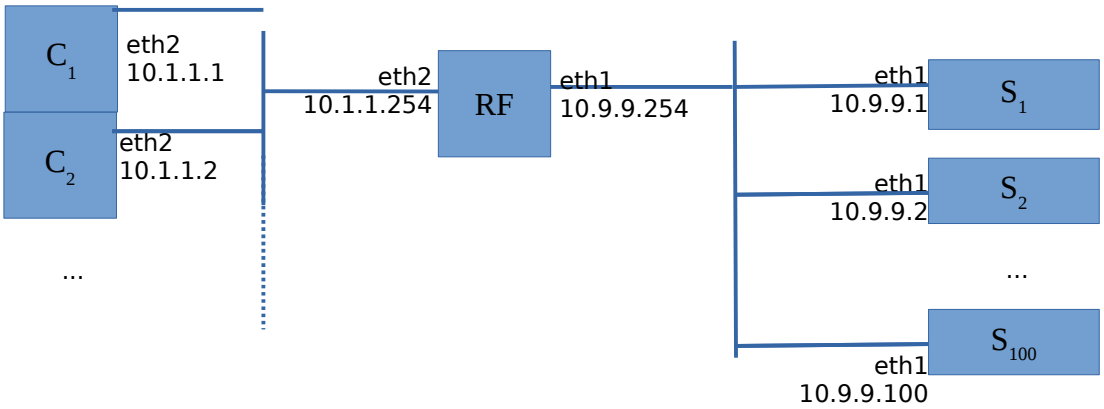
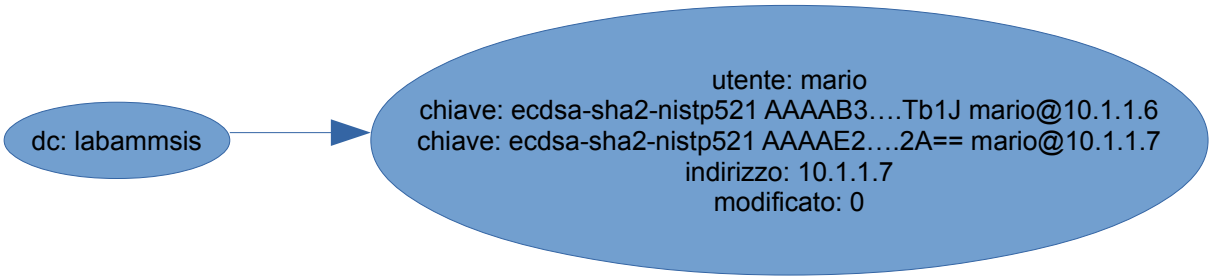
Laboratorio di Amministrazione di Sistemi T

Prova pratica del 9 luglio 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.
Le chiavi pubbliche sono conservate in una directory LDAP ospitata dal router, con questa struttura:



esempio di entry
di classe *store*

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)

keyserver.schema.ldif [R, 3] – Definire i tipi di attributo *utente*, *indirizzo*, *chiave* (stringhe) e *modificato* (intero) la classe *store* che li contenga obbligatoriamente.

addkey.sh [C, 17] - Questo script viene eseguito da utenti standard (non root); esige un nome di file come parametro sulla riga di comando e procede come segue:

- genera una coppia di chiavi SSH con algoritmo ECDSA della massima lunghezza supportata, e le salva col nome passato come parametro fornendo automaticamente passphrase vuota;
- crea (o aggiorna) la entry LDAP dell'utente inserendo (o aggiungendo) la chiave pubblica generata ai valori dell'attributo *chiave* e impostando *modificato* al valore 1 e *indirizzo* all'IP del client su cui è in esecuzione lo script.
- si pone in attesa che nella stessa entry *modificato* assuma valore 0; quando ciò accade stampa un messaggio di conferma su standard output. Se l'attesa si prolunga per più di 90 secondi, termina con un messaggio d'errore.

findservers.sh [R, 12] - Questo script utilizza SNMP per verificare su ognuno dei server nel range 10.9.9.1-10.9.9.100 se il demone SSH è attivo e in ascolto sulla poerta TCP/22. Il controllo dell'intero range dev'essere condotto con la massima rapidità possibile.
Ogni volta che lo script viene eseguito, mette nel file `/tmp/servers.on` gli indirizzi di tutti e soli i server su cui il controllo ha avuto esito positivo che alla precedente esecuzione dello script invece l'avevano fallito.
Indicare nei commenti

- come far eseguire automaticamente lo script ogni 3 minuti;
- come configurare l'agent SNMP sul server per rendere possibile il controllo richiesto.

installkeys.sh [R, 11] - Questo script ogni 10 secondi ispeziona la directory LDAP per individuare le entry di classe *store* che hanno *modificato*==1. Per ognuna installa via SSH su ogni server presente nella lista `/tmp/servers.on` le chiavi pubbliche dell'utente corrispondente alla entry, nel file appropriato per consentirgli l'accesso senza password, e imposta *modificato* a 0.
Si ipotizzi che i path delle home directory siano nella forma /home/USERNAME

track.sh [R, 14] - Questo script osserva senza sosta il traffico TCP tra i client e i server. Ogni volta che osserva una nuova connessione, configura il packet filter per poterne misurare l'attività; ogni volta che osserva il termine di una connessione elimina la regola inserita in precedenza.
Nota: si suggerisce di osservare il funzionamento di *limit.sh* per decidere come strutturare le regole.

limit.sh [R, 12] - Questo script osserva ogni secondo senza mai terminare le regole inserite da *track.sh*. Se un client ha più di 20 connessioni attive verso la rete dei server, o se una qualsiasi connessione tra quelle attive ha generato dall'ultima osservazione più di 1MB di traffico, invoca *embargo.sh* per limitare le attività di tale client verso la rete dei server.

embargo.sh[R, 9] - Questo script necessita di due parametri, il primo dev'essere una stringa di valore *start* o *stop*, il secondo un indirizzo IP di un client.
Se invocato con *start*, configura il packet filter per bloccare ogni nuova connessione TCP dal client specificato verso la rete dei server. Il blocco dev'essere automaticamente rimosso trascorso 1 minuto dal suo inserimento. La rimozione deve avvenire con l'opportuna invocazione dello stesso script con parametro *stop.* Ogni evento di inserimento o rimozione di una regola di blocco dev'essere loggato sul file `/var/log/embargo.log`
Indicare nei commenti come configurare rsyslog per ottenere l'effetto desiderato.