基于通用多核平台的入侵检测系统研究

陈 诚[1] 孙一品[1] 钟求喜[1] 侯一凡[2]
(国防科学技术大学计算机学院 长沙 410073)[1] (中国人民解放军信息工程大学测绘学院 郑州 450052)[2]

摘 要 为应对网络流量快速增长问题,提出一种基于通用多核平台的入侵检测系统结构。在系统设计基础上,分析、验证了硬件平台、资源分配模式和流量特征等关键因素对系统处理性能的影响。实验表明,网络流量的流数、单位时间内报文包数等指标对系统性能的影响更大;在启用多核处理器超线程技术并将检测引擎与CPU绑定时,系统性能可以得到有效提高;系统易于实现,性价比高。

关键词 入侵检测,多核,超线程
中图法分类号 TP393 **文献标识码** A

Research on Intrusion Detection System Based on Commodity Multi-core Platform

CHEN Cheng[1] SUN Yi-pin[1] ZHONG Qiu-xi[1] HOU Yi-fan[2]
(College of Science, National University of Defense Technology, Changsha 410073, China)[1]
(Institute of Surveying and Mapping, The PLA Information Engineering University, Zhengzhou 450052, China)[2]

Abstract To deal with the rapid increment of network traffic, an Intrusion Detection System (IDS) based on commodity multi-core platform was proposed. This paper evaluated some critical factors for the system performance, such as hardware, resource-assigning and network traffic features. Extensive experiments demonstrate that number of traffic flow and pps index have larger impact on system performance. The ids performance can be improved obviously by activating the Hyper-Threading of multi-core processor and binding the detection engine with the CPU core. Our system is easy to be realized and has low price-performance ratio.

Keywords Intrusion detection, Multi-core, Hyper-threading

1 引言

入侵检测系统是维护网络安全必不可少的基础设施之一。近几年来,随着网络用户的大量增加和互联网的快速发展,骨干网络所承载的数据流量也越来越大[1]。单纯依靠增加设备数目来处理骨干链路流量不仅提高了部署成本,也增大了管理维护的难度,因此,设计高性价比的入侵检测设备成为网络安全防护的迫切需求。

入侵检测系统的性能瓶颈主要源于报文捕获和检测匹配需要消耗大量的计算资源,商用入侵检测系统多基于专用硬件加速技术,设备成本高。随着CPU多核技术快速发展,诸如Intel Core 2 Duo i7系列[2]、Intel Xeon E7系列[3]等四核、八核处理器已经大规模产生,通用计算机平台的计算能力大幅增加,且价格低廉,为研究高性价比的入侵检测系统提供了契机。然而,传统的入侵检测系统主要基于单核平台,难以充分利用多核平台的计算能力。本文提出了基于通用多核平台的入侵检测系统结构,在原型系统基础上进一步分析并验证了影响入侵检测系统的主要因素。

2 相关工作

基于单核平台的传统入侵检测系统,难以应对网络流量的快速增长。基于通用多核平台提升入侵检测系统性能,成为了入侵检测的研究热点。国内外众多专家学者,针对多核平台入侵检测做了大量的研究和探索[4-8]。

文献[4]使用OpenMP并行编程技术来让入侵检测系统的主线程以多个副本线程fork-join的方式并行运行在多核平台上,通过把检测任务分配给各个处理核心上的副本线程来提升系统的整体处理能力,其优点在于开发简单,缺点是需要以人工的方式对程序进行加解锁来解决数据冲突以确保程序功能的正常,而加解锁又会增加程序的等待时间,增加运行开销。Snort3.0[9]提供了一个snort安全平台(SnortSP),通过抽象层将SnortSP与引擎结合起来,其默认支持多线程,处理流量的时候,各个引擎模块可以同时运行,能够充分利用多核计算的优势,但其实际运行效率较低。Intel在提高多核平台入侵检测系统的缓存命中率方面做了些探索,他们使用流水线和流绑定技术[5]修改传统的snort程序,将其拆解为hash分包和检测匹配两个模块,流水化地运行。他们把每个检测匹配引擎限制在一个核上,再把hash引擎分出来的网络数据流量分派到其所对应的检测匹配引擎上进行处理,这个方法因提高了缓存命中率而提升了系统吞吐量,但其缺点在于要对snort做较大改动,并且hash算法不能保证流的完整性。文献[6-8]对snort的修改类似于Intel[5]。不同点在于

到稿日期:2011-03-23 返修日期:2011-06-27 本文受国家自然科学基金项目(61003303)资助。
陈 诚(1985—),男,硕士生,主要研究方向为入侵检测,E-mail:test_manager@126.com。

