Apéndice C

Recogida de registros

Índice del capítulo

C.1. Instalación y configuración de *syslog-ng*

C.1.1. Instalación de *syslog-ng*

Procedemos en este punto y en los siguientes al proceso de instalación y configuración del manejador de eventos y mensajes del sistema *syslog-ng*.

La instalación se puede realizar cómodamente con el siguiente comando gracias a que se provee la utilidad en formato *paquete Debian*:

```
#> apt-get install syslog-ng
```

Si así lo deseamos, los contenidos del paquete se podrán consultar como sigue:

```
#> dpkg -L syslog-ng
```

C.1.2. Configuración de *syslog-ng*

La configuración de *syslog-ng* se realiza mediante su archivo de configuración en `/etc/syslog-ng/syslog-ng.conf` [51, 52]. El archivo de configuración es extenso pero tiene una estructura clara además de densamente comentada, y explica la filosofía de funcionamiento de *syslog-ng* a la vez que nos introduce sus

parámetros de configuración. Lo presentamos desglosando su estructura en sus secciones principales:

1. En primer lugar aparecen una serie de opciones comunes al funcionamiento del programa.

`/etc/syslog-ng/syslog-ng.conf` (I - opciones)

```
#
# Configuration file for syslog-ng under Debian
#
# attempts at reproducing default syslog behavior

# the standard syslog levels are (in descending order of priority):
# emerg alert crit err warning notice info debug
# the aliases "error", "panic", and "warn" are deprecated
# the "none" priority found in the original syslogd configuration is
# only used in internal messages created by syslogd


######
# options

options {
        # disable the chained hostname format in logs
        # (default is enabled)
        chain_hostnames(0);

        # the time to wait before a died connection is re-established
        # (default is 60)
        time_reopen(10);

        # the time to wait before an idle destination file is closed
        # (default is 60)
        time_reap(360);

        # the number of lines buffered before written to file
        # you might want to increase this if your disk isn't catching with
        # all the log messages you get or if you want less disk activity
        # (say on a laptop)
        # (default is 0)
        #sync(0);

        # the number of lines fitting in the output queue
        log_fifo_size(2048);

        # enable or disable directory creation for destination files
        create_dirs(yes);

        # default owner, group, and permissions for log files
        # (defaults are 0, 0, 0600)
        #owner(root);
        group(adm);
        perm(0640);

        # default owner, group, and permissions for created directories
        # (defaults are 0, 0, 0700)
        #dir_owner(root);
        #dir_group(root);
        dir_perm(0755);

        # enable or disable DNS usage
        # syslog-ng blocks on DNS queries, so enabling DNS may lead to
        # a Denial of Service attack
        # (default is yes)
        use_dns(no);

        # maximum length of message in bytes
        # this is only limited by the program listening on the /dev/log
            Unix
        # socket, glibc can handle arbitrary length log messages, but --
            for
        # example -- syslogd accepts only 1024 bytes
```

```
        # (default is 2048)
        #log_msg_size(2048);
};
```

2. A continuación se definen una secuencia de fuentes de los datos que *syslog-ng* manejará. Estos orígenes serán llamadas al sistema de programas locales o remotos, las cuales se recibirán mediante una conexión TCP o UDP.

`/etc/syslog-ng/syslog-ng.conf` (II - fuentes)

```
######
# sources

# all known message sources
source s_all {
        # message generated by Syslog-NG
        internal();
        # standard Linux log source (this is the default place for the
            syslog()
        # function to send logs to)
        unix-stream("/dev/log");
        # messages from the kernel
        file("/proc/kmsg" log_prefix("kernel: "));
        # use the above line if you want to receive remote UDP logging
            messages
        # (this is equivalent to the "-r" syslogd flag)
        # udp();
};
```

En este caso en vez de establecerse distintas fuentes hay definida una única fuente, `s_all`, que recibe mensajes desde una serie de puntos distintos. Se podrían haber especificado una serie de fuentes distintas, como `s_internal`, `s_unix_stream` o `s_file`, cada una de ellas con un punto de entrada distinto igualmente.

3. En tercer lugar se enumeran una serie de destinos o sumideros de los datos que el programa recoja durante su ejecución. Los destinos podrán ser ficheros de texto, programas o incluso salidas hacia otras máquinas mediante flujos TCP o UDP.

