服务器算法	64
6.2	多进程服务器	68
6.2.1	进程概念	68
6.2.2	创建进程	68
6.2.3	终止进程	70
6.2.4	多进程并发服务器	73
6.2.5	多进程并发服务器实例	77
6.3	多线程服务器	83
6.3.1	线程基础	84
6.3.2	线程函数调用(POSIX)	84

6.3.3	多线程并发服务器.....	87
6.3.4	给新线程传递参数.....	88
6.3.5	多线程并发服务器实例.....	92
6.3.6	线程安全 (MT-safe) 实例.....	96
6.4	I/O 多路复用服务器.....	112
6.4.1	I/O 模式.....	113
6.4.2	select() 函数.....	114
6.4.3	单线程并发服务器实例.....	116
6.5	套接字终止处理.....	123
6.6	小结.....	124
第 7 章	名字和 IP 地址转换.....	126
7.1	名字解析.....	126
7.2	套接字地址.....	126
7.2.1	地址结构.....	126
7.2.2	字节顺序.....	127
7.2.3	IP 地址转换函数.....	128
7.2.4	套接字地址信息函数.....	129
7.3	套接字信息函数.....	130
7.3.1	主机名转换为 IP 地址: gethostbyname() 函数.....	130
7.3.2	IP 地址转换为主机名: gethostbyaddr() 函数.....	132
7.3.3	获得服务的端口号: getservbyname() 函数.....	134
7.3.4	端口号转换为服务名: getservbyport() 函数.....	135
7.4	小结.....	135
第 8 章	同步及进程间通信.....	136
8.1	线程同步.....	136
8.1.1	线程同步基础.....	136
8.1.2	互斥锁基础.....	136
8.1.3	加锁和解锁互斥锁.....	138
8.1.4	条件变量.....	142
8.1.5	同步线程退出.....	151
8.1.6	死锁.....	158
8.2	进程同步.....	165
8.2.1	进程关系.....	165
8.2.2	信号处理.....	167
8.2.3	处理僵死进程.....	171
8.3	进程间通信.....	175
8.3.1	管道.....	176

8.3.2	FIFO	176
8.3.3	消息队列	180
8.3.4	共享内存	180
8.3.5	信号量	181
8.4	小结	182
第 9 章	异常处理	183
9.1	异常处理基础	183
9.2	函数调用的错误处理	183
9.2.1	显示错误信息	184
9.2.2	定义错误处理函数	187
9.3	I/O 超时处理	188
9.3.1	使用 alarm() 函数	188
9.3.2	使用 select 函数	189
9.4	服务器异常处理	190
9.4.1	异常处理的系统调用	190
9.4.2	服务器异常处理实例	191
9.5	客户异常处理	196
9.6	小结	196
第 10 章	创建实用套接字类库	197
10.1	创建静态链接库	197
10.1.1	创建库文件	197
10.1.2	建立库文件索引	197
10.1.3	连接库文件	197
10.2	创建动态链接库	198
10.2.1	创建库文件	198
10.2.2	使用动态链接库	198
10.2.3	相互引用的库文件	199
10.2.4	动态库与静态库并存	199
10.3	创建自定义的套接字类库	199
10.3.1	设计套接字类库	199
10.3.2	套接字系统调用: MySocket 类	201
10.3.3	多线程实现: MyThread 类	202
10.3.4	加锁/解锁: MyMutex 类和 MyCondition 类	205
10.3.5	基于 TCP 的多线程并发服务器: TcpServThr 类	209
10.3.6	TCP 多线程客户类: TcpCliThr 类	214
10.4	实例分析	218
10.4.1	实现聊天室服务器	218

10.4.2	实现聊天室客户	225
10.4.3	运行程序	227
10.5	小结	230
第四部分 高级网络编程技术		
第 11 章	守护进程	231
11.1	输出守护进程消息	231
11.1.1	syslogd 进程	231
11.1.2	syslog()函数	232
11.1.3	closelog()函数	234
11.2	创建守护进程	234
11.2.1	守护进程的创建过程	234
11.2.2	创建守护进程的代码	234
11.3	配置守护进程	235
11.4	守护进程实例	236
11.5	小结	241
第 12 章	原始套接字	243
12.1	产生原始套接字	243
12.2	写原始套接字	244
12.3	读原始套接字	244
12.4	原始套接字实例	245
12.5	小结	253
第 13 章	数据链路访问	254
13.1	数据链路访问方法	254
13.1.1	BSD 包过滤器	254
13.1.2	DLPI	254
13.1.3	SOCK_PACKET	254
13.1.4	libpcap	255
13.2	libpcap 应用	255
13.2.1	libpcap 库函数	255
13.2.2	libpcap 数据结构	258
13.2.3	过滤程序	258
13.3	数据链路访问实例	259
13.4	小结	264
第 14 章	多接口设计	265
14.1	单个服务器绑定到多个接口	265

