2018

绿盟科技 NSFOCUS

WASM双刃剑—机制剖析与逆向

演讲人：赵光远

目录
CONTENTS

01 PART 01 **WASM**

02 PART 02 **Compile**

03 PART 03 **挖矿实例**

04 PART 04 **探索方向**

KCon

PART
01

WASM

- 前端编程无法解决的问题
  - 传统的js脚本运行缓慢
  - 在浏览器中无法实现复杂功能(效率不足，影响体验)
  - 很低的渲染能力，却拥有CPU、GPU极高的占用率

- 应用场景
  - 浏览器页面3D环境渲染
  - 数据压缩
  - 图像/音频/视频的处理
  - ~~挖矿(本身就是计算的一种)~~
  - ....

- 支持度



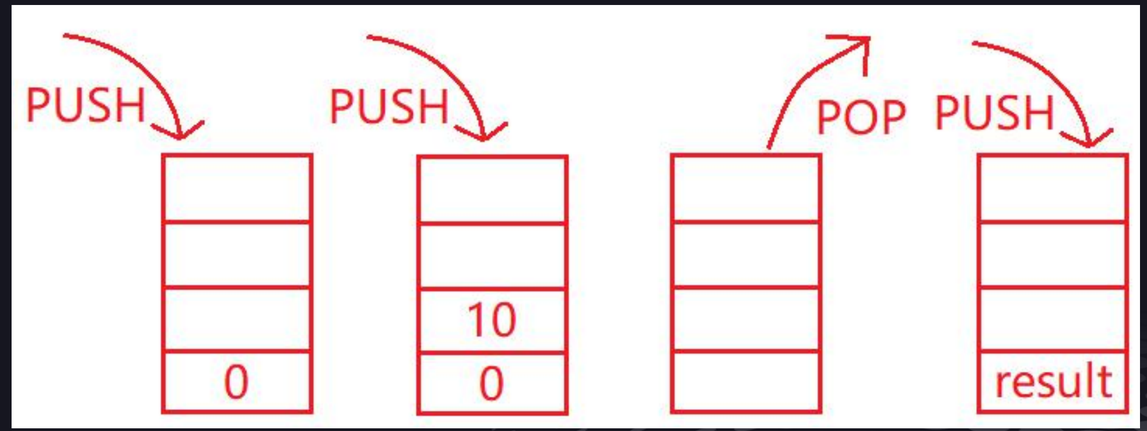

- WebAssembly(WASM)
  - 基于栈的二进制格式文件
  - 支持C/C++/Rust等高级语言的转换
  - 理论上能够将现有的C/C++代码库直接编译运行，无需重写
  - 支持Write Once,Run anywhere
  - 由JVM解释并执行，运行时处于隔离环境中
  - 不能直接操作Dom元素，不能直接调用I/O
  - 只能通过WebSocket对外通信
  - 只能使用binaryen-shell与JavaScript交互
  - 只能通过为JavaScript提供的接口进行调用

• 基于栈的运行机制

```
int add(int num) {
    return num+10;
}
```

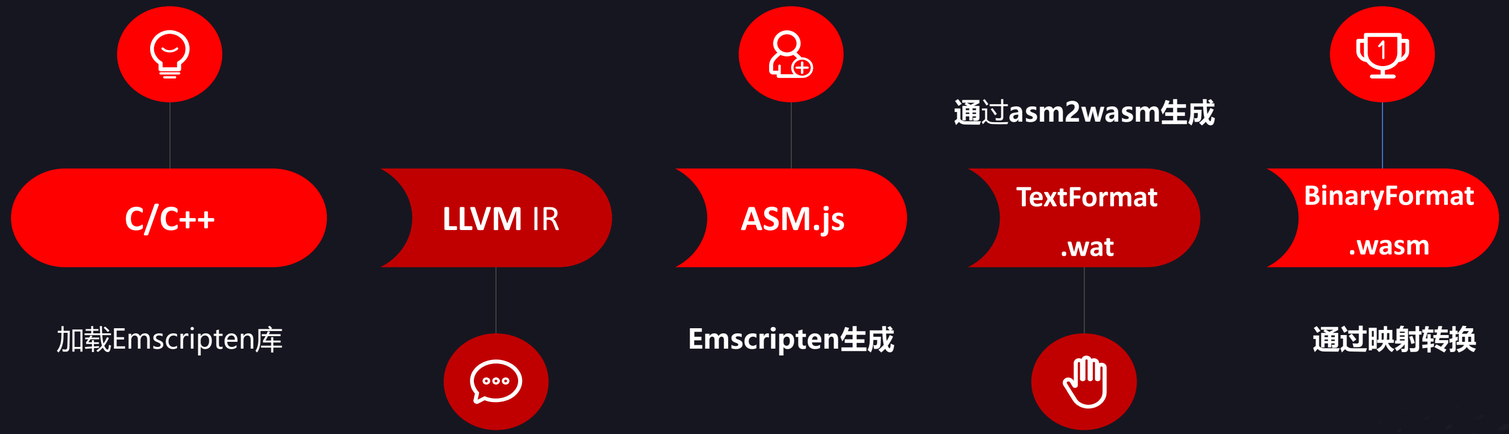
00 61 73 6D 0D 00 00 00 01 86 80 80 80 00 01 60
01 7F 01 7F 03 82 80 80 80 00 01 00 04 84 80 80
80 00 01 70 00 00 05 83 80 80 80 00 01 00 01 06
81 80 80 80 00 00 07 96 80 80 80 00 02 06 6D 65
6D 6F 72 79 02 00 09 5F 5A 35 61 64 64 34 32 69
00 00 0A 8D 80 80 80 00 01 87 80 80 80 00 00 20
00 41 0A 6A 0B

```
get_local 0
i32.const 10
i32.add
```



