CRON_SECRET Documentation

Purpose

The `CRON_SECRET` is a security token that authenticates scheduled cron job requests to protect your notification endpoints from unauthorized access.

What is CRON_SECRET?

Overview

`CRON_SECRET` is a randomly generated, cryptographically secure string used to verify that cron job requests are legitimate and authorized.

Why It’s Critical

Without `CRON_SECRET`:
- ❌ Anyone could trigger your cron endpoints
- ❌ Spam/abuse of notification system
- ❌ Unauthorized data access
- ❌ Potential DoS attacks

✅ With `CRON_SECRET`:
- ✅ Only authorized requests execute
- ✅ Protection against abuse
- ✅ Secure notification scheduling
- ✅ Audit trail for cron executions

Your Provided CRON_SECRET

```
CRON_SECRET=jYEyBMdsOrDgrmmAPi8yuUisg2C0zDfqnf/8/1VkJCo=
```

Details:

- **Format**: Base64 encoded string
- **Length**: 44 characters
- **Generation Method**: `openssl rand -base64 32`
- **Entropy**: 256 bits (very secure)
- **Status**: ✅ Ready to use

Where It’s Used

1. Vercel Cron Jobs

Vercel automatically includes `CRON_SECRET` in cron job requests:

```
// app/api/cron/send-goal-reminders/route.ts
export async function GET(request: Request) {
  // Verify the request is from Vercel Cron
  const authHeader = request.headers.get('authorization');
  const token = authHeader?.split(' ')[1];

  if (token !== process.env.CRON_SECRET) {
    return new Response('Unauthorized', { status: 401 });
  }

  // Process goal reminders
  // ...
}
```

2. Manual Cron Triggers

For testing or manual execution:

```
curl -X GET \
  https://mindfulchampion.com/api/cron/send-goal-reminders \
  -H "Authorization: Bearer jYEyBMdsOrDgrmmAPi8yuUisg2C0zDfqnf/8/1VkJCo="
```

3. Scheduled Tasks

All scheduled tasks verify CRON_SECRET:

- ✅ Goal reminders (daily)
- ✅ Video analysis cleanup
- ✅ Achievement calculations
- ✅ Subscription renewals
- ✅ Trial expiration checks

Vercel Cron Configuration

vercel.json

```
{
  "crons": [
    {
      "path": "/api/cron/send-goal-reminders",
      "schedule": "0 9 * * *"
    }
  ]
}
```

Vercel automatically:
- ✅ Adds `Authorization` header with CRON_SECRET

- ✅ Triggers at scheduled time
- ✅ Retries on failure
- ✅ Logs execution

You don't need to:
- ❌ Manually configure the header
- ❌ Set up external cron services
- ❌ Manage cron daemon

Security Implementation

API Route Pattern

```
import { NextRequest, NextResponse } from 'next/server';

export async function GET(request: NextRequest) {
  // 1. Extract authorization header
  const authHeader = request.headers.get('authorization');

  // 2. Parse Bearer token
  const token = authHeader?.split(' ')[1];

  // 3. Verify against CRON_SECRET
  if (token !== process.env.CRON_SECRET) {
    return NextResponse.json(
      { error: 'Unauthorized' },
      { status: 401 }
    );
  }

  try {
    // 4. Execute cron job logic
    const result = await performScheduledTask();

    return NextResponse.json({
      success: true,
      ...result
    });
  } catch (error) {
    return NextResponse.json(
      { error: 'Internal server error' },
      { status: 500 }
    );
  }
}
```

Current Implementation

✅ Implemented in:
- `/api/cron/send-goal-reminders`
- `/api/cron/process-notifications`
- `/api/cron/cleanup-expired-trials`

How to Regenerate

When to Regenerate

⚠️ Regenerate if:
- Secret may have been exposed
- Regular rotation (every 90 days recommended)
- Security audit requires it
- Onboarding new team member

Regeneration Process

Method 1: Using OpenSSL (Recommended)

```
# Generate new secret
openssl rand -base64 32

# Example output:
zK8mNpQrXYfGh3WvT9LcBdE6uJ2sA5nH7iO1kM4xP0Y=
```

Method 2: Using Node.js

```
const crypto = require('crypto');
const secret = crypto.randomBytes(32).toString('base64');
console.log(secret);
```

Method 3: Using Python

```
import secrets
import base64

secret = base64.b64encode(secrets.token_bytes(32)).decode()
print(secret)
```

