ChatGPT

PocketBase 后端服务 HTTPS/SSL 配置指南

PocketBase 是一款开箱即用的后端服务框架，为了保证移动客户端与服务之间的通信安全，应启用 HTTPS。Let's Encrypt 提供免费的 SSL/TLS 证书，并提供 Certbot 等工具来自动获取和续订证书 [1]。反向代理服务器（如 Nginx、Caddy）可以使用这些证书对外提供 HTTPS 服务，其中 Caddy 默认会自动获取并续订证书 [2]。本文将以 Ubuntu 服务器为例，分步介绍如何使用 Certbot 或 acme.sh 申请 Let’s Encrypt 证书，配置 Nginx/Caddy 反向代理启用 HTTPS，以及 PocketBase 的相关配置注意事项和客户端信任设置。

申请 Let’s Encrypt SSL 证书（Certbot 或 acme.sh）

1. **准备域名和 DNS 解析**：首先需要一个绑定到服务器公网 IP 的域名（如 `example.com`）。在域名管理控制台中添加 A 记录，将域名指向服务器 IP [3]。Let's Encrypt 只能为可访问的域名颁发证书，因此确保 DNS 记录已经生效并正确指向服务器。

2. **使用 Certbot 获取证书**：Certbot 是 Let's Encrypt 官方推荐的客户端。在 Ubuntu 上可以按以下步骤安装并申请证书：

3. 更新软件源并安装 Certbot：

```
sudo apt update
sudo apt install certbot python3-certbot-nginx
```

4. 检查 Nginx 配置并启动（若之前未配置 Nginx，可先创建一个基本的配置文件）。

5. 运行 Certbot 获取证书（这里以 `example.com` 为例）：

```
sudo certbot --nginx -d example.com
```

该命令会自动与 Nginx 交互，验证域名所有权并配置 SSL 证书 [4]。执行过程中需要输入邮箱地址并同意服务条款。证书文件通常会保存到 `/etc/letsencrypt/live/example.com/` 目录下，包括 `fullchain.pem`（全链证书）和 `privkey.pem`（私钥）。

6. **使用 acme.sh 获取证书（可选）**：acme.sh 是一个轻量级的 Shell 脚本 ACME 客户端，也可以用于申请 Let’s Encrypt 证书。安装和使用示例如下 [5] [6]：

7. 安装 acme.sh：

```
curl https://get.acme.sh | sh
source ~/.bashrc
```

8. 使用 Webroot 模式申请证书（假设网站根目录为 `/var/www/html`）：

```
acme.sh --issue -d example.com --webroot /var/www/html
```

或者使用临时的 standalone 模式（这会在后台启动一个临时的 Web 服务进行验证）：

```
acme.sh --issue -d example.com --standalone
```

申请成功后，证书会保存在 `~/.acme.sh/example.com/` 目录。再将证书安装到系统目录（并设置重载 Nginx）例如：

```
acme.sh --install-cert -d example.com \
 --key-file /etc/ssl/private/example.com.key \
 --fullchain-file /etc/ssl/certs/example.com.crt \
 --reloadcmd "sudo systemctl reload nginx"
```

acme.sh 默认会在用户目录下创建每天运行一次的 cron 任务，自动在证书到期 60 天前尝试续期 [7] [8] 。

配置反向代理（Nginx 或 Caddy）启用 HTTPS

使用 Nginx 或 Caddy 作为前端代理，可以在 443 端口对外提供 HTTPS 服务，并将流量转发给 PocketBase。

- **Nginx 配置示例**：假设 PocketBase 在本地监听 8090 端口（只绑定在 localhost），Nginx 可配置如下：

```
# 将 HTTP 自动跳转到 HTTPS
server {
  listen 80;
  server_name example.com;
  return 301 https://$host$request_uri;
}

server {
  listen 443 ssl;
  server_name example.com;
  ssl_certificate /etc/letsencrypt/live/example.com/fullchain.pem;
  ssl_certificate_key /etc/letsencrypt/live/example.com/privkey.pem;
  ssl_protocols TLSv1.2 TLSv1.3;
  ssl_ciphers HIGH:!aNULL:!MD5;
  client_max_body_size 10M;

  location / {
    proxy_pass http://127.0.0.1:8090/;
    proxy_http_version 1.1;
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
    proxy_set_header Connection "";
    proxy_read_timeout 360s;
```

```
  }
}
```

