Creating secure web based user interfaces for Embedded Devices

© 2011 Essensium N.V.
This work is licensed under a Creative Commons Attribution-ShareAlike 3.0 Unported License
How do you talk to an embedded system?
Web interface is the easiest!

- Custom display
- RS232 – Shell
- Ssh
- Custom protocol
- HTTP
Creating secure web based user interfaces for Embedded Devices

Arnout Vandecappelle
arnout@mind.be
Overview

1 Why web-based user interfaces?

2 Adding an HTTP server to firmware

3 Security considerations

4 Using a browser as the UI
Traditional firmware architecture

- ttyS0
  - I/O thread
- ttyS1
  - I/O thread
- ssh
  - custom_sh
  - I/O thread

Main application thread
Adding a web interface to traditional firmware architecture

Main application thread

ttyS0
I/O thread

ttyS1
I/O thread

ssh
connector
I/O thread

apache
CGI/php
I/O thread
Benefits of a web interface

- Browsers are universal
  - availability
  - familiarity

- The network is everywhere

- Browsers allow a rich interface

- A web-interface is the cheapest interface
  - only software (and a network connection)
Adding a web interface to traditional firmware architecture has many problems

- Extra processes
- Requires RPC mechanism
- Inconsistent interfaces (HTML/Text)
- Extra footprint from web server
Web interface architecture: embedded

- ttyS0
- ttyS1
- ssh
- Main application thread
- I/O thread
- I/O thread
- connector
- Embedded web server thread
- HTTP client
Web interface architecture: embedded and no other interfaces

HTTP client  HTTP client  HTTP client

Embedded web server thread

HTML formatter

Main application thread
Web interface architecture: embedded and no other interfaces

- No extra processes needed
- Only one type of interface
- Stay within your application language
The embedded web server

HTTP client  HTTP client  HTTP client

Embedded web server thread

HTML formatter

Main application thread
Tasks of the embedded web server

- Listen & accept connections
- Parse HTTP request (GET/POST/PUT/...)
- Parse HTTP headers
- Parse URI and find a handler for it
  - Serve static pages directly from filesystem
  - Provide MIME-type for static pages
  - Serve dynamic pages by calling a handler function
Mongoose as an embedded web server

- 40 Kilobytes
- License: MIT/X11 (BSD like)
- Small, easy to hack
- Actively developed
- Multiplatform (Posix, RTEMS, Windows (mingw/MSVC))
- Unit tests
- SSI (Server Side Includes), Directory listing, ...
- CGI calls
- HTTP Authenticate:, Range:, Etag:, ...
- SSL (see later)

http://code.google.com/p/mongoose/
```c
ctx = mg_start(&event_handler, NULL, options);
...
mg_stop(ctx);

static void *event_handler(enum mg_event event, 
    struct mg_connection *conn, 
    const struct mg_request_info *request_info)
{
    if (event == MG_NEW_REQUEST) {
        if (!request_info->is_ssl) {
            return redirect_to_ssl(conn, request_info);
        } else if (!is_authorized(conn, request_info)) {
            return redirect_to_login(conn, request_info);
        } else if (strcmp(request_info->uri, "/get_messages") == 0) {
            return ajax_get_messages(conn, request_info);
        }
    }
    return NULL;
}
```
Mongoose limitations

- Not used very heavily → limited testing
- Not a lot of API for embedded handlers
  - e.g. generate standard headers
  - e.g. calculate Etag: etc.
- No IPv6 support (yet)
- Still very young
  - API subject to change
  - Will it survive?

- Contributions very welcome!
Alternatives

  - GPL/Commercial
  - PHP-like embedding of C/C++
  - Slightly larger (?); certainly more complex
  - IPv6 support

- **libwthttpd** (see later)

- **Non-embedded**
  - Apache
  - Lighttpd
  - Monkey
  - ...
HTTP security

Main application thread

Embedded web server thread

HTML formatter

HTTP client

HTTP client

HTTP client

...
HTTP security

Web based embedded system is vulnerable
- Port 80 is attacked
- Anybody can try to connect
- Text based communication → buffer overflows
- Authentication → password sniffing
- Request forgery and replay attacks
Tips to protect the web service

- Time out connections otherwise you run out of threads

- HTTP Digest Authentication otherwise passwords can be sniffed

- URL-encoding of session
  - Always use a different URL
  - If bookmarked → redirect to login page first

