Wykrywanie objawów działania oprogramowania typu malware

Mirosław Skrzewski[1]

Streszczenie: Programy instalujące się w pracujących w sieci komputerach bez wiedzy i zgody użytkownika, dystrybuujące spam, wykradające informacje stają się poważną plagą. Skrytość działania tych programów i ich integracja z systemem pozostawiają często re-instalację systemu jako jedyne skuteczne remedium. W rozdziale przedstawiono opracowaną metodę wykrywania objawów działania w systemie niepożądanego oprogramowania na podstawie analizy aktywności systemu w sieci i omówiono wstępne wyniki testów jej stosowania.

Słowa kluczowe: Malware, wykrywanie działania, analiza ruchu sieciowego.

1. Charakterystyka zagrożeń

Systemy użytkowników połączone z Internetem narażone są na działanie zagrożeń, których rodzaj i ilość ciągle wzrasta. Lista zagrożeń obejmuje rosnącą liczbę różnych programów, takich jak wirusy, robaki internetowe (wormy), programy szpiegowskie, programy dystrybucji reklam, spamu, konie trojańskie. W wielu publikacjach używa się dla ich określenia nazwy malware (złośliwe oprogramowanie). Do ok. 2004 roku, duża jego część miała charakter demonstracyjny, miała zwrócić uwagę świata na zły poziom bezpieczeństwa produktów wypuszczanych przez wielkich producentów oprogramowania. Jako swoisty argument w dyskusji, programy robaków internetowych (wormy) miały jako zadanie zainfekować w jak najkrótszym czasie jak największą liczbę systemów w Internecie, nie prowadząc przy tym działalności destrukcyjnej. Czas wypuszczania nowych wersji wormów wybierany był na koniec tygodnia (np. piątek wieczór), aby w trakcie weekendu infekcja miała jak największy zasięg.

Od blisko trzech lat sytuacja uległa wyraźnej zmianie. Ilość pojawiających się nowych programów nie uległa zmniejszeniu, a według wszelkich oznak wzrosła, natomiast zmieniło się ich zachowanie i cel ich działania. Programy malware zaczęły ukrywać swoją obecność w systemach, natomiast ich działanie nakierowane zostało na przynoszenie wymiernych korzyści ich autorom. Miejsce robaków internetowych zajęły specjalizowane narzędzia do prowadzenia różnych akcji w zainfekowanym systemie, takich jak wykradanie z systemu różnego typu danych i informacji – numery kart kredytowych, hasła dostępu do systemów, informacje o środowisku sieciowym – (programy szpiegowskie *spyware*), programy logujące uderzenia klawiszy (*keylogger*), śledzące ruch w sieci (*sniffery*), czy też udostępniające ich autorom

[1] Instytut Informatyki, Wydział Automatyki, Elektroniki i Informatyki, Politechnika Śląska, Akademicka 16, 44-100 Gliwice
e-mail: mskrzewski@polsl.pl

zainfekowane systemy jako platformy (*zombi*) do prowadzenia zdalnie różnych akcji – dystrybucja spamu, prowadzenia ataków DDoS, czy też programy do wymuszania okupu (*ransomware*) na użytkowniku systemu za udostępnienie klucza do odszyfrowania potajemnie zaszyfrowanych przez program danych.

Intensywność występowania tego typu programów w sieci jest dużą niewiadomą. Raporty firm monitorujących zagrożenia internetowe podają różne liczby odnośnie ich występowania. Raport firmy Webroot Software Inc. (Moll, 2007) ocenia, że 87% amerykańskich komputerów domowych jest zainfekowanych tego typu programami, a średnia ich ilość w systemie sięga 34. W innym raporcie (Fisher, 2007) stwierdza się, że 31% badanych systemów PC posiadało zainstalowany i działający na nich przynajmniej jeden program typu koń trojański, pozwalający na dostęp z zewnątrz do komputera w sposób zupełnie niewidoczny dla użytkownika.

Raporty firm Symantec (2006) i Sophos (2007b) potwierdzają fakt, że głównym celem złośliwego oprogramowania stały się komputery domowe użytkowników, jako wyposażone w słabszą ochronę i często źle skonfigurowane. Z raportu Sophos (2006) wynika także, że dominującym typem nowego złośliwego oprogramowania stają się konie trojańskie, i ich przewaga ilościowa ok. 82% wobec 18% udziału wormów i wirusów w nowo wykrywanym złośliwym oprogramowaniu wzrasta. Głównym kanałem rozprzestrzeniania się niepożądanego oprogramowania pozostaje poczta elektroniczna, rośnie także udział serwisów www (infekcja programem na skutek otworzenia linku do strony, często przesyłanego pocztą elektroniczną), maleje natomiast udział metod dystrybucji typowych dla wormów – samodzielnego rozprzestrzeniania się poprzez sieć, wyszukując i atakując podatne na infekcję systemy.

