Security and Privacy, RSS, Dynamic Documents

Security and Privacy Technologies

- Platform for Privacy Preferences (P3P)
- Platform for Internet Content Selection (PICS)
- Transport Layer Security (TLS) / Secure Socket Layer (SSL)

Privacy and Security Concerns

User / Client
- Fraud: The server or site is "who" it appears to be. "Phishing" is a common threat.
- Safe content: Content received is safe — it is not dangerous or malicious.
- Privacy Infringement: Information given to server will be kept private.

Provider / Server
- Break-in: Server machine will not be broken into.
- "Webjacking": Content will not be altered.
- Denial-of-Service (DOS) Attacks: Server will be available.
- Authentication / Authorization: Users will not access documents not meant for them
- Fraud: Users are who they claim to be.

Common
- Eavesdropping: Third parties are not eavesdropping on the information sent between the client and server.
- Tampering: Information sent between the client and server arrives intact.

Security and Privacy Technologies

- Platform for Privacy Preferences (P3P)
- Platform for Internet Content Selection (PICS)
- Transport Layer Security (TLS) / Secure Socket Layer (SSL)

P3P Initiative

- W3C Privacy Overview
- Platform for Privacy Preferences (P3P)

The Platform for Privacy Preferences Project (P3P), developed by the World Wide Web Consortium, is emerging as an industry standard providing a simple, automated way for users to gain more control over the use of personal information on Web sites they visit. At its most basic level, P3P is a standardized set of multiple-choice questions, covering all the major aspects of a Web site’s privacy policies. Taken together, they present a clear snapshot of how a site handles personal information about its users. P3P-enabled Web sites make this information available in a standard, machine-readable format. P3P-aware browsers can “read” this data automatically and compare it to the consumer’s own set of privacy preferences. P3P enhances user control by putting privacy policies where users can find them, in a form users can understand, and, most importantly, enables users to act on what they see.

Readings
- P3P: Privacy Primer from O'Reilly Network
- Help! IE6 Is Blocking My Cookies from O'Reilly Network
- Web Privacy with P3P by Lorrie Cranor, published by O'Reilly
Example Policies: English

The following is taken from The Platform for Privacy Preferences 1.0 (P3P1.0) Specification

Copyright © 1999 W3C® (MIT, INRIA, Keio), All Rights Reserved.

CoolCatalog makes the following statement for the Web page at http://www.CoolCatalog.com/catalog/.

We use cookies and collect your gender, information about your clothing preferences, and (optionally) your home address to customize our entry catalog pages and for our own research and product development. We do not use this information in identifiable form.

We also maintain server logs that include information about visits to the http://www.CoolCatalog.com/catalog/ page, and the types of browsers our visitors use. We use this information in order to maintain and improve our web site. We do not use this information in an identifiable way.

We do not provide access capabilities to information we may have from you, but we do have retention and opt-out policies, which you can read more about at our privacy page http://www.CoolCatalog.com/PrivacyPractice.html. The third-party PrivacySeal.org provides assurance that we abide by this agreement.

Example Policies: P3P Syntax

The following is taken from The Platform for Privacy Preferences 1.0 (P3P1.0) Specification

Copyright © 1999 W3C® (MIT, INRIA, Keio), All Rights Reserved.

Entity: CoolCatalog, Inc.
Access to Identifiable Information: none
Assurance: PrivacySeal.org
Other disclosures: Change agreement, retention

We collect:
dynamic.cookies (category = state)
user.gender
dynamic.miscdata (category = pref)    user.home.  (optional)

For purpose: Customization of the site to individuals, research and development
Identifiable use: No
Recipients: Only ourselves and our agents
Consequence: A site with clothes you would appreciate

We collect:
dynamic.clickstream.server
dynamic.http.useragent

For purpose: Web site and system administration, research and development
Identifiable use: No
Recipients: Only ourselves and our agents

Example Policies: P3P Syntax in XML

The following is taken from The Platform for Privacy Preferences 1.0 (P3P1.0) Specification

