Xarxes: Pràctica 1, Programació d'aplicacions de xarxa

César Fernàndez, Enric Guitart, Carles Mateu

Febrer 2019

Finalitat de la pràctica:

- Aprendre a programar aplicacions de xarxes
- Entendre el model client-servidor
- Aprendre a programar i dissenyar un protocol de comunicacions

Introducció

La Gestió de Configuració és una de les àrees funcionals de la Gestió de Xarxa. Entre les seves tasques cal destacar el registre de canvis de configuracions i el control de versions de l'equipament de xarxa. Aquesta tasca és important per els administradors de xarxa ja que els permet minimitzar l'impacte en el servei en el cas d'una fallida hardware d'un equip que impliqui la seva substitució. En aquest cas sol caldrà copiar al nou equip la configuració guardada de l'equip avariat per tenir-lo totalment operatiu amb les mateixes funcionalitats. Per altra banda, el control de versions, els permet recuperar configuracions i funcionalitats anteriors en cas de fallida d'una nova configuració.

En aquesta pràctica es desenvoluparà part d'un sistema de gestió de configuració emprant el model de comunicació client-servidor. El software client és el que s'afegiria a l'equipament de xarxa i proporcionaria noves comandes per la gestió de la configuració. El software servidor caldria instal·lar-lo en el NMS (**N***etwork* **M***anagement* **S***tation*) afegint-hi noves funcionalitats per la Gestió de Configuració.

En les següents seccions es descriuran les característiques i funcions de cadascun dels actors del sistema.

Informació

Gestió de Xarxa: Conjunt de tasques de provisió, operació, manteniment i supervisió dels diferents elements que composen una xarxa de comunicacions. El model ISO ho descompon en cinc àrees funcionals:

- Gestió de configuració
- Gestió de rendiment
- Gestió de comptabilitat
- Gestió de fallides
- Gestió de seguretat

Client

Com s'ha dit en la introducció, el software client s'instal·larà en els equips de xarxa. Aquest software s'ha de comunicar amb el servidor emprant un protocol de comunicacions. Per la pràctica, el software client s'executarà en el PC simulant un equip de xarxa.

Bàsicament un client ha de realitzar tres tasques:

- Registrar-se en el servidor
- Mantenir comunicació periòdica amb el servidor
- Esperar comandes per l'enviament/recepció del seu arxiu de configuració

Informació

NMS: És un dels elements de l'Arquitectura de Gestió de Xarxa. Es tracta d'un equip de propòsit general (servidor) que executa les aplicacions de de xarxa. Els elements de xarxa es comuniquen amb el NMS enviant informació de control i gestió. Amb aquesta informació es pot dur a terme l'anàlisi de dades i la generació d'informes.

Dades de configuració

El software client obtindrà d'un arxiu, emmagatzemat en el seu directori de treball, les dades necessàries per la comunicació amb el servidor. Aquest arxiu ha de proporcionar un paràmetre per línia i en cada línia s'ha d'especificar el nom del paràmetre i el seu valor separats per espais en blanc. El nom per defecte d'aquest arxiu és: `client.cfg`.

Es podrà modificar el nom d'aquest arxiu amb el paràmetre `-c` a l'hora d'executar el client (`-c` <*nom_arxiu*>).

Els paràmetres que ha de contenir l'arxiu són:

- *Nom*: Nom de l'equip (codi alfanumèric de 6 dígits).
- *MAC*: Adreça MAC de l'equip (12 caràcters hexadecimals).
- *Server*: Nom o adreça IP del servidor.
- *Server-port*: Port UDP del servidor per registrar-se.

Registre en el servidor

És la primera tasca que ha de realitzar el client i fins que ho finalitzi satisfactòriament no es prosseguirà amb la resta.