PART
02

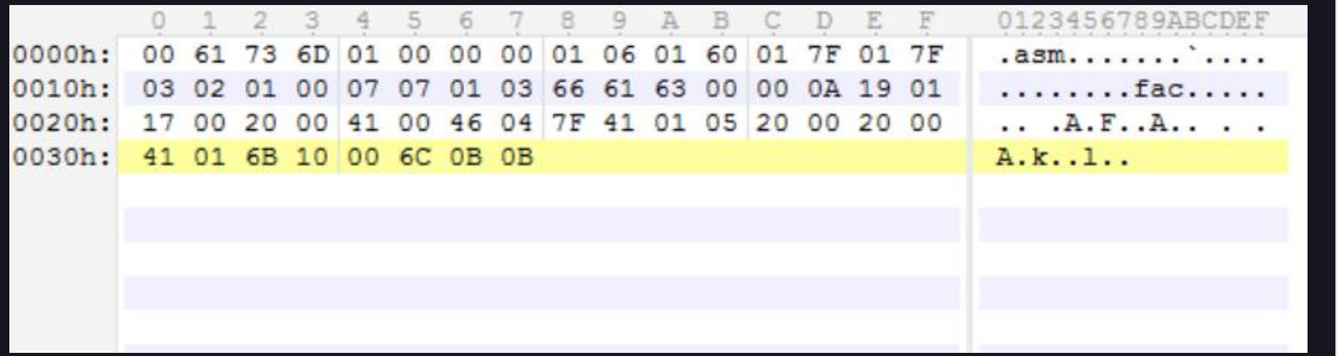
编译及反编译过程


C/C++
LLVM IR
ASM.js
通过asm2wasm生成
TextFormat
.wat
BinaryFormat
.wasm
加载Emscripten库
Emscripten生成
通过映射转换

BinaryFormat
.wasm

WASM2C

C/C++



```
          0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  0123456789ABCDEF
0000h: 00 61 73 6D 01 00 00 00 01 06 01 60 01 7F 01 7F  .asm.......`....
0010h: 03 02 01 00 07 07 01 03 66 61 63 00 00 0A 19 01  ........fac.....
0020h: 17 00 20 00 41 00 46 04 7F 41 01 05 20 00 20 00  .. .A.F..A.. . .
0030h: 41 01 6B 10 00 6C 0B 0B                          A.k..l..
```

```
static u32 fac(u32 p0) {
  FUNC_PROLOGUE;
  u32 i0, i1, i2;
  i0 = p0;
  i1 = 0u;
  i0 = i0 == i1;
  if (i0) {
    i0 = 1u;
  } else {
    i0 = p0;
    i1 = p0;
    i2 = 1u;
    i1 -= i2;
    i1 = fac(i1);
    i0 *= i1;
  }
  FUNC_EPILOGUE;
  return i0;
}
```

