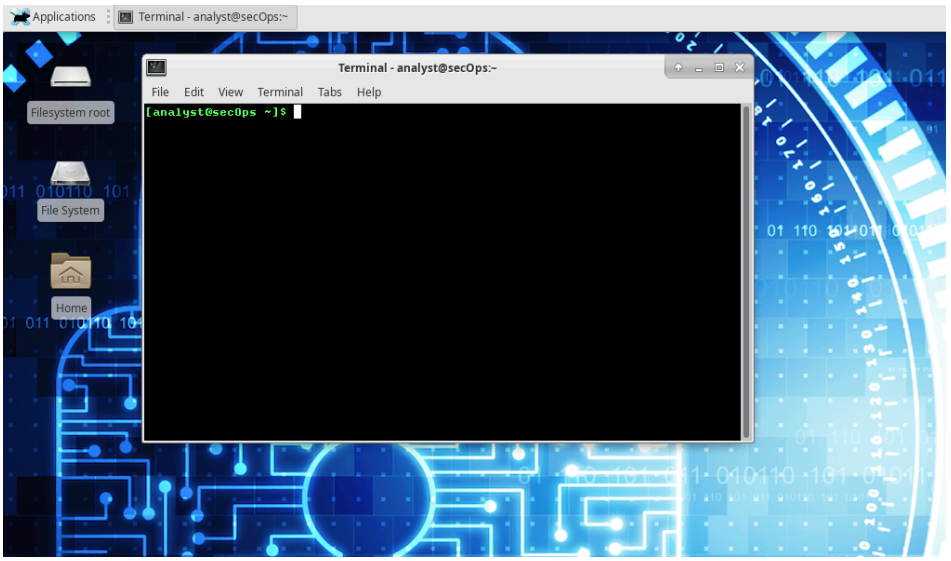
Esercizio 2: Server Linux

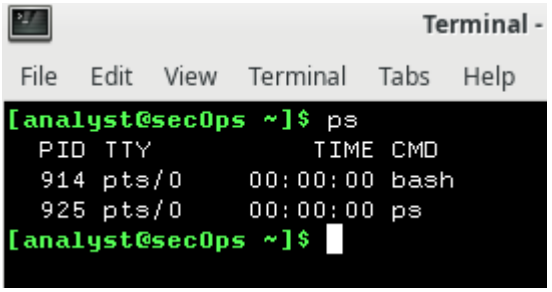
In questo esercizio l'obiettivo principale è comprendere e analizzare dei servizi e dei processi in esecuzione su un sistema Linux, vedremo processi che agiscono in **background** e cercheremo di comprenderne la **funzione.**

Parte 1: Server

Come primo passaggio andiamo ad effettuare l'accesso alla nostra VM (**CyberOps Workstation**) con le credenziali di **analyst**, usando la password **cyberops**. Una volta effettuato l'accesso apriamo il terminale.



Per visualizzare i processi, utilizziamo il comando **ps.**



Tuttavia alcuni processi non vengono visualizzati senza **permessi elevati**. Dunque, per rispondere alla domanda "**Perché è stato necessario eseguire ps come root (premettendo il comando con sudo)?",** lo usiamo perché alcuni processi non appartengono all'utente con la quale abbiamo fatto l'accesso, **ovvero analista**, e potrebbero non essere visualizzati se ps è stato eseguito da un utente **"normale"**

