第35卷 第11期 计 算 机 工 程 2009年6月
Vol.35 No.11 Computer Engineering June 2009
·安全技术· 文章编号：1000—3428(2009)11—0120—03 文献标识码：A 中图分类号：TP393.08

Snort规则链表结构的改进与仿真

孙 敏，古晓明，张志丽
(山西大学计算机与信息技术学院，太原 030006)

摘 要： Snort 系统根据规则链表对捕获的数据包进行匹配，以发现攻击行为，规则链表结构的合理性在很大程度上影响检测速度。针对Snort规则链表结构中局部聚集的现象，对其按共性选项因式分解，将规则按所含选项的信息量进一步排序。在仿真平台OPNET上的模拟结果表明，改进后的规则链表结构能减少规则匹配时间。

关键词： 误用入侵检测；规则链表；因式分解；网络仿真

Improvement and Simulation of Snort Rule Chains Structure

SUN Min, GU Xiao-ming, ZHANG Zhi-li
(School of Computer & Information Technology, Shanxi University, Taiyuan 030006)

【Abstract】 For finding attacks, Snort matches the captured packets with rule chains, therefore the rationality of the rule chains influences the detection speed of Snort greatly. This paper factors out common options from the rule chains, for solving the problem of local accumulation. It sorts the rules according to information quantity of the options. Simulation results of the OPNET shows that the improved rule chains structure can reduce the time of rules matching.

【Key words】 misuse intrusion detection; rule chains; factoring; network simulation

Snort是一个著名的开放源码的误用入侵检测系统(IDS)，它捕获计算机系统和网络中传递的所有数据包，与已知的攻击模式(即规则)比较，从而发现来自网络内部和外部的攻击行为。

1 Snort规则链表结构分析

Snort[1]将所有已知的入侵行为以规则的形式存放在规则库中，每一条规则由规则头和规则选项2个部分组成。在初始化并解析规则时，分别生成4个不同的规则树：TCP, UDP, ICMP, IP，每一个规则树即一个独立的三维链表：规则头(Rule Tree Node, RTN)，规则选项(Optional Tree Node, OTN)和指向匹配函数的指针。RTN包含动作、协议、源(目的)地址和端口以及数据流向；OTN包含报警信息(msg)、匹配内容(content)等选项。当Snort捕获一个数据包时，首先分析该数据包属于哪个规则树，然后找到相匹配的RTN节点，最后向下与OTN节点进行匹配。每个OTN节点包含一组用来实现匹配操作的函数指针。当数据包与某个OTN节点匹配时，即判断此数据包为攻击数据包。

2 Snort规则链因式分解

2.1 OTN链因式分解

2.1.1 因式分解原理

Snort在某个规则树内部的匹配过程中，在找到相应的RTN节点后，将进行深度优先搜索，即对属于同一个RTN的OTN依次进行匹配。这种方法能使程序结构清晰，具有很好的可扩展性。但在具体实现中，即使多个OTN节点的一类选项具有相同的值，仍须对每一个OTN重复匹配[2]。若将这些OTN节点中具有相同值的一类选项(共性选项)提取出来，在匹配RTN之后、OTN之前，在宽度方向增加一次匹配操作，则可使规则链表的搜索效率得到优化[3]。

基于上述问题，本文提出一种将OTN节点因式分解的新方法，即对规则中的共性选项(common options)进行提取。因式分解结果是生成2种OTN：主OTN(primary OTN)和次OTN(secondary OTN)。主OTN存放共性选项；次OTN则只包含除了共性选项之外的特殊选项，指向含有共性选项的规则。在这个新结构中，其余不包含共性选项的规则作为普通OTN(ordinary OTN)，这些规则将不被因式分解。

假如有以下5条规则：

R1: alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg: "α"; flags:S,12; classtype:I; sid:1;)

R2: alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg: "β"; classtype: II; sid:2;)

R3: alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg: "σ"; flags:S,12; ack: 0; classtype:III; sid:3;)

R4: alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg: "θ"; flags:S,12; nocase; classtype:IV; sid:4;)

R5：alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg: "γ";flags:SF,12; classtype:V; sid:5;)

这些规则均为alert类，并且有相同的协议字段，相同的源、目的IP和PORT，所以，它们属于相同的RTN。图1左

基金项目： 山西省高校科技开发基金资助项目(200512G2)；山西大学科研基金资助项目(2005103)
作者简介： 孙 敏(1966－)，女，副教授，主研方向：计算机网络，网络安全；古晓明、张志丽，硕士研究生
收稿日期： 2008-10-06 **E-mail：** gxm820911@yahoo.com.cn