Przy skrytości działania obecnej generacji złośliwego oprogramowania – braku dużych zmian aktywności w sieci, brak zakłóceń w pracy innych programów, często braku wyraźnych objawów wskazujących na działanie w systemie, bardzo istotnym problemem staje się wykrywanie działania takiego oprogramowania. Istniejące na rynku narzędzia (HKED, 2005; Messmer, 2007) bazują na technologiach wywodzących się z programów antywirusowych, wymagających analizy kopii programu dla opracowania metody jego wykrywania. Przy braku wyraźnych lokalnych objawów działania może być problem z dostarczaniem próbek programów i szybkim opracowywaniem sygnatur dla nowych wersji malware, stąd konieczność opracowania metod wykrywania śladów działania takich programów w systemie bez znajomości szczegółów ich budowy. W dalszej części przedstawiona zostanie propozycja takiego narzędzia działającego na podstawie śladów pozostawianych w obrazie aktywności sieciowej systemu.

2. Możliwości detekcji oprogramowania typu malware

Obecność złośliwego oprogramowania może zostać dość łatwo wykryta w systemie w kilku momentach, w których jego działanie jest widoczne lub gdy korzysta z komunikacji w sieci. Takimi momentami są instalacja w systemie, powstanie zmian w systemie plików, w procedurach startu systemu, próby komunikacji poprzez sieć z autorem, czy też uruchomienie obsługi dodatkowych portów w sieci. Jeżeli znana jest budowa oprogramowania (jego kod), to może zostać ono wykryte w dowolnym

momencie czasu przez programy skanujące na podstawie sygnatury danej wersji programu.

2.1. Wykrywanie w fazie instalacji

Niepożądane oprogramowanie, jak każde zagrożenie z zewnątrz, może przedostać się na komputer użytkownika poprzez jeden z dostępnych kanałów komunikacyjnych. Systemy monitorujące (systemy IDS) mogą wykryć i zarejestrować moment przedostania się tego oprogramowania do systemu, jeżeli będą w stanie rozpoznać to zagrożenie. Często złośliwe oprogramowanie dołączane jest do innych programów, ściąganych z Internetu, może stanowić ono ukryte „wyposażenie" pewnych linków serwisów internetowych lub linków stanowiących element przesyłki mailowej, na otwarcie których (i tym samym do instalacji oprogramowania) użytkownik jest w różny sposób nakłaniany. W większości takich sytuacji powstrzymanie instalacji oprogramowania w systemie jest możliwe tylko wtedy, gdy działający program antywirusowy potrafi rozpoznać intruza.

2.2. Wykrywanie zmian w systemie plików

Obecność w systemie niepożądanego oprogramowania może zostać wykryta, jeżeli w systemie działają programy kontroli integralności systemu plików, np. przy pomocy sum kontrolnych sprawdzające, czy nie pojawiły się zmiany w kodzie programów (inne wartości sum kontrolnych) lub nowe programy, nie mające jeszcze w bazie swoich danych. Inne narzędzia sprawdzają zmiany wpisów w rejestrze sterujące ładowaniem programów przy starcie systemu. Jeżeli narzędzia tego typu nie są zainstalowane w systemie lub są niepoprawnie skonfigurowane, moment instalacji i fakt rozpoczęcia działania takiego programu pozostaje nie zauważony.

2.3. Analiza aktywności systemu w sieci

Komunikacja z otoczeniem sieciowym programów typy wormy realizowana jest na ogół przez te same kanały, przez które oprogramowanie przedostało się do systemu. Jeżeli dane oprogramowanie pojawiło się jako załącznik do przesyłki mailowej, często będzie zawierało serwer pocztowy, rozsyłający kolejne kopie oprogramowania pod różne, odczytane z zasobów systemu lub na generowane mniej lub bardziej losowo, adresy. Programy typu malware na ogół nie otwierają żadnych portów w systemie, i podłączają się do już istniejących lub same nawiązują połączenia do systemów na zewnątrz sieci, za wyjątkiem programów typu RAT (Remote Access Tools) czy koni trojańskich.

