如何在Ubuntu 20.04 中為Nginx 創建自簽名SSL 證書

聆聽世界的魚 Linux公社 2021-12-16 08:30

收錄於話題
#Nginx 4 #服務器 2 #SSL 1

點擊上方藍字 ● 關注Linux公社

介紹

TLS或傳輸層安全性及其前身**SSL**代表安全套接字層，是用於保護和加密計算機網絡流量的Web 協議。

使用TLS/SSL，服務器可以安全地在服務器和客戶端之間發送流量，而不會出現消息被外部攔截的可能性。證書系統還幫助用戶驗證他們正在連接的站點的身份。

在本指南中，我們將向您展示如何在Ubuntu 20.04 服務器上設置用於Nginx Web 服務器的自簽名SSL 證書。(見https://www.linuxmi.com/ubuntu-20-04-nginx-ssl-certificate.html)

注意：自簽名證書將加密您的服務器和任何客戶端之間的通信。但是，由於它不是由Web 瀏覽器中包含的任何受信任的證書頒發機構簽署的，因此用戶無法使用該證書來自動驗證您的服務器的身份。

如果您沒有與服務器關聯的域名，並且加密的Web 界面不是面向用戶的，則自簽名證書可能是合適的。如果您*確實*有域名，則在許多情況下最好使用CA 簽署的證書。您可以通過Let's Encrypt 項目了解如何設置免費的可信證書（見https://www.linuxmi.com/ubuntu-20-04-lets-encrypt-nginx.html）。

先決條件

在開始之前，您應該有一個配置了sudo權限並啟用防火牆的非root 用戶。

您還需要安裝Nginx Web 服務器。按照我們在Ubuntu 20.04上安裝Nginx的指南進行操作（見https://www.linuxmi.com/linux-nginx-web-server.html）。確保完成本教程的**第5 步**並設置服務器塊，因為這是測試Nginx 是否能夠使用您的自簽名證書加密連接所必需的。

步驟1 – 創建SSL 證書

TLS/SSL 通過公共證書和私鑰的組合發揮作用。SSL 密鑰在服務器上保密，並對發送給客戶端的內容進行加密。SSL 證書與請求內容的任何人公開共享。它可用於解密由相關 SSL 密鑰簽名的內容。

您可以在單個命令中使用OpenSSL 創建自簽名密鑰和證書對：

- sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/ssl/private/nginx-selfsigned.key -out /etc/ssl/certs/nginx-selfsigned.crt

以下是此命令每個部分的作用的細分：

- `sudo`: 該`sudo`命令允許`sudo`組成員臨時將他們的權限提升到另一個用戶（默認為超級用戶或**root**用戶）的權限。在這種情況下這是必要的，因為我們正在`/etc/`目錄下創建證書和密鑰對，只能由**root**用戶或其他特權帳戶訪問。
- `openssl`：這是用於創建和管理OpenSSL 證書、密鑰和其他文件的基本命令行工具。
- `req`：此子命令指定我們要使用X.509 證書籤名請求(CSR) 管理。“X.509”是SSL 和 TLS 對其密鑰和證書管理所遵循的公鑰基礎設施標準。我們想創建一個新的X.509 證書，所以我們使用這個子命令。
- `-x509`：這通過告訴實用程序我們要製作自簽名證書而不是像通常發生的那樣生成證書籤名請求來進一步修改前一個子命令。
- `-nodes`：這告訴OpenSSL 跳過使用密碼保護我們的證書的選項。當服務器啟動時，我們需要Nginx 能夠在沒有用戶干預的情況下讀取文件。密碼可以防止這種情況發生，因為我們必須在每次重新啟動後輸入它。
- `-days 365`：此選項設置證書被視為有效的時間長度。我們在這裡設置了一年。
- `-newkey rsa:2048`：這指定我們要同時生成新證書和新密鑰。我們沒有在上一步中創建簽署證書所需的密鑰，因此我們需要將其與證書一起創建。該`rsa:2048`部分告訴它製作一個2048 位長的RSA 密鑰。
- `-keyout`：這一行告訴OpenSSL 在哪裡放置我們正在創建的生成的私鑰文件。
- `-out`：這告訴OpenSSL 在哪裡放置我們正在創建的證書。

