Security Issues in Software Design

"All data is evil, until proven otherwise."
- Michael Howard, Microsoft

When most people think about computer security, they think about physical security and cryptography. Physical security involves controlling access to computer assets and protecting them from physical attack, and cryptography is used to ensure that messages transmitted between computers are not intercepted or altered. For example, when you log into WebTycho, your password is encrypted by your browser and decrypted by the WebTycho server.

But errors in the design of software can also create security vulnerabilities. We have all heard about all the "security updates" that Microsoft has issued for its Windows operating systems over the past few years. A vulnerability in a software product can subject the computer on which it is running to various attacks. Attacks may be grouped in the following categories:

- **Spoofing**: An attacker pretends that he is someone else, perhaps in order to inflict some damage on the person or organization impersonated.
- **Tampering**: An attacker is able to modify data or program behavior.
- **Repudiation**: An attacker, who has previously taken some action, is able to deny that he took it.
- **Information Disclosure**: An attacker is able to obtain access to information that he is not allowed to have.
- **Denial of Service**: An attacker prevents the system attacked from providing services to its legitimate users. The victim may become bogged down in fake workload, or even shut down completely.
- **Elevation of Privilege**: An attacker, who has entered the system at a low privilege level (such as a user), acquires higher privileges (such as those of an administrator).

These six categories are encapsulated in the acronym STRIDE.

One of the goals of this class is to enhance your understanding of these ideas. It will be useful to have some common terminology:

- A **vulnerability** is a defect in the design of a software product that makes it possible for an attacker to carry out an attack on a system.
- A **threat** is a possible way to attack a software product
- An **exploit** is a specific technique devised by an attacker to take advantage of a vulnerability. It is a realization of a threat.

Vulnerabilities may be found in operating systems, library subprograms used to build applications, widely used applications such as browsers and e-mail programs, and third-party applications. Because of recent progress in fixing vulnerabilities in operating systems, most current attacks target application programs.

The largest single cause of security vulnerabilities in software is **trust in data**. The program (or component of a program) expects a certain kind of data, assumes that the incoming data is of that certain kind, and fails to examine the data to verify that it is what is expected. The program may then fail if accidentally or intentionally malformed data is received. In short, the program fails to do an adequate job of **input data validation**. In this course, we will have several occasions to consider input data validation and some consequences of absent or inadequate input data validation..

Acknowledgement: This note is based in part on material presented at the Microsoft Academic Days Conference on Trustworthy Computing, April 7-9 2006.

Suggested readings:

19 Deadly Sins of Software Security. Michael Howard, David LeBlanc, John Viega. Microsoft Press.

Threat Modeling. Frank Swiderski, Window Snyder. Microsoft Press.