Il comando ps può essere utilizzato anche per visualizzare la gerarchia di processi. Tramite la sintassi **–ejH**, si può visualizzare l'albero dei processi attualmente in esecuzione, dopo aver avviato il **server web nginx** con privilegi elevati.

```
[analyst@secOps ~]$ sudo ps -ejH
  PID  PGID    SID TTY          TIME CMD
    2     0      0 ?        00:00:00 kthreadd
    4     0      0 ?        00:00:00   kworker/0:0H
    5     0      0 ?        00:00:00   kworker/u2:0
    6     0      0 ?        00:00:00   mm_percpu_wq
    7     0      0 ?        00:00:00   ksoftirqd/0
    8     0      0 ?        00:00:00   rcu_preempt
    9     0      0 ?        00:00:00   rcu_sched
   10     0      0 ?        00:00:00   rcu_bh
   11     0      0 ?        00:00:00   rcuc/0
   12     0      0 ?        00:00:00   rcub/0
   13     0      0 ?        00:00:00   migration/0
   14     0      0 ?        00:00:00   watchdog/0
   15     0      0 ?        00:00:00   cpuhp/0
   16     0      0 ?        00:00:00   kdevtmpfs
   17     0      0 ?        00:00:00   netns
   18     0      0 ?        00:00:00   rcu_tasks_kthre
   19     0      0 ?        00:00:00   kworker/0:1
   20     0      0 ?        00:00:00   khungtaskd
   21     0      0 ?        00:00:00   oom_reaper
   22     0      0 ?        00:00:00   writeback
   23     0      0 ?        00:00:00   kcompactd0
   24     0      0 ?        00:00:00   ksmd
   25     0      0 ?        00:00:00   khugepaged
   26     0      0 ?        00:00:00   crypto
   27     0      0 ?        00:00:00   kintegrityd
   28     0      0 ?        00:00:00   kblockd
   29     0      0 ?        00:00:00   edac-poller
   30     0      0 ?        00:00:00   devfreq_wq
   31     0      0 ?        00:00:00   watchdogd
   32     0      0 ?        00:00:00   kworker/u2:1
   33     0      0 ?        00:00:00   kswapd0
   72     0      0 ?        00:00:00   kthrotld
   73     0      0 ?        00:00:00   acpi_thermal_pm
   74     0      0 ?        00:00:00   nvme-wq
```

La gerarchia dei processi ci viene mostrata **dall'indentazione**, come evidenziato nell'immagine

```
  949     949     949 ?          00:00:00  nginx
  950     949     949 ?          00:00:00   nginx
```

I processi **figli** appaiono sotto i loro **genitori**, **indentati** per livello.

I server sono essenzialmente programmi, spesso avviati dal sistema stesso al momento dell'avvio. Il compito svolto da un server è chiamato servizio. In questo modo, un web server fornisce servizi web. Il comando **netstat** è un ottimo strumento per aiutare a identificare i server di rete in esecuzione su un computer

```
[analyst@secOps ~]$ netstat
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  9      [ ]         DGRAM                    10597    /run/systemd/journal/dev-log
unix  3      [ ]         DGRAM                    10373    /run/systemd/notify
unix  7      [ ]         DGRAM                    10393    /run/systemd/journal/socket
unix  2      [ ]         DGRAM                    14255    /run/user/1000/systemd/notify
unix  3      [ ]         STREAM     CONNECTED     13138
unix  2      [ ]         DGRAM                    14033
unix  3      [ ]         STREAM     CONNECTED     14647    /run/user/1000/bus
unix  3      [ ]         STREAM     CONNECTED     13207
unix  2      [ ]         DGRAM                    14181
unix  3      [ ]         STREAM     CONNECTED     14030    /run/dbus/system_bus_socket
unix  3      [ ]         STREAM     CONNECTED     14029
unix  3      [ ]         STREAM     CONNECTED     13208    /run/systemd/journal/stdout
unix  3      [ ]         STREAM     CONNECTED     14646
unix  3      [ ]         DGRAM                    14258
unix  3      [ ]         STREAM     CONNECTED     13139    /run/systemd/journal/stdout
unix  3      [ ]         DGRAM                    14257
unix  3      [ ]         STREAM     CONNECTED     14680    @/tmp/.ICE-unix/393
unix  3      [ ]         DGRAM                    10376
unix  3      [ ]         STREAM     CONNECTED     14677    /run/user/1000/bus
unix  3      [ ]         STREAM     CONNECTED     12814    /run/systemd/journal/stdout
unix  3      [ ]         STREAM     CONNECTED     14674    @/tmp/.X11-unix/X0
unix  3      [ ]         STREAM     CONNECTED     14804    /run/user/1000/bus
unix  3      [ ]         STREAM     CONNECTED     12749
unix  3      [ ]         STREAM     CONNECTED     14673
unix  3      [ ]         STREAM     CONNECTED     14720    @/tmp/.X11-unix/X0
unix  3      [ ]         STREAM     CONNECTED     14676
unix  3      [ ]         STREAM     CONNECTED     15138
unix  3      [ ]         STREAM     CONNECTED     12813
unix  3      [ ]         STREAM     CONNECTED     14799
unix  3      [ ]         STREAM     CONNECTED     14719
unix  3      [ ]         STREAM     CONNECTED     12750    /run/systemd/journal/stdout
unix  3      [ ]         STREAM     CONNECTED     14679
unix  3      [ ]         STREAM     CONNECTED     14723    /run/user/1000/bus
unix  3      [ ]         STREAM     CONNECTED     12875
unix  3      [ ]         STREAM     CONNECTED     14800    @/tmp/.ICE-unix/393
unix  3      [ ]         STREAM     CONNECTED     14722
unix  3      [ ]         DGRAM                    10375
unix  3      [ ]         STREAM     CONNECTED     11694
unix  3      [ ]         STREAM     CONNECTED     14649
unix  3      [ ]         STREAM     CONNECTED     15275    /run/systemd/journal/stdout
unix  3      [ ]         STREAM     CONNECTED     15062
unix  3      [ ]         STREAM     CONNECTED     13518    /run/dbus/system_bus_socket
unix  3      [ ]         STREAM     CONNECTED     14792
```