Copyright © 1999 W3C® (MIT, INRIA, Keio), All Rights Reserved.

```xml
<POLICY xmlns="http://www.w3.org/2000/P3Pv1" entity="CoolCatalog, Inc.">
  <ASSURANCE-GROUP>
  ...
  discuri="http://www.CoolCatalog.com/PrivacyPractice.html" access="none" retention="yes" change_agreement="yes"/>
  <STATEMENT>
    <IDENTIFIABLE><no/></IDENTIFIABLE>
    <CONSEQUENCE-GROUP>
      <CONSEQUENCE>a site with clothes you would appreciate</CONSEQUENCE>
      <RECIPIENT><ours/></RECIPIENT>
      <PURPOSE><custom/><develop/></PURPOSE>
      <DATA-GROUP>
        <DATA name="dynamic.cookies" category="state"/>
        <DATA name="dynamic.miscdata" category="pref"/>
        <DATA name="user.gender"/>
        <DATA name="user.home." optional="yes"/>
      </DATA-GROUP>
    </STATEMENT>
    <STATEMENT>
      <IDENTIFIABLE><no/></IDENTIFIABLE>
      <RECIPIENT><ours/></RECIPIENT>
      <PURPOSE><admin/><develop/></PURPOSE>
      <DATA-GROUP>
        <DATA name="dynamic.clickstream.server"/>
        <DATA name="dynamic.http.useragent"/>
      </DATA-GROUP>
    </STATEMENT>
  </STATEMENT></POLICY>
```

Example Policies: P3P in HTTP Header, Compact P3P Policy

An example with Disney.com

```
P3P: CP="CaOD DSP CUR.LIR ADSM MkXr TALaDA PsMo PaaS dC/uSdIg CMUI C/Oa/ T Imge COM NAV INT DEM CNT STA PRE"
```

