Cloak Ransomware Variant Exhibits Advanced Persistence, Evasion and VHD Extraction Capabilities

By Halcyon RISE Team

Published: 2024-12-12 · Archived: 2026-04-29 02:05:18 UTC

The Cloak ransomware group, which surfaced in late 2022, has rapidly become a significant threat actor in the cybersecurity landscape with more than two-dozen attacks against victims like Autohaus Ruland Viersen and Dunlop Aircraft Tyres. Despite its recent prominence, the origins and organizational structure of the group remain obscure.

Cloak primarily targets small to medium-sized businesses in Europe, with Germany as a key focus. The group has extended its operations to countries in Asia and targets various sectors, including healthcare, real estate, construction, IT, food, and manufacturing.

Cloak's attack strategy involves acquiring network access through Initial Access Brokers (IABs) or social engineering methods such as phishing, malvertising, exploit kits, and drive-by downloads disguised as legitimate updates like Microsoft Windows installers. Once inside a network, the group deploys a ransomware payload—a variant of ARCrypter that appears to be derived from the leaked Babuk ransomware source code.

The group delivers ransom notes as desktop wallpapers and text files named "readme_for_unlock.txt" while deleting volume shadow copies to hinder recovery efforts. Victims who refuse to pay face further consequences, as Cloak publishes their stolen data on its Data Leak Site (DLS) for free download. The group boasts an exceptionally high payment rate of 91-96%, highlighting its effectiveness in coercing victims.

Connections between Cloak and the Good Day ransomware operation have also been observed. Good Day, a variant of the ARCrypter family first seen in May 2023, shares a data leak platform with Cloak, suggesting collaboration or overlap in their extortion activities. Cloak's association with Good Day and its sophisticated techniques emphasize its growing influence and adaptability in the ransomware ecosystem.

The group is suspected of purchasing information from initial access brokers (IABs) to infiltrate their victims' networks, but also use social engineering tactics such as phishing, malvertising, exploit kits, and drive-by downloads disguised as Microsoft Windows Update installers.

Executive Summary

The Cloak variant analyzed displays sophisticated extraction and privilege escalation mechanisms and terminates processes related to security and data backup tools:

- **Delivery and Execution**: Delivered via a loader embedding the ransomware payload, the malware employs sophisticated extraction and privilege escalation mechanisms. It terminates processes and services

related to security, backups, and databases while modifying system settings to hinder recovery and user actions.

- **Payload Behavior**: The ransomware encrypts files on local drives and network shares using the HC-128 algorithm. The encryption keys are securely generated with Curve25519 and SHA512. It employs advanced evasion techniques, including executing from virtual hard disks to avoid detection.
- **Persistence and System Impact**: The ransomware ensures persistence by modifying registry entries for startup execution and restricting user actions such as logging off or accessing the Task Manager. It disrupts system utilities, network services, and essential applications to escalate operational downtime.
- **Extortion and Encryption Techniques**: Ransom notes are deployed as desktop wallpapers and text files. The ransomware uses intermittent encryption for large files, targeting specific chunks to maximize damage while optimizing performance. Shadow copies and backups are deleted to increase leverage over victims.

Ransomware Payload Behavior Analysis

The loader contains three (3) resources, each compressed with the LZMS compression algorithm from the Compression API loaded from Cabinet.dll and encrypted with a variant of Extended Tiny Encryption Algorithm (XTEA).

After extracting the first resource and saving it, the payload creates a disk partition script file which initially contains the following commands:

```
 select vdisk file="C:\ProgramData\Q9acabd3.vhd"
attach vdisk
exit
```

This script is then loaded and executed with the diskpart command line utility using the following command, which is done several times in its attempt to mount the virtual hard disk:

```
 diskpart /s C:\ProgramData\kD2aE.tmp
```

The following script is then executed, where the result is parsed by the loader to retrieve the volume ID for the volume named ***BLA:***

```
 list volumes
exit
```

If the loader is executed with elevated privileges, the diskpart command-line utility will successfully list all volumes, including the attached virtual disk. Otherwise, the loader will use the *%APPDATA%* directory as the location for its payload.

