

a fake resume invoked China's defence-tech elite, then installed VShell

 blog.himanshuanand.com/2026/08/a-fake-resume-invoked-chinas-defence-tech-elite-then-installed-vshell

August 27, 2026

Table of Contents

Disclosure: this research was conducted using an ANY.RUN account provided as part of a collaboration. All analysis and conclusions are my own. I did not execute the malware locally. Local work was limited to archive extraction, parsing, disassembly and decoding traffic already captured by the sandbox.

TLDR

I found a Chinese-language “resume” that is actually a Windows executable. It claims to come from a student at one of China’s **Seven Sons of National Defence**, opens a real Word document, which is nice of it and quietly delivers **SNOWLIGHT and the VShell remote-access trojan**, which is less nice.

The chain is:

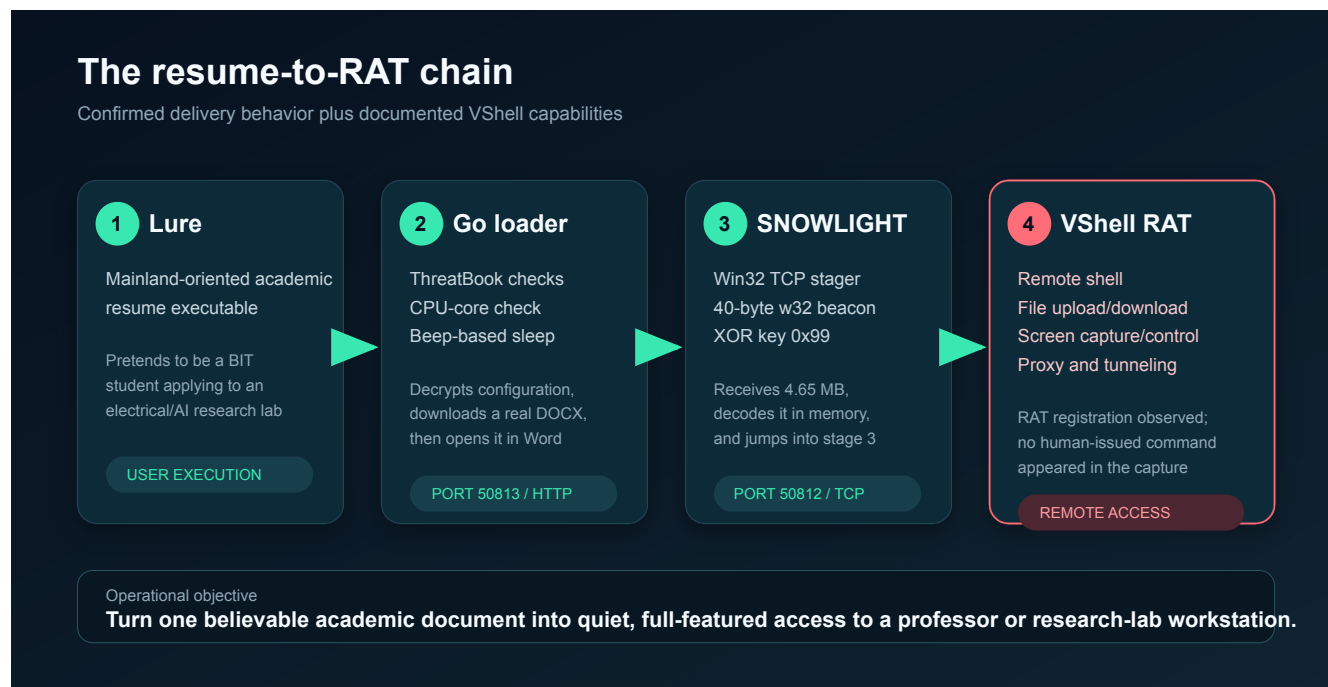
1. A fake Beijing Institute of Technology graduate-school resume is opened.
2. A custom Go loader checks for sandboxes, verifies the machine has at least four CPU cores and uses Beep as a sleep timer.
3. It downloads and opens a genuine DOCX decoy so the victim sees the document they expected.
4. It reflectively runs a 1,454-byte Windows SNOWLIGHT stager.
5. SNOWLIGHT connects to 38.207.178[.]192:50812, downloads 4.65 MB, XOR-decodes it with 0x99 and jumps into a fileless VShell RAT.

What is it meant to do? **Give an operator quiet, interactive access to a professor or research-lab workstation: command execution, file access, screen capture and a route further into the victim network.**

That is the most defensible objective. The malware proves remote access. The decoy tells us why that access was packaged as this particular document.

The lure is most consistent with a **mainland Chinese academic recipient**, at moderate confidence. That describes the intended victim, not the attacker. Attribution is less exciting: this is best described as an **unattributed actor using a mainland-China-oriented academic**

lure and the now widely available SNOWLIGHT/VShell ecosystem. The evidence does not establish the operator's language, nationality, employer or government sponsor. UNC5174 is relevant history, not a conclusion.



how this started

The sample appeared in ANY.RUN with a filename that translates roughly to:

Beijing Institute of Technology_network engineering major_
fresh graduate_Zhang Yuguang_personal resume (2)(1).zip

Inside the archive was an executable carrying nearly the same name. Windows hides known extensions by default, document icons remain a thing and humans remain human. This business model has survived more Windows releases than Internet Explorer.

ANY.RUN gave it a 100/100 malicious score, labeled the process `vshell` and showed the executable launching `cmd.exe`, opening a DOCX in Word and talking to one IP in a Hong Kong-registered netblock over two ports. Public geolocation sources disagree about where the server was physically located, so the registration is the claim I can actually support.