Jeżeli oprogramowanie takie pozostaje w stanie uśpienia, oczekując na zdalne polecenia rozpoczęcia aktywności, jego wykrycie będzie możliwe, gdy zdalny kanał komunikacji będzie obsługiwany na nietypowych, wyższych portach systemu, normalnie nieaktywnych, i porty te będą gotowe do użytku. Jeżeli działanie oprogramowania korzysta z aktywnej komunikacji – brak otwartych portów, występowanie wyłącznie jako nadawca, to działanie takiego oprogramowania może zostać wykryte przez rejestrację i analizę połączeń pomiędzy systemami w sieci.

2.4. Monitorowanie i analiza połączeń sieciowych systemu

Monitorowanie połączeń sieciowych systemów może być realizowane jako jedno z zadań sieciowego systemu IDS, w formie rejestracji danych o poszczególnych transmisjach pomiędzy różnymi systemami w sieci, lub też lokalnie na poszczególnych systemach. Przykładem modułu systemu IDS może być linuksowy program *argus* (Bullard, 2004), rejestrujący także ilości danych przesyłane w ramach poszczególnych połączeń pomiędzy różnymi systemami oraz bieżący i końcowy status połączenia. Z narzędzi systemowych do kontroli stanu komunikacji można wykorzystać program *netstat*. W najprostszej wersji monitorowanie można zrealizować rejestrując cyklicznie wyniki działania polecenia *netstat -an*, które listuje otwarte porty danego komputera i aktywne połączenia sieciowe.

Analizując zarejestrowane połączenia można wykryć transmisje, czasem trwające bardzo krótko, które w systemie nie powinny wystąpić lub są trudne do wyjaśnienia, a mogą świadczyć o działaniu nieznanych programów. Rejestracja prowadzona w systemie IDS nie pozwala powiązać wykrytych transmisji z działającymi programami, można natomiast próbować tego dokonać w systemie rejestrującym te dane lokalnie.

W dalszej części rozdziału przedstawione zostaną dostępne w systemie Windows narzędzia umożliwiające rejestrację aktywności sieciowej oraz zaprojektowane do tego celu narzędzia dedykowane. Następnie rozważone zostaną możliwe algorytmy wykrywania podejrzanych transmisji.

3. Narzędzia do monitorowania aktywności sieciowej

Podstawowe informacje o trwających w danej chwili transmisjach oraz o otwartych w systemie portach można uzyskać wywołując działający w trybie tekstowym program *netstat* (rys. 1) z opcjami *-a* (raportuj wszystkie aktywne połączenia i otwarte porty) i *-n* (wyświetlaj adresy numerycznie). W systemie Windows XP dostępna jest opcja *-o* (wyświetlaj dla każdego połączenia związany z nim identyfikator procesu), a w systemie XP z SP2 dodatkowa opcja *-b* pozwalająca administratorowi wylistować dla każdego z aktywnych portów i połączeń nazwę procesu i listę obsługujących dany port lub połączenie programów. Wykonanie polecenia w tej wersji trwa jednak dość długo, kilka sekund.

Polecenie to (rys. 2) podaje, jaki jest aktualny stan komunikacji i kto ją obsługuje w danym momencie czasu. Podobny zakres informacji o procesach komunikacyjnych w systemie dostarcza dodatkowe narzędzie udostępniane przez Microsoft, program *PortReporter*, działający jako usługa w systemie i zapisujący do logu listę wszystkich załadowanych do pamięci programów przy każdej zmianie stanu procesu komunikacyjnego. Lista ta jest bardzo obszerna, obejmuje załadowane do pamięci i związane z danym procesem programy *.exe*, moduły *.dll*, *.ime*, *.drv* i inne. Przykładowy zapis w logu programu związany z próbą otwarcia połączenia *ssh* przedstawia rys. 3. Programy te dostarczają szeregu istotnych informacji o obsłudze komunikacji w sieci, ale ukierunkowane są na diagnostykę problemów z działaniem sieci, a ze względu na powolne działanie, nie gwarantują możliwości rejestracji szybkich zmian stanu połączeń w czasie.

```
Protokół  Adres lokalny      Obcy adres          Stan          PID
TCP    0.0.0.0:21           0.0.0.0:0           NASŁUCHIWANIE 664
TCP    0.0.0.0:135          0.0.0.0:0           NASŁUCHIWANIE 1036
TCP    0.0.0.0:445          0.0.0.0:0           NASŁUCHIWANIE 4
TCP    0.0.0.0:1025         0.0.0.0:0           NASŁUCHIWANIE 664
TCP    0.0.0.0:3389         0.0.0.0:0           NASŁUCHIWANIE 968
TCP    0.0.0.0:5679         0.0.0.0:0           NASŁUCHIWANIE 2228
TCP    127.0.0.1:1028       0.0.0.0:0           NASŁUCHIWANIE 692
TCP    127.0.0.1:1040       0.0.0.0:0           NASŁUCHIWANIE 2068
TCP    192.168.27.74:139    0.0.0.0:0           NASŁUCHIWANIE 4
TCP    192.168.27.74:1048   192.168.27.41:445   USTANOWIONO   4
UDP    0.0.0.0:445          *:*                               4
UDP    0.0.0.0:500          *:*                               800
UDP    0.0.0.0:1030         *:*                               1236
UDP    0.0.0.0:3456         *:*                               664
UDP    0.0.0.0:4500         *:*                               800
UDP    127.0.0.1:123        *:*                               1132
UDP    127.0.0.1:1206       *:*                               3040
UDP    127.0.0.1:1900       *:*                               1400
UDP    192.168.27.74:123    *:*                               1132
UDP    192.168.27.74:137    *:*                               4
UDP    192.168.27.74:138    *:*                               4
UDP    192.168.27.74:1900   *:*                               1400
```