GET http://disney.go.com/
User-Agent: lwp-request/2.06
GET http://www.disney.com/ --> 301 Moved Permanently
GET http://disney.go.com/ --> 200 OK
Cache-Control: max-age=300
Connection: close
Date: Tue, 02 May 2006 19:49:52 GMT
Pragma: no-cache
From: DOLWEB10
Accept-Ranges: bytes
ETag: "80559abfeac51:e82"
Server: Microsoft-IIS/5.0
Content-Length: 8862
Content-Type: text/html
Content-Type: text/html; charset=iso-8859-1
Expires: Thu, 02 Jan 2003 16:00:00 GMT
Last-Modified: Tue, 15 Nov 2005 18:04:39 GMT
Cache-Expires: Tue, 02 May 2006 19:53:11 GMT
Client-Date: Tue, 02 May 2006 19:49:54 GMT
Client-Peer: ...
```
Privacy for Your Web Site
Making Your Web Site P3P Compliant

Third Party Privacy Services

Privacy for Your Web Site
Making Your Web Site P3P Compliant

Third Party Privacy Services

Privacy for Your Web Site
Making Your Web Site P3P Compliant

Third Party Privacy Services

Privacy for Your Web Site
Making Your Web Site P3P Compliant

Third Party Privacy Services

Privacy for Your Web Site
Making Your Web Site P3P Compliant

Third Party Privacy Services

Privacy for Your Web Site
Making Your Web Site P3P Compliant

Third Party Privacy Services

Privacy for Your Web Site
Making Your Web Site P3P Compliant

Third Party Privacy Services

Privacy for Your Web Site
Making Your Web Site P3P Compliant

Third Party Privacy Services
General Internet Security

SecurityFocus
http://www.securityfocus.com/
SecurityFocus is designed to facilitate discussion on security related topics, create security awareness, and to provide the Internet's largest and most comprehensive database of security knowledge and resources to the public. It also hosts the BUGTRAQ mailing list.

CERT/CC
http://www.cert.org/
The CERT Coordination Center (CERT/CC) is located at the Software Engineering Institute (SEI), a federally funded research and development center at Carnegie Mellon University in Pittsburgh, Pennsylvania. Following the Morris worm incident, which brought 10 percent of Internet systems to a halt in November 1988, the Defense Advanced Research Projects Agency (DARPA) charged the SEI with setting up a center to coordinate communication among experts during security emergencies and to help prevent future incidents. Since then, the CERT/CC has helped to establish other response teams and our incident handling practices have been adapted by more than 90 response teams around the world.

Basic Cryptography - the concepts of TLS / SSL / Digital Signatures

Terms

Cryptography Pieces

Symmetric Key Cryptography

Public Key Cryptography
Digital Signatures

Public key cryptography is much slower than symmetric key cryptography. In practice, the two systems are usually combined:

1. Establish a common session key.
   - Browser generates a "session key".
   - Browser sends encrypted session key to server.
   - Server decrypts session key with server's private key.
2. Send Messages.
   - Browser and server use common session key to encrypt and decrypt messages.

Digital Envelopes

Public Key + Signature

Certifying Authorities (CAs)

1. Generate a public/private key pair.
2. Send your public key to a CA.
3. CA verifies your identity and signs your public key with its own private key.
4. Your signed certificate identifies you, analogous to your "passport".

Digital Certificates and Public Key Infrastructure

Some CA Companies

- CyberTrust
- Verisign
- Type II
- RSA
- GigaTrust
- BatMix
- Phase One
- VeriSign
- Tesseract
Transport Layer Security (TLS) and Secure Sockets Layer (SSL)

- Encryption system
- Operations at the TCP/IP layer (not specific to HTTP)
- Server Certificate
- Client Certificate
- Cipher Suite (symmetric; message digests; authentication) negotiated between client and server

RSS

Rich Site Summary (RSS) is a lightweight XML format designed for sharing headlines and other Web content.

Articles: What is RSS? by Mark Pilgrim, XML.com

Example RSS feed: http://mlb.mlb.com/partnerxml/gen/news/rss/bos.xml

Item, Link, Description
<item>    <title>Loretta, Ortiz deliver Sox past Yanks</title>    <link>http://mlb.mlb.com/NASApp/mlb/news/gameday_recap.jsp?ymd=20060501&amp;content_id=1427610&amp;vkey=recap&amp;fext=.jsp&amp;c_id=bos&amp;partnered=rss_bos</link>    <pubDate>Tue, 02 May 2006 00:32:00 EDT</pubDate>    <guid>http://mlb.mlb.com/NASApp/mlb/news/gameday_recap.jsp?ymd=20060501&amp;content_id=1427610&amp;vkey=recap&amp;fext=.jsp&amp;c_id=bos&amp;partnered=rss_bos</guid>    <description>Loretta, Papi deliver Sox past Yankees</description>    <content:encoded>Any slumping hitter will tell you that sometimes it just takes a fortunate bounce to snap out of a slump. Mark Loretta might have gotten his slump buster at a most crucial moment Monday night at Fenway Park.

Ian Browne</content:encoded>  </item>

Example providers of RSS feeds
- BBC
- CNN
- NPR
- Boston.com News
- Boston.com Sports

What to do with RSS?

Produce
- Indicate RSS feeds using "link" element in page:
  <link rel="alternate" type="application/rss+xml" title="ONLamp.com Articles" href="http://www.oreillynet.com/meerkat/?_fl=rss10&t=ALL&c=5544" />
  <link rel="alternate" type="application/atom+xml" title="ONLamp.com Articles" href="http://www.oreillynet.com/meerkat/?_fl=atom&t=ALL&c=5544" />
- Link to RSS feed with the convention of an "XML/RSS" icon on page.

Consume
Consume

- External RSS reader application
- RSS reader built-in to e-mail client
- RSS reader built-in to Web browser

Firefox "Live Bookmarks"

Dynamically Created Documents

- XHTML (or image or other content) is created at the time of the request.
- Content created from dynamic process "off-line"

Dynamic Document Solutions

- CGI
- Client-Side Scripting
- Rendering Server

Issues to Consider

- Portability
  - change of server (Apache, Netscape, IIS, WebSite, JigSaw)
  - change of operating system (UNIX flavors, NT)
  - new browsers and new browser versions
- Performance
  - speed of individual request
  - throughput
- Simplicity
  - Develop
  - Maintain
- Power
  - can you do what you need to do?
  - can you do what you want to do?
Languages commonly used for programs running under CGI:
- Perl
- Python
- C/C++
- Tcl
- VBScript
- Any language (compiled or interpreted) that will run on the system.

**CGI Program**

**Life Cycle of a CGI Program**

1. For each request, the server creates a new process and the process initializes itself.
2. The HTTP server passes the request information to the CGI program as environment variables.
3. The HTTP server sends any client input (such as user-entered field values from an HTML form) to the CGI program's standard input (STDIN) or via the QUERY_STRING environmental variable.
4. The CGI program writes any output to be returned to the client as standard output (STDOUT). Error information (write to standard error (STDERR)) is logged by the HTTP server.
5. When the CGI process exits, the request is complete.

**Perl and CGI**

- *Programming Perl* (Officially) — unofficially known as the “Camel Book” this is the best Perl book available.
- *Perl in a Nutshell* (Officially) has 5 chapters that deal with CGI.
- *Perl Cookbook* (Officially) is also useful as it has hundreds of “recipes” on how to perform certain tasks in Perl, many of which are useful for CGI and many of which are specific to CGI.
- *Official Guide to Programming with CGI.pm* (Larry Wall, published by O'Reilly). This is freely available at the CGI.pm documentation.

**CGI Examples: first.cgi**

