https://doi.org/10.22670/knom.2019.22.1.42

상세한 프로토콜 구조를 추론하는 프로토콜 리버스 엔지니어링 방법에 대한 연구

채 병 민◆, 문 호 원*, 구 영 훈**, 심 규 석**, 이 민 섭**, 김 명 섭°

A Study on the Inference of Detailed Protocol Structure in Protocol Reverse Engineering

Byeong-Min Chae◆, Ho-Won Moon*, Young-Hoon Goo**, Kyu-Seok Shim**, Min-Seob Lee**, Myung-Sup Kim°

요 약

최근 네트워크 환경은 고속화, 대용량화 등으로 인터넷 트래픽 발생량이 증가하고 있으며, 모바일 및 IoT 환경, 지속적으로 증가하는 어플리케이션, 악성행위로 인해 비공개 프로토콜 데이터가 늘어나고 있다. 이러한 비공개 프로토콜들의 대다수는 구조가 전혀 알려지지 않고 있다. 효율적인 네트워크 관리 및 보안을 위해 비공개 프로토콜의 구조 분석은 반드시 선행되어야 한다. 이를 위해 많은 프로토콜 리버스 엔지니어링 방법론이 제안되었지만, 적용하기에 각기 다른 단점이 존재한다. 본 논문에서는 CSP(Contiguous Sequential Pattern)와 SP(Sequential Pattern) Algorithm을 계층적으로 결합하여 네트워크 트레이스 분석 기반의 상세한 프로토콜 구조를 추론하는 방법론을 제안한다. 제안된 방법론은 선행 연구인 A^2PRE을 개선하는 방식으로 설계 및 구현을 하였으며 다른 방법론과 성능 비교를 위해 성능지표를 정의하고 HTTP, DNS 프로토콜의 예를 통해 제안하는 방법론의 우수성을 설명한다.

Key Words : Protocol Reverse Engineering, Message Format, CSP Algorithm, SP Algorithm

ABSTRACT

Recently, the amount of internet traffic is increasing due to the increase in speed and capacity of the network environment, and protocol data is increasing due to mobile, IoT, application, and malicious behavior. Most of these private protocols are unknown in structure. For efficient network management and security, analysis of the structure of private protocols must be performed. Many protocol reverse engineering methodologies have been proposed for this purpose, but there are disadvantages to applying them. In this paper, we propose a methodology for inferring a detailed protocol structure based on network trace analysis by hierarchically combining CSP (Contiguous Sequential Pattern) and SP (Sequential Pattern) Algorithm. The proposed methodology is designed and implemented in a way that improves the preceeding study, A^2PRE, We describe performance index for comparing methodologies and demonstrate the superiority of the proposed methodology through the example of HTTP, DNS protocol.

※이 논문은 2016년도 국방과학연구소의 재원을 받아 수행된 연구임(UE161105ED).
◆ First Author : Hanwha Systems, byeongmin.chae@hanwha.com
° Corresponding Author : Korea University of Department of Computer and Information Science, tmskim@korea.ac.kr
* Hanwha Systems, moon1000@hanwha.com
** Korea University of Department of Computer and Information Science, {gyh0808, kusuk007, chenlima2}@korea.ac.kr
논문번호 : KNOM2019-01-10, Received June 24, 2019; Revised July 23, 2019; Accepted August 15, 2019

Ⅰ. 서 론

현재 전 세계 데이터량은 연평균 26%씩 증가하고 있다. 기술 발전 및 인프라의 확장으로 2017-2022 약 3배 이상의 IP 트래픽이 증가하고 IoT 기술의 발전으로 Mobile 트래픽이 연평균 성장률이 46%에 이를 전망이다[1]. 상용서비스가 시작된 5G 환경에서는 모바일, IoT 장비의 수가 증가하므로 응용 및 악성코드 수 또한 증가할 것으로 판단된다. 이런 환경에서 발생하는 프로토콜 중 다수는 특정 업체에서 개발하여 사용하는 독점적인 프로토콜이거나 봇넷의 C&C(Command and Control) 프로토콜 등과 같은 구조를 제한적으로 알 수 있거나, 전혀 알 수 없는 프로토콜이다. 한편, 2016-2018 카스퍼스키랩에서 수집한 IoT 장치용 악성 코드 샘플에 따르면 2018년 상반기 IoT 기기를 공격한 악성코드 변종은 무려 12만 종이 넘었고, 이 수치는 2017년 한 해 동안 발견된 IoT 악성코드 수의 3배가 넘는다[2]. 대부분의 IoT 기기의 악성코드는 봇넷을 생성하여 DDoS(Distributed Denial-of-Service) 공격을 촉진하는 것이 목표이다. 이러한 예로는 2016년 10월 미라이 소스코드가 공개된 이후 발생한 미라이 변종 봇넷을 통한 다수의 DDoS 공격이 있다. 이에 따라 악성코드를 제어하는 C&C 프로토콜을 분석하는 일도 알려지지 않은 비공개 프로토콜에 대한 구조분석 기술 확보가 필요한 이유 중 하나이다.

