MazeWalker

Enriching static malware analysis
and more

Yevgeny Kulakov
@p_h_0_e_n_i_x

About Me

- Malware RE @ Trusteer, IBM, Seculert
    - binary analysis automation
    - sandbox development
- Now in vEYE Security on software container problems

Agenda

- Malware vs Reverser
- General idea behind MazeWalker Tool
- How and What MazeWalker solves
- Demo
- Future work

Malware vs Reverser

Prevent or slowdown manual analysis

Make me suffer

Some examples of annoying behaviour

Code (un)packing

- New executable areas introduced
- Runtime code change
- Stack-based execution

Sample Layer 0 → **Layer 1** Decrypt code → **Layer 1** Resolve + Unpack → **Payload**

Code (un)packing - PiC

- Runtime CF change - Indirect Calls & Jumps

```
call    eax
inc     ecx
call    eax
xchg    ecx, edx
neg     edx
jmp     short loc_2D531F3
```

Environment Detection

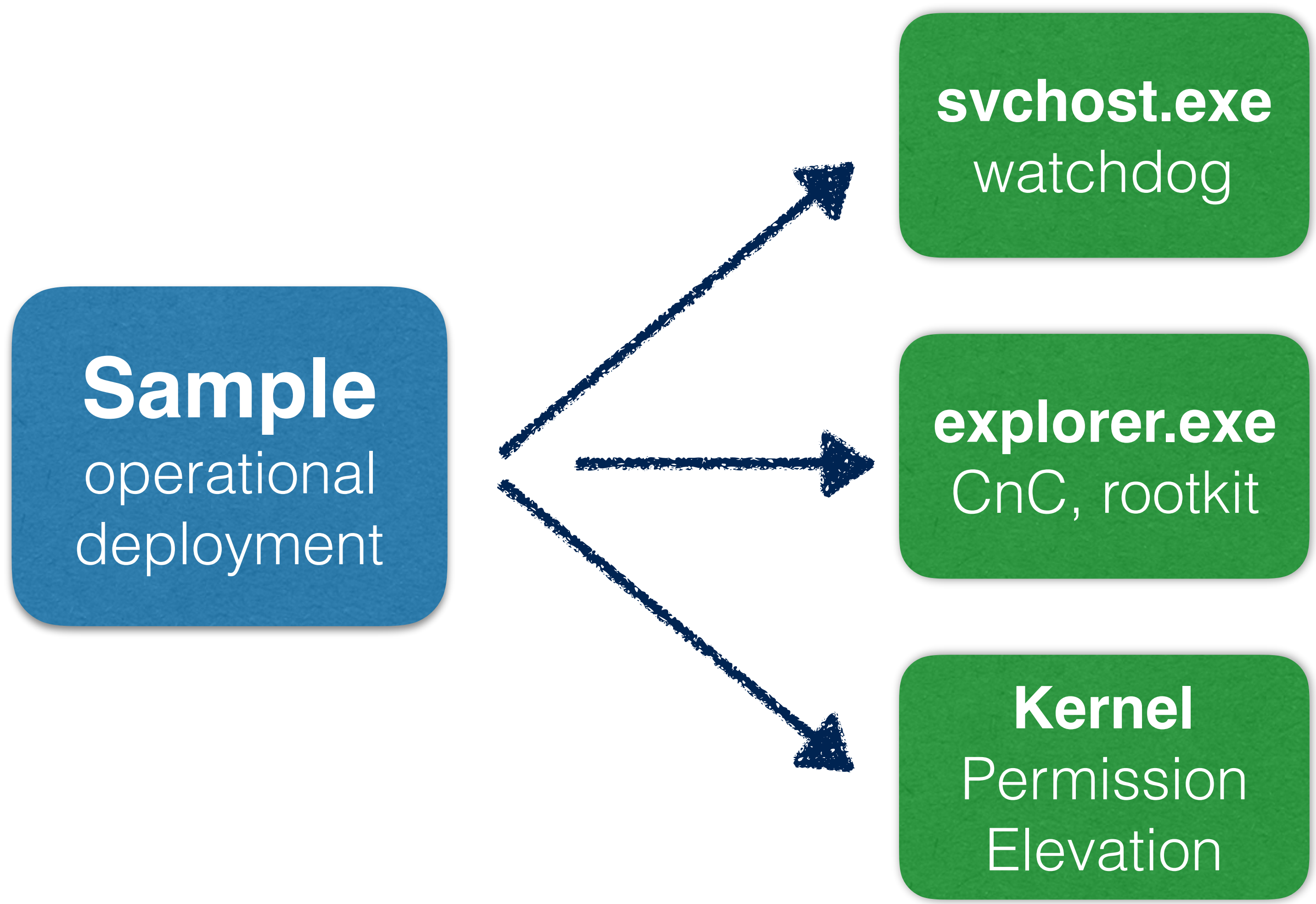
- Anti-VMs
  - API based
    - device enumeration
    - api monitoring detection (cuckoobox hooks)
  - ASM based
    - elapsed time diff

```
74D56F01  8BFF        MOV EDI,EDI
74D56F03  55          PUSH EBP
74D56F04  8BEC        MOV EBP,ESP
74D56F06  83EC 10     SUB ESP,10
74D56F09  56          PUSH ESI
74D56F0A  57          PUSH EDI
```

```
74D56F01 -E9 FA902A98 JMP 0D000000
74D56F06  83EC 10     SUB ESP,10
74D56F09  56          PUSH ESI
74D56F0A  57          PUSH EDI
```

Code dispersion

- Hard to follow - several debug sessions
- Attaching debugger may freeze the UI



Obfuscate at rest

- Encrypt all the things - cfg, code, etc
- Obfuscate API calling or resolve it on each API call
- Own API resolution - use own DLLs copies
- Abuse asm and mix code with data

```
                push    eax
sub_2A577F5     endp ; sp-analysis failed

                call    loc_2A57862
                db  76h ; v
                db  65h ; e
                db  72h ; r
                db  63h ; c
                db  6Ch ; l
                db  73h ; s
                db  69h ; i
                db  64h ; d
                db    0
; ---------------------------------------------------

loc_2A57862:                            ; CODE XREF
                push    eax
                call    dword ptr [ebx+3F6EEA5h]
                test    eax, eax
```

No Run No Fun

A word on code amount

There is a lot of code

- Malware is taken as a serious software project
    - release cycles, test labs, dev teams
    - copy & paste from other malware projects too

Carberp

Branch: master ▾ **malware** / Carberp Botnet / source - absource / pro / **all source** /

Create new file | Upload files | Find file | History

import Carberp Botnet — Latest commit 936a0e4 on Jun 23, 2016

..		
BC	import Carberp Botnet	a year ago
BJWJ	import Carberp Botnet	a year ago
BSS	import Carberp Botnet	a year ago
BinToHex	import Carberp Botnet	a year ago
BlackJoeWhiteJoe	import Carberp Botnet	a year ago
BootkitDropper	import Carberp Botnet	a year ago
Demo_Cur2	import Carberp Botnet	a year ago
Demo_Cur3	import Carberp Botnet	a year ago
Demo_cur	import Carberp Botnet	a year ago
DllLoaderHook	import Carberp Botnet	a year ago
DllLoaderHook1	import Carberp Botnet	a year ago
DropSploit	import Carberp Botnet	a year ago
DropSploit1/src	import Carberp Botnet	a year ago
FakeDllAutorun	import Carberp Botnet	a year ago
GrabberIE_FF	import Carberp Botnet	a year ago
InjectDLL	import Carberp Botnet	a year ago
Locker	import Carberp Botnet	a year ago
Mini	import Carberp Botnet	a year ago
NodInject	import Carberp Botnet	a year ago
OCR	import Carberp Botnet	a year ago

Gozi

Branch: master ▾ **malware** / **windows** / **gozi-isfb** /

Create new file | Upload files | Find file | History

Added Gozi/ISFB Source — Latest commit f76f13e on Mar 23, 2016

..		
AcDll	Added Gozi/ISFB Source	a year ago
BcClient	Added Gozi/ISFB Source	a year ago
Builder	Added Gozi/ISFB Source	a year ago
Common	Added Gozi/ISFB Source	a year ago
Handle	Added Gozi/ISFB Source	a year ago
KeyLog	Added Gozi/ISFB Source	a year ago
Lib32	Added Gozi/ISFB Source	a year ago
Lib64	Added Gozi/ISFB Source	a year ago
RsaKey	Added Gozi/ISFB Source	a year ago
SocksLib	Added Gozi/ISFB Source	a year ago
ZipLib	Added Gozi/ISFB Source	a year ago
apdepack	Added Gozi/ISFB Source	a year ago
client	Added Gozi/ISFB Source	a year ago
crypto	Added Gozi/ISFB Source	a year ago
cryptor	Added Gozi/ISFB Source	a year ago
dname	Added Gozi/ISFB Source	a year ago
release(builder)	Added Gozi/ISFB Source	a year ago
x64/release(builder)	Added Gozi/ISFB Source	a year ago
zconv	Added Gozi/ISFB Source	a year ago
Config.exe	Added Gozi/ISFB Source	a year ago

There is a lot of code (cont)

- Culminates in large codebase over time
- Takes substantial amount of time to analyze

