한국컴퓨터정보학회 동계학술대회 논문집 제26권 제1호 (2018. 1)

WMI 기반 악성코드 탐지 기법에 관한 연구

용승림*, 정명준O
O*인하공업전문대학 컴퓨터시스템과
e-mail: slyong.inhatc.ac.kr*, jamzzoon@gmail.comO

A Study on WMI-based Malicious Code Detection

Seunglim Yong*, Myeong-jun JeongO
O*Dept. of Computer Systems and engineering, Inha technical college

● 요 약 ●

최근에 WMI를 악용한 악성코드 공격이 증가하고 있다. WMI는 악성 프로그램을 설치하지 않아도 레지스트리, 파일시스템 등 중요한 정보에 접근할 수 있다. 또한, 윈도우 운영체제에 내장된 프로그램이기 때문에 백신에서 탐지하기 어렵다. 본 논문에서는 WMI를 이용한 악성코드 탐지를 위하여 제안하는 방법은 API를 후킹하여 메모리에서 실행될 DLL을 보고 악성코드를 탐지하는 방법을 제안한다.

키워드: WMI(WMI), 악성코드(malicious software)

I. Introduction

파일 없는 악성코드 공격은 공격자가 피해자의 시스템에 소프트웨어를 설치하지 않고, 윈도우 운영체제에서 제공하는 내장된 소프트웨어를 이용한다. 최근에는 악성 매크로를 이용한 파일 없는 악성코드로 Locky 랜섬웨어의 새로운 버전인 Lukitus와 뱅킹 악성코드인 Emotet의 변종이 발견되었다. [1]

McAfee Labs의 2017년 3분기 보고서에 따르면, Powershell을 이용한 파일 없는 악성코드 공격이 계속적으로 증가하고 있다. Powershell과 WMI를 이용하면 시스템 대부분 기능을 이용할 수 있으며, 이들을 통해 실행되는 명령어는 올바른 것으로 간주한다. 따라서 공격자들은 Powershell과 WMI를 공격의 도구로 자주 사용한다. [2]

본 논문에서는 WMI를 이용한 악성코드에 대해 분석하고 그에 대한 대응 방안을 제시한다.

II. Preliminaries

1. Related works

1.1 악성코드 탐지 방법

1.1.1 시그니처 기반 탐지 방법

시그니처 기반 탐지는 가장 널리 사용되는 악성코드 탐지 기술로서 1byte 이상의 패턴 매칭을 통해 악성코드를 탐지한다. 이 방법은 알려진 악성코드의 시그니처를 저장하는 데이터베이스를 유지해야 하며 탐지 속도가 빠르고 대부분의 일반 유형인 악성코드에 효과적이다. 하지만 시그니처 기반 데이터베이스에 없는 악성코드는 탐지되지 않으며 새로운 위협이 발견될 때마다 업데이트해야 한다. 또한, 비교적 단순한 난독화 기법을 사용해서 시그니처 탐지를 회피할 수 있다. [3]

1.1.2 행동 기반 탐지 방법

행동 기반 악성코드 탐지 방법은 프로그램의 동작을 모니터링하여 악의적인 행위 여부를 판단한다. 행동 기반 탐지 방법은 프로그램을 실제로 실행함으로써 프로그램의 동작을 관찰하고 미리 정의된 악의적인 동작을 수행하면 악성코드로 식별한다. [4]

1.2 WMI 기반 악성코드

1.2.1 WMI

WMI(Windows Management Instrumentation)는 윈도우98 이후 모든 버전에 내장된 강력한 관리자 프레임워크이다. [5] WMI는 파일시스템 및 레지스트리 정보 확인, 프로세스 실행, 트리거 설정 등 로컬이나 원격 시스템에서 관리 할 수 있도록 많은 기능을 제공한다. WMI는 wmic.exe, wbemtest.exe, 등을 통해 사용할 수 있고, 이를

```
Sub AutoOpen()
    x1 = "Download"
    h = "Str"
    o = "power" & "shell" & ".exe"
    Const HIDDEN_WINDOW = 0
    strComputer = "."
    abcdef = h & "ing"
    Set objWMIService = GetObject("winmgmts:\\" & strComputer & "\root\cimv2")

    Set objStartup = objWMIService.Get("Win32_ProcessStartup")
    Set objConfig = objStartup.SpawnInstance_
    objConfig.ShowWindow = HIDDEN_WINDOW
    Set objProcess = GetObject("winmgmts:\\" & strComputer & "
        \root\cimv2:Win32_Process")
    objProcess.Create o & " -ExecutionPolicy Bypass -WindowStyle Hidden -
        noprofile -noexit -c if ([IntPtr]::size -eq 4) {(new-object Net.
        WebClient)." & x1 & abcdef & "('http://rabbitons.pw/cache') | iex }
        else {(new-object Net.WebClient)." & x1 & abcdef & "('http://rabbitons.
        pw/css') | iex}", Null, objConfig, intProcessID
End Sub
```

Fig. 1. Smaple code of Powersniff

이용하여 정보수집, AV/VM탐지, 원격 코드 실행 등 다양한 기능을 수행할 수 있다. 공격자는 Powershell, VBScript 등 스크립트 언어에서 API를 통해 WMI에 접근하기 때문에 탐지하기 어렵다.

