ไวยากรณ์นำและนิยามสนับสนุนตัววิเคราะห์โปรโตคอลแบบสากล

Syntax Directed, Definition Supported Universal Protocol Analyzer

สุรศักดิ์ สงวนพงษ์ และ เอกพล โรจน์รัตนาวิชัย
กลุ่มวิจัยเครือข่ายประยุกต์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเกษตรศาสตร์ บางเขน กรุงเทพฯ
E-mail: {nguan, g4165247}@ku.ac.th.

บทคัดย่อ

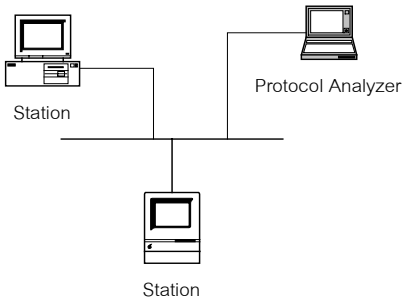
บทความนี้กล่าวถึงแนวคิดการออกแบบนิยามที่ใช้อธิบายลักษณะของโปรโตคอล เพื่อใช้สร้างโปรแกรมวิเคราะห์โปรโตคอล นิยามที่ออกแบบสามารถบรรยายลักษณะของโปรโตคอลทุกๆ ชนิดที่มีอยู่ในปัจจุบันและที่จะเกิดขึ้นใหม่ในอนาคตได้โดยครอบคลุมถึงโปรโตคอลที่อยู่ในเลเยอร์ 2 ถึง 7 ตามข้อกำหนดของโอเอสไอ ประโยชน์การใช้งานคือนำนิยามไปใช้กำหนดโปรโตคอล เพื่อสร้างโปรแกรมวิเคราะห์โปรโตคอลแบบสากล เมื่อต้องการให้โปรแกรมวิเคราะห์โปรโตคอลรู้จักกับโปรโตคอลใหม่ หรือมีการเปลี่ยนแปลงฟิลด์ในโปรโตคอลเดิม ผู้ใช้ไม่จำเป็นต้องแก้ไขโปรแกรมใหม่ หากแต่ปรับเปลี่ยนเฉพาะแฟ้มที่กำหนดลักษณะโปรโตคอลเท่านั้น

Abstract

Implementation of protocol analyzer relies on per protocol modules as existing of protocols. This results a much effort for software development as well as a modification for updated or newly designed protocols. This research proposes a new approach for implementation of universal protocol analyzer using directed syntax and definition of protocol format. With our proposed, the system needs only a configuration files described the protocol characteristics. The parsing engine is, therefore, unique and universal for any protocol. Any changed or updated of protocols do not require source code modification or recompilation of parsing modules. The definition conforms all protocols from layer 2 to 7 of the OSI reference model.

1. บทนำ

ตัววิเคราะห์โปรโตคอล ทำหน้าที่ตรวจจับแพคเก็ตในเครือข่ายและนำมาวิเคราะห์ข้อมูลที่บรรจุอยู่ภายในแพคเก็ตนั้น ตัววิเคราะห์โปรโตคอลอาจอยู่ในรูปของซอฟต์แวร์ที่ใช้งานได้กับคอมพิวเตอร์ทั่วไป หรือเป็นซอฟต์แวร์ที่ถูกออกแบบมาใช้กับคอมพิวเตอร์ที่มีฮาร์ดแวร์เฉพาะ เช่น สนิฟเฟอร์ [1] ซึ่งเป็นชื่อที่นิยมเรียกแทนตัววิเคราะห์โปรโตคอล ถึงแม้ว่าตัววิเคราะห์โปรโตคอลจะมีหน้าที่เฉพาะงานแต่ก็มีลักษณะสมบัติพื้นฐานและการจัดวางระบบเหมือนกับสถานีงานในเครือข่ายดังตัวอย่างในรูปที่ 1



รูปที่ 1 ตัววิเคราะห์โปรโตคอลในเครือข่าย

ในปัจจุบันโปรแกรมวิเคราะห์โปรโตคอลยังถูกจำกัดที่จะออกแบบมาสำหรับวิเคราะห์โปรโตคอลที่จำกัด ไม่สามารถที่จะเพิ่มเติมโปรโตคอลใหม่ๆ [2] เข้าไปได้ หรือสามารถทำได้โดยการเขียนโปรแกรมเพื่อเพิ่มโปรโตคอลใหม่ๆ โดยจำเป็นต้องคอมไพล์โปรแกรมใหม่ทุกครั้ง

