Cybersecurity con sistemi GNU/Linux

Dispensa del corso

Andrea Fazzi

1 Utilizzo avanzato del terminale

In questa sezione verranno discussi gli utilizzi più avanzati del terminale. In particolare, ci si soffermerà sui cosidetti *terminal multiplexer*.

1.1 I *terminal multiplexer*

I terminal multiplexer in Linux sono strumenti potenti che consentono agli utenti di creare più sessioni di terminale all'interno di una singola finestra del terminale. Questi strumenti sono particolarmente utili per gestire più processi contemporaneamente, mantenendo aperte le sessioni anche dopo la disconnessione, e per organizzare le finestre del terminale in modo efficiente.

Uno dei terminal multiplexer più popolari e ampiamente utilizzati è `tmux`. `tmux` permette agli utenti di creare, gestire e navigare tra più sessioni di terminale, facilitando l'organizzazione dei processi e la gestione delle finestre. Con `tmux`, è possibile dividere la finestra del terminale in più riquadri, ciascuno dei quali può eseguire un processo separato, e passare facilmente da una sessione all'altra.

Un altro esempio di terminal multiplexer è `screen`, che offre funzionalità simili a `tmux` ma con una sintassi leggermente diversa. Entrambi gli strumenti sono disponibili per la maggior parte delle distribuzioni Linux e possono essere installati tramite il gestore di pacchetti della distribuzione.

Utilizzare un terminal multiplexer può migliorare notevolmente l'efficienza e la produttività degli sviluppatori e degli amministratori di sistema, consentendo di lavorare con più processi contemporaneamente senza dover aprire molteplici finestre del terminale.

1.1.1 Riferimenti

1. https://opensource.com/article/21/5/linux-terminal-multiplexer

1.2 Introduzione a `tmux`

`tmux` è un potente strumento di gestione delle sessioni di terminale in Linux. Permette agli utenti di creare, gestire e navigare tra più sessioni di terminale all'interno di una singola finestra del terminale. Questa funzionalità è particolarmente utile per l'esecuzione di più programmi con una singola connessione, come quando si effettua una connessione remota a una macchina utilizzando Secure Shell (SSH) [1].

Per iniziare ad utilizzare `tmux`, basta digitare `tmux` nel terminale. Questo comando avvia un server `tmux`, crea una sessione predefinita (numero 0) con una singola finestra e si collega ad essa.

Una volta connessi a `tmux`, è possibile eseguire qualsiasi comando o programma come si farebbe normalmente.

1.3 Installazione di `tmux`

`tmux` è un'applicazione disponibile nel repository di `Arch Linux`. Per installarla basterà utilizzare il comando `pacman`.

```
sudo pacman -S tmux
```

Per verificare che l'installazione sia andata a buon fine:

```
tmux -V
```

Il comando restituirà la versione installata.

1.3.1 Riferimenti

1. https://www.redhat.com/sysadmin/introduction-tmux-linux
2. https://linuxhandbook.com/tmux/
3. https://hamvocke.com/blog/a-quick-and-easy-guide-to-tmux/
4. https://linuxize.com/post/getting-started-with-tmux/
5. https://linuxconfig.org/introduction-to-terminal-multiplexer-tmux
6. https://github.com/tmux/tmux/wiki/Getting-Started
7. https://www.howtogeek.com/671422/how-to-use-tmux-on-linux-and-why-its-better-than-screen/
8. https://opensource.com/article/17/2/quick-introduction-tmux
9. https://wiki.archlinux.org/title/Tmux
10. https://github.com/tmux/tmux/wiki

1.4 Primi passi con `tmux`

Per eseguire `tmux` si dovrà semplicemente invocare il comando all'interno di una sessione di terminale.

```
tmux
```

È possibile staccare la sessione `tmux` premendo `Ctrl+B` seguito da `D`. `tmux` opera utilizzando una serie di scorciatoie da tastiera (keybindings) attivate premendo la combinazione "prefisso". Di default, il prefisso è `Ctrl+B`. Dopo di che, premere `D` per staccare dalla sessione corrente. La sessione continua

ad eseguire in background anche dopo la disconnessione, permettendo di riprendere dove si è lasciati quando si è pronti a riconnettersi al server e riattaccarsi alla sessione esistente.

