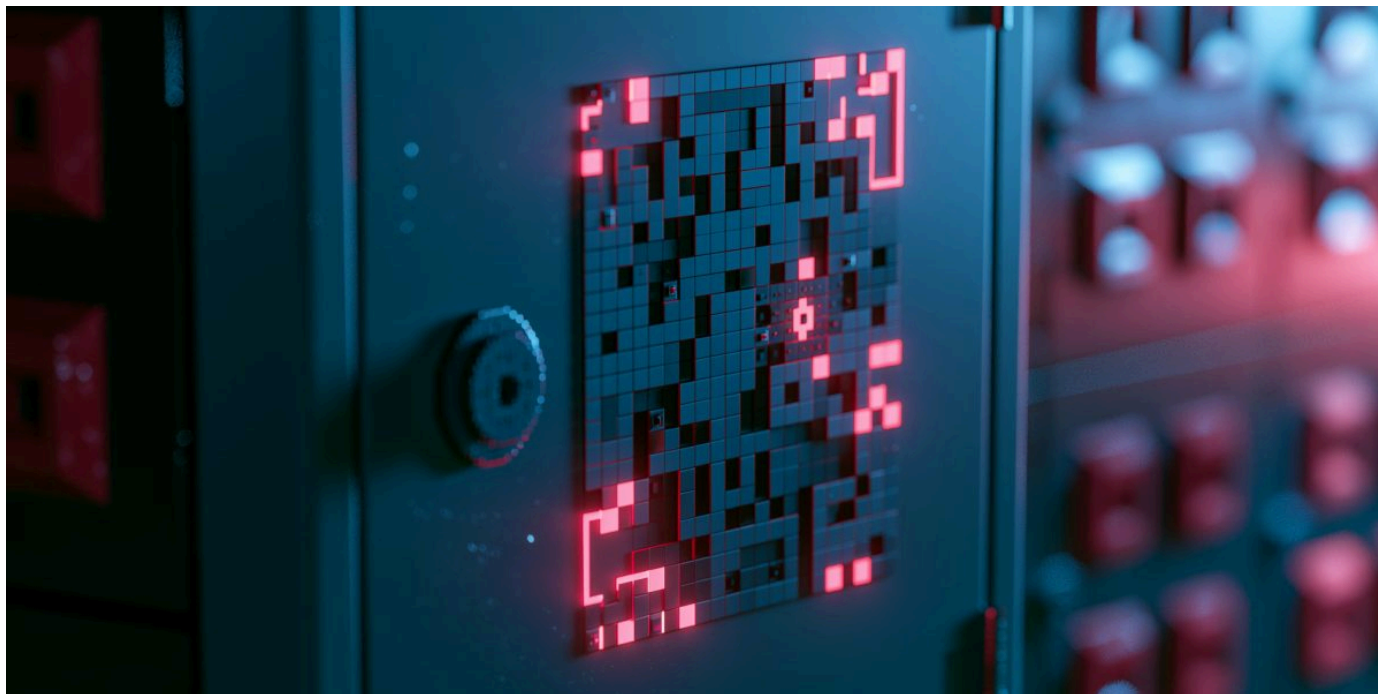


New GenieLocker ransomware for Windows, ESXi, and Linux

SL securelist.com/genielocker-ransomware-for-windows-linux-and-esxi/120843

Fedor Sinitsyn, Yanis Zinchenko

July 30, 2026



Introduction

The new GenieLocker ransomware family has been active since March 2026. It has been used in attacks against organizations in the Russian Federation, primarily in the manufacturing sector, and attributed to the Toy Ghouls group [by open-source intelligence](#) (link in Russian).

The Toy Ghouls, also known as Bearlyfy, Labubu and Laboo.booo, is a financially motivated extortion group, which previously relied on third-party encryption Trojans like RedAlert, LockBit, and Babuk. GenieLocker, apparently a custom design, upgrades their toolkit and reduces their reliance on third-party software. We discovered multiple samples of this Trojan in two variants: PE builds for Windows and ELF builds for Linux and ESXi.

