PLAINBIT

사이버보안전문단 6기 프로젝트

Cobalt Strike Out,
호스트 기반의 Cobalt Strike 위협 탐지 방안 연구

2024.10

김진국, 장원희, 김서준, 김예지

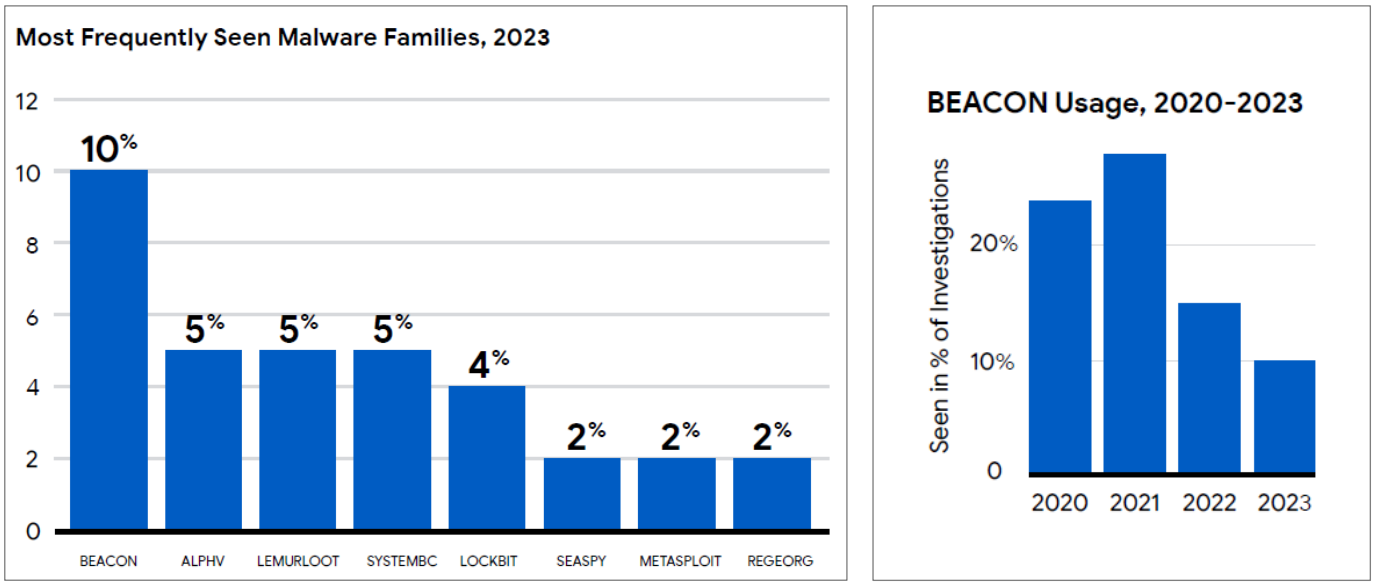
※ 해당 보고서는 사이버보안전문단 프로젝트 결과물로 작성되었습니다.

목차

1. 개 요

사이버 보안 환경은 날이 갈수록 더욱 정교해지는 위협 행위자들의 공격 기법으로 인해 고도화된 위협에 직면하고 있다. Cobalt Strike는 명령 및 제어(Command and Control, 이하 C2) 통신을 통해 감염된 네트워크에서 공격자가 자유롭게 활동할 수 있도록 하는 기능을 지원하고 있으며, 다양한 공격 기법을 제공해 공격자들이 탐지를 회피하고 공격을 장기화하는 데 유리한 환경을 제공한다. Cobalt Strike를 공격자들이 선호하는 이유는, 기존의 공격 도구와 달리 공격 환경을 유지하거나 관리하는 데 있어 별도의 업데이트나 관리 작업 필요 없이 손쉽게 사용할 수 있기 때문이다. 이러한 편의성으로 인해 Cobalt Strike는 여전히 다양한 사이버 공격에서 널리 악용되는 대표적인 도구로 자리 잡고 있다.

보안 업체 Mandiant 보고서[1]에 따르면, Cobalt Strike의 페이로드인 Beacon이 전 세계 Mandiant 사고 조사에서 가장 많이 식별된 악성코드 제품군으로, 전체 침입 사례의 10%에서 식별되었다고 한다. 특히, 2020년에 Cobalt Strike 크랙 버전이 공개된 이후 2021년까지 공격에 활발하게 활용되었으나, 2022년부터는 사용률이 다소 감소하는 양상이 보였다. 이는 다수의 보안 업체에서 Cobalt Strike를 탐지 및 대응하고 있는 결과로 보이지만, 여전히 주요 사이버 공격에서 자주 사용되는 공격 도구로 자리매김하고 있는 것을 알 수 있다. 이와 같은 사실은 Cobalt Strike가 공격자들이 정교한 공격을 계획하고 실행하는 데 있어 얼마나 중요한 역할을 하는지를 잘 보여준다.



[그림 1] 2023년에 가장 많이 식별된 악성코드 계열 순위 (Mandiant, M-Trends 2024 Special Report)

Cobalt Strike와 같이 탐지 회피 능력이 뛰어난 도구는 보안 전문가나 사고 대응 전문가들에게 새로운 도전 과제를 안겨주고 있다. 공격자들이 Cobalt Strike를 활용한 공격을 더 신속하게 탐지하고 방어하기 위해서는 관련 위협을 탐지할 수 있는 방안에 대한 연구가 더욱 필요하다. 현재 많은 기업들이 Cobalt Strike 기반의 공격에 노출되고 있지만, 이에 대한 탐지와 대응 역량이 충분하지 않아 사이버 보안 환경에서 효과적인 탐지 및 분석 방안을 마련하는 것이 매우 시급한 상황이다.

[1] services.google.com/fh/files/misc/m-trends-2024.pdf

본 연구는 Cobalt Strike를 이용한 공격이 시스템에 남기는 다양한 아티팩트를 체계적으로 분석해, 누구나 공격 흔적을 보다 신속하고 정확하게 탐지하고 대응할 수 있도록 실용적이고 구체적인 탐지 방안을 마련하는 것을 목표로 한다. 이 연구 결과는 Cobalt Strike를 악용한 위협을 모니터링하고 대응하기 위한 핵심적인 기술적 기반을 제공하는데 기여할 것이다. 이로 인한 기대 효과는 다음과 같다.

첫째, Cobalt Strike에 대한 예방적 대응 전략 강화

Cobalt Strike와 같은 공격 도구를 탐지하려면 전문적인 지식과 경험이 요구되며, 많은 기업들이 보안 솔루션을 통해 이를 탐지하고 대응하고 있다. 그러나, 대부분의 보안 솔루션이 사고 발생 후 대응에 초점을 맞추고 있는 반면, 본 연구는 침해사고의 초기 단계에서 악성 행위를 조기 탐지할 수 있는 지표를 제공하는 데 중점을 두고 있다. 이를 통해 Cobalt Strike 기반 위협을 더 정밀하게 식별하고, 로깅 설정 등 탐지 강화 방안 마련에 기여할 것이다. 연구 결과는 공격자가 공격 목적을 달성하기 전에 공격을 무력화할 수 있도록 도와줄 것이며, 향후 유사한 공격 도구에 대한 대응 전략 수립에도 중요한 기초 자료로 활용될 것이다.

둘째, 중소기업의 보안 능력 향상을 위한 접근성 제공

많은 중소기업에서는 제한된 보안 예산과 리소스로 인해 고도화된 보안 솔루션을 도입하기 어려운 실정이다. 본 연구에서 제안하는 Sysmon Configure Rule과 한국인터넷진흥원의 해킹진단도구 탐지 룰은 누구나 무료로 사용할 수 있어, 중소기업도 경제적 부담 없이 위협을 탐지하고 모니터링하는 데 활용할 수 있다. 연구를 통해 작성된 탐지 룰은 조직에서 쉽게 적용할 수 있도록 설계되어, 보안 운영의 진입 장벽을 낮추고 보다 효율적인 보안 모니터링과 침해사고 대응을 가능하게 할 것이다.

셋째, 사고 분석 속도 및 정밀도 향상

침해사고 발생 시 가장 중요한 것은 공격자가 목적을 달성하기 전에 신속하게 원인을 규명하고, 내부 이동 경로와 전파 방식을 파악해 공격을 무력화하는 대응을 빠르게 수행하는 것이다. 본 연구는 Cobalt Strike를 통한 공격이 시스템에 남기는 다양한 아티팩트를 체계적으로 분석함으로써 침해사고의 원인과 범위를 더 신속하게 파악할 수 있도록 돕는다. 또한, 제공된 Sysmon Configure Rule을 통해 수집된 정보는 정밀한 사고 분석을 가능하게 해 대응 시간을 크게 단축시킬 수 있다. 이를 통해 Cobalt Strike를 이용한 침해사고의 피해를 최소화하고, 추가적인 확산을 방지하는 데 중요한 역할을 할 것이다.

2. Cobalt Strike란?

Cobalt Strike는 위협과 관련된 보안 테스트를 위해 개발된 상용 레드팀 명령제어 프레임워크로, 다양한 공격 전술에 활용할 수 있는 명령과 기능을 지원한다. 이를 통해 공격자 행위를 시뮬레이션하거나 레드팀 훈련, 보안 평가 등의 목적으로 사용된다. Cobalt Strike는 레드팀과 실제 공격자 모두에게 인기 있는 프레임워크로, 레드팀은 대상 환경의 보안을 평가하거나 Cobalt Strike를 활용한 위협을 탐지하고 대응하는 역량을 강화하는 데 사용한다. 반면, 공격자들은 Cobalt Strike가 제공하는 다양한 공격 모듈을 활용해 네트워크 탐색, 권한 상승, 정보 수집 등의 공격 활동을 수행하고 있다.

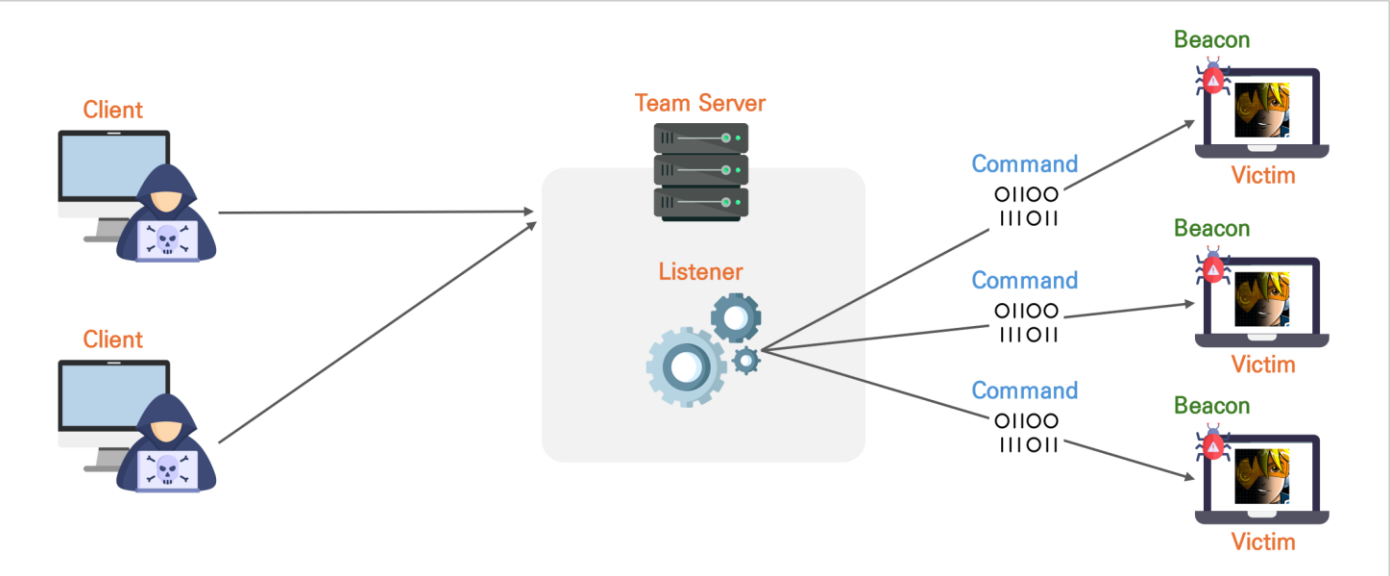
Cobalt Strike는 침투, 탐지 회피, 명령 제어, 커뮤니티 지원 등 사이버 공격에 필요한 핵심 기능들을 제공하기 때문에 공격에 많이 활용되고 있다. 공격자에게 Cobalt Strike를 매력적으로 여기는 이유는 다음과 같이 정리할 수 있다.

[표 1] 공격자가 Cobalt Strike를 공격에 많이 활용하는 이유

번호	구분	내용
1	명령 및 제어(C2) 기능	공격자가 피해 시스템에 침투한 후 원격으로 명령을 전달하고 제어할 수 있는 기능을 지원하며, 이를 통해 시스템을 조작하거나 추가 악성코드를 실행할 수 있음
2	정교한 침투 기능	대부분의 기능이 메모리에서 실행되기 때문에 파일을 드롭(Drop)하지 않고도 시스템에 접근 가능함 (파일리스(Fileless) 공격이 가능하기 때문에 탐지 회피에 유리)
3	다양한 공격 모듈	Beacon이라는 악성 페이로드를 통해 공격자와 시스템 간의 통신이 수행되며, HTTP(S), DNS 등 다양한 프로토콜을 사용할 수 있기 때문에 네트워크 트래픽을 숨기거나 정상 통신을 위장할 수 있음 (공격에 대한 통신 트래픽 탐지 회피에 유리)
4	공격 우회 기능	다양한 보안 솔루션을 우회할 수 있는 기능을 제공하며, 메모리 상에서 동작하고 침투 방식이 정교하기 때문에 백신이나 EDR 솔루션에서 탐지되지 않는 경우 존재함
5	커뮤니티 및 공격 자동화	잘 알려진 공격 프레임워크로, 관련된 많은 정보가 인터넷에 공개되어 있기 때문에 공격자는 이를 활용해 쉽게 공격을 자동화하거나 공개된 커뮤니티에서 다양한 플러그인을 쉽게 다운로드 받아 활용할 수 있음 (원하는 공격 방식을 하나의 프레임워크에 구성 가능)
6	도구 위장	Cobalt Strike가 공격에도 많이 활용되지만, 보안 전문가가 보안 평가를 위해 활용한다는 점을 악용해 실제 악성코드로 인식하지 않도록 속이고 공격자의 행위와 레드팀의 행위 구분에 혼란을 줄 수 있음
7	맞춤화된 공격에 활용	Cobalt Strike는 사용자 정의가 가능하기 때문에 특정 환경에 맞춘 공격을 설계할 수 있음 Beacon의 통신 방식, 암호화, 페이로드의 동작 등을 세밀하게 조정해 공격 환경에 맞춘 공격 캠페인 운영 가능

2.1. 구성 요소

Cobalt Strike는 Team Server, Client, Listener, Beacon으로 구성된다. 공격자는 Cobalt Strike Client를 활용해 Team Server에 연결한 후, Team Server에서 내린 명령을 Listener를 통해 Beacon으로 전달하고 전달받은 명령을 Beacon이 실행해 결과 값을 Listener에게 전달한다.



[그림 2] Cobalt Strike 동작 구성

[표 2] Cobalt Strike 구성 요소 목록

번호	구분	설명
1	Team Server	- 실질적인 명령 제어 서버로, Listener와 Beacon을 관리하고 공격자가 내리는 명령을 전달 및 실행 - 기본적으로 TCP 50050 포트를 사용해 Client와 통신
2	Client	- Team Server에 연결해 직접 명령을 제어하는 인터페이스 역할
3	Listener	- Beacon 과의 연결을 유지하고 Team Server가 Beacon에 명령을 전달할 수 있도록 돕는 역할 - Beacon이 사용할 통신 경로 설정 (8개의 다양한 통신 프로토콜 지원)
4	Beacon	- Victim에 설치된 Cobalt Strike의 악성 페이로드 (백도어로 동작하는 실질적인 악성코드) - Listener를 통해 전달받은 명령을 실행 후, 결과 값을 Listener로 전달 - 다양한 형태의 Beacon 제공 (EXE/POWERSHELL/DLL, Stager/Stageless,)

[표 3] Listener가 지원하는 통신 프로토콜 목록

번호	구분	설명
1	Beacon DNS	- DNS 요청을 사용해 통신 - 데이터 채널을 DNS TXT, DNS A, DNS AAAA로 유연하게 변경 가능 (기본 값: DNS TXT) - 'checkin' 명령을 통해 체크인하도록 요청해야 작업 수행 가능 (미 요청 시 연결하지 않음)
2	Beacon HTTP(S)	- HTTP GET 요청으로 Team Server 서버로부터 명령을 받음 - HTTP POST 요청으로 수행된 명령 결과에 대한 데이터를 Team Server에 전송

번호	구분	설명
3	Beacon SMB	- 명명된 파이프를 사용해 통신하며, 주로 Peer-to-Peer 통신에 사용
4	Beacon TCP	- TCP 소켓을 사용해 통신하며, 주로 Peer-to-Peer 통신에 사용
5	External C2	- 타 사의 프로그램이 Beacon의 Listener로 동작할 수 있도록 설정
6	Foreign HTTP(S)	- Metasploit 프레임워크 또는 Cobalt Strike의 다른 인스턴스에 세션을 전달할 수 있도록 설정

Cobalt Strike의 Team Server와 Client는 모두 다음의 항목이 충족되어야 한다.

[표 4] Team Server와 Client 시스템 요구사항 목록

번호	요구사항	구분	설명
1	Java	Team Server, Client	- Oracle Java 11, OpenJDK 11 중 1개 설치
2	OS	Team Server	- Java 요구사항을 충족하는 Linux 운영체제 (Debian, Ubuntu, Kali Linux)
		Client	- Windows 7 이상, MacOS X 10.13 이상GUI 기반의 Linux 운영체제
3	Hardware (최소요구)	Team Server	- 2GHz+ 프로세서 이상, 2GB RAM 이상, 500MB 이상의 사용 가능한 디스크 공간

2.2. Cobalt Strike 특징

Cobalt Strike의 특징은 크게 6개로 나눌 수 있으며, 다음과 같다.

2.2.1. 유연성 (Flexibility)

Cobalt Stike의 개발 철학의 원칙 중 하나가 운영자가 필요한 방식으로 제품을 사용할 수 있도록 관련 기능을 제품에 지속적으로 추가하는 것이며, 이로 인해 사용자에게 가장 많은 유연성을 제공할 수 있도록 설계되었다. 사용자는 소프트웨어에 내장된 기본 모듈에 제약 받지 않고 커스텀한 도구와 기술을 통합해 프레임워크를 쉽게 확장할 수 있으며, 도구와 스크립트는 중앙 저장소인 Community Kit[2]에서 다운로드 받아 사용할 수 있다. 현재 Cobalt Strike에서 유연성 제공을 위해 활용할 수 있는 기능은 다음과 같다.

[표 5] Cobalt Strike의 유연성을 위한 기능 목록

번호	구분	설명
1	Arsenal Kit	Cobalt Strike의 기본 동작을 변경할 수 있도록 지원하는 기능으로, 사용자는 키트를 그대로 빌드해 사용하거나 공격에 맞게 수정 가능 (Sleep Mask Kit 및 User Defined Reflective Loader와 같은 도구를 활용해 소프트웨어 동작 방식 변경 가능)
2	Community Kit	사용자 커뮤니티에서 Cobalt Strike의 기능을 확장하기 위해 작성한 중앙 저장소이며, 클라이언트 UI 기능을 추가하는 Aggressor Script부터 광범위한 작업을 수행할 수 있는 Beacon Object Files[3](이하, BOF)까지 사용 가능
3	Malleable C2	사용자가 Beacon의 HTTP 지표를 변경할 수 있는 방법으로, Beacon의 네트워크 트래픽 지표를 제어하는데 사용되며 일반적으로 Beacon의 트래픽이 대상에 맞게 포함되도록 하거나 APT를 에뮬레이트하는데 사용됨

2.2.2. 상호 운용성 (Interoperability)

Cobalt Strike는 Fortra 社의 다른 제품과 함께 사용이 가능하다. 예를 들어, Cobalt Strike와 Fortra 社의 모의 침투 테스트 도구인 'Core Impact'와 함께 사용해 리소스를 공유하고 세션 터널링 기능을 위해 Beacon 배포가 가능하다.

2.2.3. 은닉 통신 (Covert Communication)

Beacon은 탐지 회피를 위해 다양한 통신 채널을 제공하고, Malleable C2 프로필을 생성하면 네트워크 지표를 변경해 Beacon 활동을 은닉하거나 APT를 시뮬레이션 할 수 있다. 또한, Peer-to-peer Beacon 연결은 TCP를 통해 설정되거나 SMB를 사용하는 명명된 파이프를 통해 설정할 수 있다.

[2] cobalt-strike.github.io/community_kit/
[3] Beacon Object Files는 Beacon 프로세스 내에서 실행할 수 있는 컴파일된 C 프로그램이다.

2.2.4. 침투 후 공격 (Post-exploitation)

Cobalt Strike의 시그니처 페이로드인 Beacon의 통신 유형은 비동기 통신과 대화형 모드를 지원하며, 배포되면 정보를 수집, 임의의 명령 실행, 추가 페이로드를 배포하는 등의 작업을 수행할 수 있다. 또한, BOF(Beacon Object Files)를 사용해 익스플로잇 기능을 추가할 수 있다.

[표 6] Beacon 통신 유형

번호	구분	설명
1	비동기 통신 (Asynchronous Communication)	- Beacon의 명령은 대기열에 저장되며, 체크인할 때 실행됨 - Beacon의 체크인 빈도는 'sleep' 명령을 통해 설정 가능 - 통신 속도가 느리기 때문에 은밀성이 필요한 작업에 이상적임
2	대화형 모드 (Interactive Mode)	- 초당 여러 번 팀 서버에 체크인해 모든 명령 즉시 실행 - 비동기 통신보다 은밀하지는 않지만, SOCKS 프록시와 같이 즉각적인 제어가 필요한 작업에 이상적임

2.2.5. 협업 (Collaboration)

Team Server를 통해 실시간으로 소통하고 피해 시스템을 제어할 수 있는 기능으로, 세션을 공유하는 것 외에도 호스트를 공유하거나 파일을 다운로드할 수 있다.

2.2.6. 보고 (Reporting)

Cobalt Strike 활용 타임라인과 지표 목록을 제공하는 기능으로, PDF와 Word 문서로 내보내기 할 수 있다.

[표 7] Cobalt Strike에서 제공하는 보고서 목록

번호	구분	설명
1	활동 보고서 (Activity Report)	- Cobalt Strike를 통한 공격 타임라인
2	호스트 보고서 (Hosts Report)	- Cobalt Strike가 호스트 별로 수집한 정보 요약 (서비스, 자격 증명 및 세션 등)
3	침해 지표 (Indicators of Compromise)	- 위협 인텔리전스 보고서의 침해지표 부록과 유사하게 기록 - Malleable C2 프로필, 사용 도메인, 업로드 파일의 MD5 해시 분석 포함
4	세션 보고서 (Sessions Report)	- 세션 별로 지표와 활동을 문서화 - 각 세션에서 사용된 통신 경로, 세션이 연결될 동안 디스크에 저장된 파일의 MD5 해시, 익스플로잇 후 활동 타임라인 등 포함
5	사회 공학 (Social Engineering)	- 스피어 피싱 이메일의 각 라운드, 클릭한 사용자와 해당 사용자로부터 수집된 내용을 문서화
6	전술, 기술 및 절차 (Tactics, Techniques, and Procedure)	- Cobalt Strike 활동을 MITRE ATT&CK 내 전술에 매핑

2.3. 기능 구성

Cobalt Strike는 기본적으로 다양한 기능을 제공할 뿐만 아니라, 사용자가 필요에 따라 프레임워크를 맞춤형으로 구성할 수 있도록 추가적인 기능도 지원한다. Cobalt Strike는 기본 제공 모듈과 확장 모듈로 구성되며, 각 구성 요소는 다음과 같다.

2.3.1. 기본 지원 기능

기본 지원 기능은 Cobalt Strike 설치 시 함께 제공되는 필수 기능들로, 침투 테스트 및 전반적인 공격 단계에서 핵심 작업을 수행하는 데 필요한 기능들로 구성되어 있다. Cobalt Strike 4.9.1 기준으로 총 98개의 기능이 제공된다.

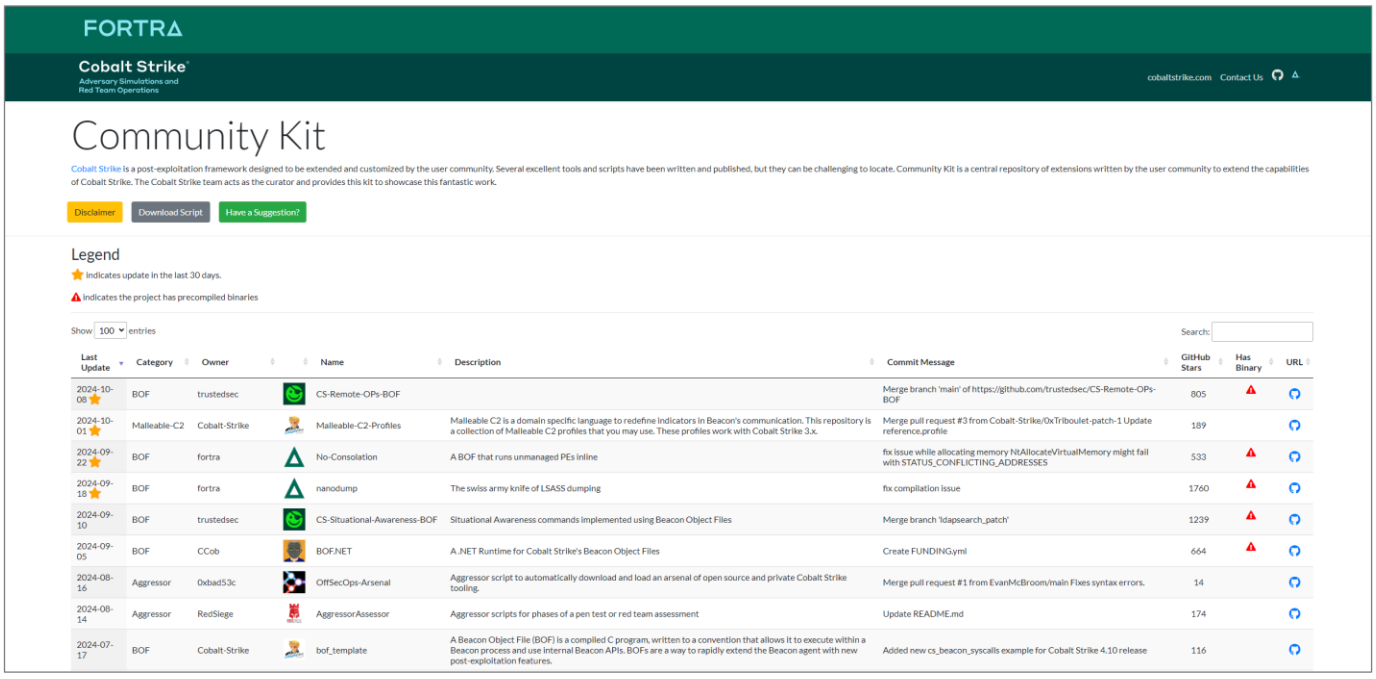
```
beacon> help

Beacon Commands
===============

    Command                 Description
    -------                 -----------
    !                       Run a command from the history
    argue                   Spoof arguments for matching processes
    blockdlls               Block non-Microsoft DLLs in child processes
    browserpivot            Setup a browser pivot session
    cancel                  Cancel a download that's in-progress
    cd                      Change directory
    checkin                 Call home and post data
    chromedump              Recover credentials from Google Chrome
    clear                   Clear beacon queue
    clipboard               Attempt to get text clipboard contents
    connect                 Connect to a Beacon peer over TCP
    covertvpn               Deploy Covert VPN client
    cp                      Copy a file
    data-store              Store post-ex items to Beacon
    dcsync                  Extract a password hash from a DC
    desktop                 View and interact with target's desktop
    dllinject               Inject a Reflective DLL into a process
    dllload                 Load DLL into a process with LoadLibrary()
    download                Download a file
    downloads               Lists file downloads in progress
    drives                  List drives on target
    elevate                 Spawn a session in an elevated context
    execute                 Execute a program on target (no output)
    execute-assembly        Execute a local .NET program in-memory on target
    exit                    Terminate the beacon session
    file_browser            Open the file browser tab for this beacon
    getprivs                Enable system privileges on current token
    getsystem               Attempt to get SYSTEM
    getuid                  Get User ID
    hashdump                Dump password hashes
    help                    Help menu
    history                 Show the command history
    inject                  Spawn a session in a specific process
    inline-execute          Run a Beacon Object File in this session
    jobkill                 Kill a long-running post-exploitation task
    jobs                    List long-running post-exploitation tasks
    jump                    Spawn a session on a remote host
    kerberos_ccache_use     Apply kerberos ticket from cache to this session
    kerberos_ticket_purge   Purge kerberos tickets from this session
    kerberos_ticket_use     Apply kerberos ticket to this session
    keylogger               Start a keystroke logger
    kill                    Kill a process
    link                    Connect to a Beacon peer over a named pipe
    logonpasswords          Dump credentials and hashes with mimikatz
    ls                      List files
    make_token              Create a token to pass credentials
    mimikatz                Runs a mimikatz command
    mkdir                   Make a directory
    mode dns                Use DNS A as data channel (DNS beacon only)
    mode dns-txt            Use DNS TXT as data channel (DNS beacon only)
    mode dns6               Use DNS AAAA as data channel (DNS beacon only)
    mv                      Move a file
    net                     Network and host enumeration tool
    note                    Assign a note to this Beacon
    portscan                Scan a network for open services
    powerpick               Execute a command via Unmanaged PowerShell
    powershell              Execute a command via powershell.exe
    powershell-import       Import a powershell script
    ppid                    Set parent PID for spawned post-ex jobs
    printscreen             Take a single screenshot via PrintScr method
    process_browser         Open the process browser tab for this beacon
    ps                      Show process list
    psinject                Execute PowerShell command in specific process
    pth                     Pass-the-hash using Mimikatz
    pwd                     Print current directory
    reg                     Query the registry
    remote-exec             Run a command on a remote host
    rev2self                Revert to original token
    rm                      Remove a file or folder
    rportfwd                Setup a reverse port forward
    rportfwd_local          Setup a reverse port forward via Cobalt Strike client
    run                     Execute a program on target (returns output)
    runas                   Execute a program as another user
    runasadmin              Execute a program in an elevated context
    runu                    Execute a program under another PID
    screenshot              Take a single screenshot
    screenwatch             Take periodic screenshots of desktop
    setenv                  Set an environment variable
    shell                   Execute a command via cmd.exe
    shinject                Inject shellcode into a process
    shspawn                 Spawn process and inject shellcode into it
    sleep                   Set beacon sleep time
    socks                   Start/Stop a SOCKS4a/SOCKS5 server to relay traffic
    spawn                   Spawn a session
    spawnas                 Spawn a session as another user
    spawnto                 Set executable to spawn processes into
    spawnu                  Spawn a session under another process
    spunnel                 Spawn and tunnel an agent via rportfwd
    spunnel_local           Spawn and tunnel an agent via Cobalt Strike client rportfw
```

[그림 3] Cobalt Strike 4.9.1 기준 지원 기능 목록

2.3.2. 확장 기능

확장 기능은 기본 지원 모듈 외에 공격 목적에 맞게 프레임워크를 고도화하기 위한 목적으로 활용되는 기능으로, 사용자 정의 스크립트나 플러그인을 활용해 확장할 수 있다. 이러한 확장 기능은 공격의 복잡성을 높이거나 특정 상황에 맞게 활용할 수 있도록 돕는 역할을 하며, 다양한 공격 시나리오에 맞게 유연한 활동을 할 수 있도록 한다.

Aggressor Script는 Cobalt Strike 3.0 버전 이상에서 지원하는 스크립팅 언어로, Cobalt Strike Client를 수정하고 기능을 확장시킬 수 있다. '.cna' 형식의 파일로 구성되어 있으며, Cobalt Strike의 Script Console에서 모듈을 추가할 수 있다. Cobalt Strike에서 지원하는 기본 모듈에서 지속성을 위한 기능이 별도로 존재하지 않아 공격자들은 Beacon이 지속적으로 실행될 수 있게 확장 모듈을 사용한다. 확장 기능은 사용자가 정의할 수 있기 때문에 Github 등 커뮤니티에 자신이 만든 도구나 스크립트를 공유해 활용하고 있으며, Fortra社에서는 이러한 도구와 스크립트를 'Community Kit' 페이지에 중앙으로 저장해 사용자가 편리하게 접근하고 활용할 수 있도록 지원하고 있다.

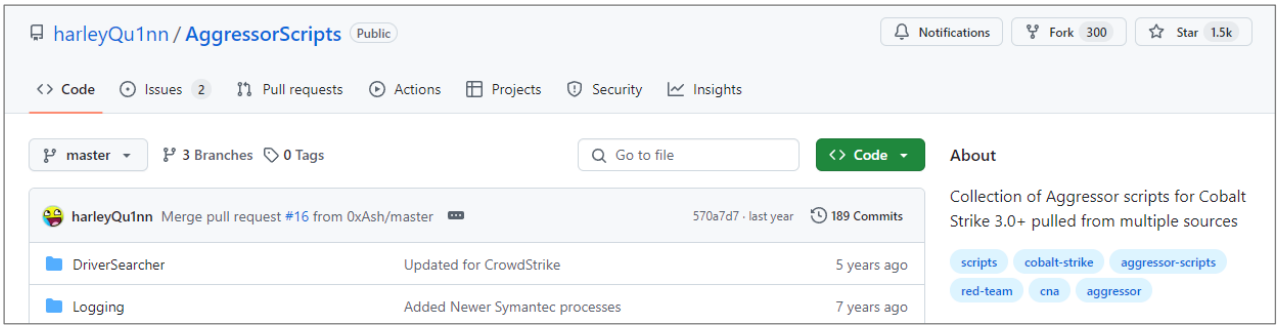


[그림 4] Fortra 社에서 공개한 Cobalt Strike Community Kit (cobalt-strike.github.io/community_kit)

Github에 공개된 대표적인 Cobalt Strike 확장 기능 프로젝트는 다음과 같다.

- **[harleyQu1nn] AggressorScripts**

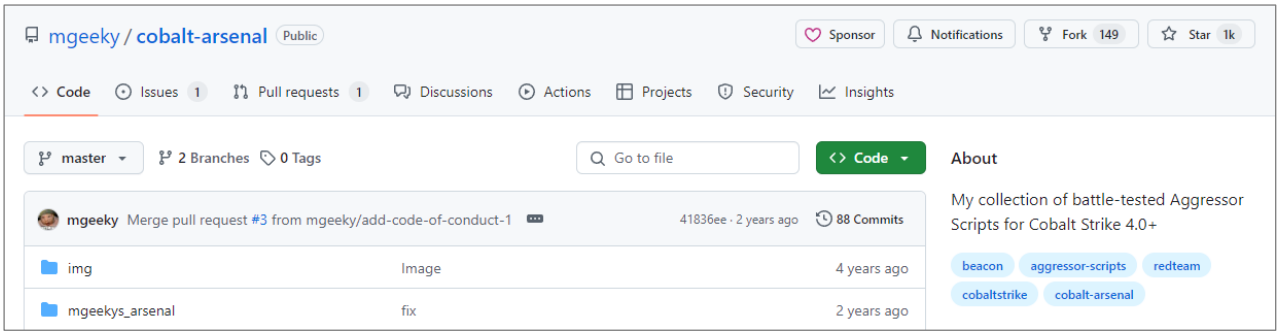
Cobalt Strike 3.0 이상 버전을 위한 스크립트 모음으로, 사용자 편의성을 높이는 기능부터 지속성을 위한 기능까지 다양한 추가 기능을 지원하고 있다. Github에서 1,480개의 Star를 보유하고 있어 가장 잘 알려져 있는 도구이다.



[그림 5] harleyQu1nn의 AggressorScripts 도구 Github 페이지 (github.com/harleyQu1nn/AggressorScripts)

- **[mgeeky] cobalt-arsenal**

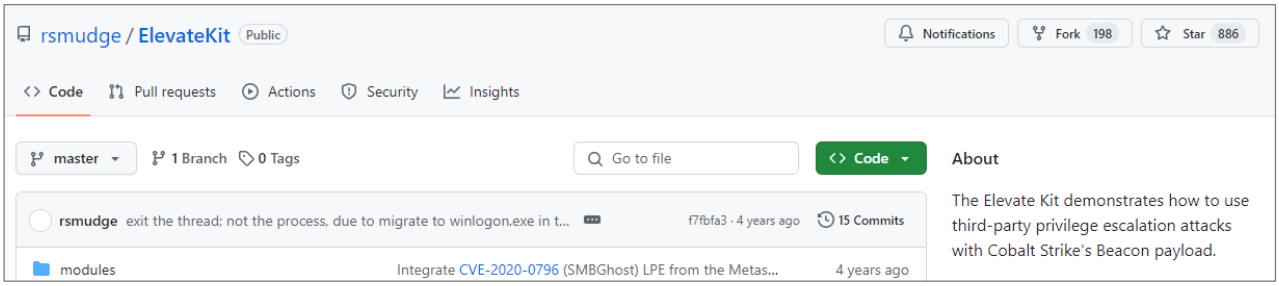
Cobalt Strike 4.0 이상 버전을 위한 스크립트 모음으로, 사용자 편의성보다는 공격에서 활용할 수 있는 기능을 구현해 지원하고 있으며, Github에서 1,030개의 Star를 보유하고 있다.



[그림 6] mgeeky의 cobalt-arsenal 도구 Github 페이지 (github.com/mgeeky/cobalt-arsenal)

- **[rsmudge] ElevateKit**

Cobalt Strike의 Beacon을 사용해 권한 상승 공격을 사용하는 스크립트로 Github에서 885개의 Star를 보유하고 있으나, 2020년 6월 이후 업데이트 되고 있지 않다.



[그림 7] rsmudge의 ElevateKit 도구 Github 페이지 (github.com/rsmudge/ElevateKit)

3. 선행 연구

2020년 11월, Cobalt Strike 4.0의 크랙 버전이 Github에 공개된 이후, 다수의 공격자들이 이를 활용해 내부 시스템을 장악하는 사례가 급증했다. 실제로 2019년과 2020년 사이 Cobalt Strike를 활용한 사고가 161% 증가했으며, 이는 해당 도구가 레드팀의 침투 테스트와 보안 점검 목적으로 주로 사용되던 시기와는 크게 대비되는 변화이다. 과거에는 주로 보안 취약점 점검 용으로 사용되어 탐지의 필요성이 상대적으로 적었지만, 2021년 이후 공격자들이 공개된 버전을 적극적으로 활용하면서 Cobalt Strike에 대한 탐지 및 방어 연구가 본격적으로 활발히 진행되었다. 아래 표는 Cobalt Strike와 관련해 국내/외에서 진행된 주요 연구들을 정리한 것이다.

[표 8] 기존에 진행된 국내/외 Cobalt Strike 연구 목록

구분	공개 유형	공개 연도	연구 제목	저자
국내	논문	2023	명령 제어 프레임워크 (Command and Control Framework) 도구 추적 방안에 대한 연구	권혁주, 곽진
	블로그	2022	침투 테스팅 도구 Cobalt Strike Part 1	이글루 코퍼레이션
국외	블로그	2023	Cobalt Strike Forensic	Balasubramanya C
		2023	Memory Forensic: Hunting Cobalt Strike in Memory	Omar Alanezi
		2020	Detecting Cobalt Strike Default Modules via Named Pipe Analysis	Riccardo Ancarani
		2021, 2022	Cobalt Strike, a Defender's Guide	The DFIR Report
		2022, 2023	Cobalt Strike Investigation	Lexfo
		2021	Hunting and detecting Cobalt Strike	Erwan Chevalier, Narimane Lavay and Sekoia TDR
		2021	Responding to a Cobalt Strike Attack	Invictus Incident Response
		2022	Cobalt Strike Analysis and Tutorial: How Malleable C2 Profiles Make Cobalt Strike Difficult to Detect	UNIT 42
		2021	Detecting CONTI CobaltStrike Lateral Movement Techniques	Cyb3rSn0rlax
	보고서	2021	Full-Spectrum Cobalt Strike Detection	Recorded Future

기존에 진행된 국내/외 Cobalt Strike 연구와 관련해 간략히 요약한 내용은 다음과 같다.

- **[권혁주, 곽진] 명령 제어 프레임워크 (Command and Control Framework) 도구 추적 방안에 대한 연구**[4]

  명령 제어 프레임워크를 사전에 추적할 수 있는 방법론(명령 제어 프레임워크 관련 서버 목록 수집, 단계적 전달 묘사, 봇넷 설정 추출, 인증서 수집 및 특징 추출)을 제안했다. 제안된 방법론을 Cobalt Strike에 적용해 명령 제어 프레임워크로부터 발생할 수 있는 사이버 위협 대응 기반을 마련했다.

- **[이글루 코퍼레이션] 침투 테스팅 도구 Cobalt Strike**[5]

  Cobalt Strike의 개요 및 기능을 설명하고, Sentinel-One에서 배포한 'CobaltStrikeParser'를 이용해 Malleable C2 Profile 분석 과정을 설명했다. 이후, Cobalt Strike를 이용한 사례를 분석해 확보 가능한 샘플을 대상으로 공격자들의 공격 패턴에 대해 설명했다.

- **[Balasubramanya C] Cobalt Strike Forensic**[6]

  Cobalt Strike Forensic을 위한 마인드맵을 소개하면서 Cobalt Strike Forensic을 수행하는 다양한 기술과 도구에 대해 설명하고, Cobalt Strike 활동을 특정하기 위해 찾아야 할 특정 아티팩트들의 예시를 설명했다.

- **[Omar Alanezi] Memory Forensic: Hunting Cobalt Strike in Memory**[7]

  아티팩트를 분석하기 위해 Didier Stevens가 작성한 '1768.py'를 이용해 메모리 이미지에 Cobalt Strike가 존재하는지 확인하고, 'Volatility' 프레임워크를 사용해 메모리 이미지를 수동 분석하는 방법에 대해 설명했다.

- **[Riccardo Ancarani] Detecting Cobalt Strike Default Modules via Named Pipe Analysis**[8]

  Cobalt Strike의 일부 모듈을 사용할 때 명명된 파이프가 생성되는 것을 통해 Cobalt Strike 사용을 탐지하는 방법에 대해 설명했다.

- **[The DFIR Report] Cobalt Strike, a Defender's Guide**[9]

  Cobalt Strike를 방어하기 위해 공격자의 공격 단계(실행, 방어 회피, 정찰, 권한 상승, 자격 증명 접근, 명령 및 제어, 측면 이동)에서 식별할 수 있는 Sysmon과 Windows 이벤트들에 대해 설명했다.

