KNOM Review '18-01 Vol.21 No.01
https://doi.org/10.22670/knom.2018.21.1.10

# Protocol Reverse Engineering에서의 정교한 정적 필드를 추출하는 방안에 대한 연구

이 민 섭◆ , 구 영 훈*, 심 규 석* , 김 명 섭°

# A Study on the Extraction of Highly Accurate Static Fields in Protocol Reverse Engineering

Min-Seob Lee◆, Young-Hoon Goo*, Kyu-Seok Shim*, Myung-Sup Kim°

요 약

최근 네트워크 환경은 4차 산업혁명과 더불어 급속도로 성장하고 있으며, 이로 인한 대용량 트래픽이 지속적으로 발생하고 있다. 또한 새로운 응용이나 악성행위가 계속 발생하고 있고 이러한 환경에서 발생하는 프로토콜들의 대부분은 알려지지 않거나 문서화 되어있지 않은 비공개 프로토콜이다. 효율적인 네트워크 관리 및 보안을 위해서 비공개 프로토콜의 구조분석은 불가피하다. 이를 위해서 많은 프로토콜 리버스 엔지니어링 방법론이 제안되었지만, Syntax를 구성하는 필드를 추출하는 표준화된 방법론은 존재하지 않는다. 따라서 본 논문에서는 프로토콜 리버스 엔지니어링에서 정교한 정적필드를 추출하는 방법론을 제안하고 HTTP 프로토콜을 예로 들어 실험을 진행하고 그 성능을 검증한다.

**Key Words :** Protocol Reverse Engineering, Field Format, Protocol Syntax, Private Protocol

ABSTRACT

Recently, the network environment has been growing rapidly along with the fourth industrial revolution, resulting in a steady stream of heavy traffic. In addition, many of the protocols that are occurring in these environments are unknown or documented and private. Structural analysis of private protocols is inevitable for efficient network management and security. While many protocol reverse engineering methodologies have been proposed to achieve this, there is no standardized methodology to extract the fields that make up the Syntax. Therefore, this thesis proposes a methodology to extract sophisticated static fields from protocol reverse engineering and conducts an experiment with, for example, the HTTP protocol and verifies its performance.

◆ First Author : Korea University Department of Computer and Information Science, chenlima2@korea.ac.kr
° Corresponding Author : Korea University Department of Computer and Information Science, tmskim@korea.ac.kr
* Korea University of Department of Computer and Information Science, {gyh0808, ,kusuk007}@korea.ac.kr
이 논문은 2018년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업(NRF-2018R1D1A1B07045742)과 2018년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.2018-0-00539-001,블록체인의 트랜잭션 모니터링 및 분석 기술개발)
논문번호 : KNOM2018-01-007, Received July 15, 2018; Revised August 13, 2018; Accepted August 15, 2018

## Ⅰ. 서 론

4차 산업혁명과 더불어 인터넷 트래픽 사용량이 증가하고 네트워크를 사용하는 응용 및 악성행위가 지속적으로 발생하고 있다. 이러한 환경에서 발생하는 프로토콜의 대부분은 알려지지 않거나 문서화 되어있지 않은 비공개 프로토콜이다. 이러한 비공개 프로토콜의 명확한 사양을 추출하는 것을 목표로 하는 연구인 프로토콜 리버스 엔지니어링은 지금까지 꾸준히 여러 방법론들을 제시하며 연구가 진행되어 왔다. 비공개 프로토콜은 우리가 살아가는 현대사회에서 어렵지 않게 찾아볼 수 있는데 최근 빗썸 해킹이나 우리은행 사이버 공격 같은 사건을 예시로 들 수 있다. 다양한 형태의 사이버 공격이 지속적으로 발생하고 있고 이에 따라 사이버 공격은 꾸준히 인류가 풀어야할 숙제 중에 하나로 대두되고 있다. 이러한 사이버 공격에 대응하기 위해서도 프로토콜 리버스 엔지니어링은 현대 사회에서 필수적으로 진행되어야 할 연구 중에 하나이다. 프로토콜 리버스 엔지니어링은 네트워크 보안 분야뿐만 아니라 네트워크 관리 분야에서도 필수적인 요소로 자리 잡고 있는데 예를 들면 신뢰성 있는 네트워크 사용 현황 파악, 한정적인 네트워크 자원을 효율적으로 사용하고 관리하기 위해 특정 프로토콜에 대한 대역폭 조절 등 여러 분야에 활용되고 있다.

