高性能网络协议还原平台的研究

贾荣来 叶建伟
(哈尔滨工业大学计算机学院网络与信息内容安全中心 黑龙江 哈尔滨 150000)

摘 要 在计算机网络日益发展的今天,网络上的信息传播正在逐步取代传统媒体,因而计算机网络上的安全问题也越来越受到人们的重视。针对当前大流量网络的普及和多核处理器的广泛应用,提出并实现一种高效的网络数据包重组还原平台。该平台完全工作在用户空间,主要对以旁路监听方式下在网络链路捕获到的网络数据包进行重组及协议还原。扼要地介绍协议还原所涉及到的数据包捕获、数据包重组以及应用层协议还原等关键技术。实验证明,该平台能够高效地将网络数据流还原到 TCP 层,并可根据需要加入多种应用层协议还原模块,具有很好的可扩展性。

关键词 网络安全 数据包重组 协议还原 可扩展性

中图分类号 TP393 TN915.08 **文献标识码** A DOI: 10.3969/j.issn.1000-386x.2013.01.065

ON HIGH-PERFORMANCE NETWORK PROTOCOLS RESTORATION PLATFORM

Jia Ronglai Ye Jianwei
(*Computer Network and Information Security Research Center, School of Computer Science anf Techndgy, Harbin Institute of Technolgy, Harbin 150000, Heilongjiang, China*)

Abstract With the increasing development of computer network, dissemination of information on Internet is gradually replacing the traditional media. Therefore, the security issue in Internet is attracting more and more attentions. In view of the popularisation of large flow networks and the extensive application of multi-core processor, in this paper we present and implement an effective reconstruction and restoration platform for network data packets, which is completely worked in user space. The platform mainly restructures the network data packets captured in internet link on bypass monitoring mode and restores the protocols. This paper briefly introduces some key technologies related to protocol restoration such as the packet capture, packet reconstruction as well as application layer protocol restoration, etc. Experiment indicates that this platform can efficiently restore the network data flow to TCP layer. Also, this platform may add a variety of application layer protocols restore modules according to the need, and has good scalability.

Keywords Network security Packet reconstruction Protocol restoration Scalability

0 引 言

随着互联网技术的高速发展以及高速网络的大规模铺设,对网络协议还原技术提出了新的挑战,传统的 TCP/IP 协议栈及现有的协议还原技术已经无法满足当前对网络内容安全的需求。本文对网络协议栈以及协议还原技术[1-3]进行了进一步的研究,提出并实现了一个高性能网络协议还原平台。

1 相关工作

协议还原技术首先需要对网络上传输的数据包进行捕获,然后通过软件实现一个模拟的 TCP/IP 协议栈,将这些捕获到的网络数据包进行数据重组,进而得到应用层信息。

Libnids[4] 是由 Rafal Wojtczuk 等开发、用于网络入侵检测开发的专业编程接口,它提供了 TCP 数据流重组功能,以及 IP 分片进行重组的功能,所以对于分析基于 TCP 协议的各种协议 Libnids 都能胜任。Libnids 实际上是一个可以将网络数据包还原到 TCP 层的 TCP/IP 协议栈。

当前多数协议还原技术都基于 Libnids。它们存在的主要问题是采用单进程串行处理模式、内存管理效率低,从而导致网络吞吐量较小。

2 关键技术

2.1 数据包捕获

Libpcap[5] 是 Unix/Linux 平台下的网络数据包捕获的函数库。它是一个独立于系统的用户层包捕获 API 接口,为底层网络监听提供了一个可移植的框架。

在网卡的缺省工作模式下,Libpcap 只能捕获到广播数据包和目的 IP 是本机的数据包。如果我们需要捕获流经网卡的所有数据包,只需要将网卡设置为混杂模式,也可以通过设置 BPF 过滤规则来获取自己需要的数据包。

