INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-05-08 14:00 UTC

PamDOORa: Commercial PAM Backdoor Sold on Russian Cybercrime Forum Targets Linux SSH Stack

THREAT CAMPAIGN | **HIGH** | CVSS 5.0

SCC Item ID	SCC-CAM-2026-0289
Type	Threat Campaign
Severity	HIGH
CVSS Base Score	5.0
Affected Products	Linux x86_64, OpenSSH, PAM (Pluggable Authentication Modules)
Published	2026-05-08T04:41:00
Discovery Source	Rss

Executive Summary

A commercially available Linux backdoor called PamDOORa is being sold on a Russian cybercrime forum. It embeds inside the Linux authentication layer to silently capture plaintext passwords during SSH logins and includes anti-forensic routines capable of suppressing audit logs. No confirmed deployments have been identified yet, but the commercial model places this capability in the hands of less sophisticated attackers, increasing the probability of broad targeting across Linux server environments.

Technical Analysis

PamDOORa is a PAM-based implant targeting Linux x86_64 systems. It hooks into the Pluggable Authentication Modules (PAM) stack, specifically the authentication pipeline used by OpenSSH, to intercept and harvest plaintext credentials at the point of authentication, before encryption is applied. The toolkit includes a builder pipeline that simplifies deployment and anti-forensic routines that tamper with system logs (wtmp/btmp and auth logs) to erase evidence of access. No CVE has been assigned; this is a malicious implant, not a software vulnerability. CWE mapping: CWE-287 (improper authentication), CWE-522 (insufficiently protected credentials), CWE-506 (embedded malicious code). MITRE ATT&CK techniques include T1556.003 (Modify Authentication Process: Pluggable Authentication Modules), T1003 (OS Credential Dumping), T1070.002 and T1070.003 (Indicator Removal: Clear Linux Logs / Clear Command History), T1098 (Account Manipulation), T1078 (Valid Accounts), T1059.004 (Command and Scripting Interpreter: Unix Shell), and T1543 (Create or Modify System Process). Patch status: no vendor patch exists, as this is a malicious implant. Remediation requires detection and removal of the malicious PAM module. Source: The Hacker News (T3), Unit 42 PAM

abuse research (T3).

Action Checklist

1. Containment: Isolate any Linux x86_64 system showing unexpected PAM module entries until reviewed. Audit all Linux x86_64 systems running OpenSSH and PAM, particularly internet-facing servers. Verify the integrity of PAM configuration files (/etc/pam.conf, /etc/pam.d/*) and loaded PAM modules against a known-good baseline.
2. Detection: Check for unauthorized or unfamiliar .so files loaded in PAM configuration (grep for unexpected entries in /etc/pam.d/sshd). Review /var/log/auth.log, /var/log/secure, wtmp, and btmp for gaps, truncations, or anomalous timestamps indicating log tampering. Use file integrity monitoring (e.g., AIDE, Tripwire) to identify unexpected changes to PAM module files under /lib/security/ or /lib64/security/. Hunt for T1556.003 behaviors: unexpected shared library loads during authentication events.
3. Eradication: Remove any identified malicious PAM module files from /lib/security/ or /lib64/security/. Restore PAM configuration to a verified-clean state from a trusted backup or by reinstalling the pam package from your distribution's official repository (e.g., apt reinstall libpam-modules on Debian/Ubuntu; dnf reinstall pam on RHEL/Fedora). Rotate all SSH credentials on affected systems immediately, as harvested credentials must be treated as compromised.
4. Recovery: After reinstalling PAM, re-run file integrity checks against all authentication-related binaries and configuration files. Re-enable and verify log integrity. Monitor authentication logs for anomalous login patterns, particularly successful logins from unusual source IPs using valid credentials, consistent with credential reuse from harvested passwords (T1078). Confirm wtmp/btmp and auth logs are writing correctly.
5. Post-Incident: This threat exposes gaps in PAM module integrity monitoring, privileged authentication layer visibility, and SSH access controls. Implement file integrity monitoring on PAM paths as a standing control. Enforce SSH key-based authentication and disable password-based SSH login where operationally feasible. Establish a baseline inventory of authorized PAM modules and alert on deviations. Review privileged access to Linux systems for excessive or unnecessary accounts.

IR / Forensic Enrichment

Triage Priority	URGENT
Escalation Criteria	Escalate immediately to CISO and legal/privacy counsel if any credential harvest file is recovered containing plaintext passwords, if successful logins from unknown IPs using valid credentials (T1078) are detected post-containment indicating active use of harvested passwords, or if affected systems process PII/PHI/PCI data triggering statutory breach notification obligations.

