A grey-hat eszközök elleni átfogó biztonsági megoldáshoz kifinomultság, részletesség és megfontoltság szükséges.

A Cobalt Strike Elleni Védekezés

Prevenciós és Threat Hunting Metodológia

Black Cell Magyarország Kft.

COBALT STRIKE
ADVANCED THREAT TACTICS FOR PENETRATION TESTERS

Tartalom

A Cobalt Strike elleni védekezés

Vezetői összefoglaló

Az elmúlt időszakban több nagyvállalatot ért kibertámadás, ami mögött feltételezhetően, illetve több esetben bizonyíthatóan a Cobalt Strike nevű offenzív kiberbiztonsági eszköz áll, állhat. A Black Cell Magyarország Kft. által kidolgozott védelmi eljárások segítségével, mind a megelőzése, mind a detektálása, mind az eltávolítása megoldható. A Cobalt Strike által használt technikák 38 különböző eljárást foglalnak magukba, mindegyikre különböző szabályt kell implementálni, logikai, fizikai és adminisztratív vonatkozásban. Ezeket a szabályokat speciálisan erre a célra fejlesztett, módosított védelmi eszközökbe kell felvinni.

Az általunk kifejlesztett metodológia jelentős hangsúlyt tesz a támadások megelőzésére. Számos mutatója van egy még bekövetkezetlen támadásnak és ezek észlelésével és a helyes reakció kiváltásával megállítható egy támadás mielőtt bármilyen kár keletkezne. Természetesen a kiberbiztonság világában sosem garantálható a 100%-os védelem, ezért nem csak a megelőzésre, hanem a bekövetkezett támadás észlelésére, beazonosítására és elhárítására is részletes eljárásokat készítettünk. A megoldásaink nem kizárólag Cobalt Strike támadások ellen effektívek, hanem a manapság legtöbbet előforduló fenyegetések hárítására is alkalmas.

A Cobalt Strike-ról

A Cobalt Strike egy kereskedelmi célú, teljes funkcionalitású, penetrációs tesztelő eszköz, amely ellenséges szimulációs szoftvernek számít, amelyet célzott támadások végrehajtására és speciálisan a *post-exploit*-álás emulálására terveztek. A Cobalt Strike interaktív utólagos kihasználási képességei lefedik az támadási taktikák szinte teljes skáláját, mindezt egyetlen integrált rendszerben hajtják végre. Saját képességein túl a Cobalt Strike kihasználja más ismert eszközök, például a *Metasploit* és a *Mimikatz* képességeit is.

Technikai képességei

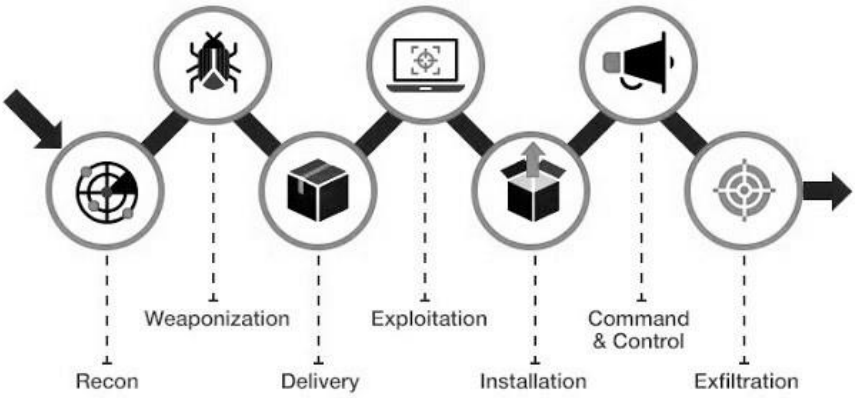
1. Access Token Manipulation
2. BITS Jobs
3. Bypass User Account Control
4. Command-Line Interface
5. Commonly Used Port
6. Component Object Model and Distributed COM
7. Connection Proxy

8. Credential Dumping
9. Custom Command and Control Protocol
10. Data from Local System
11. Execution through API
12. Exploitation for Privilege Escalation
13. Indicator Removal from Tools
14. Input Capture
15. Man in the Browser
16. Multiband Communication
17. Network Service Scanning
18. Network Share Discovery
19. New Service
20. Parent PID Spoofing
21. Pass the Hash
22. PowerShell
23. Process Discovery
24. Process Hollowing
25. Process Injection
26. Remote Desktop Protocol
27. Remote Services
28. Remote System Discovery
29. Scheduled Transfer
30. Screen Capture
31. Scripting
32. Service Execution
33. Standard Application Layer Protocol
34. Timestomp
35. Windows Admin Shares
36. Valid Accounts
37. Windows Management Instrumentation
38. Windows Remote Management

