Assurance Case (AC) Role in DevSecOps Pipeline: An Example

John Goodenough
Chuck Weinstock
Carol Woody
Bob Ellison

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Document Markings

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

DM21-1074

Goals of Presentation

Show potential role of a pipeline-oriented (DevSecOps) assurance case (AC)

- Prior use of ACs focused on gaining release decision from oversight body

Show potential value/benefits of a pipeline-oriented AC

- Justify exit criteria for pipeline stages
- Define evidence needed to meet (evolving) exit criteria
- Provide basis for reassurance activity, e.g.,
  - What evidence needs to be refreshed to maintain confidence that (relevant) exit criteria continue to be met

A properly annotated AC defines exit criteria for each pipeline stage as well as showing how each stage contributes to overall system assurance

Presentation Approach

Show (by example) how AC helps define exit criteria for the PLAN phase

- AC justifies what we need to know before something can be released to the IMPL phase
  - Show how the AC could be used in the PLAN phase on each iteration of the DevSecOps loop

Construct an example for the PLAN phase

- Choose a desired system property, e.g., availability
- Focus on one class of reasons why the property might not hold, e.g.,
  - Resource exhaustion (due to poor design or unauthorized user actions)
- Show part of a possible AC for this situation (**use CWE to start with**)
- Contrast exit criteria for the PLAN phase from criteria for the IMPL phase
- Postulate a change to the system and show the **reassurance** case

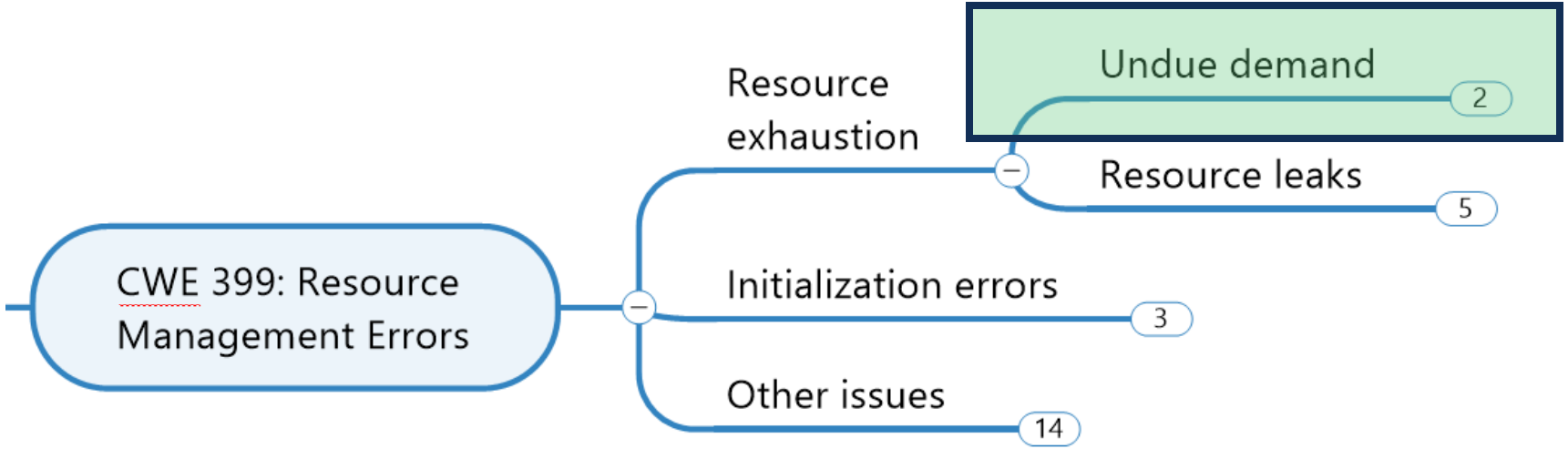
The Common Weakness Enumeration

The Common Weakness Enumeration (CWE)