SAMBA는 마이크로소프트의 SMB(Server Message Block) 프로토콜을 리버스 엔지니어링하여 윈도우와 다른 시스템 간의 파일 및 프린터 공유를 할 수 있게 해주는 프로젝트이다. SAMBA의 예와 같이 비공개 네트워크 프로토콜의 사양을 추출하는 작업인 프로토콜 리버스 엔지니어링은 이기종간의 상호운용성면에서 활용할 수 있고, 효율적인 네트워크 관리 및 보안 문제를 해결하기 위해 반드시 필요하다. 네트워크 보안 분야에서는 이전에 알려지지 않은 공격의 탐지 및 차단을 위한 방화벽과 침입탐지 시스템에 도움이 될 수 있으며 네트워크 취약성을 파악하기 위한 침투 시험 및 스마트 퍼저 시스템 구축, 급증하는 악성코드 프로토콜 분석 등에 사용되는 지능형 DPI(Deep Packet Inspection)의 일부분으로 유용한 정보를 제공할 수 있다. 하지만 이러한 정보를 제공하기 위해서는 프로토콜에 대한 깊이 있는 이해가 먼저이며 기반이 되는 기술이 프로토콜 리버스 엔지니어링이다.

수동 프로토콜 리버스 엔지니어링으로부터 시작된 프로토콜 리버스 엔지니어링은 최근 자동화 된 방법론으로 발전해 왔다. 하지만, 아직까지 프로토콜의 구조 및 사양을 명확하게 추출하는 방법론은 없으며, 각각 방법론의 장단점이 뚜렷하다.

본 논문에서는 네트워크 트레이스를 기반으로 상세한 프로토콜 사양 추출을 위한 프로토콜 리버스 엔지니어링 방법을 제안한다. 선행연구인 A^2PRE[3](Advanced Automatic Protocol Reverse Engineering)에서 사용한 CSP(Contiguous Sequential Pattern) 알고리즘과 SP(Sequential Pattern) 알고리즘을 계층적 결합시켜 사용하여 syntax를 추출하고 2단계 중복제거 필터를 통해 SP 알고리즘을 적용 시 발생가능한 문제점을 최소화한다. 이를 통해 Message Format을 상세하게 추출하는 방법을 제안한다.

본 논문의 구성은 본 장의 서론에 이어 2장에서 관련 연구 및 해결하고자 하는 문제를 정의하고 3장에서 제안하는 프로토콜 리버스 엔지니어링 방법론에 대해 상세히 기술한다. 4장에서는 Message Format의 성능지표에 따른 결과를 비교하고 제안하는 방법론의 성능을 검증한다. 마지막으로 5장에서 결론 및 향후 연구를 기술하며 끝맺음한다.

Ⅱ. 연구 배경 및 관련 연구

1. 프로토콜 리버스 엔지니어링

프로토콜은 데이터 통신을 위한 규약이며 프로토콜의 사양이라는 것은 프로토콜을 구성하는 3가지 요소인 구문, 의미, 타이밍이다. 프로토콜 리버스 엔지니어링에서는 프로토콜 데이터에서 어떤 형식을 가지고 있는지, 어떤 의미를 가지고 있는지, 어떤 순서로 동작하는지를 알아내는 것을 목표로 한다[4].

서론에서 언급한 것과 같이 프로토콜 리버스 엔지니어링은 알 수 없거나 문서화되지 않은 프로토콜 형식의 사양을 추출하는 것이다. 즉 해당 프로토콜에 대한 다수의 트레이스, 바이너리 또는 소스 코드를 수집하여 네트워크 메시지의 구조를 분석하는 것이다. 네트워크 기반 분석 방법은 네트워크 패킷 캡쳐 도구를 이용하여 수집한 네트워크 트레이스를 기반으로 분석하는 방식이고 실행기반 분석 방법은 실제 바이너리를 입수하여 이를 실행과정을 모니터링하면서 수집한 데이터를 기반으로 분석하는 것이다.

2. 관련 연구

Netzob[5]은 프로토콜 구조의 추론 과정 중 일부를 자동화하는 반자동 방법론이다.

Netzob에서는 유전자 염기서열을 분석하는 알고리즘인 Needleman & Wunsch와 UPGMA(Unweighted Pair Group Method with Arithmetic Mean) 계층적 클러스터링 알고리즘을 사용하여 클러스터링을 수행한다. Needleman & Wunsch 알고리즘으로 유사도를 측정하고, 유사도 기준으로 UPGMA를 수행하여 클러스터링을 한다. 이러한 방법론의 차이로 직접적인 성능을 비교하는 데 제약사항이 존재한다.

표 1. 선행연구 비교가능 여부
Table 1. Comparability of preceeding studies

Name	Year	Clustering	Comparability
AutoReEngine	2013	Bottom-Up	O
Netzob	2014	Top-Down	X
Ji *et al.*	2016	Bottom-Up	O
A^2PRE	2017	Bottom-Up	O

AutoReEngine[6]과 A^2PRE는 Apriori 알고리즘을 기반으로 Bottom-UP 방식의 클러스터링을 수행하는 방법론이고 본 논문에서 제시하는 방법론과 1:1 성능 비교가 가능하다.

AutoReEngine은 크게 Data Pre-processing, Protocol Keyword Extraction, Message Format Extraction, State Machine Inference의 4 단계로 구성된다. Data Pre-processing 단계에서는 입력 트래픽을 플로우로 분류하고 플로우 내의 패킷들을 메시지로 재조립한다. Protocol Keyword Extraction 단계는 크게 두 가지 step으로 진행된다.