Technical details

Modus operandi

We described typical TTPs and modus operandi of the Toy Ghouls threat actor [in the previous post](#) (link in Russian).

In this article, we aim to thoroughly describe the capabilities of Windows and Linux builds of the custom encryption Trojan GenieLocker. To give more context, we will also provide a brief overview of the attack that took place at the end of March 2026, where GenieLocker was deployed on the victim's systems.

Initial Access

During the incident, the attackers first entered the environment through an OpenVPN connection originating from an external partner's network. They likely exploited the trusted relationship with that partner and used stolen, yet still valid, credentials to connect.

Discovery and Credential Access

After breaching the target's network, the attackers installed additional tools on the compromised hosts, including OpenSSH, socks5.exe, SoftPerfect Network Scanner, and Mimikatz. They employed SoftPerfect Network Scanner for discovery and used Mimikatz to dump credentials. Forensic analysis also shows that they accessed the KeePassXC password manager already installed on several compromised machines, likely attempting to extract the stored credentials from the KeePass databases.

Lateral Movement and Command and Control

Lateral movement was performed by using RDP to reach Windows machines and SSH for Linux servers. The widespread deployment of the encryption Trojan was conducted with the legitimate utilities PsExec and PAExec. Additionally, the attackers established a reverse SSH tunnel to communicate with their command-and-control server.

Impact

During the impact phase, the attackers encrypted files on the compromised Windows machines with the PE version of the GenieLocker ransomware. On the compromised Linux and ESXi servers, they stopped active virtual machines and encrypted their disks using the ELF version of GenieLocker.

The tactics, techniques, and procedures seen here match those documented in earlier attacks attributed to the Toy Ghouls group. As in those prior incidents, forensic analysis found no evidence of data exfiltration, which is typical behavior for this threat actor. Toy Ghouls have not employed a double-extortion model and do not run a data-leak website.

Encryption Trojan for Windows

The Windows version of GenieLocker (MD5: 5d62c1349b8981c396c9a23f4f8f053c) is primarily written in C, but compiled with the C++ libraries using Microsoft Visual C/C++. The malware incorporates several ransom-related capabilities, including process termination, service shutdown, debugger evasion, and a sophisticated encryption routine. For its cryptographic operations, it relies on the open-source libsodium library.

Aligned with the recent trend supported by our expertise, as observed in attacks of some other ransomware strains, GenieLocker doesn't save the ransom notes on the victim's system. The Trojan doesn't contain any attackers' contact info or negotiation addresses. Instead, the attackers will need to deliver the ransom demands and contacts manually during the attack. This approach may be an attempt by the GenieLocker developers to avoid proactive detection of the ransomware process being triggered by the creation of multiple readme files.

```
" +-----+\n"
" | GenieLocker v%d | \n"
" +-----+\n"
"\n"
" Usage: genie_encrypt [options] [path]\n"
"\n"
" If no path given, encrypts all local + remote drives.\n"
"\n"
" Options:\n"
" -p, --percent N      Encryption percentage (0-100, default %d)\n"
"                      0 = encrypt only head (%s)\n"
" -r, --recursive      Process directories recursively\n"
" -l, --log FILE        Log file path\n"
" -h, --help            Show this help\n"
"\n"
" Build config:\n"
" Extensions:  %s\n"
" Vault ext:   %s\n"
" Head encrypt: %s\n"
" Start delay: %d sec\n"
" Self-delete: %s\n"
```

GenieLocker help message

Arguments and launch

GenieLocker supports multiple arguments for configuring its behavior.

