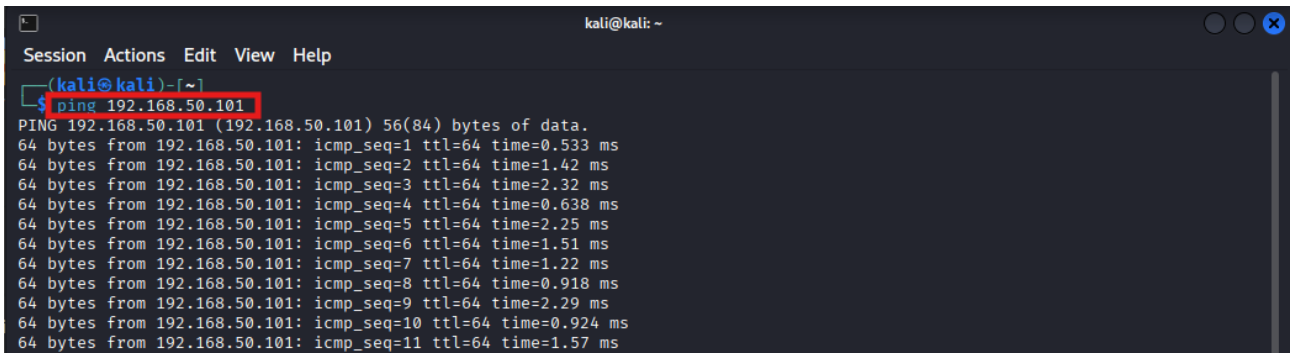
S6 – L1 EXPLOIT FILE UPLOAD

1) Configurazione laboratorio e verifica rete

Figura 1 — Ping Kali → Metasploitable

Cosa ho fatto

- Da terminale su **Kali**, hai eseguito:
    - `ping 192.168.50.101`
- Le risposte ICMP confermano che **Metasploitable è raggiungibile** dalla Kali, come richiesto nella preparazione dell'ambiente.



---

2) DVWA: impostazione sicurezza a LOW (richiesta nei suggerimenti)

Figura 2 — DVWA Security Level = LOW

Cosa ho fatto

- In DVWA sono entrato nella pagina **DVWA Security**.
- Ho selezionato **low** e confermato con **Submit**.
- In basso compare il messaggio "Security level set to low": evidenza che DVWA è configurata come suggerito prima di procedere all'exploit.