Update Process

1. **Generate new secret**
   `bash`
   `   openssl rand -base64 32`

2. **Update .env**
   `env`
   `   CRON_SECRET=<new-secret-here>`

3. **Update Vercel**
   - Go to Vercel Dashboard
   - Project Settings → Environment Variables
   - Update `CRON_SECRET`
   - Redeploy

4. **Update any external services**
   - If using external cron services (Cron-job.org, etc.)
   - Update their authorization headers

5. **Test**
   `bash`
   `npm run verify-env`

Testing CRON_SECRET

Test 1: Valid Request

```
curl -X GET \
  https://your-app.com/api/cron/send-goal-reminders \
  -H "Authorization: Bearer jYEyBMdsOrDgrmmAPi8yuUisg2C0zDfqnf/8/1VkJCo=" \
  -v
```

Expected: `200 OK` with success response

Test 2: Invalid Request (Missing Auth)

```
curl -X GET \
  https://your-app.com/api/cron/send-goal-reminders \
  -v
```

Expected: `401 Unauthorized`

Test 3: Invalid Request (Wrong Secret)

```
curl -X GET \
  https://your-app.com/api/cron/send-goal-reminders \
  -H "Authorization: Bearer wrong-secret" \
  -v
```

Expected: `401 Unauthorized`

Monitoring & Logging

Log Successful Executions

```
console.log('[CRON] Goal reminders started', {
  timestamp: new Date().toISOString(),
  endpoint: '/api/cron/send-goal-reminders'
});
```

Log Failed Auth Attempts

```
if (token !== process.env.CRON_SECRET) {
  console.error('[CRON] Unauthorized attempt', {
    timestamp: new Date().toISOString(),
    endpoint: request.url,
    ip: request.headers.get('x-forwarded-for'),
    token: token ? 'present-but-invalid' : 'missing'
  });

  return NextResponse.json(
    { error: 'Unauthorized' },
    { status: 401 }
  );
}
```

Set Up Alerts

✅ Alert on:
- Multiple failed auth attempts
- Cron job failures
- Unexpected execution times
- Missing CRON_SECRET in environment

Troubleshooting

Error: "Unauthorized"

Causes:
1. ❌ CRON_SECRET not set in environment
2. ❌ Wrong secret provided
3. ❌ Missing Authorization header
4. ❌ Typo in secret

Solutions:
1. ✅ Verify `CRON_SECRET` in .env
2. ✅ Check Vercel environment variables
3. ✅ Verify no extra spaces/characters
4. ✅ Regenerate and update if needed

Error: "Cron job not executing"

Causes:
1. ❌ Vercel cron not configured
2. ❌ Wrong schedule format
3. ❌ API route error
4. ❌ Timeout (10s limit)

Solutions:
1. ✅ Check `vercel.json` configuration
2. ✅ Verify cron schedule syntax
3. ✅ Check API route logs
4. ✅ Optimize long-running tasks

Error: “CRON_SECRET is undefined”

Causes:
1. ❌ Not set in .env
2. ❌ Not deployed to Vercel
3. ❌ Typo in variable name

Solutions:
1. ✅ Add to .env file
2. ✅ Add to Vercel environment variables
3. ✅ Check for typos: `CRON_SECRET` (not `CRON_TOKEN`, etc.)

Best Practices

Security

✅ **DO**:
- Store in .env (already in .gitignore)
- Use strong, random values (32+ bytes)
- Rotate every 90 days
- Monitor for unauthorized attempts
- Use HTTPS only

❌ **DON’T**:
- Commit to version control
- Share in public channels
- Use predictable values
- Reuse across projects
- Log the actual secret value

Deployment

✅ **DO**:
- Set in Vercel environment variables
- Test after deployment
- Document regeneration process
- Have rollback plan

❌ **DON’T**:
- Deploy without testing
- Change during active cron window
- Forget to update external services

Quick Reference

Current Value

```
CRON_SECRET=jYEyBMdsOrDgrmmAPi8yuUisg2C0zDfqnf/8/1VkJCo=
```

Generate New

```
openssl rand -base64 32
```

Test Request

```
curl -H "Authorization: Bearer jYEyBMdsOrDgrmmAPi8yuUisg2C0zDfqnf/8/1VkJCo=" \
  https://your-app.com/api/cron/send-goal-reminders
```

Verify in Code

```
const token = request.headers.get('authorization')?.split(' ')[1];
if (token !== process.env.CRON_SECRET) {
  return new Response('Unauthorized', { status: 401 });
}
```

Last Updated: December 3, 2025
Status: Production Ready
Security Level: HIGH
Rotation Schedule: Every 90 days (March 2026)