Argument	Description
First argument	"Secret" argument, hex string value
-p, -percent N	Percentage of file content to encrypt
-r, -recursive	Process directories recursively
-l, -log <filename>	Set path for log file

-h, --help	Show help message
Last argument	Path to encrypt

GenieLocker expects the first argument to be a hex string referred to in the malware code as the “secret argument”, which is required for the ransomware to start. Most likely, the purpose of this is to avoid execution on sandboxes and other automated analysis environments. Another reason may be to prevent unauthorized usage by other threat actors.

```
SetConsoleOutputCP(65001u);
SetConsoleCP(65001u);
if ( _argc < 2 )
{
    stderr = _acrt_iob_func(2u);
    Log(stderr, " Missing secret argument\n");
    return 1;
}
```

Checking the secret argument

The secret argument is a hex value with a variable size that does not exceed 4096 bytes. This hex string value is converted to bytes and hashed with the SHA-256 algorithm. The result is compared to a hardcoded value. If they match, the literal string session is appended to the secret value, and the whole string is hashed with BLAKE2b-256, but the resulting hash is never used. This may be a part of a feature still in development.

```
// BLAKE2b-256
crypto_generichash_init(&s->blake_ctx, 0i64, 0i64, 32i64);
crypto_generichash_update(&s->blake_ctx, hex_secret, hex_secret_size);
crypto_generichash_update(&s->blake_ctx, "session", 7ui64);
crypto_generichash_final(&s->blake_ctx, &state->secret_blake2b, 0x20u);
```

Secret value hashing

Anti-debugging

GenieLocker contains multiple methods to inspect if its process is under debugging. After launch it makes the first check named Environment check and uses WinAPI functions IsDebuggerPresent and CheckRemoteDebuggerPresent to detect the debugger.

```
if ( IsDebuggerPresent()
    || (pbDebuggerPresent = 0,
        CurrentProcess = GetCurrentProcess(),
        CheckRemoteDebuggerPresent(CurrentProcess, &pbDebuggerPresent),
        pbDebuggerPresent) )
{
    stderr = _acrt_iob_func(2u);
    LOG(stderr, " Environment check failed\n");
    return 0i64;
}
```

Environment check

After the secret argument validation, GenieLocker starts a new parallel thread called watchdog. It runs in an infinite loop that performs a number of checks to detect well-known debuggers every 500 milliseconds. If at least one of the checks fails, the whole GenieLocker process immediately terminates.

```

if ( IsDebuggerPresent() )
    CloseProgram(s);
// 0x70 - FLG_HEAP_ENABLE_TAIL_CHECK |
//       FLG_HEAP_ENABLE_FREE_CHECK |
//       FLG_HEAP_VALIDATE_PARAMETERS
if ( (NtCurrentPeb()->NtGlobalFlag & 0x70) != 0 )
    CloseProgram(s);
ProcessHeap = NtCurrentPeb()->ProcessHeap;
if ( ProcessHeap->Flags != 2 || ProcessHeap->ForceFlags )
    CloseProgram(s);
pbDebuggerPresent = 0;
CurrentProcess = GetCurrentProcess();
CheckRemoteDebuggerPresent(CurrentProcess, &pbDebuggerPresent);
if ( pbDebuggerPresent )
    CloseProgram(s);
if ( CheckHardwareBreakpoints() )
    CloseProgram(s);
if ( CheckVectoredExceptions() )
    CloseProgram(s);
if ( CheckWindowsWithDebuggerNames() )
    CloseProgram(s);
if ( SearchDebuggerProcessNames() )
    CloseProgram(s);
if ( TextSectionCheckCRC32(&crc32) )
    CloseProgram(s);
Sleep(500u);

```

Watchdog checks

The only thing worth elaborating on is that the GenieLocker process calculates the CRC32 of its .text section when the watchdog thread is starting, saves the resulting hash, and then recalculates it again in every loop and compares with the initial value. In case the code in this section is modified by the debugger or other program, this method allows the Trojan to detect this modification.

