面向拟态判决的可编程语义解析方法

张文建[1]，宋克[1]，谭力波[2]，魏帅[1]，董春雷[1]
（1. 信息工程大学，河南 郑州 450002；2. 天津市滨海新区信息技术创新中心，天津 300450）

摘 要：针对拟态判决领域的应用，提出了一种面向拟态判决的可编程语义解析方法。该方法基于匹配查表思想，通过域指针配置方式进行定制协议解析，解决了针对不同协议的可编程解析问题；采用流水控制的方式保证了协议解析过程无拥塞，提高了协议解析的性能；通过引入哈希运算，降低了子分组基于语义的重排序设计复杂度。性能分析结果表明，所提方法在协议解析方面具有高灵活性、高处理能力及低资源利用率等特点。

关键词：拟态判决；可编程；匹配查表；语义解析
中图分类号：TP309
文献标识码：A
doi: 10.11959/j.issn.1000−436x.2020068

Programmable semantic parsing approach for mimic arbitration

ZHANG Wenjian[1], SONG Ke[1], TAN Libo[2], WEI Shuai[1], DONG Chunlei[1]
1. Information Engineering University, Zhengzhou 450002, China
2. Information Technology Innovation Center of Tianjin Binhai New Area, Tianjin 300450, China

Abstract: Aiming at the application of mimic arbitration, a programmable semantic parsing approach for mimic arbitration was proposed. Based on the idea of matching lookup table, this method could achieve custom protocol parsing through domain pointer configuration, and solve the problem of programmable protocol parsing for different protocols. By adopting pipeline control method, the congestion free in the procedure of protocol parsing was guaranteed and the performance of protocol parsing was improved. By introducing Hash operation, the complexity of semantic reordering design of sub-packages was simplified. The performance analysis results show that this approach has the characteristics of high flexibility protocol parsing, high processing capacity and low resource utilization.

Key words: mimic arbitration, programmability, matching lookup table, semantic parsing

1 引言

随着网络安全技术的发展，拟态防御技术[1]逐渐成熟，成为一种新型网络安全防御手段。拟态防御要求系统具备动态异构冗余特性，即动态地调度异构执行体，通过拟态判决异构执行的输出来判断系统的安全状态[2]。面对表达相同语义的协议数据，拟态判决需要解析协议数据的语义信息。如图 1 所示，协议分组由分组头和负载组成，其中分组头包含多层协议，考虑到协议的实用性，协议制定者总会在分组的特定部位定义自定义字段、保留字段及可选字段，图中，TLV 表示 Tag(type)-length-value。此外，不同的协议栈也极有可能将基本的元数据根据自身算法乱序放置。当前，很多应用于网络安全领域的拟态防御技术[3-4]都采用了异构协议栈、异构操作系统、异构处理器等网络组件来构造拟态系统，并引入拟态判决来裁决[5]异构执行体的输出数据。针对由协议栈、操作系统及处理器的异构化而

收稿日期：2019–12–16；**修回日期**：2020–03–06
基金项目：国家核高基重大专项基金资助项目（No.2017ZX01030301）；国家自然科学基金资助项目（No.61572520）；上海市经信委信息化发展（大数据发展）专项基金资助项目（No. 201701046）
Foundation Items: National Core Zlectronic Devices, High-End Generic Chips and Basic Software Major Project(No.2017ZX01030301), The National Natural Science Foundation of China (No.61572520), Major Project for Committee on Economy and Informatization of Shanghai (No.201701046)

引起的协议分组信息多态化冗余化等特点，拟态判决存在着误判风险。在面向该领域的研究中，部分文献[6]从理论方法上阐述拟态判决的方法，但缺少去除多态化冗余信息的方案。

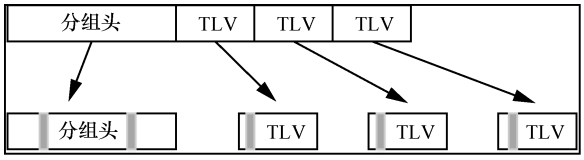


图 1　协议分组架构

目前，面向拟态判决的协议解析实现方式主要分为以下几种。1) 软件实现，即通过处理器进行解析，灵活度高，但这种方式需要通过处理器进行计算解析，效率较低。2) 硬件逻辑实现，大多数拟态判决的协议解析通过硬件逻辑实现[7]，但是硬件逻辑实现的专用包解析的灵活性不高，无法支持扩展协议。3) 利用可编程的方案实现拟态判决的协议解析，目前，研究中尚未出现面向拟态判决的协议解析方案，但交换机数据转发平面的研究为面向拟态判决的协议解析提供了解决思路。

