How to Report a Problem or Request a Feature

Prepare

First, read the excellent essay How to Report Bugs Effectively. Then read it again. Really.

Make sure that you have searched this wiki for relevant information, and search the tracker if the issue already exists and also search the forum for relevant discussion.

If you don't find the answer to your question, then think hard about how to ask.

Post a Question in the SIPfoundry Forum

You need an account on the SIPfoundry infrastructure to post to the forum.

Remember everything you learned about *How To Report Bugs Effectively* when writing your email. Be nice. Be patient - everyone who ever posts to those lists has some other job to do - if you want immediate answers, there are commercial versions of sipXecs for which you can buy support. In general, the more carefully and nicely you ask, the quicker and more useful the answer will be.

If you include information from logs, it is best **not** to reformat them or post the prettier versions output by syslogviewer: instead, *attach* the log segment to your email. This keeps the email programs from line-wrapping or otherwise modifying the log data, which often interferes with tools developers use to examine and display them.

Use the Issue Tracker

If, after discussion on the list, it seems necessary to create an issue in the project tracker, then you can do that.

You must create an account in the tracker.

Create a New Issue. Be thorough - err on the side of putting in too much information, not too little. Use the sipx-snapshot tool (either through the sipXconfig user interface or as a shell command) to capture your configuration and attach it. If possible, before creating the snapshot:

1. Reset the log level to at least INFO (this provides message tracing) on the component(s) involved in the problem - always include the sipXproxy and the registrar (since they are involved in just about everything).
2. Stop the sipXecs services and delete (or move aside) the existing log files (`${prefix}/var/log/sipxpbx/*.log`)
3. Start the sipXecs services
4. Reproduce the problem.
5. Identify the call(s) that show the problem: the best identification is the call-id value, but the calling and called addresses (phone numbers) and a precise time are OK. (Record the time in UTC, as that is how time is recorded in the log files.)
6. Create the sipx-snapshot and attach it to your issue.

If your system does not have a large disk, you may want to change your log level back to the default NOTICE level after this; the INFO level logs can get big pretty fast on a busy system.

You can see how your issue is progressing by watching its Status. If you used a valid email address when you created your account (you did, didn't you?), then you'll get email whenever the issue is updated. Because you created the issue, you have special status with respect to it - you are the Reporter.

Special Note if this is a security related issue:

Please also set the 'Security' level in the New Issue to 'Security Issue'. If this problem affects current users, we want to fix the issue as soon as possible, and not expose existing installations to additional danger until it is fixed. Please see special information on security at http://wiki.sipxcom.org/display/sipXecs/Security+Team

The workflow is illustrated below. The nodes are Issue States, and the arrows between them are Transitions.

? Unknown Attachment

The first couple of states are for issues that are new, or at least have not yet been acted on in some way and implement the community development process:

New

- This is the initial state for all issues. An issue in this state has not yet be reviewed. Someone (usually the project coordinator, but it can be any developer), will review it and either assign it back to the Reporter to ask for more information (which moves it to Need Information state), Accept it (which moves it to Open state), or Reject it for issues that have not been accepted (which moves the issues to the Closed state). The issue is moved to the Reproduce state by a developer to ask the QA team to verify or reproduce the issue. Feature requests that the project team will not work on but are relevant to the community are moved into the Community state for community development.

IceBox

- This means the issue has been accepted. The issue is legit and described well enough that it is possible to start work on it. It does not mean that anyone is doing that work yet. Issues can sit in the Open state for a long time, depending on priorities. Community can pick up issues, do the development, and attach a Patch for review.

Open

- This state is reserved for the sub-task workflow and represents an open sub-task.

Reproduce

- Issues are moved into the Reproduce state by developers if QA verification is needed. This typically happens for bugs that we would like to reproduce before accepting the issue.

Community

- New feature requests that either the project team will likely not work on or issues that are well suited for community contributions are moved into this state. The community can pick them up from here, do the development, and attach a patch for review. This will make it much more likely that the issues gets resolved.

Need Information

- This state means that something more is needed before the issue can be accepted. If an issue assigned to you is in this state, you should Return it when you provide the information, which will move it back to the New state (where it will once again go through the review process).

Patch Pending

- Community members who would like to submit a patch move issues into this state. This is similar to the New state in that it indicates that review by a developer is needed. If it stays in this state too long, send email to the project coordinator to call attention to it, but please be patient. The reviewer may apply the patch and move the issue to Resolved state, or may request that the submitter change it in some way (which moves the issue back to Open state).

The next four states reflect the development process:

Backlog

- Product Management moves issues from the Open state into Backlog to queue them for development. Backlog represents the queue of issues, sorted by priority, for the development team to work on in the next iteration.

Awaiting Development

- Issues in this state are scheduled for this iteration of development. This is the queue from where developers pick issues to work on.

In Progress

- This state means that some developer is actively working on the issue.

Code Review

- Once the developer finishes work the issue is moved into the Code Review state. Another developer will pick up the issue from here, perform code review, and put it back into the queue for this iteration.

These two states reflects the QA process:

Awaiting QA

- Once development is finished and the code has been submitted into the unstable repository. the issue is assigned for QA for verification. This represents functional test.

Testing in Progress

- The QA team is working on testing the issue.

Reopened

- The main purpose of this state is for QA to reopen an issue and assign it back to engineering if verification failed. Product management or developers will pick up issues from this state and correct the work previously performed. It can be picked up directly from here by the developer, which transitions the issue back into the In Progress state. In more special cases issues that have been Resolved or even Closed can be reopened.

The last two states are the goal states:

Resolved

- This state indicates that the issue is resolved and has passed QA. The issue is assigned back to the reporter when moved to this state to see if the reporter agrees; if so, the Reporter should Close the issue - if not, Reopen it.

Closed

- The final state - the issue is done and nothing further need happen.