Preparing for encryption

GenieLocker contains multiple exclusion lists. For example, it does not encrypt folders with names from the list below. Among those, there are mostly system folders, which are skipped to avoid corrupting the OS.

```

1 $recycle.bin;config.msi;$windows.~bt;$windows.~ws;windows;boot;program files;program files (x86);programdata;system volume
  information;tor browser;windows.old;intel;msocache;perflogs;x64dbg;public;all users;default;microsoft;appdata

```

The Trojan also avoids encrypting the following system Windows files.

```

1 autorun.inf;boot.ini;bootfont.bin;bootsect.bak;desktop.ini;iconcache.db;ntldr;ntuser.dat;ntuser.dat.log;ntuser.ini;thumbs.db;GDIPFONTCACHEV1.DA

```

The file extensions below are excluded from encryption as well.

```

1 386;adv;ani;bat;bin;cab;cmd;com;cpl;cur;deskthemepack;diagcab;diagcfg;diagpkg;dll;drv;exe;hlp;icl;icns;ico;ics;idx;ldf;lnk;mod;mpa;msc;msp;mssty
  ms;MD

```

Furthermore, the Trojan contains an exclusion list for host names. The malware retrieves the computer name using GetComputerNameA and checks it against this list, but in the sample in question, the list is empty.

```

" +-----+\n"
" | HOST WHITELISTED - ABORTING | \n"
" | Hostname: %-36s | \n"
" +-----+\n"

```

Output for whitelisted hosts

If the host name is not excluded, GenieLocker starts to kill processes that could be using the files of interest and therefore prevent the Trojan from encrypting them. These processes are listed below. The Trojan stops them by using the `TerminateProcess` function.

```
1 sql;oracle;ocssd;dbsnmp;synctime;agntsvc;isqlplussvc;xfssvcon;mydesktopservice;ocautoupds;encsvc;firefox;tbirdconfig;mydesktopqos;ocomm;dl
```

Additionally, the Trojan stops the following services using `ControlService` with the `SERVICE_CONTROL_STOP` control code.

```
1 vss;sql;svc$;memtas;mepocs;msexchange;sophos;veeam;backup;GxVss;GxBlr;GxFWD;GxCVD;GxCIMgr;1c;Mssqlserver;vmwp;vmms;vmcompute
```

Finally, GenieLocker starts encryption threads and searches for all available drives, including network shares, to encrypt them.

```
" +-----+\n"
" | GenieLocker v%d | \n"
" +-----+\n"
" | Threads: %-4d Head: %-10s Pct: %3d%% | \n"
" | Extension: %-33s | \n"
" +-----+\n"
```

Threads info output

File encryption and cryptography

The extension for the encrypted files is hardcoded in the Trojan's body. In the sample under review, it is `.03ffc1c4a3da0f02`. Before starting to encrypt each file, GenieLocker creates two auxiliary files:

- a lock file: `<filename.fileext>.03ffc1c4a3da0f02.lock`
- a journal: `<fileext>.03ffc1c4a3da0f02.journal`

The lock file helps to protect files from double encryption by other threads or instances. Inside this file, the Trojan stores the current PID obtained from the `GetCurrentProcessId` function.

The journal file contains the hardcoded string `VCJOURN`, value 1 (possibly version), some unused zeroed fields, total blocks to encrypt, and the count of blocks that are actually encrypted. The last field is a CRC32 hash sum for the integrity check of the journal content.

```
; GLJournal
db 'V', 'C', 'J', 'O', 'U', 'R', 'N', 0 ; magic
db 1 ; version
dq 0 ; field_0
dq 0 ; field_1
dq 0 ; field_2
dq 0 ; field_3
dq 20h ; total_blocks
dq 20h ; encrypted_blocks
dd 748B9E8Dh ; crc32
```

Journal content

By default GenieLocker encrypts files using 0x1000000-byte chunks. If the argument `-p` is passed (it sets the percentage of the file contents to be encrypted), the ransomware calculates how many chunks with 0x1000000 size are necessary to encrypt the specified percentage. Each chunk has a random position inside the file. Regardless of whether the percentage is set, even if it is zero, the first chunk in the beginning of the file will be encrypted anyway.

