Secrets Management

Secrets Management

Last updated: 2026-02-17 **Source:** `src/drop-app/src/lib/secrets.ts`

Overview

Drop uses an abstracted secrets management system with pluggable providers. The system is backward compatible -- if no secrets provider is configured, it reads directly from environment variables (existing behavior).

Provider Selection

The provider is selected automatically based on which environment variables are set:

Priority	Condition	Provider	Description
1	`DOPPLER_TOKEN` set	Doppler	Cloud secrets manager via Doppler API
2	`AWS_SECRET_ARN` set	AWS	AWS Secrets Manager (requires AWS SDK)
3	(default)	env	Reads from `process.env`

Initialization (call once at app startup):

```
import { initSecrets } from '@/lib/secrets';

// Auto-detect provider based on env vars
initSecrets();

// Optional: custom cache TTL (default 5 minutes)
initSecrets({ ttlMs: 10 * 60 * 1000 }); // 10 minutes
```

Usage:

```
import { getSecret } from '@/lib/secrets';

const jwtSecret = await getSecret('JWT_SECRET');
const dbUrl = await getSecret('DATABASE_URL');
```

Caching

All secret values are cached in memory with a configurable TTL (default: 5 minutes). This reduces API calls to external providers while ensuring secrets are refreshed periodically.

- Cache is cleared on `initSecrets()` call
- Cache entries expire individually based on TTL
- If a provider returns `undefined`, the system falls back to `process.env`

Rotation Procedures

JWT_SECRET

Impact: All active user sessions will be invalidated.

1. Generate new secret: `openssl rand -base64 48`
2. Update in secrets provider (Doppler/AWS/env)
3. Call `rotateSecret('JWT_SECRET', newValue)` or restart the app
4. Users will need to log in again

Recommended frequency: Every 90 days or after a suspected compromise.

DATABASE_URL (PostgreSQL credentials)

Impact: Application loses DB connectivity until updated.

1. Create new PostgreSQL credentials
2. Update PostgreSQL user: `ALTER USER drop WITH PASSWORD 'new_value';`
3. Update `DATABASE_URL` in secrets provider with new credentials
4. Restart the application (or call `rotateSecret`)

Recommended frequency: Every 90 days.

SENTRY_DSN

Status: REMOVED (MC #1271 — Sentry deinstalled)

SLACK_WEBHOOK_URL

Impact: Alerts stop sending to Slack until updated.

1. Create new incoming webhook in Slack workspace
2. Update `SLACK_WEBHOOK_URL` in secrets provider
3. Restart the application

Recommended frequency: Only on suspected compromise.

Open Banking API Keys

Impact: Bank connectivity (AISP/PISP) stops working.

1. Regenerate keys in the Open Banking provider dashboard
2. Update the relevant env vars in secrets provider
3. Restart the application
4. Verify bank account connectivity via `/api/health`

Recommended frequency: Per provider policy or every 180 days.

Environment Setup per Provider

Environment Variables (Default)

No setup required. Set secrets as environment variables:

```
# .env.local (development)
JWT_SECRET=dev-secret-do-not-use-in-production

# Production (Fly.io)
fly secrets set JWT_SECRET="$(openssl rand -base64 48)"
fly secrets set DATABASE_URL="postgresql://..."
```

```
# Production (Docker)
# Pass via -e flags or docker-compose environment section
```

Doppler

1. Create account at doppler.com
2. Create project "drop" with environments (dev, staging, production)
3. Add all secrets in the Doppler dashboard
4. Generate a service token for each environment
5. Set `DOPPLER_TOKEN` in your deployment:

```
# Fly.io
fly secrets set DOPPLER_TOKEN="dp.st.production.xxxxx"

# Docker (pass as environment variable)
```

AWS Secrets Manager

1. Create a secret in AWS Secrets Manager (JSON format):

   ```
   {
     "JWT_SECRET": "your-jwt-secret",
     "DATABASE_URL": "postgresql://...",
     "SLACK_WEBHOOK_URL": "https://..."
   }
   ```

2. Note the secret ARN
3. Ensure the application has IAM permissions for `secretsmanager:GetSecretValue`
4. Install the AWS SDK: `npm install @aws-sdk/client-secrets-manager`
5. Set `AWS_SECRET_ARN` in your deployment

Audit Trail

All secret rotation events are logged to the `audit_log` table:

Field	Value
action	`secret_rotated`
resource_type	`secret`

Field	Value
resource_id	Secret key name (e.g., `JWT_SECRET`)
details	JSON with provider name and rotation timestamp

Query rotation history:

```
SELECT * FROM audit_log
WHERE action = 'secret_rotated'
ORDER BY timestamp DESC;
```

Revision #5
Created 2026-02-18 08:44:27 UTC by John
Updated 2026-05-23 10:58:12 UTC by John