효율적인 APT 시뮬레이터 프레임워크 제안

허남정[1], 최상훈[2,*], 박기웅[2,†]

[1]세종대학교 시스템보안연구실 (지능형드론 융합전공) 석사과정

[2,†] 세종대학교 정보보호학과 교수 (*연구교수)

An Efficient APT Simulator Framework

Nam-Jung Heo[1], Sang-Hoon Choi[2], Ki-Woong Park[2,†]

[1]SysCore Lab. (Convergence Engineering for Intelligent Drone), Sejong University

[2,†] Dept. of Computer and Information Security, Sejong University

요 약

최근 지속적이고 지능적인 사이버 공격(APT)이 증가하면서 사이버 보안의 주요 위협 요소로 부상하고 있다. 이러한 APT 공격을 효과적으로 대응하기 위해 취약점 점검 도구인 APT 시뮬레이터가 사용될 수 있다. 그러나 APT 시뮬레이터는 단일 시스템 취약점 점검 도구로만 연구되어 왔으며, 다중 에이전트와 협력 및 동시성을 가진 연쇄성 취약점 테스트 구현에 어려움이 있다. 본 논문에서는 기존 APT 시뮬레이터의 성능적 오버헤드를 줄일 수 있는 새로운 프레임워크를 제안한다. 향후 연구에서는 다중 에이전트 기반 APT 시뮬레이터 연구로 발전할 것을 기대한다.

I. 서론

최근 사이버 보안 분야에서 지속적이고 지능적인 공격(APT)이 기업의 자원을 탈취하고 인프라를 망가트리는 등 사이버 보안의 주요 문제로 주목받고 있다. 최근 연구에서 APT 공격이 증가하고 있으며 APT 공격을 점검할 수 있는 연구의 필요성을 강조한다 [1]. 이처럼 APT 공격은 사이버 보안의 큰 위협이 되는 요소이다.

이러한 APT(Advanced Persistent Threat) 공격에 효과적으로 대응하기 위해서는 공격을 탐지하고 대응할 수 있는 기술뿐만 아니라 대상 시스템의 취약점을 사전에 점검할 수 있는 도구가 필요하다. APT 시뮬레이터는 대상 시스템에 사전 취약점을 점검하고 보안을 강화한다 [2]. 이러한 APT 시뮬레이터는 조직 및 기관에 보안 훈련 도구로써 사용되기도 한다. 대표적인 제품으로 MITRE에서 만든 CALDERA 제품이 있다 [3].

그러나 현재까지의 APT 시뮬레이터는 단일 시스템을 중심으로 연구되었으며, 다중 에이전트와 협력 및 동시성을 가진 복잡한 다중 시스템 취약점 점검에는 한계가 있다.

본 논문에서는 향후 멀티 에이전트 기반의 연쇄적이고 동시적인 APT 시뮬레이터 구현 연구에 도움이 되고자 기존의 APT 시뮬레이터에 비해 가볍고 성능 오버헤드가 적은 새로운 APT 시뮬레이터 프레임워크를 제안한다.

II. 관련 연구

· **Caldera** : MITRE에서 제작 및 배포한 Caldera는 APT 공격에 사용되는 기술들을 적

†교신저자: 박기웅 (세종대학교 정보보호학과 교수)

본 논문은 과학기술정보통신부의 재원으로 정보통신기획평가원(IITP)의 정보보호핵심원천기술개발(Project No. RS-2024-00438551, 50%), 국방ICT융합연구(Project No. 2022-11220701, 20%), 한국연구재단(NRF) 중견후속연구사업(Project No. RS-2023-00208460, 30%)의 지원을 받아 수행된 연구임.

용하여 대상 시스템에서 시뮬레이션하여 취약점을 찾아낸다.

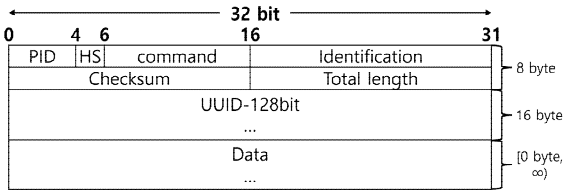
· **Atomic Red Team** : Caldera를 비롯한 다른 APT 시뮬레이터에서 사용할 수 있는 테스트 라이브러리로, 실제 APT 공격에 사용된 전략들이 모여있다 [4].

다중 에이전트 시스템에서는 일반적으로 통신 오버헤드를 줄이고 확장성을 확보하기 위해 필수 정보만을 담아 통신한다 [5]. 기존 APT 시뮬레이터에서는 이러한 통신 최적화에 관한 연구들이 부족하다.

