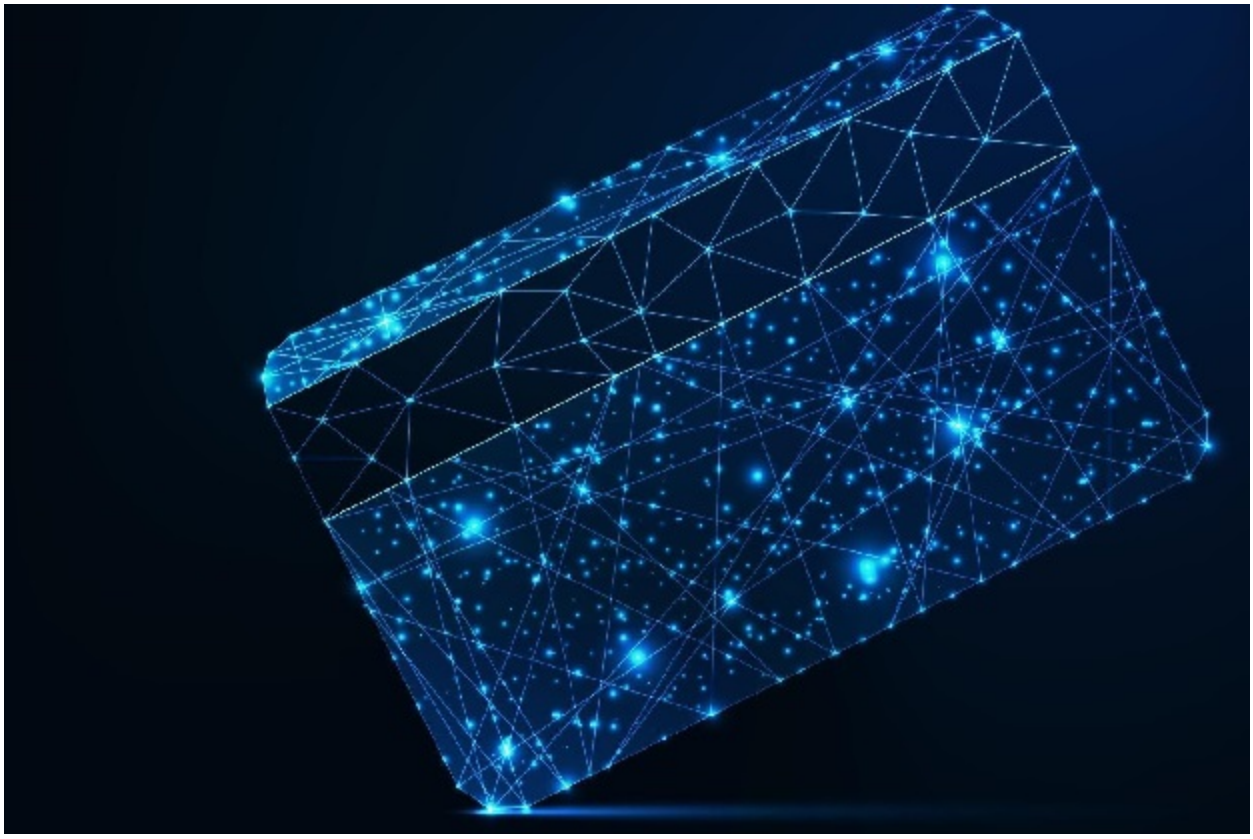
IIStealer: A server-side threat to e-commerce transactions

welivesecurity.com/2021/08/06/iistealer-server-side-threat-ecommerce-transactions/

August 6, 2021



The first in our series on IIS threats looks at a malicious IIS extension that intercepts server transactions to steal credit card information

Zuzana Hromcová
6 Aug 2021 - 03:00PM

The first in our series on IIS threats looks at a malicious IIS extension that intercepts server transactions to steal credit card information

ESET researchers have discovered and analyzed a previously undocumented trojan that steals payment information from e-commerce websites' customers. The trojan, which we named IIStealer, is detected by ESET security solutions as Win64/BadIIS.

This blogpost is the first installment in our series where ESET researchers put IIS web server threats under the microscope, with the other two parts discussing IIS malware used for cyberespionage 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 IIStealer is featured as one of the studied families (Group 5).

Anatomy of native IIS malware

Download Research Paper

Attack overview

IIStealer is implemented as a malicious extension for *Internet Information Services* (IIS), Microsoft web server software. Being a part of the server, IIStealer is able to access all the network communication flowing through the server and steal data of interest to the attackers – in this case, payment information from e-commerce transactions.