Per la comunicació d'aquesta tasca s'emprarà el protocol UDP amb la següent PDU (**P***rotocol* **D***ata* **U***nit*):

tipus:	unsigned char	char	char	char	char
	Tipus paquet	Nom equip	Adreça MAC	Número aleatori	Dades
bytes:	1	7	13	7	50

Figura 1: Format PDU UDP

Els tipus de PDU en la fase de registre són:

Valor	Mnemònic	Significat
0x00	REGISTER_REQ	Petició de resgistre
0x01	REGISTER_ACK	Acceptació de registre
0x02	REGISTER_NACK	Denegació de registre
0x03	REGISTER_REJ	Rebuig de registre
0x09	ERROR	Error de protocol

Taula 1: Tipus de paquet fase de registre

Procediment

El client ha d'enviar una petició de registre [`REGISTER_REQ`] i esperar la seva acceptació [`REGISTER_ACK`]. En la petició de registre caldrà omplir els camps de la PDU amb les dades de l'equip, el número aleatori a valors zero i el camp de dades buit. El client es mantindrà en aquesta fase fins rebre la confirmació de subscripció o bé per temporització.

El client partirà de l'estat `DISCONNECTED` i en enviar el primer paquet de petició de registre passarà a l'estat `WAIT_REG`.

La temporització que ha d'aplicar el client en un procés de registre es mostra en el diagrama de temps de la figura 2. Inicialment s'enviaran n paquets de petició de registre a intervals de t segons. Si no es rep acceptació se seguiran enviant paquets de petició de registre, però per cada paquet s'incrementarà l'interval en t segons. L'increment es produirà fins que l'interval entre paquets sigui superior a $m * t$ segons, en

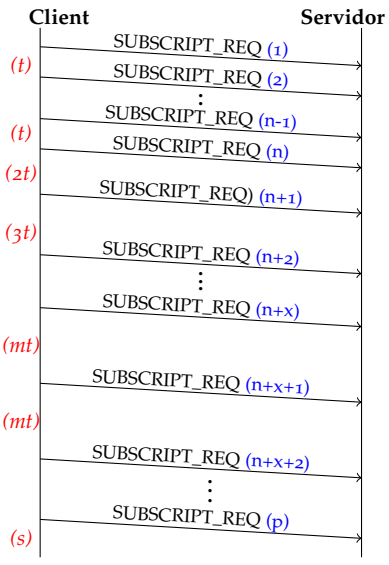

Figura 2: Temporització registre

aquest moment l'interval d'enviament es mantindrà constant en $m * t$ segons. L'enviament de paquets de petició de registre es mantindrà fins haver enviat p paquets. Si no s'ha rebut cap resposta correcta als p paquets, el client esperarà s segons i iniciarà un nou el procés de registre. Passats q processos de registre consecutius sense rebre cap resposta d'acceptació de registre, el client finalitzarà informant que no s'ha pogut contactar amb el servidor.

Per les proves del protocol s'han establert els següents valors dels temporitzadors i els llindars:

$$n= 3,\ t= 2,\ m= 4,\ p= 8,\ s= 5,\ q= 3$$

Si durant el procés de registre es rep un paquet de rebuig [REGISTER_REJ], el client finalitzarà notificant que el registre ha estat rebutjat i indicarà el motiu del rebuig. El client passarà a l'estat DISCONNECTED.

Per altra banda, si la resposta a un paquet [REGISTER_REQ] és un paquet del tipus [REGISTER_NACK], el client finalitzarà el procés de registre actual i efectuarà un nou procés de registre (si no s'han superat els q processos).

Quan el client rebi un paquet [REGISTER_ACK] es considerarà l'equip registrat en el servidor i el client passarà a l'estat REGISTERED. El paquet [REGISTER_ACK] contindrà el número aleatori assignat a l'equip i en el camp data contindrà el port per les comunicacions TCP amb el servidor.

Finalitzada la fase de registre el client iniciarà la resta de funcions.

Mantenir comunicació periòdica amb el servidor

Un cop un equip està registrat en el servidor mantindrà una comunicació periòdica amb el servidor que l'indicarà que es troba operatiu. Per aquesta comunicació s'emprarà la mateixa PDU que en la fase de registre i els tipus de paquets mostrats en la taula 2.