随着网络体系结构的不断发展，网络设备更加追求可扩展性、可编程性及转发性能。SDN（software defined networking）[8]实现了网络设备控制管理平面和数据转发平面的分离，从而为网络的可扩展、可编程提供了很好的思路。为了支持更多新生协议的需求，当前可编程协议解析的研究进行得如火如荼。文献[9]在可编程架构中引入 TCAM（ternary corrent addressable memory），并使用 RAM（random access memory）缓存匹配域的偏移数据。华为提出了 POF（protocol-oblivious forwarding）模型[10]，通过比拟 PC 机结构和 SDN 架构，定义了通用的动作集，将动作执行可编程化，实现了可定制的解析动作。文献[11]实现了比特级粒度的解析，并设计了基于元操作的查找操作，使硬件可以处理任意的协议分组数据。文献[12]创新性地提出了解析和动作执行联动的硬件结构，减少了架构的复杂度以及执行时延，并提高了硬件资源利用率。Wang 等[13]在 NetFPGA 上实现了 P4 的可编程包解析。

综上所述，面向数据转发的可编程设计方案是针对转发设计的，更注重数据转发行为及转发策略的可配置化，解析的粒度更加细致。因此，从拟态判决分组解析的实际需求出发，借助可编程协议解析思想，本文拟解决拟态判决领域多态化冗余化的协议信息提取，将解决语义相同、语法不同的协议数据提取问题，即解决协议语义的顺序混乱、掩码不一致字段及截断冗余的可选字段等问题。基于以上分析，本文提出了面向拟态判决的可编程语义解析方法，基于 NetFPGA[14]平台完成了所提方法的仿真和验证。结果表明，本文所提结构和方法满足面向拟态判决的可编程语义解析的需求。

2　设计目标

在进行拟态判决时，参与拟态判决的协议分组保持语义的严格一致性，否则，将影响拟态防御效果。但受制于协议的灵活性及各个协议栈封装协议数据的算法不同，来自不同协议栈相同功能模块的协议数据可能存在数据不一致的情况，具体分为以下几种情况。

1) 协议数据乱序。受制于协议栈及处理器的不同，协议栈下发相同语义的协议数据时可能存在同一数据帧中数据的乱序。比如下发路由表信息时，在不同协议栈的路由分发算法中，承载路由信息的 OSPF（open shortest path first）协议的 LSU（link status update）消息的路由条目数据顺序可能存在不一致的情况。如此一来，在进行拟态判决时，必须先解析协议数据，调整路由条目数据后再进行判决。

2) 保留字段或自定义字段。为实现协议的可扩展性，大多数协议都规定了保留字段或者自定义字段，不同的协议栈在使用保留字段或者自定义字段时，如果使用方式不同，或者部分不使用，则在对此类协议数据进行拟态判决时，必须将这些字段进行掩码处理，才能进行判决。

3) 可选字段。出于对协议可扩展性方面的考虑，部分协议分组头中存在可选字段，当不同的协议对可选字段的使用方式不同时，此类协议数据必须剔除可选字段才能进行判决。

针对以上情况，本文的设计目标是根据拟态判决的需求，通过解决顺序混乱问题，将冗余信息删除，并将不一致字段进行归一化处理，即对协议数据进行分组切片、掩码以及截断操作，来提取协议数据的语义。分组切片操作对协议数据分组进行分解，得到元数据；掩码是对保留字段或者自定义字段的操作，消除不一致性；截断操作是去除可选字段、删除冗余信息。本文从协议解析高灵活性、高处理性能和低资源使用率的角度出发，设计了一种可实现精确提取协议数据语义信息的面向拟态判决的可编程语义解析（MAPSP, programmable semantic parsing for

mimic arbitration）硬件结构及相关算法。

3　MAPSP 的硬件结构

基于设计目标，本文提出了如图 2 所示的 MAPSP 硬件结构，该结构主要包括解析器和语义提取器。当存在多级解析时，解析器按照级数增加，其主要目标是通过提取分组的协议数据、保留字段或自定义字段、可选字段的位域，进行分组切片、掩码和截断操作，从而达到提取协议数据的语义目的。

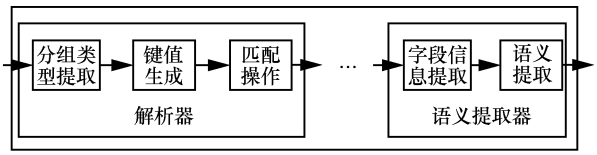


图 2　MAPSP 硬件结构

3.1　解析器

在 MAPSP 结构中，解析器根据上一级匹配操作后产生的信息提取分组类型及匹配信息等数据，完成解析树的配置，通过逻辑运算生成匹配键值，以及匹配可配置的 TCAM 及 RAM 获取与语义相关的数据信息，实现多协议分组的弹性解析。解析器的硬件结构如图 3 所示。

