

Know Your Enemy: Worms at War

The Not so Friendly World of Cyberspace

Honeynet Project

<http://www.honeynet.org>

Last Modified: 9 November, 2000

This paper was born out of pure curiosity. Our Honeynet was being pounded with UDP port 137 and TCP port 139 scans. The network was getting scanned 5-10 times a day on these ports, something was up. The goal was to learn what these scans were all about. What was out in the Internet causing all of this activity? Based on the ports, we assumed that the scans were looking for Window's based vulnerabilities. The plan was to setup a Win98 honeypot, sit back and wait. We didn't have to wait long. A continuation of the [Know Your Enemy](#) series.

The Setup

During a one month period (20 Sep - 20 Oct) we confirmed [524 unique NetBIOS scans](#) on our Honeynet network. These scans consisted of UDP port 137 (NetBIOS Naming Service) probes, usually followed by TCP port 139 (NetBIOS Session Service). That is large number of scans probing for a specific service, something was up, we decided to find out what. The Honeynet network does not advertises itself to the Internet. We just put the systems on our network and sit back and wait. That means that the majority of scans we receive are random scans that are most likely probing most of the Internet. These are the same threats your systems face. As these scans are probing Windows based systems, they are most likely focusing on the common homeowner with a DSL or Cable connection to their house. We are not talking about corporate espionage or web defacing, we are talking about simple homeowners as the target here. We were curious, who was doing these scans, what was their purpose, and why the vast number of scans? Was this a coordinated effort, were these worms? Lots of questions. So, we decided to find out and added a Windows98 honeypot to our collection. We did a default installation of Windows98 and enabled sharing of the C: drive. A Windows98 honeypot may not sound glamorous, however there are several things to be gained from setting up such a system.

1. Windows98 represent a huge number of systems connected to the Internet, and this number is growing fast. Typically, these are the systems with the weakest security, as homeowners are the ones using these systems. What people do not realize is the risk these systems are exposed to, as many of them have dedicated connections to the Internet.
2. This was our first crack at a Microsoft based honeypot. The plan was to start off simple and learn

from there.

On October 31, the system was installed, sharing was enabled, and connected to the Internet. We then sat back and waited, the wait was not long.

The First Worm

Less than 24 hours later we received our first visitor. The system 216.191.92.10 (host-010.hsf.on.ca) scanned the network looking for Window systems. It found ours and began querying it. It first began by getting the system name and determine if sharing was enabled. Once it determined that sharing was enabled, it then probed for specific binaries on our system. Its goal was to determine if a specific worm was installed, and if not, then it would install itself. In this case, the specific worm was not installed. The worm is known as the "[Win32.Bymer Worm](#)". The purpose of this worm is to take advantage of your CPU cycles to help an individual win the [distributed.net](#) contest. Distributed.net is group that uses the idle process of distributed computers for various challenges (such as cracking [RC5-64 challenge](#)). People are awarded prizes if they crack the challenge. The more computers and CPU cycles you have under your control, the better of your chances of winning. In our case, someone "volunteered" us for the project by installing the worm on our system.

An individual (in this case, bymer@inec.kiev.ua), created a self replicating worm that would find vulnerable Window systems and install the distributed.net client on unsuspecting systems. Once installed and executed, the worm utilizes your CPU cycles in attempt to help the author win the contest. Meanwhile the worm begins probing for other vulnerable systems it can take over. The goal is to have access to as many computers and CPU cycles as possible. This process grows exponentially as more systems are compromised. Lets take a look at the attack using packet captures of the network traffic (in this case, we used the [IDS sniffer snort](#)). For more advanced analysis of the NetBIOS protocol, you may want to use a protocol analyzer, such as the free utility [Ethereal](#). Throughout the sniffer traces below, the system 172.16.1.105 is the IP address of the honeypot.

