

WASM Crackme

지문

Yes, it needed to happen. **WASM Crackme 200!**

8 binaries will be generated.

find an input that prints "correct!"

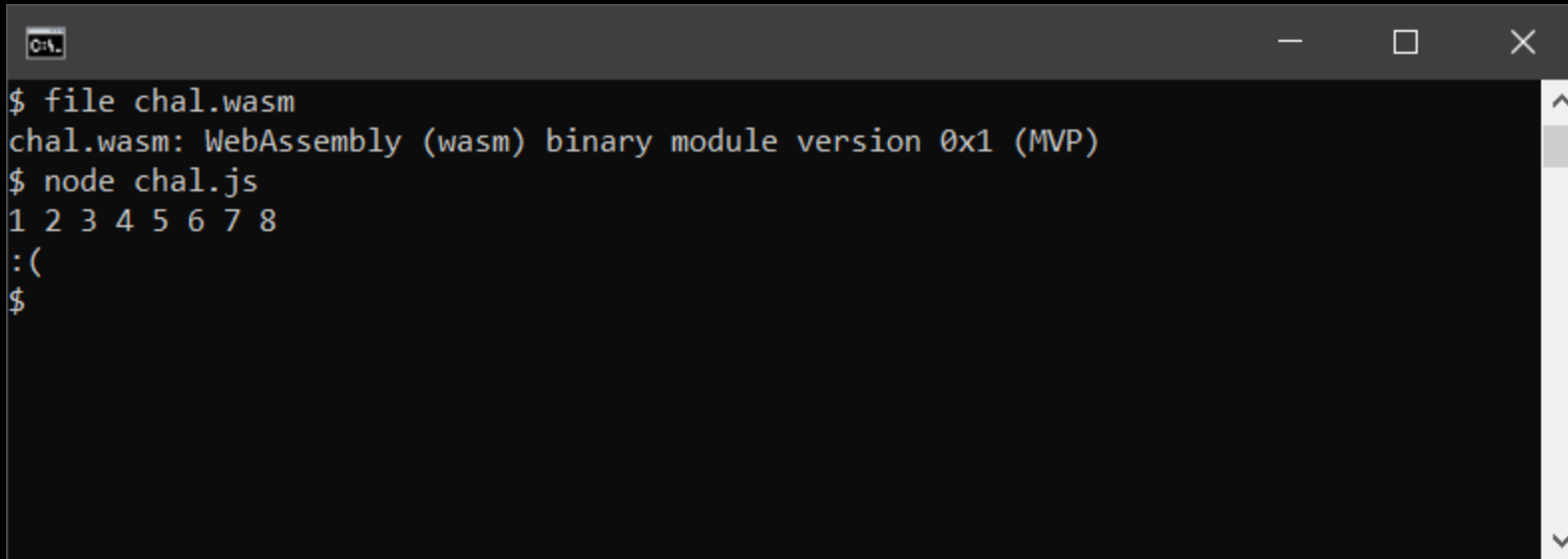
How to execute

```
$ echo 1 2 3 4 5 6 7 8 | node ./chal.js
```

correct!

