명령 지향적 프로토콜을 위한 리버스 엔지니어링 방법

구영훈, 백의준, 신무곤, 김명섭
고려대학교
{gyh0808, pb1069, tm0309, tmskim}@korea.ac.kr

A Method for Reverse Engineering of Command-Oriented Protocols

Young-Hoon Goo, Ui-Jun Baek, Mu-Gon Shin, Myung-Sup Kim
Korea Univ.

요 약

새로운 응용 및 사이버 공격의 지속적인 출현과 빈번한 갱신으로 인해 자동 프로토콜 리버스 엔지니어링의 필요성이 강조되고 있다. 특히, 근 10년 간의 꾸준한 보안사고와 사이버 갈등의 심화는 프로토콜 리버스 엔지니어링의 중요성을 대변한다. 다양한 자동 프로토콜 리버스 엔지니어링 방법론이 제안되었으나, 메시지 포맷 추론을 목적으로 하는 방법론들의 경우, 프로토콜 키워드의 빈도를 기반으로 하는 알고리즘을 사용하기 때문에, 키워드들의 분포가 균등 분포를 따르는 명령 지향적 프로토콜의 사양을 추출하기에 한계점을 가진다. 본 논문에서는 순차 패턴 마이닝 기술을 사용하여 명령 지향적 프로토콜의 사양을 추출하는 방법을 제안한다. 또한, 제안한 방법을 실제 프로토콜에 적용한 결과를 제시하여 해당 방법론의 타당성을 보인다.

Ⅰ. 서론

전세계적으로 IP 트래픽은 26.5%의 연평균 성장률로 빠르게 증가하고 있으며, 모바일 트래픽의 연평균 성장률은 IoT 기술의 발전으로 인해 46%에 달할 것으로 전망된다. 5G 네트워크의 출현은 더욱 빠른 서비스를 가능하게 하지만, 이러한 고속 네트워크에서의 보안에 취약한 모바일 및 IoT 장치의 증가는 새로운 응용 및 악성 행위도 함께 증가시킨다. 이에 따라, 알 수 없는 프로토콜의 사양을 추론하는 행위인 프로토콜 리버스 엔지니어링의 중요성은 앞으로 더욱더 크게 증가할 것이다.

프로토콜 리버스 엔지니어링 방법에 있어 네트워크 트레이스 기반 분석 방법은 실행 트레이스 기반 분석 방법에 비해 보다 자동화, 실용성 측면에서 더 우수한 방법이다 [1]. 하지만, 프로토콜 유한 상태 머신 추출이 아닌 메시지 포맷 추론을 목적으로 하는 네트워크 트레이스 기반 방법론들의 경우, 여러 네트워크 트레이스 내 프로토콜 키워드의 빈도를 기반으로 하는 알고리즘을 사용하기 때문에, 전반적으로 키워드들의 빈도가 매우 낮은 명령 지향적 프로토콜의 사양을 추출하기에 한계점을 가진다. 본 논문에서는 순차 패턴 마이닝 기술을 사용하여 명령 지향적 프로토콜의 사양을 추출하는 방법을 제안한다. 또한, 제안한 방법을 실제 프로토콜에 적용한 결과를 제시하여 해당 방법론의 타당성을 보인다.

본 논문은 2장에서 명령 지향적 프로토콜과 비명령 지향적 프로토콜에 대한 설명과 함께 선행 연구의 한계에 관해 설명하고, 3장에서 명령 지향적 프로토콜을 위한 리버스 엔지니어링 방법에 대해 설명한다. 4장에서는 실험결과를 기술하고. 5장에서는 결론 및 향후 연구를 언급한다.

Ⅱ. 명령 지향적 프로토콜과 비명령 지향적 프로토콜

네트워크 트레이스 기반의 프로토콜 리버스 엔지니어링은 방법론은 메시지 포맷 추론을 위해 Sequence Alignment, 자연어 처리, 연관 규칙 마이닝 등의 빈도와 관련된 기술을 사용한다. 명령 지향적 프로토콜이란 Request 메시지 포맷이 [Command]-[Arguments]의 형태를 가지고, Response 메시지 포맷은 [3digit Code]-[Phrase]의 형태를 가지며, 사용자가 해당 프로토콜을 따르는 프로그램 바이너리를 1회 실행 시, 다양한 명령어를 사용하는 프로토콜을 말한다. 즉, 명령 지향적 프로토콜은 1개의 세션 내에 다양한 명령어들이 적은 횟수로 균등하게 분포되는 특징을 갖는다. 예를 들어, FTP의 경우 사용자는 서버에 먼저 로그인을 한다. 그리고 *PWD* 명령어를 통해 현재 작업 디렉토리의 경로를 확인하고 *LIST* 명령어를 통해 현재 디렉토리 내 파일 항목들을 확인한다. 전송모드를 설정하기 위해서는 *TYPE* 명령어를 사용하며 파일 업로드 및 다운로드를 위해 *GET*, *PUT* 명령어를 사용한다. 이러한 명령어와 관련된 프로토콜 키워드들은 네트워크 트레이스 내에 매우 적은 횟수로 발생되며, 로그인과 관련된 키워드는 각 세션 내에서 1회씩만 발생된다. 따라서, FTP, SMTP, POP3, SIP과 같은 명령 지향적 프로토콜의 경우, 패킷 재생 혹은 유한 상태 머신 구축을 목적으로 하는 경우, 리버스 엔지니어링이 가능하나 메시지 포맷 추출을 목적으로 하는 방법론은 한계가 존재한다.

