IISpy: A complex server-side backdoor with anti-forensic features

welivesecurity.com/2021/08/09/iispy-complex-server-side-backdoor-antiforensic-features/

August 9, 2021

The second in our series on IIS threats dissects a malicious IIS extension that employs nifty tricks in an attempt to secure long-term espionage on the compromised servers

Zuzana Hromcová

9 Aug 2021 - 11:30AM

The second in our series on IIS threats dissects a malicious IIS extension that employs nifty tricks in an attempt to secure long-term espionage on the compromised servers

ESET researchers have discovered and analyzed a previously undocumented backdoor, implemented as an extension for *Internet Information Services* (IIS), Microsoft's web server software. The backdoor, which we named IISpy, uses a variety of tricks to interfere with the server's logging and to evade detection, in order to perform long-term espionage. IISpy is detected by ESET security solutions as Win{32,64}/BadIIS.

This blogpost is the second installment in our series where ESET researchers put IIS web server threats under the microscope – the other parts discuss IIS malware used for cybercrime and SEO fraud, respectively. For a comprehensive guide to how to detect, analyze and remove IIS malware, refer to our white paper Anatomy of native IIS malware, where IISpy is featured as one of the studied families (Group 7).

Anatomy of native IIS malware

Download Research Paper

Attack overview

According to ESET telemetry, this backdoor has been active since at least July 2020, and has been used with *Juicy Potato* (detected as Win64/HackTool.JuicyPotato by ESET security solutions), which is a privilege escalation tool. We suspect the attackers first obtain initial access to the IIS server via some vulnerability, and then use *Juicy Potato* to obtain the administrative privileges that are required to install IISpy as a native IIS extension.

According to our telemetry, IISpy affects a small number of IIS servers located in Canada, the USA and the Netherlands – but this is likely not the full picture, as it is still common for administrators to not use any security software on servers, and thus our visibility into IIS servers is limited.

Because IISpy is configured as an IIS extension, it can see all the HTTP requests received by the compromised IIS server, and shape the HTTP response that the server will answer with. IISpy uses this channel to implement its C&C communication, which allows it to operate as a *passive network implant*. As shown in Figure 1, the operator (not the backdoor) initiates the connection by sending a special HTTP request to the compromised server. The backdoor recognizes the attacker request, extracts and executes the embedded backdoor commands, and modifies the HTTP response to include the command output.

The following backdoor commands are supported:

- Get system information
- Upload/download files
- Execute files or shell commands
- Create a reverse shell
- Create/list/move/rename/delete files and folders
- Create a mapping between a local and a remote drive
- Exfiltrate collected data

IISpy ignores all other HTTP requests sent to the compromised IIS server by its legitimate visitors – of course, these are still handled by the benign server modules.

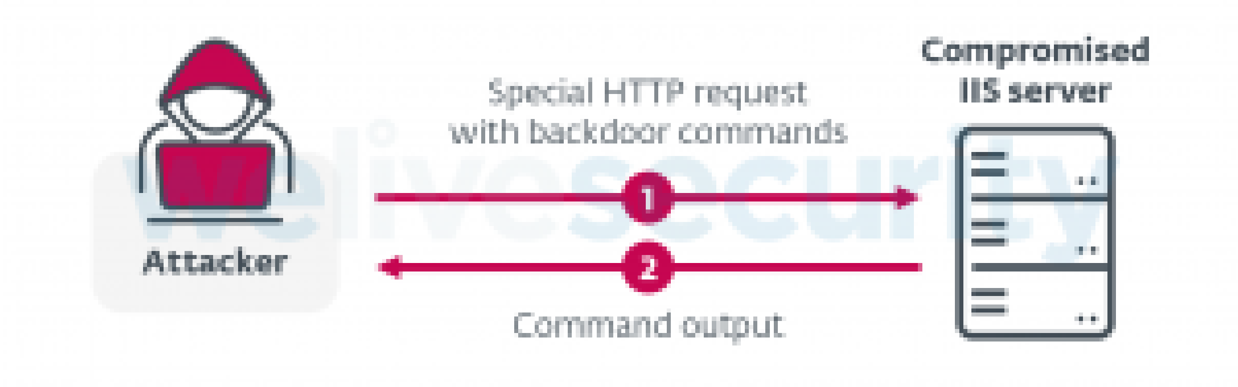


Figure 1. IISpy backdoor control mechanism

Network communication

The control requests from IISpy's operators have a predefined structure, with a specific (hidden) relationship between the Cookie and Host headers, and the URL. To identify such requests, IISpy first computes the MD5 hash of both the URL and Host header of an inbound HTTP request, and splits each MD5 into four double words:

