

[TOOL] ScanUDP, Improved UDP Scanning Tool

Source: <http://www.derkeiler.com/Mailing-Lists/Securiteam/2002-11/0021.html>

From: support@securiteam.com

Date: 11/07/02

From: support@securiteam.com

To: list@securiteam.com

Date: 7 Nov 2002 10:42:32 +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

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.

ScanUDP, Improved UDP Scanning Tool

DETAILS

This simple program written in C for Linux scan for UDP ports in remote hosts, determining which UDP services they are offering. All UDP scanners that Fryxar tried work with the following principle:

UDP datagram -> Closed Port -> ICMP Port Unreachable

UDP datagram -> Open Port -> No Reply (or application dependent)

But, if the scanned devices are behind a firewall, all the ports will seem open. So, what scanudp does is to send "dependend protocol packets", and wait for the application response. To insert new "dependen protocol packets", simple modify the "port" variable array.

Tool:

```
/* UDP Scanner (with protocol probes)
```

```
By: fryxar
```

```
Compile: gcc scanudp.c -o scanudp
```

```
*/
```

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

```
#include <sys/types.h>
```

```
#include <sys/socket.h>
```

```
#include <sys/select.h>
```

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

Securiteam: [TOOL] ScanUDP, Improved UDP Scanning Tool

```
#include <fcntl.h>
#include <unistd.h>
#include <signal.h>
#include <stdlib.h>
#include <errno.h>
#include <sys/stat.h>
#include <sys/wait.h>
#include <sys/time.h>
#include <netdb.h>

#define VERSION "1.0"
#define MAXBUF 1400
#define MAXSCAN 100
#define TIMEOUT 6

typedef struct config {
    int timeout;
} config;

typedef struct scan {
    u_int16_t number;// Number of udp port
    char *name;// Name of udp port
    char *outstring;// String to send (protocol dependent)
    int outstringlen;// Len above
    char *instring;// String to wait (protocol dependent, NULL for
anything)
    int instringlen;// Len above
    char match;// Does port match? Always initialized to 0
} scan;

// Port scanning probes definitions (protocol dependent)
struct scan port[] = {

    7, "echo",
    "probe", 5, "probe", 5, 0,

    13, "daytime",
    "\x0a", 1, NULL, 0, 0,

    19, "chargen",
    "\x0a", 1, NULL, 0, 0,

    // dig @ip localhost A
    53, "dns",

    "\x68\x6c\x01\x00\x00\x01\x00\x00\x00\x00\x00\x00\x00\x09\x6c\x6f\x63\x61\x6c\x68\x6f\x73\x74\x00\x00\x01\x00\x00",
    27, NULL, 0, 0,

    // echo "get a" | tftp ip
    69, "tftp",
    "\x00\x01\x61\x00\x6e\x65\x74\x61\x73\x63\x69\x69\x00", 13, NULL, 0,
```


Securiteam: [TOOL] ScanUDP, Improved UDP Scanning Tool

```
struct sockaddr_indest_addr;
fd_setfdset;
struct hostent *he;

// Set defaults
conf.timeout = 10;

if(argc < 2) usage( argv[0] );

while((opt = getopt(argc, argv, "t:")) != -1) {
    switch(opt) {
        case 't':
            if(strlen(optarg) == 0) usage(argv[0]);
            conf.timeout = atoi(optarg);
            break;

        default:
            usage(argv[0]);
            break;
    }
}

host = argv[argc-1];

if( (he = gethostbyname(host)) == NULL) {
    fprintf(stderr, "Error: Cannot resolve %s!\n", host);
    exit(-1);
}

FD_ZERO( &fdset );
maxfd = 0;

for( i = 0; port[i].number; i++ ) {
    if((fd[i] = socket(AF_INET, SOCK_DGRAM, 0)) < 0) {
        perror(NULL);
        exit(2);
    }
    if( maxfd < fd[i] ) maxfd = fd[i];
    FD_SET( fd[i], &fdset );

    dest_addr.sin_family = AF_INET;
    dest_addr.sin_addr = *((struct in_addr *)he->h_addr);
    dest_addr.sin_port = htons(port[i].number);

    if( connect(fd[i], (struct sockaddr *)&dest_addr, sizeof(dest_addr)) <
0) {
        perror(NULL);
        close(fd[i]);
        exit(4);
    }
}
```

Securiteam: [TOOL] ScanUDP, Improved UDP Scanning Tool

```
memcpy( buf, port[i].outstring, port[i].outstringlen );
if( send(fd[i], buf, port[i].outstringlen, 0) < 0 ) {
    perror(NULL);
    close(fd[i]);
    exit(5);
}
}

// Wait for timeout
sleep( conf.timeout );

tv.tv_sec= 0;
tv.tv_usec = 0;

if( select( maxfd+1, &fdset, NULL, NULL, &tv ) < 0 ) {
    perror(NULL);
    exit(6);
}

for( i = 0; port[i].number; i++ ) {
    if( !FD_ISSET( fd[i], &fdset ) ) {
        close( fd[i] );
        continue;
    }

    if( (nread = recv(fd[i], buf, MAXBUF, 0)) <= 0 ) {
        close( fd[i] );
        continue;
    }

    if( port[i].instring == NULL || !memcmp( buf, port[i].instring,
port[i].instringlen ) ) {
        for( repeat = 0, j = 0; j < i; j++ )
            if( port[i].number == port[j].number && port[j].match > 0 ) repeat =
1;

        if( !repeat ) {
            printf( "%s\t%d/udp\n", host, port[i].number );
            port[i].match++;
        }
        close( fd[i] );
    }
}

exit(0);
}
```

ADDITIONAL INFORMATION

The information has been provided by <<mailto:fryxar@datafull.com>> Fryxar.

Securiteam: [TOOL] ScanUDP, Improved UDP Scanning Tool

=====
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.

- **Previous message:** support@securiteam.com: "[NEWS] Com21 Cable Modem Configuration File Feeding Vulnerability"
- **Messages sorted by:** [\[date \]](#) [\[thread \]](#) [\[subject \]](#) [\[author \]](#) [\[attachment \]](#)