FUZZRAIDERS

Penetration Test Simulation Project

FR-PR-2026-003 | Nocturnal – Full Attack Chain (Public / Sanitized)

Date: 2026-02-26

Environment: Hack The Box (Retired machine – authorized training lab)

Prepared by: FuzzRaiders Team

Classification: PUBLIC (Sanitized – no payloads, credentials, session tokens, or flags)

Document version	**V1.0 (public version)**
Report code	FR-PR-2026-003
Target	Nocturnal (hostname: nocturnal.htb)
Assessment type	External web app + host compromise simulation
Team roles	Proj Lead: Z4B0, Support: Mysto
Notes	Public report excludes step-by-step exploit payloads; focuses on endpoints, impact, and remediation.

Table of Contents

1.0 Engagement information

1.1 Objective

Simulate a realistic end-to-end penetration test against an authorized lab target to evaluate security weaknesses, validate exploitability, and produce a professional report with clear impact and remediation guidance (portfolio/public version).

1.2 Rules of engagement

• Authorized lab environment only (HTB retired machine).
• No denial-of-service testing.
• No third-party targeting.
• Public report is sanitized: no flags, credentials, session tokens, or copy-paste exploit payloads.

1.3 Scope and limitations

Scope focused on the exposed web application, authentication/session behavior, document handling functionality, and post-compromise host escalation paths. The assessment does not represent a full enterprise review (e.g., codebase review, threat modeling, or multi-host lateral movement).

1.4 Risk rating methodology

Severity is assigned using a practical risk model based on: (1) Impact, (2) Likelihood, and (3) Exploit chaining potential.
Critical = full compromise likely; High = major unauthorized access or foothold; Medium = limited impact; Low = hard to exploit or low business impact.

Critical	System takeover	High	RCE/root/admin
High	Major data exposure/foothold	Medium-High	IDOR + auth compromise/SSH access
Medium	Limited exposure	Medium	Sensitive info leak with constraints
Low	Minor issue	Low	Hard to exploit / low value

2.0 High-level summary

2.1 Executive summary

A complete attack chain was achieved. The compromise began with Broken Access Control (IDOR) in document viewing. Unauthorized document access enabled discovery of sensitive information, which was chained into a remote execution foothold. That foothold enabled authenticated SSH access and culminated in privilege escalation via a vulnerable local management service.

2.2 Attack chain overview (sanitized)

Phase	Result
Recon & mapping	Identified web app + SSH; mapped document viewer endpoint.
Access control testing	Confirmed IDOR on GET /view.php using username + file parameters.
Foothold	Validated unsafe server-side file handling leading to code execution (payload redacted).
Credential exposure	Sensitive data exposure enabled authenticated SSH access (creds redacted).
Post-exploitation	Enumerated local services; identified admin panel bound to localhost:8080.
Privilege escalation	Exploited vulnerable management component to obtain root (mechanics redacted).

2.3 Findings summary

ID	Finding	Severity	Affected endpoint/component	Primary impact
FR-F-01	Broken access control (IDOR) in document viewing	High	GET /view.php	Unauthorized document access
FR-F-02	Unsafe file handling enabling RCE chain	Critical	GET /view.php	Remote code execution (web)
FR-F-03	Credential exposure enabling SSH access	High	SSH (tcp/22)	Host foothold
FR-F-04	Privilege escalation via local management interface	Critical	http://127.0.0.1:8080	Root compromise

3.0 Nocturnal (nocturnal.htb) – technical walkthrough (sanitized)

3.1 Proof files

Public version: Proof artifacts (user/root flags) and credentials are intentionally redacted. Private evidence package contains full verification screenshots.

3.2 Pre-compromise enumeration steps

Network discovery identified an HTTP service and SSH. Web application mapping discovered a document viewer endpoint with query parameters used to select a user context and a file. Testing focused on authorization enforcement, input validation, and file handling behavior.

Techniques used (sanitized):
• TCP port/service discovery
• Web content discovery and endpoint mapping
• Authorization testing (IDOR patterns)
• Parameter tampering and response-differential analysis
• Controlled validation of server-side file processing behavior

3.3 Compromise (initial access)

Initial access was achieved by chaining weaknesses in the document viewer. The endpoint accepted a user context and file selector without enforcing strict authorization boundaries. This enabled access to documents outside of the authenticated user's scope and progression into unsafe server-side handling paths that resulted in code execution. Exact payloads are omitted in this public report.

