DPI 技术白皮书

2009 年 3 月

目 录

前言

在当今高速大容量的 Internet 环境中，内容安全是网络安全的重要组成部分。对于网络管理来说，最重要的就是识别和区分网络流量，通过协议识别可以对网络进行流量控制、网络计费、内容过滤、以及流量管理。

传统的协议识别采用的是端口识别，这种识别能达到较高的速率，但是现在大量的应用层协议为了避免识别，逃避防火墙的检查，不使用固定的端口进行通信.这不仅包括众多近年新出现的 P2P 协议，而且包括了越来越多的传统协议，比如 BitTorrent、eMule 等 P2P 协议，其采用动态端口进行通信；Skype、QQ 等协议则共用 80 端口。越来越多诸如此类协议的产生，使得端口识别已无能无力，因此近年来很多的研究工作都致力于开发新的方法来识别应用层协议。

DPI（Deep Packet Inspection，深度包检测）技术是近年来出现的一种协议识别技术，DPI 技术在分析包头的基础上，增加了对应用层的分析，是一种基于应用层的流量检测和控制技术，当 IP 数据包、TCP 或 UDP 数据流经过基于 DPI 技术的网络设备时，DPI 引擎通过深入读取 IP 包载荷的内容来对 OSI 7 层协议中的应用层信息进行重组，从识别出 IP 包的应用层协议。

DPI 技术

传统的 IP 包流量识别和 QoS 控制技术，仅对 IP 包头中的“5Tuples”,即“五元组”信息进行分析，来确定当前流量的基本信息，传统 IP 路由器也正是通过这一系列信息来实现一定程度的流量识别和 QoS 保障的，但其仅仅分析 IP 包的四层以下的内容，包括源地址、目的地址、源端口、目的端口以及协议类型，随着网上应用类型的不断丰富，仅通过第四层端口信息已经不能真正判断流量中的应用类型，更不能应对基于开放端口、随机端口甚至采用加密方式进行传输的应用类型。

DPI 技术技术在分析包头的基础上，增加了对应用层的分析，是一种基于应用层的流量检测和控制技术，当 IP 数据包、TCP 或 UDP 数据流经过基于 DPI 技术的带宽管理系统时，该系统通过深入读取 IP 包载荷的内容来对 OSI7 层协议中的应用层信息进行重组，从而得到整个应用程序的内容，然后按照系统定义的管理策略对流量进行整形操作。

不同的应用通常会采用不同的协议，而各种协议都有其特殊的指纹，这些指纹可能是特定的端口、特定的字符串或者特定的 Bit 序列。基于特征字的识别技术，正是通过识别数据报文中的指纹信息来确定业务所承载的应用。根据具体检测方式的不同，基于特征字的识别技术又可细分为固定特征位置匹配、变动特征位置匹配和状态特征字匹配三种分支技术。通过对指纹信息的升级，基于特征字的识别技术可以方便的扩展到对新协议的检测。

针对不同的识别技术，DPI 可以分为以下两大类：

- ✧ 使用特征字与掩码相结合的协议识别
- ✧ 使用正则表达式库的协议识别

使用特征字与掩码相结合的协议识别

使用特征字符串进行协议识别，先统计协议实际交互过程中出现频率高的字符作为匹配串，DPI 引擎在线检查全报文以匹配多个串，往往适用于少量协议，效率一般，但其正确性有待提高，并且，对于一些变长填充的协议这种方式会显得无能为力。

使用特征字与掩码相结合的字符串匹配方式实现比较简单，往往可以采用硬件的方式来实现，例如某 DPI 厂商就采用交换机的 ACL 芯片来实现基于特征字与掩码相结合的 DPI 识别引擎。

使用正则表达式库的协议识别

最近业界越来越趋向采用正则表达式来进行匹配，实践表明，相对于传统的五元组识别和字符串识别，使用正则表达式对应用层协议进行识别的正确性有很大提高。

正则表达式(Regular Expression)描述了一种字符串匹配的模式，可以用来检查一个串是否含有某种子串、将匹配的子串做替换或者从某个串中取出符合某个条件的子串等。

一些 DPI 厂商采用只检测包头 16 字节或固定长度的特征值来实现 DPI，但是我们认为这种模式不够灵活，特别是一些协议的特征值在包的尾部，或者特征值之间间杂着动态长度的随机填充字节，这些协议，用固定的 DPI 检测就无法识别。

经过对主流协议的研究，我们认为采用正则表达式（regular expression）的方式进行协议特征值的匹配是效果最好的，因为基于正则表达式的 DPI 识别引擎从原理上来说可以识别绝大部分协议。但是由于正则表达式的复杂性，常规的正则表达式引擎相当消耗系统资源，效率比较低，故而直接采用通用的正则表达式算法会严重的影响设备的性能。因此对于

