

```
{
  "payload": {
    "allShortcutsEnabled": false,
    "fileTree": {
      "WikiLoader": {
        "items": [
          {
            "name": "Images",
            "path": "WikiLoader/Images",
            "contentType": "directory"
          },
          {
            "name": "WikiLoader Shellcode pt2.md",
            "path": "WikiLoader/WikiLoader Shellcode pt2.md",
            "contentType": "file"
          },
          {
            "name": "WikiLoader Shellcode pt3.md",
            "path": "WikiLoader/WikiLoader Shellcode pt3.md",
            "contentType": "file"
          },
          {
            "name": "WikiLoader notepad.md",
            "path": "WikiLoader/WikiLoader notepad.md",
            "contentType": "file"
          }
        ],
        "totalCount": 4
      },
      "items": [
          {
            "name": ".obsidian",
            "path": ".obsidian",
            "contentType": "directory"
          },
          {
            "name": "Pikabot",
            "path": "Pikabot",
            "contentType": "directory"
          },
          {
            "name": "WSHRAT",
            "path": "WSHRAT",
            "contentType": "directory"
          },
          {
            "name": "WikiLoader",
            "path": "WikiLoader",
            "contentType": "directory"
          },
          {
            "name": "README.md",
            "path": "README.md",
            "contentType": "file"
          }
        ],
        "totalCount": 5
      },
      "fileTreeProcessingTime": 3.319514,
      "foldersToFetch": [],
      "repo": {
        "id": 675143377,
        "defaultBranch": "main",
        "name": "MalwareAnalysisReports",
        "ownerLogin": "VenzoV",
        "currentUserCanPush": false,
        "isFork": false,
        "isEr08-05T23:42:05.000Z",
        "ownerAvatar": "https://avatars.githubusercontent.com/u/107503502?v=4",
        "public": true,
        "private": false,
        "isOrgOwned": false,
        "symbolsExpanded": false,
        "treeExpanded": true,
        "refInfo": {
          "name": "main",
          "listCacheKey": "v0:1691279081.0",
          "canEdit": false,
          "refType": "branch",
          "currentOid": "e5a302131a634bb67d77dc8e5068d0e3c14018",
          "Shellcode pt2.md",
          "currentUser": null,
          "blob": {
            "rawLines": null,
            "stylingDirectives": null,
            "csv": null,
            "csvError": null,
            "dependabotInfo": {
              "showConfigurationBanner": false,
              "configFilePath": null,
              "networkDependabotPath": "/VenzoV/MalwareAnalysisReports/network/updates",
              "dismissConfigNotice/dependabot_configuration_notice",
              "configurationNoticeDismissed": null,
              "displayName": "WikiLoader Shellcode pt2.md",
              "displayUrl": "https://github.com/VenzoV/MalwareAnalysisReports/blob/main/WikiLoader/WikiLoader%20Shellcode%20pt2.md?raw=true",
              "headerInfo": {
                "blobSize": "6.5 KB",
                "deleteTooltip": "You must be signed in to make or propose changes",
                "editTooltip": "You must be signed in to make or propose changes",
                "deleteInfo": {
                  "deleteTooltip": "You must be signed in to make or propose changes"
                },
                "editInfo": {
                  "editTooltip": "You must be signed in to make or propose changes"
                }
              },
              "ghDesktopPath": "https://desktop.github.com",
              "isGitLfs": false,
              "gitLfsPath": null,
              "onBranch": true,
              "shortPath": "afecf18",
              "siteNavLoginPath": "return_to=https%3A%2F%2Fgithub.com%2FVenzoV%2FMalwareAnalysisReports%2Fblob%2Fmain%2FWikiLoader%2FWikiLoader%2520Shellcode%20pt2.md",
              "level": 1,
              "text": "Summary part 1",
              "anchor": "summary-part-1",
              "htmlText": "Summary part 1"
            },
            "level": 1,
            "text": "Overview",
            "anchor": "overview",
            "htmlText": "Overview"
          },
          {
            "level": 2,
            "text": "Loading bingsmap.dll & First Indirect syscall",
            "anchor": "loading-bingsmapdll--first-indirect-syscall",
            "htmlText": "Loading bingsmap.dll & First Indirect syscall"
          },
          {
            "level": 2,
            "text": "Shellcode DLL injection",
            "anchor": "shellcode-dll-injection",
            "htmlText": "Shellcode DLL injection"
          },
          {
            "level": 2,
            "text": "New thread",
            "anchor": "new-thread",
            "htmlText": "New thread"
          },
          {
            "level": 2,
            "text": "Anti Debug",
            "anchor": "anti-debug",
            "htmlText": "Anti Debug"
          },
          {
            "level": 2,
            "text": "Injecting 3rd shellcode into explorer.exe",
            "anchor": "injecting-3rd-shellcode-into-explorerexe",
            "htmlText": "Injecting 3rd shellcode into explorer.exe"
          },
          {
            "level": 2,
            "text": "End of part 2",
            "anchor": "end-of-part-2",
            "htmlText": "End of part 2"
          },
          {
            "level": 2,
            "text": "References",
            "anchor": "references",
            "htmlText": "References"
          }
        ],
        "lineInfo": {
          "truncatedLoc": "162",
          "truncatedSloc": "102",
          "mode": "file",
          "image": false,
          "isCodeownersFile": null,
          "isPlain": false,
          "isValidLegacyIssueTemplate": false,
          "issue-and-pull-request-templates",
          "issueTemplate": null,
          "discussionTemplate": null,
          "language": "Markdown",
          "languageID": 222,
          "large": false,
          "loggedIn": false,
          "planSupportInfo": {
            "reposFork": null,
            "repoOwnedByCurrentUser": null,
            "requestFullPath": "/VenzoV/MalwareAnalysisReports/blob/main/WikiLoader/WikiLoader%20Shellcode%20pt2.md",
            "dismissActionNoticePath": "/settings/dismiss-notice/publish_action_from_dockerfile",
            "releasePath": "/VenzoV/MalwareAnalysisReports/releases/new?marketplace=true",
            "showPublishActionBanner": false
          },
          "rawBlobUrl": "https://github.com/VenzoV/MalwareAnalysisReports/raw/main/WikiLoader/Wiki"
        }
      }
    }
  }
}
```

