BumbleBee a New Modular Backdoor Evolved From BookWorm

b trendmicro.com/en_us/research/22/i/buzzing-in-the-background-bumblebee-a-new-modular-backdoor-evolv.html

September 2, 2022

Updated on Sept. 6, 2022, at 11:55 p.m. ET to clarify the reason behind the naming of this BumbleeBee malware.

Updated on Sept. 2, 2022, at 9:55 p.m. ET to clarify the difference between this BumbleBee malware and the Bumblebee ransomware loader.

Introduction

In March 2021, we investigated a backdoor with a unique modular architecture. Its type of modular framework made our static analysis more challenging because it required us to first rebuild its structure or use dynamic analysis to understand its functionality and behavior.

We called it "BumbleBee" since the developer of this tool originally named it as such ("bumblebee" in Chinese: 大黄蜂).

Our analysis found that BumbleBee only had little malicious code in its payload, and what it does on the surface is track keys and clipboard content. However, further investigation revealed a controller application that expands the malware's capabilities.

This type of backdoor is similar to another of its kind called BookWorm, in which it can be inferred that BumbleBee is a refactored version of BookWorm. At the time of writing, BumbleBee has only been deployed in Taiwan; together with its use of Simplified Chinese as the language for its user interface, this malware can be suspected to be deployed by malicious Chinese actors. This blog will tackle BumbleBee's capabilities and our analysis of this backdoor. It's important to note that this BumbleBee malware family is different from the <u>Bumblebee loader</u>, a loader malware that is used by ransomware groups to drop backdoors to gain access to corporate networks.

BumbleBee – a refactored modular backdoor

BumbleBee is a modular backdoor that comprises two applications, a server and a client application (a master and slaver application, respectively in the malware's jargon). Once the client application is deployed on the target computer (these are commonly local government devices), threat actors can control the machine using the server module. Let us take a deeper look into this backdoor.

Layered deployment – client application

We have encountered the client application in a security breach incident. Its unique "layer-inlayer" architecture caught our attention. The module has a self-extracted file that contains three main parts: a legitimate executable (*XcrSvr.exe*), side-loaded DLL (*XecureIO_v20.dll*) and the shellcode binary file (ore) in the file system to execute the legitimate executable.

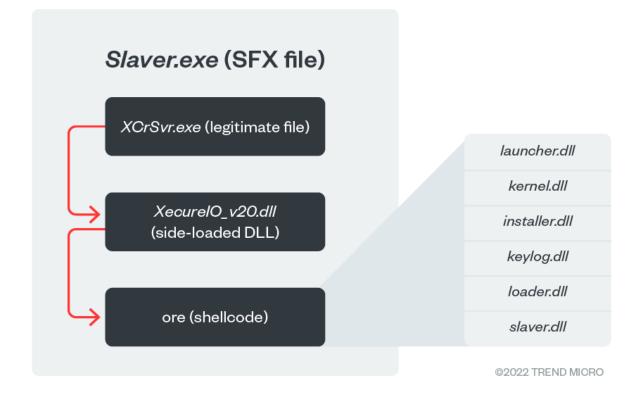


Figure 1. Architecture of BumbleBee

General	Compatibility	Digital Signature	
Security	Details	Previous Versions	
Language Original filename		ule m Co., Ltd.	Figure 2. Metadata of XcrSvr.exe

XCrSvr.exe is the executable in the XecureVistaCryptoSvr module developed by SoftForum. This file is exploited to launch the side-loaded DLL, *XecureIO_v20.dll*, which will work as the next-stage loader that executes the shellcode "ore," which is the main component in this backdoor. This shellcode contains multiple modules of its own (shown in Table 1). Each module has corresponding 32-bit and 64-bit versions of binaries in the shellcode except for *launcher.dll*.

Name	Description
launcher.dll	The first-stage launcher that loads all the subsequent modules. It decrypts a list of modules in memory and executes each in order.
kernel.dll	The utility component that controls all the other modules.
installer.dll	The module used to install components in the compromised machine.
keylog.dll	The keylog component monitors the keystrokes and clipboard content of the victim, and records actions from the victim such as running a process, entering a password, and getting the text of a window. The stolen data will then be run through a XOR logic gate with a two-byte key 0xF29D and saved under %temp%\kb\[UserName]\. The timestamp will be used as the file name.
loader.dll	The module that reads the shellcode.

slaver.dll The main module that interacts with the other methods once the backdoor is launched.

