Nightspire

Technical Analysis — Windows

REVERSE-ENGINEERED REPORT

RansomLook · ransomlook.io
File last modified: 2026-05-23

Sample SHA-256: `69f5515ff3f554233840ad2f2397b345f955013017a9ae14ed4e762f52d936af`

NightSpire Ransomware — Full Analysis

1. Sample Identification

Field	Value
Family	NightSpire (self-identified as "NightSpire.Team")
SHA-256	`69f5515ff3f554233840ad2f2397b345f955013017a9ae14ed4e762f52d936af`
MD5	`20cb8d8216061545b0b31ec8bd5f42de`
Type	PE32+ x86-64, Windows Console
Size	3,295,744 bytes (3.2 MB)
Language	Go (Golang) - compiled with Go runtime
Compile timestamp	Not available (Go binary with stripped debug info)
PDB path	None (Go binaries don't use PDB)
Image base	`0x400000`
Sections	8: `.text`, `.rdata`, `.data`, `.idata` (×2), `.pdata`, `.xdata`, `.idata`
Functions	Approximately 3,200 functions identified after Go runtime analysis

NightSpire is a Go-based ransomware targeting Windows systems with modern cryptographic implementation (AES-256-CTR + RSA-4096-OAEP-SHA512) and multi-threaded execution. The binary shows no dedicated anti-analysis techniques, string obfuscation, or evasion mechanisms, focusing purely on encryption functionality. It embeds a hardcoded 4096-bit RSA public key for key exchange and implements a robust CLI interface with extensive configuration options. The malware enumerates all Windows drives (A: through Z:) using parallel goroutines and includes functionality for file icon manipulation and Windows Explorer integration.

Imports (2 DLLs)

DLL	Count	Purpose
KERNEL32	47	Core Windows APIs: File I/O (`WriteFile`, `CreateFileW`, `FindFirstFileW`), Process/Thread management (`CreateThread`, `WaitForSingleObject`), Memory management (`VirtualAlloc`, `VirtualFree`), System info (`GetSystemDirectoryA`, `GetSystemInfo`)
Additional APIs	8	Advanced APIs: Exception handling (`AddVectoredExceptionHandler`), Console (`WriteConsoleW`, `GetConsoleMode`), Library loading (`LoadLibraryW`, `GetProcAddress`)

Notably absent: - No networking imports (`WinHttp`, `WinINet`, `Ws2_32`) → **no built-in networking capability identified in this payload** - No service control (`OpenSCManager`, `EnumServicesStatus`) → **no service termination functionality** - No process enumeration (`CreateToolhelp32Snapshot`, `TerminateProcess`) → **no process killing functionality** - No debugging detection (`IsDebuggerPresent`, `CheckRemoteDebuggerPresent`) → **no dedicated anti-debugging logic** - No WMI/COM imports → **no WMI-based operations**

2. Infrastructure

Field	Value
Primary Onion	`http://nspire7lugml7ybqyjaaxtsgrs4qn3fcon3lrjbih6wamttvdm5ke4qd.onion`
Leak Site	`http://nspirep7orjq73k2x2fwh2mxgh74vm2now6cdbnnxjk2f5wn34bmdxad.onion`
Email 1	`nightspire.team2026@onionmail.org`
Email 2	`nightspireteam.receiver@onionmail.org`
qTox ID 1	`038F61A270B8094E713E4815C4FA5086E4AD3A021575C6F90EE65A0C123D3E3BF6926C3B59EA`
qTox ID 2	`8D663FD10BF662930F4C076CBF95FACFCC4ABD8F1A5E328DE75D0B0237A74E1AE1E0C5C37E7F`
Chat UUID	`NSPIRE830NPH7ZLBRW39`
Chat Password	`769FZisalII12Rph`
Note filename	Unknown (not hardcoded, likely configurable)
Extension	`.nspire`
File Marker	`sNightspire` (appended to encrypted files)
Payment	Bitcoin (BTC)

3. Ransom Note

Content (from embedded string @ `0x5afeca`)

```
~~~ You have been attacked by NightSpire.Team ~~~

All data was encrypted.(There might be some files which extensions are remaining as original, but
they are all encrypted as well.)