Valor	Mnemònic	Significat
0x10	ALIVE_INF	Enviament d'informació d'alive
0x11	ALIVE_ACK	Confirmació de recepció d'informació d'alive
0x12	ALIVE_NACK	Denegació de recepció d'informació d'alive
0x13	ALIVE_REJ	Rebuig de recepció d'informació d'alive

Taula 2: Tipus de paquet fase manteniment de comunicació amb el servidor

Procediment

El client que passi a l'estat REGISTERED enviarà un paquet [ALIVE_INF] cada r segons i esperarà rebre un paquet [ALIVE_ACK] del servidor que confirmi la recepció correcta del paquet enviat. El client haurà de comprovar que els camps de la PDU rebuda del servidor es corresponen amb les dades obtingudes en la fase de registre, en cas contrari no es farà cas del paquet rebut. En rebre el primer [ALIVE_ACK] correcte del servidor, el client passarà a l'estat ALIVE.

Si el client no rep confirmació de recepció d'alive a u paquets [ALIVE_INF] consecutius interpretarà que hi ha problemes de comunicació amb el servidor, passarà a l'estat DISCONNECTED i iniciarà un nou procés de registre. Els paquets [ALIVE_NACK] que rebi el client no es tindran en compte i es consideraran com no haver rebut resposta del servidor.

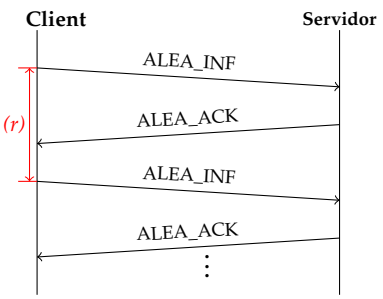

Figura 3: Manteniment de comunicació amb el servidor

La recepció d'un paquet [ALIVE_REJ] en l'estat ALIVE es considerarà com un intent de suplantació d'identitat, el client passarà a l'estat DISCONNECTED i iniciarà un nou procés de registre.

Per les proves del protocol s'han establert els següents valors dels temporitzadors i llindars:

$$r = 3,\ u = 3$$

La figura 3 mostra el diagrama de temps de un procés de manteniment de comunicació amb el servidor sense problemes.

Per el protocol de registre i manteniment de comunicació s'han definit els següents estats en les que pot estar l'equip (client):

- DISCONNECTED: Equip desconnectat (no s'ha endegat el client o bé no s'ha enviat cap petició de registre)
- WAIT_REG: Espera de confirmació de registre
- REGISTERED: Equip registrat
- ALIVE: Enviant i rebent ALIVE

Espera de comandes

Finalitzada la fase de registre el client, a banda d'enviar els paquets [ALIVE_INF], esperarà comandes per la consola del sistema. Les comandes que es poden introduir són:

- send-conf
- get-conf
- quit

Les accions associades a cadascuna de les comandes es descriuen en els següents apartats:

send-conf

En rebre aquesta comanda l'equip enviarà el seu arxiu de configuració al servidor. L'arxiu a enviar ha d'estar en el directori arrel (directori des d'on s'ha executat el client) i per defecte s'anomena boot.cfg. El nom d'aquest arxiu es pot variar emprant el paràmetre **-f** <*arxiu*> a l'hora d'executar el client. Per la transferència de l'arxiu s'emprarà el protocol TCP al port del servidor rebut en el paquet [REGISTER_ACK] de la fase de registre. L'arxiu de configuració és un arxiu de text i el seu enviament es farà línia a línia (un paquet per cada línia de l'arxiu). La longitud màxima d'una línia és de 150 caràcters. La PDU per la transferència d'arxius es mostra en la figura 4.

tipus:	unsigned char	char	char	char	char
	Tipus paquet	Nom equip	Adreça MAC	Número aleatori	Dades
bytes:	1	7	13	7	150

Figura 4: Format PDU TCP

Els tipus de PDU definits per l'enviament de l'arxiu de configuració es detallen en la taula 3.