Summary part 1

\n

In part one we looked at how shellcode was decrypted by using the Microsoft Bcrypt library. \nAES CBC mode was used to decrypt shellcode located in the file "certificate.pem". \nOnce this was done a new thread is created and entry point changed to point to the newly decrypted payload allocated within the memory. \nWe switched threads and followed the shellcode.

\n

Overview

\n

- \n
 - Loading bingmaps.dll
- \n
 - Long busy loop to slow down execution.
- \n
 - Retrieving once again API via PEB walking
- \n
 - Function used to load API from: Kernel32.dll
- \n
 - Function Used to load native API to perform indirect syscalls: ntdll.dll

- \n
- Checks if native calls are hooked.
- \n
- New thread is created and execution is switched, the thread points to bingsmap.dll and jumps back and forth to shellcode.
- \n

\n

Loading bingsmap.dll & First Indirect syscall

\n

The shellcode enters firstly into a long loop which takes a bit of time to execute until rcx value is 0x1b1. This is likely to slowdown analysis or in general to delay execution of malicious code.

\n

 "Image"


\n

PEB reference is fetched and as in part one we enter the code section where it is parsed to fetch export table. Like part one, it goes through the InMemoryOrderModuleList and compares the result to some chars to get KERNEL32.dll or NTDLL.dll based on what API it needs. So what we expect is the following:

\n

- \n
- loops through all the functions
- \n
- Calculates a hash for each one
- \n
- Find LoadLibraryA()
- \n
- JMP to LoadLibraryA() with "bingmaps.dll" as argument
- \n
- Fetching NtProtectVirtualMemory
- \n
- Call NtProtectVirtualMemory on bingmaps.dll location with 0x40 rx -> wrx
- \n
- Overwrites code of the exported function bingmaps.dll (GetBingsMapFactory). It reads from the shellcode and writes to this location, essentially injecting a part of itself into the legitimate binary. The bytes are xored before writing.
- \n

\n

 "Image"

\n

Hashing loop:

\n

 "Image"

\n

Call to LoadLibraryA:


\n

 "Image"

\n

First call to NtProtectVirtualMemory:

\n

 "Image"

\n

Memory protection alteration:

\n

 "Image" \n  "Image"

\n

Shellcode DLL injection

\n

Next, malware injects part of its shellcode into the exported function of bingmaps.dll GetBingsMapFactory.\nOnce it has finished injecting code it will call once again NtProtectVirtualMemory with 0x20 PAGE_EXECUTE_READ on the .text section of bingmaps.dll. Reverting it back to RX permissions.

\n

 "Image"

\n

 "Image"

\n

Memory and instructions seen after inject is complete:

\n

 "Image"

\n

 "Image"

\n

New thread

\n

Malware will once again run through PEB , Ntdll.dll & Kernel32.dll in search for the following API:

\n

\n

- GetModuleFileNameA -> This will be the entry point of the new thread, but RIP will be the injected code seen before
- NtCreateThreadEx -> creates thread in suspended state and hidden from debugger. Value 0x5 we can change it to 0x1 if we want to see the thread in the debugger.
- NtGetContextThread
- NtSetContextThread -> Inside the context object in memory there is pointer referencing the start of injected code which will be the threads RIP. (BP will be set here to follow execution)
- NtResumeThread
- NtWaitForSingleObject

\n

 "Image"

\n

 "Image"

\n

Running execution we finally reach the new thread and code!

