Designing and Building **Secure Software**

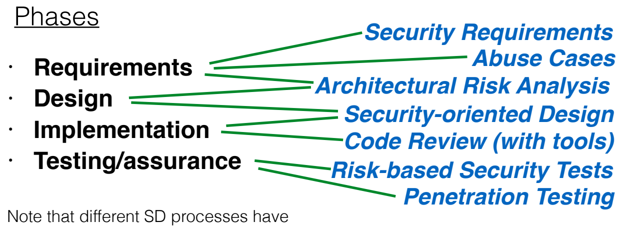
Development process

- Many development processes; **four common phases**:
  - **Requirements**
  - **Design**
  - **Implementation**
  - **Testing/assurance**
  - Phases of development apply to the whole project, its individual components, and its refinements/iterations
- Where does **security engineering** fit in?
  - **All phases!**

Making secure software

- **Flawed approach**: Design and build software, and ***ignore security at first***
  - Add security once the functional requirements are satisfied
- **Better approach**: ***Build security in*** from the start
  - Incorporate security-minded thinking into all phases of the development process

Security engineering



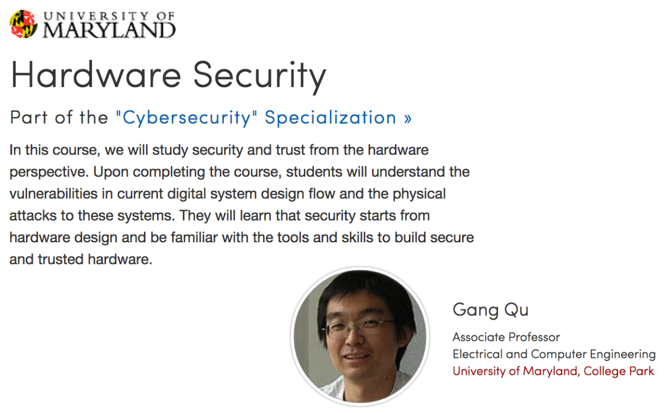
Note that different SD processes have different phases and artifacts, but all involve the basics above. We'll keep it simple and refer to these.

Activities

Note Software vs. Hardware

- **System design contains software *and hardware***
  - *Mostly, we are focusing on the software*
- **Software is malleable** and easily changed
  - Advantageous to core functionality
  - **Harmful to security** (and performance)
- **Hardware is fast**, but hard to change
  - Disadvantageous to evolution
  - **Advantage to security**
    - Can't be exploited easily, or changed by an attack

Learn more!



UNIVERSITY OF MARYLAND

Hardware Security

Part of the "Cybersecurity" Specialization »

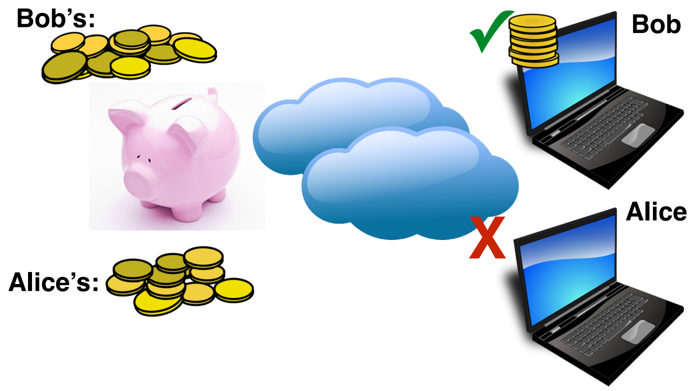
In this course, we will study security and trust from the hardware perspective. Upon completing the course, students will understand the vulnerabilities in current digital system design flow and the physical attacks to these systems. They will learn that security starts from hardware design and be familiar with the tools and skills to build secure and trusted hardware.

Gang Qu
Associate Professor
Electrical and Computer Engineering
University of Maryland, College Park

Secure Hardware

- **Security functionality in hardware**
  - Intel's AES-NI implements **cryptography instructions**
  - Intel SGX supports "**encrypted computation**"
    - For cloud computing applications
- **Hardware primitives for security**
  - **Physically uncloneable functions (PUFs)**
    - Source of unpredictable, but repeatable, randomness, useful for authentication
  - Intel MPX - **primitives for fast memory safety** enforcement

Running Example: **On-line banking**


Bob's:
Bob
✓
X
Alice
Alice's: