What are Criteria for Writing Good Requirements?

Do not write like this

• Ambiguity – *or*
  – *The same subsystem shall also be able to generate visible or audible caution or warning signal for the attention of security or business analyst*

• Multiple requirements – *and, or, with, also*
  – *The warning lamp shall light up when system intrusions is detected and the current workspace or input shall be saved*
Do not write like this

• Let-out clauses
  *if, when, except, unless, although, always*

  – The fire alarm shall always be sounded when the smoke is detected, unless the alarm is being tested when the antivirus is deployed

• Long rumpling sentences

  – Provided that the designated input signals from the specified devices are received in the correct order where the systems is able to differentiate the designators, the security solution should comply with the required framework of Section 3.1.5 to indicate the desired security states

Do not write like this

• System design:
  no *names of components, materials, software objects/procedures, database fields*

  – The antenna shall be capable of receiving FM signals, using a copper core with nylon armoring and a waterproof hardened rubber shield

• Mix of requirements and design:
  no references to *system, design, testing, or installation*

  – The user shall be able to view the current selected channel number which shall be displayed in 14pt Swiss type on an LCD panel tested to standard 657-89 and mounted with shockproof rubber washers
Do not write like this

• Speculation
  
  usually, generally, often normally, typically

  – Users normally require early indication of intrusion into the system

• Vague, undefinable terms
  
  user-friendly, versatile, approximately, as possible, efficient, improved, high-performance, modern

  – Security-related messages should be versatile and user-friendly
  – The OK status indicator lamp shall be illuminated as soon as possible after system security self-check is completed

Do not write like this

• Wishful thinking

  100% reliable/ safe/ secure. Handle all unexpected failures. Please all users. Run on all platforms. Never fail. Upgrade to all future situations.

  – The gearbox shall be 100% secure in normal operation.
  – The network shall handle all unexpected errors without crashing.
Good requirements

• Use simple direct sentences
  – Security analyst should be able to view system status

• Use a limited vocabulary
  – Security analyst should be able to change the infected component in less than 12 hours
  – Security analyst should be able to reconfigure the infected component in less than 12 hours

Good requirements

• Identify the type of user who wants each requirements
  – The navigator shall be able to…

• Focus on stating result
  – … view storm clouds by radar …

• Define verifiable criteria
  – … at least 100 km ahead.
  – Acceptance criterion: Aircraft flying at 800km/h at 10,000 meters towards a known storm cloud indicated by meteorology satellite report; storm cloud is detected at a range of at least 100 km.
Criteria for Writing Good Requirements

- **What**, not how (external observability)
  - Avoid premature design or implementation decisions
- **Understandability, clarity** (not ambiguous)
- **Cohesiveness** (one thing per requirement)
- **Testability**
  - Somehow possible to test or validate whether the requirement has been met, clear acceptance criteria
  - Often requires quantification, this is more difficult for security than e.g. for performance
    - "The response time of function F should be max 2 seconds"
    - "The security of function F should be at least 99.9 %"

Solution Examples

**SecReq.1**: After a successful submit lock the submission interface

**SR1**: ERIS shall encrypt the messages between Umpire and ERIS

**SecReq.2**: ERIS developers should revise and update security features on old codes used in system
Solution Examples

**SecReq.4**: Implementation front-end and back-end input validation that is applied to all input data

**SecReq5**: ERIS should allow Umpires to use additional factor in the authentication process

**SecReq6**: Application user should have properly authorized privileges without additional privileges

**SecReq7**: If SQL injection has occurred, script injection can be mitigated by use of content security policy in the HTML of the frontend, which disallows all script tag injections.