Malicious activity

北京理工大学_网络工程专业_应届毕业生...
 MD5: 533B7E37E4320C0179AB3F8B220377A7
 Start: 21.08.2026, 07:51 Total time: 187 s

Win10 64bit

arch-exec payload golang vshell rat

Indicators: Tracker: Remote Access Trojan

Results **new** Get sample Restart

CPU RAM

Processes 7 Browser Data 0 **new** Actions 2 **beta**

Filter by PID or name Only important

PID	Process Name	Path	Size	Count	Score
2248	WinRAR.exe	"C:\Users\admin\AppData\Local\Temp\北京理工...	2k	4k	89
2480	COM slui.exe	-Embedding	681	3k	67
5712	北京理工大学_网络工程专业_应届毕业生_张宇光...	vshell	394	212	52
5804	cmd.exe	/c start C:\Windows\Temp\北京理工大学_网络工...	1k	1k	66
6448	conhost.exe	0xffffffff -ForceV1	201	132	34
1348	WINWORD.EXE	/n "C:\Windows\Temp\北京理工大学_网...	23k	17k	198
6664	ai.exe	"AADAC0F9-C83B-4783-AF3C-862F4AA1B67B" ...	193	62	31

Process details ID 5712 Malicious

北京理工大学_网络工程专业_应届... **AI**

Username: admin
 Start: +10131ms Indicators:

100
OUT OF 100

Command line **AI**
 "C:\Users\admin\Desktop\北京理工大学_网络工程专业_应届毕业生_张宇光-个人简历 (2) .exe"

More Info Hide all

Danger 2

VSHELL has been detected (SURICATA)

Executing a file with an untrusted certificate Sigma Rule

Public task: [ANY.RUN analysis 8d27f4bf-ed8c-461d-96e6-86968464dd86](#)

The obvious answer was “it is a RAT.” The useful question was **what does each stage do and what does the combination tell us about the operator’s objective?**

So I took it apart.

the document that was trying a little too hard

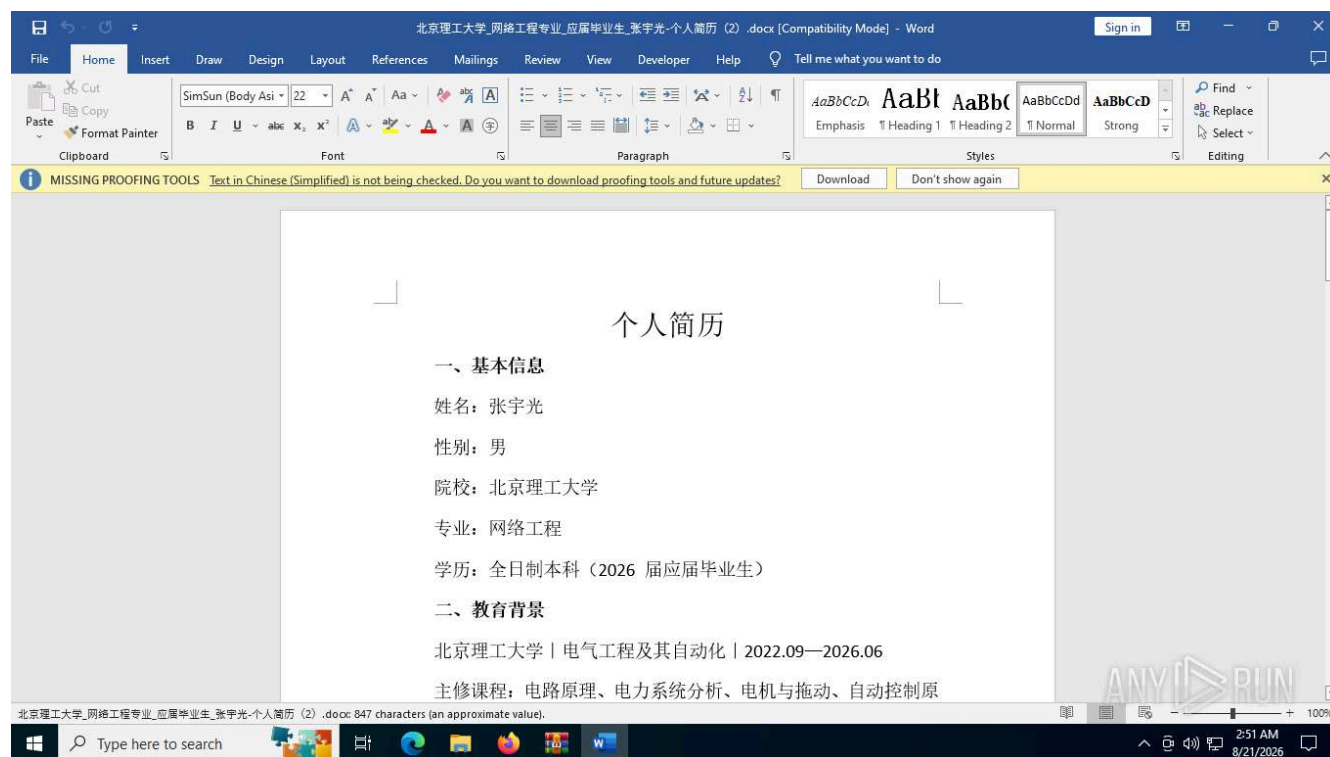
The malware downloads a real Word document from:

[http://38.207.178\[.\]192:50813/MySQL_LOG.txt](http://38.207.178[.]192:50813/MySQL_LOG.txt)

It saves it as:

C:\Windows\Temp\Beijing Institute of Technology_..._personal resume (2).docx

Then it launches Word. The victim gets the thing they clicked for, while the interesting part continues in memory.