`tmux` fornisce una serie di scorciatoie da tastiera per eseguire comandi rapidamente all'interno di una sessione `tmux`. Alcune delle più utili includono:

- `Ctrl+B D` — Stacca dalla sessione corrente.
- `Ctrl+B %` — Suddividi la finestra in due pannelli orizzontalmente.
- `Ctrl+B "` — Suddividi la finestra in due pannelli verticalmente.
- `Ctrl+B` seguito da una freccia (sinistra, destra, su, giù) — Sposta tra i pannelli.
- `Ctrl+B X` — Chiudi il pannello.
- `Ctrl+B C` — Crea una nuova finestra.
- `Ctrl+B N` o `P` — Sposta alla finestra successiva o precedente.
- `Ctrl+B 0` (1,2...) — Sposta a una finestra specifica per numero.
- `Ctrl+B :` — Entra nella riga di comando per digitare comandi. L'auto-completamento tramite tab è disponibile.
- `Ctrl+B ?` — Visualizza tutte le scorciatoie da tastiera. Premere `Q` per uscire.
- `Ctrl+B W` — Apre un pannello per navigare tra le finestre in più sessioni.

1.4.1 Esercizio

Utilizza `tmux` per suddividere il terminale in due pannelli verticali. Nel pannello di sinistra apri uno script in Python utilizzando l'editor `nano`. Nel pannello di destra esegui il comando `htop`. Se `htop` non è presente nel sistema, procedi con la sua installazione attraverso il package manager `pacman`.

1.5 Configurazione di `tmux`

Il file di configurazione di `tmux`, noto come `tmux.conf`, è un file di testo che permette agli utenti di personalizzare l'ambiente di lavoro di `tmux` secondo le proprie preferenze. Questo file può essere posizionato in due luoghi principali:

- `~/.tmux.conf` per una configurazione specifica dell'utente corrente.
- `/etc/tmux.conf` per una configurazione globale, applicabile a tutti gli utenti del sistema.

Se il file `~/.tmux.conf` non esiste, può essere creato semplicemente eseguendo il comando `touch ~/.tmux.conf` nel terminale. Questo creerà un file di configurazione vuoto che può essere modificato per aggiungere le impostazioni desiderate.

La configurazione di `tmux` può includere una vasta gamma di opzioni, tra cui:

- Cambio del prefisso di comando predefinito.

- Abilitazione della modalità mouse.
- Impostazione di due prefissi.
- Cambio del comportamento predefinito del server.
- Inizio del conteggio dei numeri delle finestre e dei pannelli (Base-Index) a 1.
- Modifica dello sfondo del pannello corrente.
- …

Per esempio, per cambiare il prefisso di comando predefinito da `Ctrl+B` a `Ctrl+A`, si potrebbe aggiungere la seguente riga al file `tmux.conf`:

```
set-option -g prefix C-a
```

Dopo aver apportato modifiche al file di configurazione, è necessario ricaricarlo per applicare le nuove impostazioni. Questo può essere fatto eseguendo il comando `tmux source-file ~/.tmux.conf` dal terminale o utilizzando il comando `source-file ~/.tmux.conf` dalla modalità di comando di `tmux`. Per facilitare il processo, è possibile aggiungere un collegamento rapido nel file `tmux.conf` per ricaricare facilmente la configurazione:

```
bind r source-file ~/.tmux.conf \; display "Reloaded!"
```

Questo permette di ricaricare la configurazione premendo il prefisso seguito da `r`, visualizzando un messaggio di conferma.