After retrieving the volume ID, it is selected using the following diskpart script:

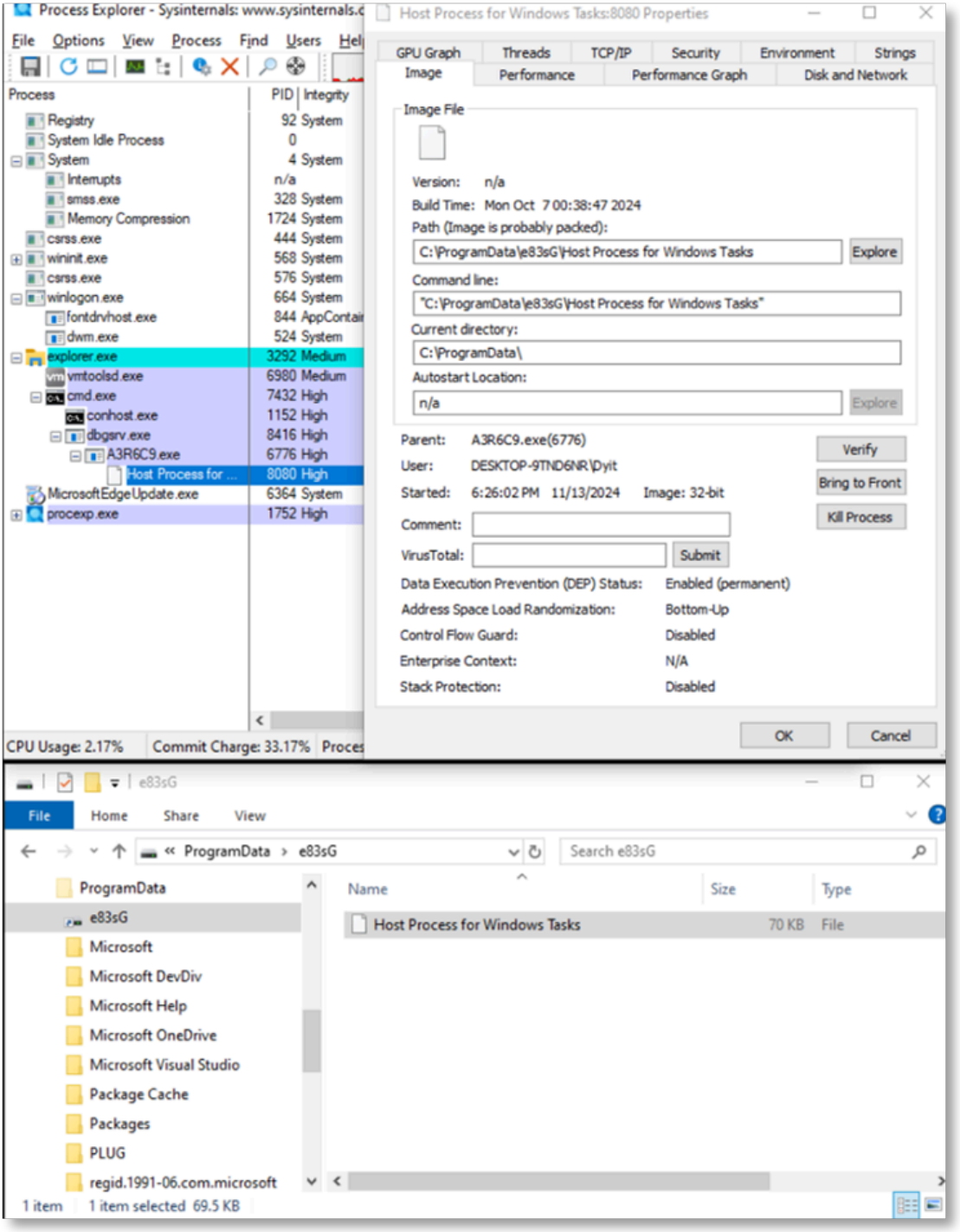
```
 select volume <virtual disk volume ID>
exit
```

