MALWARE ANALYSIS REPORT

LunaStealer / LunaGrabber

Classification	Infostealer / Dropper
Analyst	Doug Metz
Organization	Baker Street Forensics
Date	2026-05-02
TLP	TLP:WHITE

1. Executive Summary

This report documents a static malware analysis investigation of two related samples attributed to the LunaStealer / LunaGrabber malware family. Both samples are PyInstaller-compiled Windows PE executables designed to steal credentials, browser data, session tokens, and Discord accounts from victim machines.

The analysis revealed two distinct components within the same campaign: a credential-stealing payload (sdas.exe) and a more sophisticated multi-stage dropper (loader.exe) containing an LZMA-compressed second-stage payload with anti-analysis capabilities and a live Discord injection mechanism hosted on GitHub.

Key findings include a remote injection script URL, an extensive C2 IP list, anti-VM/sandbox evasion using blacklisted process names, hardware IDs, MAC addresses, and usernames, plus a self-destruct mechanism designed to eliminate forensic artifacts post-execution.

2. Sample Overview

2.1 Sample 1 — sdas.exe (LunaStealer)

Property	Value
SHA-256	`4f3b8971d6c985125cc2cb1fec0ac5e700219510a84f991a7481d060861ea0d0`
Filename	`sdas.exe`
File Size	`47.76 MB`
File Type	`Win32 PE64 EXE`
Compile Time	`2025-12-01 15:00:24 UTC`
First Seen	`2025-12-01`
Python Version	`3.13`
Packer	`PyInstaller 2.1+`
Imphash	`f3c0dbc597607baa2ea891bc3a114b19`
Entropy	`3.9 (low — uncompressed bundle)`
Code Signature	`Absent`
Family (MB)	`LunaStealer`
VT Detections	`42/71 (trojan.generickdq/python)`
HA Detections	`56 AV — Win/malicious_confidence_100%`

2.2 Sample 2 — loader.exe (LunaGrabber)

Property	Value
SHA-256	`d4f57b427a281c4dc32d6b8cc12195e0ce262c9f80ac8a8505ff3a0b47536324`
Filename	`loader.exe`
File Size	`22.17 MB`
File Type	`Win32 PE64 EXE`
Compile Time	`2026-01-03 03:47:13 UTC`

First Seen	2026-01-03
Python Version	3.11
Packer	PyInstaller 2.1+
Imphash	f3c0dbc597607baa2ea891bc3a114b19 (IDENTICAL to Sample 1)
Entropy	6.9 (high — compressed internal payload)
Code Signature	Absent
Family (MB)	LunaGrabber / LunaStealer
VT Detections	40/70 (trojan.generickdq/python)
HA Detections	100 AV — Win/malicious_confidence_100%
MB Tags	exe, LunaGrabber, LunaStealer

3. Analysis Methodology & Pivot Points

Analysis proceeded through five progressive phases, each revealing indicators that drove the next phase of investigation.

Phase 1: Threat Intelligence Query (tiquery)

Both samples were submitted to the MalChela tiquery tool, which performs multi-source lookups across MalwareBazaar, VirusTotal, Hybrid Analysis, MetaDefender, and Triage simultaneously.

- **Pivot:** Sample 1 returned confirmed LunaStealer hits across all five sources with first-seen date of 2025-12-01.
- **Pivot:** Sample 2 was tagged LunaGrabber and LunaStealer by MalwareBazaar, with Triage clustering it alongside BlankGrabber, GlassWorm, IcedID, and Luca-Stealer — suggesting a multi-tool dropper rather than a standalone stealer.
- **Pivot:** The filename loader.exe vs sdas.exe suggested different functional roles within the same campaign.

Phase 2: Static PE Analysis (fileanalyzer + mstrings)

Both samples were submitted to fileanalyzer for PE header analysis and mstrings for string extraction with MITRE ATT&CK mapping.

Key finding: both samples share an identical imphash (f3c0dbc597607baa2ea891bc3a114b19), identical section layout (7 sections, same sizes), and identical import count (146). This confirmed both were compiled from the same PyInstaller loader template with different payloads embedded inside.