The Trojan encrypts the file content using the Authenticated Encryption with Associated Data (AEAD) algorithm `XChaCha20-Poly1305`, with a unique key and nonce for each file. The Trojan also adds a footer that contains the data necessary for future decryption and metadata. The metadata parts are encrypted using the same cipher and key as the file contents, but with a different nonce. The file key is encrypted using the `Curve25519-XSalsa20-Poly1305` scheme, with the attackers' master public key hardcoded in the Trojan's body.

The metadata of each encrypted file contains the following fields.

Value or name	Size (bytes)	Description
---------------	--------------	-------------

version	1	Hardcoded byte with value 1, most likely the version.
encryption_percent	1	Percentage of file content to encrypt, value from -p argument.
file_nonce	24	Nonce used during encryption of the file content.
original_filesize	8	Original size of the file before encryption.
total_chunk_count	8	Max count of chunks inside the current file.
chunk_size	4	Size of a single encrypted chunk (by default, 0x1000000 bytes on Windows and 0x400000 on ESXi and Linux).
remain_size	4	The number of bytes remaining after splitting the file content into chunks.
blake2b_digest_of_chunks	32	BLAKE2b-256 hash calculated from the original data of all chunks before they are encrypted. Used for integrity checks.
chunk_count	4	Number of chunks that were encrypted.
extension	64	A string with the additional ransomware extension.
poly1305_tags (array)	16 bytes per chunk	Array of Poly1305 tags of encrypted chunks.
bitmask	varies, one bit per each chunk	Chunks bitmask; if set, the chunk is encrypted; otherwise, it is not.

The chunks bitmask contains as many bits as the maximum number of chunks inside a file at 100%. If a bit at a specific index is set to 1, the chunk is encrypted. The value 0 means that the chunk is not encrypted. Since the Trojan encrypts files based on the percentage value, it needs to know which chunks were encrypted.

```

db 1 ; version
db 3Dh ; encryption_percent
db 2Ah, 0A2h, 0E1h, 1Ah, 19h, 54h, 0B8h, 0ADh, 0FBh, 1, 0BCh, 3, 98h, 42h; file_nonce
db 2Dh, 1Bh, 9Ah, 8Ah, 74h, 0E1h, 0F2h, 1Dh, 65h, 0D1h; file_nonce
dq 7E12DBCh ; original_filesize
dq 20h ; total_chunk_count
dd 400000h ; chunk_size
dd 212DBCh ; remain_size
db 15h, 80h, 9Fh, 5Ah, 7Fh, 74h, 0EAh, 95h, 50h, 70h, 0CEh, 15h, 0E8h; blake2b_digest_of_chunks
db 85h, 0B0h, 0ECh, 0D2h, 0D6h, 16h, 89h, 0F9h, 2Eh, 99h, 0A2h, 53h, 0E7h; blake2b_digest_of_chunks
db 98h, 66h, 1Bh, 0F4h, 0CBh, 0DEh ; blake2b_digest_of_chunks
dd 18h ; chunk_count
db 2Eh, 34h, 38h, 34h, 33h, 63h, 35h, 61h, 63h, 61h, 39h, 66h, 39h, 33h; extension
db 31h, 38h, 32h, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0; extension
db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0; extension
db 0, 0, 0, 0, 0, 0 ; extension

```

Metadata structure at the end of an encrypted file (without a Poly1305 tags array or bitmask)