첫 번째 step인 Frequence Strings Extraction에서는 메시지 시퀀스들을 입력받아 Apriori 알고리즘을 통해 Field Format의 후보인 Keyword를 추출한다.

두 번째 step인 Variance Analysis에서는 앞 단계에서 출력된 바이트 시퀀스들에 대하여 각 영역에서 Variance를 구한다. 이 값이 특정 임계치보다 작을 시 Keyword로 추출한다.

Message Format Extraction 단계에서는 앞 단계에서 추출된 Protocol Keyword들을 length-1 항목집합으로 하고, Transaction을 메시지 시퀀스들로, Support의 단위를 Session Support rate, Site-specific session set Support rate로 설정하여 하나의 동일한 메시지에서 발생하는 Keyword의 연속이 빈번하게 발생할 시 이 Keyword Series를 추출하여 이를 Message Format으로 선정한다.

Ji *et al.*[7]는 문자열 검색 알고리즘인 Aho-Corasick을 사용하여 프로토콜 키워드를 추출한 다음, FP-growth 알고리즘을 사용하여 연관관계 분석을 통해 메시지 형식을 추출한다. Bottom-Up방식 클러스터링을 사용하는 점에서는 본 방법론과 유사하여 비교가능 할 것으로 보이나 비교하기 위해서는 해당 방법론을 구현해야하는 제약사항이 있어 비교대상에서는 제외한다.

A^2PRE는 AutoReEngine과의 유사하게 Apriori 알고리즘을 기반으로 입력된 프로토콜 네트워크 트래픽을 분석하는 방법론이다. 크게 Message Assemble, Syntax Inference, Semantics Inference, Behavior Inference의 4단계로 구성된다.

Message Format을 추론하는 과정은 Message Assemble, Syntax Inference 2단계로 이뤄진다. 전처리 과정에서 수집한 패킷들을 5-Tuple(출발지·목적지 IP, 출발지·목적지 port, 전송 계층 프로토콜)의 기준으로 시간 순으로 정렬한다. Message Assemble 단계에서는 수집한 프로토콜의 플로우를 로드하고 메시지 단위로 분할한다. Syntax Inference 단계에서는 Static Field인 SF(v)와 규칙성이 있는 Dynamic Field인 DF(v)를 추출하고 이를 기반으로 Message Format을 추출한다. 그리고 각 Message Format에 필드사이에 추가적인 Field Format을 추출한다.

3. 문제 정의

본 연구진들의 선행연구인 A^2PRE를 시험한 결과, 개선해야 할 부분이 확인되었다. 특정 트래픽을 대상으로 분석을 수행했을 때 빈도가 높은 핵심 키워드를 추출하였지만 AutoReEngine는 평균 4.3개의 필드로 Message Format을 구성하였고 A^2PRE는 평균 5.3개의 필드로 Message Format을 구성하였다. A^2PRE는 AutoReEngine 대비 동등 이상의 Message Format 추론이 가능하지만 다수의 필드로 구성되어있는 프로토콜 구조를 표현하는 데는 다소 부족한 점이 있다는 판단을 하였다. 즉, 상세하게 프로토콜의 구조를 추론하는 데에는 한계점이 존재한다.

본 연구진의 기존연구인 A^2PRE가 가지고 있는 알고리즘의 장점을 최대한 유지하고 이를 개선하기 위한 방향으로 SP 알고리즘을 적용하였다. SP를 적용하였을 때 장점으로는 Message Format을 구성하는 Field Format의 개수가 증가하는 경향이 뚜렷하

게 보였고 단점으로는 CSP를 적용하였을 때 36개였던 Message Format이 SP를 적용하였을 때 528개로 과도하게 증가하는 점이 확인되었다[8]. 많은 Message Format이 생성되는 경우 분석가 입장에서는 프로토콜의 구조를 직관적으로 파악하기가 어렵고 신속한 탐지 및 대응이 필요한 시스템에서는 Message Format과 타켓 네트워크 프로토콜의 매칭 속도가 증가하게 되는 단점이 있다.

따라서 본 논문에서는 선행연구에서 확인된 문제점을 해결하기 위해 본 연구진은 CSP와 SP 알고리즘을 계층적으로 결합하고 중복관계 필터 및 포함관계를 제거하는 필터, 즉 2단계 필터처리를 통해 Message Format을 최적화하고 상세한 프로토콜 구조를 추론하는 방법을 제안한다.

Ⅲ. 제안하는 프로토콜 구조 추출 방법론

1. 제안하는 방법론의 Overview

본 논문에서 제안하는 방법론의 전체적인 구조는 그림1과 같다. 본 방법론은 크게 Message Assemble, Field Format Inference, Message Format Inference, Inclusion Relation Filter로 4단계로 나뉘어져 있다.

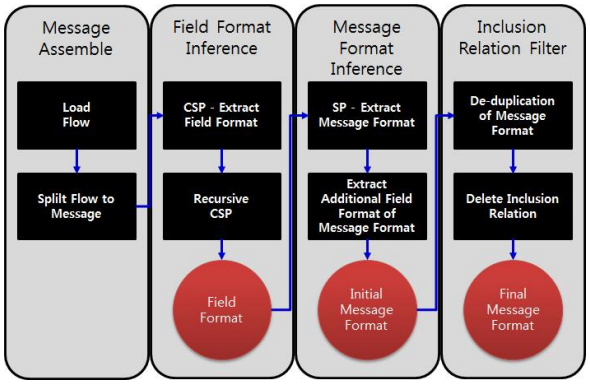

그림 1. 제안하는 방법론의 Overview
Fig. 1. Overview of Suggested Methodology

Message Assemble 단계는 수집한 프로토콜 데이터를 사용하기 위해 데이터 전처리를 하는 단계이다. 5-Tuple이 같은 패킷들을 시간 순으로 정렬하여 양방향 플로우 단위로 변환 하고 메시지 단위로 분할한다.