- <h0><h1><h2><h3> = md5(Host Header value)
- <r0><r1><r2><r3> = md5(Raw URL value)

Then, it verifies that the Cookie header contains a substring built from these values:

> <r1><h2>=<h3><r2><r3><r0><h0><h1>

Figure 2 illustrates how this substring is assembled. Backdoor commands are embedded in the HTTP body, AES-CBC encrypted and base64 encoded.

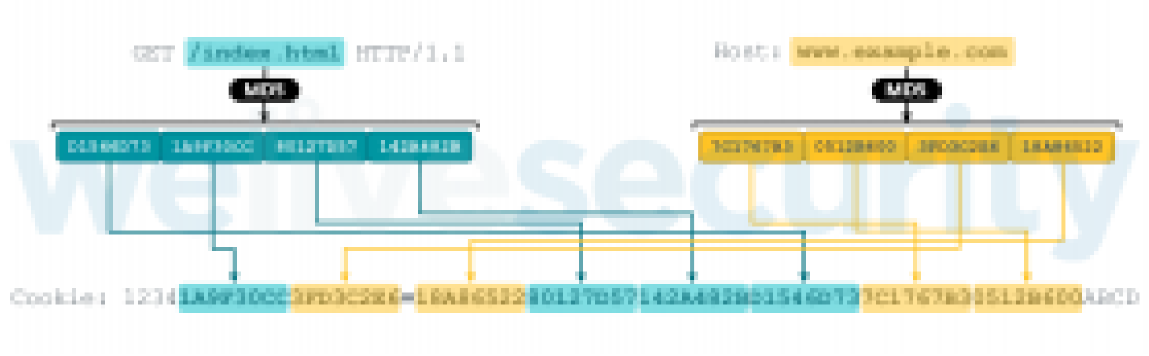


Figure 2. IISpy control HTTP request format

Note that this structure of control requests is unique to IISpy: all the other known IIS backdoors (that we have documented in our white paper *Anatomy of native IIS malware*) are controlled by hardcoded passwords, specific URIs or custom HTTP headers. As opposed to those “secrets”, IISpy’s control requests are more difficult to fingerprint and find in logs, which is an attempt to keep its C&C communication unnoticed.

Another such trick is used for the other side of the communication: IISpy embeds its encrypted and encoded response within a fake PNG image, between the PNG file headers as a TEXT or BLOB chunk. To reply to a control HTTP request, IISpy replaces the original HTTP response body (sent by the IIS server) with the fake PNG file, and sets the Content-Type header to image/png to give more credibility to this charade.

Both sides of the C&C communication are AES-CBC encrypted and base64 encoded, using these parameters:

- Encryption key: DA1F8BE19D9122F6499D72B90299CAB080E9D599C57E802CD667BF53CCC9EAB2
- IV: 668EDC2D7ED614BF8F69FF614957EF83EE

