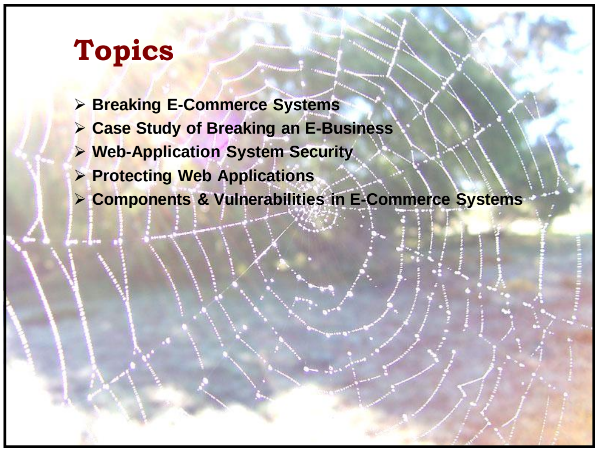
Web-Based Vulnerabilities

CSH6 Chapter 21
“Web-Based Vulnerabilities”
Anup K. Ghosh, Kurt Baumgarten, Jennifer Hadley & Steven Lovaas



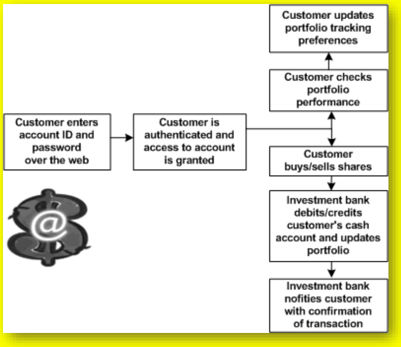
Topics

- Breaking E-Commerce Systems
- Case Study of Breaking an E-Business
- Web-Application System Security
- Protecting Web Applications
- Components & Vulnerabilities in E-Commerce Systems

Breaking E-Commerce Systems

- Thinking about how criminal hackers think
  - ❑Attack weakest link
  - ❑Look for monetary gain
  - ❑Low-hanging fruit
  - ❑Attack servers when possible
- Must harden not only perimeter but also core
- Asymmetric attacks
  - ❑Defense harder & more costly than offense
  - ❑Script kiddies have caused $M damage
    - ✓E.g., MafiaBoy 2000 vs eBay, Amazon, Schwab….

Case Study of Breaking an E-Business (1)



Customer enters account ID and password over the web

Customer is authenticated and access to account is granted

Customer updates portfolio tracking preferences

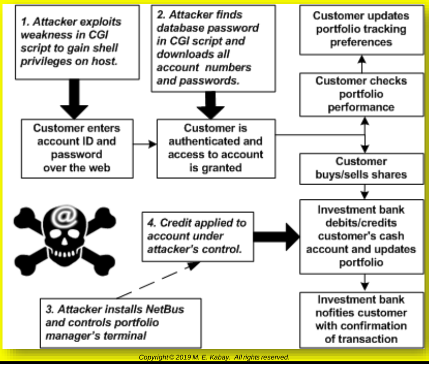
Customer checks portfolio performance

Customer buys/sells shares

Investment bank debits/credits customer's cash account and updates portfolio

Investment bank nofities customer with confirmation of transaction

Case Study of Breaking an E-Business (2)



1. Attacker exploits weakness in CGI script to gain shell privileges on host.

2. Attacker finds database password in CGI script and downloads all account numbers and passwords.

Customer enters account ID and password over the web

Customer is authenticated and access to account is granted

Customer updates portfolio tracking preferences

Customer checks portfolio performance

Customer buys/sells shares

4. Credit applied to account under attacker's control.

Investment bank debits/credits customer's cash account and updates portfolio

3. Attacker installs NetBus and controls portfolio manager's terminal

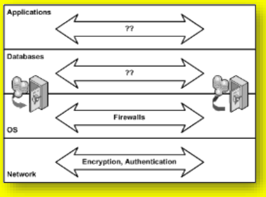
Investment bank nofities customer with confirmation of transaction

Web-Application System Security

- Absolutely require corporate security policy
  - ❑Informs decisions on specific security configurations
  - ❑Inconsistencies can doom security
- Security systems should be independently evaluated
  - ❑System audits (do measures conform with policy?)
  - ❑Vulnerability analysis (can we locate obvious gaps in security?)
  - ❑Penetration testing (can we break through the barriers using criminal hacker methods?)

