Defensive programming (1) Adopting good programming practices

Modularity

➡ Have separate modules for separate functionalities

✓ Easier to find security flaws when components are independent

Encapsulation

➡ Limit the interaction between the components

✓ Avoid wrong usage of the components

Information hiding

➡ Hide the implementation

◉ Black box model does not improve security

Defensive programming (2)
Being security aware programmer

- ✓ Check the inputs, even between components that belongs to the same application (mutual suspicion)
- ✓ Be “fault tolerant” by having a consistent policy to handle failure (managing exceptions)
- ✓ Reuse known and widely used code by using design patterns and existing libraries