

JOIN OUR FREE COMMUNITY

Get Started

SHARE ARTICLE

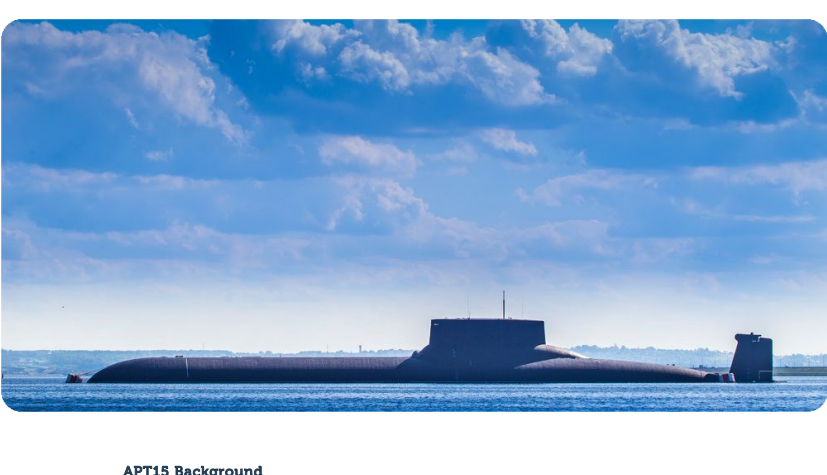






MirageFox: APT15 Resurfaces With New Tools Based On Old Ones

Written by [Jay Rosenberg](#) - 14 June 2018



APT15 Background

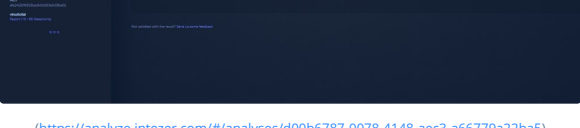
Coincidentally, following the recent hack of a US Navy contractor and theft of highly sensitive data on submarine warfare, we have found evidence of very recent activity by a group referred to as APT15, known for committing cyber espionage which is believed to be affiliated with the Chinese government. The malware involved in this recent campaign, MirageFox, looks to be an upgraded version of a tool, a RAT believed to originate in 2012, known as Mirage.

APT15 is known for committing cyberespionage against companies and organizations located in many different countries, targeting different sectors such as the oil industry, government contractors, military, and more. They are known for “living off the land,” meaning they use already available tools and software installed on the computer to operate, and once inside a target network, they will tailor their malware specifically to the target. Other names for the group are Vixen Panda, Ke3chang, Royal APT, and Playful Dragon.

There are many articles and researches online about APT15 and their activities, the most recent one by [NCC Group](#); although posted in March 2018, it refers to a campaign in 2017. In addition, although the 2017 campaign has been documented, during our research regarding MirageFox, we found a recently uploaded binary (6/8/2018) from the 2017 campaign, pretty much identical to a RAT mentioned in their RoyalAPT report, [barely detected with only 7/66 detections on VirusTotal](#).











APT15 Code Reuse

We found the new version of the RAT on VirusTotal hunting, by a YARA signature we created based off code *only found* in Mirage and Reaver, both attributed to Chinese government affiliated groups. After seeing that these binaries were new uploads to VirusTotal, with very few detections, we analyzed them using Intezer Analyze™ to see if we could find any code reuse.



(<https://analyze.intezer.com/#/analyses/d00b6787-0078-4148-aec3-a66779a22ba5>)

As can be seen in this code reuse analysis report (SHA256: 28d6a9a709b9ead84ace250889a1687c07e19f6993325ba5295410a478da30a), there is shared code with Mirage and Reaver. The compilation timestamp is from June 8, 2018 while the upload date to VirusTotal was June 9, 2018.

10 engines detected this file			
	File size: 88 KB	Last analysis: 2018-06-08 09:03:54 UTC	
10 / 66			
Detection	Details	Community	
AnginLab		TrigW32.Com[4]	Auto
Baidu		Win32.Trojan.WormQyba.16070407...	Blow
CiscoAV		Win32.Trojan.Mirage-4	CrowdStrike Falcon
Cylance		Unlabeled	Symantec
Engines		Malicious (High confidence)	Symantec
Ad-Aware		Clean	AlteLab-V3
Anti-VL		Clean	AlteLab
Avast		Clean	Avast Mobile Security
AVG		Clean	Avira

(VirusTotal)

On VirusTotal, we can see there are only 10/66 detections for this binary, 11/66 for another similar version of MirageFox (SHA256: 97813e76564aa829a359c2d12c9c6b824c532de0fc15f43765cf6b106a32b9a5), and 9/64 for the third MirageFox binary that was uploaded (SHA256: b7c1ae10f3037b7645541acb9f7421312fb1e164be964ee7acd6eb1299d6acb2).

