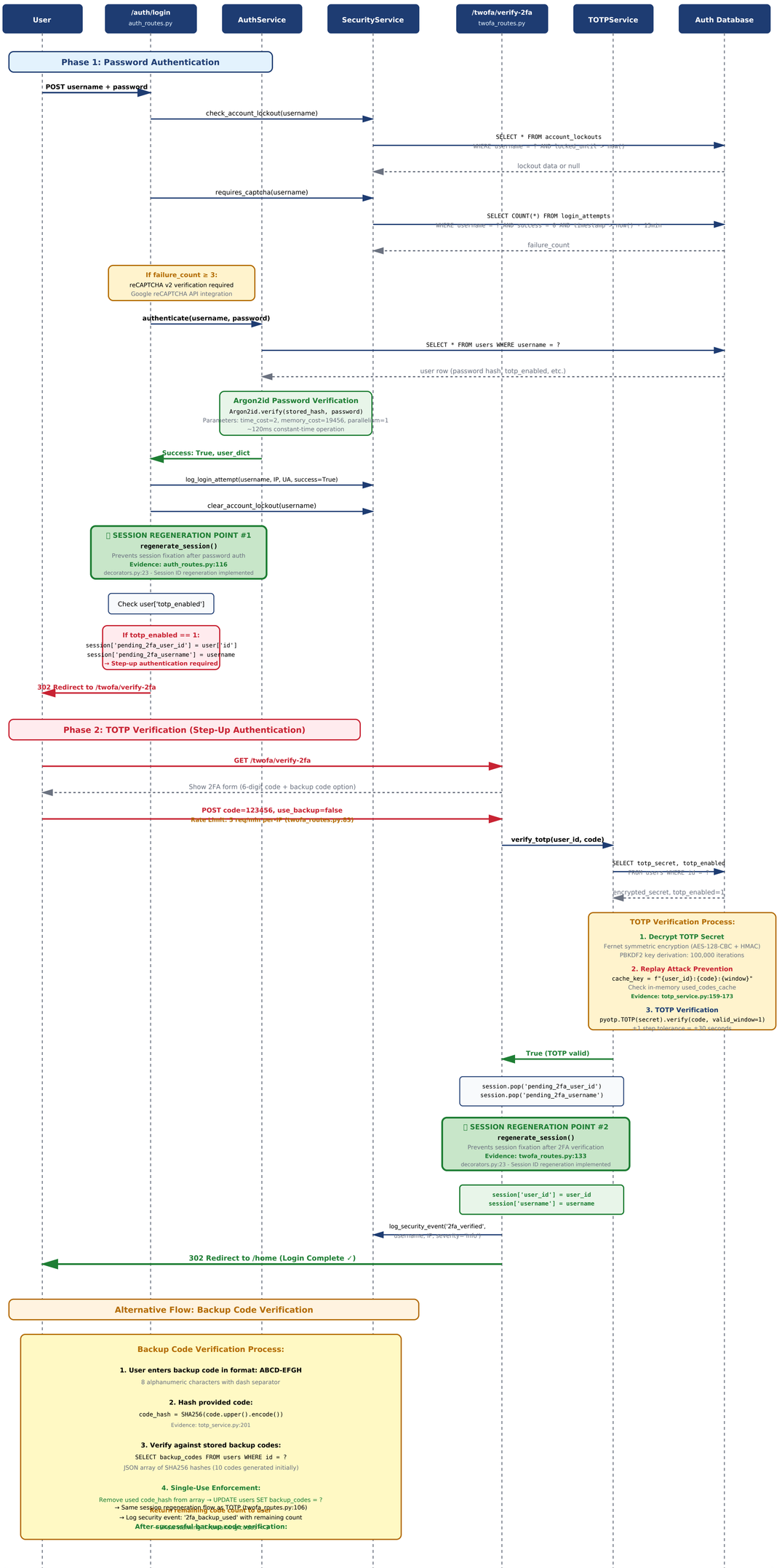
2FA Login Sequence - Complete Authentication Flow

Evidence: auth_routes.py:116, twofa_routes.py:106,133, totp_service.py:159-173



2FA Security Implementation Summary

✅ Session Fixation Prevention (IMPLEMENTED):

- Regeneration Point #1: After password authentication (auth_routes.py:116)
- Regeneration Point #2: After TOTP verification (twofa_routes.py:133) or backup code (line 106)
- Implementation: decorators.py:23 - regenerate_session() decorator
- Security: Prevents attacker from fixing session ID before authentication completes

⚡ TOTP Replay Attack Prevention:

- In-memory cache: {user_id}:{code}:{time_window} (totp_service.py:159-173)
- Time window: 30-second intervals (TOTP standard)
- Cache cleanup: Removes entries older than 90 seconds (3 windows)
- Prevention: Same code cannot be used twice in same time window

🕐 TOTP Time Window Tolerance:

- valid_window=1: Accepts codes from current window and ±1 adjacent windows
- Tolerance: ±30 seconds (1 step before and after current window)
- Balances security with clock skew tolerance (RFC 6238 recommendation)

🛡️ Additional Security Measures:

- Rate Limiting: 5 req/min per-IP on /verify-2fa
- Backup Codes: SHA256 hashed, single-use (10 total)
- TOTP Secret: Fernet encrypted at rest
- PBKDF2: 100,000 iterations for key derivation
- Security Logging: All 2FA events audited

🔐 Step-Up Authentication Pattern:

- Phase 1: Password validates identity
- Phase 2: TOTP/Backup code validates possession
- Pending session: Temporary storage between phases
- Final session: Only created after both factors pass