Recovery Notes	After PAM reinstallation, monitor `/var/log/auth.log` and `/var/log/secure` continuously for a minimum of 30 days for T1078 credential-reuse attempts using passwords harvested prior to detection, as PamDOORa's commercial distribution model means harvested credentials may be resold or reused on extended timelines. Confirm log integrity by verifying `auditd` and `rsyslog` are forwarding to an off-system destination before returning affected hosts to production — PamDOORa's log suppression capability means locally stored logs alone cannot be trusted as a complete record. All SSH key pairs and service account passwords on affected systems must be treated as permanently compromised regardless of whether a harvest file was recovered.
Forensic Artifacts	/lib/security/*.so and /lib64/security/*.so — the malicious PamDOORa shared library will appear here as an .so file not attributable to any installed package (verify with `rpm -qf ` or `dpkg -S `); its presence is the primary deployment indicator | /var/log/auth.log or /var/log/secure with time-gap analysis — PamDOORa actively suppresses audit logs during SSH authentication events; gaps or truncations in these files during periods of known SSH activity are a behavioral indicator of the module's log-hiding function | wtmp/btmp (readable via `last` and `lastb`) cross-referenced against auth.log — discrepancies between session records in wtmp and missing corresponding pam_unix authentication events in auth.log indicate PamDOORa's selective log suppression was active | Credential harvest staging file — PamDOORa writes captured plaintext SSH passwords to a file on the local filesystem, likely under /tmp/, /dev/shm/, or a hidden dot-prefixed path; recovery of this file provides direct evidence of compromised account scope | auditd SYSCALL records for write/open events on /lib/security/ and /lib64/security/ (key=pam_module_change) — if auditd was running at time of implantation, these records document the exact timestamp and process lineage (ppid, uid, exe) of the actor who dropped the malicious PamDOORa .so into the PAM module directory

Per-Action IR Details

Containment — Audit all Linux x86_64 systems running OpenSSH and PAM, particularly internet-facing servers. Verify the integrity of PAM configuration files (/etc/pam.conf, /etc/pam.d/*) and loaded PAM modules against a known-good baseline. Isolate any system showing unexpected PAM module entries until reviewed.

NIST Phase: Containment

Reference: NIST 800-61r3 §3.3 — Containment Strategy

Controls: NIST IR-4 (Incident Handling), NIST SI-7 (Software, Firmware, and Information Integrity), NIST CM-3 (Configuration Change Control), CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory), CIS 4.6 (Securely Manage Enterprise Assets and Software)

Compensating: Run: `for f in /etc/pam.d/*; do md5sum $f; done` and diff against a stored baseline (even a plain text file from a clean peer system). Use `lsof | grep /lib/security` or `lsof | grep /lib64/security` to enumerate any processes currently holding open handles to PAM .so files. Network-isolate suspect systems using host-based iptables rules (`iptables -I INPUT -j DROP; iptables -I OUTPUT -j DROP`) rather than pulling the NIC, to preserve volatile state for analysis.

Evidence: Before isolating, capture: (1) output of `cat /etc/pam.d/sshd` and all files under `/etc/pam.d/` to document the live PAM stack configuration; (2) `ls -lahR /lib/security/ /lib64/security/` with timestamps to identify any .so files with unexpected modification dates or names inconsistent with distribution-packaged PAM modules; (3) `md5sum /lib/security/*.so /lib64/security/*.so` for comparison against known-good hashes from the distribution's package manager (`rpm -V pam` on RHEL or `debsums libpam-modules` on Debian/Ubuntu); (4) output of `ss -tnp` and `netstat -tnp` to document any active SSH sessions at time of containment.

Detection — Check for unauthorized or unfamiliar .so files loaded in PAM configuration (grep for unexpected entries in /etc/pam.d/sshd). Review /var/log/auth.log, /var/log/secure, wtmp, and btmp for gaps, truncations, or anomalous timestamps indicating log tampering. Use file integrity monitoring (e.g., AIDE, Tripwire) to identify

unexpected changes to PAM module files under /lib/security/ or /lib64/security/. Hunt for T1556.003 behaviors: unexpected shared library loads during authentication events.

