65.21.0 (2026-10-26)

Note

💙 **Is django-allauth's authentication the entrance to your business?** Please consider supporting its continued development by becoming a sponsor at https://allauth.org/sponsors/. Your support helps keep this project thriving!

Note worthy changes

- To give more granular control over when reauthentication is required, two new settings are introduced: `MFA_REAUTHENTICATION_REQUIRED` and `SOCIALACCOUNT_REAUTHENTICATION_REQUIRED`. Additionally, you can provide custom logic via the new adapter method `is_reauthentication_required()`.

Security notice

- Headless: Reaching the phone verification attempt limit did not abort pending login/signup flows. Note that code expiry and SMS sending rate limits remained in effect.
- MFA: Fixed an issue where authentication records from an *inactive* user's passwordless WebAuthn login could carry over to a subsequent login in the same browser session, causing MFA to be skipped for the subsequent user.

65.20.1 (2026-10-20)

Security notice

- MFA: Fixed a denial-of-service vulnerability where unauthenticated passwordless WebAuthn login attempts could consume the per-user rate limit and lock a targeted account out of TOTP authentication.
- MFA: Fixed a vulnerability (introduced in 65.20.0) that allowed a TOTP code to be reused within its validity window by submitting it in an alternative representation.

65.20.0 (2026-10-16)

Note worthy changes

- A recent login no longer counts for views that require reauthentication. Users must now explicitly reauthenticate.
- OpenID Connect providers can now override values from the server's discovery document using `openid_configuration_overrides`.
- Account: Visiting signup now cancels pending login flows.

Fixes

- Headless: Cached JWT signing configuration to avoid repeatedly parsing RSA keys.
- MFA: Non-ASCII authentication codes no longer cause a server error.
- IdP: Fixed authorization code exchanges using `client_secret_basic` not returning an ID token.
- Account: The password reset-by-code timeout now restarts after the code is confirmed, allowing time to choose a new password.
- Account: On Python 3.15, every 404 response no longer emits a deprecation warning for `os.path.commonprefix()`.

Security notice

- Account: Cancelling a multi-stage login did not discard authentication records already collected. When another account was subsequently logged in using the same session, records from both accounts were mingled. Aborted login flows now discard their authentication records.
- Authorization state is now bound to the provider that initiated the flow, preventing it from being used at another provider's callback.

Backwards incompatible changes

- Dropped support for Python 3.10.
- Custom provider implementations using the `SocialLogin.stash_state()` or `SocialLogin.unstash_state()` methods must switch to `Provider.stash_redirect_state()` and `Provider.unstash_redirect_state()`, respectively. The `SocialLogin` methods have been removed.

65.19.7 (2026-10-01)

Security notice

- The Tumblr providers (both `tumblr` and `tumblr_oauth2`) used mutable blog names as the identifier for third-party accounts. If a blog name changed and was then claimed by another user, that user could log in to the original owner's account. The providers now use the stable blog UUID instead. Thanks to Sanjeev Kumar for reporting.

Backwards incompatible changes

- Tumblr: Existing `SocialAccount` records will no longer be linked due to the switch to stable identifiers. You will need to manually handle this situation, see the "Account ID Change" section in the Tumblr provider docs.

65.19.6 (2026-09-30)

Security notice

- The minimum OAuthLib version for the `socialaccount` and `idp-oidc` extras is now 4.0.0, incorporating upstream security fixes for GHSA-hj66-6f7g-4r5v and GHSA-xpv3-w29h-x7cv.
- The Bitbucket and Pinterest providers used mutable usernames as the identifier for third-party accounts. If a username changed and was then claimed by another user, that user could log in to the original owner's account. The providers now use stable identifiers instead. Thanks to Sanjeev Kumar for reporting.

Backwards incompatible changes

- Bitbucket and Pinterest: Existing `SocialAccount` records will no longer be linked due to the switch to stable identifiers. You will need to manually handle this situation by populating `SocialAccount.uid` based on `account_id` (Bitbucket) or `id` (Pinterest) located in `SocialAccount.extra_data`.

65.19.5 (2026-09-28)

Security notice

- Account: When the potentially unsafe `ALLOWED_HOSTS = ["*"]` configuration was used, arbitrary redirect URLs were considered safe, allowing open redirects. The wildcard is now ignored when validating redirect destinations. Thanks to John Tipton and Yuji Egami for reporting.
- Headless: `X-Session-Token` authentication for your own APIs (DRF / Ninja) now uses Django's standard session auth hash check. A token is rejected after the password changes outside of the allauth password-change flow, instead of still loading the user by primary key while the session row exists.