检测匹配引擎：文献[6]分别设置了针对 TCP、UDP 流的专用检测引擎以及针对其他流的通用检测引擎，但是仅仅止步于模型阶段；文献[7]全部使用通用检测引擎，遗憾的是，并未给出系统处理能力的数据；文献[8]允许一个流在不同的检测线程上面进行切换，但是开发代价较大。

与上述现有工作[4-8]不同，本文将传统入侵检测系统作为单个检测引擎，在通用多核计算平台上同时启动运行多个检测引擎来提高系统的整体处理能力。相比已有方案，本方案开发代价小、可扩展性高，而且能更有效地利用多核平台的计算能力。

3 基于通用多核平台的入侵检测系统

3.1 系统设计

当前，主流服务器乃至个人主机均采用多核处理器，本文统称为通用多核平台。如图 1 所示，多核处理器[10]是指，在一个单独的芯片上封装多个独立的计算核心（Core0 — Core3），但操作系统会把这些计算核心识别为多个独立的处理器。多核处理器能够在给定的时钟周期内执行相对较多的任务，但在运行基于单核的传统应用的时候，难以充分发挥其整体计算能力。

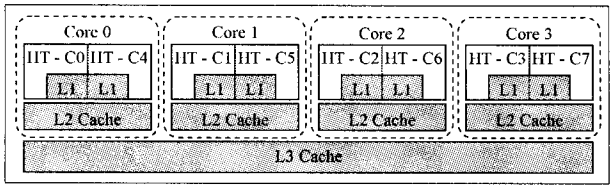

图 1 支持超线程的多核处理器示意图

传统的入侵检测系统是基于单核处理器的单检测引擎系统，以开源入侵检测程序 snort 为例，其工作流程如图 2 所示：snort 获取一个报文后，首先进行的是解码操作，将报文按照协议类型重新装填到 snort 的数据格式中，然后转到预处理插件，预处理之后的报文部分直接转到报警步骤，部分则是经规则匹配阶段再转到报警步骤，最后是输出结果，如果报文中有恶意数据，则会触发报警并被记录下来。

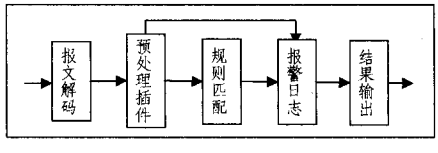

图 2 snort 工作流程

目前，传统串行程序在多核平台上的优化思路有两种：模块流水化与模块并行化。前者是指把串行程序拆解成若干个不同的功能模块，每个模块运行在一个处理核心上，再让各个模块流水运行；后者是指，在多个计算核心上同时运行多个相同的模块来处理同一类的数据。已有研究同时采用这两种技术对 snort 进行改进[5-8]，然而，值得指出的是，这些研究基本上止步于原型系统阶段。事实上，综合流水化、并行化技术改进 snort，需要对 snort 进行较大改动，不仅开发难度大，而且性能收益也并不理想。一方面，流水化之后，模块之间的协调同步开销将大幅增加。另一方面，不同模块的单位处理能力不同，特别是规则匹配模块的处理开销远大于其他模块[11]。各模块之间的流水部件个数设置、计算资源分配比较复杂，容易造成多核平台的计算资源空闲浪费。

针对上述问题，本文采用并行化思想，将单个 snort 程序视为一个匹配引擎，通过多引擎并行化提升系统整体性能。针对单个引擎难以应对千兆以上流量的问题，如图 3 所示，本文基于网卡虚拟化思想，将单个网卡虚拟成多个子网卡。修改网卡驱动程序，增加报文分流功能，在流保持的前提下，将流量均匀分配到各个子网卡。每个子网卡的流量由一个 snort 引擎处理。该架构屏蔽了底层分流对 snort 的影响，不需要改动 snort 主体代码，不仅开发代价小，可扩展性好，而且能够充分利用多核平台计算优势。

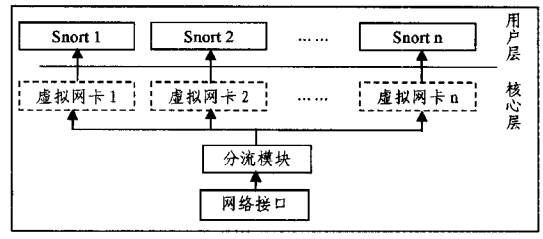

图 3 多核平台入侵检测系统设计图

3.2 性能分析

影响入侵检测系统性能的因素有很多，核心因素有硬件平台、计算资源分配模式、流量特征等。下面针对此 3 大关键因素进行分析，并将在第 4 节给出实验验证。

• 硬件平台要素

CPU 主频、处理核心数目、内存的带宽、大小是决定硬件平台计算能力的关键要素，对入侵检测系统性能的影响是显著的。值得关注的是超线程技术，它具有多核 CPU 平台可由用户决定是否启用的重要功能，其对入侵检测系统的影响值得分析。