Protecting Web Applications

- Layered view of systems
- Network, OS flaws usually documented
  - Alerts
  - National Vulnerability Database http://nvd.nist.gov/
- Vulnerability scanners available (see *CSH6* Ch 46)
- Firewalls critical element



- Application servers (Java etc) must be secured
- Application security = function of how programs are configured & used (not just of patches)

Components & Vulnerabilities in E-Commerce Systems

- Client-Side Risks
- Network-Protocol Risks
- Business-Application Logic
- CGI-Script Vulnerabilities
- Application Subversion
- Web-Server Exploits
- Database Security
- Platform Security

Client-Side Risks

- Most e-commerce uses browsers
  - Also extending to hand-held devices
- Threats from malicious mobile code (*CSH6* Ch 16 & 17); e.g., Web scripts, Java applets, ActiveX controls, Trojan horse programs
- Serious risk from loss of privacy
  - Identity theft against *data subjects*
  - Business & legal consequences for corporate victims
  - Browsers typically convey much private info
  - Spyware tracks computer usage

Network-Protocol Risks

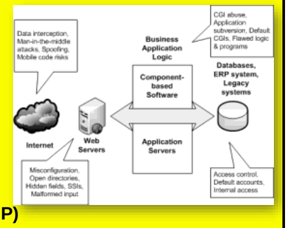
- Primarily result from sending unencrypted data over the 'Net
- Several protocols preserve confidentiality by using encryption
  - SET (Secure Electronic Transaction)
  - SSL (Secure Sockets Layer)
  - S/HTTP (Secure HTTP)(superseded)
  - S/MIME (Secure Multipurpose Internet Mail Extensions)
  - CyberCash (proprietary credit-card system)(bankrupt 2001, bought by VeriSign & First Data Merchant Services Corp.)
- See *CSH6* Ch 30

Network-Protocol Attacks

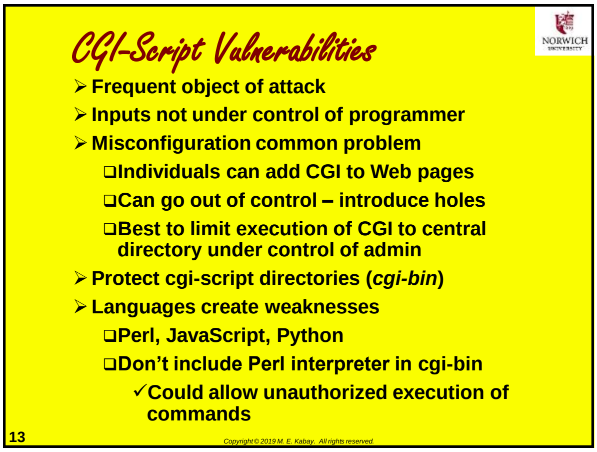
- Man-in-the-middle (intercepting, inserting)
- DNS attacks (altering tables to misdirect users)
- War dialing (scanning all phone numbers in block for modems) – equivalent today is scanning for unsecured Wireless Access Points
- Exploiting software holes (FTP, Bind, SMTP, HTTP)
- Internal access (unauthorized behavior by authorized personnel)
- Leveraging trusted hosts (attack from linked system)
- Brute-force decryption (test all possible keys)

Business-Application Logic

- Key area of vulnerability
  - Usually custom SW
  - Complex
  - May not be tested as thoroughly as COTS
- Critical elements include
  - Common Gateway Interface (CGI)

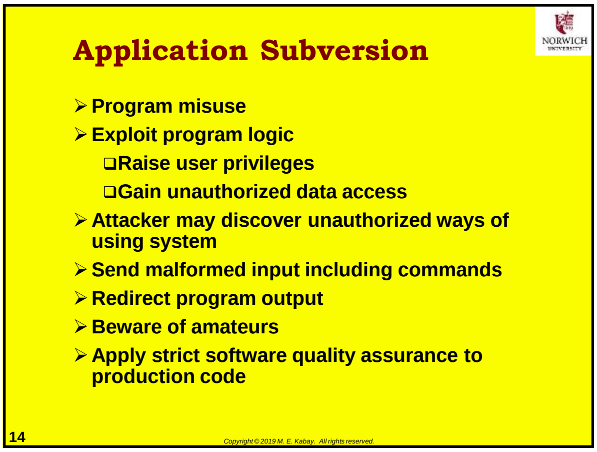

Data interception, Man-in-the-middle attacks, Spoofing, Mobile code risks
Business Application Logic
CGI abuse, Application subversion, Default CGIs, Flawed logic & programs
Component-based Software
Databases, ERP system, Legacy systems
Internet
Web Servers
Application Servers
Misconfiguration, Open directories, Hidden fields, SSIs, Malformed input
Access control, Default accounts, Internal access

  - Hypertext Processor (PHP)
  - Component-based software (CBS)
    - ✓Enterprise JavaBeans (EJB)
    - ✓Java 2 Enterprise Edition (J2EE)
    - ✓Common Object Request Broker Architecture (CORBA)
    - ✓Common Object Model (COM & DCOM)