At first glance it looks normal: Zhang Yuguang, male, Beijing Institute of Technology, class of 2026, network engineering.

Then you read past the first paragraph.

The education section says the major is **Electrical Engineering and Automation**, not Network Engineering. The grades field is blank. The honors say “delete as needed.” The certificates say “adjust as needed.” There is no phone number, email address, employer or verifiable internship. It is a three-page resume that forgot the small detail of telling the reader how to contact the applicant.

The research interests are much more specific:

- AI-based power-grid fault diagnosis
- intelligent electrical systems
- renewable-energy converter control
- power-system automation
- joining “your research group” and completing experiments and papers

This is not really a corporate job application. It reads like an unfinished template for contacting a **prospective graduate supervisor**.

That changes the victim model. The named university is probably the applicant’s claimed school, not the target. The likely recipient is a professor or research group working on electrical engineering, energy systems or applied AI, possibly at another institution.

the university name is doing a lot of work

Beijing Institute of Technology is not just a convenient university name copied from a rankings table.

[ASPI’s China Defence Universities Tracker](#) categorizes BIT as one of the **Seven Sons of National Defence**, a group of universities administered by China’s Ministry of Industry and Information Technology and closely associated with the country’s defence research ecosystem. The tracker lists BIT research areas including armament science, control systems, communications, optical engineering and unmanned aerial vehicles.

BIT’s [own laboratories page](#) prominently lists the State Key Laboratory of Explosion Science and Technology and the Key Laboratory of Intelligent Control and Decision of Complex System. Subtle institution. Very difficult to imagine why a social engineer might consider the name technically prestigious.

The harder-edged fact comes from the U.S. government. In December 2020, the Commerce Department [added BIT to the Entity List](#). The published rule says BIT was added for “acquiring and attempting to acquire U.S.-origin items in support of programs for the People’s Liberation Army.”

That makes the lure more interesting. A supposed student from a prestigious, defence-linked technical university is a credible reason for a professor working on power systems, intelligent control or applied AI to open a research statement. The affiliation adds authority and may signal that the sender’s work is worth reading.

It does **not** prove any of the following:

- That the applicant is real or has any connection to BIT

- That BIT created, distributed or knew about the file
- That BIT itself was the intended victim
- That defence research was the confirmed collection target
- That the operator was foreign to China, state-sponsored or conducting espionage

The malware gives us a claimed affiliation, not an enrollment record. The accurate headline is that the lure **invoked the name of a defence-linked university**. Anything stronger would turn an excellent hook into a false allegation.

Metadata supports recent preparation but does not identify an operator:

```
Created:      2026-06-05 07:27 UTC
Modified:     2026-08-13 05:42 UTC
Application:  WPS Office 12.1.0.28043
Creator:      Work let me happyyy
Last editor:  Fifty-one (Chinese: wu shi yi)
WPS locale:   2052 (Simplified Chinese, PRC)
Document:     3 pages, 923 words, 986 characters
Template UID: 1319704924
```

The sample ran on 21 August, eight days after the last edit. The WPS user ID and author fields may have come from a template. Treating inherited Office metadata as an operator passport is how threat-intel fan fiction begins.

who was this written for?

There are four different questions here and combining them produces bad attribution very efficiently:

1. What language and theme does the lure use?
2. Who does the document claim the applicant is?
3. Where is the likely recipient?
4. Who built or operated the malware?

The first two are directly observable. The document uses Simplified Chinese and claims to be from a Beijing Institute of Technology student. The claim is unverified. The fourth is unknown. The third can be assessed, but only probabilistically.

The mainland-China hypothesis is strongest because the document combines several local academic markers:

- 应届毕业生, the standard mainland term for a fresh graduate
- 个人简历, 课题组, 读研 and the deferential phrase 贵课题组

- 大学英语六级, the mainland College English Test Band 6
- 计算机二级, the National Computer Rank Examination's second level
- 三好学生, a PRC education-system honor
- WPS locale 2052, plus 宋体 and 黑体 fonts
- A supervisor-centered pitch promising to follow the research group's arrangements, complete experiments and publish papers

A Taiwan-oriented version would be expected to use Traditional Chinese forms such as 應屆畢業生, 履歷, 研究室 or 指導教授. It does not. Hong Kong, Macau, Singapore and overseas Chinese-speaking supervisors remain possible, but an English-language, department- or laboratory-framed application would be less surprising in those settings.

None of this proves the server operator was in mainland China. It says the lure writer wanted the file to look native to a mainland academic workflow. The likely recipient is a professor or research-group leader working on power systems, electrical engineering, renewable-energy control or applied AI. Beijing Institute of Technology is the claimed applicant's institution, not necessarily the victim's.

Assessment: mainland China is the leading intended-victim geography at moderate confidence; attacker geography remains unknown.

stage 1: a Go loader with trust issues

The first executable is a 5,073,728-byte, 32-bit Go binary built with Go 1.22.0 for Windows 386, with CGO_ENABLED=0, a zeroed linker timestamp and a nominal Kingsoft signature that fails validation with HashMismatch. Its internal project name is pdfrehuai and unlike the final payload it left useful symbols and source paths behind:

```
pdfrehuai/sandbox.AntiWeibu
pdfrehuai/sandbox.BeepSleep
pdfrehuai/loder.ADsMemLoad
pdfrehuai/crypto.Base64XorAesBase32Decode
pdfrehuai/mylib.DownloadFile
./main-Base64XorAesBase32-encrpurl.go
```

The spelling is not mine. Malware developers also ship on Fridays.

