Execution(실행)

Execution(실행)은 공격자가 제어하는 코드가 로컬 또는 원격 시스템에서 실행되도록 하는 기술로 구성된다. 악성 코드를 실행하는 기술은 네트워크 탐색이나 데이터 탈취 같이 보다 광범위한 목표를 달성하기 위해 다른 전술의 기술과 결합하는 경우가 많다. 이는 공격자가 원격 접속 도구를 실행시키기 위하여 PowerShell 스크립트를 사용하는 것을 예시로 들 수 있다.

- Command and Scripting Interpreter(명령 및 스크립팅 인터프리터)

[서브 기술] PowerShell, AppleScript, Windows Command Shell, Unix Shell, Visual Basic, Python, JavaScript, Network Device CLI

공격자는 Command and Scripting Interpreter 를 남용하여 명령, 스크립트 및 바이너리를 실행할 수 있다. 이러한 인터페이스와 언어는 컴퓨터 시스템과 상호 작용하는 방법을 제공하며 다양한 플랫폼에서 공통적인 기능이다. 대부분 시스템에는 command-line 인터페이스와 스크립팅 기능이 내장되어 있다. 예를 들어 macOS 및 Linux 배포판에는 Unix Shell 의 일부가 포함되어 있으며, Windows 에는 Windows Command Shell 및 PowerShell 이 포함된다.

Python 과 같은 플랫폼 간 인터프리터는 물론 JavaScript 및 Visual Basic 과 같은 클라이언트 응용 프로그램과 일반적으로 연결된 인터프리터도 있다.

공격자는 임의의 명령을 실행하는 수단으로 이러한 기술을 다양한 방식으로 남용할 수 있다. 명령과 스크립트는 피해자에게 유인 문서로 전달되거나 기존 C2 에서 내려받은 보조 페이로드로 전달되는 초기 액세스 페이로드에 포함될 수 있다. 공격자는 대화형 터미널/쉘을 통해 명령을 실행할 수도 있다.

*인터프리터 : 프로그래밍 언어의 소스 코드를 바로 실행하는 컴퓨터 프로그램

- Container Administration Command(컨테이너 관리 명령)

공격자는 컨테이너 관리 서비스를 남용하여 컨테이너 내에서 명령을 실행할 수 있다. Docker 데몬, Kubernetes API 서버 또는 kubelet 과 같은 컨테이너 관리 서비스를 사용하면 환경 내에서 컨테이너를 원격으로 관리할 수 있다.

Docker 에서 공격자는 컨테이너 배포 중에 스크립트 또는 명령을 실행하는 진입점을 지정하거나 docker exec 와 같은 명령을 사용하여 실행 중인 컨테이너 내에서 명령을 실행할 수 있다. Kubernetes 에서 공격자는 충분한 권한이 있는 경우 Kubernetes API 서버, kubelet 과의 상호 작용을 통해 또는 kubectl exec 와 같은 명령을 실행하여 클러스터의 컨테이너에서 원격 실행을 얻을 수 있다.

- Deploy Container(컨테이너 배포)

공격자는 Execution(실행)을 쉽게 하거나 방어를 회피하기 위해 환경에 컨테이너를 배포할 수 있다. 때에 따라 공격자는 악성 코드를 실행하거나 다운로드하는 프로세스와 같이 특정 이미지 또는 배포와 관련된 프로세스를 실행하기 위해 새 컨테이너를 배포할 수 있다. 다른 곳에서는 공격자가 환경 내의 기존 방어를 우회하기 위해 네트워크 규칙, 사용자 제한 등 없이 구성된 새 컨테이너를 배포할 수 있다.

컨테이너는 Docker 의 생성 및 시작 API 또는 Kubernetes 대시보드 또는 Kubeflow 와 같은 웹 애플리케이션을 통해 다양한 방법으로 배포할 수 있다. 공격자는 검색되거나 구축된 악성 이미지를 기반으로 하거나 런타임에 악성 페이로드를 다운로드하고 실행하는 무해한 이미지에서 컨테이너를 배포할 수 있다.

- Exploitation for Client Execution(프로그램 취약점 악용)

공격자는 클라이언트 애플리케이션의 소프트웨어 취약점을 악용하여 코드를 실행할 수 있다. 예상치 못한 동작으로 이어질 수 있는 안전하지 않은 코딩 관행으로 인해 소프트웨어에 취약점이 존재할 수 있다. 공격자는 임의 코드 실행을 위한 표적 공격을 통해 특정 취약점을 이용할 수 있다. 종종 공격적인 툴킷에 대한 가장 가치 있는 취약점 공격은 원격 시스템에 대한 액세스 권한을 얻는 데 사용될 수 있기 때문에 원격 시스템에서 코드 실행을 획득하는 데 사용할 수 있는 취약점 공격이다. 사용자는 일반적으로 작업을 수행하는 데 사용하는 응용 프로그램과 관련된 파일을 볼 것으로 예상하므로 높은 유용성으로 인해 익스플로잇 연구 및 개발에 유용한 대상이다.

