Authentik Forward Auth e Nginx: Protezione SSO per Web Radio e Streaming (senza timeout)

Authentik Forward Auth e Nginx: Protezione SSO per Web Radio e Streaming (senza timeout)

Quando si decide di mettere in sicurezza un'applicazione web aziendale dietro un Identity Provider come **Authentik**, la soluzione più immediata consiste nel deployare un avamposto in modalità **Proxy Standalone**. In questa configurazione, l'avamposto (Outpost) fa da vero e proprio scudo protettivo (Reverse Proxy) davanti all'applicazione: tutto il traffico attraversa il container dell'avamposto prima di raggiungere il server web reale.

Questo approccio funziona magnificamente per la stragrande maggioranza dei siti web e delle API. Tuttavia, si rivela un **disastro totale** se l'applicazione in questione effettua uno streaming multimediale continuo, come nel caso di una **Web Radio** (es. AzuraCast, Icecast).

Vediamo in dettaglio perché lo streaming audio si interrompe con i proxy classici, come risolvere il problema in modo definitivo usando la modalità **Forward Auth** con Nginx, e come superare le insidie di riscrittura degli header introdotte da Cloudflare Tunnel.

Il Problema: I timeout dei proxy in Go

Gli avamposti di Authentik sono scritti in Go. I web server scritti in Go utilizzano delle impostazioni interne molto rigide per gestire i timeout delle connessioni TCP (in particolare `ReadTimeout` e `WriteTimeout`).

Quando un utente si collega ad uno stream audio (es. `/listen/radio.mp3`), la connessione HTTP stabilita è di tipo *infinite chunked transfer*: la connessione rimane aperta indefinitamente per inviare continuamente i pacchetti audio alle orecchie dell'ascoltatore.

Dopo pochi minuti di riproduzione, il flusso audio si interrompe bruscamente. Questo accade perché la connessione supera il limite del timeout di scrittura dell'avamposto Go, il quale interpreta la sessione infinita come un potenziale attacco o un socket rimasto appeso e la abbatte di forza. Aumentare all'infinito questi timeout non è mai una buona idea, poiché espone il server a vulnerabilità di tipo Denial of Service (DoS).

La Soluzione: Nginx Forward Auth (`auth_request`)

Per combinare la sicurezza dell'SSO e la stabilità dello streaming infinito, dobbiamo invertire i ruoli e utilizzare la modalità **Forward Auth (Opzione B)**.

In questo scenario:

1. **Nginx** diventa l'entry-point pubblico (porta 80).
2. Quando arriva una richiesta, Nginx usa il modulo **`auth_request`** per fare una mini-richiesta interna e velocissima all'avamposto per controllare se l'utente possiede il cookie di sessione corretto.
3. Se l'avamposto risponde `200 OK`, Nginx serve il contenuto.
4. Se risponde `401 Unauthorized`, Nginx reindirizza l'utente al login.

La magia per la Web Radio sta nel fatto che **Nginx gestisce direttamente il flusso di dati audio verso il client**, bypassando completamente il container dell'avamposto. Poiché Nginx è ottimizzato per connessioni asincrone infinite e non soffre dei timeout rigidi di Go, lo stream rimarrà stabile all'infinito, pur rimanendo protetto dall'autenticazione a monte!

```
sequenceDiagram
    autonumber
    actor Client
    participant Cloudflare as Cloudflare Tunnel
    participant Nginx as Nginx (Porta 80)
    participant Outpost as Authentik Outpost (Porta 9000)
    participant AzuraCast as AzuraCast (Porta 24443)

    Client->>Cloudflare: Richiesta audio (/listen/stream.mp3)
    Cloudflare->>Nginx: Inoltra richiesta
    Nginx->>Outpost: Verifica Cookie (auth_request)
    Note over Outpost: Controlla sessione
```

```
    Outpost-->>Nginx: 200 OK (Utente autenticato)
    Nginx->>AzuraCast: Proxy dello stream audio
    AzuraCast-->>Nginx: Invio pacchetti audio (infinito)
    Nginx-->>Client: Stream audio diretto (Stabilità infinita)
```

Step 1: Configurazione di Authentik Centrale

Nel pannello di controllo del tuo Authentik centrale:

1. **Crea il Proxy Provider**:
    - Vai su *Directory -> Applicazioni -> Provider -> Crea*.
    - Seleziona **Proxy Provider**.
    - Imposta la modalità su **Autenticazione inoltrata (applicazione singola)** (Forward Auth).
    - Imposta l'host esterno su `https://radio.example.org`.
    - Imposta l'host interno su `http://192.168.5.3` (l'IP locale del server della radio).
2. **Crea l'Avamposto Remoto**:
    - Vai su *Directory -> Applicazioni -> Avamposti -> Crea*.
    - Scegli come tipo **Proxy** ed assegna l'applicazione creata sopra.
    - Imposta il campo *Integrazione* su **Nessuna integrazione**.
    - Clicca su salva, apri i dettagli dell'avamposto appena creato e copia l'**Outpost Token** generato.

Step 2: Esecuzione dell'Avamposto su Docker (Server Remoto)

Sulla macchina dove gira la Web Radio (`192.168.5.3`), creiamo un file `docker-compose.yml` per far girare il proxy locale.