Affected endpoint(s) and parameters

• GET /view.php

- username (query)

- file (query)

Validation steps (public/sanitized)

1) Authenticate as a normal user.
2) Request /view.php with your own username to observe allowed file listing behavior.
3) Modify the username value to another valid username and observe unauthorized file listings.
4) Attempt to retrieve an available document and confirm access is granted outside intended scope.
5) Validate that file processing logic contains unsafe server-side handling paths (details redacted).

3.4 Post-exploitation enumeration steps

After obtaining a stable foothold, post-exploitation focused on:
• Locating credential material and sensitive configuration data
• Enumerating running services and local-only listeners
• Identifying privilege escalation opportunities (misconfigurations, vulnerable admin components)
• Verifying path to administrative control in a controlled manner

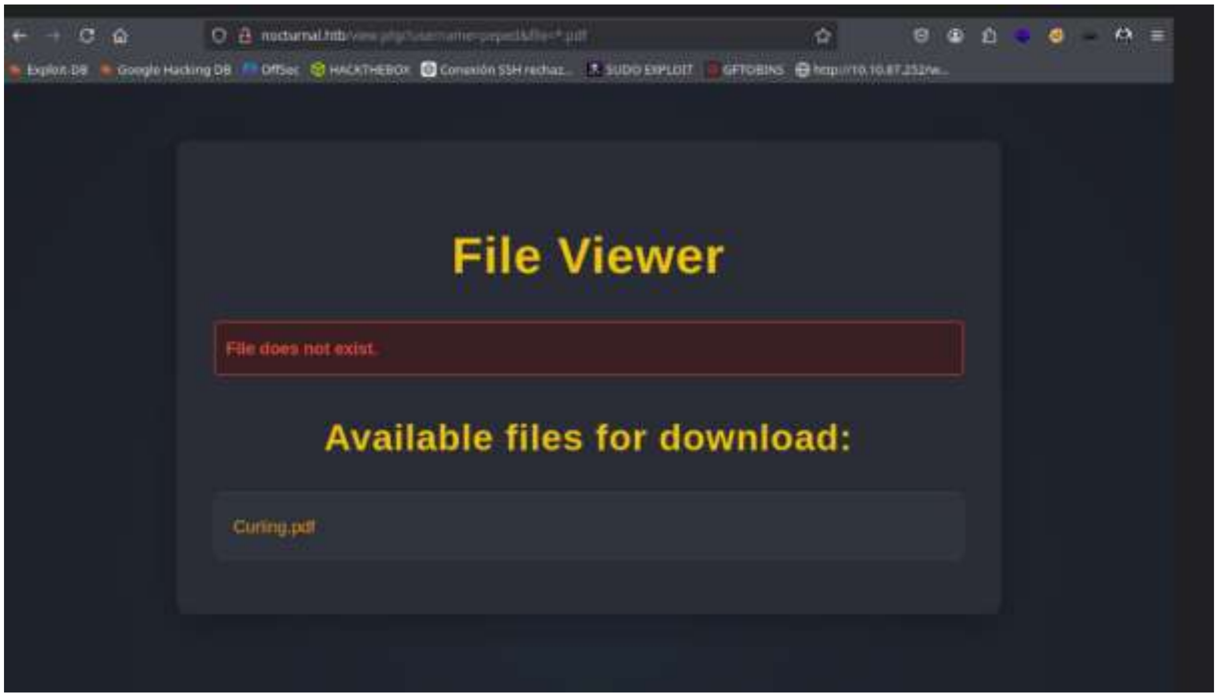
3.5 Local privilege escalation

A management interface bound to localhost (observed on http://127.0.0.1:8080) was identified. The component was vulnerable, allowing escalation from a standard user foothold to full root-level control. Exploitation mechanics are redacted for public release.

3.6 Screenshots (sanitized)

The following screenshots demonstrate key phases while removing sensitive values (tokens/passwords/flags).

Figure 1 – Document viewer behavior and available files (IDOR context).