侧部分为原 Snort 规则链，右侧部分为因式分解后的规则链。在新的规则树中 flags 字段的匹配次数由原来的 4 次减少为 2 次。随着规则数量的增加，该方法将明显减少匹配次数。

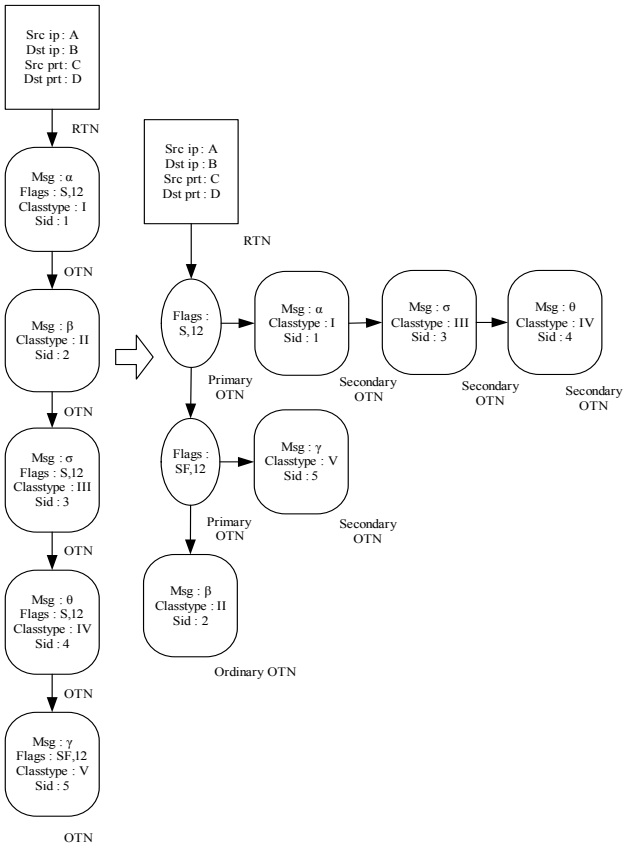

图 1　OTN 因式分解

2.1.2　因式分解实现方法

为完成因式分解，必须将同一 RTN 下的规则按共性选项分配到不同主 OTN 下作为次 OTN。本文引入一个用来计算每条规则权重的函数 F，将具有相同权重(即含相同共性选项)的规则放在同一个主 OTN 下。函数 F 计算方法如下：

(1)集合 $D=\{x_1, x_2,\cdots, x_n\}$，包含所有的 Snort 选项，$x_i$ 为第 i 个选项。

(2)集合 $Y=\{y_1,y_2,\cdots,y_m\}$ 为 D 的子集，包含共性选项，y_i 为第 i 个共性选项。

(3)函数 $p(y)$，用来给出集合 Y 中每个元素的权值，此函数必须能使具有不同共性字段的规则有不同的权重值，定义为

$$p(y)=2^i \tag{1}$$

其中，i 为该规则选项在集合 Y 中的序号。

$$F=\sum_{r_i\in Y} p(r_i) \tag{2}$$

其中，r_i 为在规则 r 中出现的第 i 个 Y 集合中的元素。

2.1.3　因式分解算法

因式分解的算法描述如下：

(1)构建 D 集合，将 Snort 中所有规则选项放入 D 集合。

(2)构建集合 Y，按 2.2.1 节中计算规则选项信息量的方法，从 D 中选出信息量最大的前 5 个选项，按信息量由小到大的次序置于集合 Y 中。

(3)对 Y 中元素赋以权值，由式(1)算出。

(4)将 Y 中元素依权值由大到小依次链在 RTN 下，这些新链入的节点即主 OTN。

(5)由 F 函数计算规则权重值，相同权重值归入同一集合。

(6)将每个规则集合链入权值小于或等于其权重值的权值最大的主 OTN 下，对每个集合内规则执行规则排序算法，形成三维链表。

(7)将剩余权重值为 0 的规则链在最后一个主 OTN 之后。

(8)完成。

2.2　动态调整规则匹配次序

在对 OTN 因式分解基础上，还可在主 OTN 下的次 OTN 上对规则进行排序。次 OTN 中的规则包含的选项均属于 Y 在 D 中的补集，因此，可通过对这些选项包含的信息量进行测度来判断次 OTN 中的规则的重要性。