A támadás azonosítása

Indikátorok

Az indikátorok azonosításának lényege, hogy kezdeti fázisban kiszúrjuk a támadási kísérletet és a támadás korai fázisában megelőző, védekező intézkedéseket vezethessünk be, ehhez azonban szükséges a támadás ismerete. A támadás egy malícius tartalommal bíró dokumentum célszemélynek történő elküldésével kezdődik (a felderítési tevékenységek után). A fájlt megnyitva egy makró vagy egy végrehajtható fájl futtatásával kezdődik a támadás. A hálózatban történő perzisztencia kialakítása után beacon-t létrehozva tartja fenn a kommunikációt a control (C2) szerverrel.



Cyber Kill Chain © Mitre

A Cobalt Strike már a cyber kill chain delivery fázisában kiszűrhető, a fájl célszemélynek történő eljuttatáskor.

A legegyszerűbb és egyben egyik legfontosabb védekezési módszer az alkalmazottak biztonsági tudatosságának növelése, hogy ne nyissanak meg gyanús forrásból származó fájlokat, hanem ezeket továbbítsák a gyanús fájlok elemzésére szakosodott szakemberek felé.

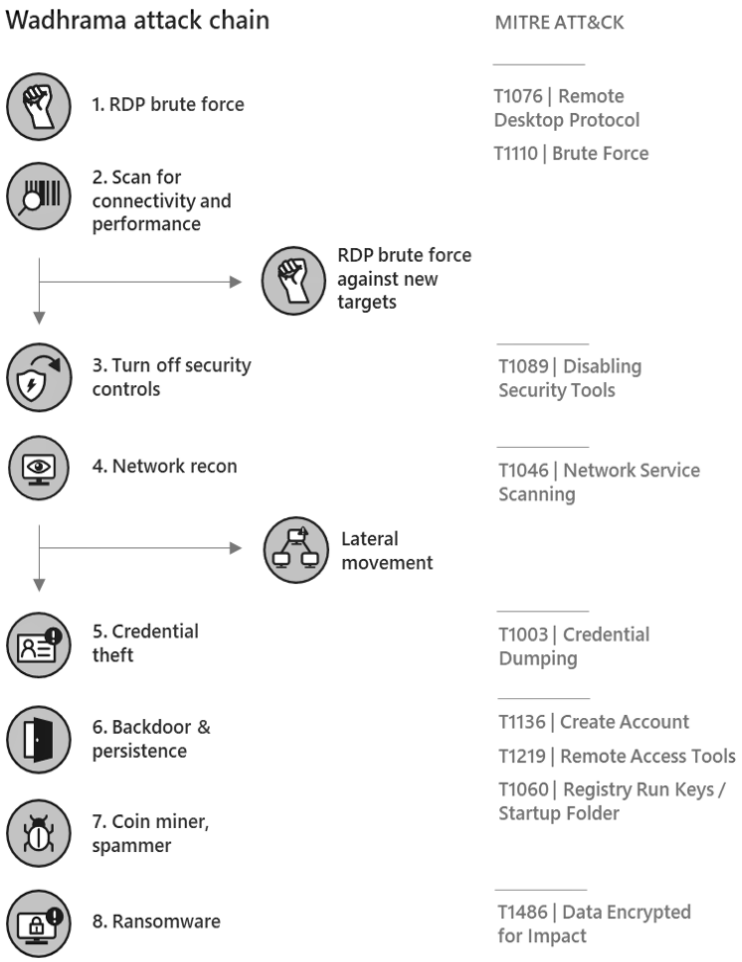
A Cobalt Strike felsorolt technikai képességei közül logok gyűjtésének beállításával kiszűrhető több módszer, míg vannak olyanok, melyek kiszűrésére további szoftverek telepítése és konfigurálása szükséges. Logok gyűjtésével hárítható a parancssor, illetve PowerShell, service futtatása, felhasználói fiókok létrehozása, hálózati szkennek, valamint a távoli elérés jelentette biztonsági kockázat.

A Cobalt Strike első fázisa a felderítés és a vállalati információ gyűjtése. Ilyenkor a támadó különböző szkennekkel megpróbálja beazonosítani a vállalati infrasktruktúra alkotóelemeit és esetleges sebezhetőségeit. Ezek a szkennek úgymond „hangosak” mivel sok könnyen érzékelhető kérés érkezik az infrastruktúrát alkotó erőforrásokra. Csupán egy SIEM rendszerre és jól beállított naplófájl gyűjtésre van szükség az esetleges felderítési műveletek riasztására. Hozzá kell tenni, hogy ezek a risztások önmagukban nem tudják jelezni, hogy kifejezetten Cobalt Strike támadásról van szó, hanem elegendő felkészülési időt biztosít az IT biztonsági szakembereknek a közelgő támadásra való felkészülésre.