Field Format Inference는 CSP 알고리즘을 적용하였다. 메시지 시퀀스 데이터베이스 내에서 특정 Support를 만족하는 연속적인 바이트스트림을 Field Format의 SF(v)로 추출하고 일정조건을 가진 SF(v)를 Recursive CSP를 통해 DF(v)로 추출한다.

Message Format Inference는 Field Format Inference에서 추출된 Field Format을 기준으로 SP 알고리즘을 적용하고 가능한 모든 조합을 수행하여 특정 Support를 만족하는 Field Format의 조합을 Message Format으로 추출한다. 이후 Message Format의 SF(v)와 DF(v) 사이에 해당하는 부분의 데이터를 분석하여 조건에 따라 추가적인 필드를 추출한다.

Inclusion Relation Filter는 Message Format Inference에서 추출된 Message Format을 최적화 작업을 수행한다. 우선 생성된 Message Format을 기준으로 Message Format의 분석하는 메시지리스트에 따른 중복제거 작업을 수행한다. 그리고 포함관계 기준으로 우선순위 작업을 수행하고 Message Format의 중복제거 작업을 수행한다. 즉 두 단계의 필터 처리를 통해 최종적으로 상세한 구조를 가진 Message Format을 추출하게 된다.

2. CSP와 SP의 결합

본 방법론은 Message Format에 안에 최대한 많은 필드를 추출하여 상세한 프로토콜 구조를 추론하는 목적을 가지고 있다. 선행연구에서는 3단계의 Hierarchical CSP를 적용하여 Field Format, Message Format, Flow Format을 추출하였다. 하지만 CSP 알고리즘을 사용하여 Message Format을 추출할 경우 항상 인접하여 연결되어 있는 순차 패턴만을 추출하므로 다양한 Message Format의 추출이 제한된다는 단점을 가지고 있다. 물론 Field Format을 추출하기 위해서는 연속된 바이트 스트림으로 구성된 키워드들을 조합을 통해 선발해야한다. 즉 CSP를 적용시켜서 일정 Support 이상을 만족시키는 후보자를 선택하는 것을 Level 별로 증가시키면서 수행하고 최종적으로 Field Format을 추출해야한다. Field Format을 추출할 때는 연속성이 있는 키워드들을 추출해야하므로 반드시 CSP를 사용해야하지만 Message Format을 추출해야 할 때에는 꼭 CSP를 사용할 이유는 없다.

그림2은 4개의 메시지가 각각 A-B-C, A-C, A-D-B, A-C-D 키워드로 구성되어있다고 가정하고 support 50%이상을 기준으로 CSP와 SP 알고리즘 적용 시킨 결과이다. 연속성 있는 순차 패턴을 추출하는 CSP대비 순차 패턴을 추출하는 SP를 적용할 경우에 더 많은 패턴을 추출하는 것이 확인된다.

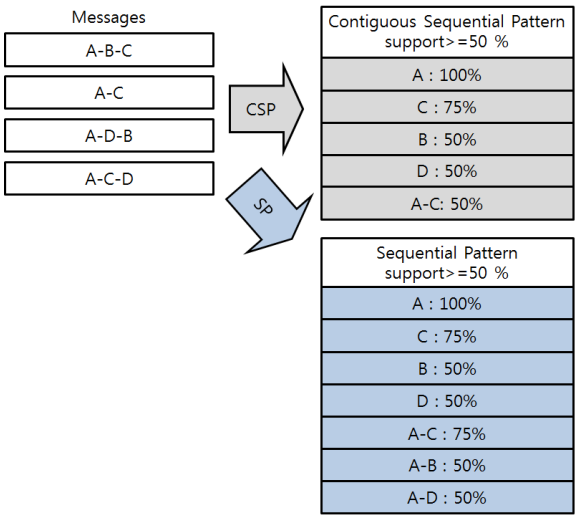

그림 2. CSP와 SP 알고리즘 예
Fig. 2. Example of CSP and SP Algorithm

다만, 모든 시계열 순차 패턴을 추출하게 되므로 수행시간이 CSP 보다 길어지는 단점이 있지만 위의 예처럼 다양한 조합의 Message Format의 추출이 가능한 장점이 있다.