2.2.1　选项所包含信息量测度

大量实验统计结果表明，对于一个给定的数据包样本和一个给定的规则集，让样本中的每一个数据包和规则集中的所有规则进行一次匹配，某个选项能选择出的规则数越多，说明在检测过程中越重要，据此可以测度选项包含的信息量。本文对实验结果作归一化处理：设通过匹配选项 r_i 能匹配的规则数为 n_i，则选项 r_i 的信息量由下式可得：

$$N_{r_i}=\sum_{i=1}^{m} n_i / mR \tag{3}$$

其中，m 为样本中数据包总数为；R 为规则总数。

2.2.2　规则重要性计算

由每个选项的信息量不同，定义某条规则的重要性为

$$W_{R_i}=\sum_{r_i\in D-Y} N_{r_i} \tag{4}$$

其中，r_i 为规则 R_i 中出现的选项；D 为所有 Snort 选项集合；Y 为共性选项集合；N_{r_i} 为选项 r_i 的信息量。

2.2.3　规则排序算法

规则排序算法的具体描述如下：

(1)求出共性选项集合 Y 在 Snort 规则选项集合 D 中补集 D-Y，r_i 是该补集中第 i 个元素。

(2)用攻击样本中每一个数据包 p_i 去匹配所有的规则，记录能匹配成功的规则数 n_i，如图 2 所示，按列相加，得到每个规则选项 r_i ($1\leqslant i\leqslant t$，t 为 D-Y 中元素个数)所能选择的平均规则数为 m_{r_i}。

	r_1	r_2	r_3	…	r_t
P_1	n_1	–	n_1	…	n_1
P_2	–	n_2	–	…	–
P_3	n_3	–	n_3	…	–
⋮	⋮	⋮	⋮	…	⋮
P_m	–	n_m	–	…	n_m
$m_{r_i}=\sum_{i=1}^{m} n_i / m$	m_{r_1}	m_{r_2}	m_{r_3}	…	m_{r_t}

图 2　数据包匹配记录

(3)对选项 r_i 求得信息量为 N_{r_i}，由式(3)算得。

(4)规则 R_i 的重要性 W_{R_i} 由式(4)计算得出。

(5)对每个主 OTN 下的次 OTN 中规则按重要性值由大到小排序。

3　仿真实验及结果分析

3.1　仿真实现思路

实验采用网络仿真软件 OPNET[4-5]Modeler 构建 IDS，步骤如下：

(1)IDS 运行的网络环境模拟。

(2)IDS 建模。采用 Modeler 的 3 层建模机制，最底层为进程(process)模型，以状态机来描述协议；其次为节点(node)模型，由相应的协议模型构成，反映设备特性；最上层为网络模型。

(3)对 IDS 测试评估。比较加入 Snort 规则链因式分解算法前后的 IDS 执行效率。

3.2 IDS 建模及仿真实现过程

本文模拟的 IDS 是基于规则匹配的误用入侵检测系统。已知攻击特征的规则库采用开源的 Snort2.0 中使用的规则文件，IDS 的检测过程的模拟在 Modeler 的节点模型层上实现。

在节点域构建了 2 个模块：数据采集器和数据分析器。数据采集器利用 ACE 收集网络流量并还原网络场景，数据分析器则检测数据采集器中的网络数据，并将结果记入日志。仿真通过数据采集器和数据分析器的合作完成，其结构如图 3 所示。

数据采集器工作在以太网环境中，用 10 Base 双绞线联向一个集线器。数据采集器由 3 个模块组成：队列模块(mac)，发送模块(hub_tx_0_0)，接收模块(hub_rx_0_0)。发送模块和接收模块之间有 1 条逻辑关联，其间通过外部的链路和内部的标准队列模块通信。队列模块用来管理缓冲器和所有收集到的数据。队列模块中运行 Ethernet_mac_v2 进程。

数据分析器(Traffic_analyzer)是 IDS 中最重要的模块。该模块相当于检测引擎，它构建规则树并将其载入内存。由 3 个强制状态和 5 个非强制状态组成，如图 4 所示，其中，空心为强制状态；实心为非强制状态。