- **Pivot:** Sample 1 entropy 3.9 (low for PyInstaller) vs Sample 2 entropy 6.9 (high) — suggesting the embedded payload in Sample 2 is compressed or encrypted more aggressively.
- **Pivot:** mstrings identified 22-23 ATT&CK-mapped indicators including anti-debug (IsDebuggerPresent), timing evasion (QueryPerformanceCounter), token manipulation (OpenProcessToken), DLL hijacking (SetDllDirectoryW), and staging (GetTempPathW + ExpandEnvironmentStringsW).
- **Pivot:** The .fptable PE section and Tcl_CreateThread import — both PyInstaller artifacts — confirmed the Python-compiled nature, consistent with VT's trojan.generickdq/python classification.

Phase 3: PyInstaller Extraction (pyinstxtractor-ng)

Both samples were extracted using pyinstxtractor-ng to reveal the bundled Python module sets and entry point pyc files.

- **Finding:** Sample 1 entry point: sdas.pyc (Python 3.13, 112 files, 752 PYZ modules)
- **Finding:** Sample 2 entry point: cleaner.pyc (Python 3.11, 113 files, 760 PYZ modules)
- **Pivot:** The name cleaner.pyc in a file called loader.exe is a strong indicator of staged delivery or post-infection cleanup functionality.
- **Finding:** Both bundles carry identical core libraries: requests, requests_toolbelt, Cryptodome, cryptography, psutil, PIL, sqlite3, win32 — confirming same stealer framework.
- **Pivot:** Sample 2 uniquely contains a l.js reference (T1059 — Command and Scripting Interpreter), absent from Sample 1, indicating a JavaScript component not present in the earlier build.
- **Finding:** OpenSSL version difference: Sample 1 uses libcrypto-3.dll (OpenSSL 3.x) vs Sample 2 uses libcrypto-1_1.dll (OpenSSL 1.1) — different build environments approximately one month apart.

Phase 4: Bytecode Decompilation (pycdc)

Both pyc files were decompiled using pycdc, which supports Python 3.11 and 3.13 bytecode.

- **Finding:** sdas.pyc decompiled cleanly to reveal a classic infostealer import stack: CryptUnprotectData (browser credential decryption), AES (Cryptodome), ImageGrab (screenshots), sqlite3 (browser DB access), MultipartEncoder (exfil).
- **Pivot:** cleaner.pyc revealed a multi-layer obfuscation scheme using byte array encoding to reconstruct eval, getattr, and __import__ at runtime — designed to evade static string detection.

The cleaner obfuscation chain decoded as:

```
bytes([98,97,115,101,54,52]) → "base64"
bytes([90,88,90,104,98,65,61,61]) → b64decode("ZXZhbA==") → "eval"
bytes([90,50,86,48,...]) → "getattr"
bytes([88,49,57,112,...]) → "__import__"
```

Phase 5: LZMA Stage 2 Extraction

The cleaner.pyc contained a large embedded binary blob (_____ variable) with an LZMA magic header (\xfd7zXZ). The blob was located at offset 0x17d4 and decompressed to 102,923 bytes of stage 2 Python source.

- **Pivot:** The LZMA decompression is wrapped in a try/except LZMAError block with an os._exit(0) call on failure — a sandbox bail-out mechanism to prevent analysis in emulated environments.
- **Pivot:** The decompressed stage 2 used a further custom string encoding scheme (LjNNNNNN... alphabet substitution) to hide all IOCs from static analysis, with final execution via compile() + exec().
- **Finding:** Full decoding of the stage 2 revealed the complete malware capability set, C2 infrastructure, injection URL, and evasion lists documented in Section 5.