Technical analysis

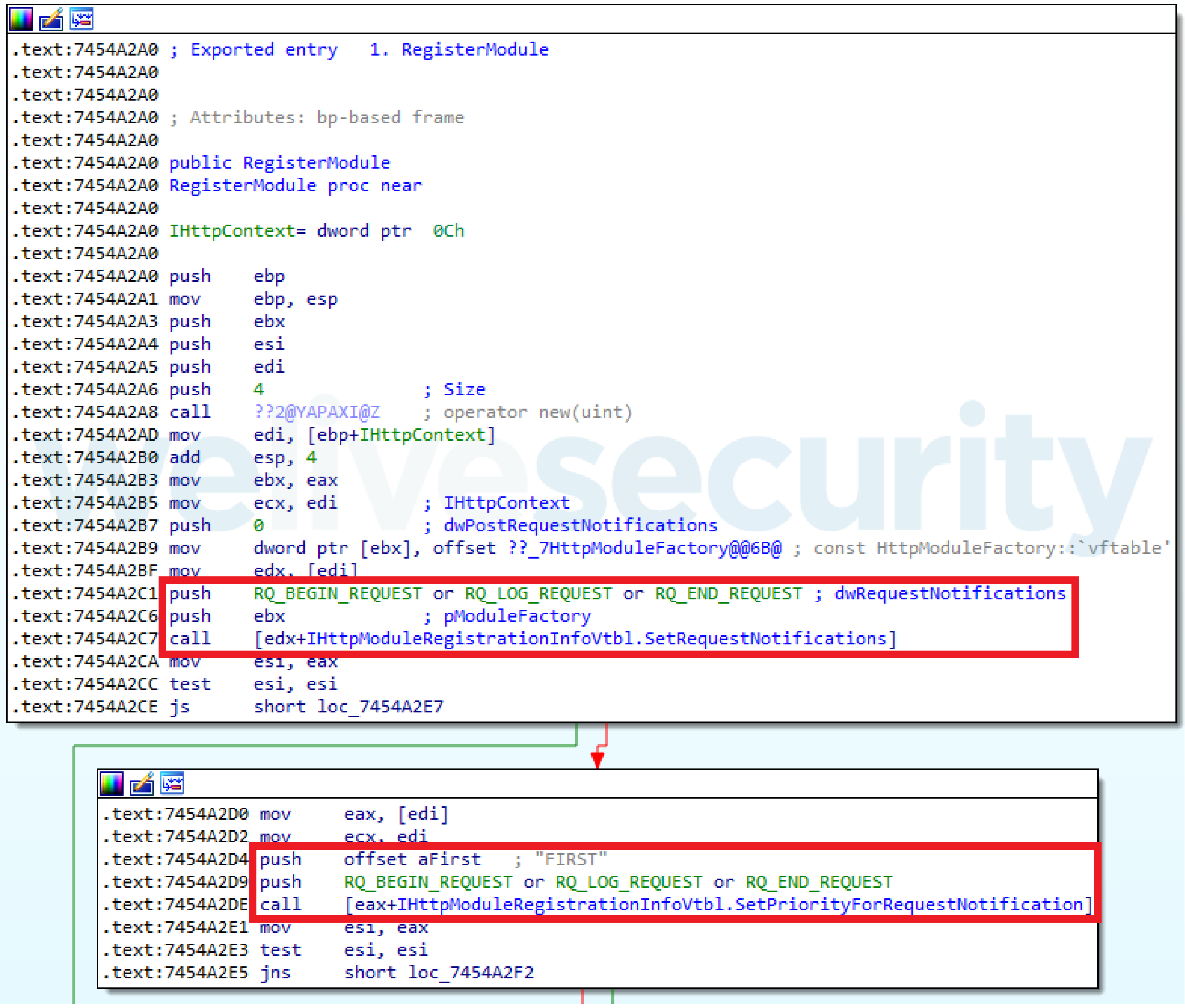
From the technical standpoint, IISpy is implemented as a native IIS module – a C++ DLL deployed in the %windir%\system32\inetsrv\ or the %windir%\SysWOW64\inetsrv folder on the compromised IIS server, under the name cache.dll or logging.dll.

IISpy is configured as an IIS extension in the %windir%\system32\inetsrv\config\ApplicationHost.config configuration file, and so it is loaded automatically by the IIS Worker Process (w3wp.exe), which handles all requests sent to the IIS web server. As far as execution and persistence goes, configuring IISpy as an IIS module itself checks all the boxes – all that’s left to implement inside the malicious module is the actual request processing (and as a bonus, a few anti-detection and anti-forensic tricks). We cover both in this section.

Module design

IISpy is written using the IIS C++ API, and uses instances of IHttpContext, IHttpRequest and IHttpResponse interfaces to parse HTTP requests and manipulate the HTTP responses.