바이너리 분석

- chal.js: chal.wasm을 실행해주는 스크립트
- chal.wasm: WebAssembly 파일

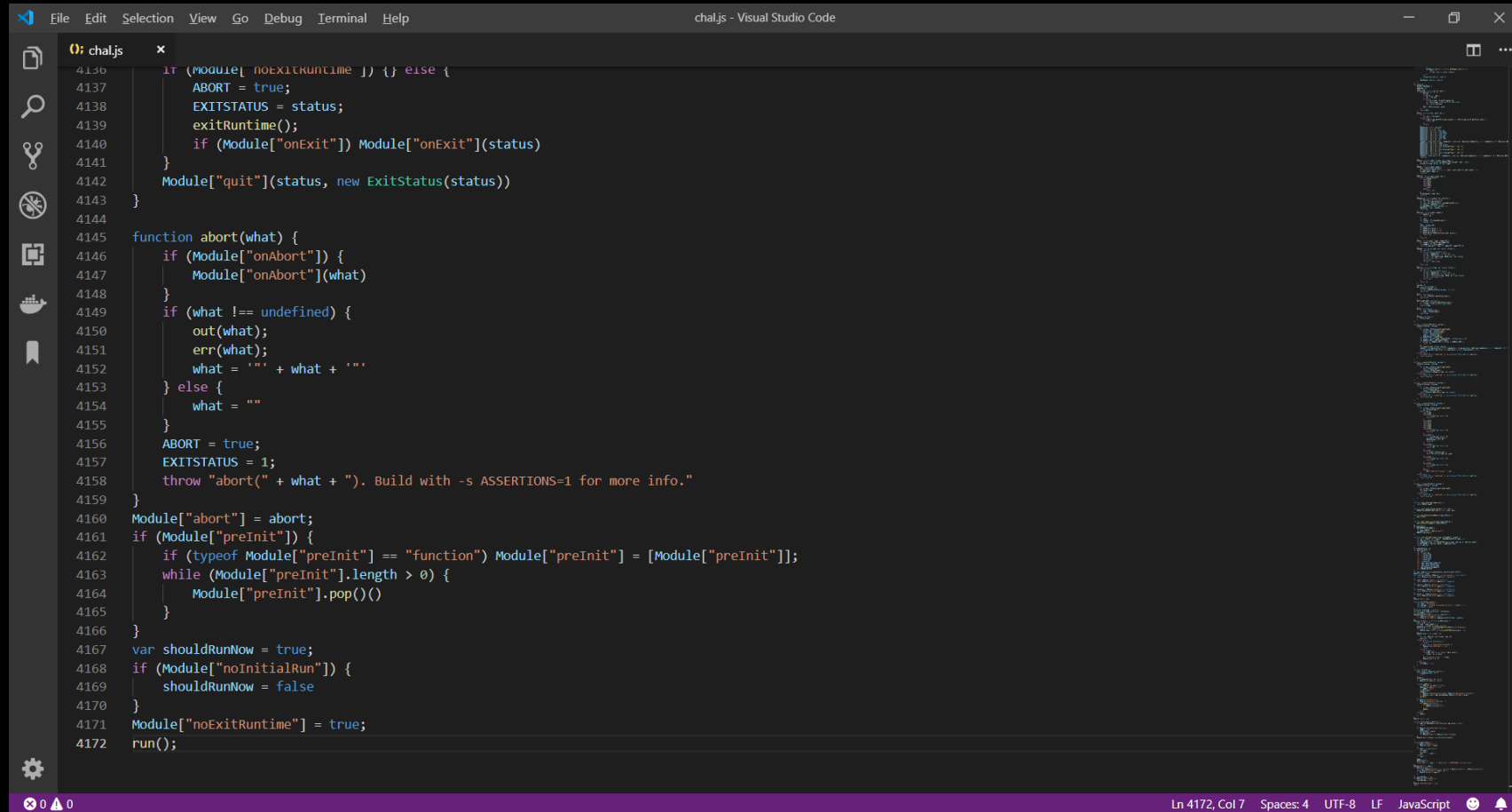
A terminal window with a dark background and light gray text. The window title bar shows a small icon and standard window controls (minimize, maximize, close). The terminal content shows the following commands and output:

```
$ file chal.wasm
chal.wasm: WebAssembly (wasm) binary module version 0x1 (MVP)
$ node chal.js
1 2 3 4 5 6 7 8
:(
$
```