[4] dbpia.co.kr/journal/articleDetail?nodeId=NODE11554248
[5] igloo.co.kr/security-information/침투-테스팅-도구-cobalt-strike-part-1-기능편/
[6] medium.com/@balasubramanya.c/cobaltstrike-forensics-32e6ce68ce42
[7] cybersync.org/blogs-en/hunting_cobalt_strike_in_memory
[8] labs.withsecure.com/publications/detecting-cobalt-strike-default-modules-via-named-pipe-analysis
[9] thedfirreport.com/2021/08/29/cobalt-strike-a-defenders-guide/

- **[Lexfo] Cobalt Strike Investigation**[10]

Cobalt Strike를 통해 내부 이동(Lateral Movement) 전술에서 주로 사용할 수 있는 명령어(remote-exec, jump)가 사용됐을 때 식별할 수 있는 아티팩트에 대해 설명했다.

- **[Erwan Chevalier, Narimane Lavay and Sekoia TDR] Hunting and detecting Cobalt Strike**[11]

기본 인증서와 공개된 Malleable C2 profile을 사용하는 Cobalt Strike C2 서버를 찾는 방법에 대해 설명하고, 공격자가 측면 이동할 때의 아티팩트들과 명명된 파이프를 통해 Cobalt Strike를 탐지하는 것에 대해 설명했다.

- **[Invictus Incident Response] Responding to a Cobalt Strike Attack**[12]

Cobalt Strike에 대해 설명하고 Cobalt Strike 분석에 활용할 수 있는 OSINT/위협 인텔리전스를 소개했다. 또한, Cobalt Strike 공격을 두 그룹(단계적 공격, 무단계 공격)으로 분류한 뒤 실행 파일과 메모리 분석 방법에 대해 설명했다.

- **[UNIT 42] Cobalt Strike Analysis and Tutorial: How Malleable C2 Profiles Make Cobalt Strike Difficult to Detect**[13]

Cobalt Strike의 프로필에 대해 설명하고 기본 프로필을 사용할 때와 맞춤형 프로필을 사용할 때 네트워크 통신에서의 변화에 대해 설명했다.

- **[Cyb3rSn0rlax] Detecting CONTI CobaltStrike Lateral Movement Techniques**[14]

CONTI 랜섬웨어 그룹의 Cobalt Strike를 이용한 측면 이동 기술에 대해 MITRE ATT&CK에 매칭하고, Cobalt Strike의 내장 모듈을 사용한 측면 이동을 탐지할 수 있는 이벤트에 대해 설명했다.

- **[Recorded Future] Full-Spectrum Cobalt Strike Detection**[15]

Cobalt Strike 활동을 효과적으로 탐지하기 위해 호스트 기반 모니터링, 네트워크 기반 모니터링, Cobalt Strike C2 식별을 위한 위협 인텔리전스 등 전체 스펙트럼의 탐지에 대해 설명하는 보고서를 발표했다.

[10] blog.lexfo.fr/Cobalt Strike Investigation Part 1.html
[11] blog.sekoia.io/hunting-and-detecting-cobalt-strike
[12] linkedin.com/pulse/responding-cobalt-strike-attack-part-i-invictus-incident-response/?trk=article-ssr-frontend-pulse_little-text-block
[13] unit42.paloaltonetworks.com/cobalt-strike-malleable-c2-profile
[14] unh4ck.com/detection-engineering-and-threat-hunting/lateral-movement/detecting-conti-cobaltstrike-lateral-movement-techniques-part-1
[15] go.recordedfuture.com/hubfs/reports/mtp-2021-0914.pdf

4. 활용 공격 사례

Cobalt Strike가 공격 과정에서 어떻게 사용되는지를 파악하기 위해 최근 3년간 발생한 침해사고 사례를 조사했다. 조사 결과, 공격자들이 다양한 유형의 공격에서 Cobalt Strike를 주요 도구로 활용하고 있음을 확인할 수 있었다. 특히 랜섬웨어 공격과 APT(Advanced Persistent Threat) 캠페인에서 Cobalt Strike는 명령 제어(C2) 통신을 유지하고 내부 네트워크를 장악하며 공격을 확장하는데 자주 사용된 것으로 확인된다. 아래 표는 Cobalt Strike가 활용된 국내/외 침해사고 사례를 정리한 것이다.

[표 9] 국내/외에서 발생한 Cobalt Strike 활용 사례 목록

구분	공격 그룹	공개 연도	공격 사례 내용 요약
랜섬웨어 감염	Conti	2022	스팸 메일을 통해 최초 접근 후, Windows 유틸리티로 호스트에서 정찰을 수행하고 Cobalt Strike를 통해 자격 증명을 덤프했다. SMB를 사용해 Cobalt Strike DLL을 도메인 컨트롤러와 다른 서버로 전송했고, 도메인 전체를 랜섬웨어에 감염시켰다.
	BlueSky	2023	인터넷에 노출된 MS-SQL 서버를 대상으로 무차별 대입 공격을 통해 최초 접근한 후, xp_cmdshell을 사용해 Cobalt Strike Beacon을 다운로드 받고 명령 제어 서버와 연결했다. SMBExec를 사용해 SMB에 대한 정보를 수집하고, SMB를 통해 네트워크의 모든 시스템을 랜섬웨어에 감염 시켰다.
	Rhysida	2024	스피어 피싱을 통해 사용자가 Cobalt Strike Beacon을 다운로드 및 실행하도록 유도했다. 이후 Cobalt Strike를 활용해 권한 상승하고 PowerShell, WinSCP, Cobalt Strike, PsExec 등으로 내부 이동하면서 랜섬웨어 감염 및 정보를 유출했다.
	Storm-1811	2024	Microsoft Teams에서 헬프 데스크 직원으로 사칭해 Quick Assist 기능을 통해 시스템에 접근했다. 이후 악성 파일을 시스템에 설치하고, PsExec를 사용해 네트워크 전체에 랜섬웨어를 배포했다.
	ALPHV	2024	스팸 메일을 통해 최초 접근한 후 Cobalt Strike Beacon을 다운로드 받고, Cobalt Strike를 통해 자격증명을 덤프 했다. 이후 ScreenConnect와 SMB를 통해 내부 이동하다가 랜섬웨어 감염 및 정보를 유출했다.
	BlackSuit	2024	공격자에 의해 커스터마이징된 Cobalt Strike Beacon으로 네트워크에 최초로 접근했다. 이후 Cobalt Strike를 통해 원격에서 명령을 실행하고, Kerberoasting 공격을 수행해 Kerberos 티켓을 탈취했다. SMB를 이용해 Cobalt Strike Beacon을 전파한 후 랜섬웨어를 감염 시켰다.
	APT41	2024	웹 서버에 웹셸을 업로드해 Cobalt Strike를 포함한 악성 파일들을 유입시켰다. 자격증명 덤프, 네트워크 스캐닝 등 정찰 활동을 수행하고 패치되지 않은 취약점을 활용해 권한을 상승했다. 이후 랜섬웨어 감염 및 정보를 유출했다.
악성코드 감염 (백도어)	미확인	2021	MS Exchange 서버를 대상으로 취약점을 악용해 시스템 인증을 우회하고, 웹셸을 업로드 했다. 웹셸을 이용해 Cobalt Strike 비콘을 다운로드하고 실행했다.
	미확인	2022	부적절하게 관리되고 있는 MS-SQL 서버를 대상으로 무차별 대입 공격을 통해 최초 접근했다. 취약점을 악용해 MS-SQL 서버 관련 프로세스를 통해 Cobalt Strike Beacon을 실행했다.

구분	공격 그룹	공개 연도	공격 사례 내용 요약
악성코드 감염 (백도어)	Dalbit	2023	WebLogic 취약점이나 파일 업로드 취약점을 악용해 웹셸을 업로드 했다. 웹셸을 통해 악성 파일을 다운로드 받고 권한 상승, 네트워크 스캐닝을 수행하기 위해 Cobalt Strike를 백도어로 활용했다.
	UAC-0057	2022	피싱 메일을 통해 최초 접근한 후 추가적인 악성 파일을 다운로드 받았다. 공격자가 유입시킨 악성 파일 중에 명령 제어 서버와 통신하는 Cobalt Strike Beacon이 포함됐다.
악성코드 감염 (코인마이너)	미확인	2023	아파치 웹 서버를 대상으로 웹셸을 업로드해 추가 악성 파일을 다운로드하는 다운로더를 유입시켰다. 다운로더를 통해 Cobalt Strike를 다운로드하고 명령 제어 서버와 통신하고 내부 이동하다가 코인을 채굴하는 악성 프로그램을 설치했다.

국내/외에서 발생한 Cobalt Strike 활용 침해사고 사례에서 Cobalt Strike 활용 방안을 간략히 요약한 내용은 다음과 같다.

- **[랜섬웨어 감염] Conti, 2022**[16]

  Cobalt Strike를 통해 LSASS 메모리에서 자격증명을 덤프하고 내부 이동에 활용했다. Cobalt Strike Beacon을 실행시킬 때 정상적인 프로그램(explorer.exe)에 주입해 보안 프로그램을 회피했다.

- **[랜섬웨어 감염] BlueSky, 2023**[17]

  Cobalt Strike를 명령 제어 서버와 통신해 원격 명령 실행과 내부 이동에 활용했다. 해당 공격자들은 MSSQL 서버의 잘 관리되지 않는 관리자 계정을 탈취해 시스템에 접근했다.

- **[랜섬웨어 감염] Rhysida, 2024**[18]

  Cobalt Strike를 권한 상승과 내부 이동에 활용했다. 해당 공격자들은 인프라(웹 서버, 이메일 서버 등)의 패치 되지 않은 취약점을 악용했다.

- **[랜섬웨어 감염] Storm-1811, 2024**[19]

  Cobalt Strike를 명령 제어 서버와 통신해 원격 명령 실행에 활용했다. 해당 공격 그룹은 "Help Desk", "Help Desk IT", "Help Desk Support", "IT Support" 등 헬프 데스크 직원으로 사칭해 접근했다.

- **[랜섬웨어 감염] ALPHV, 2024**[20]

  Cobalt Strike를 통해 LSASS 메모리에서 자격증명을 덤프하고 내부 이동에 활용했다. Cobalt Strike Beacon을 실행시킬 때 정상적인 프로그램(winlogon.exe)에 주입해 보안 프로그램의 탐지를 회피했다.

[16] thedfirreport.com/2022/04/04/stolen-images-campaign-ends-in-conti-ransomware
[17] thedfirreport.com/2023/12/04/sql-brute-force-leads-to-bluesky-ransomware
[18] blog.barracuda.com/2024/05/09/rhysida-ransomware--the-creepy-crawling-criminal-hiding-in-the-d
[19] www.microsoft.com/en-us/security/blog/2024/05/15/threat-actors-misusing-quick-assist-in-social-engineering-attacks-leading-to-ransomware
[20] thedfirreport.com/2024/06/10/icedid-brings-screenconnect-and-csharp-streamer-to-alphv-ransomware-deployment

- **[랜섬웨어 감염] BlackSuit, 2024**[21]

Cobalt Strike를 명령 제어 서버와 통신해 원격 명령 실행과 내부 이동에 활용했다. 최초 접근에 공격자가 커스텀하게 제작한 Cobalt Strike를 이용해 탐지를 어렵게 만들었다.

- **[랜섬웨어 감염] APT41, 2024**[22]

Cobalt Strike를 명령 제어 서버와 통신해 원격 명령 실행에 활용했다. 백신을 우회하기 위해 Anti-AV 로더를 사용해 Windows Defender에서 Cobalt Strike를 탐지하지 못하도록 했다.

- **[악성코드 감염(백도어)] 미확인, 2021**[23]

MS Exchange 서버를 대상으로 공격을 수행한 공격자는 공격 과정에서 Cobalt Strike를 실행해 백도어로 활용했다. Cobalt Strike Beacon을 실행시킬 때 정상적인 프로그램(msdtc.exe)에 주입해 보안 프로그램의 탐지를 회피했다.

- **[악성코드 감염(백도어)] 미확인, 2022**[24]

MS-SQL 서버를 대상으로 공격을 수행한 공격자는 공격 과정에서 Cobalt Strike를 실행해 백도어로 활용했다. Cobalt Strike Beacon을 실행시킬 때 정상적인 프로그램(mshta.exe, rundll32.exe)에 주입해 보안 프로그램을 회피했다.

- **[악성코드 감염(백도어)] Dalbit, 2023**[25]

Cobalt Strike를 실행해 백도어로 활용했다. 추가로 LSASS 메모리에서 자격증명을 덤프하고 호스트의 화면 이미지를 명령 제어 서버로 전송하는 것이 확인됐다.

- **[악성코드 감염(백도어)] UAC-0057, 2022**[26]

공격 과정에서 Cobalt Strike를 백도어로 활용했다. 추가로 Cobalt Strike로 취약점을 악용해 악성 파일 배포에 사용했다.

- **[악성코드 감염(코인마이너)] 미확인, 2023**[27]

Cobalt Strike를 명령 제어 서버와 통신해 원격 명령 실행과 내부 이동에 활용했다. 명령 제어 서버와의 통신에서는 HTTP 프로토콜을 사용하고, 내부 이동에는 SMB 프로토콜을 사용했다.

---

[21] thedfirreport.com/2024/08/26/blacksuit-ransomware

[22] blog.talosintelligence.com/chinese-hacking-group-apt41-compromised-taiwanese-government-affiliated-research-institute-with-shadowpad-and-cobaltstrike-2

[23] asec.ahnlab.com/ko/27448

[24] asec.ahnlab.com/ko/31657

[25] asec.ahnlab.com/ko/46431

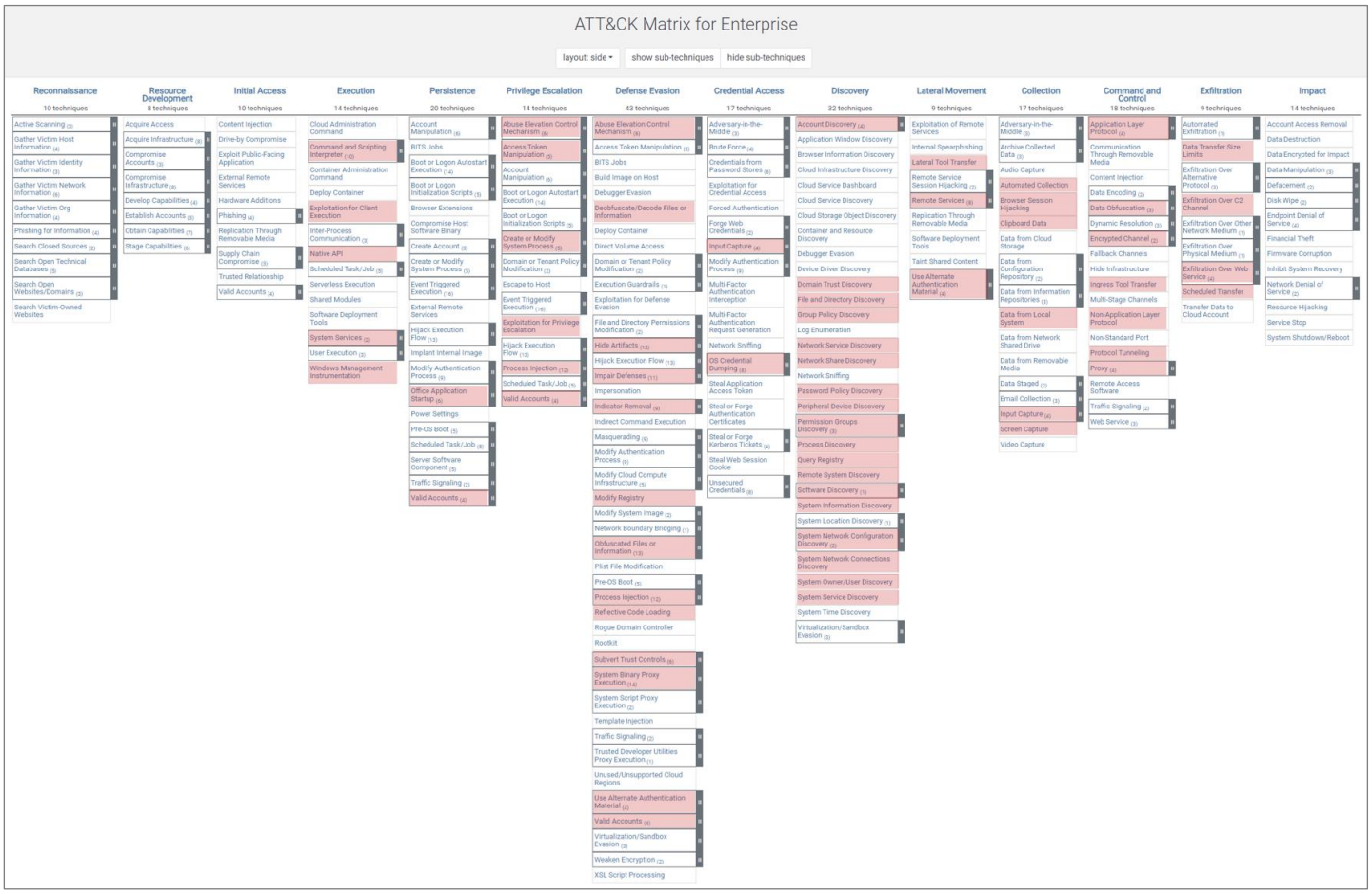
[26] socprime.com/blog/cobalt-strike-beacon-malware-spread-via-targeted-phishing-emails-related-to-azovstal-cyber-attack-on-ukrainian-government-entities

[27] asec.ahnlab.com/ko/58882

5. 공격 전술

5.1. 활용 가능한 공격 전술

Cobalt Strike가 수행할 수 있는 공격 기법을 MITRE ATT&CK 프레임워크 전술과 매칭해보면, 총 10개의 전술에 해당된다. 이 전술들을 보면, 정찰(Reconnaissance)이나 최초 침투(Initial Access) 단계에서는 매칭되지 않고, 실행(Execution) 단계부터 매칭되는 것을 확인할 수 있다. 이는 Cobalt Strike가 초기 침투 단계에서 사용되고 있는 도구가 아니라 침투가 완료된 후 후속 공격(Post-exploitation) 단계에서 최적화된 도구임을 의미한다.



[그림 8] MITRE ATT&CK 기반 Cobalt Strike에서 활용 가능한 공격 전술과 기법 (v15 기준, 빨간색 음영으로 표시)

5.2. 주로 활용되는 공격 전술

Cobalt Strike는 여러 전술과 기법에 걸쳐 활용될 수 있으며, 실제 침해사고 사례를 보면 공격자들이 주로 원격 명령 실행과 내부 이동 단계에서 많이 사용하고 있음을 확인할 수 있다. MITRE ATT&CK 전술별로 Cobalt Strike에서 주로 사용되는 공격 기법과 명령은 다음과 같다.

[표 10] MITRE ATT&CK 기반 Cobalt Strike에서 주로 활용 가능한 공격 전술과 기법 목록 (v15 기준)

전술(Tactics)	TID	기법(Techniques)	Cobalt Strike 명령
Execution	T1059	Command and Scripting Interpreter	shell
			run
			powershell
Privilege Escalation	T1543	Create or Modify System Process	elevate svc-exe
	T1068	Exploitation for Privilege Escalation	getsystem
Defense Evasion	T1055	Process Injection	inject
			dllload
			spawnto
	T1620	Reflective Code Loading	dllinject
Credential Access	T1003	OS Credential Dumping	hashdump
			logonpasswords
			mimikatz
Discovery	T1046	Network Service Discovery	portscan
	T1018	Remote System Discovery	net
Lateral Movement	T1021	Remote Services	remote-exec
			jump
	T1570	Lateral Tool Transfer	upload
Collection	T1119	Automated Collection	screenwatch
Command and Control	T1090	Proxy	socks
Exfiltration	T1041	Exfiltration Over C2 Channel	download

6. 연구 내용

6.1. 연구 개요

본 연구는 Cobalt Strike의 Beacon 페이로드와 기본 제공 명령이 시스템에 남기는 흔적을 체계적으로 분석해, 이를 바탕으로 Cobalt Strike를 활용한 사이버 위협에 대한 탐지 방안과 실무 적용 전략을 제시하는 것을 목표로 한다. 또한, 다년간의 침해사고 분석 경험을 바탕으로, 국내에서 자주 발생하는 Active Directory(이하, AD) 환경에서의 침해사고 시나리오를 재연해 실제 발생 가능한 위협에 대한 탐지 방안을 제안하고자 한다. 연구는 크게 세 가지로 구분해 진행되었다.

- **Cobalt Strike 분석**

Beacon의 동작 방식을 프로토콜과 페이로드 유형별로 구분하고, 각 동작 유형이 호스트에 남기는 흔적을 분석한다. 또한, Cobalt Strike에서 기본으로 제공하는 98개의 명령을 수행한 후, 로깅 설정이 강화된 환경과 기본 로깅 환경에서 남는 흔적의 차이를 분석한다. 이 과정에서 Sysmon 로그와 호스트 시스템 아티팩트를 모두 분석해 각 명령이 남기는 세부적인 행위 흔적을 심층적으로 연구한다. 이를 통해 침해사고 탐지에 필요한 유의미한 지표들을 도출하고자 한다.

- **Cobalt Strike 탐지 방안 연구**

Microsoft에서 시스템 활동과 네트워크 연결을 상세하게 기록하기 위해 제작한 Sysmon과 한국인터넷진흥원에서 해킹 사고의 여부를 원클릭으로 확인할 수 있도록 제작한 해킹진단도구를 활용해 Cobalt Strike와 관련된 위협을 탐지할 수 있는 탐지 룰을 설계한다. 이를 기반으로 다양한 공격 경로를 고려하고 실제 환경에서 탐지 효과를 극대화할 수 있는 탐지 룰 구성 방법을 제안한다.

- **시나리오 기반 탐지 룰 테스트**

AD 환경을 기반으로 한 침해사고 시나리오를 설정하고, Cobalt Strike를 활용한 공격을 모의로 실행해 탐지 방안 연구에서 개발된 Sysmon Configure Rule과 해킹진단도구의 탐지 룰을 검증한다. 이를 통해 탐지 규칙의 정확도와 성능을 평가하고, 실제 환경에서의 효과적인 위협 탐지 능력을 검증함으로써 Cobalt Strike에 대한 방어 역량을 강화하는 데 기여할 수 있도록 한다.

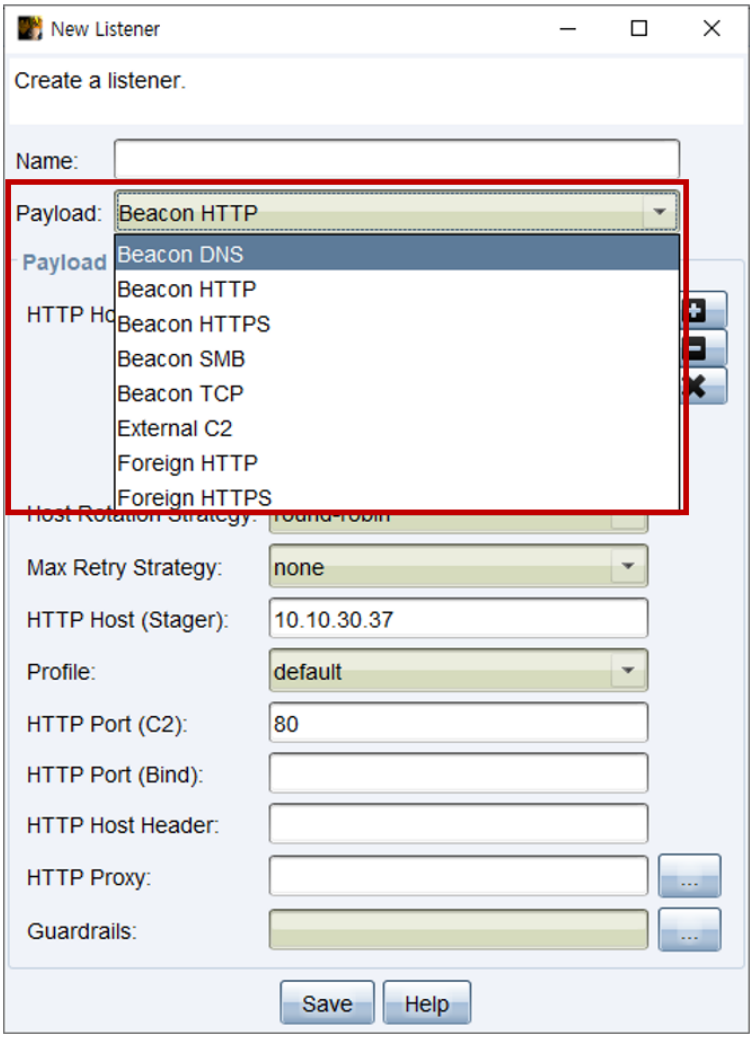
6.2. Cobalt Strike 분석

6.2.1. Beacon 동작 방식별 흔적 분석

본 연구에서는 시스템에서 실행되는 Beacon의 흔적을 탐지하기 위해 Beacon의 동작 방식을 변화시킬 수 있는 프로토콜 유형, 페이로드 유형, 동작 방식에 따른 실행 이벤트를 분석했다. 이를 위해 Cobalt Strike의 Beacon 페이로드를 Listener 프로토콜, 페이로드 종류, 동작 방식에 따라 세분화하고, 각 유형에서 발생하는 호스트 시스템의 흔적을 체계적으로 연구했다. 분석은 Beacon 실행, 주기적 통신, 명령어 실행, Beacon 종료 등의 이벤트로 구분해 진행했다.

6.2.1.1. Listener 프로토콜 유형

Cobalt Strike의 Beacon 페이로드는 5개의 Listener 프로토콜(HTTP, HTTPS, SMB, TCP, DNS)을 통해 통신한다.



[그림 9] Listener 프로토콜 유형

해당 연구에서는 DNS를 제외한 4개의 Listener 프로토콜을 통해 Beacon의 동작 방식을 분류한 후, Beacon의 동작 방식을 분석하기 위해 7개의 이벤트(Beacon 실행, 주기적 통신, 명령어 실행, Beacon 종료, 연결(Connect), 연결(Link), 연결 해제(UnLink)로 구분했다. 본 연구를 위해 Beacon의 형태는 Windows EXE, Beacon의 동작 방식은 Stageless로 설정했다. 연구 결과, Beacon이 통신하는 Listener 프로토콜 유형에 대한 차이점은 외부 Team Server 또는 Beacon 간 통신에 사용되는 통신 포트의 차이가 존재했다. Listener 프로토콜 유형에 따른 Beacon 통신의 차이점에 대한 연구 결과는 다음과 같다.

[표 11] Listener 프로토콜 유형에 따른 Beacon 통신의 차이점

프로토콜 구분	연구 결과
HTTP	외부 Team Server와의 통신에서 주로 사용되는 유형으로, HTTP GET 요청으로 명령을 가져오고, HTTP POST 요청으로 데이터를 보내도록 기본 값이 설정되어 있기 때문에 80번 포트(HTTP)를 통신에 사용한다.
HTTPS	외부 Team Server와의 통신에서 주로 사용되는 유형으로, HTTP GET 요청으로 명령을 가져오고, HTTP POST 요청으로 데이터를 보내도록 기본 값이 설정되어 있기 때문에 443번 포트(HTTPS)를 통신에 사용한다. HTTP와의 차이점은 Beacon이 실행되는 과정에서 인증서 관련 레지스트리, 인터넷 익스플로러(IE) 및 Windows의 기타 인터넷 관련 기능에 대한 레지스트리를 생성하고 수정한다.
SMB	내부 Beacon 간 통신에서 주로 사용되는 유형으로, 명명된 파이프를 사용해 445번 포트(SMB)를 통신에 사용한다. SMB Listener를 생성할 때 Pipe 이름의 기본 값이 'msagent_88'로 설정되어 있으며, 이는 수정이 가능하다. 기본적으로 제공되는 명령인 'jump'를 사용해 SMB Beacon을 생성하게 되면 자동으로 연결(Link)되지만, 그 외에는 SMB 연결을 수행하고자 하는 대상에서 Beacon을 Listening 상태로 대기시킨 다음 'link' 명령을 사용해 연결해야 한다. SMB Beacon을 연결(Link)할 때 네트워크 공유 개체 'IPC$'를 사용한다.
TCP	내부 Beacon 간 통신에서 주로 사용되는 유형으로, TCP 소켓을 통신에 사용한다. TCP Listener를 생성할 때, 기본 포트는 '4444'로 설정되어 있으며, 이는 수정이 가능하다.

각 이벤트 별 Listener 프로토콜에 해당하는 Beacon의 상세 동작 방식은 다음과 같다.

1) Beacon 실행

Beacon을 실행했을 때 모든 프로토콜에서 이미지(DLL) 로드 이벤트와 Pipe 이벤트가 공통적으로 발생했으며, HTTPS와 SMB에서 추가 이벤트가 발생했다. 이미지(DLL) 로드 이벤트는 공통적으로 발생했지만 각 프로토콜 별로 로드되는 DLL의 차이는 존재했다. HTTPS에서는 레지스트리 이벤트에서 인터넷 익스플로러(IE) 또는 Windows 기타 인터넷 관련 기능에 대한 레지스트리 값 변경 이벤트가 발생했으며, SMB에서는 Team Server에서 SMB Listener를 생성할 때 설정한 Pipe를 생성하는 이벤트가 추가로 발생했다.

[표 12] Listener 프로토콜 유형에 따른 Beacon 실행 이벤트의 차이점

프로토콜 구분	이벤트 유형	설명
HTTP	이미지(DLL) 로드	HTTP Beacon 프로세스가 생성되면서 이미지(DLL) 로드 이벤트 발생
	Pipe 생성 및 연결	명명된 Pipe를 생성하고 연결하는 이벤트 발생 (명명된 Pipe의 이름 형태는 '\MSSE-{랜덤 숫자 4자리}-server'로 구성)
HTTPS	이미지(DLL) 로드	HTTPS Beacon 프로세스가 생성되면서 이미지(DLL) 로드 이벤트 발생
	Pipe 생성 및 연결	명명된 Pipe를 생성하고 연결하는 이벤트 발생 (명명된 Pipe의 이름 형태는 '\MSSE-{랜덤 숫자 4자리}-server'로 구성)
	레지스트리 수정	HTTP Beacon 프로세스가 생성되면서 인터넷 익스플로러(IE) 또는 Windows 기타 인터넷 관련 기능에 대한 레지스트리 값을 변경해 보안 설정 비활성화
SMB	이미지(DLL) 로드	SMB Beacon 프로세스가 생성되면서 이미지(DLL) 로드 이벤트 발생
	Pipe 생성 및 연결	명명된 Pipe를 생성하고 연결하는 이벤트 발생 (명명된 Pipe의 이름 형태는 '\MSSE-{랜덤 숫자 4자리}-server'로 구성) 이후, Team Server에서 SMB Listener를 생성할 때 설정한 Pipe를 생성하는 이벤트 발생 (SMB Listener를 생성할 때 Pipe 이름이 기본 값으로 '\msagent_88'로 설정되어 있으며, Pipe 이름은 수정할 수 있다.)
TCP	이미지(DLL) 로드	TCP Beacon 프로세스가 생성되면서 이미지(DLL) 로드 이벤트 발생
	Pipe 생성 및 연결	명명된 Pipe를 생성하고 연결하는 이벤트 발생 (명명된 Pipe의 이름 형태는 '\MSSE-{랜덤 숫자 4자리}-server'로 구성)

[표 12-1] Beacon 실행 시 발생하는 이벤트 패턴(HTTPS)

이벤트 ID	이벤트 유형	내용
7	Image loaded	Image=%HTTPS Beacon File%, ImageLoaded=C:\Windows\System32\ntdll.dll
7	Image loaded	Image=%HTTPS Beacon File%, ImageLoaded=C:\Windows\System32\KERNEL32.DLL
7	Image loaded	(생략)

이벤트 ID	이벤트 유형	내용
7	Image loaded	Image=%HTTPS Beacon File%, ImageLoaded=C:\Windows\System32\ncryptsslp.dll
17	Pipe Created	PipeName=\MSSE-*-server, Image=%HTTPS Beacon File%
18	Pipe Connected	PipeName=\MSSE-*-server, Image=%HTTPS Beacon File%
13	Registry - Value set	TargetObject=HKU*\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\UNCAsIntranet, Details=DWORD (0x00000000)
13	Registry - Value set	TargetObject=HKU*\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\AutoDetect, Details=DWORD (0x00000000)
12	Registry - CreateKey	(생략)
12	Registry - CreateKey	TargetObject=*\EnterpriseCertificates\Trust\CRLs
12	Registry - CreateKey	TargetObject=*\EnterpriseCertificates\Trust\CTLs
7	Image loaded	Image=%HTTPS Beacon File%, ImageLoaded=C:\Windows\System32\ntdll.dll
7	Image loaded	Image=%HTTPS Beacon File%, ImageLoaded=C:\Windows\System32\KERNEL32.DLL

2) 연결(Connect)

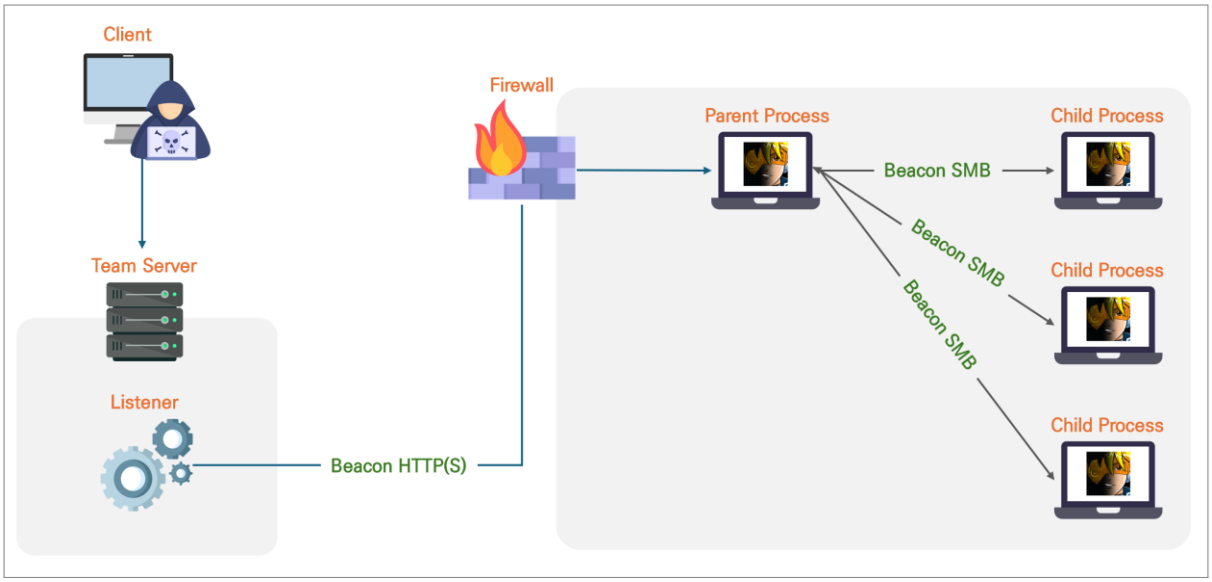
'Connect' 명령은 TCP 프로토콜에서만 사용하는 방식으로, 해당 명령을 통해 TCP Beacon과 Link 수립 시, TCP 포트로 네트워크가 연결된 이벤트가 발생했다.

[표 13] TCP 포트로 연결된 네트워크 연결 이벤트

이벤트 ID	이벤트 유형	내용
3	Network connection	Image=%Parent Beacon File%, SourceIp=%Parent Host IP%, SourcePort=%N%, DestinationIp=%TCP Host IP%, DestinationPort=443, DestinationPortName=https
3	Network connection	Image=%Parent Beacon File%, SourceIp=%Parent Host IP%, SourcePort=%N+1%, DestinationIp=%TCP Host IP%, DestinationPort=%TCP Port%
3	Network connection	Image=%Parent Beacon File%, SourceIp=%Parent Host IP%, SourcePort=%N+2%, DestinationIp=%TCP Host IP%, DestinationPort=443, DestinationPortName=https
3	Network connection	Image=%TCP Beacon File%, SourceIp=%TCP Host IP%, SourcePort=%N%, DestinationIp=%Parent Host IP%, DestinationPort=%TCP Port%

3) 연결(Link)

'Link' 명령은 SMB 프로토콜에서만 사용하는 방식이다. 통신 시, Parent Beacon 호스트와 SMB Beacon 호스트(Child Process)가 구분되고, 각 호스트 별로 시스템에 남기는 흔적이 다르다.



[그림 10] SMB Beacon 동작 방식

SMB Beacon 호스트(Child Process)에서는 'link' 명령을 통해 SMB Beacon과 Link 수립 시, System 프로세스에서 Beacon 실행 당시 생성된 Pipe에 연결하고 Parent Beacon IP와 SMB(445) 포트로 네트워크 연결 이벤트가 발생했다. 또한, SMB Beacon이 동작 중인 계정으로 네트워크 로그온이 수행되고, 네트워크 공유 개체 '*\IPC$'에 접근하는 이벤트가 발생했다.

[표 14] SMB Beacon(Child Process) – Pipe 연결 후 Parent Beacon IP와 SMB(445) 포트로의 네트워크 연결 이벤트

이벤트 ID	이벤트 유형	내용
18	Pipe Connected	PipeName=\msagent_88, Image=System
3	Network connection	Image=System, SourceIp=%Parent Host IP%, SourcePort=%N%, DestinationIp=%Host IP%, DestinationPort=445, DestinationPortName=microsoft-ds
3	Network connection	Image=System, SourceIp=%Parent Host IP%, SourcePort=%N+1%, DestinationIp=%Host IP%, DestinationPort=445, DestinationPortName=microsoft-ds
3	Network connection	Image=System, SourceIp=%Parent Host IP%, SourcePort=%N+2%, DestinationIp=%Host IP%, DestinationPort=445, DestinationPortName=microsoft-ds
3	Network connection	Image=System, SourceIp=%Parent Host IP%, SourcePort=%N+3%, DestinationIp=%Host IP%, DestinationPort=445, DestinationPortName=microsoft-ds

[표 15] SMB Beacon(Child Process) – 네트워크 로그온 수행 후 네트워크 공유 개체 접근 이벤트

이벤트 ID	이벤트 유형	내용
4624	네트워크 로그온	TargetUserName=Administrator, TargetLogonId=4576956, LogonType=3, LogonProcessName=NtLmSsp , AuthenticationPackageName=NTLM, IpAddress=%Parent Host IP%, IpPort=%N%
5140	네트워크 공유 개체 접근	SubjectUserName=Administrator, SubjectLogonId=4576956, ObjectType=File, IpAddress=%Parent Host IP%, IpPort=%N%, ShareName=*\IPC$, ShareLocalPath=, AccessMask=1, AccessList=%%4416

Parent Beacon 호스트에서는 Parent Beacon 프로세스에서 Team Server와 네트워크 연결 이벤트가 발생했다.

[표 16] Parent Process – Team Server와의 네트워크 연결 이벤트

이벤트 ID	이벤트 유형	내용
3	Network connection	Image=%Parent Beacon File%, SourceIp=%Parent Host IP%, SourcePort=%N%, DestinationIp=%Team Server IP%, DestinationPort=443, DestinationPortName=HTTPS
3	Network connection	Image=%Parent Beacon File%, SourceIp=%Parent Host IP%, SourcePort=%N+1%, DestinationIp=%Team Server IP%, DestinationPort=443, DestinationPortName=HTTPS

4) Beacon 주기적 통신

Beacon을 통해 주기적인 통신을 수행했을 때, SMB를 제외한 모든 프로토콜에서 각 프로토콜의 포트로 주기적인 네트워크 연결 이벤트가 발생하는 패턴이 확인되었다. SMB의 경우, 주기적인 통신 과정에서 네트워크 연결 이벤트가 발생하지 않았다.

[표 17] Listener 프로토콜 유형에 따른 Beacon 주기적 통신 이벤트의 차이점

프로토콜 구분	이벤트 유형	설명
HTTP	네트워크 연결	HTTP Beacon 프로세스에서 Team Server와 HTTP(80) 포트로 주기적인 네트워크 연결 이벤트 발생 (이때, 네트워크 연결 주기는 설정된 sleep 값에 따라 다름)
HTTPS	네트워크 연결	HTTPS Beacon 프로세스에서 Team Server와 HTTPS(443) 포트로 주기적인 네트워크 연결 이벤트 발생 (이때, 네트워크 연결 주기는 설정된 sleep 값에 따라 다름)
SMB	–	네트워크 연결 이벤트가 발생하지 않음
TCP	네트워크 연결	TCP Beacon을 연결시킨 Parent Beacon 프로세스에서 Team Server와 주기적인 네트워크 연결 이벤트 발생 (이때, 네트워크 연결 주기는 설정된 sleep 값에 따라 다름)

[표 17-1] Beacon으로 주기적 통신 수행 시 발생하는 이벤트 패턴(HTTPS)

이벤트 ID	이벤트 유형	내용
3	Network connection	Image=%HTTPS Beacon File%, SourceIp=%Host IP%, SourcePort=%N%, DestinationIp=%Team Server IP%, DestinationPort=443, DestinationPortName=https

5) 연결 해제(UnLink)

'UnLink' 명령은 SMB 프로토콜에서만 사용하는 방식이다. 해당 명령을 실행했을 때 연결(Link)때 발생된 네트워크 로그온에 해당하는 Logon ID로 로그오프 이벤트가 발생했다.

[표 18] 연결(Link)때 발생한 네트워크 로그온에 해당하는 Logon ID로 로그오프 이벤트

이벤트 ID	이벤트 유형	내용
4634	네트워크 로그오프	TargetUserName=Administrator, TargetLogonId=4576956, LogonType=3

6) Beacon 종료

Beacon을 종료했을 때, 모든 프로토콜에서 생성한 Beacon 프로세스 종료 이벤트가 발생한 후 네트워크 연결 이벤트가 두 번 연속으로 발생하는 패턴이 확인됐다.