14.2	多个服务器绑定到多个接口.....	269
14.3	小结.....	274
第 15 章	路由套接字.....	275
15.1	创建路由套接字.....	275
15.2	读写路由套接字.....	275
15.3	读取路由信息.....	277
15.4	路由套接字实例.....	279
15.5	小结.....	282
第 16 章	简单路由器实例分析.....	283
16.1	设计专用路由器.....	283
16.2	实现专用路由器.....	286
16.2.1	捕获数据包: myCap 类和 myCapIP 类.....	286
16.2.2	查询系统路由: myRoute 类.....	293
16.2.3	发送 IP 包: myRaw 类.....	297
16.2.4	封装串口通信: SerialComm 类.....	300
16.2.5	处理专用数据传输网络协议: myDevice 类.....	300
16.2.6	同时发送和接收: sendThr、recvThr 和 myRouter 类.....	302
16.3	小结.....	310
附录 A	套接字 Wrapper 类源程序.....	311
附录 B	串口通信类源程序.....	337

第一部分 网络基础

第 1 章 Unix 系统基础

本章介绍 Unix 系统、常用 Unix 网络命令及软件开发环境，并通过简单的实例说明网络程序开发的过程。

1.1 Unix 系统概述

1.1.1 Unix 系统的历史

Unix 系统是 20 世纪 70 年代由 AT&T 的 Bell 实验室开发的。到目前为止，Unix 有两大流派：那就是 AT&T 的 System V 与 BSD (Berkeley Software Distribution)。SVR4 是两大流派融合后的产物。1991 年底，与 System V 针锋相对的 Open Software Foundation 推出了 OSF/1。其中 SVR4 融合了 System V、BSD、SunOS，是各种 Unix 中的主流。

1.1.2 Unix 系统的特点

Unix 问世以来，经过几十年的发展，现已成为功能最为强大和稳定的网络操作系统。它具有如下特点：

- 真正的多任务、多用户操作系统
- 可移植性强
- 完整的网络功能(TCP/IP 网络支持)
- 虚拟内存
- 共享库

1.1.3 Unix 系统的体系结构

Unix 系统分为 3 个层次：用户、核心以及硬件。其中系统调用是用户程序与核心间的接口，通过系统调用进程可由用户模式转入核心模式，在核心模式下完成一定的服务请求后再返回用户模式。系统调用接口通过库把函数调用映射成进入操作系统所需要的原语。

系统调用是在特权方式下运行的，可以存取核心数据结构和它所支持的用户级数据。系统调用的主要功能是用户可以使用操作系统提供的有关设备管理、文件系统、进程控制、进程通信以及存储管理方面的功能，而不必了解操作系统的内部结构和有关硬件的细节，从而减轻用户负担和保护系统，并提高资源利用率。

1.1.4 Unix 系统的地址空间

Unix 虚拟地址空间分为两个部分：用户空间和系统空间。在用户模式下只能访问用户空间，而在核心模式下可以访问系统空间和用户空间。系统空间在每个进程的虚拟地址空间中都是固定的。典型的 Unix 系统地址空间分配见图 1.1。

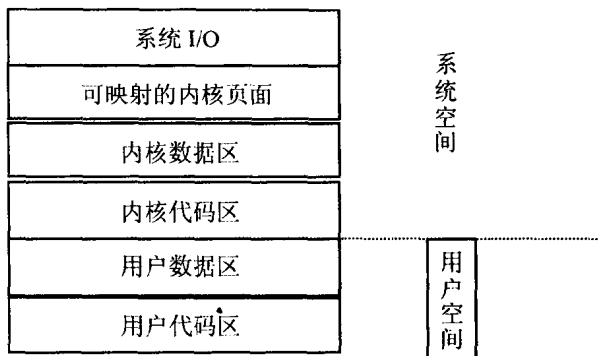


图 1.1 典型 Unix 系统地址空间分配

每个进程有自己的虚拟地址空间，对虚拟地址的引用通过地址转换机制转换为物理地址的引用。进程只能访问自己的地址空间所对应的页面，而不能访问或修改其他进程的地址空间对应的页面。Unix 通过虚存管理机制实现了这种保护。

