Nevada Ransomware, Nokoyawa Variant | ThreatLabz

By Brett Stone-Gross

Published: 2023-03-06 · Archived: 2026-04-05 13:46:31 UTC

Technical Analysis

ThreatLabz has identified at least four distinct versions of Nokoyawa ransomware. For clarity, we will use the version numbers 1.0, 1.1, 2.0 and 2.1 (Nevada) based on code similarities. Table 1 illustrates the similarities and differences between all four versions of Nokoyawa ransomware including Nevada.

Attribute	**Nokoyawa 1.0**	**Nokoyawa 1.1**	**Nokoyawa 2.0**	**Nokoyawa 2.1 (Nevada)**
Encryption algorithms	SECT233R1 + Salsa20	SECT233R1 + Salsa20	X25519 + Salsa20	X25519 + Salsa20
Encryption library	Tiny-ECDH	Tiny-ECDH	x25519_dalek	x25519_dalek
Programming language	C/C++	C/C++	Rust	Rust
Encryption Parameters	Hardcoded	Passed via command-line	Passed via command-line	Hardcoded
Import Hashing	No	Yes	No	No
CIS Exclusion	No	No	Yes	Yes
Architecture	x64	x64	x64	x64

Earliest known compilation date	February 2022	January 2023	September 2022	January 2023

Table 1. Comparison between different versions of Nokoyawa ransomware

There are a few commonalities between all Nokoyawa variants such as being compiled only for 64-bit versions of Windows and using a relatively obscure method to delete Windows Shadow Copies. The latter entails calling the function *DeviceIoControl* (shown in Figure 1) with the undocumented control code parameter *IOCTL_VOLSNAP_SET_MAX_DIFF_AREA_SIZE* (0x53C028) with a maximum size of 1, which causes Windows to delete all shadow copies as a result.

```
loc_140005F1E:                          ; CODE XREF: DeleteShadowCopies_0:DeleteShadowCopies↑j
                lea     rax, [rbp+70h+BytesReturned]
                mov     [rsp+0F0h+var_C0], rax ; lpBytesReturned
                mov     [rsp+0F0h+var_B8], 0 ; lpOverlapped
                mov     [rsp+0F0h+nOutBufferSize], 0 ; nOutBufferSize
                mov     qword ptr [rsp+0F0h+var_D0], 0 ; lpOutBuffer
                lea     r8, [rbp+70h+InBuffer] ; lpInBuffer
                mov     rcx, rdi        ; hDevice
                mov     edx, 53C028h    ; dwIoControlCode
                mov     r9d, 18h        ; nInBufferSize
                call    DeviceIoControl
                mov     esi, eax
                mov     rcx, rdi        ; hObject
                call    cs:__imp_CloseHandle
                mov     eax, esi
                add     rsp, 0E0h
                pop     rdi
                pop     rsi
                pop     rbp
                retn
```


Figure 1. Nokoyawa/Nevada code to delete Windows Shadow Copies

All versions of Nokoyawa support the command-line parameters *--file* (to encrypt a single file) and *--dir* (to encrypt a directory). However, Nokoyawa 1.1 and 2.0 require a configuration to execute the ransomware via the *--config* command-line parameter. The configuration parameter is a Base64 encoded JSON object that has the following keys and values shown in Table 2.

Key	**Description**
NOTE_NAME	Ransom note filename
NOTE_CONTENT	Ransom note content

EXTENSION	Encrypted file extension (also used as the Salsa20 nonce)
ECC_PUBLIC	Curve25519 public key
SKIP_EXTS	File extensions that will not be encrypted
SKIP_DIRS	Directories that will not be encrypted
ENCRYPT_NETWORK	Encrypt network shares
DELETE_SHADOW	Delete Windows shadow copies
LOAD_HIDDEN_DRIVES	Unhide hidden drives and encrypt files

Table 2. Nokoyawa 1.1 and Nokoyawa 2.0 ransomware configuration parameters

Nokoyawa 1.1 also has a *--safe-mode* command-line option to reboot the system into Windows safe mode prior to file encryption to maximize the number of files that can be encrypted by loading the minimal set of applications, and therefore, minimize the number of open file handles that may interfere with encryption. In addition, Nokoyawa 1.1 is the only variant that obfuscates the Windows API functions that are called during runtime by resolving each name via CRC32 hash.

In Nevada ransomware, the encryption parameters are hardcoded in the binary, but the other command-line options are virtually identical to Nokoyawa 1.1 and 2.0 (with the exception of a new feature to self-delete the ransomware binary after file encryption is complete). Nevada also supports a *-help* command-line argument, which prints the usage shown below in Figure 2.