We have taken your sensitive data.
If you want check the list of files we have stolen, you can find it on our WEBSITE CHAT (http://
nspire7lugml7ybqyjaaxtsgrs4qn3fcon3lrjbih6wamttvdm5ke4qd.onion).

*********************************************************************
We want Only $500K in BTC.
+ If you contact us within 24 hours, you will receive a 30% discount on the price.
+ If you pay us within 48 hours, you will receive a 40% discount on the price.
+ And also if you pay us within 24 hours, you will receive a 50% discount on the price.

* We can adjust the LARGER DISCOUNT if you engage in ACTIVE NEGOTIATION.
* After 24 hours, we will public your company's real name and conduct a full-scale investigation
into your data.
* In 72 hours, we will begin the leak, starting with the oldest data.
*********************************************************************

***If you pay the ransom, we will fulfill all the terms we agreed upon during the negotiation
process.
Provides DECRYTPION TOOL & KEY and permanently DELETE the stolen data.
And also  we'll help reinforce your internal network with strong security measures to make sure
you never face another ransomware attack.

You can recover your system and files within only 5 hours with our help***

>>>>>What happens if you don't pay?

1. We will provide your all data to CLIENTS, GOVERNMENT and LEGAL authorities.
2. We will transfer information to your competitors.
3. We will make all the information public on our website.(http://
nspirep7orjq73k2x2fwh2mxgh74vm2now6cdbnnxjk2f5wn34bmdxad.onion/)
4. We will do our utmost to ensure your Company's faces significant fines and reputational
damage.


>>>>> Warning! Do not delete or modify encrypted files, it will lead to irreversible problems
with decryption of files!

*********************************************************************
>>>>>Contact us.


>>> Using Our WEBSITE CHAT(You can only connect our site with Tor Browser)

+ Download and install Tor Browser at https://www.torproject.org/
+ Through Tor Browser, enter below url
http://nspire7lugml7ybqyjaaxtsgrs4qn3fcon3lrjbih6wamttvdm5ke4qd.onion

+ Enter UUID "NSPIRE830NPH7ZLBRW39" and PASSWORD "769FZisalII12Rph" on Login Page

>>> Using qTox Chat App

qTox ID1: 038F61A270B8094E713E4815C4FA5086E4AD3A021575C6F90EE65A0C123D3E3BF6926C3B59EA
```

```
qTox ID2: 8D663FD10BF662930F4C076CBF95FACFCC4ABD8F1A5E328DE75D0B0237A74E1AE1E0C5C37E7F

>>>Using e-Mail:

nightspire.team2026@onionmail.org
nightspireteam.receiver@onionmail.org
****************************************************************

DO NOT SHARE UUID TO ANYONE!!!
```

---

4. Execution Flow (`main.main` @ `0x511d40`)