2.2 数据重组

数据重组包括 IP 分片重组和 TCP 会话重组,这两部分参考

收稿日期: 2012-05-02。贾荣来,硕士,主研领域: 计算机网络与信息安全。叶建伟,讲师。

了 Libnids 的实现，与 Libnids 不同的是：① 在本系统实现中采用的多线程设计，而 Libnids 采用的单进程模式；② IP 分片重组中直接于存储数据包的内存缓冲区中进行，无需开辟新的内存空间用于存储 IP 分片；③ TCP 会话重组中采用内存池方式对数据包进行存储管理。这样实现的目的是解决数据包处理过程中小块内存频繁申请释放所带来的低效率问题。

2.3 内存管理

内存的分配与回收算法有很多详尽的讨论，在 Linux 系统或标准 C 语言环境下，使用的是内存分配与回收函数 malloc 和 free。但这种内存分配方式存在以下几个问题，一是在多线程环境下的互斥会造成 malloc 的性能随着线程数增多而下降，二是当频繁使用时会造成大量的内存碎片并进而降低性能，三是 malloc 和 free 函数为系统调用，耗时较长，频繁调用会降低系统性能。因而对于一个高速的数据处理系统需要重新设计内存管理的策略。

在 TCP/IP 协议中，网络层和数据链路层都对数据包大小有明确限制，通过对大量网络数据包的大小进行捕获统计，发现数据包大小比较集中在 1.5kB 左右和 64B 左右，在 TCP 流重组处理中每次对内存大小的需求有明显的局部性特征，大部分集中在 4kB 到 8kB 之间，因此在进行内存管理时可以利用此特性，以提高内存分配的效率。在该内存管理中包含两个主要部分，一部分用于存储网络数据包的内存缓冲区，一部分用于 TCP 流重组的内存池。

首先是创建一个较大的内存缓冲区，该内存缓冲区由很多个大小固定的内存块构成，其中每一个内存块用于存储一个网络数据包。该内存缓冲区可以由所有线程进行操作，数据包捕获线程将捕获到的数据包放入该内存缓冲区，数据包任务线程则从该内存缓冲区中取得数据然后进行重组操作。

然后是 TCP 流重组中的内存池设计，其结构如图 1 所示，该内存池是由 N 个内存池链表组成的，当一个内存池满了以后，就会从下一个内存池中提取空间来使用。其中 last 表示当前数据区域的已经使用的数据的结尾；end 表示当前内存池的结尾；next 表示下一个内存池；failed 表示申请空间时在本内存池中失败的次数；max 表示本内存池中最大可用空间；current 表示当前正在使用的内存池；cleanup_pt 是一个函数指针，其中保存着需要清理的数据指针以及相应的清理函数，让内存池销毁或其它需要清理内存池的时候，可以调用此结构体中的 handler。

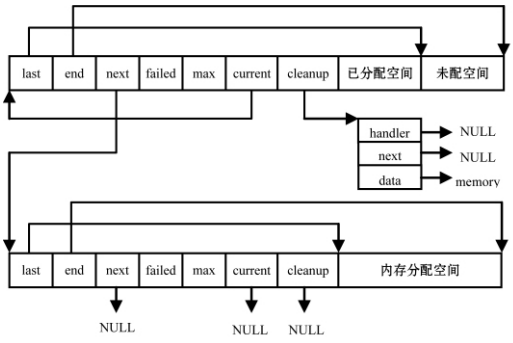

图 1 TCP 流重组内存池

2.4 系统并行化

在单核处理器时代，虽然程序是可以在处理器上并发执行的，但对于同一个程序而言，只能串行地运行在处理器上。随着多核处理器的普及和多线程编程技术的出现，使得同一个应用程序能够以多任务的方式并行运行在不同的处理器内核上，防止出现处理器核心有闲有忙的情况，充分发挥了多核处理器的强大处理能力。本系统采用多线程程序设计[6]，一个线程负责数据包的捕获，其他线程（称为任务线程）负责数据包的处理。其中任务线程数可以通过配置文件设定。