chal.js

```
1 |var Module=typeof Module!=="undefined"?Module:{};var moduleOverrides={};var key;for(key in Module){if(Module.hasOwnProperty(key)){moduleOverrides[key]=Module[key]}}
Module["arguments"]=[];Module["thisProgram"]="./this.program";Module["quit"]=function(status,toThrow){throw toThrow};Module["preRun"]=[];Module["postRun"]=[];var
ENVIRONMENT_IS_WEB=false;var ENVIRONMENT_IS_WORKER=false;var ENVIRONMENT_IS_NODE=false;var ENVIRONMENT_HAS_NODE=false;var ENVIRONMENT_IS_SHELL=false;
ENVIRONMENT_IS_WEB=typeof window==="object";ENVIRONMENT_IS_WORKER=typeof importScripts==="function";ENVIRONMENT_HAS_NODE=typeof process==="object"&&typeof
require==="function";ENVIRONMENT_IS_NODE=ENVIRONMENT_HAS_NODE&&!ENVIRONMENT_IS_WEB&&!ENVIRONMENT_IS_WORKER;ENVIRONMENT_IS_SHELL=!ENVIRONMENT_IS_WEB&&
!ENVIRONMENT_IS_NODE&&!ENVIRONMENT_IS_WORKER;var scriptDirectory="";function locateFile(path){if(Module["locateFile"]){return Module["locateFile"](path,scriptDirectory)}
else{return scriptDirectory+path}}if(ENVIRONMENT_IS_NODE){scriptDirectory=_dirname+"/";var nodeFS;var nodePath;Module["read"]=function shell_read(filename,binary){var
ret;if(!nodeFS)nodeFS=require("fs");if(!nodePath)nodePath=require("path");filename=nodePath["normalize"](filename);ret=nodeFS["readFileSync"](filename);return binary?
ret:ret.toString();}Module["readBinary"]=function readBinary(filename){var ret=Module["read"](filename,true);if(!ret.buffer){ret=new Uint8Array(ret)}assert(ret.buffer);
return ret};if(process["argv"].length>1){Module["thisProgram"]=process["argv"][1].replace(/\\/g,"/")}Module["arguments"]=process["argv"].slice(2);if(typeof
module!=="undefined"){module["exports"]=Module}process["on"]("uncaughtException",function(ex){if(!(ex instanceof ExitStatus)){throw ex}});process["on"]
("unhandledRejection",abort);Module["quit"]=function(status){process["exit"](status)};Module["inspect"]=function(){return[Emscripten Module object]}"}else if
(ENVIRONMENT_IS_SHELL){if(typeof read!=="undefined"){Module["read"]=function shell_read(f){return read(f)}}Module["readBinary"]=function readBinary(f){var data;if(typeof
readbuffer==="function"){return new Uint8Array(readbuffer(f))}data=read(f,"binary");assert(typeof data==="object");return data};if(typeof scriptArgs!="undefined"){Module
["arguments"]=scriptArgs}else if(typeof arguments!="undefined"){Module["arguments"]=arguments}if(typeof quit==="function"){Module["quit"]=function(status){quit(status)}}
}else if(ENVIRONMENT_IS_WEB||ENVIRONMENT_IS_WORKER){if(ENVIRONMENT_IS_WORKER){scriptDirectory=self.location.href}else if(document.currentScript)
{scriptDirectory=document.currentScript.src}if(scriptDirectory.indexOf("blob:")!==0){scriptDirectory=scriptDirectory.substr(0,scriptDirectory.lastIndexOf("/")+1)}else
{scriptDirectory=""}}Module["read"]=function shell_read(url){var xhr=new XMLHttpRequest;xhr.open("GET",url,false);xhr.send(null);return xhr.responseText};if
(ENVIRONMENT_IS_WORKER){Module["readBinary"]=function readBinary(url){var xhr=new XMLHttpRequest;xhr.open("GET",url,false);xhr.responseType="arraybuffer";xhr.send(null);
return new Uint8Array(xhr.response)}}Module["readAsync"]=function readAsync(url,onload,onerror){var xhr=new XMLHttpRequest;xhr.open("GET",url,true);
xhr.responseType="arraybuffer";xhr.onload=function xhr_onload(){if(xhr.status===200||xhr.status===0&&xhr.response){onload(xhr.response);return}onerror();
xhr.onerror=onerror;xhr.send(null)};Module["setWindowTitle"]=function(title){document.title=title};else{}var out=Module["print"]||(typeof console!=="undefined"?
console.log.bind(console):typeof print!=="undefined"?print:null);var err=Module["printErr"]||(typeof printErr!=="undefined"?printErr:typeof console!=="undefined"&&
console.warn.bind(console)||out);for(key in moduleOverrides){if(moduleOverrides.hasOwnProperty(key)){Module[key]=moduleOverrides[key]}}moduleOverrides=undefined;
function dynamicAlloc(size){var ret=HEAP32[DYNAMICTOP_PTR>>2];var end=ret+size+15&-16;if(end>_emscripten_get_heap_size()){abort()}HEAP32[DYNAMICTOP_PTR>>2]=end;return
ret}function getNativeTypeSize(type){switch(type){case"i1":case"i8":return 1;case"i16":return 2;case"i32":return 4;case"i64":return 8;case"float":return 4;
case"double":return 8;default:{if(type[type.length-1]==="*"){return 4}else if(type[0]==="i"){var bits=parseInt(type.substr(1));assert(bits%8===0,"getNativeTypeSize
invalid bits "+bits+", type "+type);return bits/8}else{return 0}}}}var asm2wasmImports={"f64-rem":function(x,y){return x/y},"debugger":function(){debugger}};var
functionPointers=new Array(0);if(typeof WebAssembly!=="object"){err("no native wasm support detected")}function setValue(ptr,value,type,noSafe){type=type||"i8";if
(type.charCodeAt(type.length-1)==="*")type="i32";switch(type){case"i1":HEAP8[ptr>>0]=value;break;case"i8":HEAP8[ptr>>0]=value;break;case"i16":HEAP16[ptr>>1]=value;break;
case"i32":HEAP32[ptr>>2]=value;break;case"i64":tempI64=[value>>>0,(tempDouble=value,+Math_abs(tempDouble)>=1?tempDouble>0?(Math_min(+Math_floor(tempDouble/4294967296),
4294967295)|0)>>>0:~+Math_ceil((tempDouble-+(~tempDouble>>>0))/4294967296)>>>0:0],HEAP32[ptr>>2]=tempI64[0],HEAP32[ptr+4>>2]=tempI64[1];break;case"float":HEAPF32
[ptr>>2]=value;break;case"double":HEAPF64[ptr>>3]=value;break;default:abort("invalid type for setValue: "+type)}}var wasmMemory;var wasmTable;var ABORT=false;var
```

