There are a bunch of different ways an “anonymous user API” could happen. None of the different methods are particularly complicated to have in core and all of them would be a huge win for the comment module and any contrib module that works with anonymous users. All of them are greatly enhanced with profile going into core. However the complicated is there are so many ways of doing the same thing and so the biggest blocker to this issue is just figuring out which one to do.

Unique Consequences are unique to that specific method.
General Consequences are not necessarily true to all methods but are consequences shared by multiple methods.
Bold methods have a patch.

Method	Description	Unique Consequences	General Consequences
Anon Flag + Unique Anon Names (Sun, DamZ, Fago seem to prefer this)	● We put a flag “anonymous” on user entities ● We make modules like comment work with users properly.	● Anonymous names will need to be unique (constrained in the DB). ○ This means anon users will have to use the same Username + E-mail combination or it will fail. ○ This can be achieved by acquisitions. All future posts by the same anonymous user will have to use the same anon user (otherwise you wouldn’t be able to post twice) ● Contrib that wants non-unique names will have to use another field and fill out a randomly generated unique username (probably from UID) ● We will need contrib to use hook_user_format_name_alter to use nicer names from other information (such as profile fields) ○ This was “hook_username_alter” in Drupal 7 ● Anonymous names take up the potential user names authenticated users could have.	● Whilst Usernames and Emails have to be typed in identically to use the same user. All other data (such as website) would probably have to be overwritten every time there is a new comment.. ○ Or, using Revisions the revision id could be stored against the comment ○ Or, if I change the website in one comment it will change for all comments. ● Fully anonymous comments can be allowed by simply not attaching a user to the comment. (This can be configurable). ○ Therefore IP address should be stored against the comment. ● There will be lots of users with an anonymous flag in the DB.UIs will have to changed to handle this. ● Information about users will mainly be stored on Profile entities.
Anon Flag + Non-Unique Anon Usernames	● We put a flag “anonymous” on user entities ● We change the database so that usernames don’t have to be unique ● We implement a higher level layer that enforces unique user names in situations where they need to be enforced that first checks if the thing being saved to is anonymous or authenticated.	● The Database schema is changed which means that modules that by-pass higher level things could potentially accidently put in non-unique usernames for authenticated users. ○ This introduces a potential DX WTF as people who make this mistake will get no feedback showing them they have made this mistake, but logins will happen strangely throughout the site. ● This will allow comments to be configurable to demand unique anonymous names or not. ● Validation can be put into entity save but will also have to be put into every form that uses anonymous and authenticated users increasing developer learning curve.	● Apart from name and e-mail. All other data (such as website) would probably have to be overwritten every time there is a new comment. ○ Or, using Revisions the revision id could be stored against the comment ○ Or, if I change the website in one comment it will change for all comments. ● Fully anonymous comments can be allowed by simply not attaching a user to the comment. (This can be configurable). ○ Therefore IP address should be stored against the comment. ● There will be lots of users with an anonymous flag in the DB.UIs will have to

			be changed to handle this ● Information about users will mainly be stored on Profile entities.
Separating the User into 2 entities - Account and User. (fubhy prefers this. This method is what the party people have implemented in contrib)	● We will create an account entity and move all data required for authentication to this entity (Username and password) ● User entities will then be exactly the same for anonymous and authenticated users. Other modules and core will continue just working with users. ● There will be a name on User and a Name on Account. For anonymous users they will be writing directly to the user name and for authenticated users they will write to the account name (which is then copied to the user name) - This is like an entity label	● Whilst this introduces the largest change to Drupal, it probably introduces the smallest change to most contrib modules having to interact with Users as the user entity will largely continue unchanged. ● There are some situations where account and user tables will have to be loaded which may hit performance but these situations are minimised. ● Allows the possibility of a fully pluggable authentication system that doesn't store irrelevant data. ● Account fields can still maintain all their constraints at the database level like core currently has. ● The configurable user label introduces some complexity into core but allows contrib a greater deal of flexibility. ○ Full Name module could be really easily made to work with core Profile module and would just be easy ○ Complicated logic involving "hook_username_alter" can be put into the save method rather than load, potentially increasing performance.	● Apart from name and e-mail. All other data (such as website) would probably have to be overwritten every time there is a new comment. ○ Or, using Revisions the revision id could be stored against the comment ○ Or, if I change the website in one comment it will change for all comments. ● Fully anonymous comments can be allowed by simply not attaching a user to the comment. (This can be configurable). ○ Therefore IP address should be stored against the comment. ● There will be lots of users with an anonymous flag in the DB.UIs will have to be changed to handle this ● Information about users will mainly be stored on Profile entities.
Separating the User into 2 bundles - anonymous and authenticated	● Users would have 2 bundles. Anonymous and Authenticated which have different fields on them	● Not fully realised as a solution yet so work needs to go into thinking how this could happen ● Makes "Acquisitions", converting an anonymous user into an authenticated user much harder, as changing the bundle on an entity is complicated (where would all the data go?) ● Means users have to be fieldable, ideally users should not be fieldable and instead profile should handle all extra information on a user. ● This may require unique	● Fully anonymous comments can be allowed by simply not attaching a user to the comment. (This can be configurable). ○ Therefore IP address should be stored against the comment. ● There will be lots of users with an anonymous flag in the DB.UIs will have to be changed to handle this
Separating the authentication "plugin"	"fubhy, I would not necessarily make a field, but keep it stored with the entity - just optional. Then I'd make a class ("plugin") that contains the logic to read account data from wherever you want." - Fago	● Means the database needs to be changed to make usernames optional or non unique and so inherits many of the issues of the Anonymous Flag. ● Don't fully understand the implications of this. ● This means that we'd store all the data on the user entity	