III. 효율적인 APT 프레임워크 제안

에이전트와 C2 서버 간의 통신에서는 JSON, YAML, Google Protocol Buffer 등 다양한 데이터 표준 포맷을 사용할 수 있다. 이러한 데이터 표준 포맷은 바이트(Byte) 수준의 통신이 이루어지며, 불필요한 메타 데이터 등이 추가되어 트래픽을 증가시킨다 [6].

본 논문에서는 기존 에이전트에서 사용될 수 있는 데이터 표준 포맷 대신에 비트(Bit) 수준의 새로운 프로토콜을 제안한다. [그림 1]은 제안된 프로토콜의 내부 구조를 보여주며, 자주 사용되는 필드 위주로 헤더 정보를 구성한다.

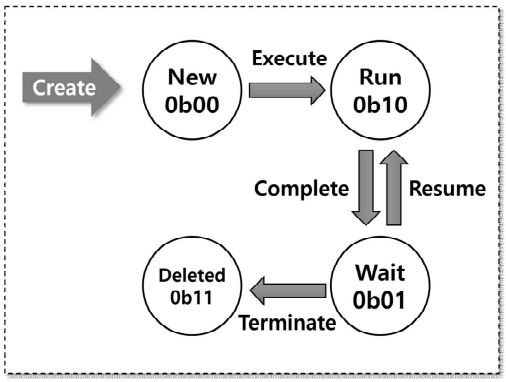


(그림 1) 제안된 프로토콜 구조

에이전트와 C2 서버 간의 통신을 위해 사용되는 프로토콜로 HTTP, HTTPS, TCP, DNS 등이 존재한다. 이 중 HTTPS 프로토콜은 암호화된 방식으로 통신 내용을 감추어 보안 솔루션이 탐지를 어렵게 만든다.

A. Dabit et al. [7] 연구에 따르면 에이전트는 Discord, Twitter와 같은 소셜 미디어 플랫폼으로도 C2 서버와 통신을 할 수 있다. Protocol ID (PID) 필드에서는 에이전트가 사용하는 프로토콜 또는 통신 매개체 등을 나타낸다.

Health Status(HS) 필드는 에이전트의 상태를 나태는 값으로 [그림 2]는 에이전트의 상태 네 단계를 나타낸다. 해당 필드는 에이전트 간의 협력 시 서로의 상태를 확인하고 동작할 수 있도록 정보를 제공한다. 처음 에이전트가 시작되면 New(0b00) 상태이다. 에이전트가 명령을 수행할 때는 Run(0b10) 상태로 변화며 명령이 수행한 후에는 Wait(0b01) 상태가 된다. 에이전트가 더 이상 동작하지 않을 때는 Deleted(0b11) 상태를 된다.



(그림 2) Agent Status 4단계

Command 필드에서는 C2 서버와 에이전트 간의 상호 운용성을 위해 사용된다. 명령을 지시하거나 받을 수 있으며, 에이전트 간 협력 통신에도 사용할 수 있다.

Identification 필드에서는 패킷의 순서를 보장한다. 일부 프로토콜 통신에는 데이터의 크기를 제한하고 있다. 제한된 크기 이상의 데이터는 동일한 순서 번호로 패킷을 나누어 전송한다. 엔드 포인트에서 여러 패킷을 하나로 합쳐서 사용한다.

Checksum 필드는 데이터의 무결성을 검증한다. Total Length는 헤더 크기를 포함한 전체 패킷의 크기를 의미한다. UUID 필드는 에이전트를 구분하는 128비트 식별자로 사용한다. Data 필드에서는 Command 필드를 해석하고 결과를 처리하기 위한 데이터를 포함한다.

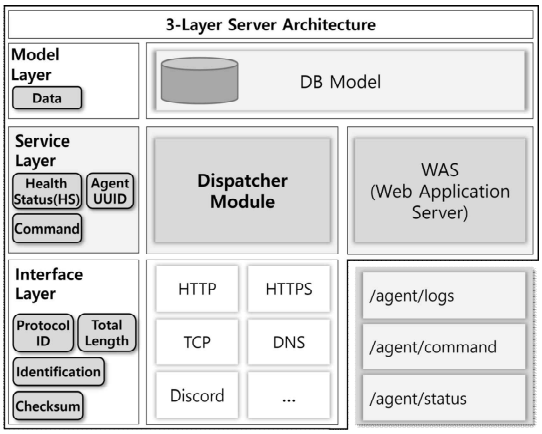
[그림 3]은 C2 서버의 전체적인 아키텍처를 나타낸다. 각 레이어(layer)에서 사용되는 필드에 따라 레이어를 구분한 형태이다.