该配置中，`server_name` 填写你的域名，`ssl_certificate` 和 `ssl_certificate_key` 指向 Certbot 或 acme.sh 生成的证书文件路径。`location /` 中的 `proxy_pass` 将所有流量转发到 PocketBase 服务（监听在本地 8090 端口）。同时通过 `proxy_set_header X-Real-IP`、`X-Forwarded-For` 等头部，将客户端真实 IP 信息传递给 PocketBase [9] [10]。注意 `proxy_pass` 后面添加了 `/`，以避免路径问题（如未包含尾部 `/` 可能导致 404 问题 [11]）。配置完成后，重启 Nginx：

```
sudo nginx -t
sudo systemctl restart nginx
```

- **Caddy 配置示例**：Caddy 使用简单的 Caddyfile，可实现自动 HTTPS。示例配置：

```
example.com {
    reverse_proxy 127.0.0.1:8090
}
```

只需将上述配置保存为 `/etc/caddy/Caddyfile`，然后启动 Caddy。Caddy 会自动解析域名、申请 Let's Encrypt 证书并处理续签，同时默认将 HTTP 跳转到 HTTPS [2] [12]。无需手动指定证书文件路径。启动后，Caddy 会监听 443 端口并转发请求到本地 PocketBase 服务。

自动续期证书

- **Certbot 自动续期**：Certbot 在安装时会自动创建一个 systemd 定时任务，每天运行两次，自动续期到期前 30 天的证书 [13]。你可以手动测试续期：

```
sudo certbot renew --dry-run
```

如果没有错误发生，表示自动续期正常。证书续期后 Certbot 会自动重载 Nginx，使其使用新证书 [14]。

- **acme.sh 自动续期**：acme.sh 安装后会自动在用户 crontab 中添加 cron 任务（每天定时运行 `acme.sh --cron`），默认在证书有效期剩余 60 天时尝试续期 [7] [8]。一般无需额外配置。你也可以手动运行：

```
acme.sh --cron --home ~/.acme.sh
```

或检查各域名的续期状态。请确保续期命令中的 `--reloadcmd`（如示例所示）能够在续证后重载代理服务（Nginx/Caddy），以让它们加载新证书。

PocketBase 配置注意事项

1. **监听地址**：如果使用 Nginx/Caddy 作为前端代理，建议将 PocketBase 绑定到本地接口，不直接对外。例如，可以这样启动 PocketBase：

```
pocketbase serve --http=127.0.0.1:8090
```

这样 PocketBase 只在本地监听 8090 端口，外部无法直接访问，只能通过代理访问，更安全。

2. **代理头配置**：由于客户端 IP 已由 Nginx/Caddy 转发，你应在 PocketBase 管理后台（设置 > 系统）启用「代理头 (Proxy Headers)」支持，将 `X-Real-IP`、`X-Forwarded-For` 等头部纳入使用 [9]。这样 PocketBase 才能正确获取并记录真实的客户端 IP，避免日志中记录代理服务器的 IP。

3. **TLS/SSL 配置**：PocketBase 本身无需做额外的 HTTPS 配置，因 HTTPS 已由 Nginx/Caddy 终止。只要 PocketBase 正常监听 HTTP，前端代理即可加密流量。若你使用的是 Caddy，Caddy 将自动处理证书和加密；若使用 Nginx，确保证书路径配置正确即可。请检查防火墙或安全组设置，确保 80、443 端口已开放给客户端访问。