chal.js -> <https://beautifier.io>



```
0: chal.js x
4136   if (Module["noExitRuntime"] || !{}) else {
4137       ABORT = true;
4138       EXITSTATUS = status;
4139       exitRuntime();
4140       if (Module["onExit"]) Module["onExit"](status)
4141   }
4142   Module["quit"](status, new ExitStatus(status))
4143 }
4144
4145 function abort(what) {
4146     if (Module["onAbort"]) {
4147         Module["onAbort"](what)
4148     }
4149     if (what !== undefined) {
4150         out(what);
4151         err(what);
4152         what = ' ' + what + ' '
4153     } else {
4154         what = ''
4155     }
4156     ABORT = true;
4157     EXITSTATUS = 1;
4158     throw "abort(" + what + "). Build with -s ASSERTIONS=1 for more info."
4159 }
4160 Module["abort"] = abort;
4161 if (Module["preInit"]) {
4162     if (typeof Module["preInit"] == "function") Module["preInit"] = [Module["preInit"]];
4163     while (Module["preInit"].length > 0) {
4164         Module["preInit"].pop()()
4165     }
4166 }
4167 var shouldRunNow = true;
4168 if (Module["noInitialRun"]) {
4169     shouldRunNow = false
4170 }
4171 Module["noExitRuntime"] = true;
4172 run();
```

chal.js 분석

run()