As required by all native IIS modules, it exports a function called RegisterModule, where it creates an instance of its core classes and registers their methods for server events using the IHttpModuleRegistrationInfo::SetRequestNotifications method, as shown in Figure 3.



```
.text:7454A2A0 ; Exported entry   1. RegisterModule
.text:7454A2A0
.text:7454A2A0
.text:7454A2A0 ; Attributes: bp-based frame
.text:7454A2A0
.text:7454A2A0 public RegisterModule
.text:7454A2A0 RegisterModule proc near
.text:7454A2A0
.text:7454A2A0 IHttpContext= dword ptr  0Ch
.text:7454A2A0
.text:7454A2A0 push    ebp
.text:7454A2A1 mov     ebp, esp
.text:7454A2A3 push    ebx
.text:7454A2A4 push    esi
.text:7454A2A5 push    edi
.text:7454A2A6 push    4               ; Size
.text:7454A2A8 call    ??2@YAPAXI@Z    ; operator new(uint)
.text:7454A2AD mov     edi, [ebp+IHttpContext]
.text:7454A2B0 add     esp, 4
.text:7454A2B3 mov     ebx, eax
.text:7454A2B5 mov     ecx, edi        ; IHttpContext
.text:7454A2B7 push    0               ; dwPostRequestNotifications
.text:7454A2B9 mov     dword ptr [ebx], offset ??_7HttpModuleFactory@@6B@ ; const HttpModuleFactory::`vftable'
.text:7454A2BF mov     edx, [edi]
.text:7454A2C1 push    RQ_BEGIN_REQUEST or RQ_LOG_REQUEST or RQ_END_REQUEST ; dwRequestNotifications
.text:7454A2C6 push    ebx             ; pModuleFactory
.text:7454A2C7 call    [edx+IHttpModuleRegistrationInfoVtbl.SetRequestNotifications]
.text:7454A2CA mov     esi, eax
.text:7454A2CC test    esi, esi
.text:7454A2CE js      short loc_7454A2E7
```

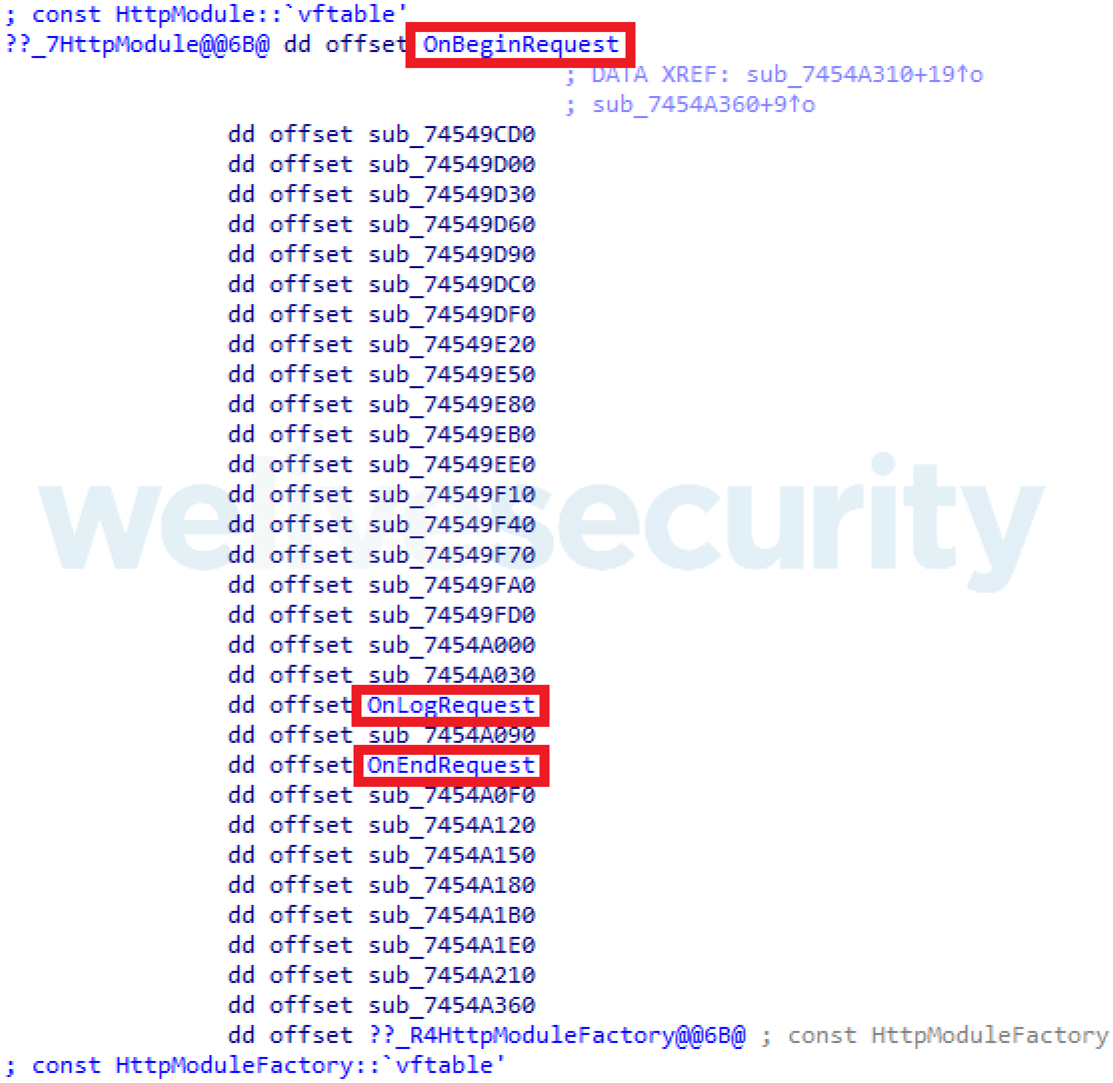
```
.text:7454A2D0 mov     eax, [edi]
.text:7454A2D2 mov     ecx, edi
.text:7454A2D4 push    offset aFirst   ; "FIRST"
.text:7454A2D9 push    RQ_BEGIN_REQUEST or RQ_LOG_REQUEST or RQ_END_REQUEST
.text:7454A2DE call    [eax+IHttpModuleRegistrationInfoVtbl.SetPriorityForRequestNotification]
.text:7454A2E1 mov     esi, eax
.text:7454A2E3 test    esi, esi
.text:7454A2E5 jns     short loc_7454A2F2
```