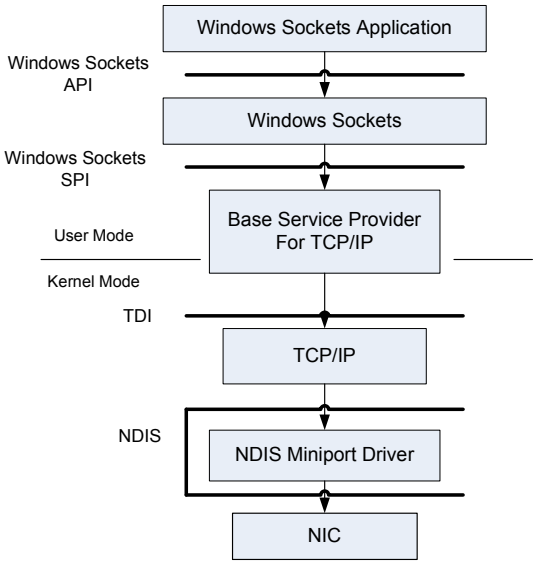
Rysunek 1. Przykładowy wynik wykonania polecenia netstat -ano

```
Protokół  Adres lokalny      Obcy adres          Stan          PID
  TCP    0.0.0.0:21           0.0.0.0:0           NASŁUCHIWANIE 664
  [inetinfo.exe]

  TCP    0.0.0.0:135          0.0.0.0:0           NASŁUCHIWANIE 1036
  c:\windows\system32\WS2_32.dll
  C:\WINDOWS\system32\RPCRT4.dll
  c:\windows\system32\rpcss.dll
  C:\WINDOWS\system32\svchost.exe
  C:\WINDOWS\system32\ADVAPI32.dll
  [svchost.exe]

  TCP    0.0.0.0:3389         0.0.0.0:0           NASŁUCHIWANIE 968
  -- nieznane składniki --
  c:\windows\system32\rpcss.dll
  C:\WINDOWS\system32\svchost.exe
  C:\WINDOWS\system32\ADVAPI32.dll
  [svchost.exe]

 UDP    0.0.0.0:3456         *:*                               664
  [inetinfo.exe]
```