기존의 다양한 연구에서 프로토콜 리버스 엔지니어링에 대한 여러 방법론을 제시하였지만 각각 몇 가지 한계점들이 존재한다. 전통적인 프로토콜 리버스 엔지니어링은 대부분 수동으로 수행되며 시간이 오래 걸리고 오류가 발생하기 쉬운 단점이 존재한다. 이를 해결하기 위해서 자동 프로토콜 리버스 엔지니어링 방법들이 제안되었다. 하지만 제안된 방법론들중 일부는 명확한 필드추출이 이루어지지 않고 일부는 단순하게 프로토콜에서 빈번히 발생하는 값만으로 필드를 추출하고 있기 때문에 프로토콜 리버스 엔지니어링을 진행하는데 있어서 가장 중요한 필드가 제대로 추출되고 있지 않다. 따라서 본 논문에서는 프로토콜 리버스 엔지니어링에서 정적필드를 정교하게 추출하는 방법을 제안하고 HTTP프로토콜을 예로 들어 실험을 진행한다. 본 논문의 구성은 본 장의 서론에 이어서 2장에서 관련 연구 및 문제를 정의하고 3장에서 제안하는 프로토콜 리버스 엔지니어링에서 정교한 정적필드 추출방법론에 대해서 자세히 설명한다. 마지막으로 4장에서는 실험결과에 대한 분석을 하고 마지막으로 5장에서 결론 및 향후 연구를 제시하며 마무리 한다.

## Ⅱ. 관련 연구

### 2.1 프로토콜 리버스 엔지니어링 구성요소

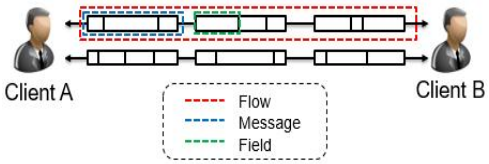


그림 1. 프로토콜 리버스 엔지니어링 구성 요소
Fig. 1. Components of Protocol Reverse Engineering

필드를 정의하기 전에 먼저 프로토콜 리버스 엔지니어링의 입력으로 사용되는 플로우와 메시지를 먼저 정의한다. 플로우란, 5-tuple(Source IP Address, Destination IP Address, Source Port, Destination Port, L4 Protocol)이 동일한 패킷들의 집합이며 메시지의 연속적인 시퀀스를 의미한다. 일반적으로 메시지의 단위는 TCP플로우의 경우 하나의 TCP세그먼트를 하나의 메시지라고 정의하고 UDP플로우인 경우에는 하나의 패킷을 하나의 메시지라고 정의한다. 메시지는 필드의 연속적인 시퀀스로 이루어져 있는데 필드는 프로토콜 리버스 엔지니어링에서 의미를 가지는 가장 작은 단위를 의미한다. 예를 들어 HTTP 프로토콜 같은 경우에 GET, User-Agent, HTTP/1.1 Host 처럼 의미를 가지는 Method, Header Name 같은 것 들을 필드라고 정의한다.

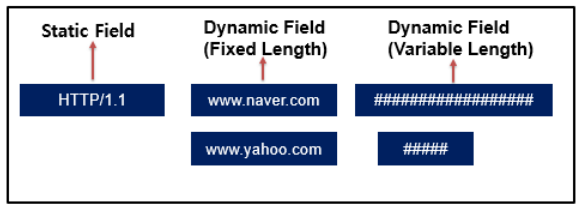


그림 2. 필드의 종류
Fig. 2. Types of Fields

필드는 3가지 유형으로 구성되어 있는데 정적 필드(Static Field), 동적 필드 이면서 길이가 고정적인 필드(Dynamic Field(Fixed Length)), 동적 필드이면서 길이가 가변적인 필드(Dynamic Field(Variable Length))로 나타낼 수 있다. 본 논문에서는 값이 고정적인 정적필드(Static Field)를 정교하게 추출하는 방법에 대해 기술한다.

### 2.2 프로토콜 리버스 엔지니어링

프로토콜 리버스 엔지니어링은 알 수 없거나 문서화 되어있지 않은 네트워크 프로토콜의 명확한 사양을 추출하는 것을 목표로 하는 연구이다. 보통 OSI 7계층에서 알려지지 않은 응용 계층 프로토콜의 사양을 도출하는 것을 목표로 한다.[1] 프로토콜의 사양이라는 것은 프로토콜을 구성하는 3가지 요소를 의미하는데 3가지 요소는 각각 구문, 의미, 타이밍이다. 프로토콜 리버스 엔지니어링 에서는 타겟 프로토콜에 어떤 유형의 메시지들이 존재하고, 이 메시지들은 어떤 형식으로 구성되어 있는지, 어떤 순서로 동작하는지를 알아내는 것을 최종 목표로 한다.