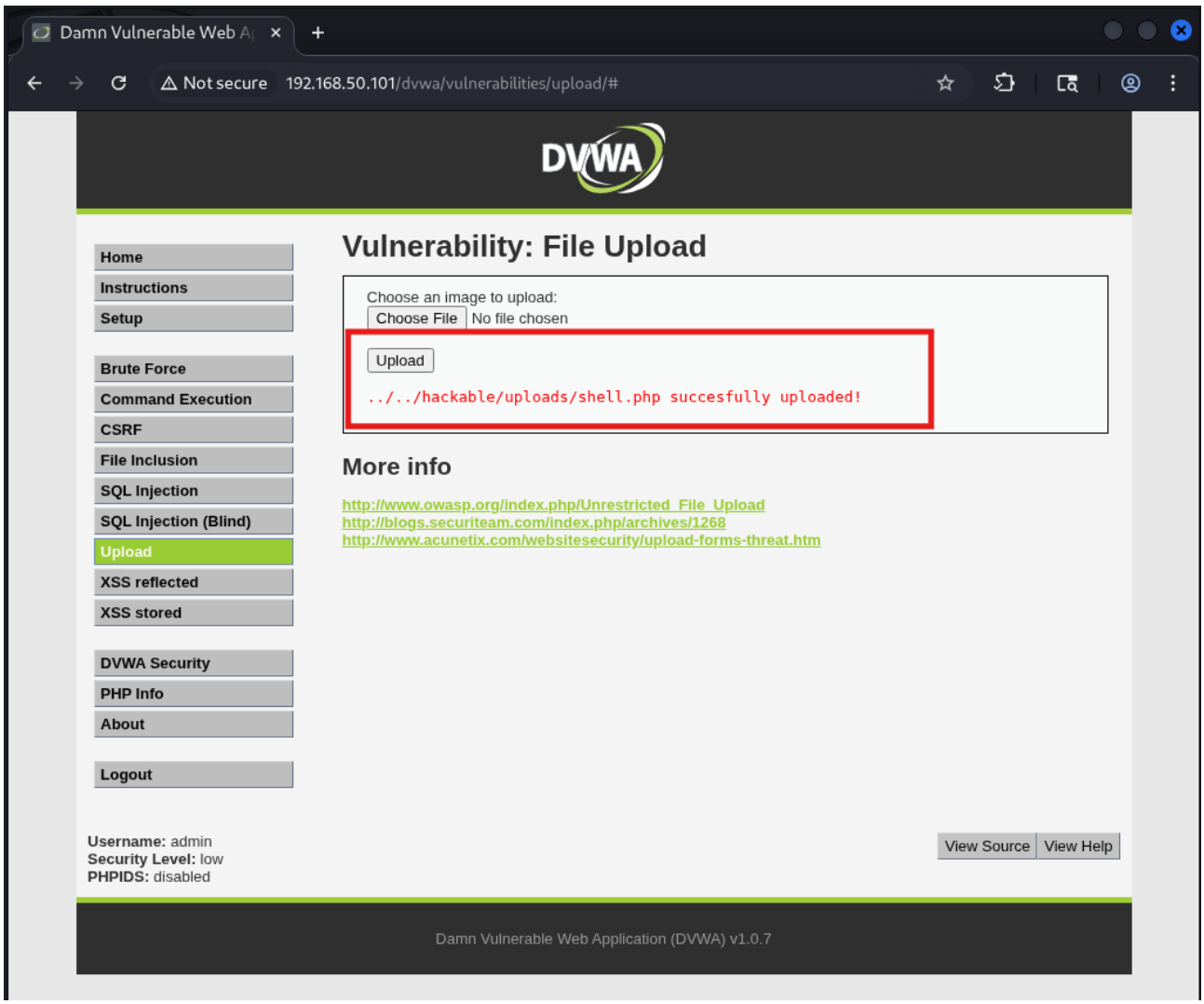
---

3) Exploit: File Upload della shell PHP

Figura 3 — Upload riuscito di `shell.php`

Cosa ho fatto

- Sono andato su **Vulnerability: File Upload** (sezione Upload).
- Ho caricato la tua shell PHP (`shell.php`).
- DVWA conferma l'upload con messaggio rosso:
  `../../hackable/uploads/shell.php successfully uploaded!`
  Questa è la prova richiesta "Risultato del caricamento (screenshot del browser)".



