Dr. Pebber

Faking Windows Structures for ESIL to parse

A broken Emotet sample: My Inspiration

- Need to pull malicious domains from sample
- Sample segfaults early before it calls out at all
- Crashes trying to write to a non-writable address (0x012xxxxx)
- Lots of global pointer function calls, in the same non-writeable memmory (`call [0x012xxxxx]`)
- Empty Import table, no cross-refs to open, connect,
- How do I damask those global function pointers?

Shellcode: Same problem

- No import table
- Windows lacks a reliable syscall table
- Shelcode needs to find Win API functions (usually)
- To the PEB (Process Environment Block)!

Finding Win API Functions (32 bit)

- Grab PEB from fs:[0x30]
- Dereference PEB_LDR_DATA (PEB+0xc)
- More dereferencing, walking links, etc
- Hash lookup function names
- Profit

Back to Emotet

- Static analysis => duplicating hashing alg
- dynamic => fixing binary (probably best solution)
- emulate in ESIL => need windos structures

Introducing Dr. Pebbers

- All the needed windows structures in a minmized binary blob
- https://github.com/swoops/dr_pebber
- Demo

Potential Uses

- Annotated Output to Learn Windows Structure
- Avoid starting a Windows VM to reverse the discovery of Win API functions
- Reverse shellcode outside of any executable
- Reverse second stage payloads in restricted environments
- Automation: r2 + dr_pebber + docker
- You control the functions, you can make them do anything
- No OS to touch the debugger flag
- Whatever you guys come up with!

Drawbacks

- ESIL emulation is slow compared to a debugger
- Currently 32 bit only
- Discrepancies, like distance between sections
- Lazy loading API calls, like the GOT table

Thanks

- Hurricane Labs
- My Wife