*브라우저 기반 악용
웹 브라우저는 Drive-by Compromise 및 Spearphishing Link 를 통한 일반적인 대상이다. 엔드포인트 시스템은 일반적인 웹 브라우징을 통해 또는 웹 브라우저를 악용하는 데 사용되는 공격자가 제어하는 사이트로 연결되는 스피어피싱 이메일 링크의 표적이 되는 특정 사용자로부터 손상될 수 있다.

*사무용 애플리케이션
Microsoft Office 와 같은 일반 사무실 및 생산성 응용 프로그램도 피싱을 통해 표적이 된다. 악성 파일은 첨부 파일로 직접 전송되거나 다운로드 링크를 통해 전송된다. 이를 위해서는 사용자가 취약점 공격을 실행하기 위해 문서나 파일을 열어야 한다.

*일반적인 타사 애플리케이션
일반적으로 보거나 대상 네트워크에 배포된 소프트웨어의 일부인 다른 응용 프로그램도 악용에 사용될 수 있다. 기업 환경에서 일반적으로 사용되는 Adobe Reader 및 Flash 와 같은 응용 프로그램은 시스템에 대한 액세스 권한을 얻으려는 공격자의 일상적인 표적이 되어 왔다. 소프트웨어 및 취약점의 특성에 따라 일부는 브라우저에서 악용되거나 사용자가 파일을 열도록 요구할 수 있다. 예를 들어, 일부 Flash 취약점 공격은 Microsoft Office 문서 내의 개체로 전달되었다.

- Inter-Process Communication(프로세스 간 통신)

[서브 기술] Component Object Model, Dynamic Data Exchange

공격자는 로컬 코드 또는 명령 실행을 위해 IPC(프로세스 간 통신) 메커니즘을 남용할 수 있다. IPC 는 일반적으로 프로세스에서 데이터를 공유하거나, 서로 통신하거나, 실행을 동기화하는 데 사용된다. IPC 는 또한 프로세스가 순환 대기 패턴에 갇혀 있을 때 발생하는 교착 상태와 같은 상황을 피하기 위해 일반적으로 사용된다.

공격자는 IPC 를 남용하여 임의의 코드나 명령을 실행할 수 있다. IPC 메커니즘은 OS 에 따라 다를 수 있지만 일반적으로 프로그래밍 언어/라이브러리 또는 Windows Dynamic Data Exchange 또는 Component Object Model 과 같은 기본 인터페이스를 통해 액세스할 수 있는 형태로 존재한다. 명령 및 스크립팅 인터프리터와 같은 상위 수준 실행 매체도 기본 IPC 메커니즘을 활용할 수 있다.

- Native API(네이티브 API)

공격자는 기본 OS API(응용 프로그래밍 인터페이스)와 직접 상호 작용하여 행동을 실행할 수 있다. 네이티브 API 는 하드웨어/장치, 메모리 및 프로세스와 관련된 것과 같은 커널 내에서 하위 수준 OS 서비스를 호출하는 제어된 수단을 제공한다. 이러한 기본 API 는 시스템 부팅 동안(다른 시스템 구성 요소가 아직 초기화되지 않은 경우) OS 에 의해 활용될 뿐만 아니라 일상적인 작업 중에 작업 및 요청을 수행한다.

네이티브 API 가 제공하는 기능은 종종 인터페이스와 라이브러리를 통해 사용자 모드 애플리케이션에 노출된다. 예를 들어 Windows API CreateProcess() 또는 GNU fork()와 같은 함수를 사용하면 프로그램과 스크립트가 다른 프로세스를 시작할 수 있다. 이를 통해 API 호출자는 바이너리를 실행하고, CLI 명령을 실행하고, 모듈을 로드하는 등의 작업을 수행할 수 있다. 다양한 시스템 작업에 대해 수천 개의 유사한 API 기능이 존재하기 때문이다.

Microsoft .NET 및 macOS Cocoa 와 같은 상위 수준 소프트웨어 프레임워크도 기본 API 와 상호 작용하는 데 사용할 수 있다. 이러한 프레임워크는 일반적으로 API 기능에 언어 래퍼/추상화를 제공하며 코드의 사용 용이성/이식성을 위해 설계되었다.

공격자는 행동을 실행하는 수단으로 이러한 기본 API 기능을 남용할 수 있다. 명령 및 스크립팅 인터프리터와 유사하게 기본 API 및 인터페이스 계층 구조는 피해를 입은 시스템의 다양한 구성 요소와 상호 작용하고 이를 활용하는 메커니즘을 제공한다.

- Scheduled Task/Job(예약 작업)

[서브 기술] Cron, Launchd, Scheduled Task, Systemd Timers, Container Orchestration Job

공격자는 작업 예약 기능을 악용하여 악성 코드의 초기 또는 반복 실행을 용이하게 할 수 있다. 모든 주요 운영 체제에는 지정된 날짜와 시간에 실행될 프로그램이나 스크립트를 예약하는 유틸리티가 있다. 적절한 인증이 충족되는 경우 원격 시스템에서 작업을 예약할 수도 있다(예: Windows 환경에서 RPC 및 파일 및 프린터 공유). 원격 시스템에서 작업을 예약하려면 일반적으로 원격 시스템에서 관리자 또는 권한이 있는 그룹의 구성원이어야 한다.