超线程[12]是英特尔研发的一个技术，如图 1 所示，超线程技术可以在一个实体处理器中，提供两个逻辑线程（Core0 所对应的 HT-C0、HT-C4，Core1 所对应的 HT-C1、HT-C5 等），使其在逻辑上变成两个共享 2 级缓存虚拟处理器，让一般情况下闲置的计算资源可以得到有效利用，超线程技术在少量增加处理器晶体管面积的同时，大幅度提升了处理器的性能。但是，由于启用超线程的处理器实际上只有一个物理核心，一旦发生资源互抢，比如 cache 更新，整体性能可能会下降。

• 资源分配模式要素

单个检测引擎是否应该固定地绑定在某个处理核心上？理论上，这有利于避免检测引擎在处理核心之间的切换，减少进程的调度开销，从而提高入侵检测系统的性能。改进效果有待于实验验证。

• 流量特征要素

网络流量处理能力是评估入侵检测系统性能的关键依据，其主要特征如 pps、bps、流的数量等同样会影响最终评估结果。

1）pps 与 bps 的对比评估

pps 与 bps 均是衡量入侵检测系统负载的流量指标，这两个指标紧密相关，但并不完全一致。比如，网络流量 bps 的增加可能是因为 pps 的增长，也可能是因为报文长度发生了变化。对 snort 而言，特征规则大多限定了距离报文头部的

匹配范围，因此，报文长度的增加并不会显著增加匹配开销。换言之，pps 相对于 bps 能够更好地反映入侵检测系统的处理能力。但事实是否如此，还需实验来评估。

2)流的数目的影响

流是指一段时间内具有相同源目地址、源目端口以及协议的一组顺序数据包。为提高检测匹配的准确性，snort 使用预处理插件来维护流状态表。理论上，网络流量中流的个数越多，状态表也将越大，查表的开销也随之增加，从而可能影响入侵检测系统的处理能力。

4 实验分析

4.1 测试条件

为评估上述系统方案及各影响要素，本文使用配置 Intel Xeon E5540 多核处理器、16GB Kingston DDR Ⅲ内存的主机搭建原型系统。

测试数据采用骨干链路上采集的真实流量，其中共有 5615260 个流。采用 TcpReplay[13] 重放数据包，根据 TcpReplay 发出的报文数和 snort 有效处理的报文数来计算丢包率。使用 Intel Vtune Amplifier XE 2011[14] 来测量系统缓存失效率。

针对后续实验的侧重点不同，对真实网络流量做如下处理：

1)在第一、第二个实验中，未对原始真实流量做处理，直接使用。

2)针对于第三个实验，以报文字节长度为划分参数，从原始真实流量中分别提取出 7 组流量，并且为了尽量减少其他因素的影响，再分别从中提取出 UDP 报文，提取后的结果见表 1。

表 1 不同报文长度的 UDP 流量包

序号	报文长度范围
1	0～79kB
2	80～159kB
3	160～319kB
4	320～639kB
5	640～1279kB
6	1280～1400kB
7	1400kB 以上

3)针对于第四个实验，在保证流的完整性的前提下，从原始真实流量中提取出 3 个包，其所含的流的数目各不相同，分别是原始流量的 1/2、1/4、1/8。

4.2 测试结果

1)实验一 超线程技术有效性测试

本实验针对有无超线程的两种情况分别进行了 6 组实验，以位于 50Mbps～ 300Mbps 区间的 6 个不同的速度来重放网络流量，记录每一组实验中 snort 的丢包率以及相对应的缓存失效率。如图 4、图 5 所示，启用超线程以后，系统的缓存失效率高于未启用超线程的情况，但是其所对应的 snort 丢包率低于后者。缓存失效率的增加是因为两个 snort 检测引擎竞争使用缓存资源，将自己所需的数据读入缓存的同时淘汰了对方尚未使用的数据缓存。但是，因为超线程技术更加有效地利用了系统计算资源，所以其整体性能即 snort 丢包率，要优于未启用超线程的时候。