采用基于正则表达式的 DPI 引擎的厂商来说，如何解决 DPI 引擎的性能问题，是一件非常重要的工作。

经过我们研发人员的努力，我们开发了一套高性能的正则表达式算法，该算法采用以空间换时间的方式大大的提高了正则表达式匹配的性能，根据我们设计的算法的原理与我们实际测试的数据，该算法实现了性能与正则表达式的长度和数目无关，这也就代表设备的性能与设备所加载的协议特征码（实际上是一系列的正则表达式）的数量无关。

我们在此正则表达式算法的基础上实现了我们的 DPI 算法，从性能上来说基本上与固定长度的 DPI 算法相差无几，但是却大大提高了其灵活性。

DPI 引擎

我们采用了基于正则表达式的 DPI 引擎，在最初的版本中，DPI 引擎中的正则表达式算法采用的是 NFA 算法，在最近的版本中，为了提高整个 DPI 引擎的性能，我们采用 DFA 算法代替了 NFA 算法。

NFA 与 DFA 的区别

NFA（Non-deterministic finite automaton，不确定有穷自动机），是基于表达式的（Regex-Directed）；而 DFA（Deterministic finite automaton，确定的有穷自动机）是基于文本的（Text-Directed）。

举例来说，对于正则表达式 to(nite|knight|night)，NFA 在匹配最开始两个字符（to）之后，剩下的三个组件（component）是 nite, knight 和 night，于是正则算法会依次尝试这三个选择分支（每次尝试一个）；而 DFA 在匹配最开始两个字符之后，会将剩下的三个选择拆分作字符，并行尝试，也就是说，匹配 to 之后，先匹配 k 或者 n ，如果 k 不能匹配，则放弃 knigth 所在的分支，再匹配 i ，再匹配 t 或 g ……这样继续下去，直到匹配结束。

DFA 引擎在任意时刻必定处于某个确定的状态，而 NFA 引擎可能处于一组状态之中的任何一个，所以，NFA 引擎必须记录所有的可能路径（trace multiple possible routes through the NFA），NFA 之所以能够提供 Backtrack 的功能，原因就在这里。从理论上说，如果我们不需要 Backtrack，或者仅仅需要很小级别的 Backtrack，完全可以从 NFA 构造出等价的

DFA，再进行匹配，这样能大大提高速度——代价是，DFA 需要更多的空间。

NFA 因为不确定，所以限制比 DFA 要少，构造起来也比较方便一点，但匹配速度较慢. 而 DFA 依次匹配并记录匹配过程中每个字符的位置，因此每个字符最多被匹配一次，其匹配速度比 NFA 要高许多。虽然要把正则表达式先转化为 NFA，再把 NFA 转换为 DFA，其转换时间相对要长，但是只需在匹配报文前对正则表达式编译一次，因此匹配报文的总体时间要比 NFA 的匹配引擎快很多。

正则表达式特征库至 DFA 的转换

我们采用扩展的正则表达式对应用层协议的特征进行描述，这个协议特征描述的集合被称之为特征库。

特征库到 DFA 的转换过程需要通过以下步骤：

正则表达式->NFA

给出一个正则串的输入，得到一个 NFA 的输出。被广泛采用的是 Thompson Algorithm，也就是所谓的子集算法。该算法的实现和算术表达式的求值非常的类似，需要一个符号栈存放操作符，一个自动机栈存放生成的自动机。算法结束后，可以从自动机栈中得到一个最终的结果。