4. Malware Capabilities

4.1 Anti-Analysis & Evasion

The loader.exe stage 2 implements comprehensive pre-execution environment checks before any malicious activity begins:

- Process blacklist check — terminates if any analysis tool is running (see IOC section for full list)
- MAC address blacklist — over 80 known VM/sandbox MAC prefixes
- Hardware ID (HWID) blacklist — known virtualization HWIDs
- IP address blacklist — known sandbox infrastructure IPs
- Username and PC name blacklists — common analysis environment identifiers
- LZMA decompression sandbox bail-out — os._exit(0) on decompression failure
- Timing check — QueryPerformanceCounter to detect accelerated sandbox execution
- IsDebuggerPresent — active debugger detection

4.2 Discord Token Theft & Injection

Both samples target Discord desktop installations across all three release channels (stable, canary, PTB):

- Extracts tokens from Discord LevelDB local storage
- Reads Discord Local State for encrypted token storage
- Injects remote JavaScript payload into Discord desktop core module: `https://raw.githubusercontent.com/Smug246/luna-injection/main/obfuscated-injection.js`
- Injection persists across Discord restarts by patching discord_desktop_core
- Exfiltrates Discord user profile, payment methods, MFA status, Nitro type, and billing information

4.3 Browser Credential Theft

Targets 20+ Chromium and non-Chromium browsers using CryptUnprotectData for master key decryption and direct SQLite access to credential databases:

- Browsers: Chrome (all profiles), Edge, Brave, Opera, Opera GX, Yandex, Vivaldi, Kometa, Orbitum, CentBrowser, 7Star, Sputnik, Iridium, Epic Privacy Browser, Uran, Amigo, Torch, Lightcord
- Data collected: saved passwords, cookies, credit card numbers, browser history
- Uses requests_toolbelt MultipartEncoder for structured HTTP exfiltration

4.4 Roblox Session Cookie Theft

Dedicated Roblox cookie stealer targeting .ROBLOSECURITY session cookies, with subsequent API call to validate and enrich stolen sessions:

- Queries https://www.roblox.com/mobileapi/userinfo with stolen cookie
- Collects username, Robux balance, and avatar thumbnail URL

4.5 System Reconnaissance

- Desktop screenshot via PIL ImageGrab
- System information collection (CPU count, OS version, hostname, username)
- Network interface enumeration
- Running process enumeration

4.6 Self-Destruct

After execution completes, the malware deletes its own executable:

```
ping localhost -n 3 > NUL && del /F "{executable_path}"
```

The ping delay (3 seconds) allows the process to fully exit before the delete command executes.

5. Indicators of Compromise

5.1 File Hashes

Hash Type	Sample	Value
SHA-256	sdas.exe	4f3b8971d6c985125cc2cb1fec0ac5e700219510a84f991a7481d060861ea0d0
SHA-256	loader.exe	d4f57b427a281c4dc32d6b8cc12195e0ce262c9f80ac8a8505ff3a0b47536324
Imphash	Both	f3c0dbc597607baa2ea891bc3a114b19

5.2 Network Indicators

Injection URL (HIGH PRIORITY)

C2 IP Addresses

The following IP addresses were extracted from the stage 2 payload C2 list. These should be blocked at the network perimeter.

```
88.132.231.71, 78.139.8.5, 20.99.160.173, 88.153.199.169, 84.147.62.12,
194.154.78.160, 92.211.109.160, 195.74.76.222, 188.105.91.116,
34.105.183.68
92.211.55.199, 79.104.209.33, 95.25.204.90, 34.145.89.174,
109.74.154.90, 109.145.173.169, 34.141.146.114, 212.119.227.151,
195.239.51.59, 192.40.57.234
64.124.12.162, 34.142.74.220, 188.105.91.173, 109.74.154.91,
34.105.72.241, 109.74.154.92, 213.33.142.50, 93.216.75.209,
192.87.28.103, 88.132.226.203
195.181.175.105, 88.132.225.100, 92.211.192.144, 34.83.46.130,
188.105.91.143, 34.85.243.241, 34.141.245.25, 178.239.165.70,
84.147.54.113, 193.128.114.45
95.25.81.24, 92.211.52.62, 88.132.227.238, 35.199.6.13, 80.211.0.97,
34.85.253.170, 23.128.248.46, 35.229.69.227, 34.138.96.23,
192.211.110.74
35.237.47.12, 87.166.50.213, 34.253.248.228, 212.119.227.167,
193.225.193.201, 34.145.195.58, 34.105.0.27, 195.239.51.3,
35.192.93.107
```