Rysunek 2. Przykładowy wynik wykonania polecenia netstat -abn

```
Log number: 632
Log entry below recorded at: 07/3/16,19:47:47
================================================================
Process ID: 3000 (putty.exe)
User context: PC300\baca
Process doesn't appear to be a service
PID     Port          Local IP          State            Remote IP:
    Port
3000    TCP 2168  192.168.27.13      SYN SENT
    192.168.27.19:22
Port Statistics
TCP mappings: 1
UDP mappings: 0
TCP ports in a SYN SENT state:          1 = 100.00%
Loaded modules:
D:\Program Files\Tools\putty.exe (0x00400000)
C:\WINDOWS\system32\ntdll.dll (0x7C900000)
C:\WINDOWS\system32\kernel32.dll (0x7C800000)
C:\WINDOWS\system32\ADVAPI32.dll (0x77DC0000)
C:\WINDOWS\system32\RPCRT4.dll (0x77E70000)
C:\WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls_6595b64 144
    ccf1df_6.0.2600.2982_x-ww_ac3f9c03\COMCTL32.dll (0x773C0000)
C:\WINDOWS\system32\msvcrt.dll (0x77C00000)
C:\WINDOWS\system32\GDI32.dll (0x77F10000)
C:\WINDOWS\system32\USER32.dll (0x77D30000)
C:\WINDOWS\system32\SHLWAPI.dll (0x77F60000)
C:\WINDOWS\system32\comdlg32.dll (0x76380000)
C:\WINDOWS\system32\SHELL32.dll (0x7C9C0000)
C:\WINDOWS\system32\IMM32.dll (0x76360000)
C:\WINDOWS\system32\WINMM.dll (0x76B20000)
C:\WINDOWS\system32\WINSPOOL.DRV (0x72F90000)
C:\WINDOWS\system32\WSOCK32.dll (0x71A70000)
C:\WINDOWS\system32\WS2_32.dll (0x71A50000)
C:\WINDOWS\system32\WS2HELP.dll (0x71A40000)
C:\WINDOWS\system32\uxtheme.dll (0x5B1D0000)
C:\WINDOWS\system32\PROCHLP.DLL (0x10000000)
C:\WINDOWS\system32\MSCTF.dll (0x746D0000)
C:\WINDOWS\system32\SynTPFcs.dll (0x63000000)
C:\WINDOWS\system32\VERSION.dll (0x77BF0000)
C:\WINDOWS\system32\msctfime.ime (0x75180000)
C:\WINDOWS\system32\ole32.dll (0x774D0000)
C:\WINDOWS\system32\Amhooker.dll (0x00FF0000)
C:\WINDOWS\system32\mswsock.dll (0x719F0000)
C:\WINDOWS\system32\hnetcfg.dll (0x66780000)
C:\WINDOWS\System32\wshtcpip.dll (0x71A30000)
```

Rysunek 3. Log programu PortReporter dla zdarzenia próba otwarcia połączenia TCP na port 22



Rysunek 4. Architektura interfejsów protokołów komunikacyjnych systemu Windows

4. Monitor aktywności sieciowej *TDILog*

Do wykrywania śladów obecności niepożądanego oprogramowania w systemie na podstawie ruchu sieciowego potrzebne jest zupełnie inne narzędzie. Powinno ono dokładnie rejestrować momenty występowania wszystkich zmian stanu połączeń sieciowych w systemie i związane z nimi procesy, robić to w sposób umożliwiający jego ciągłą pracę bez obawy wyczerpania zasobów pamięciowych systemu oraz dostarczać zbiorczych informacji o ilości danych wymienianych w ramach poszczególnych połączeń.

Dla zapewnienia minimalnego obciążenia rejestracją systemu, powinno ono nie korzystać z mechanizmu ciągłego przepytywania o stan wybranych interfejsów programowych systemu lecz przechwytywać zdarzenia generowane przy zmianach stanu połączeń w systemie. Na rys. 4 przedstawiono architekturę interfejsów programowych związanych z działaniem stosu protokołów komunikacyjnych systemu Windows (Szromek, 2007). Z interfejsów poziomu jądra systemu (*kernel mode*) do rejestracji ruchu najlepiej nadaje się interfejs NDIS, przez który przechodzi całość ruchu sieciowego we wszystkich obsługiwanych przez system protokołach.

Zdarzenia generowane na tym poziomie nie zawierają jednak informacji o programach, z/do których ten ruch jest kierowany, ani też informacji o adresach wewnętrznych protokołów wyższych warstw, zależnych od stosu komunikacyjnego (portach dla protokołów UDP/TCP, gniazdach IPX/SPX czy nazwach NetBEUI). Informacje te można uzyskać na interfejsie warstwy transportowej TDI (*Transport Driver Interface*), choć ograniczone są tylko do protokołów transportowych – nie obejmują protokołów pomocniczych np. protokołu ICMP czy ARP.

```
00037  09:18:04:484  LISTEN  UDP      0.0.0.0;1028      408  C:\
    WINDOWS\system32\svchost.exe  {ZARZĄDZANIE NT\USŁUGA SIECIOWA}
00038  09:18:04:718  CONNECT  TCP  OUT  192.168.27.13;1029
    212.72.49.131;80  2760  C:\Program Files\Skype\Phone\Skype.exe
    { PC300\baca}
00039  09:18:04:796  CLOSED  TCP  *  192.168.27.13;1029
    212.72.49.131;80  175/303  {PC300\baca}
00040  09:18:05:375  PROCESS  596
C:\WINDOWS\system32\svchost.exe  {ZARZĄDZANIE NT\USŁUGA LOKALNA}
```