ThreatBook sandbox check

AntiWeibu refers to **Weibu / ThreatBook**, a Chinese threat-intelligence and sandbox provider. The function obtains its own executable directory, lowercases it and checks whether it contains:

C:\Users\Administrator\Desktop

If that sandbox-like path is present, it enumerates the directory and looks for a filename containing 2016. Matching the condition terminates execution. If the path is absent, the function returns and execution continues.

This is not generic “maybe I am in a VM” behavior. It is a check written for a particular analysis environment.

CPU check

The loader reads Go’s cached logical-processor count and exits when the machine has fewer than four. Small sandbox VMs often receive one or two. A real professor’s workstation probably has more. Very scientific threat model: if the computer can handle MATLAB, it can handle malware.

BeepSleep

Instead of trusting Sleep, the loader dynamically resolves kernel32!Beep, calls it at 30,000 Hz for ten seconds, records the time before and after and checks that ten whole seconds elapsed. If the sandbox accelerates time, it exits. On normal desktops that frequency is inaudible; the function name is funnier than the user experience.

encrypted configuration

The loader’s strings are not stored as normal URLs. The decoding pipeline is:

```
Base32hex decode
-> AES-CBC decrypt
-> PKCS#7 unpad
-> bitwise NOT every byte
-> Base64 decode
```

AES key and IV:

YtWzxwZimsZoeMen

Decoded values:

```
http://38.207.178[.]192:50813/EasyConnectUpdata_Log.txt
http://38.207.178[.]192:50813/MySQL_LOG.txt
C:\Windows\Temp\Beijing Institute of Technology_..._personal resume (2).docx
```

This is reproducible directly from the embedded bytes. The first value begins as 128 printable Base32hex characters, becomes 80 bytes after Base32hex decoding, 72 after AES-CBC and

padding removal, then 53 bytes after inversion and Base64 decoding: the full staging URL. The other two values follow the same pipeline. No live request is required to recover them.

MySQL_LOG.txt is the Word decoy. EasyConnectUpdate_Log.txt is encrypted shellcode. The names are camouflage for anyone glancing at HTTP logs and apparently losing interest after the file extension.



memory execution

The function named ADsMemLoad dynamically resolves:

```
Activeds.dll!AllocADsMem
Activeds.dll!ReallocADsMem
kernel32.dll!VirtualProtect
kernel32.dll!RtlMoveMemory
kernel32.dll!CreateThread
kernel32.dll!WaitForSingleObject
```

It allocates memory, copies the downloaded bytes, marks them executable, creates a thread at the payload and waits. The shellcode never needs to become a normal executable on disk.

At this point the user is reading a resume and the loader is starting stage 2. Multitasking.

stage 2: SNOWLIGHT, now with Windows

The HTTP server identifies itself as SimpleHTTP/0.6 Python/3.11.0. Its 6,224-byte response decodes to **1,454 bytes of x86 shellcode**:

SHA-256: 0524619d2471d77aba4b7993f5ffbaa4b8be6d2c0d91e63a02943851dc4b6404

It uses a classic ROR-13 API resolver and loads functions without a useful import table. All 16 hashes resolve to WSASStartup, WSASocketA, connect, send, recv, closesocket, inet_addr, gethostbyname, VirtualAlloc, GetTempPathA, _access, strcpy, strcat, strlen, printf and wsprintfA.

The behavior is a byte-for-byte conceptual match for the Windows SNOWLIGHT stager documented by [SOC Radar](#), [Sekoia](#) and [HPE Threat Labs](#).

Its job is intentionally small:

1. Build %TEMP%\log_de.log.
2. Exit if that file exists. It is a kill switch or operator exclusion marker.
3. Connect to 38.207.178[.]192:50812, retrying every ten seconds.

4. Send a 40-byte architecture and server check-in.
5. Allocate exactly 30,000,000 bytes of executable memory.
6. Receive the next payload in 0x64000-byte chunks.
7. XOR every byte with 0x99.
8. Close the socket and call the decoded buffer.

The exact client check-in was captured in the PCAP:

[illegible]

That 40-byte packet is useful for detection because it is much more specific than “a Windows computer used TCP.”

SNOWLIGHT does not authenticate the server, verify a signature or hash, validate an MZ header or receive an explicit payload size. It keeps copying `recv` results into the fixed 30,000,000-byte region and appears not to enforce that boundary. The panel stager is small because it delegates both trust and memory safety to whoever answered the socket. What could go wrong has been scheduled for the next stage.



stage 3: the payload the server did not want us to keep

The sandbox network table showed a 4 MB response, but no file appeared in the dropped-file list. That makes sense: SNOWLIGHT receives the payload directly into executable memory.

The PCAP still had it.

I downloaded the recorded PCAP from ANY.RUN, reassembled only the server-to-client stream for 38.207.178[.]192:50812 and XORed it with 0x99. No connection to the live server was made.

Result:

Encoded stream size: 4,649,984 bytes
Encoded SHA-256: ed2eaa6ef3eda95383b6efc35b88acbca742ad7bd118931f74727a3139ff7e97

Decoded magic: MZ
Architecture: Windows x86
Decoded SHA-256: c666ac4f1a1b8df7ccfe8b19705279acd8b7eb7a4d0b3802bb3465064883ab25

The result is a valid nine-section, 32-bit Go PE with image base 0x00400000, entry point 0x00401480, image size 4,648,960 and a zeroed timestamp. Its 3,088,384-byte .data section has entropy 7.984, almost the theoretical maximum of 8, which is consistent with the embedded configuration and much of the program being encrypted or obfuscated. Go build information exists but reports unknown; symbols are obfuscated and there is no helpful plaintext VShell label. The author did not leave a README inside the implant. Disappointing.