在分组解析的开始阶段，分组类型提取模块首先进行分组的预解析，提取分组首段偏移及协议类型信息并将相应的信息存储到中间信息寄存器中。若非分组解析的开始阶段，则中间信息寄存器的数据由上一级解析器通过解析得到的分组相关信息组成。具体的数据信息包括当前解析节点类型偏移指针、分组类型长度偏移指针、比较指示信号、夹层指示信号、树叶节点指示信号等。其中，解析节点类型偏移指针和分组类型长度偏移指针决定了匹配域从分组提取的类型信息；比较指示信号以及夹层指示信号表示是否需要进行键值预处理；树叶节点指示信号用于指示该层解析为树叶节点，是最后一层解析。

匹配域提取模块提取中间信息寄存器数据用于键值预处理和整合。键值预处理主要包括比较操作及夹层处理操作，其中比较操作是针对 VLAN（virtual local area network）分组等需要时操作才能决定协议类型的分组。夹层处理则是指针对 MPLS（multiprotocol label switching）等多层标签协议的处理，即如果存在多层标签，直接在夹层处理模块进行分组头偏移的累加操作，并将相应信息反馈给最终信息寄存器。键值整合模块是指将协议类型、比较结果信息、夹层处理信息等进行键值映射，从而达到准确匹配的目的。TCAM 和 RAM 模块是可编程的核心模块，即用户可以通过配置 TCAM 和 RAM 模块达到提取用户定制的协议数据的目的。TCAM 存储协议类型以及比较结果新词和夹层处理信息用以匹配当前协议树的数据。RAM 分为 2 种：一种为非树叶节点协议数据 RAM，即包含下一级解析信息的数据，该数据将更新到中间信息寄存器中；另一种为树叶节点协议数据 RAM，即包含最终分组负载的分组切片指针，该数据将更新到最终信息寄存器中，以备语义提取器提取语义信息使用。