As illustrated in Figure 1, IIStealer operates by intercepting regular traffic between the compromised server and its clients (the seller and the buyers), targeting HTTP POST requests made to specific URI paths: /checkout/checkout.aspx or /checkout/Payment.aspx.

Whenever a legitimate website visitor makes a request to these checkout pages (1), IIStealer logs the HTTP request body into a log file (2), without, in any way, interfering with the HTTP reply generated by the components of the legitimate website (3).

Adversaries can then exfiltrate the collected data by making a special HTTP request to the compromised IIS server: once IIStealer detects a request made to a specific URI (/privacy.aspx) with an attacker password included in the X-IIS-Data header (4), it embeds the collected data in the HTTP response for that request (5,6).

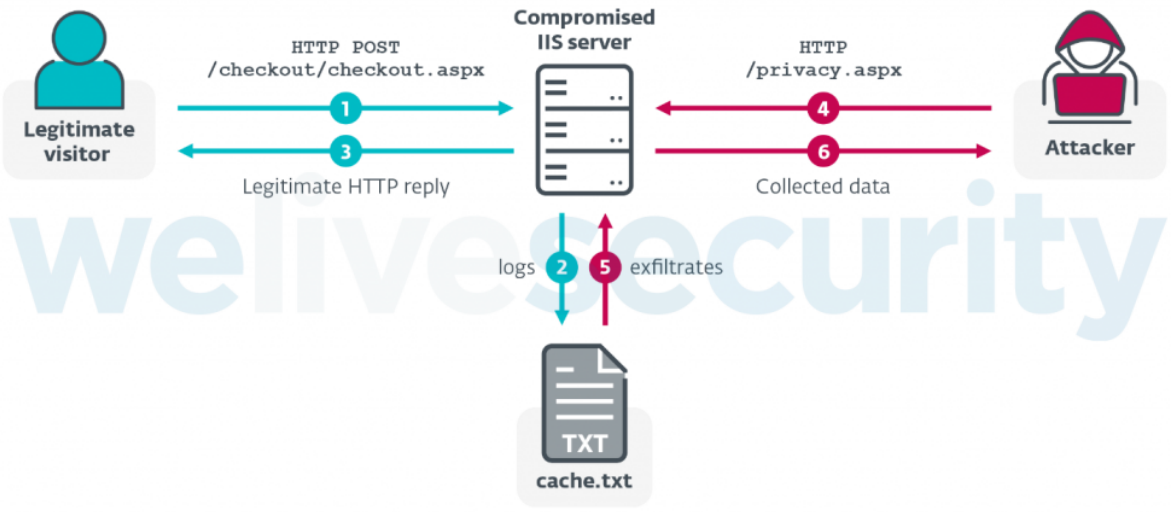


Figure 1. IIStealer: collection and exfiltration mechanisms

With these capabilities, IIStealer is able to steal credit card information sent to e-commerce websites that don't use third-party payment gateways. Note that SSL/TLS and encrypted communication channels don't secure these transactions against IIStealer, as the malware can access all data handled by the server – which is where the credit card information is processed in its unencrypted state.

The samples of this malware that we analyzed seem to be tailored for specific e-commerce websites (with hardcoded checkout page URIs). According to our telemetry, targeted were a small number of IIS servers in the USA, between September 2020 and January 2021, but this is likely affected by our limited visibility into IIS servers – it is still common for administrators to not use any security software on these servers.

Technical analysis

IIStealer is implemented as a malicious, native IIS module – a C++ DLL dropped in the %windir%\system32\inetsrv\ folder on the compromised IIS server and configured in the %windir%\system32\inetsrv\config\ApplicationHost.config file. In some cases, IIStealer is deployed under the name dir.dll and, as seen in Figure 2, uses a forged VERSIONINFO resource to mimic a legitimate Windows IIS module called dirlist.dll.