Netstat **restituisce molte informazioni** se utilizzato **senza opzioni**. Queste opzioni possono essere utilizzate per filtrare e formattare l'output di netstat, rendendolo più utile. L'opzione che andremo ad utilizzare sarà **–tunap**, che **ci mostra le connessioni di rete attive**, porte in ascolto e i processi associati.
Le informazioni per il server nginx sono evidenziate.

```
[analyst@secOps ~]$ netstat -tunap
(Not all processes could be identified, non-owned process info
 will not be shown, you would have to be root to see it all.)
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:6633            0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:21              0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      -
tcp6       0      0 :::22                   :::*                    LISTEN      -
udp        0      0 10.0.2.15:68            0.0.0.0:*                           -
```

Domanda 1: **Qual è il significato delle opzioni –t, -u, –n, –a e –p in netstat?**
(usa man netstat per rispondere)
R:
-t = Mostra solo le connessioni **TCP** (Transmission Control Protocol).
-u = Mostra solo le connessioni **UDP** (User Datagram Protocol).
-n = Mostra indirizzi IP e numeri di porta invece dei nomi (es: **80** invece di **http**)
-a = Mostra tutte le connessioni attive e tutte le porte in ascolto
-p = Mostra il **PID e il nome del processo** che ha aperto la connessione o la porta. Richiede privilegi root per vedere i processi di altri utenti.

Domanda 2: **L'ordine delle opzioni è importante per netstat?**

R: No, l'ordine delle opzioni è **irrilevante**.

L'output di **netstat** visualizza alcuni servizi che sono attualmente in ascolto su porte specifiche. Le colonne interessanti sono:

- La prima colonna mostra il protocollo livello 4 in uso (UDP o TCP, in questo caso).
- La terza colonna utilizza il formato per visualizzare l'indirizzo IP locale e la porta su cui è raggiungibile un server specifico. L'indirizzo IP 0.0.0.0 significa che il server è attualmente in ascolto su tutti gli indirizzi IP configurati nel computer.
- La quarta colonna utilizza lo stesso formato socket per visualizzare l'indirizzo e la porta del dispositivo all'estremità remota della connessione. 0.0.0.0:* significa che nessun dispositivo remoto sta attualmente utilizzando la connessione.
- La quinta colonna visualizza lo stato della connessione
- La sesta colonna visualizza l'ID del processo (PID) del processo responsabile della connessione. Visualizza anche un nome breve associato al processo.

```
[analyst@secOps ~]$ sudo netstat -tunap
[sudo] password for analyst:
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:6633            0.0.0.0:*               LISTEN      293/python2.7
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN      949/nginx: master p
tcp        0      0 0.0.0.0:21              0.0.0.0:*               LISTEN      319/vsftpd
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      318/sshd
tcp6       0      0 :::22                   :::*                    LISTEN      318/sshd
udp        0      0 10.0.2.15:68            0.0.0.0:*                           219/systemd-network
```