3.2　语义提取器

在该结构中，语义提取器的主要功能是根据解析器解析的信息对整个分组进行语义提取，即根据语义进行分组切片、掩码及截断操作来进行特征提取，其硬件结构如图 4 所示。

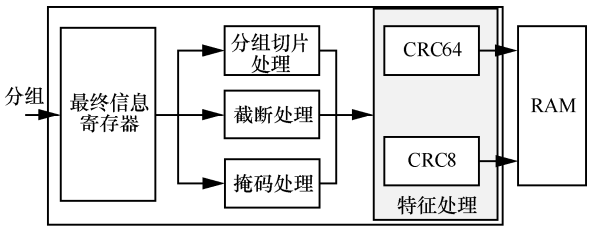


图 4　语义提取器硬件结构

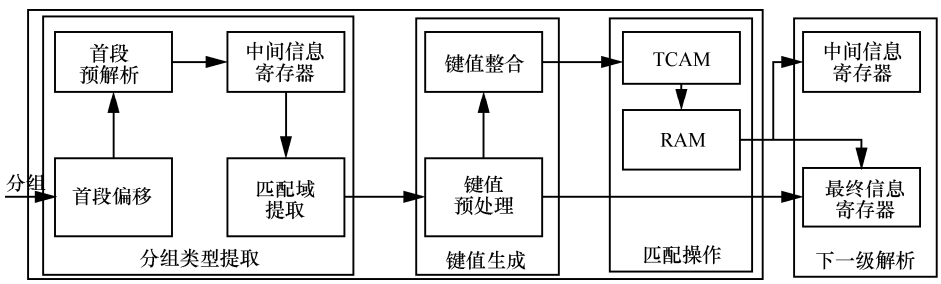


图 3　解析器的硬件结构

最终信息寄存器的信息是根据各级查表操作的结果生成的，包括原始分组分组头偏移指针、掩码信息、截断信息、负载分组头长度信息等。其中原始分组分组头偏移指针是由各级协议解析累加而成，目的是提取整个分组的分组头。当负载数据为典型 TLV 数据时，负载分组头长度信息包含分组头长度偏移指针、分组头长度、分组长度域指针等，当负载数据无乱序数据时，该部分保留，分组切片处理模块根据提供的分组切片处理信息进行分组切片处理，将分离的数据与分组头进行组合生成子分组，从而拆解顺序混乱的协议数据。掩码信息指分组中需要做掩码处理的指针偏移及掩码长度信息。掩码处理根据掩码信息进行掩码操作，将协议规定的保留或者自定义字段进行掩码操作，从而保证了语义一致性。截断信息指分组中需要做截断处理的指针和截断长度信息。截断处理根据截断信息进行截断处理，将协议规定的可选字段进行截断操作，从而实现分组长度的一致性。分组经过分组切片处理、掩码处理、截断处理等操作后，已经形成了语义独立的子分组。特征处理模块的功能是对处理后的语义独立的子分组进行特征码以及特征地址提取，并将提取的特征缓存在 RAM 中，用于拟态判决。其中特征码的作用是指示当前数据分组的特征，即当前子分组的语义信息，通过 CRC64 实现；特征地址的作用是通过 CRC8 实现数据分组的序号；RAM 用来缓存特征。缓存在 RAM 中的数据经过 MAPSP 的处理，已经完全提取出了原始分组的语义信息，剔除了乱序、保留字段或自定义字段、可选字段等与协议无关的信息，为拟态判决提供了精确的语义。

4　MAPSP 算法

MAPSP 解析器的协议可编程语义解析需将用户的协议配置信息根据特定方式借助南向接口存储在协议解析器中。其映射过程主要是将用户对所使用协议的类型信息和匹配域信息以一定的格式存储在缓存（TCAM/RAM）中。根据 MAPSP 的硬件结构，本文提出协议解析算法和语义提取算法，表 1 给出了算法所使用的符号及其含义。

中间寄存器 mid_reg 存储首段字段形成的信息或者是经过匹配 RAM_1 后得出，其数据格式与 RAM_1 一致，包括 sh_len、t_cmp、t_inlaye、inlayer_offset 等元素，最终信息寄存器包括 h_offset、slen_offset、cut_offset、cut_len、mask_offset、mask_len、TLV_flag 等信息。大部分数据和 RAM_2 数据一致，不同之处在于 fin_reg 的 h_offset 由原始值加上当前解析的协议层分组头长度。其中，协议解析算法主要对应解析器的处理，该算法运行的前提是将解析器所支持的协议类型按照约定的方式将对应的键值和数据配置到解析器中。算法伪代码如算法 1 所示。

表 1　　**算法符号及其含义**

符号	含义
node	节点
pkt	原始分组
h_offset	整体分组头偏移
sh_len	当前层协议分组头长度
type_key	与协议类型有关的键值
t_cmp	类型比较指示
t_inlayer	夹层协议指示
inlayer_offset	夹层分组头偏移
m_field	匹配域，用于匹配 TCAM
sub_pkt	子分组分组
pointer	分组切片指针
slen_offset	子分组负载长度偏移
cut_offset	截断偏移
cut_len	截断长度
mask_offset	掩码偏移
mask_len	掩码长度
TLV_flag	TLV 指示
ram_addr	子分组特征标签
feature	子分组特征
N	固定分组切片长度

算法 1　协议解析算法

1) begin

2) key_hit ← 0

3) h_msg←pkt//分组首字段提取

4) while(1)

5)　if(node != node0) mid_reg ← mid_reg //中间寄存器信息缓存

6)　else mid_reg ← h_msg

7)　end if

8)　m_field ← mid_reg //根据 mid_reg 进行匹配域提取

9)　while (t_inlayer == 1) m_field ← m_field(h_offset (inlayer_offset))

10) end while
11) if(t_cmp == 1) m_field ← m_ field(h_ offset(sh_len))
12) else m_field ← m_field(h_offset(0));
13) end if
14) type_key ← m_field //键值生成
15) for j=1 to N
16) if(type_key == TCAM$_j$) //匹配操作
17) mid_reg ← RAM$_{1j}$; fin_reg ← RAM$_{2j}$; key_hit ← 1
18) end if
19) end for
20) if (key_hit != 1)
21) mid_reg ← mid_reg ; fin_reg ← fin_reg
22) end if
23) key_hit ← 0
24) if (the last node) break
25) end if
26) end while
27) end