Afterward, the loader creates a folder at *%APPDATA%\e83sG*, where the virtual disk is assigned and mounted using the following diskpart script:

```
 assign mount="C:\ProgramData\e83sG"
exit
```

At this stage, the loader extracts and saves the UPX-compressed ransomware payload into the mounted virtual disk as *"%APPDATA%\e83sG\Host Process for Windows Services"* if the virtual disk was successfully mounted; otherwise, it is saved to *"%APPDATA%\Host Process for Windows Services,"* and then executed from that path.

Here's what the mounted virtual disk looks like from Windows Explorer in the screenshot below:



Executing the ransomware payload from a mounted virtual disk facilitates evasion from antivirus and security software, as the virtual disk can be quickly detached after the malicious tasks are completed.

It subsequently extracts the final resource into *"%APPDATA%\sichost.exe"*, which in turn places a copy of the loader into *"%APPDATA%\A3R6C9.exe"* and removes the loader from its original execution path.

Execution

Upon execution of the ransomware payload, if first enables the *SeDebugPrivilege* for its process, essentially escalating its privilege. It then respawns itself. If it finds out that its process is running under a debugger it

immediately terminates the debugger and a few application processes, and stops some services listed in the Process and Service Termination section.

Once it verifies that it is not running a debugger, it modifies the Windows registry keys under the *HKEY_LOCAL_MACHINE and HKEY_CURRENT_USER* hives to potentially restrict user actions such as:

- logging off
- shutting down
- switching users
- accessing the Task Manager

It then sends a *WM_SETTINGCHANGE* message to update the system with these changes. The ransomware then checks for the following command-line arguments:

Command Line Parameter	**Description**
`--target <target file or directory paths delimited with space>`	List of target file or directory paths to encrypt
`--debug=<log file path>`	Log errors to specified log file path

If the number of command line arguments is greater than one (1) and no target path is specified, the ransomware clears all configurations and exits.

Process and Service Termination

After respawning to free itself from being debugged, it terminates the process associated with debuggers, reverse engineering, and performance profiling applications as listed below:

Debuggers and Reverse Engineering Processes	**Code and Performance Profiling Processes**
SND	uVision
S-Ice	CodeTalker
ImmunityDebugger	valgrind
OLLYDBG	cppcheck
devenv	clang-cl
idaq	PVS-Studio
devenv	Parasoft
windbg	Understand
gdb	Deleaker
lldb	CodeBot
SoftICE	appverif

Debuggers and Reverse Engineering Processes	Code and Performance Profiling Processes
Immunity	amplxe-gui
Hopper	nsight
radare2	
ida64	
ghidra	
ntsd	
x64dbg	
x32dbg	
windbg	
cdb	
syserx32	
pdb2sdsx32	
unpackx32	
w32dsm89	
w32dsm88	
w32dsm87	

This ransomware stops services associated with AV and security, backup and restore and database services:

AV and Security Services	Backup and Restore Services	Database Services
sophos	veeam	sql
kavfsslp	backup	MsDtsServer
KAVFSGT	YooBackup	MsDtsServer100
KAVFS	YooIT	MsDtsServer110
mfefire	VeeamTransportSvc	MSOLAP$SQL_2008
DefWatch	VeeamDeploymentService	MSOLAP$SYSTEM_BGC
ccEvtMgr	VeeamNFSSvc	MSOLAP$TPS
ccSetMgr	BackupExecVSSProvider	MSOLAP$TPSAMA
SavRoam	BackupExecAgentAccelerator	MSSQL$BKUPEXEC
RTVscan	BackupExecAgentBrowser	MSSQL$ECWDB2
zhudongfangyu	BackupExecDiveciMediaService	MSSQL$PRACTICEMGT
Sophos Agent	BackupExecJobEngine	MSSQL$PRACTTICEBGC
Sophos Clean Service	BackupExecManagementService	MSSQL$PROFXENGAGEMENT
Sophos Health Service	BackupExecRPCService	MSSQL$SBSMONITORING
Sophos MCS Agent	vss	MSSQL$SHAREPOINT
Sophos MCS Client	QBFCService	MSSQL$SQL_2008
Sophos Message Router	QBIDPService	MSSQL$SYSTEM_BGC
Antivirus	Intuit.QuickBooks.FCS	MSSQL$TPS
EraserSvc11710	QBCFMonitorService	MSSQL$TPSAMA
EsgShKernel	stc_raw_agent	MSSQL$VEEAMSQL2008R2