Encryption Trojan for ESXi and Linux

Compared with its Windows counterpart, the Linux and ESXi version of GenieLocker (MD5: 9201e35e2993612612919a3c71302cab) is simpler: there is no secret argument, anti-debugging techniques, or exclusion lists. However, the sample has ESXi-specific features, such as double-fork support and the ability to modify the Welcome Message. The sample has the version v1 and, similarly to the Windows version, uses the libsodium library for cryptography.

```
GENIELOCKER v1 ESXi/Linux x64

GLOBAL OPTIONS:
  -h                Display help information

PROCESSING OPTIONS:
  -p <perc>         Encryption percentage (0-100, default 2, 0: Encrypt only file header)
  -j <workers>      Parallel worker threads (0: Auto-detect CPU cores)
  -r <dir>          Recursive directory processing (default: /vmfs/volumes)

RUNTIME CONTROL:
  -w <sec>          Initial delay before execution
  -d                Daemon mode (background process)
  -l <logfile>       Log output destination

BUILD CONFIGURATION:
  Target extensions:
  Lock extension:    .4843c5aca9f93182
  Auto-start delay:  0 seconds
  Self-destruct:     no

USAGE:
  genie_encrypt [options] <target_path>
```

ESXi version description

The command-line help output mirrors LockBit's styling, reinforcing the theory that GenieLocker's creators set out to craft a LockBit-style replacement for their own operations.

```
LOCKBIT5.0 ChuongDong Locker v1.01 Windows x64

USAGE
chuongdong64.exe [options]
* Command line length is limited to 500 characters.

BASIC OPTIONS
  -h                Show this help
  -p <dirs>         Semicolon-separated list of directories to encrypt
  -b <dirs>         Semicolon-separated list of directories to bypass

OPERATION MODES
  -i                Invisible mode (don't change extensions, no notes, don't change modification date)
  -v                Run in verbose visible mode with status bar in console
                   * Not available when using -p
  -d                Run in visible mode with debug output

NOTES SETTINGS
  -n <0/1/2>        Notes storage mode (0: none, 1: everywhere, 2: C:\ only)
                   * This option is ignored when using -i (invisible mode)
```

LockBit output design, possibly the source layout for the GenieLocker ESXi variant

Based on the default path of the encryption directory /vmfs/volumes, we can assume that this version is intended primarily for ESXi. Nonetheless, it can still be executed on Linux distributions.

Argument	Description
-p <perc>	Percentage of file content to encrypt
-j <workers>	Number of encryption threads
-r <dir>	Process directories recursively
-w <sec>	Delay before start
-d	Daemonizing the process
-l <logfile>	Path to log file

ESXi and Linux features

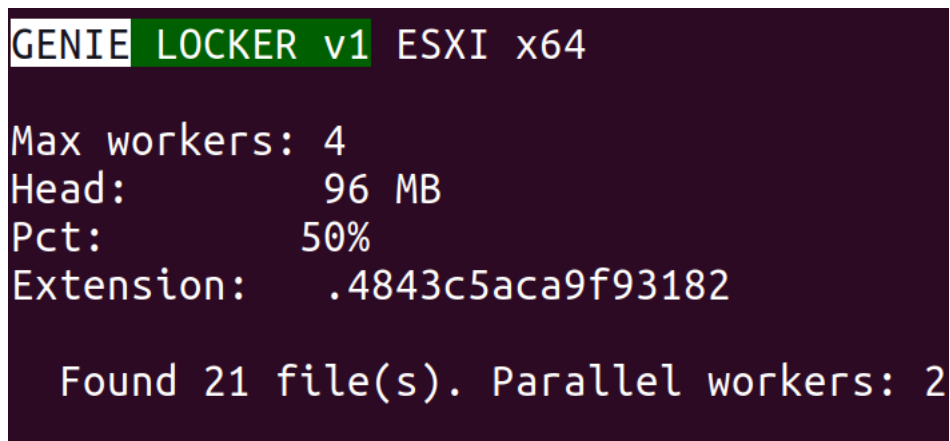
This build allows daemonizing its process with the -d flag, employing the classic double-fork method so the new process becomes fully detached from its parent.

This variant also modifies the /etc/vmware/welcome file, which contains the Welcome Message (Message of the Day) on the ESXi operating system. On Linux distributions, it does not change anything, because they use different paths for the Message of the Day. In the GenieLocker sample examined here, the message is left empty.