Time is Money

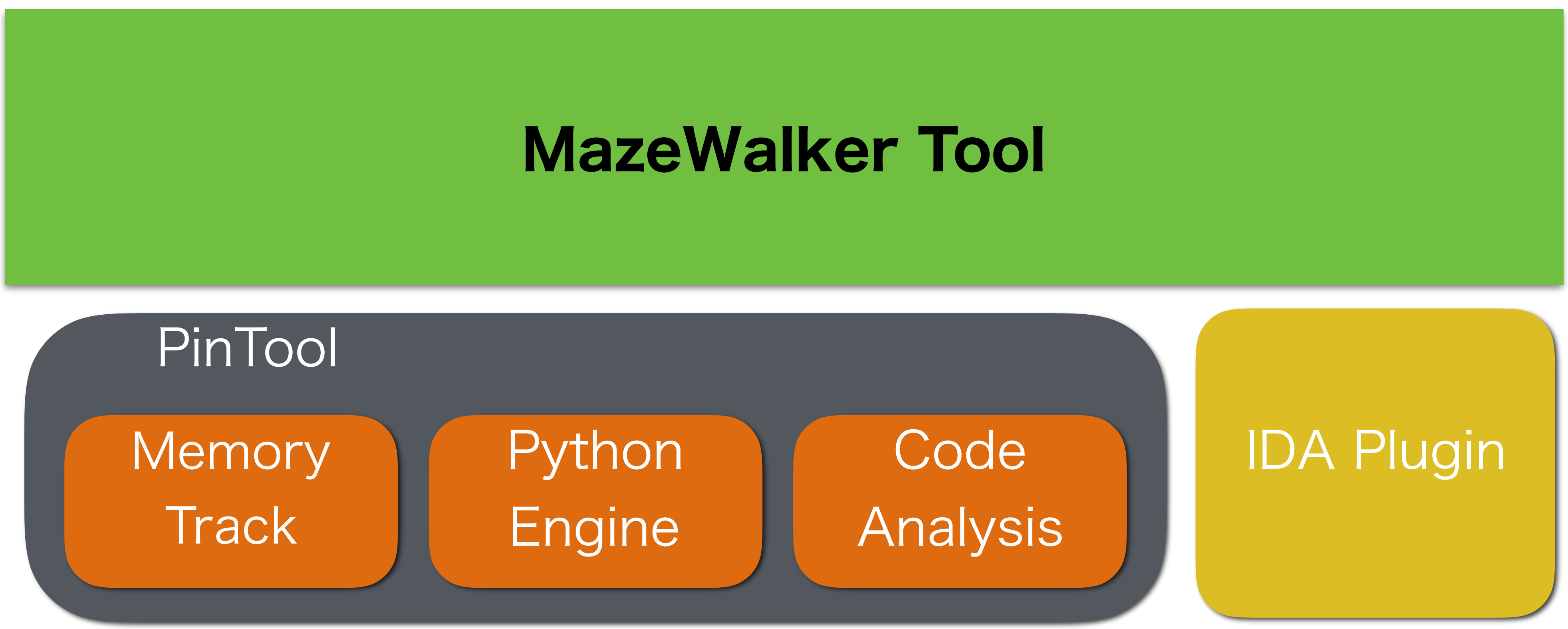
both are at most insufficient

Ideas behind MazeWalker

MazeWalker - Main Ideas

- It must save time !!!!
- Maximize time spent in IDA vs time in Debugger
- Work with non modified VMs
- Retrieve all runtime info and push into IDA
- Help with overall malware understanding
    - dig into asm on an interest - basis
    - enable research focusing

Architecture

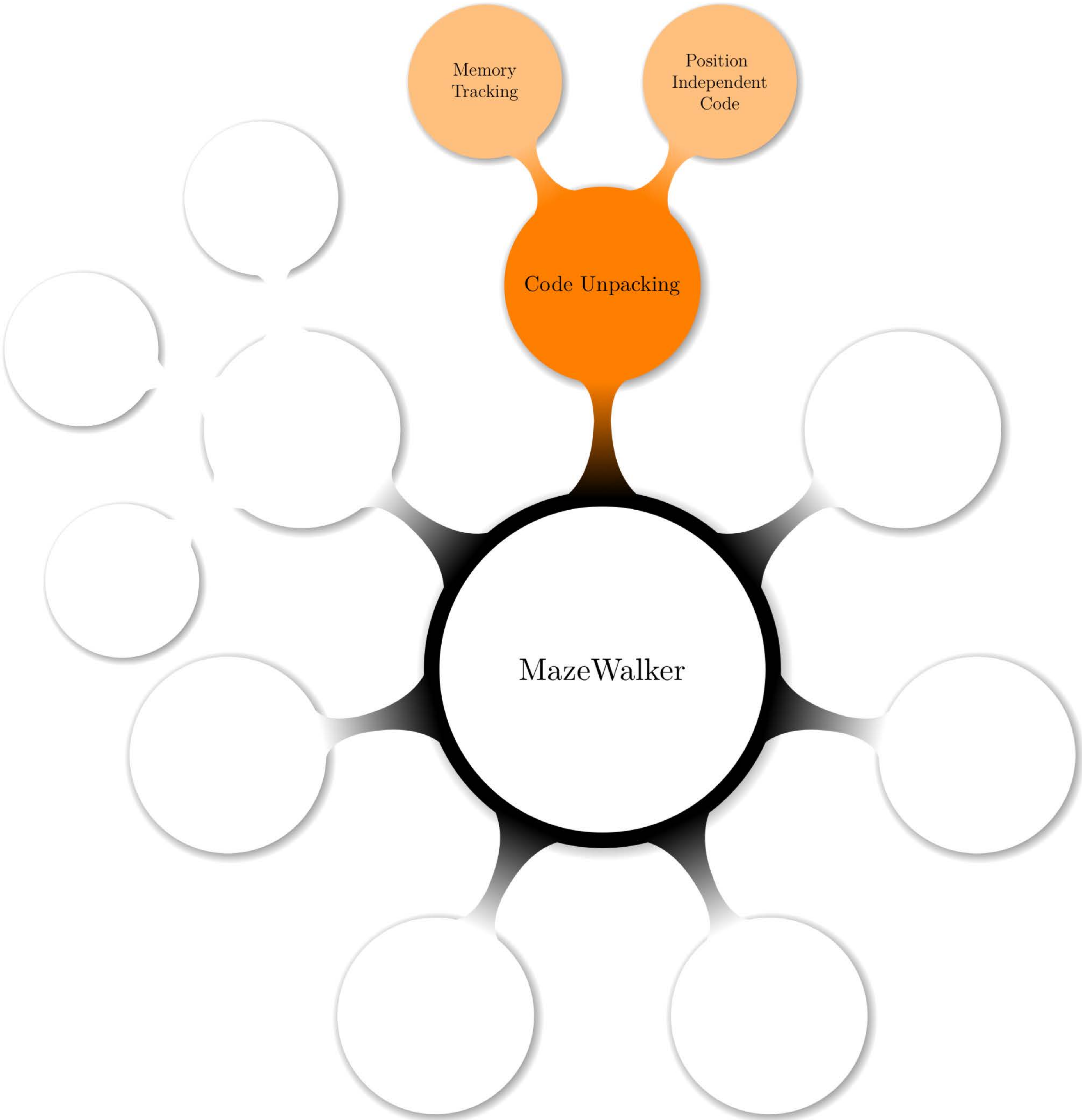


Intel’s Pin Framework

Pin is a dynamic binary instrumentation framework for the IA-32, x86-64 and MIC instruction-set architectures that enables the creation of dynamic program analysis tools.

- VM in essence
- Multi-platform

- Callbacks on everything
    - instructions
    - API calls
    - Image loading
    - Threads, Exceptions
    - memory read/writes