AV and Security Services	Backup and Restore Services	Database Services
FA_Scheduler	VSNAPVSS	MSSQL$VEEAMSQL2012
macmnsvc	PDVFSService	MSSQLFDLauncher
masvc	AcrSch2Svc	MSSQLFDLauncher$TPS
MBAMService	AcronisAgent	MSSQLSERVER
MBEndpointAgent	CASAD2DWebSvc	MySQL80
McAfeeEngineService	CAARCUpdateSvc	MySQL57
McAfeeFramework	Acronis VSS Provider	OracleClientCache80
McShield	ARSM	ReportServer
McTaskManager	bedbg	ReportServer$SQL_2008
mfemms	DCAgent	ReportServer$TPS
mfevtp	EPSecurityService	ReportServer$TPSAMA
ntrtscan	EPUpdateService	SNAC
SAVAdminService	MMS	SQLAgent$BKUPEXEC
SAVService	mozyprobackup	SQLAgent$ECWDB2
SepMasterService	SDRSVC	SQLAgent$PRACTTICEBGC
ShMonitor	VeeamBackupSvc	SQLAgent$PRACTTICEMGT
Smcinst	VeeamBrokerSvc	SQLAgent$SHAREPOINT
SmcService	VeeamCatalogSvc	SQLAgent$SQL_2008
SntpService	VeeamCloudSvc	SQLAgent$SYSTEM_BGC
sophossps	VeeamDeploySvc	SQLAgent$TPS
svcGenericHost	VeeamMountSvc	SQLAgent$TPSAMA
swi_filter	VeeamRESTSvc	SQLAgent$VEEAMSQL2012
swi_service	wbengine	SQLBrowser
swi_update_64	VeeamHvIntegrationSvc	SQLSafeOLRService
TmCCSF	Zoolz 2 Service	SQLSERVERAGENT
tmlisten		SQLTELEMETRY
WRSVC		SQLTELEMETRY$ECWDB2
swi_update		SQLWriter
EhttpSrv		SQLAgent$CXDB
ekrn		SQL Backups
ESHASRV		MSSQL$PROD
AVP		MSSQLServerADHelper
klnagent		SQLAgent$PROD
		msftesql$PROD
		MSSQL$SOPHOS
		SQLAgent$SOPHOS
		MSSQL$SQLEXPRESS
		SQLAgent$SQLEXPRESS

System and Utility Services	Network and Mail Services	Password Management Services
svc$ memtas GxVss GxBlr GxFWD GxCVD GxCIMgr sacsvr SamSs UI0Detect NetMsmqActivator	mepocs IISAdmin IMAP4Svc MSExchangeES MSExchangeIS MSExchangeMGMT MSExchangeMTA MSExchangeSA MSExchangeSRS POP3Svc RESvc SMTPSvc SstpSvc W3Svc	TrueKey TrueKeyScheduler TrueKeyServiceHelper

After closing the above services, it also attempts to terminate processes associated with database, productivity, email, web and security applications as listed below:

Database Applications	Microsoft Office Applications	Web Browser Applications/Components
sql isqlplussvc sqbcoreservice msaccess msftesql mysqld mysqld-nt mysqld-opt sqlagent sqlbrowser sqlservr steam oracle ocautoupds mydesktopqos dbsnmp xfssvccon mydesktopservice ocssd ocomm	visio winword wordpad outlook powerpnt excel onenote notepad mspub infopath Notepad	firefox firefoxconfig