งานวิจัยและพัฒนาเกี่ยวกับโปรแกรมวิเคราะห์โปรโตคอลที่ปรับเปลี่ยนได้กับทุกโปรโตคอลที่มีอยู่ในปัจจุบันได้แก่ โครงการฟอลคอน [3] โดยฟอลคอนยังมีขีดจำกัดคือ สามารถกำหนดแฟ้มกำหนดการทำงานในเลเยอร์บนสุดของแต่ละเลเยอร์ และทำได้ในบางโปรโตคอล

จากการค้นคว้าเอกสารและงานวิจัยที่เกี่ยวข้องยังไม่พบเอกสารอ้างอิงใดกล่าวถึงการออกแบบนิยามที่ใช้อธิบายโปรโตคอลแบบสากลและการกำหนดรูปแบบโปรโตคอลจากแฟ้มกำหนดการทำงาน (Configuration file)

บทความนี้จะนำเสนองานแนวคิดที่จะออกแบบนิยามที่ผู้ใช้สามารถบรรยายลักษณะของโปรโตคอลชนิดใดๆ ที่มีอยู่ทั้งในปัจจุบันและอนาคตได้ หัวข้อถัดไปจะกล่าวถึงงานวิจัยที่ทำมาแล้วในส่วนของตัววิเคราะห์โปรโตคอล ในหัวข้อที่ 3 จะกล่าวถึงนิยามที่ใช้อธิบายโปรโตคอลแบบสากล และหัวข้อที่ 4 เป็นบทสรุป

2. ตัววิเคราะห์โปรโตคอลในฟอลคอน

โปรแกรมเริ่มทำงานด้วยการสร้างฐานข้อมูลโปรโตคอลพลวัตจากแฟ้มกำหนดคุณสมบัติของโปรโตคอลพลวัต (Dynamic Protocol Configuration File) ซึ่งผู้ใช้สามารถแก้ไขได้ ฐานข้อมูลนี้มีความสำคัญต่อการวิเคราะห์โปรโตคอล เนื่องจากหากโปรแกรมไม่ทราบรูปแบบของโปรโตคอลชนิดใดก็จะมาตรวจเทียบดูกับฐานข้อมูลนี้ ตัวอย่างรูปแบบของแฟ้มกำหนดคุณสมบัติของโปรโตคอลพลวัต แสดงในรูปที่ 2 ประกอบด้วยชื่อโปรโตคอล (ProtocolName) รายละเอียดสั้นๆ ของโปรโตคอล (ProtocolDescription) ค่าที่ระบุชนิดโปรโตคอล (ProtocolTypeValue) ซึ่งเป็นค่าที่อยู่ในโปรโตคอลในระดับชั้นที่ต่ำกว่าหนึ่งระดับชั้นตามมาตรฐานโอเอสไอ พร้อมกับระบุเป็นตัวเลขฐานสิบ ชื่อของฟิลด์และขนาดของฟิลด์ (ProtocolField) โดยระบุขนาดของฟิลด์เป็นจำนวนไบต์ เมื่อระบุจนครบทุกฟิลด์แล้วจะต้องมีข้อความจบฟิลด์ (End) และปิดท้ายด้วยขอบเขตของโปรโตคอล (ProtocolEnd) ข้อความใดที่ปรากฏอยู่หลังเครื่องหมาย “ # ” หรือบรรทัดใดที่ไม่มีข้อความใดอยู่ โปรแกรมก็จะไม่แปลความหมาย

```
# Put the protocol name in UPPER CASE
ProtocolName=ICMP
# [OPTION] Description of protocol
ProtocolDescription=Message Error
# The value of this protocol in decimal #
specified in lower-layer
ProtocolTypeValue=\
IP=1
# Protocol field name and size of field # in
byte
ProtocolField=\
Type=1
Code=1
Checksum=2
# End of field
End=-1
# End of protocol
ProtocolEnd=ICMP

ProtocolName=DNS
ProtocolDescription=Domain Name System
ProtocolTypeValue=\
UDP=53
TCP=53
ProtocolField=\
Identification=2
Flags=2
NumberOfQuestions=2
NumberOfAnswerRRs=2
NumberOfAuthorityRRs=2
NumberOfAdditionalRRs=2
End=-1
ProtocolEnd=DNS
```

รูปที่ 2 ตัวอย่างแฟ้มกำหนดคุณสมบัติโปรโตคอลพลวัต

แฟ้มกำหนดคุณสมบัติโปรโตคอลพลวัตที่ออกแบบไม่สามารถที่จะอธิบายลักษณะของโปรโตคอลในแบบทั่วๆ ไปได้ ในหัวข้อถัดไป จะเสนอแนวคิดที่จะออกแบบนิยามที่จะอธิบายโปรโตคอลทุกชนิดได้

