CSE 543 - Computer Security (Fall 2004)

Lecture 21 - Web Security
November 18, 2004
URL: http://www.cse.psu.edu/~cg543/
Another bedtime story ...
What is the web?

• A collection of application-layer services used to distribute content
  – Web content (HTML)
  – Multimedia
  – Email
  – Instant messaging

• Many applications
  – News outlets, entertainment, education, research and technology, …
  – Commercial, consumer and B2B
Web security: the high bits

• The largest distributed system in existence
  – threats are as diverse as applications and users
  – But need to be thought out carefully …

• The stakeholders are …
  – Consumers (users, businesses, *agents*, …)
  – Providers (web-servers, IM services, …)

• Another way of seeing web security is
  – Securing the web *infrastructure* such that the *integrity*,
    *confidentiality*, and *availability* of content and user
    information is maintained
Secure socket LAYER

• Used to authenticate servers
  – Uses certificates, “root” CAs
• Can authenticate clients
• Inclusive security protocol
• Security at the socket layer
  – Transport Layer Security

Diagram:

HTTP

SSL

TCP

IP
SSL Operation

- Phase 1: the SSL Handshake
  - Establishes algorithms used throughout
  - Authenticates parties
  - Establishes master secret
    - Used to create other secrets
      1. Encryption Key (client-server)
      2. Encryption Key (server-client)
      3. Authentication Key (client-server)
      4. Authentication Key (server-client)
SSL Handshake

(1) Client Hello (algorithms, …)

(2) Server Hello (alg. selection, …)

(3) Server Certificate

(4) ClientKeyRequest

(5) ChangeCipherSuite

(6) ChangeCipherSuite

(7) Finished

(8) Finished
Advantages of SSL

• Confidential session
• Server authentication*
• GUI clues for users
• Built into every browser
• Easy to configure on the server
• Protocol has been analyzed like crazy
• Seems like you are getting security “for free”
Disadvantages of SSL

• Users don’t check certificates
  – most don’t know what they mean
• Too easy to obtain certificates
• Too many roots in the browsers
• Default settings are terrible
  – ssl v2 is on
  – totally insecure cipher suites are included
• very little use of client-side certificates
• performance! sites turning off
  – getting better (crypto coprocessors, etc.)
Reality of SSL

• SSL is here to stay no matter what
• credit card over SSL connection is probably safer than credit card to waiter
• biggest hurdles:
  – performance
  – user education (check those certificates)
  – too many trusted sites (edit your browser prefs)
  – enabled version 2 (disable it on the server too)
  – misconfiguration (turn off bad ciphersuites)
Cookies

- Cookies were designed to offload server state to browsers
  - Not initially part of web tools (Netscape)
  - Allows users to have cohesive experience
  - E.g., flow from page to page,
- Someone made a design choice
  - Use cookies to authenticate and authorize users
  - E.g. Amazon.com shopping cart, WSJ.com
Cookie Issues ...

- New design choice means
  - Cookies must be protected
    - Against forgery (integrity)
    - Against disclosure (confidentiality)
- Cookies not robust against web designer mistakes
  - Were never intended to be
  - Need the same scrutiny as any other tech.

Many security problems arise out of a technology built for one thing incorrectly applied to something else.
Cookie Design 1: mygorilla.com

- Requirement: authenticate users on site

mygorilla.com

- Design:
  1. use digest authentication to login user
  2. set cookie containing hashed username
  3. check cookie for hashed username

- Q: Is there anything wrong with this design?
Cookie Design 2: mygorilla.com

- Requirement: authenticate users on site

mygorilla.com

- Design:
  1. use digest authentication to login user
  2. set cookie containing encrypted username
  3. check cookie for encrypted username

- Q: Is there anything wrong with this design?
Library Attack ....

- I am sitting in the local library using the computer ...
- ... to buy some stuff ...
- ... and walk away ...
Dynamic Content

• Server generates content at run time
  – For time-sensitive information (stock ticker)
  – For user customization (Amazon.com)
  – Provide HTML interface to complex system (e.g., course management system)
Dynamic Content: CGI

• Common Gateway Interface (CGI)
  – Generic way to call out to external applications on the server
  – Passes URL to external program (e.g., form)
  – Result is captured and return to requestor

• Historically
  – “shell” scripts used to generate content
    • Very, very dangerous
  – NOTE: “perl” is no better (in some ways worse)
DC: Embedded Scripting

• Program placed directly in content, run at during request time and output returned in content
  – MS active server pages (ASP)
  – PHP
  – mod_perl
  – server-side JavaScript

• Nice at generating output
  – Dangerous if tied to user input
Warning: Cross-Site Scripting

- Note Assume the following is posted to a message board on your favorite website:

  Hello message board.
  <SCRIPT>malicious code</SCRIPT>
  This is the end of my message.

- Now a reasonable ASP (or some other dynamic content generator) uses the input to create a webpage (e.g., blogger nonsense).

- Now a malicious script is now running
  – Applet, ActiveX control, …
Dynamic Content Security

- Largely just applications
  - Inasmuch as applications are secure
  - Command shells, interpreters, are dangerous

- Three things to prevent DC vulnerabilities
  - Validate input
    - Input often received as part of user supplied data
    - E.g., cookie
  - Limit program functionality
    - Don’t leave open ended-functionality
  - Execute with limited privileges