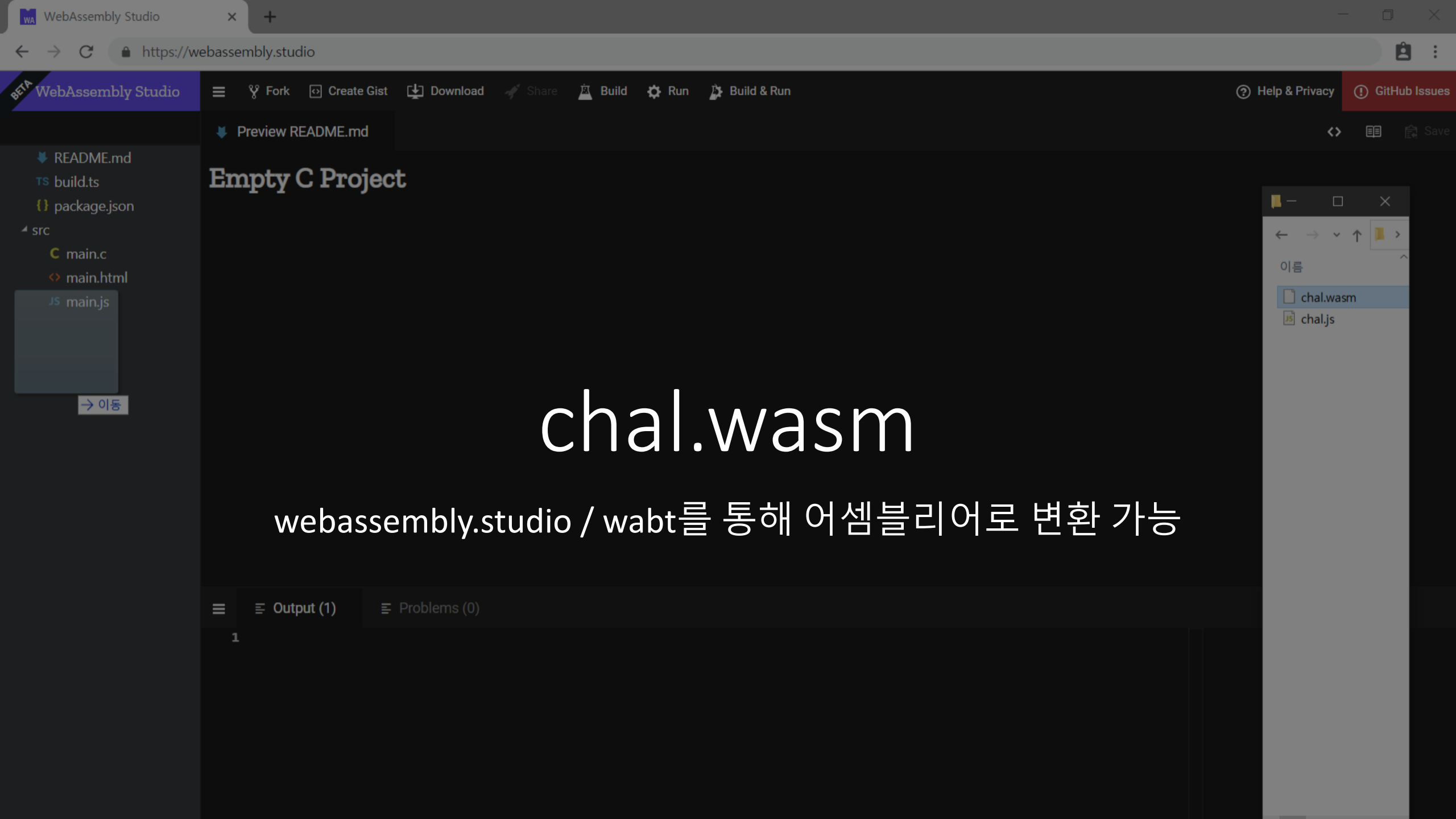
→Module["callMain"](args);

→var ret = Module["_main"](argc, argv, 0);

→return Module["asm"]["n"].apply(null, arguments)

Module["asm"] -> chal.wasm

```
617 var wasmBinaryFile = "chal.wasm";
618
619 function createWasm(env) {
620   function receiveInstance(instance, module) {
621     Module["asm"] = exports;
622   }
623
624   function receiveInstantiatedSource(output) {
625     receiveInstance(output["instance"])
626   }
627
628   function instantiateAsync() {
629     if (/*...*/ true) {
630       fetch(wasmBinaryFile, {
631         credentials: "same-origin"
632       }).then(function(response) {
633         return WebAssembly.instantiateStreaming(response, info)
634           .then(receiveInstantiatedSource)
635       })
636     }
637   }
638
639   instantiateAsync();
640   return {}
641 }
```



WA

WebAssembly Studio

←

→

↺

https://webassembly.studio

👤

⋮

BETA

WebAssembly Studio

☰

Fork

Create Gist

Download

Share

Build

Run

Build & Run

🔍

Help & Privacy

🔔

GitHub Issues

📄

Preview README.md

📁

chal.wasm

📄

Save

📄

README.md

TS

build.ts

{}

package.json

src

C

main.c

<>

main.html

JS

main.js

📁

chal.wasm

This .wasm file is editable as a .wat file, and is automatically reassembled to .wasm when saved.

15911

i32.const -1

15912

set_local \$p0

15913

end

15914

get_local \$l0

15915

set_global \$g4

15916

get_local \$p0)

15917

(func \$n (export "n") (type \$t6) (result i32)

15918

(local \$l0 i32) (local \$l1 i32) (local \$l2 i32) (local \$l3 i32) (local \$l4 i32) (local \$l5 i64)

15919

get_global \$g4

15920

set_local \$l1

15921

get_global \$g4

15922

i32.const 16

15923

i32.add

15924

set_global \$g4

15925

get_local \$l1

15926

i32.const 8

15927

i32.add

15928

set_local \$l3

15929

i32.const 2772

15930

i32.load

15931

tee_local \$l0

export "n"

