Backup con *burp*

Stefano Antonelli

Servizi Nazionali

Tutorial days CCR - 10.10.2022

Indice

Introduzione

- *burp* è un software per il backup e ripristino dati [1, url sito]
- È un *free open source software* rilasciato con licenza *AGPLv3*
- Lo scopo è ridurre il traffico di rete e l'occupazione di spazio disco dei backup
- Sviluppato nel tentativo di migliorare alcuni svantaggi, secondo lo sviluppatore, di *Bacula* [2, bacula] e [3, features]
  - complessità nella configurazione, 4 componenti ciascuna con il suo file di configurazione
  - complessità nel debug
  - rivolto più al backup su nastro che disco
  - non implementa il *delta* nella differenza dei file nel caso in cui cambi solo qualche byte
  - non riprende un backup interrotto

Software

- Server (sistemi *Unix like*):
  - RHEL/CentOS 7,8
  - Debian, Ubuntu
- Client:
  - Linux, Windows, Mac
- Aggiornamenti: un'occhiata al CHANGELOG [6, changelog]
- Installazione: compilazione dei sorgenti, rpm, deb [4, software]
  - il software è stato portato in Github. Qualcosa, vecchie release ed altro, resta ancora sulla pagina di download del sito originale [5, sito originale download]

Funzionalità

- Un file di configurazione per il server ed uno per il client; alcune opzioni sul server sovrascrivono quelle sul client (*e.g.* `include`) o viceversa (*e.g* `keep`, `timer_arg`), leggere il man del file di conf [7, man]
  - opzioni del tipo `client_can_` oppure `server_can_` consentono, o meno, operazioni da parte del client o del server
- Trasmissione dei dati client/server criptata *SSL*
- È possibile trasmettere dati già criptati sul client (se non ci si fida del server...) utilizzando il parametro `password_encryption` ma con alcuni *caveat* tra cui: nomi dei file non sono criptati, si perde il *delta* backup incrementale, se perdo la password perdo i backup
- Espressioni regolari per includere/escludere dal backup file o directory [8, regex]
- Backup su server di destinazione in altro sito/cloud: in fase di sviluppo
- Un parametro imp. è `max_children=5` (è nel file di configurazione del server). È il numero di processi figli del processo originale ed equivale al numero di client che possono connettersi simultaneamente al server
- Notifiche: si può configurare il server in modo che invii notifiche al termine del backup (in caso di successo o di fallimento)
- Possibilità di eseguire script sul client, prima o dopo il backup/restore

Installazione del server/client

- Il modo più rapido è mediante i gestori di pacchetti *rpm, deb* (server o client) o *exe,dmg* (solo client)
- Un file di configurazione per il server (`/etc/burp/burp-server.conf`) con il parametro `mode = server` ed un file di conf. per il client (`/etc/burp/burp.conf`) con il parametro `mode = client`
- Sul server, inoltre, c'è un file di configurazione per ogni client in (`/etc/burp/clientconfdir`)
- L'*installer Windows* all'avvio, richiede la configurazione [10, conf windows]

Aggiungere un client

- Sul server, ogni client (*e.g. hostname*) è identificato da un file nella dir `/etc/burp/clientconfdir` con nome *hostname*
  - se il nome del file inizia con . o ~ viene ignorato
  - il file deve contenere almeno l'argomento `password = mia_pass` e la password deve essere la stessa presente nel file di configurazione del client, a meno di non utilizzare `password_check = 0` sul server
  - il nome del file deve essere uguale al valore del parametro `cname` = presente nel file di conf sul client
- Dopo l'installazione di *burp* sul client, si configura il file `/etc/burp/burp.conf` che conterrà almeno le tre righe:
  - `cname = hostname`
  - `password = mia_pass`
  - `server = burp_server`

Come effettuare il backup

- Sul client posso forzare il backup (attenzione all'opzione `client_can_force_backup` sul file di configurazione del server)
  - Client Linux: forzo il backup lanciando il comando `burp -a b` oppure aggiungo in `cron` il comando `burp -a t` (*e.g.* `7,27,47 * * * * root /usr/sbin/burp -a t`) ed il server decide quando far partire il backup in base al `timer_arg` ed al suo carico (l'opzione sul server sovrascrive quella sul client)
  - Client Windows: forzo il backup lanciando il comando `burp.exe -a b` dalla directory `C:\Program Files\Burp\bin` oppure configuro un *Windows Scheduler task*
- Sul client possono essere eseguiti degli script prima/dopo il backup utilizzando l'opzione `backup_script_pre/post=/path/to/the/script`