Figure 3. IISpy's RegisterModule export

IISpy's core class is inherited from CHttpModule and, as seen in Figure 4, overrides three of its methods – event handlers for the server events:

- OnBeginRequest is called every time the server starts processing a new HTTP request, and IISpy uses this handler to parse it in search of attacker requests
- OnEndRequest, called with the last step within the HTTP request-processing pipeline, implements IISpy's backdoor interpreter
- OnLogRequest, called right before the IIS server logs a processed HTTP request, implements IISpy's anti-logging feature

IISpy registers these handlers with the highest priority (via the IHttpModuleRegistrationInfo::SetPriorityForRequestNotification API). Since several IIS modules (malicious and regular) can be registered for the same event, this ensures that IISpy's handler will be executed before any other handlers registered for the same event.



```
; const HttpModule::`vftable'
??_7HttpModule@@6B@ dd offset OnBeginRequest
                                        ; DATA XREF: sub_7454A310+19↑o
                                        ; sub_7454A360+9↑o
                dd offset sub_74549CD0
                dd offset sub_74549D00
                dd offset sub_74549D30
                dd offset sub_74549D60
                dd offset sub_74549D90
                dd offset sub_74549DC0
                dd offset sub_74549DF0
                dd offset sub_74549E20
                dd offset sub_74549E50
                dd offset sub_74549E80
                dd offset sub_74549EB0
                dd offset sub_74549EE0
                dd offset sub_74549F10
                dd offset sub_74549F40
                dd offset sub_74549F70
                dd offset sub_74549FA0
                dd offset sub_74549FD0
                dd offset sub_7454A000
                dd offset sub_7454A030
                dd offset OnLogRequest
                dd offset sub_7454A090
                dd offset OnEndRequest
                dd offset sub_7454A0F0
                dd offset sub_7454A120
                dd offset sub_7454A150
                dd offset sub_7454A180
                dd offset sub_7454A1B0
                dd offset sub_7454A1E0
                dd offset sub_7454A210
                dd offset sub_7454A360
                dd offset ??_R4HttpModuleFactory@@6B@ ; const HttpModuleFactory
; const HttpModuleFactory::`vftable'
```

Figure 4. IISpy's core class implements three event handlers

Backdoor commands

In its OnEndRequest handler, IISpy decrypts the HTTP body of an attacker's request and extracts its parameters, which are organized as key-value pairs and listed in Table 1.