KCon

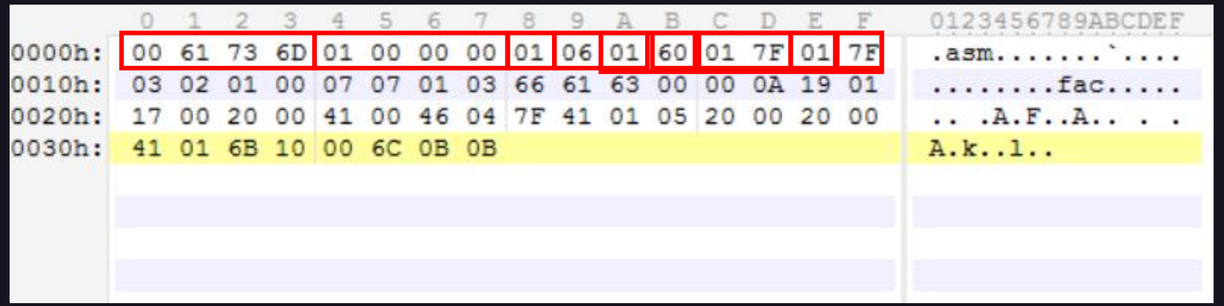
- 一个模块中包含的节(必需)
  - type
  - function
  - code
- Type类型(部分)
  - int
    - i32
    - i64
  - float
    - f32
    - F64
  - functype

$$\begin{array}{lcll} \text{section}_N(\text{B}) & ::= & N{:}\text{byte}\ \ size{:}\text{u32}\ \ cont{:}\text{B} & \Rightarrow\ cont\ \ (\text{if}\ size = ||\text{B}||) \\ & | & \epsilon & \Rightarrow\ \epsilon \end{array}$$

```
(module
 (type $0 (func (param i32 i32) (result i32)))
 (memory $0 256 256)
 (export "add" (func $add))
 (func $add (; 0 ;) (type $0) (param $x i32) (param $y i32) (result i32)
  (i32.add
   (get_local $x)
   (get_local $y)
  )
 )
)
```

$$\text{functype} ::= \text{0x60}\ t_1^*{:}\text{vec}(\text{valtype})\ t_2^*{:}\text{vec}(\text{valtype}) \Rightarrow [t_1^*] \to [t_2^*]$$

- wasm的二进制文件格式



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	0123456789ABCDEF
0000h:	00	61	73	6D	01	00	00	00	01	06	01	60	01	7F	01	7F	.asm.....`....
0010h:	03	02	01	00	07	07	01	03	66	61	63	00	00	0A	19	01	........fac.....
0020h:	17	00	20	00	41	00	46	04	7F	41	01	05	20	00	20	00	.. .A.F..A.. . .
0030h:	41	01	6B	10	00	6C	0B	0B									A.k..l..

- 0x0 flag
- 0x4 magic
- 0x8 section_id
- 0x9 length
- 0xA type
- 0xB function type start
- 0xC val:type
- 0xE val:type
- ……