```
1.  Decode embedded RSA public key from base64 (sLs0tls1crudjti variable)
2.  Parse command-line arguments using Go flag package
3.  Initialize configuration flags (-p, -e, -s, -r, -t, -k, -noreadme)
4.  If icon flag set: configure .nspire file icons + refresh Windows Explorer
5.  If specific path provided (-p): encrypt target directory
6.  Else: enumerate all drives A: through Z: in parallel goroutines
7.  For each drive: spawn Go goroutine calling main.EncryptDir
8.  Use sync.WaitGroup to coordinate parallel execution
9.  Each goroutine recursively encrypts files in its assigned drive
```

Multi-threaded Architecture

NightSpire employs Go's goroutine-based concurrency model: - **Drive-level parallelism**: One goroutine per Windows drive letter (A: through Z:) - **Thread control**: `-k` flag controls goroutine count (default: 1, warning about file corruption) - **Synchronization**: Uses Go's `sync.WaitGroup` for coordination - **Memory safety**: Go's garbage collector prevents typical memory corruption issues

---

5. Encryption System

Key Exchange

Parameter	Value
Algorithm	RSA-OAEP-SHA512
Key size	4096 bits
Implementation	Go crypto/rsa standard library
Attacker public key	Base64-embedded in binary @ `0x5afa9e`
Public exponent	65537 (standard)
OAEP hash	SHA-512

Symmetric Cipher

Parameter	Value
Algorithm	AES-256-CTR
Mode	Counter (CTR)
Key size	256 bits (32 bytes)
Nonce/IV	128 bits (16 bytes)
Per-file key	YES — generated via `main.GenerateRandomKey` @ `0x50f480`
Per-file nonce	YES — generated via `main.GenerateRandomNonce` @ `0x50f500`

File Encryption Process (`main.EncryptFile` @ `0x510460`)

1. **Key Generation**: Generate 32-byte AES key + 16-byte nonce using crypto/rand
2. **File Check**: Call `main.checkPossibility` to verify file is not already encrypted
3. **RSA Wrapping**: Encrypt AES key using RSA-4096-OAEP-SHA512 (`main.EncryptRSA` @ `0x50f7a0`)
4. **AES Encryption**: Encrypt file content using AES-256-CTR (`main.encryptMethod` @ `0x50f620`)
5. **Signature**: Append "sNightspire" marker to file end (`main.writeToTail` @ `0x50ffa0`)
6. **Extension**: Change file extension to .nspire (`main.changeFileName` @ `0x510260`)
7. **Icon**: If enabled, set custom icon for .nspire files

Encrypted File Format

File Structure Analysis: The exact encrypted file metadata structure could not be fully reconstructed during static analysis. Based on the observed RSA and AES operations:

- **Encrypted Data**: File content encrypted with AES-256-CTR
- **Key Material**: RSA-4096-OAEP encrypted AES key (512 bytes expected)
- **Signature Marker**: "sNightspire" string appended to file end
- **Extension**: Changed to .nspire

Note: The precise layout of RSA ciphertext, nonce storage, and metadata positioning within the encrypted file footer requires dynamic analysis or additional reverse engineering.

Go Build Information

Compiler Version: Analysis indicates Go 1.24.11 (identified via runtime strings) **Build Target**: Windows x86-64 **CGO**: Disabled (pure Go implementation) **Build Optimizations**: Standard Go compiler optimizations applied

Crypto Implementation Details

AES-256-CTR Implementation (`main.encryptMethod`):

```
// Pseudocode based on decompiled function
func encryptMethod(data []byte, keyNonce []byte) []byte {
    key := keyNonce[:32]    // First 32 bytes = AES key
    nonce := keyNonce[32:]  // Next 16 bytes = CTR nonce

    cipher, err := aes.NewCipher(key)
    if err != nil { return nil }

    stream := cipher.NewCTR(cipher, nonce)

    encrypted := make([]byte, len(data))
    stream.XORKeyStream(encrypted, data)

    return encrypted
}
```

RSA Key Wrapping (`main.EncryptRSA`):

```
// Pseudocode based on decompiled function
func EncryptRSA(data []byte, publicKey *rsa.PublicKey) []byte {
    hash := sha512.New()

    encrypted, err := rsa.EncryptOAEP(hash, rand.Reader, publicKey, data, nil)
    if err != nil { return nil }