Dobbiamo fare in modo che il container dell'avamposto parli con Authentik centrale via VPN interna (NetBird) sulla porta `9000`, ma indichi al browser dell'utente di fare il login usando l'indirizzo HTTPS pubblico. Per fare questo, valorizziamo la variabile `AUTHENTIK_HOST_BROWSER`:

```
services:
  authentik_proxy:
```

```
    image: ghcr.io/goauthentik/proxy:2026.5.3
    container_name: authentik_proxy
    ports:
      - "9000:9000"
      - "9443:9443"
    environment:
      # Connessione interna all'API di Authentik centrale via NetBird
      AUTHENTIK_HOST: "http://auth.example.org:9000"
      # URL pubblico per i reindirizzamenti dei browser client
      AUTHENTIK_HOST_BROWSER: "https://auth.example.org"
      AUTHENTIK_TOKEN: "<IL_TUO_TOKEN_AK_OUTPOST>"
      AUTHENTIK_INSECURE: "true"
    restart: unless-stopped
```

Avvia il container con `docker compose up -d`.

Step 3: Configurazione Nginx con `auth_request` e risoluzione dei problemi Cloudflare Tunnel

Nginx deve fare da gatekeeper. C'è però un'insidia: **Cloudflare Tunnel di default riscrive l'header `Host`** sostituendolo con l'hostname della destinazione interna (es. `docker` o `localhost`). Questo rompe la validazione di Authentik, che si aspetta di vedere l'host esterno `radio.example.org` e l'URL di redirect in HTTPS.

Per risolvere questo problema in modo robusto, **forziamo l'Host a `radio.example.org` e il protocollo a `https` direttamente all'interno delle location di Authentik** in Nginx.

Ecco il file di configurazione `/etc/nginx/sites-available/azuracast.conf` completo:

```
server {
    listen 80 default_server;
    listen [::]:80 default_server;
    server_name _;

    client_max_body_size 250M;

    # Buffer ottimizzati per lo stream
    proxy_buffers 8 16k;
    proxy_buffer_size 32k;
```

```
    root /var/www/azuracast;
    index index.html;

    # --- INTEGRAZIONE AUTHENTIK FORWARD AUTH ---

    # 1. Endpoint dell'avamposto per login, callback e auth check
    location /outpost.goauthentik.io/ {
        auth_request off; # Evita controlli ricorsivi su se stesso

        proxy_pass http://127.0.0.1:9000;
        proxy_ssl_verify off;

        # Sovrascriviamo le riscritture di Cloudflare Tunnel forzando Host e
Proto reali
        proxy_set_header Host radio.example.org;
        proxy_set_header X-Original-URL https://radio.example.org$request_uri;
        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 https;
        proxy_set_header X-Forwarded-Host radio.example.org;
        proxy_set_header X-Forwarded-URI $request_uri;

        proxy_http_version 1.1;
        proxy_set_header Connection "";
    }

    # 2. Intercettazione dell'errore 401 per rimandare l'utente al login
    error_page 401 = @goauthentik_to_login;
    location @goauthentik_to_login {
        # Reindirizzamento relativo sul dominio locale gestito dall'avamposto
        return 302 /outpost.goauthentik.io/start?rd=https://
radio.example.org$request_uri;
    }

    # --- PROTEZIONE APPLICAZIONE ---

    # Document root protetta
    location / {
        auth_request /outpost.goauthentik.io/auth/nginx;
        try_files $uri $uri/ =404;
    }

    # API protette
    location /api/ {
        auth_request /outpost.goauthentik.io/auth/nginx;
        proxy_pass https://127.0.0.1:24443/api/;
```

```
        proxy_ssl_verify off;

        proxy_set_header Host radio.example.org;
        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 https;

        proxy_http_version 1.1;
        proxy_set_header Connection "";
    }

    # --- FLUSSO AUDIO PROTETTO MA SERVITO DA NGINX (STABILE) ---
    location /listen/ {
        # Nginx chiede ad Authentik se la sessione è valida (Subrequest
istantanea)
        auth_request /outpost.goauthentik.io/auth/nginx;

        # Se OK, Nginx fa il proxy dello stream audio direttamente (Niente
timeout Go!)
        proxy_pass https://127.0.0.1:24443/listen/;
        proxy_ssl_verify off;

        proxy_buffering off;
        proxy_cache off;
        proxy_http_version 1.1;
        proxy_set_header Connection "";
        proxy_read_timeout 36000s;

        proxy_set_header Host radio.example.org;
        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 https;
    }
}
```

Ricarichiamo la configurazione con `sudo systemctl reload nginx`.

Step 4: Configurazione finale del Cloudflare Tunnel

Ora che Nginx è il vero punto di ingresso e si occupa di orchestrare l'autenticazione, dobbiamo assicurarci che il Cloudflare Tunnel punti a **Nginx (porta 80)** e non più all'avamposto (porta 9000).

Nel tuo pannello di controllo Cloudflare Zero Trust, modifica la rotta del dominio `radio.example.org`:

- **Service Type**: `HTTP`
- **URL**: `http://localhost:80` (oppure l'IP interno sulla porta 80).

Conclusioni

Grazie alla combinazione di Cloudflare Tunnel, Nginx (`auth_request`) ed Authentik Forward Auth, siamo riusciti ad ottenere:

- Un portale web radio protetto da SSO aziendale.
- La certezza che nessuno possa intercettare o riprodurre il flusso audio `/listen/` dall'esterno senza autenticazione.
- Uno streaming multimediale solido all'infinito, liberato dai timeout di scrittura imposti dal proxy Go di Authentik.

Fluxer Icon

Canale di discussione e supporto

Canale di discussione e supporto. Per confrontarti su configurazioni o altro entra su fluxer.dreadful.work (https://fluxer.dreadful.work/invite/dreadful).