특히 HTTP의 Header Line과 같이 항상 필드의 위치와 순서가 가변적이며 선택적으로 존재하는 레코드 유형의 필드가 존재하는 메시지 구조의 경우 메시지의 처음부터 끝까지 표현하는 Message Format이 아닌 일부분만을 표현하는 단편적인 Message Format만을 추출하게 된다. 예를 들면 정상적인 HTTP메시지는 헤더정보가 0x0d0a0d0a로 끝나는 것으로 항상 나타나기 때문에 Field Format으로 추출이 가능하다. 하지만 Message Format을 추출하는 단계에서는 CSP를 적용한 경우 Message Format의 필드로 추출되지 않을 수 있다. 그러나 SP 알고리즘을 사용할 경우 모든 조합을 처리하기 때문에 위와 같은 가변위치의 필드도 추출이 용이하다. 즉, 각 Length에서 더 다양한 패턴을 추출할 수 있고 이 패턴들은 다음 Length 조합의 후보자가 된다. 이와 같이 각 Length별로 Support를 만족시키는 만큼 반복하게 되면 결과적으로 많은 필드조합을 통해 Message Format의 세분화를 가능하게 한다. 이를 통해 HTTP, DNS처럼 레코드 유형의 필드가 존재하는 메시지 구조 또한 더 상세하게 추출이 가능하다.

3. Message Format 중복 제거 필터

Message Format Inference에서 SP 알고리즘을 적용하여 가능한 모든 조합의 Message Format을 생성한다. 너무 많은 조합의 결과를 추론하기 때문에 이를 최적화하는 방법이 필요하다. 1차적으로 적용하는 Message Format 종복제거 방법은 아래의 그림 3과 같다.

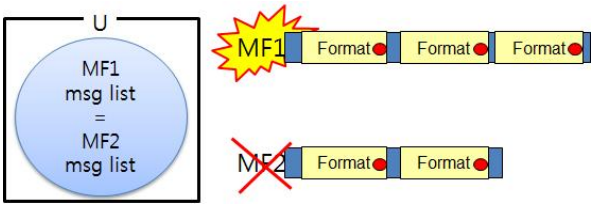

그림 3. Message Format 중복 제거 개념
Fig. 3. The Concept of Message Format De-duplication

두 개의 Message Format을 비교하였을 때 방향이 동일하고 각각의 Message Format의 메시지들의 집합이 같다면 전체 메시지를 일부 메시지 집합을 표현하는 두 개의 Message Format이라고 볼 수 있다. 즉 두 개의 Message Format은 중복되어있다고 판단 할 수 있다. 이럴 경우 좀 더 상세한 구조를 가진 Message Format을 선택하고 나머지는 제거한다. 이 과정을 전체 Message Format을 대상으로 수행하고 새로운 Message Format 리스트를 생성한다.

4. Message Format 포함관계 제거 필터

기본적인 중복제거 필터를 통해 Message Format의 개수를 다소 줄였지만 Message Format의 최적화는 충분하지 않다. 이를 해결하기 위해 Message Format의 포함관계를 검토하였다. 각각의 Message Format는 일정 Support 기준 이상의 메시지들에서 추출된 것이다. 따라서 각각의 메시지는 1개 이상의 Message Format에 포함될 수 있다. 여기서 착안한 필터의 기본적인 원리는 다음과 같다. 각 Flow의 하나의 Message는 여러 개의 Message Format이 아닌 하나의 Message Format에만 포함되도록 하는 것이다. 이를 위해 일반적인 프로토콜의 구조 유형을 분석한 다음에 그림4과 같은 통찰을 기반으로 조건을 정하였다.

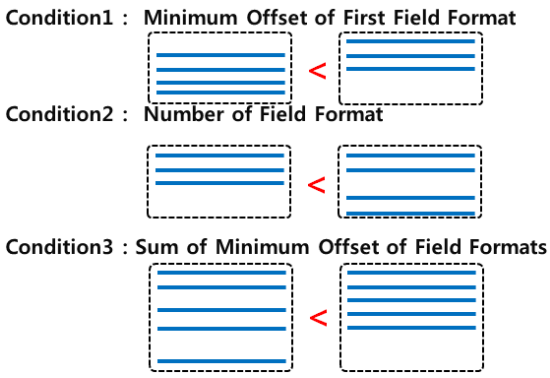

그림 4. Message Format과 메시지와 매핑
Fig. 4. Mapping condition of Message Format to Message

첫째, 일반적인 프로토콜은 앞에서부터 Field가 존재한다. 즉 첫 번째 Field Format의 offset이 첫 번째 조건이다.

둘째, Message Format을 구성하는 Field Format의 개수가 많을수록 세부적인 프로토콜 구조를 추론할 수 있다. 즉 Field Format의 수가 두 번째 조건이다.

셋째, 일반적인 프로토콜 구조는 앞쪽이 필드가 존재하고 뒤쪽에 페이로드가 있다. 즉 메시지 내 Field Format들의 offset의 합이 세 번째 조건이다.

이러한 세 가지 조건을 기본으로 Message Format과 메시지들의 매핑을 수행한다. 최종적으로 메시지와 매핑되지 않은 Message Format을 삭제한다. 이 과정을 통해 최종 Message Format을 생성한다.

Ⅳ. 실험

직관적이며 명확한 분석을 위해 가독성이 좋고 수집도 용이한 Text 기반 프로토콜인 HTTP 프로토콜의 두 가지 트래픽 셋과 Binary 기반 프로토콜인 DNS 프로토콜의 한 가지 트래픽 셋을 수집하고 실험을 진행하였다. 표 2는 실험에 사용한 트래픽의 정보이다.