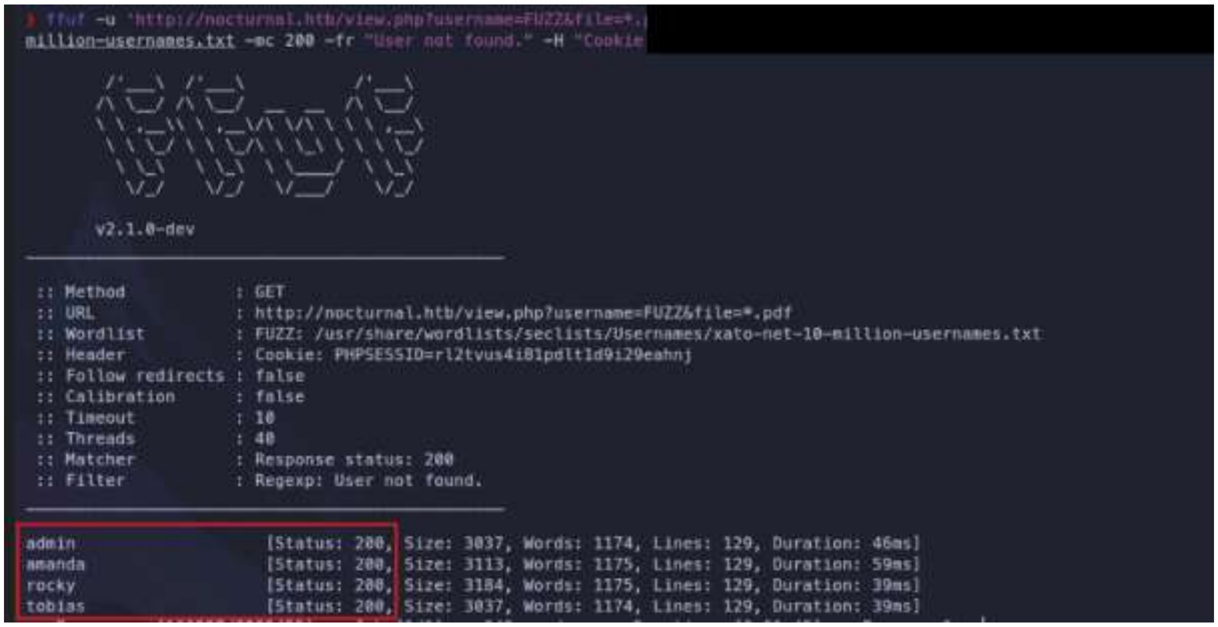
File Viewer

File does not exist.

Available files for download:

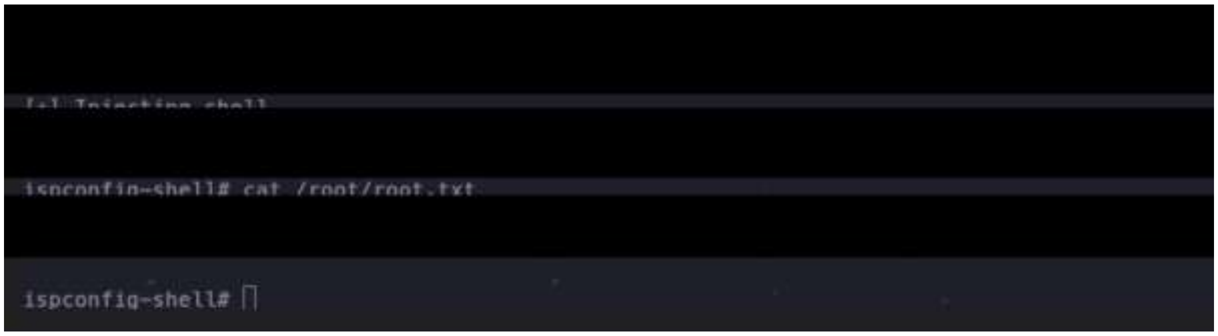
Curling.pdf

Figure 2 – Username enumeration via response-differential testing (session token redacted).



```
❯ ffuf -u 'http://nocturnal.htb/view.php?username=FUZZ&file=*.[redacted]
million-usernames.txt -mc 200 -fr "User not found." -H "Cookie

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v2.1.0-dev
________________________________________________

 :: Method           : GET
 :: URL              : http://nocturnal.htb/view.php?username=FUZZ&file=*.pdf
 :: Wordlist         : FUZZ: /usr/share/wordlists/seclists/Usernames/xato-net-10-million-usernames.txt
 :: Header           : Cookie: PHPSESSID=rl2tvus4i81pdlt1d9i29eahnj
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200
 :: Filter           : Regexp: User not found.
________________________________________________

admin                   [Status: 200, Size: 3037, Words: 1174, Lines: 129, Duration: 46ms]
amanda                  [Status: 200, Size: 3113, Words: 1175, Lines: 129, Duration: 59ms]
rocky                   [Status: 200, Size: 3184, Words: 1175, Lines: 129, Duration: 39ms]
tobias                  [Status: 200, Size: 3037, Words: 1174, Lines: 129, Duration: 39ms]
```