Quando effettuare il backup

- Il parametro che regola l'inizio di un backup è `timer_arg`. Si può configurare sul client o sul server (client sovrascrive il server). Il primo argomento `timer_arg` presente nel file di conf, indica il tempo minimo dall'ultimo backup utile; gli altri passano dei parametri allo script di backup per verificare se l'intervallo di tempo del backup è quello desiderato *e.g.*:
  - `timer_arg = 20h`
  - `timer_arg = Mon,Tue,Wed,Thu,Fri,00,01,02,03,04,05,19,20,21,22,23`
  - `timer_arg = Sat,Sun,00,01,02,03,04,05,06,07,08,17,18,19,20,21,22,23`

Quanti backup conservare

- *burp* assegna un numero ed un timestamp ad ogin backup completato (*e.g.* `0000576 2022-10-05 00:37:18 +0200`). Il numero viene incrementato di una unità rispetto al precedente. La conservazione dei backup si basa sul numero del backup e non sul timestamp
- Il numero di backup da tenere è deciso dal parametro `keep` presente nel file di configurazione del server o del client (client sovrascrive il server). I parametri `keep` vengono letti a cascata *e.g.*:

```
keep=7
keep=4
keep=6
```

Se ho un backup giornaliero, vuol dire tieni 7 backup giornalieri, 4 settimanali e 6 mensili

Cosa mettere sotto backup

- Per decidere di quali file o directory fare il backup o meno, utilizzo i parametri `include/exclude` sul file di conf del client o del server (server sovrascrive il client a meno del parametro `server_can_override_includes=[0|1]`). I parametri `include/exclude` possono avere come valore delle espressioni regolari per gruppi di files o directory [8, regex]

```
include=/home
include=/etc
exclude=/dev
exclude=/proc
```

Rimuovere un client

Un client può essere disabilitato o rimosso

- Disabilitato:
  - si rinomina il file di conf presente sul server `/etc/burp/clientconfdir`. Questo non ferma le richieste di connessione verso il server se è presente un *cron* o un *Windows scheduler*
- Rimosso:
  - Il client utilizza il suo certificato per comunicare con il server. Per rimuovere il client, va revocato il suo certificato e questo si fa utilizzando il comando `burp_ca` sul server

Installazione

- mini test di installazione di un server e due client (*Linux* e *Windows*)

Burp-UI webuser interface

- *burp* non ha una interfaccia web. Questa è stata sviluppata indipendentemente
- *URL* `https://git.ziirish.me/ziirish/burp-ui` (*screenshots* e demo per verificare il funzionamento)
- *Doc* `https://burp-ui.readthedocs.io/en/latest/#documentation`

Riferimenti

[1] *URL*: `https://burp.grke.org`

[2] *Bacula*: `https://burp.grke.org/why.html`

[3] *features*: `https://burp.grke.org/features.html`

[4] *software*: `https://github.com/grke/burp/wiki/Binary-packages`

[5] *download*: `https://burp.grke.org/download.html`

[6] *CHANGELOG*: `https://burp.grke.org/changelog.html`

[7] *man conf*: `https://burp.grke.org/docs/manpage.html`

[8] *regex*: `https://github.com/deajan/linuxscripts/tree/master/burp/incexc`

[9] *tuning*: `https://github.com/grke/burp/wiki/Performance-Tips`

[10] *configurazione windows*: `https://burp.grke.org/docs/windows-installer.html`

[11] *security*: `https://burp.grke.org/docs/security-models.html`

Security models di *burp*

- La comunicazione client/server è sempre criptata *SSL*. I certificati server/client garantiscono l'identità [11, security]
- Modello server "inaffidabile", agisco sul client
  - criptare i dati lato client con l'opzione `encryption_password = .` *Caveat*: se perdo la pwd perdo i backup, nomi delle dir e dei file non sono criptati, si perdono i *deltas* nei backup
  - uso `server_can_restore=0` default 1
  - uso `server_can_override_includes=0` per evitare che il server faccia il backup di file ai quali non voglio dare accesso, default 1
- Modello client "inaffidabile", agisco sul server, diverse opzioni
  - `client_can_delete=0`, default 1 (anche per i successivi)
  - `client_can_diff=0`
  - `client_can_list=0`
  - `client_can_monitor=0`
  - `client_can_restore=0`
  - `client_can_verify=0`

4 fasi del backup

Il backup di un client avviene in 4 fasi

- Fase 1 client: scansione del filesystem ed invio delle statistiche al server
- Fase 1 server: riceve le statistiche dal client

- Fase 2 client: invia le modifiche richieste dal server e qui termina il suo compito
- Fase 2 server: richiede e riceve le modifiche dal client e crea una lista dei non modificati e dei modificati

- Fase 3 server: crea la nuova lista per il backup a partire da quella dei non modificati e dei modificati

- Fase 4 server: termina il backup riorganizzando i dati ricevuti e posizionandoli correttamente. Genera i *reverse deltas* in modo da risparmiare spazio per i precedenti backup