Taula 3: Tipus de paquet per l'enviament de l'arxiu de configuració

Valor	Mnemònic	Significat
0x20	SEND_FILE	Petició d'enviament d'arxiu de configuració
0x21	SEND_ACK	Acceptació de la petició d'enviament d'arxiu de configuració
0x22	SEND_NACK	Denegació de la petició d'enviament d'arxiu de configuració
0x23	SEND_REJ	Rebuig de la petició d'enviament d'arxiu de configuració
0x24	SEND_DATA	Bloc de dades de l'arxiu de configuració
0x25	SEND_END	Fi de l'enviament de dades de l'arxiu de configuració

Procediment:

El client ha d'enviar un paquet [SEND_FILE] al servidor amb els camps de la PDU als valors corresponents i en el camp data el nom de l'arxiu de configuració local i la seva longitud en bytes separats per una coma. Si la resposta del servidor és un paquet [SEND_ACK] amb els camps de la PDU correctes i en el camp data el nom de l'arxiu que es guardarà en el servidor (<*nom_equip*>.cfg) , es passarà a enviar l'arxiu de configuració amb paquets [SEND_DATA] (un per línia).Per finalitzar l'enviament de l'arxiu s'enviarà un paquet [SEND_END] i es tancarà el canal TCP amb el servidor.

Cas que la resposta al paquet [SEND_FILE] sigui diferent a [SEND_ACK] s'informarà del motiu (indicat en el camp data de la PDU) i es finalitzarà la comunicació TCP amb el servidor.

Si la resposta del servidor triga més de w segons es considerarà que no hi ha una comunicació correcta i es tancarà el canal TCP. Per les proves del protocol s'ha establert:

$$w = 4$$

El procediment correcte de l'enviament d'un arxiu de configuració al servidor es mostra en el diagrama de temps de la figura 5.

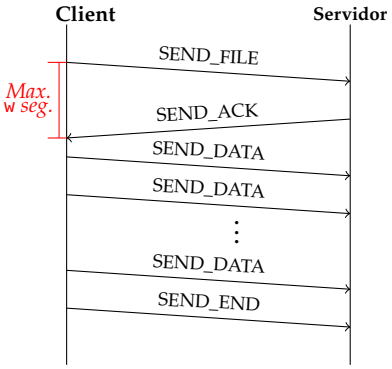

Figura 5: Enviament d'arxiu de configuració al servidor

`get-conf`

Aquesta comanda permet a l'equip obtenir el seu arxiu de configuració emmagatzemat al servidor. Igual que per la comanda `send-conf` s'emprarà el protocol TCP per la transferència de dades i la PDU també serà la mateixa. En aquest cas s'han definit els tipus de PDU que es mostren en la taula 4.

Taula 4: Tipus de paquet per l'obtenció de l'arxiu de configuració

Valor	Mnemònic	Significat
0x30	GET_FILE	Petició d'obtenció d'arxiu de configuració
0x31	GET_ACK	Acceptació d'obtenció d'arxiu de configuració
0x32	GET_NACK	Denegació d'obtenció d'arxiu de configuració
0x33	GET_REJ	Rebuig d'obtenció d'arxiu de configuració
0x34	GET_DATA	Bloc de dades de l'arxiu de configuració
0x35	GET_END	Fi de l'obtenció de l'arxiu de configuració

Procediment:

El procediment inicial per l'obtenció de l'arxiu de configuració és idèntic al de la comanda `send-conf` descrit en l'apartat anterior però canviant el tipus de PDU. Per la

transferència de dades el servidor enviarà els paquets de dades i el client ho emmagatzemarà en l'arxiu local de configuració (`boot.cfg` o bé l'especificat en l'opció **-f** en la crida del client) sobre escrivint el seu contingut.

Igual que en el cas de la comanda `get-conf` , si es la resposta al paquet [`GET_FILE`] és diferent a [`GET_ACK`] o bé la cadència de recepció de paquets és superior a `w` segons, es tancarà el canal de comunicació TCP indicant el motiu del tancament.

En la figura 6 es pot veure el diagrama de temps de l'obtenció de l'arxiu de configuració.