    return encrypted
}
```

---

6. File Targeting

Targeted Files

NightSpire encrypts all files by default except those matching exclusion criteria.

Excluded Directories

Directory	Reason
`System Volume Information`	Windows system integrity
`Program Files (x86)`	Application binaries
AppData (optional)	User application data (controllable via `-s` flag)

File Exclusion Logic (`main.checkPossibility` @ `0x50f8c0`)

NightSpire implements file exclusion logic:

1. **Size check**: Skip files < 6 bytes
2. **Extension patterns**: Skip files ending with specific patterns: - Pattern `1768977262` + `25970` (hex: `0x696E7370` + `0x7273`) → likely ".nsprs" or similar - 3-byte patterns: `31091` + `'s'`, `27748` + `'l'`, `30821` + `'e'` → various extensions
3. **Signature check**: Read last bytes of file to detect "sNightspire" marker
4. **Already encrypted**: Skip files already containing the ransomware signature

---

7. Recovery Inhibition

NightSpire does not implement traditional recovery inhibition techniques:

- **No VSS deletion** (no `vssadmin delete shadows` commands)
- **No bcdedit modifications** (no boot recovery disabling)
- **No wbadmin operations** (no backup deletion)
- **No safe mode blocking**

The ransomware relies purely on encryption with strong cryptography rather than system sabotage.

---

8. Targeted Services (0)

NightSpire does not terminate services. No service control manager operations detected.

---

9. Targeted Processes (0)

NightSpire does not kill processes. No process enumeration or termination logic detected.

---

10. Persistence & Evasion

Icon Integration (`main.setFileIcon` @ `0x50eae0`)

Function: Custom icon assignment for .nspire files **Purpose:** Windows Explorer integration to display custom icons for encrypted files

Explorer Refresh (`main.refreshWindowsExplorer` @ `0x50ef00`)

Function: Forces Windows Explorer to refresh and recognize new file type **Implementation:** Uses Windows Shell APIs to update file associations

File Marker System (`main.checkPossibility`)

Function: Prevents re-encryption by checking for "sNightspire" signature **Implementation:** Seeks to end of file and reads signature before encryption

Anti-Analysis Summary

Technique	Unprotect ID	Address	Description
None detected	—	—	No dedicated anti-analysis or anti-debugging logic was identified

NightSpire shows minimal obfuscation: - No string obfuscation - No API hashing - No dedicated anti-debugging logic - No VM detection
- No sandbox evasion techniques - Plain-text configuration strings - Standard Go runtime calls

---

11. Command-Line Arguments

Argument	Description
`-p <path>`	Specify target file/directory path
`-e <mode>`	Encryption method: 0 = with extension change, 1 = without extension
`-s <flag>`	Skip AppData folders: 0 = encrypt AppData, 1 = skip AppData
`-r <mode>`	Readme method: 0 = create everywhere, 1 = root folder only
`-t <seconds>`	Sleep time between file operations (throttling)
`-k <count>`	Thread/goroutine count (default: 1, **corruption warning**)
`-noreadme`	Disable ransom note creation entirely

Advanced Configuration: - **Icon mode**: Embedded flag to enable .nspire icon configuration - **Target validation**: All paths validated before encryption begins - **Error handling**: Go's error handling prevents crashes on inaccessible files

---

12. Static Imports Summary

Category	Key APIs
Crypto	Implemented in Go standard library (crypto/aes, crypto/rsa, crypto/rand)
File I/O	`WriteFile`, `CreateFileW`, `FindFirstFileW`, `FindNextFileW`, `ReadFile`, `SetEndOfFile`
Process	`CreateThread`, `WaitForSingleObject`, `VirtualAlloc`, `VirtualFree`, `GetCurrentThreadId`
System	`GetSystemDirectoryA`, `GetSystemInfo`, `LoadLibraryW`, `GetProcAddress`, `CloseHandle`
Volume	`FindFirstVolumeW`, `FindNextVolumeW`, `GetVolumeInformationW`, `GetVolumePathNameW`
Console	`WriteConsoleW`, `GetConsoleMode`, `GetStdHandle`
Registry	None detected
Network	None detected

---

13. IDA Analysis — Key Functions

Address	Name	Size	Description
`0x511d40`	`main.main`	1541 B	Main entry point, CLI parsing, drive enumeration orchestration
`0x511260`	`main.EncryptDir`	2395 B	Recursive directory encryption with goroutine management
`0x510aa0`	`main.startEncrypting`	1963 B	Encryption orchestrator and thread coordinator
`0x510460`	`main.EncryptFile`	921 B	Individual file encryption with error handling