서론에서 언급하였듯이 전통적인 프로토콜 리버스 엔지니어링은 대부분 수동적으로 진행되었다. 수동적인 프로토콜 리버스 엔지니어링은 모든 프로토콜의 요소를 정확하게 복구할 수 있는 장점이 있지만 분석을 수행하는 분석자의 능력에 따라서 결과가 매우 상이하게 나타날 수 있으며 오류가 발생하기 쉽고 무엇보다 시간이 매우 오래 걸리는 치명적인 단점이 존재한다. 시간이 매우 오래 걸리기 때문에 현대 사회에서 급증하는 대용량의 비공개 프로토콜 트래픽을 분석하기에는 타당하지 않다. 이러한 수동적인 프로토콜 리버스 엔지니어링의 단점을 보완하기 위해서 현재까지 프로토콜 리버스 엔지니어링의 자동화를 제안하는 여러 연구들이 진행되어 왔다. 자동 프로토콜 리버스 엔지니어링은 일반적으로 네트워크 트레이스 기반 분석, 실행 트레이스 기반 분석 두 가지 방법으로 분류 된다. 두 가지 방법의 가장 큰 차이점[2]은 입력으로 사용되는 트레이스의 종류이다. 트레이스는 실행 트레이스, 네트워크 트레이스로 나뉜다.

실행 트레이스 기반 분석 방법은 타겟 프로토콜을 사용하는 프로그램 바이너리의 실행을 모니터링 해서 로깅한 실행 트레이스를 입력으로 사용한다. 해당 방법은 타겟 프로토콜을 구현하는 프로그램 바이너리를 사용할 수 있는 환경에서만 분석이 가능하다. 비공개 프로토콜을 구현하는 프로그램 바이너리에 접근하는 것은 현실적으로 어렵기 때문에 비공개 프로토콜의 구조를 분석하는 자동 프로토콜 리버스 엔지니어링에는 적합하지 않다.

다른 한 가지 방법인 네트워크 트레이스 기반 분석 방법은 타겟 프로토콜의 네트워크 트래픽을 캡쳐한 각 네트워크 트레이스를 입력으로 사용하여 분석하는 방법이다. 해당 방법은 타겟 프로토콜을 구현하는 프로그램 바이너리에 접근하지 못하더라도 최 앞단 라우터에서 발생하는 트래픽을 캡쳐해서 클라이언트와 서버간의 송신, 수신 메시지를 모두 분석할 수 있기에 비공개 프로토콜의 구조를 분석하는 방법으로 적합하다. 따라서 본 논문에서는 네트워크 트레이스 기반 분석 방법을 사용한다.

### 2.3 선행 연구

네트워크 트레이스 기반 분석 방법을 사용한 선행연구에는 J. Z. Luo et al.이 제안한 [3]AutoReEngine, Georges Bossert이 제안한 [4]Netzob, A. Trifilo et al이 제안한 [5]Trifilo, Y. Wang et al이 제안한 [6]Veritas, J. Antunes et al.이 제안한 [7]ReverX, M.Shevertalov et al.이 제안한 [8]Pext등이 있다.

| Name | Year | Field Extraction |
|---|---|---|
| Pext | 2007 | X |
| Trifilo | 2009 | X |
| ReverX | 2011 | O |
| Veritas | 2011 | O |
| AutoReEngine | 2013 | O |
| Netzob | 2014 | O |

그림 3. 선행 연구 필드 추출 여부
Fig. 3. Field extraction of preceding studies

그림3을 보면 Pext와 Trifilo이외에 다른 방법론들은 모두 필드를 추출하고 있다. 위 2가지 방법론을 제외하고 필드를 추출하는 총 4가지 방법론 중에서 비교적 최근에 연구되었고 오픈소스로 공개되어있는 Netzob과 최근에 연구되었고 본 논문에서 제안하는 방법론의 토대가 된 AutoReEngine을 본 논문의 저자가 직접 개발한 프로그램으로 실험하고 비교를 통해 제안한 방법론의 성능을 검증한다.