A pre-exploit tevékenységeket követi a kezdetleges hozzáférés megszerzése és a beacon telepítése. Ez általában egy spear phishing kampányként valósul meg, ahol rosszindulatú dokumentumokat tartalmazó emaileket küldenek számos belső email címre, annak reményében, hogy valaki megnyitja. Ez a spear phishing kampány az egyik legkönnyebben detektálható része egy Cobalt Strike támadásnak. Amennyiben ismeretlen címről rövid időn belül nagy mennyiségű email érkezik belsős email címekre, riasztás keletkezik, amiből hamar kiderül egy képzett IT biztonsági szakembernek, hogy valójában phishing támadás érkezett. Ezek után az emailt fogadó felhasználók értesíthetők, és össze gyűjthető, hogy kik nyitották meg a csatolmányt. Ezen kívül az email csatolmányok hash értékét automatikusan lehet ellenőrizni, hogy tartalmaznak e rosszindulatú kódot, így megkönnyítve a rosszindulatú phishing támadások beazonosítását.

A Cobalt Strike által alkalmazott dokumentumokba ágyazott rosszindulatú makrók esetleges megnyitása is egyszerűen detektálható. A dokumentumokba ágyazott VBA scriptek új folyamatokat hoznak létre, amelyek a megfelelő process creation logolással riasztást váltanak ki egy SIEM rendszerben akkor is, ha a támadó sikeresen kiiktatja az antivírust, vagy utólag kiüríti a naplófájlt.

A fent említett viszonylag alapvető logolás egy SIEM rendszerrel párosítva rendkívül valószínűtlenné teszi a sikeres Cobalt Strike támadásokat, sőt minimális kiegészítéssel a legtöbb gyakori támadás elkerülhető, mint például az ábrán látható RDP brute force támadás. Ha mégis sikerülne egy támadónak hozzáférést szereznie és egy tartós fenyegetés alakulna ki, még mindig marad számos kisebb-nagyobb védekezési módszer, amely értesít komolyabb kár okozása előtt.



© Microsoft

A sikeres Cobalt Strike támadás esetén a telepített beacon-nek előbb vagy utóbb kommunikálnia kell egy C2 szerverrel. Ezt a kommunikációt gyakran álcázni próbálják különböző obfuszkáló eljárásokkal.

Ezt az álcázott kommunikációt könnyen be lehet azonosítani egy next-generation tűzfallal vagy egyéb intrusion detection rendszerrel. Ezek az eszközök mélyen elemzik a hálózati kommunikációt alkotó csomagokat és anomália esetén riasztanak.

A fent említett védekezési metódusok együtt szinte lehetetlenné teszik a sikeres és detektálatlan Cobalt Strike támadást. Ennek ellenére még létezik eddig nem említett, főleg lokális hatású támadói képesség, amelynek érzékeléséhez, már nem elég egyszerűen naplófájlokat gyűjteni és elemezni. Ezekhez már fejlett Endpoint Detection and Response és vulnerability management eszközökre lesz szükség.

Elegendő Logolás és SIEM Rendszer	Intrusion Detection System és/vagy Next Generation Firewall	Endpoint Detection és/vagy Vulnerability Management
• Network Service Scanning (ID: T1046) • Network Share Discovery (ID: T1135) • New Service (ID: T1050) • Command-Line Interface (ID: T1059) • Credential Dumping (ID: T1003) • Data from Local System (ID: T1005) • PowerShell (ID: T1086) • Process Discovery (ID: T1057) • Process Injection (ID: T1055) • Remote System Discovery (ID: T1018) • Scripting (ID: T1064) • Service Execution (ID: T1035) • Timestomp (ID: T1099) • Windows Admin Shares (ID: T1077) • Valid Accounts (ID: T1078) • Windows Management Instrumentation (ID: T1047)	• Remote Desktop Protocol (ID: T1076) • Remote Services (ID: T1021) • Scheduled Transfer (ID: T1029) • Standard Application Layer Protocol (ID: T1071) • Commonly Used Port (ID: T1043) • Connection Proxy (ID: T1090) • Custom Command and Control Protocol (ID: T1094) • Multiband Communication (ID: T1026) • Windows Remote Management (ID: T1028)	• Access Token Manipulation (ID: T1134) • BITS Jobs (ID: T1197) • Bypass User Account Control (ID: T1088) • Component Object Model and Distributed COM (ID: T1175) • Execution through API (ID: T1106) • Exploitation for Privilege Escalation (ID: T1068) • Indicator Removal from Tools (ID: T1066) • Input Capture (ID: T1056) • Man in the Browser (ID: T1185) • Parent PID Spoofing (ID: T1502) • Pass the Hash (ID: T1075) • Process Hollowing (ID: T1093) • Screen Capture (ID: T1113)

A különböző védekezési megoldások, az általuk lefedett Cobalt Strike képességek és a hozzá tartozó Mitre Att&ck reference.

