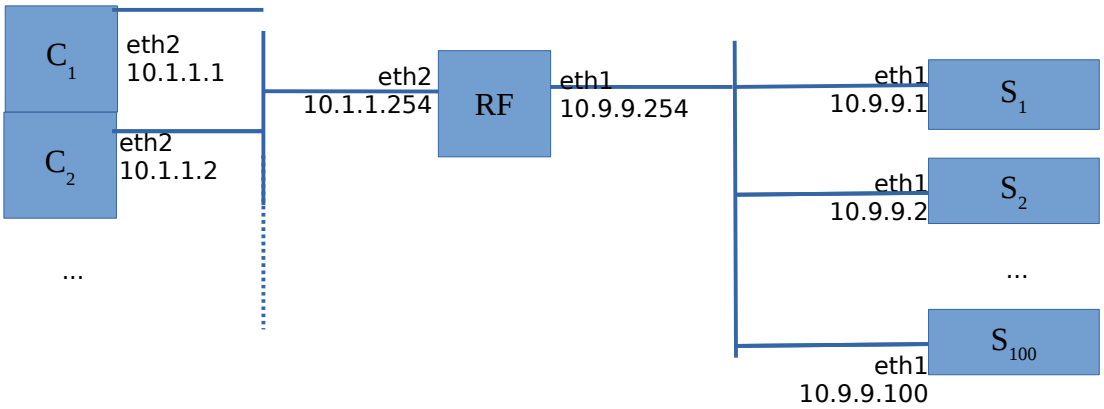
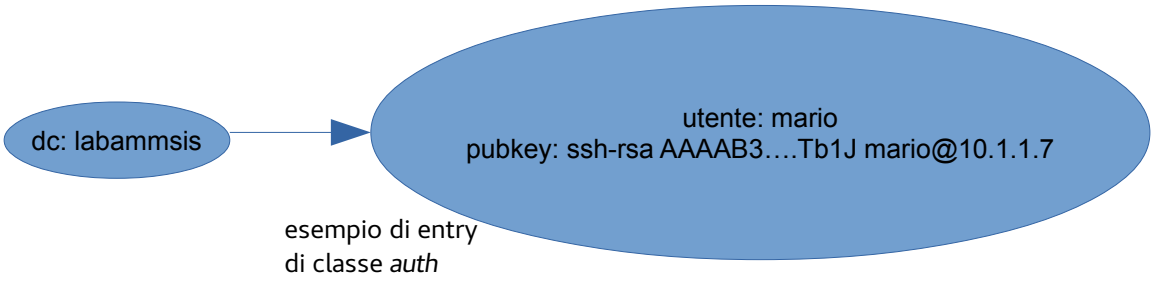
Laboratorio di Amministrazione di Sistemi T

Prova pratica del 17 giugno 2019

Descrizione generale del problema



Un sistema di calcolo distribuito mette a disposizione di un pool di client un gruppo di 100 server.
I client si connettono ai server via SSH, il sistema prevede un meccanismo per distribuire automaticamente le chiavi sui server, che possono essere rimossi e aggiunti (non si consideri il problema di selezionare automaticamente i server presenti e funzionanti).
Le chiavi pubbliche sono conservate in una directory LDAP ospitata dal router, con questa struttura:



esempio di entry
di classe *auth*

File da consegnare

(tra parentesi quadre la collocazione: C = disponibile su tutti i client / R = disponibile sul router / S = disponibile su tutti i server, e il punteggio indicativo; non fanno parte del nome)

auth.schema.ldif [R, 3] – Definire i tipi di attributo *utente*, *pubkey* (stringhe) e la classe *auth* che li contenga obbligatoriamente entrambi.

init.sh [C, 13] - Questo script

- genera una coppia di chiavi SSH con algoritmo RSA e modulo di 2048 bit per l'utente che lo esegue, chiedendo preventivamente (non dal comando di generazione) in quale file salvare la chiave e fornendo automaticamente passphrase vuota
- logga l'indirizzo del client e la chiave pubblica generata sulla facility local1.info
- si pone in attesa che sulla directory LDAP compaia una entry con lo username dell'utente che lo esegue e la relativa chiave, controllando ogni secondo per un massimo di 10 secondi. Se la entry non appare entro il tempo limite, stampa un messaggio d'errore.

Indicare nei commenti come configurare rsyslog sui client e sul router perché i messaggi loggati dallo script vengano scritti nel file **`/var/log/client-keys`** del router.

register.sh [R, 12] - Questo script esamina senza sosta il file **`/var/log/client-keys`** e per ogni nuova chiave che compare

- estrae dalla chiave il nome dell'utente proprietario
- controlla via SNMP che tale utente stia eseguendo *init.sh* sulla macchina da cui proviene il messaggio
- in caso positivo crea la entry LDAP di classe *auth* coi dati richiesti

Indicare nei commenti come configurare snmpd sui client per consentire la verifica

install.sh [R, 11] - Questo script esegue la seguente sequenza di operazioni per ogni server nel range 10.9.9.1-10.9.9.100:

- verifica via SNMP se il server ha il processo *sshd* attivo
- in caso di esito positivo, lancia via ssh sul server il comando *getkeys.sh*

L'intero range deve essere processato nel giro di pochi secondi.
Indicare nei commenti

- come far eseguire automaticamente lo script ogni 2 minuti
- come configurare snmpd sui server per consentire la verifica

getkeys.sh [S, 11] - Questo script interroga la directory LDAP e per ogni entry di classe auth trovata

- verifica se esiste già l'utente in locale
- in caso negativo
  - lo crea
  - ne disattiva la password
  - installa la chiave pubblica nel file appropriato per consentirgli l'accesso senza password

Curare i dettagli delle directory e dei file creati.

fwinit.sh [R, 14] - Questo script configura il packet filter del router per bloccare tutto il traffico non indispensabile per questo scenario. Le connessioni ssh dai client ai server devono essere consentite

limit.sh [C, 9] - Questo script osserva ogni secondo senza mai terminare quante connessioni ssh attive verso i server ci sono sul client su cui viene lanciato. Se ne conta più di 10, configura iptables per bloccare ogni nuova connessione ssh verso i server, se ne conta meno di 8 rimuove il blocco eventualmente inserito in precedenza.