如前所述，這些選項將創建密鑰文件和證書。運行此命令後，系統會詢問您一些有關服務器的問題，以便將信息正確嵌入到證書中。

適當填寫提示。**最重要的一行是請求`Common Name (e.g. server FQDN or YOUR name)`. 您需要輸入與您的服務器相關聯的域名，或者更有可能是您服務器的公共IP 地址。**

整個提示將如下所示：

```
Country Name (2 letter code) [AU]:US
State or Province Name (full name) [Some-State]:New York
Locality Name (eg, city) []:New York City
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Bouncy Cas
Organizational Unit Name (eg, section) []:Ministry of Water Slides
Common Name (e.g. server FQDN or YOUR name) []:server_IP_address
Email Address []:admin@your_domain.com
```

您創建的兩個文件都將放置在該/etc/ssl目錄的相應子目錄中。

在使用OpenSSL 時，您還應該創建一個強大的Diffie-Hellman (DH) 組，用於與客戶端完全正向保密（perfect forward secrecy）。

您可以通過鍵入以下內容來執行此操作：

- 須藤 openssl dhparam -out /etc/nginx/dhparam.pem 4096

這將需要一段時間，但完成後，您將擁有一個強大的DH 組，/etc/nginx/dhparam.pem該組將在配置期間使用。

步驟2 – 配置Nginx 以使用SSL

現在您的/etc/ssl目錄下的密鑰和證書文件已經創建，您需要修改Nginx 配置以利用它們。

首先，您將創建一個配置片段，其中包含有關SSL 密鑰和證書文件位置的信息。然後，您將創建一個具有強SSL 設置的配置片段，將來可以與任何證書一起使用。最後，您將使用您創建的兩個配置片段調整您的Nginx 服務器塊，以便可以適當地處理SSL 請求。

這種配置Nginx 的方法將允許您保持乾淨的服務器塊並將常見的配置段放入可重用的模塊中。

創建指向SSL 密鑰和證書的配置片段

首先，使用您喜歡的文本編輯器在/etc/nginx/snippets目錄中創建一個新的Nginx 配置片段。以下示例使用nano.

要正確區分此文件的用途，請將其命名為self-signed.conf：

- 須藤納米 /etc/nginx/snippets/self-signed.conf

在此文件中，您需要將ssl_certificate指令設置為您的證書文件和ssl_certificate_key關聯的密鑰。這將如下所示：

/etc/nginx/snippets/self-signed.conf

```
ssl_certificate /etc/ssl/certs/nginx-selfsigned.crt;
ssl_certificate_key /etc/ssl/private/nginx-selfsigned.key;
```

添加這些行後，保存文件並退出編輯器。如果您曾經nano編輯過該文件，則可以通過按CTRL + X、Y、來進行編輯ENTER。

使用強加密設置創建配置片段

接下來，您將創建另一個片段來定義一些SSL 設置。這將為Nginx 設置強大的SSL 密碼套件，並啟用一些有助於確保服務器安全的高級功能。

您設置的參數可以在以後的Nginx 配置中重複使用，因此您可以給文件一個通用名稱：

- 須藤納米/etc/nginx/snippets/ssl-params.conf

為了安全地設置Nginx SSL，我們將調整來自Cipherlist.eu的建議。Cipherlist.eu 是一種有用且易於理解的資源，可用於了解用於流行軟件的加密設置。

注意：來自Cipherlist.eu 的這些建議設置提供了強大的安全性。有時，這是以更高的客戶端兼容性為代價的。如果您需要支持舊客戶端，則可以通過單擊頁面上標有“Yes, give me a ciphersuite that works with legacy / old software.（是的，給我一個適用於舊軟件/舊軟件的密碼套件）”的鏈接來訪問該列表。如果需要，您可以用下一個示例代碼塊的內容替換該列表。

選擇使用哪種配置在很大程度上取決於您需要支持的內容。他們都將提供極大的安全性。

出於您的目的，請完整復制提供的設置，但首先，您需要進行一些小的修改。

首先，為上游請求添加首選的DNS 解析器。我們將在本指南中使用Google 的(8.8.8.8和8.8.4.4)。

其次，註釋掉設置嚴格傳輸安全標頭的行。在取消對這一行的註釋之前，您應該花點時間閱讀HTTP 嚴格傳輸安全性或HSTS，特別是關於“預加載”功能的內容。預加載HSTS 可提供更高的安全性，但如果意外啟用或錯誤啟用，也會產生深遠的負面影響。