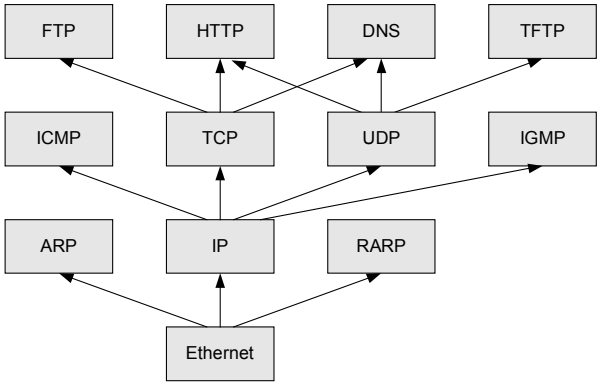
3. นิยามที่ใช้อธิบายโปรโตคอลแบบสากล

นิยามที่จะอธิบายลักษณะของโปรโตคอลแบบสากล จะต้องอธิบายภาพรวมของโครงสร้างของโปรโตคอลต่างๆ ที่เชื่อมต่อกันอยู่ และสามารถอธิบายฟิลด์ต่างๆ ในโปรโตคอลได้ โดยจะต้องอธิบายตำแหน่งเริ่มต้นและความยาวของฟิลด์ได้

3.1. โครงสร้างแบบลำดับชั้นของโปรโตคอล

ลักษณะทางโครงสร้างของโปรโตคอลจะประกอบเป็นโครงสร้างแบบลำดับชั้น ดังรูปที่ 3 ตามโครงสร้างของโอเอสไอ โดยชั้นนอกสุดอยู่ที่เลเยอร์ล่างสุด ในแต่ละเลเยอร์จะประกอบไปด้วยฟิลด์ ที่ใช้เก็บข้อมูลต่างๆ

โครงสร้างของนิยามที่ออกแบบ จะมองโปรโตคอลที่ละชนิดแบบลำดับชั้น คือมองที่เฮดเดอร์ (header) ของโปรโตคอลในลำดับชั้นนั้น และจะส่งส่วนที่เป็นดาต้า (data) ไปให้กับโปรโตคอลในลำดับชั้นถัดไปทำงานต่อ



รูปที่ 3 ตัวอย่างโครงสร้างแบบลำดับชั้นของโปรโตคอล

สิ่งที่นิยามทำในแต่ละลำดับชั้นของโปรโตคอล คือการตีความฟิลด์ต่างๆ ในโปรโตคอลนั้นๆ สิ่งที่วิจัยอยู่คือ การออกแบบนิยามที่จะใช้อธิบายโครงสร้าง และฟิลด์ทั้งหมด ต้นแบบโครงสร้างการนิยามโปรโตคอลในแต่ละลำดับชั้น แสดงดังรูปที่ 4

```
<protocol name> = {
<field name> = <start position>, <length>,
<type>, <description>;
   ……
[if <condition> then goto <next protocol
configuration file>;]
};
```

รูปที่ 4 การนิยามโครงสร้างของโปรโตคอล

<protocol name>	ชื่อของโปรโตคอลที่ตั้งขึ้น
<field name>	ชื่อฟิลด์
<start position>	ตำแหน่งเริ่มต้น ตามด้วยชนิดของข้อมูล เช่น *bit, byte, word*
<length>	ความยาวของฟิลด์ตามด้วยชนิดของข้อมูล เช่น *bit, byte, word*
<type>	ชนิดของข้อมูลในฟิลด์นั้น เช่น จำนวนเต็ม (*int*), ตัวอักษร (*char*), ไบนารี (*bin*)
<description>	คำอธิบายฟิลด์นั้นๆ
<condition>	เงื่อนไข
<next protocol configuration file>	ชื่อแฟ้มกำหนดการทำงานถัดไปที่จะส่งดาต้าไปตีความต่อ

3.2. การส่งต่อให้โปรโตคอลอื่น

เมื่อนิยามตีความเฮดเดอร์เรียบร้อยแล้ว และจะส่งดาต้าให้กับโปรโตคอลในเลเยอร์ถัดไปตีความต่อ เช่น โปรโตคอลไอพี [4] หากฟิลด์ *protocol* มีค่าเท่ากับ 6 จะส่งดาต้าตำแหน่งที่ *hlen*4* ถึง *len* ไปให้แฟ้มกำหนดการทำงานชื่อ *tcp.conf* แสดงตัวอย่างดังรูปที่ 5

```
if <condition> then goto <next protocol
configuration file>;
if  protocol = 6 then tcp.conf, hlen*4, len
```