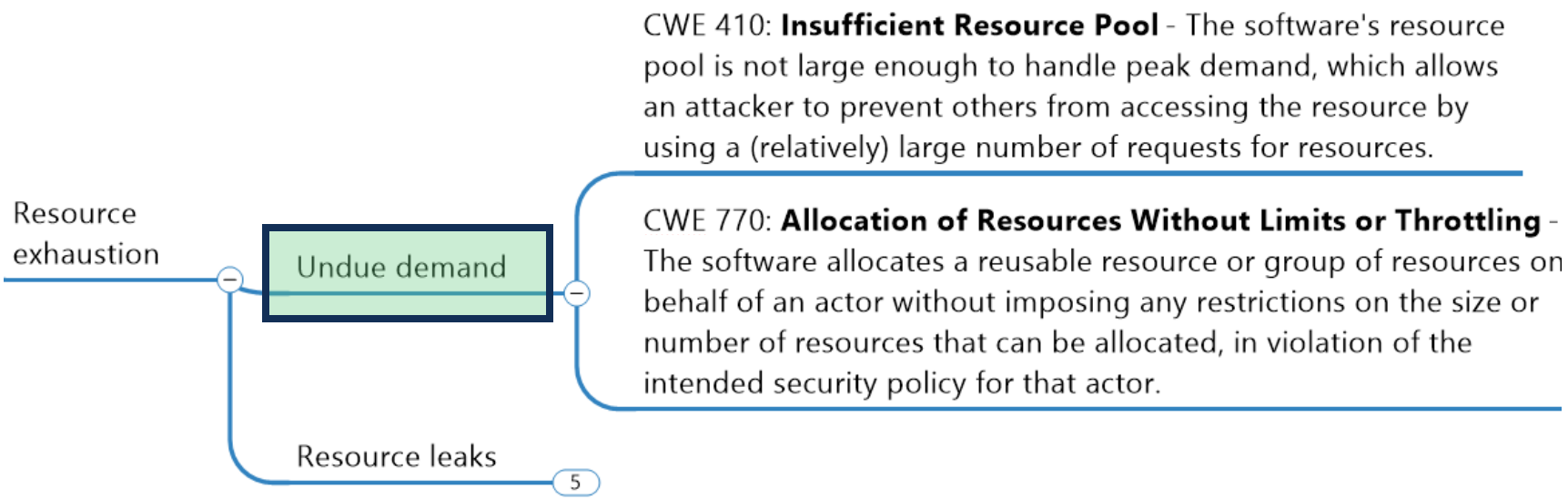
- A list of commonly seen types of SW/HW design and implementation weaknesses
- Examples of categories: Authentication Errors, Authorization Errors, Bad Coding Practices, Concurrency Issues, Data Validation Issues, Privilege Issues, Resource Management Errors
- There are 21 CWEs in the **Resource Management Errors** category (CWE 399)
  - We have grouped these into
    - Resource Exhaustion Errors (4 CWEs)
    - Initialization Errors (3 CWEs)
    - Other Issues (14)
- In this presentation we focus on **Resource Exhaustion Errors**