dir.dll (malicious)

```
1 VERSIONINFO
FILEVERSION 10,0,17763,1
PRODUCTVERSION 10,0,17763,1
FILEOS 0x40004
FILETYPE 0x2
{
BLOCK "StringFileInfo"
{
        BLOCK "000004B0"
        {
                VALUE "CompanyName", "Microsoft Corporation"
                VALUE "FileDescription", "Directory Listing handler"
                VALUE "FileVersion", "10.0.17763.1 (WinBuild.160101.0800)"
                VALUE "InternalName", "dirlist.dll"
                VALUE "LegalCopyright", "© Microsoft Corporation. All rights reserved."
                VALUE "OriginalFilename", "dirlist.dll"
                VALUE "ProductName", "Internet Information Services"
                VALUE "ProductVersion", "10.0.17763.1"
        }
}

BLOCK "VarFileInfo"
{
        VALUE "Translation", 0x0000 0x04B0
}
}
```

dirlist.dll (benign)

```
1 VERSIONINFO
FILEVERSION 10,0,18362,1
PRODUCTVERSION 10,0,18362,1
FILEOS 0x40004
FILETYPE 0x2
{
BLOCK "StringFileInfo"
{
        BLOCK "000004B0"
        {
                VALUE "CompanyName", "Microsoft Corporation"
                VALUE "FileDescription", "Directory Listing handler"
                VALUE "FileVersion", "10.0.18362.1 (WinBuild.160101.0800)"
                VALUE "InternalName", "dirlist.dll"
                VALUE "LegalCopyright", "© Microsoft Corporation. All rights reserved."
                VALUE "OriginalFilename", "dirlist.dll"
                VALUE "ProductName", "Internet Information Services"
                VALUE "ProductVersion", "10.0.18362.1"
        }
}

BLOCK "VarFileInfo"
{
        VALUE "Translation", 0x0000 0x04B0
}
}
```

Figure 2. IIStealer's VERSIONINFO resource (left) mimics legitimate dirlist.dll module (right)

Because it is an IIS module, IIStealer is loaded automatically by the IIS Worker Process (w3wp.exe), which handles the requests sent to the IIS web server – this is how IIStealer achieves persistence, and how it can affect the processing of incoming requests.

We don't have any information about how the malware is spread, but we know that administrative privileges are required to install it as a native IIS module, which narrows down the candidates for the initial compromise. A configuration weakness or vulnerability in a web application, or the server itself, are likely culprits.

As for its technical characteristics, IIStealer implements a core class inherited from CHttpModule (module class) and overrides the CHttpModule::OnPostBeginRequest method with its malicious code. As with all native IIS modules, IIStealer exports a function named RegisterModule (see Figure 3), where it instantiates the module class and registers its methods for server events – more specifically, it registers for the RQ_BEGIN_REQUEST post-event notification that is generated every time the server starts processing an inbound HTTP request. As a result, the OnPostBeginRequest method is called with each new request, which allows IIStealer to affect the request processing.