將以下內容添加到您的ssl-params.conf代碼段文件中：

/etc/nginx/snippets/ssl-params.conf

```
ssl_protocols TLSv1.3;
ssl_prefer_server_ciphers on;
ssl_dhparam /etc/nginx/dhparam.pem;
ssl_ciphers EECDH+AESGCM:EDH+AESGCM;
ssl_ecdh_curve secp384r1;
```

```
ssl_session_timeout  10m;
ssl_session_cache shared:SSL:10m;
ssl_session_tickets off;
ssl_stapling on;
ssl_stapling_verify on;
resolver 8.8.8.8 8.8.4.4 valid=300s;
resolver_timeout 5s;
# Disable strict transport security for now. You can uncomment the fo
# line if you understand the implications.
#add_header Strict-Transport-Security "max-age=63072000; includeSubDo
add_header X-Frame-Options DENY;
add_header X-Content-Type-Options nosniff;
add_header X-XSS-Protection "1; mode=block";
```

因為您使用的是自簽名證書，所以不會使用SSL 裝訂。Nginx 將輸出警告並禁用我們的自簽名證書的裝訂，但隨後將繼續正常運行。

完成後按“CTRL + X然後”Y和“保存”並關閉文件ENTER。

調整Nginx 配置以使用SSL

現在您有了代碼片段，您可以調整Nginx 配置以啟用SSL。

我們將在本指南中假設您在/etc/nginx/sites-available目錄中使用自定義服務器塊配置文件。本指南還遵循先決條件Nginx 教程中的約定並用於此示例。根據需要替換您的配置文件名。/etc/nginx/sites-available/your_domain

在繼續之前，請備份您當前的配置文件：

- 須藤 cp /etc/nginx/sites-available/your_domain /etc/nginx/sites-available/your_domain.bak

現在，打開配置文件進行調整：

- 須藤納米 /etc/nginx/sites-available/your_domain

在內部，您的服務器塊可能類似於以下內容：

/etc/nginx/sites-available/your_domain

```
server {
        listen 80;
        listen [::]:80;
```

```
        root /var/www/your_domain/html;
        index index.html index.htm index.nginx-debian.html;

        server_name your_domain www.your_domain;

        location / {
                try_files $uri $uri/ =404;
        }
}
```

您的文件可能以不同的順序，並且代替root和index指令，你可能有一些location，proxy_pass或其它自定義的配置語句。這很好，因為您只需要更新listen指令並包含SSL 片段。然後修改這個現有的服務器塊以在端口上提供SSL 流量443，並創建一個新的服務器塊來響應端口80並自動將流量重定向到端口443。

注意：使用302 重定向，直到您確認一切正常。之後，您將其更改為永久301 重定向。

在您現有的配置文件中，更新兩個listen語句以使用port 443and ssl，然後包含您在前面的步驟中創建的兩個片段文件：

/etc/nginx/sites-available/your_domain

```
server {
    listen 443 ssl;
    listen [::]:443 ssl;
    include snippets/self-signed.conf;
    include snippets/ssl-params.conf;

root /var/www/your_domain/html;
        index index.html index.htm index.nginx-debian.html;

  server_name your_domain.com www.your_domain.com;

  location / {
                try_files $uri $uri/ =404;
        }
}
```

接下來，在第一個塊的右括號(})之後將第二個服務器塊添加到配置文件中：

/etc/nginx/sites-available/your_domain.com

```
server {
    listen 80;
    listen [::]:80;

    server_name your_domain.com www.your_domain.com;

    return 302 https://$server_name$request_uri;
}
```

這是一個簡單的配置，它偵聽端口80並執行到HTTPS 的重定向。保存並關閉文件，CTRL + X然後按然後Y，ENTER完成編輯後。

步驟3 – 調整防火牆

如果您ufw按照先決條件指南的建議啟用了防火牆，則需要調整設置以允許SSL 流量。幸運的是，Nginxufw在安裝時註冊了一些配置文件。

您可以通過鍵入以下內容來查看可用的配置文件：

- sudo ufw 應用程序列表

將出現如下列表：

```
Available applications:
  Nginx Full
  Nginx HTTP
  Nginx HTTPS
  OpenSSH
```