Memory Tracking
Position Independent Code
Code Unpacking
MazeWalker

Code unpacking - memory

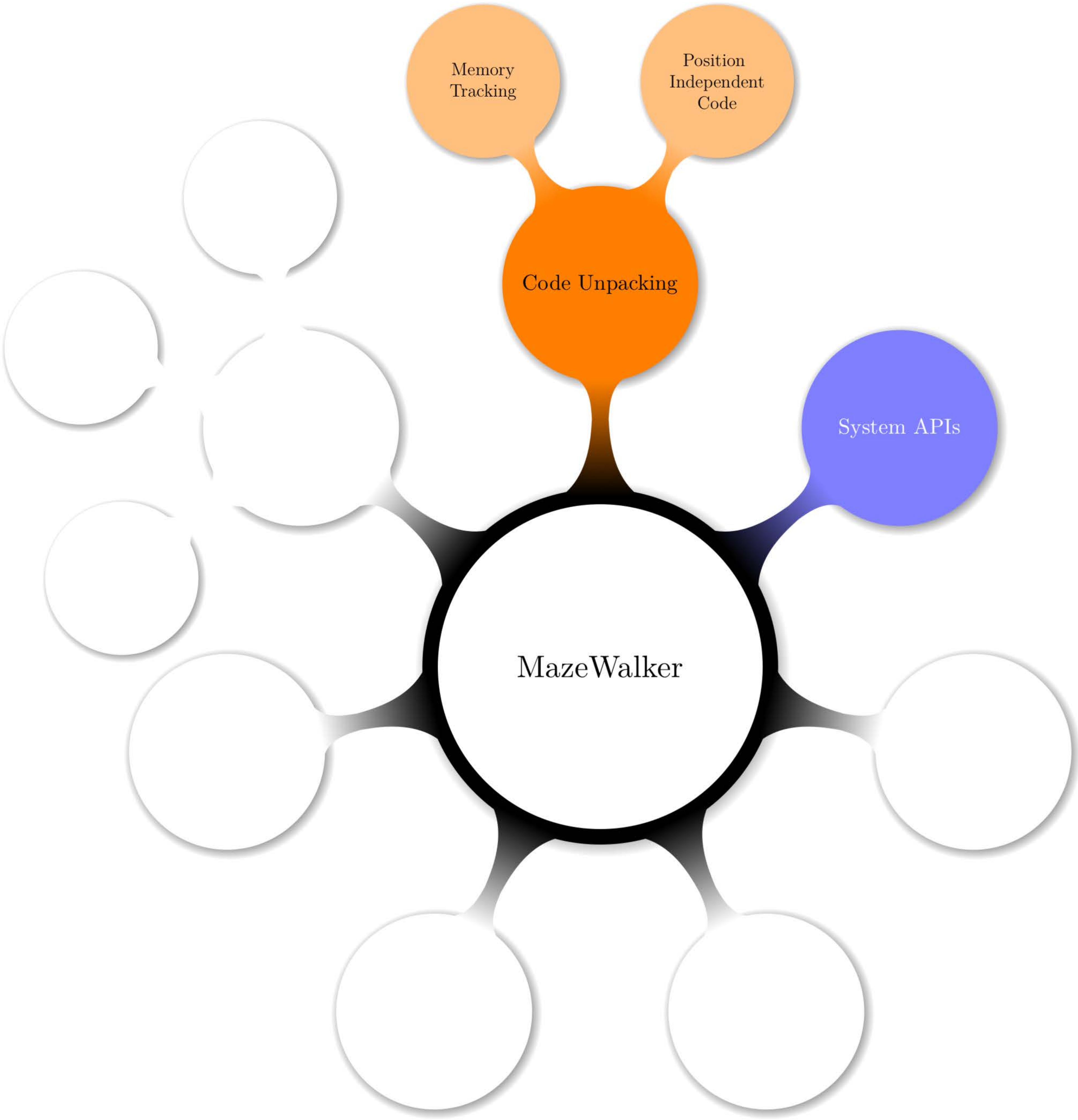
- Rely on allocated page analysis
- Tracks all executed memory by comparing executing BBL to older copy
  - detect new PEs
  - identify known (dynamically) loaded DLLs


```
"whitelist": {
  "imphash": [
    {
      "name": "wow64cpu.dll",
      "hash": "99760ef4e9760fe74c20aa23cc71b9b6"
    },
    {
      "name": "kernel32.dll",
      "hash": "51d53c5eba00dd0eb29d977440ba62d9"
    },
    {
      "name": "cryptsp.dll",
      "hash": "ebc7b47d85441b0f3dce38e782316e8c"
    },
    {
      "name": "advapi32.dll",
      "hash": "56357721dfdf00b68c7be9d465e71475"
    }
  ],
  "exphash": [
    {
      "name": "ntdll.dll",
      "hash": "302ce8c1fc2c0c08531dd6637cd5e81f"
    },
    {
      "name": "ntdll.dll",
      "hash": "4a40a87fd83beb5f83fdd4e5be70262e"
    }
  ],
  "path": [
    "C:\\Windows\\"
  ],
```

Code unpacking - PiC

- Pin helps to do Call/Jump site analysis
- Logging call-site <-> target pair

```
call    eax             ; GetProcAddress
inc     ecx
call    eax             ; RtlDecompressBuffer
xchg    ecx, edx
neg     edx
jmp     short loc_2D531F3
```


Memory Tracking
Position Independent Code
Code Unpacking
System APIs
MazeWalker

System API monitoring

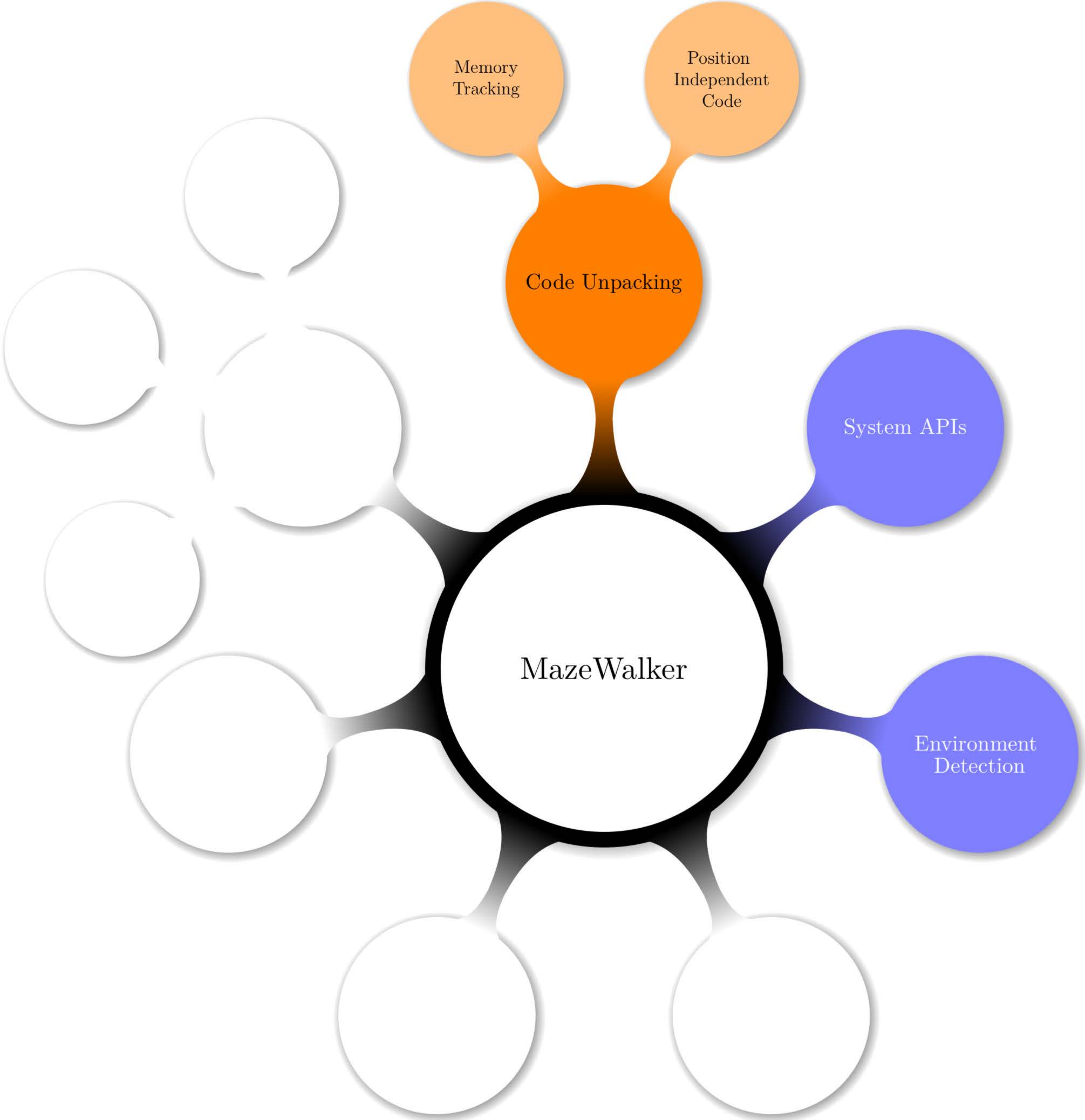
- Pin’s Routine Objects
    - Harder to detect
- Configurable
- API Agnostic monitor interface
- Scriptable

```
"apis": [
  {
    "name": "SetupDiGetDeviceRegistryPropertyA",
    "lib": "setupapi",
    "num": 7,
    "pre_parser": "",
    "post_parser": "post_antivm1"
  },
  {
    "name": "wsprintfA",
    "lib": "user32",
    "num": 2,
    "pre_parser": "",
    "post_parser": "post_wsprintfa"
  },
```

