Interview guide

This interview guide is organised into headings that identify the main themes or lines of enquiry, under which individual questions to be asked are given. The main questions are listed first, with possible follow-on 'probes' underneath in *italics* that may be used if required.

Due to the nature of a semi-structured interview, the questions may not be asked exactly as written or in the same sequence, but may be adjusted depending on how the conversation flows. The intention is that they serve mainly as a reminder or checklist for the interviewer.

Introduction (~5 mins)

Question

- What were your first experiences of regular expressions?
  - *How did you view them at first?*
  - *Have your views about them changed?*

Regular expression development processes (~30 mins)

Context clarification

Developers use regexes in a variety of different contexts (e.g. in text editors). To provide a clearer focus for the interviews and allow for consistent analysis of responses, participants should be asked to answer the following questions specifically in the context of **production code** usage.

Questions

- What is your typical process when creating a regular expression?
  - *Why do you follow this process?*
  - *Which do you find the most problematic parts of this process?*
  - *How often do you get a regex right first time?*
- How is your process different when modifying an existing regex you are not familiar with?
  - *What steps do you take to make this process easier or safer?*
  - *What about the original code helps you in this process? Or makes it harder?*
- When and how do tools or other resources feature in your process?
  - *Why do you use tools in this way/not use tools?*
  - *Are there any situations where you would be more likely to consider using a tool?*
- Could you explain how you usually validate your regular expressions?
  - *Do you ever write automated tests for them? Why/why not?*
  - *How important is it to you to validate regular expressions?*
  - *Do you ever ask colleagues for input or advice when working on regular expressions?*
- How do you obtain sample input for testing regular expressions?
  - *Do you preserve any sample input that you use?*
  - *Have you ever heard of/used input generation tools? (e.g. brics, rex)*
- Do you document your regular expressions?

  - *Why do you document in this way/not document?*
- What is your approach to reviewing code that includes regular expressions?
  - *Why do you code review regexes in this way?*
  - *What are the biggest challenges you face in such a task?*
  - *What would help you most? (e.g. documentation, tooling)*
- What recommendations would you give an inexperienced developer about regex development?
  - *For example, what about: testing, validation, documentation*
  - *Are there any things you would advise them against doing?*

Performance and ReDoS (~5 min)

Example

The following example may be given if required (condensed from survey instrument):

> Some regular expressions' execution time increases disproportionately as the input gets longer. If such a regex is running on a server and matching against user input, an attacker may be able to send a malicious request that seriously slows down the server or causes it to hang indefinitely.

Questions

- Have you ever had to deal with a performance issue in a regular expression?
  - *How did/would you go about resolving it?*
- Are you familiar with the concept of Regular Expression Denial-of-Service (ReDoS)?
  - *If not, give example above*
- How would you handle a confirmed report of a ReDoS vulnerability in your code?
  - *How serious a risk do you consider ReDoS?*
  - *What tools or information would help you handle vulnerabilities more effectively?*

Closing (~5 min)

Questions

- How important is it for an effective developer to have regex knowledge?
  - *Why do you think this is?*
  - *In your view, has this importance changed over time? Why?*
  - *Does the importance depend on the stack/developer's role?*