먼저 Model 레이어는 순수하게 데이터의 처리만을 수행한다. 내부적으로 DB를 가지고 있으며, 요청에 따라 데이터를 저장하고 반환한다.

Service 레이어에서는 에이전트의 UUID 값을 기준으로 에이전트의 상태(HS)를 확인하고 에이전트에 명령을 내릴 수 있다. C2 서버와 에이전트 간의 의미론적 상호 작용을 위해서 Command 필드를 사용한다.

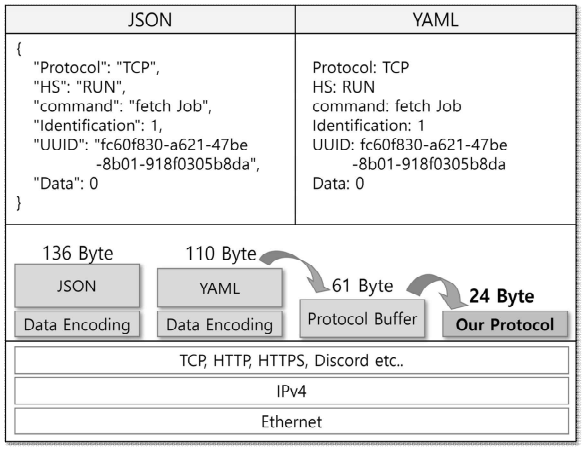
Interface 레이어는 어떤 프로토콜(Protocol)로든 들어오든 데이터 통신이 보장되도록 처리한다. Protocol ID (PID)를 식별하고 Checksum으로 데이터의 무결성을 검증한다. 데이터가 여러 패킷으로 나누어진 경우에는 동일한 Identification 필드 값을 기준으로 데이터를 하나로 합쳐서 처리한다. TotalLength 필드로 헤더와 데이터 영역을 분리하고 데이터를 가공한다.

C2 서버는 다른 프레임워크와의 상호 운용성을 확보하기 위해 Service 레이어에 별도로 WAS(Wab Application Server)를 운영한다. 매개변수로 에이전트의 UUID 값을 식별하고 어떤 URL 경로(Path)로 들어왔는지에 따라 명령을 수행하고 결과를 반환한다.



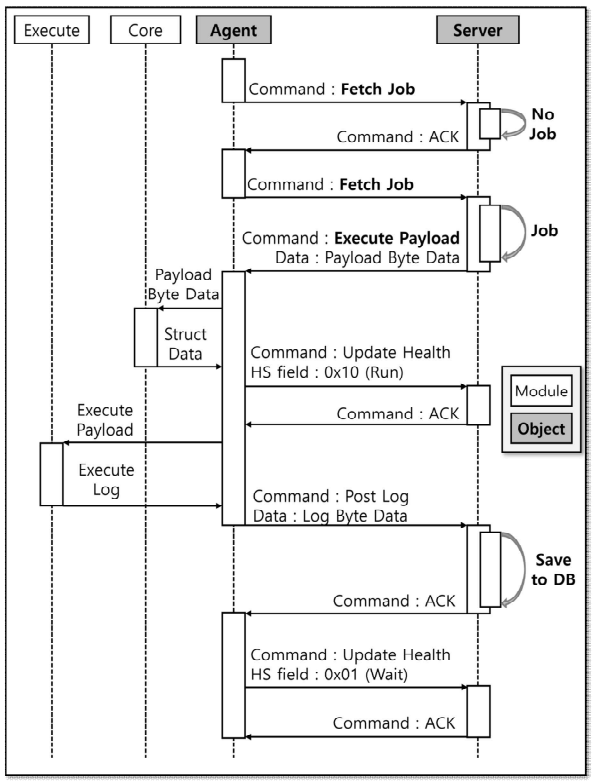
(그림 3) C2 Server 3-Layer Architecture

[그림 4]는 제안된 프로토콜이 다른 프로토콜과 비교했을 데이터 전송 효율을 보여주는 예이다. JSON과 YAML 같은 데이터 표준 포맷은 비효율적인 문자열 기반의 메타 데이터가 들어간다. 이와 달리 Google Protocol Buffer 문자열을 사용하지 않아 더욱 효율적이나 바이트(Byte) 수준의 데이터 표준 포맷의 한계를 벗어나지 못한다. 우리가 제안한 프로토콜은 에이전트에 필요한 필수 요소만으로 데이터 교환을 효율적으로 할 수 있도록 설계되어 더욱 낮은 데이터 크기로 통신할 수 있다.



