Secure Software Design

Andey Robins

Spring 23 - Week 2

Foundations of Security

Outline

- What is security?
- The CIA Triad
- The Gold Standard
- The Rest of Security
- Differences
- Example Designs

What is Security About

Security is all about ***trust***

- Who has it?
- Who do we give it to?
- What does it get you?

Some Terms

Information Security: The protection of data

Software Security: The design, implementation, and operation of trustworthy systems

Trust Decision: At some point, trust must be given, and what happens at that point

Trusting Too Little

- Creates excess work
- Requires more upkeep/maintenance
- Drains resources
- More difficult

I have a book which I trust nobody to read without being under my direct supervision. I place this book inside of a safety deposit box at the bank. I place the key to this box inside the safety deposit box at a different bank. This safety deposit box is only accessible after giving the teller a form of ID and a passphrase.

Trusting Too Much

- Can lead to being blindsided later
- Creates a culture of insecurity

The same book from before, but I just leave it sitting out on my desk.

Security is a game of tradeoffs

A Reasonable Middleground

I place the book in a fireproof safe in the bottom drawer of my desk. It uses a keypad for password entry, and only I know the password. My desk locks with a key I keep on my keyring.

Therefore, security is about tradeoffs regarding trust.

Trust is a Spectrum

Implicit Trust

Trustworthiness

The CIA Triad

Confidentiality

Confidentiality: Your secrets should remain secret

Expectations of Confidentiality

- User assumptions
- Misuse
- Legal requirements

Example: Levels of Confidentiality

Imagine that you work for a password utility company. Your company hosts password syncing servers and a password keeper desktop application that goes along with the online service.

1. An employee’s email address is leaked with their identity.
2. –
3. –

1. An employee’s email address is leaked with their identity.
2. Company source code is exfiltrated.
3. –

1. An employee's email address is leaked with their identity.
2. Company source code is exfiltrated.
3. User password vaults are exfiltrated.

All three are compromises of *confidentiality* but, they all clearly have different levels of impact.

Information Leakage

Assume a system doesn't provide any explicit subversion of confidentiality.

```
CREATE TABLE Users (
    uid INT AUTOINCREMENT PRIMARY KEY,
    email TEXT NOT NULL,
    bio TEXT
);
```

If the link to view your profile is website.com/user/<uid>/profile.html, what information does this setup leak?

If the link to view your profile is website.com/user/<uid>/profile.html, what information does this setup leak?

The number of users

An Attack

I run a rival business and I want to determine if I'm converting more users than my competitor. I can write a simple script like:

```
# pseudocode
num_users = 123456 # current count of users
page = curl website.com/user/$num_users/profile.html
if page.error == 404 {
    echo $num_users
} else {
    num_user++
    bash ./competitors.sh
}
```

Integrity

Integrity: Nothing should be changed without your knowledge

Availability

Availability: You can get what you need when you need it

The Gold Standard

Authentication

Authentication: You should know who is interacting with your system

Authorization

Authorization: You should know if the user is allowed to do what they want

Auditability

Auditability: You should be able to see what happened

The Rest

The Hand

- CIA
- Think like an adversary
- Keep it Simple
- Defense in Depth

Think Like an Adversary

- Who attacks us?
- What are they going to do?

These two questions lead us to the concluding question of what are we going to do about it?

Figure 1: Bank Layout

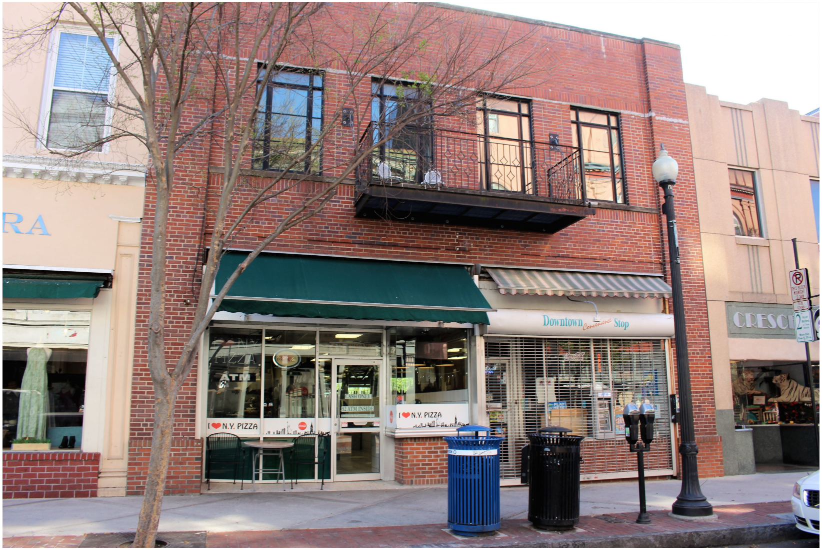

Figure 2: An example storefront

Keep it Simple

The simple design is the one with easily seen problems

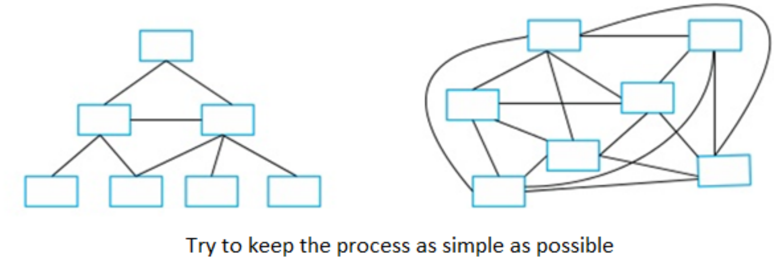

Figure 3: Assume one block requires trust, which is easier to define the boundaries of trust for?

Other Questions

- If a module needs to be replaced, which design is better?
- If two modules need to be combined, which design is better?
- If a module is failing, which is easier to debug?
- If we need to ship a new feature, which is easier to graft it onto?

Defense in Depth

A good defense will have multiple layers

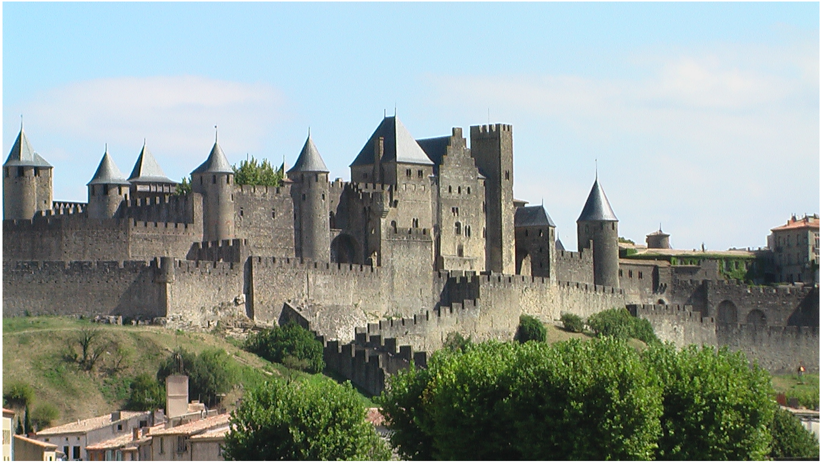
Figure 4: The layered walls of Carcassonne

Differences

Integrity vs Auditability

Confidentiality vs Authentication

Authentication vs Authorization

Tradeoffs

Confidentiality vs Availability

Authentication vs Anonymity

Integrity vs Availability

Example Designs