[표 19] Listener 프로토콜 유형에 따른 Beacon 종료 이벤트의 차이점

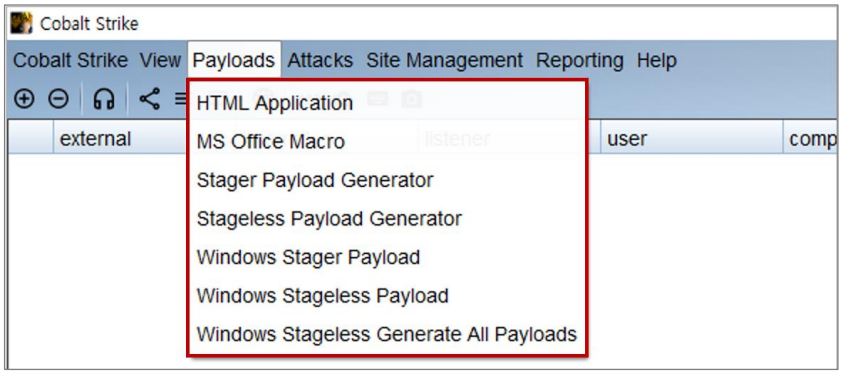
프로토콜 구분	이벤트 유형	설명
HTTP	프로세스 종료	HTTP Beacon 프로세스 종료 이벤트 발생 후 네트워크 연결 이벤트가 두 번 연속 발생
HTTPS	프로세스 종료	HTTPS Beacon 프로세스 종료 이벤트 발생 후 네트워크 연결 이벤트가 두 번 연속 발생
SMB	프로세스 종료	SMB Beacon 프로세스 종료 이벤트 발생 후 네트워크 연결 이벤트가 두 번 연속 발생
TCP	프로세스 종료	TCP Beacon 프로세스 종료 이벤트 발생 후 네트워크 연결 이벤트가 두 번 연속 발생

[표 19-1] Beacon 종료 시 발생하는 이벤트 패턴(HTTPS)

이벤트 ID	이벤트 유형	내용
5	Process terminated	Image=%HTTPS Beacon File%
3	Network connection	Image=%HTTPS Beacon File%, SourceIp=%Host IP%, SourcePort=%N%, DestinationIp=%Team Server IP%, DestinationPort=443, DestinationPortName=https
3	Network connection	Image=%HTTPS Beacon File%, SourceIp=%Host IP%, SourcePort=%N+1%, DestinationIp=%Team Server IP%, DestinationPort=443, DestinationPortName=https

6.2.1.2. Beacon 페이로드 유형

Cobalt Strike의 Beacon 페이로드는 7개의 유형(HTML Application, MS Office Macro, PowerShell Script, Windows EXE, Windows Service EXE, Windows DLL, Raw)이 존재한다.



[그림 11] Beacon 페이로드 유형

해당 연구에서는 6개의 Beacon 페이로드 별로 Beacon을 실행했을 때의 이벤트를 분석했다. 본 연구를 위해 Beacon의 Listener 프로토콜은 HTTPS로 Beacon의 동작 방식은 Stageless로 설정했다. 분석 결과, Beacon을 실행하는 주체 별로 Beacon 실행 형태에 따른 차이점이 존재했다.

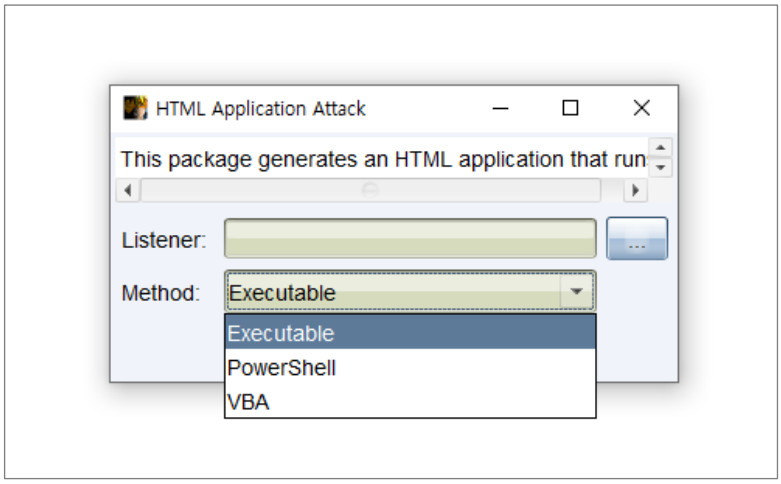
[표 20] Beacon 페이로드 유형에 따른 Beacon 통신의 차이점

페이로드 구분	연구 결과
HTML Application	세부적으로 Executable, PowerShell, VBA가 존재하는데, HTML Application이기 때문에 mstha.exe 프로세스가 실행되면서 하위 프로세스로 Beacon이 실행된다. 해당 형태는 Stager 동작 방식만 지원하기 때문에 HTTP(S) Listener만 생성 가능하다. Executable은 VBScript로 작성되어 있으며, Shellcode를 Beacon EXE 파일로 생성한 뒤 백그라운드에서 실행하고 실행이 끝나면 생성된 EXE 파일과 임시 폴더를 삭제한다. PowerShell은 VBScript로 작성되어 있으며, PowerShell을 실행해 Base64로 인코딩된 악성 페이로드를 실행하기 때문에 mshta.exe 프로세스가 powershell.exe 프로세스를 생성한다. VBA는 VBScript로 작성되어 있으며, Excel을 실행해 악성 매크로 코드를 실행하기 때문에 mshta.exe 프로세스가 excel.exe 프로세스를 생성한다.
MS Office Macro	Microsoft Word 또는 Excel 문서에 저장할 수 있는 VBA 매크로를 생성한다. 해당 형태는 Stager 동작 방식만 지원하기 때문에, HTTP(S) Listener만 생성이 가능하다. 문서의 매크로를 활용하기 때문에 winword.exe, excel.exe 프로세스가 rundll32.exe 프로세스를 실행한다.
PowerShell Script	Beacon을 실행시킬 수 있는 PowerShell Script를 제공하고, 공격자가 PowrShell Script를 어떻게 실행하는가에 따라 실행하는 주체가 달라진다. PowerShell Script 내부 코드 구조에서 Base64 Encoding과 XOR 연산을 수행하는 점을 통해 식별 가능하다.
Windows EXE	Beacon을 실행 파일(.exe)로 제공하기 때문에 공격자가 가장 다양한 방법을 통해 실행할 수 있다.
Windows Service EXE	Beacon을 실행 파일(.exe)로 제공하지만 해당 파일이 서비스를 통해 실행되어야 정상적으로 동작한다. 서비스를 통해 실행되기 때문에 service.exe 프로세스가 Beacon 프로세스를 생성하고, Beacon 프로세스는 rundll32.exe 프로세스를 생성한다.
Windows DLL	DLL(Dynamic Link Library)로 제공하기 때문에 rundll32.exe나 DLL Injection 등의 방법을 통해 실행할 수 있다.

각 Beacon 페이로드 유형에 해당하는 실행 방식은 다음과 같다.

1) HTML Application

HTML Application은 주로 피싱 이메일에 삽입되어 사용자가 클릭하도록 유도함으로써 Cobalt Strike의 Beacon이 실행될 수 있게 하기 위해 사용한다. 해당 페이로드는 HTTP(S) Listener만 생성이 가능하고, 3개의 Method(Executable, PowerShell, VBA)를 통해 제작할 수 있다.



[그림 12] HTML Application 유형

Executable은 VBScript로 작성되어 있으며, Beacon Shellcode를 EXE 파일로 생성한 뒤, 백그라운드에서 실행되고 실행이 끝나면 생성된 EXE 파일과 임시 폴더를 삭제한다. PowerShell은 VBScript로 작성되어 있으며, PowerShell을 실행해 Base64 인코딩된 악성 페이로드를 실행한다. VBA는 VBScript로 작성되어 있으며, Excel을 실행해 악성 매크로 코드를 실행한다. HTTP Application 유형의 Beacon이 실행되었을 때 3개의 메소드 모두 mshta.exe 프로세스가 생성된 후, 해당 프로세스를 통해 각 메소드에서 활용하는 프로세스(rundll32.exe, powershell.exe, excel.exe)가 추가로 생성되는 패턴이 확인되었다.

[표 21] HTML Application Beacon 실행 시, Method 별 발생 이벤트 차이점

구분	이벤트 유형	설명
Executable	Process Create	mshta.exe 프로세스를 통해 인자 값이 존재하지 않는 rundll32.exe 프로세스 생성
PowerShell		mshta.exe 프로세스를 통해 powershell.exe 프로세스가 생성되며, powershell.exe를 통해 인코딩된 페이로드를 실행
VBA		mshta.exe 프로세스를 실행한 후, excel.exe 프로세스가 svchost.exe 프로세스를 통해 생성

[표 21-1] Executable Method로 생성된 Beacon 실행 시 발생하는 프로세스 생성 이벤트

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\SysWOW64\mshta.exe, CommandLine="C:\Windows\SysWOW64\mshta.exe" %HTML Application Beacon File%
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\System32\rundll32.exe, ParentImage=C:\Windows\SysWOW64\mshta.exe

[표 21-2] PowerShell Method로 생성된 Beacon 실행 시 발생하는 프로세스 생성 이벤트

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\SysWOW64\mshta.exe, CommandLine="C:\Windows\SysWOW64\mshta.exe" %HTML Application Beacon File%
1	Process Create	Image=C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe, CommandLine="C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" -nop -w hidden -encodedcommand %Encoding Code%, ParentImage=C:\Windows\SysWOW64\mshta.exe

[표 21-3] VBA Method로 생성된 Beacon 실행 시 발생하는 프로세스 생성 이벤트

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\SysWOW64\mshta.exe, CommandLine="C:\Windows\SysWOW64\mshta.exe" %HTML Application Beacon File%
1	Process Create	Image=C:\Program Files\Microsoft Office\root\Office16\EXCEL.EXE, CommandLine="C:\Program Files\Microsoft Office\Root\Office16\EXCEL.EXE" /automation -Embedding, ParentImage=C:\Windows\System32\svchost.exe, ParentCommandLine=C:\Windows\system32\svchost.exe -k DcomLaunch -p

2) MS Office Macro

MS Office Macro는 Microsoft Word 또는 Microsoft Excel 문서에 포함될 매크로를 생성하는 페이로드 유형으로, 주로 피싱 이메일에 첨부파일로 전달되어 사용자가 매크로를 허용해 Cobalt Strike의 Beacon이 실행될 수 있게 하기 위해 사용한다. 해당 형태는 Stager 동작 방식만 지원하기 때문에, HTTP(S) Listener만 생성이 가능하다. MS Office Macro 유형의 Beacon이 실행되었을 때, 매크로를 삽입하는 애플리케이션에 따라 WINWORD.EXE 또는 EXCEL.EXE 프로세스가 생성된다. 해당 프로세스가 rundll32.exe 프로세스를 생성한 후, 자신의 프로세스를 통해 rundll32.exe에 접근하고 원격 스레드를 생성하는 것으로 확인되었다.

[표 22] MS Office Macro로 생성된 Beacon 실행 시 발생하는 이벤트

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Program Files\Microsoft Office\root\Office16\WINWORD.EXE, CommandLine="C:\Program Files\Microsoft Office\Root\Office16\WINWORD.EXE" /n %MS Office Macro Beacon File% /o ""
1	Process Create	Image=C:\Windows\SysWOW64\rundll32.exe, CommandLine=C:\Windows\SysWOW64\rundll32.exe, ParentImage=C:\Program Files\Microsoft Office\root\Office16\WINWORD.EXE, ParentCommandLine="C:\Program Files\Microsoft Office\Root\Office16\WINWORD.EXE" /n %MS Office Macro Beacon File% /o ""
10	Process accessed	SourceImage=C:\Program Files\Microsoft Office\Root\Office16\WINWORD.EXE, TargetImage=C:\Windows\SysWOW64\rundll32.exe, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044 ~
8	CreateRemoteTread	SourceImage=C:\Program Files\Microsoft Office\root\Office16\WINWORD.EXE, TargetImage=C:\Windows\SysWOW64\rundll32.exe, StartAddress=%Random%, StartModule=-, StartFunction=-

3) PowerShell Script

PowerShell 명령을 통해 시스템 명령을 직접 실행하는 페이로드 유형으로, 공격자는 PowerShell을 이용해 C2 서버와 통신을 설정하거나 추가 페이로드를 다운로드하기 위해 사용한다. PowerShell Script 내부 코드 구조에서 Base64 Encoding과 XOR 연산을 수행하는 점을 통해 식별 가능하다. PowerShell Script 유형의 Beacon이 실행되었을 때, powershell.exe 프로세스가 생성되면서 PowerShell Script 형태의 Beacon이 실행된다. 실행 방법에 따라 부모 프로세스는 변경되는 패턴이 확인되었다.

[표 23] PowerShell Script로 생성된 Beacon 실행 시 발생하는 프로세스 생성 이벤트

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe, CommandLine=powershell %PowerShell Script Beacon File%

4) Windows EXE

실행 파일 형태(.exe)의 페이로드 유형으로, 시스템에서 직접 실행해야 한다. 해당 페이로드 유형은 공격자가 가장 다양한 방법을 통해 실행할 수 있다. Windows EXE 유형의 Beacon이 실행되었을 때, Windows EXE 형태의 Beacon 프로세스가 생성된다. 실행 방법에 따라 부모 프로세스는 변경되는 패턴이 확인되었다.

[표 24] Windows EXE로 생성된 Beacon 실행 시 발생하는 프로세스 생성 이벤트

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=%Windows EXE Beacon File%, CommandLine=%Windows EXE Beacon File%

5) Windows Service EXE

실행 파일 형태(.exe)의 페이로드 유형으로, 독립적으로 실행해서는 동작되지 않고 서비스를 통해 실행해야 정상적으로 동작한다. Windows Service EXE 유형의 Beacon이 실행되었을 때, services.exe 프로세스를 통해 Windows Service 형태의 Beacon 프로세스가 생성되고, Beacon 프로세스가 인자 값이 존재하지 않는 rundll32.exe 프로세스를 생성한 뒤 종료하는 패턴이 확인되었다.

[표 25] Windows Service EXE로 생성된 Beacon 실행 시 발생하는 이벤트

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=%Windows Service EXE Beacon File%, ParentImage=C:\Windows\System32\services.exe
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\System32\rundll32.exe, ParentImage=%Windows Service EXE Beacon File%
5	Process terminated	Image=%Windows Service EXE Beacon File%

6) Windows DLL

동적으로 호출하는 라이브러리 파일(DLL) 형태의 페이로드 유형으로, 공격자는 주로 프로세스 인젝션(Process Injction) 공격을 통해 활용한다. Windows DLL 유형의 Beacon이 실행되었을 때, rundll32.exe 프로세스가 Windows DLL 형태의 Beacon 프로세스가 생성된다.

[표 26] Windows DLL로 생성된 Beacon 실행 시 발생하는 이벤트

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=rundll32 %Windows DLL Beacon File%,Start

6.2.1.3. Beacon 페이로드 동작 방식

Beacon 페이로드는 총 2가지 방식(Stager, Stageless)로 동작한다. 해당 연구에서는 2가지 동작 방식의 Beacon 페이로드를 실행했을 때의 이벤트를 분석했다. 본 연구를 위해 Beacon의 Listener 프로토콜은 HTTPS로 Beacon페이로드 유형은 Windows EXE로 설정했다. 동작 방식은 Stageless로 설정했다. 분석 결과, Beacon이 실행되는 과정에서 네트워크 연결 이벤트의 발생 횟수에서 차이가 존재했다.

[표 27] Beacon 페이로드 동작 방식에 따른 Beacon 통신의 차이점

동작 구분	연구 결과
Stager	외부 Team Server에서 Beacon 데이터를 다운로드 받아 메모리 상에서 실행하기 때문에 네트워크 연결 이벤트가 두 번 발생한다. 이러한 통신 특성으로 HTTP(S) Listener 형태만 생성이 가능하다. 또한, Beacon에 실제 데이터가 존재하지 않기 때문에 19 ~ 20KB의 매우 작은 크기로 구성되어 있다.
Stageless	Beacon 내부에 데이터가 포함되어 실행되기 때문에 네트워크 연결 이벤트가 한 번만 발생한다. 외부 Team Server와 통신할 필요가 없기 때문에 모든 Listener 형태에서 생성이 가능하다. 또한, Beacon에 실제 데이터가 존재하기 때문에 300 ~ 400KB의 Stager 방식보다 큰 크기로 구성되어 있다.

각 Beacon 페이로드 동작 방식에 해당하는 실행 방식은 다음과 같다.

1) Stager

Stager Beacon은 메모리에 작은 코드를 먼저 로드한 뒤, 네트워크를 통해 완전한 페이로드를 다운로드하는 방식으로 동작한다. 작은 크기로 탐지를 회피하고, 감염된 호스트에서 외부 페이로드를 가져와 실행한다. 이러한 방식으로 인해 HTTP(S) Listener에서만 생성이 가능하고, 3개의 Beacon 유형(Windows EXE, Windows Service, DLL)으로만 생성이 가능하다. Stager 방식의 Beacon이 실행되었을 때, Beacon 프로세스에 Beacon을 실행한 후 네트워크 연결 이벤트가 두 번 연속으로 발생하는 패턴이 확인되었으며 SourcePort가 1씩 증가했다.

[표 28] Stager 방식의 Beacon 실행 시 발생하는 네트워크 연결 이벤트

이벤트 ID	이벤트 유형	내용
3	Network connection	Image=%Beacon File%, SourceIp=%Host IP%, SourcePort=%N%, DestinationIp=%Team Server IP%, DestinationPort=443, DestinationPortName=https
3	Network connection	Image=%Beacon File%, SourceIp=%Host IP%, SourcePort=%N+1%, DestinationIp=%Team Server IP%, DestinationPort=443, DestinationPortName=https

2) Stageless

Stageless Beacon은 모든 명령어와 기능을 포함한 완전한 페이로드로, 처음부터 전송되어 실행된다. 외부 서버와 추가 통신 없이 즉시 실행 가능해 빠르게 공격을 시작할 수 있다는 장점이 있다. 페이로드가 한 번에 전송되기 때문에 네트워크에서의 추가 트래픽을 줄이고 탐지 가능성을 낮출 수 있다. 이러한 방식으로 인해 모든 Listener 형태로 생성이 가능하고, 5개의 Beacon 유형(PowerShell, Windows EXE, Windows Service EXE, DLL, Raw)으로 생성이 가능하다.

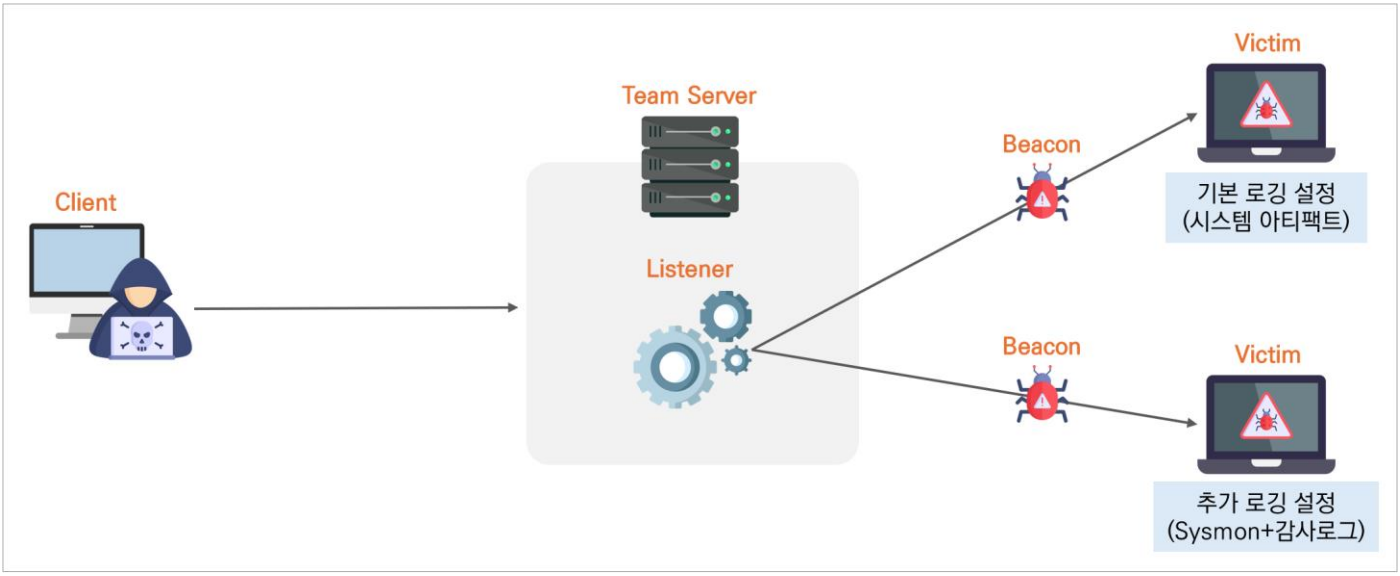
Stageless 방식의 Beacon이 실행되었을 때, Beacon 프로세스에서 Beacon을 실행한 후 네트워크 연결 이벤트가 한 번만 발생하는 패턴이 확인되었다.

[표 29] Stageless 방식의 Beacon 실행 시 발생하는 네트워크 연결 이벤트

이벤트 ID	이벤트 유형	내용
3	Network connection	Image=%Beacon File%, SourceIp=%Host IP%, SourcePort=%N%, DestinationIp=%Team Server IP%, DestinationPort=443, DestinationPortName=https

6.2.2. 기본 지원 기능별 흔적 분석

Cobalt Strike 버전 4.9.1을 기준으로 기본으로 제공되는 명령은 98개이며, 다양한 전술에서 공격이 가능하다. 고도화된 공격 기법을 수행하기 위해서는 Aggressor Script를 사용해야 하나 일반적인 침해사고에서 많이 활용되는 기법은 기본 모듈을 통해 충분히 수행 가능하다. Cobalt Strike의 대부분의 기능이 메모리 내에서 동작하기 때문에, 기본 로그 설정만으로는 명령어 실행 흔적을 식별하기 어려운 한계가 존재한다. 본 연구에서는 Cobalt Strike에서 기본으로 지원하는 명령을 직접 실행해 각 명령이 시스템에 남기는 흔적을 분석했으며, Cobalt Strike의 동작 방식을 기반으로 한 흔적 분석 방안을 보완하기 위해 추가 로그 설정이 적용된 시스템(Sysmon 로그, 감사로그)과 기본 로깅 설정이 적용된 시스템에서 각각 98개의 명령어를 실행했다.



[그림 13] 기본 지원 기능별 흔적 테스트 환경 구성도

1) 추가 로그 설정 사항

Sysmon로그 설정은 시스템에서 Cobalt Strike가 동작될 때의 패턴과 아티팩트를 확인하기 위해 관련된 모든 이벤트들을 로깅하도록 설정했으며, 감사 로그는 한국인터넷진흥원에서 권장하는 로그 설정 가이드(한눈에 보는 로그 설정 노트)를 기준으로 설정했다. 아래 표는 추가 로깅을 설정한 로그 목록이다.

[표 30] 로그 추가 설정 목록

로그 구분	설정 로그 목록
Sysmon	Process Create, NetworkConnect, ProcessTerminate, DriverLoad, ImageLoad, CreateRemoteThread, ProcessAccess, FileCreate, RegistryEvent, PipeEvent, WmiEvent, FileDelete, ProcessTampering

로그 구분	설정 로그 목록
감사로그	계정 로그온 (자격 증명 유효성 검사, 기타 계정 로그온 이벤트 감사), 계정 관리 (컴퓨터 계정 관리 감사, 사용자 계정 관리 감사) 세부 추적 (프로세스 만들기 감사, RPC 이벤트 감사) 로그온/로그오프 (로그오프 감사, 로그온 감사, 기타 로그온/로그오프 이벤트 감사, 특수 로그온 감사) 개체 액세스 (파일 공유 감사, 기타 개체 액세스 이벤트 감사) 시스템 (보안 상태 변경 감사)

2) 기본 지원 기능 분류

Cobalt Strike에서 기본으로 제공하는 98개의 명령을 수행하는 기능과 공격 가능한 전술을 기준으로 12개의 항목으로 분류했으며, 각 기능에 대한 카테고리는 아래와 같다.

[표 31] 기본으로 지원하는 명령 카테고리 구분

번호	명령 카테고리	기준
1	Beacon 제어 및 관리	Beacon이 C2 서버와 통신하는 방법과 행동을 제어하는 명령으로 구성
2	프로세스 제어	대상 시스템의 프로세스를 탐색, 생성, 종료 또는 인젝션할 수 있는 명령으로 구성
3	파일 및 디렉터리 작업	대상 시스템의 파일 시스템에 접근하고 조작하는 명령으로 구성
4	네트워크 탐색 및 이동	대상 시스템의 네트워크 정보 수집 및 네트워크 내 이동을 수행하는 명령으로 구성
5	권한 상승 및 크리덴셜 수집	대상 시스템에서 권한을 상승시키거나 인증 정보(크리덴셜)을 수집하는데 사용되는 명령으로 구성
6	사용자 활동 모니터링	대상 시스템에서 사용자의 활동을 모니터링하는 명령어로 구성
7	통신 및 데이터 암호화	Beacon과 C2 서버 간의 통신에 사용되는 데이터 암호화 설정, 통신 방식 설정에 사용하는 명령으로 구성
8	명령 및 스크립트 실행	대상 시스템에서 임의의 명령을 실행하거나 스크립트를 로드하는 데 사용되는 명령으로 구성
9	네트워크 트래픽 조작 및 우회	Beacon을 통해 네트워크 트래픽을 조작하고 우회하는 명령으로 구성
10	디버깅 및 진단	Beacon의 세션 상태를 파악하거나 작업 중인 명령어를 관리하는 데 사용되는 명령으로 구성
11	시스템 정보 수집	대상 시스템의 상태를 파악하고 추가 공격에 필요한 정보를 수집하는 데 사용되는 명령으로 구성
12	시스템 환경 제어	대상 시스템의 환경 설정이나 레지스트리를 조작해 시스템의 환경을 제어하는 데 사용되는 명령으로 구성

각 명령 카테고리 별로 해당되는 명령어 목록과 설명은 다음과 같다.

[표 32] 명령 카테고리별 기본 지원 명령 목록

번호	명령 카테고리	명령어	설명
1	Beacon 제어 및 관리	checkin	Beacon이 Sleep 시간을 대기하지 않고 즉시 Team Server와 통신
2		link	SMB Beacon을 연결하고 제어권을 획득
3		mode dns	DNS Beacon의 데이터 채널을 DNS A로 설정
4		mode dns-txt	DNS Beacon의 데이터 채널을 DNS TXT로 설정
5		mode dns6	DNS Beacon의 데이터 채널을 DNS AAAA로 설정
6		sleep	Beacon과 Team Server 통신 주기 설정
7		unlink	SMB Beacon이 연결된 세션 해제
8	프로세스 제어	argue	프로세스 환경 블록에서 메모리 구조를 조작해 실행 중인 프로세스의 명령줄 인수를 스푸핑
9		blockdlls	프로세스 공간에서 Microsoft DLL이 아닌 DLL을 차단하는 바이너리 서명 정책을 활성화/비활성화
10		dllinject	대상 프로세스에 Reflective DLL 파일 주입
11		dllload	지정한 원격 프로세스에 LoadLibrary 함수를 통해 DLL 로드
12		inject	대상 프로세스에 Beacon 페이로드 주입
13		kill	지정한 프로세스 종료
14		ppid	Beacon에서 실행하는 프로세스의 부모 프로세스 지정
15		process_browser	대상 시스템의 Process Browser 탭 실행
16		ps	대상 시스템의 프로세스 목록 출력
17		psinject	대상 프로세스에 .NET 런타임을 주입해 파워셸 명령 실행
18		shininject	대상 프로세스에 ShellCode를 주입해 실행
19		shspawn	대상 프로세스를 실행하고 ShellCode를 주입해 실행
20	파일 및 디렉터리 작업	cd	Beacon 프로세스의 작업 디렉터리 전환
21		cp	대상 시스템 내부의 파일을 지정된 경로로 복사
22		download	대상 시스템의 파일을 Team Server에 다운로드

번호	명령 카테고리	명령어	설명
23	파일 및 디렉터리 작업	drives	대상 시스템의 드라이브 목록 출력
24		file_browser	현재 작업 디렉터리를 기준으로 File Browser 탭 실행
25		ls	디렉터리의 파일 및 폴더 목록 출력
26		mkdir	대상 시스템에 디렉터리 작성
27		mv	대상 시스템 내부의 파일을 지정된 경로로 이동
28		rm	대상 시스템 내부의 파일 및 디렉터리 삭제
29		timestomp	대상 파일의 생성, 수정, 접근 시간을 지정한 파일과 동일하게 변경
30		upload	공격자 시스템의 파일을 대상 시스템으로 업로드
31	네트워크 탐색 및 이동	jump	지정한 익스플로잇 방식을 사용해 원격 대상에 세션 생성 (익스플로잇 방식: psexec, psexec64, psexec_psh, winrm, winrm64)
32		net	네트워크 및 호스트 정보를 수집하는 데 사용할 수 있는 net 명령어 사용
33		portscan	프로세스에 Portscan 도구를 주입해 네트워크에 열려 있는 서비스 포트 스캐닝
34		remote-exec	지정한 익스플로잇 방식을 사용해 원격 대상에 명령 실행 (익스플로잇 방식: psexec, winrm, wmi)
35		rportfwd	특정 포트를 열고 해당 포트로 들어오는 트래픽을 전달하는 리버스 포트 포워드 설정
36		rportfwd_local	트래픽을 클라이언트를 통해 전달하는 리버스 포트 포워드 설정
37		ssh	SSH 세션 설정을 사용해 SSH 세션 생성
38		ssh-key	SSH 세션 키를 사용해 SSH 세션 생성
39	권한 상승 및 크리덴셜 수집	chromedump	Mimikatz를 활용해 Google Chrome에 저장된 자격증명 획득
40		dcsync	Mimikatz를 활용해 AD Domain Controller에서 도메인 사용자의 NTLM 비밀번호 해시 획득
41		elevate	지정한 익스플로잇 방식을 사용해 권한 상승 시도 (익스플로잇 방식: svc-exe, uac-token-duplication)
42		getprivs	대상 시스템에서 사용 가능한 권한 목록 출력
43		getsystem	Beacon의 세션 권한을 SYSTEM으로 상승
44		hashdump	지정된 프로세스에 hashdump 도구를 주입해 계정들의 NTLM 해시 획득

번호	명령 카테고리	명령어	설명
45	권한 상승 및 크리덴셜 수집	kerberos_ccache_use	Kerberos CCache 파일을 사용해 자격 증명을 로드 (CCache 파일을 통해 해당 자격 증명으로 Kerberos 티켓을 발급받아 사용)
46		kerberos_ticket_purge	대상 시스템에서 모든 Kerberos 티켓 삭제 (기존의 티켓을 정리하거나 새로운 티켓을 발급받기 위한 준비 가능)
47		kerberos_ticket_use	지정된 Kerberos 티켓을 사용해 세션에 인증 (티켓을 직접적으로 세션에 적용해 현재 세션을 해당 티켓의 권한으로 설정 가능)
48		logonpasswords	Mimikatz를 이용해 시스템의 평문 자격증명과 NTLM 해시 획득
49		make_token	특정 사용자 자격 증명으로 액세스 토큰 생성
50		mimikatz	내장되어 있는 Mimikatz 명령어 사용
51		pth	Mimikatz를 이용해 사용자 권한을 사칭하는 Pass-The-Hash 공격 수행
52		rev2self	현재 프로세스가 사용 중인 임시 액세스 토큰을 원래 사용자 액세스 토큰으로 전환
53		steal_token	대상 프로세스의 액세스 토큰 탈취
54	사용자 활동 모니터링	clipboard	대상의 텍스트 클립보드의 내용을 획득
55		desktop	대상 시스템에 VNC 서버를 주입하고 연결
56		keylogger	대상 시스템에서 키 입력을 기록해 데이터베이스에 저장
57		printscreen	PrintScr 방식을 통해 대상 시스템의 단일 스크린샷 출력
58		screenshot	대상 프로세스에 ScreenShot 도구를 주입해 스크린샷 출력
59		screenwatch	대상 프로세스에 ScreenWatch 도구를 주입해 주기적으로 스크린샷 출력
60	통신 및 데이터 암호화	connect	Beacon TCP에 연결해 제어권을 획득
61		covertvpn	대상 시스템에 Covert VPN 배포 및 실행
62		data-store	Beacon 객체 파일(BOF, Beacon Object File)과 .NET 어셈블리를 Beacon의 메모리에 저장
63		socks	SOCKS 서버를 시작해 Beacon이 트래픽을 프록시할 수 있도록 설정
64		spawn	x86 또는 x64 아키텍처의 새로운 프로세스를 생성해 지정된 리스너를 위한 Shell Code 주입
65		spawnas	특정 사용자 계정의 자격 증명(도메인, 사용자, 이름, 비밀번호)을 사용해 페이로드 실행
66		spawnto	Beacon이 Shell Code를 주입할 프로세스 지정

번호	명령 카테고리	명령어	설명
67	통신 및 데이터 암호화	spawnu	지정된 프로세스 ID(PID)를 부모 프로세스로 사용해 새로운 세션 생성
68		spunnel	에이전트(Agent)를 스폰(spawn)하고 리버스 포트 포워딩 터널 생성
69		spunnel_local	에이전트(Agent)를 생성하고 리버스 포트 포워딩 설정
70		token-store	프로세스의 액세스 토큰을 탈취해 저장소에 저장
71	명령 및 스크립트 실행	execute	출력 없이 백그라운드에서 프로그램을 실행
72		execute-assembly	대상 시스템에서 .NET 어셈블리 실행
73		inline-execute	대상 시스템에서 Beacon 객체 파일(BOF) 실행
74		powerpick	.NET 라이브러리와 어셈블리만 사용해 PowerShell 프로세스 없이 파워셸 명령어 실행
75		powershell	PowerShell 프로세스를 통해 명령어 실행
76		powershell-import	대상 시스템에 PowerShell 스크립트 실행
77		run	CMD 프로세스 없이 Win32 API 호출을 사용해 OS 명령어 실행
78		runas	다른 사용자 계정으로 프로그램 실행
79		runasadmin	Beacon이 실행 중인 사용자 세션에서 특정 명령을 관리자 권한으로 실행
80		runu	프로세스 실행 시, 다른 또는 이미 실행 중인 프로세스의 부모 프로세스를 지정해 실행
81		shell	Windows 명령 프롬프트(CMD)를 사용해 명령어 실행
82	네트워크 트래픽 조작 및 우회	browserpivot	지정된 프로세스로 브라우저 피벗 설정
83	디버깅 및 진단	!	이전에 실행된 명령어를 다시 실행
84		cancel	현재 진행 중인 파일 다운로드를 취소
85		clear	Beacon의 명령어 대기열 지움
86		downloads	현재 진행 중인 파일 다운로드 목록 출력
87		exit	Beacon 세션(프로세스) 종료
88		help	명령어에 대한 도움말 출력
89		history	현재 세션에서 입력한 명령어 히스토리 목록 출력

번호	명령 카테고리	명령어	설명
90	디버깅 및 진단	jobkill	지속해서 수행 중인 Beacon 작업 중지
91		jobs	지속해서 수행 중인 Beacon 작업 목록 출력
92		note	Beacon에 메모 작성
93		setenv	Beacon 내부의 환경 변수 설정
94		syscall-method	시스템 호출 방법을 변경해 AV 및 EDR 탐지 우회
95		windows_error_code	Windows 에러 코드 번호에 대한 설명 출력
96	시스템 정보 수집	getuid	현재 세션과 연결된 계정명(Hostname/Username) 출력
97		pwd	Beacon의 현재 작업 디렉토리 출력
98	시스템 환경 제어	reg	대상 시스템의 레지스트리 정보 출력

3) 기본 지원 기능 별 흔적 분석 결과

총 39개의 명령에서 Cobalt Strike 사용과 관련된 흔적이 확인되었으며, 명령 카테고리 별 흔적이 발견된 명령은 다음과 같다.

[표 33] 기본 지원 명령 카테고리 별 흔적이 확인된 명령어 목록

명령 카테고리	명령어
Beacon 제어 및 관리	link
프로세스 제어	dllinject, dllload, inject, psinject, shinject, shspawn
파일 및 디렉터리 작업	흔적이 확인된 명령 없음
네트워크 탐색 및 이동	jump, net, portscan, remote-exec
권한 상승 및 크리덴셜 수집	chromedump, dcsync, elevate, hashdump, logonpasswords, mimikatz, pth
사용자 활동 모니터링	desktop, keylogger, printscreen, screenshot, screenwatch
통신 및 데이터 암호화	covertvpn, socks, spawn, spawnas, spawnu, spunnel, spunnel_local
명령 및 스크립트 실행	esecute, execute-assembly, powerpick, powershell, powershell-import, runas, runu, shell
네트워크 트래픽 조작 및 우회	browserpivot
디버깅 및 진단	흔적이 확인된 명령 없음
시스템 정보 수집	흔적이 확인된 명령 없음
시스템 환경 제어	흔적이 확인된 명령 없음

다음은 명령 카테고리 별로 흔적이 확인된 명령에 대한 상세 내용을 정리한 것이다. 흔적이 확인된 명령은 노란색 음영으로 표시했으며, 상세 내용 중 탐지 방안으로 활용할 수 있는 이벤트는 빨간색으로 표시했다.

6.2.2.1. Beacon 제어 및 관리

Cobalt Strike에서 지원하는 기본 기능 중 Beacon 제어 및 관리에 포함되는 명령은 7개이며, 이 중 Cobalt Strike를 통한 행위로 특정할 수 있는 흔적이 확인된 명령은 1개이다.

[표 34] Beacon 제어 및 관리 카테고리 명령 별 흔적 존재 여부

번호	명령어	시스템 아티팩트 내 흔적	추가 설정된 로그 내 흔적
1	checkin	존재하지 않음	존재하지 않음
2	link	Volatile, EventLog	Sysmon Event
3	mode dns	존재하지 않음	존재하지 않음
4	mode dns-txt	존재하지 않음	존재하지 않음
5	mode dns6	존재하지 않음	존재하지 않음
6	sleep	존재하지 않음	존재하지 않음
7	unlink	존재하지 않음	존재하지 않음

명령 별로 확인된 흔적에 대한 내용은 다음과 같다.

1) link

- **추가 설정 로그 (Sysmon Event)**

SMB Beacon을 연결하기 위해 link 명령어를 사용하면 System 프로세스가 SMB Beacon이 실행됐을 때 생성된 Pipe에 연결한다. 이후 link 명령을 수행한 대상(Parent Host)에 SMB(445) 네트워크 연결 이벤트가 발생한다. Cobalt Strike에서 SMB Listener를 생성하면 Pipe Name의 기본 값은 '\msagent_88'로 설정되어 있으며, 해당 Pipe Name은 임의로 수정 가능하다.

[표 35] link 명령 실행에 따른 Sysmon 이벤트 로그 패턴

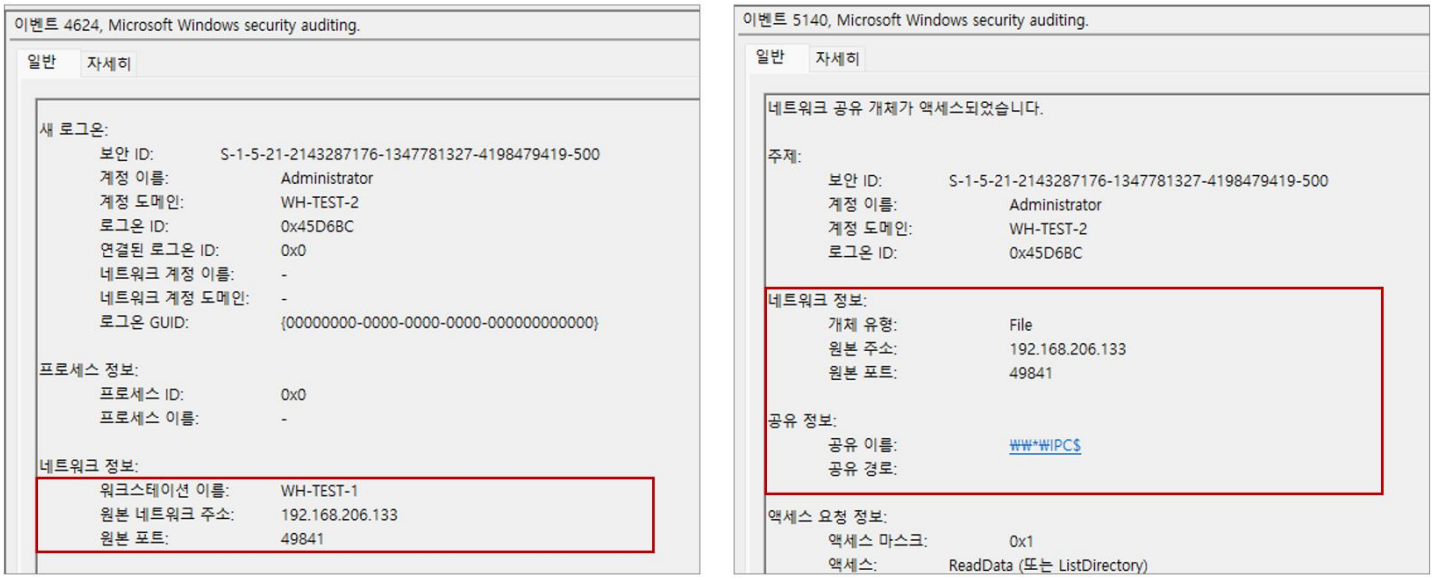
이벤트 ID	이벤트 유형	내용
18	Pipe Connected	PipeName=\msagent_88, Image=System
3	Network connection	Image=System, SourceIp=%Parent Host IP%, SourcePort=%N%, DestinationIp=%Host IP%, DestinationPort=445, DestinationPortName=microsoft-ds

- **시스템 아티팩트 (EventLog)**

이벤트 로그에서는 link 명령을 통해 SMB를 연결한 대상(Parent Host)이 SMB 포트로 연결이 수행된 대상(Child Host)에 네트워크 로그온한 이벤트가 발생한 후, 네트워크 공유 개체(*\IPC$)에 액세스한 이벤트가 확인된다. Parent Host와 Child Host가 SMB 포트로 연결된 상태에서 명령을 실행하는 경우, 별도로 시스템에 남는 흔적은 확인되지 않았다.

[표 36] link 명령 실행에 따른 이벤트 로그 패턴(Security.evtx)

이벤트 ID	이벤트 유형	내용
4624	계정 로그온 (네트워크)	TargetUserName= %Connected User Name%, TargetLogonId= %Connected Logon ID%, Logon Type= 3, IpAddress= %Parent Host Ip Address%
5140	네트워크 공유 개체 접근	SubjectUserName= %Connected User Name%, SubjectLogonId= %Connected Logon ID%, ObjectType= File, IpAddress: %Parent Host Ip Address%, ShareName= *\IPC$



[그림 14] link 명령 실행에 따른 이벤트 로그 예시(Security.evtx)

6.2.2.2. 프로세스 제어

Cobalt Strike에서 지원하는 기본 기능 중 프로세스 제어에 포함되는 명령은 12개이며, 이 중 Cobalt Strike를 통한 행위로 특정할 수 있는 흔적이 확인된 명령은 6개이다.