Aa

Abi

*

1 of 1

←

→

☰

✕

☰

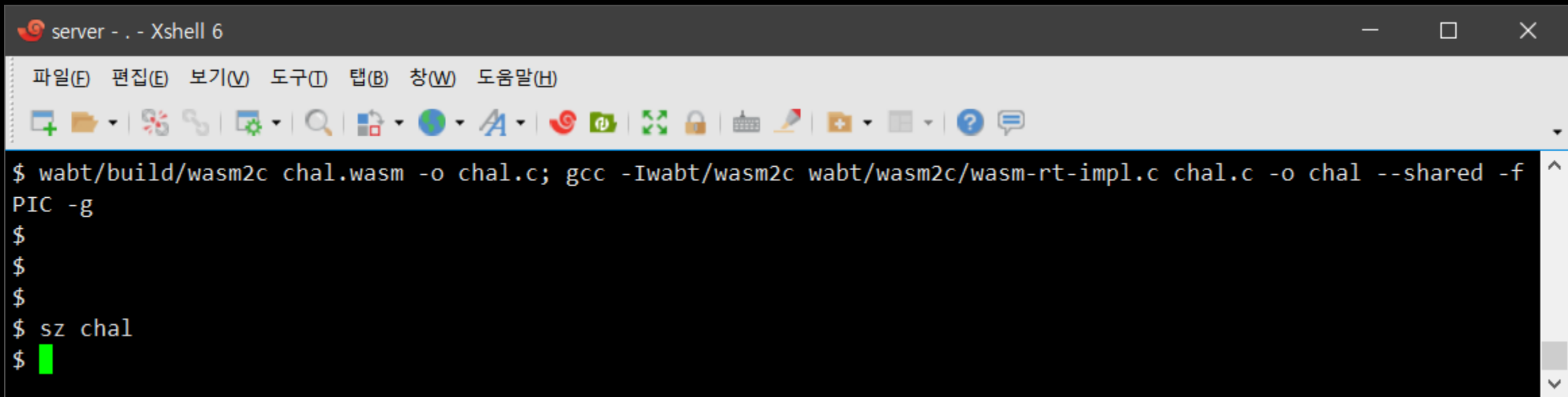
☰

Output (1)

☰

Problems (0)

1



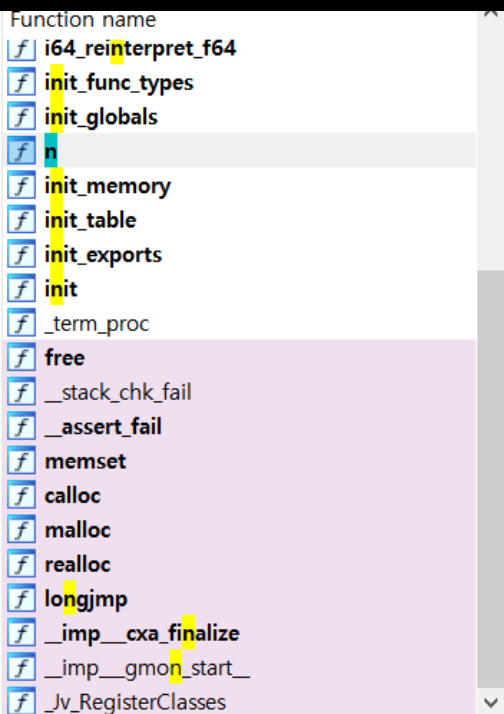
The screenshot shows an Xshell 6 terminal window with a menu bar in Korean (파일(F), 편집(E), 보기(V), 도구(T), 탭(B), 창(W), 도움말(H)) and a toolbar. The terminal content is as follows:

```
$ wabt/build/wasm2c chal.wasm -o chal.c; gcc -Iwabt/wasm2c wabt/wasm2c/wasm-rt-impl.c chal.c -o chal --shared -fPIC -g
$
$
$
$ sz chal
$
```

chal.wasm

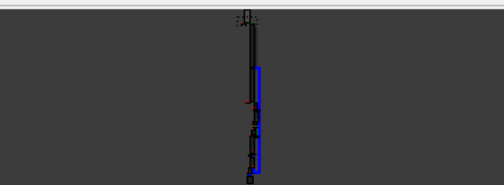
- JEB 또는 wasm2c + Ghidra/IDA를 통해 C로 변환 가능
- callMain 함수 내에 `require("fs").writeFileSync("memory", new Uint8Array(wasmMemory.buffer))`를 넣어 메모리 덤프 (권장)