(그림 4) 다른 데이터 표준 포맷과 비교했을 때 제안된 프로토콜의 우수성

IV. APT 프레임워크 : 에이전트 및 C2 Server 통신 과정



(그림 5) Agent Communication with C2 Server

[그림 5]는 에이전트가 C2 서버로부터 공격 지시를 받고, 공격을 수행한 후 결과를 다시 C2 서버에 보내는 과정을 보여주는 시퀀스 다이어그램이다.

에이전트는 주기적으로 C2 서버로부터의 명

령 지시 여부를 확인하기 위해 Fetch Job 요청을 보낸다. 명령이 없다면 ACK 응답을 받지만, 명령이 있으면 Execute Payload 응답을 받아 Data 필드에 실행해야 할 데이터(Payload)를 얻는다. 공격을 수행한 후 결과 로그를 Data 필드에 담아 Post Log 요청으로 C2 서버에 전송한다. 명령을 수행하는 동안 에이전트는 C2 서버에 상태 정보를 Health Status(HS) 필드에 담아 Update Health 요청을 보낸다.

V. 제안된 프레임워크의 한계점

본 연구에서는 제안된 프로토콜이 TCP 또는 HTTP와 같이 암호화되지 않는 프로토콜을 사용하면 스누핑(Snooping) 또는 스푸핑(Spoofing)과 같은 보안 취약점이 존재할 수 있다. APT 시뮬레이터를 포함한 에이전트를 사용하는 보안 솔루션에서는 에이전트의 통신 패킷을 가로채고 정보를 도청하거나 에이전트를 조작하기 위한 스푸핑 공격 등에 취약할 수 있다[8]. 본 논문에서는 이러한 보안 취약점을 대비할 수 있는 구체적인 기술은 구현되지 않았다. 향후 연구에서 드론 패킷에서 흔히 사용되는 암호화 방식이나, 패킷의 끝에 ECC(elliptic curve cryptography) 필드를 추가하여 데이터를 암호화하는 등의 방법으로 데이터의 보안을 강화할 수 있을 것이다.

VI. 결론

본 논문에서는 기존 데이터 표준 포맷의 데이터 교환보다는 가볍고 성능 오버헤드가 적은 새로운 형태의 APT 시뮬레이터 프레임워크를 제안한다. 이 구조는 에이전트와 C2 서버 간의 효율적인 통신을 가능하게 한다. 하지만, 기존의 Master-Slave 관계를 완전히 벗어나지 못한다.

향후 연구에서 다수의 시스템에 대한 연쇄적이고 동시적인 APT 시뮬레이터를 구현하기 위하여 에이전트 간의 협력 통신을 할 수 있는 프레임워크가 제안될 수 있어야 한다. 제안된 APT 시뮬레이터가 향후 연구에 소중한 자료로 활용될 수 있기를 기대한다.

[참고문헌]

[1] 장성우, 인용준, APT(Advanced Persistent Threat) 공격 시나리오 검증 시스템 연구, 한국산학기술학회, Apr, 2023.

[2] A. Dabit, O. A. AI-Haija, M. AI-Fayoumi, Identifying Weaknesses: A Guide to Conducting an Effective Network Vulnerability Assessment, International Arab Conference on Information Technology (ACIT), Dec, 2023.

[3] N. Mohamed, Study of bypassing Microsoft Windows Security using the MITRE CALDERA Framework, Sep, 2022.

[4] M. Okuma et al., Automated Mapping Method for Sysmon Logs to ATT&CK Techniques by Leveraging Atomic Red Team, International Conference on Signal Processing and Information Security (ICSPIS), pp.104-109, Nov, 2023.

[5] A. Dorri, S. S. Kanhere and R. Jurdak, Multi-Agent Systems: A Survey, IEEE Access, vol.6, pp.28573-28593, Apr, 2018.

[6] D. C. Castillo, J. Rosales and G.-A. T. Balnco, Optimizing Binary Serialization with an Independent Data Definition Format, International Journal of Computer Applications, vol.180, no.28, Mar, 2018.

[7] A. Sidhardhan, S. Keerthana and J.-M. Kannimoola, Weaponizing Real-world Applications as C2 (Command and Control), International Conference on Innovative Data Communication Technologies and Application (ICIDCA), pp.458-463, Mar, 2023.

[8] M. Suliman and B. Alluhaybi, Protecting Mobile Agent against Man-In-TheMiddle Attack: The Dummy Agent Model, The International Journal of Advanced Networking and Applications (IJANA), vol.13, no.04, pp.5024-5028, Mar, 2022.