CS356: More Web Attacks

W. Michael Petullo

University of Wisconsin–La Crosse

As of November 10, 2021

Does TLS/HTTPS solve everything?

Web Security: Cookies

Small bits of information that a browser will store on behalf of a website: stateful information for stateless HTTP protocol (e.g., login info. and preferred language).

- Cookie size limited to around 4 KB
- Browser will provide cookie only to site that created cookie

Dangers:

- Cross-site scripting can gather cookies
- Cookies transmitted without TLS can be intercepted
- Pages can opt to include scripts from tracking sites

```
HTTP/2.0 200 OK
Content-Type: text/html
Set-Cookie: name=value
...
```

Response

```
GET /index.html HTTP/2.0
Host: example.com
Cookie: name=value
...
```

Request

Web Security: Cross-Site Request Forgery

- A cross-site scripting attack exploits the trust a user/browser has in a site.
- A cross-site request forgery exploits the trust a site has in a browser.

Goal: cause the victim to submit maliciously-crafted request to a site the victim has privileged access to.

1. Attacker identifies reproducible request that executes a desirable action (e.g., change password, make purchase, or post comment)
2. Victim authenticates to target site (browser now holds authentication cookie)
3. Attacker hosts JavaScript at attacker.com; JavaScript sends request from victim's browser to target site.
4. Victim visits attacker.com; browser executes script, thereby sending request to target site

Protection: tokens that render requests irreproducible; SameSite=Strict cookie attr.

More Web Attacks: CSRF Exercise

1. Compile `xssd.go` using `go build xssd.go`; run `xssd`.
2. Compile `bankd.go` using `go build bankd.go`; run `bankd`.
3. Run Wireshark, and watch the loopback interface.
4. Use a browser to connect to `localhost:8082`.
5. Interact with the banking website; make a transfer.
6. Visit the forum by visiting `localhost:8080`.
7. Enter the bank's fraud-alert HTML as a forum post.
8. Click on the button that appears in the forum.
9. What did you see in Wireshark?

Software Vulnerabilities: Command Injection

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>

5  int main(void)
6  {
7      const char *prog = "/bin/ping -c 1";
8      int len = strlen(prog) + 1;
9      char c[BUFSIZ];

11     strcpy(c, prog);
12     strcat(c, " ");
13     fgets(c + len, BUFSIZ - len, stdin);

15     system(c);
16 }
```

- `system` takes command *c* and executes it through a shell
- Attacker controls argument (?) that follows "ping -c 1"
- Shell will interpret shell syntax

More Web Attacks: Command-Injection Exercise

1. Compile `cmdinjectd.go` using `go build cmdinjectd.go`; run `cmdinjectd`.
2. Use a browser to connect to `localhost:8083`.
3. Exploit server to run a command of your choosing.

Databases: SQL

create	CREATE TABLE students (last string, first string);
insert	INSERT INTO students (last, first) VALUES ('Washington', 'George');
insert	INSERT INTO students (last, first) VALUES ('Lincoln', 'Abraham');
select	SELECT * FROM students; SELECT last FROM students; SELECT first, last FROM students;
delete	DELETE FROM students WHERE last == 'Lincoln';
drop	DROP TABLE students;
comment	-- This is a comment!

Databases: SQL Injection

```
1  in := bufio.NewScanner(os.Stdin)
2  if !in.Scan() {
3      fmt.Fprintf(os.Stderr, "error reading statement\n")
4      os.Exit(1)
5  }

7  rows, err := db.Query("SELECT "+in.Text()+" FROM public;")
8  if err != nil {
9      fmt.Fprintf(os.Stderr, ...)
10     os.Exit(1)
11 }
```

- `Query` takes query q and executes it through SQL engine
- Attacker controls argument that follows "SELECT"
- SQL engine will interpret SQL syntax

More Web Attacks: SQL-Injection Exercise

1. Compile `sqlinjectd.go` using:
   1. `go mod init sqlinjectd`.
   2. `go mod tidy`.
   3. `go build sqlinjectd.go`.
2. Run `sqlinjectd`.
3. Use a browser to connect to `localhost:8084`.
4. Exploit server to reveal secret value in database.

More Web Attacks: Assignments

Graded Homework Aquinas: xss

https://www.flyn.org/courses/cs356/schedule