표 2. 실험 트래픽 정보
Table 2. Traffic Information

Traffic set	Flow	Pkt	Message	Byte
HTTP Set1	359	3,841	1,189	4,674,813
HTTP Set2	33	2,827	129	2,531,785
DNS Set	963	2,000	2,000	354,872

위 실험 트래픽을 기준으로 AutoReEngine과 선행 연구한 A^{2}PRE, 본 논문에서 제시한 A^{2}PRE(SP), 3가지 알고리즘을 적용하여 실험을 수행하였다. 객관적인 평가 및 검증 방법을 위해 선행연구[9]를 참고하여 HTTP, DNS 프로토콜의 정답지와 성능지표를 정의하였다.

1. 정답지 정의

정답지를 정의하기 전에 먼저 용어 정리를 하겠다. 입력되는 트래픽에서 알고리즘이 추출해야하는 키워드가 TF(True Field)이다. TF들의 조합으로 구성되는 Message Format을 TM(True Message Format)이라고 정했다. EF(Extracted Field)는 알고리즘으로 추출된 Field Format을 의미하고 EM(Extracted Message Format)을 구성한다.

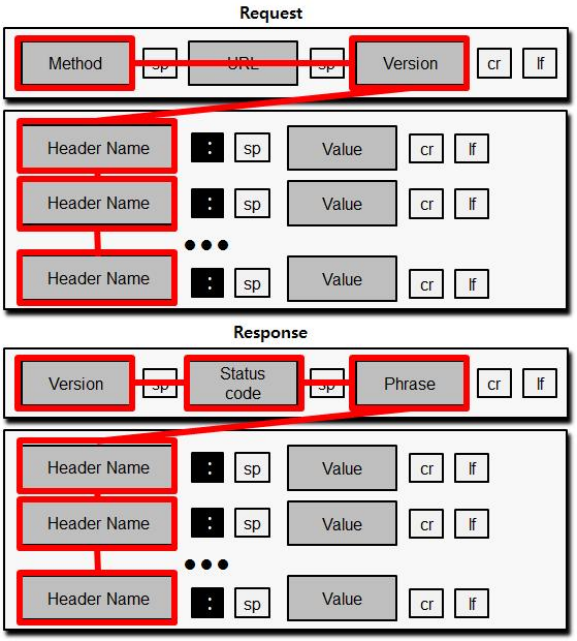

그림 5. HTTP 정답지 정의
Fig. 5. Definition of Answer for HTTP

HTTP의 정답지는 그림5와 같이 HTTP 프로토콜을 Request, Response로 구분하여 헤더 부분에서 나타나는 정보인 Method, Version, Status Code, Phrase, Header Name 등 HTTP 프로토콜을 구성하는 중요한 키워드들을 TF로 선정하여 정답지를 구성하였다.

입력 트래픽에서 정답지로 설정된 TF를 기준으로 TM을 추출한 결과가 그림6와 같다. 주요 키워드인 GET, HTTP/1.1 등의 정보가 순서대로 나열되어있다. 각각의 TM은 연속된 키워드들의 순서로 출력되며 중복되지 않는다.

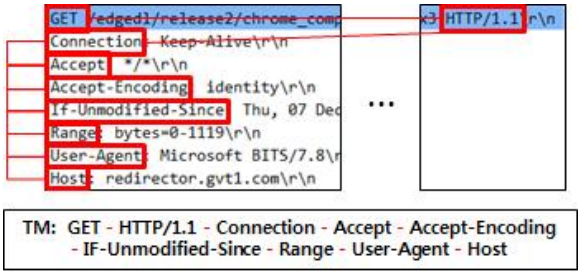

그림 6. TM의 예
Fig. 6. Example of TM

DNS 프로토콜은 그림 7의 구조를 가지고 있다. DNS 프로토콜의 구조는 Query, Response 2가지로 나뉘는데 Identification에서 Query Class까지의 구조는 동일하다. Response는 추가로 Domain Name부터 Resource Data까지 1회 이상 반복되는 가변 레코드 구조를 가지고 있다. 그림 7의 (1), (2), (3) 영역의 각 Byte를 TF로 선정하였고 Query 정답지는 경우 (1), (2) 영역을, Response 정답지는 (1), (2), (3) 영역을 TM으로 정하였다.

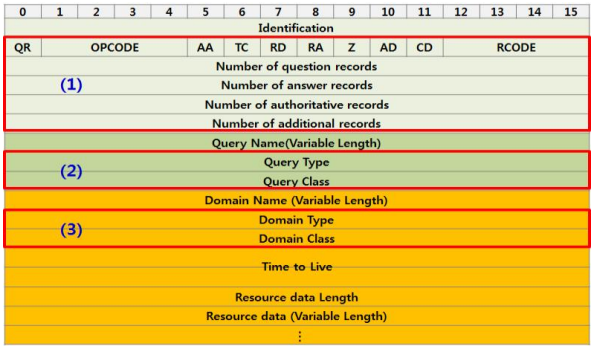

그림 7. DNS 정답지 정의
Fig. 7. Definition of Answer for DNS

2. 성능지표 정의

앞서 정의한 TM과 알고리즘을 통해 추출된 EM의 비교를 통해 Message Format의 상세화 정도를 확인한다.