Rysunek 5. Przykłady wpisów w logu programu *TDILog*

Do rejestracji zdarzeń wybrano ostatecznie interfejs TDI, i w ramach pracy dyplomowej (Szromek, 2007) opracowany został działający zgodnie z tymi założeniami program *TDILog* monitorujący aktywność sieciową systemu opartą na protokołach TCP i UDP, działający jako usługa systemowa i rejestrujący czas zachodzących na interfejsie TDI zdarzeń z dokładnością do pojedynczych milisekund, z dobową rotacją plików logu. Log programu dostarcza informacji o tworzonych połączeniach, podaje wielkości ruchu przesyłanego w danym połączeniu, proces obsługujący dane połączenie i kontekst użytkownika procesu, a dla otwartych portów procesy odpowiedzialne za ich obsługę.

Na rys. 5 przedstawiono fragment logu programu *TDILog* pokazujący sposób rejestracji aktywności systemu w sieci. Każdy wiersz opisuje jedno zdarzenie. Kolejno umieszczone są numer zdarzenia, czas wystąpienia, opis zdarzenia (listen – otwarcie gniazda, connect – nawiązanie połączenia, closed – zakończenie połączenia, process – wykrycie procesu komunikującego się przez sieć, timeout, reset – nieudane połączenie), protokół, kierunek komunikacji (in – połączenie z zewnątrz, out – nawiązanie połączenia z zewnętrznym systemem, * – dwukierunkowy przesył danych), adres IP i port, lokalny, zdalny, identyfikator procesu (przy closed – liczba bajtów wysłanych i odebranych na zamykanym porcie), nazwa (pełna) programu obsługującego port oraz kontekst działania programu obsługi.

5. Wykrywanie śladów działania - algorytm

Logi programu *TDILog* zawierają chronologiczny zapis wszystkich operacji związanych z komunikacją systemu w sieci. Większość wpisów jest wynikiem akcji podejmowanych przez użytkownika lub działania algorytmów realizowanych przez system operacyjny lub programy aplikacyjne. Dla wykrycia śladów obecności programów typu malware należy określić dokładny model zachowania się takich programów w sieci i szukać w logach zapisów pasujących do przyjętego modelu. W pracach Skrzewski (2004 i 2005) analizowano formy komunikacji sieciowej występujące w trakcie działania różnego rodzaju zagrożeń. Opierając się na przedstawionych tam rozważaniach należy poszukiwać w logach następujących objawów aktywności programów malware w systemie:

- Występowanie obsługi dodatkowych portów TCP lub UDP

- Występowanie połączeń z zewnątrz na otwarte dodatkowe porty
- Występowanie połączeń o niewielkiej aktywności otwieranych na zewnątrz sieci lokalnej do nieznanych systemów w nietypowych porach (poza godzinami pracy użytkowników) lub nie związanych z działalnością użytkowników w sieci.

Ponieważ większość systemów w sieci lokalnej znajduje się pod osłoną mechanizmów filtracji pakietów, blokujących możliwość komunikacji z zewnątrz na nietypowe porty, głównym źródłem informacji o działających w systemie programach będą zapisy o tworzonych przez system połączeniach i ich adresatach. Ich analizę należy podzielić na kilka etapów:

- tworzymy zestawienie otwieranych z danego systemu połączeń do innych systemów, przy ustalonym adresie IP nadawcy (monitorowany system) dla każdej pary numer portu przeznaczenia, adres przeznaczenia zliczamy ilość utworzonych połączeń (zdarzenie connect)
- dla połączeń z serwisami www (port przeznaczenia 80 lub 443) usuwamy z listy połączenia tworzone w czasie trwania innych połączeń lub obejmujące inne połączenia (znajdujące się w logu pomiędzy zdarzeniami connect a closed dla określonego numeru portu nadawcy), są one związane z odczytem elementów odczytywanych stron internetowych.
- dla pozostałych na liście adresów przeznaczenia sprawdzamy nazwy domenowe (dns) systemów, nazwy domenowe związane z podsieciami IP, dla określenia systemów związanych z serwisami aktualizacji zainstalowanego w systemie oprogramowania.
- pozostałe na liście połączenia z zewnętrznymi systemami mogą świadczyć o działaniu w systemie nieautoryzowanego oprogramowania.