- SSL/TLS
Comparison of SSL libraries

- **GnuTLS**
  - License: LGPL
  - Pretty complete

- **OpenSSL**
  - License: BSD with advertising clause
  - Most well-known
  - Large and clumsy

- **CyaSSL**
  - License: GPL/Commercial
  - Specifically targeted at embedded
    - focuses on most used features
  - Optimized for speed (e.g. assembly for embedded uPs)
  - OpenSSL API (simplified)
## Comparison of SSL libraries: Protocol support and library size

<table>
<thead>
<tr>
<th>Library</th>
<th>SSLv2</th>
<th>SSLv3</th>
<th>TLSv1.0</th>
<th>TLSv1.1</th>
<th>TLSv1.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>GnuTLS</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>OpenSSL</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>CyaSSL</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Platform</th>
<th>GnuTLS</th>
<th>OpenSSL</th>
<th>CyaSSL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Debian x86</td>
<td>944K</td>
<td>1649K</td>
<td>90K</td>
</tr>
<tr>
<td>OpenWrt MIPS</td>
<td>323K</td>
<td>506K</td>
<td>60K</td>
</tr>
</tbody>
</table>
This Connection is Untrusted

You have asked Firefox to connect securely to localhost, but we can't confirm that your connection is secure.

Normally, when you try to connect securely, sites will present trusted identification to prove that you are going to the right place. However, this site's identity can't be verified.

What Should I Do?

If you usually connect to this site without problems, this error could mean that someone is trying to impersonate the site, and you shouldn't continue.

Get me out of here!

Technical Details

localhost uses an invalid security certificate.

The certificate is not trusted because it is self-signed. The certificate is only valid for 10.1.2.3

(Error code: sec_error_untrusted_issuer)

I Understand the Risks
Certificate management

- Root CA
- Production CA
- Device certificate
Generating HTML content

HTTP client  HTTP client  HTTP client

Embedded web server thread

HTML formatter

Main application thread
Generating HTML content

- Status pages
  - AJAX or long polling to refresh it
- Forms to manipulate settings
  - CSS to make it look nice
  - Javascript (JQuery) to pre-verify constraints
- Graphical output
  - Image maps
  - HTML5 canvas
  - Javascript plotting
- Internationalization: serve pages in the user's language
  - Accept-Language
  - Translations
Cross-browser compatibility

TIME BREAKDOWN OF MODERN WEB DESIGN

- Time spent trying to get the bastard to work in Internet fucking Explorer.
- Time spent wishing a slow, painful death on Bill Gates and more swearing.
- Time spent looking for that one extra space character in the JavaScript that Fire Fox is throwing a wobbly over.
- Time spent actually designing anything.
- Time spent making the site W3C compliant.
- Swearing
- Time spent trying to get the layout to work using only CSS before giving up and using tables.

©2006 Alan "IE users must DIE" Foreman poisonedminds.com
Libraries for cross-browser compatibility

  - 30K
  - License: MIT
  - *The standard on the web*
  - Still need to write a lot of Javascript
    → doesn't solve all cross-browser issues

- Wt [http://www.webtoolkit.eu/wt](http://www.webtoolkit.eu/wt)
  - C++ UI library
  - 2MB (incl. Webserver)
  - License: GPL/Commercial
  - Stay within 1 programming language
    no javascript required
Wt is a UI library for web applications

Server: application session

WContainerWidget: root()

WContainerWidget

WText
  text = "Your name, please"

WLineEdit
  text = "Koen"

WPushButton
  text = "Greet me"

WText
  text = "Hello there, Koen"

widget tree (C++)

Browser

<div id="...">
  <div id="...">
    <span id="...">Your name, please</span>
  
  <input type="text" value="Koen" onkeypress="..."/>
  <button onclick="..." >Greet me</button>

  <span id="...">Hello there, Koen</span>
</div>

DOM tree ((X)HTML/CSS/JavaScript)

HTTP

{ (Modified) form field values
  Events
}

Updated widget tree
Conclusions

- Web interface is the cheapest UI for embedded systems
- Embed it directly into your application
  mongoose, klone, libwthttp
- Don't forget about security
- Cross-browser support is difficult
  → use jQuery or Wt
www.mind.be

www.essensium.com

Essensium NV
Mind - Embedded Software Division
Gaston Geenslaan 9, B-3001 Leuven
Tel : +32 16-28 65 00
Fax : +32 16-28 65 01
email : info@essensium.com