Windows Defender blocked the cleartext PE immediately after reconstruction. I did not disable or bypass it. Further inspection decoded the captured stream only in the analysis process's memory.

Why call it VShell if the string is hidden?

- ANY.RUN's Suricata rules identified the protocol as VShell.
- The delivery stub is the exact SNOWLIGHT family used to stage VShell.
- The final payload has the expected obfuscated Go core structure.
- The panel behavior, w32 architecture tag, %TEMP%\log_de.log, 0x99 XOR and cross-platform stager design all match public VShell/SNOWLIGHT research.

That is stronger than trusting one antivirus label. Independent static and network features all land on the same family.

the RAT checked in again

Execution did not stop at the final PE. Immediately after the 4.65 MB transfer, the host opened three new TCP sessions to the same 50812 listener from local ports 49805, 49806 and 49807.

I reassembled both directions and parsed **198 complete VShell frames**. Every frame uses the public protocol's four-byte little-endian length followed by a 12-byte nonce, AES-GCM ciphertext and a 16-byte tag. Subtracting that 28-byte encryption overhead exposes a very recognizable startup sequence:

Encrypted size	Plaintext size	Protocol role
37 bytes	9 bytes	length + five-byte VShell version
60 bytes	32 bytes	MD5 challenge or response

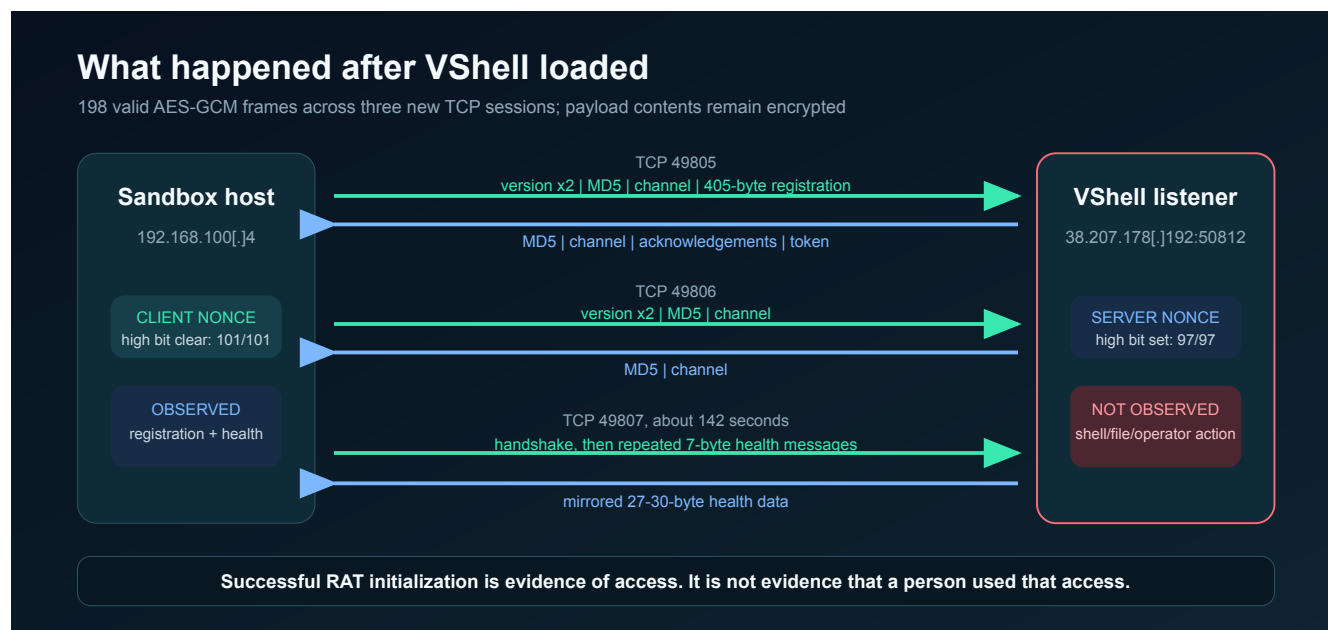
32 bytes	4 bytes	channel type, such as conf/main
433 bytes	405 bytes	client host-registration record
35 bytes	7 bytes	repeated health message
55-58 bytes	27-30 bytes	repeated health response/data

The first connection is consistent with completing registration, including the 405-byte client record. The second performs another channel handshake. The third repeats small, nearly mirrored health frames for about 142 seconds. [Public VShell protocol research](#) documents the same version negotiation, MD5 verification, conf registration and heartbeat loop.

VShell derives its AES-256-GCM key as the ASCII hex digest of MD5(salt). The salt is obfuscated inside this final payload. Default and common values including qwe123qwe, qwe123, 123456, vshell, veo, admin and an empty string failed authentication, so I cannot show the plaintext fields.

The sizes, ordering, direction-marked nonces and cadence nevertheless prove that the RAT initialized and exchanged its normal control protocol. They do **not** prove a human was at the console. I found no unusual frame sizes, file transfer, shell stream or other break from registration and health traffic.

Observed: successful VShell registration and C2 health traffic. Not observed: an operator issuing a command.



okay, but what does the malware actually do?