算法 1 中，首先提取首字段的关键信息，并将关键信息赋值给中间信息寄存器 mid_reg；当解析非首字段信息时，直接使用 mid_reg 的数据（如 2)~8)所示）。其次根据 mid_reg 提供的信息从分组中提取当前层分组类型等信息后，判断是否存在夹层协议，比如，MPLS 标签等，如果存在夹层协议则直接进行标签剥离，从而减少了匹配操作，直至标签剥离到最后一级（如 9)所示）。再次根据中间信息寄存器中的 t_cmp 指示信息判断该层协议是否存在需要进行比较大小的操作，如果是 VLAN 数据，则直接剥离进入下一层解析，如果非 VLAN 数据则正常解析（如 11)~13)所示）。最后根据前期提取的协议类型、比较操作、夹层处理等消息生成键值（如 14)所示）。根据键值进行 TCAM 匹配操作得到 index，并索引 RAM$_1$ 和 RAM$_2$ 得到下一层协议的中间信息和最终信息（如 15)~23)所示）。如果是最后一层解析，则直接进入语义提取阶段，否则继续进行下一层协议的解析（如 24)所示）。

语义提取算法与语义提取器对应，即根据解析器解析得到的最终结果信息，对协议数据进行分组切片、掩码、截断等处理，并通过循环冗余校验运算，最终达到语义提取的目的。算法伪代码如算法 2 所示。

算法 2 语义提取算法

1) begin
2) //sub_pkt$_0$ 为原始分组头
3) sub_pkt$_0$ ← pkt|fin_reg(h_offset) // 根据 fin_reg 的 h_offset 信息提取分组头
4) sub_pkt$_0$ ← mask(sub_pkt$_0$, mask_offset, mask_len) //掩码操作
5) sub_pkt$_0$ ← cut(sub_pkt$_0$, cut_offset, cut_len) //分组切片操作
6) pointer ← h_offset
7) j←1
8) while(1)
9) if(TLV_flag == 1)
10) sub_len ← pkt|(pointer, slen_offset)// 提取子分组长
11) sub_pkt$_j$ ← pkt|(pointer, sub_len)//分组切片
12) sub_pkt$_j$ ← mask(sub_pkt$_{j-1}$, mask_ offset, mask_len); //掩码
13) sub_pkt$_j$ ← cut(sub_pkt$_{j-1}$, cut_ offset, cut_len) //截断
14) ram_addr$_j$ ← CRC8(sub_pkt$_j$)
15) feature$_j$ ← CRC64(sub_pkt$_j$)
16) else
17) sub_len ← N
18) sub_pkt$_j$ ← pkt|(pointer, sub_len)//分组切片
19) sub_pkt$_j$←mask(sub_pkt$_{j-1}$, mask_offset, mask_ len) //掩码
20) sub_pkt$_j$ ← cut(sub_pkt$_{j-1}$, cut_ offset, cut_ len) //截断
21) ram_addr$_j$ ← CRC8(sub_pkt$_j$)
22) feature$_j$ ← CRC64(sub_pkt$_j$)
23) end if
24) if (the last sub_pkt)
25) j←1; pointer ← 0; break
26) else
27) j←j+1; pointer←(pointer, sub_len) //指针更新
28) end if
29) end while
30) end

算法 2 中，首先根据 fin_reg 提供的分组头信息

进行分组头提取，并针对分组头中存在的自定义、保留及选项字段进行截断或者掩码处理（如 3)~5)所示）。然后，对分组负载进行分组切片，判断负载数据是否为 TLV 结构的数据，如果是 TLV 结构的数据则按照 TLV 进行分组切片，否则按照固定长度进行分组切片直至整个数据分组被完全处理，从而形成包括分组头、负载元数据等子分组，并根据掩码参数、截断参数进行掩码和截断操作（如 10)~13)、17)~20)所示）。最后对这些子分组进行 CRC 运算。CRC 运算分为 2 个部分，其中 CRC8 为数据子分组的标签，标识数据的序号，CRC64 为数据子分组的特征值（如 14)~15)、21)~22)所示）。如果所处理子分组为最后的子分组则跳出循环，否则继续语义提取过程（如 24)~28)所示）。

MAPSP 算法通过掩码和截断操作去除了影响语义的冗余和模糊信息，通过分组切片提取了数据分组的基本语义，利用 CRC 运算解决了基本语义紊乱问题，并简化了子分组基于语义的重排序设计复杂度，从而保证了整体语义的精确表述。

5　性能测试与分析

从理论上对解析算法和语义提取算法的存储使用情况进行分析，在 Net-FPGA 平台上部署了 MAPSP 的硬件架构，并对实现的架构进行资源分析，最后针对其处理性能进行了验证和分析。

