All this text wasn’t written by me (Tiago a.k.a TigaxMT).

All of it was transcribed by me from HackerOne video lessons.

All credits and thanks go to them.

Session Fixation

As the name implies, it's a bug which the session becomes fixated.
By putting the session ID in the query string or another insecure location, it's possible for an attacker to trick a victim into using a given session ID to log in.
Then the attacker can use that session ID for any purpose, without needing credentials.

EXAMPLE FLOW

Attacker → Victim: Go to http://innocentsite.org/login?PHPSESSID=0123456789ABCDEF
Victim → innocentsite: My session ID is 0123456789ABCDEF, please transfer all funds to account X

TWO PROBLEMS IN ONE

There are really two problems here, that are generally combined under "Session Fixation", but they're worth talking about separately:

- Session IDs in the query string are easy to leak to other users, or to other sites via Referer
- Session IDs are used as the sole mechanism to tie a browser to a user

VISIBLE SESSION IDS

These have mostly gone away, but were a huge problem up until recently. You'll still see them, mostly in legacy PHP apps.

These not only enable session fixation, but complete leakage of session credentials.

SESSION == AUTHENTICATION

Sessions are most commonly used for consistent authentication throughout the user's continued use of an application, right? That's how we usually see it.
But they're often used to store things like:

- Number of login attempts
- Saved email/username
- Many other ancillary pieces of data

MITIGATION

Getting rid of visible Session IDs is critical, but if your app/server still allows them to be passed in that way, you're no more secured from session fixation.
The key is simple: When a user logs in, they get a brand new session with a brand new ID. No matter if they just got a session for visiting the site 30 seconds ago.

This cuts session fixation off at the knees by not allowing a single session ID to persist across logins, and reinforce the proper session == authentication paradigm.