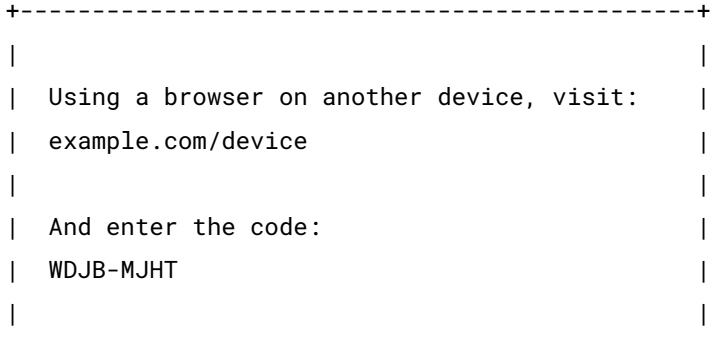
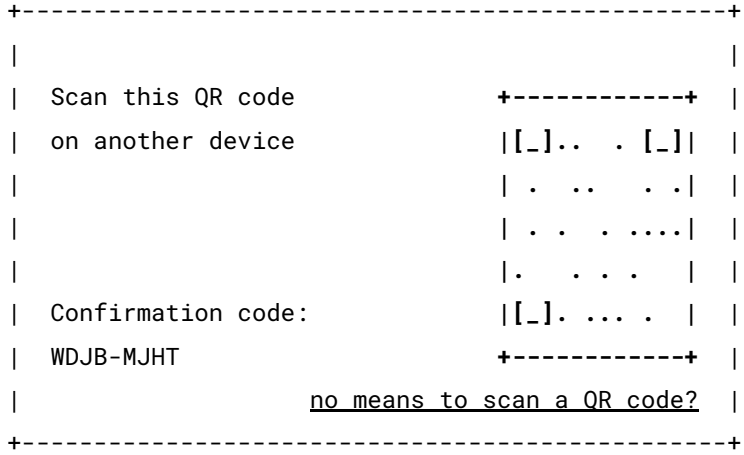
The importance of non-textual code transmission in RFC 8628

IETF 113, March 24 2022, Filip Skokan

Textual vs. non-textual



vs.



3.3.1. Non-Textual Verification URI Optimization

When "verification_uri_complete" is included in the authorization response (Section 3.2), clients MAY present this URI in a non-textual manner using any method that results in the browser being opened with the URI, such as with QR (Quick Response) codes or NFC (Near Field Communication), to save the user from typing the URI.

For usability reasons, it is RECOMMENDED for clients to still display the textual verification URI ("verification_uri") for users who are not able to use such a shortcut. Clients MUST still display the "user_code", as the authorization server will require the user to confirm it to disambiguate devices or as remote phishing mitigation (see Section 5.4).

If the user starts the user interaction by navigating to "verification_uri_complete", then the user interaction described in Section 3.3 is still followed, with the optimization that the user does not need to type in the "user_code". The server SHOULD display the "user_code" to the user and ask them to verify that it matches the "user_code" being displayed on the device to confirm they are authorizing the correct device. As before, in addition to taking steps to confirm the identity of the device, the user should also be afforded the choice to approve or deny the authorization request.

Description

1. end-user mistypes *`verification_uri`* (or enters it to a search engine) and either
   a. lands on a phishing site directly
   b. lands on a search engine result page full of landmines, ends up on a phishing site anyway
2. enters the *`user_code`* and is presented with a phone number to call, a scripted scenario takes place
3. agent will first ask for an email address, with the end-user on the phone it is easy to drive the end-user's attention away from the fact they're being guided to hand over a password reset code to the agent ("magic" login link is another possibility)
4. agent is now authenticated, they can run the rest of their script (buy digital products with a saved payment method, attempt to upsell fake services, etc)
5. **at the end** of it the agent enters the right code (asks for it over the phone) at the *`verification_uri`* and **the end-user is logged in on their consumption device, completely unaware of that they've just been wronged and their account was compromised**

Key points

- End-user initiated the authorization flow, not the attacker
- End-user has a legitimate stake in completing the authorization flow
- Improving consent or confirmation screens on the “other” device does nothing to prevent this type of attack
- End-user walks away satisfied and detection is delayed

What’s next?

- Is this an end-user problem?
  - Yup. There are warning posts along the way that the end-user ignored, but we could still do better
- **If we preferred non-textual code transmission in our interfaces, end-user would not end up on a phishing site**
- Implement password reset links, not password reset codes

Question: Other than QR code scanning (present in every default mobile phone “camera” app since a few major mobile OS versions ago), what other convenient and readily available non-textual code transmission method can we recommend?