รูปที่ 5 การส่งต่อให้โปรโตคอลอื่น

3.3. รูปแบบการอธิบายฟิลด์

สิ่งสำคัญที่สุดในการออกแบบนิยามที่จะใช้อธิบายลักษณะของโปรโตคอลได้คือ การอธิบายลักษณะของฟิลด์ได้ เนื่องจากลักษณะฟิลด์มีความแตกต่างกันในแต่ละโปรโตคอล ฉะนั้นก่อนการออกแบบนิยามจะต้องวิเคราะห์ลักษณะของฟิลด์ต่างๆ ก่อน การบรรยายลักษณะโดยทั่วๆ ไปของฟิลด์จะต้องกำหนดตำแหน่งเริ่มต้น และความยาวของแต่ละฟิลด์ โดยจัดแบ่งได้ดังนี้

3.3.1. การระบุตำแหน่งของฟิลด์ ในแต่ละโปรโตคอล แบ่งลักษณะของฟิลด์ออกเป็น 2 ชนิด คือ

- **ฟิลด์ตำแหน่งคงที่** เป็นฟิลด์ที่มีตำแหน่งปรากฏอยู่ตายตัวในโปรโตคอลนั้นๆ ซึ่งเป็นฟิลด์แบบปกติที่พบกันอยู่โดยทั่วๆ ไป อย่างในโปรโตคอลไอพี เช่นฟิลด์ *version, header len., tos,* etc. ตัวอย่างดังรูปที่ 6

```
<field name> = <start position>, <length>,
<type>, <description>;
ver = 0 bit, 4 bit, int, "version";
```

รูปที่ 6 การอธิบายโครงสร้างฟิลด์ตำแหน่งคงที่

- **ฟิลด์ตำแหน่งแปรเปลี่ยน** เป็นฟิลด์ที่อาจจะมีหรือไม่มีปรากฏในโปรโตคอลนั้นก็ได้ วิธีการตรวจสอบจำเป็นจะต้องดูจากฟิลด์ข้างเคียงเป็นตัวบอก เช่น โปรโตคอล *IP* [4] ฟิลด์ *hlen* จะบอกว่ามีฟิลด์ *option* หรือไม่โดยคำนวนจาก *hlen*4-20* หากได้ค่าเป็น 0 แสดงว่าไม่มี *option* ดังรูปที่ 7

```
<field name> = <start position>, <length>,
<type>, <description>;
hlen = 4 bit, 4 bit, int, "header length";
option = 20 byte, hlen *4 -20, bin, "IP
option";
```

รูปที่ 7 การอธิบายโครงสร้างฟิลด์ตำแหน่งแปรเปลี่ยน

3.3.2. การระบุความยาวของฟิลด์ ในแต่ละฟิลด์จะต้องมีความยาว ซึ่งความยาวจะแบ่งเป็น 2 ประเภท

- **ความยาวคงที่** เป็นความยาวฟิลด์แบบแน่นอนตายตัว เป็นลักษณะส่วนมากของฟิลด์ต่างๆ เช่น ในโปรโตคอลไอพีมีฟิลด์ *version, header len,* etc. ต่างก็มีความยาวแน่นอน ตัวอย่างดังรูปที่ 8

```
<field name> = <start position>, <length>,
<type>, <description>;
ver = 0 bit, 4 bit, int, "version";
```

รูปที่ 8 การอธิบายโครงสร้างฟิลด์แบบความยาวคงที่

- **ความยาวแปรเปลี่ยน** เป็นความยาวฟิลด์แบบไม่แน่นอน การหาความยาวของฟิลด์นั้นทำได้ 2 วิธีคือ

  - **มีฟิลด์ข้างเคียงเป็นตัวบอก** เช่น การคำนวณหาความยาวฟิลด์ *option* ใน ไอพีทำได้โดยคำนวณจาก *hlen *4 –20* ดังรูปที่ 9

```
<field name> = <start position>, <length>,
<type>, <description>;
hlen = 4 bit, 4 bit, int, "header length";
option = 20 byte, hlen *4 -20, bin, "IP
option";
```