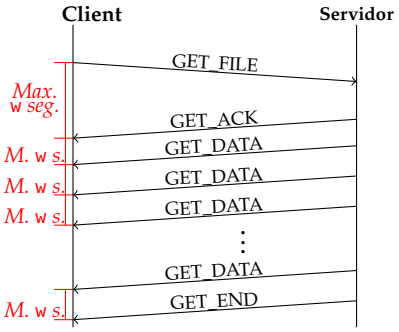

Figura 6: Obtenció de l'arxiu de configuració del servidor

`quit`

Finalitza el client tancant tots els canals de comunicació.

Servidor

Per la seva part, el servidor genèricament ha d'efectuar quatre tasques:

- Obrir un canal UDP per atendre les peticions de registre dels clients, comprovar si són adients i respondre en conseqüència.
- Controlar el manteniment de comunicació amb els equips registrats en el sistema i gestionar el seu estat.
- Obrir un canal TCP i esperar peticions per rebre o enviar els arxius de configuració als clients. Els arxius rebuts s'emmagatzemaran en el directori local.
- Esperar comandes per la consola del sistema.

El servidor haurà d'atendre múltiples clients a l'hora (servidor concurrent).

Igual que el client, el servidor ha de llegir d'un arxiu el seus paràmetres de configuració. Aquest arxiu ha de tenir la mateixa estructura que l'esmentada per el client i per defecte s'anomenarà: `server.cfg`. Els paràmetres que ha de contenir aquest arxiu seran:

- *Nom*: Nom del servidor (codi alfanumèric de 6 dígits).
- *MAC*: Adreça MAC del servidor (12 caràcters hexadecimals).
- *UDP-port*: Port UDP per on espera rebre les comunicacions UDP.
- *TCP-port*: Port TCP per on espera rebre les connexions TCP.

El servidor ha de tenir emmagatzemats en un arxiu el nom i la MAC de tots els equips autoritzats a emprar el sistema. Aquests arxiu es llegirà a l'inici del servidor i es mantindrà en memòria amb una estructura de dades que li permeti controlar l'estat dels clients, els codis aleatoris, adreça IP, etc. El nom de l'arxiu és `equips.dat` i contindrà en cada línia el nom i la MAC, separats per un espai, de cada equip autoritzat a emprar el sistema de gestió de configuració.

En iniciar-se el servidor haurà d'obrir el dos canals de comunicació, UDP i TCP, i esperar les comunicacions dels clients. Per cada client, i cada protocol, haurà de destinar un procés que l'atengui.

Es descriuen a continuació cadascuna de les quatre tasques que ha d'efectuar el servidor:

Atendre peticions de registre

En rebre una petició de registre, el servidor haurà de comprovar si l'equip està autoritzat per emprar el sistema. Si no està autoritzat s'enviarà un paquet [REGISTER_REJ] amb tots els camps de la PDU a valor zeros i en el camp [data] el motiu del rebuig. Cas d'estar el client autoritzat, haurà de comprovar si les dades de l'equip són correctes. En cas afirmatiu es generarà un nombre aleatori per assignar a la comunicació amb l'equip i s'enviarà un paquet [REGISTER_ACK] amb les dades de la PDU corresponents i el port TCP en el camp [data]. Si es rep una petició de registre per un equip que està REGISTERED o ALIVE es comprovarà que les dades de l'equip siguin correctes i, en cas afirmatiu, s'enviarà un paquet [REGISTER_ACK] amb el nombre aleatori generat en el primer procés de registre.

Si en el paquet de petició de registre hi ha discrepàncies amb el número aleatori o es diferent de zeros en el primer paquet, el servidor enviarà un paquet [REGISTER_NACK] amb tots els camps de la PDU a valor zeros i en el camp [data] el motiu de la denegació. Igualment si hi ha discrepàncies amb l'adreça IP de l'equip, en paquets posteriors al primer, es respondrà amb un paquet [REGISTER_NACK] i els camps de la PDU als valors descrits anteriorment.

Mentre no finalitzi el registre l'equip es mantindrà en l'estat DISCONNECTED i sol passarà a l'estat REGISTERED després d'enviar un paquet [REGISTER_ACK].