1) 算法存储分析及硬件资源评估

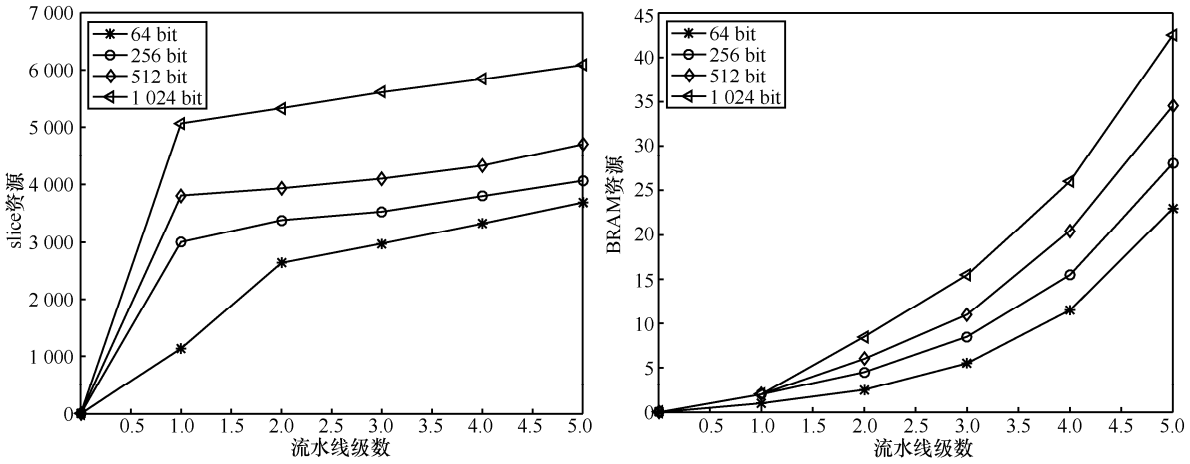
本文算法是针对拟态判决的语义解析模型，其使用的存储分配情况与面向转发协议解析模型有所不同。该设计的主要存储包括 3 个部分：存储类型的 TCAM、存储中间信息的 RAM_1 和存储最终信息的 RAM_2。

根据算法的特点，每一层协议可完成一次 TCAM 查找，并进行匹配。当存在夹层协议或者需要对类型进行比较时，则直接在 TCAM 匹配之前进行下一夹层或下一层协议数据进行类型提取，因此，TCAM 的宽度可按照 $\sum_{1}^{M}\max(\text{t\_len})_i$ 进行设计，其中 $\max(\text{t\_len})_i$ 为第 i 层协议类型长度的最大值，M 为需解析分组的最大层数。TCAM 的条目数与协议节点数相等，假设协议节点数为 $N(T)$，则 TCAM 的条目数为 $N(T)$，TCAM 的最大存储为 $N(T)\sum_{1}^{M}\max(\text{t\_len})_i$。

假设协议树中的非叶子节点和叶子节点的数目分别为 $K(T)$和 $l(T)$，则 $N(T) = K(T)+l(T)$。RAM_1 的条目数为 $K(T)$，RAM_2 的条目数为 $l(T)$。设 RAM_1 和 RAM_2 的单个条目宽度分别为 W_1 和 W_2，则存储的最大空间为 $K(T)W_1 + L(T)W_2$。

本文以以太网、MPLS、802.1Q、IPv4、IPv6、TCP（transmission control protocol）、UDP（user datagram protocol）、OSPF 等协议为例，对 TCAM 和 RAM 资源进行分析。由于面向拟态判决的语义解析需要将数据分组进行深度解析，因此针对 OSPF 的解析需要解析到消息层。OSPF 共 5 类消息，因此总的节点数为 13 个，其中叶子节点 6 个，非叶子节点 7 个。按照上述协议类型分属不同层级，按照上述对 TCAM 和 RAM 的计算式，可得出硬件资源需求：TCAM 需求为 12 条目，且单个条目长度为 10 B；RAM 资源需求为 2 648 bit。

本文算法可以根据需求进行流水线级数设定，流水线级数与 slice 和 BRAM 资源的关系如图 5 所示。



(a) slice 资源使用情况　　(b) BRAM 资源使用情况

图 5　资源开销情况

图 5 表明，随着流水线级数的增加，slice 随之增加，但相对于其他资源每一级流水资源占用较少，所以流水线级数增加造成的 slice 的增加非常有限；另一方面，随着流水线级数的增加，BRAM 的增加幅度变大，这是因为除了随着流水级线性增加之外，还出现了 BRAM 额外占用的情况。

2) 处理性能分析

吞吐量和时延是评价语义解析结构处理性能的重要指标。虽然可以通过扩展总线宽度来增强数据处理能力，但是受限于后续语义提取模块的工作模式。当总线宽度过宽时，后续特征处理速度会受到影响，因此选定总线宽度为 64 bit。在相应的 FPGA 开发平台上进行实验，实现结果最大时钟频率可达到 201.5 MHz，因此，可以实现在吞吐量为 12.896 GB 情况下协议数据的无阻塞处理。时延方面，每一级解析器需要约 3 个时钟周期，语义提取模块可在一个时钟周期内完成。实验中本文采用 4 次匹配查找的 MAPSP 架构，图 6 为 MAPSP 算法的时延仿真结果，其中协议数据从进入解析器开始到完成子分组切分后的时延大概为 13 个时钟周期。