AutoReEngine은 단일 프로토콜에 대한 네트워크 트래픽을 입력으로 받는다. AutoReEngine은 크게 "Data Pre-Processing", "Protocol Keyword Extraction", "Message Format Extraction", "State Machine Inference"의 4가지 단계로 이루어져 있다. AutoReEngine은 순차패턴 마이닝 기법중 하나인 Apriori 알고리즘을 이용하여 타겟 프로토콜에서 빈번하게 발생하는 문자열 후보를 추출한 뒤 문자열 후보들이 메시지와 라인에서 얼마나 고정적인 위치에 발생하는지를 따져서 특정 Threshold 미만의 분산값을 가지는 문자열 후보들만 최종 필드로 선택한다.

Netzob은 Top-Down 방식으로 프로토콜 리버스 엔지니어링을 수행한다. 우선 입력으로 받은 단일 프로토콜에 대한 네트워크 트래픽을 메시지 단위로 재조립 한 다음 Needleman-Wunch알고리즘을 사용하여 Sequence Alignment를 진행하고 하나의 Symbol을 추출한다. Symbol이란 AutoReEngine의 Message Format과 동일한 개념으로써 필드들의 연속적인 시퀀스를 뜻한다. Sequence Alignment가 완료되면 각 Symbol에서 공통적으로 나타나는 문자열들을 정적필드(Data Variable)로 추출하고 각각 다른 값을 가지는 부분을 동적필드(Alternative Variable)로 추출한다. 그 후에 UPGMA알고리즘을 사용하여 각 Symbol간의 유사도를 측정하고 사용자가 입력한 유사도 이상의 유사도를 가지는 Symbol들을 클러스터링 한다.

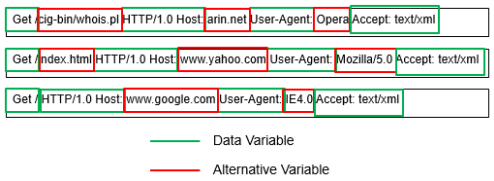


그림 4. Netzob에서의 필드
Fig. 4. Field of Netzob

### 2.4 문제 정의

프로토콜 리버스 엔지니어링은 크게 Syntax, Semantics, FSM 세 가지를 출력한다. Syntax는 타겟 프로토콜에서 메시지의 형식을 의미하고 Semantics는 메시지 유형을 구성하는 필드가 가지는 의미를 나타낸다. FSM은 메시지 유형들이 어떤 순서로 어떻게 동작하는지 표현하는 유한 오토마타이다. 본 논문에서는 프로토콜 리버스 엔지니어링의 출력중 하나인 Syntax에 해당하는 메시지 유형 안의 필드를 추출하는 방법에 초점을 맞추고 있다. 여러 선행 방법론들에서 각자 다른 방법으로 필드를 추출하고 있지만 몇 가지 한계점들이 있다. 첫째, 단순히 알고있는 구분자(space, tab,...)로 필드를 구분한다는 점이다. 이러한 방법은 타겟 프로토콜의 필드가 알려져 있는 구분자로 잘 구분되어 있을 경우에만 필드가 추출되는데 비공개 프로토콜의 경우 필드를 구분하고 있는 정보가 전혀 없기 때문에 이러한 방법은 적합하지 않다. 둘째, 타겟 프로토콜에서 특정 Threshold 이상으로 빈번히 발생하는 문자열들을 필드로 추출한다는 점이다. 이러한 방법은 필드가 가지는 위치정보(offset, depth)를 고려하지 않고 단순히 통계적으로만 접근하기 때문에 타겟 프로토콜에서만 발생하는 정적필드를 추출하지 못하는 한계점이 존재한다. 따라서 본 논문에서는 필드가 가지는 위치정보와 통계정보를 모두 활용하여 타겟 프로토콜에서 빈번히 발생하면서 고정적인 위치에만 발생하는 정교한 정적필드를 추출하는 방법을 제안한다.