Additionally, the ESXi version supports a few basic features that are not included in the Windows version. For instance, there is a launch-delay option and the ability to set the number of encryption worker threads. This build also includes several features that already exist in the Windows variant, such as configuring the percentage of a file to encrypt, choosing the target directory, and setting the log file location.

File encryption

The encryption scheme for files is identical to the Windows version. The Trojan uses XChaCha20-Poly1305 to encrypt the file content and metadata, and Curve25519-XSalsa20-Poly1305 for key encryption.



```
GENIE LOCKER v1 ESXi x64

Max workers: 4
Head:        96 MB
Pct:         50%
Extension:   .4843c5aca9f93182

Found 21 file(s). Parallel workers: 2
```

File encryption summary

Victims

According to KSN telemetry, GenieLocker detections are overwhelmingly concentrated on endpoints located in the Russian Federation. In the March 2026 campaign, the primary sector under siege was manufacturing, with construction trailing closely, followed by financial services, retail, and technology.

Conclusions

Toy Ghouls are ramping up their campaign against Russian enterprises. The rollout of their home-grown encryption Trojan GenieLocker marks a major upgrade to the group's ransomware toolkit. By engineering bespoke ransomware that runs natively on Windows, Linux, and ESXi, the actor has cut their dependence on off-the-shelf ransomware families and unified the cryptographic backbone across all targeted platforms.

Kaspersky's products detect this malware as Trojan-Ransom.Win64.Agent.genie, HEUR:TrojanRansom.Win64.Generic, Trojan-Ransom.Linux.Agent.genie.

Indicators of compromise

Additional information about this threat is available to customers of the [Kaspersky Threat Intelligence Reporting](#) service. Contact: intelreports@kaspersky.com.

GenieLocker for Windows

[A50EAAF514F4F84E61CA2455A8789753](#) kftd.exe, genie_encrypt.exe
[F08F476F26B01D142CA73923DE65FC0C](#)
[FD46A80C2F45577263328984EDF7F4DC](#)
[DE3CFBB50F66079BFEE20A6F64E59433](#)
[780C8F4C6F077DA4DA96582987920362](#)
[D87D0B01D95ACC936B7DC47B8F41937A](#) run.exe, genie_encrypt.exe
[34A7F28E0BB69B0D49BACC88BDF20AC1](#) run.exe, run2.exe, genie.exe
[5D62C1349B8981C396C9A23F4F8F053C](#) genie_encrypt.exe
[A8842616C9057D5CF6E1FE1FA8C3C160](#)
[34B8828635F88078735799A3C1AC8E28](#)
[D3E06EB34D8EEEE7EF92CAC3AD0A20FF5](#)
[C68B6862725777651085650DB34947FC](#) consultant.exe
[9CD514FF2809CE0B993E3B8649E82A94](#)
[824CA1E906CC073EE5B0F3519DF69A8F](#)

[25480DAD40152EF3D0C6D38EECC9BD9B](#)
[7DAD78584795AA5C160520CC6ACCF260](#)
[18F61C6D686CFFD131C9FD3F3437064B](#) tempo.exe, kernel.exe
[9969A8221312DBA70DD5CBDDF83A146C](#)
[F7B9E36E94163A9A303160945F99267A](#)
[B893EAFED0659F70D4AC250F09073723](#)
[D661CF666B9ACBAB7CFEAE1127A261A9](#) genie.exe
[3A4479B51890373BFC4A011EF41FE376](#)
[58C0DDA52B8F069660166D61FD74F911](#)

GenieLocker for Linux and ESXi

[9201E35E2993612612919A3C71302CAB](#) vzdump

C2

[89\[.\]125.66.101](#)

- [Malware Technologies](#)
- [Linux](#)
- [Microsoft Windows](#)
- [Ransomware](#)
- [Malware Descriptions](#)
- [Data Encryption](#)
- [Malware](#)
- [Trojan](#)
- [ESXi](#)