Figure 3 – Privilege escalation confirmation (sensitive values redacted).



```
[redacted]
[+] Injecting shell
[redacted]
ispconfig-shell# cat /root/root.txt
[redacted]
ispconfig-shell# 
```

4.0 Findings and remediation details

FR-F-01 – Broken access control (IDOR) in document viewing (High)

OWASP mapping: A01:2021 – Broken Access Control

CWE mapping: CWE-639 (Authorization Bypass Through User-Controlled Key)

Affected endpoint(s)/component(s): GET /view.php (username, file)

Description:

The application uses a user-controlled parameter (username) to determine which user context is used when listing or retrieving files. Server-side authorization checks are insufficient, allowing a logged-in user to request documents associated with other users. This is an Insecure Direct Object Reference (IDOR) pattern where the server trusts client-provided identity selectors.

Impact:

Unauthorized access to private documents. In a real environment, this can expose PII, internal documents, credentials, tokens, or other sensitive operational data. This finding also served as the entry point for the broader attack chain.

Likelihood: High (simple parameter manipulation; no special conditions beyond authentication).

Recommendations:

• Remove client-controlled identity selection. Derive user context from session/authentication state.

• Implement object-level authorization checks on every request (ownership/ACL validation).

• Use unpredictable identifiers for stored objects and validate ownership server-side.

• Add monitoring/rate-limiting for rapid enumeration of usernames/files.

Fix verification guidance:

• Attempt to request another user's documents; server must return 403/404 consistently.

• Review logs to ensure enumeration attempts are flagged/blocked.

• Perform automated access-control tests as part of CI/CD (DAST for IDOR patterns).

Evidence reference(s): EV-FR-F-01, EV-FR-F-03 (sanitized)

FR-F-02 – Unsafe server-side file handling enabling RCE chain (Critical)

OWASP mapping: A03:2021 – Injection / A05:2021 – Security Misconfiguration (context-dependent)

CWE mapping: CWE-78 (OS Command Injection) / CWE-20 (Improper Input Validation) (public classification)

Affected endpoint(s)/component(s): GET /view.php (file) and server-side document processing

Description:

The server processes user-selected files using insufficient input validation and unsafe handling logic. The weakness allows a malicious request to influence server-side processing paths, culminating in remote code execution under the web application context. Payload mechanics are intentionally omitted in this public report.

Impact:

Remote code execution enables full compromise of the application layer and can lead to system compromise depending on privileges, environment configuration, and exposure of internal services/secrets.

Likelihood: Medium-High (requires understanding of processing paths; exploit chaining demonstrated).

Recommendations:

• Eliminate any shell-based processing of user input; avoid invoking OS commands with user-controlled values.

• Implement strict allowlist validation and canonicalization for file names and paths.

• Store files outside web root; enforce safe content handling.

• Run document processing in a restricted worker/sandbox with minimal privileges and resource limits.

• Apply code review gates and security testing for file handling and command execution paths.

Fix verification guidance:

• Attempt previously successful malicious input patterns; server must reject and log them.

• Confirm file processing occurs without shell execution and under restricted permissions.

• Add unit tests for validation logic and negative test cases for bypass attempts.

Evidence reference(s): Private evidence pack (withheld)

FR-F-03 – Credential exposure enabling SSH access (High)

OWASP mapping: A07:2021 – Identification and Authentication Failures (context-dependent)

CWE mapping: CWE-200 (Exposure of Sensitive Information to an Unauthorized Actor)

Affected endpoint(s)/component(s): SSH (tcp/22) and credential material stored in accessible documents

Description:

Sensitive material retrieved through unauthorized document access provided valid SSH authentication details. Credentials are not disclosed in this public report. This reflects a common real-world failure mode: secrets stored in documents/backups/configs that become accessible due to an upstream access-control flaw.

Impact:

Authenticated SSH access provides stable host foothold, enabling deeper enumeration, persistence opportunities, and privilege escalation.

Likelihood: High (credential material was directly usable once obtained).

Recommendations:

• Remove secrets from documents; use a secrets manager and least-privilege access controls.

• Rotate any exposed credentials immediately; implement credential hygiene and auditing.

• Disable password SSH authentication where possible; enforce key-based auth and MFA.

• Monitor SSH authentication events and alert on anomalies.