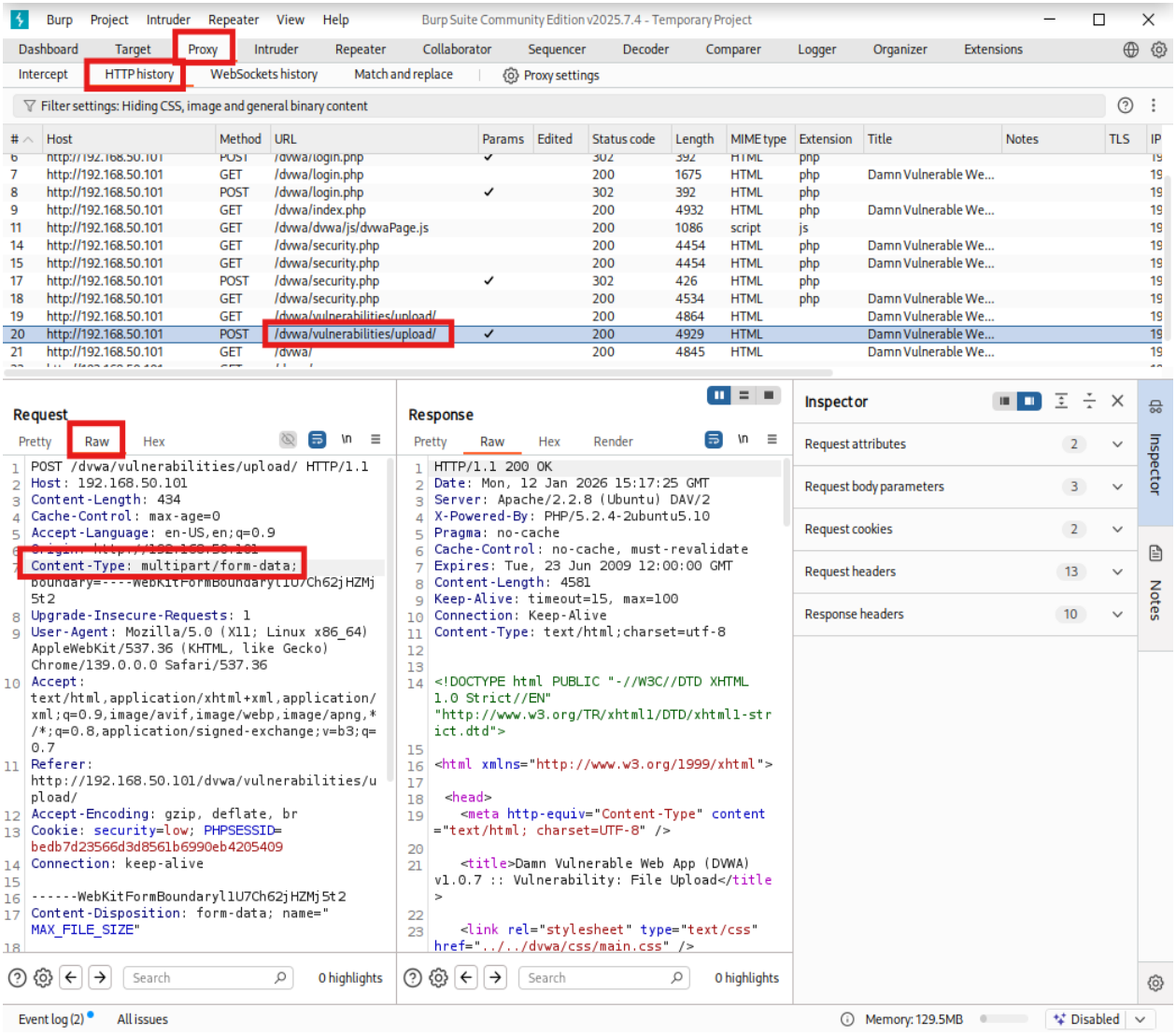
3.1) Intercettazione con BurpSuite durante l’upload (obbligatorio in consegna)

Figura 4 — BurpSuite: richiesta POST di upload (multipart/form-data)

Cosa hai fatto

- Ho tenuto BurpSuite aperto e ho intercettato/registrato la richiesta verso DVWA durante l’upload.
- Nella request si vede:
  - metodo **POST**
  - endpoint `/dvwa/vulnerabilities/upload/`

- header chiave: `Content-Type: multipart/form-data`
  Questo dimostra tecnicamente che l'upload passa come form multipart e soddisfa la consegna "Intercettazioni (screenshot di burpsuite)".



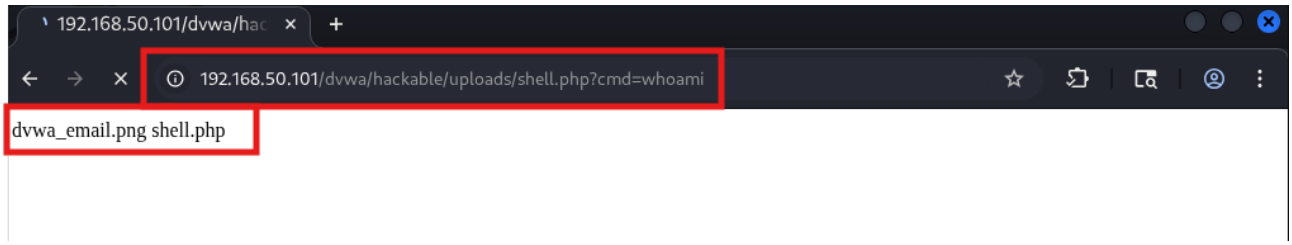
Durante il caricamento della shell PHP ho intercettato con BurpSuite la richiesta HTTP POST verso la pagina di upload. La richiesta utilizza `multipart/form-data`, confermando l'invio del file tramite form web e permettendo l'analisi di header/parametri e risposta del server.