반면, HTTP, DNS, ICMP, MIME과 같은 비명령 지향적 프로토콜은 명령 지향적 프로토콜과 달리 일반적으로 사용하는 프로토콜 키워드가 존재하므로 빈도와 관련된 기술을 활용하여 리버스 엔지니어링이 가능하다. 예를 들어, HTTP의 경우, *Host*, *Referer*, *User-Agent*와 같은 키워드는 다양한 메시지에서 높은 횟수를 가지고 발생하며, 주로 사용되는 메시지 타입은 *GET* 타입이다. DNS는 주소를 이름으로 매핑하는 기능보다는 이름을 주소로 매핑하는 기능을 많이 사용하기 때문에 이와 관련된 필드의 값이 네트워크 트레이스 내에서 높은 비율을 차지한다.

본 논문에서는 이러한 한계를 해결하기 위해 순차 패턴 마이닝 기술 중 하나인 GSP 알고리즘 [2]을 변형하여 명령 지향적 프로토콜을 위한 리버스 엔지니어링 방법을 제안한다.

Ⅲ. 명령 지향적 프로토콜을 위한 리버스 엔지니어링 방법

이 논문은 2018년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업(NRF-2018R1D1A1B07045742) 및 2018년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2018-0-00539-002, 블록체인의 트랜잭션 모니터링 및 분석 기술개발)

본 방법론은 필드 포맷 추출, 메시지 포맷 추출, 메시지 포맷 최적화, 유한 상태 머신 추출과 같이 총 4 단계로 수행된다. 필드 포맷 추출, 메시지 포맷 추출 모듈에서는 본 방법론의 목적에 맞게 변형한 GSP 알고리즘을 사용한다. GSP 알고리즘은 대량의 데이터베이스에서 빈번하게 발생하는 시계열 항목들의 시퀀스를 추출하는 알고리즘이다. 제안하는 방법론은 GSP의 최소 gap constraint와 최대 gap constraint를 0으로 설정하고, 지지도의 개념을 수정한 순차 패턴 마이닝 기술을 사용하여 메시지 포맷을 추출한다. 본 방법론에서는 세션과 메시지에 대한 두 가지 단위의 지지도를 사용하며, 각각 수식 1과 수식 2와 같이 정의한다. 수식 1은 전체 세션 중 해당 패턴을 포함하는 세션의 비율을 의미하며, 수식 2는 전체 메시지 중 해당 패턴을 포함하는 메시지의 비율을 의미한다.

$$Supp_{session} = \frac{n(Sessions\ Having\ Target\ Subsequence)}{n(Total\ Sessions)} \quad (1)$$

$$Supp_{msg} = \frac{n(Messages\ Having\ Target\ Subsequence)}{n(Total\ Messages)} \quad (2)$$

필드 포맷 추출 모듈에서는 먼저, 시퀀스의 항목을 1byte로 설정하고 세션 단위의 지지도가 일정 지지도 이상인 패턴을 추출한다. 추출한 패턴들에 대하여 메시지 단위의 지지도가 일정 지지도 이하인 패턴만을 선별하여 정적 필드 포맷으로 추출한다.

메시지 포맷 추출 모듈에서는 시퀀스의 항목을 정적 필드 포맷으로 설정하고 1 단계와 마찬가지로 세션 단위의 지지도가 일정 지지도 이상인 패턴을 추출한 후, 메시지 단위의 지지도가 일정 지지도 이하인 패턴만을 선별하여 메시지 포맷으로 추출한다.

메시지 포맷 최적화 모듈은 추출된 메시지 포맷을 구성하는 정적 필드 포맷 사이의 빈 부분을 동적 필드 포맷으로 정의하고 입력 받은 실제 메시지들과 비교하여 해당 부분의 모든 값을 저장한다. 이렇게 함으로써 메시지 포맷을 더 구체화시킨다.