“n” (main)



Line 14 of 30

Graph overview



```
j4= qword ptr -8  
; __unwind {  
push    rbp  
mov     rbp, rsp  
sub     rsp, 60h  
mov     [rbp+10], 0  
mov     [rbp+11], 0  
mov     [rbp+12], 0  
mov     [rbp+13], 0  
mov     [rbp+14], 0  
mov     [rbp+15], 0  
mov     rax, cs:wasm_rt_call_stack_depth_ptr  
mov     eax, [rax]  
lea     edx, [rax+1]  
mov     rax, cs:wasm_rt_call_stack_depth_ptr  
mov     [rax], edx  
mov     rax, cs:wasm_rt_call_stack_depth_ptr  
mov     eax, [rax]  
cmp     eax, 1F4h  
jbe     short loc_31047
```

```
mov     edi, 7 ; code  
call    _wasm_rt_trap
```

```
loc_31047:  
mov     eax, cs:g4  
mov     [rbp+i0], eax  
mov     eax, [rbp+i0]  
mov     [rbp+11], eax  
mov     eax, cs:g4  
mov     [rbp+i0], eax
```

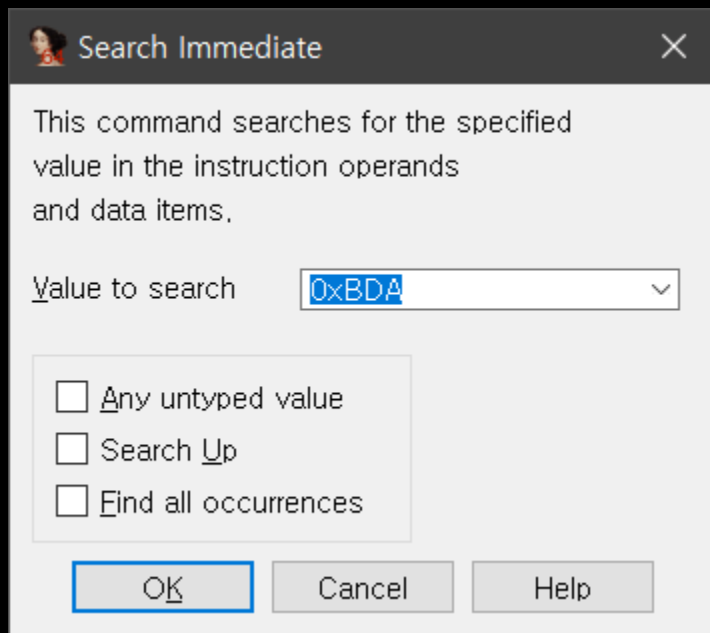
메모리 덤프 -> 코드

- 빠른 분석을 위해 덤프한 메모리 파일을 열어보자
- ex. “correct!” at 0xBDA, “:(” at 0xBE3

```
00000BB0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
00000BC0  00 00 00 00 00 00 00 00 00 00 00 00 5F 70 89 00  ....._p%.
00000BD0  FF 09 2F 0F 20 25 6C 6C 64 00 63 6F 72 72 65 63  y./.%lld.correct
00000BE0  74 21 00 3A 28 00 00 01 02 04 07 03 06 05 00 69  t!.:(......i
00000BF0  6E 66 69 6E 69 74 79 00 6E 61 6E 00 00 00 00 00  nfinity.nan.....
00000C00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
00000C10  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
```

메모리 덤프 -> 코드

- IDA 기준 Search 메뉴의 immediate value... (Alt + I) 기능으로 해당 상수를 찾음(포인터 또한 정수의 형태를 띠므로)



```
74 while ( l2a < 7 );  
75 i2a = (unsigned __int8)l0 == 0;  
76 l0a = (unsigned __int8)l0 == 0;  
77 if ( i2a )  
78     v7 = 0xBE3;           // :(  
79 else  
80     v7 = 0xBDA;           // correct!  
81 f116(v7);
```

“n” (main). f116은 puts임을 유추할 수 있음.