Control del manteniment de comunicació

El servidor ha de controlar que tots els equips registrats en el sistema mantinguin una comunicació periòdica per constatar que estan operatius. En aquest sentit contestarà amb un paquet [ALIVE_ACK] als paquets [ALIVE_INF] conformats correctament: Nom, MAC, IP, nombre aleatori i estat correctes.

Els paquets [ALIVE_INF] rebuts amb equips no autoritzats o equips que no estan registrats es respondran amb un paquet [ALIVE_REJ] en el que tots els camps de la PDU contindran valor zero i en el camp [data] el motiu del rebuig.

Si en el paquet [ALIVE_INF] hi ha discrepàncies amb l'adreça IP i/o amb el número aleatori es respondrà amb un paquet [ALIVE_NACK] indicant en el camp data el motiu de la denegació i la resta de camps a valor zero.

En el servidor un equip registrat pot canviar d'estat per tres motius:

- *Recepció primer ALIVE*: En rebre el primer paquet [ALIVE_INF] d'un equip en estat REGISTERED aquest passarà a l'estat ALIVE.
- *Temporització primer ALIVE*: Si un equip en estat REGISTERED no rep un paquet [ALIVE_INF] abans de j intervals d'enviament d'ALIVE l'equip passarà a l'estat DISCONNECTED. Per les proves del protocol s'ha establert j=2.
- *Pèrdua d'ALIVES*: Si un equip en l'estat ALIVE deixa de rebre k ALIVES consecutius passarà a l'estat DISCONNECTED. Per les proves del protocol el valor k s'ha establert en 3.

Espera de peticions per el canal TCP

Tal com s'ha comentat en la introducció del servidor, per cada connexió TCP s'haurà de crear un procés que l'atengui (servidor concurrent). El comportament de cada procés dependrà del tipus de petició rebuda del client:

[SEND_CONF]: El client vol enviar el seu arxiu de configuració per emmagatzemar-lo al servidor. Caldrà comprovar que es tracta d'un client autoritzat, en l'estat correcte i amb les dades addicionals correctes. En funció del resultat de la comprovació es podran respondre tres tipus de paquets:

- [SEND_ACK]: Dades de l'equip correctes. S'enviaran els camps de la PDU amb els seus corresponents valors (del servidor) i en el camp data s'enviarà el nom amb el que s'emmagatzemarà localment l'arxiu de configuració de l'equip:<*nom_equip*>.cfg. Un cop enviat aquest paquet s'esperarà rebre paquets [SEND_DATA] per cadascuna de les línies de l'arxiu de configuració. Aquesta informació s'anirà guardant en l'arxiu local descrit anteriorment. La recepció d'un paquet [SEND_END] indicarà el final de la transmissió per part del client. Es tancarà l'arxiu de configuració escrit i també el canal de comunicació TCP.

  Si l'interval de recepció de paquets és superior a w segons es considerarà que hi ha problemes en la comunicació i es tancarà el canal TCP. En la figura 7 es mostra el diagrama de temps d'aquest procediment.

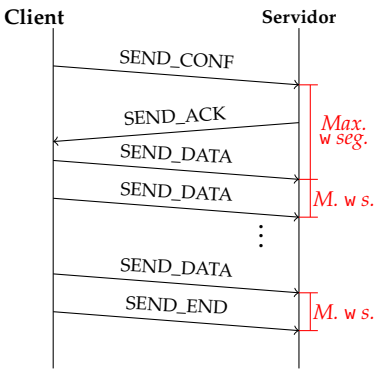

Figura 7: Enviament d'arxiu de configuració al servidor

- [SEND_NACK]: Dades addicionals de l'equip incorrectes. Implica error en l'adreça IP rebuda o bé error en el número aleatori. En la PDU de resposta es ficaran els camps de l'adreça MAC i el del número aleatori a valor zeros, el camp del nom de l'equip a valor buit i en el camp de dades el motiu de la denegació de recepció de l'arxiu de configuració. Un cop enviat el paquet es tancarà el canal de comunicació TCP.