$$P(A,B) = \begin{cases} 1, \text{ if } A \subset B \\ 0, \text{ if } A \not\subset B \end{cases} \quad (1)$$

$$Detail_{EMi} = \frac{\sum_{TM_j \in EM_i} \sum_{y \in EM_i} \sum_{x \in TM_j} P(x,y)}{n(f \mid f \in (TM \in EM_i)} \quad (2)$$

$$Detail_{Total} = \sum_{i=1}^{n} \frac{Detail_{EMi}}{n(EM)} \quad (3)$$

$$Coverage = \sum_{i=1}^{n} \frac{n(Msg \mid Msg \in EMi)}{n(Msg)} \quad (4)$$

수식 (1)은 A가 B에 속하는 경우 1을 반환하는 수식을 의미하며 A는 TM에 포함된 TF, B는 EM에 포함된 EF이다. 수식 (2)의 $Detail_{EMi}$는 하나의 EM기준으로 EM에 포함되는 TM의 TF와 EF가 매칭되는 개수를 확인하여 개별 Message Format의 상세화 정도를 구하는 수식이다. 수식 (3)의 $Detail_{Total}$은 각각의 $Detail_{EMi}$으로 전체 Message Format에 대한 상세화 정도를 구한다. 즉, 위의 수식들은 EM이 얼마나 TM을 상세하게 반영하는지 수치화 한 것이다. EM에 속하는 EF와 TM에 속하는 TF가 매칭되는 항목에 점수를 부여하여 계산하는 방식으로 Detail을 정의하였으며 각각의 EM별 Detail을 구하고 이를 바탕으로 최종 $Detail_{Total}$를 구해 알고리즘 별 성능을 비교하였다. 수식 (4)의 Coverage는 전체 EM이 분석할 수 있는 메시지 개수/전체 메시지 개수이고 성능평가의 보조지표로 사용한다.

3. 실험결과

A^{2}PRE와 A^{2}PRE(SP)는 Support를 Field Format Min.Supp : 65%, Msg Format Min.Supp : 50%, Flow Format Min.Supp : 50%로 입력하였고 AutoReEngine의 경우에는 Site-Session set Min.Supp : 50%, Session Min.Supp : 50%로 입력하였다.

표 3. 메시지 포맷들의 평균 필드 수
Table 3. Average Number of Fields in Message Formats

Traffic set	AutoReEngine	A^{2}PRE	A^{2}PRE(SP)
HTTP Set1	4.35	5.39	9.74
HTTP Set2	3.44	4.37	8.64
DNS Set	3.67	3.38	7.25

표3에서 보듯이 Message Format을 구성할 때 A^{2}PRE는 AutoReEngine 대비 많은 필드를 가지고 있다. 그리고 A^{2}PRE(SP)는 A^{2}PRE 대비 2배정도의 필드 수를 가지고 있다. 즉, 필드 추출 개수가 많은 것을 확인할 수 있다.

앞서 정의한 Detail 성능 평가 지표를 기반으로 프로토콜별 Request(Query)와 Response로 구분하여 알고리즘별 성능을 비교해 보았다. 실험을 통해 본 연구진이 제안한 방법론이 정답지 기준으로 얼마나 Message Format을 상세하게 추출하였는지를 확인하였다.

그림 8에서 확인되는 것과 같이 A²PRE(SP)는 AutoReEngine, A²PRE 대비 전반적인 성능에 우위가 있다는 것을 확인하였다.

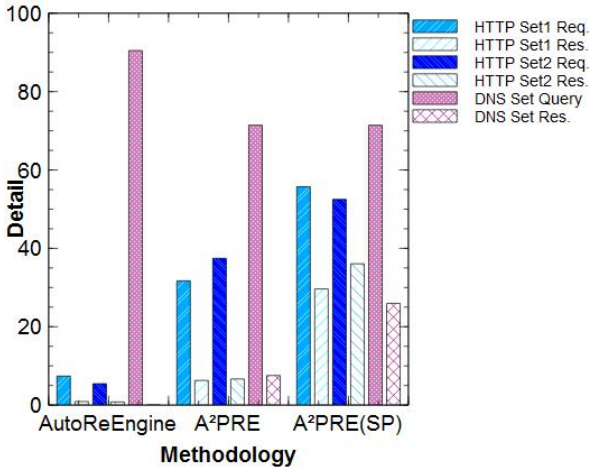

그림 8. Detail 지표를 통한 성능비교
Fig. 8. Performance Comparison by Detail Index

HTTP 프로토콜의 경우는 A²PRE(SP)는 AutoReEngine 대비 약 10배의 수치 상세화 정도를 보였고 A²PRE 대비 약 2배의 수치 향상을 기록하였다. DNS 프로토콜의 경우는 AutoReEngine이 Request추출에서는 A²PRE, A²PRE(SP) 대비 우수함을 보였지만 복잡한 구조의 Response 정보를 추출하지 못하는 것이 확인됐다.

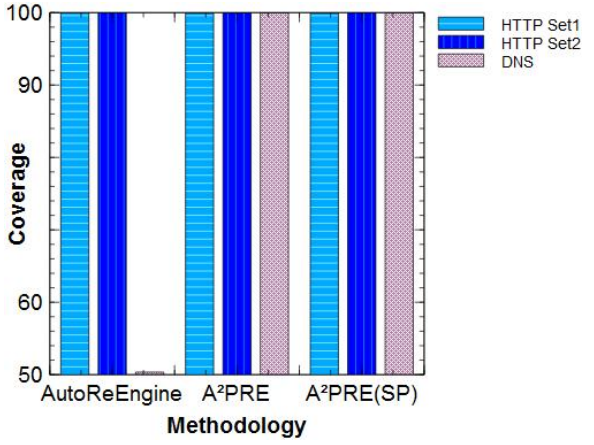

그림 9. Coverage 지표를 통한 성능비교
Fig. 9. Performance Comparison by Coverage Index

그 결과가 그림 9에서 AutoReEngine의 DNS Set Coverage가 약 50%로 나타나는 이유이다. A²PRE(SP)는 Request에서는 A²PRE와 동일한 결과를 보여주었지만 레코드 유형이 반복되는 구조인 Response에서는 약 3배 이상의 성능을 보였다.