+

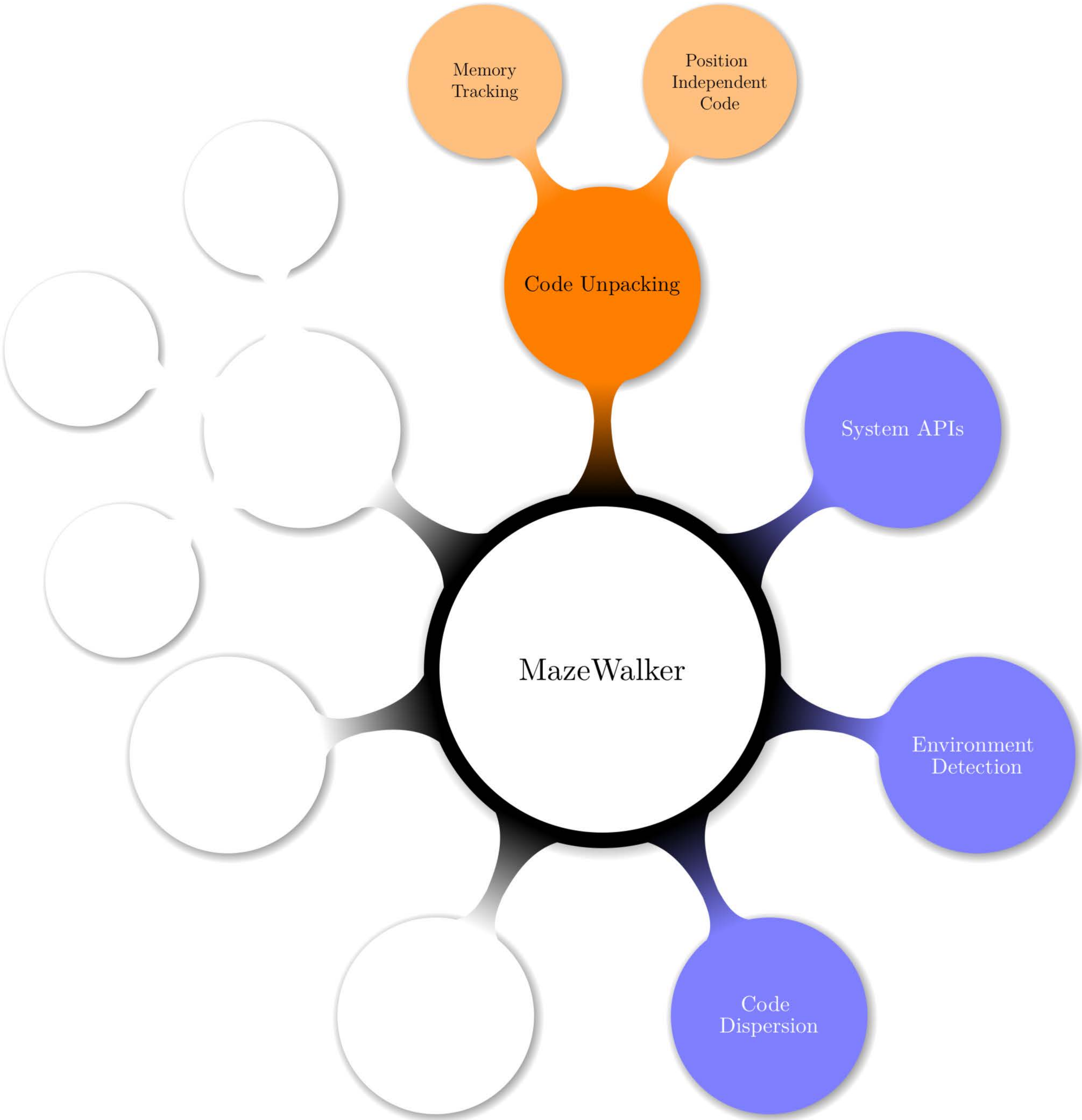
System APIs - CreateFileW

```
import ctypes
import json

def pre_analyzer(LPCTSTR_lpFileName,
                 DWORD_dwDesiredAccess,
                 DWORD_dwShareMode,
                 LPSECURITY_ATTRIBUTES_lpSecurityAttributes,
                 DWORD_dwCreationDisposition,
                 DWORD_dwFlagsAndAttributes,
                 HANDLE_hTemplateFile,
                 **kwargs):

    FileName = ctypes.c_wchar_p.from_address(LPCTSTR_lpFileName)
    res = []
    if (FileName and FileName.value):
        result = {'name': 'lpFileName', 'data': FileName.value}
        res.append(result)
    return json.dumps(res)
```


Memory Tracking
Position Independent Code
Code Unpacking
System APIs
MazeWalker
Environment Detection

Environment Detection

```
#SetupDiGetDeviceRegistryPropertyA
def post_analyzer(HDEVINFO_DeviceInfoSet,
                  PSP_DEVINFO_DATA_DeviceInfoData,
                  pProperty,
                  PDWORD_PropertyRegDataType,
                  PBYTE_PropertyBuffer,
                  pPropertyBufferSize,
                  PDWORD_RequiredSize,
                  **kwargs):

    Property = ctypes.c_ulong.from_address(pProperty)
    if (Property.value == 0xC):
        PropertyBufferSize = ctypes.c_ulong.from_address(pPropertyBufferSize)
        if (PropertyBufferSize.value > 0):
            res = []
            pPropertyBuffer = ctypes.c_ulong.from_address(PBYTE_PropertyBuffer)
            PropertyBuffer = ctypes.cast(pPropertyBuffer.value, ctypes.c_char_p)
            buffer = (c_char * PropertyBufferSize.value).from_address(pPropertyBuffer.value)

            res.append({'name': 'PropertyBufferSize', 'data': PropertyBufferSize.value})
            res.append({'name': 'original_PropertyBuffer', 'data': PropertyBuffer.value})

            replace_string(buffer, PropertyBuffer, ['vmware', 'virtual'], [b'NewTek', b'Digital'])

            res.append({'name': 'fixed_PropertyBuffer', 'data': PropertyBuffer.value})

            return json.dumps(res)
    return None
```



Memory Tracking
Position Independent Code
Code Unpacking
System APIs
MazeWalker
Environment Detection
Code Dispersion

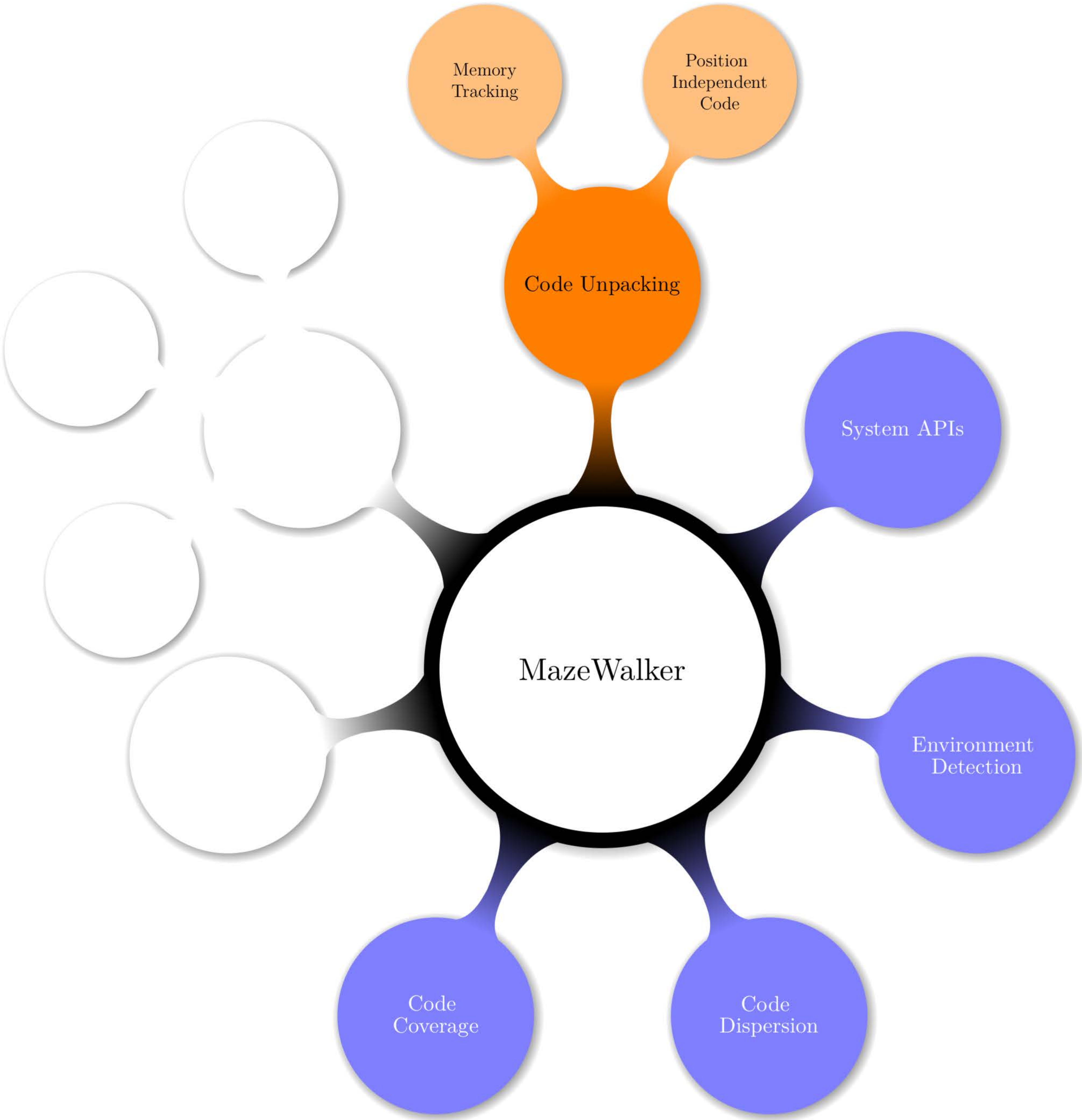
Code dispersion

- Use scriptable APIs monitoring for code injection tracking
  - this helps Pin to find target process
- Use Pin's existing ability to track child processes

Code dispersion

```python
import ctypes
import json
import os
import subprocess

def pre_analyzer(DWORD_dwDesiredAccess,
                 BOOL_bInheritHandle,
                 DWORD_dwProcessId,
                 **kwargs):

    pid = ctypes.c_int.from_address(DWORD_dwProcessId)
    if (pid and pid.value and os.getpid() != pid.value):
        if "pin_dir" in kwargs and "out_dir" in kwargs:
            process = subprocess.Popen(kwargs["pin_dir"] +
                                       "/pin.exe -unique_logfile -pid " + str(pid.value) +
                                       " -t " + kwargs["pin_dir"] + "/MazeWalker.dll -cfg " +
                                       kwargs["pin_dir"] + "/config.json" +
                                       " -out " + kwargs["out_dir"] + " -unique_logfile")
        res = []
        result = {'name': 'dwProcessId', 'data': pid.value}
        res.append(result)
        return json.dumps(res)
```