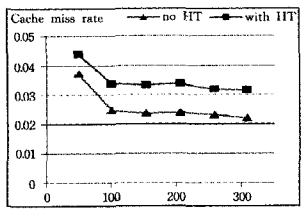

图 4 有无超线程的缓存失效率

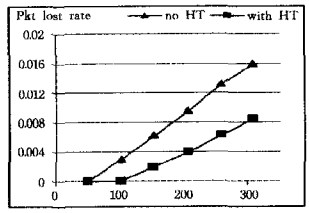

图 5 有无超线程的丢包率对比

对比

2)实验二 CPU 绑定必要性测试

实验一表明，超线程技术对于提高系统性能有帮助，所以这里在启用超线程的环境下，对有、无 CPU 绑定的情况进行对比测试，本实验中，以 300Mbps 的速度来重放流量，对两种情况分别做了 3 次实验，每一次都记录其缓存失效率及其相对应的 snort 丢包率，并取其平均值，汇总为表 2 中的数据。

表 2 有无 CPU 绑定的缓存失效率、丢包率对比

CPU 绑定	缓存失效率	snort 丢包率
是	0.028568299	0.02429425
否	0.030738723	0.03251413

从表 2 中可以看到，由于没有多余的检测引擎切换所导致的计算开销，进行流绑定后所得到的缓存失效率以及 snort 丢包率都要低于不进行流绑定的情况。也就是说，对流进行绑定，有利于提高系统性能。

3)实验三 数据流 pps 及 bps 敏感性测试

在本实验中，单独启动一个 snort 检测引擎，从 pps、bps 角度分别评估其对系统性能的影响。首先以 15kpps 的速度把提取出来的流量包以报文长度从小到大的顺序进行重放，并记录发送流量的 bps 值，因为流量的字节长度是从小到大变化的，所以 Mbps 也有类似变化，同时记录 snort 的平均 CPU 占用率，见表 3；然后选择表 3 数据中的中间点（折合 Mbps 为 56.95Mbps、加粗标识）作为依据，以 56.95Mbps 的速度再次按照之前的顺序重放流量包，并记录下对应的 pps 以及 CPU 占用率的数值，见表 4。

表 3 固定 pps

cpu_usage	pps	折合 Mbps
14.94%	14998	7.53
16.09%	14985	12.37
18.20%	14925	23.28
20.51%	14870.29	**56.95**
23.92%	14961	124.3
26.88%	15074.78	153.74
28.21%	14669.19	163.03

表 4 固定 bps

cpu_usage	折合 pps	Mbps
46.23%	114919	57.72
39.86%	69401.7	57.29
32.69%	36606.7	57.1
20.51%	14870.3	56.95
18.23%	6879.97	57.16
16.23%	5574.61	56.85
11.65%	5097.87	56.66

最后以两个表格中 Mbps 为 56.95 所在行的数据作为参考点（零点），来计算 CPU 占用率以及发包速率相对于零点的变化量，计算公式如下：

Y 轴（CPU 相对占用率）：$Y_i = Y_i - Y_0$

X 轴(发包相对速率)：$X_i=(X_i-X_0)\ /\ X_0$

计算结果见表 5。

表 5　CPU、发包速率相对变化

固定 pps		固定 bps	
y	x	y	x
25.72%	672.81%	−5.57%	−86.78%
19.35%	366.71%	−4.42%	−78.28%
12.18%	146.17%	−2.31%	−59.12%
0.00%	0.00%	0.00%	0.00%
−2.28%	−53.73%	3.41%	118.26%
−4.28%	−62.51%	6.37%	169.96%
−8.86%	−65.72%	7.70%	186.27%

其变化曲线如图 6 所示，其中三角形标记点曲线是固定 bps 的情况下不同的 pps 值所对应的 CPU 的变化值的曲线，从图 6 中可以看出，三角形标记点曲线的斜率，在第一、三象限都要大于方形标记点曲线，可见，相对于 bps，pps 的变化对 snort 的 CPU 占用率的影响要大。

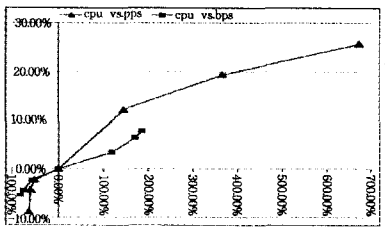

图 6　pps、bps 的变化率 vs CPU 变化

4)实验四　流数影响测试

本实验以 60kpps 的速率发送提取出来的含有不同数目流的数据包，这个发包速率是通过前期实验确定的，可以确保流数最少的流量，在 snort 接收端不会产生丢包。这里以相同的发包速率重放这 4 组流量，并测得系统丢包率以及相对应的 CPU 占用率，见表 6。

表 6　不同流数目所对应的 CPU 占用率以及系统丢包率

	流数	系统丢包率	CPU 占用率
N 个流	5615260	0.015578725	0.6853
1/2 N 个流	2649918	4.38755E-05	0.6055
1/4 N 个流	1316748	1.13103E-06	0.5921
1/8 N 个流	638975	0	0.5725