  Aquest paquet també es pot enviar si arriba una petició d'un equip que ja està efectuant una operació amb el seu arxiu de configuració (send o get).

- [SEND_REJ]: Discrepància amb les dades principals de l'equip. Si el nom de l'equip o la MAC no es corresponen amb un equip autoritzat o bé l'equip no està registrat, s'enviarà aquest tipus de paquet amb tots els camps de l'adreça MAC i el del número aleatori a valor zeros, el camp del nom de l'equip a valor buit i en el camp de dades el motiu del rebuig de recepció de l'arxiu de configuració. Igual que en el cas anterior, després d'enviar aquest paquet es tancara el canal de comunicació TCP.

[GET_CONF]: El client vol obtenir l'arxiu de configuració emmagatzemat en el servidor. Igual que en el cas anterior caldrà comprovar si les dades rebudes són correctes i també es poden respondre tres tipus de paquets:

- [GET_ACK]: Dades de l'equip correctes. S'enviaran els camps de la PDU amb els seus valors corresponents (del servidor) i en el camp data s'enviarà el nom amb el que està emmagatzemat l'arxiu de configuració de l'equip:<*nom_equip*>.cfg. Un cop enviat aquest el servidor enviarà l'arxiu de configuració emprant paquets [GET_DATA], un per línia. Per finalitzar l'enviament de l'arxiu enviarà un paquet [GET_END] i es tancarà el canal de comunicació TCP.

- [GET_NACK]: Dades addicionals de l'equip incorrectes. Implica error en l'adreça IP rebuda o bé error en el número aleatori. En la PDU de resposta es ficaran els camps de l'adreça MAC i el del número aleatori a valor zeros, el camp del nom de l'equip a valor buit i en el camp de dades el motiu de la denegació d'enviament de l'arxiu de configuració. Un cop enviat el paquet es tancarà el canal de comunicació TCP.

  Aquest paquet també es pot enviar si arriba una petició d'un equip que ja està efectuant una operació amb el seu arxiu de configuració (send o get).

- [GET_REJ]: Discrepància amb les dades principals de l'equip. Si el nom de l'equip o la MAC no es corresponen amb un equip autoritzat o bé l'equip no està registrat, s'enviarà aquest tipus de paquet amb tots els camps de l'adreça MAC i el del número aleatori a valor zeros, el camp del nom de l'equip a valor buit i en el camp de dades el motiu del rebuig d'enviament de l'arxiu de configuració.

  Aquest paquet també s'enviarà si en el servidor no existeix cap arxiu de configuració de l'equip.

  Igual que en el cas anterior, després d'enviar aquest paquet es tancara el canal de comunicació TCP.

Espera de comandes per consola

En la consola del servidor inicialment sol es podran introduir dues comandes:

- `list`: Visualitza per pantalla una taula amb tots els equips autoritzats en el sistema amb les següents dades: Nom, MAC i estat. Per els equips que estan registrats també ha de mostrar l'adreça IP des de la que s'ha connectat i el nombre aleatori assignat en la fase de registre.
- `quit`: Finalitza l'execució del servidor.

Implementació

La implementació de la pràctica s'haurà de fer en dos llenguatges:

- **Servidor**: Python (2.x [superior a 2.6])
  - Sol es poden emprar llibreries estàndard
  - De les llibreries de comunicacions s'han d'emprar `socket` i `select`
  - S'ha d'emprar *shebang* (`#!`) amb la versió de de Python

- **Client**: Ansi C (C99).
  - Sol es poden emprar llibreries estàndard
  - S'emprarà l'eina `make` per compilar
  - S'ha de compilar amb `gcc` i les opcions `-ansi -pedantic -Wall`

Tant el client com el servidor han de permetre els següents paràmetres d'entrada:

-d : Fer *debug*. S'ha de presentar per consola un missatge per cada esdeveniment significatiu. Aquests missatges han de permetre fer un seguiment del funcionament de la implementació i del protocol.

-c <*arxiu*>: Arxiu de dades de configuració del software. Especifica el nom de l'arxiu d'on es llegiran les dades necessàries per la comunicació entre client i servidor. Si no s'especifica aquest paràmetre el nom per defecte serà especificat en la documentació.