您可以通過鍵入sudo ufw status以下內容來檢查當前設置：

- 須藤 ufw 狀態

它可能會生成以下響應，這意味著Web 服務器只允許HTTP 流量：

```
Status: active

To                         Action      From
--                         ------      ----
OpenSSH                    ALLOW       Anywhere
Nginx HTTP                 ALLOW       Anywhere
```

```
OpenSSH (v6)               ALLOW       Anywhere (v6)
Nginx HTTP (v6)            ALLOW       Anywhere (v6)
```

要允許HTTPS 流量，您可以更新“Nginx Full”配置文件的權限，然後刪除多餘的“Nginx HTTP”配置文件限額：

- sudo ufw 允許'Nginx Full'
- sudo ufw delete 允許'Nginx HTTP'

運行後`sudo ufw status`，您應該會收到以下輸出：

- 須藤 ufw 狀態

```
Status: active

To                         Action      From
--                         ------      ----
OpenSSH                    ALLOW       Anywhere
Nginx Full                 ALLOW       Anywhere
OpenSSH (v6)               ALLOW       Anywhere (v6)
Nginx Full (v6)            ALLOW       Anywhere (v6)
```

此輸出確認對防火牆的調整已成功，您已準備好在Nginx 中啟用更改。

步驟4 - 在Nginx 中啟用更改

完成對防火牆的更改和調整後，您可以重新啟動Nginx 以實現新更改。

首先，檢查文件中是否有語法錯誤。您可以通過鍵入`sudo nginx -t`：

- 須藤nginx -t

如果一切順利，您將得到如下結果：

```
nginx: [warn] "ssl_stapling" ignored, issuer certificate not found for
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
```