[표 37] 프로세스 제어 카테고리 명령 별 흔적 존재 여부

번호	명령어	시스템 아티팩트 내 흔적	추가 설정된 로그 내 흔적
1	argue	존재하지 않음	존재하지 않음
2	blockdlls	존재하지 않음	존재하지 않음
3	dllinject	존재하지 않음	Sysmon Event
4	dllload	존재하지 않음	Sysmon Event
5	inject	존재하지 않음	Sysmon Event
6	kill	존재하지 않음	존재하지 않음
7	ppid	존재하지 않음	존재하지 않음
8	process_browser	존재하지 않음	존재하지 않음
9	ps	존재하지 않음	존재하지 않음
10	psinject	존재하지 않음	Sysmon Event
11	shininject	존재하지 않음	Sysmon Event
12	shspawn	존재하지 않음	Sysmon Event

명령 별로 확인된 흔적에 대한 내용은 다음과 같다.

1) dllinject

- **추가 설정 로그 (Sysmon Event)**

Beacon 프로세스는 dllinject 명령을 수행할 대상 프로세스에 원격 스레드를 생성한다.

[표 38] dllinject 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
8	CreateRemoteTread	SourceImage=%Beacon File%, TargetImage=%Target Process%, StartAddress={Random}, StartModule=-, StartFunction=

2) dllload

- **추가 설정 로그 (Sysmon Event)**

Beacon 프로세스는 dllload 명령을 수행할 대상 프로세스에 KERNEL32.DLL 모듈의 LoadLibraryA() 함수를 사용해 원격 스레드를 생성한다.

[표 39] dllload 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
8	CreateRemoteTread	SourceImage=%Beacon File%, TargetImage=%Target Process%, StartAddress={Random}, StartModule=C:\Windows\System32\KERNEL32.DLL, StartFunction=LoadLibraryA

3) inject

- **추가 설정 로그 (Sysmon Event)**

Beacon 프로세스는 대상 프로세스에 원격 스레드를 생성한다. 이후 대상 프로세스에서 Beacon 실행 패턴처럼 이미지(DLL) 로드, 레지스트리 이벤트가 발생되는 패턴이 확인되지만 Pipe 이벤트는 발생하지 않는다.

[표 40] inject 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
8	CreateRemoteTread	SourceImage=%Beacon File%, TargetImage=%Target Process%, StartAddress=%Random%, StartModule=-, StartFunction=-
-	-	이후 Target Process에서 Beacon 실행 패턴처럼 이미지(DLL) 로드, 레지스트리 이벤트가 발생되나, Pipe 이벤트는 발생되지 않음

4) psinject

- **추가 설정 로그 (Sysmon Event)**

Beacon 프로세스는 사용자에게 입력 받은 Command를 PID에 원격 스레드를 생성해 실행한다.

[표 41] psinject 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
8	CreateRemoteThread	,SourceImage=%Beacon File%, TargetImage=%Target Process%, StartAddress=%[Random]%, StartModule=-, StartFunction=-
17	Pipe Created	PipeName=\postex_*, Image=C:\Windows\System32\rundll32.exe
18	Pipe Connected	PipeName=\postex_*, Image=%Beacon File%

5) shinject

- **추가 설정 로그 (Sysmon Event)**

Beacon 프로세스가 원격 프로세스에 페이로드 주입을 위해 원격 스레드를 생성한다. 이때 발생하는 프로세스 접근 이벤트는 페이로드에서 프로그램을 실행하는 경우에만 발생한다.

[표 42] shinject 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
8	CreateRemoteThread	SourceImage=%Beacon File%,TargetImage=%Target Process%, StartAddress=%[Random]%
10	Process accessed	SourceImage=%Target Process%, TargetImage=%Payload Process%, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72| C:\Windows\System32\KERNELBASE.dll+48a8a| C:\Windows\System32\KERNELBASE.dll+48616| C:\Windows\System32\KERNEL32.DLL+5f34b|UNKNOWN(%[Random]%)

6) shspwan

- **추가 설정 로그 (Sysmon Event)**

Beacon 프로세스가 페이로드 주입을 위해 기본 설정 값(프로세스 생성)인 rundll32.exe를 실행하고 접근하는 이벤트 패턴이 확인된다.

[표 43] shspwan 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\system32\rundll32.exe, ParentImage=%Beacon File%
10	Process accessed	SourceImage=%Beacon File%, TargetImage=C:\Windows\system32\rundll32.exe, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72| C:\Windows\System32\KERNELBASE.dll+48a8a|C:\Windows\System32\KERNELBASE.dll+48616| C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(%[Random]%)

6.2.2.3. 파일 및 디렉터리 작업

Cobalt Strike에서 지원하는 기본 기능 중 파일 및 디렉터리 작업에 포함되는 명령은 11개이며, 해당 명령들 모두 Cobalt Strike를 통한 행위로 특정할 수 있는 흔적은 발견되지 않았다.

[표 44] 파일 및 디렉터리 작업 카테고리 명령 별 흔적 존재 여부

번호	명령어	시스템 아티팩트 내 흔적	추가 설정된 로그 내 흔적
1	cd	존재하지 않음	존재하지 않음
2	cp	존재하지 않음	존재하지 않음
3	download	존재하지 않음	존재하지 않음
4	drives	존재하지 않음	존재하지 않음
5	file_browser	존재하지 않음	존재하지 않음
6	ls	존재하지 않음	존재하지 않음
7	mkdir	존재하지 않음	존재하지 않음
8	mv	존재하지 않음	존재하지 않음
9	rm	존재하지 않음	존재하지 않음
10	timestomp	존재하지 않음	존재하지 않음
11	upload	존재하지 않음	존재하지 않음

6.2.2.4. 네트워크 탐색 및 이동

Cobalt Strike에서 지원하는 기본 기능 중 네트워크 탐색 및 이동에 포함되는 명령은 8개이며, 이 중 Cobalt Strike를 통한 행위로 특정할 수 있는 흔적이 확인된 명령은 4개이다.

[표 45] 네트워크 탐색 및 이동 카테고리 명령 별 흔적 존재 여부

번호	명령어	시스템 아티팩트 내 흔적	추가 설정된 로그 내 흔적
1	jump	Metadata, EventLog, Registry	Sysmon Event
2	net	존재하지 않음	Sysmon Event
3	portscan	존재하지 않음	Sysmon Event
4	remote-exec	Metadata, EventLog, Registry	Sysmon Event
5	rportfwd	존재하지 않음	존재하지 않음
6	rportfwd_local	존재하지 않음	존재하지 않음
7	ssh	테스트 대상 아님	테스트 대상 아님
8	ssh-key	테스트 대상 아님	테스트 대상 아님

명령 별로 확인된 흔적에 대한 내용은 다음과 같다.

1) jump

- **추가 설정 로그 (Sysmon Event)**

테스트를 수행했던 익스플로잇은 3개(psexec64, psexec_psh, winrm64)이며, 각 익스플로잇 별로 구분한 내용은 다음과 같다.

[표 46] jump 명령을 통해 테스트한 3개의 익스플로잇(psexec64, psexec_psh, winrm64) Sysmon 이벤트 패턴

익스플로잇 구분	연구 결과
psexec64	System 프로세스를 통해 ADMIN$(C:\Windows) 경로에 %Random%.exe 파일이 생성한다. 생성한 파일을 서비스로 등록하고 실행하며, 이후 프로세스 변조 이벤트가 발생된다. %Random%.exe 프로세스는 명명된 Pipe를 생성 및 연결하고, 인자 값이 존재하지 않는 rundll32.exe 프로세스를 생성한다. 이후 rundll32.exe 프로세스에서 Beacon 실행 패턴처럼 이미지(DLL) 로드, 레지스트리 이벤트가 발생되지만 Pipe 이벤트는 발생하지 않는다.
psexec_psh	cmd.exe를 통해 서비스가 설치(Encoding Code를 실행시키는 PowerShell 동작)된다. PowerShell이 동작하면서 임시 파일들이 생성 및 삭제되고, SMB(445)로 네트워크 연결 이벤트가 발생한다. 이후 powershell.exe 프로세스는 명명된 Pipe를 생성한다.
winrm64	Beacon 프로세스에서 Encoding Code를 실행하는 powershell.exe 프로세스를 생성하고 접근한다. PowerShell이 동작하면서 임시 파일들이 생성 및 삭제되고, 대상 호스트에 WinRM(5985)으로 네트워크 연결 이벤트가 발생한다. 대상 호스트에서는 svchost.exe 프로세스가 wsmprovhost.exe 프로세스를 생성하고 PowerShell이 동작하며, 이후 WinRM 관련 Pipe가 생성된다.

[표 46-1] jump psexec64 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
11	File created	Image=System, TargetFilename=C:\Windows\%Random%.exe
12	Registry – CreateKey	Image=C:\Windows\system32\services.exe, TargetObject=HKLM\System\CurrentControlSet\Services\%Random%
13	Registry – SetValue	Image=C:\Windows\system32\services.exe, TargetObject=HKLM\System\CurrentControlSet\Services\%Random%\Type, Details=DWORD (0x00000010)
13	Registry – SetValue	Image=C:\Windows\system32\services.exe, TargetObject=HKLM\System\CurrentControlSet\Services\%Random%\Start, Details=DWORD (0x00000003)
13	Registry – SetValue	Image=C:\Windows\system32\services.exe, TargetObject=HKLM\System\CurrentControlSet\Services\%Random%\ErrorControl, Details=DWORD (0x00000000)
13	Registry – SetValue	Image=C:\Windows\system32\services.exe, TargetObject=HKLM\System\CurrentControlSet\Services\%Random%\ImagePath, Details=\\%HOSTNAME%\ADMIN$\%Random%.exe

이벤트 ID	이벤트 유형	내용
13	Registry – SetValue	Image=C:\Windows\system32\services.exe, TargetObject=HKLM\System\CurrentControlSet\Services\%Random%\ObjectName, Details=LocalSystem
1	Process Create	Image=\\%HOSTNAME%\ADMIN$\%Random%.exe, ParentImage=C:\Windows\System32\services.exe
25	Process Tampering	Image=\\%HOSTNAME%\ADMIN$\%Random%.exe, Type=Image is locked for access
17	Pipe Created	PipeName=\MSSE-*-server, Image=\\%HOSTNAME%\ADMIN$\%Random%.exe
3	Network connection	Image=System, SourceIp=%Parent Host IP%, SourcePort=%N%, DestinationIp=%Host IP%, DestinationPort=445, DestinationPortName=microsoft-ds
18	Pipe Connected	PipeName=\MSSE-*-server, Image=\\%HOSTNAME%\ADMIN$\%Random%.exe
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\System32\rundll32.exe, ParentImage=\\%HOSTNAME%\ADMIN$\{Random}.exe
5	Process terminated	Image=\\%HOSTNAME%\ADMIN$\%Random%.exe
23	File Delete	Image=System, TargetFilename=C:\Windows\%Random%.exe
–	–	이후 rundll32.exe 프로세스에서 Beacon 실행 패턴처럼 이미지(DLL) 로드, 레지스트리 이벤트가 발생되나, Pipe 이벤트는 발생되지 않음

[표 46-2] jump psexec_psh 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
12	Registry – CreateKey	Image=C:\Windows\system32\services.exe, TargetObject=HKLM\System\CurrentControlSet\Services\%Random%
13	Registry – SetValue	Image=C:\Windows\system32\services.exe, TargetObject=HKLM\System\CurrentControlSet\Services\%Random%\Type, Details=DWORD (0x00000010)
13	Registry – SetValue	Image=C:\Windows\system32\services.exe, TargetObject=HKLM\System\CurrentControlSet\Services\%Random%\Start, Details=DWORD (0x00000003)
13	Registry – SetValue	Image=C:\Windows\system32\services.exe, TargetObject=HKLM\System\CurrentControlSet\Services\%Random%\ErrorControl, Details=DWORD (0x00000000)
13	Registry – SetValue	Image=C:\Windows\system32\services.exe, TargetObject=HKLM\System\CurrentControlSet\Services\%Random%\ImagePath, Details=%COMSPEC% /b /c start /b /min powershell -nop -w hidden -encodedcommand %Encoding Code%
13	Registry – SetValue	Image=C:\Windows\system32\services.exe, TargetObject=HKLM\System\CurrentControlSet\Services\%Random%\ObjectName, Details=LocalSystem

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe, CommandLine="c:\windows\syswow64\windowspowershell\v1.0\powershell.exe" -Version 5.1 -s -NoLogo -NoProfile, ParentImage=C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe, ParentCommandLine=powershell -nop -w hidden -encodedcommand %Encoding Code%
11	File created	Image=c:\windows\syswow64\windowspowershell\v1.0\powershell.exe, TargetFilename=C:\Windows\Temp__PSScriptPolicyTest_*.ps1
11	File created	Image=c:\windows\syswow64\windowspowershell\v1.0\powershell.exe, TargetFilename=C:\Windows\Temp__PSScriptPolicyTest_*.psm1
23	File Delete	Image=c:\windows\syswow64\windowspowershell\v1.0\powershell.exe, TargetFilename=C:\Windows\Temp__PSScriptPolicyTest_*.ps1
23	File Delete	Image=c:\windows\syswow64\windowspowershell\v1.0\powershell.exe, TargetFilename=C:\Windows\Temp__PSScriptPolicyTest_*.psm1
3	Network connection	Image=System, SourceIp=%Parent Host IP%, SourcePort=%N%, DestinationIp=%Host IP%, DestinationPort=445, DestinationPortName=microsoft-ds
17	Pipe Created	PipeName=\status_*, Image=C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe

[표 46-3] jump winrm64 명령 실행에 따른 Sysmon 이벤트 로그 패턴(Parent Host)

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe, CommandLine=powershell -nop -exec bypass -EncodedCommand %Encoding Code%, ParentImage=%Beacon File%
10	Process accessed	SourceImage=%Beacon File%, TargetImage=C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72|C:\Windows\System32\KERNELBASE.dll+48a8a|C:\Windows\System32\KERNELBASE.dll+48616|C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(0000000003B51019)
11	File created	Image=C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe, TargetFilename=C:\Users\Administrator\AppData\Local\Temp\1__PSScriptPolicyTest_*.ps1
11	File created	Image=C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe, TargetFilename=C:\Users\Administrator\AppData\Local\Temp\1__PSScriptPolicyTest_*.psm1
23	File Delete	Image=C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe, TargetFilename=C:\Users\Administrator\AppData\Local\Temp\1__PSScriptPolicyTest_*.ps1
23	File Delete	Image=C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe, TargetFilename=C:\Users\Administrator\AppData\Local\Temp\1__PSScriptPolicyTest_*.psm1
3	Network connection	Image=C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe, SourceIp=%Parent Host IP%, SourcePort=%N%, DestinationIp=%Target Host IP%, DestinationPort=5985

[표 46-4] jump winrm64 명령 실행에 따른 Sysmon 이벤트 로그 패턴(Child Host)

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\wsmprovhost.exe, CommandLine=C:\Windows\system32\wsmprovhost.exe -Embedding, ParentImage=C:\Windows\System32\svchost.exe, ParentCommandLine=C:\Windows\system32\svchost.exe -k DcomLaunch -p
11	File created	Image=C:\Windows\system32\wsmprovhost.exe, TargetFilename=C:\Users\Administrator\AppData\Local\Temp__PSScriptPolicyTest_*.ps1
11	File created	Image=C:\Windows\system32\wsmprovhost.exe, TargetFilename=C:\Users\Administrator\AppData\Local\Temp__PSScriptPolicyTest_*.psm1
23	File Delete	Image=C:\Windows\system32\wsmprovhost.exe, TargetFilename=C:\Users\Administrator\AppData\Local\Temp__PSScriptPolicyTest_*.ps1
23	File Delete	Image=C:\Windows\system32\wsmprovhost.exe, TargetFilename=C:\Users\Administrator\AppData\Local\Temp__PSScriptPolicyTest_*.psm1
3	Network connection	Image=System, SourceIp=%Parent Host IP%, SourcePort=%N%, DestinationIp=%Target Host IP%, DestinationPort=5985
17	Pipe Created	PipeName=\PSHost.*.DefaultAppDomain.wsmprovhost, Image=C:\Windows\System32\wsmprovhost.exe

- **시스템 아티팩트 (Metadata, EventLog, Registry)**

테스트를 수행했던 익스플로잇은 3개(psexec64, psexec_psh, winrm64)이며, 각 익스플로잇 별로 구분한 내용은 다음과 같다.

[표 47] jump 명령을 통해 테스트한 3개의 익스플로잇(psexec64, psexec_psh, winrm64) 시스템 아티팩트 결과

익스플로잇 구분	아티팩트 구분	연구 결과
psexec64	Metadata	# $UsnJrnl PsExec 연결이 성공한 경우, C:\Windows\{랜덤 파일명}.exe 파일이 생성되었다가 삭제되는 정황이 발생하며, 생성되는 파일은 설치된 서비스명과 동일하다.
	EventLog	# System.evtx PsExec연결이 성공한 경우, 서비스 설치 이벤트(Event ID: 7045)가 발생한다. 이때, 서비스 이름은 랜덤 문자열(숫자, 영어 소문자 포함) 7자리로 구성되어 있다. 서비스 파일 이름은 C:\Windows\{랜덤 파일명}.exe으로, 설치된 서비스 명과 동일하다. PsExec 콘솔에서 실행된 명령에 관련된 이벤트는 로깅되지 않는다.
	Registry	# SYSTEM 서비스 설치 키가 생성되며, ImagePath Valume를 통해 명령 구문을 확인할 수 있다.
psexec_psh	Metadata	# UsnJrnl PsExec 연결이 성공한 경우, C:\Windows\Temp 폴더에 임시 파워셸 파일이 생성된 이력이 존재한다.
	EventLog	# System.evtx PsExec 연결이 성공한 경우, 서비스 설치 이벤트(Event ID: 7045)가 존재한다. 서비스 이름은 랜덤 문자열(숫자, 영어 소문자 포함) 7자리로 구성되어 있다. 서비스 파일 이름은 encodedcomand로 인코딩된 PowerShell 명령 구문으로 구성된다. PsExec 콘솔에서 실행된 명령에 관련된 이벤트는 로깅되지 않는다.
	Registry	# SYSTEM 서비스 설치 키가 생성되며, ImagePath Valume를 통해 명령 구문을 확인할 수 있다.
winrm64	EventLog	# Microsoft-Windows-WinRM/Operational.evtx 'WSMan 셸 만드는 중' 이라는 이벤트(Event ID: 6)에서 연결된 문자열은 '%Attacker IP%/wsman?PSVersion=%PowerShell Version%입니다.'와 같이 공격자 IP 및 PowerShell 실행 흔적 확인이 가능하다. # Security.evtx 위 이벤트가 발생한 이후 '자격 증명 유효성 검사', '특수 권한 할당', '계정 로그온(Logon Type: 3)' 이벤트가 연속적으로 발생한다. 이때, 계정 로그온 이벤트에서 공격자 IP를 확인할 수 있다.

2) net

- **추가 설정 로그 (Sysmon Event)**

net 명령은 네트워크 및 호스트 정보를 수집하는데 사용할 수 있으며, [pid]나 [arch] 없이 net 명령만 실행할 경우, 임시 프로세스를 사용해 실행한다. 테스트는 2가지 방안(net 명령, net [pid] 명령)으로 수행했으며, 결과는 다음과 같다.

[표 48] net 명령 실행 결과

명령 구분	연구 결과
net	Beacon 프로세스가 net 모듈을 사용하기 위해서는 기본 설정 값(프로세스 생성)인 rundll32.exe를 실행하고 접근한다. 통신을 위해 생성 및 실행된 Pipe 이름의 형태는 '\postex_{랜덤한 문자 및 숫자 4자리}'로 구성되어 있다.
net [pid]	Beacon 프로세스가 net 모듈을 사용하기 위해 사용자가 지정한 Process에 스레드가 생성된다. 통신을 위해 생성 및 실행된 Pipe의 이름의 형태는 '\postex_{랜덤한 문자 및 숫자 4자리}'로 구성되어 있다.

[표 48-1] net 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\system32\rundll32.exe, ParentImage=%Beacon File%
7	Image Loaded	Image=C:\Windows\System32\rundll32.exe,ImageLoaded=C:\Windows\System32\netapi32.dll
10	Process accessed	SourceImage=%Beacon File%, TargetImage=C:\Windows\system32\rundll32.exe, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72|C:\Windows\System32\KERNELBASE.dll+48a8a|C:\Windows\System32\KERNELBASE.dll+48616|C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(%[Random]%)
17	Pipe Created	PipeName=\postex_*, Image=C:\Windows\System32\rundll32.exe
18	Pipe Connected	PipeName=\postex_*, Image=%Beacon File%

[표 48-2] net [pid] 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
8	CreateRemoteThread	SourceImage=%Beacon File%, TargetImage=%Target Process%, StartAddress=%[Random]%, StartModule=-, StartFunction=
17	Pipe Created	PipeName=\postex_*, Image=%Target Process%
18	Pipe Connected	PipeName=\postex_*, Image=%Beacon File%

3) portscan

- **추가 설정 로그 (Sysmon Event)**

portscan 명령 사용 시 특성 프로세스를 지정하지 않으면 기본적으로 rundll32.exe 프로세스를 사용한다. Beacon 프로세스가 인자 없는 rundll32.exe 프로세스를 생성하고 접근한다. 해당 프로세스는 명명된 Pipe를 생성하고 Beacon 프로세스가 생성된 Pipe에 연결한다. 대상 네트워크의 Port를 스캔하며, 열려져 있는 Port에 대해 네트워크 연결 이벤트가 발생한다. Sysmon에서는 Portscan 과정에서 연결이 성공(Port가 Open되어 있는)한 이력에 대해서 네트워크 연결 이벤트가 발생하는 패턴이 확인되었다.

[표 49] portscan 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\system32\rundll32.exe, ParentImage=%Beacon File%
10	Process accessed	SourceImage=%Beacon File%, TargetImage=C:\Windows\system32\rundll32.exe, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044| C:\Windows\System32\KERNELBASE.dll+4be72| C:\Windows\System32\KERNELBASE.dll+48a8a| C:\Windows\System32\KERNELBASE.dll+48616| C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(0000000000961019)
17	Pipe Created	PipeName=\postex_*, Image=C:\Windows\system32\rundll32.exe
18	Pipe Connected	PipeName=\postex_*, Image=%Beacon File%
3	Network connection	Image=C:\Windows\system32\rundll32.exe, SourceIp=%Host IP%, SourcePort=%N%, DestinationIp=%Target IP%, DestinationPort=3389
3	Network connection	Image=C:\Windows\system32\rundll32.exe, SourceIp=%Host IP%, SourcePort=%N%, DestinationIp=%Target IP%, DestinationPort=445
3	Network connection	Image=C:\Windows\system32\rundll32.exe, SourceIp=%Host IP%, SourcePort=%N%, DestinationIp=%Target IP%, DestinationPort=135
3	Network connection	Image=C:\Windows\system32\rundll32.exe, SourceIp=%Host IP%, SourcePort=%N%, DestinationIp=%Target IP%, DestinationPort=80

4) remote-exec

- **추가 설정 로그 (Sysmon Event)**

remote-exec 명령은 원격 시스템에 PsExec, WinRm, WMI를 이용해 명령을 실행할 수 있으며, 테스트는 2개의 Method(WinRM, WMI)를 사용해 진행했다.

[표 50] remote-exec 명령 실행 결과

명령 구분	연구 결과
WinRM	WinRM 세션 연결 이벤트가 로깅되며, 해당 이벤트를 통해 공격자 IP 확인이 가능하다. 또한, Event ID 6에서 'PSVersion='이 기록되면 WinRM을 통한 파워셸 명령이 실행된 것으로 판단할 수 있다. 파워셸 명령일 경우, 세션 연결 이후 "wsmprovhost.exe -Embedding" 이벤트를 Windows PowerShell.evtx에서 실행 이력을 확인할 수 있다.
WMI	WIM를 활용할 때, WMI 관련 레지스트리 키 생성 및 Beacon이 원격지로 연결되는 이벤트가 발생한다. 원격 프로세스 실행 시, WmiPrvSE.exe를 통해 실행한다.

[표 50-1] remote-exec WinRM 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe, CommandLine=powershell -nop -exec bypass -EncodedCommand %Encode Command%, ParentImage=%Beacon File%
1	Process Create	Image=C:\Windows\System32\conhost.exe, CommandLine=\??\C:\Windows\system32\conhost.exe 0xffffffff -ForceV1, ParentImage=C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe, ParentCommandLine=powershell -nop -exec bypass -EncodedCommand %Encode Command%
11	File created	Image=C:\Windows\system32\wsmprovhost.exe, TargetFilename=C:\Users\Administrator\AppData\Local\Temp__PSScriptPolicyTest_*.ps1
11	File created	Image=C:\Windows\system32\wsmprovhost.exe, TargetFilename=C:\Users\Administrator\AppData\Local\Temp__PSScriptPolicyTest_*.psm1
1	Process Create	Image=%Command Process%, CommandLine=%Command Process%, ParentImage=C:\Windows\System32\wsmprovhost.exe, ParentCommandLine=C:\Windows\system32\wsmprovhost.exe -Embedding
1	Process Create	Image=C:\Windows\System32\conhost.exe, CommandLine=\??\C:\Windows\system32\conhost.exe 0xffffffff -ForceV1, ParentImage=%Command Process%, ParentCommandLine=%Command Process%
23	File Delete	Image=C:\Windows\system32\wsmprovhost.exe, TargetFilename=C:\Users\Administrator\AppData\Local\Temp__PSScriptPolicyTest_*.ps1
23	File Delete	Image=C:\Windows\system32\wsmprovhost.exe, TargetFilename=C:\Users\Administrator\AppData\Local\Temp__PSScriptPolicyTest_*.psm1
3	Network connection	Image=System, SourceIp=%Parent Host IP%, SourcePort=%N%, DestinationIp=%Target Host IP%, DestinationPort=5985

[표 50-2] remote-exec WMI 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=%Command%, CommandLine="%Command%", ParentImage=C:\Windows\System32\wbem\WmiPrvSE.exe
3	Network connection	Image=%Beacon File%, SourceIp=%Host IP%, SourcePort=%N%, DestinationIp=%Target IP%, DestinationPort=%N%, DestinationPortName=-
12	Registry - CreateKey	Image=%Beacon File%, TargetObject=HKLM\SOFTWARE\Microsoft\Wbem\CIMOM
17	Pipe Created	PipeName=\msagent_*, Image=C:\Windows\System32\rundll32.exe
18	Pipe Connected	PipeName=\msagent_*, Image=%Beacon File%

[표 51] remote-exec 명령을 통해 테스트한 3개의 익스플로잇(psexec64, psexec_psh, winrm64) 시스템 아티팩트 결과

익스플로잇 구분	아티팩트 구분	연구 결과
psexec	Metadata	# $UsnJrnl PsExec 연결이 성공한 경우, C:\Windows\{랜덤 파일명}.exe 파일이 생성되었다가 삭제되는 정황이 발생하며, 생성되는 파일은 설치된 서비스명과 동일하다.
	EventLog	# System.evtx PsExec연결이 성공한 경우, 서비스 설치 이벤트(Event ID: 7045)가 발생한다. 이때, 서비스 이름은 랜덤 문자열(숫자, 영어 소문자 포함) 7자리로 구성되어 있다. 서비스 파일 이름은 C:\Windows\{랜덤 파일명}.exe으로, 설치된 서비스 명과 동일하다. PsExec 콘솔에서 실행된 명령에 관련된 이벤트는 로깅되지 않는다.
	Registry	# SYSTEM 서비스 설치 키가 생성되며, ImagePath Valume를 통해 명령 구문을 확인할 수 있다.
winrm	EventLog	# Microsoft-Windows-WinRM/Operational.evtx 'WSMan 셸 만드는 중' 이라는 이벤트(Event ID: 6)에서 연결된 문자열은 '%Target Computer Name%/wsman?PSVersion=%PowerShell Version%입니다.'와 같은 문자열이 확인되면 공격자 IP 및 PowerShell 실행 흔적 확인이 가능하다. # Security.evtx 위 이벤트가 발생한 이후 '자격 증명 유효성 검사', '특수 권한 할당', '계정 로그온(Logon Type: 3)' 이벤트가 연속적으로 발생한다. 이때, 계정 로그온 이벤트에서 공격자 IP를 확인할 수 있다.

6.2.2.5. 권한 상승 및 크리덴셜 수집

Cobalt Strike에서 지원하는 기본 기능 중 권한 상승 및 크리덴셜 수집에 포함되는 명령은 15개이며, 이 중 Cobalt Strike를 통한 행위로 특정할 수 있는 흔적이 확인된 명령은 7개이다.

[표 52] 권한 상승 및 크리덴셜 수집 카테고리 명령 별 흔적 존재 여부

번호	명령어	시스템 아티팩트 내 흔적	추가 설정된 로그 내 흔적
1	chromedump	존재하지 않음	Sysmon Event
2	dcsync	존재하지 않음	Sysmon Event
3	elevate	Metadata	Sysmon Event
4	getprivs	존재하지 않음	존재하지 않음
5	getsystem	존재하지 않음	존재하지 않음
6	hashdump	존재하지 않음	Sysmon Event
7	kerberos_ccache_use	테스트 대상 아님	테스트 대상 아님
8	kerberos_ticket_purge	테스트 대상 아님	테스트 대상 아님
9	kerberos_ticket_use	테스트 대상 아님	테스트 대상 아님
10	logonpasswords	존재하지 않음	Sysmon Event
11	make_token	존재하지 않음	존재하지 않음
12	mimikatz	존재하지 않음	Sysmon Event
13	pth	EventLog	Sysmon Event
14	rev2self	존재하지 않음	존재하지 않음
15	steal_token	존재하지 않음	존재하지 않음

명령 별로 확인된 흔적에 대한 내용은 다음과 같다.

1) chromedump

- **추가 설정 로그 (Sysmon Event)**

chromedump 명령어 사용 시 특정 프로세스를 지정하지 않으면 기본적으로 rundll32.exe 프로세스를 사용한다. Beacon 프로세스가 인자 값이 없는 rundll32.exe 프로세스를 생성하고 접근한다. rundll32.exe 프로세스는 명명된 Pipe를 생성하고, Beacon 프로세스가 해당 Pipe에 연결한다. 명명된 Pipe 이름의 형태는 '\postex_{랜덤한 문자 및 숫자 4자리}'로 구성된다.

[표 53] chromedump 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\system32\rundll32.exe, ParentImage=%Beacon File%
10	Process accessed	SourceImage=%Beacon File%, TargetImage=C:\Windows\system32\rundll32.exe, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72|C:\Windows\System32\KERNELBASE.dll+48a8a|C:\Windows\System32\KERNELBASE.dll+48616|C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(0000000000641019)
17	Pipe Created	PipeName=\postex_*, Image=C:\Windows\System32\rundll32.exe
18	Pipe Connected	PipeName=\postex_*, Image=%Beacon File%

2) dcsync

- **추가 설정 로그 (Sysmon Event)**

dcsync 명령어 사용 시 특정 프로세스를 지정하지 않으면 기본적으로 rundll32.exe 프로세스를 사용한다. Beacon 프로세스가 인자 값이 없는 rundll32.exe 프로세스를 생성하고 접근한다. rundll32.exe 프로세스는 명명된 Pipe를 생성하고, Beacon 프로세스가 해당 Pipe에 연결한다. 명명된 Pipe 이름의 형태는 '\postex_{랜덤한 문자 및 숫자 4자리}'로 구성된다.

[표 54] dcsync 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\system32\rundll32.exe, ParentImage=%Beacon File%
10	Process accessed	SourceImage=%Beacon File%, TargetImage=C:\Windows\system32\rundll32.exe, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72| C:\Windows\System32\KERNELBASE.dll+48a8a|C:\Windows\System32\KERNELBASE.dll+48616| C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(0000000000641019)
17	Pipe Created	PipeName=\postex_*, Image=C:\Windows\System32\rundll32.exe
18	Pipe Connected	PipeName=\postex_*, Image=%Beacon File%

3) elevate

- **추가 설정 로그 (Sysmon Event)**

elevate svc-exe 명령을 실행하면, System 프로세스를 통해 ADMIN$(C:\Windows) 경로에 %Random%.exe 파일이 생성된다. 생성한 %Random%.exe 파일을 서비스로 등록하고 실행하며, 프로세스 변조 이벤트가 발생된다. 해당 프로세스는 명명된 Pipe를 생성 및 연결하고, 인자 값이 존재하지 않는 rundll32.exe 프로세스를 생성한다. 이후, rundll32.exe 프로세스에서 Beacon 실행 패턴처럼 이미지(DLL) 로드, 레지스트리 이벤트가 발생되지만 Pipe 이벤트는 발생되지 않는다. 이벤트 패턴이 'jump psexec64' 명령어와 유사하지만 ADMIN$ 경로를 사용할 때 루프백 주소(127.0.0.1)를 사용하고, 네트워크 공유 개체 접근 이벤트에서 루프팩 주소(127.0.0.1)를 사용하는 차이가 존재한다.

[표 55] elevate 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
11	File created	Image=System, TargetFilename=C:\Windows\%Random%.exe
12	Registry – CreateKey	Image=C:\Windows\system32\services.exe, TargetObject=HKLM\System\CurrentControlSet\Services\%Random%

이벤트 ID	이벤트 유형	내용
13	Registry – SetValue	Image=C:\Windows\system32\services.exe, TargetObject=HKLM\System\CurrentControlSet\Services\%Random%\Start, Details=DWORD (0x00000003)
13	Registry – SetValue	Image=C:\Windows\system32\services.exe, TargetObject=HKLM\System\CurrentControlSet\Services\%Random%\ErrorControl, Details=DWORD (0x00000000)
13	Registry – SetValue	Image=C:\Windows\system32\services.exe, TargetObject=HKLM\System\CurrentControlSet\Services\%Random%\ImagePath, Details=\\127.0.0.1\ADMIN$\%Random%.exe
13	Registry – SetValue	Image=C:\Windows\system32\services.exe, TargetObject=HKLM\System\CurrentControlSet\Services\%Random%\ObjectName, Details=LocalSystem
1	Process Create	Image=\\127.0.0.1\ADMIN$\%Random%.exe, ParentImage=C:\Windows\System32\services.exe
25	Process Tampering	Image=\\127.0.0.1\ADMIN$\%Random%.exe, Type=Image is locked for access
17	Pipe Created	PipeName=\MSSE-*-server, Image=\\127.0.0.1\ADMIN$\%Random%.exe
18	Pipe Connected	PipeName=\MSSE-*-server, Image=\\127.0.0.1\ADMIN$\%Random%.exe
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\System32\rundll32.exe, ParentImage=\\127.0.0.1\ADMIN$\%Random%.exe
5	Process terminated	Image=\\127.0.0.1\ADMIN$\%Random%.exe
23	File Delete archived	Image=System, TargetFilename=C:\Windows\{Random}.exe

- **시스템 아티팩트 (Metadata)**

elevate 익스플로잇마다 파일을 생성하는 경로와 형태가 다르다. svc-exe 익스플로잇을 활용해 명령을 수행하면, 'C:\Windows\랜덤 파일명.exe' 파일이 생성되었다 삭제되는 패턴이 확인되며, use-token-duplication 익스플로잇을 활용해 명령을 수행하면 'C:\Windows\Temp\랜덤 파일명' 파일이 생성되었다 삭제되는 패턴이 확인된다.

4) hashdump

- **추가 설정 로그 (Sysmon Event)**

hashdump 명령 사용 시, 특정 프로세스를 지정하지 않으면 기본적으로 rundll32.exe 프로세스를 사용한다. Beacon 프로세스는 인자 값이 존재하지 않는 rundll32.exe 프로세스를 생성하고 접근한다. rundll32.exe 프로세스는 명명된 Pipe를 생성하고 Beacon 프로세스가 생성된 Pipe에 연결한다. rundll32.exe 프로세스는 hashdump를 수행하기 위해 lsass.exe 프로세스에 원격 스레드를 생성하고, 명령이 종료되면 rundll32.exe 프로세스를 종료한다. 통신을 위해 생성 및 실행된 Pipe 이름의 형태는 '\postex_{랜덤한 문자 및 숫자 4자리}'로 구성된다.

[표 56] hashdump 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\system32\rundll32.exe, ParentImage=%Beacon File%
10	Process accessed	SourceImage=%Beacon File%, TargetImage=C:\Windows\system32\rundll32.exe, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72|C:\Windows\System32\KERNELBASE.dll+48a8a|C:\Windows\System32\KERNELBASE.dll+48616|C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(0000000003B41019)
17	Pipe Created	PipeName=\postex_*, Image=C:\Windows\system32\rundll32.exe
18	Pipe Connected	PipeName=\postex_*, Image=%Beacon File%
8	CreateRemoteTread	SourceImage=C:\Windows\System32\rundll32.exe, TargetImage=C:\Windows\System32\lsass.exe, StartAddress=%Random%, StartModule=-, StartFunction=-
5	Process terminated	Image=C:\Windows\System32\rundll32.exe

5) logonpasswords

- **추가 설정 로그 (Sysmon Event)**

Beacon 프로세스는 인자 값이 존재하지 않는 rundll32.exe 프로세스를 생성하고 접근한다. rundll32.exe 프로세스는 명명된 Pipe를 생성하고 Beacon 프로세스가 생성된 Pipe에 연결한다. 명령이 종료되면 rundll32.exe 프로세스를 종료한다. 통신을 위해 생성 및 실행된 Pipe 이름의 형태는 '\postex_{랜덤한 문자 및 숫자 4자리}'로 구성된다.

[표 57] logonpasswords 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\system32\rundll32.exe, ParentImage=%Beacon File%
10	Process accessed	SourceImage=%Beacon File%, TargetImage=C:\Windows\system32\rundll32.exe, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72|C:\Windows\System32\KERNELBASE.dll+48a8a|C:\Windows\System32\KERNELBASE.dll+48616|C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(0000000003B41019)
17	Pipe Created	PipeName=\postex_*, Image=C:\Windows\system32\rundll32.exe
18	Pipe Connected	PipeName=\postex_*, Image=%Beacon File%
5	Process terminated	Image=C:\Windows\System32\rundll32.exe

6) mimikatz

- **추가 설정 로그 (Sysmon Event)**

Beacon 프로세스는 인자 값이 존재하지 않는 rundll32.exe 프로세스를 생성하고 접근한다. rundll32.exe 프로세스는 명명된 Pipe를 생성하고 Beacon 프로세스가 생성된 Pipe에 연결한다. mimikatz 명령에 따라 추가 이벤트가 발생되고, 명령이 종료되면 rundll32.exe 프로세스를 종료한다. 통신을 위해 생성 및 실행된 Pipe 이름의 형태는 '\postex_{랜덤한 문자 및 숫자 4자리}'로 구성된다.

[표 58] mimikatz 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\system32\rundll32.exe, ParentImage=%Beacon File%
10	Process accessed	SourceImage=%Beacon File%, TargetImage=C:\Windows\system32\rundll32.exe, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72|C:\Windows\System32\KERNELBASE.dll+48a8a|C:\Windows\System32\KERNELBASE.dll+48616|C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(0000000003B41019)
17	Pipe Created	PipeName=\postex_*, Image=C:\Windows\system32\rundll32.exe
18	Pipe Connected	PipeName=\postex_*, Image=%Beacon File%
5	Process terminated	Image=C:\Windows\System32\rundll32.exe

7) pth

- **추가 설정 로그 (Sysmon Event)**

Beacon 프로세스는 Pass-the-Hash 공격을 수행하며, 시스템에 LogonType 9로 접근한 후 명령 송수신을 위해 Pipe를 생성 및 연결한다.

[표 59] pth 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\system32\rundll32.exe, ParentImage=%Beacon File%
1	Process Create	Image=C:\Windows\System32\cmd.exe, CommandLine=C:\Windows\system32\cmd.exe /c echo %[Random][11]%〉\\.\pipe\%[Random][6]%, ParentImage=C:\Windows\system32\rundll32.exe
17	Pipe Created	PipeName=\postex_*, Image=C:\Windows\System32\rundll32.exe
18	Pipe Connected	PipeName=\postex_*, Image=%Beacon File%

6.2.2.6. 사용자 활동 모니터링

Cobalt Strike에서 지원하는 기본 기능 중 사용자 활동 모니터링에 포함되는 명령은 6개이며, 이 중 Cobalt Strike를 통한 행위로 특정할 수 있는 흔적이 확인된 명령은 5개이다.

[표 60] 사용자 활동 모니터링 카테고리 명령 별 흔적 존재 여부

번호	명령어	시스템 아티팩트 내 흔적	추가 설정된 로그 내 흔적
1	clipboard	존재하지 않음	존재하지 않음
2	desktop	존재하지 않음	Sysmon Event
3	keylogger	존재하지 않음	Sysmon Event
4	printscreen	존재하지 않음	Sysmon Event
5	screenshot	존재하지 않음	Sysmon Event
6	screenwatch	존재하지 않음	Sysmon Event

명령 별로 확인된 흔적에 대한 내용은 다음과 같다.

1) desktop

- **추가 설정 로그 (Sysmon Event)**

desktop 명령 사용 시 특정 프로세스를 지정하지 않으면 기본적으로 rundll32.exe 프로세스를 사용한다. Beacon 프로세스가 인자 값이 존재하지 않는 rundll32.exe 프로세스를 생성하고 접근한다. rundll32.exe 프로세스는 루프백 IP(127.0.0.1)로 네트워크 통신 이벤트를 발생시킨다. 이후, desktop 연결이 수행되는 동안 매초 다량의 네트워크 통신 이벤트를 발생시킨다.

[표 61] desktop 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\system32\rundll32.exe, ParentImage=%Beacon File%
10	Process accessed	SourceImage=%Beacon File%, TargetImage=C:\Windows\system32\rundll32.exe, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72|C:\Windows\System32\KERNELBASE.dll+48a8a|C:\Windows\System32\KERNELBASE.dll+48616|C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(0000000003B41019)
12	Registry – CreateKey	Image=%Beacon File%, TargetObject=HKLM\System\CurrentControlSet\Services\Tcpip\Parameters
3	Network connection	Image=C:\Windows\System32\rundll32.exe, SourceIp=127.0.0.1, SourcePort=%N%, DestinationIp=127.0.0.1, DestinationPort=%M%
3	Network connection	Image=%Beacon File%, SourceIp=%Host IP%, SourcePort=%N%, DestinationIp=%Team Server IP%, DestinationPort=443, DestinationPortName=https
3	Network connection	Image=%Beacon File%, SourceIp=%Host IP%, SourcePort=%N+1%, DestinationIp=%Team Server IP%, DestinationPort=443, DestinationPortName=https
3	Network connection	…

2) keylogger

- **추가 설정 로그 (Sysmon Event)**

keylogger 명령 사용 시 특정 프로세스를 지정하지 않으면 기본적으로 rundll32.exe 프로세스를 사용한다. Beacon 프로세스는 인자 값이 존재하지 않는 rundll32.exe 프로세스를 생성하고 접근한다. rundll32.exe 프로세스는 명명된 Pipe를 생성하고 Beacon 프로세스가 생성된 Pipe에 연결한다. 통신을 위해 생성 및 실행된 Pipe 이름의 형태는 '\postex_{랜덤한 문자 및 숫자 4자리}'로 구성된다.