\n

 "Image"

\n

Anti Debug

\n

From the bingmaps.dll injected code, the malware proceeds with an initial anti debug trick.\nIt seems it has a hash table hardcoded and checks running processes if they match, if so it proceeds to exit.\nFor this, from the PEB it fetches the following API:

\n

- CreateToolHelp32SnapShot

\n

- Process32First

\n

- Process32Next

\n

\n

Each name from CreateToolHelp32SnapShot section, is hashed with the same algorithm identified above and then compared with the hashtable results.\nI haven't checked all the hashes but for it seems it quit when seeing x64dbg.exe and pe-bear.exe which I had running.

\n

Example hash of Pe-Bear -> C0A4EC617E002F0 -> check performed on 17E002F0 portion.

\n

To avoid this, from Process Hacker I opened the section created by CreateToolHelp32SnapShot and modified the hex values to match svchost.exe, and it worked, we get to the next stage.

\n

Also, worth noting that there is a specific check for explorer.exe, when it matches it runs another section of code that saves the PID of the process. This is needed for the next step.

\n

Hashtable:

\n



\n

Call to Process32Next:

\n



\n

Injecting 3rd shellcode into explorer.exe

\n

Malware from the context of the bingmaps.dll will now proceed to inject other shellcode to explorer.exe.\nTo do this the following is done:

\n

- ZwOpenProcess -> supplying the PID of explorer.exe

\n

- ZwAllocateVirtualMemory -> Allocates two memory buffers inside explorer.exe.

\n

- GetModuleFileA -> Get the full path name of the current process running all the malware

\n


- ZwWriteVirtualMemory -> Writes the shellcoded and the results of GetModuleFileA to explorer.exe section.

\n

\n

For the shellcode, a single byte is written at a time. The byte to write is obtained by a single XOR loop before the call, the result is loaded into R8 which is the 3rd argument for ZwWriteVirtualMemory.

\n


Call to open process:\n

\n


Call to ZwWriteVirtualMemory for current process name:

\n



\n
 "Image"

\n
Call to ZwWriteVirtualMemory for shellcode:

\n
 "Image"

\n
Memory inside explorer.exe after loop write:

\n
 "Image"

\n

End of part 2

\n
So we have found another shellcode inject and the code seems to proceed with some other anti analysis checks as I saw it loads:

\n
 \n
 • ZwQueryInformationProcess
 \n
 • RtlWow64GetCpuAreaInfo
 \n

\n
But I will look into this next time with part 3!