Exploit CWE

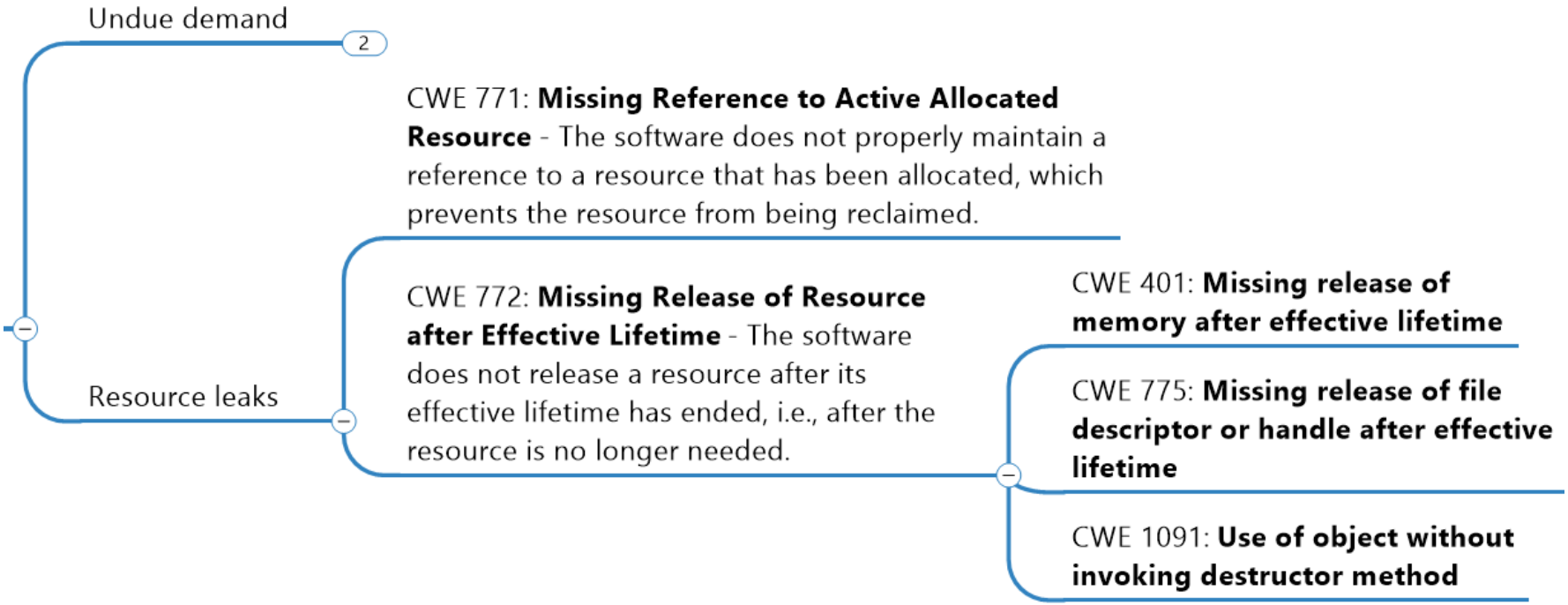
Use CWE to identify possible problems, e.g.:



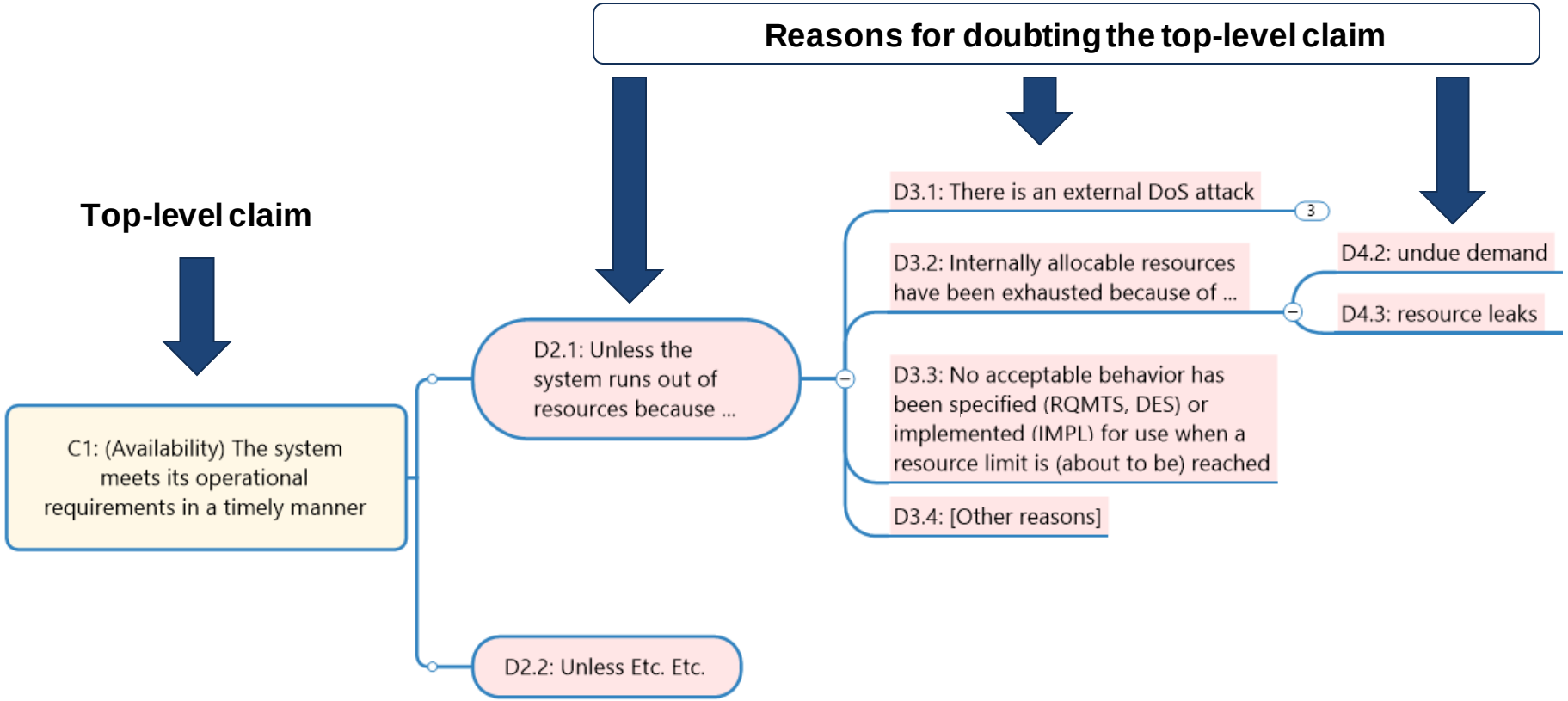
Exploit CWE



Exploit CWE

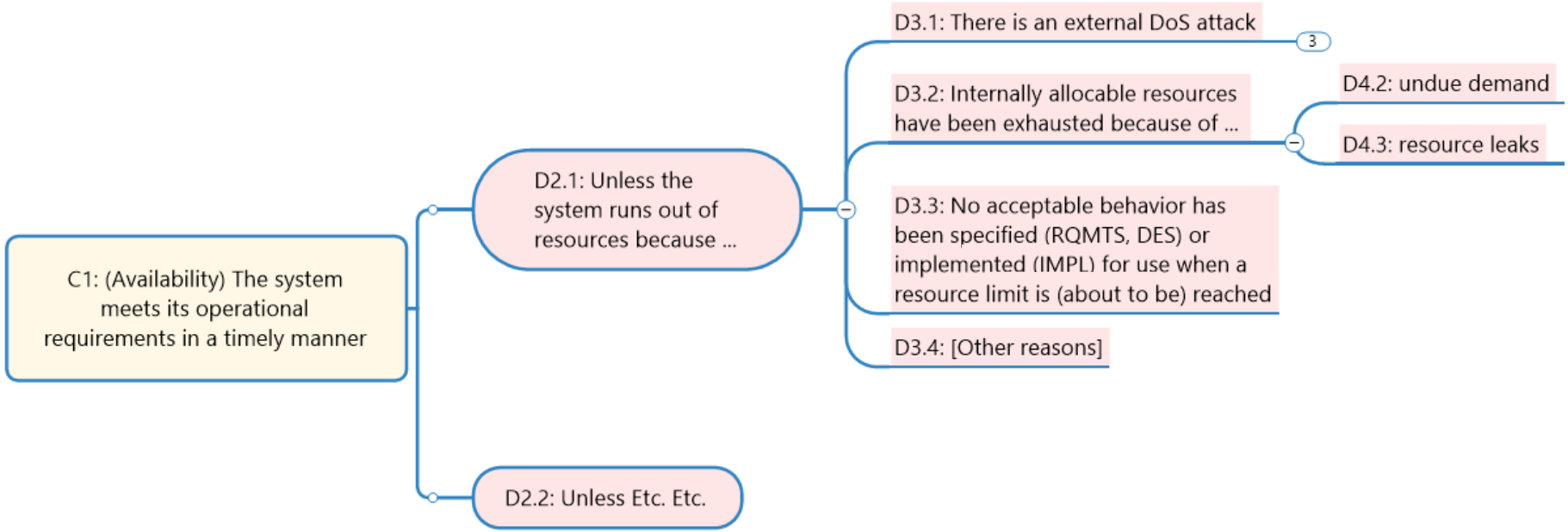


Assurance Case Linked to CWEs

