Relazione sul Progetto di Gestione Di Rete

Alessandra Fais
Matricola 481017

A.A. 2013/14

1. Introduzione

Il progetto consiste in uno script Lua che permette il monitoraggio dei processi attualmente in esecuzione sulla macchina mediante l'utilizzo della libreria Sysdig. All'utente viene data la possibilità di rilevare informazioni su tutti i processi o su uno in particolare. Alla sua esecuzione lo script sarà in grado di mostrare, ad intervalli regolari, il tempo di esecuzione e PID del/i processo/i e il numero di socket aperti e chiusi. I dati sono ricavati dall'intercettazione di system call `socket` e relative `close`.

2. Application Performance Management

Un'applicazione di questo tipo rientra nell'ambito del monitoraggio e della gestione delle performance delle applicazioni. Gli aspetti di cui si tiene conto per le misurazioni sono il grado di soddisfazione degli utenti nei confronti del servizio che utilizzano e le risorse computazionali richieste dalle applicazioni. La qualità del servizio può essere stimata mediante il rilevamento delle transazioni e delle richieste effettuate da una applicazione e dei suoi tempi di risposta alle azioni degli utenti.

2.1. Packet capture

Sulle reti LAN è possibile intercettare i pacchetti mediante l'utilizzo di una monitoring port sul network switch: scopo della porta è fare mirroring, ossia inviare una copia dei pacchetti che transitano nello switch a una connessione di controllo, in modo che questi possano essere poi analizzati.

Alternativa alle monitorning ports è il network tap, un dispositivo hardware che permette di accedere al flusso di dati nella rete: un caso tipico può essere l'analisi dei pacchetti che transitano su un collegamento punto-punto full-duplex tra due stazioni, in questo caso il tap avrà almeno 3 porte (una verso ognuna delle 2 stazioni e una porta di monitoring) e permetterà il passaggio dei dati in maniera trasparente per le stazioni ma allo stesso tempo invierà una copia di ogni pacchetto in transito attraverso la porta di monitoring in modo che una terza parte che sta in ascolto su tale porta sia in grado di fare l'analisi del traffico.

Una volta catturato il traffico e reso in una forma leggibile esso viene sfruttato per la determinazione di errori o problemi sulla rete, il controllo del suo utilizzo e la valutazione della qualità del servizio.

Pcap: Packet Capture Library

Uno degli strumenti ideati per la cattura dei pacchetti è la libreria `pcap`, il cui scopo è fornire un'interfaccia per intercettare i pacchetti a basso livello sulla rete. La libreria si aggancia al livello più basso dello stack protocollare TCP/IP e ciò consente la cattura e successiva gestione dei pacchetti che arrivano alla scheda di rete del dispositivo ma che non sono necessariamente destinati a lei (tutto il traffico che arriva alla scheda di rete è accessibile).

`Wireshark` e `tcpdump` sono due analizzatori di pacchetti che usano la libreria `pcap` per la cattura del traffico.

2.2. System call analysis

Gli scopi del monitoraggio del traffico sulla rete sono gli stessi che portano all'analisi delle operazioni tra processi su una macchina. In questo secondo scenario però non è possibile usare l'approccio descritto sopra e fare sniffing sui pacchetti: la motivazione è che le comunicazioni inter-processo non viaggiano sotto forma di pacchetto. L'unico modo per analizzare i dati scambiati è l'intercettazione delle system call.

Sysdig

Sysdig è un progetto Open Source focalizzato sul monitoraggio e la risoluzione di problemi a livello di sistema. La libreria permette di mettere assieme gli elementi raccolti mediante osservazione della rete con l'attività interna alla macchina, mostrando la relazione tra i dati e i processi che li hanno spediti.

Per quanto riguarda la risoluzione di problemi, Sysdig consente la creazione di un trace file sul quale poi poter applicare diversi filtri al fine di controllare il funzionamento delle attività di sistema (analisi offline). In alternativa il monitoraggio dell'attività di sistema può essere fatto in real time. Una volta acquisite le informazioni la ricerca dei dati utili può essere effettuata in diversi modi: tra questi è possibile utilizzare degli script, scritti in Lua, chiamati `chisels`.

Lua chisels

I chisels sono script che analizzano il flusso di eventi di Sysdig al fine di ricavarne qualcosa di utile. Gli eventi sono arricchiti di un contesto e portati a livello utente, così qualsiasi linguaggio di scripting può essere utilizzato.