Table 1. BumbleBee's modules

If a victim is compromised for the first time, *launcher.dll* loads and launches all the other modules. The installer modules will be responsible for the installation and establishing persistence on the compromised machine via the following steps:

- 1. Drop a copy of the XecureIO_v20.dll in %APPDATA%\LOCAL\TEMP folder.
- Encrypt original shellcode file (to be a "bin" file) and path information (to be a "path" file) by using RC4 algorithm (key is the value of "ProductID" from "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Internet Explorer\Registration")
- 3. Drop bpu.dll (used to bypass UAC) and launched by rundll32.exe.
- 4. Establish persistence on compromised machine.
- 5. Delete the original SFX file.

Notably, as *XecureIO_v20.dll* is loaded by *XcrSvr.exe*, it will check if the parent process is *"XcrSvr.exe*." If so, it will patch the entry point of *XcrSvr.exe* with a long jump instruction to direct execution flow to the malicious code.

```
result = GetModuleHandleW(0);
base = (PIMAGE DOS HEADER)result;
if ( *(_WORD *)result != 'ZM' )
 return result;
v5 = base->e_lfanew;
nt_headers_1 = *(PIMAGE_NT_HEADERS *)((char *)&base->e_magic + v5);
result = (HMODULE)((char *)base + v5);
nt_headers = (PIMAGE_NT_HEADERS)((char *)base + v5);
if ( nt_headers_1 != (PIMAGE_NT_HEADERS)'EP' )
 return result;
entrypoint = (char *)base + nt_headers->OptionalHeader.AddressOfEntryPoint;// get entrypoint VA
result = (HMODULE)VirtualProtect(
                    (char *)base + nt headers->OptionalHeader.AddressOfEntryPoint,
                    0x10u,
                    0x40u,
                    &fl0ldProtect);
base = (PIMAGE_DOS_HEADER)result;
if ( !result )
 return result;
v3 = (char *)load_malicious_shellcode - (char *)entrypoint - 5;
entrypoint[1] = (char *)load_malicious_shellcode - (char *)entrypoint - 5;// patch entrypoint
entrypoint[4] = HIBYTE(v3);
                                              // call
*entrypoint = 0xE9;
entrypoint[2] = (unsigned __int16)((char *)load_malicious_shellcode - (char *)entrypoint - 5) >> 8;
entrypoint[3] = (unsigned int)((char *)load_malicious_shellcode - (char *)entrypoint - 5) >> 16;
return (HMODULE)VirtualProtect(entrypoint, 0x10u, floldProtect, &floldProtect);
```

Figure 3. XecureIO_v20.dll hooks its parent process' entry point

•				
$\rightarrow \bullet$	00401000	55	push ebp	EntryPoint
۰	00401001	8BEC	mov ebp,esp	
	00401003	83EC 44	sub esp,44	
•	00401006	56	push esi	
۰	00401007	FF15 DC504000	<pre>call dword ptr ds:[<&GetCommandLineA></pre>	
•	0040100D	8BF0	mov esi,eax	eax:&L"XCrSvr.exe"
<u> </u>	4 1			

Figure 4. The original entry point

•	00401000 00401005 00401006 00401007 0040100D	44 56	<pre>jmp xecureio_v20.10001CD0 inc esp push esi call dword ptr ds:[<&GetCommandLineA>] mov esi,eax</pre>	EntryPoint	Figure
- T 1		Lander in a link			

5. The patched entry point

Based on our analysis, we think the reason is that the malicious code embedded in XecureIO_v20 will not run if it followed the normal execution flow of *XCrSvr.exe*. Hence, once *XecureIO_v20.dll* is loaded by *XCrSvr.exe*, it will patch the entry point of *XCrSvr.exe* and jump to the address of the malicious code to make sure the code can be executed properly. After the client is installed and the persistence is established, the loader, *XecureIO_v20.dll*, will retrieve the value of "ProductID" from the registry key

"*HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Internet Explorer\Registration*" and use it as the key to decrypt the encrypted payload (the file "bin") dropped in the first installation. Using the information on the compromised machine as a key to encrypt the payload makes it much more difficult for analysts to decrypt and debug the malware in the analysis environment.