`0x50f8c0`	`main.checkPossibility`	602 B	File exclusion logic and re-encryption prevention
`0x50f620`	`main.encryptMethod`	377 B	AES-256-CTR implementation wrapper
`0x50f7a0`	`main.EncryptRSA`	273 B	RSA-OAEP key wrapping implementation
`0x50ffa0`	`main.writeToTail`	596 B	Signature appending ("sNightspire" marker)
`0x510860`	`main.MakeReadMeFile`	471 B	Ransom note creation and placement
`0x510260`	`main.changeFileName`	510 B	File extension change to .nspire
`0x50f480`	`main.GenerateRandomKey`	123 B	AES key generation (32 bytes via crypto/rand)
`0x50f500`	`main.GenerateRandomNonce`	123 B	CTR nonce generation (16 bytes via crypto/rand)
`0x50eae0`	`main.setFileIcon`	854 B	Windows file icon configuration for .nspire
`0x50ef00`	`main.refreshWindowsExplorer`	734 B	Explorer refresh for file association changes

Runtime Functions (Go-specific): - `runtime.main` @ `0x43cf20` : Go runtime initialization - Multiple goroutine management functions for concurrent execution - Garbage collector integration for memory management

---

14. Indicators of Compromise (IOCs)

Hashes

Type	Value
SHA-256	`69f5515ff3f554233840ad2f2397b345f955013017a9ae14ed4e762f52d936af`
MD5	`20cb8d8216061545b0b31ec8bd5f42de`

Network

Type	Value
Onion (Chat)	`nspire7lugml7ybqyjaaxtsgrs4qn3fcon3lrjbih6wamttvdm5ke4qd.onion`
Onion (Leak)	`nspirep7orjq73k2x2fwh2mxgh74vm2now6cdbnnxjk2f5wn34bmdxad.onion`
Email	`nightspire.team2026@onionmail.org`
Email	`nightspireteam.receiver@onionmail.org`

Files

Indicator	Value
Encrypted extension	`.nspire`
File signature	`sNightspire` (at end of encrypted files)
Chat credentials	UUID: `NSPIRE830NPH7ZLBRW39`, Password: `769FZisalII12Rph`

Registry

Key	Description
File associations	`.nspire` file type registration with custom icons

Behavioral

- Multi-threaded drive enumeration (A: through Z:)
- Windows Explorer refresh operations
- Volume API usage for drive discovery
- Go goroutine-based parallel processing
- File marker checking to prevent re-encryption

Distinctive Strings

- `"NightSpire.Team"`
- `"sNightspire"`
- `"~~~ You have been attacked by NightSpire.Team ~~~"`
- `".nspire"`
- `"NSPIRE830NPH7ZLBRW39"`
- `"769FZisalII12Rph"`
- `"noreadme"`
- `"_rt0_amd64_windows"` (Go runtime entry)

---

15. MITRE ATT&CK Mapping

ID	Technique	Implementation
T1486	Data Encrypted for Impact	AES-256-CTR + RSA-4096-OAEP-SHA512 hybrid encryption
T1083	File and Directory Discovery	Recursive directory enumeration across all Windows drives

Notable Absences: - **T1055** (Process Injection): Not implemented - **T1112** (Modify Registry): Minimal registry interaction - **T1082** (System Information Discovery): No system profiling - **T1614** (System Location Discovery): No geolocation checks - **T1490** (Inhibit System Recovery): No VSS deletion or backup removal - **T1562** (Impair Defenses): No security tool disabling

---

16. Summary

NightSpire represents a well-structured Go-based ransomware with modern cryptographic implementation and multi-threading architecture. The malware demands $500,000 in Bitcoin with time-based discount incentives and operates dual Tor infrastructure for negotiation and data leak hosting.

Key technical characteristics: - **Modern cryptography**: AES-256-CTR + RSA-4096-OAEP-SHA512 using standard Go crypto libraries - **Go implementation**: Modern language choice offers memory safety and concurrent execution - **Multi-threaded execution**: Parallel drive encryption via goroutines for performance - **Minimal evasion**: Straightforward implementation with no dedicated anti-analysis features - **Infrastructure setup**: Dedicated .onion sites, qTox integration, and structured negotiation process

Defensive considerations: - No cryptographic weaknesses were identified during static analysis - Minimal obfuscation makes behavioral detection and analysis straightforward
- File signature system enables identification of encrypted files - CLI configuration suggests possible automation or affiliate usage - Prevention is critical as no recovery weaknesses were identified in the analyzed payload

The ransomware's straightforward implementation and lack of obfuscation, combined with robust crypto and established infrastructure, suggests an operator prioritizing operational effectiveness over evasion complexity.

Analysis Date: 2026-05-15 Analyzed with: IDA Pro 9.3 + Custom MCP Tooling