Memòria

Objectius del projecte

Procés servidor (monitor-servidor)

1. Crear un programa anomenat **monitor-servidor.py** que detecti quins ordinadors estan encesos a la xarxa. Als ordinadors encesos instal·lar l'anomena't **monitor-client.py** que serà l'encarregat de transmetre'ns les dades d'us del sistema dels ordinadors remots.
2. Les dades que volem que ens siguin transferides, han de ser en un fitxer anomenat reglas.txt, també especificarem la xarxa i l'interval de temps en el que volem rebre les dades.
3. Rebre l'informacio del monitor-client i guardar-la en una base de dades per al posterior tractament.
4. Crear uns servei `/sbin/service monser [start|stop|restart|reload]`

Procés client (monitor-client)

1. Acceptar la sol·licitud d'informació de monser, obtenir-la i transmetre-li-la.
2. S'ha de poder aturar en rebre un senyal SIGTERM.

Procés de reporter (monitor-reporter)

1. Generar plantilles xhtml a partir de les dades guardades a la base de dades anteriorment.

Acompliment dels objectius

Procés servidor (monitor-servidor)

1. He creat un programa anomenat monitor-servidor.py que mitjançant "*nmap*" es monitoritza la xarxa per saber l'estat dels ordinadors. Posteriorment intenta entrar per "*ssh*" per comprovar si està instal·lat el monitor-client.py, si no és així l'instal·la i posteriorment l'executa.
2. Al fitxer reglas.txt podem especificar amb si/no les dades d'us del sistema que volem que el monitor-client ens transmeti. També s'ha d'especificar la xarxa que es vol monitoritzar i l'interval de temps en segons, en que és vol repetir el procés.
3. Un cop el monitor-client ha estat instal·lat, executat i han estat configurades les regles, es rep l'informació i s'insereix a la taula corresponent de la base de dades. Aquest procés es repeteix segons l'interval de temps escollit.
4. He configurat un servei amb totes les opcions per al monitor-servidor.py

Procés client (monitor-client)

1. S’accepta una cadena de text del monitor-servidor on s’especifiquen les dades que han de ser transferides. S’executen al monitor-client les funcions corresponents a les regles. Després es transfereix tot en una sola cadena de text. Si la cadena de text on s’especifiquen les regles és “*nodata*” el monitor-client.py es tancarà automàticament sense transferir res, ja que el servidor haurà fet el mateix.
2. Si s’envia una senyal SIGUSR1 el monitor-client.py s’atura.

Procés de reporter (monitor-reporter)

1. Es generen plantilles en xhtml especificant paràmetres com les taules desitjades, un interval de data i hora i/o un filtre per ordinadors. Podem demanar plantilles de varies taules i ordinadors a la vegada i es generaran totes amb diferents noms.

Problemes trobats i solucions adoptades

Problema: Volia que el monitor-servidor.py es conectés per “ssh” als ordinadors remots sense tenir que posar la contrasenya, nomes especificant-la al monitor servidor.py.

Solució: El modul de python “paramiko” que permet posar-li el password i ho implementa tot automaticament.

Problema: Transmetre el fitxer monitor-client.py a l’ordinador remot, sense que demani el password com passa quan s’intenta amb la ordre “scp”.

Solució: Amb el protocol TFTP i el mòdul paramiko això es possible.

Problema: Al monitor-reporter.py volia poder dir-li que automaticament em generes varies plantilles en una sola execució especificant-li varies taules o pc concrets pels cuals filtrar els resultats.

Solució: Investigar i pensar una mica i fer funcions per que sigui possible.

Problema: No volia tindre que instal·lar a mà la base de dades, el sistema de logs, configurar el logrotate, crear les carpetes... cada cop que hagués d’instal·lar el programa.

Solució: Crear un script en bash d’instal·lació. Primer de tot crea les carpetes i copia els fitxers necessaris incluint un “readme”. Després configura el sistema de logs si no ho esta ja i igual amb el logrotate. Per últim crea un usuari per la base de dades, crea la base de dades i per acabar les taules. Comprova que tot estigui correcte i et diu si s’ha instal·lat correctament o el problema que ha succeït.

Memòria de les activitats

Els **primers dos dies** em vaig dedicar a fer la connexió entre el monitor-servidor i el monitor-client mitjançant un socket pel port 50020 i que li envies les regles. No hem va costar massa ja que tenia exemples de classe ben fets per poder aprofitar-los. Després vaig fer que el monitor-servidor llegís el fitxer “*regles.txt*”, l’envies i que el monitor-client el rebés.

Els **següents dos dies** hem vaig dedicar a fer les funcions necessàries al monitor-client per que agafes l’informació del ordinadors i la transmetés al monitor-servidor en una sola cadena de text. També vaig fer el DER de la base de dades i la vaig crear junt amb el seu usuari.

A parti del **cinquè** dia fins al seté vaig estar fent el tractament de les dades per tal de separar-les i poder-les afegir a la base de dades juntament amb la data i l'hora en que habien estat transmeses.

Fins aquest **vuité** dia habia estat fent les proves amb una llista fixa de tres ips, llavors vaig fer una funció per monitoritzar la xarxa amb "*nmap*" i obtindre les ips i l'estat dels ordinadors de la xarxa.

Llavors també vaig implementar el sistema de logs per tenir constància dels errors en un log i per tenir constància dels ordinadors encesos o apagats en un altre. Vaig configurar el logrotate per tal de que els log no s'emplenessin.

El **nové** i **desé** dia vaig estar comprovant que tot funcionava i polint aspectes com el control d'errors i també les senyals que podia rebre el monitor-servidor per tal de tancar-lo o per tal de que rellegís el fitxer de regles.

Quan tot anava mes o menys correcte vaig començar a fer l'script d'instal·lació del programa en bash.

L'onzè i **dotzè** dia vaig fer el monitor-reporter.py per generar les plantilles en xhtml a partir de l'informació de la base de dades. Vaig fer un "*css*" per tal de fer mes atractiva la presentació de les taules. Per últim vaig crear el servei anomenat "*monser*" per al monitor-servidor.py, que permet d'encendre'l, aturar-lo, reiniciar-lo i veure'n el seu estat.

El **penúltim** dia de treball vaig fer la documentació corresponent i la comprovació de que tot funcionava correctament en diversos ordinadors que no fossin el meu.

Fonts de consulta i inspiració

Python Documentation - http://www.python.org/doc/

Mòdul Paramiko de python - http://jessenoller.com/2009/02/05/ssh-programming-with-paramiko-completely-different/

Conclusions

Crec que aquest projecte m'ha servit per unificar coneixements i deferents pràctiques que havia après en assignatures separades. Ha estat un molt bon aprenentatge degut a la complexitat d'algunes situacions. He utilitzat tot tipus d'eines com la programació donada aquest segon curs d'ASI sobre els sockets, la creació de bases de dades a partir d'un DER de manera que tot resulti mes senzill i estigui ben fet o la creació de plantilles en format html vàlid. Ha sigut bó haver de buscar per Internet mòduls i procediment necessaris per tal de fer una determinada cosa.

Com opinió personal dir, que al principi no m'agradava el tema de projecte, el trobava avorrit. Però un cop ho vaig entendre tot bé m'hi vaig posar de ple i era un gran repte acabar-lo i fer-lo el millor possible.

Autor: Albert Navas Mallo hisi46997513