4) Accesso alla shell caricata via browser (inizio controllo remoto)

Figura 5 — Apertura shell via URL con parametro `cmd`

Cosa ho fatto

- Ho aperto nel browser il file caricato nella cartella `hackable/uploads`:
  - `.../dvwa/hackable/uploads/shell.php?cmd=whoami`
- Questa è la fase richiesta “Esecuzione della shell via browser” e mostra chiaramente **come passi `cmd` nella GET**

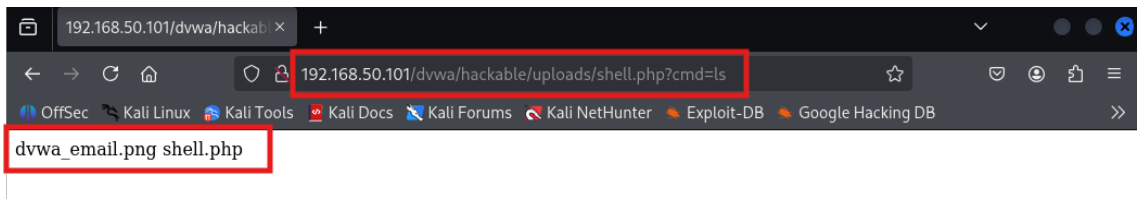


⚠ Nota: in questo screenshot l’output visualizzato è una lista file (non “www-data”). Non è un problema, perché poi lo dimostro correttamente nelle figure successive: significa solo che stavo facendo test multipli (es. `ls` e poi `whoami`).

Figura 6 — Esecuzione comando remoto: `ls` (output in pagina)

Cosa ho fatto

- Ho usato la shell per eseguire **comandi remoti** passando `cmd` via GET.
- Con `cmd=ls` ottengo l’output della directory, dove si vedono:
  - `dvwa_email.png`
  - `shell.php`
    Questa è la prova “controllo remoto tramite shell”, richiesta dalla traccia.

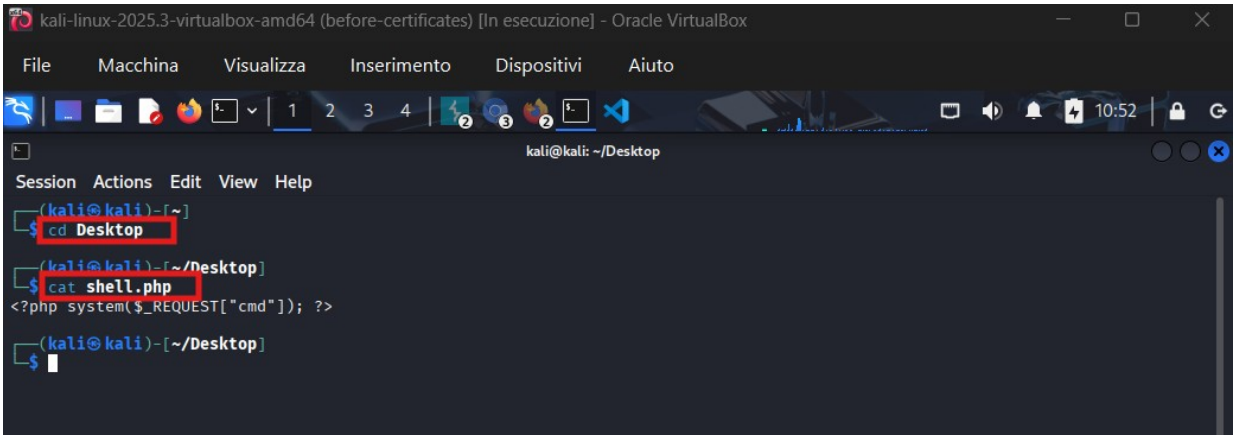


5) Evidenza del codice PHP (obbligatorio)

Figura 7 — Codice della shell (`cat shell.php`)

Cosa ho fatto

- Da terminale su Kali sono entrato nella cartella (Desktop) e ho mostrato il contenuto:
    - `cat shell.php`
- Il codice è minimale e usa `$_REQUEST["cmd"]` per eseguire il comando.
  Questo soddisfa la consegna “Codice php”.