The worm begins by first checking to see if the file *dnetc.ini* is on the system. This is the standard configuration file for the distributed.net client. This configuration file tells the main server who should get credit for all the CPU cycles. This is also the person that most likely created the worm. Here we see the packet trace where the remote system (NetBIOS name GHUNT, account GHUNT, domain HSFOPROV) copies the configuration file to our honeypot.

```
11/01-15:29:18.580895 216.191.92.10:2900 -> 172.16.1.105:139
TCP TTL:112 TOS:0x0 ID:50235 DF
*****PA* Seq: 0x12930C6 Ack: 0x66B7068 Win: 0x2185
00 00 00 5B FF 53 4D 42 2D 00 00 00 00 01 00 ... [.SMB-.....
00 00 00 00 00 00 00 00 00 00 00 00 00 C8 57 1C .....W.
00 00 82 D1 0F FF 00 00 00 07 00 91 00 16 00 20 .....
00 DC 1C 00 3A 10 00 00 00 00 00 00 00 00 00 00 .....:.....
```

```
00 00 00 1A 00 5C 57 49 4E 44 4F 57 53 5C 53 59 ..... \WINDOWS\SY
53 54 45 4D 5C 64 6E 65 74 63 2E 69 6E 69 00 STEM\dnetc.ini.
```

Below we see the actual file transfer of the configuration file *dnetc.ini*. Notice who is the point of contact for this, *bymer@inec.kiev.ua*. This is the individual that receives the credit for the CPU cycles, and most likely the author of the worm attacking us.

```
11/01-15:29:18.729337 216.191.92.10:2900 -> 172.16.1.105:139
TCP TTL:112 TOS:0x0 ID:50747 DF
*****PA* Seq: 0x1293125 Ack: 0x66B70AD Win: 0x2140
00 00 01 11 FF 53 4D 42 0B 00 00 00 00 01 00 .....SMB.....
00 00 00 00 00 00 00 00 00 00 00 00 00 00 C8 57 1C .....W.
00 00 02 D2 05 00 00 E1 00 00 00 00 00 E1 00 E4 .....
00 01 E1 00 5B 6D 69 73 63 5D 20 0D 0A 70 72 6F ....[misc] ..pro
6A 65 63 74 2D 70 72 69 6F 72 69 74 79 3D 4F 47 ject-priority=OG
52 2C 52 43 35 2C 43 53 43 2C 44 45 53 0D 0A 0D R,RC5,CSC,DES...
0A 5B 70 61 72 61 6D 65 74 65 72 73 5D 0D 0A 69 .[parameters]..i
64 3D 62 79 6D 65 72 40 69 6E 65 63 2E 6B 69 65 d=bymer@inec.kie
76 2E 75 61 0D 0A 0D 0A 5B 72 63 35 5D 0D 0A 66 v.ua....[rc5]..f
65 74 63 68 2D 77 6F 72 6B 75 6E 69 74 2D 74 68 etch-workunit-th
72 65 73 68 6F 6C 64 3D 36 34 0D 0A 72 61 6E 64 reshould=64..rand
6F 6D 70 72 65 66 69 78 3D 32 31 37 0D 0A 0D 0A omprefix=217....
5B 6F 67 72 5D 0D 0A 66 65 74 63 68 2D 77 6F 72 [ogr]..fetch-wor
6B 75 6E 69 74 2D 74 68 72 65 73 68 6F 6C 64 3D kunit-threshold=
31 36 0D 0A 0D 0A 5B 74 72 69 67 67 65 72 73 5D 16....[triggers]
0D 0A 72 65 73 74 61 72 74 2D 6F 6E 2D 63 6F 6E ..restart-on-con
66 69 67 2D 66 69 6C 65 2D 63 68 61 6E 67 65 3D fig-file-change=
79 65 73 0D 0A yes..
```

The next file to be transferred is the actual distributed.net client, *dnetc.exe*. This is a valid executable, it is not malicious in nature. We confirmed this by taking an MD5 signature of the client found on the honeypot. We then downloaded the client from distributed.net and took an MD5 hash of the dnetc.exe client. The MD5 hashes were identical (d0fd1f93913af70178bff1a1953f5f7d), indicating that this code is not the worm. This is the binary that uses your CPU cycles as part of the distributed.net challenge. However, the worm intends on using this binary without your permission nor knowledge, all for the author's gain.

```
11/01-15:34:09.044822 216.191.92.10:2900 -> 172.16.1.105:139
TCP TTL:112 TOS:0x0 ID:33084 DF
*****PA* Seq: 0x129341A Ack: 0x66B71C0 Win: 0x202D
00 00 00 5B FF 53 4D 42 2D 00 00 00 00 01 00 ...[.SMB-.....
00 00 00 00 00 00 00 00 00 00 00 00 00 00 C8 57 1C .....W.
```

```

00 00 04 26 0F FF 00 00 00 07 00 91 00 16 00 20  ...&.....
00 FE 1D 00 3A 10 00 00 00 00 00 00 00 00 00 00  ....:.....
00 00 00 1A 00 5C 57 49 4E 44 4F 57 53 5C 53 59  ....\WINDOWS\SY
53 54 45 4D 5C 64 6E 65 74 63 2E 65 78 65 00      STEM\dnetc.exe.