1.5.1 Riferimenti

1. https://hamvocke.com/blog/a-guide-to-customizing-your-tmux-conf/
2. https://www.hostinger.com/tutorials/tmux-config
3. https://dev.to/iggredible/useful-tmux-configuration-examples-k3g
4. https://github.com/gpakosz/.tmux
5. https://arcolinux.com/everthing-you-need-to-know-about-tmux-configuration/
6. https://thevaluable.dev/tmux-config-mouseless/
7. https://github.com/samoshkin/tmux-config
8. https://wiki.archlinux.org/title/tmux
9. https://medium.com/@bhavik.n/customize-tmux-to-use-it-effectively-28b262c8b692

1.6 Creazione di layout personalizzati

Si supponga di voler creare un file di configurazione specifico per un progetto in cui il terminale viene diviso in tre parti: due colonne verticali di cui una a sua volta suddivisa orizzontalmente. In questo

caso è possibile utilizzare uno script di configurazione personalizzato per `tmux`. Questo script può essere salvato in un file separato, ad esempio in `~/.config/tmux/split.conf`.

```
new -s splitted_session # crea una nuova sessione

selectp -t 0 # seleziona il primo pannello
splitw -h -p 50 # divide il pannello corrente orizzontalmente in due parti

selectp -t 1 # seleziona il nuovo secondo pannello
splitw -v -p 50 # divide il pannello corrente verticalmente in due parti
selectp -t 0 # torna al primo pannello
```

Una volta creato il file di configurazione, occorrerà "eseguirlo" con

```
tmux source-file ~/.config/tmux/split.conf
```

1.6.1 Esercizio

Crea un file di configurazione `tmux` per produrre un layout con due righe di cui la prima suddivisa in due colonne secondo uno schema simile a quello riportato sotto.

```
-----------
|    |    |
|    |    |
-----------
|         |
-----------
```

2 Secure Shell

Il protocollo SSH (Secure Shell) è un protocollo di rete che fornisce una connessione sicura e crittografata tra due sistemi informatici su una rete non sicura. OpenSSH è una delle implementazioni più comuni di SSH, che offre una serie di strumenti per la gestione delle chiavi SSH, l'autenticazione e la sicurezza delle connessioni.

2.1 Comando ssh

Il comando `ssh` è utilizzato per stabilire una connessione sicura con un server remoto. Ad esempio, per connettersi a un server remoto con l'indirizzo IP `192.168.1.100` come utente `utente`, il comando è:

```
ssh utente@192.168.1.100
```

Questo comando avvia una sessione SSH con il server specificato, richiedendo l'autenticazione dell'utente.

2.2 Comando ssh-keygen

`ssh-keygen` è uno strumento per generare una coppia di chiavi pubbliche/private per l'autenticazione SSH. Per generare una nuova chiave SSH, si può eseguire il comando `ssh-keygen` come nell'esempio sotto

```
ssh-keygen -t rsa
```

Agendo sulle opzioni del comando, si può specificare un percorso per salvare la chiave impostare una passphrase per la chiave privata per una maggioreq sicurezza.

```
ssh-keygen -t rsa -b 2048 -f ~/.ssh/mykey
```

Questo comando genera una chiave RSA di 2048 bit e la salva nel file `~/.ssh/mykey`. La chiave pubblica corrispondente sarà salvata in `~/.ssh/mykey.pub`.

