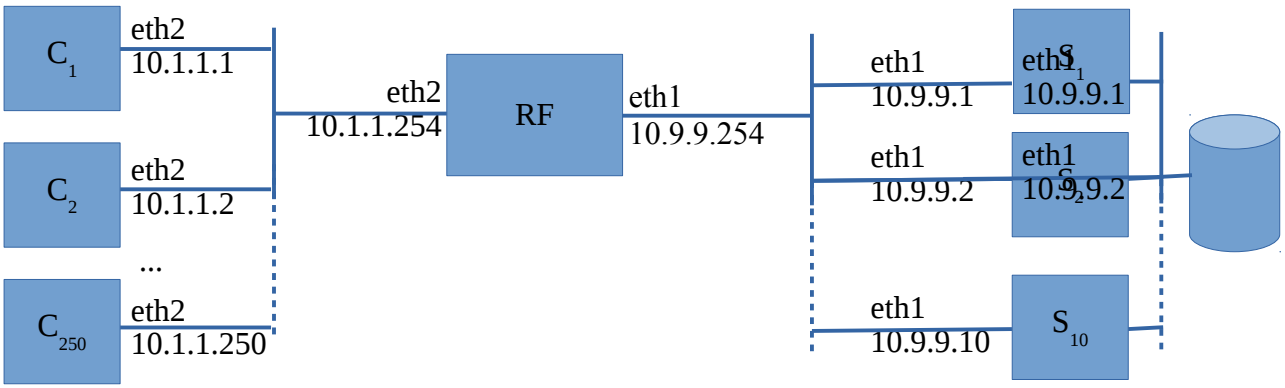
Laboratorio di Amministrazione di Sistemi T

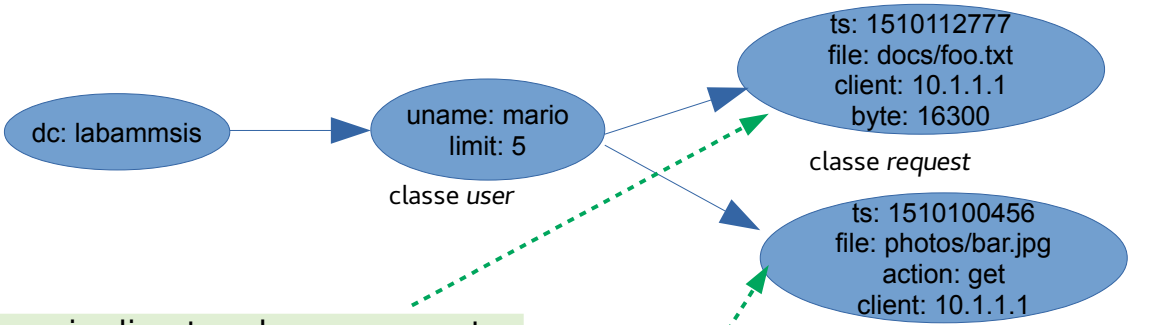
Prova pratica del 3 settembre 2018

Descrizione generale del problema



Il sistema di archiviazione rappresentato in figura prevede un insieme di 250 Client, 10 Server che condividono un filesystem, e un Router-Firewall (RF) tra le rispettive reti. Il RF media le richieste di archiviazione e recupero file inviate dai client ai server.

La base di utenti autorizzata all'uso degli script sui server è registrata sulla directory LDAP ospitata da RF. Le entry servono sia a rappresentare le identità degli utenti che a gestire le richieste da essi effettuate, come nell'esempio di struttura qui riportato:



Esempio di entry che rappresenta un recupero di *file* completato all'istante *ts* e relativo numero di *byte* trasferiti

Esempio di entry che rappresenta una richiesta di recuperare il *file*, inserita in LDAP all'istante *ts* e non ancora soddisfatta

Si trascurino eventuali problemi legati a richieste concorrenti sullo stesso file, o all'esistenza stessa dei file, e si ipotizzi che tutti i percorsi coinvolti nei trasferimenti siano leggibili e scrivibili da tutti gli utenti.

File da consegnare

(tra parentesi quadre la collocazione e il punteggio, non fanno parte del nome)

file.schema.ldif [router - 5] – Definire i tipi di attributo *ts, limit, byte* (intero), *uname, file, action*, *client* (stringhe) e le classi *user* e *request* che li utilizzino come in figura: le entry di classe *user* hanno l'attributo *uname* e il numero massimo di richieste simultanee che l'utente può fare (*limit*); le entry di classe *request* hanno sempre gli attributi *ts, file* e *client*, le richieste si riconoscono dalla presenza dell'attributo *action*, i trasferimenti completati con successo dalla presenza dell'attributo *byte* che registra a fini statistici la quantità di dati trasferiti.

start.sh [client - 21] – Lo script accetta come parametri un'azione (*put* o *get*) e un nome di un file, ed è usato per chiedere a RF di scegliere un server e di abilitare la connessione ssh dal client al server.
Se l'utente ha già un numero di richieste attive pari al valore di *limit*, lo script esce segnalando la situazione, altrimenti la nuova richiesta viene fatta creando una entry LDAP di classe *request* con *ts* corrispondente all'istante di creazione e attributi *action* e *file* presi dai parametri dello script, e attributo *client* contenente l'IP del client su cui lo script è in esecuzione.
Lo script si pone successivamente in attesa che *action* cambi. Non appena al posto dell'azione compare un indirizzo IP di un server, lo script esegue il trasferimento del file tra client e server nella direzione indicata dall'azione passata come parametro, e modifica la entry per togliere l'attributo *action* e inserire l'attributo *byte* col relativo valore.
Se trascorrono 20 secondi senza che compaia l'IP, lo script termina con un errore.
Ogni situazione di terminazione anomala deve essere sia riportata su stdout che registrata sui log di sistema; indicare nei commenti come configurare rsyslog in modo che i messaggi vengano scritti su `/var/log/start.log`.

select.sh [router - 9] – Lo script verifica con la massima rapidità possibile quale tra i 10 server ha il minor numero di connessioni ssh attive provenienti da un qualsiasi client, e ne scrive l'IP su standard output.
Inserire nei commenti come configurare *snmpd* sui server per consentire la verifica.

manage.sh [router - 9] – Lo script sorveglia continuamente il traffico proveniente dalla rete dei client. Ogni volta che rileva la fine di una connessione TCP sulla porta di LDAP, individua le entry che contengono un attributo *action* con valore *put* o *get*, e per ognuna:

- invoca *select.sh* che restituisce un IP di un server
- invoca *fw.sh* per abilitare il traffico ssh tra client e server
- modifica *action* per inserirvi l'IP del server

fw.sh [router - 4] – accetta come parametri un comando, gli ip di un client e di un server;

- se il comando è **open**, autorizza sul packet filter le connessioni ssh dirette dal client al server;
- Se il comando è **close** elimina le regole inserite con **open**;

init.sh [router - 5] – Lo script configura il packet filter del RF perché consenta inizialmente solo il traffico strettamente necessario al funzionamento dei vari script del sistema.