Az előzőekben megfogalmazott detektációs eljárások szilárd és megbízható védelmet nyújtanak Cobalt Strike támadások ellen, a pre- és post-exploit képességeinek érzékelésével. Viszont léteznek különböző észlelési metódusok, amik kifejezetten a mögöttes keretrendszer egyes tulajdonságait keresi.

- Default certificate hash: A Cobalt Strike alapértelmezett tanúsítványa sok esetben nem kerül lecserélésre. Ezért, ha ennek a tanúsítványnak a hash értékét érzékeljük a hálózatunkban, akkor tudjuk, hogy támadás történt.

- Default controller port: A Cobalt Strike alapértelmezett portja az 50050. Ha ilyen portot találunk nyitva, vagy látjuk szerepelni hálózati kommunikációban, akkor tudjuk, hogy támadás történt.

- HTTP response whitespace: A Cobalt Strike 3.13-as verzió előtti szerverek HTTP válaszaiban létezik egy könnyen detektálható anomália.

```
0030  ff ff e9 13 00 00 48 54  54 50 2f 31 2e 31 20 32   ·····HT TP/1.1 2
0040  30 30 20 4f 4b 20 0d 0a  43 6f 6e 74 65 6e 74 2d   00 OK · Content-
0050  54 79 70 65 3a 20 74 65  78 74 2f 68 74 6d 6c 0d   Type: te xt/html·
0060  0a 44 61 74 65 3a 20 46  72 69 2c 20 33 20 4d 61   ·Date: F ri, 3 Ma
0070  79 20 32 30 31 39 20 31  34 3a 30 30 3a 30 39 20   y 2019 1 4:00:09
0080  47 4d 54 0d 0a 43 6f 6e  6e 65 63 74 69 6f 6e 3a   GMT··Con nection:
0090  20 6b 65 65 70 2d 61 6c  69 76 65 0d 0a 43 6f 6e    keep-al ive··Con
00a0  74 65 6e 74 2d 4c 65 6e  67 74 68 3a 20 35 0d 0a   tent-Len gth: 5··
00b0  0d 0a                                             ··
```

© Recorded Future

A fenti képen látható, hogy a státusz kód után szerepel egy rendkívüli szóköz, amely más hiteles válaszokban nem szerepel.

- TLS negotiation fingerprint: A TLS egy protokoll, amely titkosítja a hálózati kommunikációt, biztonságos információközvetítés érdekében. A titkosított kommunikációs csatorna létrehozásához, szükséges egy szinkronizációs folyamat, amely titkosítatlanul történik. Erről a folyamatról készíthetünk egy „ujjlenyomatot”, ami alapján be tudunk azonosítani hasonló kommunikációt. A megfelelő IDS megoldással elemezhetjük a hálózati kommunikációt Cobalt Strike ujjlenyomatokat keresve.