## Ⅲ. 본 론

### 3.1 제안하는 방법론의 Overview

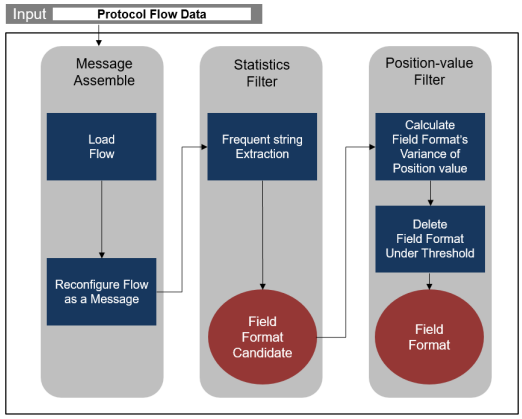


그림 5. 제안하는 방법론의 Overview
Fig. 5. Overview of Suggested Methodology

본 논문에서 제안하는 방법의 전체적인 구성은 그림5과 같다. 본 방법론은 크게 “Message Assemble”, “Statistics Filter”, “Position-Value Filter” 3가지 단계로 이루어져 있다.

Message Assemble 단계는 타겟 프로토콜 플로우 데이터를 사용하기 위해 데이터 전처리를 하는 단계로써 플로우 데이터를 필드들의 연속적인 시퀀스로 이루어져 있는 메시지들의 시퀀스로 재구성 한다.

Statistics Filter 단계 에서는 Apriori 알고리즘[9]을 사용하여 지정한 Support Unit단위에서 빈번하게 발생하는 문자열(필드후보)들을 추출한다. 이때 Apriori 알고리즘에서 가장 중요한 값인 Minimum Support Unit으로써 3가지 Support Unit을 사용한다. Statistics Filter 단계를 거쳐서 추출된 필드후보들은 3가지 Unit에서 모두 빈번하게 발생하는 문자열들이다.

Position-Value Filter 단계에서는 필드후보들의 위치정보를 사용하여 고정적인 위치에 발생하는 필

드후보들만 선정하는 필터링을 수행하고 특정 Threshold를 기준으로 고정적인 위치에 발생하는 필드후보들을 최종 정적필드를 추출하게 된다.

### 3.2 Statistics Filter

Statistics Filter 단계에서는 Apriori 알고리즘에 3가지 Support를 적용하여 모든 Support Unit에서 빈번히 발생하는 문자열들을 추출한다. Apriori 알고리즘은 "특정 시퀀스가 빈번하다면 그 서브시퀀스들도 역시 빈번하다"는 Apriori속성에 기반을 둔 알고리즘이다. Apriori 알고리즘은 각 level별로 접근하여 length(k-1)아이템 부터 length(k) 아이템까지 추출하는 알고리즘인데 k-1아이템과 k아이템의 포함관계를 제거하기 위하여 Apriori Step마다 포함관계 제거를 수행한다. 각 level에서 k-1아이템으로부터 파생된 k아이템이 있을때 k-1아이템이 분석하는 메시지들과 k-1아이템으로부터 파생된 k아이템들이 분석하는 메시지들의 합집합이 같으면 k-1아이템이 다른 k아이템에 존재하지 않다는 것을 알 수 있기에 k-1아이템을 삭제한다.

$$flow_{supp} = \frac{number\ of\ flows\ containing\ item}{total\ number\ of\ flows}$$

$$message_{supp} = \frac{number\ of\ (Req\ or\ Res) messages\ containing\ item}{total\ number\ of\ (Req\ or\ Res) messages}$$

$$site\_set_{supp} = \frac{number\ of\ site\_set\ containing\ item}{total\ number\ of\ site\_set}$$

그림 6. 3가지 Support Unit 정의
Fig. 6. Definition of 3 Support Units

본 방법론에서는 Apriori 알고리즘에 Flow, Message, Site_set Support를 동시에 적용하여 세가지 Support Unit 모두에서 빈번히 발생하는 문자열들을 필드후보로 추출한다. 즉, 3가지 Support를 모두 만족해야만 필드 후보로 추출된다. 각 Support Unit에 대한 정의는 그림6과 같다. Site_set이란 하나의 서버와 해당 서버랑 통신하는 클라이언트 간에 연결을 맺고 있는 모든 플로우들을 원소로 하는 집합을 의미한다. Flow Support는 전체 플로우에서 해당 후보필드가 얼마나 빈번하게 발생하는지에 대한 지표로써 몇몇 플로우에서만 발생하는 Noise문자열들을 필터링 할 수 있다. Message Support는 메시지의 방향(Request, Response)을 구분하여 해당 후보필드가 Request메시지, Response메시지에서 얼마나 빈번하게 발생하는지에 대한 지표로써 본 방법론에서 가장 중요한 Support이다. Support를 적용할 수 있는 가장 작은 단위인 메시지에 Support를 적용하여 Flow, Site_set에서 빈번하게 발생한다 하더라도 Message에서는 빈번하게 발생하지 않는 Noise를 제거할 수 있다. Site Support는 Site_set에서 해당 후보필드가 얼마나 빈번하게 발생하는지에 대한 지표로써 몇몇 플로우에서만 빈번하게 발생하는 Noise문자열들을 필터링 할 수 있다.