`/etc/syslog-ng/syslog-ng.conf` (III - destinos)

```
######
# destinations

# some standard log files
destination df_auth { file("/var/log/auth.log"); };
destination df_syslog { file("/var/log/syslog"); };
destination df_cron { file("/var/log/cron.log"); };
destination df_daemon { file("/var/log/daemon.log"); };
destination df_kern { file("/var/log/kern.log"); };
destination df_lpr { file("/var/log/lpr.log"); };
destination df_mail { file("/var/log/mail.log"); };
destination df_user { file("/var/log/user.log"); };
destination df_uucp { file("/var/log/uucp.log"); };

# these files are meant for the mail system log files
# and provide re-usable destinations for {mail,cron,...}.info,
# {mail,cron,...}.notice, etc.
destination df_facility_dot_info { file("/var/log/$FACILITY.info"); };
destination df_facility_dot_notice { file("/var/log/$FACILITY.notice"); };
destination df_facility_dot_warn { file("/var/log/$FACILITY.warn"); };
destination df_facility_dot_err { file("/var/log/$FACILITY.err"); };
destination df_facility_dot_crit { file("/var/log/$FACILITY.crit"); };

# these files are meant for the news system, and are kept separated
```

```
# because they should be owned by "news" instead of "root"
destination df_news_dot_notice { file("/var/log/news/news.notice" owner("
    news")); };
destination df_news_dot_err { file("/var/log/news/news.err" owner("news"))
    ; };
destination df_news_dot_crit { file("/var/log/news/news.crit" owner("news
    ")); };

# some more classical and useful files found in standard syslog
    configurations
destination df_debug { file("/var/log/debug"); };
destination df_messages { file("/var/log/messages"); };

# pipes
# a console to view log messages under X
destination dp_xconsole { pipe("/dev/xconsole"); };

# consoles
# this will send messages to everyone logged in
destination du_all { usertty("*"); };
```

Aquí, por ejemplo, se define el destino `df_syslog` que será el que nos creará el archivo de *syslog-ng* en `/var/log/syslog`.

4. Lo siguiente que encontramos en el archivo de configuración son una lista de filtros que permiten identificar con mayor exactitud el origen del mensaje y nos permiten realizar una separación de los mensajes.

`/etc/syslog-ng/syslog-ng.conf` (IV - filtros)

```
######
# filters

# all messages from the auth and authpriv facilities
filter f_auth { facility(auth, authpriv); };

# all messages except from the auth and authpriv facilities
filter f_syslog { not facility(auth, authpriv); };

# respectively: messages from the cron, daemon, kern, lpr, mail, news,
    user,
# and uucp facilities
filter f_cron { facility(cron); };
filter f_daemon { facility(daemon); };
filter f_kern { facility(kern); };
filter f_lpr { facility(lpr); };
filter f_mail { facility(mail); };
filter f_news { facility(news); };
filter f_user { facility(user); };
filter f_uucp { facility(uucp); };

# some filters to select messages of priority greater or equal to info,
    warn,
# and err
# (equivalents of syslogd's *.info, *.warn, and *.err)
filter f_at_least_info { level(info..emerg); };
filter f_at_least_notice { level(notice..emerg); };
filter f_at_least_warn { level(warn..emerg); };
filter f_at_least_err { level(err..emerg); };
filter f_at_least_crit { level(crit..emerg); };

# all messages of priority debug not coming from the auth, authpriv, news,
    and
# mail facilities
filter f_debug { level(debug) and not facility(auth, authpriv, news, mail)
    ; };

# all messages of info, notice, or warn priority not coming form the auth,
# authpriv, cron, daemon, mail, and news facilities
filter f_messages {
        level(info,notice,warn)
```

```
                and not facility(auth,authpriv,cron,daemon,mail,news);
};

# messages with priority emerg
filter f_emerg { level(emerg); };

# complex filter for messages usually sent to the xconsole
filter f_xconsole {
    facility(daemon,mail)
        or level(debug,info,notice,warn)
        or (facility(news)
                and level(crit,err,notice));
};
```

En este caso, fijándonos de nuevo en el *syslog-ng*, se define un filtro `f_syslog`.