- In memory detection: Ha egy Cobalt Strike támadás gyanaúja merül fel, akkor a végpont memóriájából készített képfájlból megállapítható, hogy fut-e beacon, ha igen akkor milyen paraméterekkel. Ez nem automatizálható folyamat, de támadás esetén nélkülözhetetlen információt biztosít az elhárításhoz.

```
[root@localhost vm]# python vol.py cobaltstrikeconfig -f mem.image --profile=Win7SP1x64
Volatility Foundation Volatility Framework 2.5
config addr: 0002F35C

--------------------------------------------------------------------------
Process: powershell.exe (2508)

[CobaltStrike Config Info]
BeaconType              : 1 (Hybrid HTTP and DNS)
Port                    : 80
Polling(ms)             : 60000
Unknown1                : '\x00\x10\x00('
Jitter                  : 0
Maxdns                  : 255
Unknown2                : '0\x81\x9f0\r\x06\t*\x86H\x86\xf7\r\x01\x01\x01\x05\x00\x03\x81\x8d\x000\x81\x89\x02\x81\x81\x00\xb3\
\x8cK\xdeH.\xc4W$HnCn\xb2\xf0\xf2\x92\x0f\xfe\x9f\x05\x85\x14\x1e\xe7x\x15\x9d\x96\xe7v[i\xb8_d\xa8*\x8el!\xed\x8c\xe8\xe5S\xe1
5h\xae\x85\x02\x03\x01\x00\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\
0\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\
C2Server                : nl01.misaupdate.com,/__utm.gif,nl02.misaupdate.com,/__utm.gif,nl03.misaupdate.com,/__utm.gif
UserAgent               : Mozilla/5.0 (compatible; MSIE 10.0; Windows NT 6.2; WOW64; Trident/6.0; MATBJS)
HTTP_Method2_Path       : /___utm.gif
Unknown3                : '\x00\x00\x00\x04\x00\x00\x00\x02\x00\x00\x00\x0f\x00\x00\x00\x02\x00\x00\x00\x0f\x00\x00\x00\x02\x00
\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00
x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00
00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00
0\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00'
Header1                 : utmac=UA-2202604-2
                        : utmcn=1
                        : utmcs=ISO-8859-1
                        : utmsr=1280x1024
                        : utmsc=32-bit
                        : utmul=en-US
                        : __utma
                        : utmcc
Header2                 : Content-Type: application/octet-stream
                        : UA-220
                        : -2
                        : utmac
                        : utmcn=1
                        : utmcs=ISO-8859-1
                        : utmsr=1280x1024
                        : utmsc=32-bit
                        : utmul=en-US
Injection_Process       :
PipeName                : \\%s\pipe\msagent_%x
Year                    : 0
Month                   : 0
Day                     : 0
DNS_idle                : 0.0.0.0
DNS_sleep(ms)           : 0
Method1                 : GET
Method2                 : POST
Unknown4                : '\x00\x00\x00\x00'
Spawnto_x86             : %windir%\syswow64\rundll32.exe
Spawnto_x64             : %windir%\sysnative\rundll32.exe
Unknown5                : '\x00\x01'
Proxy_HostName          :
Proxy_UserName          :
Proxy_Password          :
Proxy_AccessType        : 2 (use IE settings)
create_remote_thread _  : Enable
```

Minta kimenet, amely egy Cobalt Strike beacon-t jelez a memóriaképfájlban. © JPCERT

YARA

A YARA egy olyan eszköz, melynek célja a malware és az arra utaló minták azonosítása és osztályozása. A YARA segítségével azonosíthatóak a Cobalt Strike-ra utaló minták is. Az alábbiakban néhány ilyen céllal létrehozott YARA szabályok láthatók bemutatásképp:

Hosztokat azonosító szabály:

```
rule CobaltStrike_C2_Host_Indicator {
        meta:
                description = "Detects CobaltStrike C2 host artifacts"
                author = "yara@s3c.za.net"
                date = "2019-08-16"
        strings:
                $c2_indicator_fp = "#Host: %s"
                $c2_indicator = "#Host:"
        condition:
                $c2_indicator and not $c2_indicator_fp
                and not uint32(0) == 0x0a786564
                and not uint32(0) == 0x0a796564
}
```

Konfigurációs fájlt azonosító szabály:

```
rule CobaltStrike_C2_Encoded_Config_Indicator {
        meta:
                description = "Detects CobaltStrike C2 encoded profile configuration"
                author = "yara@s3c.za.net"
                date = "2019-08-16"
        strings:
                $c2_enc_config = {69 68 69 68 69 6B ?? ?? 69 6B 69 68 69 6B ?? ?? 69 6A 69 6B 69 6D ?? ?? ?? ?? 69 6D 69 6B 69 6D
?? ?? ?? ?? 69 6C 69 68 69 6B ?? ?? 69 6F 69 68 69 6B ?? ?? 69 6E 69 6A 68 69}
        condition:
                $c2_enc_config
}
```