Fix verification guidance:

• Run secret scanning across repos/storage and confirm no plaintext secrets remain.

• Attempt password-based SSH auth; confirm it is disabled or tightly controlled.

• Validate monitoring: ensure alerts trigger on abnormal SSH behavior.

Evidence reference(s): Private evidence pack only (public screenshots redacted)

FR-F-04 – Privilege escalation via vulnerable local management interface (Critical)

OWASP mapping: A06:2021 – Vulnerable and Outdated Components / A05:2021 – Security Misconfiguration

CWE mapping: CWE-1104 (Use of Unmaintained Third Party Components) (public classification)

Affected endpoint(s)/component(s): Local management service at http://127.0.0.1:8080

Description:

A locally accessible management component was identified during post-exploitation enumeration. The component contained a known-vulnerable condition that enabled escalation to root-level control. Exploit details are omitted in the public report to prevent copy-paste abuse.

Impact:

Root compromise results in total loss of confidentiality, integrity, and availability. This enables full persistence, data destruction/exfiltration, and onward attacks against connected infrastructure.

Likelihood: Medium (requires foothold + local access; exploitability demonstrated).

Recommendations:

• Patch/upgrade management components immediately; remove unused/admin software from production hosts.

• Restrict admin interfaces and enforce strong authentication and access controls.

• Apply OS hardening and least privilege; isolate services; reduce attack surface.

• Implement monitoring for admin interface access and suspicious post-auth actions.

Fix verification guidance:

• Verify component version is patched and vulnerable routes are no longer exploitable.

• Confirm localhost-only services cannot be reached without authenticated controls.

• Validate logging/alerting on admin interface authentication and configuration changes.

Evidence reference(s): EV-FR-F-05 (sanitized)

5.0 Detection and hardening recommendations

5.1 Logging and telemetry

Recommended telemetry sources:
• Web server access logs (URI, status, response size, auth context)
• Application audit logs (document access events with user + object IDs)
• SSH auth logs (successful/failed logins, source, user, key fingerprints)
• Process execution logs (child processes from web worker)
• Local service logs for management interface activity

5.2 Detection ideas

High-signal detections for this attack chain:
• Multiple /view.php requests across many usernames from one session/IP
• Requests with unusual 'file' values or rapid file enumeration patterns
• Sudden spikes in 200 responses for resources not owned by the user
• New SSH logins shortly after document access anomalies
• Port-forwarding activity combined with local admin panel access

5.3 Hardening checklist

• Enforce object-level authorization on all file/document resources
• Remove shell execution paths and unsafe file processing
• Patch and minimize admin components; reduce installed software
• Disable password SSH auth; enforce key-based auth; implement rate limiting
• Centralize logs and enable alerting for access-control abuse patterns

6.0 Remediation roadmap

Prioritization is based on exploit chaining risk and time-to-fix.

Timeline	Actions	Owner	Verification
0–7 days	Fix IDOR on /view.php; remove username param trust; add auth checks	App team	Attempt cross-user access → denied
0–7 days	Remove unsafe file processing paths; hotfix + monitoring	App team	Negative tests + logs
7–30 days	Rotate secrets; deploy secrets management; scan docs/repos for secrets	Ops/Sec	No secrets found; rotation complete
7–30 days	Patch/upgrade management interface; restrict access	Ops/Sec	Version verified; access controls validated
30–90 days	Implement SDLC controls: code review gates, SAST/DAST, secret scanning	Engineering leadership	Pipeline evidence + metrics

7.0 Appendices

Appendix A – Tools used (high level)

• Network scanning: Nmap
• Web enumeration: ffuf / manual mapping
• Interception/testing: browser + proxy tooling
• Host access: SSH
• Local enumeration: standard Linux enumeration
• Exploit research: public advisories (details withheld in public report)

Appendix B – Evidence index (public)

Evidence ID	Description	Public status
EV-FR-F-01	Document viewer listing behavior (IDOR context).	Included
EV-FR-F-03	Username enumeration output (token redacted).	Included (sanitized)
EV-FR-F-04	SSH access proof (password/flag redacted).	Excluded (sensitive)
EV-FR-F-05	Privilege escalation proof (password/flag redacted).	Included (sanitized)

Appendix C – Glossary

IDOR:	Insecure Direct Object Reference (broken object-level authorization).
Foothold:	Initial stable access on target.
RCE:	Remote Code Execution.
Privilege escalation:	Moving from low privileges to admin/root.
Sanitized report:	Public version with sensitive details removed.