我们给出右线性文法：

S → 0S | 1S | 1A | 0B

A → 1C | 1

B → 0C | 0

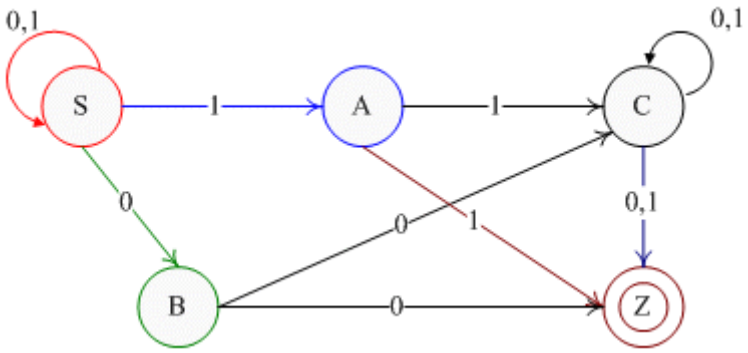
C → 0C | 1C | 0 | 1

右线性文法特点：终结符＋非终结符， 通过右线性文法构造 NFA：

1） 每个非终结符在图中对应一个结点，文法开始符号表示图中的初态结点，添加一终态结点 Z 。

2） 对形如 A → cB 的产生式，画一 A 到 C 的弧，标记为 c 。

3） 对形如 A → c 的产生式，画一 A 到 Z 的弧，标记为 c 。



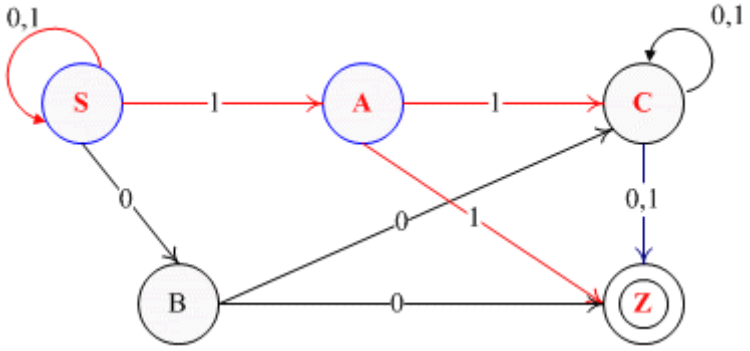
NFA->DFA

一个 NFA 在读入符号串之后，并不确切地知道自动机的下一个状态是什么。但可以肯定的是，下一个状态一定处于某个状态集中。不妨该状态集记做 {q1,q2,…qk} 。而一个等价的 DFA 读入同样的符号串一定处于某个确定的状态上。

这样，都是读入同样的 w， DFA 到达某一个状态，而 NFA 到达某一个状态集。由 w 的任意性，可将 NFA 的所有的状态集和 DFA 的状态一一对应起来。这种对应的前提就是能识别同样的输入串。即 L(M1)=L(M2) 。

所以可以看出，我们要做的就是将 NFA 状态集归并为 DFA 中的状态。其实质是将结点的跳转转化为结点集之间的跳转。由初始结点 S 出发，找出其闭包构成的集合，在本例中， 由于不含 ε 路径，S 的闭包集合就是 {S}。然后由{S}出发，找出通过不同的路径所能到达的集合。

为清晰起见，将选择图中一个集合跳转，重新标上颜色，和下表对应。



集合编号	集合	0	1
S0	{S}	{S,B}	{S,A}
S1	{S,B}	{S,B,C,Z}	{S,A}

S2	{S,A}	{S,B}	{S,A,C,Z}
S3	{S,B,C,Z}	{S,B,C,Z}	{S,A,C,Z}
S4	{S,A,C,Z}	{S,B,C,Z}	{S,A,C,Z}

最小化 DFA

有穷自动机分为确定的有穷自动机 DFA 和不确定的有穷自动机 NFA 两种。

定义 1DFA ：一个确定的有穷自动机，M 是一个五元组，M=(K,Σ, f , S , Z)，其中:K 是一个有穷集,其元素称为状态；Σ 是一个有穷字母表,其元素称为输入符号；S∈K,称为初态;Zì K，是终态集；f 是转换函数，是 K×Σ→K 上的映射，f(ki,a)=kj,(ki∈K,kj∈K)表示状态 ki，输入符为 a 时，转换为状态 kj。

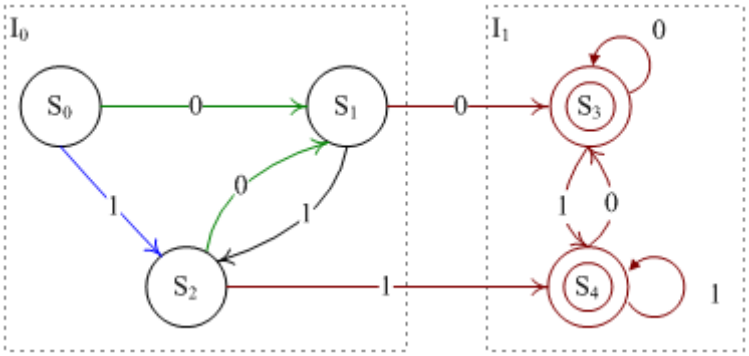
定义 2 无用状态：从自动机的开始状态出发,任何输入串也不能到达的那个状态;或者从这个状态没有通路到达终态的状态。

定义 3 等价状态：如果说两个状态 s 和 t 是等价的，应满足如下条件:(a) 一致性条件：s 和 t 必须同时为终态或为非终态;(b) 蔓延性条件：对于所有输入符号，状态 s 和 t 必须转换到等价的状态里。

一个 DFAM 可以通过消除无用状态和合并等价状态而转化为一个最小化的与之等价的 DFAM'。该过程称为 DFA 的最小化。最小化的思想是在不改变 DFA 识别的语言的前提下，合并相应的结点，使合并后的 DFA 与合并前的 DFA 等效，而结点数目减少。