File Description name

path	An RC4-encrypted path string used to find the location of next-stage shellcode. It could be a file path or a registry path starting with HKLM or HKCU.

bin The next-stage RC4-encrypted shellcode payload.

Table 2. Payload file names

Expanded control – server application

Due to BumbleBee's complex client application, it took some time for us to fully analyze its functionality. While doing so, we ran across the server application of the malware that acts as a controller. This provided us with further understanding on how BumbleBee works. As the client application is running on the infected device, it will communicate with the server application and show the information of the machine it is in. Details, such as computer name, external IP address, geographic location, OS, CPU, and memory, are collected by the client application.

🔍 大黄蛇	\$													 @ <u>X</u>
文件(F)	操作(O) 帮助(H)												
-	_	ī	0,		Ô	<u></u>	Ð							
文件管理	里 屏幕控制	进程管理	服务管理	注册表编辑		交互式控制台	反向代理	键盘记录						
名称		外网地址	地理位置	描述	空闲 开机器	j间 操	作系统			CPU 内存	硬盘 屏幕	内网地址	杀毒软件	版本
- *	(未分组)													
		1.00							Intel(R) Core(TM) i7-8700 CPU @ 3.20			10.00	Windows Defender	
-					2022-0	6-13 08:43:54 Mi	crosoft Windov	ws 7 Professional 64-bit 6.1.7601	Intel(R) Core(TM) i7-8700 CPU @ 3.20	IGHz 1023MB		1000	Windows Defender	
时间		说明												
2022	2-06-13 15:00:35 2-06-13 14:52:57	新增主机												
2022	-06-13 14:52:57	主机	-	Ŀ	线.									
¶⊒k 2022	-06-13 14:42:39	主机	۲ ==	錢.										
을 2022	-06-13 14:09:32 -06-13 13:59:51 -06-13 13:58:43	新增主机	-											
2022	-06-13 13:59:51	新增主机												
\$ 2022	-06-13 13:58:43	http://		/update/ 监听F										
0														

Figure 6. Connection established

🔍 大黄蜂									- O X
文件(F) 操作(O) 帮助(H)							
_		a	0,		Ō	6	Ð		
文件管理	屏幕控制	进程管理	服务管理	注册表编辑	控制台面	冷 交互式控制	治 反向代理	键盘记录	
名称		外网地址	地理位置	描述	空闲	开机时间	操作系统		
- 🎎 (未分组))								

Figure 7. Built-in options in server application

Based on the options in the server application shown in Figure 7, we can determine that it supports the following functions for controlling the compromised machine:

Functions	Description
文件管理 (File management)	Upload/download/delete/list files from the victim's environment
屏幕控制 (Remote desktop control)	Control the victim's desktop remotely
进程管理 (Process management)	List and manage running processes with the image names, current folder, process id and parent process id
服务管理 (Service management)	List and manage current services status
注冊表编辑 (Registry editor)	List and manage the victim's registry key
控制台命令 (Command shell)	Execute the command shell

交互式控制台 (Interactive console)	Execute the command shell							
反向代理 (Reverse proxy)	Reverse proxy to help expose a local server behind a NAT or firewall to the internet							
键盘记录 (Keylogger)	Log keystrokes and clipboard contents							

Table 3. Supported functions

BumbleBee's modular framework allowed it to embed a small amount of malicious code that involves stealing keystrokes and clipboard content in the client's shellcode. However, it could expand its capabilities through its server application by loading additional modules. This design proves that BumbleBee is flexible, allowing its developers to focus on the development of additional modules instead of having to rebuild the malware itself. Its structure could also reduce the risk of exposing itself to analysts and their own modules for comparison.

Network communication

BumbleBee communicates over the HTTP protocol. It first creates an HTTP request that acts as a network beacon to notify the command and control (C&C) server. The POST request with the following URL, *http://<C&C server>/update/*, is the initial network beacon. The client application will send information of the compromised machine, which is encrypted by RC4 (see Figure 8 and Figure 9) once the first connection is established successfully. All other communication traffic, except for the victim information, are encrypted between server and client applications using the RC4 and compressed by LZO (Lempel–Ziv–Oberhumer) algorithm.