W celu finalnej weryfikacji otrzymanych danych można spróbować połączyć się do danego portu w zewnętrznym systemie i sprawdzić odpowiedź, przy jej braku można sprawdzić trasę przez sieć do danego systemu, np. poleceniem traceroute (tracert). Podobną analizę można przeprowadzić dla połączeń otwieranych do monitorowanego systemu, tworząc zestawienie adresów systemów otwierających połączenia do dowolnych portów. Z zestawienia można usunąć połączenia na porty charakterystyczne dla komunikacji w otoczeniu sieciowym, np. dla systemu Windows do portów 139, 445, 3389, jeśli połączenia otwierane są przez systemy z tej samej podsieci.

6. Badania aktywności systemów

Dla weryfikacji skuteczności proponowanego algorytmu, program *TDILog* zainstalowany został na kilku systemach działających w laboratorium w trybie 24/7 (24 godziny, 7 dni w tygodniu) oraz na wybranych komputerach użytkowników. Badane systemy laboratoryjne podłączono do jednego hub-a, do którego podłączone był

Tabela 1. Zestawienie podejrzanych połączeń do adresów zewnętrznych IP

Data	16.04		17.04		18.04		19.04		21.04		24.04	
Adres IP \ Porty	80	443	80	443	80	443	80	443	80	443	80	443
150.254.186.69							1					
150.254.186.70			1		2				1		1	
207.46.209.126			1	1							1	1
213.155.151.113	1											
62.146.68.11					1							
64.4.21.189					1	1						
64.4.52.189											1	
66.150.14.20	2		2		2		2		2		2	

```
GET / HTTP/1.1
Accept: image/gif, ..., application/vnd.ms-excel, ..., */*
Host: 213.155.151.113     Connection: Keep-Alive

HTTP/1.0 400 Bad Request
Server: AkamaiGHost ....
Expires: Thu, 1 May 2007 15:44:40 GMT
Date: Thu, 1 May 2007 15:44:40 GMT
Connection: close
<HTML><HEAD>
<TITLE>Invalid URL</TITLE></HEAD><BODY>
<H1>Invalid URL</H1>
The requested URL "&#47;", is invalid.<p>
Reference&#32;&#35;9&#46;6d979bd5&#46;1179416680&#46;0
</BODY></HTML>
```

Rysunek 6. Połączenie http z systemem 213.155.151.113, z serwerem AkamaiGhost

także komputer z działającym programem argus, rejestrującym dodatkowo początkowe 196 bajtów każdego pakietu do pliku. Dobowe logi programu *TDILog* mają różną wielkość, zależnie od funkcji systemu w sieci. Na włączonym, nie używanym systemie Windows XP mają wielkość ok. 20-30 kB, na używanym systemie od 200 do 700 i więcej kB, zależnie od aktywności użytkownika, na serwerze w domenie sięgają kilku MB. Dla wybranego komputera przeanalizowano dane obejmujące okres kilku dni pracy, pozostałe po analizie zestawienie połączeń przedstawia tabela 1.

Występujące w tablicy adresy IP w większości nie mają przypisanych nazw domenowych, przy próbie otwarcia strony w przeglądarce internetowej najczęściej pojawia się kod błędu lub pusta strona. Przy adresach 207.46.209.126 i 64.4.21.189 otworzyła się strona Microsoft Windows Update. W celu wyjaśnienia, co kryje się za pozostałymi adresami, skorzystano z danych zbieranych przez program *argus*, i zarejestrowane początki ramek związane z komunikacją z podanymi w tabeli adresami przeanalizowano analizatorem protokołów ethereal. Przebieg dialogu http z wybranymi stacjami przedstawiają rys. 6 i 7.

Analizując zawartość wysyłanych ramek transmisji z rys. 7 widać ewidentnie,

```
13:14:40.094828 157.158.57.74 -> 66.150.14.20  TCP [TCP segment of a
    reassembled PDU]
TCP, Src Port: 1412 (1412), Dst Port: http (80), Seq: 1, Ack: 1, Len: 4   '
    POST'

13:14:40.095397 157.158.57.74 -> 66.150.14.20  HTTP  Continuation or non-
    HTTP traffic
TCP, Src Port: 1412 (1412), Dst Port: http (80), Seq: 5, Ack: 1, Len: 1460
Hypertext Transfer Protocol, Data (238 bytes)
0000 /index.php HTTP  /1.1..Accept-Enc oding: gzip..Con tent-Type: appli
0040 cation/x-www-for m-urlencoded..Co ntent-Length: 13  32..Host: public
0080 .windupdates.com  ....data=436C84A  E4139B9F9EBADFB6  9AE8467A41F51F50
00c0 964D643A3026CB2A  2DDB02837F96E3E2  4B13D303A363E9
[Packet size limited during capture: HTTP truncated]

13:14:40.095405 157.158.57.74 -> 66.150.14.20  HTTP  Continuation or non-
    HTTP traffic
TCP, Src Port: 1412 (1412), Dst Port: http (80), Seq: 1465, Ack: 1, Len:
    20
Hypertext Transfer Protocol  Data (20 bytes)
0000  A4A9AEBB267F3C6F  F3B3
```