There are two answers: what this sample **demonstrably does** and what the delivered VShell platform **allows the operator to do**.

confirmed directly in this sample

- Detect and avoid analysis environments
- Refuse low-CPU systems
- Detect accelerated sleep behavior
- Decrypt hidden network configuration
- Download a benign-looking decoy
- Open the decoy in Word to preserve the illusion
- Download shellcode from remote infrastructure
- Execute shellcode in its own memory
- Establish a raw TCP staging connection
- Download and decode a full RAT without writing it normally to disk
- Transfer execution into the VShell core
- Complete VShell registration and exchange sustained encrypted health traffic

available to the VShell operator

Public VShell analysis and recovered console material document:

- Interactive terminal and arbitrary command execution
- File browsing, upload and download
- Screenshot capture and interactive screen access
- Host, user, process and network reconnaissance
- Reverse proxying and tunneling into otherwise unreachable systems
- Multiple C2 transports, including raw TCP, WebSocket, KCP and others

This matters because VShell is not a one-shot password stealer. It is an **operator platform**. The initial executable does not need to know which documents are valuable or which neighboring machine should be targeted. A human can decide after the victim checks in.

I did not observe an operator issuing commands in this sandbox run. I also did not prove persistence, document theft or lateral movement on this particular host. Those are capabilities and plausible follow-on actions, not facts I am going to promote to “observed” because the conclusion sounds cooler.

showing the objective without making things up

Malware rarely contains a string saying `OBJECTIVE=STEAL_PROFESSOR_RESEARCH`. You infer objective by combining behavior, access level, targeting and concealment.

Full VShell RAT, not a simple downloader: the operator wants interactive control after infection. That supports continued access, collection and human-directed follow-on decisions.

Fileless staged execution: reducing obvious disk artifacts helps the implant stay quiet long enough for somebody to use the access.

Real DOCX opened after infection: keeping the victim from suspecting failure is part of the operation, not a side effect.

Graduate-supervisor language: the document is written for an academic recipient, making professor or research-group targeting more likely than corporate recruiting.

AI, power-grid and renewable-energy themes: research data, institutional accounts and access to technical networks are all plausible targets.

Screen, file, shell and proxy capabilities: the workstation is a foothold. It does not have to be the operator's final destination.

My confidence-graded assessment:

High confidence: the immediate objective is remote access to the victim machine.

Moderate confidence: the intended victim is mainland Chinese university faculty or a technical research group, not ordinary corporate HR.

Moderate confidence: likely follow-on goals include collecting research material, credentials and internal access, with lateral movement available if the host is valuable.

Low confidence: whether the ultimate motivation is state-directed espionage, commercial theft or access brokerage.

One-line version: **turn a believable academic document into quiet, full-featured control of a research workstation, then let the operator decide what is worth stealing.**

attribution: please keep your red string on the corkboard

SNOWLIGHT has history. Mandiant named it while tracking UNC5174 and [Sysdig linked SNOWLIGHT-delivered VShell to UNC5174](#), a China-nexus contractor associated with espionage and access brokering.

If this analysis stopped in 2025, "possible UNC5174" would be tempting.

But the ecosystem changed. Cracked VShell releases and loader-generation panels circulated widely. More importantly, SNOWLIGHT is not merely malware that several actors happened to copy. It is a **default stager generated by the VShell management panel**. The operator selects Windows 32-bit, TCP and staged delivery; the framework produces the same broad implementation seen here.

[Trend Micro demonstrated why that distinction matters](#). Another report had assigned a SNOWLIGHT-to-VShell intrusion to UNC5174. Trend instead tied it to Earth Lamia with actor-specific VOIDMAW packaging and a repeated PDB path, while explicitly warning that SNOWLIGHT alone was unreliable because any VShell user could generate it. The panel-generated stager invalidated the tool-only attribution; the packaging artifacts supplied the positive one.

[HPE reached the same practical conclusion](#) in a 2026 PyPI investigation: leaked and cracked VShell builds have moved the framework beyond an exclusive APT toolkit.

What supports a mainland-China-oriented **victim context** here:

- Simplified-Chinese academic lure and PRC-specific education vocabulary
- WPS Office locale 2052
- Supervisor-centered research-group language

Separate facts place parts of the **development and tooling context** in Chinese-language security communities:

- A function specifically named AntiWeibu
- Evasion tailored to the Chinese ThreatBook sandbox
- VShell's origin and popularity in Chinese-language offensive-security communities

Neither category is a nationality test. The lure describes the intended social context; it does not tell us who sat behind the panel. The tooling facts may belong to an upstream loader author, copied source or a commodity framework rather than this operator. The AntiWeibu technique was also published in Chinese offensive-development material before this sample, so copying it does not identify a private toolmaker.

actor comparison

Candidate	Meaningful overlap	Contradicting or missing evidence
UNC5174	Historical SNOWLIGHT/VShell use	Published chains were mainly Linux, exploitation-led; no IOC/lure match
Earth Lamia	Windows VShell; universities	Published access uses public-facing exploits and VOIDMAW/DLL artifacts

UAT-8302 and match	Windows SNOWLIGHT -> VShell, 0x99	Government intrusions, sideloading different malware/IOCs; no lure
UNC6586 Windows,	SNOWLIGHT/VShell component	Linux React2Shell chain; no phishing or infrastructure match
Commodity needs VShell user assumption	Exact panel-generated w32/TCP stager; cracked builds available	Does not identify one operator, but no unsupported actor-specific

The last row requires the fewest leaps. Cisco Talos calls SNOWLIGHT a generic VShell stager in its [UAT-8302 research](#) and Google describes VShell as publicly available and used by actors with varying motivations in its [UNC6586 reporting](#). Neither source links this campaign to those groups.