（1）任务分配

在本系统的实现中，由数据包捕获线程将捕获到的数据包放入内存缓冲区，然后由任务分发器交给各个任务线程处理。由于捕获到的 IP 数据包有可能是分片的，如果同属于一个 IP 报文的不同 IP 分片被分发给了不同的线程处理，就会导致 IP 分片重组失败；同样的，如果属于同一个连接的不同 TCP 报文被分发给了不同的线程，也会导致 TCP 流重组的失败。因此必须保证属于同一个连接的数据包必须交由同一个线程进行处理。在本实现中采用 hash 算法，根据数据包所属二元组信息（源 IP 和目的 IP）将属于同一连接的数据包分发给同一线程处理，既防止了上述问题的出现，又最大限度地保证了各个线程之间的负载均衡问题。

（2）线程间通信及其无锁化算法的实现

在该系统中，各个线程之间是一个单生产者-多消费者模型关系，其中内存缓冲区是临界资源。线程间通信的传统解决方法是采用锁机制来保护临界资源，但这样会导致临界区无法并行执行，进入临界区的线程就需要等待，从而带来效率的下降。本小节实现了一个无锁算法，保证各个进程之间能够在无锁环境下并行执行。

对于每一个任务线程，我们给其分配两个循环队列，由 DR、FR 指向，其中 DR 所指向的队列中存放的是该线程将要处理的数据包所在内存缓冲区中的地址，FR 所指向的队列中存放的是该线程已经处理过的数据包所在内存缓冲区中的地址。当数据包捕获线程捕获到一个数据包并将其放入内存缓冲区时，同时将该数据包在内存缓冲区中的地址放入指定线程（由任务分配算法计算出）的 DR 队列中，相应的任务线程则根据 DR 内容从内存缓冲区读取数据包并处理，处理结束后将该数据包的内存地址放入 FR 队列中。数据包捕获线程的另一个任务就是读取每一个线程的 FR 队列，并释放对应于内存缓冲区中的数据包。数据包捕获线程和任务处理线程的工作流程示意如图 2 和图 3 所示。

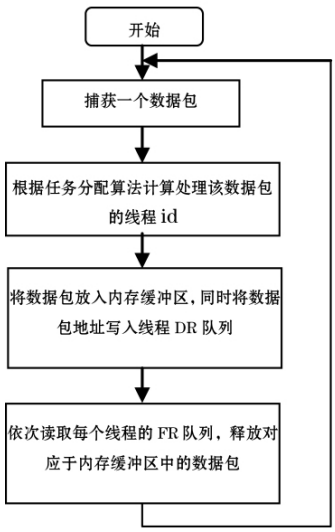

图 2 数据捕获线程流程

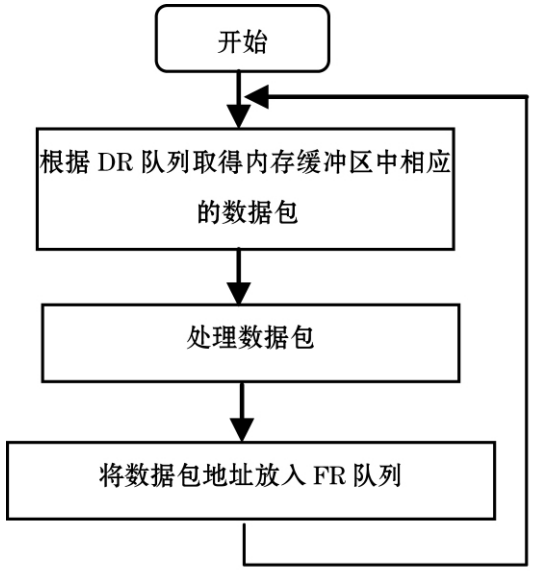

图 3　任务线程流程

3　系统实现及性能评估

3.1　系统实现

系统框架如图 4 所示，本系统采用模块化编程思想，由数据包捕获模块、IP 分片重组模块、TCP 流重组模块及应用层协议还原模块组成。应用层协议模块以压栈的形式加入到系统中，因而可以根据需要编写不同应用层协议的还原模块并加入到系统中。