5. Por último, unas definiciones de registro que asociarán las fuentes, filtros y destinos en unidades de registro.

`/etc/syslog-ng/syslog-ng.conf` (y V - registros)

```
######
# logs
# order matters if you use "flags(final);" to mark the end of processing
    in a
# "log" statement

# these rules provide the same behavior as the commented original syslogd
    rules

# auth,authpriv.*                 /var/log/auth.log
log {
        source(s_all);
        filter(f_auth);
        destination(df_auth);
};

# *.*;auth,authpriv.none          -/var/log/syslog
log {
        source(s_all);
        filter(f_syslog);
        destination(df_syslog);
};

# this is commented out in the default syslog.conf
# cron.*                          /var/log/cron.log
#log {
#        source(s_all);
#        filter(f_cron);
#        destination(df_cron);
#};

# daemon.*                        -/var/log/daemon.log
log {
        source(s_all);
        filter(f_daemon);
        destination(df_daemon);
};

# kern.*                          -/var/log/kern.log
log {
        source(s_all);
        filter(f_kern);
        destination(df_kern);
};

# lpr.*                           -/var/log/lpr.log
log {
        source(s_all);
        filter(f_lpr);
        destination(df_lpr);
};
```

```
# mail.*                          -/var/log/mail.log
log {
        source(s_all);
        filter(f_mail);
        destination(df_mail);
};

# user.*                          -/var/log/user.log
log {
        source(s_all);
        filter(f_user);
        destination(df_user);
};

# uucp.*                          /var/log/uucp.log
log {
        source(s_all);
        filter(f_uucp);
        destination(df_uucp);
};

# mail.info                       -/var/log/mail.info
log {
        source(s_all);
        filter(f_mail);
        filter(f_at_least_info);
        destination(df_facility_dot_info);
};

# mail.warn                       -/var/log/mail.warn
log {
        source(s_all);
        filter(f_mail);
        filter(f_at_least_warn);
        destination(df_facility_dot_warn);
};

# mail.err                        /var/log/mail.err
log {
        source(s_all);
        filter(f_mail);
        filter(f_at_least_err);
        destination(df_facility_dot_err);
};

# news.crit                       /var/log/news/news.crit
log {
        source(s_all);
        filter(f_news);
        filter(f_at_least_crit);
        destination(df_news_dot_crit);
};

# news.err                        /var/log/news/news.err
log {
        source(s_all);
        filter(f_news);
        filter(f_at_least_err);
        destination(df_news_dot_err);
};

# news.notice                     /var/log/news/news.notice
log {
        source(s_all);
        filter(f_news);
        filter(f_at_least_notice);
        destination(df_news_dot_notice);
};


# *.=debug;\
#         auth,authpriv.none;\
#         news.none;mail.none     -/var/log/debug
```

```
log {
        source(s_all);
        filter(f_debug);
        destination(df_debug);
};


# *.=info;*.=notice;*.=warn;\
#          auth,authpriv.none;\
#          cron,daemon.none;\
#          mail,news.none          -/var/log/messages
log {
        source(s_all);
        filter(f_messages);
        destination(df_messages);
};

# *.emerg                         *
log {
        source(s_all);
        filter(f_emerg);
        destination(du_all);
};


# daemon.*;mail.*;\
#          news.crit;news.err;news.notice;\
#          *.=debug;*.=info;\
#          *.=notice;*.=warn       |/dev/xconsole
log {
        source(s_all);
        filter(f_xconsole);
        destination(dp_xconsole);
};
```

Aquí es donde, finalmente, se asocian la fuente `s_all`, el filtro `f_syslog` y el destino `df_syslog` en una misma regla para crear el archivo `/var/log/syslog` del sistema.

Si hacemos ahora las siguientes suposiciones sobre la estructura de la red:

- El sistema de monitorización se aloja en el sistema con dirección IP 192.168.100.24.
- El servicio *syslog-ng* del sistema de monitorización atenderá los mensajes en el puerto 514
- El único sistema generador de registros se aloja en la dirección IP 192.168.100.24, es decir, está en el mismo sistema que el receptor de los mensajes.

Visto cuál es el funcionamiento de *syslog-ng* y la estructura, será fácil entender el uso que le daremos:

- En los sistemas remotos, desde donde queremos recibir los registros, estableceremos los siguientes parámetros adicionales al resto de su configuración para habilitar el envío sobre TCP:

  `/etc/syslog-ng/syslog-ng.conf` (líneas a añadir en generadores de registros)

```
destination dest_tcp {
        tcp("192.168.100.24" port(514));
};
```

```
log {
        source(s_all);
        destination(dest_tcp);
};
```

Con esta configuración haremos que todos los registros se manden por TCP hacia el puerto 514 de la máquina 192.168.100.24.