표 4. TM 대비 메시지 포맷의 수
Table 4. Number of Message Format per TM

Traffic set	TM	AutoReEngine	A²PRE	A²PRE(SP)
HTTP Set1	129	60	36	43
HTTP Set2	40	32	30	25
DNS Set	72	3	14	20

또한, 선행연구에서 SP를 적용시켰을 때 너무 많은 Message Format을 생성하여 발생하던 문제가 있었다. 표4을 보면 TM 대비 각각의 방법론에서 추출된 Message Format의 개수를 확인이 가능하다. 중복제거 필터가 적용된 A²PRE(SP)는 TM 대비 낮은 수치를 가지는 것뿐만 아니라 기존 방법론과 대동소이한 결과를 보여준다. 즉 입력 메시지들을 충분히 압축하여 클러스터링 한 결과를 보여준다. 단순 SP를 적용시켰을 때 나타난 문제가 해소됨이 확인되었다.

Ⅴ. 결 론

본 논문에서는 상세한 프로토콜 구조를 추론하는 프로토콜 리버스 엔지니어링을 위한 방법론을 제안하였다. Bottom-Up방식의 프로토콜 구조 분석 기법을 기반으로 CSP와 SP를 결합한 알고리즘 및 2단계 중복 제거 필터를 적용하여 상세한 Message Format을 추출하는 방법을 구현 및 검증하였다. 기존 방법론인 AutoReEngine, A²PRE와 제안한 방법론인 A²PRE(SP)과 성능검증을 수행하였으며, Detail 성능지표를 적용한 실험을 통해 적으면 2배에서 많으면 10배까지 기존 방법론 대비 향상된 점을 확인할 수 있다. 다만 패턴과 통계기반으로 분석을 수행하는 본 방법론으로는 전체 페이로드가 암호화된 프로토콜을 분석을 수행할 경우 유의미한 결과를 추출하지 못하는 한계가 있다.

본 방법론의 성능개선을 위해 향후 연구로는 Top-Down 방법론과 Bottom-Up 방법론이 가지고 있는 각각의 장점을 취할 수 있는 연구를 진행할 계획이다.

References

[1] Cisco, Cisco Visual Networking Index: Forecast and Trends, 2017-2022 (2019), Retrieved Jun., 10, 2019 from https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.pdf

[2] Mikhail Kuzin, Yaroslav Shmelev, Vladimir Kuskov, New trends in the world of IoT threats (2018), Retrieved Jun., 10, 2019 from https://securelist.com/new-trends-in-the-world-of-iot-threats/87991/

[3] 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

[4] Min-Seob. Lee, Young-Hoon. Goo, Kyu-Seok Shim, and Myung-Sup Kim, "A Study on the Extraction of Highly Accurate Static Fields in Protocol Reverse Engineering", *KNOM Review*, Vol. 21, No. 1, pp.10-17, Aug. 2018

[5] Bossert, Georges, Frederic Guihery, 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

[6] 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

[7] Ran Ji, Haifeng Li and Chaojing Tang, "Extracting keywords of UAVs wireless communication protocols based on association rules learning", *in Proceeding 12th International Conference on Computational Intelligence and Security*, pp.310-313, Wuxi, China, Dec. 2016

[8] Young-Hoon. Goo, Kyu-Seok Shim, Min.-Seob. Lee, Jee-Tae Park, and Myung-Sup Kim, "Comparison of Message Format Extraction Performance Using CSP Algorithm and Using SP Algorithm in Protocol Structure Analysis", *in Proceeding KICS Winter Conference 2019*, pp.1-2, Pyeongchang, Korea, Jan. 2019

[9] Young-Hoon. Goo, Kyu-Seok Shim, Min.-Seob. Lee, Jee-Tae Park, and Myung-Sup Kim, "A Study of Evaluation and Validation Method for Protocol Reverse Engineering", *in Proceeding KICS 2018 Summer Conference* pp.746-747, Jeju Island, Korea, Jun. 2018

채 병 민 (Byeong-Min Chae)

2007. 02 충남대학교, 물리학과 학사
2012. 02 충남대학교 컴퓨터공학과 석사
2007 ~ 2008 삼성전자 연구원
2008 ~ 현재 한화시스템 연구원
<관심분야> 네트워크 관리 및 보안, 트래픽 모니터링 및 분석

문 호 원 (Ho-Won Moon)

2000. 02 한양대학교, 수학과 학사
2000 ~ 2011 삼성전자 연구원
2011 ~ 현재 한화시스템 연구원
<관심분야> 네트워크 관리 및 보안, 트래픽 모니터링 및 분석, 라우팅

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

이 민 섭 (Min-Seob Lee)

2018 고려대학교 컴퓨터정보학과 학사
2018 - 현재 고려대학교 컴퓨터정보학과 석사과정
<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