1.1.5 POSIX 标准

POSIX 表示可移植操作系统接口 (Portable Operating System Interface)。电气和电子工程师协会 (IEEE) 开发 POSIX 标准，是为了提高 Unix 环境下应用程序的可移植性。

目前，大多数 Unix 系统都支持该标准。

1.2 常用 Unix 网络命令

在网络编程中，为了调试程序，经常会用到一些网络命令，以查看或修改网络配置、监视网络运行。以下介绍几种常用 Unix 网络命令。

1.2.1 ping

ping 检测主机连接状况。

1. 命令格式

```
ping [hostname | IP address]
```

2. 例子

```
ping 192.9.200.1
```

该命令用于检测本机是否与主机“192.9.200.1”连接。如果收到远程主机（IP 地址为 192.9.200.1）的回应，则说明本机与远程主机通信状态良好，否则存在网络故障。如果存在网络故障，应检查网络设备的连接及配置。

1.2.2 netstat

netstat 命令显示与网络有关的各种数据结构。如显示网络连接、路由表和网络接口信息。

1. 命令格式

```
netstat -[rli][n]
```

-r: 显示路由表，同 route -e。

-i: 显示所有网络接口的信息，格式同 ifconfig -e。

-n: 以网络 IP 地址代替主机名称，显示出网络连接情形。

2. 例子

显示所有网络接口的信息:

```
$ netstat -i -n
```

Name	Mtu	Net/Dest	Address	Ipkts	Ierrs	Opkts	Oerrs	Collis	Queue
lo0	8232	127.0.0.0	127.0.0.1	380	0	380	0	0	0
eth0	1500	192.9.200.0	192.9.200.1	408	0	504	0	0	0

显示路由表:

```
$ netstat -r -n
```

```
Routing Table: IPv4
```

Destination	Gateway	Flags	Ref	Use	Interface
224.0.0.0	127.0.0.1	U	1	0	lo0
127.0.0.1	127.0.0.1	UH	20	440	lo0
192.9.201.0	192.9.200.2	UG	1	500	eth0

1.2.3 ifconfig

ifconfig 显示当前有效网络接口的状态。

1. 命令格式

```
ifconfig [接口]
```

☞ 注意: 如以单个接口作为参数, 它只显示给出的那个接口的状态; 如果给出一个 -a 参数, 它会显示所有接口的状态, 包括那些停用的接口。

2. 例子

```
ifconfig -a
```

显示所有网络接口。

1.2.4 route

`route` 对内核的 IP 路由表进行操作。它主要用于给那些已经用 `ifconfig` 配置过的接口指定主机或网络设置静态路由。

1. 命令格式

```
route [add|del] [-net|-host] target [gw Gw] [netmask Nm] [metric N]
      [[dev] If]
route [-CFvnee]
```

☞ 注意：当使用了 `add` 或 `del` 选项时，`route` 将修改路由表。如果没有这些选项，`route` 显示当前路由表的内容。

2. 例子

```
route add -net 192.9.200.0 netmask 255.255.255.0 eth0
```

给路由表添加一条指向网络 192.9.200.* 的路由，其通过接口为 `eth0`。

```
route -e
```

显示当前路由表的内容。

1.2.5 tcpdump

`tcpdump` 用于转储网络上的数据流。它可显示出在某个网络接口上，匹配布尔表达式 `expression` 的报文。

1. 命令格式

```
tcpdump [-adeflnNOPqStvx] [-c count] [-F file] [-i interface]
        [-r file] [-s snaplen] [-T type] [-w file] [expression]
```

-a: 把网络和广播地址转换成名字。

-c: 当收到数目为 `count` 个报文后退出。

-i: 监听 `interface`。

-r: 从 `file` 中读入数据报(文件是用 `-w` 选项创建的)。如果 `file` 是 "-", 就读取标准输入。

-s: 从每个报文中截取 `snaplen` 字节的数据。

-x: 以十六进制数形式显示每一个报文。

`expression`: 用来选择要转储的数据报。如果没有指定 `expression`, 就转储网络的全部报文。否则, 只转储相对 `expression` 为真的数据报。