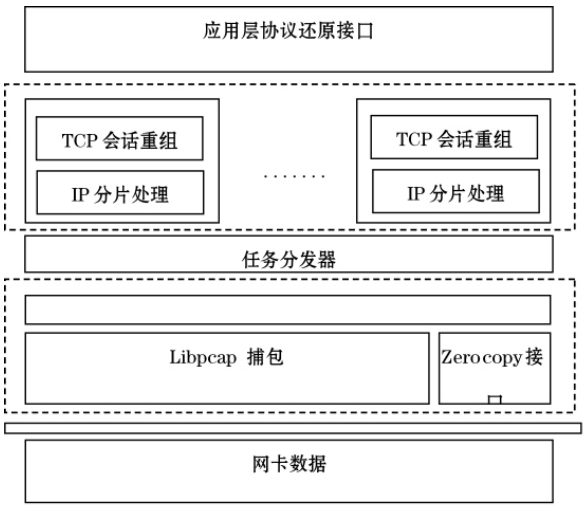

图 4　系统框架

系统工作流程是: 系统首先通过循环调用 libpcap 函数库的捕包函数 pcap_next 函数进行捕包，该函数返回指向捕获到的网络数据包的内存地址，每当捕获一个数据包，我们将此数据包拷贝到内存缓冲区中保存下来，然后通过任务分发器根据数据包的二元组信息将数据包交由不同的线程进行并行处理，IP 分片重组模块负责进行分片重组，最后将重组后的 IP 数据包交由 TCP 流重组模块进行 TCP 会话重组，最后，由应用层协议还原模块进行应用层协议还原并根据需要进行存储。系统工作流程如图 5 所示。

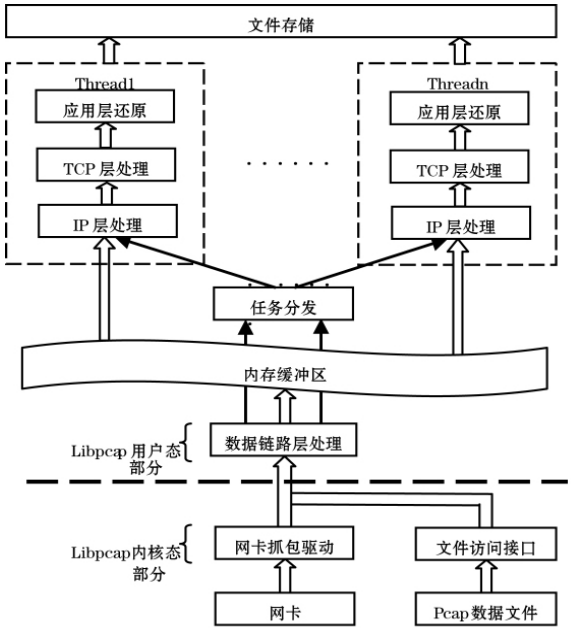

图 5　系统流程图

3.2　实验环境及实验方法

实验所用测试服务器带有 25GB 内存和两个 Quad-Core AMD Opteron(tm) Processor 2382 处理器，每个处理器有 4 个运行在 2.61GHz 的核心，采用 64 位 Redhat Enterprise Linux Server release 5. 4 (Tikanga) 系统，网卡型号为 nVidia Corporation MCP55 Ethernet (rev a3) 。测试数据为局域网网关真实网络数据。

我们通过 BPF 规则限定只捕获 TCP 数据包，并在数据包处理过程中加入适当延迟，然后统计成功还原到应用层的数据包占 Libpcap 捕获的原始数据包的百分比，来验证该协议还原平台的性能。作为参照，我们将在同一个网络中同时运行基于 Libnids 的协议还原测试程序。