Domanda 1: In base all'output netstat mostrato al punto (d), qual è il protocollo Layer 4, lo stato della connessione e il PID del processo in esecuzione sulla porta 80?

```
[analyst@secOps ~]$ sudo netstat -tunap
[sudo] password for analyst:
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:6633            0.0.0.0:*               LISTEN      293/python2.7
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN      949/nginx: master p
tcp        0      0 0.0.0.0:21              0.0.0.0:*               LISTEN      319/vsftpd
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      318/sshd
tcp6       0      0 :::22                   :::*                    LISTEN      318/sshd
udp        0      0 10.0.2.15:68            0.0.0.0:*                           219/systemd-network
```

R: Il Layer è **TCP**, lo stato della connessione è **LISTEN**, il **PID è 949**

Domanda 2: Sebbene i numeri di porta siano solo una convenzione, puoi indovinare che tipo di servizio è in esecuzione sulla porta 80 TCP?

R: Un server Web, questo perché la porta 80/TCP è al protocollo HTTP, usato dai web server per servire pagine web.

A volte è utile incrociare le informazioni fornite da **netstat** con **ps**. Sappiamo che il processo con **PID 949** è associato alla porta **TCP 80**. Usando **ps e grep** possiamo elencare tutte le righe dell'output di ps che contengono **PID 949**.

```
[analyst@secOps ~]$  sudo ps -elf | grep 949
[sudo] password for analyst:
1 S root       949     1  0  80   0 -  7192 -      06:08 ?        00:00:00 nginx: master process nginx
5 S http       950   949  0  80   0 -  8457 SyS_ep 06:08 ?        00:00:00 nginx: worker process
0 S analyst   1292   701  0  80   0 -  2720 -      08:38 pts/0    00:00:00 grep 949
```

La prima riga mostra un processo di proprietà dell'utente root (terza colonna), avviato da un altro processo con PID 1 (quinta colonna), alle 6.08 (dodicesima colonna)

La seconda riga mostra un processo con **PID 950**, di proprietà dell'utente **http**, avviato dal processo **949**, alle 6.08

La terza riga mostra un processo di proprietà dell'utente analyst, con **PID 701**, avviato da un processo con **PID 1292**, come comando **grep 949**.

Domanda 3: Il processo PID 949 è nginx. Come si potrebbe concludere ciò dall'output di cui sopra?

R: Alla riga 1, l'output mostra la riga di comando nginx: **master proces nginx**

Domanda 4: Cos'è Nginx? Qual è la sua funzione? (Ricerca google)

R: NGinx è un Web server estremamente snello ma completo dal punto di vista delle funzionalità. **In esso troviamo ad esempio il supporto per il protocollo di Rete IPv6 così come per i protocolli di messaggistica WebSocket**. Tra le altre feature di NGinx è possibile segnalare anche le funzionalità integrate per il **load balancing**, indispensabile per la gestione di sessioni ad alto traffico, e la possibilità di agire come **reverse proxy.** Per

quanto riguarda la sicurezza, il Web server consente inoltre di utilizzare connessioni crittografate, con la possibilità di archiviare i **certificati SSL** (*Secure Sockets Layer*), di filtrare i package veicolati sulla base di direttive definite dall'amministratore di sistema, di installare servizi che operano in background come per esempio gli antivirus e, sempre tramite reverse proxy, di anonimizzare i server di destinazione delle richieste.

Ciò che davvero distingue Nginx dagli altri server web classici come Apache è la sua architettura interna. Mentre Apache utilizza un modello basato su processi o thread, in cui ogni richiesta apre un thread o un processo separato (un approccio più pesante), **Nginx utilizza un'architettura asincrona basata sugli eventi**.

Cosa significa questo? Quella **Invece di creare un processo per ogni richiesta, Nginx ha un processo master che gestisce più processi worker** e ciascuno di questi worker può gestire migliaia di connessioni simultanee grazie a tecniche di I/O non bloccanti e alla programmazione basata sugli eventi. Ciò riduce l'utilizzo di memoria e CPU anche in situazioni di traffico elevato.