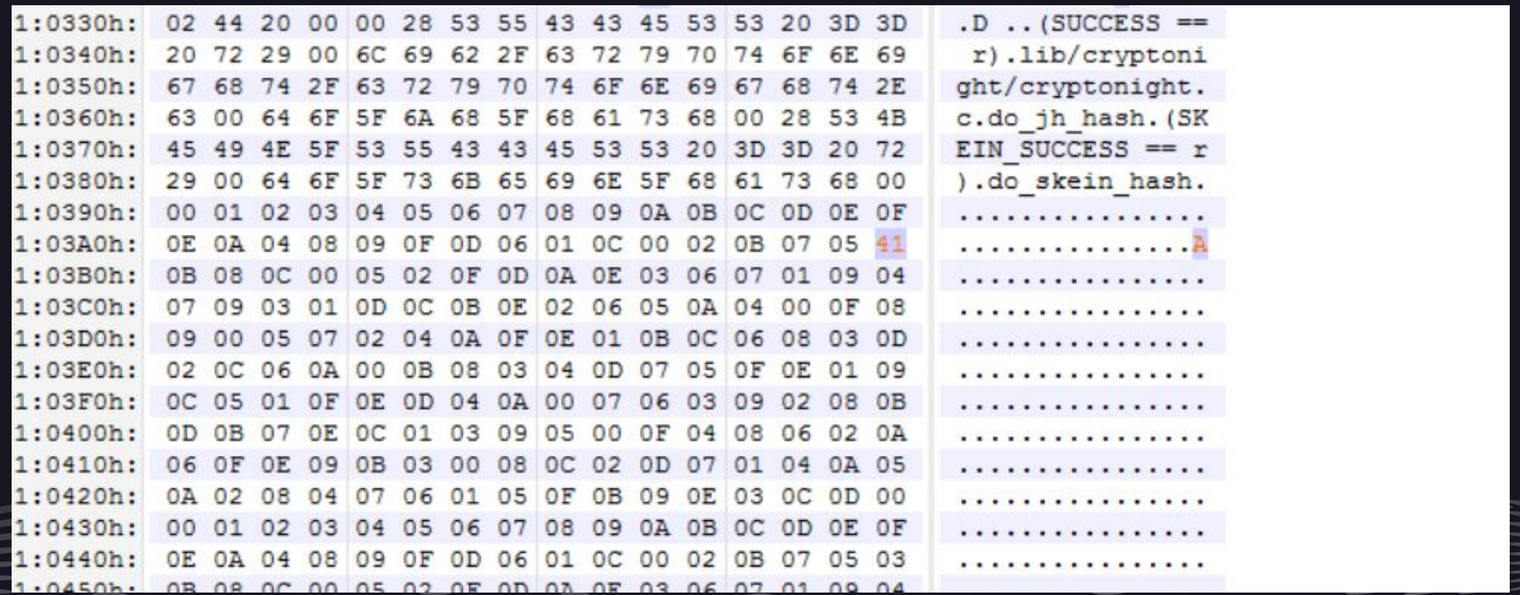
PART
03

挖矿实例


In-browser mining: Coinhive and WebAssembly | Forcepoint
https://www.forcepoint.com/zh.../browser-mining-coinhive-and-webassembl...
After a few evolutionary steps via Application Specific Integrated Circuits (ASICs) mining
seem to have returned to their roots: the 'humble' personal ...
JavaScript-Emscripten-Bitcoin-Miner | WebAss
www.wasmrocks.com › Coin Minners
This project implements a
Security
Another day, another cryptocurrency miner lurking in a Google Chrome extension
Plus: A new stealthier Monero crafter emerges
https://github.com/ohac/wasmminer
wasmminer. build wsproxy and run. sudo apt install golang export GO
github.com/gorilla/websocket cd wsproxy go build ./wsproxy ...
ero WebAssembly based miner
im of this project is to provide a completely open source, browser based, Monero miner. It has a companion project,
f Snipa's xmr-node-proxy which adds a WebSocket based branch for allowing this miner to connect through to
various mining pools.
Whilst browser mining can be used maliciously, there are many non-malicious use-cases, such as charitable donations (the very
reason this project started). A defining aspect to whether web mining is done in a fair way is that of user consent. My
recommendation is therefore to ensure the end-user knows the page will be mining and offer them controls such as starting
and stopping, and options for how much processing time the mining should use.
Compiling
Behavior
Miner.Wasmwe
Note
ryptocurrency miner that runs in web browsers.
n.Webcoinminer.
resources:
, 2017 revision 005
ust 05, 2018 revision 024
n December 26, 2017 revision 006
d version August 06, 2018 revision 002
y Certified release date December 27, 2017

KCon

- JavaScript脚本编写的挖矿程序，利用coinhive提供的API
- 将JavaScript脚本编译为wasm，在支持运行wasm运行的浏览器中执行挖矿行为，不支持的浏览器会执行JavaScript脚本



PART
04

研究方向

- wasm在运行效率方面比JavaScript表现的更加出色，因此wasm将被广泛应用在游戏，虚拟场景模拟等方面，这意味着wasm在以后的应用中会占据一席之地
- wasm运行在JVM沙箱中，对外的交互基本上都由JavaScript进行接管，JavaScript不了解WASM中运行的内容，只能获取到指定的返回信息
- wasm对于JavaScript来说就是一个黑盒，获知其中的运行状态及内容很难
- 大多数的WAF产品目前没有wasm解析器，因此无法对其进行有效的特征检测及拦截
- wasm由JVM解释执行，是否会支持C/C++等高级语言也可以导入wasm完成某些功能，将这些功能隐藏在wasm中具有相当好的隐蔽性，如何对其进行检测？

网络数据流实时检测

在下载或发送时对其进行检测
加密流量的分离及分析

漏洞挖掘

利用其能将高级语言编译为webbytecode的特性，挖掘相关漏洞

本地动态行为检测

WASM文件执行情况及功能的动态检测方案

代码保护

wasm文件格式清晰，非常容易直接获取代码，对代码应如何混淆并加以保护

KCon

引用及参考

- https://github.com/WebAssembly
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/WebAssembly/
- https://webassembly.github.io/spec/core/

Question?

谢谢观看

演讲人：赵光远