3) 误判率分析

语义提取算法中所提到的使用 CRC8 和 CRC32 进行特征提取的方法必然会引起冲突。当算法存在冲突时，不同语义的数据极有可能映射到相同的特征中去，导致不同的协议数据解析成相同的语义，从而形成判决逃逸。在实验室环境下，通过不同的方式进行数据构造，来验证语义提取算法的判决逃逸率。一方面，利用 CRC8(md5(*i*++))和 CRC32 (md5(*i*++))构造随机输入数据作为协议负载数据，并在协议分组头中进行部分随机化构造，即与语义相关的协议数据使用特定值，无关部分使用随机值进行构造数据集；另一方面，通过注入协议数据测试判决逃逸次数。

在实验过程中，按照 12 GB 的速率进行数据分组解析，发送分组时间为 24 h，实验结果如图 7 所示。其中，当子分组长度为 8 B 和 16 B 时，出现了少量的判决逃逸。与总的发送分组量相比，逃逸概率达到 10^{-10}，这种量级的逃逸概率满足拟态判决的要求。当子分组长度为 4 B/ 32 B/ 64 B/ 128 B 时，24 h 的发送分组数据未能触发判决逃逸现象。考虑到实际网络中数据分组头部和数据负载的常用长度，子分组长度越小，切分的次数越多，越能影响系统的性能；子分组长度越大，占用的系统资源越多。因此，结合实验中得出的逃逸次数，子分组长度选取的最优值为 64 B。

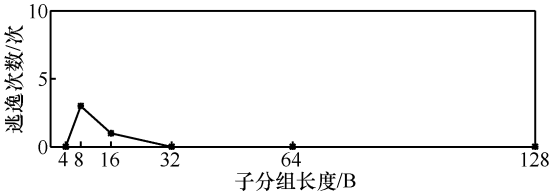


图 7　子分组长度与逃逸次数关系

6　结束语

针对目前拟态判决面临的传输协议数据的多态化冗余化问题，本文借鉴可编程思想和协议数据转发技术，提出了面向拟态判决的可编程语义解析方法，并针对所提解析方法搭建了硬件结构，设计了解析算法及语义提取算法。最后基于 Net-FPGA 平台对所提架构的硬件资源开销和处理性能进行了验证。本文所提方法对拟态判决领域的协议数据可编程化语义解析的发展具有重要意义。

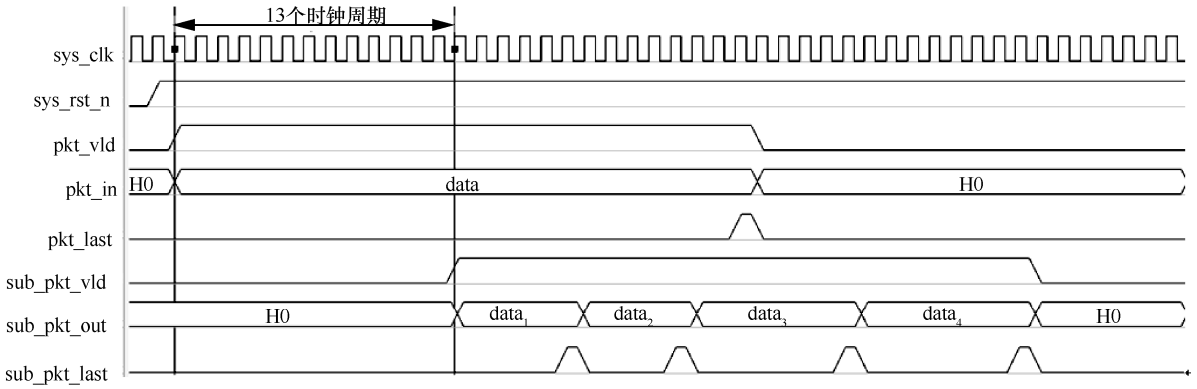


图 6　MAPSP 算法的时延仿真结果

参考文献：

[1] 邬江兴. 网络空间拟态防御研究[J]. 信息安全学报, 2016, 1(4):1-10.
WU J X. Research on cyber mimic defense[J]. Journal of Cyber Security, 2016,1(4):1-10.