Table 1. IISpy attacker request parameters

Key	Value
/mode	Command type

Key	Value
/action	Command
/path /binary /data …	Command arguments (see Table 2 for full list)
/credential/username	Local user username, used for impersonation
/credential/password	Local user password, used for impersonation

If the credentials are present, IISpy uses them to log in as the user (via LogonUserW, ImpersonateLoggedOnUser) to execute the backdoor commands in the user's context. The backdoor commands and arguments are also organized as nested key-value pairs, as listed in Table 2.

Table 2. IISpy backdoor commands and arguments

Command type (/mode value)	Command (/action value)	Arguments (key names)	Command description	Returned data (map structure or description)
init	N/A	N/A	Collects basic system information: computer name and domain, username and domain, logical drives information.	/computer/domain /computer/name /user/domain /user/name /- /name /type
file	list	/path	Collects information about the files in the specified folder.	/- /name /attr /size /create /access /write
get	/path /binary	Downloads the file with the specified name from the compromised IIS server.	The contents of the file, encrypted and embedded within a fake PNG image (a PNG header followed by non-image data).	

Command type (/mode value)	Command (/action value)	Arguments (key names)	Command description	Returned data (map structure or description)
create	/path /directory /data	Creates a new file or directory in the specified path. Optional /data argument can hold the file content.	/- /file /attr /size /create /access /write	
upload	/path /data	Uploads a file with the specified name to the compromised server. The /data entry contains base64-encoded file content.	/- /file /attr /size /create /access /write	
delete	/path /files /name /attr	Deletes the list of files/directories in the given path.	/files /code /name	
move	/path /dest /copy /files /name /new	Copies or renames files from the list, from the source directory to the destination directory.	/files /code /name	
time	/path /create /access /write	Modifies file timestamps	N/A	
drive	map	/letter /share /username /password	Creates a mapping between a local and a remote drive, using the specified credentials for the network resource.	N/A
remove	/letter	Removes an existing drive mapping	N/A	

Command type (/mode value)	Command (/action value)	Arguments (key names)	Command description	Returned data (map structure or description)
cmd	exec	/cmd	Executes the specified command, either under the context of the current user, or the user provided in arguments. Returns the command output.	/output

After executing the backdoor command, IISpy encrypts and encodes its return data and uses it to modify the HTTP response to the attacker's request. The return data is also organized as key-value pairs, with the entries listed in Table 2, plus two additional entries based on the GetLastError result (or custom error messages):