To make sure the received payload is correct, BumbleBee adopts a CRC32 checksum with reversed-presentation mode to verify the received data. For the CRC32 calculation, a self-defined value, "20200105" is used as the initial value (typically, the value is 0xffffffff) for checksum calculation.

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	oc	OD	0E	OF	Decoded text
00000000	32	30	35	OD	OA	2C	00	02	05	C9	86	C2	FB	74	79	40	205,ɆÂûty@
00000010	B7	9E	27	BF	F1	20	5F	58	7F	4C	9C	ЗA	49	1A	E9	5B	ž'¿ñ X.Lœ:I.é[
00000020	70	0E	69	74	81	OF	A4	EF	4C	C1	7C	0C	5D	E4	71	38	p.it. ¤ïLÁ .]äq8
00000030	05	ЗF	EE	55	C8	4B	D4	E9	32	86	92	23	BC	90	B6	16	.?îUÈKÔé2†′#4.¶.
00000040	8D	AA	зc	E8	0B	76	DGr	42	6B	5F	5F	8B	BF	DC	F5	6D	.°<è.vĐBk <¿Üõm
00000050	B5	FB	D1	4E	94	91	F5L	D6	D5	96	4A	89	1B	CA	2F	62	µûÑN″ \õÖÕ−J‰.Ê/b
00000060	F3	9B	DA	C9	BD	DO	13	F2	D7	AD	69	F6	зc	E9	A4	2C	ó>ÚÉ3⊕.ò×.iö<é¤,
00000070	10	F2	00	B 3	6D	E1	22	FB	27	В4	8C	18	CE	B3	3D	В3	.ò.³má"û''Œ.γ=³
00000080	AA	77	D8	OD	36	ЗF	54	57	25	68	9D	25	F5	11	D4	7C	*wØ.6?TW%h.%õ.Ô
00000090	7D	EE	77	ЗD	F6	93	46	79	59	28	10	63	C3	CB	DO	E2	}îw=ö`FyY(.cÃËĐâ
000000A0	DC	4C	84	39	76	D3	28	59	D9	7E	CE	1A	2A	32	1C	18	ÜL"9∀Ó(YÙ~Î.*2
00000B0	54	01	67	7A	92	A 5	57	34	AO	7D	C3	08	6D	ED	DE	EE	T.gz′¥W4 }Ã.mí⊧î
00000000	DO	EB	8B	45	49	A 9	91	42	4C	A4	FD	BB	34	C4	1B	E3	Đë <ei©`bl¤ý≫4ä.ã< td=""></ei©`bl¤ý≫4ä.ã<>
00000D0	C9	C1	25	66	C4	62	45	E8	67	B 3	33	EF	89	A1	B2	87	ÉÁ%fÄbEèg³3ï‰;*‡
000000E0	FO	AB	13	DB	09	C9	D8	3E	7A	AA	04	05	C7	01	6C	4C	ð≪.Û.ÉØ≻z≞Ç.lL
000000F0	75	97	FO	6B	47	90	60	CA	02	00	EA	32	9C	30	37	7B	u—ðkG.`Êê2œ07{
00000100	01	E2	C6	12	0A	7B	35	29	0A	36	Α4	40	05	CB	C2	FC	.âÆ{5).6¤@.ËÂü
00000110	F6	31	49	BA	E6	92	27	65	D5	22	C5	1D	2B	08	46	C2	ö1I°æ′'eÕ"Å.+.FÂ
00000120	6D	D6	4B	60	B8	DD	Α2	F7	54	BA	86	EB	6A	98	09	8F	mÖK`,Ý¢÷T°†ëj~
00000130	E1	D2	A1	D2	D4	32	B7	5C	54	AC	99	61	05	E2	E8	42	áÒ;ÒÔ2 ·∖T⊣™a.âèB
00000140	05	77	A3	23	53	AB	OF	88	4B	E1	B4	36	75	02	ED	49	.w£#S«.^Ká′6u.íI
00000150	9E	69	96	51	FF	DE	77	F9	99	3E	BB	C3	59	57	83	2D	ži-QÿÞwù™>≫ÃYWf-
00000160	B6	A 8	A 5	D5	B 9	47	8A	38	DD	9E	ED	6C	A6	D6	DA	23	¶"¥Õ¹GŠ8Ýžíl¦ÖÚ#
00000170	ЗA	51	AO	EE	C 6	B8	D8	53	C5	AE	EF	1F	AA	E2	78	C6	:Q îÆ,ØSÅ©ï.≜âxÆ
00000180	E2	3B	28	D4	61	20	CD	7B	C6	14	90	FB	6F	BA	ЗA	2E	â;(Ôa Í{Æûo°:.
00000190	C9	77	F4	C5	61	8E	50	1E	B6	6F	C3	F5	B6	AE	6E	E3	ÉwôÅaŽP.¶oÃõ¶©nã
000001A0	97	9E	79	8B	51	93	E4	CD	55	ЗB	91	Α4	C0	56	FO	F3	—žy <q"äíu; `¤àvðó<="" td=""></q"äíu;>
000001B0	E9	51	ED	14	9E	61	12	0B	4B	97	0E	5B	37	64	D6	A1	éQí.žaK—.[7dÖ;
000001C0	94	6D	A3	03	92	FF	96	96	87	1E	FB	3D	E2	7F	24	FD	"m£.'ÿ‡.û=â.\$ý
000001D0	07	62	DB	95	74	3B	79	BE	4C	B5	CB	24	0D	E1	AA		.bÛ•t;y%LµË\$.á°Ä
000001E0	FD	0E	2A	46	79	34	73	68	95	7F	21	32	5E		54	80	ý.*Fy4sh•.!2^.T€
000001F0	46	E1	01	B8	76	52	F1	A9			E3		18	D5	OD	CC	Fá.,vRñ©5>ã\$.Õ.Ì
00000200	6D	F1	AA	DF	06	DD	92	E9	OD	64	OD	0A					mñªß.Ý′é.d