Ogni chisel si compone di quattro variabili globali che vanno definite necessariamente: `description`, `short description`, `category`, `list of arguments`. Le altre parti permettono di inizializzare i filtri da utilizzare, elaborare gli eventi che accadono durante l'esecuzione del chisel, settare gli argomenti (se presenti)... Ogni chisel può essere quindi personalizzato in ogni componente per essere adattato al particolare scopo per il quale viene scritto.

3. Architettura del progetto

Il progetto prevede due modalità di utilizzo: monitoraggio di tutti i processi o di un processo a scelta. Il funzionamento nella prima maniera piuttosto che nella seconda viene scelto mediante l'inserimento di un parametro all'avvio dello script, rispettivamente il parametro `all` oppure il `nome del processo`. Il chisel prevede questo parametro come unico argomento; il tipo dell'argomento determina il filtro usato, impostato nella `on_set_arg()`.

La funzione `on_init()` permette di indicare i filtri che Sysdig deve utilizzare al fine di ottenere solo i campi necessari agli scopi del chisel (per esempio filtra solo gli eventi relativi alle system call `socket` e `close`). Sysdig estrae i dati relativi ai campi indicati e lo script ottiene le informazioni necessarie su questi campi mediante `chisel.request_field(sysdig_filter_field)`. I valori verranno estratti per ogni evento nella `on_event()`. Se non si usassero i filtri messi a disposizione da Sysdig la `on_event()` riceverebbe tutti gli eventi per poi dover lavorare soltanto su un piccolo sottoinsieme di questi: il chisel

funzionerebbe comunque ma il codice diverrebbe più complesso e perderebbe di efficienza.

La funzione `on_event()` permette di elaborare i dati raccolti da Sysdig per ogni evento. Vengono aggiornate le strutture dati che tengono il conto delle system call `socket` e delle rispettive `close`; inoltre vengono salvate informazioni proprie dei processi quali nome, PID, tempo di esecuzione.

La funzione `on_interval()` consente di fare stampe periodiche dei dati elaborati; tali stampe comprendono risultati parziali aggiornati a ogni intervallo (numero di `socket` e `close` nell'ultimo intervallo) e risultati accumulati a partire dall'inizio dell'esecuzione del processo o del chisel. Le strutture dati usate per gestire i dati parziali vengono svuotate e sovrascritte con i nuovi risultati a ogni aggiornamento.

La funzione `on_capture_end()` rende possibile la gestione della fine della cattura del traffico. Nel caso di questo chisel i dati sono presi in tempo reale e il monitoraggio viene arrestato inviando un segnale SIGINT al processo (ossia digitando CTRL-C e provocando l'interruzione dell'esecuzione del chisel).

3.1. Strutture Dati

Vengono impiegate diverse strutture dati per memorizzare i dati necessari per calcolare i risultati da mostrare all'utente. In Lua si utilizza un solo tipo di struttura dati, la `table`, ossia un hash table di associazioni `<key, value>`. Alcune table sono solamente aggiornate e contengono il numero globale di `socket` e di `close`, la durata dell'esecuzione del processo e l'associazione tra il nome del processo e il suo PID. Le altre table usate contengono risultati parziali, come il numero di `socket` e `close` nell'ultimo intervallo di tempo o i file descriptor utilizzati; queste strutture dati vengono svuotate a ogni intervallo di tempo, per essere poi sovrascritte con i dati raccolti nel successivo intervallo.

3.2. Funzioni extra

Oltre alle funzioni sopra descritte sono state implementate altre funzioni che permettono di ricavare informazioni utili da visualizzare poi all'utente.

La funzione `findLast(pathname, pattern)` prende il pathname assoluto del processo (restituito da Sysdig) e ne estrae il nome del processo. Il nome verrà utilizzato nella stampa a schermo e usato come identificativo del processo stesso anche nelle strutture dati (la chiave usata nelle table è sempre il nome del processo ad eccezione della tabella che tiene traccia dei file descriptor aperti).

La funzione `deleteStoppedProcess(proc)` si occupa di svuotare i campi delle strutture dati usati per il processo `proc` nel momento in cui tale processo viene arrestato o chiuso, oppure si sospende.

La funzione `calcExecTime(pname, extime)` usa il tempo `extime`, ossia il tempo di sistema nel quale si verifica l'evento: se il processo `pname` è stato appena avviato o comunque questa è la prima misurazione che lo riguarda (primo intervallo di misurazione dello script) allora la sua durata è impostata a 0 e `extime` è memorizzato come suo tempo iniziale. A ogni successiva misurazione la durata del processo sarà calcolata come differenza tra l'ultimo timestamp in secondi e il tempo iniziale del processo.