Database Applications	Microsoft Office Applications	Web Browser Applications/Components
dbeng50 sqlwriter		

Email Clients	Antivirus and Security Software	System Utilities
thebat thebat64 thunderbird tbirdconfig	agntsvc tmlisten PccNTMon CNTAoSMgr Ntrtscan mbamtray	synctime encsvc

Backup and Restore Software
zoolz

File Selection/Enumeration

If the `--*target*` command-line argument is used to run the ransomware payload, it will only search for files to encrypt within the specified files and directories. Without this argument, it will search for files across all volumes, directories, and network shares. The ransomware achieves this by initializing worker threads that will:

1. **Queue Files for Encryption**

Files are selected by enumerating all files from a given directory or network resource, skipping files that match any of the folder names, file names or file extensions listed below:

Folder Names	File Names	File Extensions
Boot	readme_for_unlock.txt	crYpt
BOOTNXT	autorun.inf	crYptA1
System Volume Information	bootfont.bin	crYptA2
Windows	bootsect.bak	crYptA3
Windows.old	bootmgr	sys
Tor Browser	ntuser.dat.log	tmp
Internet Explorer	thumbs.db	efi
Google	iconcache.db	exe
Opera	ntldr	bat
Opera Software	ntuser.dat	dll
Mozilla	d3d9caps.dat	ini
Mozilla Firefox	#recycle	drv
$Recycle.Bin	..	msc
	.	

Folder Names	File Names	File Extensions
ProgramData All Users		

2. Encrypt the Queued Files

There are two (2) modes of encryption implemented by this ransomware, full and intermittent encryption, which depend on the size of the file being encrypted. Detailed information for the encryption methods is tabulated below:

File size condition	Encryption Mode	Encryption Mode Parameters
> 0x00 bytes and <= 0x500000 (5MiB)	Full	Whole file encryption
> 0x500000 (5MiB) and < 0x1400000 (20MiB)	Intermittent	Step = 0x1000000 (16MiB) Skip = 1/3 of the file Chunks = 3 Step encrypt every chunk size of 0x1000000 starting from offset 0x00, then sip every 1/3 of the file. Encrypting a total of 3 chunks.
> 0x1400000 (20MiB)	Intermittent	Step = 0x1000000 (16MiB) Skip = 0xA00000 (10MiB) Chunks = File size/0xA00000 Step encrypt every chunk size of 0x1000000 from offset 0x00, then skip 0xA00000. Encrypting a total of (file size/0xA00000) chunks.

Encryption and Decryption

When a file is queued for encryption, its file attributes is set to *FILE_ATTRIBUTE_NORMAL* first using *SetFileAttributesW*. Once ensured that the file attribute is not set to *FILE_ATTRIBUTE_READONLY and FILE_ATTRIBUTE_SYSTEM*, it is renamed by appending *.crYpt* as its file extension.

Based on our analysis, this ransomware is derived from the leaked Babuk source code, evidently seen in the decompiled code below:

```
00409774    struct mw_session_struct mw_session;
00409774    CryptGenRandom(mw_global_hprov, 0x20, &mw_session.curve25519_private);
00409798    mw_session.curve25519_private[0] &= 0xf8;
004097ba    mw_session.curve25519_private[0x1f] &= 0x7f;
004097dc    mw_session.curve25519_private[0x1f] |= 0x40;
004097f8    mw_curve25519_donna(&mw_footer, &mw_session.curve25519_private, &basepoint);
00409813    mw_curve25519_donna(&mw_session, &mw_session.curve25519_private, &mw_mpubl);
0040982b    struct mw_keys_struct mw_keys;
0040982b    mw_sha512_simple(&mw_session, 0x20, &mw_keys);
0040984a    void _mw_ctx;
0040984a    struct ECRYPT_ctx* mw_ctx = &_mw_ctx;
0040984b    mw_ecrypt_keysetup(mw_ctx, &mw_keys, 0x100, 0x100);
00409861    mw_ecrypt_ivsetup(&_mw_ctx, &mw_keys.hc256_vec);
0040987a    mw_footer.xcrc32_hash = mw_xcrc32(&mw_keys, 0x40);
```