Domanda 5: La seconda riga mostra che il processo 950 è di proprietà di un utente chiamato http e ha il processo numero 949 come processo genitore. Cosa significa? È un comportamento comune?

R: Significa che il processo 950 è un **worker process** di NGINX, eseguito dall'utente http, e il suo **processo padre con** PID 949 è il **master process nginx**, eseguito da root.. Questo è normale poiché nginx viene eseguito da solo per ogni client che si connette alla porta 80 TCP.
È **comportamento standard** nei server web come **nginx perché il master proces** viene avviato dal root per aprire la porta **80,** poi si genera il processo figlio che **gestisce le successive richieste.**

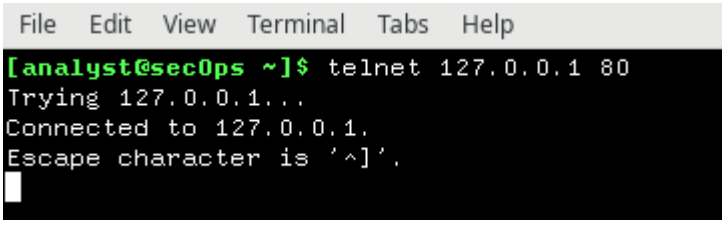
Domanda 6: Perché l'ultima riga mostra `grep 949`?

R: Grep 949 è stato utilizzato per filtrare l'output di ps. Grep 949 era ancora in esecuzione quando l'output è stato compilato, apparendo nell'elenco.

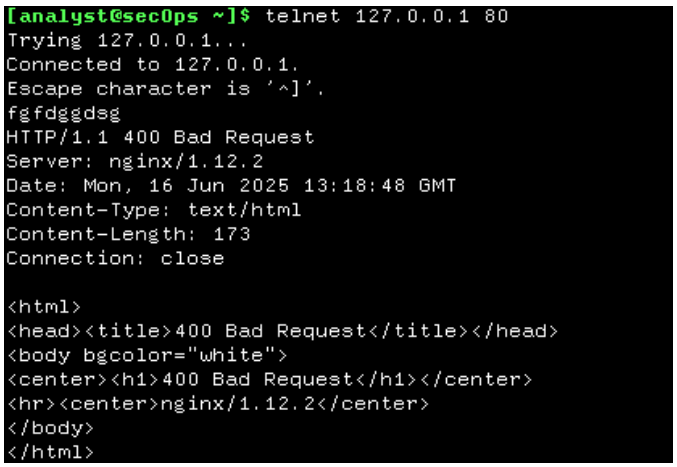
Parte 2 : Usare Telnet per Testare i Servizi TCP

Telnet è una semplice applicazione di shell remota ed è considerato insicuro perché non fornisce crittografia. Gli amministratori che scelgono di usarlo per gestire remotamente dispositivi di rete e server esporranno le credenziali di accesso a quel server, poiché il servizio trasmetterà i dati della sessione in chiaro. Sebbene Telnet non sia raccomandato come applicazione di shell remota, **può essere molto utile per testare rapidamente o raccogliere informazioni sui servizi TCP**. Il protocollo Telnet opera sulla porta 23 usando TCP per impostazione predefinita. Il client telnet, tuttavia, permette di specificare una porta diversa. Cambiando la porta e connettendosi a un server, il client telnet permette a un analista di rete di valutare rapidamente la natura di un server specifico comunicando direttamente con esso.

Nella Parte 1, si è scoperto che **nginx** era in esecuzione e assegnato alla porta 80 TCP. Sebbene una rapida ricerca su internet abbia rivelato che nginx è un server web leggero, come potrebbe un analista esserne sicuro? Cosa succederebbe se un attaccante avesse cambiato il nome di un programma malware in nginx, solo per farlo sembrare il popolare webserver? Usiamo telnet per connettersi all'host locale sulla porta 80 TCP



Grazie al protocollo Telnet, è stata stabilita una connessione TCP in chiaro, da parte del client Telnet, direttamente al server **nginx**, in ascolto sulla porta **80 TCP 127.0.0.1**. **Questa**



connessione ci consente di inviare i dati direttamente al server. Premendo alcune lettere sulla tastiera ci verrà restituito un errore nel formato di una pagina web.