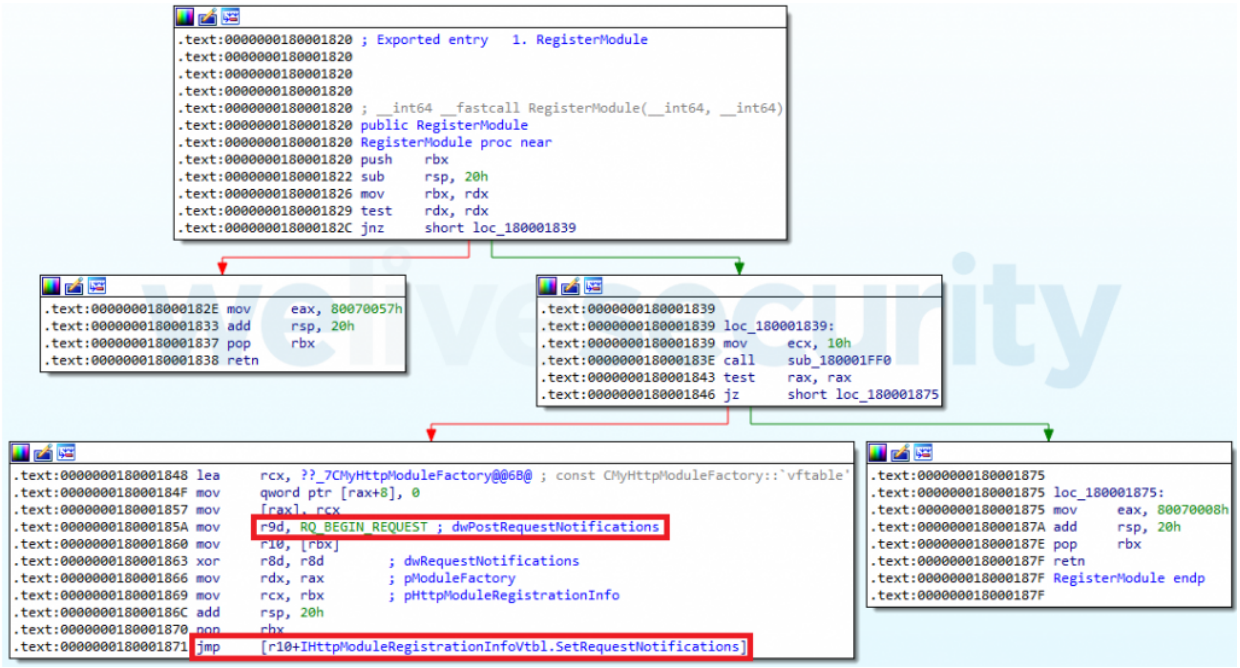


Figure 3. IIStealer's RegisterModule entry point

In the OnPostBeginRequest handler, IIStealer filters incoming HTTP requests by request URIs. All POST requests made to /checkout/checkout.aspx or /checkout/Payment.aspx are logged – along with their full HTTP bodies – into a file named C:\Windows\Temp\cache.txt. These requests are made by legitimate visitors of the compromised e-commerce websites and can contain sensitive information such as personal details and credit card numbers.

The collected data can be exfiltrated via a specifically crafted HTTP request from the attacker. This request must have an X-IIS-Data HTTP header set to a hardcoded, 32-byte alphanumeric password (that we have chosen not to disclose), and must be sent to a URL path specified in the malware sample:

- /privacy.aspx
- /checkout/Payment.aspx

Once the malicious module detects such a request, it uses the IHttpResponse::Clear method to delete any HTTP response prepared by the IIS server, and copies the unencrypted contents of the log file into the HTTP response body using the IHttpResponse::WriteEntityChunks API function, as seen in Figure 4.

```
75  hFile1 = CreateFileA(logFileName, 3u, 3u, 0i64, 3u, 0x80u, 0i64);
76  hFile2 = hFile1;
77  if ( hFile1 != -1i64 )
78  {
79    fileSize1 = GetFileSize(hFile1, 0i64);
80    logFileSize = fileSize1;
81    if ( fileSize1 == -1 )
82      goto LABEL_21;
83    buffer = HeapAlloc(hHeap, 8u, fileSize1);
84    logFileContent = buffer;
85    if ( !buffer )
86      goto LABEL_21;
87    if ( !ReadFile(hFile2, buffer, logFileSize, &numberOfBytesRead, 0i64) )
88    {
89      HeapFree(hHeap, 0, logFileContent);
90      v16 = hFile2;
91LABEL_13:
92      CloseHandle(v16);
93      return 0i64;
94    }
95    pHttpResponse1 = (*(*pHttpContext + offsetof(IHttpContext2Vtbl, GetResponse)))(pHttpContext);
96    pHttpResponse2 = pHttpResponse1;
97    if ( !pHttpResponse1 )
98    {
99LABEL_21:
100     CloseHandle(hFile2);
101     if ( logFileContent )
102       HeapFree(hHeap, 0, logFileContent);
103     return 0i64;
104   }
105   (*(*pHttpResponse1 + offsetof(IHttpResponse, Clear)))(pHttpResponse1);
106   IHttpResponse = *pHttpResponse2;
107   httpDataChunk.DataChunkType = HttpDataChunkFromMemory;
108   httpDataChunk.FromFileHandle.ByteRange.StartingOffset.QuadPart = logFileContent;
109   httpDataChunk.FromMemory.BufferLength = logFileSize;
110   v20 = (IHttpResponse->WriteEntityChunks)(pHttpResponse2, &httpDataChunk, 1i64);
111   HeapFree(hHeap, 0, logFileContent);
112   v16 = hFile2;
113   if ( v20 < 0 )
114     goto LABEL_13;
115   if ( SetFilePointer(hFile2, 0, 0i64, 0) != -1 )
116   {
117     if ( SetEndOfFile(hFile2) )
118       FlushFileBuffers(hFile2);
119   }
120   CloseHandle(hFile2);
121   return 2i64;
122 }
123 return 0i64;
124}
```

Figure 4. IIStealer replaces the HTTP response body with its own data

This allows the operators of IIStealer to access and exfiltrate the collected data by simply sending a special request to the compromised IIS server – there is no need for the malware to implement additional C&C channels, or embed any C&C server domains in its configuration.