공격자는 작업 일정을 사용하여 시스템 시작 시 또는 지속성을 위해 일정에 따라 프로그램을 실행할 수 있다. 이러한 메커니즘을 악용하여 지정된 계정(예: 높은 권한/권한이 있는 계정)의 컨텍스트에서 프로세스를 실행할 수도 있다.

- Shared Modules

공격자는 공유 모듈을 악용하여 악성 페이로드를 실행할 수 있다. Windows 모듈 로더는 임의의 로컬 경로 및 임의의 UNC(범용 명명 규칙) 네트워크 경로에서 DLL 을 로드하도록 지시할 수 있다. 이 기능은 NTDLL.dll 에 있으며 Win32 API 의 CreateProcess, LoadLibrary 등과 같은 함수에서 호출되는 Windows Native API 의 일부이다.

공격자는 피해자 시스템에서 임의의 코드를 실행하는 방법으로 이 기능을 사용할 수 있다. 예를 들어, 맬웨어는 공유 모듈을 실행하여 추가 구성 요소나 기능을 로드할 수 있다.

- Software Deployment Tools(소프트웨어 배포 도구)

공격자는 관리, 모니터링 및 배포 시스템과 같은 엔터프라이즈 네트워크 내에 설치된 타사 소프트웨어 제품군에 액세스하고 이를 사용하여 네트워크를 통해 Lateral Movement(시스템 내부 이동) 할 수 있다. 타사 응용 프로그램 및 소프트웨어 배포 시스템은 관리 목적으로 네트워크 환경에서 사용 중일 수 있다(ex SCCM, HBSS, Altiris 등).

타사 네트워크 또는 전사적 소프트웨어 시스템에 대한 액세스는 공격자가 그러한 시스템에 연결된 모든 시스템에서 원격 코드 실행을 가능하게 할 수 있다. 액세스는 다른 시스템으로 횡적으로 이동하거나, 정보를 수집하거나, 모든 엔드포인트에서 하드 드라이브를 지우는 것과 같은 특정 효과를 일으키는 데 사용될 수 있다.

이 작업에 필요한 권한은 시스템 구성에 따라 다릅니다. 로컬 자격 증명으로 타사 시스템에 직접 액세스할 수 있거나 특정 도메인 자격 증명이 필요할 수 있다. 그러나 시스템에 로그인하거나 의도한 목적을 수행하기 위해 관리 계정이 필요할 수 있다.

- System Services(시스템 서비스)

[서브 기술] Launchctl, Service Execution

공격자는 시스템 서비스나 데몬을 남용하여 명령이나 프로그램을 실행할 수 있다. 공격자는 서비스와 상호 작용하거나 서비스를 생성하여 악성 콘텐츠를 실행할 수 있다. 많은 서비스가 부팅 시 실행되도록 설정되어 지속성(시스템 프로세스 생성 또는 수정)을 달성하는 데 도움이 될 수 있지만, 공격자는 일회성 또는 임시 실행을 위해 서비스를 남용할 수도 있다.

- User Execution(사용자 실행)

[서브 기술] Malicious Link, Malicious File, Malicious Image

공격자는 실행을 얻기 위해 사용자의 특정 행동에 의존할 수 있다. 예를 들어 사용자는 악성 문서 파일이나 링크를 열어 악성 코드를 실행하도록 소셜 엔지니어링을 받을 수 있다. 이러한 사용자 행동은 일반적으로 피싱 형태의 후속 행동으로 관찰된다.

사용자 실행은 초기 액세스 직후에 자주 발생하지만, 공격자는 사용자가 파일을 클릭하기를 바라는 공유 디렉토리나 사용자의 데스크탑에 파일을 배치하는 경우와 같이 침입의 다른 단계에서 발생할 수 있다. 이 활동은 내부 스피어피싱 직후에도 나타날 수 있다.

- WMI(Windows Management Instrumentation) 윈도우 관리 도구

공격자는 WMI(Windows Management Instrumentation)를 남용하여 Execution(실행)을 달성할 수 있다. WMI 는 Windows 시스템 구성 요소에 대한 로컬 및 원격 액세스를 위한 균일한 환경을 제공하는 Windows 관리 기능이다. 로컬 및 원격 액세스를 위한 WMI 서비스와 원격 액세스를 위한 서버 메시지 블록(SMB) 및 원격 프로시저 호출 서비스(RPCS)에 의존한다. RPC 는 포트 135 를 통해 작동한다.

공격자는 WMI 를 사용하여 로컬 및 원격 시스템과 상호 작용하고 이를 Lateral Movement(시스템 내부 이동)의 일부로 파일 검색 및 원격 실행을 위한 정보 수집과 같은 많은 전술 기능을 수행하는 수단으로 사용할 수 있다.