4. **实时功能支持**：PocketBase 支持实时订阅功能，需要长连接（WebSocket/HTTP-长轮询）。上文 Nginx 配置中的 `proxy_http_version 1.1` 和 `proxy_set_header Connection ""` 已确保支持长连接。如果你使用其他代理设置，也应确保支持 WebSocket（例如为升级请求添加 `Upgrade` 和 `Connection: upgrade` 头）。

移动 App 客户端信任与访问

移动客户端使用 HTTPS 调用 PocketBase API 时，只需将请求地址改为 `https://yourdomain.com`。由于使用了正规 CA 签发的证书，无需在客户端额外信任配置。Let's Encrypt 的根证书已经包含在主流操作系统的信任列表中——例如 **iOS 10** 以上、**Android 7.1.1** 以上的系统默认信任 ISRG 根证书 [15]。也就是说，正常情况下无需在 App 内添加额外的信任链配置。开发时，只需将 API 域名配置为 `https://yourdomain.com`，并保证使用 HTTPS 进行请求。部分平台可能要求目标服务器支持 TLS1.2+，但 Let’s Encrypt 证书和现代代理配置一般都能满足这些要求。

注意：如果你的 App 曾使用 HTTP（尤其在 iOS 上需要白名单配置），在切换到 HTTPS 后请移除相关临时设置，让 App 强制使用 HTTPS 以符合应用商店要求。

总结

通过上述步骤，你可以在 Ubuntu 服务器上为 PocketBase 后端部署免费 SSL 证书并启用 HTTPS 服务。主要流程包括：准备域名并解析到服务器，使用 Certbot 或 acme.sh 向 Let’s Encrypt 申请证书，配置 Nginx/Caddy 反向代理以实现 HTTPS 访问，同时设置证书自动续期。最后保证 PocketBase 仅监听本地接口并启用代理头设置，使服务既安全又稳定。移动客户端只需使用 https:// 域名调用 API，系统会自动信任由 Let’s Encrypt 签发的证书 [15]。这样即可完成 PocketBase 后端的 HTTPS 加密配置，使移动客户端与服务间的数据传输安全可靠。

参考资料： Let’s Encrypt 官方文档、Certbot 教程、PocketBase 文档 [1] [9] [2] [15]。

---

[1] [3] [13] [14] How To Secure Nginx with Let's Encrypt on Ubuntu 20.04 | DigitalOcean
https://www.digitalocean.com/community/tutorials/how-to-secure-nginx-with-let-s-encrypt-on-ubuntu-20-04

[2] [12] Automatic HTTPS — Caddy Documentation
https://caddyserver.com/docs/automatic-https

4 Deploying Pocketbase with Docker, Nginx and SSL - DEV Community
https://dev.to/_russell/deploying-pocketbase-with-docker-nginx-and-ssl-323l

5 How to create Let's Encrypt SSL certificates with acme.sh on Linux | TechRepublic
https://www.techrepublic.com/article/how-to-create-lets-encrypt-ssl-certificates-with-acme-sh-on-linux/

6 8 GitHub - acmesh-official/acme.sh: A pure Unix shell script implementing ACME client protocol
https://github.com/acmesh-official/acme.sh

7 Certificate Renewal with acme.sh cronjob - Help - Let's Encrypt Community Support
https://community.letsencrypt.org/t/certificate-renewal-with-acme-sh-cronjob/208329

9 10 Going to production - Docs - PocketBase
https://pocketbase.io/docs/going-to-production/

11 nginx - Can't get Pocketbase running behind Ngnix as a reverse proxy - Stack Overflow
https://stackoverflow.com/questions/75360904/cant-get-pocketbase-running-behind-ngnix-as-a-reverse-proxy

15 Certificate Compatibility - Let's Encrypt
https://letsencrypt.org/docs/certificate-compatibility/