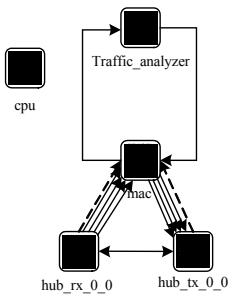

图 3 IDS 节点一般模型

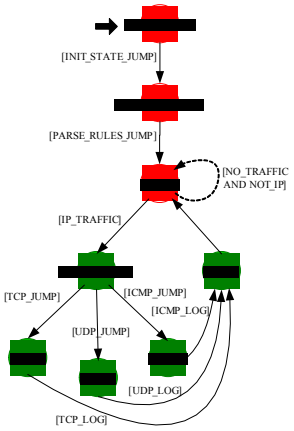

图 4 数据分析器状态

其中，INIT_STATE 初始化一些全局变量；PARSE_RULES 解析规则并载入内存；WAIT 等待数据的到达，每当一个数据到达，进程就从 WAIT 跳转到 IP_ARRIVAL；IP_ARRIVAL 检测收到的数据包，若不与任何规则匹配，就检测该 IP 数据包是否传送其他协议，若是，就跳转至下列 3 个之中的相应状态：TCP, UDP 或 ICMP。

3.3 结果分析

本文仿真过程采用从本地实验室环境收集的网络数据。每 10 000 个包分为一组，利用 ACE 模拟 IDS 的工作环境。IDS 检测过程通过数据分析器完成，在其第 2 个状态 PARSE_RULES 中加入 Snort 规则链因式分解算法和规则排序算法，通过比较加入算法前后 IDS 执行效率，评价本文算法的可行性。检测结果以处理每 10 000 个包所用的时间为评价值，结果如图 5 所示。

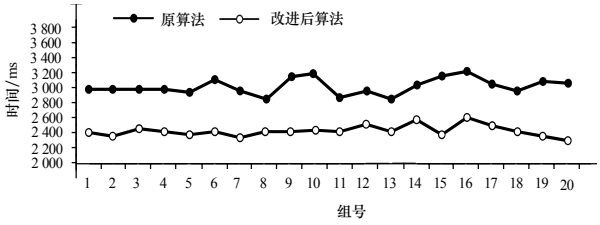

图 5 2 种算法每 10 000 个数据包检测时间对比

实验结果表明，本方法较改进前的规则链表结构每 10 000 个包的检测时间平均缩短约 1/5，具有一定的可行性。

4 结束语

本文将 Snort 规则链表结构中的 OTN 节点进行分流，增加宽度优先搜索，在仿真平台 OPNET 上测试结果证明了该算法的有效性。下一步工作是在真实的 Snort 上使用该算法，以提高 Snort 规则匹配速度。

参考文献

[1] Caswel B, Beale J. Snort2.0 入侵检测[M]. 宋劲松, 译. 北京: 国防工业出版社, 2004.

[2] Abbes T, Bouhoula A, Rusinowitch M. Protocol Analysis in Intrusion Detection Using Decision Tree[C]//Proc. of ITCC'04. Las Vegas, Nevada, USA: IEEE Computer Society, 2004.

[3] Abbes T, Bouhoula A, Rusinowitch M. On the Fly Pattern Matching for Intrusion Detection with SNORT[J]. Annals of Telecommunications, 2004, 59(9/10): 1-23.

[4] 陈 敏. OPNET 网络仿真[M]. 北京: 清华大学出版社, 2004.

[5] 王文博, 张金文. OPNET Modeler 与网络仿真[M]. 北京: 人民邮电出版社, 2003.

编辑 金胡考

(上接第 109 页)

参考文献

[1] Chu J, Labonte K, Bissias G, et al. A Trace-driven Evaluation of Chord[R]. Department of Computer Science University of California, Technical Report: 04-38, 2004.

[2] Gupta A, Liskov B, Rodrigues R. Efficient Routing for Peer-to-Peer Overlays[C]//Proceedings of the 1st Symposium on Networked Systems Design and Implementation. San Francisco, CA, USA: [s. n.], 2004.

[3] Xu Jun, Kumar A, Yu Xingxing. On the Fundamental Tradeoffs Between Routing Table Size and Network Diameter in Peer-to-Peer Networks[J]. IEEE Journal on Selected Areas in Communications, 2004, 22(1): 151-163.

[4] Ganesan P, Manku G S. Optimal Routing in Chord[C]//Proc. of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms. Philadelphia, USA: [s. n.], 2004: 176-185.

编辑 索书志