OpenProcess API

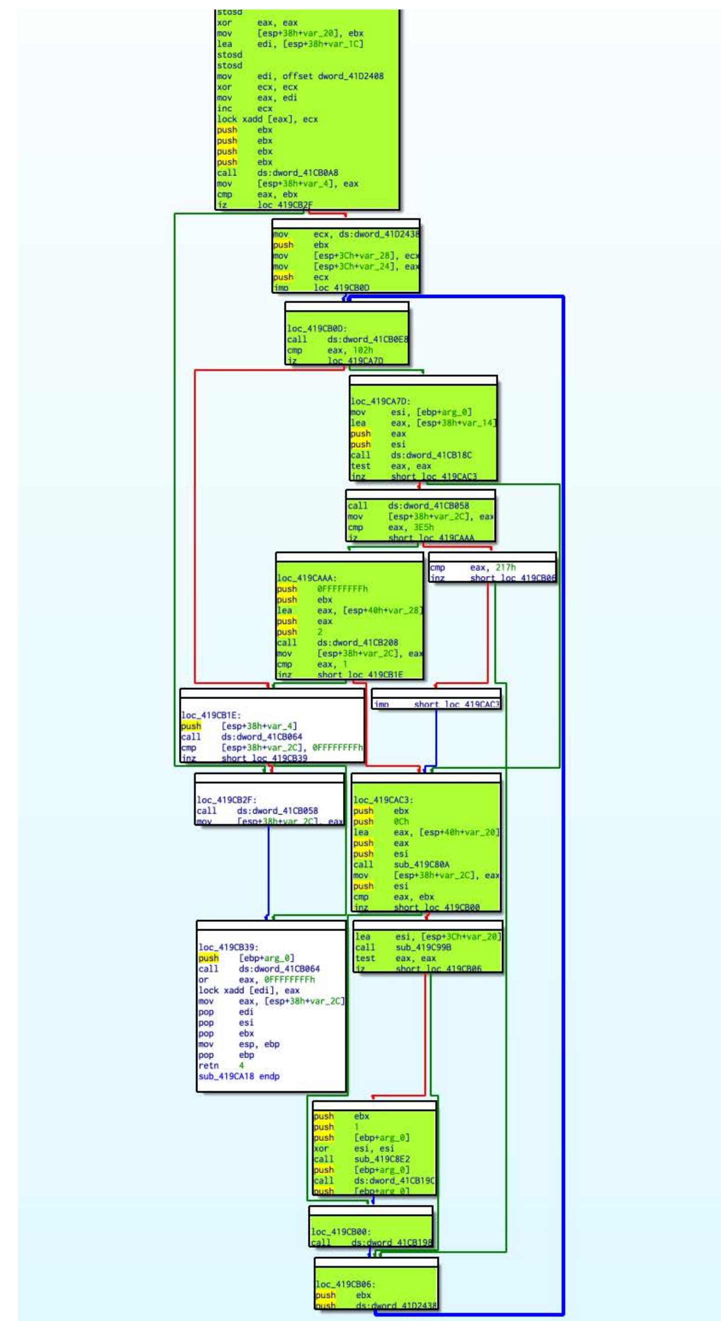

MazeWalker
Code Unpacking
Memory Tracking
Position Independent Code
System APIs
Environment Detection
Code Dispersion
Code Coverage