รูปที่ 9 การอธิบายโครงสร้างฟิลด์ความยาวแปรเปลี่ยนแบบมีฟิลด์อื่นบอกความยาว

  - **มีอักขระพิเศษเป็นตัวขั้น** เช่น ในโปรโตคอลเอชทีทีพี [5] ในส่วนเฮดเดอร์และดาต้าจะถูกขั้นด้วยอักขระแอสกี ENTER (13) 2 ตัวติดกัน ดังรูปที่ 10

```
<field name> = <start position>, <length>,
<type>, <description>;
http_header = x, until (13, 13), char, "HTTP
header";
```

รูปที่ 10 การอธิบายโครงสร้างฟิลด์ความยาวแปรเปลี่ยนแบบมีอักขระพิเศษขั้น

3.3.3. ฟิลด์ที่ให้ความหมายเดี่ยวและหลายความหมาย ในฟิลด์ทั่วๆ ไปจะเป็นแบบความหมายเดี่ยวคือในฟิลด์นั้นจะให้ความหมายเพียงอย่างเดียว แต่ในบางฟิลด์อาจจะมีความหมายย่อยๆ ลงไปอีกเราเรียกว่า หลายความหมาย เช่น โปรโตคอลไอพี ในฟิลด์ `TOS` ประกอบด้วยฟิลด์ย่อย เช่น *precedence, normal delay, normal throughput* การออกแบบในส่วนหลายความหมายนี้ จะออกแบบในลักษณะเป็นเซ็ต ของกลุ่มของค่าในฟิลด์ว่ามีช่วงเท่าไร จะมีคำอธิบายใน `description` ว่าอะไร ดังตัวอย่างในรูปที่ 11

```
tos = x, y, int, case { 0 .. 5 "tos aaa";
                        6 .. 10 "tos bbb";
                      }
```

รูปที่ 11 การอธิบายโครงสร้างฟิลด์หลายความหมาย

4. สรุป

โครงการฟอลคอนเป็นงานวิจัยที่พัฒนาไปแล้ว และสามารถกำหนดลักษณะของโปรโตคอล ในแฟ้มกำหนดการทำงานได้ในระดับหนึ่ง คือสามารถกำหนดแฟ้มกำหนดการทำงานในเลเยอร์บนสุดของแต่ละเลเยอร์ และทำได้เพียงบางโปรโตคอล งานที่กำลังพัฒนาต่อจากโครงการฟอลคอนคือ การออกแบบและสร้างนิยามสำหรับอธิบายโปรโตคอลทุกๆ ชนิดในปัจจุบัน และที่จะเกิดขึ้นใหม่ในอนาคตได้

สิ่งที่ได้กล่าวไปในบทความนี้เป็นแนวคิดที่จะสร้างนิยามสำหรับอธิบายโปรโตคอลแบบสากล โดยส่วนนี้เป็นส่วนหนึ่งของงานวิจัย Nontri Analyzer ซึ่งกำลังวิจัยเพื่อสร้างโปรแกรมวิเคราะห์โปรโตคอลที่สามารถรู้จักโปรโตคอลทุกชนิดที่มีอยู่ทั้งในปัจจุบันและอนาคตได้ โดยเพียงแค่บรรยายลักษณะของโปรโตคอลด้วยนิยามที่ออกแบบขึ้น

หลังจากที่โปรแกรมตีความโปรโตคอลแล้วผลลัพท์ จะส่งต่อให้กับโมดูลอื่นเพื่อวิเคราะห์ต่อ เช่น การเก็บสถิติ การวิเคราะห์การจู่โจมระบบ การเฝ้าดูข้อมูลที่ผ่านเข้ามา

5. เอกสารอ้างอิง

[1] Network associates (1999). "Sniffer Total Network Visibility" [WWW document]. URL http://www.sniffer.com/asp_set/products/tnv/intro.asp

[2] นภัทร สระเอี่ยม, กอบชัย เดชหาญ, ชัชรินทร์ สูนย์ขัน, เอกชัย พรหมมาส. "การตรวจจับแพ็กเก็ตข้อมูลและการประยุคต์ใช้งานบนระบบทีซีพี/ไอพี". การประชุมวิชาการทางวิศวกรรมไฟฟ้า ครั้งที่ 20. พ.ศ. 2540. หน้า 569-573.

[3] อุรุพงษ์ กัลยาสิริ. "การพัฒนาตัววิเคราะห์โปรโตคอลบนเครือข่ายแลน". โครงงานวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเกษตรศาสตร์. พ.ศ. 2541.

[4] Postel, J.B. "Internet Protocol", RFC 791. 1981

[5] Berners-Lee T., Fielding, R. T., and Nielsen, H. F. "Hypertext Transfer Protocol-HTTP/2.0", RFC 1866. 1995