This ransomware uses *CryptGenRandom*, a cryptographically secure pseudo random number generator, to generate a random 32-byte (0x20) Curve25519 private key. It uses Curve25519 _donna to derive a 32-byte public key from the generated private key, and uses *Curve25519_donna* again to derive a 32-byte shared key from the generated private key and a hard-coded 32-byte public key, which in this case is:

```
 00000000: 7a 15 f0 aa 58 7d 9d 6a b5 54 bb ae 0f 8c 41 8a  z...X}.j.T....A.
00000010: 73 5c ac ea e9 e6 80 8b 82 f0 87 f4 78 82 74 0f  s\..........x.t.
```

A 64-byte (512 bits) hash is then generated by getting the SHA512 hash of the Curve25519 shared key, where the first 32-bytes used as the HC-128 key and the remaining 32-bytes as the HC-128 initial vector (IV).

Depending on the encryption mode described in the File Selection/Enumeration section, data chunks from the file are encrypted using HC-128. The structure below describes a 0x48 byte footer structure appended at the end of the encrypted file:

```
    struct mw_footer_struct __packed
    {
00      BYTE curve25519_pub[0x20];
20      DWORD xcrc32_hash;
24      DWORD unused;
28      BYTE marker[0x20];
48  };
```

This ransomware variant may have adapted its cryptographic algorithms from the following sources:

- Elliptic Curve Cryptography (ECC) function Curve25519_donna - https://github.com/agl/curve25519-donna/blob/master/curve25519-donna.c
- SHA512 - https://github.com/Maximus5/plink/blob/607ca3416722096a75555009b3422de97c37e65a/sshsh512.c#L303

Extortion Notifications

During the C Run-Time initialization of the ransomware payload, the ransom note is decrypted with a modified variant of Extended Tiny Encryption Algorithm (XTEA) using a 16-byte key derived with four (4) hard-coded bytes as shown in the code snippet below:

```
00415e50    uint32_t (*)[0x4] __stdcall mw_decrypt_ransom_note(LPVOID enc_data, int32_t size)

00415e50    {
00415e50        int32_t key[0x4];
00415e50        _memset(&key, 0, 4);
00415e6a        void cipher;
00415e7a        uint32_t (* result)[0x4] = _memset(&cipher, 0, 8);
00415e7a        key[0] = 0xbc;
00415e99        key[1] = 0x24;
00415eb4        key[2] = 0xa1;
00415ece        key[3] = 0xcc;
00415ee9        int32_t j = 0;
00415eed        int32_t var_10 = 8;
00415ef4        int32_t var_c = 0;
00415efb
00415efb        for (uint32_t (* i)[0x4] = nullptr; i < size >> 3; i = result)
00415f1a        {
00415f1a            for (; j < var_10; j += 1)
00415f1a            {
00415f31                int32_t eax_3;
00415f31                eax_3 = *(uint8_t*)(enc_data + j);
00415f3c                *(uint8_t*)(&cipher + var_c) = eax_3;
00415f3e                var_c += 1;
00415f48            }
00415f31
00415f31            mw_modified_xtea_decrypt(0x20, &cipher, &key);
00415f5a            j -= 8;
00415f65            int32_t var_c_1 = 0;
00415f68
00415f68            for (; j < var_10; j += 1)
00415f80            {
00415f80                int32_t eax_4;
00415f80                eax_4 = *(uint8_t*)(&cipher + var_c_1);
00415f8b                *(uint8_t*)(enc_data + j) = eax_4;
00415f8f                var_c_1 += 1;
00415f97            }
00415f80
00415f80            _memset(&cipher, 0, 8);
00415fa4            var_10 += 8;
00415fb2            var_c = 0;
00415fb5            result = (char*)i + 1;
00415f0e        }
00415f1a
00415f1a        return result;
00415fc4    }
00415e50
```