1.2.2 WMI 기반 악성코드 탐지 방법

Matt Graeber는 WMI Event Subscriptions를 이용하여 WMI 이벤트를 탐지하는 IDS 작성 방법을 제안했다. [6] Matt에 따르면 공격자가 자주 사용하는 WMI 쿼리 패턴을 IDS에 등록하여 공격을 탐지할 수 있다. 하지만 WMI에 익숙한 공격자는 방어를 위해 작성한 WMI Event Subscriptions 제거를 통해 IDS 기능을 차단할 수 있다. 또한 Sysinternals Autoruns, Kansa 툴을 이용해서 WMI 데이터베이스에 저장된 악성코드를 발견할 수 있다. 하지만 이러한 도구들은 특정 스냅 샷에서 WMI Persistence 아티팩트가 제거되면 탐지가 불가능하다.

III. The Proposed Scheme

본 논문에서는 WMI를 이용한 악성코드 탐지 방법을 제안한다. 먼저 WMI를 이용한 대표적인 악성코드인 Powersniff를 살펴보고 제안하는 방법을 설명한다.

1.1 WMI를 이용한 악성코드 Powersniff

Powersniff는 MSword 매크로와 Powershell을 악용한 악성코드이다[7]. 2016년 Palo Alto Networks 연구소를 통해 발견된 악성코드로 주로 미국이나 유럽에 위치한 병원, 산업 분야가 주된 공격 대상이 되었다. 사용자가 해당 MSword를 실행하게 되면 WMI가 포함된 AutoOpen() 매크로 함수가 실행된다. 해당 코드는 WMI를 이용하여 사용자 시스템 정보를 수집하고, 악성 Powershell 코드를 실행시킨다. 이후 해당 코드는 셸코드가 포함된 새로운 Powershell 코드를 다운로드 하여 메모리에서 복호화한 후 실행시킨다.

MSword에서 WMI는 사용자 정보를 수집하고, 공격자가 구축한 C&C 서버에 접속한 후 암호화된 악성코드를 다운로드하여 실행시키는 드로퍼(dropper) 역할로 자주 사용된다. 다운로드된 악성코드는 메모리에서 복호화 과정을 통해 실행되기 때문에 기존의 방법으로는 탐지하기 어렵다. 또한, WMI는 운영체제가 제공하는 애플리케이션이므로 백신에서는 악성 프로그램으로 식별하지 않는다. 그러므로 해당 악성코드를 방어하기 위해서 WMI 실행을 사전에 차단하는 것이 효과적이다.

1.2 제안 방법

본 논문에서는 API hooking을 통해 WMI 실행 차단하는 방법을 제안한다. 현재 대부분 애플리케이션은 기능별로 모듈화되어 API 호출을 통해 시스템 자원에 접근할 수 있기 때문에, API Hooking을 이용하면 난독화에 상관없이 애플리케이션에서 사용되는 API의 파라미터를 확인할 수 있다.

winword.exe에서 WMI는 Kernel32.dll의 LoadlibraryExW() 함수를 통해 메모리에서 실행된다. WMI는 %SystemRoot%\system32\wbem에 기능별로 49개 DLL 파일로 구성되어 있으며, 해당 DLL 파일들을 Blacklist로 작성한다. 이후 winword.exe에서 DLL 로딩할 때 사용되는 LoadLiraryExW()를 후킹 하여 파라미터 lpFileName를 모니터링 한다. 파라미터 lpFileName은 LoadLiraryExW()를 통해 실행될 DLL 파일의 이름이다. 모니터링 중 lpFileName가 Blacklist에 존재하면 winword.exe를 종료시킨다.

특정 대상을 공격하기 위해 만들어진 악성 MSword가 포함된 메일 공격은 문서 암호화 기능을 통해 시그니처 기반으로 탐지하는 백신을 쉽게 우회할 수 있다. 하지만 제안하는 방법은 후킹을 통해 메모리에 실행될 DLL을 보고 판단하기 때문에 WMI 실행을 실시간으로 방어할 수 있다.

