Proxying con DeleGate

Recentemente in questa rubrica ci siamo occupati di filtraggio di contenuti nella navigazione web, e quindi di proxy. In questo articolo continuiamo ad occuparci di proxy ma da un punto di vista più generale. Un uso comune di un proxy è quello di schermare i veri server dagli utilizzatori, fornendo un accesso ai soli servizi che devono essere offerti in modo più o meno sicuro. Una situazione tipica in cui è spesso usato un proxy, è quella di un server presente nella rete interna di una azienda che deve però offrire dei servizi anche all'esterno. Possiamo mettere nella zona DMZ tra la rete interna e quella esterna, un proxy che riceve le connessioni dei client dall'esterno e le rimanda al vero server interno. I client esterni non sanno quale è il vero server, anzi di solito scambiano il proxy per esso.

Il più semplice proxy che può svolgere questa funzione è un *Circuit Level Proxy* il cui funzionamento è molto semplice. Supponendo che la connessione sia realizzata con il protocollo TCP, il Circuit Level Proxy termina su se stesso la connessione TCP con il client ricevendo dal client quindi tutti i pacchetti destinati al server. Apre poi una diversa connessione TCP con il server a cui invia i pacchetti ricevuti dal client, e viceversa. In questo caso il proxy non esamina il contenuto dei pacchetti e quindi non può effettuare alcuna operazione di filtro, ma ottiene già un piccolo risultato dal punto di vista della sicurezza: un attaccante esterno che vuole attaccare la macchina e non il servizio, deve prima riuscire a prendere il controllo del proxy per poi poter attaccare il vero server. Il proxy può essere una macchina molto semplice, con pochi servizi attivi, con un sistema operativo diverso da quello del server, e tenuta sotto costante controllo. Quindi un attacco contro il proxy ha di solito meno possibilità di riuscire e buone possibilità di essere rilevato in tempo.

Un Circuit Level Proxy non protegge però dagli attacchi al servizio, poiché esso invia direttamente al server finale tutti i pacchetti ricevuti senza modificarli. Per poter proteggere anche il servizio, il proxy deve conoscere la sintassi del servizio stesso ed agire da *Application Layer Gateway* (ALG). Visto che un ALG conosce il protocollo a livello 7 della pila OSI, è in grado di filtrare i comandi che potrebbero essere nocivi o ancora meglio, limitare i comandi inviati al server ad un solo sottoinsieme di comandi fidati. Si noti come un ALG funziona in modo diverso da un Circuit Level Proxy: il Circuit Level Proxy invia al server finale pacchetto per pacchetto i dati ricevuti dal client, mentre un ALG colleziona tutti i pacchetti che formano un comando, esamina il comando e solo dopo che lo ha validato lo invia al vero server. Lo stesso ovviamente avviene per i pacchetti di ritorno dal server al client. Ovviamente il problema degli ALG è che bisogna costruirne uno per ogni applicazione.

Capita spesso di trovarsi nella situazione di avere un servizio per cui non esiste un ALG in commercio che si adatti alle nostre esigenze, in questo caso possiamo sicuramente adottare un Circuit Level Proxy, ma con un attimo di impegno possiamo fare molto di più.

DeleGate

In questo articolo utilizzeremo DeleGate, un proxy open-source gratuito solo per uso personale, che ci permette di indicare alcune soluzioni possibili al problema appena citato. Si può scaricare DeleGate dal sito omonimo, per le piattaforme Microsoft vi è una versione precompilata, mentre per le altre si può direttamente compilare il sorgente con il solito comando `make`. Durante la compilazione verrà richiesto l'indirizzo email da indicare nei banner del programma ed al quale il programma invierà un email in caso di problemi. Il programma è molto semplice visto che è composto da un solo eseguibile chiamato `delegated` e nessun file di configurazione. Di solito il programma non gira con le priorità dell'amministratore ma con quelle dell'utente *nobody* o dell'utente che ha lanciato il processo se diverso dall'amministratore. DeleGate può essere attivato come un ALG per una lunga lista di protocolli quali http, ftp, telnet, nntp, smtp, pop, imap, lpr, ldap, icp, dns, ssl, socks ecc., oppure può essere usato come un Circuit Level Proxy per tcp o udp. DeleGate implementa un controllo degli accessi sugli indirizzi IP in qualunque modalità esso sia attivato, e quando in funzione ALG, a seconda del protocollo, può anche implementare l'autenticazione dei client o degli utenti permettendo quindi di aggiungerla quando il server finale non la offrisse. Ma la funzione di maggior interesse per noi è il fatto che in ogni modalità è possibile inserire dei filtri sui dati in transito nel proxy. Vi sono già dei filtri predefiniti ma altri possono essere costruiti semplicemente realizzando un programma in un linguaggio qualsiasi in grado di leggere da `stdin` e scrivere su `stdout`. I filtri possono essere posizionati in vari punti del percorso dei dati, possono essere unidirezionali o bidirezionali, ovvero ricevere il traffico in entrambe le direzioni o solo quello in una direzione.

