11

CHAPTER

独自ドメインと HTTPS 設定

https:// で安全に — Route 53 と Let's Encrypt で無料 SSL

本書のサンプル章

AGENDA

この章で学ぶこと

01 ドメインと DNS の基礎
ドメイン名が IP に変わる仕組み

02 ドメインの取得
お名前 .com で .com を取る

03 Route 53 で DNS 設定
ホストゾーン作成 + A レコード

04 Let's Encrypt で無料 SSL
Certbot で証明書取得 + 自動更新

05 Nginx + Django 設定更新
HTTPS リダイレクト + セキュリティヘッダ

06 つまずきポイント TOP 4
DNS 伝播 / Certbot / リダイレクトループ

PART 1

ドメインと DNS

覚えやすい名前 → 数字の IP に変換する仕組み

DNS = インターネットの電話帳

ANALOGY

電話帳の例え

IP アドレス

`192.0.2.1`

= 電話番号（覚えにくい）

↓ DNS が変換

ドメイン名

`google.com`

= 名前（覚えやすい）

→ ブラウザは DNS に問い合わせて IP を取得

WHY DOMAIN

ドメイン取得のメリット

✓ 覚えやすい	*3.x.x.x → my-game.com*
✓ 無料 SSL 証明書	*Let's Encrypt が使える*
✓ SEO に有利	*検索エンジンの評価 UP*
✓ 信頼性 UP	*プロフェッショナルな印象*
✓ メアド取得可	*contact@my-game.com*

RECORDS

主要な DNS レコードタイプ

A — IPv4 アドレス

ドメイン → IPv4

```
example.com → 192.0.2.1
```

AAAA — IPv6 アドレス

ドメイン → IPv6

```
example.com → 2001:db8::1
```

CNAME — 別名

ドメイン → ドメイン

```
www.example.com → example.com
```

MX — メールサーバー

メール宛先

```
example.com → mail.example.com
```

TXT — テキスト情報

認証 / 検証

```
"v=spf1 ..."
```

NS — ネームサーバー

DNS サーバー指定

```
example.com → ns1.example.com
```

本書では A レコード（ドメイン → EC2 の IP）を Route 53 で設定する

REGISTRAR

ドメイン取得サービスの比較

Route 53

$12/ 年

AWS と統合

AWS 慣れに最適

お名前 .com

¥1,408/ 年

日本語サポート

本書推奨（初心者）

Cloudflare

$10/ 年

無料 CDN 付き

高速 + DDoS 対策

Namecheap

$8.88/ 年

海外サービス

安価

お名前 .com 取得手順（約 10 〜 20 分）

01	02	03	04	05	06
検索	選択	アカウント	メール認証	支払い	完了

PART 2

Route 53 + DNS 設定

AWS の DNS でドメインを EC2 に紐付ける

ROUTE 53

AWS の DNS サービス

FEATURES

Route 53 の主要機能

1. **ホストゾーン管理**
   DNS レコードを一元管理
2. **高可用性**
   AWS グローバルネットワーク
3. **AWS 統合**
   EC2 ・ S3 ・ CloudFront 連携
4. **ヘルスチェック**
   障害時の自動切替

PRICING

料金

ホストゾーン	$0.50/ 月（1 ドメイン）
DNS クエリ	100 万まで無料
ドメイン取得	$12 〜 15/ 年（.com）

HOSTED ZONE

ホストゾーンとは

1 つのドメインの DNS レコードを
まとめて管理する単位。
A / CNAME / MX を中に持つ。

ホストゾーン作成 + ネームサーバー切替

STEP 1 — Route 53 でホストゾーンを作成

```
$ aws route53 create-hosted-zone \
    --name adventure-dev-journey.com \
    --caller-reference $(date +%s) \
    --hosted-zone-config Comment="..."

→ NameServers: ns-1837.awsdns-37.co.uk, ...
```

STEP 2 — お名前 .com でネームサーバーを切替

1. ns-1837.awsdns-37.co.uk
2. ns-1428.awsdns-50.org
3. ns-169.awsdns-21.com
4. ns-793.awsdns-35.net

→ お名前 .com Navi → ネームサーバー設定 → 「その他のサービス」に 4 つ入力

A レコード作成 + DNS 伝播確認

A レコード作成

```
# 環境変数
export HOSTED_ZONE_ID="Z123…"
export PUBLIC_IP="3.112.xxx.xxx"

# A レコードを作成（ルートドメイン）
$ aws route53 \
    change-resource-record-sets \
    --hosted-zone-id $HOSTED_ZONE_ID \
    --change-batch '{
      "Changes": [{
        "Action": "CREATE",
        "ResourceRecordSet": {
          "Name": "adventure-dev-journey.com",
          "Type": "A",
          "TTL": 300,
          "ResourceRecords": [{
            "Value": "'$PUBLIC_IP'"}]
        }}]}'
```

DNS 伝播の確認

```
$ dig adventure-dev-journey.com
```

```
adventure-dev-journey.com.
  300 IN A 3.112.xxx.xxx
```