- En el sistema de monitorización que recibe los registros de los demás será necesario establecer lo siguiente:

`/etc/syslog-ng/syslog-ng.conf` (líneas a añadir en receptores de registros)

```
options {
        ...
        check_hostname(yes);
        keep_hostname(yes);
};

source s_tcp {
        tcp(ip(192.168.100.24) port(514) keep-alive(yes));
};

destination hosts {
        file("/var/log/HOSTS/$HOST/$YEAR/$MONTH/$DAY/
             $FACILITY$YEAR$MONTH$DAY"
        owner(root) group(root) perm(0600) dir_perm(0700) create_dirs(yes)
             );
};
destination df_syslog_common {
        file("/var/log/syslog_common");
};

log {
        source(s_all);
        filter(f_syslog);
        destination(df_syslog_common);
};

log {
        source(s_tcp);
        filter(f_syslog);
        destination(df_syslog_common);
};

log {
        source(s_all);
        destination(hosts);
};

log {
        source(s_tcp);
        destination(hosts);
};
```

Con esta configuración haremos lo siguiente:

- Las opciones `check_hostname` y `keep_hostname` evitarán que se pierda el nombre del sistema que generó el registro, es decir, estas opciones serán las que nos mantendrán la autoría del mensaje.
- Establecemos una fuente, `s_tcp`, que escucha las llegadas de los mensajes de los sistemas remotos.
- Especificamos un destino, `hosts`, que cree una estructura de directorios a partir de `/var/log/HOSTS` en la que podremos diferencias los mensajes por fecha, máquina y facilidad de la máquina que lo originó.

- Especificamos un destino, `df_syslog_common` para la aglutinación conjunta de todos los `syslog`, en el archivo `/var/log/syslog_common`.
- Se hacen unas especificaciones de registros para que tanto los mensajes locales como remotos se almacenen tanto en la estructura de directorios de `/var/log/HOSTS` como en el archivo `/var/log/syslog_common`.

Por tanto, nos quedaría funcionando el sistema según se indica en la Figura C.1.

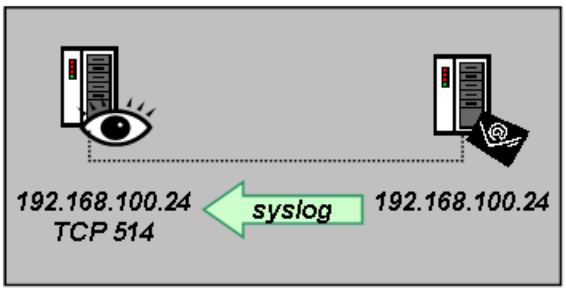

Figura C.1: Esquema de funcionamiento de *syslog-ng*

Si aceptamos esta configuración, será importante reconfigurar *logrotate* para que el archivo `syslog_common` sea también rotado junto al resto de registros del sistema. Para ellos atendemos a su configuración en `/etc/logrotate.d/syslog-ng`. En dicho archivo, podremos añadir al final las siguientes líneas para que nuestro nuevo `syslog_common` sea tratado igual que el viejo `syslog`:

`/etc/logrotate.d/syslog-ng` (líneas a añadir)

```
    /var/log/syslog_common {
       rotate 7
       daily
       compress
5      postrotate
          /etc/init.d/syslog-ng reload >/dev/null
       endscript
    }
```

C.2. Consideraciones de seguridad: transporte mediante *stunnel*

C.2.1. Instalación de *stunnel*

Como se adelantó en la sección 6.4, "*Consideraciones de seguridad: stunnel y openssl*", de este documento, *stunnel* nos va a permitir realizar un transporte seguro de los mensajes de los registros del sistema entre las distintas aplicaciones *syslog-ng* de los sistemas de nuestra red. Para ello utilizaremos un transporte sobre TCP de los mensajes de *syslog-ng*, algo imposible de hacer con *syslogd*

ya que tan sólo admitía su transporte sobre UDP, además de una configuración específica que se verá a lo largo de esta sección.

Al igual que la mayoría de las utilidades estudiadas en este documento, ***stunnel*** se proporciona en un *paquete Debian* y su instalación se realiza cómodamente:

```
#> apt-get install stunnel
```

Los contenidos del paquete se podrán consultar como sigue:

```
#> dpkg -L stunnel
```

C.2.2. Configuración de *stunnel*

Para realizar el transporte seguro de los mensajes de *syslog-ng* sobre *stunnel*, este último tendrá que ser configurado en dos configuraciones distintas, una como sistema cliente y otra como sistema servidor (nuestro sistema de monitorización) al que se conectarán los distintos clientes emisores de los mensajes. Para ello es necesario conocer algunos de los parámetros de configuración de esta utilidad, que comentamos en el Cuadro C.1. Para más detalle sobre estos y otros parámetros puede consultarse el manual de la aplicación [56].

Parámetro	**Descripción**
-d `[host:]port`	Modo demonio. *stunnel* escuchará peticiones en la dirección (opcional) y puerto indicados. Si no se especifica dirección escuchará en INADDR_ANY.
-r `[host:]port`	Destino. Indican a *stunnel* la dirección (opcional) y puerto a los que mandar la información. Si no se especifica dirección los mandará a INADDR_LOOPBACK.
-c	Modo cliente. Usar este parámetro hará que la comunicación hacia el destino (especificada con -r) sea la comunicación SSL.
-A `certfile`	Fichero de la Autoridad de Certificación.
-p `pemfile`	Fichero del certificado del servidor.
-v `level`	Nivel de la certificación: 1 Comprobar la validez del certificado si se presenta un certificado. 2 Comprueba la validez del certificado del cliente (exigiendo uno). 3 Comprueba la validez del certificado del cliente contra un certificado local. Por defecto no hay verificación.

