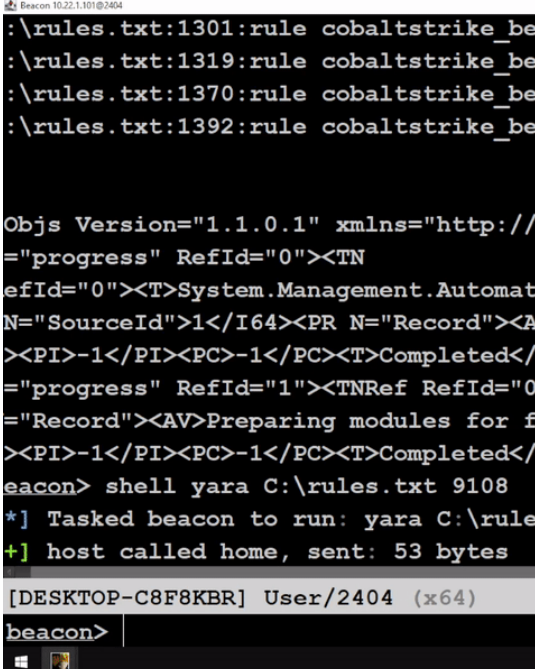


---

## AceLdr - Avoid Memory Scanners

A position-independent reflective loader for Cobalt Strike. Zero results from Hunt-Sleeping-Beacons,



```
Beacon 10.22.1.101@2404
:\rules.txt:1301:rule cobaltstrike_be
:\rules.txt:1319:rule cobaltstrike_be
:\rules.txt:1370:rule cobaltstrike_be
:\rules.txt:1392:rule cobaltstrike_be

Obj's Version="1.1.0.1" xmlns="http://
="progress" RefId="0"><TN
efId="0"><T>System.Management.Automat
N="SourceId">1</I64><PR N="Record"><A
><PI>-1</PI><PC>-1</PC><T>Completed</
="progress" RefId="1"><TNRef RefId="0
="Record"><AV>Preparing modules for f
><PI>-1</PI><PC>-1</PC><T>Completed</
eacon> shell yara C:\rules.txt 9108
*] Tasked beacon to run: yara C:\rule
+] host called home, sent: 53 bytes

[DESKTOP-C8F8KBR] User/2404 (x64)
beacon>
```

BeaconHunter, BeaconEye, Patriot, Moneta, PE-sieve, or MalMemDetect.

### Features

**Easy to Use** Import a single CNA script before generating shellcode.

**Dynamic Memory Encryption** Creates a new heap for any allocations from Beacon and encrypts entries before sleep.

**Code Obfuscation and Encryption** Changes the memory containing CS executable code to non-executable and encrypts it (FOLIAGE).

**Return Address Spoofing at Execution** Certain WinAPI calls are executed with a spoofed return address (InternetConnectA, NtWaitForSingleObject, RtlAllocateHeap).

**Sleep Without Sleep** Delayed execution using WaitForSingleObjectEx.

**RC4 Encryption** All encryption performed with SystemFunction032.

---

## Known Issues

- Not compatible with loaders that rely on the shellcode thread staying alive.

## References

This project would not have been possible without the following: - FOLIAGE - x64 return address spoofing (source + explanation)

Other features and inspiration were taken from the following: - <https://www.arashparsa.com/bypassing-pesieve-and-moneta-the-easiest-way-i-could-find/> - <https://github.com/secidiot/TitanLdr> - <https://github.com/JLospinoso/gargoyle> - <https://www.forrest-orr.net/post/masking-malicious-memory-artifacts-part-ii-insights-from-moneta> - <https://www.arashparsa.com/hook-heaps-and-live-free/> - <https://blog.f-secure.com/hunting-for-gargoyle-memory-scanning-evasion/> - <https://www.elastic.co/blog/detecting-cobalt-strike-with-memory-signatures>