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis

Controls: NIST SI-4 (System Monitoring), NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST AU-9 (Protection of Audit Information), NIST IR-5 (Incident Monitoring), CIS 8.2 (Collect Audit Logs), CIS 7.1 (Establish and Maintain a Vulnerability Management Process)

Compensating: Run `grep -v pam_unix\|pam_env\|pam_securetty\|pam_limits\|pam_deny\|pam_permit /etc/pam.d/sshd` to surface unfamiliar module references in the SSH PAM stack. Check for log suppression artifacts with: `stat /var/log/auth.log` and `wc -l /var/log/auth.log` then compare file sizes against a known-good peer; PamDOORa's audit log suppression would manifest as truncated or zero-byte growth periods correlating with active SSH login times. For shared library auditing without EDR, configure auditd with: `auditctl -w /lib/security/ -p wa -k pam_module_change` and `auditctl -w /lib64/security/ -p wa -k pam_module_change` — then review `/var/log/audit/audit.log` for SYSCALL records with key=pam_module_change. Use `ausearch -k pam_module_change` to query. For T1556.003 hunting, use `strace -e trace=open,openat -p $(pgrep sshd)` on a non-production clone to observe which .so files sshd loads during an authentication attempt.

Evidence: PamDOORa-specific forensic indicators to capture: (1) `/var/log/auth.log` or `/var/log/secure` — look for time-gap anomalies where SSH login events are absent during known active periods, or where `pam_unix(sshd:auth)` success/failure records are missing for sessions documented in `wtmp`; (2) `last -f /var/log/wtmp` and `lastb -f /var/log/btmp` — compare session counts against auth.log entries to identify suppressed authentication events, a hallmark of PamDOORa's log-hiding capability; (3) the exfiltration target file where PamDOORa writes harvested plaintext credentials — likely a hidden file under `/tmp/`, `/dev/shm/`, or a dot-prefixed path in a world-writable directory (run `find /tmp /dev/shm /var/tmp -name '.*' -o -newer /usr/sbin/sshd 2>/dev/null`); (4) `strings` output on any suspect .so file under `/lib/security/` — PamDOORa is a commercial tool and may contain hardcoded C2 hostnames, file paths, or credential staging paths identifiable via string analysis.

Eradication — Remove any identified malicious PAM module files from /lib/security/ or /lib64/security/. Restore PAM configuration to a verified-clean state from a trusted backup or by reinstalling the pam package from your distribution's official repository (e.g., apt reinstall libpam-modules on Debian/Ubuntu; dnf reinstall pam on RHEL/Fedora). Rotate all SSH credentials on affected systems immediately, as harvested credentials must be treated as compromised.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: NIST IR-4 (Incident Handling), NIST SI-2 (Flaw Remediation), NIST SI-3 (Malicious Code Protection), NIST IA-5 (Authenticator Management), CIS 7.2 (Establish and Maintain a Remediation Process), CIS 7.3 (Perform Automated Operating System Patch Management), CIS 5.2 (Use Unique Passwords)

Compensating: Before removing the malicious .so, preserve a forensic copy: `cp /lib/security/.so /root/ir-evidence/$(hostname)_$(date +%Y%m%d)_pam_malicious.so` and hash it (`sha256sum`). Then: `apt reinstall libpam-modules` (Debian/Ubuntu) or `dnf reinstall pam` (RHEL/Fedora) — this overwrites PAM module files from the signed distribution package. Verify post-reinstall with `debsums libpam-modules` or `rpm -V pam`. For SSH credential rotation on many systems without a PAM or secrets manager, use a bash loop over your inventory: `for host in $(cat hosts.txt); do ssh-copy-id -i ~/.ssh/newkey.pub user@$host; done`. Disable password auth immediately in `/etc/ssh/sshd_config` by setting `PasswordAuthentication no` and `ChallengeResponseAuthentication no`, then `systemctl restart sshd`.

Evidence: Before eradicating, capture: (1) a bit-for-bit copy of the malicious PAM .so file(s) for malware analysis — run `sha256sum`, `file`, and `strings` against the copy to extract hardcoded paths, C2 indicators, or credential staging locations specific to this PamDOORa build; (2) the credential harvest file (identified during detection) — its contents represent confirmed compromised accounts requiring immediate rotation; (3) `ldd .so` output to identify any additional shared library dependencies that may indicate further persistence mechanisms; (4) `/proc//maps` snapshot showing which memory-mapped libraries were loaded into the sshd process prior to eradication, to confirm the malicious module was actively loaded in memory.