5.3 Blacklisted Analysis Process Names

The malware checks for these process names on startup and terminates silently if any are found running:

```
httpdebuggerui, wireshark, fiddler, regedit, taskmgr, vboxservice,
df5serv
processhacker, vboxtray, vmtools, vmwareuser, vgauthservice, vmacthlp,
x96dbg
vmsrvc, x32dbg, vmwaretray, prl_cc, prl_tools, xenservice, qemu-ga
joebocontrol, ksdupmerclient, ksduper, joeboserver, ollydbg, pestudio
vboxheartbeat, idax64, vboxheartbeat, vmtoolsd
```

5.4 YARA Detection Opportunities

The following characteristics provide reliable YARA detection anchors:

- Both samples share identical imphash: f3c0dbc597607baa2ea891bc3a114b19
- PE section .fptable (PyInstaller frozen module table) at consistent offset
- Tcl_CreateThread import present in both samples
- String: ping localhost -n 3 > NUL && del /F (self-destruct command)
- String: luna-injection (injection repo reference in stage 2)
- String: .ROBLOSECURITY= (Roblox cookie targeting)
- String: discord_desktop_core (Discord injection target)

6. Campaign Timeline

Date	Sample	Event
2025-12-01	sdas.exe	Sample 1 compiled (PE timestamp 15:00:24 UTC)
2025-12-01	sdas.exe	First seen on MalwareBazaar, VirusTotal, Hybrid Analysis
2026-01-03	loader.exe	Sample 2 compiled (PE timestamp 03:47:13 UTC)
2026-01-03	loader.exe	First seen on MalwareBazaar and VirusTotal
2026-01-04	loader.exe	Triage sandbox report published — clustered with BlankGrabber, GlassWorm, IcedID, Luca-Stealer
2026-01-05	loader.exe	MetaDefender detection added
2026-05-02	Both	Analysis conducted by Baker Street Forensics

7. Tools & Analysis Environment

Tool	Purpose
MalChela (tiquery)	Multi-source threat intelligence hash lookup (MalwareBazaar, VirusTotal, Hybrid Analysis, MetaDefender, Triage)
MalChela (fileanalyzer)	Static PE analysis — hashes, entropy, section layout, imports, packing detection
MalChela (mstrings)	String extraction with MITRE ATT&CK mapping and IOC detection
pyinstxtractor-ng	PyInstaller archive extraction to recover bundled Python modules and entry point pyc files
pycdc (Decompyle++)	Python bytecode decompilation supporting Python 3.11 and 3.13
Python 3.9 (lzma)	LZMA stage 2 payload extraction from embedded binary blob
MalChela MCP Plugin	Claude Desktop integration — tools invoked directly within the analysis session
Claude Desktop	AI-assisted analysis orchestration and pivot identification

8. Recommendations

8.1 Detection & Blocking

- Block outbound connections to the GitHub raw URL: raw.githubusercontent.com/Smug246/luna-injection/
- Add all C2 IPs from Section 5.2 to network blocklists
- Deploy YARA rules based on the imphash, .fptable section, and string indicators from Section 5.4
- Alert on process creation of ping followed by del /F within 5 seconds (self-destruct pattern)

8.2 Incident Response

- On suspected infection: immediately revoke and regenerate all Discord tokens
- Audit browser saved credentials, cookies, and payment methods — treat all as compromised
- Check for modified discord_desktop_core files in Discord AppData directory
- Check startup locations and scheduled tasks for persistence mechanisms
- Review network logs for outbound connections to C2 IP list

8.3 Reporting

- Report the GitHub repository (Smug246/luna-injection) to GitHub Trust & Safety for takedown
- Submit samples to any TI platforms not yet covered