Just before the encryption threads are initialized, a bitmap image is generated from the decrypted ransom note, saved to *C:\ProgramData\wallpaper.bmp*, and set as the desktop wallpaper using the *SystemParametersInfoW* function as shown in the code snippet and wallpaper preview below:

```
00407a30    BOOL mw_set_wallpaper(int32_t arg1)

00407a30    {
00407a30        SystemParametersInfoW(SPI_SETDESKWALLPAPER, 0, arg1, 0x3);
00407a61        HKEY var_c;
00407a61
00407a61        if (!RegOpenKeyExW(HKEY_CURRENT_USER, u"Control Panel\Desktop", 0, KEY_SET_VALUE, &var_c))
00407a61        {
00407a63            wchar16 const* const lpData = u"Center";
00407a6e            int32_t eax_2 = lstrlenW(u"Center");
00407a8a            RegSetValueExW(var_c, u"WallpaperStyle", 0, REG_SZ, lpData, eax_2 * 2 + 2);
00407a90            void* const lpData_1 = &data_4235d8;
00407a9b            int32_t eax_4 = lstrlenW(&data_4235d8);
00407ab7            RegSetValueExW(var_c, u"TileWallpaper", 0, REG_SZ, lpData_1, eax_4 * 2 + 2);
00407ac1            RegCloseKey(var_c);
00407a61        }
00407a61
00407adc        return SystemParametersInfoW(SPI_SETDESKWALLPAPER, 0, arg1, 0x3);
00407a30    }
```

Alternatively, for every directory traversed by the file enumeration thread, a ransom note in text format is saved as *readme_for_unlock.txt*.

Backup Disruptions

Before initializing the encryption threads, the ransomware empties the recycle bin by calling the *SHEmptyRecycleBinA* function. It then deletes volume shadow copies by running the following command line:

```
cmd.exe /c vssadmin.exe delete shadows /all /quiet
```

This ransomware adds the following registry to make sure it executes every time the system starts:

Registry Component	**Value**
Hive	`HKEY_LOCAL_MACHINE`

Registry Component	Value
Key	`SOFTWARE\Microsoft\Windows\CurrentVersion\Run`
Entry	Windows Update
Path	C:\ProgramData\Host Process for Windows Services or C:\ProgramData\e83sG\Host Process for Windows Services

System Modifications

To restrict the user from logging out, shutting down, switching to another user or accessing the Task Manager; the ransomware sets the following registry keys:

Registry Component	Value
Hive	`HKEY_LOCAL_MACHINE` `HKEY_CURRENT_USER`
Key	`SOFTWARE\MICROSOFT\WINDOWS\CURRENTVERSION\POLICIES\EXPLORER`

Entries	`NoLogoff`
	`NoClose`
	`StartMenuLogOff`
	`DisableChangePassword`
	`DisableSwitchUser`
	`DisableTaskMgr`
	`HideFastUserSwitching`

Conclusion

The Cloak ransomware variant analyzed demonstrates a high level of sophistication in its operational tactics, combining advanced privilege escalation, process termination, and encryption techniques. Its delivery mechanism embeds the payload seamlessly, while its use of the HC-128 algorithm and robust key generation ensures secure and effective file encryption.

By targeting security tools, backups, and databases, Cloak maximizes disruption and complicates recovery efforts. Its persistence mechanisms, including registry modifications and user restrictions, further ensure prolonged impact and operational downtime. With its strategic use of intermittent encryption and aggressive deletion of recovery tools, Cloak exemplifies a modern ransomware threat designed to exert maximum pressure on victims while evading detection and countermeasures.

Halcyon.ai eliminates the business impact of ransomware. Modern enterprises rely on Halcyon to prevent ransomware attacks, eradicating cybercriminals' ability to encrypt systems, steal data, and extort companies – talk to a Halcyon expert today to find out more and check out the Halcyon Attacks Lookout resource site. Halcyon also publishes a quarterly RaaS and extortion group reference guide, Power Rankings: Ransomware Malicious Quartile.