[표 62] keylogger 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\system32\rundll32.exe, ParentImage=%Beacon File%
10	Process accessed	SourceImage=%Beacon File%, TargetImage=C:\Windows\system32\rundll32.exe, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72|C:\Windows\System32\KERNELBASE.dll+48a8a|C:\Windows\System32\KERNELBASE.dll+48616|C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(0000000003B41019)
17	Pipe Created	PipeName=\postex_*, Image=C:\Windows\system32\rundll32.exe
18	Pipe Connected	PipeName=\postex_*, Image=%Beacon File%

3) printscreen

- **추가 설정 로그 (Sysmon Event)**

printscreen 명령은 PrintScr 방법을 통해 단일 스크린을 찍을 수 있으며, PID를 지정하면 지정한 프로세스에 스크린샷 도구를 주입할 수 있다. 테스트는 2가지 동작 방식(printscreen, printscreen [pid])으로 진행했다.

[표 63] printscreen 명령 실행 결과

명령 구분	연구 결과
printscreen	Beacon 프로세스가 printscreen 모듈을 사용하기 위해 기본 설정 값(프로세스 생성)인 rundll32.exe를 실행하고 접근한다.
printscreen [pid]	Beacon 프로세스가 printscreen 모듈을 사용하기 위해 사용자가 지정한 process에 스레드를 생성한다. 통신을 위해 생성 및 실행된 Pipe 이름의 형태는 '\postex_{랜덤한 문자 및 숫자 4자리}'로 구성된다.

[표 63-1] printscreen 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\system32\rundll32.exe, ParentImage=%Beacon File%
10	Process accessed	SourceImage=%Beacon File%, TargetImage=C:\Windows\system32\rundll32.exe, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72|C:\Windows\System32\KERNELBASE.dll+48a8a|C:\Windows\System32\KERNELBASE.dll+48616|C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(%[Random]%)
17	Pipe Created	PipeName=\postex_*, Image=C:\Windows\System32\rundll32.exe
18	Pipe Connected	PipeName=\postex_*, Image=%Beacon File%

[표 63-2] printscreen [pid] 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
8	CreateRemoteThread	SourceImage=%Beacon File%, TargetImage=%Target Process%, StartAddress=%[Random]%, StartModule=-, StartFunction=
17	Pipe Created	PipeName=\postex_*, Image=%Target Process%
18	Pipe Connected	PipeName=\postex_*, Image=%Beacon File%

4) screenshot

- **추가 설정 로그 (Sysmon Event)**

screenshot 명령은 스크린샷을 찍는 기능을 제공하며, PID를 지정하면 지정한 프로세스에 스크린샷 도구를 주입할 수 있다. 테스트는 2가지 동작 방식(screenshot, screenshot [pid])으로 진행했다.

[표 64] screenshot 명령 실행 결과

명령 구분	연구 결과
screenshot	Beacon 프로세스가 screenshot 모듈을 사용하기 위해 기본 설정 값(프로세스 생성)인 rundll32.exe를 실행하고 접근한다. 통신을 위해 생성 및 실행된 Pipe 이름의 형태는 '\postex_{랜덤한 문자 및 숫자 4자리}'로 구성된다.
screenshot [pid]	Beacon 프로세스가 screenshot 모듈을 사용하기 위해 사용자가 지정한 프로세스에 스레드를 생성한다. 통신을 위해 생성 및 실행된 Pipe 이름의 형태는 '\postex_{랜덤한 문자 및 숫자 4자리}'로 구성된다.

[표 64-1] screenshot 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\system32\rundll32.exe, ParentImage=%Beacon File%
7	Image Loaded	Image=%Beacon File%, ImageLoaded=C:\Windows\System32\oleaut32.dll
10	Process accessed	SourceImage=%Beacon File%, TargetImage=C:\Windows\system32\rundll32.exe, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72|C:\Windows\System32\KERNELBASE.dll+48a8a|C:\Windows\System32\KERNELBASE.dll+48616|C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(%[Random]%)
17	Pipe Created	PipeName=\postex_*, Image=C:\Windows\System32\rundll32.exe
18	Pipe Connected	PipeName=\postex_*, Image=%Beacon File%

[표 64-2] screenshot [pid] 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
7	Image Loaded	Image=%Beacon File%, ImageLoaded=C:\Windows\System32\oleaut32.dll
8	CreateRemoteThread	SourceImage=%Beacon File%, TargetImage=%Target Process%, StartAddress=%[Random]%, StartModule=-, StartFunction=
17	Pipe Created	PipeName=\postex_*, Image=C:\Windows\System32\rundll32.exe
18	Pipe Connected	PipeName=\postex_*, Image=%Beacon File%

5) screenwatch

- **추가 설정 로그 (Sysmon Event)**

screenwatch 명령은 시스템에서 지속적으로 스크린샷을 찍는 도구를 특정 프로세스에 주입해 사용자의 데스크톱 활동을 주기적으로 캡처할 수 있게 한다. 테스트는 2가지 동작 방식(screenwatch, screenwatch [pid])으로 진행했다.

[표 65] screenwatch 명령 실행 결과

명령 구분	연구 결과
screenwatch	Beacon 프로세스가 screenwatch 모듈을 사용하기 위해 기본 설정 값(프로세스 생성)인 rundll32.exe를 실행하고 접근한다. 통신을 위해 생성 및 실행된 Pipe 이름의 형태는 '\postex_{랜덤한 문자 및 숫자 4자리}'로 구성된다.
screenwatch [pid]	Beacon 프로세스가 screenwatch 모듈을 사용하기 위해 사용자가 지정한 프로세스에 스레드를 생성한다. 통신을 위해 생성 및 실행된 Pipe 이름의 형태는 '\postex_{랜덤한 문자 및 숫자 4자리}'로 구성된다.

[표 65-1] screenwatch 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\system32\rundll32.exe, ParentImage=%Beacon File%
7	Image Loaded	Image=%Beacon File%, ImageLoaded=C:\Windows\System32\oleaut32.dll
10	Process accessed	SourceImage=%Beacon File%, TargetImage=C:\Windows\system32\rundll32.exe, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72|C:\Windows\System32\KERNELBASE.dll+48a8a|C:\Windows\System32\KERNELBASE.dll+48616|C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(%[Random]%)
17	Pipe Created	PipeName=\postex_*, Image=C:\Windows\System32\rundll32.exe
18	Pipe Connected	PipeName=\postex_*, Image=%Beacon File%

[표 65-2] screenwatch [pid] 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
7	Image Loaded	Image=%Beacon File%, ImageLoaded=C:\Windows\System32\oleaut32.dll
8	CreateRemoteThread	SourceImage=%Beacon File%, TargetImage=%Target Process%, StartAddress=%[Random]%, StartModule=-, StartFunction=
17	Pipe Created	PipeName=\postex_*, Image=C:\Windows\System32\rundll32.exe
18	Pipe Connected	PipeName=\postex_*, Image=%Beacon File%

6.2.2.7. 통신 및 데이터 암호화

Cobalt Strike에서 지원하는 기본 기능 중 통신 및 데이터 암호화에 포함되는 명령은 11개이며, 이 중 Cobalt Strike를 통한 행위로 특정할 수 있는 흔적이 확인된 명령은 7개이다.

[표 66] 통신 및 데이터 암호화 카테고리 명령 별 흔적 존재 여부

번호	명령어	시스템 아티팩트 내 흔적	추가 설정된 로그 내 흔적
1	connect	존재하지 않음	존재하지 않음
2	covertvpn	Metadata, EventLog, Registry,	Sysmon Event
3	data-store	존재하지 않음	존재하지 않음
4	socks	존재하지 않음	Sysmon Event
5	spawn	존재하지 않음	Sysmon Event
6	spawnas	EventLog	Sysmon Event
7	spawnto	존재하지 않음	존재하지 않음
8	spawnu	존재하지 않음	Sysmon Event
9	spunnel	존재하지 않음	Sysmon Event
10	spunnel_local	존재하지 않음	Sysmon Event
11	token-store	존재하지 않음	존재하지 않음

명령 별로 확인된 흔적에 대한 내용은 다음과 같다.

1) covertvpn

- **추가 설정 로그 (Sysmon Event)**

Beacon 프로세스가 인자 값이 존재하지 않는 rundll32.exe 프로세스를 생성하고 접근한다. rundll32.exe 프로세스가 C:\Users\ADMIN~1\Appdata\Local\Temp 경로에 파일을 생성하고 이미지 로드 및 서비스 설치 이벤트를 발생시킨다. %TEMP%(C:\Users\ADMIN~1\AppData\Local\Temp) 하위 경로에 3개의 파일을 rundll32.exe가 생성하고 이미지 로드하는 이벤트가 발생하며, 일반적으로 rundll32.exe가 이미지를 로드 하지 않는 위치에서 발생한다.

[표 67] covertvpn 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\SysWOW64\rundll32.exe, CommandLine=C:\Windows\system32\rundll32.exe, ParentImage=%Beacon File%
10	Process accessed	SourceImage=%Beacon File%, TargetImage=C:\Windows\syswow64\rundll32.exe, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72|C:\Windows\System32\KERNELBASE.dll+48a8a|C:\Windows\System32\KERNELBASE.dll+48616|C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(%[Random]%)
11	FileCreate	Image=C:\Windows\syswow64\rundll32.exe, TargetFilename=C:\Users\ADMINI~1\AppData\Local\Temp\1\wpcap.dll
11	FileCreate	Image=C:\Windows\syswow64\rundll32.exe, TargetFilename=C:\Users\ADMINI~1\AppData\Local\Temp\1\Packet.dll
11	FileCreate	Image=C:\Windows\syswow64\rundll32.exe, TargetFilename=C:\Users\ADMINI~1\AppData\Local\Temp\1\npf.sys
7	Image loaded	Image=C:\Windows\syswow64\rundll32.exe, ImageLoaded=C:\Users\ADMINI~1\AppData\Local\Temp\1\wpcap.dll
7	Image loaded	Image=C:\Windows\syswow64\rundll32.exe, ImageLoaded=C:\Users\ADMINI~1\AppData\Local\Temp\1\Packet.dll
12	Registry – CreateKey	Image=C:\Windows\system32\services.exe, TargetObject=HKLM\System\CurrentControlSet\Services\npf
13	Registry – SetValue	Image=C:\Windows\system32\services.exe, TargetObject=HKLM\System\CurrentControlSet\Services\npf\ImagePath, Details=\??\C:\Users\ADMINI~1\AppData\Local\Temp\1\npf.sys

- **시스템 아티팩트 (Metadata, EventLog, Registry)**

covertvpn 명령을 실행하면 시스템에 남는 흔적은 다음과 같다.

[표 68] covertvpn 명령 실행 시 시스템 아티팩트에서 확인 가능한 패턴

아티팩트 구분	연구 결과
Metadata	# $MFT, $UsnJrnl C:\Users\Administrator\AppData\Local\Temp\1 폴더 하위에 3개의 파일(wpcap.dll, Packet.dll, npf.sys)이 생성됨
EventLog	# System.evtx npf 서비스 설치 이벤트(Event ID: 7045)가 발생하며, 등록되는 서비스 파일 이름은 'C:\Users\ADMIN~1\AppData\Local\Temp\1\npf.sys이다.
Registry	# SYSTEM Session Manager 키의 PendingFileRenameOperations Value의 값으로 생성된 3개의 파일(wpcap.dll, Packet.dll, npf.sys)이 등록됨

2) socks

- **추가 설정 로그 (Sysmon Event)**

Beacon 프로세스가 Socks를 생성한 후, 매초 다량의 네트워크 통신 이벤트를 발생시킨다.

[표 69] socks 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
3	Network connection	Image=%Beacon File%, SourceIp=%Host IP%, SourcePort=%N%, DestinationIp=%Team Server IP%, DestinationPort=443, DestinationPortName=https
3	Network connection	Image=%Beacon File%, SourceIp=%Host IP%, SourcePort=%N+1%, DestinationIp=%Team Server IP%, DestinationPort=443, DestinationPortName=https
3	Network connection	…

3) spawn

- **추가 설정 로그 (Sysmon Event)**

Beacon 프로세스가 페이로드 주입을 위해 기본 설정 값(프로세스 생성)인 rundll32.exe를 실행하고 접근한다.

[표 70] spawn 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\system32\rundll32.exe, ParentImage=%Beacon File%
10	Process accessed	SourceImage=%Beacon File%, TargetImage=C:\Windows\system32\rundll32.exe, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72|C:\Windows\System32\KERNELBASE.dll+48a8a|C:\Windows\System32\KERNELBASE.dll+48616|C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(%[Random]%)

4) spawnas

- **추가 설정 로그 (Sysmon Event)**

Beacon 프로세스가 다른 사용자 계정으로 새로운 Beacon을 생성하면서 기본 설정 값인 rundll32.exe를 실행한다.

[표 71] spawnas 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\system32\rundll32.exe, ParentImage=%Beacon File%

- **시스템 아티팩트 (EventLog)**

현재 로그인된 사용자가 아닌 다른 사용자 계정으로 로그인을 수행해 프로세스를 실행하기 때문에 Secondary Logon 서비스가 활성화된다.

[표 72] spawnas 명령 실행에 따른 이벤트 로그(System.evtx)

이벤트 ID	이벤트 유형	내용
7036	서비스 상태 변경	Secondary Logon 서비스가 running 상태로 들어갔습니다.

5) spawnu

- **추가 설정 로그 (Sysmon Event)**

Beacon 프로세스가 새로운 Beacon을 생성하면서 기본 설정 값(프로세스 생성)인 rundll32.exe를 실행한다.

[표 73] spawnu 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\system32\rundll32.exe, ParentImage=%Beacon File%
10	Process accessed	SourceImage=%Beacon File%, TargetImage=C:\Windows\system32\rundll32.exe, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72|C:\Windows\System32\KERNELBASE.dll+48a8a|C:\Windows\System32\KERNELBASE.dll+48616|C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(%[Random]%)

6) spunnel

- **추가 설정 로그 (Sysmon Event)**

Beacon 프로세스가 페이로드 실행을 위해 rundll32.exe를 생성한 후, 매초 다량의 네트워크 통신 이벤트를 발생시킨다.

[표 74] spunnel 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\system32\rundll32.exe, ParentImage=%Beacon File%
10	Process accessed	SourceImage=%Beacon File%, TargetImage=C:\Windows\system32\rundll32.exe, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM3\2\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72|C:\Windows\System32\KERNELBASE.dll+48a8a|C:\Windows\System32\KERNELBASE.dll+48616|C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(%[Random]%)
3	Network connection	Image=%Beacon File%, SourceIp=%Host IP%, SourcePort=%N%, DestinationIp=%Team Server IP%, DestinationPort=443, DestinationPortName=https
3	Network connection	Image=%Beacon File%, SourceIp=%Host IP%, SourcePort=%N+1%, DestinationIp=%Team Server IP%, DestinationPort=443, DestinationPortName=https
3	Network connection	…

7) spunnel_local

- **추가 설정 로그 (Sysmon Event)**

Beacon 프로세스가 페이로드 실행을 위해 rundll32.exe를 생성한 후, 매초 다량의 네트워크 통신 이벤트를 발생시킨다.

[표 75] spunnel_local 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\system32\rundll32.exe, ParentImage=%Beacon File%
10	Process accessed	SourceImage=%Beacon File%, TargetImage=C:\Windows\system32\rundll32.exe, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72|C:\Windows\System32\KERNELBASE.dll+48a8a|C:\Windows\System32\KERNELBASE.dll+48616|C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(%[Random]%)
3	Network connection	Image=%Beacon File%, SourceIp=%Host IP%, SourcePort=%N%, DestinationIp=%Team Server IP%, DestinationPort=443, DestinationPortName=https
3	Network connection	Image=%Beacon File%, SourceIp=%Host IP%, SourcePort=%N+1%, DestinationIp=%Team Server IP%, DestinationPort=443, DestinationPortName=https
3	Network connection	…

6.2.2.8. 명령 및 스크립트 실행

Cobalt Strike에서 지원하는 기본 기능 중 통신 및 데이터 암호화에 포함되는 명령은 11개이며, 이 중 Cobalt Strike를 통한 행위로 특정할 수 있는 흔적이 확인된 명령은 8개이다.

[표 76] 명령 및 스크립트 실행 카테고리 명령 별 흔적 존재 여부

번호	명령어	시스템 아티팩트 내 흔적	추가 설정된 로그 내 흔적
1	execute	존재하지 않음	Sysmon Event
2	execute-assembly	존재하지 않음	Sysmon Event
3	inline-execute	테스트 대상 아님	테스트 대상 아님
4	powerpick	EventLog	Sysmon Event
5	powershell	EventLog	Sysmon Event
6	powershell-import	EventLog, Registry	존재하지 않음
7	run	존재하지 않음	존재하지 않음
8	runas	EventLog	Sysmon Event
9	runasadmin	존재하지 않음	존재하지 않음
10	runu	존재하지 않음	Sysmon Event
11	shell	존재하지 않음	Sysmon Event

명령 별로 확인된 흔적에 대한 내용은 다음과 같다.

1) execute

- **추가 설정 로그 (Sysmon Event)**

Beacon 프로세스는 해당 명령을 수행할 대상 프로그램을 생성하고 접근한다.

[표 77] execute 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=%Target Program%, ParentImage=%Beacon File%
10	Process accessed	SourceImage=%Beacon File%, TargetImage=%Target Program%, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72|C:\Windows\System32\KERNELBASE.dll+48a8a|C:\Windows\System32\KERNELBASE.dll+48616|C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(0000000003B81019)

2) execute-assembly

- **추가 설정 로그 (Sysmon Event)**

Beacon 프로세스는 인자 값이 존재하지 않는 rundll32.exe 프로세스를 생성하고 접근한다.

[표 78] execute-assembly 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\system32\rundll32.exe, ParentImage=%Beacon File%
10	Process accessed	SourceImage=%Beacon File%, TargetImage=C:\Windows\system32\rundll32.exe, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72|C:\Windows\System32\KERNELBASE.dll+48a8a|C:\Windows\System32\KERNELBASE.dll+48616|C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(0000000003B41019)

3) powerpick

- **추가 설정 로그 (Sysmon Event)**

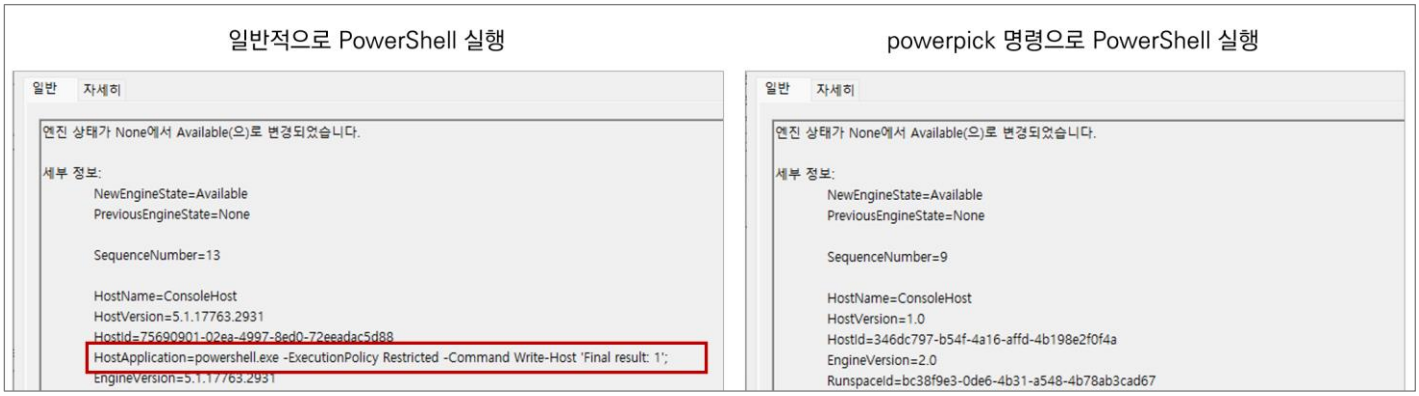
Beacon 프로세스는 사용자에게 입력 받은 Command를 powershell.exe로 실행한다. rundll32.exe는 기본적으로 Windows 기본 DLL(예. KERNEL32.DLL, user32.dll, gdi32.dll)을 호출해 시스템 수준의 작업을 수행하며, .NET Framework 환경과 관련된 DLL을 직접적으로 사용하는 경우가 없어 이상 정황으로 판단할 수 있다.

[표 79] powerpick 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\rundll32.exe, CommandLine=C:\Windows\system32\rundll32.exe, ParentImage=%Beacon File%
7	Image Loaded	Image=%Beacon File%, ImageLoaded=C:\Windows\Microsoft.NET\Framework64\%[Version]%\mscoreei.dll
7	Image Loaded	Image=%Beacon File%, ImageLoaded=C:\Windows\Microsoft.NET\Framework64\%[Version]%\mscorwks.dll
7	Image Loaded	Image=%Beacon File%, ImageLoaded=C:\Windows\Microsoft.NET\Framework64\%[Version]%\mscorjit.dll
7	Image Loaded	Image=%Beacon File%, ImageLoaded=C:\Windows\Microsoft.NET\Framework64\%[Version]%\Culture.dll
10	Process accessed	SourceImage=%Beacon File%, TargetImage=C:\Windows\system32\rundll32.exe, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72| C:\Windows\System32\KERNELBASE.dll+48a8a|C:\Windows\System32\KERNELBASE.dll+48616| C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(%[Random]%)
17	Pipe Created	PipeName=\postex_*, Image=C:\Windows\System32\rundll32.exe
18	Pipe Connected	PipeName=\postex_*, Image=%Beacon File%

- **시스템 아티팩트 (EventLog)**

일반적으로 파워셸을 실행할 경우, 파워셸 이벤트(Event ID: 400)에 'HostApplication='에 Command가 기록되지만 PowerPick을 사용해 메모리 상에서 실행할 경우, 'HostApplication=' 자체가 존재하지 않는다.



[그림 15] HostApplication 데이터 존재 여부 비교 이벤트(Windows Powershell.evtx)

4) powershell

- **추가 설정 로그 (Sysmon Event)**

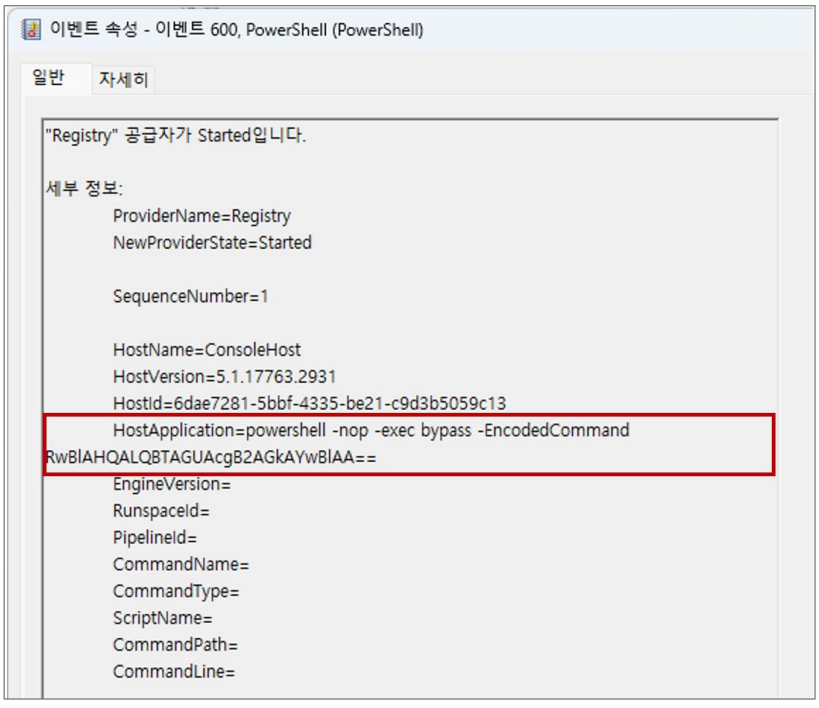
Beacon 프로세스는 사용자에게 입력 받은 Command를 powershell.exe로 실행한다.

[표 80] powershell 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe, CommandLine=powershell -nop -exec bypass -EncodedCommand %BASE64_ENC(Command)%, ParentImage=%Beacon File%
1	Process Create	Image=C:\Windows\System32\conhost.exe, ParentImage=C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe, ParentCommandLine=powershell -nop -exec bypass - EncodedCommand %BASE64_ENC(Command)%
10	Process accessed	SourceImage=%Beacon File%, TargetImage=C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72| C:\Windows\System32\KERNELBASE.dll+48a8a|C:\Windows\System32\KERNELBASE.dll+48616| C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(%[Random]%)

- **시스템 아티팩트 (EventLog)**

powershell 명령 실행 시, HostApplication의 값에서 인코딩된 파워셸 명령 구문이 실행된 이력 확인이 가능하다.



[그림 16] powershell 명령 실행 시 로깅되는 이벤트 (Windows PowerShell.evtx)

5) powershell-import

- **시스템 아티팩트 (EventLog, Registry)**

powershell-import 명령을 실행할 경우, EventLog에서는 Base64로 인코딩된 파워셸 명령 구문 확인이 가능하고, Registry에서는 관련된 레지스트리 키가 변경되는 이력이 확인되었다.

[표 81] powershell-import 명령 실행 시 시스템 아티팩트에서 확인 가능한 패턴

아티팩트 구분	연구 결과
EventLog	# Windows PowerShell.evtx (Event ID: 600) HostApplication의 값에서 인코딩된 파워셸 명령 구문으로 실행된 이력 확인 가능
Registry	# SOFTWARE 2개의 레지스트리 키가 변경되는 이력 확인 가능 (Microsoft\Tracing\powershell_RASAPI32, Microsoft\Tracing\powershell_RASMANCS)

6) runas

- **추가 설정 로그 (Sysmon Event)**

사용자가 입력한 “Command args”를 특정 사용자 계정으로 실행할 때 프로세스 생성 이벤트가 발생한다.

[표 82] runas 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=%Command%, CommandLine=”%Command% %args%”, ParentImage=%Beacon File%

- **시스템 아티팩트 (EventLog)**

현재 로그온한 사용자가 아닌 다른 사용자 계정으로 로그인을 수행해 프로세스를 실행하기 때문에 Secondary Logon 서비스가 활성화된다.

[표 83] runas 명령 실행에 따른 이벤트 로그(System.evtx)

이벤트 ID	이벤트 유형	내용
7036	서비스 상태 변경	Secondary Logon 서비스가 running 상태로 들어갔습니다.

7) runu

- **추가 설정 로그 (Sysmon Event)**

사용자가 입력한 “Command args”를 특정 프로세스 하위에서 생성하면서 Beacon이 생성된 프로세스에 접근하는 이벤트가 발생한다.

[표 84] run 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=%Command%, CommandLine=”%Command% %args%”, ParentImage=%Beacon File%
10	Process accessed	SourceImage=%Beacon File%, TargetImage=%Command%, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72|C:\Windows\System32\KERNELBASE.dll+48a8a|C:\Windows\System32\KERNELBASE.dll+48616|C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(%[Random]%)

8) shell

- **추가 설정 로그 (Sysmon Event)**

Beacon 프로세스가 Shell 명령을 수행하기 위해 cmd.exe를 생성하고 접근하는 이벤트가 확인되며, 해당 이벤트를 통해 공격자가 입력한 Command를 확인할 수 있다.

[표 85] shell 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
1	Process Create	Image=C:\Windows\System32\cmd.exe, CommandLine=C:\Windows\system32\cmd.exe /C %Command%, ParentImage=%Beacon File%
10	Process accessed	SourceImage=%Beacon File%, TargetImage=C:\Windows\system32\cmd.exe, GrantedAccess=2097151, CallTrace=C:\Windows\SYSTEM32\ntdll.dll+a2044|C:\Windows\System32\KERNELBASE.dll+4be72|C:\Windows\System32\KERNELBASE.dll+48a8a|C:\Windows\System32\KERNELBASE.dll+48616|C:\Windows\System32\KERNEL32.DLL+1c153|UNKNOWN(%[Random]%)

6.2.2.9. 네트워크 트래픽 조작 및 우회

Cobalt Strike에서 지원하는 기본 기능 중 네트워크 트래픽 조작 및 우회에 포함되는 명령은 1개이며, 해당 명령을 통해 Cobalt Strike를 통한 행위로 특정할 수 있는 흔적 존재했다.

[표 86] 네트워크 트래픽 조작 및 우회 카테고리 명령 별 흔적 존재 여부

번호	명령어	시스템 아티팩트 내 흔적	추가 설정된 로그 내 흔적
1	browserpivot	존재하지 않음	Sysmon Event

browserpivot 명령 실행을 통해 확인된 흔적에 대한 내용은 다음과 같다.

1) browserpivot

- 추가 설정 로그 (Sysmon Event)

Beacon 프로세스가 iexplorer.exe를 대상으로 원격 스레드를 생성한 후 매초 다량의 네트워크 통신 이벤트를 발생시킨다.

[표 87] browserpivot 명령 실행에 따른 Sysmon 이벤트 로그 패턴

이벤트 ID	이벤트 유형	내용
8	CreateRemoteTread	SourceImage=%Beacon File%, TargetImage=C:\Program Files\internet explorer\iexplore.exe, StartAddress=%Random%
3	Network connection	Image=%Beacon File%, SourceIp=%Host IP%, SourcePort=%N%, DestinationIp=%Team Server IP%, DestinationPort=443, DestinationPortName=https
3	Network connection	Image=%Beacon File%, SourceIp=%Host IP%, SourcePort=%N+1%, DestinationIp=%Team Server IP%, DestinationPort=443, DestinationPortName=https
3	Network connection	…

6.2.2.10.디버깅 및 진단

Cobalt Strike에서 지원하는 기본 기능 중 디버깅 및 진단에 포함되는 명령은 13개이며, 해당 명령들 모두 Cobalt Strike를 통한 행위로 특정할 수 있는 흔적은 발견되지 않았다.

[표 88] 디버깅 및 진단 카테고리 명령 별 흔적 존재 여부

번호	명령어	시스템 아티팩트 내 흔적	추가 설정된 로그 내 흔적
1	!	존재하지 않음	존재하지 않음
2	cancel	존재하지 않음	존재하지 않음
3	clear	존재하지 않음	존재하지 않음
4	downloads	존재하지 않음	존재하지 않음
5	exit	존재하지 않음	존재하지 않음
6	help	존재하지 않음	존재하지 않음
7	history	존재하지 않음	존재하지 않음
8	jobkill	존재하지 않음	존재하지 않음
9	jobs	존재하지 않음	존재하지 않음
10	note	존재하지 않음	존재하지 않음
11	setenv	존재하지 않음	존재하지 않음
12	syscall-method	존재하지 않음	존재하지 않음
13	windows_error_code	존재하지 않음	존재하지 않음

6.2.2.11.시스템 정보 수집

Cobalt Strike에서 지원하는 기본 기능 중 시스템 정보 수집에 포함되는 명령은 2개이며, 해당 명령들 모두 Cobalt Strike를 통한 행위로 특정할 수 있는 흔적은 발견되지 않았다.

[표 89] 시스템 정보 수집 카테고리 명령 별 흔적 존재 여부

번호	명령어	시스템 아티팩트 내 흔적	추가 설정된 로그 내 흔적
1	getuid	존재하지 않음	존재하지 않음
2	pwd	존재하지 않음	존재하지 않음

6.2.2.12. 시스템 환경 제어

Cobalt Strike에서 지원하는 기본 기능 중 시스템 환경 제어에 포함되는 명령은 1개이며, Cobalt Strike를 통한 행위로 특정할 수 있는 흔적은 발견되지 않았다.

[표 90] 시스템 환경 제어 정보 수집 카테고리 명령 별 흔적 존재 여부

번호	명령어	시스템 아티팩트 내 흔적	추가 설정된 로그 내 흔적
1	reg	존재하지 않음	존재하지 않음

6.3. Cobalt Strike 탐지 방안

본 연구에서는 '6.2. Cobalt Strike 분석'에서 확인된 호스트 아티팩트를 기반으로, Cobalt Strike 위협을 탐지하고 침해사고 대응에 활용할 수 있는 탐지 룰을 설계했다. 두 가지 도구(Sysmon, 해킹진단도구)를 활용해 Cobalt Strike의 침해 흔적을 탐지할 수 있는 구체적인 탐지 룰을 개발했으며, 다양한 공격 시나리오를 고려해 실제 환경에서 효과적으로 적용할 수 있는 탐지 룰 구성 방안을 마련했다. 각 탐지 룰에 대한 상세 설명은 다음과 같다.

6.3.1. Sysmon Configure Rule

Sysmon(System Monitor)은 Microsoft 사의 Sysinternals Suite에 포함된 시스템 모니터링 도구로, 시스템에서 일어나는 이벤트 행위(프로세스 생성, 이벤트 연결 등)를 더 자세하게 기록한다. 이벤트 로그가 남기지 않는 정보를 추가로 기록할 수 있기 때문에 공격자(또는 악성코드)가 수행한 악성 행위와 시스템 이상 징후 원인 등을 식별하는데 많은 도움이 된다. Sysmon을 기본 설정으로 설치하게 되면 로깅 되지 않는 이벤트가 존재하고, 설정을 모두 활성화하면 많은 이벤트가 로깅되어 필요 목적에 맞는 이벤트를 놓칠 수 있다. 따라서, 운영하고 있는 시스템 용도와 모니터링 목적에 맞는 Configure를 작성해 반영해야 한다. Configure 파일은 XML 형식으로 작성되어 있으며, 이벤트 유형이 정의된 필드에 논리 연산(AND, OR)을 적용해 필터링 옵션을 사용할 수 있어 높은 유연성을 가진다.

2023년에 진행한 'Sysmon을 이용한 호스트 기반의 사이버 위협 로깅 방안 연구'에서 다년간의 침해사고 분석 반영해 공격자가 주로 수행하는 행위를 시뮬레이션하고, 이에 따른 행위가 Sysmon 로그에 어떻게 로깅 되는지를 분석했다. 이를 기반으로 공격자의 행위를 탐지하기 위한 Sysmon Configure Rule을 제작했다.

```
<!--

######   ##        #####    ####    ######   ######    ####    ######
##   ##  ##       ##   ##    ##     ##   ##  ##   ##    ##       ##
##   ##  ##       ##   ##    ##     ##   ##  ##   ##    ##       ##
######   ##       #######    ##     ##   ##  ######     ##       ##
##       ##       ##   ##    ##     ##   ##  ##         ##       ##
##       ##       ##   ##    ##     ##   ##  ##         ##       ##
##       #######  ##   ##   ####    ##   ##  ######    ####      ##

 #####    ##  ##   #####    #####    #####   ######          #####    #####   ######   #######   ####     #####   ##   ##  ######   #######
##   ##   ##  ##  ##   ##  ## # ##  ##   ##  ##   ##        ##   ##  ##   ##  ##   ##  ##         ##     ##   ##  ##   ##  ##   ##  ##
##        ##  ##  ##       ## # ##  ##   ##  ##   ##        ##       ##   ##  ##   ##  ##         ##     ##       ##   ##  ##   ##  ##
 #####    ####    #####   ## # ##  ##   ##  ##   ##        ##       ##   ##  ##   ##  #####      ##     ## ###   ##   ##  ######   #####
     ##    ##         ##  ##   ##  ##   ##  ##   ##        ##       ##   ##  ##   ##  ##         ##     ##  ##   ##   ##  ##       ##
##   ##    ##    ##   ##  ##   ##  ##   ##  ##   ##        ##   ##  ##   ##  ##   ##  ##         ##     ##  ##   ##   ##  ##       ##
 #####     ##     #####   ##   ##   #####   ##   ##         #####    #####   ##   ##  ##        ####     #####    #####   ##       #######

-->
<!-- BIT-sysmon-config v2.0 -->
<!-- Update Date: 2024-10-15 -->
<Sysmon schemaversion="4.90">
    <!-- Configure Entries -->
    <HashAlgorithms>MD5,SHA1</HashAlgorithms>
    <ArchiveDirectory>SYSMON_Archive</ArchiveDirectory>

    <!-- Event Filtering -->
    <EventFiltering>
        <!-- Event ID 1. Process create -->
        <!-- 설명: Process create 이벤트는 새로 생성된 프로세스에 대한 정보를 제공한다. -->
        <RuleGroup name="" groupRelation="or">
            <ProcessCreate onmatch="include">
                <Image condition="begin with">C:\PerfLogs\</Image> <!-- 공격자가 주로 활용하는 시스템 경로에서 생성된 프로세스 탐지 -->
                <Image condition="begin with">C:\$Recycle.bin\</Image>
                <Image condition="begin with">C:\Users\Default\</Image>
                <Image condition="begin with">C:\Users\Public\</Image>
                <Image condition="begin with">C:\Windows\Fonts\</Image>
                <Image condition="begin with">C:\Windows\Debug\</Image>
                <Image condition="begin with">C:\Windows\Media\</Image>
```

[그림 17] PLAINBIT Sysmon Configure Rule 일부

Sysmon Configure Rule의 작성 기준을 고려하기 위해서는 Rule 작성이 가능한 명령을 구분해야 한다. 이때, Cobalt Strike에서 기본으로 지원하는 기능은 입력만 수행하는 기능, 입출력을 모두 수행하는 기능, 추가 이벤트를 발생하는 기능으로 구분할 수 있다. 이 중 추가 이벤트가 발생하는 기능을 기준으로 Sysmon Configure Rule이 작성되었다.

1) 입력만 수행하는 기능

Cobalt Strike에서 명령을 입력 받고 별도의 출력이 필요하지 않는 명령들은 네트워크 연결 이벤트가 한 번만 발생하기 때문에 Sysmon으로 구분하기 어렵다. 입력만 수행하는 명령은 다음과 같다.

[표 91] Cobalt Strike에서 명령 실행 시 입력만 수행하는 기능

Cobalt Strike 명령어 목록
argue, blockdlls, cancel, cd, cp, jobkill, mkdir, mv, powershell-import, reg, rev2self, rm, rportfwd, rportfwd_local, runasadmin, setenv, sleep, spawnto, syscall-method, timestomp, unlink, upload

2) 입출력을 수행하는 기능

Cobalt Strike에서 명령을 입력 받고 해당하는 출력 값을 반환해야 하는 명령들은 네트워크 연결 이벤트가 두 번 발생하기 때문에 Sysmon으로 구분하기 어렵다.

[표 92] Cobalt Strike에서 명령 실행 시 입출력을 수행하는 기능

Cobalt Strike 명령어 목록
clipboard, connect, data-store, download, drives, exit, file_browser, getprivs, getsystem, getuid, jobs, ls, printscreen, process_browser, ps, pwd, steal_token, token_store

3) 추가 이벤트가 발생하는 기능

Cobalt Strike에서 명령을 입력 받고 수행할 때 추가적으로 이벤트가 발생해 Sysmon Configure Rule로 구분할 수 있는 명령들이 있다. 해당 명령들 중 고유한 이벤트 패턴을 가진 명령과 공통으로 발생하는 패턴을 가진 명령을 구분해 Rule을 작성했다.

[표 93] Cobalt Strike에서 명령 실행 시 추가 이벤트가 발생하는 기능

Cobalt Strike 명령어 목록
browserpivot, chromedump, covertvpn, dcsync, desktop, dllinject, dllload, elevate, execute, execute-assembly, hashdump, inject, jump, keylogger, link, logonpasswords, mimikatz, net, portscan, powerpick, powershell, printscreen, psinject, pth, remote-exec, run, runas, runu, screenshot, screenwatch, shell, shinject, shspawn, socks, spawn, spawnas, spawnu, spunnel, spunnel_local

6.3.1.1. 주요 Rule 설명

Cobalt Strike에서 명령을 실행하는 과정에서 추가 이벤트를 발생해 탐지할 수 있는 기능을 기반으로 총 23개의 Sysmon Configure Rule을 작성했으며, 작성된 주요 Rule 목록은 다음과 같다.

[표 94] Cobalt Strike를 활용한 행위를 탐지할 수 있는 주요 Sysmon Rule 목록

이벤트 ID	이벤트 명	Rule 목록
1	Process Create	HTML Application 형태의 Cobalt Strike Beacon 프로세스 생성 이벤트 탐지
		Encoding된 명령어를 수행하는 powershell.exe 프로세스 생성 이벤트 탐지
		Encoding된 명령어를 수행하는 powershell.exe 프로세스가 특정 명령 구문을 수행하는 프로세스 생성 이벤트 탐지
		인자 값이 없는 rundll32.exe 프로세스 생성 이벤트 탐지
		Pipe에 문자열을 전달하는 명령어를 가진 cmd.exe 프로세스 생성 이벤트 탐지
		WinRM 프로세스가 부모인 프로세스 생성 이벤트 탐지
		WMI 프로세스가 부모인 프로세스 생성 이벤트 탐지
3	NetworkConnect	Cobalt Strike Beacon에서 사용할 수 있는 프로토콜을 통한 네트워크 연결 이벤트 탐지
		rundll32.exe 프로세스가 LocalHost로 네트워크 연결하는 이벤트 탐지
7	ImageLoad	rundll32.exe 프로세스가 NPcap DLL을 로드하는 이벤트 탐지
		rundll32.exe 프로세스가 .NET 관련 DLL을 로드하는 이벤트 탐지
8	CreateRemoteThread	KERNEL32.DLL의 LoadLibraryA 함수를 사용하는 원격 스레드 생성 이벤트 탐지
		시스템 드라이브에서 원격 스레드 생성 이벤트 탐지
		rundll32.exe 프로세스가 lsass.exe 프로세스에 원격 스레드 생성 이벤트 탐지
		browserpivot 명령어 수행 시 발생하는 원격 스레드 생성 이벤트 탐지
10	ProcessAccess	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
11	FileCreate	rundll32.exe 프로세스가 %TEMP% 하위 경로에 파일 생성 이벤트 탐지
12, 13	RegistryEvent	Cobalt Strike Beacon 실행 시 발생하는 레지스트리 이벤트 탐지
		레지스트리를 통해 ADMIN$ 경로의 실행 파일을 서비스로 등록하는 이벤트 탐지
		레지스트리를 통해 PowerShell 명령어를 서비스로 등록하는 이벤트 탐지
		레지스트리를 통해 NPcap 파일을 서비스로 등록하는 이벤트 탐지
17, 18	PipeEvent	Cobalt Strike 실행 시 발생하는 명명된 Pipe 이벤트 탐지
		Cobalt Strike 명령어 수행 시 발생하는 명명된 Pipe 이벤트 탐지

각 이벤트와 관련해 작성된 Sysmon Configure Rule의 상세 설명은 다음과 같다.

1) ProcessCreate

ProcessCreate 이벤트는 새로 생성된 프로세스에 대한 정보를 기록하는 이벤트 항목으로, Cobalt Strike 사용에 따라 프로세스 생성 이벤트를 탐지하는 Rule을 작성했다.