Offset	Size	Description
0x00	Unfixed	Size of the packet
0x05	4	(Resevered)
0x09	4	Checksum of the Decrypted Paylaod
0x0D	Size of the packet - 8	RC4 Encrypted Payload

Figure 8. Encrypted information of the compromised machine

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	OF	Decoded text
00000000	01	34	34	30	42	58	20	44	65	73	6B	74	6F	70	20	52	.440BX Desktop R
00000010	65	66	65	72	65	6E	63	65	20	50	6C	61	74	66	6F	72	eference Platfor
00000020	6D	00	01	4E	6F	6E	65	00	00	00	00	00	01	4E	6F	6E	mNoneNon
00000030	65	00	01	47	65	6E	75	69	6E	65	49	6E	74	65	6C	00	eGenuineIntel.
00000040	01	49	6E	74	65	6C	28	52	29	20	43	6F	72	65	28	54	.Intel(R) Core(T
00000050	4D	29	20	69	37	2D	38	37	30	30	20	43	50	55	20	40	M) i7-8700 CPU @
00000060	20	33	2E	32	30	47	48	7A	00	01	30	46	38	42	46	42	3.20GHz0F8BFB
00000070	46	46	30	30	30	39	30	36	45	41	00	00	20	00	09	00	FF000906EA
00000080	40	00	02	00	00	0C	78	00	00	00	01	01	56	4D	77	61	@
00000090	72	65	2C	20	49	6E	63	2E	00	01	56	4D	77	61	72	65	re, IncVMware
000000A0	20	56	69	72	74	75	61	6C	20	50	6C	61	74	66	6F	72	Virtual Platfor
000000B0																	
00000000																	
00000080 00000090 000000A0 000000B0	40 72	00 65	02 2C	00 20	00 49	0C 6E	78 63	00 2E	00	00 01	01 56	01 4D	56 77	4D 61	77 72	61 65	0VMwa re, IncVMware

Figure 9. Decrypted information (by RC4)

Persistence

During the investigation, we found that BumbleBee adopted several techniques for persistence. It will use different techniques depending on the configuration. Here are the techniques adopted by the BumbleBee sample we found:

- Abuse registry run key to repeatedly execute the malware once system boot
- · Create Windows services to repeatedly execute malicious payloads
- Use Windows logon scripts automatically executed at logon initialization to establish persistence via adding a Registry key HKEY_CURRENT_USER\Environment "UserInitMprLogonScript"

Attribution

Due to the unique modular structure and installation procedures, we started to work on a literature review to clarify if it is an exclusive tool used by a certain threat actor. We found a similar backdoor, "BookWorm," revealed by Palo Alto in 2015. They share the following features:

- 1. Both are self-extracted files and abuse legitimate executables to load self-made malware.
- 2. Both use the same registry value as RC4 encryption key to encrypt their payload.
- 3. Both use modular architecture in the conception of the backdoor.
- 4. Both appeared in Southeast Asia, targeting local government-related organizations (similar victimology).
- 5. Both use RC4 and LZO algorithms in C&C communications (similar network protocol).

We think BumbleBee is likely to be the refactored BookWorm backdoor. They have similar tactics, techniques, and procedures (TTPs), unique encryption approach, and similar target sectors. According to the language (Simplified Chinese) shown in server application, we suspect that the origins and developers of BumbleBee may be in China and of Chinese descent.

Conclusion

Since BumbleBee and Bookworm share the same features, BumbleBee is likely a refactored form of the latter. Focusing on Asian local government targets, all signs point to a suspect linked to a Chinese hacker group.

BumbleBee, being a modular framework, is not only flexible but sophisticated as it will require analysts to investigate its structure and behavior. Another aspect of having a modular framework is that they can just keep developing additional modules since it can easily be integrated with the current version of said malware.

With its modular capabilities, the threat may deploy additional modules that may prove dangerous. Thus, an advanced layer of protection and quick detection is needed to prevent the backdoor from taking root in the system. <u>Trend Micro Vision One™</u> offers both within different entry points of a backdoor.

IOCs

Trojan.Win32.MULTICOM.ZTIC

f8809c6c56d2a0f8a08fe181614e6d9488eeb6983f044f2e6a8fa6a617ef2475 slaver.exe

Trojan.Win32.REGLOAD.ZTI

ea5db8d658f42acad38106cbc46eea5944607eb709fb00f8adb501d4779fbea0 XecureIO_v20.dll

3fc6c5df4a04d555d5cbf2ca53bed7769b5595fc6143a2599097cb6193ef8810 XecureIO_v20.dll

eeca34fba68754e05e7307de61708e4ce74441754fcc6ae762148edf9e8e2ca0	ore
6690b7ace461b60b7a72613c202d70f4684c8cdc5afbb4267c67b5fe5dbf828e	bin
4ecde81a476f1e4622d192fe2f120f7c5c3ec58bf118b791d5532f3ff61c09ee	bin
8ab8bb836b074e170c129b7f0523d256930fd1f8cf126ca1875b450fdb6c4c05	bin
515cb31b2c89df83ea6d54d5c0c3e4fe9a024319d9bd8fd76ad351860bd67ea3	ore
8e340746339614ca105a1873dad471188b24421648d080e37d52b87f4ced5e6d	bin

C&C:

- http[:]//www[.]synolo[.]ns01[.]biz:80/update
- http[:]//118[.]163[.]105[.]130:80/update

MITRE

Tactics	Techniques
Defense Evasion	T1574.002 - Hijack Execution Flow: DLL Side- Loading
T1070.004 - Indicator Removal on Host: File Deletion	
T1055 - Process Injection	_
T1480.001 - Execution Guardrails: Environmental Keying	_
Persistence	T1547.001 - Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder
T1037.001 - Boot or Logon Initialization Scripts: Logon Script (Windows)	
T1548.003 - Create or Modify System Process: Windows Service	_
Privilege Escalation	T1548.002 - Abuse Elevation Control Mechanism: Bypass User Account Control
Collection	T1056.001 - Input Capture: Keylogging
Reconnaissance	T1592 - Gather Victim Host Information

Command and Control

T1090 - Proxy

T1573.001 - Encrypted Channel: Symmetric Cryptography

T1132.001 - Data Encoding: Standard Encoding

Resource Development

T1587.001 - Develop Capabilities: Malware