Statistics Filter 단계에서는 첫 번째 과정으로 1바이트 모든 Character를 Apriori알고리즘의 입력으로 사용하여 k-길이의 빈번한 문자열(필드후보)집합을 추출한다. 그리고 필드후보들 중에 완벽하게 포함되는 관계에 있는 두 필드후보가 있을때 길이가 짧은 후보필드를 제거하는 포함관계 제거를 수행한다.

### 3.3 Position-Value Filter

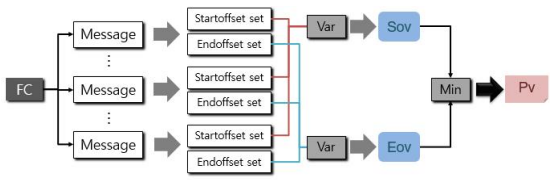


그림 7. Position-Value Filter 상세 구조
Fig. 7. Detailed Structure of Position-Value Filter

Position-Value Filter 단계에서는 Statistics Filter 단계를 거쳐서 추출된 필드후보들의 위치정보를 검사하여 특정 Threshold 미만인, 즉 얼마나 고정적인 위치에서 나타나고 있는지를 기준으로 필터링을 진행한다. 타겟 필드후보(FC)는 그림7과 같이 여러 메시지에서 발생하는데 FC가 발생하는 각각의 메시지마다 Startoffset set과 Endoffset set이 존재한다. Startoffset set이란 메시지 시작부분을 기준으로 FC의 위치값(offset)들의 집합이고 Endoffset set은 메시지 끝부분을 기준으로 한 FC의 위치 값들의 집합이다. 그림7에서 볼 수 있듯이 FC가 나타나는 모든 메시지들의 Startoffset들의 분산값(Sov)을 계산하고 Endoffset(Eov)들의 분산값을 계산한다. Sov와 Eov 중에 최솟값을 Pv라고 정의한다. Pv는 특정 FC가 메시지 내에서 어떻게 분포되어 있는지 나타내는 지표이다. Pv가 작으면 작을수록 해당 FC는 메시지 내에서 고정적인 위치에 나타난다는 것을 의미한다. 따라서 Pv가 특정 Threshold 미만의 값을 가진다면 해당 FC를 최종 정적필드로 선정한다.

## Ⅳ. 실험

| Netzob | AutoReEngine | Suggested Method |
|---|---|---|
| HTTP/1.1\|20\|200\|20\|OK\|0d\|\|0a\|Server:\|20\|nginx\|0d\|\|0a\|Date:\|20\|Tue,\|20\|10\|20\|Apr\|20\|2018\|20\|07:01:11\|20\|GMT\|0d\|\|0a\|Content-Type:\|20\|image/gif\|0d\|\|0a\|Content-Length:\|20\| | e\|0d\|\|0a\|User-Agent: Mozilla/5.0 (Windows NT 10.0\|3b\| W | GET / |
| | e\|0d\|\|0a\|User-Agent: Mozilla/5.0 (Windows NT 10.0\|3b\| Win64\|3b\| x64 | Safari/537.36\|0d\|\|0a\|Accept: |
| | e\|0d\|\|0a\|Accept | HTTP/1.1\|0d\|\|0a\|Host: |
| | e\|0d\|\|0a\|Accept- | \|0d\|\|0a\|User-Agent: Mozilla/5.0 (Windows NT 10.0\|3b\| Win64\|3b\| x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.1 |
| | User-Agent | \|0d\|\|0a\|Referer: http |
| \|20\|GMT\|0d\|\|0a\|Connection:\|20\|keep-alive\|0d\|\|0a\|Keep-Alive:\|20\|timeout=5\|0d\|\|0a\|ETag:\|20\|\|22\|55502c9 | HTTP/1.1 | \|0d\|\|0a\|Host: |
| | HTTP/1.1 20 | \|0d\|\|0a\|Connection: keep-alive\|0d\|\|0a\|U |
| | HTTP/1.1 200 OK\|0d\|\|0a\| | \|0d\|\|0a\|Accept: |
| | HTTP/1.1 \|0d\|\|0a\| | \|0d\|\|0a\|Accept-Language: ko-KR |
| | HTTP/1.1 \|0d\|\|0a\|Host: | \|0d\|\|0a\|Accept-Encoding: gzip, deflate\|0d\|\|0a\|Accept-Language: ko-KR,ko\|3b\|q=0.9,en-US\|3b\|q=0.8,en\|3b\|q=0.7\|0d\|\|0a\| |
| \|22\|\|0d\|\|0a\|Expires:\|20\|Thu,\|20\|10\|20\|May\|20\|2018\|20\|07:01:11\|20\|GMT\|0d\|\|0a\|Cache-Control:\|20\|max-age=2592000\|0d\|\|0a\|Accept-Ranges:\|20\|bytes\|0d\|\|0a\|\|0d\|\|0a\|GIF89a\|0b\|\|00\| | \|0a\|Expires | HTTP/1.1 |
| | Expires | \|0d\|\|0a\|Date: |
| | Accept-Encoding | \|0d\|\|0a\|Content-Type: |
| | GET / | \|0d\|\|0a\|Content-Length: |
| | Content | \|0d\|\|0a\|Connection: |