3.3　实验步骤及实验结果分析

实验中分别将协议还原平台任务线程数设定为 1、2、4、7、10，测试并统计实验结果。结果如图 6 所示，其中横轴代表系统处理数据包过程中的延迟，单位为 100us，纵轴代表成功还原到 TCP 的数据包占 Libpcap 捕获的原始数据包的百分比。

由图 6 可见，在延迟很小的情况下，该协议还原平台与 Libnids 均可保证 100% 的 TCP 数据还原率，但随着延迟的增加，该协议还原平台的性能逐渐突显出来。

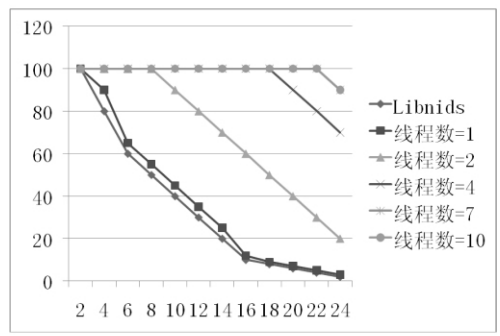

图 6　实验结果

(下转第 266 页)

6 进一步的研究

6.1 一个实体名的多个变体

使用自然语言来表示域名为域名的表达提供了极大的丰富性和多样性。NatureDNS 除了占用了“-”(hyphen)作为标志符和分节符,其余所有的语法形式和可打印的字符形式都是被允许的。这可能面临一个问题是,具体到一个实体名,可能有多种表达方式,如同一个人有全名、昵称、俗称、简称一样,到底使用哪个来注册,还是全部都要注册需要制定的相应的规则。

6.2 Unicode 进化带来的影响

由于兼容国际化语言,经过权衡我们选择了 Unicode 统一编码字符集,并推荐了 Punycode 和 Base62x 两个将 Unicode 字符 ASCII 化的编码方案。

然而,Unicode 本身也在进化中,这种进化尽管幅度很小、步子很慢,但可能因此影响到 NatureDNS 的稳定性——作为互联网基础核心服务,DNS 对稳定性的要求无论多么苛刻都不为过。对此,随着 Unicode 的日趋成熟,其变化频次逐渐减少;另外只要 NatureDNS 有相应的更新修正机制即可应对。

参 考 文 献

[1] Klensin J. IETF RFC3467 [OL]. [2012 - 06 - 15]. http://tools.ietf.org/html/rfc3467.

[2] Liu Z, Liu L, et al. Dot-base62x: A Compact Textual Representation of IPv6 Address for Clouds [C]//UCC 11 Proceedings of the 2011 Fourth IEEE International Conference on Utility and Cloud Computing, Melbourne, 2011.

[3] Root Zone Database [OL]. [2012 - 06 - 15]. http://www.iana.org/domains/root/db/.

[4] New gTLD Reveal Day - Applied-for Strings [OL]. [2012 - 06 - 15]. http://newgtlds.icann.org/en/announcements-and-media/announcement-13jun12-en.

[5] Mockapetris P. IETF RFC882 [OL]. [2012 - 06 - 15]. http://tools.ietf.org/html/rfc882.

[6] Pang J, Hendricks J, et. al. Availability, usage, and deployment characteristics of the domain name system [C]//IMC 04 Proceedings of the 4th ACM SIGCOMM conference on Internet measurement, Sicily, Italy, 2004.

[7] Global Domain Registry Statistics [OL]. [2012 - 06 - 15]. http://www.webhosting.info/registries/.

[8] Domain Name Length Allocation [OL]. [2012 - 06 - 15]. http://www.searchengineknowledge.com/domains/length.php.

[9] The World's Longest Domain Name [OL]. [2012 - 06 - 15]. http://www.oreillynet.com/onlamp/blog/2005/06/the_worlds_longest_domain_name.html.

[10] Liu D, Chen Y, Xie K, et al. Research on the Structures and Resolutions of Internet Namespaces [J]. Journal of Software, 2005, 16(8): 1445-1455.