[표 95] Sysmon Configure Rule에서 ProcessCreate 이벤트를 통해 탐지할 수 있는 명령어 목록

번호	명령 카테고리	명령어	설명
1	-	Beacon (HTML Application)	- HTML Application 형태의 Cobalt Strike Beacon 프로세스 생성 이벤트 탐지
2	네트워크 탐색 및 이동	jump	- Encoding된 명령어를 수행하는 powershell.exe 프로세스 생성 이벤트 탐지 - WinRM 프로세스가 부모인 프로세스 생성 이벤트 탐지 - Encoding된 명령어를 수행하는 powershell.exe 프로세스가 특정 명령 구문을 수행하는 프로세스 생성 이벤트 탐지
3		net	- 인자 값이 없는 rundll32.exe 프로세스 생성 이벤트 탐지
4		portscan	- 인자 값이 없는 rundll32.exe 프로세스 생성 이벤트 탐지
5		remote-exec	- Encoding된 명령어를 수행하는 powershell.exe 프로세스 생성 이벤트 탐지 - WinRM 프로세스가 부모인 프로세스 생성 이벤트 탐지 - WMI 프로세스가 부모인 프로세스 생성 이벤트 탐지
6	권한 상승 및 크리덴셜 수집	chromedump	- 인자 값이 없는 rundll32.exe 프로세스 생성 이벤트 탐지
7		dcsync	- 인자 값이 없는 rundll32.exe 프로세스 생성 이벤트 탐지
8		hashdump	- 인자 값이 없는 rundll32.exe 프로세스 생성 이벤트 탐지
9		logonpasswords	- 인자 값이 없는 rundll32.exe 프로세스 생성 이벤트 탐지
10		mimikatz	- 인자 값이 없는 rundll32.exe 프로세스 생성 이벤트 탐지
11		pth	- 인자 값이 없는 rundll32.exe 프로세스 생성 이벤트 탐지 - Pipe에 문자열을 전달하는 명령어를 가진 cmd.exe 프로세스 생성 이벤트 탐지
12	사용자 활동 모니터링	desktop	- 인자 값이 없는 rundll32.exe 프로세스 생성 이벤트 탐지
13		keylogger	- 인자 값이 없는 rundll32.exe 프로세스 생성 이벤트 탐지
14		printscreen	- 인자 값이 없는 rundll32.exe 프로세스 생성 이벤트 탐지
15		screenshot	- 인자 값이 없는 rundll32.exe 프로세스 생성 이벤트 탐지
16		screenwatch	- 인자 값이 없는 rundll32.exe 프로세스 생성 이벤트 탐지
17	통신 및 데이터 암호화	covertvpn	- 인자 값이 없는 rundll32.exe 프로세스 생성 이벤트 탐지

번호	명령 카테고리	명령어	설명
18	통신 및 데이터 암호화	spawnas	- 인자 값이 없는 rundll32.exe 프로세스 생성 이벤트 탐지
19		spawnu	- 인자 값이 없는 rundll32.exe 프로세스 생성 이벤트 탐지
20		spunnel	- 인자 값이 없는 rundll32.exe 프로세스 생성 이벤트 탐지
21		spunnel_local	- 인자 값이 없는 rundll32.exe 프로세스 생성 이벤트 탐지
22		execute-assembly	- 인자 값이 없는 rundll32.exe 프로세스 생성 이벤트 탐지
23		Powerpick	- 인자 값이 없는 rundll32.exe 프로세스 생성 이벤트 탐지
24		powershell	- Encoding된 명령어를 수행하는 powershell.exe 프로세스 생성 이벤트 탐지

ProcessCreate 이벤트 탐지와 관련해 작성된 Rule 설명은 다음과 같다.

- **HTML Application 형태의 Cobalt Strike Beacon 프로세스 생성 이벤트 탐지**

Cobalt Strike Beacon이 HTML Application 실행 형태로 동작하는 경우, mshta.exe 프로세스가 부모 프로세스로 생성되는 이벤트가 탐지된다. PowerShell 프로세스를 생성하게 될 때, 32bit 시스템 파일을 사용하기 때문에 HTML Application 형태의 Cobalt Strike Beacon 프로세스가 생성되는 것을 탐지하도록 작성했다.

```
<Rule name="Turn on Cobalt Strike Beacon(HTML Application)" groupRelation="and"> <!-- Cobalt Strike에서 HTML Application 형태의 Beacon 실행 탐지 -->
  <Image condition="is">C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe</Image>
  <CommandLine condition="contains">-encodedcommand</CommandLine>
  <ParentImage condition="is">C:\Windows\SysWOW64\mshta.exe</ParentImage>
</Rule>
<ParentImage condition="image">mshta.exe</ParentImage> <!-- HTML Application(HTA) 파일을 실행하는 프로세스 탐지 -->
```

[그림 18] HTML Application 형태의 Cobalt Strike Beacon 프로세스 생성 이벤트 탐지 룰

- **인코딩된 명령어를 수행하는 powershell.exe 프로세스 생성 이벤트 탐지**

Cobalt Strike Beacon이 특정 명령을 수행할 때 PowerShell 프로세스에서 Encoding된 명령어가 실행되는 이벤트가 탐지된다. 따라서, Encoding된 명령어를 수행하는 powershell.exe 프로세스가 생성되는 것을 탐지하도록 작성했다.

```
<Rule name="Execute Encoded PowerShell Commands" groupRelation="and"> <!-- Encoding된 PowerShell 명령어 수행 탐지 -->
  <OriginalFileName condition="is">PowerShell.EXE</OriginalFileName>
  <CommandLine condition="contains">-nop</CommandLine>
  <CommandLine condition="contains any">-EncodedCommand;-encodedcommand</CommandLine>
</Rule>
```

[그림 19] 인코딩된 명령어를 수행하는 powershell.exe 프로세스 생성 이벤트 탐지 룰

- **인코딩된 명령어를 수행하는 powershell.exe 프로세스가 특정 명령 구문을 수행하는 프로세스 생성 이벤트 탐지**

Cobalt Strike Beacon이 jump psexec_psh 명령을 수행할 때 Encoding된 명령어를 수행하는 PowerShell 프로세스가 '-Version', '-s', '-NoLogo', '-NoProfile' 명령 구문을 가진 PowerShell 프로세스를 생성하는 이벤트가 탐지된다. 따라서, Encoding된 명령어를 수행하는 powershell.exe 프로세스가 특정 명령 구문을 수행하는 프로세스를 생성하는 것을 탐지하도록 작성했다.

```
<Rule name="Cobalt Strike Command Execute(Jump-PsExec_psh)" groupRelation="and"> <!-- Cobalt Strike에서 jump psexec_psh 명령 수행 탐지 -->
  <Image condition="contains all">C:\Windows\;\WindowsPowerShell\;\powershell.exe</Image>
  <CommandLine condition="contains all">-Version;-s -NoLogo -NoProfile</CommandLine>
  <ParentImage condition="contains all">C:\Windows\;\WindowsPowerShell\;\powershell.exe</ParentImage>
  <ParentCommandLine condition="contains all">-nop;-encodedcommand</ParentCommandLine>
</Rule>
```

[그림 20] 인코딩된 명령어를 수행하는 powershell.exe 프로세스가 명령 구문을 수행하는 프로세스 생성 이벤트 탐지 룰

- **인자 값이 없는 rundll32.exe 프로세스 생성 이벤트 탐지**

Cobalt Strike Beacon이 실행되거나 특정 명령을 수행할 때, Beacon 프로세스가 인자 값이 없는 rundll32.exe 프로세스를 생성하는 이벤트가 탐지된다. 일반적으로 rundll32.exe 프로세스는 뒤에 인자 값으로 DLL 파일을 가지고 생성되기 때문에 인자 값이 없는 rundll32.exe 프로세스가 생성되는 것을 탐지하도록 작성했다.

```
<Rule name="rundll32.exe with No Arguments" groupRelation="and"> <!-- 인자 값이 없는 rundll32.exe 프로세스 탐지 -->
  <Image condition="contains all">C:\Windows\;\rundll32.exe</Image>
  <OriginalFileName condition="is">RUNDLL32.EXE</OriginalFileName>
  <CommandLine condition="end with">\rundll32.exe</CommandLine>
</Rule>
```

[그림 21] 인자 값이 없는 rundll32.exe 프로세스 생성 이벤트 탐지 룰

- **Pipe에 문자열을 전달하는 명령어를 가진 cmd.exe 프로세스 생성 이벤트 탐지**

Cobalt Strike Beacon이 pth 명령을 수행할 때 Pipe에 문자열을 전달하는 명령어를 가진 CMD 프로세스가 생성하는 이벤트가 탐지된다. 따라서, Pipe에 문자열을 전달하는 명령어를 가진 cmd.exe 프로세스를 생성하는 것을 탐지하도록 작성했다.

```
<Rule name="Cobalt Strike Command Execute(pth)" groupRelation="and"> <!-- Cobalt Strike에서 pth 명령 수행 탐지 -->
  <Image condition="contains all">C:\Windows\;\cmd.exe</Image>
  <CommandLine condition="contains all">C:\Windows\;\cmd.exe /c echo;>;\\.\pipe\</CommandLine>
</Rule>
```

[그림 22] Pipe에 문자열을 전달하는 명령어를 가진 cmd.exe 프로세스 생성 이벤트 탐지 룰

- **WinRM 프로세스가 부모인 프로세스 생성 이벤트 탐지**

Cobalt Strike Beacon이 특정 명령을 수행할 때 WinRM 프로세스를 활용하기 때문에 WinRM 프로세스가 부모 프로세스로 생성되는 이벤트가 탐지된다. 따라서, WinRM 프로세스인 wsmprovhost.exe가 부모인 프로세스로 생성되는 것을 탐지하도록 작성했다.

```
<Rule name="WinRM Process Creates Child Process" groupRelation="or"> <!-- WinRM 프로세스가 생성하는 프로세스 탐지 -->
  <ParentImage condition="contains all">C:\Windows\;\wsmprovhost.exe</ParentImage>
</Rule>
```

[그림 23] WinRM 프로세스가 부모인 프로세스 생성 이벤트 탐지 룰

- **WMI 프로세스가 부모인 프로세스 생성 이벤트 탐지**

Cobalt Strike Beacon이 특정 명령을 수행할 때 WMI 프로세스를 활용하기 때문에 WMI 프로세스가 부모 프로세스로 생성되는 이벤트가 탐지된다. 따라서, WMI 프로세스인 WmiPrvSE.exe가 부모인 프로세스로 생성되는 것을 탐지하도록 작성했다.

```
<Rule name="WMI Process Creates Child Process" groupRelation="or"> <!-- WMI 프로세스가 생성하는 프로세스 탐지 -->
  <ParentImage condition="contains all">C:\Windows\;\wbem\WmiPrvSE.exe</ParentImage>
</Rule>
```

[그림 24] WMI 프로세스가 부모인 프로세스 생성 이벤트 탐지 룰

2) NetworkConnect

NetworkConnect 이벤트는 네트워크를 통해 TCP/UDP 연결하는 프로세스에 대한 정보를 기록하는 이벤트 항목으로, Cobalt Strike 사용에 따라 네트워크 연결 이벤트를 탐지하는 Rule을 작성했다.

[표 96] Sysmon Configure Rule에서 NetworkConnect 이벤트를 통해 탐지할 수 있는 명령어 목록

번호	명령 카테고리	명령어	설명
1	-	Beacon	Cobalt Strike Beacon에서 사용할 수 있는 프로토콜을 통한 네트워크 연결 이벤트 탐지
2	사용자 활동 모니터링	desktop	rundll32.exe 프로세스가 LocalHost로 네트워크 연결하는 이벤트 탐지

NetworkConnect 이벤트 탐지와 관련해 작성된 Rule 설명은 다음과 같다.

- **Cobalt Strike Beacon 에서 사용할 수 있는 프로토콜을 통한 네트워크 연결 이벤트 탐지**

Cobalt Strike를 사용할 때 Beacon 프로세스에서 Team Server 또는 다른 Beacon과 지속적으로 네트워크 연결하는 이벤트가 탐지된다. 따라서, Cobalt Strike Beacon에서 사용할 수 있는 SSH, DNS, HTTP, HTTPS, SMB 등의 프로토콜을 통해 네트워크 연결되는 것을 탐지하도록 작성했다.

```
<!-- Ports Detected -->
<DestinationPort name="SSH" condition="is">22</DestinationPort> <!-- SSH 연결을 탐지 -->
<DestinationPort name="Telnet" condition="is">23</DestinationPort> <!-- Telnet 연결을 탐지 -->
<DestinationPort name="SMTP" condition="is">25</DestinationPort> <!-- SMTP 연결을 탐지 -->
<DestinationPort name="DNS" condition="is">53</DestinationPort> <!-- DNS 연결을 탐지 -->
<DestinationPort name="HTTP" condition="is">80</DestinationPort> <!-- HTTP 연결을 탐지 -->
<DestinationPort name="IMAP" condition="is">143</DestinationPort> <!-- IMAP 연결을 탐지 -->
<DestinationPort name="HTTPS" condition="is">443</DestinationPort> <!-- HTTPS 연결을 탐지 -->
<DestinationPort name="SMB" condition="is">445</DestinationPort> <!-- SMB 연결을 탐지 -->
<DestinationPort name="RDP" condition="is">3389</DestinationPort> <!-- RDP 연결을 탐지 -->
<DestinationPort name="VNC" condition="is">5800</DestinationPort> <!-- VNC 연결을 탐지 -->
<DestinationPort name="VNC" condition="is">5900</DestinationPort> <!-- VNC 연결을 탐지 -->
```

[그림 25] Cobalt Strike Beacon에서 사용할 수 있는 프로토콜을 통한 네트워크 연결 이벤트 탐지 룰

- **rundll32.exe 프로세스가 루프백 IP(127.0.0.1)로 네트워크 연결하는 이벤트 탐지**

Cobalt Strike Beacon이 desktop 명령을 수행할 때 기본 값으로 rundll32.exe 프로세스가 루프백 IP (127.0.0.1)에서 루프백 IP(127.0.0.1)로 네트워크 연결하는 이벤트가 탐지된다. 따라서, rundll32.exe 프로세스가 LocalHost로 네트워크 연결되는 것을 탐지하도록 작성했다.

```
<Rule name="Cobalt Strike Command Execute(Desktop)" groupRelation="and"> <!-- Cobalt Strike에서 desktop 명령어 수행 탐지 -->
  <Image condition="contains all">C:\Windows\;\rundll32.exe</Image>
  <SourceIp condition="is">127.0.0.1</SourceIp>
  <DestinationIp condition="is">127.0.0.1</DestinationIp>
</Rule>
```

[그림 26] rundll32.exe 프로세스가 루프백 IP(127.0.0.1)로 네트워크 연결하는 이벤트 탐지 룰

3) ImageLoad

ImageLoad 이벤트는 프로세스에서 모듈(DLL)이 로드될 때 프로세스에 대한 정보를 기록하는 이벤트 항목으로, Cobalt Strike 사용에 따라 특정 모듈(DLL)이 로드되는 이벤트를 탐지하는 Rule을 작성했다.

[표 97] Sysmon Configure Rule에서 ImageLoad 이벤트를 통해 탐지할 수 있는 명령어 목록

번호	명령 카테고리	명령어	설명
1	통신 및 데이터 암호화	covertvpn	rundll32.exe 프로세스가 NPcap DLL을 로드하는 이벤트 탐지
2	명령 및 스크립트 실행	powerpick	rundll32.exe 프로세스가 .NET 관련 DLL을 로드하는 이벤트 탐지

ImageLoad 이벤트 탐지와 관련해 작성된 Rule 설명은 다음과 같다.

- **rundll32.exe 프로세스가 NPcap DLL 을 로드하는 이벤트 탐지**

Cobalt Strike Beacon이 covertvpn 명령을 수행할 때 생성한 NPcap DLL 파일들을 rundll32.exe 프로세스에서 로드하는 이벤트가 탐지된다. 따라서, rundll32.exe 프로세스가 NPcap DLL(wpcap.dll, Packet.dll)을 로드하는 것을 탐지하도록 작성했다.

```
<!-- Cobalt Strike에서 covertvpn 명령 사용 탐지 -->
<Rule name="Cobalt Strike Command Execute(CovertVPN)" groupRelation="and">
  <Image condition="contains all">C:\Windows\;\rundll32.exe</Image>
  <ImageLoaded condition="contains all">C:\Users\;\AppData\Local\Temp\;\wpcap.dll</ImageLoaded>
</Rule>
<Rule name="Cobalt Strike Command Execute(CovertVPN)" groupRelation="and">
  <Image condition="contains all">C:\Windows\;\rundll32.exe</Image>
  <ImageLoaded condition="contains all">C:\Users\;\AppData\Local\Temp\;\Packet.dll</ImageLoaded>
</Rule>
```

[그림 27] rundll32.exe 프로세스가 NPcap DLL을 로드하는 이벤트 탐지 룰

- **rundll32.exe 프로세스가 .NET 관련 DLL 을 로드하는 이벤트 탐지**

Cobalt Strike Beacon이 powerpick 명령을 수행할 때 .NET 환경과 관련된 DLL 파일들을 rundll32.exe 프로세스에서 로드하는 이벤트가 탐지된다. 따라서, rundll32.exe 프로세스가 .NET관련 DLL(mscoreei.dll, mscorwks.dll, mscorjit.dll, Culture.dll)을 로드하는 것을 탐지하도록 작성했다.

```
<!-- Cobalt Strike에서 powerpick 명령 사용 탐지 -->
<Rule name="Cobalt Strike Command Execute(Powerpick)" groupRelation="and">
  <Image condition="contains all">C:\Windows\;\rundll32.exe</Image>
  <ImageLoaded condition="contains all">C:\Windows\Microsoft.NET\Framework64\;\mscoreei.dll</ImageLoaded>
</Rule>
<Rule name="Cobalt Strike Command Execute(Powerpick)" groupRelation="and">
  <Image condition="contains all">C:\Windows\;\rundll32.exe</Image>
  <ImageLoaded condition="contains all">C:\Windows\Microsoft.NET\Framework64\;\mscorwks.dll</ImageLoaded>
</Rule>
<Rule name="Cobalt Strike Command Execute(Powerpick)" groupRelation="and">
  <Image condition="contains all">C:\Windows\;\rundll32.exe</Image>
  <ImageLoaded condition="contains all">C:\Windows\Microsoft.NET\Framework64\;\mscorjit.dll</ImageLoaded>
</Rule>
<Rule name="Cobalt Strike Command Execute(Powerpick)" groupRelation="and">
  <Image condition="contains all">C:\Windows\;\rundll32.exe</Image>
  <ImageLoaded condition="contains all">C:\Windows\Microsoft.NET\Framework64\;\Culture.dll</ImageLoaded>
</Rule>
```

[그림 28] rundll32.exe 프로세스가 .NET 관련 DLL을 로드하는 이벤트 탐지 룰

4) CreateRemoteThread

CreateRemoteThread 이벤트는 다른 프로세스에서 스레드를 생성하는 프로세스에 대한 정보를 기록하는 이벤트 항목으로, Cobalt Strike 사용에 따라 원격 스레드를 생성하는 이벤트를 탐지하는 Rule을 작성했다.

[표 98] Sysmon Configure Rule에서 CreateRemoteThread 이벤트를 통해 탐지할 수 있는 명령어 목록

번호	명령 카테고리	명령어	설명
1	프로세스 제어	dllload	KERNEL32.DLL의 LoadLibraryA 함수를 사용하는 원격 스레드 생성 이벤트 탐지
2		dllinject	시스템 드라이브에서 원격 스레드 생성 이벤트 탐지
3		inject	시스템 드라이브에서 원격 스레드 생성 이벤트 탐지
4		psinject	시스템 드라이브에서 원격 스레드 생성 이벤트 탐지
5	권한 상승 및 크리덴셜 수집	hashdump	rundll32.exe 프로세스가 lsass.exe 프로세스에 원격 스레드 생성 이벤트 탐지
6	네트워크 트래픽 조작 및 우회	browserpivot	browserpivot 명령어 수행 시 발생하는 원격 스레드 생성 이벤트 탐지

CreateRemoteThread 이벤트 탐지와 관련해 작성된 Rule 설명은 다음과 같다.

- **KERNEL32.DLL 의 LoadLibraryA 함수를 사용하는 원격 스레드 생성 이벤트 탐지**

Cobalt Strike Beacon이 dllload 명령을 수행할 때 DLL을 로드하기 위해 원격 스레드를 생성해 LoadLibraryA 함수를 사용하는 이벤트가 탐지된다. 따라서, KERNEL32.DLL의 LoadLibraryA 함수를 사용하는 원격 스레드를 생성하는 것을 탐지하도록 작성했다.

```
<Rule name="DLL Load" groupRelation="and">
  <StartModule condition="contains all">C:\Windows\;\KERNEL32.DLL</StartModule>
  <StartFunction condition="is">LoadLibraryA</StartFunction>
</Rule>
```

[그림 29] KERNEL32.DLL LoadLibraryA 함수를 사용하는 원격 스레드 생성 이벤트 탐지 룰

- **시스템 드라이브에서 원격 스레드 생성 이벤트 탐지**

Cobalt Strike Beacon이 특정 명령을 수행할 때 원격 스레드를 생성하는 이벤트가 탐지된다. 따라서, 시스템 드라이브에서 원격 스레드를 생성하는 것을 탐지하도록 작성했다.

```
<Rule name="Process Injection" groupRelation="or">
  <SourceImage condition="begin with">C:\</SourceImage>
  <SourceImage condition="begin with">\\</SourceImage>
</Rule>
```

[그림 30] 시스템 드라이브에서 원격 스레드 생성 이벤트 탐지 룰

- **rundll32.exe 가 lsass.exe 프로세스에 원격 스레드 생성 이벤트 탐지**

Cobalt Strike Beacon이 hashdump 명령을 수행할 때 크리덴셜을 수집하기 위해 LSASS 프로세스에 원격 스레드를 생성하는 이벤트가 탐지된다. 따라서, rundll32.exe 프로세스가 lsass.exe 프로세스에 원격 스레드를 생성하는 것을 탐지하도록 작성했다.

```
<Rule name="Hash Dump" groupRelation="and">
  <SourceImage condition="contains all">C:\Windows\;\rundll32.exe</SourceImage>
  <TargetImage condition="is">C:\Windows\System32\lsass.exe</TargetImage>
</Rule>
```

[그림 31] rundll32.exe가 lsass.exe 프로세스에 원격 스레드 생성 이벤트 탐지 룰

- **browserpivot 명령어 실행 시 발생하는 원격 스레드 생성 이벤트 탐지**

Cobalt Strike Beacon이 browserpivot 명령을 수행할 때 IE Explorer 프로세스를 대상으로 원격 스레드를 생성하는 이벤트가 탐지된다. 따라서, iexplore.exe 프로세스를 대상으로 원격 스레드를 생성하는 것을 탐지하도록 작성했다.

```
<Rule name="Cobalt Strike Command Execute(BrowserPivot)" groupRelation="or"> <!-- Cobalt Strike에서 browserpivot 명령 사용 탐지 -->
  <TargetImage condition="is">C:\Program Files\internet explorer\iexplore.exe</TargetImage>
  <TargetImage condition="is">C:\Program Files (x86)\Internet Explorer\iexplore.exe</TargetImage>
</Rule>
```

[그림 32] browserpivot 명령어 실행 시 발생하는 원격 스레드 생성 이벤트 탐지 룰

5) ProcessAccess

ProcessAccess 이벤트는 다른 프로세스에 접근하는 프로세스에 대한 정보를 기록하는 이벤트 항목으로, Cobalt Strike 사용에 따라 다른 프로세스에 접근하는 이벤트를 탐지하는 Rule을 작성했다.

[표 99] Sysmon Configure Rule에서 ProcessAccess 이벤트를 통해 탐지할 수 있는 명령어 목록

번호	명령 카테고리	명령어	설명
1	프로세스 제어	shinject	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
2		shspawn	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
3	네트워크 탐색 및 이동	jump	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
4		net	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
5		portscan	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
6	권한 상승 및 크리덴셜 수집	chromedump	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
7		dcsync	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
8		hashdump	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
9		logonpasswords	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
10		mimikatz	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
11	사용자 활동 모니터링	desktop	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
12		keylogger	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
13		printscreen	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
14		screenshot	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
15		screenwatch	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
16	통신 및 데이터 암호화	covertvpn	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
17		spawn	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
18		spawnu	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
19		spunnel	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
20		spunnel_local	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지

번호	명령 카테고리	명령어	설명
21	명령 및 스크립트 실행	execute	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
22		execute-assembly	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
23		powerpick	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
24		powershell	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
25		runu	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지
26		shell	특정 Cobalt Strike 명령어 수행 시 발생하는 프로세스 접근 이벤트 탐지

ProcessAccess 이벤트 탐지와 관련해 작성된 Rule 설명은 다음과 같다.

- **특정 Cobalt Strike 명령어 실행 시 발생하는 프로세스 접근 이벤트 탐지**

Cobalt Strike Beacon이 특정 명령을 수행할 때 프로세스를 생성하고 해당 프로세스에 접근하는 이벤트가 기록된다. 프로세스에 접근할 때 GrantedAccess(프로세스 권한에 관한 Access Flags)의 값과 CallTrace(열린 프로세스를 호출하는 위치)의 값이 고정되어 있기 때문에 특정 값으로 접근하는 것을 탐지하도록 작성했다. 정상 프로세스에서도 해당 값으로 접근할 수 있어 다른 Rule에서 탐지한 이벤트들과 같이 분석해야 한다.

```
<Rule name="Cobalt Strike Command Execute" groupRelation="and">
  <CallTrace condition="contains">UNKNOWN</CallTrace>
  <GrantedAccess condition="contains any">0x1028;0x1fffff</GrantedAccess>
</Rule>
```

[그림 33] 특정 Cobalt Strike 명령어 실행 시 발생하는 프로세스 접근 이벤트 탐지 룰

6) FileCreate

FileCreate 이벤트는 파일을 생성하거나 덮어쓰는 프로세스에 대한 정보를 기록하는 이벤트 항목으로, Cobalt Strike 사용에 따라 파일이 생성하는 이벤트를 탐지하는 Rule을 작성했다.

[표 100] Sysmon Configure Rule에서 FileCreate 이벤트를 통해 탐지할 수 있는 명령어 목록

번호	명령 카테고리	명령어	설명
1	통신 및 데이터 암호화	covertvpn	- rundll32.exe 프로세스가 %TEMP% 하위 경로에 파일 생성 이벤트 탐지

FileCreate 이벤트 탐지와 관련해 작성된 Rule 설명은 다음과 같다.

- **rundlle2.exe 프로세스가 %TEMP% 하위 경로에 파일을 생성하는 이벤트 탐지**

Cobalt Strike Beacon이 covertvpn 명령을 수행할 때 %TEMP% 하위 경로에 NPcap 파일들을 생성하는 이벤트가 탐지된다. 따라서, rundll32.exe 프로세스가 %TEMP% 하위 경로에 NPcap 파일(wpcap.dll, Packet.dll, npf.sys)을 생성하는 것을 탐지하도록 작성했다.

```
<!-- Cobalt Strike에서 covertvpn 명령 사용 탐지 -->
<Rule name="Cobalt Strike Command Execute(CovertVPN)" groupRelation="and">
  <Image condition="contains all">C:\Windows\;\rundll32.exe</Image>
  <TargetFilename condition="contains all">C:\Users\;\AppData\Local\Temp\;\wpcap.dll</TargetFilename>
</Rule>
<Rule name="Cobalt Strike Command Execute(CovertVPN)" groupRelation="and">
  <Image condition="contains all">C:\Windows\;\rundll32.exe</Image>
  <TargetFilename condition="contains all">C:\Users\;\AppData\Local\Temp\;\Packet.dll</TargetFilename>
</Rule>
<Rule name="Cobalt Strike Command Execute(CovertVPN)" groupRelation="and">
  <Image condition="contains all">C:\Windows\;\rundll32.exe</Image>
  <TargetFilename condition="contains all">C:\Users\;\AppData\Local\Temp\;\npf.sys</TargetFilename>
</Rule>
```

[그림 34] rundll32.exe 프로세스가 %TEMP% 하위 경로에 파일을 생성하는 이벤트 탐지 룰

7) RegistryEvent

RegistryEvent 이벤트는 레지스트리 키 또는 값을 조작(생성, 수정, 삭제 등)하는 프로세스에 대한 정보를 기록하는 이벤트 항목으로, Cobalt Strike 사용에 따라 레지스트리 키 또는 값을 조작하는 이벤트를 탐지하는 Rule을 작성했다.

[표 101] Sysmon Configure Rule에서 RegistryEvent 이벤트를 통해 탐지할 수 있는 명령어 목록

번호	명령 카테고리	명령어	설명
1	-	Beacon	- Cobalt Strike Beacon 실행 시 발생하는 레지스트리 이벤트 탐지
2	네트워크 탐색 및 이동	jump	- 레지스트리를 통해 ADMIN$ 경로의 실행 파일을 서비스로 등록하는 이벤트 탐지 - 레지스트리를 통해 PowerShell 명령어를 서비스로 등록하는 이벤트 탐지
3	권한 상승 및 크리덴셜 수집	elevate	- 레지스트리를 통해 ADMIN$ 경로의 실행 파일을 서비스로 등록하는 이벤트 탐지
4	통신 및 데이터 암호화	covertvpn	- 레지스트리를 통해 NPcap 파일을 서비스로 등록하는 이벤트 탐지

RegitryEvent 이벤트 탐지와 관련해 작성된 Rule 설명은 다음과 같다.

- **Cobalt Strike Beacon 실행 시 발생하는 레지스트리 이벤트 탐지**

Cobalt Strike Beacon이 실행될 때 로컬 네트워크 설정에 대한 레지스트리 값을 변경하는 이벤트가 탐지된다. 따라서, 로컬 네트워크 설정과 관련된 레지스트리 키가 생성하고, 값을 변경해 보안 설정을 비활성화하는 것을 탐지하도록 작성했다.

```
<!-- Cobalt Strike의 Beacon 실행 시 Internet ZoneMap Setting이 변경되는 것을 탐지 -->
<Rule name="Change Internet ZoneMap Setting" groupRelation="and">
  <EventType condition="is">CreateKey</EventType>
  <TargetObject condition="contains all">HKU\;\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap</TargetObject>
</Rule>
<Rule name="Change Internet ZoneMap Setting" groupRelation="and">
  <EventType condition="is">SetValue</EventType>
  <TargetObject condition="contains all">HKU\;\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\UNCAsIntranet</TargetObject>
  <Details condition="is">DWORD (0x00000000)</Details>
</Rule>
<Rule name="Change Internet ZoneMap Setting" groupRelation="and">
  <EventType condition="is">SetValue</EventType>
  <TargetObject condition="contains all">HKU\;\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\AutoDetect</TargetObject>
  <Details condition="is">DWORD (0x00000000)</Details>
</Rule>
```

[그림 35] Cobalt Strike Beacon 실행 시 발생하는 레지스트리 이벤트 탐지 룰

- **레지스트리를 통해 ADMIN$ 경로의 실행 파일을 서비스로 등록하는 이벤트 탐지**

Cobalt Strike Beacon이 elevate, jump 명령을 수행할 때 ADMIN$ 경로의 실행 파일을 서비스로 등록하는 이벤트가 탐지된다. 따라서, 레지스트리를 통해 ADMIN$ 경로의 실행 파일이 서비스로 등록되는 것을 탐지하도록 작성했다.

```
<Rule name="Create Service in ADMIN$" groupRelation="and"> <!-- ADMIN$ 경로의 실행파일이 서비스로 실행되도록 등록 -->
  <EventType condition="is">SetValue</EventType>
  <TargetObject condition="contains all">HKLM\System\CurrentControlSet\Services\;\ImagePath</TargetObject>
  <Details condition="contains all">\ADMIN$\;.exe</Details>
</Rule>
```

[그림 36] 레지스트리를 통해 ADMIN$ 경로의 실행 파일을 서비스로 등록하는 이벤트 탐지 룰

- **레지스트리를 통해 파워셸 명령을 서비스로 등록하는 이벤트 탐지**

Cobalt Strike Beacon이 jump 명령을 수행할 때 Encoding된 PowerShell 명령어를 서비스로 등록하는 이벤트가 탐지된다. 따라서, 레지스트리를 통해 CMD로 인코딩된 PowerShell 명령어가 서비스로 등록되는 것을 탐지하도록 작성했다.

```
<Rule name="Create Service in PowerShell" groupRelation="and"> <!-- PowerShell 명령어가 서비스로 실행되도록 등록 -->
  <EventType condition="is">SetValue</EventType>
  <TargetObject condition="contains all">HKLM\System\CurrentControlSet\Services\;\ImagePath</TargetObject>
  <Details condition="contains all">COMSPEC;powershell;-nop;-encodedcommand</Details>
</Rule>
```

[그림 37] 레지스트리를 통해 파워셸 명령을 서비스로 등록하는 이벤트 탐지

- **레지스트리를 통해 NPcap 파일을 서비스로 등록하는 이벤트 탐지**

Cobalt Strike Beacon이 covertvpn 명령을 수행할 때 생성한 NPcap 시스템 파일을 서비스로 등록하는 이벤트가 탐지된다. 따라서, 레지스트리를 통해 NPcap 시스템 파일(npf.sys)이 서비스로 등록되는 것을 탐지하도록 작성했다.

```
<!-- NPcap 설치 탐지(Cobalt Strike의 covertvpn 명령어에서 사용) -->
<Rule name="Install NPcap" groupRelation="and">
  <EventType condition="is">CreateKey</EventType>
  <TargetObject condition="is">HKLM\System\CurrentControlSet\Services\npf</TargetObject>
</Rule>
<Rule name="Install NPcap" groupRelation="and">
  <EventType condition="is">SetValue</EventType>
  <TargetObject condition="is">HKLM\System\CurrentControlSet\Services\npf\ImagePath</TargetObject>
  <Details condition="contains all">C:\Users\;\AppData\Local\Temp\;\npf.sys</Details>
</Rule>
```

[그림 38] 레지스트리를 통해 NPcap 파일을 서비스로 등록하는 이벤트 탐지

8) PipeEvent

PipeEvent 이벤트는 명명된 Pipe를 생성하거나 연결하는 프로세스에 대한 정보를 기록하는 이벤트 항목으로 Cobalt Strike 사용에 따라 명명된 Pipe를 생성하거나 연결하는 이벤트를 탐지하는 Rule을 작성했다.

[표 102] Sysmon Configure Rule에서 PipeEvent 이벤트를 통해 탐지할 수 있는 명령어 목록

번호	명령 카테고리	명령어	설명
1	-	Beacon	Cobalt Strike 실행 시 발생하는 명명된 Pipe 이벤트 탐지
2	Beacon 제어 및 관리	link	Cobalt Strike 실행 시 발생하는 명명된 Pipe 이벤트 탐지
3	프로세스 제어	psinject	Cobalt Strike 실행 시 발생하는 명명된 Pipe 이벤트 탐지
4	네트워크 탐색 및 이동	jump	Cobalt Strike 실행 시 발생하는 명명된 Pipe 이벤트 탐지
5		net	Cobalt Strike 실행 시 발생하는 명명된 Pipe 이벤트 탐지
6		portscan	Cobalt Strike 실행 시 발생하는 명명된 Pipe 이벤트 탐지
7		remote-exec	Cobalt Strike 실행 시 발생하는 명명된 Pipe 이벤트 탐지
8	권한 상승 및 크리덴셜 수집	chromedump	Cobalt Strike 실행 시 발생하는 명명된 Pipe 이벤트 탐지
9		dcsync	Cobalt Strike 실행 시 발생하는 명명된 Pipe 이벤트 탐지
10		hashdump	Cobalt Strike 실행 시 발생하는 명명된 Pipe 이벤트 탐지
11		logonpasswords	Cobalt Strike 실행 시 발생하는 명명된 Pipe 이벤트 탐지
12		mimikatz	Cobalt Strike 실행 시 발생하는 명명된 Pipe 이벤트 탐지
13		pth	Cobalt Strike 실행 시 발생하는 명명된 Pipe 이벤트 탐지
14	사용자 활동 모니터링	keylogger	Cobalt Strike 실행 시 발생하는 명명된 Pipe 이벤트 탐지
15		printscreen	Cobalt Strike 실행 시 발생하는 명명된 Pipe 이벤트 탐지
16		screenshot	Cobalt Strike 실행 시 발생하는 명명된 Pipe 이벤트 탐지
17		screenwatch	Cobalt Strike 실행 시 발생하는 명명된 Pipe 이벤트 탐지
18	명령 및 스크립트 실행	powerpick	Cobalt Strike 실행 시 발생하는 명명된 Pipe 이벤트 탐지

PipeEvent 이벤트 탐지와 관련해 작성된 Rule 설명은 다음과 같다.

- **Cobalt Strike 실행 시 발생하는 명명된 Pipe 이벤트 탐지**

Cobalt Strike Beacon이 실행될 때 명명된 Pipe를 생성 및 연결하는 이벤트가 탐지된다. 따라서, Cobalt Strike Beacon 실행 시 사용할 수 있는 명명된 Pipe가 생성되고 연결하는 것을 탐지하도록 작성했다.

```
<Rule name="Turn on Cobalt Strike Beacon" groupRelation="or"> <!-- Cobalt Strike의 Beacon이 실행 -->
  <PipeName condition="contains all">MSSE-;-server</PipeName>
</Rule>
```

[그림 39] Cobalt Strike 실행 시 발생하는 명명된 Pipe 이벤트 탐지 룰

- **Cobalt Strike 명령어 실행 시 발생하는 명명된 Pipe 이벤트 탐지**

Cobalt Strike Beacon이 특정 명령을 수행할 때 명명된 Pipe를 생성 및 연결하는 이벤트가 탐지된다. 따라서, Cobalt Strike Beacon이 명령어 수행 시 사용할 수 있는 명명된 Pipe가 생성되고 연결하는 것을 탐지하도록 작성했다.

```
<Rule name="Cobalt Strike Command Execute" groupRelation="or"> <!-- Cobalt Strike의 Beacon이 명령 수행 -->
  <PipeName condition="begin with">\postex_</PipeName>
  <PipeName condition="begin with">\postex_ssh_</PipeName>
  <PipeName condition="begin with">\status_</PipeName>
  <PipeName condition="begin with">\msagent_</PipeName>
</Rule>
```

[그림 40] Cobalt Strike 명령어 실행 시 발생하는 명명된 Pipe 이벤트 탐지 룰

9) ProcessTampering

ProcessTampering 이벤트는 프로세스를 변조하는 프로세스에 대한 정보를 기록하는 이벤트 항목으로 Cobalt Strike 사용에 따라 프로세스를 변조하는 이벤트를 탐지하는 Rule을 작성했다.

[표 103] Sysmon Configure Rule에서 ProcessTampering 이벤트를 통해 탐지할 수 있는 명령어 목록

번호	명령 카테고리	명령어	설명
1	권한 상승 및 크리덴셜 수집	elevate	모든 프로세스 변조 이벤트 탐지

ProcessTampering 이벤트 탐지와 관련해 작성된 Rule 설명은 다음과 같다.

- **모든 프로세스 변조 이벤트 탐지**

Cobalt Strike Beacon이 elevate 명령을 수행할 때 프로세스 변조 이벤트가 탐지된다. Cobalt Strike 명령 수행으로 구분할 수 없지만 프로세스 변조는 악성 행위의 가능성이 높아 모든 프로세스 변조 이벤트를 탐지하도록 작성했다.

```
<ProcessTampering onmatch="exclude"> <!-- 모든 프로세스 변조 행위를 기록 -->
</ProcessTampering>
```

[그림 41] 모든 프로세스 변조 이벤트 탐지 룰

6.3.1.2. 주요 개선 사항

2023년에 작성했던 Sysmon Configure Rule과 비교했을 때, 크게 2가지의 사항이 개선되었다. 주요 개선된 사항은 다음과 같다.

1) Cobalt Strike 탐지 Rule 추가

Cobalt Strike를 통한 공격을 수행할 경우, 시스템에 남기는 이벤트를 분석해 Cobalt Strike의 공격 흔적을 탐지하는 Rule이 추가되었다. 공격자가 목적 달성을 수행하기 전에 빠르게 원인을 규명하고 내부 이동 경로와 전파 방식을 파악하는 데 도움을 줄 수 있는 정보를 제공한다.

2) Sysmon Configure Rule 최적화

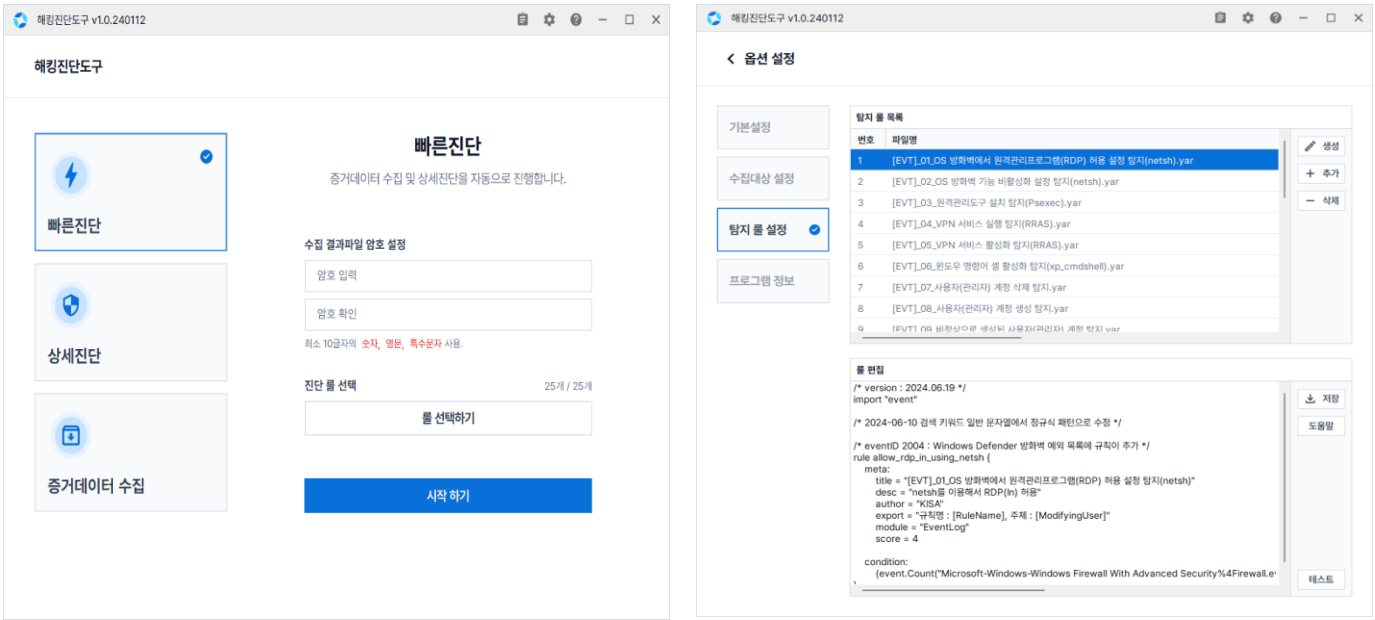
기존 Sysmon Configure Rule에서 ProcessCreate, DriverLoad, ImageLoad 이벤트 항목에 악성 파일의 해시 값을 탐지하도록 작성했다. Rule 파일에 악성 해시 값이 포함되다 보니 백신(AV)에서 악성 파일로 탐지되는 불편함이 존재했다. 따라서, 악성 파일의 해시 값을 탐지하는 Rule을 사용자가 필요에 따라 작성할 수 있도록 모두 제거했다. 최종적으로 개선된 Sysmon Configure Rule 현황은 다음과 같다.

