Building a Dumb Web Server

And Why That Can Be a Smart Thing to Do

Alan Dewar
President, Calgary UNIX Users' Group
http://www.cuug.ab.ca/dewara
dewara@cuug.ab.ca

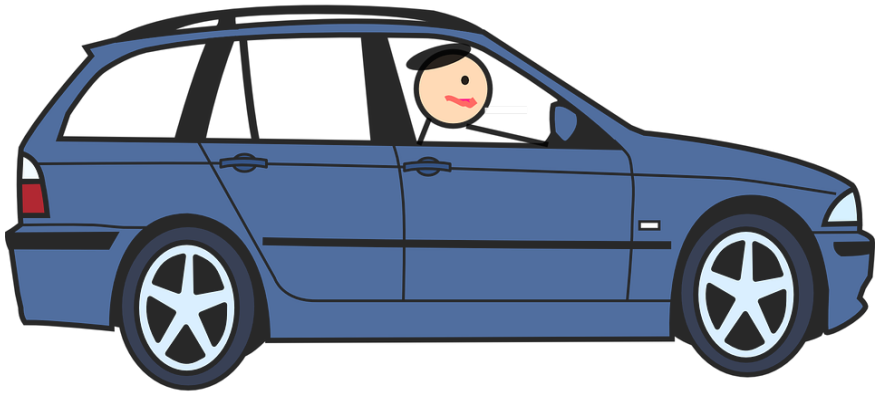
Building a Dumb Web Server

- Need to present information
- Desire to do much more
- Sophistication increases risk
- What do you really need?
  - How to get that
  - What you still need to watch out for

The Situation

- You are part of a group
  - Small business
  - Volunteer organization
  - Personal interest
- You have information
  - Documents
  - Photos
  - Videos
  - Contacts
- You want to make it available
  - Web server

The Dream



"My toaster is on the Internet, so I can have hot bagels ready when I get home!"

The Problem

"His toaster is on the Internet, so I can burn his house down before he gets home!"

The Solution?

The Solution?

The Problem with the Solution

"Kitchen Sink 1.0 includes Faucet 0.9, which uses Washer 0.3.1, which has a known leak I can exploit…"

Keep Patches Up to Date!

- Equifax
  - Apache Struts vulnerability: CVE-2017-5638
  - Exposed full names, social security numbers, birth dates, addresses, driver license numbers
  - 143 million US people affected (44 percent of population)
- CUUG?

The Alternative

- Keep it simple!
  - Static web pages
  - Client-side scripting

Starting from Scratch

- HTTP
- Simple implementation
- Complications

Uniform Resource Locator (URL)

`http://www.cuug.ab.ca:80/upcoming/meeting.html?id=42&x=foo#hi`

- Protocol: `http`
- Host: `www.cuug.ab.ca`
- Port: `80`
- Path: `/upcoming/meeting.html`
- Search: `id=42&x=foo`
- Position: `hi`

Browser/Server Conversation
Hypertext Transfer Protocol (HTTP)

- Request

  `GET` *path* `HTTP/1.1`

  other stuff

  blank line

- Response

  `HTTP/1.1` *status_code message*

  other stuff

  blank line

  content of web page

Browser/Server Conversation Example: http://www.yoyodyne.com/

- Browser:

```
GET / HTTP/1.1
Host: www.yoyodyne.com
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:45.0) Gecko/20100101 Firefox/45.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
```

- Server:

```
HTTP/1.1 200 OK
Date: Sun, 24 Sep 2017 02:46:17 GMT
Server: Apache/2.4.27 (FreeBSD)
Last-Modified: Tue, 05 Sep 2017 13:49:53 GMT
ETag: "73a-55871807646e0"
Accept-Ranges: bytes
Content-Length: 1850
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html
```

Web Server Pseudo-Code

- Open listen socket
- Upon connection:
  - Read up to blank line
  - Extract path from “GET” line
  - Find specified file
  - Write HTTP status line, blank line
  - Copy file to socket
  - Close connection