---

6) Risultato delle varie richieste

Figura 8 — `whoami` → utente `www-data`

Cosa ho fatto

- Ho eseguito:
    - `cmd=whoami`
- L’output è `www-data`, quindi confermo che il comando gira nel contesto del web server (Apache/PHP).
  Questa riga è anche ottima come “commento di sicurezza”: accesso remoto ottenuto, ma con privilegi iniziali limitati.

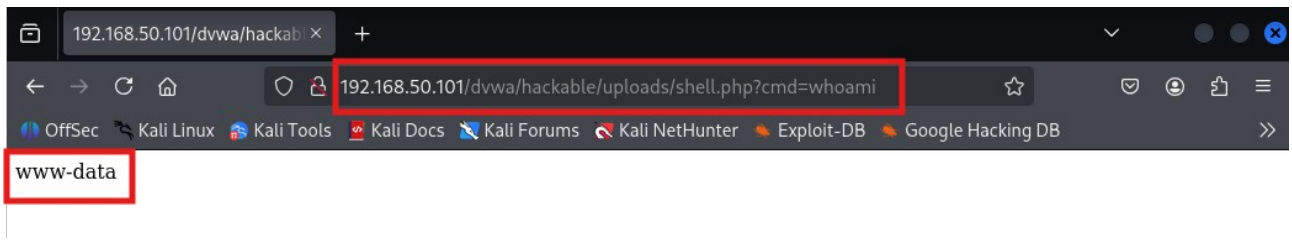
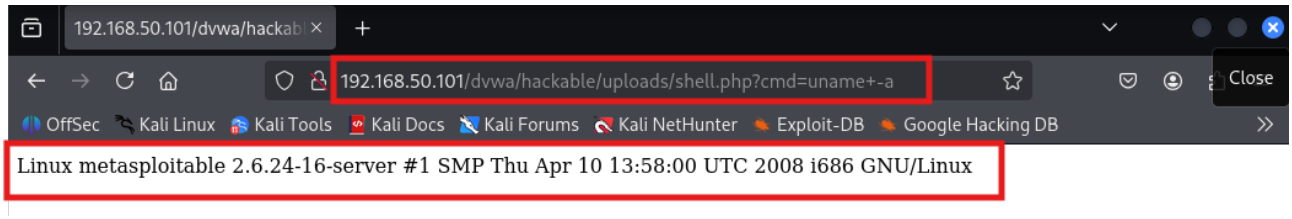


Figura 9 — `uname -a` → informazioni sistema

Cosa ho fatto

- Ho eseguito:
    - `cmd=uname -a`
- Ottieni dettagli su kernel/OS (“Linux metasploitable … 2.6.24… i686 …”).
  Questo copre perfettamente la consegna “Eventuali altre informazioni scoperte della macchina interna”.



---

7) Verifica finale della comunicazione bidirezionale

Figura 10 — Ping Metasploitable → Kali (comunicazione bidirezionale)