Recovery — After reinstalling PAM, re-run file integrity checks against all authentication-related binaries and configuration files. Re-enable and verify log integrity. Monitor authentication logs for anomalous login patterns — particularly successful logins from unusual source IPs using valid credentials — consistent with credential reuse from harvested passwords (T1078). Confirm wtmp/btmp and auth logs are writing correctly.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery

Controls: NIST IR-4 (Incident Handling), NIST SI-7 (Software, Firmware, and Information Integrity), NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST AU-12 (Audit Record Generation), CIS 8.2 (Collect Audit Logs), CIS 6.3 (Require MFA for Externally-Exposed Applications)

Compensating: Post-reinstall integrity check: `debsums -c libpam-modules libpam-runtime` (Debian/Ubuntu) or `rpm -Va pam openssh-server` (RHEL/Fedora) — flag any files with checksum mismatches. Verify log writing is restored by tailing auth.log during a controlled test SSH login: `tail -f /var/log/auth.log` while running `ssh localhost` from a second terminal — a healthy system will immediately produce `pam_unix(sshd:auth): authentication failure` or `Accepted publickey` entries. For T1078 credential-reuse monitoring without SIEM, deploy a cron job that runs every 15 minutes: `awk '/Accepted password/{print $1,$2,$3,$9,$11}' /var/log/auth.log | sort | uniq -c | sort -rn > /root/ir-watch/ssh_logins_$(date +%H%M).txt` and diff against the prior run to surface new source IPs using valid credentials.

Evidence: Recovery validation artifacts to document: (1) output of post-reinstall integrity check (`debsums` or `rpm -Va`) showing clean state — this becomes the new baseline; (2) a 72-hour window of `/var/log/auth.log` and `/var/log/secure` entries post-recovery, preserving evidence of any T1078 credential-reuse attempts using passwords harvested by PamDOORa before detection; (3) `last -f /var/log/wtmp` output post-recovery to confirm new sessions are being correctly logged, confirming PamDOORa's log suppression mechanism is no longer active; (4) `systemctl status sshd` and `sshd -T` output confirming `PasswordAuthentication no` is active in the running configuration, not just the config file.

Post-Incident — This threat exposes gaps in PAM module integrity monitoring, privileged authentication layer visibility, and SSH access controls. Implement file integrity monitoring on PAM paths as a standing control. Enforce SSH key-based authentication and disable password-based SSH login where operationally feasible. Establish a baseline inventory of authorized PAM modules and alert on deviations. Review privileged access to Linux systems for excessive or unnecessary accounts.

NIST Phase: Post Incident

Reference: NIST 800-61r3 §4 — Post-Incident Activity

Controls: NIST IR-4 (Incident Handling), NIST IR-8 (Incident Response Plan), NIST SI-7 (Software, Firmware, and Information Integrity), NIST AU-2 (Event Logging), NIST IA-5 (Authenticator Management), NIST AC-2 (Account Management), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 5.4 (Restrict Administrator Privileges to Dedicated Administrator Accounts), CIS 6.5 (Require MFA for Administrative Access), CIS 8.2 (Collect Audit Logs)

Compensating: Deploy AIDE for standing PAM integrity monitoring: initialize a baseline with `aide --init` immediately after a confirmed-clean reinstall, then schedule `aide --check` via cron daily and alert on any changes to `/lib/security/`, `/lib64/security/`, `/etc/pam.d/`, and `/etc/pam.conf`. Write a YARA rule targeting PamDOORa's behavioral fingerprint — specifically, any .so file under `/lib/security/` that imports `pam_get_authtok` or `pam_sm_authenticate` but was not installed by the system package manager (cross-reference with `dpkg -S` or `rpm -qf`). For the authorized PAM module inventory, generate it once from a clean system: `for f in /lib/security/*.so /lib64/security/*.so; do echo "$(sha256sum $f) $(rpm -qf $f 2>/dev/null || dpkg -S $f 2>/dev/null)"; done > /root/pam_module_baseline.txt`. Store this baseline off-system.

Evidence: Post-incident documentation to retain: (1) the full lessons-learned report documenting the detection gap — specifically, the absence of standing integrity monitoring on `/lib/security/` that allowed PamDOORa to operate undetected, and the duration of that exposure window; (2) the credential harvest file contents (if recovered) cross-referenced against your IAM inventory to confirm scope of compromised accounts requiring rotation; (3) the final AIDE or `rpm -Va` baseline from confirmed-clean systems, stored in a read-only location as the authoritative reference

for future deviations; (4) threat intelligence submission — if a PamDOORa .so sample was recovered, submit to a private threat intel feed or internal TIP with the SHA-256 hash, `strings` output, and any recovered C2 indicators to enable organization-wide hunting across other Linux systems.