Web Server Actual Code (Tcl)

```
proc serve {sock} {
    if {[catch {
        set request [read $sock]
        regexp {^GET ([^\n]*) HTTP/} $request dummy path
        set fd [open "./$path" r]
        fconfigure $fd -translation binary
        set contents [read $fd]
        close $fd
        puts $sock "HTTP/1.0 200 OK\r\n\r"
        puts -nonewline $sock $contents
    }]} {
        puts $sock "HTTP/1.0 404 Not Found\r\n\r"
        puts $sock "<p>Sorry, not found.</p>"
    }
    close $sock
}
proc connect {sock ip port} {
    fconfigure $sock -translation binary -blocking 0
    fileevent $sock readable "serve $sock"
}
socket -server connect 8080
vwait forever
```

Directories

- Path ending with trailing “/”
  - `GET / HTTP/1.1`
  - Append “`index.html`”
- Directory but no trailing “/”
  - `GET /dewara HTTP/1.1`
  - `HTTP/1.1 301 Moved Permanently`
  - `Location: http://www.cuug.ab.ca/dewara/`

Digression: HTTP Status Codes

- 1xx: Informational
- 2xx: Successful
  - `200 OK`
- 3xx: Redirection
  - `301 Moved Permanently`
- 4xx: Client Error
  - `404 Not Found`
  - `418 I'm a teapot`
- 5xx: Server Error

Giving Away Too Much

- Malicious requests
  - `GET /../../../../../../../etc/passwd HTTP/1.1`
  - `GET /../../../../../../../dev/sda HTTP/1.1`
  - `GET /../../../../../../../proc/12345/fd/1 HTTP/1.1`
- Sanitize requests
- Run as dedicated user with minimal privileges

Spaces and Other Special Characters

- Hexadecimal escape codes
  - `GET /foo/bar/hello%20world.html`
- Decode *before* sanitizing
  - `GET /%2E%2E/%2E%2E/%2E%2E/%2E%2E/etc/passwd HTTP/1.1`

Long Headers

- Attempted buffer overrun
  - `GET` */(1_million_characters)(executable code)*
- Reject long paths
  - `HTTP/1.1 414 URI Too Long`

Denial of Service

- Client send partial request, then hangs
- Enforce timeout
  - `HTTP/1.1 408 Request Timeout`

Dumbing It Down

- It's your web site, so you have control over content
  - No links to directories
  - No spaces in paths
  - No excessively-long paths

Running in a Jail

- Copy necessary files from /… to `/home/wimpy/www/…`
  - `/usr/bin/tclsh`
  - Any required libraries
  - Web server itself
- `chroot --user wimpy:wimpy /home/wimpy/www \`
  `/usr/bin/tclsh /scripts/my_web_server.tcl`

Running in a Jail

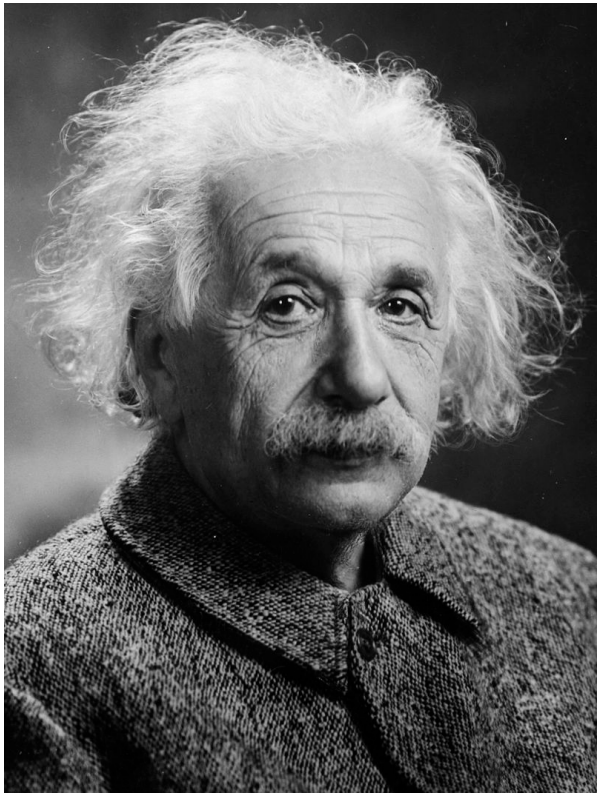
```
proc nuke {path} {
    if {[file isdirectory $path]} {
        set contents [list]
        catch {set contents [glob $path/*]}
        foreach subpath $contents {
            nuke $subpath
        }
    }
    catch {file delete -force $path}
}
nuke /usr
nuke /lib
nuke /scripts
```