유한 상태 머신 추출 모듈은 메시지 포맷과 실제 트래픽을 매칭시켜 각 메시지 포맷을 노드로 하는 유한 상태 머신을 추출한다.

Ⅳ. 실험 및 결과

본 장에서는 제의한 알고리즘을 사용하여 명령 지향적 프로토콜의 사양을 추출한 결과를 제시한다. 이를 위해 FTP 프로토콜의 트래픽을 수집하였다. 실험 결과의 타당성을 높이기 위해 6대 이상의 서로 다른 호스트에서 수집하였으며, 세션 단위의 지지도와 메시지 단위의 지지도에 대한 임계값을 각각 80%, 15%로 설정하였다. 이론적으로, 명령 지향적 프로토콜의 특정 행위는 모든 세션에서 거의 1회씩 발생하며, 모든 메시지 내에서는 매우 적은 횟수로 발생한다. 따라서 세션 단위의 지지도에 대한 임계값은 100%에 가깝게, 메시지 단위의 지지도에 대한 임계값은 전체 세션의 수를 전체 메시지 수로 나눈 비율에 가깝도록 설정하는 것이 옳다. 데이터의 아웃라이어를 고려하여, 두 지지도를 위와 같이 설정하였다.

표 1은 본 논문에서 제안하는 방법을 사용하여 추출한 메시지 포맷의 예를 보여준다. 각 메시지 포맷이 정적 필드와 동적 필드로 세분화되어 있으며 각 필드는 값과 위치에 대한 정보를 가지고 있다. 그림 1은 추출된 유한 상태 머신이며, 각 노드는 표 1의 각 메시지 포맷을 나타낸다. 표 1과 그림 1은 실제 FTP의 로그인 과정을 정확히 반영하고 있다.

Ⅴ. 결론 및 향후 연구

본 논문에서는 기존의 메시지 포맷 추출 목적 리버스 엔지니어링 방법의 한계를 설명하고, 명령 지향적 프로토콜을 위한 리버스 엔지니어링을 위한 방법을 제안하였다. 향후 연구로는 본 방법론을 허니팟 시스템과 연동하여 실시간으로 공격자의 행위에 반응할 수 있는 시스템을 설계할 예정이다.

표 1. 추출된 FTP 프로토콜 메시지 포맷

field type	value	offset	depth
Message ID: 4 / Direction : Response			
static	220 ProFTPD 1.3.4rc2 Server (Debian) [::ffff:172.16.	0	51
dynamic	1	52	52
	3		
	7		
	6		
	2		
	4		
	8		
static	.101]|0d||0a|	53	59
Message ID: 0 / Direction : Request			
static	USER	0	4
dynamic	dbsnmp|0d||0a|	5	15
	scott|0d||0a|		
	sys|0d||0a|		
	system|0d||0a|		
	moderator|0d||0a|		
	anonymous|0d||0a|		
	ftp|0d||0a|		
Message ID: 2 / Direction : Response			
static	331 Password required for	0	25
dynamic	dbsnmp|0d||0a|	26	36
	scott|0d||0a|		
	sys|0d||0a|		
	system|0d||0a|		
	moderator|0d||0a|		
	anonymous|0d||0a|		
Message ID: 1 / Direction : Request			
static	PASS	0	4
dynamic	tiger|0d||0a|	5	27
	pass123|0d||0a|		
	password|0d||0a|		
	password@example.com|0d||0a|		
	|0d||0a|		
Message ID: 3 / Direction : Response			
static	530 Login incorrect.|0d||0a|	0	21

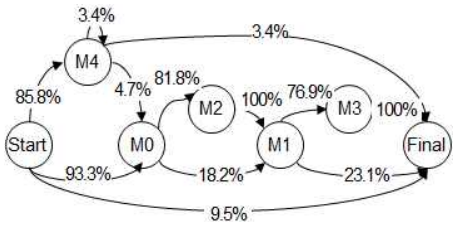

그림 1. 추출된 FTP 프로토콜 유한 상태 머신

참 고 문 헌

[1] Y.-H. Goo, K.-S. Shim, M.-S. Lee, and M.-S. Kim, “Analyzing the Differences Between Network Trace-based and Execution Trace-basd Protocol Reverse Engineering in Three Perspectives”, in Proc. KICS 2017 Summer Conf. pp.82-83, Jun. 2017.

[2] R. Srikant and R. Agrawal, “Mining sequential patterns: Generalizations and performance improvements”, in Proc. 5th Int. Conf. Extending Databased Technol. Adv. Database Technol. Springer, pp. 1 - 17, Mar. 1996.