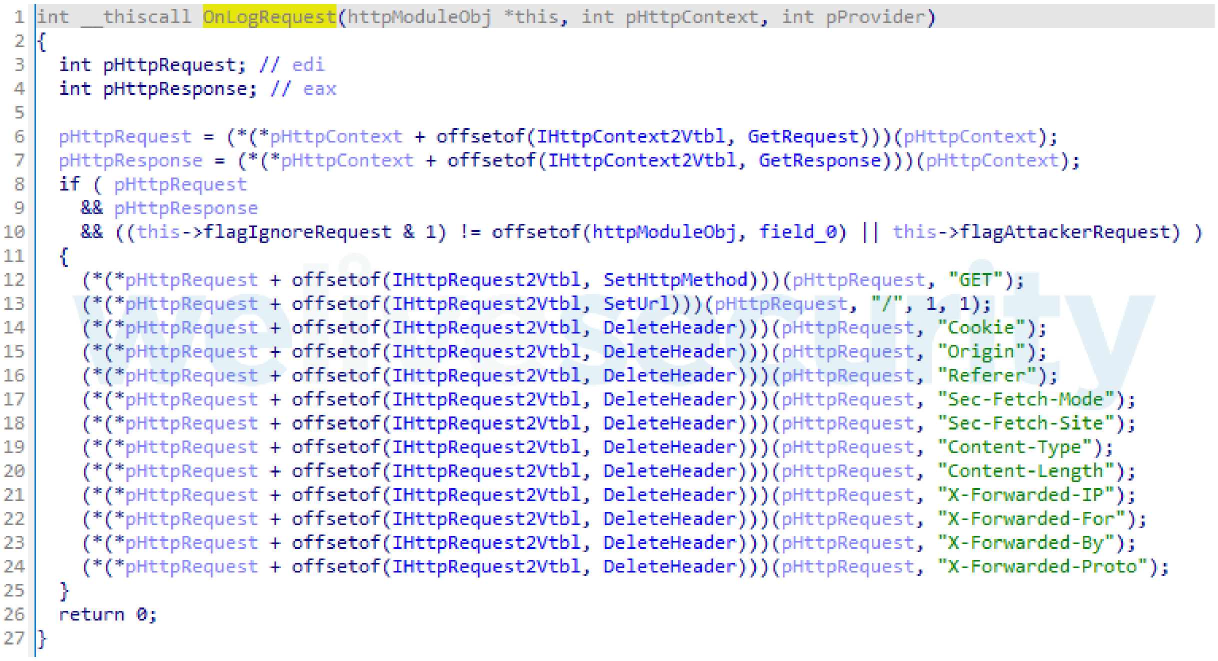
- /error/code
- /error/message

Anti-logging feature

Finally, IISpy implements the OnLogRequest event handler – called right before the IIS server logs a processed HTTP request. The backdoor uses this handler to modify the log entries for requests coming from the attackers to make them look like casual requests. As shown in Figure 5, these steps are taken:

- Rewrite the HTTP method in the request to GET
- Rewrite the URL from the request to /
- Delete these headers from the request: Cookie, Origin, Referer, Sec-Fetch-Mode, Sec-Fetch-Site, Content-Type, Content-Length, X-Forwarded-IP, X-Forwarded-For, X-Forwarded-By, X-Forwarded-Proto

With the log entries modified this way, the attackers attempt to further hide traces of their malicious activities, to make potential forensic analysis more difficult.



```
int __thiscall OnLogRequest(httpModuleObj *this, int pHttpContext, int pProvider)
{
  int pHttpRequest; // edi
  int pHttpResponse; // eax

  pHttpRequest = (*(*pHttpContext + offsetof(IHttpContext2Vtbl, GetRequest)))(pHttpContext);
  pHttpResponse = (*(*pHttpContext + offsetof(IHttpContext2Vtbl, GetResponse)))(pHttpContext);
  if ( pHttpRequest
    && pHttpResponse
    && ((this->flagIgnoreRequest & 1) != offsetof(httpModuleObj, field_0) || this->flagAttackerRequest) )
  {
    (*(*pHttpRequest + offsetof(IHttpRequest2Vtbl, SetHttpMethod)))(pHttpRequest, "GET");
    (*(*pHttpRequest + offsetof(IHttpRequest2Vtbl, SetUrl)))(pHttpRequest, "/", 1, 1);
    (*(*pHttpRequest + offsetof(IHttpRequest2Vtbl, DeleteHeader)))(pHttpRequest, "Cookie");
    (*(*pHttpRequest + offsetof(IHttpRequest2Vtbl, DeleteHeader)))(pHttpRequest, "Origin");
    (*(*pHttpRequest + offsetof(IHttpRequest2Vtbl, DeleteHeader)))(pHttpRequest, "Referer");
    (*(*pHttpRequest + offsetof(IHttpRequest2Vtbl, DeleteHeader)))(pHttpRequest, "Sec-Fetch-Mode");
    (*(*pHttpRequest + offsetof(IHttpRequest2Vtbl, DeleteHeader)))(pHttpRequest, "Sec-Fetch-Site");
    (*(*pHttpRequest + offsetof(IHttpRequest2Vtbl, DeleteHeader)))(pHttpRequest, "Content-Type");
    (*(*pHttpRequest + offsetof(IHttpRequest2Vtbl, DeleteHeader)))(pHttpRequest, "Content-Length");
    (*(*pHttpRequest + offsetof(IHttpRequest2Vtbl, DeleteHeader)))(pHttpRequest, "X-Forwarded-IP");
    (*(*pHttpRequest + offsetof(IHttpRequest2Vtbl, DeleteHeader)))(pHttpRequest, "X-Forwarded-For");
    (*(*pHttpRequest + offsetof(IHttpRequest2Vtbl, DeleteHeader)))(pHttpRequest, "X-Forwarded-By");
    (*(*pHttpRequest + offsetof(IHttpRequest2Vtbl, DeleteHeader)))(pHttpRequest, "X-Forwarded-Proto");
  }
  return 0;
}
```

Figure 5. IISpy modifies log entries for attacker requests

Conclusion

IISpy is a complex server-side backdoor misusing the extensibility of IIS web server software for its persistence, execution and C&C mechanisms. With its tricks to blend in with the regular network traffic, and to clear incriminating logs, it is designed for long term espionage on compromised IIS servers.