[2] 扈红超, 陈福才, 王禎鹏. 拟态防御 DHR 模型若干问题探讨和性能评估[J]. 信息安全学报, 2016, 1(4):40-51.
HU H C, CHEN F C, WANG Z P. Performance evaluations on DHR for cyberspace mimic defense[J]. Journal of Cyber Security, 2016, 1(4): 40-51.

[3] 仝青, 张铮, 张为华, 等. 拟态防御 Web 服务器设计与实现[J]. 软件学报, 2017, 28(4): 883-897.
TONG Q, ZHANG Z, ZHANG W H, et al. Design and implementation of mimic defense Web server[J]. Journal of Software, 2017, 28(4): 883-897.

[4] 魏帅, 于洪, 顾泽宇, 等. 面向工控领域的拟态安全处理机架构[J]. 信息安全学报, 2017,2(1): 54-73.
WEI S, YU H, GU Z Y, et al. Architecture of mimic security processor for industry control system[J]. Journal of Cyber Security, 2017, 2(1): 54-73.

[5] 李卫超, 张铮, 王立群, 等. 基于拟态防御架构的多余度裁决建模与风险分析[J]. 信息安全学报, 2018, 3(5): 68-78.
LI W C, ZHANG Z, WANG L Q, et al. The modeling and risk assessment on redundancy adjudication of mimic defense[J]. Journal of Cyber Security, 2018, 3(5): 68-78.

[6] 赵博. 一种基于输出子集权重分配的拟态判决方法及装置: 201710543007.1[P]. (2017-07-05)[2019-12-16].
ZHAO B. A mimic arbitration method and device based on weight assignment of output subset: 201710543007.1[P]. (2017-07-05) [2019-12-16].

[7] 汪涟. 拟态判决方法、装置及系统: 201811336007.5[P]. (2018-11-09) [2019-12-16].
WANG L. Method device and system of mimic arbitration: 201811336007.5[P]. (2018-11-09) [2019-12-16].

[8] MCKEOWN N. Software-defined networking[J]. INFOCOM Keynote talk, 2009, 17(2): 30-32.

[9] BOSSHART P, GIBB G, KIM H K, et al. Forwarding metamorphosis: fast programmable match-action processing in hardware for SDN[C]// Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM. New York: ACM Press, 2013. 99-110.

[10] SONG H. Protocol-oblivious forwarding: unleash the power of SDN through a future-proof forwarding plane [C]//ACM Sigcomm Workshop on Hot Topics in Software Defined Networking. New York: ACM Press, 2013: 127-132.

[11] 刘中金, 李勇, 苏厉, 等. 弹性协议可定制的网络数据平面结构及其映射算法[J]. 电子与信息学报, 2014, 36(7): 1713-1719.
LIU Z J, LI Y, SU L, et al. Design on the elastic protocol customizable data plane and its mapping algorithm[J]. Journal of Electronics & Information on Technology, 2014, 36(7): 1713-1719.

[12] 孙鹏浩, 兰巨龙, 胡宇翔, 等. 一种解析与执行联动的 SDN 可编程数据平面[J]. 电子学报, 2017, 45(5):1103-1108.
SUN P H, LAN J L, HU Y X, et al. A configurable SDN data-plane based on linkage between parsing and executing[J]. Journal of Electronics, 2017, 45(5): 1103-1108.

[13] WANG H, SOULE R, DANG H T, et al. P4FPGA: a rapid prototyping framework for P4[C]//Proceedings of the Symposium on SDN Research. New York: ACM Press, 2017: 122-135.

[14] ZILBERMAN N, AUDZEVICH Y, KALOGERIDOU G, et al. NetFPGA - rapid prototyping of networking devices in open source[J]. ACM Sigcomm Computer Communication Review, 2015, 45(5): 363-364.

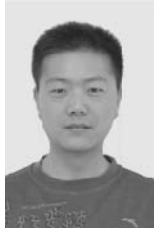
[作者简介]

张文建（1987− ），男，河南商丘人，信息工程大学博士生、助理研究员，主要研究方向为网络空间安全、网络可编程设计、集成电路设计。

宋克（1976− ），男，河南郑州人，信息工程大学博士生、副研究员，主要研究方向为网络空间安全、集成电路设计。

谭力波（1981− ），男，内蒙古赤峰人，天津市滨海新区信息技术创新中心高级工程师，主要研究方向为交换结构设计及计算机结构设计。

魏帅（1984− ），男，河南南阳人，博士，信息工程大学讲师，主要研究方向为嵌入式计算。



董春雷（1987− ），男，河南周口人，信息工程大学助理研究员，主要研究方向为片上系统芯片技术。