Rysunek 7. Transmisja http do systemu 66.150.14.20, rozbita na 3 pakiety

że jest to przesył jakichś informacji z komputera w świat, raczej w zakodowanej (zaszyfrowanej?) formie. W transmisji z rys. 6 można dopatrywać się przekazywania informacji w drugim kierunku, przez niewidoczny na ekranie przeglądarki ciąg znaków w wierszu Reference.

7. Uwagi końcowe

Przedstawione przykłady pokazują, że proponowany behawioralny algorytm wykrywania obecności ukrytego w systemie oprogramowania oparty o analizę rejestrowanych śladów komunikacji systemu w sieci pozwala na wyselekcjonowanie podejrzanych przypadków transmisji. Dla jego pełnej funkcjonalności należy zautomatyzować proces analizy logów tworzonych przez program *TDILog*. Ostateczne zlokalizowanie niepożądanego programu w systemie będzie wymagało skorzystania z innych narzędzi, np. z firmowego programu Microsoftu *PortReporter*, listującego wszystkie moduły programowe składające się na realizację procesu obsługującego daną fazę komunikacji.

Literatura

Bullard, C. (2004) *Argus - audit record generation / utilization system.* `http://www.die.net/doc/linux/man/man8/argus.8.html`.

Fisher, D. (2007) *Spyware war may be a losing battle, experts say.* `http://searchwindowssecurity.techtaget.com/originalContent/0,289142,sid45_gci1209110,00.html?bucket=NEWS&topic=299913`.

HKED (2005) *A brief history of Malware Detection.* `http://www.cybertechhelp.com/tutorial/article/history-of-malware-detection`.

Messmer, M. (2007) *New approaches to malware detection coming into view.* Network World, 04/25/07.

Moll, D. (2007) *State of spyware, Q2 2006.* Webroot Software, Inc. http://h30307.www3.hp.com /pdf/SOS_Q206_USA.pdf.

Skrzewski, M. (2004) *Zagrożenie prywatności w technologii www.* Konferencja *Internet w Społeczeństwie Informacyjnym*, Zakopane, 2004.

Skrzewski, M. (2005) *Algorytmy wykrywania zagrożeń. Nowe Technologie w Komputerowych Systemach Zarządzania*, WkiŁ, Warszawa, 2005.

Sophos (2006) *Sophos Security Threat Management Report.* Update July 2006, http://www.sophos.com/ sophos/docs/eng/papers/sophos-security-report-jun06-srus.pdf.

Sophos (2007a) *New Sophos security report reveals United States is worst for malware hosting and spam-relaying.* http://www.sophos.com/pressoffice/news/articles/2007/01/secrep2007.html.

Sophos (2007b) *Sophos Security threat report 2007.* http://www.sophos.com/sophos/docs/eng marketing_material/sophos-security-threats-2007_wsrus.pdf.

Symantec (2006) *Symantec Report Demonstrates that Cyber Attacks Increasingly Target Home Users for Financial Gain*, http://www.symantec.com/about/news/release/article.jsp?prid=20060925_.

Symantec (2007) *Vulnerabilities in desktop applications and use of stealth techniques are on the rise.* http://www.symantec.com/about/news/release/article.jsp?prid=20060925_02.

Szromek, J. (2007) *Program do analizy połączeń sieciowych komputera.* praca dyplomowa magisterska, kierunek Informatyka, Wydział Automatyki, Elektroniki i Informatyki, Gliwice, 2007.

Behavioral detection of the running malware programs

Malware programs, installing themselves on user systems without the user intent or knowledge and turning them into spam distribution or zombie platform, grown recently to the most important Internet threat. The paper presents details of the new method of discovery of the malware programs activity on the user systems on the basis of network system activity records, generated by the *TDILog* tool. Preliminary algorithm of TDILog's log analysis and malware presence detection are also presented.