Questo perché nginx è un server web, e non comprende la sequenza di lettere casuali inviate, e restituisce un errore nel formato di una pagina web.

Domanda 1: **Perché l'errore è stato inviato come pagina web?**

R: Questo perché nginx è un server web, progettato per rispondere alle richieste secondo il protocollo **HTTP**, e quindi restituisce sempre una risposta nel formato di una pagina web.

Non tutti i servizi sono uguali. Alcuni servizi sono progettati per accettare dati non formattati e non termineranno se vengono inseriti dati spazzatura tramite tastiera. Guardando l'output di **netstat** presentato prima, è possibile vedere un processo associato alla porta 22. Usa Telnet per connettersi ad esso. **La porta 22 TCP è assegnata al servizio SSH**. SSH permette a un amministratore di connettersi a un computer remoto in modo sicuro.

```
[analyst@secOps ~]$ telnet 127.0.0.1 22
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
SSH-2.0-OpenSSH_7.7
sgdfrh
Protocol mismatch.
Connection closed by foreign host.
```

Domanda 2: **Usa Telnet per connettersi alla porta 68. Cosa succede? Spiega.**

```
[analyst@secOps ~]$ telnet 127.0.0.1 68
Trying 127.0.0.1...
telnet: Unable to connect to remote host: Connection refused
```

R: Non si riesce a connettersi perché Telnet usa **TCP**, ma la porta **68 è UDP**, e poi perché nessun servizio è in ascolto su quella porta su **127.0.0.1**

Domanda 3: **Quali sono i vantaggi dell'uso di netstat?**

R: **Netstat** permette di avere una panoramica completa delle connessioni attive sul sistema: possiamo vedere quali connessioni sono attualmente aperte, quali porte sono in ascolto, e quali processi stanno utilizzando queste porte. Un altro aspetto positivo è che netstat è uno strumento già incluso nella maggior parte dei sistemi operativi, quindi non è necessario installare software aggiuntivo per usarlo.

Domanda 4: Quali sono i vantaggi dell'uso di Telnet? È sicuro?

R: Telnet è uno strumento semplice ma molto potente.Uno dei suoi vantaggi principali è la possibilità di **connettersi a una porta specifica su un host e inviare comandi manuali**, il che lo rende utile per fare debug di applicazioni di rete o per verificare se un determinato servizio è raggiungibile.Tuttavia, dal punto di vista della sicurezza, **Telnet non è affatto sicuro**. Il motivo è che tutto il traffico che invia, viene trasmesso in chiaro, **cioè senza alcuna cifratura.** Questo lo rende estremamente vulnerabile a intercettazioni e attacchi di tipo "**man-in-the-middle**", soprattutto se usato su reti non sicure.

Conclusione

In questo esercizio abbiamo esplorato il funzionamento dei server e dei processi su un sistema Linux, acquisendo competenze pratiche nell'utilizzo di comandi fondamentali come ps, netstat e telnet. Abbiamo compreso l'importanza dei privilegi elevati per visualizzare tutti i processi attivi, osservato la gerarchia dei processi attraverso ps -ejH, e identificato i servizi di rete in ascolto sulle varie porte con netstat -tunap.

In particolare, ci siamo concentrati sul server web nginx, analizzandone i processi master e worker, le porte utilizzate e il comportamento in risposta a connessioni dirette via Telnet. Questo ci ha permesso di riconoscere le caratteristiche tipiche di un server HTTP e distinguere comportamenti normali da possibili anomalie.

Infine, abbiamo confrontato due strumenti preziosi per l'analisi di rete: netstat, utile per avere una visione globale e strutturata delle connessioni attive e dei servizi in esecuzione; e telnet, uno strumento semplice ma efficace per testare i servizi TCP in modo diretto. Abbiamo anche sottolineato come Telnet, sebbene utile, non offra garanzie di sicurezza, e quindi non sia adatto all'amministrazione remota di sistemi moderni.

Questa attività fornisce una base solida per la comprensione dei servizi Linux e per le attività di analisi di rete e troubleshooting, fondamentali nel contesto della sicurezza informatica e dell'amministrazione di sistema.