```

Next we see the actual worm being transferred, *msi216.exe*. This is the self-replicating worm that randomly probes for vulnerable systems and copies itself. This is the worm that is most likely causing a great number of the scans we are receiving.

```

11/01-15:37:23.083643 216.191.92.10:2900 -> 172.16.1.105:139
TCP TTL:112 TOS:0x0 ID:40765 DF
*****PA* Seq: 0x12C146A Ack: 0x66C248B Win: 0x20B2
00 00 00 5C FF 53 4D 42 2D 00 00 00 00 00 01 00  ...\.SMB-.....
00 00 00 00 00 00 00 00 00 00 00 00 00 00 C8 57 1C  ....W.
00 00 02 F3 0F FF 00 00 00 07 00 91 00 16 00 20  ....
00 C0 1E 00 3A 10 00 00 00 00 00 00 00 00 00 00  ....:.....
00 00 00 1B 00 5C 57 49 4E 44 4F 57 53 5C 53 59  ....\WINDOWS\SY
53 54 45 4D 5C 6D 73 69 32 31 36 2E 65 78 65 00      STEM\msi216.exe.

```

Last, the worm first modifies then uploads a new *win.ini* file. The worm does this so the system will execute the worm upon reboot. Remember, it can be difficult to remotely execute a file on a Win98 system, so this is the worm's method of getting it executed. It does this by adding itself to the boot up configuration file *c:\windows\win.ini* and has itself loaded during the boot process. The new *win.ini* file is then uploaded to our compromised system.

```

11/01-15:38:55.352810 216.191.92.10:2900 -> 172.16.1.105:139
TCP TTL:112 TOS:0x0 ID:1342 DF
*****A* Seq: 0x12C6F55 Ack: 0x66C95FC Win: 0x1FBFB
00 00 0B 68 FF 53 4D 42 1D 00 00 00 00 00 01 00  ...h.SMB.....
00 00 00 00 00 00 00 00 00 00 00 00 00 00 C8 57 1C  ....W.
00 00 02 F9 0C 0D 00 61 19 00 00 00 00 00 00 00  ....a.....
00 00 00 00 00 00 00 00 00 2C 0B 3C 00 2D 0B 00  ....,.<.-..
5B 77 69 6E 64 6F 77 73 5D 0D 0A 6C 6F 61 64 3D  [windows]..load=
63 3A 5C 77 69 6E 64 6F 77 73 5C 73 79 73 74 65  c:\windows\sysste
6D 5C 6D 73 69 32 31 36 2E 65 78 65 0D 0A 72 75  m\msi216.exe..ru
6E 3D 0D 0A 4E 75 6C 6C 50 6F 72 74 3D 4E 6F 6E  n=..NullPort=Non
65 0D 0A 0D 0A 5B 44 65 73 6B 74 6F 70 5D 0D 0A  e....[Desktop]..
57 61 6C 6C 70 61 70 65 72 3D 28 4E 6F 6E 65 29  Wallpaper=(None)
0D 0A 54 69 6C 65 57 61 6C 6C 70 61 70 65 72 3D  ..TileWallpaper=
31 0D 0A 57 61 6C 6C 70 61 70 65 72 53 74 79 6C  1..WallpaperStyl
65 3D 30 0D 0A 0D 0A 5B 69 6E 74 6C 5D 0D 0A 69  e=0....[intl]..i

```

That's it. The Worm is now complete and the honeypot has now been infected. All that needs to happen now is the system to reboot and the Worm will take effect. Once it takes effect, several things happen.

1. The distributed.net client begins, using the CPU cycles in the contest.
2. The Worm begins searching for other vulnerable systems to replicate itself to. This is what is causing all the UDP 137 and TCP 139 scans.
3. The worm may add the following keys to the registry.

