Lecture notes for “web application security”

Mix of emerging technology, privacy, & databases

Web applications
- Def: client-server programs communicating over HTTP, client UI rendered by web browser
- Mashup def: single web page displaying multiple web applications
  - Often written & hosted by different developers
- Examples
  - Mashups: Facebook, iGoogle, MySpace (we'll come back to these)
  - Shopping carts: Amazon
  - At GT: Buzzport, Oscar, T-Square, Library, Course Critique

Architecture
- [Draw picture: client browser, server web, server DB]
- [Draw communication: browser to web server: strings ; web server to DB server: SQL strings]
- Client: Javascript, AJAX (asynchronous javascript and xml)
- Web server: PHP, ASP, et al
- DB server: MySql, MS Sql Server, Oracle

SQL requests
- DB made of tables
- Tables made of columns & rows
- Each row is an “entry” of the DB
- Read/write DB via SQL (structured query language)
- SQL commands are strings interpreted by DB engine
- Keywords: “SELECT, WHERE, INSERT, DROP”, etc
- Data: column names, values to match
- Web server code (e.g. php) often assembles SQL command via concatenation

Example:
- [Suppose DB of students in CS 4235, has name, GTID, grade]
- [You want to access your grade via web page with text entry box taking GTID number]
- [Text entry box sets php variable $idnumber]
- Server code assembles:
  - SELECT name, grade FROM cs4235 WHERE gtid=$idnumber

- Intent: return one row of table matching student ID number

Attack:
- No distinction between code and data in sql queries
- Use string concatenation to add additional SQL code
- Suppose $idnumber = -1 OR 1=1
- Server code assembles:
  - SELECT name, grade FROM cs4235 WHERE gtid=-1 OR 1=1
- Effect: returns all rows of table, see everyone's grades

Complexities:
- String values, quote marks [show example]
- Web page may only show first result
  - Exclude values known already
- Complex queries using JOIN, HAVING, etc
- Structure of query may not be known
- Column names may not be known
- Table names may not be known
- Some sql engines have commands to reveal internal information

More SQL attacks:
- DROP TABLE
- INSERT

Other web application vulnerabilities
- Command injection [write example]
- Cross-site scripting [http://library.gatech.edu/ search box demo]
  - <script>document.location.href=”http://www.uga.edu”</script>
- Web api
  - [Explain how invoked]
  - [May contain functionality designers did not want to expose]

Facebook mashup architecture
- Third party apps hosted at third party servers
  - [Draw picture]
- Servers get access to your personal information
- Server can take any action with your information
  - Outside of facebook's control