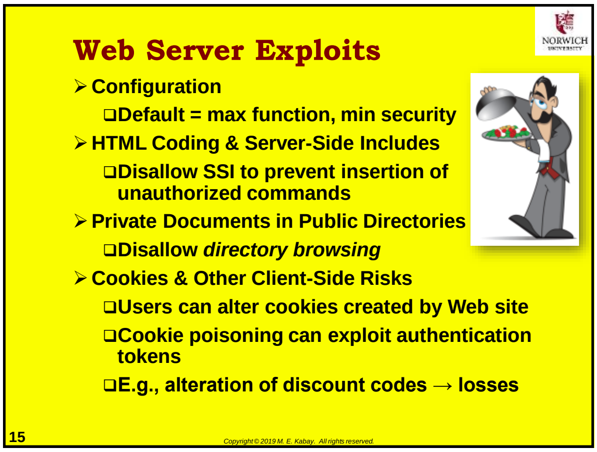
CGI-Script Vulnerabilities

- Frequent object of attack
- Inputs not under control of programmer
- Misconfiguration common problem
  - ❑Individuals can add CGI to Web pages
  - ❑Can go out of control – introduce holes
  - ❑Best to limit execution of CGI to central directory under control of admin
- Protect cgi-script directories (*cgi-bin*)
- Languages create weaknesses
  - ❑Perl, JavaScript, Python
  - ❑Don't include Perl interpreter in cgi-bin
    - ✓Could allow unauthorized execution of commands



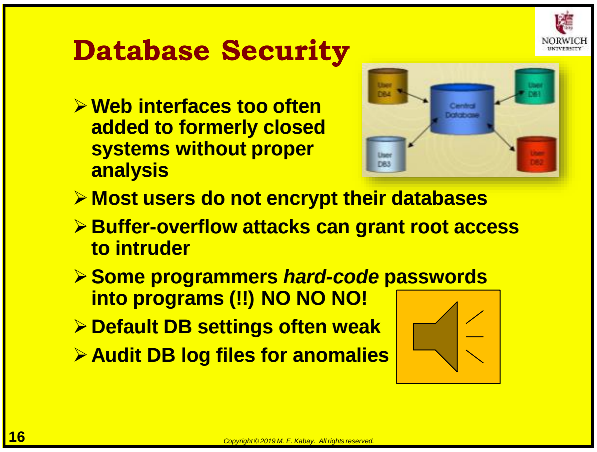
Application Subversion

- Program misuse
- Exploit program logic
  - ❑Raise user privileges
  - ❑Gain unauthorized data access
- Attacker may discover unauthorized ways of using system
- Send malformed input including commands
- Redirect program output
- Beware of amateurs
- Apply strict software quality assurance to production code



Web Server Exploits

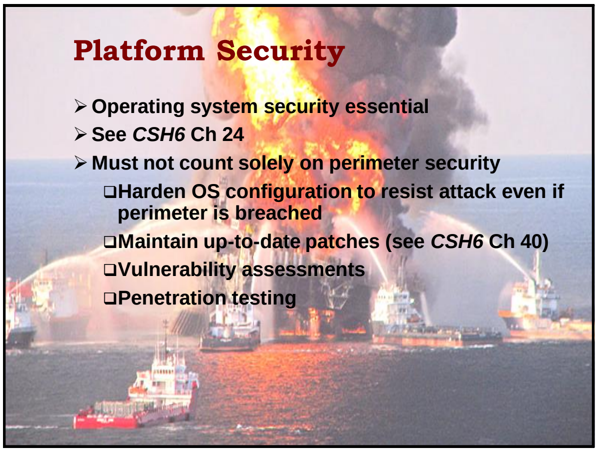
- Configuration
  - ❑Default = max function, min security
- HTML Coding & Server-Side Includes
  - ❑Disallow SSI to prevent insertion of unauthorized commands
- Private Documents in Public Directories
  - ❑Disallow *directory browsing*
- Cookies & Other Client-Side Risks
  - ❑Users can alter cookies created by Web site
  - ❑Cookie poisoning can exploit authentication tokens
  - ❑E.g., alteration of discount codes → losses



Database Security

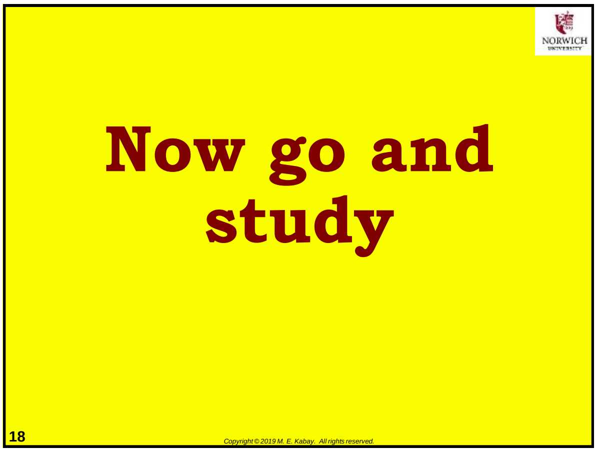
- Web interfaces too often added to formerly closed systems without proper analysis
- Most users do not encrypt their databases
- Buffer-overflow attacks can grant root access to intruder
- Some programmers *hard-code* passwords into programs (!!) NO NO NO!
- Default DB settings often weak
- Audit DB log files for anomalies

User DB4
Central Database
User DB1
User DB3
User DB2



Platform Security

- Operating system security essential
- See *CSH6* Ch 24
- Must not count solely on perimeter security
  - ❑Harden OS configuration to resist attack even if perimeter is breached
  - ❑Maintain up-to-date patches (see *CSH6* Ch 40)
  - ❑Vulnerability assessments
  - ❑Penetration testing



Now go and study