이벤트 ID	이벤트 명	include	exclude	비고
1	ProcessCreate	229	150	-
2	FileCreateTime	4	10	-
3	NetworkConnect	42	12	-
5	ProcessTerminate	8	5	-
6	DriverLoad	0	3	-
7	ImageLoad	39	16	-
8	CreateRemoteThread	6	9	-
9	RawAccessRead	-	-	기록하지 않음
10	ProcessAccess	27	15	-
11	FileCreate	50	23	-
12, 13, 14	RegistryEvent	50	3	-
15	FileCreateStreamHash	13	-	모두 기록
17, 18	PipeEvent	12	3	-
19, 20, 21	WmiEvent	0	-	모두 기록
22	DNSQuery	0	92	-
23	FileDelete	40	4	-
24	ClipboardChange	-	-	기록하지 않음
25	ProcessTampering	0	-	모두 기록
26	FileDeleteDetected	-	-	기록하지 않음
27	FileBlockExecutable	-	-	기록하지 않음
28	FileBlockShredding	-	-	기록하지 않음
29	FileExecutableDetected	1	0	-

[그림 42] 본 연구를 통해 개선한 Sysmon Configure Rule

6.3.2. 한국인터넷진흥원 해킹진단도구 진단 룰

해킹진단도구[28]는 한국인터넷진흥원(KISA)이 가진 침해사고 분석 전문 지식과 노하우를 플랫폼화해 일반기업도 쉽고 간단하게 해킹 여부를 점검할 수 있는 진단도구이다. 대기업, 비영리 기업 제한 없이 민간기업 전체를 대상으로 기업에서 운영하는 윈도우 서버(Windows Server 2008 이상)를 진단할 수 있다. 해킹진단도구의 빠른 진단 기능을 통해 누구나 쉽게 원클릭으로 해킹 여부를 진단하고, 상세 진단 기능을 통해 많은 시스템의 해킹 여부를 한 번에 진단할 수 있다. 해킹진단도구에는 현재 총 25개의 탐지 룰(이벤트 로그 21개, 레지스트리 4개)이 포함되어 배포되고 있으며, 공격자 내부 이동 단계에서 주로 사용하는 행위를 탐지한다.



[그림 43] 해킹진단도구 메인 화면 및 탐지 룰 목록

해킹진단도구의 탐지 룰은 Cobalt Strike에서 위협이라고 특정할 수 있는 지표를 진단할 수 있는 룰과 공격이라고 특정할 수 있는 지표를 진단할 수 있는 룰로 구분해 작성했다. 해킹진단도구에서는 탐지된 지표의 위험도 점수를 1점(관심)부터 4점(심각)까지 측정해 일반인들도 진단된 결과를 직관적으로 확인할 수 있도록 제공하고 있으며, 작성한 해킹진단도구 탐지 룰은 아래와 같은 기준으로 위험도를 측정했다.

[표 104] 해킹진단도구 탐지 룰 작성 기준

위험도 점수	상태	조건
4	심각	Cobalt Strike를 통한 위협이라고 특정할 수 있는 지표
3	경계	Cobalt Strike를 통한 위협이라고 판단하기는 어렵지만, 공격이라고 특정할 수 있는 지표
2	주의	해당 지표만으로는 공격이라고 판단하기는 어렵지만, 공격에 많이 활용되는 이벤트
1	관심	해당 지표만으로는 공격이라고 판단하기 어려운 이벤트

[28] https://boho.or.kr/kr/subPage.do?menuNo=205112

6.3.2.1. 주요 Rule 설명

본 연구를 통해 작성된 주요 해킹진단도구 탐지 룰 목록은 다음과 같다.

[표 105] Cobalt Strike를 활용한 행위를 탐지할 수 있는 주요 해킹진단도구 탐지 룰 목록

탐지 모듈	탐지 대상 이벤트 구분	탐지 룰 목록
EventLog	Security.evtx	네트워크 공유 개체 접근
		비정상 윈도우 로그온 탐지
	System.evtx	네트워크 공유를 통한 비정상 서비스 설치
		비정상 PowerShell 명령이 포함된 서비스 설치
		NPcap 서비스 설치
	Windows PowerShell.evtx	Command가 없는 비정상 PowerShell 명령
	Microsoft-Windows-WinRM.evtx	PowerShell을 활용하는 WinRM 탐지
	Microsoft-Windows-Sysmon/Operation.evtx	Cobalt Strike 프로세스 통신 Pipe 생성 및 연결 탐지
		Cobalt Strike Beacon(HTML Application) 실행 탐지
		Cobalt Strike browserpivot 명령어 실행 탐지
		Cobalt Strike desktop 명령어 실행 탐지
		Cobalt Strike jump psexec_psh 명령어 실행 탐지
		Cobalt Strike powerpick 명령어 실행 탐지
		Cobalt Strike pth 명령어 실행 탐지
		프로세스 접근 이벤트 탐지
		비정상 lsass.exe 접근 이벤트 탐지
		ADMIN$ 경로에 실행 파일 생성 이벤트 탐지
		ADMIN$ 경로에 실행 파일 삭제 이벤트 탐지
		ADMIN$ 경로에 실행 파일이 서비스로 등록되는 이벤트 탐지
		인자가 존재하지 않는 비정상 Rundll32.exe 실행 탐지
		NPcap 파일 생성
		NPcap 이미지 로드
Registry	SYSTEM	난독화된 PowerShell 명령어가 레지스트리 서비스 등록 탐지
		ADMIN$ 경로의 실행 파일이 레지스트리 서비스 등록 탐지

1) [EventLog] Security.evtx

Security.evtx를 통해 총 4개의 이벤트를 탐지할 수 있도록 룰을 작성했다.

[표 106] 이벤트 로그(Security.evtx)에서 해킹진단도구 탐지 룰을 통해 탐지할 수 있는 명령어 목록

번호	명령 카테고리	명령어	탐지 룰 이름
1	네트워크 탐색 및 이동	jump	네트워크 공유 개체 접근
2		net	네트워크 공유 개체 접근
3	권한 및 크리덴셜 수집	pth	비정상 윈도우 로그온 탐지
4		elevate	네트워크 공유 개체 접근

Security.evtx 이벤트 로그 대상으로 작성된 탐지 룰 설명은 다음과 같다.

- **네트워크 공유 개체 접근**

Cobalt Strike Beacon이 jump, net, elevate 명령을 수행할 때 네트워크 공유 개체(ADMIN$, IPC$)에 접근하는 이벤트가 발생한다. 따라서, Security.evtx의 이벤트 ID가 5140인 이벤트 로그에서 Loopback IP(127.0.0.1, ::1 등)에서 네트워크 공유 개체(ADMIN$, IPC$)에 접근하는 이벤트가 발생했는지 진단하도록 작성했다.

```
/* version : 2024.10.15 */
import "event"

/* CobaltStrike elvate, jump access sharefolder */
/* SubjectUserSid=S-1-5-18, ObjectType=File, IpAddress=127.0.0.1, IpPort=%N%, ShareName=\\*\ADMIN$ */
/*SubjectUserName=%Account running Beacon%, ObjectType=File, IpAddress=fe80::*, IpPort=%N%, ShareName=\\*\ADMIN$*/


private rule Access_ShareFolder_loopbackipv4{
    condition:
        (event.Count("Security.evtx","EventID:5140","ObjectType:/File/", "IpAddress:/127\\.0\\.0\\.1/", "ShareName:/\\\\*\\ADMIN\\$/") > 0) or
        (event.Count("Security.evtx","EventID:5140","ObjectType:/File/", "IpAddress:/127\\.0\\.0\\.1/", "ShareName:/\\\\*\\IPC\\$/") > 0) or
        (event.Count("Security.evtx","EventID:5140","ObjectType:/File/", "IpAddress:/::1/", "ShareName:/\\\\*\\ADMIN\\$/") > 0) or
        (event.Count("Security.evtx","EventID:5140","ObjectType:/File/", "IpAddress:/::1/", "ShareName:/\\\\*\\IPC\\$/") > 0)
}

private rule Access_ShareFolder_loopbackipv6{
    condition:
        (event.Count("Security.evtx","EventID:5140","ObjectType:/File/", "IpAddress:/fe80::\\.*/", "ShareName:/\\\\*\\ADMIN\\$/") > 0) or
        (event.Count("Security.evtx","EventID:5140","ObjectType:/File/", "IpAddress:/fe80::\\.*/", "ShareName:/\\\\*\\IPC\\$/") > 0)
}

rule System_5140_Access_ShareFolder{
    meta:
        title = "[EVT] System_5140_Access_ShareFolder"
        desc = "루프백 아이피를 통한 네트워크 공유 폴더 접근"
        author = "PLAINBIT"
        export = "사용자 명: [SubjectUserName], IP: [IpAddress], 공유폴더 이름: [ShareName], 공유폴더 경로: [ShareLocalPath]"
        module = "EventLog"
        score = 2
    condition:
        Access_ShareFolder_loopbackipv4 or Access_ShareFolder_loopbackipv6
}
```

[그림 44] 네트워크 공유 개체 접근 이벤트 탐지 룰

- **비정상 윈도우 로그온 탐지**

Cobalt Strike Beacon이 pth 명령을 수행할 때 주로 사용되지 않는 프로세스를 통한 비정상 윈도우 로그온 이벤트가 발생한다. 따라서, Security.evtx의 이벤트 ID가 4624인 이벤트 로그에서 Logon Type이 9이면서 seclogo, svchost.exe 프로세스를 사용하는 이벤트가 발생했는지 진단하도록 작성했다.

```
/* version : 2024.10.15 */
import "event"

rule PowerShell_400_Suspicious_Network_Logon{
    meta:
        title = "[EVT] Security_4624_Suspicious_Network_Logon"
        desc = "NTLM Hash를 통한 네트워크 로그온 탐지"
        author = "PLAINBIT"
        export = "로그온 계정: [TargetUserName], 로그온 이전 계정: [TargetOutboundUserName], IP: [IpAddress]"
        module = "EventLog"
        score = 3
    condition:
        (event.Count("Security.evtx","EventID:4624","LogonType:9", "LogonProcessName:seclogo", "ProcessName:/\\\\Windows\\\\System32\\\\svchost\\.exe/") > 0)
}
```

[그림 45] 비정상 윈도우 로그온 이벤트 탐지 룰

2) [EventLog] System.evtx

System.evtx를 통해 총 3개의 이벤트를 탐지할 수 있도록 룰을 작성했다.

[표 107] 이벤트 로그(System.evtx)에서 해킹진단도구 탐지 룰을 통해 탐지할 수 있는 명령어 목록

번호	명령 카테고리	명령어	탐지 룰 이름
1	네트워크 탐색 및 이동	jump	- 네트워크 공유를 통한 비정상 서비스 설치 - 비정상 PowerShell 명령이 포함된 서비스 설치
2	권한 및 크리덴셜 수집	elevate	- 네트워크 공유를 통한 비정상 서비스 설치
3	통신 및 데이터 암호화	covertvpn	- NPcap 서비스 설치

System.evtx 이벤트 로그 대상으로 작성된 탐지 룰 설명은 다음과 같다.

- **네트워크 공유를 통한 비정상 서비스 설치**

Cobalt Strike Beacon이 jump, elevate 명령을 수행할 때 ADMIN$ 경로의 랜덤한 파일 명을 가진 실행 파일을 서비스로 등록하는 이벤트가 발생한다. 따라서, System.evtx의 이벤트 ID가 7045인 이벤트 로그에서 등록된 파일이 ADMIN$ 경로의 실행 파일인 이벤트가 발생했는지 진단하도록 작성했다.

```
/* version : 2024.10.15 */
import "event"

/* CobaltStrike elevate svc-exe and jump*/
/* ServiceName=%Random%, ImagePath=\\127.0.0.1\ADMIN$\%Random%.exe */
/* ServiceName={Random}, ImagePath=\\%HOSTNAME%\ADMIN$\%Random%.exe */


private rule ServiceInstall_ShareFolder_loopback{
    condition:
        (event.Count("System.evtx","EventID:7045","ServiceName:/.*/", "ImagePath:/\\\\127\\.0\\.0\\.1\\\\ADMIN\\$\\\\.*\\.exe/") > 0)
}

private rule ServiceInstall_ShareFolder_hostname{
    condition:
        (event.Count("System.evtx","EventID:7045","ServiceName:/.*/", "ImagePath:/\\\\__MYHOST__\\\\ADMIN\\$\\\\.*\\.exe/") > 0)
}

rule System_7045_ServiceInstall_ShareFolder{
    meta:
        title = "[EVT] System_7045_ServiceInstall_ShareFolder"
        desc = "네트워크 공유를 통한 비정상 서비스 설치"
        author = "PLAINBIT"
        export = "서비스 명: [ServiceName], 파일: [ImagePath]"
        module = "EventLog"
        score = 3
    condition:
        ServiceInstall_ShareFolder_loopback or ServiceInstall_ShareFolder_hostname
}
```

[그림 46] 네트워크 공유를 통한 비정상 설치 이벤트 탐지 룰

- **비정상 파워셸 명령이 포함된 서비스 설치**

Cobalt Strike Beacon이 jump 명령을 수행할 때 Encoding된 PowerShell 명령어를 실행하도록 서비스로 등록하는 이벤트가 발생한다. 따라서, System.evtx의 이벤트 ID가 7045인 이벤트 로그에서 등록된 명령이 CMD를 통해 Encoding된 PowerShell 명령어를 실행하는 이벤트가 발생했는지 진단하도록 작성했다.

```
/* version : 2024.10.15 */
import "event"

/* CobaltStrike jump psexec_psh */
/* ServiceName=%Random%, ImagePath=%COMSPEC% /b /c start /b /min powershell -nop -w hidden -encodedcommand %Encoding Code% */


private rule ServiceInstall_suspicious_powershell{
    condition:
        (event.Count("System.evtx","EventID:7045","ServiceName:/.*/", "ImagePath:/%COMSPEC% \\/b \\/c start \\/b \\/min powershell \\-nop \\-w hidden \\-encodec
}

rule System_7045_ServiceInstall_PowerShell{
    meta:
        title = "[EVT] System_7045_Serviceinstall_PowerShell"
        desc = "비정상 PowerShell 명령이 포함된 서비스 설치"
        author = "PLAINBIT"
        export = "서비스 명: [ServiceName], 파일: [ImagePath]"
        module = "EventLog"
        score = 3
    condition:
        ServiceInstall_suspicious_powershell
}
```

[그림 47] 비정상 파워셸 명령이 포함된 서비스 설치 이벤트 탐지 룰

- **NPcap 서비스 설치**

Cobalt Strike Beacon이 covertvpn 명령을 수행할 때 생성한 %Temp% 경로의 NPcap 시스템 파일을 서비스로 등록하는 이벤트가 발생한다. 따라서, System.evtx의 이벤트 ID가 7045인 이벤트 로그에서 등록된 파일이 %Temp% 경로의 NPcap 시스템 파일(npf.sys)인 이벤트가 발생했는지 진단하도록 작성했다.

```
/* version : 2024.10.15 */
import "event"

/* CobaltStrike CovertVPN install NPcap Service */
/* ServiceName=npf, ImagePath=C:\Users\ADMINI~1\AppData\Local\Temp\1\npf.sys */


private rule ServiceInstall_NPcap_Driver{
    condition:
        (event.Count("System.evtx","EventID:7045","ServiceName:/npf/","ImagePath:/\\AppData\\\\Local\\\\Temp\\\\.*\\\\npf\\.sys$/") > 0) or
        (event.Count("System.evtx","EventID:7045","ServiceName:/npf/","ImagePath:/\\\\Windows\\\\Temp\\\\npf\\.sys$/") > 0)
}

rule System_7045_ServiceInstall_NPcap_Driver{
    meta:
        title = "[EVT] System_7045_ServiceInstall_NPcap_Driver"
        desc = "NPcap 서비스 설치"
        author = "PLAINBIT"
        export = "서비스 명: [ServiceName], 파일: [ImagePath]"
        module = "EventLog"
        score = 1
    condition:
        ServiceInstall_NPcap_Driver
}
```

[그림 48] NPcap 서비스 설치 이벤트 탐지 룰

3) [EventLog] Windows PowerShell.evtx

Windows PowerShell.evtx를 통해 총 1개의 이벤트를 탐지할 수 있도록 룰을 작성했다.

[표 108] 이벤트 로그(Windows PowerShell.evtx)에서 해킹진단도구 탐지 룰을 통해 탐지할 수 있는 명령어 목록

번호	명령 카테고리	명령어	탐지 룰 이름
1	명령 및 스크립트 실행	powerpick	Command가 없는 비정상 PowerShell 명령 탐지

Windows PowerShell.evtx 이벤트 로그 대상으로 작성된 탐지 룰 설명은 다음과 같다.

- **Command 가 없는 비정상 파워셸 명령 탐지**

Cobalt Strike Beacon이 powerpick 명령을 수행할 때 메모리 상에서 명령이 수행되기 때문에 Command 필드 내 데이터가 없는 이벤트가 발생한다. 따라서, Windows PowerShell.evtx의 이벤트 ID가 400인 이벤트 로그에서 HostApplication 필드가 없는 이벤트가 발생했는지 진단하도록 작성했다.

```
/* version : 2024.10.15 */
import "event"

/* CobaltStrike WinRM PowerShell */
/* 600  PowerShell 실행  HostApplication=powershell -nop -w hidden -encodedcommand %Encoding Code% */
/* 6  WSMan 세션 생성  connection=%HOSTNAME%/wsman?PSVersion=5.1.17763.2931 */
/* 91 WSMan 세션 생성  resourceUri=http://schemas.microsoft.com/powershell/Microsoft.PowerShell (%Account running Beacon% clientIP: %Parent Host IP%) */



private rule PowerShell_using_WinRM{
    condition:
        (event.Count("Microsoft-Windows-WinRM%4Operational.evtx","EventID:6","connection:/.*\\/wsman\\?PSVersion=\\.*/") > 0)
}

private rule Create_WinRM{
    condition:
        (event.Count("Microsoft-Windows-WinRM%4Operational.evtx","EventID:91","resourceUri:/ServerManagerWorkflows \\(.* clientIP: .*/") > 0)
}

rule WinRM_6_91_Suspicious_WinRM_PowerShell{
    meta:
        title = "[EVT] WinRM_6_91_Suspicious_WinRM_PowerShell"
        desc = "PowerShell을 활용하는 WinRM 이벤트 탐지"
        author = "PLAINBIT"
        export = "Winrm: [connection], [resourceUri] | "
        module = "EventLog"
        score = 2
    condition:
        PowerShell_using_WinRM and Create_WinRM
}
```

[그림 49] Command가 없는 비정상 파워셸 명령 탐지 룰

4) [EventLog] Microsoft-Windows-WinRM.evtx

Microsoft-Windows-WinRM.evtx를 통해 총 2개의 이벤트를 탐지할 수 있도록 룰을 작성했다.

[표 109] 이벤트 로그(Microsoft-Windows-WinRm.evtx)에서 해킹진단도구 탐지 룰을 통해 탐지할 수 있는 명령어 목록

번호	명령 카테고리	명령어	탐지 룰 이름
1	네트워크 탐색 및 이동	jump	PowerShell을 활용하는 WinRM 이벤트 탐지
2		remote-exec	PowerShell을 활용하는 WinRM 이벤트 탐지

Microsoft-Windows-WinRM.evtx 이벤트 로그 대상으로 작성된 탐지 룰 설명은 다음과 같다.

- **PowerShell 을 활용하는 WinRM 이벤트 탐지**

Cobalt Strike Beacon이 jump, remote-exec 명령을 수행할 때 WinRM을 활용해 PowerShell 명령을 수행하는 이벤트가 발생한다. 따라서, 이벤트 ID가 6인 이벤트 로그에서 'wsman?PSVersion' 문자열이 존재하면서, 이벤트 ID가 91인 이벤트가 발생했는지 진단하도록 작성했다.

```
/* version : 2024.10.15 */
import "event"

/* CobaltStrike WinRM PowerShell */
/* 600  PowerShell 실행  HostApplication=powershell -nop -w hidden -encodedcommand %Encoding Code% */
/* 6  WSMan 세션 생성  connection=%HOSTNAME%/wsman?PSVersion=5.1.17763.2931 */
/* 91 WSMan 세션 생성  resourceUri=http://schemas.microsoft.com/powershell/Microsoft.PowerShell (%Account running Beacon% clientIP: %Parent Host IP%) */



private rule PowerShell_using_WinRM{
    condition:
        (event.Count("Microsoft-Windows-WinRM%4Operational.evtx","EventID:6","connection:/.*\\/wsman\\?PSVersion=\\.*/") > 0)
}

private rule Create_WinRM{
    condition:
        (event.Count("Microsoft-Windows-WinRM%4Operational.evtx","EventID:91","resourceUri:/ServerManagerWorkflows \\(.* clientIP: .*/") > 0)
}

rule WinRM_6_91_Suspicious_WinRM_PowerShell{
    meta:
        title = "[EVT] WinRM_6_91_Suspicious_WinRM_PowerShell"
        desc = "PowerShell을 활용하는 WinRM 이벤트 탐지"
        author = "PLAINBIT"
        export = "Winrm: [connection], [resourceUri] | "
        module = "EventLog"
        score = 2
    condition:
        PowerShell_using_WinRM and Create_WinRM
}
```

[그림 50] PowerShell을 활용하는 WinRM 이벤트 탐지 룰

5) [EventLog] Microsoft-Windows-Sysmon/Operation.evtx

Microsoft-Windows-Sysmon/Operation.evtx를 통해 총 33개의 이벤트를 탐지할 수 있도록 룰을 작성했다.

[표 110] 이벤트 로그(Microsoft-Windows-Sysmon/Operation.evtx)에서 해킹진단도구 탐지 룰을 통해 탐지할 수 있는 명령어 목록

번호	명령 카테고리	명령어	탐지 룰 이름
1	-	Beacon	- Cobalt Strike 프로세스 통신 Pipe 생성 및 연결 탐지 - Cobalt Strike Beacon(HTML Application) 실행 탐지
2	Beacon 제어 및 관리	link	- Cobalt Strike 프로세스 통신 Pipe 생성 및 연결 탐지
3	프로세스 제어	psinject	- Cobalt Strike 프로세스 통신 Pipe 생성 및 연결 탐지
4		shinject	- 프로세스 접근 이벤트 탐지
5		shspawn	- 프로세스 접근 이벤트 탐지
6	네트워크 탐색 및 이동	jump	- Cobalt Strike 프로세스 통신 Pipe 생성 및 연결 탐지 - Cobalt Strike jump psexec_psh 명령어 실행 탐지 - 프로세스 접근 이벤트 탐지 - ADMIN$ 경로에 실행 파일 생성 이벤트 탐지 - ADMIN$ 경로에 실행 파일 삭제 이벤트 탐지 - ADMIN$ 경로에 실행 파일이 서비스로 등록되는 이벤트 탐지
7		net	- Cobalt Strike 프로세스 통신 Pipe 생성 및 연결 탐지 - 프로세스 접근 이벤트 탐지 - 인자가 존재하지 않는 비정상 Rundll32.exe 실행 탐지
8		portscan	- Cobalt Strike 프로세스 통신 Pipe 생성 및 연결 탐지 - 프로세스 접근 이벤트 탐지 - 인자가 존재하지 않는 비정상 Rundll32.exe 실행 탐지
9		remote-exec	- Cobalt Strike 프로세스 통신 Pipe 생성 및 연결 탐지
10	권한 상승 및 크리덴셜 수집	chromedump	- Cobalt Strike 프로세스 통신 Pipe 생성 및 연결 탐지 - 프로세스 접근 이벤트 탐지 - 인자가 존재하지 않는 비정상 Rundll32.exe 실행 탐지
11		dcsync	- Cobalt Strike 프로세스 통신 Pipe 생성 및 연결 탐지 - 프로세스 접근 이벤트 탐지 - 인자가 존재하지 않는 비정상 Rundll32.exe 실행 탐지
12		elevate	- ADMIN$ 경로에 실행 파일 생성 이벤트 탐지 - ADMIN$ 경로에 실행 파일 삭제 이벤트 탐지 - ADMIN$ 경로에 실행 파일이 서비스로 등록되는 이벤트 탐지

번호	명령 카테고리	명령어	탐지 룰 이름
13	권한 상승 및 크리덴셜 수집	hashdump	- Cobalt Strike 프로세스 통신 Pipe 생성 및 연결 탐지 - 프로세스 접근 이벤트 탐지 - 인자가 존재하지 않는 비정상 Rundll32.exe 실행 탐지 - 비정상 lsass.exe 접근 이벤트 탐지
14		logonpasswords	- Cobalt Strike 프로세스 통신 Pipe 생성 및 연결 탐지 - 프로세스 접근 이벤트 탐지 - 인자가 존재하지 않는 비정상 Rundll32.exe 실행 탐지
15		mimikatz	- Cobalt Strike 프로세스 통신 Pipe 생성 및 연결 탐지 - 프로세스 접근 이벤트 탐지 - 인자가 존재하지 않는 비정상 Rundll32.exe 실행 탐지
16		pth	- Cobalt Strike 프로세스 통신 Pipe 생성 및 연결 탐지 - Cobalt Strike pth 명령어 실행 탐지 - 인자가 존재하지 않는 비정상 Rundll32.exe 실행 탐지
17	사용자 활동 모니터링	desktop	- Cobalt Strike desktop 명령어 실행 탐지 - 프로세스 접근 이벤트 탐지 - 인자가 존재하지 않는 비정상 Rundll32.exe 실행 탐지
18		keylogger	- Cobalt Strike 프로세스 통신 Pipe 생성 및 연결 탐지 - 프로세스 접근 이벤트 탐지 - 인자가 존재하지 않는 비정상 Rundll32.exe 실행 탐지
19		printscreen	- Cobalt Strike 프로세스 통신 Pipe 생성 및 연결 탐지 - 프로세스 접근 이벤트 탐지 - 인자가 존재하지 않는 비정상 Rundll32.exe 실행 탐지
20		screenshot	- Cobalt Strike 프로세스 통신 Pipe 생성 및 연결 탐지 - 프로세스 접근 이벤트 탐지 - 인자가 존재하지 않는 비정상 Rundll32.exe 실행 탐지
21		screenwatch	- Cobalt Strike 프로세스 통신 Pipe 생성 및 연결 탐지 - 프로세스 접근 이벤트 탐지 - 인자가 존재하지 않는 비정상 Rundll32.exe 실행 탐지
22	통신 및 데이터 암호화	covertvpn	- 프로세스 접근 이벤트 탐지 - 인자가 존재하지 않는 비정상 Rundll32.exe 실행 탐지 - NPcap 파일 생성 - NPcap 이미지 로드
23		spawn	- 프로세스 접근 이벤트 탐지
24		spawnas	- 인자가 존재하지 않는 비정상 Rundll32.exe 실행 탐지
25		spawnu	- 프로세스 접근 이벤트 탐지 - 인자가 존재하지 않는 비정상 Rundll32.exe 실행 탐지
26		spunnel	- 프로세스 접근 이벤트 탐지 - 인자가 존재하지 않는 비정상 Rundll32.exe 실행 탐지

번호	명령 카테고리	명령어	탐지 룰 이름
27	통신 및 데이터 암호화	spunnel_local	- 프로세스 접근 이벤트 탐지 - 인자가 존재하지 않는 비정상 Rundll32.exe 실행 탐지
28	명령 및 스크립트 실행	execute	- 프로세스 접근 이벤트 탐지
29		execute-assembly	- 프로세스 접근 이벤트 탐지 - 인자가 존재하지 않는 비정상 Rundll32.exe 실행 탐지
30		powerpick	- Cobalt Strike 프로세스 통신 Pipe 생성 및 연결 탐지 - Cobalt Strike powerpick 명령어 실행 탐지 - 프로세스 접근 이벤트 탐지 - 인자가 존재하지 않는 비정상 Rundll32.exe 실행 탐지
31		powershell	- 프로세스 접근 이벤트 탐지 - 인자가 존재하지 않는 비정상 Rundll32.exe 실행 탐지
32		runu	- 프로세스 접근 이벤트 탐지
33		shell	- 프로세스 접근 이벤트 탐지

Microsoft-Windows-Sysmon/Operation.evtx이벤트 로그 대상으로 작성된 탐지 룰 설명은 다음과 같다.

- **Cobalt Strike 프로세스 통신 Pipe 생성 및 연결 탐지**

Cobalt Strike Beacon이 실행되거나 특정 명령을 수행할 때 명명된 Pipe를 생성 및 연결한다. 따라서, 이벤트 ID 17, 18인 이벤트 로그에서 Cobalt Strike에서 사용되는 명명된 Pipe 패턴을 가지는 Pipe가 생성 및 연결하는 이벤트가 발생했는지 진단하도록 작성했다.

```
/* version : 2024.10.15 */
import "event"

/* Default Cobalt Strike Artifact Kit binaries */
private rule Detect_NamedPipe_CobaltStrike_Connection{
    condition:
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:/17|18/","PipeName:/MSSE\\-[0-9a-f]{4}\\-server/") > 0)
}

/* Default psexec_psh */
private rule Detect_NamedPipe_status{
    condition:
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:/17|18/","PipeName:/status\\_[0-9a-f]{2}/") > 0)
}

/* Default SSH beacon */
private rule Detect_NamedPipe_msagent{
    condition:
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:/17|18/","PipeName:/msagent\\_[0-9a-f]{2}/") > 0)
}

/* Default SMB beacon */
private rule Detect_NamedPipe_postex_ssh{
    condition:
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:/17|18/","PipeName:/postex\\_ssh\\_[0-9a-f]{4}/") > 0)
}

/* Default Post Exploitation job (v4.2+) */
private rule Detect_NamedPipe_postex{
    condition:
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:/17|18/","PipeName:/postex\\_[0-9a-f]{4}/") > 0)
}
```

[그림 51] Cobalt Strike 프로세스 통신 Pipe 생성 및 연결 탐지 룰 일부

- **Cobalt Strike Beacon(HTML Application) 실행 탐지**

Cobalt Strike Beacon이 HTML Application 실행 형태로 동작하는 경우, mshta.exe 프로세스가 부모 프로세스로 생성된다. 따라서, Microsoft-Windows-Sysmon/Operation.evtx의 이벤트 ID 1인 이벤트 로그에서 부모 프로세스가 mshta.exe이면서 자식 프로세스가 powershell.exe, rundll32.exe, wscript.exe인 프로세스를 생성하는 이벤트가 발생했는지 진단하도록 작성했다.

```
/* version : 2024.10.15 */
import "event"

/* CobaltStrike HTML Application Beacon*/
private rule Detect_mshta_parentprocess{
    condition:
        (event.Count("Microsoft-Windows-WinRM%4Operational.evtx","EventID:1","Image:/powershell\\.exe/","CommandLine:/encodedcommand|EncodedCommand|\\-enc|\\-Enc/i","ParentImage:/mshta\\.exe/") > 0) or
        (event.Count("Microsoft-Windows-WinRM%4Operational.evtx","EventID:1","Image:/rundll32\\.exe/","ParentImage:/mshta\\.exe/") > 0) or
        (event.Count("Microsoft-Windows-WinRM%4Operational.evtx","EventID:1","Image:/wscript\\.exe/","ParentImage:/mshta\\.exe/") > 0)
}

rule Sysmon_1_CobaltStrike_Beacon_HTMLApplication{
    meta:
        title = "[EVT] Sysmon_1_CobaltStrike_Beacon_HTMLApplication"
        desc = "HTML Application형태의 CobaltStrike Beacon 실행 이력 탐지"
        author = "PLAINBIT"
        export = "프로세스: [Image], Command: [CommandLine], 부모 프로세스: [ParentImage]"
        module = "EventLog"
        score = 4
    condition:
        Detect_mshta_parentprocess
}
```

[그림 52] Cobalt Strike Beacon(HTML Application) 실행 탐지 룰 일부

- **Cobalt Strike browserpivot 명령어 실행 탐지**

Cobalt Strike Beacon이 browserpivot 명령을 수행할 때 IE Explorer 프로세스를 대상으로 원격 스레드를 생성한다. 따라서, Microsoft-Windows-Sysmon/Operation.evtx의 이벤트 ID 8인 이벤트 로그에서 iexplore.exe 프로세스를 대상으로 원격 스레드를 생성하는 이벤트가 발생했는지 진단하도록 작성했다.

```
/* version : 2024.10.15 */
import "event"

/* CobaltStrike HTML Application Beacon*/
private rule Detect_CreateRemoteThread_iexpolrer{
    condition:
    /*\\[Ii]nternet [Ee]xplorer\\iexplore\\.exe/*/
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:8","TargetImage:/[Pp]rogram [Ff]iles( \\(x86\\))?\\\\[Ii]nternet [Ee]xplorer\\\\iexplore\\.exe/") > 0)
}

rule Sysmon_8_CobaltStrike_Command_browserpivot{
    meta:
        title = "[EVT] Sysmon_8_CobaltStrike_Command_browserpivot"
        desc = "CobaltStrike browserpivot 명령어 실행 탐지"
        author = "PLAINBIT"
        export = "프로세스: [SourceImage], 대상 프로세스: [TargetImage], Address: [StartAddress]"
        module = "EventLog"
        score = 4
    condition:
        Detect_CreateRemoteThread_iexpolrer
}
```

[그림 53] Cobalt Strike browserpivot 명령어 실행 탐지 룰

- **Cobalt Strike desktop 명령어 실행 탐지**

Cobalt Strike Beacon이 desktop 명령을 수행할 때 기본 값으로 rundll32.exe 프로세스가 루프백 IP(127.0.0.1)에서 루프백 IP (127.0.0.1)로 네트워크 연결을 한다. 따라서, Microsoft-Windows-Sysmon/Operation.evtx의 이벤트 ID 3인 이벤트 로그에서 rundll32.exe 프로세스가 루프백 IP(127.0.0.1)에서 루프백 IP(127.0.0.1)로 네트워크 연결을 수행한 이벤트가 발생했는지 진단하도록 작성했다.

```
/* version : 2024.10.15 */
import "event"


/* CobaltStrike Command Execute Desktop */
private rule Detect_NetowrkConnection_rundll32_loopback{
    condition:
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:3","Image:/rundll32\\.exe/","SourceIp:/127\\.0\\.0\\.1/","DestinationIp:/127\\.0\\.0\\.1/") > 0)
}

rule Sysmon_3_CobaltStrike_Command_desktop{
    meta:
        title = "[EVT] Sysmon_3_CobaltStrike_Command_desktop"
        desc = "CobaltStrike Desktop 명령어 실행 탐지"
        author = "PLAINBIT"
        export = "프로세스: [Image], SourceIP: [SourceIp], DestIP: [DestinationIp], DestPort: [DestinationPort]"
        module = "EventLog"
        score = 4
    condition:
        Detect_NetowrkConnection_rundll32_loopback
}
```

[그림 54] Cobalt Strike desktop 명령 실행 탐지 룰

- **Cobalt Strike jump psexec_psh 명령어 실행 탐지**

Cobalt Strike Beacon이 jump psexec_psh 명령을 수행할 때 Encoding된 명령어를 수행하는 PowerShell 프로세스가 '-Version', '-s', '-NoLogo', '-NoProfile' 명령 구문을 가진 PowerShell 프로세스를 생성한다. 따라서, Microsoft-Windows-Sysmon/Operation.evtx의 이벤트 ID 1인 이벤트 로그에서 Encoding된 명령어를 수행하는 powershell.exe 프로세스가 특정 명령 구문을 수행하는 powershell.exe 프로세스를 생성하는 이벤트가 발생했는지 진단하도록 작성했다.

```
/* version : 2024.10.15 */
import "event"

/* CobaltStrike Command Execute jump psexec_psh */
private rule Detect_Suspicious_PowerShell_ProcessCreate{
    condition:
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:1","Image:/powershell\\.exe/","CommandLine:/\\-Version [0-9]\\.[0-9] \\-s \\-NoLogo \\-NoProfile/","ParentImage:/pow
}

rule Sysmon_1_CobaltStrike_Command_jump{
    meta:
        title = "[EVT] Sysmon_1_CobaltStrike_Command_jump"
        desc = "CobaltStrike jump psexec_psh 명령어 실행 탐지"
        author = "PLAINBIT"
        export = "프로세스: [Image], ParentProcess: [ParentImage], ParentCommand: [ParentCommandLine]"
        module = "EventLog"
        score = 4
    condition:
        Detect_Suspicious_PowerShell_ProcessCreate
}
```

[그림 55] Cobalt Strike jump psexec_psh 명령어 실행 탐지 룰 일부

- **Cobalt Strike powerpick 명령어 실행 탐지**

Cobalt Strike Beacon이 powerpick 명령을 수행할 때 .NET 환경과 관련된 DLL 파일들을 rundll32.exe 프로세스에서 로드한다. 따라서, 이벤트 ID 7인 이벤트 로그에서 rundll32.exe 프로세스가 .NET 관련 DLL(mscoreei.dll, mscorwks.dll, mscorjit.dll, Culture.dll)을 로드하는 이벤트가 발생했는지 진단하도록 작성했다.

```
/* version : 2024.10.15 */
import "event"

/* CobaltStrike Command Execute Powerpick */
private rule Sysmon_NetFramework_ImageLoaded_mscoreei{
    condition:
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:7","Image:/rundll32\\.exe/","ImageLoaded:/mscoreei\\.dll/") > 0)
}

private rule Sysmon_NetFramework_ImageLoaded_mscorwks{
    condition:
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:7","Image:/rundll32\\.exe/","ImageLoaded:/mscorwks\\.dll/") > 0)
}

private rule Sysmon_NetFramework_ImageLoaded_mscorjit{
    condition:
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:7","Image:/rundll32\\.exe/","ImageLoaded:/mscorjit\\.dll/") > 0)
}

private rule Sysmon_NetFramework_ImageLoaded_Culture{
    condition:
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:7","Image:/rundll32\\.exe/","ImageLoaded:/Culture\\.dll/") > 0)
}

rule Sysmon_7_CobaltStrike_Command_powerpick{
    meta:
        title = "[EVT] Sysmon_7_CobaltStrike_Command_powerpick"
        desc = "CobaltStrike Powerpick 명령어 실행 탐지"
        author = "PLAINBIT"
        export = "프로세스: [Image], 모듈: [ImageLoaded]"
        module = "EventLog"
        score = 4
    condition:
        Sysmon_NetFramework_ImageLoaded_mscoreei or
        Sysmon_NetFramework_ImageLoaded_mscorwks or
        Sysmon_NetFramework_ImageLoaded_mscorjit or
        Sysmon_NetFramework_ImageLoaded_Culture
}
```

[그림 56] Cobalt Strike powerpick 명령어 실행 탐지 룰

- **Cobalt Strike pth 명령어 실행 탐지**

Cobalt Strike Beacon이 pth 명령을 수행할 때 Pipe에 문자열을 전달하는 명령어를 가진 CMD 프로세스가 생성된다. 따라서, Microsoft-Windows-Sysmon/Operation.evtx의 이벤트 ID 1인 이벤트 로그에서 Pipe에 문자열을 전달하는 명령어를 실행하는 cmd.exe 프로세스를 생성하는 이벤트가 발생했는지 진단하도록 작성했다.

```
/* version : 2024.10.15 */
import "event"

/* CobaltStrike Command Execute Pth(Pass-the-hash) */
private rule Detect_Suspicious_ProcessCreate_CMD{
    condition:
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:1","Image:/cmd\\.exe/","CommandLine:/\\/c echo [0-9a-f]{11}\\s+>\\s+\\\\.*pipe/") > 0)
}

rule Sysmon_1_CobaltStrike_Command_pth{
    meta:
        title = "[EVT] Sysmon_1_CobaltStrike_Command_pth"
        desc = "CobaltStrike Pth(Pass-the-hash) 명령어 실행 탐지"
        author = "PLAINBIT"
        export = "프로세스: [Image], Command: [CommandLine]"
        module = "EventLog"
        score = 4
    condition:
        Detect_Suspicious_ProcessCreate_CMD
}
```

[그림 57] Cobalt Strike pth 명령어 실행 탐지 룰

- **프로세스 접근 이벤트 탐지**

Cobalt Strike Beacon이 특정 명령을 수행할 때 프로세스를 생성하고 접근하는 이벤트가 기록된다. 따라서, Microsoft-Windows-Sysmon/Operation.evtx의 이벤트 ID 10인 이벤트 로그에서 GrantedAccess(프로세스 권한에 관한 Access Flags)의 값과 CallTrace(열린 프로세스를 호출하는 위치)의 값이 특정 값으로 접근하는 이벤트가 발생했는지 진단하도록 작성했다. 해당 룰의 경우, 정상 프로세스에서도 해당 값으로 접근할 수 있기 때문에 다른 Rule에서 탐지한 이벤트들과 같이 분석해야 한다.

```
/* version : 2024.10.15 */
import "event"

/* CobaltStrike Process Injection(Process Access Pattern) */
private rule Detect_Suspicious_ProcessAccess{
    condition:
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:10","GrantedAccess:0x1FFFFF","CallTrace:/c\\:\\\\windows\\\\system32\\\\ntdll\\.dll+.*\\|c\\:\\\\windows\\\\system32\
}

rule Sysmon_10_CobaltStrike_ProcessAccess{
    meta:
        title = "[EVT] Sysmon_10_CobaltStrike_ProcessAccess"
        desc = "CobaltStrike Beacon이 타 프로세스 메모리에 접근하는 행위 탐지"
        author = "PLAINBIT"
        export = "프로세스: [SourceImage], 대상 프로세스: [TargetImage], 권한: [GrantedAccess], CallTrace: [CallTrace]"
        module = "EventLog"
        score = 2
    condition:
        Detect_Suspicious_ProcessAccess
}
```

[그림 58] 프로세스 접근 이벤트 탐지 룰 일부

- **비정상 lsass.exe 접근 이벤트 탐지**

Cobalt Strike Beacon이 hashdump 명령을 수행할 때 크리덴셜을 수집하기 위해 LSASS 프로세스에 원격 스레드를 생성한다. 따라서, Microsoft-Windows-Sysmon/Operation.evtx의 이벤트 ID 8인 이벤트 로그에서 rundll32.exe 프로세스가 lsass.exe 프로세스에 원격 스레드를 생성하는 이벤트가 발생했는지 진단하도록 작성했다.

```
/* version : 2024.10.15 */
import "event"

/* Detect CreateRetmoeThread rundll32 to lsass */
private rule Detect_Suspicious_CreateRemoteThread_lsass{
    condition:
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:8","SourceImage:/rundll32\\.exe$/","TargetImage:/lsass\\.exe$/") > 0)
}

rule Sysmon_8_Supicious_CreateRemoteThread_lsass{
    meta:
        title = "[EVT] Sysmon_8_Supicious_CreateRemoteThread_lsass"
        desc = "Cobalt Strike Beacon이 hashdump 명령을 수행할 때 크리덴셜을 수집하기 위해 LSASS 프로세스에 원격 스레드를 생성하는 것을 탐지"
        author = "PLAINBIT"
        export = "프로세스: [SourceImage], 대상 프로세스: [TargetImage], Address: [StartAddress]"
        module = "EventLog"
        score = 3
    condition:
        Detect_Suspicious_CreateRemoteThread_lsass
}
```