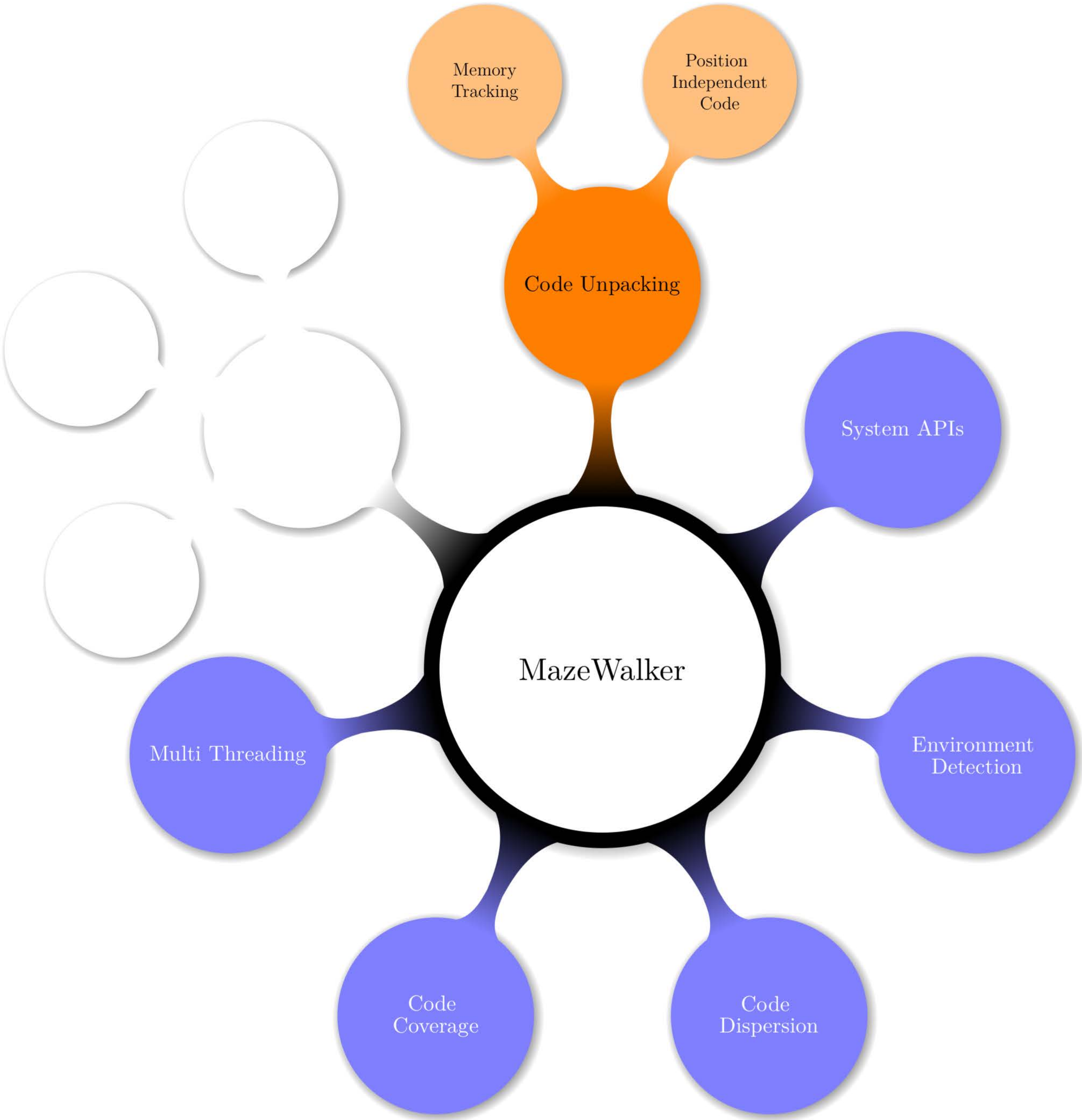
Control Flow Graph

With PIN's BBL callbacks

Covers all memory regions

Covers across different processes

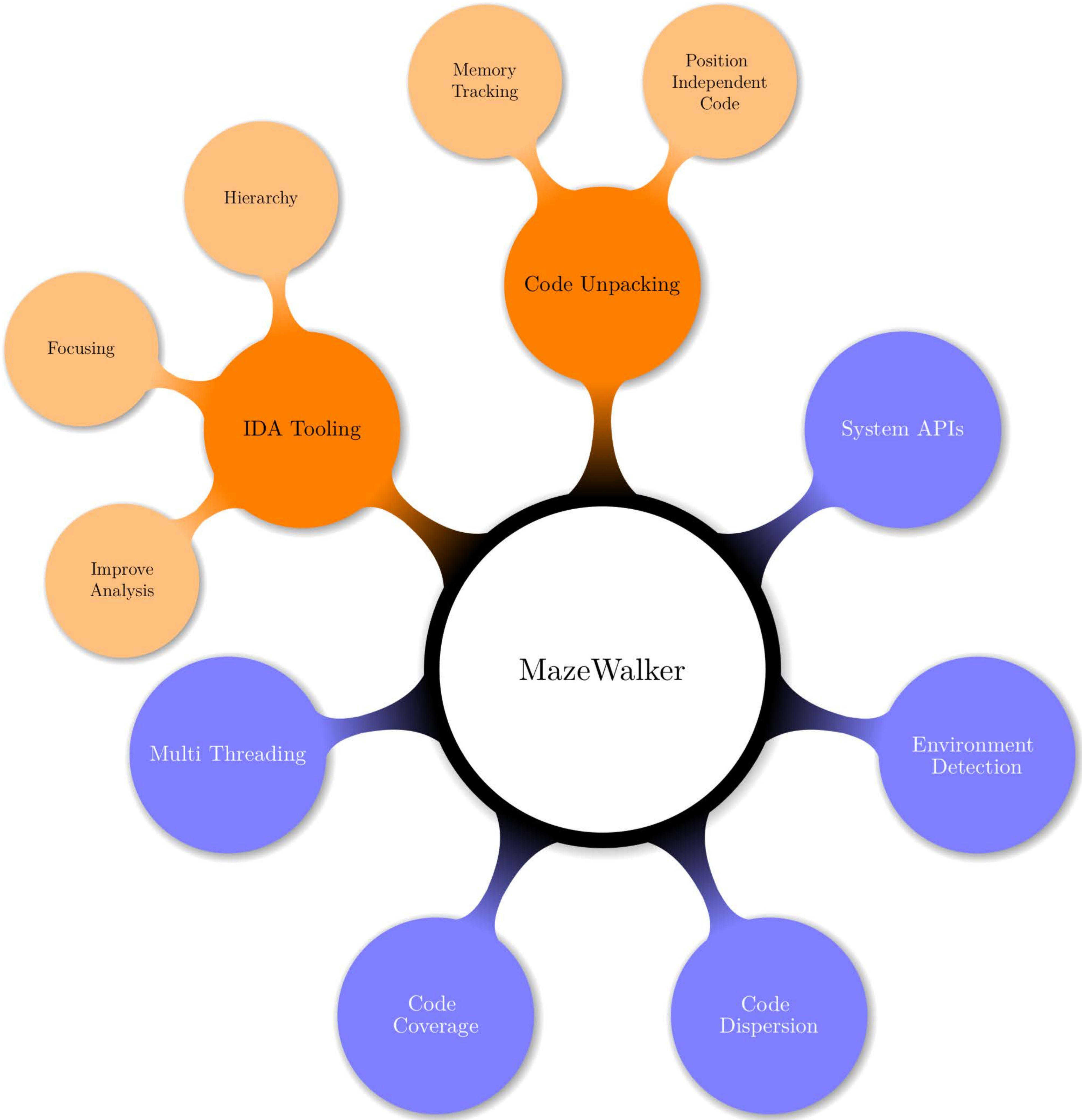




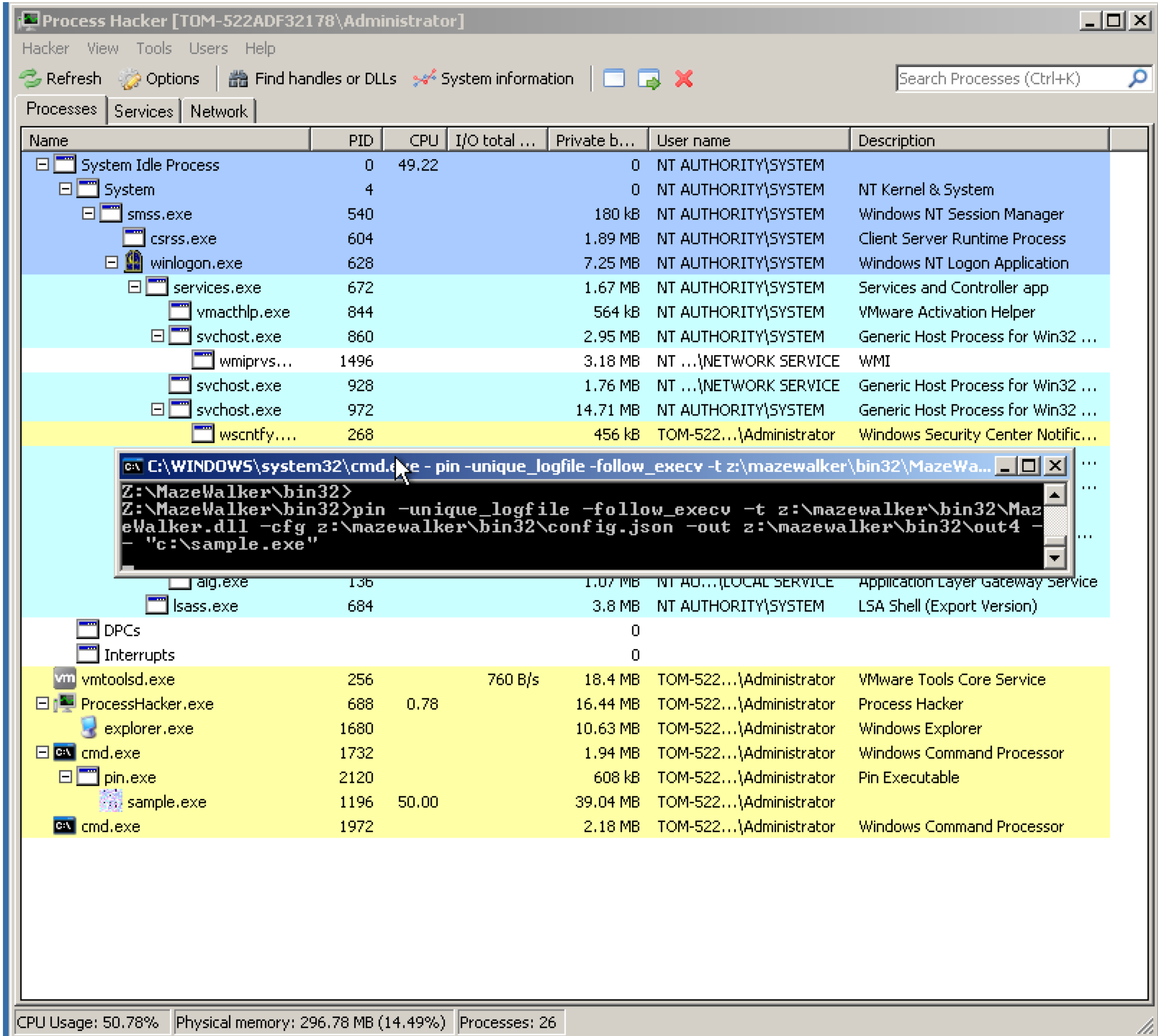
Memory Tracking
Position Independent Code
Code Unpacking
System APIs
MazeWalker
Environment Detection
Multi Threading
Code Coverage
Code Dispersion

Threads everywhere

All execution metadata is on thread basis


MazeWalker
IDA Tooling
Hierarchy
Focusing
Improve Analysis
Code Unpacking
Memory Tracking
Position Independent Code
System APIs
Environment Detection
Code Dispersion
Code Coverage
Multi Threading

Demo


Process Hacker [TOM-522ADF32178\Administrator]
Hacker View Tools Users Help
Refresh Options Find handles or DLLs System information
Search Processes (Ctrl+K)
Processes Services Network
Name PID CPU I/O total ... Private b... User name Description
System Idle Process 0 49.22 0 NT AUTHORITY\SYSTEM
System 4 0 NT AUTHORITY\SYSTEM NT Kernel & System
smss.exe 540 180 kB NT AUTHORITY\SYSTEM Windows NT Session Manager
csrss.exe 604 1.89 MB NT AUTHORITY\SYSTEM Client Server Runtime Process
winlogon.exe 628 7.25 MB NT AUTHORITY\SYSTEM Windows NT Logon Application
services.exe 672 1.67 MB NT AUTHORITY\SYSTEM Services and Controller app
vmacthlp.exe 844 564 kB NT AUTHORITY\SYSTEM VMware Activation Helper
svchost.exe 860 2.95 MB NT AUTHORITY\SYSTEM Generic Host Process for Win32 ...
wmiprvs... 1496 3.18 MB NT ...\NETWORK SERVICE WMI
svchost.exe 928 1.76 MB NT ...\NETWORK SERVICE Generic Host Process for Win32 ...
svchost.exe 972 14.71 MB NT AUTHORITY\SYSTEM Generic Host Process for Win32 ...
wscntfy.... 268 456 kB TOM-522...\Administrator Windows Security Center Notific...
C:\WINDOWS\system32\cmd.exe - pin -unique_logfile -follow_execv -t z:\mazewalker\bin32\MazeWa...
Z:\MazeWalker\bin32>
Z:\MazeWalker\bin32>pin -unique_logfile -follow_execv -t z:\mazewalker\bin32\MazeWalker.dll -cfg z:\mazewalker\bin32\config.json -out z:\mazewalker\bin32\out4 -- "c:\sample.exe"
lsass.exe 684 3.8 MB NT AUTHORITY\SYSTEM LSA Shell (Export Version)
DPCs 0
Interrupts 0
vmtoolsd.exe 256 760 B/s 18.4 MB TOM-522...\Administrator VMware Tools Core Service
ProcessHacker.exe 688 0.78 16.44 MB TOM-522...\Administrator Process Hacker
explorer.exe 1680 10.63 MB TOM-522...\Administrator Windows Explorer
cmd.exe 1732 1.94 MB TOM-522...\Administrator Windows Command Processor
pin.exe 2120 608 kB TOM-522...\Administrator Pin Executable
sample.exe 1196 50.00 39.04 MB TOM-522...\Administrator
cmd.exe 1972 2.18 MB TOM-522...\Administrator Windows Command Processor
CPU Usage: 50.78% Physical memory: 296.78 MB (14.49%) Processes: 26

Collected Data

Name
0_1196_sample.exe
0_400000_2eb448_1196.mz
1_2d50000_a000_1196.mem
2_2d50000_a000_1196.mem
3_400000_66000_1196.mz
maze_walk_1196.json
1_396_dgsesock.exe
0_400000_2eb448_396.mz
1_2d50000_a000_396.mem
2_2d50000_a000_396.mem
3_400000_66000_396.mz
396_0x7c96149b_0x4.mem
396_0x4250000_0x318.mem
maze_walk_396.json
2_1680_explorer.exe
0_14c0000_10000_1680.mem
1_4190000_90000_1680.mem
2_2240000_1000_1680.mem
3_4250000_1000_1680.mem
maze_walk_1680.json


```
[
    {
        "process": {
            "name": "",
            "pid": 1196,
            "threads_num": 2,
            "threads": [
                {
                    "tid": 0,
                    "tfunc": 4572402,
                    "bbls": [
                    ],
                    "calls": [
                    ],
                    "api_parameters": [
                    ]
                },
                {
                }
            ],
            "mem_areas": [
                {
                    "id": 0,
                    "start": 4194304,
                    "end": 7255112,
                    "size": 3060808,
                    "entry": 4572402,
                    "tids": [
                        0
                    ]
                },
```

Hierarchy matters

- sub_401000
- start
- sub_40112E
- sub_401143
- sub_401158
- sub_40125A
- sub_4014EC
- sub_4015D4
- sub_4016C0
- sub_401748
- sub_401824
- sub_4018ED
- sub_401938
- sub_401A18
- sub_401B16
- sub_401B61
- sub_401EE9
- sub_401F2D
- sub_401FE5
- sub_4020D4
- sub_40216F
- sub_40256A