注意開頭的警告。如前所述，此特定設置會生成警告，因為您的自簽名證書無法使用SSL 裝訂。這是意料之中的，您的服務器仍然可以正確加密連接。

如果您的輸出與我們的示例匹配，則您的配置文件沒有語法錯誤。如果是這種情況，那麼您可以安全地重新啟動Nginx 以實現更改：

- 須藤 systemctl 重啟 nginx

現在系統已使用新更改重新啟動，您可以繼續進行測試。

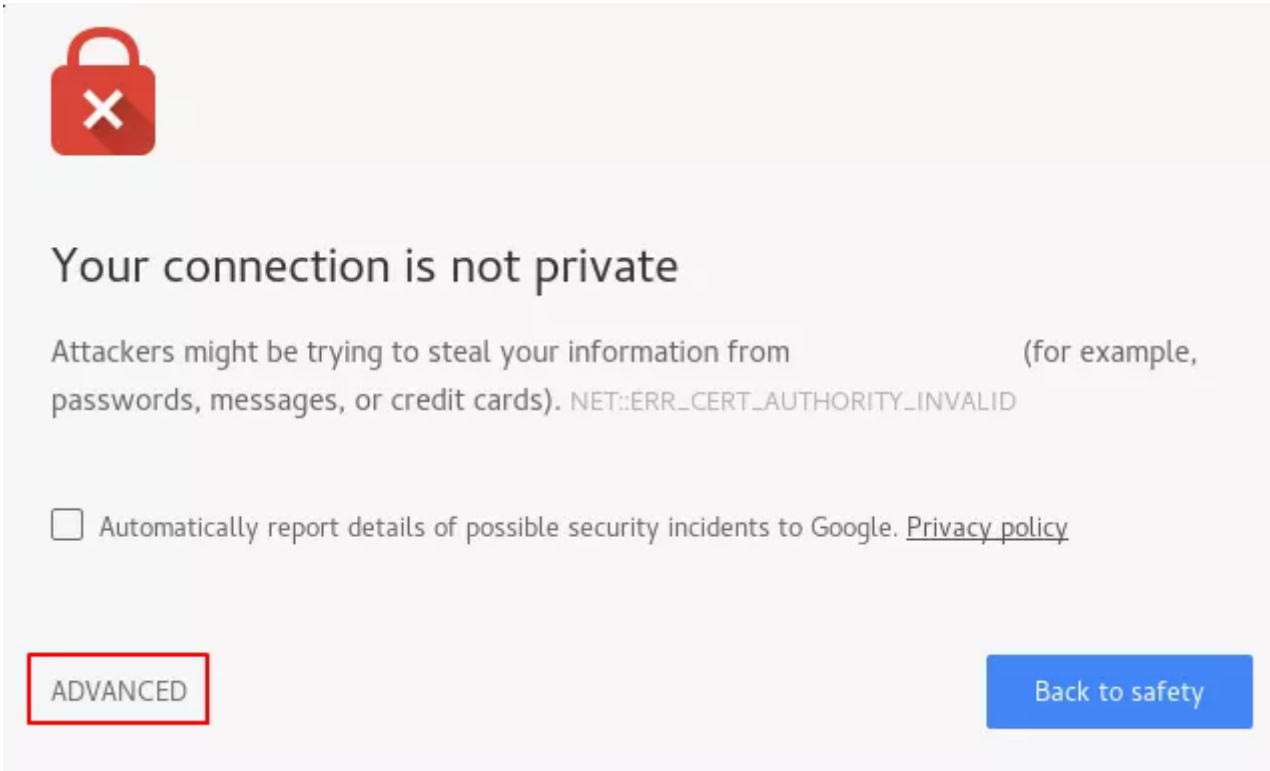
步驟5 – 測試加密

現在，您已準備好測試SSL 服務器。

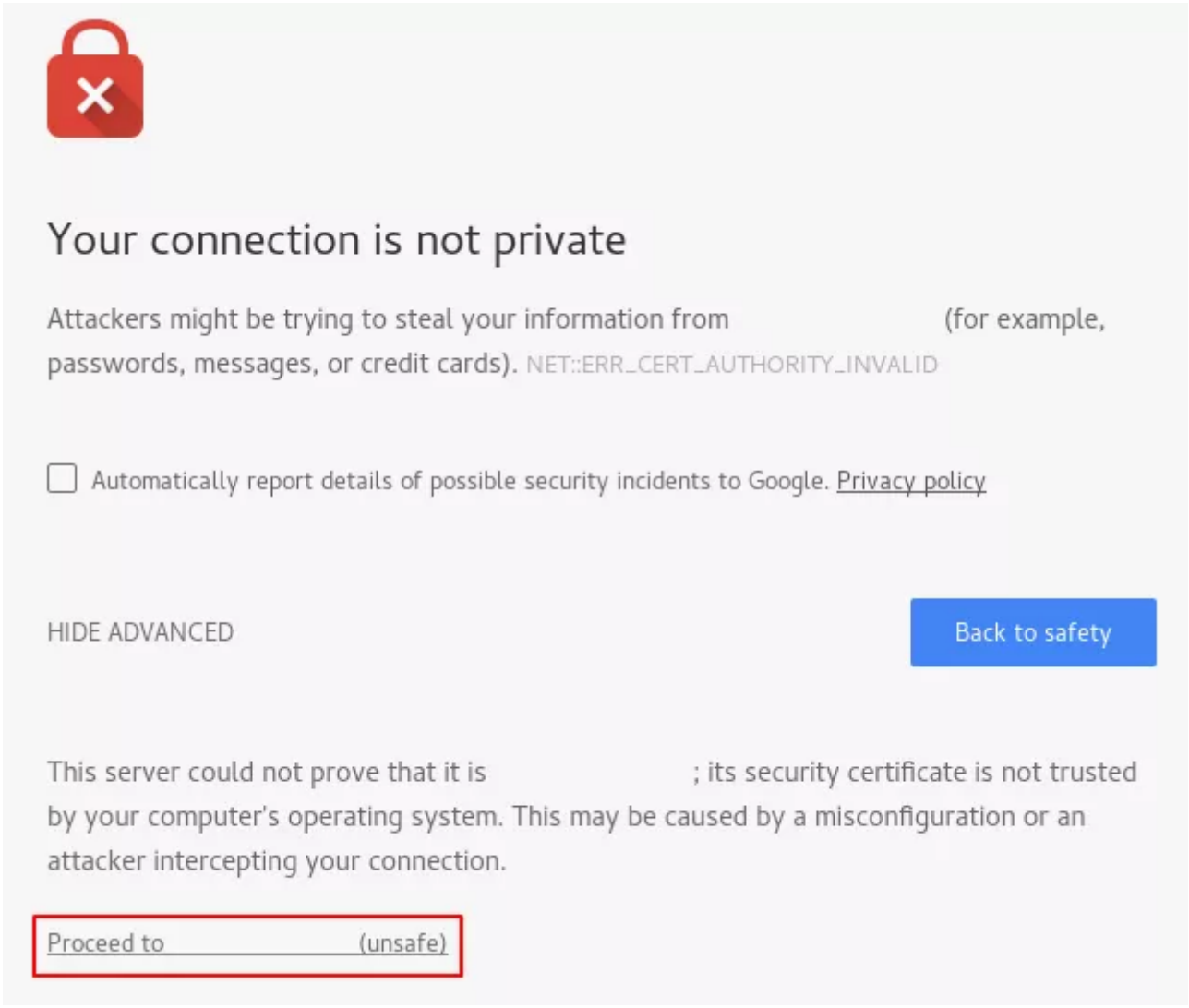
打開您的網絡瀏覽器，`https://`然後在地址欄中輸入您服務器的域名或IP：

```
https://server_domain_or_IP
```