但在实现上，采用是从合到分的方法。先从最精简的结构出发，即，整个系统就两个集合，所有的终态结点并为一个结点，所有的非综态结点合并为一个结点 。

然后看这种划分是否可行，如不可行，再分解出更多的集合。而合并可行的依据就是集合中的每个元素通过同一条弧到达同一个目标集合。



如上图，首先将所有的非综态结点合并为 I0 ，。将综态结点合并为 I1 。分别考察两个集合。对于 I0 ，通过 0 弧，有如下跳转： S0 → S1 ， S2 → S1 ， S1 → S3 。而 S1 ∈ I0 ， S3 ∈ I1 ，据此， S1 和 S0 、 S2 要分开位于不同集合。

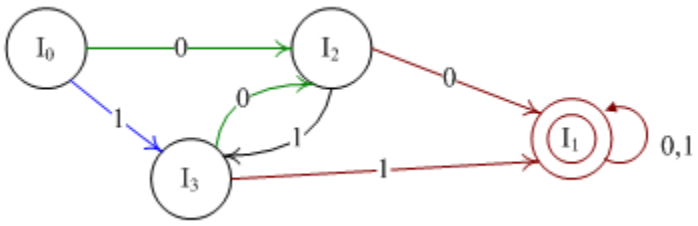
现假定 I0={ S0 ,S2} ， I1={ S3 ,S4} ， I2 ＝ { S1 } 。考察 I0 ，同样是通过 1 弧，则有如下跳转：

S0 → S2 、 S2 → S4 ，而 S2 ∈I 0 ,S4∈I 1 。故， S2 与 S4 分开。可以看出，只要发现一条弧不满足上述条件，就可以将其分离。并非要考察完每条弧。

再来考察 I1 ，通过 0 弧可得： S3 → S3 ∈ I1 ， S4 → S3 ∈ I1 。通过 1 弧可得： S3 → S4 ∈ I1 ， S3 → S4 ∈ I1 。显然， I1 不可再分。

如此重复，直到每个集合 Ii (I=0,1,...) 均不可再分为止。

最小化后的状态转换图如下：



数据包匹配模式

单包匹配方式

当一个会话的每个报文到达时都进行 DPI 引擎匹配，直到匹配成功或者达到每个会话的最大匹配包数（特征值基本上都出现在一个会话的最初几个包中，后续的包中的特征不明显，甚至会导致协议的误判），则后续的报文不再进行匹配。

匹配成功的会话会被打上协议标签，未匹配成功的会话则被打上 Others 的标签。

单报文匹配方式匹配速度快，而且准确性高，基本不存在误判。

有选择的多包匹配方式

当 DPI 引擎发现某个报文符合一定的特征，则自动进入多包匹配方式，这种模式在针对类 HTTP 协议会很有帮助，DPI 引擎会匹配某个会话的多个报文，进行更准确的协议识别。

这两种数据包匹配技术分别适用于不同类型的协议，相互之间无法替代，只有综合的运用这两大技术，才能有效的灵活的识别网络上的各类应用，从而实现有效的控制。

DFA 匹配方式

在 DFA 下有两种匹配扫描算法：

- ✧ One-Pass Scan 算法（单遍扫描算法）
- ✧ Repeated Scan 算法（多遍扫描算法）

One-Pass 算法只扫描一次，从一个位置开始扫描，如果有匹配，则结束搜索，返回结果，直至扫描结束；Repeated Scan 算法从一个位置开始扫描，如果没有匹配，则从下一个位置重新开始扫描，匹配过程中，需要对输入进行重复搜索。

我们首先采用 One-Pass Scan 算法对有 Leading 标记的特征值进行匹配，若匹配成功则结束匹配，若匹配不成功，则用 Repeated Scan 对非 Leading 的特征值进行匹配。

综述

通过 DPI 对网络数据的快速识别与分类，不仅为用户提高服务质量(QoS)、分层服务等提供技术支持，也可以为网络数据上的内容监管(如恶意代码识别、病毒防御)提供技术保障。

深层包检测 DPI 技术易于理解、升级方便、维护简单，是目前运用最有效的流量识别方法。目前，产品支持 600 多种协议和应用的自动识别，涵盖了 P2P、IM（即时通讯）、视频/流媒体、VoIP 协议、网络游戏、炒股软件、企业内部核心应用等应用和协议，基本覆盖了目前主流的网络协议和应用类型，可为用户提供全面的、有效的协议支持。

采用 DPI 技术的优点包括：准确性高、健壮性好、具有分类功能等。准确性高是由于该方法执行精确特征匹配，因此极少存在误判问题。健壮性好是由于可以处理数据包丢失、重组等，因此能适应如今复杂的网络应用。具有分类功能是由于深层数据包检测技术可以依据不同应用的载荷特征来准确分类各网络应用，因此可以为实施流量监管策略提供准确的信息。