

Shy One-Time

Keep password-reset and one-time login links working when mail scanners, link previews and bots click them first.

Sound familiar?

“You have tried to use a one-time login link that has either been used or is no longer valid. Please request a new one using the form below.”

Your user didn't click it twice. Something else clicked it first. This guide explains why and shows how to fix it in about five minutes.

Project	Version · Core	Security advisory coverage
drupal/shy_one_time	2.0.x · ^9.5 ^10 ^11	Covered

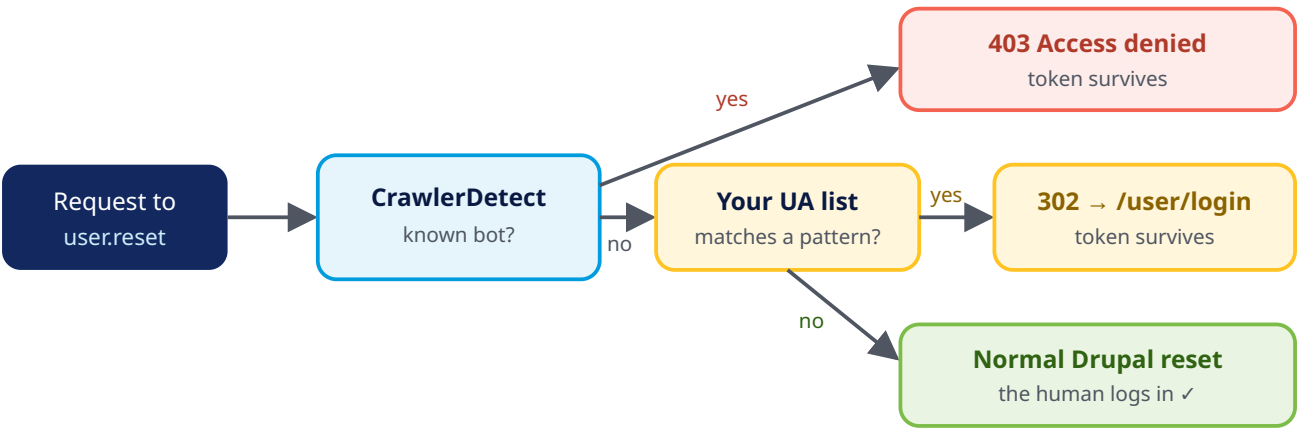
Who clicked my reset link?

Drupal's one-time login links (password reset, account activation, passwordless login) work **exactly once**. Many mail systems fetch every link in an email before the person sees it: corporate mail-security gateways, link-preview services and search crawlers. That fetch spends the token, so the user's real click fails.

It's common in enterprise email, and baffling for users. The usual result is a support ticket and a second reset request, which gets eaten the same way.

Look at who's asking before spending the token

The module listens to every request on the two reset routes, `user.reset` and `user.reset.login`, and checks the caller before Drupal spends the token:



Caller	What happens	Token
Bot recognised by the CrawlerDetect library	403 Access denied	survives
User-Agent matches a pattern you configured	302 redirect to <code>/user/login</code>	survives
Everyone else (your user's browser)	200 normal Drupal one-time login	used, as intended

Every reset-route request is logged

Each request to the reset route writes a `notice` to the site log (*Reports → Recent log messages*, channel `shy_one_time`) with the caller's User-Agent. That's how you find scanners CrawlerDetect doesn't know.

Five minutes, two commands

1 Install with Composer

```
$ composer require drupal/shy_one_time
```

No modules outside core are needed. Composer pulls in the CrawlerDetect library for you.

2 Enable the module

```
$ drush en shy_one_time -y
```

Or use *Extend* in the admin UI. It works immediately: with an empty list, CrawlerDetect alone protects your reset links.

3 (Optional) Add the scanners CrawlerDetect misses

Go to `/admin/config/system/shy_one_time` (needs the **administer site configuration** permission). Enter one User-Agent pattern **per line**, then click **Save configuration**.

```
# one per line • case-insensitive • matches anywhere
AcmeMailGateway
LinkSafety-Preview
```

4 Verify

Request a password reset to a mailbox behind your mail gateway, then check *Recent log messages* for `shy_one_time` entries. A scanner hit shows up as a logged User-Agent, and your user's click still works.

Finding the right pattern

When a user still reports “link already used”, open the log, find the `User-Agent | ...` entries just before their attempt, and copy the distinctive part of the scanner's name into the list.

Good patterns

```
AcmeMailGateway
LinkSafety-Preview/1.0
```

Specific product names that only the scanner sends.

Never do this

```
Mozilla • Chrome • Windows
```

Every real browser contains these, so **no real user can ever complete a reset**. They get redirected to login every time.

A trade, not a fix

Single-use is a security feature. It limits the damage when a reset URL leaks into a shared inbox, a forwarded message or a proxy log. Shy One-Time deliberately lets some fetches through without spending the token, so be clear about what you're trading:

A link that survives a third-party fetch can still be used by that third party

The scanner saw the URL. If it, or anyone with its logs, follows the link like a real browser, the token still works.

The User-Agent header is spoofable

The exempt group is defined by a header anyone can set. It tells you nothing about who actually fetched the link, so treat it as a convenience filter, not a security control.

Broad patterns lock real users out

A user whose real browser matches a configured pattern is redirected to login and **can never complete their reset**. Keep the list short, narrow and specific.

Typical use cases

- Stop a mail scanner consuming a reset link
- Fix “link already used” complaints
- Survive a corporate email gateway
- Prevent link-preview services burning tokens
- Keep account-activation links working
- Make passwordless or email-only login reliable
- Diagnose reset-link consumption from the log
- Reduce password-reset support tickets

Quick reference

Item	Value
Composer	<code>composer require drupal/shy_one_time</code>
Enable	<code>drush en shy_one_time -y</code>
Settings	<code>/admin/config/system/shy_one_time</code>
Permission	<code>administer site configuration</code>
Routes protected	<code>user.reset</code> , <code>user.reset.login</code>
Known bots	403 via CrawlerDetect (always on)
Your patterns	302 → <code>/user/login</code> ; one per line, case-insensitive substring
Log channel	<code>shy_one_time</code> (one notice per reset-route request)