What is missing for actor-level attribution:

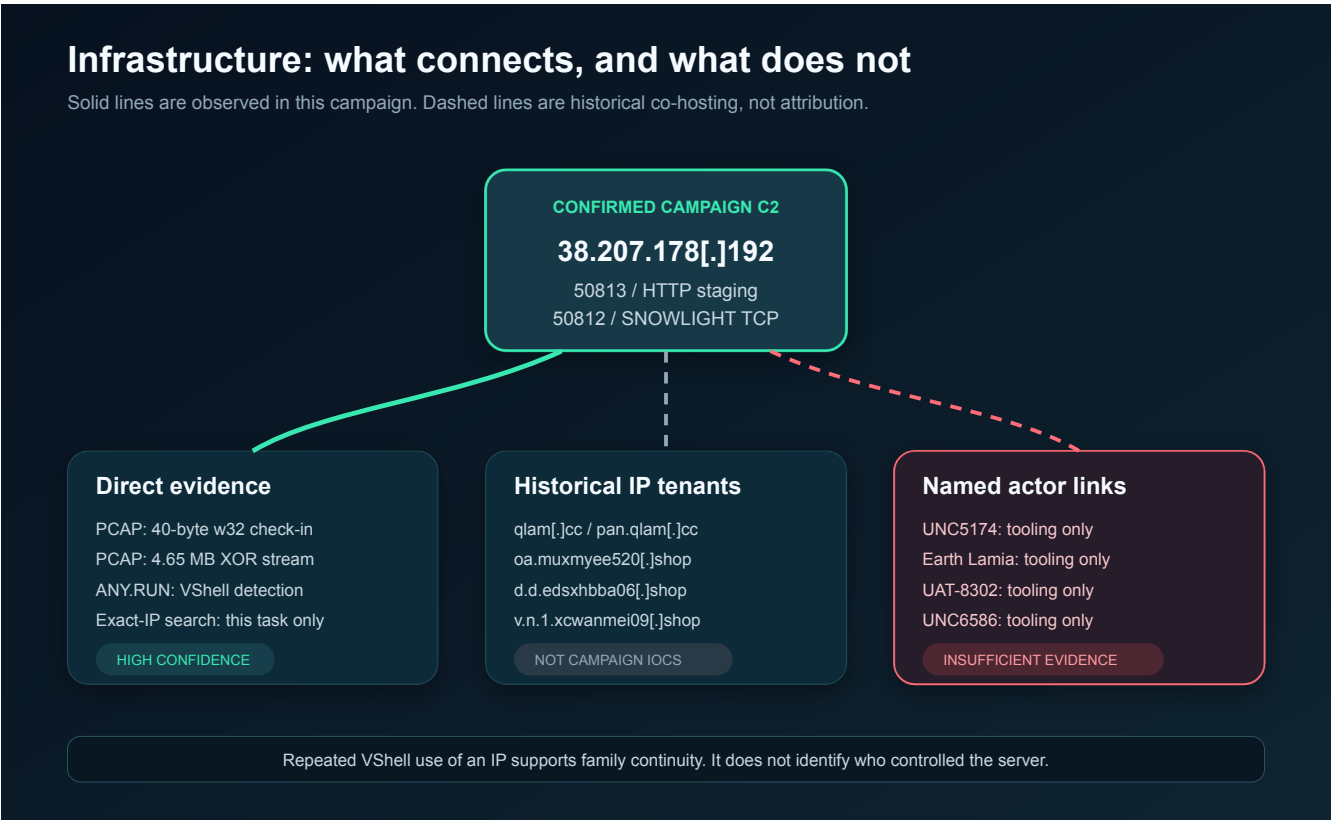
- A unique UNC5174 domain or known C2
- A matching cryptographic configuration
- A reused final-payload hash
- Operator logs or identity artifacts
- A campaign-specific overlap stronger than commodity tooling

The infrastructure adds one useful but limited fact. A ThreatFox-derived feed listed 38.207.178[.]192:40010 as VShell on 15 April 2026, months before this sample used 50812 and 50813. That supports repeated VShell use of the host. The original historical ThreatFox record is no longer available through the unauthenticated API, so I treat it as secondary-source evidence, not a campaign bridge.

Best label: **unattributed actor using a mainland-China-oriented academic lure and the leaked/commodity SNOWLIGHT and VShell ecosystem.**

Less exciting than naming an APT. More likely to survive contact with evidence.

infrastructure: one server, several stories



The exact campaign infrastructure is compact:

HIGH confidence	38.207.178[.]192:50813	HTTP staging and DOCX decoy
HIGH confidence	38.207.178[.]192:50812	SNOWLIGHT check-in and VShell transfer
MEDIUM confidence	38.207.178[.]192:40010	ThreatFox-derived VShell sighting, 2026-04-15

An ANY.RUN TI search for the exact destination IP across the available 180-day window returned one analysis: this one. It exposed the two staging URLs, three recorded connections and the VShell network alerts, but no second sample or campaign link.

Passive DNS tells a longer and mostly unrelated hosting story:

2024-01-15	v.n.1.xcwanmei09[.]shop
2024-05-03	d.d.edsxhbba06[.]shop
2024-05-06	oa.muxmyee520[.]shop
2024-06-04..2025-01-09	qlam[.]cc and www.qlam[.]cc
2024-12-20	pan.qlam[.]cc observed on the IP with its own TLS certificate
2026-04-15	secondary feed lists VShell on port 40010
2026-08-21	this sample uses ports 50812 and 50813

[OTX passive DNS](#) directly supports the old DNS resolutions. [urlscan independently observed](#) [pan.qlam.cc](#) serving nginx content from this IP in December 2024 with a certificate valid only

for that hostname. No named domain appears in the malware configuration or PCAP, the .shop hosts have no matching public urlscan results and no certificate, sample or URL joins them to VShell.