Here's a couple examples of code reuse similarities found in the Mirage family between one of the newer binaries and older ones.

Remote Shell:

Address	Disassembly	Comment	Address	Disassembly	Comment
10012980	push esi		10012980	push esi	
10012981	push edi		10012981	push edi	
10012982	push ebx		10012982	push ebx	
10012983	push ecx		10012983	push ecx	
10012984	push edx		10012984	push edx	
10012985	push ebp		10012985	push ebp	
10012986	push i386_edi		10012986	push i386_edi	
10012987	push i386_ebx		10012987	push i386_ebx	
10012988	push i386_edi		10012988	push i386_edi	
10012989	push i386_ebx		10012989	push i386_ebx	
10012990	push i386_edi		10012990	push i386_edi	
10012991	push i386_ebx		10012991	push i386_ebx	
10012992	push i386_edi		10012992	push i386_edi	
10012993	push i386_ebx		10012993	push i386_ebx	
10012994	push i386_edi		10012994	push i386_edi	
10012995	push i386_ebx		10012995	push i386_ebx	
10012996	push i386_edi		10012996	push i386_edi	
10012997	push i386_ebx		10012997	push i386_ebx	
10012998	push i386_edi		10012998	push i386_edi	
10012999	push i386_ebx		10012999	push i386_ebx	
100129a0	push i386_edi		100129a0	push i386_edi	
100129a1	push i386_ebx		100129a1	push i386_ebx	
100129a2	push i386_edi		100129a2	push i386_edi	
100129a3	push i386_ebx		100129a3	push i386_ebx	
100129a4	push i386_edi		100129a4	push i386_edi	
100129a5	push i386_ebx		100129a5	push i386_ebx	
100129a6	push i386_edi		100129a6	push i386_edi	
100129a7	push i386_ebx		100129a7	push i386_ebx	
100129a8	push i386_edi		100129a8	push i386_edi	
100129a9	push i386_ebx		100129a9	push i386_ebx	
100129aa	push i386_edi		100129aa	push i386_edi	
100129ab	push i386_ebx		100129ab	push i386_ebx	
100129ac	push i386_edi		100129ac	push i386_edi	
100129ad	push i386_ebx		100129ad	push i386_ebx	
100129ae	push i386_edi		100129ae	push i386_edi	
100129af	push i386_ebx		100129af	push i386_ebx	
100129b0	push i386_edi		100129b0	push i386_edi	
100129b1	push i386_ebx		100129b1	push i386_ebx	
100129b2	push i386_edi		100129b2	push i386_edi	
100129b3	push i386_ebx		100129b3	push i386_ebx	
100129b4	push i386_edi		100129b4	push i386_edi	
100129b5	push i386_ebx		100129b5	push i386_ebx	
100129b6	push i386_edi		100129b6	push i386_edi	
100129b7	push i386_ebx		100129b7	push i386_ebx	
100129b8	push i386_edi		100129b8	push i386_edi	
100129b9	push i386_ebx		100129b9	push i386_ebx	
100129ba	push i386_edi		100129ba	push i386_edi	
100129bb	push i386_ebx		100129bb	push i386_ebx	
100129bc	push i386_edi		100129bc	push i386_edi	
100129bd	push i386_ebx		100129bd	push i386_ebx	
100129be	push i386_edi		100129be	push i386_edi	
100129bf	push i386_ebx		100129bf	push i386_ebx	
100129c0	push i386_edi		100129c0	push i386_edi	
100129c1	push i386_ebx		100129c1	push i386_ebx	
100129c2	push i386_edi		100129c2	push i386_edi	
100129c3	push i386_ebx		100129c3	push i386_ebx	
100129c4	push i386_edi		100129c4	push i386_edi	
100129c5	push i386_ebx		100129c5	push i386_ebx	
100129c6	push i386_edi		100129c6	push i386_edi	
100129c7	push i386_ebx		100129c7	push i386_ebx	
100129c8	push i386_edi		100129c8	push i386_edi	
100129c9	push i386_ebx		100129c9	push i386_ebx	
100129ca	push i386_edi		100129ca	push i386_edi	
100129cb	push i386_ebx		100129cb	push i386_ebx	
100129cc	push i386_edi		100129cc	push i386_edi	
100129cd	push i386_ebx		100129cd	push i386_ebx	
100129ce	push i386_edi		100129ce	push i386_edi	
100129cf	push i386_ebx		100129cf	push i386_ebx	
100129d0	push i386_edi		100129d0	push i386_edi	
100129d1	push i386_ebx		100129d1	push i386_ebx	
100129d2	push i386_edi		100129d2	push i386_edi	
100129d3	push i386_ebx		100129d3	push i386_ebx	
100129d4	push i386_edi		100129d4	push i386_edi	
100129d5	push i386_ebx		100129d5	push i386_ebx	
100129d6	push i386_edi		100129d6	push i386_edi	
100129d7	push i386_ebx		100129d7	push i386_ebx	
100129d8	push i386_edi		100129d8	push i386_edi	
100129d9	push i386_ebx		100129d9	push i386_ebx	
100129da	push i386_edi		100129da	push i386_edi	
100129db	push i386_ebx		100129db	push i386_ebx	
100129dc	push i386_edi		100129dc	push i386_edi	
100129dd	push i386_ebx		100129dd	push i386_ebx	
100129de	push i386_edi		100129de	push i386_edi	
100129df	push i386_ebx		100129df	push i386_ebx	
100129e0	push i386_edi		100129e0	push i386_edi	
100129e1	push i386_ebx		100129e1	push i386_ebx	
100129e2	push i386_edi		100129e2	push i386_edi	
100129e3	push i386_ebx		100129e3	push i386_ebx	
100129e4	push i386_edi		100129e4	push i386_edi	
100129e5	push i386_ebx		100129e5	push i386_ebx	
100129e6	push i386_edi		100129e6	push i386_edi	
100129e7	push i386_ebx		100129e7	push i386_ebx	
100129e8	push i386_edi		100129e8	push i386_edi	
100129e9	push i386_ebx		100129e9	push i386_ebx	
100129ea	push i386_edi		100129ea	push i386_edi	
100129eb	push i386_ebx		100129eb	push i386_ebx	
100129ec	push i386_edi		100129ec	push i386_edi	
100129ed	push i386_ebx		100129ed	push i386_ebx	
100129ee	push i386_edi		100129ee	push i386_edi	
100129ef	push i386_ebx		100129ef	push i386_ebx	
100129f0	push i386_edi		100129f0	push i386_edi	
100129f1	push i386_ebx		100129f1	push i386_ebx	
100129f2	push i386_edi		100129f2	push i386_edi	
100129f3	push i386_ebx		100129f3	push i386_ebx	
100129f4	push i386_edi		100129f4	push i386_edi	
100129f5	push i386_ebx		100129f5	push i386_ebx	
100129f6	push i386_edi		100129f6	push i386_edi	
100129f7	push i386_ebx		100129f7	push i386_ebx	
100129f8	push i386_edi		100129f8	push i386_edi	
100129f9	push i386_ebx		100129f9	push i386_ebx	
100129fa	push i386_edi		100129fa	push i386_edi	
100129fb	push i386_ebx		100129fb	push i386_ebx	
100129fc	push i386_edi		100129fc	push i386_edi	
100129fd	push i386_ebx		100129fd	push i386_ebx	
100129fe	push i386_edi		100129fe	push i386_edi	
100129ff	push i386_ebx		100129ff	push i386_ebx	