65.19.4 (2026-09-17)

Security notice

- Account: On databases with broad Unicode collations (e.g. common MySQL/MariaDB configurations), usernames such as `admin` and `ádmin`, or email addresses containing compatibility characters, could authenticate the same account while consuming different rate limit keys. This resulted in a larger than intended per account rate limit. To mitigate, matching rows are now checked against the submitted username or email address before the password is verified. On these configurations, projects using username authentication must remove Django's `ModelBackend` from `AUTHENTICATION_BACKENDS`, as it does not perform this additional check. Thanks to Yuji Egami for reporting.

65.19.3 (2026-09-11)

Fixes

- Account: The email address data migrations (`0006` and `0008`) did not honor the database selected using `migrate --database`, potentially querying or updating the default database instead. Fixed.
- MFA: Fixed various race conditions involving TOTP and recovery codes.

Security notice

- Headless: When `HEADLESS_JWT_STATEFUL_VALIDATION_ENABLED` is on, JWT access tokens are now rejected after the password changes outside of the allauth password-change flow (for example via `set_password()` in a shell). Stateful validation now uses the same session auth hash check that refresh tokens already used.
- A known flaw in the built-in rate limiting was that the configured limits could be exceeded under a high volume of concurrent requests. This was documented as an acceptably small margin of error. However, as Kaya Emre Arikan (kemrec) demonstrated, the limits could be substantially exceeded under a sufficiently high volume of concurrent requests. This is now fixed by serializing rate limit updates.

65.19.2 (2026-09-01)

Fixes

- Headless: Posting a well-formed JSON payload that was not an object (e.g. a list or a string) to the headless endpoints resulted in a server error.

Security notice

- MFA: TOTP enrollment code verification was not rate limited. Impact is limited, as to exploit this you would need to be already fully authenticated, pass (rate-limited) reauthentication, and brute force within a 30s TOTP window.

65.19.1 (2026-08-13)

Fixes

- IdP: Fixed an incorrect URL reverse in the OpenID Connect authorization template. It silently resolved to an empty form action (posting back to the same URL), so the flow kept working regardless.
- IdP: Redirectable errors from the authorization endpoint were not delivered to clients registered with an app native (non-http(s)) `redirect_uri`.

Security notice

- IdP: The OpenID Connect RP-initiated logout endpoint honored any `post_logout_redirect_uri` that could not be tied to a registered client, making it an open redirector. It now only redirects to URIs that can be verified as registered. The impact is low: no tokens or secrets are exposed, it merely allowed redirecting the user agent to an arbitrary URL after logout.

65.19.0 (2026-08-06)

Noteworthy changes

- Officially support Django 6.1.
- IdP: Added support for Client ID Metadata Document (CIMD).
- IdP: Added `IDP_OIDC_REFRESH_TOKEN_EXPIRES_IN`, allowing refresh tokens to expire. Combined with `IDP_OIDC_ROTATE_REFRESH_TOKEN`, this results in a sliding (inactivity) window. Defaults to `None` (no expiry), preserving the previous behavior. Whenever a refresh token carries an expiry, its remaining lifetime is returned to the client as `refresh_expires_in` in the token response.
- IdP: Added an `oidc_cleartokens` management command that deletes expired OpenID Connect tokens.
- IdP: Added key rotation support via `IDP_OIDC_PRIVATE_KEYS`, as well as cache control for the JWKS endpoint.
- IdP: Added RFC 7662 OAuth 2.0 Token Introspection support.
- The `jwt` and `cryptography` dependencies are now imported lazily. Deployments that register JWT-verifying providers (e.g. Google, or OpenID Connect) - for example, no longer pay their memory cost (roughly 8-10 MiB per process) unless a token is actually verified.
- On the “Change Password” form, the “Forgot password?” link has been moved into the password field help text, and the “Reauthenticate” form now shows it too. As on the login form, the link can be customized via the `account/password_reset_help_text.html` template.

Fixes

- IdP: App native URLs wouldn’t be accepted as `post_logout_redirect_uri`.
- IdP: Fix `IDP_OIDC_ID_TOKEN_EXPIRES_IN` always returning the default value, even if set in settings.

65.18.0 (2026-05-29)

Noteworthy changes

- The `password2` (password confirmation) field on the set/change/reset password forms now carries `autocomplete="new-password"`, matching `password1`. This allows browser and password manager "suggest strong password" features to fill both fields as a pair.
- IdP: Added support for Dynamic Client Registration.
- IdP: Added support for `client_secret_basic`.
- IdP: Added support for Resource Indicators (RFC 8707).
- IdP: The `.well-known/openid-configuration` endpoint previously derived `response_types_supported` and `grant_types_supported` from configured clients. Per RFC 8414, these fields should reflect server capabilities, not the configuration of existing clients. They are now statically derived. Additionally, `scopes_supported` is now included. Use the new `populate_server_metadata()` adapter method to customize the metadata.