Mitigation

IIStealer is a server-side threat that eavesdrops on the communications between a compromised e-commerce website and its customers, with the goal of stealing sensitive payment information – but of course, malicious IIS modules can also target credentials and other information. Even though SSL/TLS is vital in securing the transmission of the data between the client and the server, it doesn't prevent this attack scenario as IIStealer is a part of the server. This should be disturbing for all serious web portals that want to protect their visitors' data, including authentication and payment information.

The best way to harden an IIS server against IIStealer and other threats is to:

- Use dedicated accounts with strong, unique passwords for the administration of the IIS server.
- Regularly patch your OS, and carefully consider which services are exposed to the internet, to reduce the risk of server exploitation.
- Only install native IIS modules from trusted sources.
- Consider using a web application firewall, and/or endpoint security solution on your IIS server.
- Regularly check the configuration file %windir%\system32\inetsrv\config\ApplicationHost.config, as well as the %windir%\system32\inetsrv\ and %windir%\SysWOW64\inetsrv folders to verify that all the installed native modules are legitimate (signed by a trusted provider, or installed on purpose).

For web developers: Even if you don't have control over the IIS server where your web service is hosted, you can still take steps to reduce the impact on users of your web service in the case of a compromise, especially:

- Do not send the password itself to the server (not even over SSL/TLS); use a protocol such as Secure Remote Password (SRP) to authenticate users without the need for the unencrypted password to be transmitted to the server, nor data that could be used to reauthenticate. IIS infostealers are a good example of why server-side hashing is not good enough.
- Avoid unnecessarily sending sensitive information from the web application; use payment gateways.

- If you identify a successful compromise: notify all parties involved in any security breach so they can take quick action.

For consumers: from the visitor's perspective, it is impossible to know whether an IIS server is compromised, but these tips will help you reduce the risk:

- Be careful about where you enter your credit card number. Consider using payment gateways by trusted third-party providers on e-commerce websites whose reputation is unknown to you: with payment gateways, such websites won't handle the sensitive payment information.
- Keep an eye on your credit statement for small or unusual payments: often small amounts are processed to test whether the cards are valid.
- If you spot something unusual, notify your bank immediately.

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.

Read next:
Anatomy of native IIS malware
IISpy: A complex server-side backdoor with anti-forensic features
IISerpent: Malware-driven SEO fraud as a service

Indicators of Compromise (IoCs)

ESET detection names

Win64/BadIIS.F
Win64/BadIIS.O

SHA-1

706EAB59C20FCC9FBC82C41BF955B5C49C644B38
7A2FA07A7DC05D50FE8E201A750A3DC7F22D6549
A1C5E7424E7C4C4C9902A5A1D97F708C6BB2F53A

Filenames and paths

dir.dll
isapicache___.dll
isapicache_.dll_
C:\Windows\Temp\cache.txt

Network indicators

Targeted URIs

/checkout/checkout.aspx
/checkout/Payment.aspx
/privacy.aspx

HTTP header

X-IIS-Data

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	IIStealer is a custom-made malware family.
Execution	T1569.002	System Services: Service Execution	IIS server (and by extension, IIStealer) persists as a Windows service.
Persistence	T1546	Event Triggered Execution	IIStealer is loaded by IIS Worker Process (w3wp.exe) when the IIS server receives an inbound HTTP request.
Defense Evasion	T1036.005	Masquerading: Match Legitimate Name or Location	IIStealer has been deployed under the name dir.dll, in an attempt to mimic a legitimate Microsoft IIS module called dirlist.dll.
T1027	Obfuscated Files or Information	IIStealer uses string stacking in an attempt to avoid some string-based detection.	
Credential Access	T1056	Input Capture	IIStealer intercepts network traffic between the IIS server and its clients to collect sensitive information such as credit card details.
Collection	T1119	Automated Collection	IIStealer automatically collects information from inbound HTTP requests, such as credit card details.

Tactic	ID	Name	Description
T1074.001	Data Staged: Local Data Staging	IIStealer uses a local file to stage collected information.	
Command and Control	T1071.001	Application Layer Protocol: Web Protocols	Adversaries send HTTP requests to the compromised IIS server to control IIStealer.
Exfiltration	T1041	Exfiltration Over C2 Channel	IIStealer uses its C&C channel to exfiltrate collected data: HTTP requests are sent by the adversary to the compromised IIS server.

6 Aug 2021 - 03:00PM

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

Newsletter

Discussion