Item 327

git_comments:

1. Veto authentication result if account is required but unavailable due to account restrictions

git_commits:

1. **summary:** GUACAMOLE-284: Veto authentication result if a database account is required but unavailable.
   message: GUACAMOLE-284: Veto authentication result if a database account is required but unavailable.

github_issues:

github_issues_comments:

github_pulls:

github_pulls_comments:

github_pulls_reviews:

jira_issues:

1. **summary:** Database "Account Restrictions" not applied when using LDAP
   description: When using LDAP authentication and a database backend the options under "Account Restrictions" are not working. When we set the option "Disabled" or "Enable/Disable account after" this has no effect. For us the users who managing Guacamole (users and connections) do not have access to LDAP to enable/disable accounts. So it would be nice to do have these options working when using LDAP authentication with a database.
2. **summary:** Database "Account Restrictions" not applied when using LDAP
   description: When using LDAP authentication and a database backend the options under "Account Restrictions" are not working. When we set the option "Disabled" or "Enable/Disable account after" this has no effect. For us the users who managing Guacamole (users and connections) do not have access to LDAP to enable/disable accounts. So it would be nice to do have these options working when using LDAP authentication with a database.
3. **summary:** Database "Account Restrictions" not applied when using LDAP
   description: When using LDAP authentication and a database backend the options under "Account Restrictions" are not working. When we set the option "Disabled" or "Enable/Disable account after" this has no effect. For us the users who managing Guacamole (users and connections) do not have access to LDAP to enable/disable accounts. So it would be nice to do have these options working when using LDAP authentication with a database.
4. **summary:** Database "Account Restrictions" not applied when using LDAP
   description: When using LDAP authentication and a database backend the options under "Account Restrictions" are not working. When we set the option "Disabled" or "Enable/Disable account after" this has no effect. For us the users who managing Guacamole (users and connections) do not have access to LDAP to enable/disable accounts. So it would be nice to do have these options working when using LDAP authentication with a database.
5. **summary:** Database "Account Restrictions" not applied when using LDAP
   description: When using LDAP authentication and a database backend the options under "Account Restrictions" are not working. When we set the option "Disabled" or "Enable/Disable account after" this has no effect. For us the users who managing Guacamole (users and connections) do not have access to LDAP to enable/disable accounts. So it would be nice to do have these options working when using LDAP authentication with a database.
6. **summary:** Database "Account Restrictions" not applied when using LDAP
   description: When using LDAP authentication and a database backend the options under "Account Restrictions" are not working. When we set the option "Disabled" or "Enable/Disable account after" this has no effect. For us the users who managing Guacamole (users and connections) do not have access to

LDAP to enable/disable accounts. So it would be nice to do have these options working when using LDAP authentication with a database.

7. **summary:** Database "Account Restrictions" not applied when using LDAP
   description: When using LDAP authentication and a database backend the options under "Account Restrictions" are not working. When we set the option "Disabled" or "Enable/Disable account after" this has no effect. For us the users who managing Guacamole (users and connections) do not have access to LDAP to enable/disable accounts. So it would be nice to do have these options working when using LDAP authentication with a database.
   label: code-design
8. **summary:** Database "Account Restrictions" not applied when using LDAP
   description: When using LDAP authentication and a database backend the options under "Account Restrictions" are not working. When we set the option "Disabled" or "Enable/Disable account after" this has no effect. For us the users who managing Guacamole (users and connections) do not have access to LDAP to enable/disable accounts. So it would be nice to do have these options working when using LDAP authentication with a database.
9. **summary:** Database "Account Restrictions" not applied when using LDAP
   description: When using LDAP authentication and a database backend the options under "Account Restrictions" are not working. When we set the option "Disabled" or "Enable/Disable account after" this has no effect. For us the users who managing Guacamole (users and connections) do not have access to LDAP to enable/disable accounts. So it would be nice to do have these options working when using LDAP authentication with a database.

jira_issues_comments:

1. It sounds like maybe there's some confusion or missing information with how you have authentication set up. Do you have MySQL authentication only, or are you layering MySQL with LDAP? Based on your description it sounds like you're doing the later, and, the way authentication layering currently works in Guacamole, disabling the account will only disable authentication of the account via the database module, it won't actually block a login, as authentication will succeed via the LDAP module. When authentication succeeds, the user will be logged in, and then the user's permissions will be aggregated from other authentication sources that contain the same username. So, disabled, time restrictions, and account expiration settings inside the database modules will not impact logins that happen via another module when multiple modules are layered.
2. {quote} So, disabled, time restrictions, and account expiration settings inside the database modules will not impact logins that happen via another module when multiple modules are layered. {quote} While it's true that account restrictions defined within the database auth shouldn't affect whether another authentication mechanism succeeds/fails, I'd say those restrictions should still take effect when it comes to providing access to the data actually defined within the database. It makes sense that the LDAP authentication would succeed, but I'm not sure it makes sense that access to the connections, etc. within the database would be granted for an account which is disabled (or otherwise restricted) within the database, particularly with respect to the {{mysql-user-required}} / {{postgresql-user-required}} properties.
3. {quote} While it's true that account restrictions defined within the database auth shouldn't affect whether another authentication mechanism succeeds/fails, I'd say those restrictions should still take effect when it comes to providing access to the data actually defined within the database. {quote} I agree. I was commenting on how it currently works, not, necessarily, on how it should work :-). However, the flip-side of this is making sure that it's understood how to properly secure database accounts in the above scenario, if necessary, to prevent accounts that may not have a password set on them from being exploited. That may already be taken care of in the Guacamole code - I did try to create a database user without a password and log in with it and it did not work, so this may not be a concern at all? Anyway, I agree that disabling the account in the DB module should result in the connection information for that user being inaccessible, even if another module succeeds.
4. {quote} That may already be taken care of in the Guacamole code - I did try to create a database user without a password and log in with it and it did not work, so this may not be a concern at all? {quote} Yep. You can't set an account without a password in the database auth. When you leave the password blank for a new account, a lengthy random password is set.
5. Cool. So simple solution to this seems to be to check for disabled/expired accounts in the methods that return configurations for the accounts, and just return empty or null if that's the case. Is that the route to go, or is there a more proper/elegant way you'd go about it?

6. More or less, yes. I'm going to try moving those checks around, such that they are taken into account when the "mysql-user-required" or "postgresql-user-required" options are enabled.
7. **body:** So far so good - moving the enforcement of account restrictions seems to solve this issue and actually cleans things up a bit.
   label: code-design