Beacon-öket azonosító szabálycsoport:

```
 privete rule cobaltstrike_beacon_raw
{
        strings:
                $s1 = "%d is an x64 process (can't inject x86 content)" fullword
                $s2 = "Failed to impersonate logged on user %d (%u)" fullword
                $s3 = "powershell -nop -exec bypass -EncodedCommand \"%s\"" fullword
                $s4 = "IEX (New-Object Net.Webclient).DownloadString('http://127.0.0.1:%u/'); %s" fullword
                $s5 = "could not run command (w/ token) because of its length of %d bytes!" fullword
                $s6 = "could not write to process memory: %d" fullword
                $s7 =
"%s.4%08x%08x%08x%08x%08x.%08x%08x%08x%08x%08x%08x%08x.%08x%08x%08x%08x%08x%08x%08x.%08x%08x%08x%08x%08x%08
x%08x.%x%x.%s" fullword
                $s8 = "Could not connect to pipe (%s): %d" fullword

                $b1 = "beacon.dll"    fullword
                $b2 = "beacon.x86.dll" fullword
                $b3 = "beacon.x64.dll" fullword

        condition:
                uint16(0) == 0x5a4d and
                filesize < 1000KB and
                (
                        any of ($b*) or
                        5 of ($s*)
                )
}

 private rule cobaltstrike_beacon_exe
{
        condition:
                cobaltstrike_template_exe and
                filesize > 100KB and filesize < 500KB and
                pe.sections[pe.section_index(".data")].raw_data_size > 200000 and
                math.entropy(pe.sections[pe.section_index(".data")].raw_data_offset + 1024, 150000 ) >= 7
}

private rule cobaltstrike_beacon_b64
{
        strings:
                $s1a = "JWQgaXMgYW4geDY0IHByb2Nlc3MgKGNhbid0IGluam"
                $s1b = "ZCBpcyBhbiB4NjQgcHJvY2VzcyAoY2FuJ3QgaW5qZW"
                $s1c = "IGlzIGFuIHg2NCBwcm9jZXNzIChjYW4ndCBpbmplY3"

                $s2a = "RmFpbGVkIHRvIGltcGVyc29uYXRlIGxvZ2dlZCBvbi"
                $s2b = "YWlsZWQgdG8gaW1wZXJzb25hdGUgbG9nZ2VkIG9uIH"
                $s2c = "aWxlZCB0byBpbXBlcnNvbmF0ZSBsb2dnZWQgb24gdX"

                $s3a = "cG93ZXJzaGVsbCAtbm9wIC1leGVjIGJ5cGFzcyAtRW"
                $s3b = "b3dlcnNoZWxsIC1ub3AgLWV4ZWMgYnlwYXNzIC1Fbm"
                $s3c = "d2Vyc2hlbGwgLW5vcCAtZXhlYyBieXBhc3MgLUVuY2"

                $s4a = "SUVYIChOZXctT2JqZWN0IE5ldC5XZWJjbGllbnQpLk"
                $s4b = "RVggKE5ldy1PYmplY3QgTmV0LldlYmNsaWVudCkuRG"
                $s4c = "WCAoTmV3LU9iamVjdCBOZXQuV2ViY2xpZW50KS5Eb3"

        condition:
                filesize < 1000KB and
                5 of ($s*)
}
```

```
rule hacktool_windows_cobaltstrike_beacon
{
	meta:
		description = "Detection of the Beacon payload from Cobalt Strike"
		reference = "https://www.cobaltstrike.com/help-beacon"
		author = "@javutin, @joseselvi"
	condition:
		cobaltstrike_beacon_b64 or
		cobaltstrike_beacon_raw or
		cobaltstrike_beacon_exe
}
```

Post-exploitation modult azonosító szabály:

```
rule hacktool_windows_cobaltstrike_postexploitation
{
	meta:
		description = "Detection of strings in the post-exploitation modules of Cobalt Strike"
		reference = "https://www.cobaltstrike.com/support"
		author = "@javutin, @mimeframe"
	strings:
		$s1 = "\\devcenter\\aggressor\\external\\"

	condition:
		filesize > 10KB and filesize < 1000KB and
		all of ($s*)
}
```

Payloadot azonosító szabály:

```
rule cobaltstrike_template_exe
{
	meta:
		description = "Template to provide executable detection Cobalt Strike payloads"
		reference = "https://www.cobaltstrike.com"
		author = "@javutin, @joseselvi"
	strings:
		$compiler = "mingw-w64 runtime failure" nocase

		$f1 = "VirtualQuery"   fullword
		$f2 = "VirtualProtect" fullword
		$f3 = "vfprintf"       fullword
		$f4 = "Sleep"          fullword
		$f5 = "GetTickCount"   fullword

		$c1 = { // Compare case insensitive with "msvcrt", char by char
				0f b6 50 01 80 fa 53 74 05 80 fa 73 75 42 0f b6
				50 02 80 fa 56 74 05 80 fa 76 75 34 0f b6 50 03
				80 fa 43 74 05 80 fa 63 75 26 0f b6 50 04 80 fa
				52 74 05 80 fa 72 75 18 0f b6 50 05 80 fa 54 74
		}
	condition:
		uint16(0) == 0x5a4d and
		filesize < 1000KB and
		$compiler and
		all of ($f*) and
		all of ($c*)
}
```