Cuadro C.1: Parámetros de *stunnel*

Si queremos hacer ahora que *syslog-ng* utilice esta comunicación segura [53], será necesario lo siguiente:

1. Establecer la siguiente configuración en la sistema servidor, que será el mismo sistema donde *syslog-ng* esperará recibir los mensajes de los clientes (es decir, nuestro sistema de monitorización):

`/etc/syslog-ng/syslog-ng.conf` (líneas a añadir en recolector de registros)

```
source s_tcp_ssl {
        tcp(ip(127.0.0.1) port(514) keep-alive(yes));
};

destination df_tcp_ssl_file { file("/var/log/syslog_tcp_ssl"); };

log {
        source(s_tcp_ssl);
        filter(f_syslog);
        destination(df_tcp_ssl_file);
};
```

   Con esto haremos que los mensajes recibidos por la fuente `df_tcp_ssl_file` sean enviados hacia un nuevo fichero en `/var/log/syslog_tcp_ssl`. En lugar de hacer esto podríamos realizar una salida hacia una estructura de directorios o hacia alguna de las otras que se mostraron en la sección C.1.1, "*Instalación de syslog-ng*".

2. En los sistemas remotos, que albergan los *syslog-ng* generadores de mensajes, estableceremos la siguiente configuración:

`/etc/syslog-ng/syslog-ng.conf` (líneas a añadir en emisores de registros)

```
destination df_tcp_ssl_server { tcp("127.0.0.1" port(51400)); };

log {
        source(s_all);
        destination(df_tcp_ssl_server);
};
```

   Con esta nueva definición de registro hacemos que todos los mensajes que se generen a partir de la fuente `s_all` (en principio la única fuente existente por lo que todos los mensajes provienen de ella) se envíen hacia el nuevo destino `df_tcp_ssl_server`. Nótese que si esta configuración y la anterior fuesen establecidas sobre un mismo *syslog-ng* los mensajes se entregarían a través del bucle local.

3. Por último, tenemos que arrancar los demonios de *stunnel*. En primer lugar en la máquina servidora donde está configurado *syslog-ng* para atender los mensajes de los clientes:

```
#> stunnel -d 192.168.100.24:5140 -r 127.0.0.1:514
```

   Y en la máquina cliente:

```
#> stunnel -c -d 127.0.0.1:51400 -r 192.168.100.24:5140
```

Con esto tenemos ya una comunicación segura entre nuestro cliente y nuestro servidor. Para clarificar todo esto, veamos la Figura C.2.

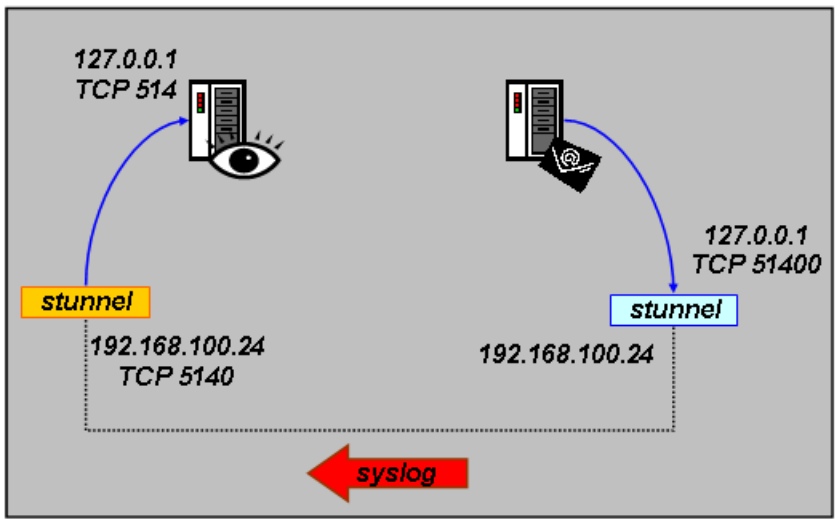

Figura C.2: Esquema de funcionamiento de *syslog-ng* con *stunnel*

C.2.3. Autenticación mediante *stunnel*: el paquete *openssl*

Una vez establecidos los túneles de comunicación segura, ahora lo ideal sería establecer una autenticación entre el cliente y el servidor para asegurarnos que sólo nuestros clientes van a mandarnos sus registros y que clientes no deseados nos llenen nuestros *logs* con falsos registros. Esto lo podemos realizar gracias a los certificados y firmas digitales, y vamos a ver a continuación cómo usarlos.