Classification: **historical tenants of the same IP, not campaign IOCs**. The same applies to unrelated domains on the subnet or ASN. Infrastructure graphing becomes fiction if every former neighbor is invited to the conspiracy.

confidence ledger

HIGH	This sample delivered SNOWLIGHT and VShell from 38.207.178[.]192.
HIGH	SNOWLIGHT is a standard VShell-generated stager, not actor-exclusive.
MODERATE	The lure was customized for a mainland Chinese academic context.
MODERATE	The IP hosted VShell before this run, based on a secondary feed.
INSUFFICIENT	The operator's language, nationality or physical location.
INSUFFICIENT	Any named actor, state sponsor or continuity with old DNS tenants.

detection and hunting

network

Block or investigate:

```
38.207.178[.]192:50812
38.207.178[.]192:50813
```

High-signal SNOWLIGHT check-in:

```
77 33 32 20 20 20 c6 7c 33 38 2e 32 30 37 2e 31
37 38 2e 31 39 32 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
```

This is w32 , network-order port 50812, the ASCII IP and zero-filled fields. Adapt the IP and port portions if hunting generically across generated SNOWLIGHT samples.

endpoint

Hunt for:

- %TEMP%\log_de.log
- Resume-themed executables launching cmd.exe and then Word
- AllocADsMem or ReallocADsMem followed by VirtualProtect and CreateThread
- Processes allocating tens of megabytes as executable memory immediately after a large raw TCP receive

- Go binaries with an invalid or hash-mismatched certificate and document-style names
- Outbound TCP beginning with six-byte architecture tags such as w32, w64, l32, l64, a32 or a64
- Post-stage TCP frames beginning with plausible little-endian lengths and direction-marked 12-byte AES-GCM nonces

The Kingsoft certificate attached to the first loader is invalid with HashMismatch. It is camouflage, not evidence that Kingsoft or WPS participated in the operation.

iocs

Original archive SHA-256

c25d4412f7f93e7de5b2aaf41747175d94cffd983ca20efbd0efcdd718b58c4d

Go loader SHA-256

81c51138d5527ca7dcc258171eb36659c479f66c86a8947b6340d040b3860a30

Go loader MD5

a7cc7e3cdd2f0f9210044911a483fa5d

Encrypted HTTP response SHA-256

f6d4da5afc89bf9e536a9002c4d256c696df2389d77279f4c5daf6979557e74e

SNOWLIGHT shellcode SHA-256

0524619d2471d77aba4b7993f5ffbaa4b8be6d2c0d91e63a02943851dc4b6404

XOR-encoded VShell stream SHA-256

ed2eaa6ef3eda95383b6efc35b88acbc742ad7bd118931f74727a3139ff7e97

Decoded VShell SHA-256

c666ac4f1a1b8df7ccfe8b19705279acd8b7eb7a4d0b3802bb3465064883ab25

DOCX decoy SHA-256

de3f56d0d5b71f2a1a1905f0b01b84fbab237e9d4c5179a05b127d806823f83c

C2 / staging IP	38.207.178[.]192
HTTP staging	38.207.178[.]192:50813
SNOWLIGHT TCP	38.207.178[.]192:50812
Loader AES key	YtWzxwZimsZoeMen
SNOWLIGHT XOR key	0x99
Kill-switch marker	%TEMP%\log_de.log

final thoughts

The interesting part of this sample is not that a resume delivered a RAT. We have been putting malware in job applications since recruiters first learned to double-click.

The interesting part is how cleanly the **behavior and lure answer different halves of the objective question.**

The code says the operator wanted an interactive foothold. It spends effort on sandbox detection, time checks, encrypted staging and fileless execution, then installs a framework built for shells, files, screens and tunnels.

The document says whose foothold might be useful: somebody reading graduate applications about AI, electrical systems, power grids and renewable energy, presented under the name of a university with real defence significance. Probably a professor or research lab. Not necessarily Beijing Institute of Technology and not necessarily a government target, but clearly more specific than “any Windows user with a mouse.”

That is how I would present the objective: not “VShell equals espionage,” and not “China lure equals APT.” Instead:

A targeted academic lure was used to establish stealthy VShell remote access, most likely enabling collection of research material and credentials and providing a pivot into the recipient institution. The exact operator and final motivation remain unconfirmed.

Precise enough to be useful. Honest enough not to become malware astrology.

If you are a professor reading unsolicited resumes: the applicant may be excellent, the research statement may be inspiring and the file extension should still be visible.

Stay safe. And maybe do not let `personal_resume.exe` join the lab before the student does.

references

- [ANY.RUN public analysis](#)
- [ASPI China Defence Universities Tracker: Beijing Institute of Technology](#)
- [U.S. Commerce Department: 2020 Entity List additions](#)
- [Beijing Institute of Technology: research laboratories and centers](#)
- [China Education Examinations Authority: College English Test](#)
- [China Education Examinations Authority: National Computer Rank Examination](#)
- [Sysdig: UNC5174's evolution from SNOWLIGHT to VShell](#)

- [Sekoia: SNOWLIGHT configuration extraction](#)
- [HPE Threat Labs: Unmasking the SNOWLIGHT stager](#)
- [SOCRadar: Tracing SNOWLIGHT](#)
- [Esonhugh: reverse engineering the VShell client and protocol](#)
- [Trend Micro: Earth Lamia and why SNOWLIGHT is not attribution](#)
- [Cisco Talos: UAT-8302 and its SNOWLIGHT/VShell deployment](#)
- [Google Threat Intelligence: UNC6586 React2Shell activity](#)
- [OTX: passive DNS for 38.207.178.192](#)
- [urlscan: historical pan.qlam.cc result](#)

Read other posts