\n

References

\n
 \n
 • <https://bazaar.abuse.ch/sample/bef04e3b2b81f2dee39c42ab9be781f3db0059ec722ae3b5434c2e63512a68/>
 \n
 • <https://www.unpac.me/results/612d6d2c-c45d-47ba-a2bb-a218ec753d3f>
 \n
 • <https://twitter.com/Cryptolaemus1/status/1747394506331160736>
 \n
 • <https://www.proofpoint.com/us/blog/threat-insight/out-sandbox-wikiloader-digs-sophisticated-evasion>
 \n
 • <https://www.geoffchappell.com/studies/windows/km/ntoskrnl/inc/api/pebteb/peb/index.htm>
 \n
 • <https://mohamed-fakroud.gitbook.io/red-teamings-doj/shellcoding/leveraging-from-pe-parsing-technique-to-write-x86-shellcode>
 \n

```
\n","renderedFileInfo":null,"shortPath":null,"symbolsEnabled":true,"tabSize":8,"topBannersInfo":\n{"overridingGlobalFundingFile":false,"globalPreferredFundingPath":null,"repoOwner":"VenzoV","repoName":"MalwareAnalysisReports","showInvalid\ncloning-and-archiving-repositories/creating-a-repository-on-github/about-citation-\nfiles","actionsOnboardingTip":null},"truncated":false,"viewable":true,"workflowRedirectUrl":null,"symbols":\n{"timed_out":false,"not_analyzed":false,"symbols":[{"name":"Summary part\n1","kind":"section_1","ident_start":3,"ident_end":17,"extent_start":0,"extent_end":1,"fully_qualified_name":"Summary part 1","ident_utf16":\n{"start":{"line_number":1,"utf16_col":2},"end":{"line_number":1,"utf16_col":16}},"extent_utf16":{"start":{"line_number":0,"utf16_col":0},"end":\n{"line_number":1,"utf16_col":0}}},{"name":"Summary part\n1","kind":"section_1","ident_start":3,"ident_end":17,"extent_start":1,"extent_end":377,"fully_qualified_name":"Summary part 1","ident_utf16":\n{"start":{"line_number":1,"utf16_col":2},"end":{"line_number":1,"utf16_col":16}},"extent_utf16":{"start":{"line_number":1,"utf16_col":0},"end":\n{"line_number":7,"utf16_col":0}}},\n{"name":"Overview","kind":"section_1","ident_start":379,"ident_end":387,"extent_start":377,"extent_end":6659,"fully_qualified_name":"Overview","ic\n{"start":{"line_number":7,"utf16_col":2},"end":{"line_number":7,"utf16_col":10}},"extent_utf16":{"start":{"line_number":7,"utf16_col":0},"end":\n{"line_number":162,"utf16_col":0}}},{"name":"Loading bingsmap.dll & First Indirect\nsyscall","kind":"section_2","ident_start":784,"ident_end":829,"extent_start":781,"extent_end":2198,"fully_qualified_name":"Loading bingsmap.dll
```

```

& First Indirect syscall", "ident_utf16": {"start": {"line_number": 16, "utf16_col": 3}, "end": {"line_number": 16, "utf16_col": 48}}, "extent_utf16": {"start": {"line_number": 16, "utf16_col": 0}, "end": {"line_number": 53, "utf16_col": 0}}, {"name": "Shellcode DLL injection", "kind": "section_2", "ident_start": 2201, "ident_end": 2224, "extent_start": 2198, "extent_end": 2752, "fully_qualified_name": "Shellcode DLL injection", "ident_utf16": {"start": {"line_number": 53, "utf16_col": 3}, "end": {"line_number": 53, "utf16_col": 26}}, "extent_utf16": {"start": {"line_number": 53, "utf16_col": 0}, "end": {"line_number": 67, "utf16_col": 0}}, {"name": "New thread", "kind": "section_2", "ident_start": 2755, "ident_end": 2765, "extent_start": 2752, "extent_end": 3580, "fully_qualified_name": "New thread", "ident_utf16": {"start": {"line_number": 67, "utf16_col": 3}, "end": {"line_number": 67, "utf16_col": 13}}, "extent_utf16": {"start": {"line_number": 67, "utf16_col": 0}, "end": {"line_number": 86, "utf16_col": 0}}, {"name": "Anti Debug", "kind": "section_2", "ident_start": 3583, "ident_end": 3593, "extent_start": 3580, "extent_end": 4735, "fully_qualified_name": "Anti Debug", "ident_utf16": {"start": {"line_number": 86, "utf16_col": 3}, "end": {"line_number": 86, "utf16_col": 13}}, "extent_utf16": {"start": {"line_number": 86, "utf16_col": 0}, "end": {"line_number": 113, "utf16_col": 0}}, {"name": "Injecting 3rd shellcode into explorer.exe", "kind": "section_2", "ident_start": 4738, "ident_end": 4779, "extent_start": 4735, "extent_end": 5847, "fully_qualified_name": "Injecting 3rd shellcode into explorer.exe", "ident_utf16": {"start": {"line_number": 113, "utf16_col": 3}, "end": {"line_number": 113, "utf16_col": 44}}, "extent_utf16": {"start": {"line_number": 113, "utf16_col": 0}, "end": {"line_number": 146, "utf16_col": 0}}, {"name": "End of part 2", "kind": "section_2", "ident_start": 5850, "ident_end": 5863, "extent_start": 5847, "extent_end": 6097, "fully_qualified_name": "End of part 2", "ident_utf16": {"start": {"line_number": 146, "utf16_col": 3}, "end": {"line_number": 146, "utf16_col": 16}}, "extent_utf16": {"start": {"line_number": 146, "utf16_col": 0}, "end": {"line_number": 154, "utf16_col": 0}}, {"name": "References", "kind": "section_2", "ident_start": 6100, "ident_end": 6110, "extent_start": 6097, "extent_end": 6659, "fully_qualified_name": "References", "ident_utf16": {"start": {"line_number": 154, "utf16_col": 3}, "end": {"line_number": 154, "utf16_col": 13}}, "extent_utf16": {"start": {"line_number": 154, "utf16_col": 0}, "end": {"line_number": 162, "utf16_col": 0}}}, {"copilotInfo": null, "copilotAccessAllowed": false, "csrf_tokens": {""/VenzoV/MalwareAnalysisReports/branches": {"post": "MnivvJ1flurjEHmuiPQZ681Dnjg6oMrb4jIwm7qG2e-wQcngD0_8CnZQV70Fx-rk78sw1kgVvrQL-ReubMjqxA"}, "/repos/preferences": {"post": "1O99XIsdBO_ScmXQsWVGIBnubvhlHMnhU-kWEW_W8FqrY5UR4GjS36dqyZgH5x4dNFfDOh9oV0IspEfCmoRyA"}}, "title": "MalwareAnalysisReports/WikiLoader/WikiLoader Shellcode pt2.md at main · VenzoV/MalwareAnalysisReports"}

```