Organizations that handle sensitive data on their servers should be on the lookout, such as organizations that have the Outlook on the web (OWA) service enabled on their Exchange email servers – OWA is implemented via IIS, and makes an interesting target for espionage. In any case, the best way to keep IISpy out of your servers is to keep them up to date, and carefully consider which services are exposed to the internet, to reduce the risk of server exploitation.

Additional technical details on the malware, Indicators of Compromise and YARA rules can be found in our comprehensive white paper, and on GitHub. For any inquiries, or to make sample submissions related to the subject, contact us at: threatintel@eset.com.

Indicators of Compromise (IoCs)

ESET detection names

Win32/BadIIS.F
Win64/BadIIS.U

SHA-1

22F8CA2EB3AF377E913B6D06B5A3618D294E4331
435E3795D934EA8C5C7F4BCFEF2BEEE0E3C76A54
CED7BC6E0F1A15465E61CFEC87AAEF98BD999E15

Filenames

cache.dll
logging.dll

MITRE ATT&CK techniques

Note: This table was built using version 9 of the MITRE ATT&CK framework.

Tactic	ID	Name	Description
Resource Development	T1587.001	Develop Capabilities: Malware	IISpy is a custom-made malware family.
T1588.002	Obtain Capabilities: Tool	Operators of IISpy have used Juicy Potato , a local privilege escalation tool.	
Initial Access	T1190	Exploit Public-Facing Application	IISpy likely obtains its initial access to the IIS server via some vulnerability in the web application or on the server, before it uses the privilege escalation tool Juicy Potato to obtain the administrative privileges that are required to install a native IIS module.
Execution	T1059.003	Command and Scripting Interpreter: Windows Command Shell	IISpy supports a backdoor command that uses the Windows command shell to execute shell commands on the compromised IIS server.
T1569.002	System Services: Service Execution	IIS server (and by extension, IISpy) persists as a Windows service.	
Persistence	T1546	Event Triggered Execution	IISpy is loaded by IIS Worker Process (w3wp.exe) when the IIS server receives an inbound HTTP request.

Tactic	ID	Name	Description
Privilege Escalation	T1068	Exploitation for Privilege Escalation	Operators of IISpy have used a local privilege escalation tool Juicy Potato to elevate privileges.
Defense Evasion	T1134.001	Access Token Manipulation: Token Impersonation/Theft	IISpy has the ability to execute backdoor commands in another user’s context (via LogonUserW, ImpersonateLoggedOnUser).
T1070	Indicator Removal on Host	IISpy has the ability to sanitize logging of attacker requests on the IIS server.	
T1070.006	Indicator Removal on Host: Timestomp	IISpy supports a backdoor command to modify file timestamps.	
Collection	T1005	Data from Local System	IISpy supports a backdoor command to collect and exfiltrate files from the compromised IIS server.
Command and Control	T1071.001	Application Layer Protocol: Web Protocols	IISpy is a passive network implant: Adversaries send HTTP requests to the compromised IIS server to control the backdoor.
T1001	Data Obfuscation	IISpy operators send commands with a specially constructed combination of URLs, Host headers and cookies. IISpy exfiltrates data in a fake PNG file (a PNG header followed by non-image data), in an attempt to make its C&C traffic look like regular network traffic.	

Tactic	ID	Name	Description
	T1132.001	Data Encoding: Standard Encoding	IISpy encodes the C&C communication with base64 encoding.
	T1573.001	Encrypted Channel: Symmetric Cryptography	IISpy uses AES-CBC to encrypt C&C communication.
	T1105	Ingress Tool Transfer	IISpy supports a backdoor command to upload additional tools to the compromised IIS server.
Exfiltration	T1041	Exfiltration Over C2 Channel	IISpy supports a backdoor command to exfiltrate data and files from the compromised IIS server.

9 Aug 2021 - 11:30AM

Sign up to receive an email update whenever a new article is published in our Ukraine Crisis – Digital Security Resource Center

Newsletter

Discussion