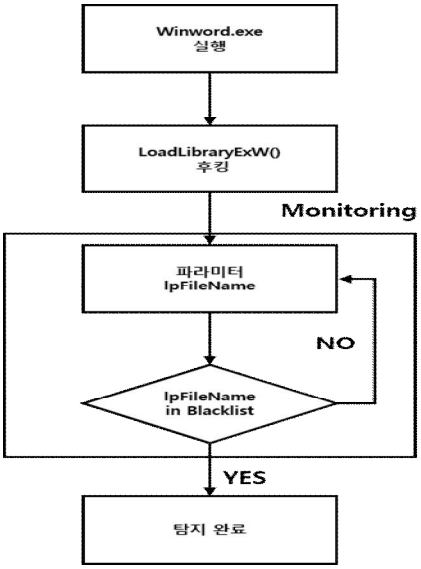

Fig. 2. System Architecture

1.3 실험 결과

본 장에서는 기존의 악성코드 탐지 방법과 논문에서 새로 제안하는 방법을 비교 분석한다. 실험은 windows7 운영체제에서 Microsoft Word 2010 기반으로 진행했다. 기존의 탐지 방법은 officemalscanner 라는 악의적인 Microsoft Office 파일을 분석할 수 있는 포렌식 도구를 활용한 방법과[8] 구글에서 운영하는 클라우드 기반의 온라인 백신 virustotal을 대상으로 비교하였다[9]. 실험 데이터는 [표 1]과 같이 '문서 암호화', '개체 삽입', '편집제한' 기능을 적용한 5가지의 Powersniff 변종이다.

Table 1. Number of Powersniff variant

번호	Powersniff 변종
(A)	매크로 포함, 문서 암호화되지 않음
(B)	매크로 포함, 문서 암호화됨
(C)	'개체' 삽입하여 매크로 포함, 암호화되지 않음
(D)	'개체' 삽입하여 매크로 포함, 암호화됨
(E)	'개체' 삽입 및 '편집제한' 사용, 문서 암호화되지 않음

Table 2. Result

탐지방법 / 변종번호	제안하는 방법	office malscanner[8]	virustotal[9]
(A)	○	○	23/58
(B)	○	×	×
(C)	○	○	23/58
(D)	○	×	×
(E)	○	○	14/58

[표 2] 은 기존의 탐지 방법과 제안한 방법으로 탐지한 결과이다. [8]의 방법은 문서 암호화된 Powersniff 파일은 탐지하지 못했다. [9]의 방법은 각기 다른 58개의 백신 엔진 중 (A),(C)에서는 23개, (E)에서는 14개 엔진이 탐지하였으나, 문서 암호화된 Powersniff 파일인 (B),(D)에서는 탐지하지 못했다. 그러나 제안하는 방법은 API를 후킹하여 메모리에서 실행될 DLL을 보고 탐지하였으며 5가지의 변종 모두를 탐지하였다. 특히 다른 방법들이 탐지하지 못한 문서 암호화가 적용된 Powersniff 파일을 검출하였다.

Ⅳ. Conclusions

본 논문에서는 WMI를 이용한 악성코드 탐지 방법으로 API 후킹을 이용한 방법을 제안하고 차단 프로그램을 구현함으로써 Msword에서 WMI 악성코드가 탐지되는 것을 확인하였다. 파일 없는 악성코드 탐지를 위하여 메모리에 로딩하는 DLL을 확인함으로써 암호화가 적용된 악성 문서 파일도 검출하였다. 제안한 방법은 애플리케이션에서 실행되는 모든 WMI를 악성코드로 탐지할 수 있기 때문에 향후 메모리 분석을 통하여 순기능을 가진 WMI도 악성코드로 오탐할 가능성을 낮추는 연구를 계속할 계획이다.

REFERENCES

[1] https://www.cybereason.com/blog/fileless-malware

[2] https://www.mcafee.com/us/about/newsroom/press-releases/press-release.aspx?news_id=20171217005042

[3] Anusha Damodaran, Fabio Di Troia, Corrado Aaron Visaggio, Thomas H. Austin, Mark Stamp, "A comparison of static, dynamic, and hybrid analysis for malware detection", Journal of Computer Virology and Hacking Techniques, February 2017, Volume 13, Issue 1, pp 1-12

[4] Ashwini Mujumdar, Gayatri Masiwal, Dr. B. B.Meshram, "Analysis of Signature-Based and Behavior-Based Anti-Malware Approaches", International Journal of Advanced Research in Computer Engineering and Technology (IJARCET), Vol. 2, pp. 2037-2039, June 2013.

[5] https://msdn.microsoft.com/en-us/library/aa384642(v=vs.85).aspx

[6] Matt Graeber, "Abusing Windows Management Instrumentation (WMI) to Build a Persistent, Asynchronous, and Fileless Backdoor", Blackhat 2015

[7] https://researchcenter.paloaltonetworks.com/2016/03/Powersniff-malware-used-in-macro-based-attacks

[8] http://www.reconstructer.org/code.html
[9] https://www.virustotal.com