65.17.0 (2026-05-20)

Noteworthy changes

- Added new socialaccount provider: Klaviyo.
- Rate limiting now truncates IPv6 addresses to their network prefix (default `/64`) to prevent bypass via address rotation. Configurable via `ALLAUTH_RATE_LIMIT_IPV6_PREFIX`.
- Added `authenticate_by_email` hook to `DefaultSocialAccountAdapter`, allowing customization of user lookup and email matching during social login.

Fixes

- BitBucket: When using the BitBucket API, the token is passed in the headers instead of the query parameters, which no longer works since May 4th, 2026 (deprecation notice).

65.16.1 (2026-04-17)

Security notice

- The `state` parameter is a critical part of the OAuth2 handshake, used to prevent CSRF attacks. The Edx, AngelList and Questrade providers were originally added without `state` support, as these providers did not support it at the time. Edx and Questrade have since added support, so their configuration has been updated accordingly. AngelList is no longer operational and has been removed. Thanks to Adil Ahmadzada for reporting.

65.16.0 (2026-04-13)

Noteworthy changes

- MFA: You can now configure recovery codes to be only shown once (`MFA_RECOVERY_CODES_SHOW_ONCE = True`).
- New signals for audit trail purposes: `login_code_rejected`, `password_reset_code_rejected`, `email_verification_code_rejected` (in `allauth.account.signals`) and `authentication_failed` (in `allauth.mfa.signals`).

65.15.1 (2026-04-02)

Fixes

- The context data for the various entrance views was inconsistent, e.g. some where missing `site` or `login_url`. Ensured all entrance views are now handed over the same base context.
- MFA: accessing the WebAuthn login view while already being authenticated resulted in a 500, fixed.
- OAuth 1.0: Fixed an argument call order issue when performing requests.

65.15.0 (2026-03-09)

Noteworthy changes

- All user facing codes (e.g. those that the user needs to manually input over at password reset, email/phone verification, login code, OIDC device codes) now follow the recommendations over at RFC 8628, Section 6.1. It uses dashed codes, such as “WDJB-MJHT”, by default. You can control the format of all codes via a new setting *ALLAUTH_USER_CODE_FORMAT*`, or, adjust the format per use case via one of `ACCOUNT_LOGIN_BY_CODE_FORMAT`, `ACCOUNT_PHONE_VERIFICATION_CODE_FORMAT`, `ACCOUNT_PASSWORD_RESET_BY_CODE_FORMAT`, `ACCOUNT_EMAIL_VERIFICATION_BY_CODE_FORMAT`, `IDP_OIDC_USER_CODE_FORMAT`.
- Added optional support for requesting new login codes. See `ACCOUNT_LOGIN_BY_CODE_SUPPORTS_RESEND`.

Backwards incompatible changes

- Dropped support for Python 3.8 and 3.9. Both these Python versions are end-of-life.

65.14.3 (2026-02-13)

Fixes

- Version 65.14.2 was not compatible with Python 3.8/3.9 due to use of an unsupported typing construct, fixed.

65.14.2 (2026-02-13)

Security notice

- Rate limiting and IP address detection: as Django applications cannot reliably determine client IP addresses out of the box, you must override `get_client_ip()` to match your deployment architecture. If you omitted to do so, the default implementation trusted `X-Forwarded-For`, which can be spoofed to bypass rate limits. Now, `X-Forwarded-For` is distrusted by default. You must either configure `ALLAUTH_TRUSTED_PROXY_COUNT`, rely on `ALLAUTH_TRUSTED_CLIENT_IP_HEADER`, or override `get_client_ip()`. Thanks to Ayato Shitomi for reporting.

65.14.1 (2026-02-07)

Fixes

- When using `ACCOUNT_CHANGE_EMAIL = True`, if the user initiating the change email process had no verified email address, `user.email` would still reflect the old email address while the verification process was pending.

Security notice

- SAML: When IdP initiated SSO was enabled (it is by default disabled), any URL found in the SAML `RelayState` parameter would be used to redirect to, potentially redirecting the authenticated user to a wrong site. Thanks to Ayato Shitomi and Funabiki Keisuke for reporting.

65.14.0 (2026-01-17)

Noteworthy changes

- Steam: the provider now supports initiating headless logins per redirect.
- Shopify: if `email_verified` is present in the user payload, it will be used to mark the email address retrieved as verified accordingly.
- IdP: added support for JWT based access tokens (see `IDP_OIDC_ACCESS_TOKEN_FORMAT`).
- IdP: added support for pointing to a custom userinfo endpoint (see `IDP_OIDC_USERINFO_ENDPOINT`)
- For OpenID Connect providers, you can now configure the field to be used as the account ID by setting `"uid_field"` in the relevant `SocialApp.settings`.
- Headless: the JWT algorithm is now configurable, supporting HS256.

Fixes

- IdP: Access tokens without a user attached (client credentials) were no longer recognized in DRF/Ninja endpoints.
- `requests` sessions are now disposed of after use to avoid resource leaks.