Mimikatz kiegészítő szoftvert azonosító szabályok:

```
rule hacktool_windows_mimikatz_copywrite
{
	meta:
		description = "Mimikatz credential dump tool: Author copywrite"
		reference = "https://github.com/gentilkiwi/mimikatz"
		author = "@fusionrace"
		md5_1 = "0c87c0ca04f0ab626b5137409dded15ac66c058be6df09e22a636cc2bcb021b8"
		md5_2 = "0c91f4ca25aedf306d68edaea63b84efec0385321eacf25419a3050f2394ee3b"
		md5_3 = "0fee62bae204cf89d954d2cbf82a76b771744b981aef4c651caab43436b5a143"
		md5_4 = "004c07dcd04b4e81f73aacd99c7351337f894e4dac6c91dcfaadb4a1510a967c"
		md5_5 = "09c542ff784bf98b2c4899900d4e699c5b2e2619a4c5eff68f6add14c74444ca"
		md5_6 = "09054be3cc568f57321be32e769ae3ccaf21653e5d1e3db85b5af4421c200669"
	strings:
		$s1 = "Kiwi en C" fullword ascii wide
		$s2 = "Benjamin DELPY `gentilkiwi`" fullword ascii wide
		$s3 = "http://blog.gentilkiwi.com/mimikatz" fullword ascii wide
		$s4 = "Build with love for POC only" fullword ascii wide
		$s5 = "gentilkiwi (Benjamin DELPY)" fullword wide
		$s6 = "KiwiSSP" fullword wide
		$s7 = "Kiwi Security Support Provider" fullword wide
		$s8 = "kiwi flavor !" fullword wide
	condition:
		any of them
}
```

Ajánlott megoldások

A fentiekben meghatározásra került számos Cobalt Strike támadás azonosítására alkalmas eljárás, viszont ezeknek az implementációjához különféle eszközökre van szükség. A Black Cell Magyarország Kft. által kidolgozott metódusokhoz ajánlott rendszereket az alábbiakban megtalálhatók.

Naplófájl gyűjtő és elemző rendszer: *Splunk*

A Splunk valós idejű adatok rögzítésére, indexelésére és korrelálására fejlesztett platform amely egy kereshető adattárból generál grafikonokat, jelentéseket, riasztásokat, irányítópultokat és vizualizációkat. A Splunk agent segítségével begyűjtésre kerülnek a naplófájlok és a fent meghatározott védekező riasztási szabályok itt kerülnek implementálásra.

IDS (Intrusion Detection System): *Suricata*

A Suricata egy nyílt forrású hálózati fenyegetésérzékelő szoftvereszköz, amely behatolás észlelési (IDS) és behatolás megelőzési (IPS) képességeket biztosít. Rendkívül jól teljesíti a hálózati csomagok mély elemzését és különböző minták észlelését, ami nélkülözhetetlenné teszi a Cobalt Strike hálózati indikátorainak észlelésében.

NGFW (Next-Generation Firewall): *Palo Alto*

A Palo Alto Next-Generation tűzfalai kombinálják a hagyományos tűzfalat más hálózati eszközök szűrési funkcióival, például alkalmazás-tűzfalak, mély csomag ellenőrzési eszközök és behatolás-megelőző rendszerek (IPS). Ezek a tűzfalak számos iparág vezető technológiákat foglalnak magukba, mint például TLS / SSL titkosított forgalomellenőrzés, QoS, fejlett vírusvédelem, fenyegetés megelőzés, alkalmazás-alapú szabályérvényesítés, és felhasználó azonosítás. A next-generation tűzfal segít megelőzni a Cobalt Strike támadásokat a vírus és fenyegetésvédelmi funkcióival, továbbá sikeres támadás esetén segít az azonosító minták keresésében.

EDR (Endpoint Detection and Response) & Vulnerability management: *MDATP*

A Microsoft Defender Advanced Threat Protection egy olyan platform, amelyet arra terveztek, hogy segítse a vállalkozásokat a fejlett fenyegetések megelőzésében, felderítésében, kivizsgálásában és az ezekre való reagálásban. Az MDATP végpont viselkedésérzékelőket, felhőalapú biztonsági elemzéseket, threat intelligence forrásokat és vulnerability managementet használ a legfejlettebb fenyegetések kiküszöbölésére. A Cobalt Strike végpontokat fenyegető funkcióinak megelőzésére, detektálására és támadás esetén az okozott kár enyhítésére is alkalmas.

Kifejezések

Beacon: Rosszindulatú beágyazott szoftver egy kompromizált rendszeren, amely folyamatos kapcsolatban van a támadó által irányított vezérlőszerverrel, és várja az új parancsok beérkezését, amelyeket majd végrehajt a rendszeren.