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run
\Bymer.scanner HKEY_LOCAL_MACHINE\Software\Microsoft\Windows
\CurrentVersion\RunServices\Bymer.scanner
```

One may think that having to wait for a system to reboot is an unreliable way to execute. Keep in mind, the targets are Windows desktop systems. How often do you reboot your Windows desktop?

The Second Worm

It is a busy week, and our second worm comes the next day. This worm is a similar variation to the first worm, its purpose is to gain control of your CPU cycles to help an individual in the distributed.net contest. The only difference with this worm is that all the files are combined in one single executable, *wininit.exe*. Default installations of Windows98 already have a binary *c:\windows\wininit.exe* installed on their system. This worm calls itself the same in an attempt to obscure itself, but installs itself in *c:\windows\system\wininit.exe*. If anyone should happen to stumble across the binary, the author hopes they will assume it is part of the operating system and not a worm. This is a very common tactic amongst the blackhat community. Once executed, the worm acts just as the previous worm does. Below we see our honeypot being infected with the second worm, *wininit.exe*. The remote systems has the NetBIOS name WINDOW, account WINDOW, domain LVCW.

```
11/02-21:41:17.287743 216.234.204.69:2021 -> 172.16.1.105:139
TCP TTL:113 TOS:0x0 ID:38619 DF
*****PA* Seq: 0x21CC0AC Ack: 0xCE6736B Win: 0x2185
00 00 00 5D FF 53 4D 42 2D 00 00 00 00 01 00 ...].SMB-.....
00 00 00 00 00 00 00 00 00 00 00 00 00 00 D0 4F 1F .....O.
00 00 84 EE 0F FF 00 00 00 07 00 91 00 16 00 20 .....
00 20 BB 01 3A 10 00 00 00 00 00 00 00 00 00 00 00 . . .:.....
00 00 00 1C 00 5C 57 49 4E 44 4F 57 53 5C 53 59 ..... \WINDOWS\SY
53 54 45 4D 5C 77 69 6E 69 6E 69 74 2E 65 78 65 STEM\wininit.exe
00
```

Once the worm has installed itself, the remote system then modifies the win.ini file to ensure it is executed on reboot. Notice how this executable adds to the already modified *c:\windows\win.ini* file, which has an entry from our previous worm.

```

11/02-21:41:48.538643 216.234.204.69:2021 -> 172.16.1.105:139
TCP TTL:113 TOS:0x0 ID:21212 DF
*****A* Seq: 0x22021C9 Ack: 0xCE68EC7 Win: 0x1FA3
00 00 0B 68 FF 53 4D 42 1D 00 00 00 00 00 01 00 ...h.SMB.....
00 00 00 00 00 00 00 00 00 00 00 00 00 00 D0 4F 1F .....O.
00 00 84 F4 0C 0F 00 7F 19 00 00 00 00 00 00 00 .....
00 00 00 00 00 00 00 00 00 00 2C 0B 3C 00 2D 0B 00 .....<.-..
5B 77 69 6E 64 6F 77 73 5D 0D 0A 6C 6F 61 64 3D [windows]..load=
63 3A 5C 77 69 6E 64 6F 77 73 5C 73 79 73 74 65 c:\windows\system
6D 5C 77 69 6E 69 6E 69 74 2E 65 78 65 20 63 3A m\wininit.exe c:
5C 77 69 6E 64 6F 77 73 5C 73 79 73 74 65 6D 5C \windows\system\
6D 73 69 32 31 36 2E 65 78 65 0D 0A 72 75 6E 3D msi216.exe..run=
0D 0A 4E 75 6C 6C 50 6F 72 74 3D 4E 6F 6E 65 0D ..NullPort=None.
0A 0D 0A 5B 44 65 73 6B 74 6F 70 5D 0D 0A 57 61 ...[Desktop]..Wa

```

Upon reboot, this worm, like the previous one, will start up and begin the same processes. The thing to keep in mind is that the remote systems attacking us are most likely not evil blackhats out to own the world. More likely the remote systems are innocent bystanders who were compromised. The owners have no idea there is a worm running on their system, nor any idea their computers are being used to scan for and exploit other vulnerable systems on the Internet. However, their systems have dedicated connections to the Internet, making them primary targets. Even systems that dial-up to the Internet are at risk for such attacks. There is a 'war' going on as automated worms seek out and compromise other systems. They then use these systems as launching points to gain control of other systems, such as our honeypot.

The Day After

The next day, other variations of the same worm probed our honeypot. They first determine if sharing is enabled, and if so, they check if the same version of the worm is already installed. In both cases for this day, the worm was already installed, so the remote systems left us alone. The first remote system checked to see if the *wininit.exe* worm was installed. Later on that day, another system checked to see if the worm *msi216.exe* was installed.