2.3 Comando ssh-copy-id

`ssh-copy-id` è uno strumento che copia la chiave pubblica SSH di un utente in un server remoto, consentendo l'accesso senza password. Per copiare la chiave pubblica `mykey.pub` dell'utente corrente al server `192.168.1.100`, il comando è:

```
ssh-copy-id -i ~/.ssh/mykey.pub utente@192.168.1.100
```

Questo comando copia la chiave pubblica nel file `~/.ssh/authorized_keys` dell'utente remoto, consentendo l'accesso senza password. Dopo aver copiato la chiave, è possibile testare l'accesso senza password con:

```
ssh -i ~/.ssh/mykey utente@192.168.1.100
```

I nomi di default per le chiavi pubbliche e private sono rispettivamente `id_rsa.pub` e `id_rsa`. Utilizzando queste chiavi, l'accesso al server senza password si semplifica. Basterà infatti eseguire

```
ssh utente@192.168.1.100
```

2.4 Mobile Shell (mosh)

Mosh, acronimo di Mobile Shell, è un'applicazione di terminale remoto che risolve i problemi di connettività tipici di SSH, specialmente su reti mobili o instabili. A differenza di SSH, che utilizza TCP e richiede una connessione stabile, Mosh utilizza UDP, che è un protocollo senza connessione, permettendo una connessione più stabile e reattiva anche in presenza di interruzioni temporanee della connessione. Mosh mantiene una connessione attiva attraverso cambiamenti di indirizzo IP e sospensioni del dispositivo, rendendolo ideale per l'uso su dispositivi mobili.

2.4.1 Come funziona Mosh

Mosh inizia stabilendo una connessione SSH per l'autenticazione, utilizzando le stesse credenziali di SSH (ad esempio, password o chiavi pubbliche). Successivamente, avvia un server Mosh sul dispositivo remoto e stabilisce una connessione UDP per la comunicazione. Questo approccio consente a Mosh di gestire meglio la perdita di pacchetti e di mantenere una connessione attiva anche in presenza di interruzioni temporanee della rete.

2.4.2 Vantaggi di Mosh rispetto a SSH

- **Robustezza e Responsività**: Mosh è più robusto e reattivo rispetto a SSH, soprattutto su connessioni Wi-Fi, cellulari e lunghe distanze. Questo è dovuto al suo protocollo basato su UDP che gestisce meglio la perdita di pacchetti e imposta il tasso di fotogrammi in base alle condizioni della rete.
- **Eco Locale Intelligente**: Mosh fornisce un eco locale speculativo delle pressioni dei tasti, permettendo all'utente di vedere le proprie pressioni dei tasti quasi istantaneamente, senza attendere il round trip di rete. Questo migliora l'esperienza utente, specialmente su connessioni ad alta latenza.
- **Supporto per Roaming**: Mosh supporta il roaming, permettendo all'utente di cambiare la propria posizione fisica senza interrompere la sessione. Questo è particolarmente utile per gli utenti mobili che si spostano tra diversi luoghi.

2.4.3 Esempi

Per connettersi a un server remoto utilizzando Mosh, il comando è simile a quello di SSH, ma utilizzando `mosh` al posto di `ssh`. Ad esempio:

```
mosh utente@host
```

E' possibile utilizzare `mosh` e `tmux` insieme. Ad esempio, con la seguente linea di comando è possibile connettersi al server remoto ed eseguire tmux.

```
mosh utente@host -- tmux
```

2.5 Riferimenti

[1] https://www.ssh.com/academy/ssh/copy-id [2] https://www.techtarget.com/searchsecurity/tutorial/Use-ssh-keygen-to-create-SSH-key-pairs-and-more [3] https://alexhost.com/faq/using-ssh-copy-id-ssh-keygen-commands-in-linux/ [4] https://www.ssh.com/academy/ssh/keygen [5] https://www.digitalocean.com/commun essentials-working-with-ssh-servers-clients-and-keys [6] https://docs.oracle.com/en/operating-systems/oracle-linux/openssh/openssh-WorkingwithSSHKeyPairs.html [7] https://www.redhat.com/sysadmin/configur ssh-keygen [8] https://www.digitalocean.com/community/tutorials/how-to-configure-ssh-key-based-authentication-on-a-linux-server [9] https://www.thegeekstuff.com/2008/11/3-steps-to-perform-ssh-login-without-password-using-ssh-keygen-ssh-copy-id/ [10] https://mosh.org/