根據您的瀏覽器，您可能會收到警告，因為您創建的證書不是由您瀏覽器的受信任證書頒發機構之一簽署的，



此警告在意料之中且正常。我們只對我們證書的加密方面感興趣，而不是對我們主機真實性的第三方驗證。單擊**“高級”**，然後單擊提供的鏈接以繼續訪問您的主機：



此時，您應該被帶到您的站點。在我們的示例中，瀏覽器地址欄顯示一個帶有“x”的鎖，這意味著無法驗證證書。它仍在加密您的連接。請注意，此圖標可能會有所不同，具體取決於您的瀏覽器。

如果Nginx 配置了兩個服務器塊，自動將HTTP 內容重定向到HTTPS，您還可以檢查重定向是否正常工作：

```
http://server_domain_or_IP
```

如果這導致相同的圖標，則表示您的重定向工作正常。

步驟6 – 更改為永久重定向

如果您的重定向工作正常並且您確定只允許加密流量，您應該修改Nginx 配置以使重定向永久。

再次打開您的服務器塊配置文件：

- 須藤納米 /etc/nginx/sites-available/your_domain

找到return 302並將其更改為return 301：

/etc/nginx/sites-available/your_domain.com

```
return 301 https://$server_name$request_uri;
```

按CTRL + X然後Y和保存並關閉文件ENTER

檢查您的配置是否存在語法錯誤：

- 須藤nginx -t

準備好後，重新啟動Nginx 以使重定向永久生效：

- 須藤 systemctl 重啟 nginx

重新啟動後，將實施更改並且您的重定向現在是永久性的。

結論

您已將Nginx 服務器配置為對客戶端連接使用強加密。這將使您能夠安全地處理請求並防止外部各方讀取您的流量。或者，您可以選擇使用自簽名SSL 證書，該證書可以從Let's Encrypt 獲得，Let's Encrypt 是一個證書頒發機構，可安裝免費的TLS/SSL 證書並在Web 服務器上啟用加密的HTTPS。從我們關於如何在Ubuntu 20.04 上使用Let's Encrypt 保護Nginx 的教程中了解更多信息（見https://www.linuxmi.com/ubuntu-20-04-lets-encrypt-nginx.html）。

來自：Linux迷
鏈接：https://www.linuxmi.com/ubuntu-20-04-nginx-ssl-certificate.html

關注我們

長按或掃描下面的二維碼關注Linux公社

關注Linux公社，添加“星標”
每天獲取技術乾貨，讓我們一起成長
合作聯繫：root@linuxidc.net

閱讀原文

喜歡此內容的人還喜歡

就這麼簡單！SQLite 數據庫增刪改查實例入門

Linux公社

羅熙表情包：不要吵啦

羅熙表情包

一個不懂得反省的人，這輩子也就只剩下一首《涼涼》了！（深思）

心靈拾憶茶館