```

11/03-04:42:11.596636 210.111.145.180:2341 -> 172.16.1.105:139
TCP TTL:115 TOS:0x0 ID:12574 DF
*****PA* Seq: 0x2345C04 Ack: 0xE65CC94 Win: 0x2171
00 00 00 5D FF 53 4D 42 2D 00 00 00 00 00 01 00 ...].SMB-.....
00 00 00 00 00 00 00 00 00 00 00 00 00 00 D8 B5 1D .....
00 00 81 3E 0F FF 00 00 00 07 00 91 00 16 00 20 ...>.....
00 3A 26 02 3A 10 00 00 00 00 00 00 00 00 00 00 ..:&:.....
00 00 00 1C 00 5C 57 49 4E 44 4F 57 53 5C 53 59 ..... \WINDOWS\SY
53 54 45 4D 5C 77 69 6E 69 6E 69 74 2E 65 78 65 STEM\wininit.exe
00 .

```

Remote system, NetBIOS name MATTHEW, account MPYLE, domain MPYLE

```
11/03-16:39:38.723572 216.23.6.24:3946 -> 172.16.1.105:139
TCP TTL:113 TOS:0x0 ID:3309 DF
*****PA* Seq: 0x1A7105F Ack: 0x10F8C0F2 Win: 0x2159
00 00 00 5B FF 53 4D 42 2D 00 00 00 00 00 01 00 ...[.SMB-.....
00 00 00 00 00 00 00 00 00 00 00 00 00 00 E0 AD 20 .....
00 00 81 D9 0F FF 00 00 00 07 00 91 00 16 00 20 .....
00 14 CE 02 3A 10 00 00 00 00 00 00 00 00 00 00 .....:.....
00 00 00 1A 00 5C 57 49 4E 44 4F 57 53 5C 53 59 ..... \WINDOWS\SY
53 54 45 4D 5C 64 6E 65 74 63 2E 69 6E 69 00 STEM\dnetc.ini.
```

The fun begins the following day, on November 04th. First, it begins with the system 207.224.254.206 (dialupF206.sttl.uswest.net, NetBIOS name SOCCERDOG, account SCOTT, domain RONS) checking to see if *dnetc.ini* is installed on our honeynet. It determines that the binary is already installed and leaves the honeypot allone. That makes a total of 5 system probing our honeypot for this worm in 3 days. Later on that day our honeypot initiates a http connection to the system bymer.boom.ru. This connection was most likely initiated by the worm and is attempting to update the master server. The system bymer.boom.ru was most likely at one time the master controller for this worm. However, the system name bymer.boom.ru now resolves to an RFC 1918 IP address, 192.168.0.1. Most likely an attempt by the domain owner to stop the worm. Also, for the worm to execute like this, the system would need to have been rebooted at some time. That is the one thing we have not figured out, if the system was rebooted, and if so, how. One of the drawbacks of a Windows based honeypot is the limited availability of information, due to nonexistent logs. Below we see the honeypot initiating a connection to bymer.boom.ru, most likely the master server for the worm.

```
11/04-23:56:38.855453 172.16.1.105:1027 -> 192.168.0.1:80
TCP TTL:127 TOS:0x0 ID:65300 DF
**S**** Seq: 0x17AF8D9A Ack: 0x0 Win: 0x2000
TCP Options => MSS: 1460 NOP NOP SackOK
```

Immediately following this the *dnetc.exe* client connects to the distributed.net server and begins a data transfer. This is part of the the distributed.net client and not part of the worm replication process. However, this is the end purpose of the worm, to burn CPU cycles and upload the results to distributed.net.