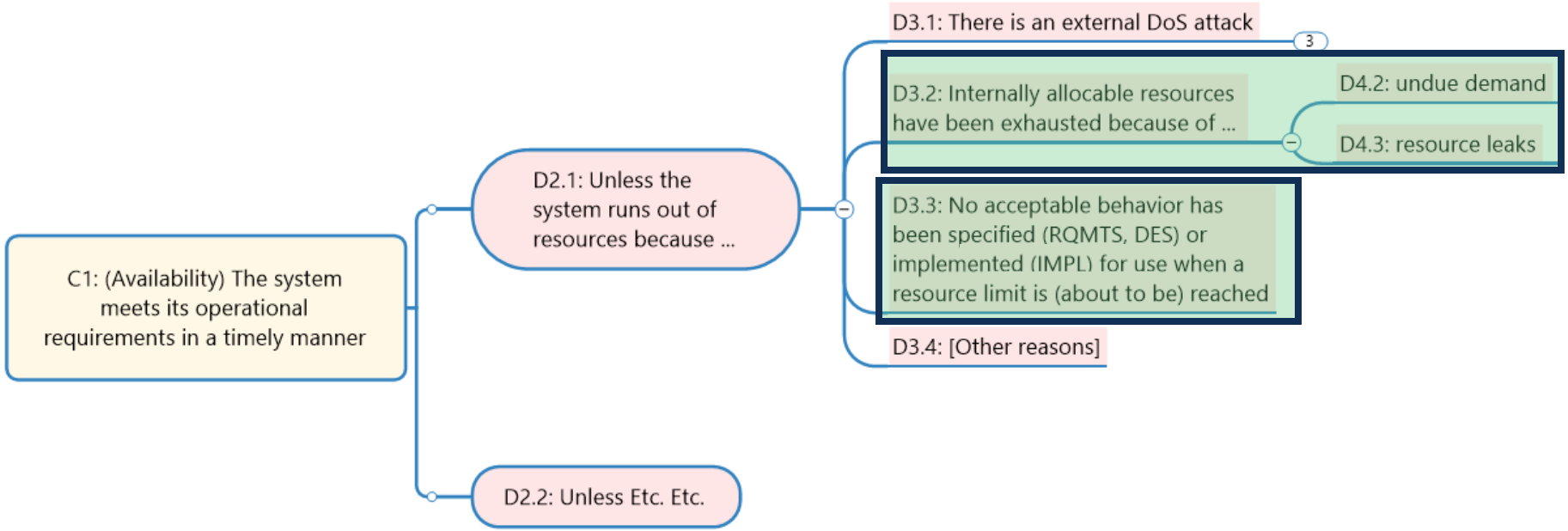


Assurance Case Linked to CWEs

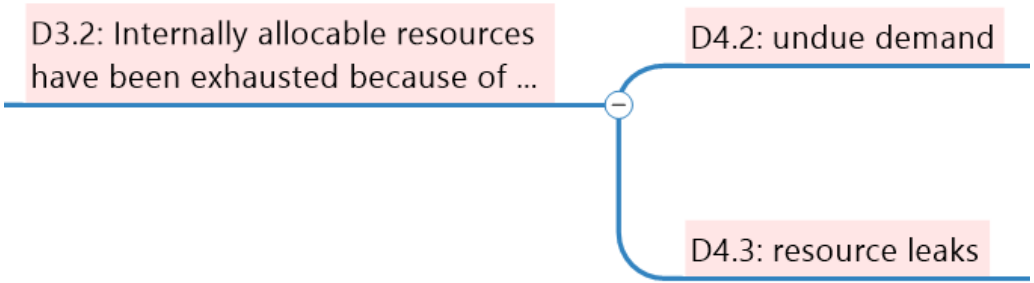
Confidence increases as doubts are reduced