많이 보이는 코드 패턴

1. 함수 프로로그 - 스택 초기화

- emscripten (binaryen) 에서 C -> asm.js로 컴파일 할 때 쓰이는 시스템 스택 구현임 (x86의 esp와 같음)

```
// from wasm2c
if ( ++wasm_rt_call_stack_depth > 500 )
    wasm_rt_trap(WASM_RT_TRAP_EXHAUSTION);
// from emscripten
l1 = g4; // g4 = stack pointer (like esp)
g4 += 16; // add stack pointer
arguments = l1 + 8; // l1 + 0..15 = stack variable
```

많이 보이는 코드 패턴

2. 함수 프로로그 – call depth 제한

- wasm_rt...는 wasm -> c로 변환 중에 생김 (실제 웹어셈블리에는 없음)
- call depth를 제한하는 것으로 보임

```
// from wasm2c
if ( ++wasm_rt_call_stack_depth > 500 )
    wasm_rt_trap(WASM_RT_TRAP_EXHAUSTION);
// from emscripten
l1 = g4; // g4 = stack pointer (like esp)
g4 += 16; // add stack pointer
arguments = l1 + 8; // l1 + 0..15 = stack variable
```

많이 보이는 코드 패턴

3. 전역 변수 접근

- emscripten에서 생성하는 전역 변수 load/store 패턴
- i32_store, ... load/store는 wasm2c에서 생성한 메모리 접근 헬퍼 함수
 - 어셈블리는 “i32.store”, ...

(4, 8바이트 정수 불러오기)

```
index = i32_load(Z_envZ_memory, 0x1430uLL);
```

```
if ( encrypted != i64_load(Z_envZ_memory, (8 * 12a + 0x730)) && i2 )
```

```
    valid = 0;
```

(4바이트 정수 저장)

```
i32_store(Z_envZ_memory, 0x1430uLL, index + 1);
```

```
→0x1430: int32_t index, 0x730: int64_t target[]
```


분석할 시간!

- 지역 변수, 함수 리네이밍
 - f116 -> puts, l0 -> valid, f115 -> scanf + “ %lld”, ...
- 전역 변수 위치 기록
 - 0x770: values, 0x1430: counter, 0x730: target, ...
- f115는 분명 C로는 scanf(“ %lld”, &input)으로 했는데...
 - 포맷 스트링에 대한 함수가 따로 만들어짐
`int f116(int *n) {return scanf(“ %lld”, n);}`
 - 출제자도 당황했음

루틴 정리

```
int main() {
    uint64_t input;
    char valid = 1;
    setvbuf(stdin, 0, 2, 0);
    setvbuf(stdout, 0, 2, 0);
    for(offset = 0; offset < 8; offset++) {
        r = keys[offset];
        if(scanf(" %lld", &input) != 1) {
            valid = 0;
        }
        if(!check(input)) valid = 0;
    }
    puts(valid ? "correct!" : ":(");
    return !valid;
}
```

```
void nop() {}
uint64_t next_ptr() {
    r = r * 7 / 8;
    if(!r) {
        return sizeof(table) / sizeof(table[0]) - 1;
    }
    return r % (sizeof(table) / sizeof(table[0]) - 1);
}

void func0() {a = a * 226896743758358843uLL;... table[next_ptr()]();}
void func199() {a = a + 82630934117059952uLL;... table[next_ptr()]();}
void (*table[])() = {func0, ..., func199, nop};

uint64_t values[8] = {<generated>}, keys[8] = {<generated>};
int offset;
int check(uint64_t input) {
    a = input;
    table[next_ptr()]();
    return a == values[offset];
}
```

역연산 코딩

- 디컴파일된 코드 또는 어셈블리 파싱

```
7165 (func $f48 (type $t0)
7166   (local $l0 i32) (local $l1 i64)
7167   i32.const 5160
7168   i32.const 5160
7169   i64.load
7170   i64.const 118569198327502503
7171   i64.xor
7172   i64.const 210829669964035999
7173   i64.mul
7174   i64.const 703053700851818
7175   i64.xor
7176   i64.const 524719026176714179
7177   i64.mul
7178   i64.const 6855619049331684745
7179   i64.add
7180   i64.store
7181   i32.const 5152
7182   i32.const 5152
7183   i64.load
```

또는...

- 출제자의 특권을 사용하여...
원본 C코드 생성 시 파싱 후 역연산 생성
- mbagen.py 54~76번 참조

Q&A

Thanks for reading!

https://github.com/codeengn/codeengn_ctf