```
11/04-23:56:40.286898 172.16.1.105:1029 -> 204.152.186.139:2064
TCP TTL:127 TOS:0x0 ID:1301 DF
*****PA* Seq: 0x17AF8F47 Ack: 0xBE445ED3 Win: 0x2238
AE 23 E2 77 F6 42 91 51 3E 61 3F EE 86 7F EE 8B .#.w.B.Q>a?.....
CE 9E 9D 28 16 BD 4B C5 5E DB FA 62 A6 FA A8 FF ...(..K.^..b....
EF 19 57 9C 37 38 06 39 7F 56 B4 D6 C7 75 63 73 ..W.78.9.V...ucs
```

```

0F 94 12 10 57 B2 C0 AD 9F D1 6F 4A E7 F0 1D E7   ....W.....oJ....
30 0E CC 84 78 2D 7B 21 C0 4C 29 BE 08 6A D8 5B   0...x-{!.L)..j.[
50 89 86 F8 98 A8 35 95 E0 C6 E4 32 28 E5 92 CF   P.....5.....2(...
71 04 41 6C B9 22 F0 09 01 41 9E A6 49 60 4D 43   q.A1."...A..I`MC
91 7E FB E0 D9 9D AA 7D 21 BC 59 1A 69 DB 07 B7   .~.....}!.Y.i...
B1 F9 B6 54 FA 18 64 F1 42 37 13 8E 8A 55 C2 2B   ...T..d.B7...U.+
CF 32 45 19 1A 93 1F 65 62 B1 CE 02 AA D0 7C 9E   .2E....eb.....|.
C5 46 78 29 F0 13 97 04                               .Fx).....
    
```

Once the upload is complete, the worm kicks into high gear and begins searching the Internet for other vulnerable system to replicate and spread itself. It randomly selects IP addresses and begins scanning those systems on ports 137 and 139. The worm identifies vulnerable systems (similar to our honeypot) and the replicates itself to the remote system. Compromised systems like these are one of the reasons for the high number of scans we have seen. Keep in mind, the Honeynet environment is designed to block any malicious traffic initiated by a compromised honeypot, so these scans never reach the Internet. The Honeynet is kind of like the 'roach motel'. It lets the bad guys in, but won't let them out. Below you see the worm attempting to find other vulnerable systems.

```

11/04-23:58:05.946299 172.16.1.105:137 -> 39.202.248.187:137
UDP TTL:127 TOS:0x0 ID:30485
Len: 58
0E 94 00 10 00 01 00 00 00 00 00 00 20 43 4B 41   ..... CKA
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41   AAAAAAAAAAAAAAAAAA
41 41 41 41 41 41 41 41 41 41 41 41 41 00 00 21   AAAAAAAAAAAAAA...!
00 01
    
```

One thing I found interesting was that the configuration file *c:\windows\win.ini* had been modified once again, most likely by the *wininit.exe* worm. The worm removed the entry of the *msi216.exe* worm from the startup configuration file, leaving itself in control. Also, the *dnetc.ini* file had been modified once again, changing the email address from *bymer@inec.kiev.ua* to the new email address *bymer@ukrpost.net*. This indicates that the second worm attempted to take over the first by eliminating it from the configuration files. This shows an extremely aggressive nature of worms, where one worm competes with another worm for real estate, or in this case CPU cycles..

If you would like to review this data yourself, you can download the file [win98.tar.gz](#). This gzipped file contains the four days of snort captures in binary format and all of the worms binaries on the honeypot, including *wininit.exe* and *msi216.exe*. Keep in mind, these are worms found in the wild, so you are working with malicious material. Be extremely careful working with it. For those of you who prefer not to mess with the worm binaries, you can download [win98-wo.tar.gz](#). This gzipped file contains everything win98.tar.gz contains, except for the two worm binaries *wininit.exe* and *msi216.exe*.

Conclusion

We have covered how in a 4 day period a Windows98 system was compromised by several worms. These worms are automated probes that identify and exploit vulnerable systems, exponentially replicating themselves. Its systems like these that are most likely scanning the Internet for NetBIOS vulnerabilities. This does not imply that every NetBIOS scan you receive is an automated worm. Nor that all worms are based for distributed.net. Consider if this worm was modified to look for confidential information on your system. The worm could easily search for documents with the words finance, confidential, secret or SSN. Once it found these documents, the information could easily be forwarded to an anonymous email account, IRC channel, or compromised webserver. The attacks are limited only by the imagination of the black-hat community.

Acknowledgements

In addition to the Honeynet project, this paper is the result of hard work and great input from two key individuals. A big thank you goes out to [H Carvey](#) and [Ryan Russell](#). They were the main contributors to the technical decoding of the events that happened here. For additional information about these vulnerabilities, both [distributed.net](#) and [CERT](#) have posted advisories.

The Honeynet Project