Brute force: Ebben a kontextusban a brute-force támadás abból áll, hogy a támadó rengeteg jelszóval folyamatosan próbál bejelentkezni azzal a reménnyel, hogy az egyik jelszó helyes lesz.

Post-expolit: A hoszt kihasználást követő szakasz, amelynek célja a veszélyeztetett hoszt értékének meghatározása és a hoszt irányításának fenntartása későbbi felhasználás céljából. A hoszt értékét a rajta tárolt adatok érzékenysége és a gép hasznossága határozza meg a hálózat további veszélyeztetésekor. Az ebben a szakaszban leírt módszerek célja, hogy segítsék a tesztelőt/támadót az érzékeny adatok azonosításában és dokumentálásában, a konfigurációs beállítások, a kommunikációs csatornák és a más hálózati eszközökkel fennálló kapcsolatok azonosításában, amelyek felhasználhatók a hálózathoz való további hozzáféréshez, és beállítsanak egy vagy több módszert a későbbi perzisztens hozzáférés céljából.

Cyber Kill Chain: A Cyber Kill Chain a támadási folyamat lépéseit mutatja be. 1. lépés a felderítés, mely során a szervezethez köthető email címek megszerzése, vállalati információ gyűjtése a cél. 2. lépés: felfegyverkezés, a megszerzett információk alapján a megfelelő exploit írása. 3. lépés az előkészített támadó eszközök célszemélyhez eljuttatása (delivery), melyet 4. lépésként a sérülékenység kihasználása követ. 5. lépés a malware telepítése, majd 6. lépésként kommunikáció kiépítése a kontroll szerverrel, hogy 7. lépésként elérhesse a támadó a célját.

EDR: Egy olyan technológia, amely védi a végpontokat, avagy számítógépes hardver eszközöket, a biztonsági fenyegetésektől. Adatokat gyűjt a végpontokról, majd elemzi ezen adatokat, hogy felfedjen lehetséges biztonsági fenyegetéseket és problémákat.

Hash: A hash algoritusok úgymond „ujjlenyomatot" készítenek, avagy bemeneti adatból a következő feltételekkel képeznek kimeneti adatot:

- Adott bemeneti adatból mindig ugyanazt a kimenetet adja.
- A kimeneti adat egyértelműen utal a bemeneti adatra, de a kimeneti adatból nem állítható elő a bemeneti adat.
- A bemeneti adat legkisebb változása is teljesen más kimenetet eredményez.

IDS: Illetéktelen hálózati behatolást jelző rendszer, amely azonosítja a hálózatban a gyanús vagy kártékony aktivitásokat, észrevesz minden olyan tevékenységet, ami eltér a rendszerek normális működésétől. Naplózza, katalogizálja és osztályozza a rendszerfolyamatokat.

Log: A naplófájl alkotó eleme, egy esemény rögzítő napló.

Metasploit: A Metasploit projekt egy számítógépes biztonsági projekt, amely információt nyújt a biztonsági résekről, valamint segít a penetrációs tesztelésben és az IDS szabályok fejlesztésében. A legismertebb alprojekt a nyílt forráskódú Metasploit Framework, amely célrendszerek elleni exploit kódok fejlesztéséhez és ezek végrehajtásához ad keretrendszert.

Mimikatz: A Mimikatz egy post-exploit eszköz, amely jelszavakat, valamint hash-eket, PIN-kódokat és Kerberos jegyeket gyűjt ki a memóriából. Továbbá pass-the-hash, pass-the-ticket vagy golden ticket támadásokat tesz lehetővé.

Payload: A rosszindulatú program azon része, amely magát a kártékony műveletet végrehajtja.

Phishing: Magyarul adathalászat. Bizalmas információ vagy hozzáférés megszerzésének csalárd kísérlete azáltal, hogy megbízható forrásúnak álcázzák az elektronikus kommunikációt.

Process creation log: Egy napló egy folyamat létrehozásáról.

QoS: A számítógépes hálózat teljesítményének mérése vagy fokozása, különösen a hálózat felhasználói által észlelt teljesítményre tekintettel.

RDP: Egy Microsoft által fejlesztett protokoll, amely grafikus felületet biztosít a felhasználó számára egy másik számítógéphez való csatlakozáshoz és annak vezérléséhez hálózati kapcsolaton keresztül.

Threat intelligence: Különböző nyílt forrásokból szerzett, veszélyekről és rosszindulatú egyénekről szóló információ, amely segít elhárítani kiberbiztonsági eseményeket.

Vulnerability management: A biztonsági rések kezelése, avagy szoftver sebezhetőségének " meghatározása, osztályozása, priorizálása, orvosolása és elhárítása.