그림 8. 실험결과
Fig. 8. Result of Experiment

본 장에서는 본 논문에서 제안한 방법론의 타당성을 검증하기 위해서 선행연구의 방법론인 “Netzob”, “AutoReEngine”과의 실험결과를 비교한다. 실험에 사용한 트래픽은 모두 동일한 HTTP 프로토콜 이다. 실험에서 사용한 Flow, Message, Site_set Support는 모두 50%로 설정하였고 Netzob의 경우 유사도를 50%로 설정하였다. AutoReEngine에서 사용하는 Position Variance Threshold값과 본 논문에서 사용하는 PV Threshold값은 동일하게 0.05로 설정하여 실험을 진행하였다. 필드가 잘 추출되었는지 평가하기 위해서 입력된 HTTP 프로토콜 트래픽에 대하여 Method, Version, Status code, Phrase, Header Name들을 정답이라고 정의하고 이를 True Field라고 한다. 그림7은 각 방법론들의 value값을 가지는 필드들을 제외하고 TrueField를 포함하는 필드들만 정리한 결과이다. 현재 프로토콜 리버스 엔지니어링에서의 추출된 필드를 평가하는 방법론은 존재하지 않기에 휴리스틱하게 실험결과를 분석한다. 필드는 프로토콜 리버스 엔지니어링에서 의미를 가지는 가장 작은 단위이기 때문에 각 필드당 1~2개의 True Field를 포함하는 필드가 정교한 필드라고 판단한다.

Netzob같은 경우에는 빨간색으로 표시된 부분이 정답 부분인데 하나의 필드가 너무 많은 True Field를 포함하고 있다. 프로토콜 리버스 엔지니어링에서 필드는 의미를 가장 작은 단위이기 때문에 1~2개의 TrueField를 포함하는 것이 적절한데 Netzob같은 경우에는 하나의 필드에 TrueField를 너무 많이 포함하고 있고 필드의 길이가 너무 길기 때문에 성능이 다른 방법론에 비해 비교적 좋지 않다고 판단된다.

AutoReEngine같은 경우에는 Netzob에 비하면 비교적 준수한 성능을 보이고 있다. Header Name과 Method 들을 잘 추출하고 있으나 너무 많은 필드들이 추출된다는 한계점이 있다. 또한 Apriori 알고리즘을 적용하는 과정에서 필드들간의 포함관계 제거가 존재하지 않기 때문에 특정 필드의 서브 시퀀스인 Noise필드들이 많이 존재한다. 프로토콜 리버스 엔지니어링에서 메시지 포맷을 추출할 때 이러한 Noise필드들이 많이 존재하면 너무 많은 메시지 포맷들이 추출되어 비공개 프로토콜의 구조를 분석하기에 부적합하다.

본 논문에서 제안하는 방법론은 하나의 필드에 1개 혹은 2개의 TrueField로 필드들이 구성되어 있고 HTTP 프로토콜의 Method, Version, Header Name들이 적절히 잘 추출 되었다는 것을 알 수 있다.