El client ha de permetre un paràmetre addicional:

-f <*arxiu*>: Arxiu de configuració de l'equip. Especifica el nom de l'arxiu de configuració de l'equip de xarxa. Si no s'especifica aquest paràmetre serà el valor per defecte: `boot.cfg`.

Per la seva banda el servidor també té un paràmetre addicional:

-u <*arxiu*>: Arxiu d'equips autoritzats. Especifica el nom de l'arxiu que conté el nom i l'adreça MAC dels equips autoritzats en el sistema. Per defecte el seu valor ha de ser: `equips.dat`.

Independentment de l'opció de *debug* esmentada anteriorment, tant el client com el servidor han de mostrar per consola un missatge amb els canvis d'estat dels clients.

Per facilitar el desenvolupament de la pràctica, es proposa dividir-la en dues parts i cada part dividir-la en quatre fases progressives:

1. Implementació del client
    (a) Fase de registre
    (b) Fase de manteniment de la comunicació
    (c) Fase d'enviament d'arxiu de configuració
    (d) Fase de recepció d'arxiu de configuració
2. Implementació del servidor
    (a) Fase de registre
    (b) Fase de manteniment de la comunicació
    (c) Fase de recepció d'arxiu de configuració
    (d) Fase d'enviament d'arxiu de configuració

Per poder implementar el client sense la dependència d'haver de construir un servidor, es proporcionarà un servidor bàsic per avaluar el desenvolupament del client. Igualment es proporcionaran un client bàsic per avaluar el desenvolupament del servidor. Tant el client com el servidor permetran efectuar tests bàsics de funcionament de cadascuna de les parts.

Lliurament

Documentació

La pràctica és individual, cada alumne haurà de lliurar un únic arxiu amb les següents característiques:

- *Nom*: **XARXES-P1-***nom* (On *nom* son els dos cognoms de l'autor separats per guionet (-))
- *Format*: ZIP, TGZ o TBZ.
- *Contingut*:
  - Codi font degudament comentat pel seu seguiment.
  - Arxius necessaris pel correcte funcionament del codi (configuracions clients, servidor, etc).
  - Un informe en format PDF que inclogui:
    1. L'estructura, a nivell esquemàtic (diagrama de blocs), del client i del servidor.
    2. L'estratègia emprada per el manteniment de la comunicació, tant en el client com en el servidor.
    3. Un diagrama d'estats del protocol implementat sobre UDP (registre i manteniment de la comunicació).
    4. Totes aquelles consideracions que cregueu oportunes.

    Aquest document ha de complir les següents condicions:

    * Una bona estructuració.
    * Un format dels elements de text correctes.
    * Un contingut clar i una redacció correcta (no s'acceptaran errades ortogràfiques ni abreviatures o símbols per substituir paraules).
- S'ha de lliurar al campus virtual (apartat Activitats) i no s'acceptarà per cap altre mitjà.

> **Important**
>
> **La documentació que no compleixi tots aquests requisits no serà avaluada.**

Termini

El termini per rebre la documentació relacionada amb aquesta pràctica finalitza el dia **23 d'abril de 2019**.

Avaluació

Per a que una pràctica pugui ser avaluada haurà de complir els següents requisits:

- El codi del client no ha de donar cap tipus d'error en el procés de compilació ni en la execució.
- El codi del servidor s'ha d'executar sense errors.
- S'executarà el servidor i dos clients en la mateixa màquina i, com a mínim, han d'arribar a la fase de manteniment de la comunicació (els clients enviant periòdicament ALIVE i el servidor responent a cadascun d'ells). El servidor ha de poder llistar els equips autoritzats en els sistema (comanda `list`) i tant en el servidor com en el client s'ha de poder emparar (i funcionar) la comanda `quit`.
- Complir tots els requeriments de l'apartat *Documentació* de la secció **Lliurament**

La avaluació de la pràctica es realitzara en un sistema Linux (Fedora fc29.x86_64).