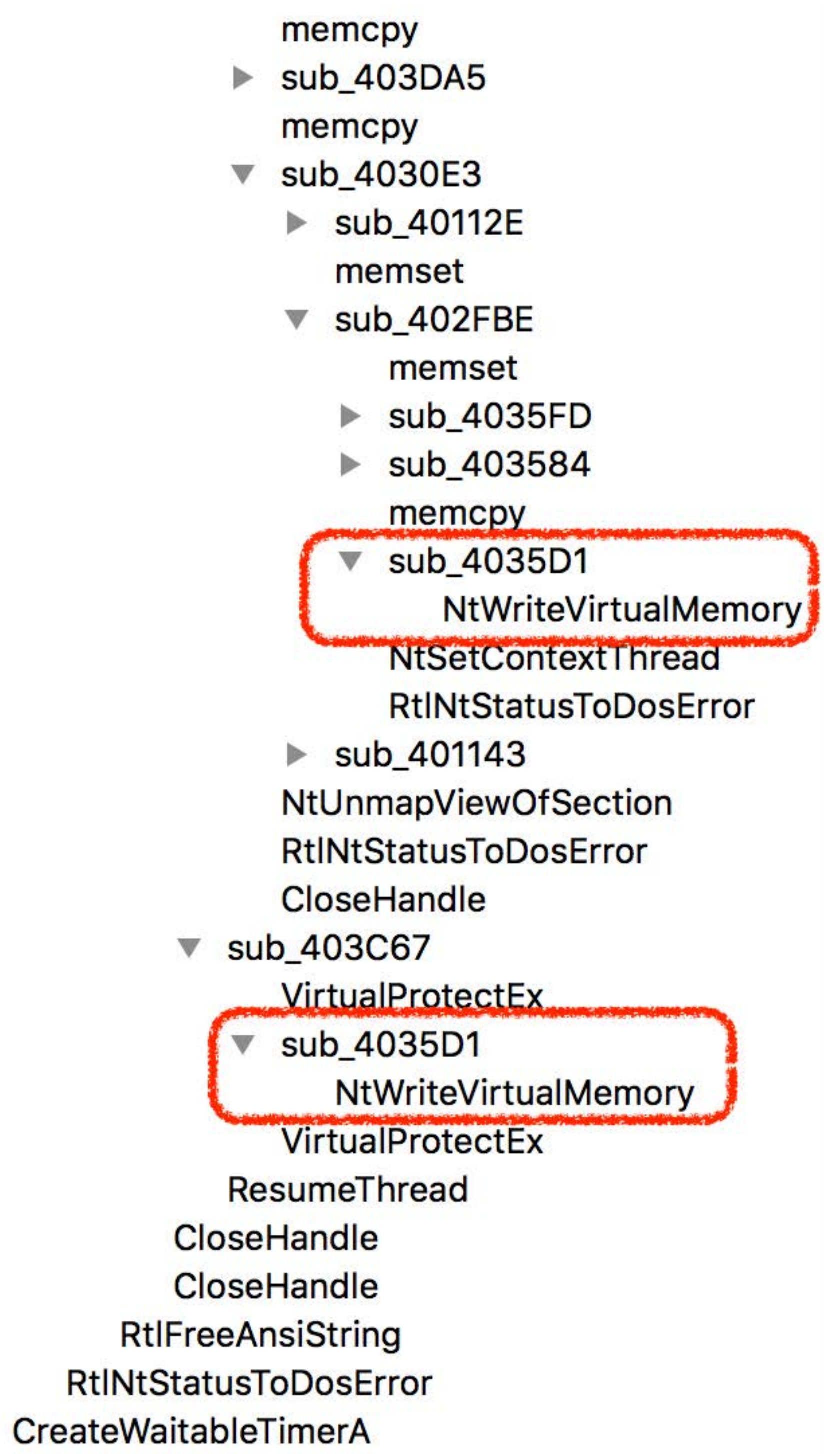
Original IDA

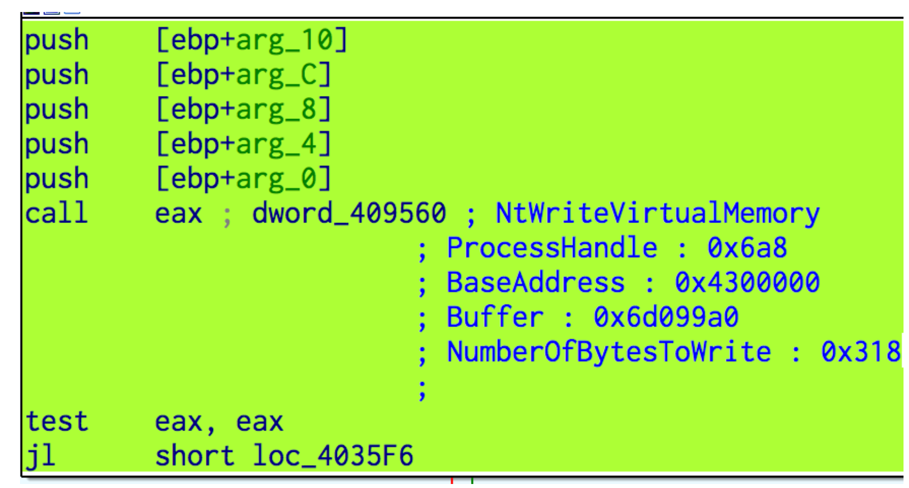
Navigate
the execution flow

- start
  - HeapCreate
  - GetModuleHandleA
  - GetCommandLineW
  - sub_40216F
    - GetModuleHandleA
    - SwitchToThread
    - sub_40375F
    - sub_40256A
    - sub_402A1E
    - CreateFileA
    - sub_401000
    - sub_401EE9
    - sub_4039EB
    - sub_4014EC
    - GetModuleHandleA
    - sub_403AB5
    - sub_4015D4
    - ConvertStringSecurityDescriptor
    - CreateEventA
    - RtlGetLastWin32Error
    - CloseHandle
    - sub_40364D
    - StrChrW
    - StrChrW
    - sub_4016C0
    - sub_40350E
      - sub_40112E
      - NtQuerySystemInformation