## V. 결론 및 향후 연구

본 논문에서는 프로토콜 리버스 엔지니어링에서 필드가 가질 수 있는 통계적인 정보와 위치적인 정보 두 가지를 융합한 방법으로 정교한 정적필드를 추출하는 방법을 제안하였다. 제안한 방법과 선행연구에서 연구된 기존방법론 2가지를 HTTP 프로토콜에 대하여 실험을 진행하고 결과를 분석하면서 제안한 방법론의 타당성을 검증하였다. 향후 연구로는 각 방법론에서 추출된 필드들을 객관적으로 평가할 수 있는 방안에 대한 연구와 입력 트래픽에서 최대한 많은 TrueField를 추출할 수 있도록 하는 최적의 Minimum Support값을 찾는 연구를 진행할 계획이다.

## References


[1] Young-Hoon Goo, Kyu-Seok Shim, Jee-Tae Park, Byeong-Min Chae, Ho-Won Moon, Myung-Sup Kim, "A Method of Protocol Reverse Engineering for Clear Protocol Specification Extraction", KNOM Review, Vol. 20, No. 2, pp. 11-23, Dec.2017

[2] Young-Hoon Goo, Baraka D. Sija, Sung-Ho Lee, Myung-Sup Kim, "Analyzing the Difference Between Network Trace-based and Execution Trace-based Protocol Reverse Engineering in Three Perspectives", Proceedings of Symposium of the Korean Institute of communications and Information Sciences, pp. 82-83, Jeju Island, Korea, June 2017.

[3] Jian-Zhen Luo, Shun-Zheng Yu "Position-based automatic reverse engineering of network protocols", Jounal of Network and Computer Applications, Vol. 36, No. 3, Issue. 3, pp. 1070 - 1077 Feb.2013

[4] Bossert, Georges, Frédéric Guihéry, and Guillaume Hiet, "Towards automated protocol reverse engineering using semantic information.", Proceedings of the 9th ACM symposium on Information, computer and communications security. ACM, pp. 51-62, Kyoto, Japan, June.2014

[5] Trifilò, A., Burschka, S., & Biersack, E. ,"Traffic to protocol reverse engineering", In Computational Intelligence for Security and Defense Applications, 2009. CISDA 2009. IEEE Symposium on IEEE, pp. 1-8, Ottawa, Canada, Dec.2009

[6] Wang, Y., Zhang, Z., Yao, D. D., Qu, B., & Guo, L., "Inferring protocol state machine from network traces: a probabilistic approach", In International Conference on Applied Cryptography and Network Security, pp. 1-18,Nerja, Spain, June.2011

[7] Antunes, Joao, Nuno Neves, and Paulo Verissimo, "Reverse engineering of protocols from network traces.", Reverse Engineering (WCRE), 2011 18th Working Conference on. IEEE, pp. 169-178, Limerick, Ireland, Oct.2011

[8] Shevertalov, Maxim, and Spiros Mancoridis, "A reverse engineering tool for extracting protocols of networked applications", Reverse Engineering (WCRE), 2007 14th Working Conference on. IEEE, pp. 229-238, Vancouver, Canada, Oct.2007

[9] Young-Hoon Goo, Kyu-Seok Shim, Woo-suk Jung, Myung-Sup Kim, "SnorGen: Web-based Automatic Signature Generation System", KNOM Review, Vol. 18, No. 2, pp. 1-11, Dec.2015



**이 민 섭** (Min-Seob Lee)

2018 고려대학교 컴퓨터정보학과 학사

2018 - 현재 고려대학교 컴퓨터정보학과 석사과정

<관심분야> 네트워크 관리 및 보안, 트래픽 모니터링 및 분석

**구 영 훈** (Young-Hoon Goo)

2016 고려대학교 컴퓨터정보학과 학사

2016 - 현재 고려대학교 컴퓨터정보학과 석/박사통합과정

<관심분야> 네트워크 관리 및 보안, 트래픽 모니터링 및 분석

**심 규 석** (Kyu-Seok Shim)

2014 고려대학교, 컴퓨터정보학과 학사과정

2016 고려대학교 컴퓨터정보학과 석사과정

2016 - 현재 고려대학교 컴퓨터정보학과 박사과정

<관심분야> 네트워크 관리 및 보안, 트래픽 모니터링 및 분석

**김 명 섭** (Myung-Sup Kim)

1998 포항공과대학교 전자계산학과 학사

2000 포항공과대학교 전자계산학과 석사

2004 포항공과대학교 전자계산학과 박사

2006 Dept. of ECS, Univ of Toronto Canada

2006 - 현재 고려대학교 컴퓨터정보학과 교수

<관심분야> 네트워크 관리 및 보안, 트래픽 모니터링 및 분석, 멀티미디어 네트워크