Nos basaremos en el paquete ***openssl*** que se suministra para **Debian Sarge** en su versión 0.9.7e-3. Su instalación es la siguiente:

```
#> apt-get install openssl
```

Nos centraremos sólo en los pasos necesarios para hacer uso de los certificados que es capaz de crear, sin estudiar el paquete en detalle sino sirviéndonos únicamente de lo necesario del mismo. Para trabajar con él, nos trasladaremos al siguiente directorio:

```
#> cd /usr/lib/ssl/misc
```

En el se encuentran las herramientas que utilizaremos para fabricar los certificados [54, 55]. Los pasos que realizamos son los siguientes:

1. Crear una llave de Autoridad de Certificación. Durante el proceso se nos pedirá distinta información:

```
#> ./CA.sh -newca

CA certificate filename (or enter to create)

Making CA certificate ...
Generating a 1024 bit RSA private key
.....................................................++++++
.....................................++++++
writing new private key to './demoCA/private/./cakey.pem'
```

```
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:ES
State or Province Name (full name) [Some-State]:Sevilla
Locality Name (eg, city) []:Sevilla
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Escuela Superior
      de Ingenieros
Organizational Unit Name (eg, section) []:Autoridad Certificadora de la
     Escuela Superior de Ingenieros
Common Name (eg, YOUR name) []:Certificador
Email Address []:certificador@esi.us.es
```

Esto nos crea una estructura de directorios en `/usr/lib/ssl/misc/demoCA/`. Debemos tomar nota del *PEM pass phrase* que nos hará falta con posterioridad.

2. Crear la llave privada del servidor y un certificado:

```
#> ./CA.pl -newreq-nodes

Generating a 1024 bit RSA private key
.++++++
......++++++
writing new private key to 'newreq.pem'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:ES
State or Province Name (full name) [Some-State]:Sevilla
Locality Name (eg, city) []:Sevilla
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Escuela Superior
      de Ingenieros
Organizational Unit Name (eg, section) []:Laboratorio de Software de
     Fuentes Abiertas
Common Name (eg, YOUR name) []:rmicmir
Email Address []:rmicmir@sfa.es

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:password
An optional company name []:company
```

Este comando nos creará el archivo `/usr/lib/ssl/misc/newreq.pem`.