Un Esempio

Consideriamo qui un esempio semplicissimo anche se poco realistico. Supponiamo di dover dare accesso telnet ad una macchina interna dall'esterno ma che vogliamo anche limitare quello che i client possono fare con questo accesso telnet. Possiamo usare DeleGate come Circuit Level Proxy creando i nostri filtri per trasformarlo in un ALG. Per prima cosa creiamo una directory, ad esempio `/delegate`, di proprietà dell'utente sotto cui girerà `delegated`, ed in essa una sotto-directory `bin` nella quale mettiamo l'eseguibile `delegated` ed il filtro `toserver.pl`, il semplicissimo filtro che abbiamo scritto in Perl. Attiviamo quindi `delegated` con la linea di

comando riportata in tabella 1. La linea di comando, che abbiamo diviso su più righe per convenienza con "\" come carattere di continuazione, indica che `delegated` deve

- mostrare nei banner ed inviare messaggi di errore all'amministratore pippo@pluto.com
- deve girare all'interno della directory `/delegate/` ove crea varie sotto-directory, ad esempio `log/` per i log, `etc/`, `work/` ove l'eseguibile viene eseguito ecc.
- l'opzione `-P` indica il numero IP e la porta (tcp) su cui `delegated` si mette in ascolto
- l'opzione SERVER indica che viene fatto un *relay tcp*, quindi non ALG, verso il server 192.168.1.33 sulla porta 23
- i due comandi `HOSTLIST` creano due access-list chiamate *clients* e *server*
- il comando `PERMIT` permette solo le connessioni di tipo *tcprelay* tra i *clients* ed il *server*
- il comando `FTOSV` indica di usare il programma `toserver.pl` come Filtro verso il Server, il quale introduce alcune caratteristiche ALG al nostro Circuit Level proxy.

Il Filtro

Per ovvie ragioni di spazio, il filtro che riportiamo è molto rudimentale ed incompleto. In ogni caso ci sembra interessante indicare quello che fa. Questo filtro unidirezionale legge ogni pacchetto ricevuto dal client, lo elabora anche accorpandolo con altri pacchetti già ricevuti e poi invia al server i pacchetti modificati. La parte centrale del filtro è perciò un loop che legge al più 2 byte alla volta. Finché non si incontra per due volte il carriage-return, si inviano i dati al server senza controlli, questo poiché le prime due stringhe inviate al server sono la username e la password. Dopo di ciò si accumulano nell'array `buffer` i caratteri inviati singolarmente dal client (telnet invia per lo più un carattere per pacchetto) sino a che non si riceve dal client il carriage-return `\r\0`. A questo punto si esamina il contenuto del buffer e se la prima parola non è `exit`, la si sostituisce con `ls`. In questo modo abbiamo ottenuto che i client possono dare solo due comandi: `exit` e `ls`. E' ovvio come estendendo questo semplicissimo esempio possiamo filtrare e limitare i client a solo certi particolari comandi, limitare la lunghezza delle righe di comando ad un certo numero massimo di caratteri, fare sostituzioni ecc. E' anche chiaro come questo filtro debba essere migliorato prima di poter essere implementato veramente: ad esempio è necessario introdurre una gestione degli errori del fitro stesso, il reporting e log delle sue attività, inoltre il filtro deve essere bidirezionale (quello indicato è unidirezionale) altrimenti il client non vede i caratteri che digita sino a che non preme il carriage-return, bisogna poi gestire il backspace e l'editing della riga di comando da parte del client e così via.

Andrea Pasquinucci

pasquinucci@ucci.it

Riferimenti Bibliografici

[1] http://www.delegate.org/

```
delegated ADMIN="pippo@pluto.com"\
  DGROOT="/delegate/" UMASK="077” -P10.0.0.7:23\
  SERVER=tcprelay://192.168.1.33:23\
  HOSTLIST="clients:10.1.0.0/24" HOSTLIST="server:192.168.1.33/32"\
  PERMIT="tcprelay:{server:23}:clients"\
  FTOSV="toserver.pl"
```

Tabella 1. Comando di attivazione di delegated

```
# ... lines omitted …

$stat=0;
while(sysread (STDIN,$char,2)){
  if ($char eq "\r\0"){
    if ( $stat >= 2) {
      undef $a;
      undef $b;
      @b=split(/ /,join('',@buffer));
      if ($b[0] ne "exit"){
        $b[0]="ls";
      }
      $a=join(' ',@b);
      syswrite (STDOUT,$a,length($a));
      undef @buffer;
    }
    $stat++;
    syswrite (STDOUT,$char,2);
  } elsif ( $stat < 2) {
    syswrite (STDOUT,$char,2);
  } else {
    push(@buffer,$char);
  }
}

# ... lines omitted ...
```

Tabella 2. Il filtro toserver.pl