The function above is seen throughout many of the binaries in the Mirage family and is executed when a command is sent from the C&C. It is responsible for executing commands in cmd.exe (later down in the functions, not seen in the screenshot, it looks for cmd.exe and executes it using `CreateProcessA`).

Configuration Decryption:

```
graph TD
    sub_1000309E["sub_1000309E proc near  
xor     eax, eax  
ret     4"]
    sub_10028413["sub_10028413 proc near  
xor     eax, eax  
ret     4"]
    sub_1000309E --> sub_10028413
```

The diagram illustrates two assembly code blocks. The first block, labeled `sub_1000309E`, contains the following instructions: `push esi`, `push edi`, `push ebx`, `push ecx`, `push edx`, `push ebp`, `push i386_edi`, `push i386_ebx`, `inc_100030A0`, `add eax, [0013268] eax`, `inc eax`, `cmp eax, 130`, `jb short 100030A0`, `ret 4`, and `sub_1000309E endp`. The second block, labeled `sub_10028413`, contains the following instructions: `inc_10028415`, `inc eax`, `add [eax], al`, `inc eax`, `inc ecx`, `jb short 10028415`, `ret 4`, and `sub_10028413 endp`. A yellow arrow points from the `inc_100030A0` instruction in the first block to the `inc_10028415` instruction in the second block, indicating a jump.

Another small, but same important function in the photo above, is the function for decrypting the data containing the C&C configuration. Similar to Reaver as posted by [Palo Alto](#), it gets the IP or domain of the C&C server, the port, name of the binary, a sleep timer,