3. Firmar, contra nuestra propia Autoridad de Certificación, el certificado `newreq.pem` recién creado . Para ello necesitaremos el *PEM pass phrase* que usamos anteriormente al crear la llave de la Autoridad de Certificación.

```
#> ./CA.sh -sign

Using configuration from /usr/lib/ssl/openssl.cnf
Enter pass phrase for ./demoCA/private/cakey.pem:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 1 (0x1)
```

```
        Validity
            Not Before: Dec 13 11:28:21 2005 GMT
            Not After : Dec 13 11:28:21 2006 GMT
        Subject:
            countryName               = ES
            stateOrProvinceName       = Sevilla
            localityName              = Sevilla
            organizationName          = Escuela Superior de Ingenieros\
            organizationalUnitName    = Laboratorio de Software de Fuentes
                 Abiertas
            commonName                = rmicmir
            emailAddress              = rmicmir@sfa.es
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                6E:05:79:E1:40:DE:14:FF:72:21:CB:A3:5B:FF:DC:B5:96:F3:04:CD
            X509v3 Authority Key Identifier:
                keyid:7F:45:5C:B4:91:B4:E9:66:7D:4B:2E:D2:7A:CD:48:77:17:D5
                     :C2:92
                DirName:/C=ES/ST=Sevilla/L=Sevilla/O=Escuela Superior de
                     Ingenieros/OU=Autoridad Certificadora de la Escuela
                     Superior de Ingenieros/CN=Certificador/emailAddress=
                     certificador@esi.us.es
                serial:86:E6:23:2B:AF:64:E0:A9

Certificate is to be certified until Dec 13 11:28:21 2006 GMT (365 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 1 (0x1)
        Signature Algorithm: md5WithRSAEncryption
        Issuer: C=ES, ST=Sevilla, L=Sevilla, O=Escuela Superior de
             Ingenieros, OU=Autoridad Certificadora de la Escuela Superior
             de Ingenieros, CN=Certificador/emailAddress=certificador@esi.us
             .es
        Validity
            Not Before: Dec 13 11:28:21 2005 GMT
            Not After : Dec 13 11:28:21 2006 GMT
        Subject: C=ES, ST=Sevilla, L=Sevilla, O=Escuela Superior de
             Ingenieros , OU=Laboratorio de Software de Fuentes Abiertas, CN
             =rmicmir/emailAddress=rmicmir@sfa.es
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:a2:3c:4b:78:b5:6b:d3:e7:01:7b:a4:90:6d:93:
                    45:51:22:94:1d:22:8f:11:c3:d7:f7:2f:c2:ae:4d:
                    46:8f:28:c1:af:96:98:27:83:40:ae:70:cb:73:38:
                    2c:f3:fb:7b:1a:83:f6:3c:22:44:07:ca:93:d7:dd:
                    bc:67:af:7e:db:7b:64:72:da:43:3b:a4:4e:47:54:
                    7e:0f:ac:6d:32:52:dc:7a:72:0d:cc:4c:f7:8c:9a:
                    f5:9a:0c:f6:b6:e4:6b:db:ed:0e:2f:01:b3:29:6d:
                    e6:21:ed:c3:16:5c:fd:f7:a3:46:5b:c8:45:3b:50:
                    f9:69:77:f7:61:ad:d6:27:d1
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                6E:05:79:E1:40:DE:14:FF:72:21:CB:A3:5B:FF:DC:B5:96:F3:04:CD
            X509v3 Authority Key Identifier:
                keyid:7F:45:5C:B4:91:B4:E9:66:7D:4B:2E:D2:7A:CD:48:77:17:D5
                     :C2:92
```

```
                DirName:/C=ES/ST=Sevilla/L=Sevilla/O=Escuela Superior de
                    Ingenieros/OU=Autoridad Certificadora de la Escuela
                    Superior de Ingenieros/CN=Certificador/emailAddress=
                    certificador@esi.us.es
                serial:86:E6:23:2B:AF:64:E0:A9

    Signature Algorithm: md5WithRSAEncryption
        30:09:37:a8:c7:05:a6:b9:fb:eb:b1:58:0b:bb:be:c1:2c:a0:
        97:ee:d5:47:2b:34:6a:69:90:22:8f:cc:68:5c:41:32:a7:98:
        7b:0a:f8:73:da:e7:b1:21:53:00:56:fa:ff:47:9b:b9:d6:0f:
        6b:21:14:13:20:ff:0b:11:45:6f:41:d4:2a:90:e5:33:bb:8c:
        94:aa:01:62:f7:a4:d3:ed:8f:6a:d4:ef:93:55:e4:64:df:44:
        c8:56:b7:92:b9:89:1d:da:b5:31:8e:56:aa:ef:15:3b:dc:bd:
        8b:5b:84:ca:f9:36:15:80:48:a9:da:2f:3e:02:7b:0a:66:17:
        14:ef
-----BEGIN CERTIFICATE-----
MIIEkTCCA/qgAwIBAgIBATANBgkqhkiG9w0BAQQFADCB3zELMAkGA1UEBhMCRVMx
EDAOBgNVBAgTB1NldmlsbGExEDAOBgNVBAcTB1NldmlsbGExJzAlBgNVBAoTHkVz
Y3VlbGEgU3VwZXJpb3IgZGUgSW5nZW5pZXJvczFFMEMGA1UECxM8QXV0b3JpZGFk
IENlcnRpZmljYWRvcmEgZGUgbGEgRXNjdWVsYSBTdXBlcmlvciBkZSBJbmdlbml
cm9zMRUwEwYDVQQDEwxDZXJ0aWZpY2Fkb3IxJTAjBgkqhkiG9w0BCQEWFmNlcnRp
ZmljYWRvckBlc2kudXMuZXMwHhcNMDUxMjEzMTEyODIxWhcNMDYxMjEzMTEyODIx
WjCBwjELMAkGA1UEBhMCRVMxEDAOBgNVBAgTB1NldmlsbGExEDAOBgNVBAcTB1Nl
dmlsbGExKDAmBgNVBAoTH0VzY3VlbGEgU3VwZXJpb3IgZGUgSW5nZW5pZXJvcyAx
NDAyBgNVBAsTK0xhYm9yYXRvcmlvIGRlIFNvZnR3YXJlIGRlIEZ1ZW50ZXMgQWJp
ZXJ0YXMxEDAOBgNVBAMTB3JtaWNtaXIxHTAbBgkqhkiG9w0BCQEWDnJtaWNtaXJA
c2ZhLmVzMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCiPEt4tWvT5wF7pJBt
k0VRIpQdIo8Rw9f3L8KuTUaPKMGvlpgng0CucMtzOCzz+3sag/Y8IkQHypPX3bxn
r37be2Ry2kM7pE5HVH4PrG0yUtx6cg3MTPeMmvWaDPa25Gvb7Q4vAbMpbeYh7cMW
XP33o0ZbyEU7UPlpd/dhrdYn0QIDAQABo4IBdjCCAXIwCQYDVR0TBAIwADAsBglg
hkgBhvhCAQ0EHxYdT3BlblNTTCBHZW5lcmF0ZWQgQ2VydGlmaWNhdGUwHQYDVR0O
BBYEFG4FeeFA3hT/ciHLo1v/3LWW8wTNMIIBFgYDVR0jBIIBDTCCAQmAFH9FXLSR
tOlmfUsu0nrNSHcX1cKSoYHlpIHiMIHfMQswCQYDVQQGEwJFUzEQMA4GA1UECBMH
U2V2aWxsYTEQMA4GA1UEBxMHU2V2aWxsYTEnMCUGA1UEChMeRXNjdWVsYSBTdXBl
cmlvciBkZSBJbmdlbmllcm9zMUUwQwYDVQQLEzxBdXRvcmlkYWQgQ2VydGlmaWNh
ZG9yYSBkZSBsYSBFc2N1ZWxhIFN1cGVyaW9yIGRlIEluZ2VuaWVyb3MxFTATBgNV
BAMTDENlcnRpZmljYWRvcjElMCMGCSqGSIb3DQEJARYWY2VydGlmaWNhZG9yQGVz
aS51cy5lc4IJAIbmIyuvZOCpMA0GCSqGSIb3DQEBBAUAA4GBADAJN6jHBaa5++ux
WAu7vsEsoJfu1UcrNGppkCKPzGhcQTKnmHsK+HPa57EhUwBW+v9Hm7nWD2shFBMg
/wsRRW9B1CqQ5TO7jJSqAWL3pNPtj2rU
```