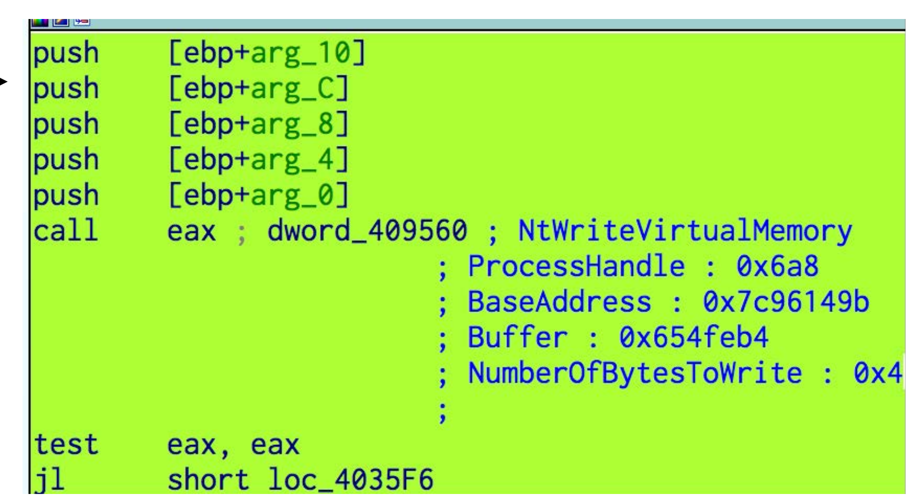
Maze Walker

Hierarchy matters

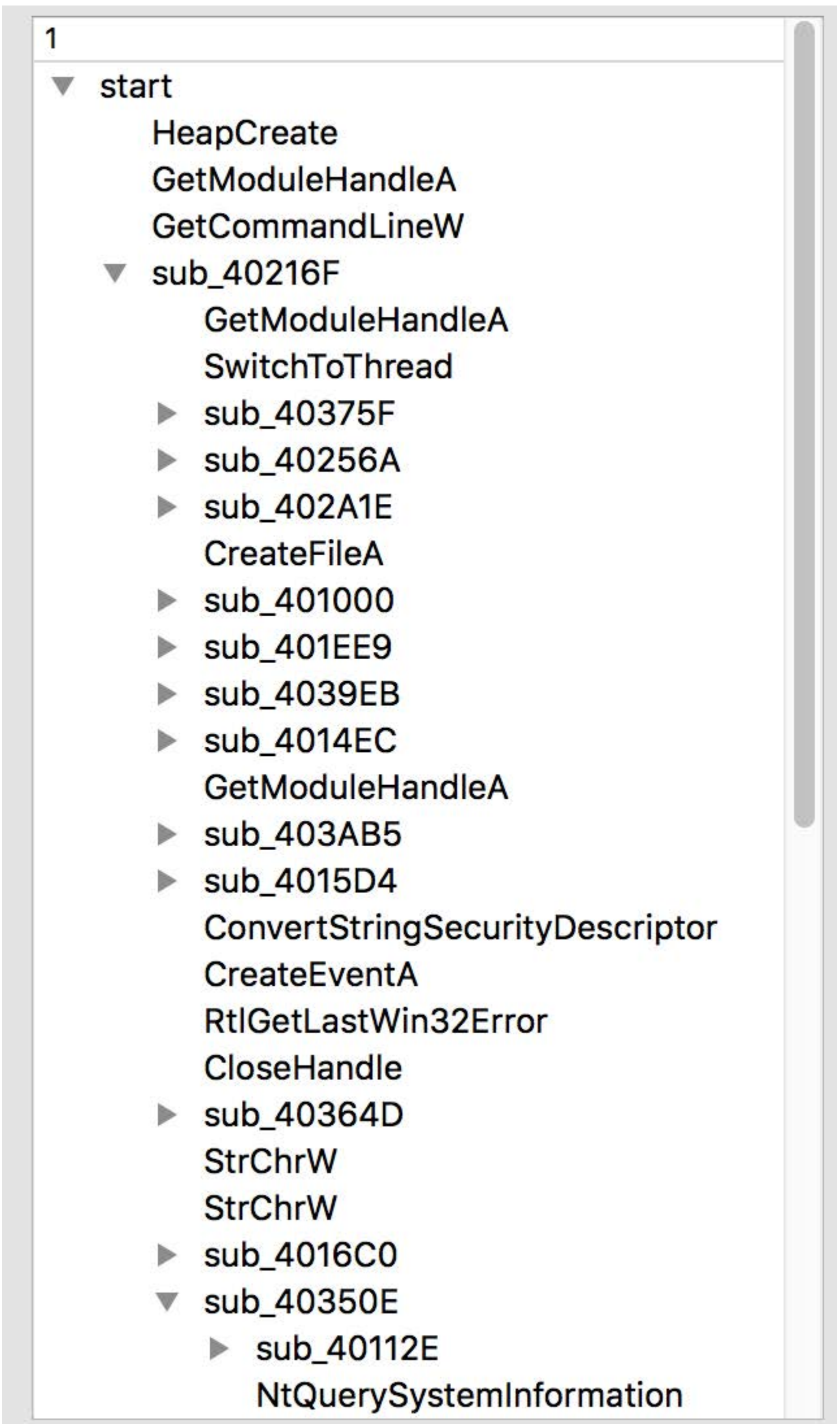




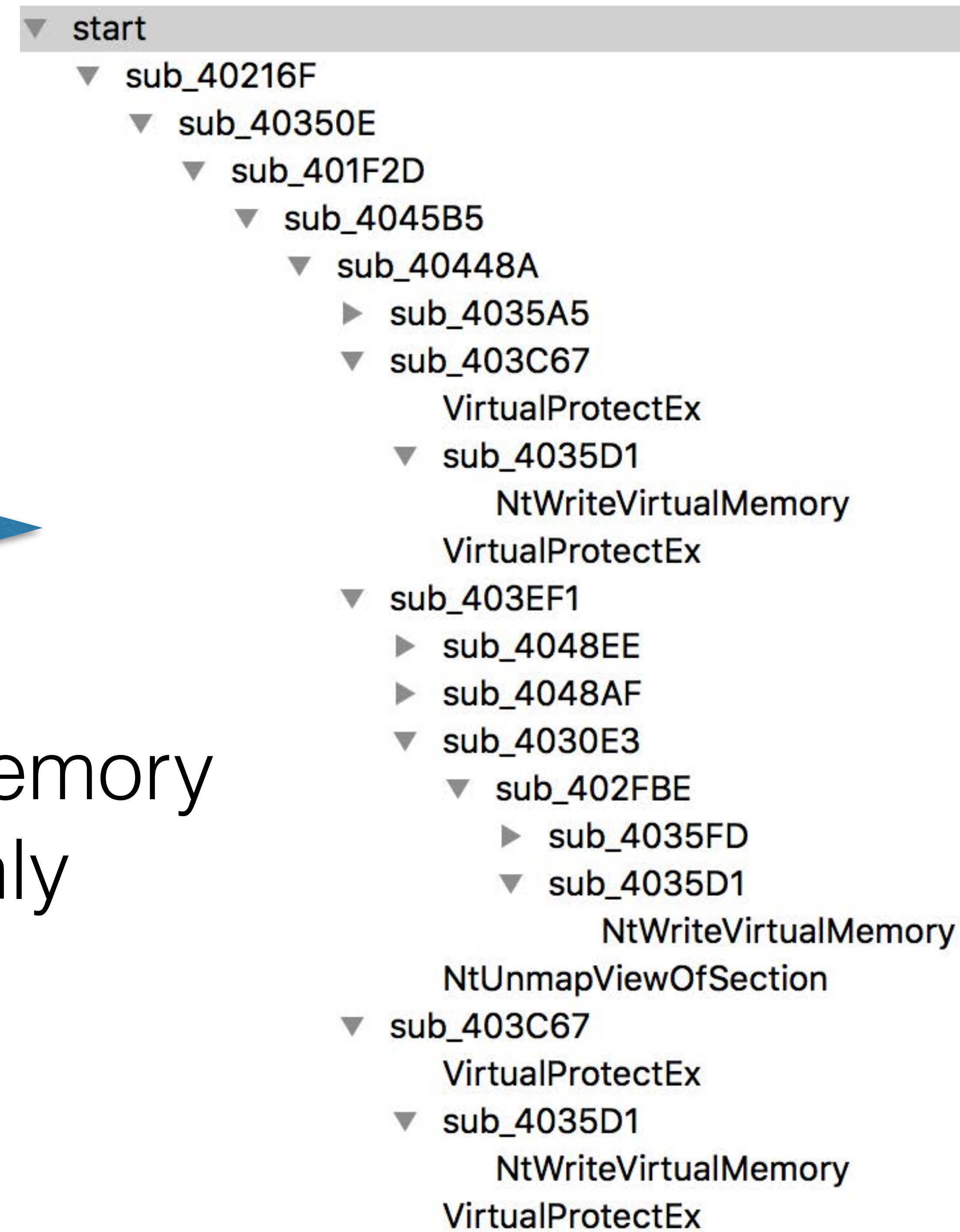
Wrapped functions get different meaning with context



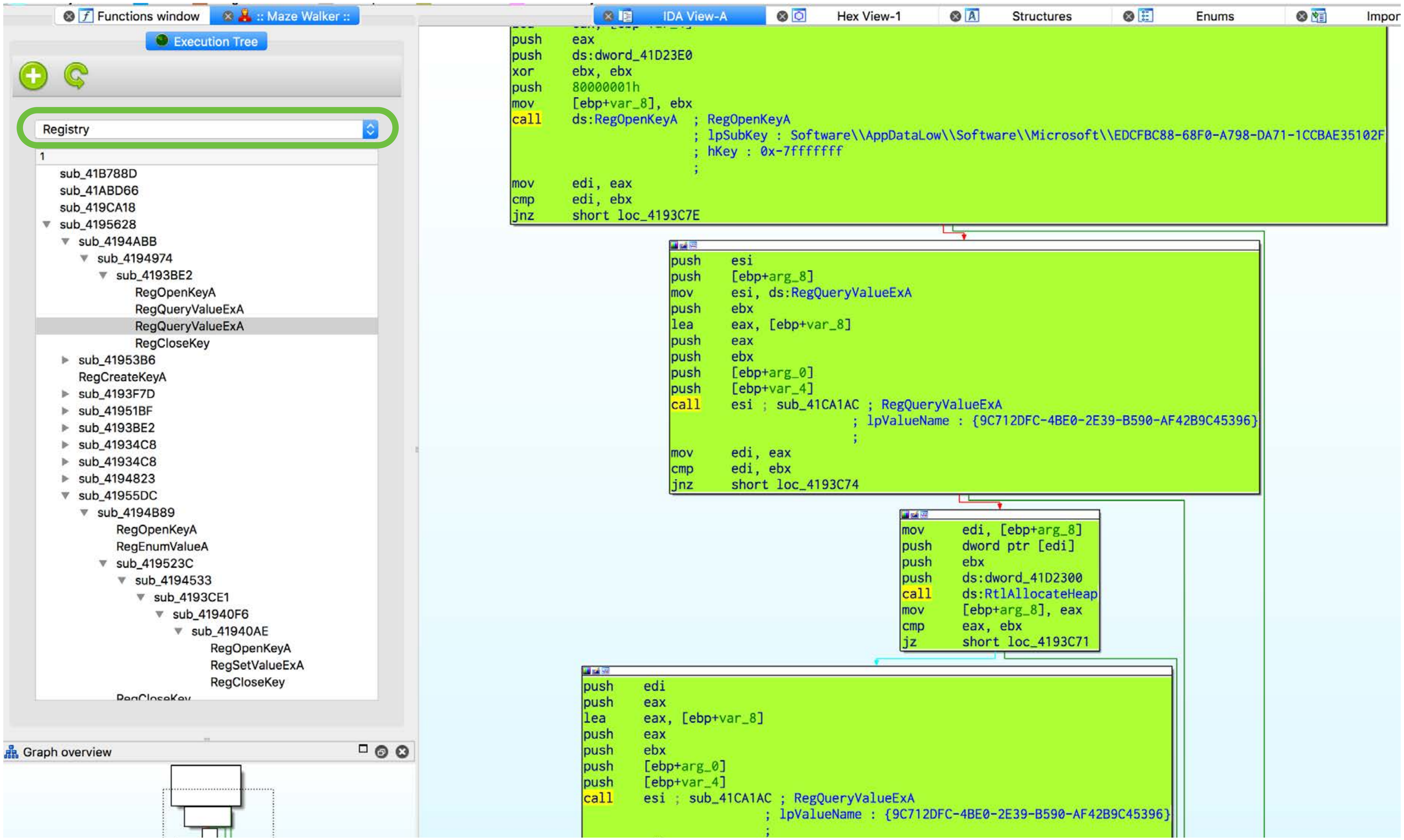
Focus



Work on Memory Part Only



Focus



Focussing on Registry only

ToDo…

Further development

- Stability and Memory consumption reduction
- Memory dumps consolidation
- Custom IDA Loader
- “Maze Walk” in kernel space
- Implement anti-instrumentation prevention logic
    - Dynamic Binary Instrumentation Frameworks: I know you're there spying on me (ReCon 2012)

https://github.com/0xPhoeniX/MazeWalker.git

Thank you!

@p_h_0_e_n_i_x