```
    str = X509_NAME_oneline(X509_get_issuer_name(server_cert), 0, 0);
    printf("       issuer: %s\n\n", str);

    /* certificate verification would happen here */

    X509_free(server_cert);

    /*---------------------------------------------------------*/
    /* (7) handshake complete --- send HTTP request over SSL */
    /*---------------------------------------------------------*/

    sprintf(host_header,"Host: %s:443\r\n",hostname);
    strcpy(outbuf,"GET / HTTP/1.0\r\n");
    strcat(outbuf,host_header);
    strcat(outbuf,"Connection: close\r\n");
    strcat(outbuf,"\r\n");

    err = SSL_write(ssl, outbuf, strlen(outbuf));
    shutdown (sd, 1); /* send EOF to server */

    printf("(7) sent HTTP request over encrypted channel:\n\n%s\n",outbuf);

    /*-------------------------------------------------*/
    /* (8) read back HTTP response from the SSL stack */
    /*-------------------------------------------------*/

    err = SSL_read(ssl, inbuf, sizeof(inbuf) - 1);
    inbuf[err] = '\0';
    printf ("(8) got back %d bytes of HTTP response:\n\n%s\n",err,inbuf);

    /*-----------------------------------------------*/
    /* (9) all done, so close connection & clean up */
    /*-----------------------------------------------*/

    SSL_shutdown(ssl);
    close (sd);
    SSL_free (ssl);
    SSL_CTX_free (ctx);

    printf("(9) all done, cleaned up and closed connection\n\n");
}
```

这个例子是在 Sun Solaris 上面编译运行的，但它说明了 SSL 在很多 OS 平台上的工作原理。由于 OpenSSL 提供了一些强有力的特性，包括所有加密、密钥以及证书管理在内的整个程序都可以在一个几页左右的 C 程序中实现。

331 ~ 332

下面按部分分析下这个程序。

- 程序的顶端包含了一些用于支持 TCP 联网和 SSL 的支撑文件。
- 第 1 部分通过调用 SSL_CTX_new 创建了本地上下文，以记录握手参数及与 SSL 连接有关的其他状态。
- 第 2 部分通过 Unix 的 gethostbyname 函数将（由一个命令行变元提供的）输入主机名转换成了 IP 地址。其他平台可能会通过其他方式来提供这项功能。