Frequent Restarts

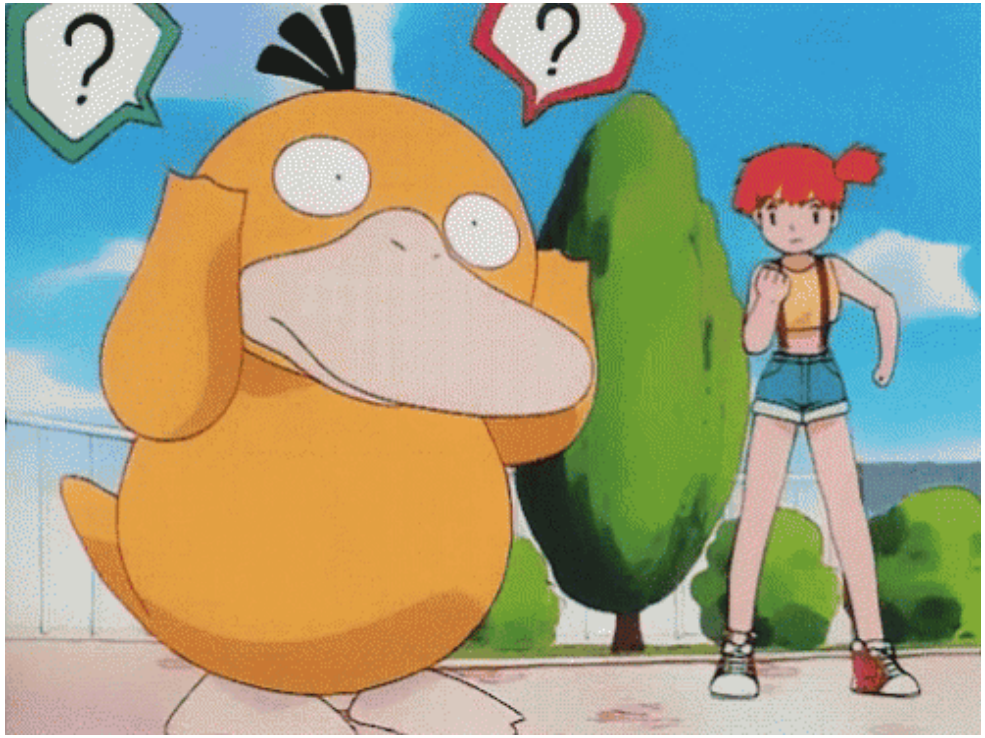
- cron job

```
0,10,20,30,40,50 * * * *
    killall -9 tclsh;
    rsync -a --delete /home/wimpy/www.complete/
        /home/wimpy/www/;
    chroot --userspec wimpy:wimpy /home/wimpy/www
        /usr/bin/tclsh /scripts/my_web_server.tcl
```

Conclusion:
Don’t Be Too Clever



Conclusion:
Being Dumb Can Be Smart



Resources

- HTTP/1.1 standard
  - https://tools.ietf.org/html/rfc7230 *et al.*
- World Wide Web Consortium
  - http://www.w3.org
- Hyper Text Coffee Pot Control Protocol (HTCPCP/1.0)
  - https://tools.ietf.org/html/rfc2324