How does work get scheduled?
Aim of this talk

- Give some clarity about how we plan the work
- Show how public GitHub activity reflects our work
- Explain how you can influence the roadmap
The planning process
Where does the work come from?

• Many sources
  • Responding to bug reports  Unplanned
  • Community PRs
  • User requests
  • Internal plans  Planned

• Not enough capacity to handle all unplanned work immediately
  • Some of this work will get done immediately
  • Some will get prioritised and scheduled

• Roadmap reflects the larger planned work items
  • Subject to change according to end-user needs
The basic process

- Roadmap gives **quarterly** deliverables
  - Community PRs and issues not shown (but expected to be part of our work)

- **Monthly** sprints
  - Before each sprint, time is spent planning sprint activities
  - These go into a GitHub board used to track progress

- **Daily** triage on incoming items (bug reports & PRs)
  - Typically should see initial triage activity within a working day

- Small community items get handled as they come in
  - Aim to be responsive – median time to close a community PR is around 12 days

- Large community items get scheduled for a future sprint
  - May be added to the roadmap for major items
  - Example: [DTLS-SRTP support](#) recently merged as a planned roadmap item
Public GitHub activity

How does this process map to activity in GitHub?
Our GitHub setup

- **mbedtls** repo is the main project
- We have a **private repo** for tracking security issues and PRs
  - When these are ready they are merged into the main project
- **mbedtls-test** holds test infrastructure

- Community issues and PRs are managed via various labels

- Work is tracked via some GitHub boards
  - Unified board
  - Product backlog
  - Epics board
Incoming issue or PR

- Triage (normally within one working day)
  - Item will be labelled
  - Author may receive comments, feedback, questions

- Labels may include:

<table>
<thead>
<tr>
<th>Label</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Community</td>
<td>Applied to all community items – you will always see this on your PRs and issues</td>
</tr>
<tr>
<td>Bug</td>
<td>Bug reports</td>
</tr>
<tr>
<td>Product Backlog</td>
<td>PR is too big to process immediately - Or, an issue that goes into the backlog</td>
</tr>
<tr>
<td>Needs:review</td>
<td>PR needs review</td>
</tr>
<tr>
<td>Needs:reviewer</td>
<td>PR needs someone (else) to review it</td>
</tr>
<tr>
<td>Needs:CI</td>
<td>PR can't progress until it passes CI</td>
</tr>
<tr>
<td>Needs:work</td>
<td>PR has been sent back to the author for additional work</td>
</tr>
<tr>
<td>Needs:changelog</td>
<td>PR needs to have a changelog entry added</td>
</tr>
<tr>
<td>Needs:backports</td>
<td>PR requires backporting to LTS branches</td>
</tr>
<tr>
<td>Ready For Merge</td>
<td>PR is ready to merge and is waiting for someone to push the button</td>
</tr>
<tr>
<td>Bug</td>
<td>Bug report</td>
</tr>
<tr>
<td>Good first issue</td>
<td>Suitable for a novice contributor to pick up this issue (simple, self-contained)</td>
</tr>
<tr>
<td>Help wanted</td>
<td>There are no immediate plans to work on this, but we would certainly welcome a community PR</td>
</tr>
<tr>
<td>Question</td>
<td>Issue is a question (normally these should now go to the mailing list)</td>
</tr>
<tr>
<td>Wontfix</td>
<td>Issue or PR won't be accepted (e.g., bug report that is actually user error)</td>
</tr>
<tr>
<td>Enhancement</td>
<td>Issue is an enhancement to current features, thus unlikely to get backported.</td>
</tr>
</tbody>
</table>
What to watch for in your PR

• Community
  • When you see this label get added, you know that triage has picked it up
  • This should happen relatively promptly (normally within a working day)

• needs: reviewer and needs: review
  • When these disappear, you can see that someone has been assigned, or started review

• Comments
  • After review, you’ll likely get some comments and requests for further work

• Ready for merge
  • When you see this, you can expect it to land soon with no further work required
PR lifecycle example

Median time from start to finish is currently about 12 days

- **PR submitted**
- **PR triaged**
  - Labels added: *community, needs: review, needs: reviewer*
- **Reviewers assigned**
  - Label removed: *needs: reviewer*
  - PR now has people assigned to review it
- **Review completed**
  - Label removed: *needs: review*
  - Label added: *needs: work*
  - Comments left highlighting some changes that are needed
- **Author updates the PR**

- **Reviewer spots that it’s been updated**
  - Label removed: *needs: work*
  - Label added: *needs: review*

- **Review completed**
  - Label removed: *needs: review*
  - Label added: *ready for merge*

- **PR merged**
  - Label removed: *ready for merge*
  - PR closed
Getting a feature into Mbed TLS
Main routes to influencing the roadmap

- **Existing roadmap:** [https://developer.trustedfirmware.org/w/mbed-tls/roadmap/](https://developer.trustedfirmware.org/w/mbed-tls/roadmap/)
- **Contact us to discuss**
  - Mailing list: [mbed-tls@lists.trustedfirmware.org](mailto:mbed-tls@lists.trustedfirmware.org)

- **Submit a PR**
  - Small updates will go in very quickly
  - For large features, the review time can be significant
  - If necessary, we will schedule review so that time is set aside for the feature

- **Collaborate**
  - Example: Silicon Labs are helping with a range of tasks relating to hardware accelerators
    - PRs
    - Reviews
    - Specification work
    - Discussions on the design of new features