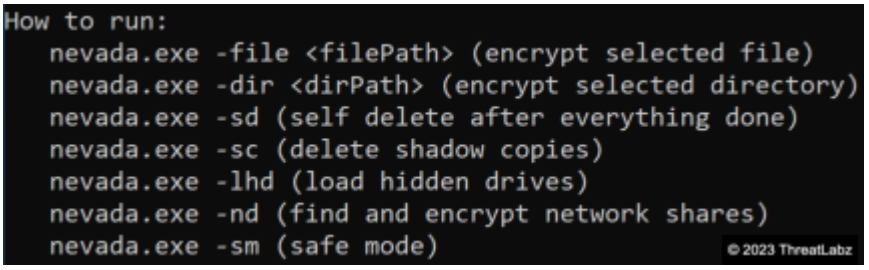

Figure 2. Nevada ransomware command-line help

In order to reduce the risk of law enforcement actions, Both Nokoyawa 2.0 and Nevada check whether the infected system is located in a former Commonwealth of Independent States (CIS) country. The former calls the

Windows API *GetSystemDefaultLCID* for language IDs (between 1049-1092 or 2073) and the latter calls *GetUserDefaultUILanguage* (between 1049-1090) to determine the system's locale and language, respectively. Some of these language IDs include countries outside of the CIS countries, which may be to simplify the code by adding a range of values rather than individually checking each value.

Nokoyawa 1.0 and Nokoyawa 1.1 share about 39% of the same code, while Nokoyawa 2.0 and Nevada share more than 87% of the same code according to BinDiff.

Debug Print Statements

Another similarity between Nokoyawa 2.0 and Nevada are debug print statements, which are very similar or identical. Figure 3 shows an example for a function that creates a thread and prints a debug statement to the console.

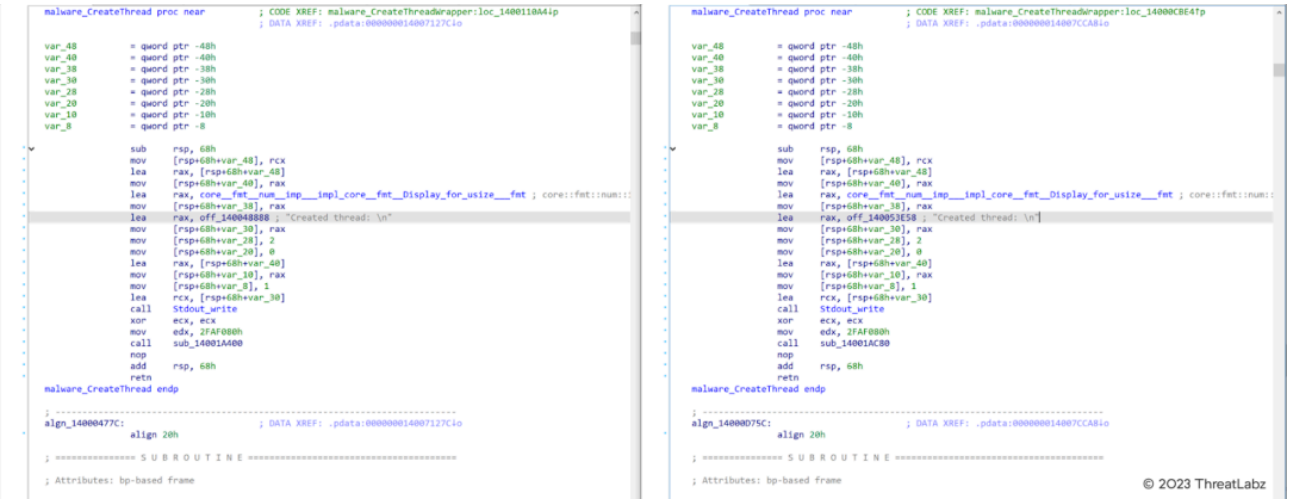

Figure 3. Comparison of *CreateThread* function and debug print statements in Nokoyawa 2.0 (left) and Nevada (right)

Many strings have also been slightly altered between Nokoyawa 2.0 and Nevada as shown in Table 3.

Nokoyawa 2.0	**Nokoyawa 2.1** **(Nevada)**
CIS lang detected! Stop working...	CIS. STOP!
Successfully deleted shadow copies from	Shadow copies deleted from
Couldn't create ransom note	Failed to create ransom note

Couldn't seek file:	Failed to seek file:
Couldn't read file:	Failed to read file:
Couldn't write to file:	Failed to write file:
Couldn't rename file	Failed to rename file

Table 3. Comparison between debug print strings in Nokoyawa 2.0 (left) and Nevada (right)

Encryption Algorithms

Nokoyawa 1.0 and 1.1 use the elliptic curve SECT233R1 (NIST B-233) via the Tiny-ECDH library to generate a per file Salsa20 key. Nokoyawa 2.0 and Nevada use Curve25519 via the open source x25519_dalek Rust library to derive a Salsa20 encryption key per file. In Nokoyawa 1.1 and 2.0, the file extension (as described in Table 2) is used as the nonce. The original version of Nokoyawa and Nevada ransomware use the hardcoded nonce values *lvcelvce* and *pmarpmar*, respectively.

Explore more Zscaler blogs