反映時間の目安

数分〜数時間

（最大で 24 時間）

→ DNS Checker で世界中から確認可

PART 3

HTTPS 化

Let's Encrypt で無料 SSL → Nginx と Django を更新

VIRTUAL HOST

Nginx バーチャルホスト設定 (HTTP)

/etc/nginx/sites-available/adventure-game

```
server {
    listen 80;
    server_name \
        adventure-dev-journey.com \
        www.adventure-dev-journey.com;

    # 静的ファイル
    location /static/ {
        alias /home/ubuntu/.../staticfiles/;
    }

    # Gunicorn へのプロキシ
    location / {
        include proxy_params;
        proxy_pass http://unix:/.../sock;
    }
}
```

ポイント

server_name

ドメイン名を指定

Host ヘッダーで振り分け

ALLOWED_HOSTS

Django 側にも追加

.env.prod を更新
Gunicorn 再起動

HTTP 状態でテスト

Certbot の前提条件

ドメインでアクセス可能に

CERTBOT

Let's Encrypt で無料 SSL 証明書

01

INSTALL

Certbot を入れる

```
$ sudo snap install \
  --classic certbot
```

02

REQUEST

証明書を発行

```
$ sudo certbot --nginx \
  -d adventure-dev-journey.com \
  -d www.adventure-...
```

03

AUTO RENEW

自動更新を確認

```
$ sudo certbot renew \
  --dry-run
```

証明書の保存先 + 自動更新の仕組み

```
/etc/letsencrypt/live/<domain>/
├── fullchain.pem   #  証明書
├── privkey.pem     #  秘密鍵
├── chain.pem
└── cert.pem
```

90 日有効

systemd timer が 1 日 2 回チェック
残り 30 日で自動更新

改善版 Nginx 設定（HTTP/2 + HSTS + Cache）

改善版（Certbot 自動生成 + 手動改善）

```
server {
  listen 443 ssl http2;
  server_name adventure-dev-journey.com;

  ssl_certificate     /etc/letsencrypt/...;
  ssl_certificate_key /etc/letsencrypt/...;

  # セキュリティヘッダー
  add_header Strict-Transport-Security \
      "max-age=31536000" always;
  add_header X-Frame-Options "SAMEORIGIN";
  add_header X-Content-Type-Options "nosniff";

  location /static/ {
    expires 30d;
    add_header Cache-Control "public";
  }
}

# HTTP → HTTPS リダイレクト
server { listen 80; return 301 https://$host$request_uri; }
```

改善ポイント

HTTP/2
高速化

HSTS
常に HTTPS を強制

X-Frame-Options
クリックジャッキング対策

X-Content-Type
MIME スニッフィング対策

expires 30d
静的ファイルキャッシュ

settings.py の HTTPS セキュリティ設定

server/settings.py — 末尾に追加

```
# HTTPS 設定（本番のみ）
if not DEBUG:
    # 全リクエストを HTTPS に
    SECURE_SSL_REDIRECT = True

    # クッキーは HTTPS のみ
    SESSION_COOKIE_SECURE = True
    CSRF_COOKIE_SECURE = True

    # HSTS（1 年間 HTTPS 強制）
    SECURE_HSTS_SECONDS = 31536000
    SECURE_HSTS_INCLUDE_SUBDOMAINS = True
    SECURE_HSTS_PRELOAD = True

    # MIME スニッフィング対策
    SECURE_CONTENT_TYPE_NOSNIFF = True

    # クリックジャッキング対策
    X_FRAME_OPTIONS = 'SAMEORIGIN'
```

各設定の意味

SSL_REDIRECT

全 HTTP を HTTPS へ

COOKIE_SECURE

クッキー = HTTPS only

HSTS

ブラウザに HTTPS 強制

CONTENT_TYPE

MIME Sniffing 防止

X_FRAME

iframe 禁止

PART 4

確認とつまずき

https:// で公開されているか確認 + 詰まりポイント

TROUBLESHOOTING

つまずきポイント TOP 4

01

DNS 伝播待ち

```
NXDOMAIN / DNS problem
```

→ dig で確認
→ 数分〜数時間待つ

02

Certbot 検証失敗

```
could not connect to verify
```

→ SG で port 80 開放
→ Nginx 起動確認

03

リダイレクトループ

```
リダイレクトが繰り返し
```

→ X-Forwarded-Proto 確認
→ proxy_params に $scheme があるか

04

証明書更新失敗

```
90 日後に切れる
```

→ certbot renew (手動)
→ snap.certbot.renew .timer の状態確認

🔒 SECURE

https:// で世界に安全に公開した。

この章で学んだこと

- ✓ ドメインと DNS の仕組み
- ✓ お名前 .com でドメイン取得
- ✓ Route 53 ホストゾーン作成
- ✓ ネームサーバー切替 + A レコード
- ✓ Let's Encrypt + Certbot で SSL
- ✓ Nginx で HTTPS + HTTP リダイレクト
- ✓ Django HTTPS セキュリティ設定

NEXT

第 12 章

運用とトラブルシュート

- ▸ ログの確認
- ▸ パフォーマンス監視
- ▸ バックアップ

→ 公開した後の世話の仕方

Let's Encrypt は 90 日有効・自動更新の動作確認は必ず 1 度。