把这 4 组数据所对应的 CPU 占用率以及系统丢包率的变化曲线分别呈现在图 7、图 8 中。

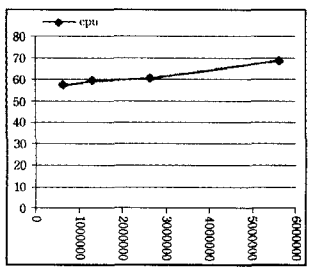

图 7　流数(x 轴) vs CPU 占用率(y 轴)

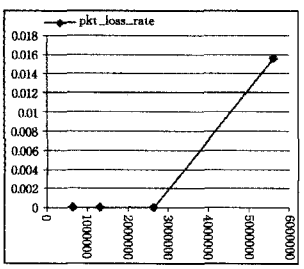

图 8　流数(x 轴) vs 系统丢包率(y 轴)

从图 7、图 8 可以看出，数据流中流的数量，对于 snort 的丢包率及其 CPU 占用率的变化有着很大的影响，并且从表 6 中可以看到，在 snort 的 CPU 占用率达到 59.21%时，系统就已经开始出现丢包，而并不是在到达满载 100%之后出现丢包。

5)实验五　综合性能测试

通过之前的 4 个实验，确定了各个因素对于入侵检测系统性能的影响程度。这里把本文所提出的多核平台入侵检测系统与传统入侵检测系统做对比测试。本实验中，发包速度为 38kpps～220kpps，进行两组实验，一组实验中，只启动一个 snort 进程，不使用超线程技术与流绑定；另外一组实验中，使用 CPU 的全部 4 个计算核心并使用超线程技术进行流绑定。图 9 是两者的丢包率的对比，其中 x 轴为发包速度，y 轴为丢包率，可见，多核平台入侵检测系统(三角形标记点曲线)比起传统入侵检测系统(方形标记点曲线)，在丢包率以及处理速度方面都有很大的提升。

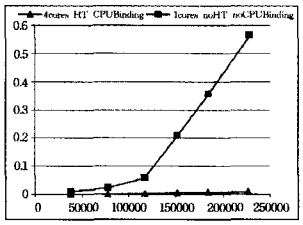

图 9　入侵检测系统丢包率对比

结束语　本文提出了一种基于通用多核平台的入侵检测系统，并分析验证了影响其性能的关键要素。后续将进一步优化底层报文捕获及分流算法，进一步提高原型系统的处理能力。

参 考 文 献

[1] CNNIC. 中国互联网络发展状况统计报告[OL]. http://www.cnnic.net.cn/dtygg/dtgg/201107/W020110719521725234632.pdf,2011.7

[2] Intel Core 2 Duo i7 [OL]. http://www.intel.com/zh_CN/products/processor/corei7/index.htm,2011

[3] Intel Xeon E7[OL]. http://www.intel.com/zh_CN/products/server/processor/xeone7/index.htm,2011

[4] 李永强. 多核技术在网络入侵检测中的应用研究[D]. 沈阳：东北大学，2009

[5] INTEL. Supra-linear Packet Processing Performance with Intel? Multi-core Processors[OL]. http://www.intel.com/technology/advanced_comm/311566.htm,2006

[6] Zhuang Z, Luo Y, Li M, et al. An Abstract Model for Intrusion Detection on Multi-Core Platform[C]// CHINAGRID '08. Washington, DC, USA: IEEE Computer Society, 2008: 202-208

[7] 庄卓俊. 基于多核平台的入侵检测系统的设计与实现[D]. 上海：上海交通大学，2009

[8] Schuff D L, Choe Y R, Pai V S. Conservative vs. optimistic parallelization of stateful network intrusion detection[C]// PPoPP '07. New York, NY, USA: ACM, 2007: 138-139

[9] Snort3.0 [OL]. http://www.snort.org/snort-downloads/snortsp/,2011

[10] Multi-core[OL]. http://www.intel.com/cn/technology/multi-core/index.htm,2011

[11] Schuff D L, Pai V S, Willmann P, et al. Parallel Programmable Ethernet Controllers: Performance and Security [J]. Network, IEEE, 2007, 21(4): 22-28

[12] Hyper-Threading [OL]. http://www.intel.com/cn/technology/platform-technology/hyper-threading/index.htm,2011

[13] Tcpreplay[OL]. http://tcpreplay.synfin.net/,2011

[14] Intel Vtune Amplifier XE[OL]. http://software.intel.com/en-us/articles/intel-vtune-amplifier-xe,2011