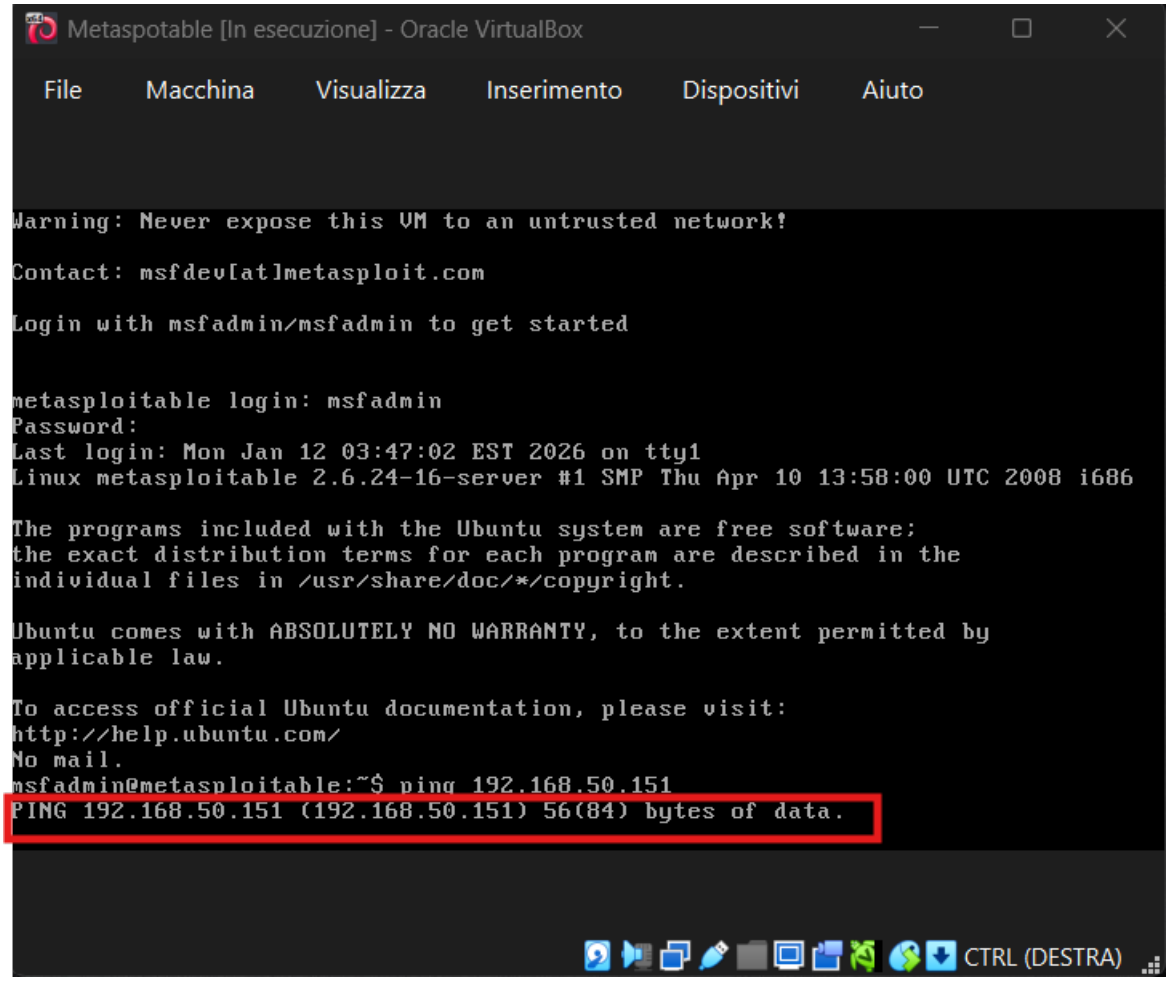
Cosa ho fatto

- Dopo aver completato lo sfruttamento della vulnerabilità, ho eseguito una verifica finale della rete direttamente dalla macchina **Metasploitable**.
- Effettuato il login sulla console di Metasploitable, ho eseguito il comando:

`ping 192.168.50.151`

(dove `192.168.50.151` è l’indirizzo IP della macchina Kali).

- Il comando ha restituito risposte ICMP corrette, confermando che **Metasploitable è in grado di raggiungere Kali**.



Questo passaggio dimostra in modo definitivo la **comunicazione bidirezionale tra le due macchine**, confermando la corretta configurazione dell’ambiente di laboratorio e la piena connettività necessaria per l’esecuzione dell’esercizio.

La verifica della comunicazione bidirezionale è stata eseguita sia prima che dopo lo sfruttamento, a conferma della stabilità della rete del laboratorio durante tutte le fasi dell’attività.

Conclusione

Ho configurato il laboratorio assicurando la raggiungibilità e la **comunicazione bidirezionale** tra Kali Linux e Metasploitable, verificata tramite test di ping in entrambe le direzioni.

Ho impostato DVWA a **Security Level "LOW"** e ho sfruttato la vulnerabilità di **File Upload** caricando una shell PHP.

Accedendo al file caricato, ho eseguito comandi remoti tramite parametro GET `cmd` (es. `ls`, `whoami`, `uname -a`), ottenendo prova di controllo remoto e raccogliendo informazioni sul sistema target.

Durante le attività ho intercettato le richieste HTTP con **BurpSuite**, analizzando la richiesta **POST** di upload del file e la **GET** utilizzata per l'esecuzione dei comandi.