Summary

Currently Zabbix does not provide any granular control on what parts of UI and what functionality (like ability to acknowledge problems) is available to end users.

It would be great if Zabbix could provide such functionality, it would allow Zabbix to support a broad range of new use cases.

Zabbix acceptance

1. Zabbix will support user roles
   a. User role is a combination of
      i. Name: unique name of the user role
         1. Special role "Super Administrator" will be defined and cannot be edited or removed, it will provide access to all functionality of Zabbix
      ii. User type: User, Admin or Super Admin
      iii. Access to UI elements
         1. It will contain a table of all top-level UI menu items and "Enabled" flag [X] - yes, [ ] - no
            a. Default access to new elements [X]
            b. Monitoring: Dashboards [X]
            c. Monitoring: Problems [X]
            d. Monitoring: Latest Data
            e. ...
            f. Inventory: Overview [X]
            g. ...
            h. Administration: Queue [X]
         2. If UI element is disabled then it will be excluded from the menu and cannot be reached by URL
         3. If user type User or Admin is selected then it will be impossible to control access to Configuration and Administration and some UI elements (like Monitoring→Discovery), it will be read-only and disabled
         4. If a menu item is disabled then Zabbix UI will not contain links to this and underlying pages
            a. For example:
               i. Monitoring->Problems [ ]: no access to Monitoring→Problems, event details
                  1. Therefore there will be no links from Dashboards widgets to event details, for example
      iv. Access to modules
         1. It will contain a table of all enabled modules and "Enabled" flag [X] - yes, [ ] - no
            a. Default access to new modules [X]
            b. Module A [X]
            c. Module B [ ]
      v. Access to API:
         1. [ ] - no access
         2. [X] - access to API is enabled
            a. API methods whitelist: list of allowed API methods
               i. Empty list means that all methods are allowed
            b. API methods blacklist: list of disabled API methods
               i. Empty list means that there are no restrictions
               ii. Note sure if white/blacklist naming is politically correct nowadays. Feel free to propose an alternative. 🙂
            c. If a API method matches two lists then the method will be disabled
            d. The lists will support wildcard for API object and method names. For example: host.*, *.delete, *.*
            e. Note that user.login and user.logout API methods will be available if at least one method is allowed

            vi. Allowed actions
                1. It will contain a table of various user actions
                    a. Default access to new actions [X]
                    b. Create and edit dashboards and screens [X]: allow users to create, edit, delete and share dashboards and global screens
                    c. Create and edit maps [X]: allow users to create, edit and delete maps
                    d. Create and edit maintenance [X]: allow users to create, edit and delete maintenance periods
                    e. Ability to acknowledge (update) problems [X]: allow users to use acknowledgement functions
                    f. Ability to execute scripts [X]: allow users to execute scripts
                2. If some action is disabled then Zabbix UI will not provide this functionality
                    a. For example:
                        i. Create and edit dashboards and screens [ ]: there will be no buttons or other controls to edit, create and share dashboard and screens

2. Each user must have one and only one role assigned
3. Administration→Users must be modified to
    a. Replace column and filter "User type" by "User role"
    b. Add a column and filtering option "API access"
    c. Tab "Permissions" must be extended to display all information related to permissions: user role, user type, access to UI elements, access to modules, access to API and allowed actions
4. A database patch must be created in order to create three roles "Super Administrator", "Administrator" and "User" and assign them to existing users of corresponding user types
5. Breaking change: users of user type "User" will have read-only access to everything
    a. Now we need ONLY read-only permissions to hosts in order to enable event acknowledgements, creation of maintenance periods, etc
    b. It must be well documented in release notes

Nonfunctional requirements

1. N/A

Use cases

1. I want some users to have permissions to dashboards only
2. I want to hide Monitoring→Services for all users
3. I want to disable editing of dashboards and screens for some users
4. I want to restrict problem acknowledgements for certain users
5. I want to restrict creation of maintenance periods for certain users
6. I want to restrict execution of scripts for certain users

Decisions made

1. N/A

Open questions

1. Perhaps we should also enable/disable front-end access on user role-level (move it from user groups)

    a. It would be great to split front-end access and authentication method
2. Perhaps we should control debug mode here as well (move it from user groups)

Changes log

- **1.1**
    - Updated 1.a.v.e: Note that user.login and user.logout API methods will be available if at least one method is allowed
    - Removed "comma delimited" from description of white/black-lists for API access