Summary of the Summary

We’ve covered a lot

Let’s put this in some context

This is not chronological, nor exhaustive

But we’ll begin at the beginning...
Basic Concepts

Security Properties

- Confidentiality
- Integrity
- Authenticity
- Trust

System Participants

- Principals
- Identities
- Subjects
Security Properties

Why don’t we have them?

Money (and time)

Mistakes are easy to make
- Buffer overflows
- SQL commands by string concatenation

Security is a secondary task
⇒ It’s also frequently confusing for users

Users become habituated to frequent or overly technical warnings

Security failures are externalities for the organizations that can prevent them
Attackers

Why do we have attackers?

- Boredom
- Technical Challenge
- Ideology
- Money
- Spite/Revenge
- Personal Gain (eg, changing grades)
Attacks

What kinds of attacks do we see?

- Programming Errors
- Social Engineering
- Protocol Vulnerabilities

Not all attacks fit cleanly into one category.
Programming Errors

Some languages make errors easy to make

- Buffer overflows in C
- SQL injection in PHP

Some languages make errors hard to avoid

- XSS, CSRF, RFI in PHP or Javascript
- Fingerprinting in Javascript or HTML
Social Engineering

Turns otherwise-trustworthy insiders into adversaries
⇒ Including against themselves!

- Phishing
- Malware
- Fraudulent phone calls
- Spam
Protocol Vulnerabilities

Flooding

Brute-Force Flooding

- Pings
- SYN floods

More Sophisticated Flooding

- Botnet-based DDoS (strength in numbers)
- Smurfs (broadcast pings from spoofed source)
- Shrews (low-volume attack against TCP)
Protocol Vulnerabilities
Low-Volume Attacks

Implementation Flaws

- Ping of Death
- LAND

Man-in-the-Middle

- Kaminsky’s DNS attack
Good Security Practices

- Operational security
- Updating software
- Configuration management
Defense
System Designers

Gold Standard

- Authentication
- Authorization
- Audit

Cryptography

Digital Certificates

Anonymity and Privacy

Fault Tolerance (eg, Replication)
Defense
Network Engineers

Firewalls

Deep-Packet Inspection

Application Session Inspection (eg, Spam filtering)

Virtual Private Networks
Defense
Programmers (that's you)

WRITE BETTER CODE!

Know API details before you start coding

Make security part of design at the beginning, not the end

Understand the risk/benefit trade-offs for

- Choice of language (Do I want to use C? PHP? Java? Python?)

- Software libraries (Does OpenSSL make me sad?)

- Protocol/implementation decisions (Do I need AES256? Do I even need encryption?)

Now go forth, and create secure systems!