Detection Guidance

Primary detection target: unauthorized shared library entries in PAM configuration. Inspect /etc/pam.d/sshd and all files under /etc/pam.d/ for module references to .so files not present in your distribution's default package manifest. Verify loaded PAM modules: list files under /lib/security/ and /lib64/security/ and compare against known-good package content using package verification commands (dpkg --verify libpam-modules or rpm -V pam). Check for log tampering artifacts: missing or truncated entries in /var/log/auth.log or /var/log/secure, gaps in wtmp/btmp binary logs (use 'last' and 'lastb' to surface anomalies), and cleared bash/shell history files on SSH-accessible accounts (T1070.002, T1070.003). Behavioral indicators to hunt: authentication events that succeed without corresponding log entries; SSH sessions appearing in process tables without auth.log correlation; new or modified .so files in PAM module directories with recent modification timestamps. SIEM query focus: file write events to /etc/pam.d/*, /lib/security/*.so, /lib64/security/*.so via auditd (if enabled with appropriate -w rules). Auditd rules: add watches on PAM configuration directories and module paths if not already in your audit policy.

Framework Mappings

MITRE-ATTACK

- **T1003** — OS Credential Dumping
- **T1098** — Account Manipulation
- **T1505.003** — Web Shell
- **T1078** — Valid Accounts
- **T1059.004** — Unix Shell
- **T1543** — Create or Modify System Process
- **T1070.002** — Clear Linux or Mac System Logs
- **T1070.003** — Clear Command History
- **T1556.003** — Pluggable Authentication Modules

NIST-800-53R5

- **AC-6** — Least Privilege
- **IA-5** — Authenticator Management
- **SI-4** — System Monitoring
- **CM-2** — Baseline Configuration
- **CM-7** — Least Functionality
- **SI-3** — Malicious Code Protection
- **SI-7** — Software, Firmware, and Information Integrity
- **AC-2** — Account Management
- **IA-2** — Identification and Authentication (Organizational Users)

- **IA-8** — Identification and Authentication (Non-Organizational Users)

OWASP-TOP10-2021

- **A04:2021** — Insecure Design
- **A07:2021** — Identification and Authentication Failures

CIS-V8

- **5.2** — Use Unique Passwords
- **6.3** — Require MFA for Externally-Exposed Applications
- **6.4** — Require MFA for Remote Network Access
- **6.5** — Require MFA for Administrative Access
- **8.2** — Collect Audit Logs

HIPAA-SECURITY

- **164.308(a)(5)(ii)(D)** — Password Management
- **164.312(d)** — Person or Entity Authentication

SOC2-TSC

- **CC6.1** — The entity implements logical access security software, infrastructure, and architectures over protected information assets
- **CC6.3** — Authorizes, modifies, or removes access

ISO-27001-2022

- **A.5.34** — Privacy and protection of personal information

NIST-CSF-2

- **DE.CM-01** — Networks and network services are monitored
- **DE.AE-08** — Incidents are declared when adverse events meet the defined incident criteria

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1003	OS Credential Dumping	Credential-Access
T1098	Account Manipulation	Persistence
T1505.003	Web Shell	Persistence
T1078	Valid Accounts	Defense-Evasion
T1059.004	Unix Shell	Execution
T1543	Create or Modify System Process	Persistence
T1070.002	Clear Linux or Mac System Logs	Defense-Evasion

Technique ID	Technique Name	Tactic
T1070.003	Clear Command History	Defense-Evasion
T1556.003	Pluggable Authentication Modules	Credential-Access

Sources

Source	URL	Tier
Security News	https://thehackernews.com/2026/05/new-linux-pamdoora-backdoor-uses-...	T3
Critical Vulnerabilities in Linux Pluggable Authentication Modules ...	https://knowledge.broadcom.com/external/article/388169/critical-vul...	T3
The Linux Security Journey — PAM (Pluggable Authentication ...	https://medium.com/@boutnaru/the-linux-security-journey-pam-pluggab...	T3
An introduction to Pluggable Authentication Modules (PAM) in Linux	https://www.redhat.com/en/blog/pluggable-authentication-modules-pam	T3
When PAM Goes Rogue: Malware Uses Authentication Modules for ...	https://unit42.paloaltonetworks.com/linux-pam-apis/	T3

Generated 2026-05-08 14:00 UTC by TJS Security Command Center