4. Unir el certificado firmado y el certificado del servidor:

```
#> cat newreq.pem newcert.pem > server_stunnel.pem
```

   Para mayor comodidad, copiaremos el archivo creado en `/etc/stunnel/server_stunnel.pem`.

5. Distribuir el archivo `/usr/lib/ssl/misc/demoCA/cacert.pem` a todos los clientes que se vayan a conectar mediante *stunnel*. Suponemos que se guardará en todos los clientes en `/etc/stunnel/cacert.pem`

6. En el servidor, habrá que arrancar *stunnel* de la siguiente forma:

```
#> stunnel -d 192.168.100.24:5140 -r 127.0.0.1:514 -p /etc/stunnel/
     server_stunnel.pem
```

7. En los clientes, se arrancará *stunnel* como sigue:

```
#> stunnel -c -d 127.0.0.1:514 -r 192.168.100.24:5140 -A /etc/stunnel/
     cacert.pem -v2
```

Tras esto, nuestro servidor de *stunnel* que espera las conexiones de los clientes sólo aceptará a los clientes que tienen un certificado válido asociado a su propia certificación. Su funcionamiento se indica en la Figura C.3.

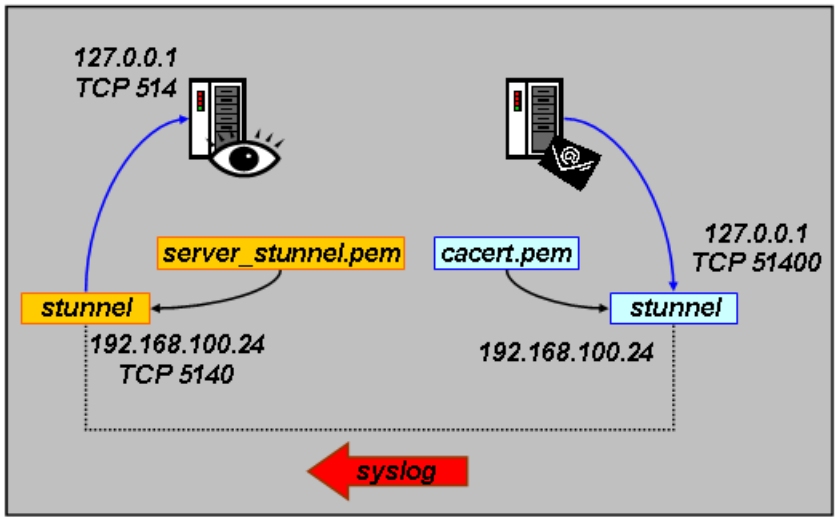

Figura C.3: Esquema de funcionamiento de *syslog-ng* con *stunnel* y certificados de *openssl*