[그림 59] 비정상 lsass.exe 접근 이벤트 탐지 룰

- **ADMIN$ 경로에 실행 파일 생성 이벤트 탐지**

Cobalt Strike Beacon이 jump, elevate 명령을 수행할 때 System 프로세스가 ADMIN$ 경로에 랜덤한 파일 명을 가진 실행 파일을 생성한다. 따라서, Microsoft-Windows-Sysmon/Operation.evtx의 이벤트 ID 11인 이벤트 로그에서 System 프로세스가 ADMIN$ 경로에 랜덤한 파일 명을 가진 실행 파일을 생성한 이벤트가 발생했는지 진단하도록 작성했다.

```
/* version : 2024.10.15 */
import "event"

/* ADMIN$(C:\Windows) File Create */
private rule Detect_Suspicious_cmd_ProcessCreate{
    condition:
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:11","Image:/System/","TargetFilename:/c\\:\\\\windows\\\\[0-9a-zA-Z_\\-~!#$%^&(){}@'`.+,=]+\\.exe/") > 0)
}

rule Sysmon_11_CreateFile_ADMINShareFolder{
    meta:
        title = "[EVT] Sysmon_11_CreateFile_ADMINShareFolder"
        desc = "ADMIN(C:\\Windows) 내 실행파일(.exe) 생성 탐지"
        author = "PLAINBIT"
        export = "프로세스: [Image], 파일: [TargetFilename]"
        module = "EventLog"
        score = 3
    condition:
        Detect_Suspicious_cmd_ProcessCreate
}
```

[그림 60] ADMIN$ 경로에 실행 파일 생성 이벤트 탐지 룰

- **ADMIN$ 경로에 실행 파일 삭제 이벤트 탐지**

Cobalt Strike Beacon이 jump, elevate 명령을 수행한 뒤 System 프로세스가 ADMIN$ 경로에 생성했던 랜덤한 파일 명을 가진 실행 파일을 삭제한다. 따라서, Microsoft-Windows-Sysmon/Operation.evtx의 이벤트 ID 23인 이벤트 로그에서 System 프로세스가 ADMIN$ 경로에 랜덤한 파일 명을 가진 실행 파일을 삭제한 이벤트가 발생했는지 진단하도록 작성했다.

```
/* version : 2024.10.15 */
import "event"

/* ADMIN$(C:\Windows) File Delete */
private rule Detect_supicious_file_delete_ADMIN{
    condition:
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:23","Image:/System/","TargetFilename:/c\\:\\\\windows\\\\[0-9a-zA-Z_\\-~!#$%^&(){}@'`.+,=]+\\.exe/") > 0)
}

rule Sysmon_23_DeleteFile_ADMINShareFolder{
    meta:
        title = "[EVT] Sysmon_23_DeleteFile_ADMINShareFolder"
        desc = "ADMIN(C:\\Windows) 내 실행파일(.exe) 삭제 탐지"
        author = "PLAINBIT"
        export = "프로세스: [Image], 파일: [TargetFilename]"
        module = "EventLog"
        score = 3
    condition:
        Detect_supicious_file_delete_ADMIN
}
```

[그림 61] ADMIN$ 경로에 실행 파일 삭제 이벤트 탐지 룰

- **ADMIN$ 경로에 실행 파일이 서비스로 등록되는 이벤트 탐지**

Cobalt Strike Beacon이 jump, elevate 명령을 수행할 때 ADMIN$ 경로의 랜덤한 파일 명을 가진 실행 파일을 서비스로 등록한다. 따라서, Microsoft-Windows-Sysmon/Operation.evtx의 이벤트 ID 13인 이벤트 로그에서 'HKLM\System\CurrentControlSet\Services' 경로에 등록된 서비스 레지스트리 항목에 ADMIN$ 경로의 실행 파일이 등록되는 이벤트가 발생했는지 진단하도록 작성했다.

```
/* version : 2024.10.15 */
import "event"

/* Detect creation of registry service with ADMIN$ with sysmon */
private rule Detect_reg_supicious_service{
    condition:
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:13","TargetObject:/HKLM\\\\System\\\\CurrentControlSet\\\\Services\\\\.*\\\\ImagePath/","Details:/\\\\.*\\ADMIN\\$\\\\
}

rule Sysmon_13_ServiceInstall_ADMINShareFolder{
    meta:
        title = "[EVT] Sysmon_13_ServiceInstall_ADMINShareFolder"
        desc = "ADMIN$(C:\\Windows) 경로 내 실행 파일이 서비스로 실행되도록 등록되는 이벤트를 탐지"
        author = "PLAINBIT"
        export = "KeyPath: [TargetObject], Value: [Details]"
        module = "EventLog"
        score = 3
    condition:
        Detect_reg_supicious_service
}
```

[그림 62] ADMIN$ 경로에 실행 파일이 서비스로 등록되는 이벤트 탐지 룰 일부

- **인자가 존재하지 않는 비정상 rundll32.exe 실행 탐지**

Cobalt Strike Beacon이 실행되거나 특정 명령을 수행할 때 Beacon 프로세스가 인자 값 없는 rundll32.exe 프로세스를 생성한다. 따라서, Microsoft-Windows-Sysmon/Operation.evtx의 이벤트 ID 1인 이벤트 로그에서 인자 값이 존재하지 않는 rundll32.exe 프로세스를 생성하는 이벤트가 발생했는지 진단하도록 작성했다.

```
/* version : 2024.10.15 */
import "event"

/* Detect Execute Supicious Rundll32 */
private rule Detect_Suspicious_ProcessCreate_rundll32{
    condition:
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:1","Image:/rundll32\\.exe/","CommandLine:/rundll32\\.exe$/","OriginalFileName:/RUNDLL32\\.EXE/") > 0)
}

rule Sysmon_1_Suspicious_ProcessCreate_Rundll32{
    meta:
        title = "[EVT] Sysmon_1_Suspicious_ProcessCreate_Rundll32"
        desc = "Cobalt Strike Beacon이 실행되거나 특정 명령어를 수행할 때 Beacon 프로세스가 인자 값 없는 rundll32.exe 프로세스를 생성하는 것을 탐지"
        author = "PLAINBIT"
        export = "프로세스: [Image], Command: [CommandLine], ParentProcess: [ParentImage], ParentCommand: [ParentCommandLine]"
        module = "EventLog"
        score = 3
    condition:
        Detect_Suspicious_ProcessCreate_rundll32
}
```

[그림 63] 인자가 존재하지 않는 비정상 rundll32.exe 실행 탐지 룰

- **NPcap 파일 생성**

Cobalt Strike Beacon이 covertvpn 명령을 수행할 때 rundll32.exe 프로세스가 %Temp% 하위 경로에 NPcap 파일들을 생성한다. 따라서, Microsoft-Windows-Sysmon/Operation.evtx의 이벤트 ID 11인 이벤트 로그에서 rundll32.exe 프로세스가 %Temp% 하위 경로에 NPcap 파일(wpcap.dll, Packet.dll, npf.sys)을 생성하는 이벤트가 발생했는지 진단하도록 작성했다.

```
/* version : 2024.10.15 */
import "event"

/* CobaltStrike ConvertVPN by 11 */
private rule Sysmon_NPcap_install_wpcap{
    condition:
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:11","Image:/rundll32\\.exe/","TargetFilename:/\\AppData\\\\Local\\\\Temp\\\\.*\\wpcap\\.dll$/") > 0) or
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:11","Image:/rundll32\\.exe/","TargetFilename:/\\\\Windows\\\\Temp\\\\wpcap\\.dll/") > 0)
}

private rule Sysmon_NPcap_install_Packet{
    condition:
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:11","Image:/rundll32\\.exe/","TargetFilename:/\\AppData\\\\Local\\\\Temp\\\\.*\\Packet\\.dll$/") > 0) or
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:11","Image:/rundll32\\.exe/","TargetFilename:/\\\\Windows\\\\Temp\\\\Packet\\.dll/") > 0)
}

private rule Sysmon_NPcap_install_npf{
    condition:
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:11","Image:/rundll32\\.exe/","TargetFilename:/\\AppData\\\\Local\\\\Temp\\\\.*\\npf\\.sys$/") > 0) or
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:11","Image:/rundll32\\.exe/","TargetFilename:/\\\\Windows\\\\Temp\\\\npf\\.sys/") > 0)
}

rule Sysmon_11_Suspicious_FileCreate_NPcap{
    meta:
        title = "[EVT] Sysmon_11_Suspicious_FileCreate_NPcap"
        desc = "비정상 프로세스에서 NPcap 파일 생성"
        author = "PLAINBIT"
        export = "프로세스: [Image], 파일: [TargetFilename]"
        module = "EventLog"
        score = 2
    condition:
        Sysmon_NPcap_install_wpcap or
        Sysmon_NPcap_install_Packet or
        Sysmon_NPcap_install_npf
}
```

[그림 64] NPcap 파일 생성 이벤트 탐지 룰

- **NPcap 이미지 로드**

Cobalt Strike Beacon이 covertvpn 명령을 수행할 때 rundll32.exe 프로세스가 %Temp% 하위 경로의 NPcap DLL 파일들을 로드한다. 따라서, Microsoft-Windows-Sysmon/Operation.evtx의 이벤트 ID 7인 이벤트 로그에서 rundll32.exe 프로세스가 %Temp% 하위 경로의 NPcap DLL 파일(wpcap.dll, Packet.dll)을 로드하는 이벤트가 발생했는지 진단하도록 작성했다.

```
/* version : 2024.10.15 */
import "event"

/* CobaltStrike ConvertVPN by 7 */
private rule Sysmon_NPcap_ImageLoaded_wpcap{
    condition:
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:7","Image:/rundll32\\.exe/","ImageLoaded:/\\AppData\\\\Local\\\\Temp\\\\.*\\wpcap\\.dll$/") > 0) or
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:7","Image:/rundll32\\.exe/","ImageLoaded:/\\\\Windows\\\\Temp\\\\wpcap\\.dll$/") > 0)
}

private rule Sysmon_NPcap_ImageLoaded_Packet{
    condition:
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:7","Image:/rundll32\\.exe/","ImageLoaded:/\\AppData\\\\Local\\\\Temp\\\\.*\\Packet\\.dll$/") > 0) or
        (event.Count("Microsoft-Windows-Sysmon%4Operational.evtx","EventID:7","Image:/rundll32\\.exe/","ImageLoaded:/\\\\Windows\\\\Temp\\\\Packet\\.dll$/") > 0)
}

rule Sysmon_7_Suspicious_ImageLoaded_NPcap{
    meta:
        title = "[EVT] Sysmon_7_Suspicious_ImageLoaded_NPcap"
        desc = "비정상 프로세스에서 NPcap 이미지 로드"
        author = "PLAINBIT"
        export = "프로세스: [Image], 모듈: [ImageLoaded]"
        module = "EventLog"
        score = 2
    condition:
        Sysmon_NPcap_ImageLoaded_wpcap and Sysmon_NPcap_ImageLoaded_Packet
}
```

[그림 65] NPcap 이미지 로드 이벤트 탐지 룰

6) [Registry] SYSTEM

SYSEM 레지스트리 하이브 키를 통해 총 2개의 이벤트를 탐지할 수 있도록 룰을 작성했다.

[표 111] 레지스트리(SYSTEM)에서 해킹진단도구 탐지 룰을 통해 탐지할 수 있는 명령어 목록

번호	명령 카테고리	명령어	탐지 룰 이름
1	네트워크 탐색 및 이동	jump	- 난독화된 PowerShell 명령어 레지스트리 서비스 등록 탐지 - ADMIN$ 경로의 실행 파일이 레지스트리 서비스 등록 탐지
2	권한 및 크리덴셜 수집	elevate	- ADMIN$ 경로의 실행 파일이 레지스트리 서비스 등록 탐지

Microsoft-Windows-WinRM.evtx 이벤트 로그 대상으로 작성된 탐지 룰 설명은 다음과 같다.

- **난독화된 파워셸 명령이 레지스트리 서비스에 등록되는 이벤트 탐지**

Cobalt Strike Beacon이 jump 명령을 수행할 때 Encoding된 PowerShell 명령어를 실행하도록 서비스 레지스트리 항목에 등록한다. 따라서, SYSTEM HIVE의 'CurrentControlSet\Services' 경로에 등록된 서비스 레지스트리 항목에 Encoding된 PowerShell 명령어가 등록되어 있는지 진단하도록 작성했다. 단, 명령이 정상적으로 수행되면 등록한 서비스 레지스트리 항목을 삭제하기 때문에 정상적으로 수행되지 못한 명령만 진단할 수 있다.

```
/* version : 2024.10.15 */
import "registry"

rule SYSTEM_Serviceinstall_EncodedCommand_PowerShell{
   meta:
      title = "[REG] SYSTEM_Serviceinstall_EncodedCommand_PowerShell"
      desc = "난독화된 PowerShell 명령어가 레지스트리 내 서비스로 등록되어 있는지 확인"
      author = "PLAINBIT"
      export = "경로 : [PATH], 값 : [VALUE], 데이터 : [DATA]"
      module = "registry"
      score = 3
   condition:
      (registry.searchValue("HKEY_LOCAL_MACHINE\\SYSTEM\\CurrentControlSet\\Services\\/.*/","ImagePath","/encodedcommand|EncodedCommand|\\-enc|\\-Enc/") > 0)
}
```

[그림 66] 난독화된 파워셸 명령이 레지스트리 서비스에 등록되는 이벤트 탐지 룰

- **ADMIN$ 경로의 실행 파일이 레지스트리 서비스 등록 탐지**

Cobalt Strike Beacon이 jump, elevate 명령을 수행할 때 ADMIN$ 경로의 랜덤한 파일 명을 가진 실행 파일을 서비스 레지스트리 항목에 등록한다. 따라서, SYSTEM HIVE의 'CurrentControlSet\Services' 경로에 등록된 서비스 레지스트리 항목에 ADMIN$ 경로의 실행 파일이 등록되어 있는지 진단하도록 작성했다. 단, 명령이 정상적으로 수행되면 등록한 서비스 레지스트리 항목을 삭제하기 때문에 정상적으로 수행되지 못한 명령만 진단할 수 있다.

```
/* version : 2024.10.15 */
import "registry"

rule SYSTEM_Serviceinstall_ADMINShareFolder_Exe{
   meta:
      title = "[REG] SYSTEM_Serviceinstall_ADMINShareFolder_Exe"
      desc = "ADMIN$(C:\\Windows) 경로의 실행 파일이 레지스트리 내 서비스로 등록되어 있는지 확인"
      author = "PLAINBIT"
      export = "경로 : [PATH], 값 : [VALUE], 데이터 : [DATA]"
      module = "registry"
      score = 3
   condition:
      (registry.searchValue("HKEY_LOCAL_MACHINE\\SYSTEM\\CurrentControlSet\\Services\\/.*/","ImagePath","/\\.*\\ADMIN\\$\\\\[0-9a-zA-Z_\\-~!#$%^&(){}@'`.+,=]+\\.exe$/") > 0)
}
```

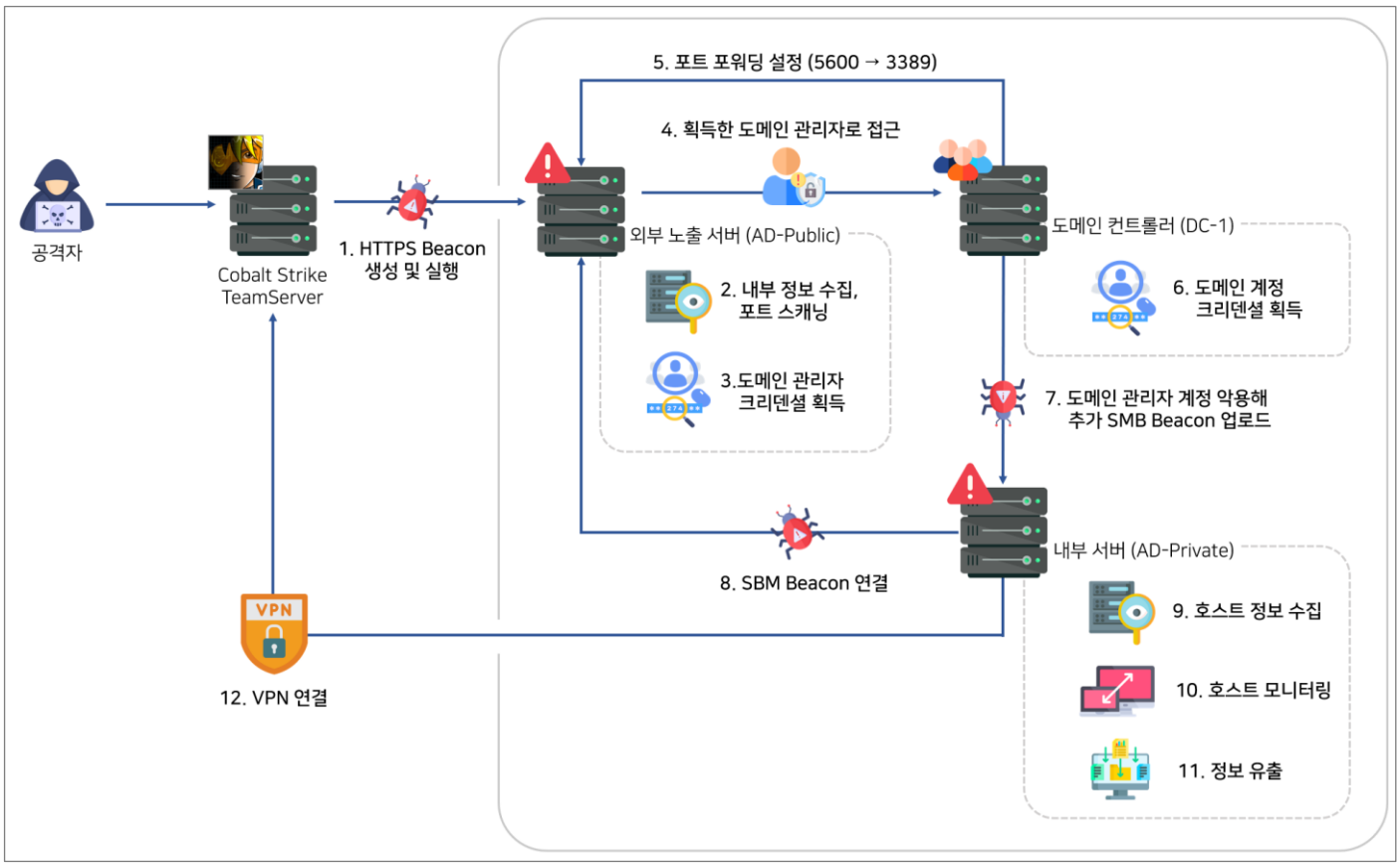
[그림 67] ADMIN$ 경로의 실행 파일이 레지스트리 서비스에 등록되는 이벤트 탐지 룰

6.4. 시나리오 기반 탐지 룰 테스트

작성한 Sysmon Configure Rule과 해킹진단도구 탐지 룰을 국내 AD환경에서 흔히 발생할 수 있는 가상의 시나리오를 제작한 후 테스트를 수행했으며, 그 결과는 다음과 같다.

6.4.1. 테스트 시나리오 개요

1) 테스트 시나리오 내용



[그림 68] 테스트 시나리오 개요도

공격자는 외부에 노출되어 있는 서버(AD-Public)에 침투한 후 Cobalt Strike Beacon을 백도어로 실행해 내부 정보 수집을 통해 AD 환경임을 확인하고 도메인 관리자 계정의 크리덴셜을 획득했다. 획득한 도메인 관리자 계정의 크리덴셜로 도메인 컨트롤러에 접근해 내부 서버로 Beacon을 추가 전파했다. 이후 내부 서버에서 호스트 정보를 수집하고, 사용자 활동을 모니터링하면서 중요 정보를 외부로 유출했다.

2) 테스트 시나리오 환경

테스트 시나리오 환경의 대상 시스템들은 모두 Windows Server 2019로 구성했으며, Cobalt Strike는 4.9.1 버전을 사용했다. 시스템에 '6.3.1. Sysmon Configure Rule'에서 설명한 Sysmon Configure Rule을 적용한 상태로 공격을 수행했다. 이후 해킹진단도구를 통해 시스템 아티팩트 수집 및 진단 룰을 통한 분석 순서로 진행했다. 시나리오에서 Cobalt Strike 명령을 활용해 수행된 주요 공격 타임라인은 다음과 같다.

[표 112] 테스트 시나리오에서 수행된 주요 공격 타임라인

일시	행위 수행 대상	이벤트 내용	활용 명령(Cobalt Strike)
2024-10-16 11:21:25	AD-Public	HTTPS Beacon 실행	-
2024-10-16 11:22:31	Team Server → AD-Public	내부 정보 수집	net, powerpick, portscan
2024-10-16 11:23:21	Team Server → AD-Public	크리덴셜 획득	mimikatz, pth
2024-10-16 11:25:01	AD-Public → DC1	내부 이동	jump psexec_psh
2024-10-16 11:25:20	DC1 → AD-Public	포트 포워딩	rportfwd
2024-10-16 11:25:36	Team Server → DC1	크리덴셜 획득	dcsync, pth
2024-10-16 11:26:15	DC1 → AD-Private	Beacon 파일 전파	upload
2024-10-16 11:27:30	AD-Private → AD-Public	SMB Beacon 연결	remote-exec psexec, link
2024-10-16 11:29:00	Team Server → AD-Private	호스트 정보 수집	powershell
2024-10-16 13:35:19	Team Server → AD-Private	사용자 활동 모니터링	desktop
2024-10-16 13:37:02	AD-Private → Team Server	정보 유출	download
2024-10-16 13:38:43	AD-Private → Team Server	VPN 설정	covertvpn

3) 시나리오 기반 탐지 룰 테스트 결과

시나리오를 기반으로 Sysmon Configure Rule과 해킹진단도구 탐지 룰을 테스트한 결과, 공격 과정에서 Cobalt Strike를 활용해 수행된 모든 행위를 탐지할 수는 없었지만 대부분의 공격 행위가 수행된 것을 탐지할 수 있었다. 또한, 특정 명령(powerpick, pth, desktop 등)을 제외한 명령들은 어떤 명령이 수행되었는지 정확히 판단할 수 없었지만, Cobalt Strike Beacon이 명령을 수행하기 위한 패턴을 탐지해 시스템 내 Cobalt Strike가 동작 중인 사실을 확인할 수 있었다.

- **HTTPS Beacon 생성 및 실행**

HTTPS Beacon의 실행 행위는 다음과 같은 룰로 탐지할 수 있었다.

[표 113] Beacon 실행을 탐지할 수 있는 Sysmon Configure Rule과 해킹진단도구 탐지 룰 목록

번호	탐지 룰 구분	탐지 룰 이름
1	Sysmon Configure Rule	PipeEvent – Turn on Cobalt Strike Beacon
2		RegistryEvent – Change Internet ZoneMap Setting
3		NetworkConnect – HTTPS
4	해킹진단도구 탐지 룰	[EVT] Sysmon_17_18_CobaltStrike_CreateAndAccess_NamedPipe

- **내부 정보 수집, 포트 스캐닝**

내부 정보를 수집하고 포트 스캐닝을 수행하는 행위는 다음과 같은 룰로 탐지할 수 있었다.

[표 114] 내부 정보 수집 및 포트 스캐닝을 탐지할 수 있는 Sysmon Configure Rule과 해킹진단도구 탐지 룰 목록

번호	탐지 룰 구분	탐지 룰 이름
1	Sysmon Configure Rule	ProcessCreate – rundll32.exe with No Arguments
2		ProcessAccess – Cobalt Strike Command Execute
3		PipeEvent – Cobalt Strike Command Execute
4		ImageLoad – Cobalt Strike Command Execute(Powerpick)
5	해킹진단도구 탐지 룰	[EVT] Sysmon_1_Suspicious_ProcessCreate_Rundll32
6		[EVT] Sysmon_10_CobaltStrike_ProcessAccess
7		[EVT] Sysmon_17_18_CobaltStrike_CreateAndAccess_NamedPipe
8		[EVT] Sysmon_7_CobaltStrike_Command_powerpick

- **도메인 관리자 크리덴셜 획득**

도메인 관리자 계정의 크리덴셜을 획득하는 행위는 다음과 같은 룰로 탐지할 수 있었다.

[표 115] 도메인 관리자 크리덴셜 획득을 탐지할 수 있는 Sysmon Configure Rule과 해킹진단도구 탐지 룰 목록

번호	탐지 룰 구분	탐지 룰 이름
1	Sysmon Configure Rule	ProcessCreate – rundll32.exe with No Arguments
2		ProcessAccess – Cobalt Strike Command Execute
3		PipeEvent – Cobalt Strike Command Execute
4		ProcessAccess – Credential Dumping
5		ProcessCreate – Cobalt Strike Command Execute(pth)
6	해킹진단도구 탐지 룰	[EVT] Sysmon_1_Suspicious_ProcessCreate_Rundll32
7		[EVT] Sysmon_17_18_CobaltStrike_CreateAndAccess_NamedPipe
8		[EVT] Sysmon_1_CobaltStrike_Command_pth
9		[EVT] Security_4624_Suspicious_Network_Logon

- **획득한 도메인 관리자로 접근**

획득한 도메인 관리자 계정을 통해 내부 이동하는 행위는 다음과 같은 룰로 탐지할 수 있었다.

[표 116] 내부 이동을 탐지할 수 있는 Sysmon Configure Rule과 해킹진단도구 탐지 룰 목록

번호	탐지 룰 구분	탐지 룰 이름
1	Sysmon Configure Rule	RegistryEvent – Create Service in PowerShell
2		ProcessCreate – Remote Commands
3		ProcessCreate – Execute Encoded PowerShell Commands
4		ProcessCreate – Cobalt Strike Command Execute(Jump-PsExec_psh)
5		PipeEvent – Cobalt Strike Command Execute
6	해킹진단도구 탐지 룰	[EVT] System_7045_Serviceinstall_PowerShell
7		[EVT] Sysmon_1_CobaltStrike_Command_jump
8		[EVT] Sysmon_17_18_CobaltStrike_CreateAndAccess_NamedPipe

- **포트 포워딩 설정**

포트 포워딩을 설정하는 rportfwd 명령은 추가 이벤트를 발생하지 않기 때문에 작성된 룰로는 탐지할 수 없었다.

- **도메인 계정 크리덴셜 획득**

도메인 계정의 크리덴셜을 획득하는 행위는 다음과 같은 룰로 탐지할 수 있었다.

[표 117] 도메인 계정 크리덴셜 획득을 탐지할 수 있는 Sysmon Configure Rule과 해킹진단도구 탐지 룰 목록

번호	탐지 룰 구분	탐지 룰 이름
1	Sysmon Configure Rule	ProcessCreate – rundll32.exe with No Arguments
2		ProcessAccess – Cobalt Strike Command Execute
3		CreateRemoteTread – Process Injection
4		PipeEvent – Cobalt Strike Command Execute
5		ProcessCreate – Cobalt Strike Command Execute(pth)
6		ProcessAccess – Credential Dumping
7	해킹진단도구 탐지 룰	[EVT] Sysmon_1_Suspicious_ProcessCreate_Rundll32
8		[EVT] Sysmon_10_CobaltStrike_ProcessAccess
9		[EVT] Sysmon_17_18_CobaltStrike_CreateAndAccess_NamedPipe
10		[EVT] Sysmon_1_CobaltStrike_Command_pth
11		[EVT] Security_4624_Suspicious_Network_Logon

- **추가 SMB Beacon 업로드**

SMB Beacon 파일을 다른 시스템으로 전파하는 행위는 다음과 같은 룰로 탐지할 수 있었다.

[표 118] 파일 업로드를 탐지할 수 있는 Sysmon Configure Rule과 해킹진단도구 탐지 룰 목록

번호	탐지 룰 구분	탐지 룰 이름
1	Sysmon Configure Rule	FileCreate
2	해킹진단도구 탐지 룰	[EVT] Sysmon_11_CreateFile_ADMINShareFolder

- **SMB Beacon 연결**

전파한 SMB Beacon을 실행하고 연결하는 행위는 다음과 같은 룰로 탐지할 수 있었다.

[표 119] SMB Beacon 연결을 탐지할 수 있는 Sysmon Configure Rule과 해킹진단도구 탐지 룰 목록

번호	탐지 룰 구분	탐지 룰 이름
1	Sysmon Configure Rule	ProcessCreate – Remote Commands
2		PipeEvent – Turn on Cobalt Strike Beacon
3		NetworkConnect – SMB
4	해킹진단도구 탐지 룰	[EVT] Sysmon_17_18_CobaltStrike_CreateAndAccess_NamedPipe

- **호스트 정보 수집**

호스트 정보를 수집하기 위한 PowerShell 스크립트 실행 행위는 다음과 같은 룰로 탐지할 수 있었다.

[표 120] 호스트 정보 수집을 탐지할 수 있는 Sysmon Configure Rule과 해킹진단도구 탐지 룰 목록

번호	탐지 룰 구분	탐지 룰 이름
1	Sysmon Configure Rule	ProcessCreate – Execute Encoded PowerShell Commands
2		ProcessAccess – Cobalt Strike Command Execute
3		ImageLoad – PowerShell
4	해킹진단도구 탐지 룰	[EVT]_16_윈도우 파워셸을 이용한 난독화 명령어 실행 탐지
5		[EVT] Sysmon_10_CobaltStrike_ProcessAccess

- **호스트 모니터링**

사용자 활동을 모니터링할 수 있는 호스트 모니터링 행위는 다음과 같은 룰로 탐지할 수 있었다.

[표 121] 호스트 모니터링을 탐지할 수 있는 Sysmon Configure Rule과 해킹진단도구 탐지 룰 목록

번호	탐지 룰 구분	탐지 룰 이름
1	Sysmon Configure Rule	ProcessCreate – rundll32.exe with No Arguments
2		ProcessAccess – Cobalt Strike Command Execute
3		NetworkConnect – Cobalt Strike Command Execute(Desktop)
4	해킹진단도구 탐지 룰	[EVT] Sysmon_1_Suspicious_ProcessCreate_Rundll32
5		[EVT] Sysmon_10_CobaltStrike_ProcessAccess
6		[EVT] Sysmon_3_CobaltStrike_Command_desktop

- **정보 유출**

정보 유출 행위를 위해 수행한 download 명령은 추가 이벤트를 발생하지 않기 때문에 작성된 룰로는 탐지할 수 없었다.

- **VPN 연결**

지속적인 접근을 위한 VPN 연결 행위는 다음과 같은 룰로 탐지할 수 있었다.

[표 122] VPN 연결을 탐지할 수 있는 Sysmon Configure Rule과 해킹진단도구 탐지 룰 목록

번호	탐지 룰 구분	탐지 룰 이름
1	Sysmon Configure Rule	ProcessCreate – rundll32.exe with No Arguments
2		ProcessAccess – Cobalt Strike Command Execute
3		FileCreate – Cobalt Strike Command Execute(CovertVPN)
4		RegistryEvent – Install NPcap
5		DriverLoad
6		ImageLoad – Cobalt Strike Command Execute(CovertVPN)
7	해킹진단도구 탐지 룰	[EVT] Sysmon_1_Suspicious_ProcessCreate_Rundll32
8		[EVT] Sysmon_10_CobaltStrike_ProcessAccess
9		[EVT] Sysmon_11_Suspicious_FileCreate_NPcap
10		[EVT] System_7045_ServiceInstall_NPcap_Driver
11		[EVT] Sysmon_7_Suspicious_ImageLoaded_NPcap

7. 결론

본 연구에서는 Cobalt Strike 4.9.1 버전을 기준으로 제공되는 98개의 명령이 시스템에 남기는 다양한 아티팩트를 체계적으로 분석해, Cobalt Strike를 악용한 위협을 탐지할 수 있는 실질적인 방안을 제시했다. 이를 통해 로그 설정이 강화되지 않은 환경에서는 Cobalt Strike 관련 행위를 파악하기 어렵다는 점을 확인했으며, Cobalt Strike와 같은 고도화된 공격 도구를 탐지하기 위해서는 Sysmon과 같은 추가 로그 설정의 중요성을 확인할 수 있었다. 로그는 침해사고의 원인을 규명하고 내부 전파 경로를 식별하는 데 핵심적인 역할을 하기 때문에, 용량 문제나 설정의 차이를 고려하더라도 반드시 로그 강화 설정을 적용해야 할 필요가 있다.

또한, 본 연구는 Cobalt Strike를 악용한 위협을 일반인도 신속하고 쉽게 식별할 수 있도록 하고, 보안 담당자나 사고 대응 전문가가 빠르게 대응할 수 있는 탐지 룰을 설계했다. Sysmon과 해킹진단도구를 활용해 총 47개의 탐지 룰을 개발했으며, 이를 검증하기 위해 AD 환경에서 발생할 수 있는 침해사고 시나리오를 기반으로 테스트를 진행했다. 그 결과, 대부분의 행위를 탐지 룰을 통해 탐지할 수 있어 실제 악성 행위에서 탐지 룰에 대한 효과성과 유효성을 확인할 수 있었다.

다만, Sysmon Configure Rule과 해킹진단도구 탐지 룰은 단일 이벤트에 기반한 규칙 적용의 한계가 있어, Cobalt Strike 위협을 보다 정확히 탐지하기 위해서는 여러 이벤트 패턴을 결합한 탐지 방안이 필요하다. 향후 더 정교한 탐지를 위해 다양한 이벤트에서 위협 패턴을 찾아낼 수 있는 도구 개발이 요구된다. 또한, Cobalt Strike 확장 기능 사용에 따른 흔적 탐지 방안에 대한 추가 연구도 필요하다. 특히, 공격자들이 가장 많이 활용하는 확장 기능 기반으로 중점적으로 분석해 해당 기능을 효과적으로 탐지할 수 있는 방안을 마련하는 것이 중요하다. 따라서, 이와 같은 사항들을 향후 연구 방향으로 설정해 진행할 예정이다.

결론적으로, 본 연구는 Sysmon과 해킹진단도구를 활용한 탐지 룰의 설계와 검증을 통해 다양한 공격 시나리오에서 위협을 조기에 탐지하고 대응할 수 있는 효과적인 방안을 제안했다. 이러한 탐지 방안은 조직의 효율적인 위협 모니터링과 대응 프로세스를 지원해 사이버 방어 역량을 근본적으로 향상시키는 데 기여할 것으로 기대된다.

참고 자료

번호	참고 자료
1	Mandiant, “M-Trends 2024 Special Report”, services.google.com/fh/files/misc/m-trends-2024.pdf
2	Cobalt Strike, “Cobalt Strike Features”, www.cobaltstrike.com/product/features
3	Cobalt Strike, “Cobalt Strike User Guide”, hstechdocs.helpsystems.com/manuals/cobaltstrike/current/userguide/content/topics/welcome_main.htm
4	Cobalt Strike, “Cobalt Strike Installation Guide”, hstechdocs.helpsystems.com/manuals/cobaltstrike/current/cobalt-strike-install.pdf
5	Cobalt Strike, “Cobalt Strike Adversary Simulations and Red Team Operations, www.cobaltstrike.com/resources/datasheets/cobalt-strike
6	harleyQu1nn, “AggressorScripts”, github.com/harleyQu1nn/AggressorScripts
7	mgeeky, “cobalt-arsenal”, github.com/mgeeky/cobalt-arsenal
8	rsmudge, “ElevateKit”, github.com/rsmudge/ElevateKit
9	SentinelOne, “What is Cobalt Strike? Examples & Modules”, www.sentinelone.com/cybersecurity-101/threat-intelligence/what-is-cobalt-strike/
10	Sean Gallagher, “Sophos 2023 Threat Report: the continued evolution of “Crime-as-a-Service”, news.sophos.com/en-us/2022/11/17/sophos-2023-threat-report-the-continued-evolution-of-crime-as-a-service/
11	John Shier, Angela Gunn, “It’s Oh So Quiet (?): The Sophos Active Adversary Report for 1H 2024”, news.sophos.com/en-us/2024/04/03/active-adversary-report-1h-2024/
12	The DFIR Report, “Stolen Images Campaign Ends in Conti Ransomware”, thedfirreport.com/2022/04/04/stolen-images-campaign-ends-in-conti-ransomware/
13	The DFIR Report, “SQL Brute Force Leads to BlueSky Ransomware”, thedfirreport.com/2023/12/04/sql-brute-force-leads-to-bluesky-ransomware/
14	Christine Barry, “Rhysida ransomware: The creepy crawling criminal hiding in the dark”, blog.barracuda.com/2024/05/09/rhysida-ransomware--the-creepy-crawling-criminal-hiding-in-the-d
15	Microsoft Threat Intelligence, “Threat actors misusing Quick Assist in social engineering attacks leading to ransomware”, microsoft.com/en-us/security/blog/2024/05/15/threat-actors-misusing-quick-assist-in-social-engineering-attacks-leading-to-ransomware/
16	The DFIR Report, “IcedID Brings ScreenConnect and CSharp Streamer to ALPHV Ransomware Deployment”, thedfirreport.com/2024/06/10/icedid-brings-screenconnect-and-csharp-streamer-to-alphv-ransomware-deployment/
17	The DFIR Report, “BlackSuit Ransomware”, thedfirreport.com/2024/08/26/blacksuit-ransomware/
18	Joey Chen, Ashley Shen, Vitor Ventura, “APT41 likely compromised Taiwanese government-affiliated research institute with ShadowPad and Cobalt Strike”, blog.talosintelligence.com/chinese-hacking-group-apt41-compromised-taiwanese-government-affiliated-research-institute-with-shadowpad-and-cobaltstrike-2/
19	Ahnlab, “Cobalt Strike와 MS Exchange Server 취약점을 이용한 침해사례 포렌식 분석”, asec.ahnlab.com/ko/27448/
20	Ahnlab, “달빗(Dalbit,m00nlight): 중국 해커 그룹의 APT 공격 캠페인”, asec.ahnlab.com/ko/46431/

번호	참고 자료
21	Ahnlab, “취약한 MS-SQL 서버를 대상으로 유포 중인 코발트 스트라이크 (2)”, asec.ahnlab.com/ko/31657/
22	Andrii Bezverkhyi, “Cobalt Strike Beacon Malware Spread Via Targeted Phishing Emails Related to Azovstal: Cyber-Attack on Ukrainian Government Entities”, socprime.com/blog/cobalt-strike-beacon-malware-spread-via-targeted-phishing-emails-related-to-azovstal-cyber-attack-on-ukrainian-government-entities/
23	Ahnlab, “CobaltStrike를 이용한 아파치 웹 서버 대상 크립토재킹 공격 캠페인”, asec.ahnlab.com/ko/58882/
24	권혁주, 곽진, "명령 제어 프레임워크 (Command and Control Framework) 도구 추적 방안에 대한 연구", dbpia.co.kr/journal/articleDetail?nodeId=NODE11554248
25	오원보, “침투 테스팅 도구 Cobalt Strike Part.1 기능편”, www.igloo.co.kr/security-information/침투-테스팅-도구-cobalt-strike-part-1-기능편/
26	오원보, “침투 테스팅 도구 Cobalt Strike Part.2 실전편”, www.igloo.co.kr/security-information/침투-테스팅-도구-cobalt-strike-part-2-실전편/
27	Balasubramanya C, “CobaltStrike Forensics”, medium.com/@balasubramanya.c/cobaltstrike-forensics-32e6ce68ce42
28	Omar Alanezi, “Memory Forensics:Hunting Cobalt Strike in Memory”, cybersync.org/blogs-en/hunting_cobalt_strike_in_memory
29	Riccardo Ancarani, “Detecting Cobalt Strike Default Modules via Named Pipe Analysis”, labs.withsecure.com/publications/detecting-cobalt-strike-default-modules-via-named-pipe-analysis
30	The DFIR Report, “Cobalt Strike, a Defender’s Guide”, thedfirreport.com/2021/08/29/cobalt-strike-a-defenders-guide/
31	Lexfo, “Cobalt Strike Investigation Part 1”, https://blog.lexfo.fr/Cobalt Strike Investigation Part 1.html
32	Lexfo, “Cobalt Strike Investigation – Part 2”, blog.lexfo.fr/cobalt-strike-investigation-part-2.html
33	Erwan Chevalier, Narimane Lavay, “Hunting and detecting Cobalt Strike”, blog.sekoia.io/hunting-and-detecting-cobalt-strike/
34	Invictus Incident Response, “Responding to a Cobalt Strike attack – Part I”, linkedin.com/pulse/responding-cobalt-strike-attack-part-i-invictus-incident-response
35	Invictus Incident Response, “Responding to a Cobalt Strike attack – Part II”, linkedin.com/pulse/responding-cobalt-strike-attack-part-ii-invictus-incident-response
36	Invictus Incident Response, “Responding to a Cobalt Strike attack – Part III”, linkedin.com/pulse/responding-cobalt-strike-attack-part-iii-invictus-incident-response
37	Chris Navarrete, Durgesh Sangvikar, Andrew Guan, Yu Fu, Yanhui Jia, Siddhart Shibiraj, “Cobalt Strike Analysis and Tutorial: How Malleable C2 Profiles Make Cobalt Strike Difficult to Detect”, unit42.paloaltonetworks.com/cobalt-strike-malleable-c2-profile/
38	Cyb3rSn0rlax, “Detecting CONTI CobaltStrike Lateral Movement Techniques – Part 1”, unh4ck.com/detection-engineering-and-threat-hunting/lateral-movement/detecting-conti-cobaltstrike-lateral-movement-techniques-part-1
39	Cyb3rSn0rlax, “Detecting CONTI CobaltStrike Lateral Movement Techniques – Part 2”, www.unh4ck.com/detection-engineering-and-threat-hunting/lateral-movement/detecting-conti-cobaltstrike-lateral-movement-techniques-part-2
40	Insikt Group, “Full-Spectrum Cobalt Strike Detection”, go.recordedfuture.com/hubfs/reports/mtp-2021-0914.pdf
41	Soft2000, “국내 기업 위협하는 코발트 스트라이크 심층 분석”, www.soft2000.com/24701

PLAINBIT Co., Ltd.

사이버위협대응센터 경기도 성남시 분당구 대왕판교로 670, A동 504호
디지털포렌식센터 서울시 서초구 서초대로 56길 22, 5층

T. 031-8039-7455 F. 031-8016-0913
plainbit.co.kr