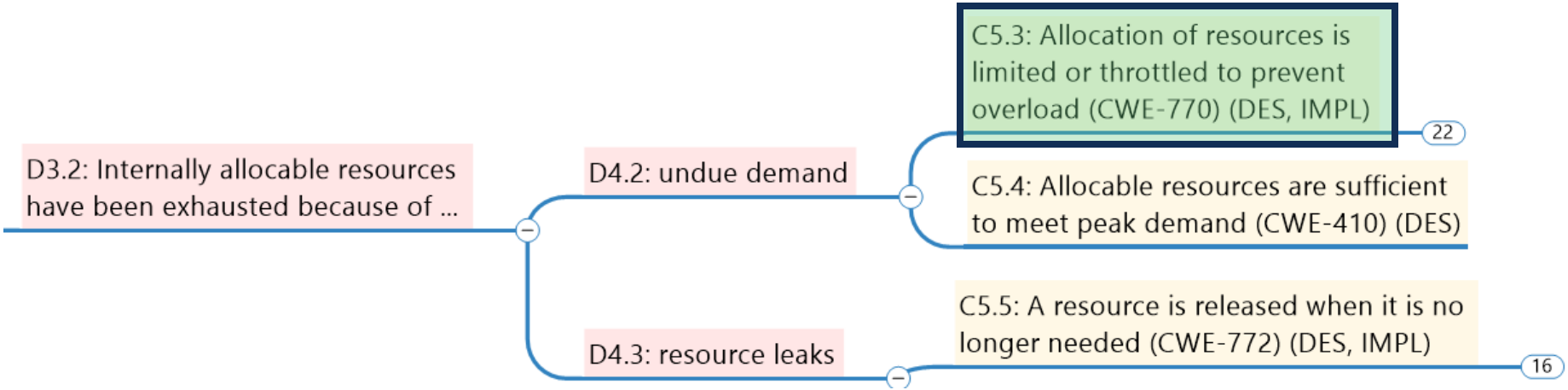
Assurance Case Linked to CWEs



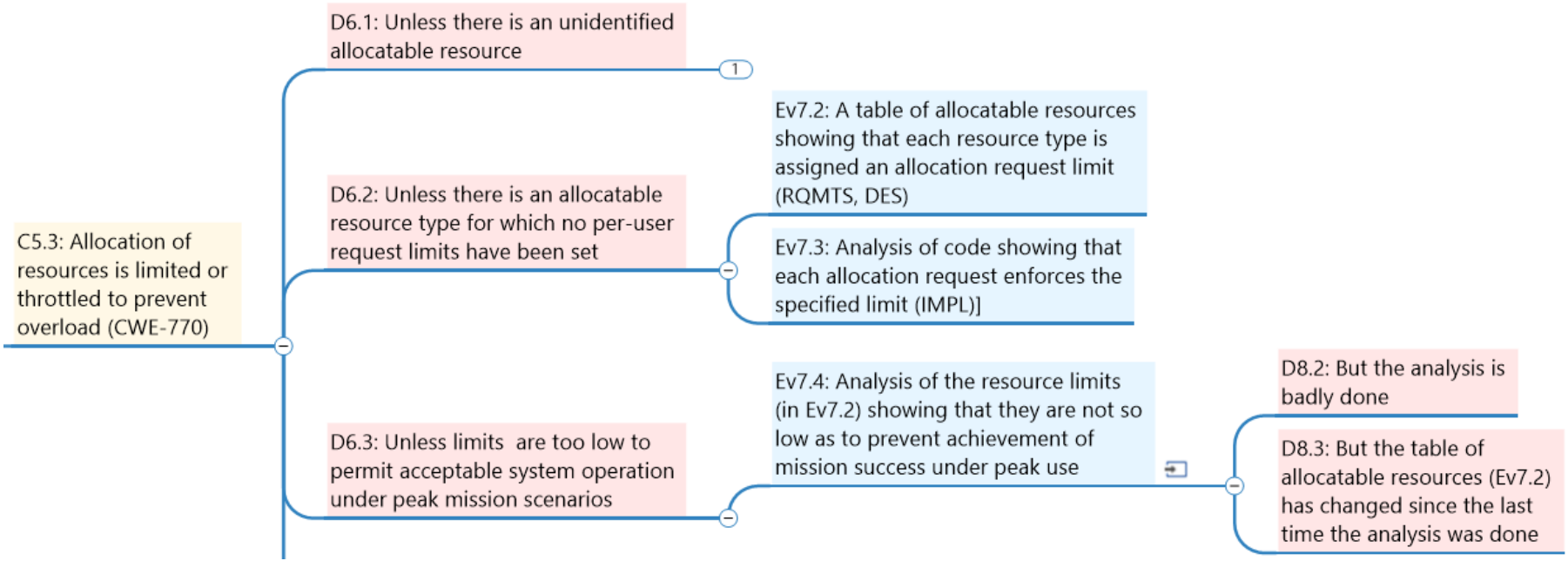
Assurance Case Linked to CWEs



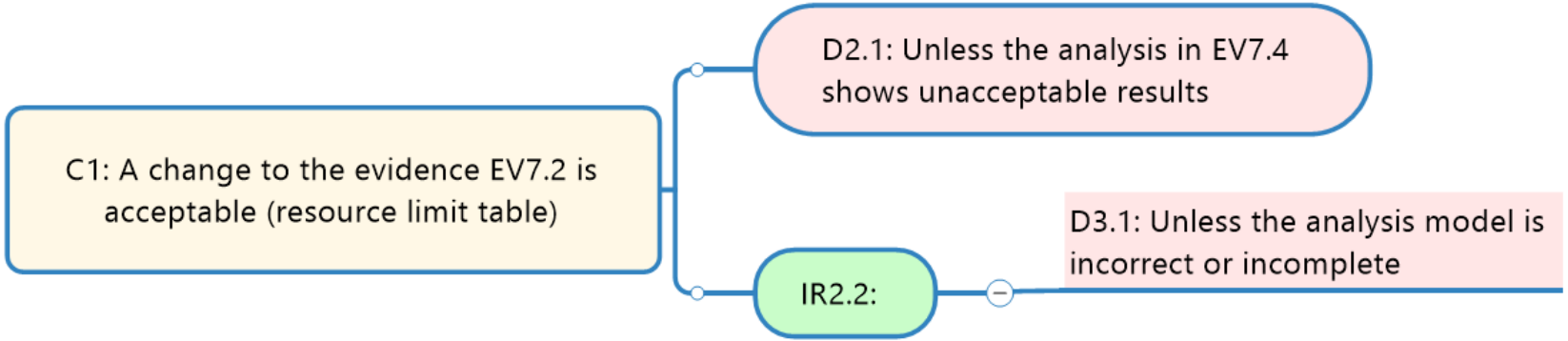
Assurance Case Linked to CWEs



Assurance Case Linked to CWEs



Reassurance Case Example



Summary

We have shown:

- How an AC based on CWEs can suggest the evidence that needs to be gathered to increase confidence in a system's behavior
- How the AC can identify exit criteria for a stage in the DevSecOps pipeline
- What evidence needs to be refreshed to maintain confidence that (relevant) exit criteria continue to be met after a change (the reassurance case)