[11] Stockbrand B. IPv6 in Practice —A Unixer's Guide to the Next Generation Intern [M]. Verlag Berlin Heidelberg: Springer, 2007: 22 - 29.

[12] Google Public DNS: world's largest DNS service [OL]. [2012 - 06 - 15]. http://9to5google.com/2012/02/14/google-public-dns-worlds-largest-dns-service-with-70-billion-requests-a-day/.

[13] Mao W, Wang Y, Wang F. The New Generation Technologies of Internet Resources Naming and Addressing [J]. Application Research of Computers, 2004, 21(4): 233 - 235, 250.

[14] Zhang H, Deng X, Qian H. Analysis of Internationalized Domain Name System [J]. Journal of Computer Applications, 2002, 22(10): 9 - 11, 2002.

[15] IETF RFC 3492. Punycode: A Bootstring encoding of Unicode for IDNA [OL]. [2012 - 06 - 15]. http://tools.ietf.org/html/rfc3492.

[16] ICANN. New Generic Top-Level Domains [OL]. [2012 - 06 - 15]. http://newgtlds.icann.org/en/about.

[17] NetC. New gTLD [OL]. [2012 - 06 - 15]. http://www.net-chinese.com.tw/new_gtld/new_gtld.asp.

[18] IANA. Repository of IDN Practices [OL]. [2012 - 06 - 15]. http://www.iana.org/domains/idn-tables.

[19] Tuwang. Google 一共有多少个域名 [OL]. [2012 - 06 - 15]. http://www.tuwang.org/394.html.

[20] Gan J, Huang L. The Research on Translating of Domain name and Practice of Server Configuration [J]. JOURNAL OF YULIN NORMAL UNIVERSITY: Natural Science, 2007, 28(5): 136 - 141.

[21] Liu Z, Liu L, Hill R, et al. Base62x: An alternative approach to Base64 for non-alphanumeric characters [C]//Fuzzy Systems and Knowledge Discovery (FSKD), 2011 Eighth International Conference, Shanghai, 2011.

[22] 人民网. 两个开心网之争 [OL]. [2012 - 06 - 15]. http://media.people.com.cn/GB/40606/9355412.html.

(上接第 255 页)

4 结 语

本文设计并实现了一个高性能网络协议还原平台,充分利用了多核处理器的强大处理能力,实现了系统内部数据包的并行处理。相比于已有的网络协议还原系统,在性能上有了很大的提升。

参 考 文 献

[1] Wright G R, Richard Stevens W. TCP/IP 详解,卷 1: 协议 [M]. 北京: 机械工业出版社,2000.

[2] Wright G R, Richard Stevens W. TCP/IP 详解,卷 2: 实现 [M]. 北京: 机械工业出版社,2000.

[3] 郭士秋. TCP/IP 协议堆栈运用基础 1: IP 协议体系 [M]. 北京: 电子工业出版社,2002.

[4] Libnids. An Implement ation of an E-Component of Network Intrusion Detection System [EB/OL]. [2010 - 03 - 04]. http://libnids.sourceforge.net/.

[5] Libpcap. Packet Capture Library [EB/OL]. [2012 - 01 - 01]. http://www.tcpdump.org/.

[6] David R Butenhof. POSIX 多线程程序设计 [M]. 北京: 中国电力出版社,2003.

[7] 陈莉君. Linux 操作系统内核分析 [M]. 北京: 人民邮电出版社,2000.

[8] 刘文涛. 网络安全开发包详解 [M]. 北京: 电子工业出版社,2005.

[9] 刘文涛. Linux 网络入侵检测系统 [M]. 北京电子工业出版社,2004.

[10] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, et al. 算法导论 [M]. 北京: 机械工业出版社,2006.

[11] Douglas E Comer. 用 TCP/IP 进行网际互联(卷 1) [M]. 北京: 电子工业出版社,2001.

[12] 谢希仁. 计算机网络 [M]. 北京: 电子工业出版社,2005.