```
#!/usr/local/bin/perl
print "Content-type: text/html\n\n";
print "<HTML><BODY><H1>Hello, World!</H1></BODY><HTML>
```
CGI Examples: second.cgi

#!/usr/local/bin/perl
use CGI qw(:all); # CGI.pm module is very useful!
print header,
start_html,
h1("Hello, World!"),
end_html;

CGI Examples: third.cgi

#!/usr/local/bin/perl
use CGI qw(:all); # CGI.pm module is very useful!
my $q = new CGI; # create a new CGI object
my $name = $q->param('name');
print header,
start_html,
h1("Hello, $name!")
end_html;

CGI Examples: fourth.cgi

#!/usr/local/bin/perl
use CGI qw(:all); # CGI.pm module is very useful!
my $q = new CGI; # create a new CGI object
my $name = $q->param('name');
if ($name) {
    print header,
    start_html,
    h1("Hello, $name!")
    end_html;
} else {
    print header,
h1("Enter name:"),
start_form,
textfield(-name="name"),
br,
submit,
br,
reset,
end_form,
end_html;
}

Server API

- NSAPI
- ISAPI
- Apache API
- Apache Modules
- Many third-party modules exist for Apache that allow you add a wide array of functionality to Apache.
Web Development Languages / Frameworks

- Apache SSI and XSSI
- JSP
- ASP, JSP, Struts
- AJAX, XSLT
- Cold Fusion, Astrobook
- The Apache Port Language Integration Project
- Ruby on Rails

HTTP Client and Server

CGI Life Cycle

External and Persistent

- JSP Servlets
- ColdFusion
- FastCGI
- CGI: SpeddyCGI
Internal

- CGI
- Perl/CGI
- PHP
- Python under Apache
- Ruby under Apache
- Java under Apache/mod_perl

Dynamic Documents: "Developer End"

- Programs with HTML embedded
- HTML with programs embedded
- Separate concerns: programs that look like programs, HTML that looks like HTML

Programs that output HTML

- CGI
- Servlets

Example

```java
package simple;
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
/** Very simplistic servlet. */
public class HelloWorld extends HttpServlet {
  public void doGet(HttpServletRequest request,
                      HttpServletResponse response)
                   throws ServletException, IOException {
    PrintWriter out = response.getWriter();
    out.println("<html>");
    out.println("<body>");
    out.println("<h1>");
    out.println("hello");
    out.println("</h1>");
    out.println("</body>");
    out.println("</html>");
  }
}
```
### HTML with program statements

- SSI
- ASP
- JSP
- PHP
- Various Perl-based systems
  - Embperl (Embedded Perl)
  - Mason
  - Template Toolkit

### Simple JSP Example

```html
<html>
<body>
<h1>Some dynamic content created using various JSP mechanisms:</h1>
<ul>
  <li><strong>Expression.</strong>
      Your hostname: 
      <% out.println(request.getRemoteHost()); %>
  </li>
  <li><strong>Scriptlet.</strong>
      Attached GET data: 
      <% out.println(request.getQueryString()); %>
  </li>
  <li><strong>Declaration (plus expression).</strong>
      Accesses to page since server restart: 
      <%= ++accessCount %>
  </li>
  <li><strong>Directive (plus expression).</strong>
      Current date: <%= new Date() %>
</ul>
</body>
</html>
```

### Model, View, Controller (MVC)

MVC separates:
- data model
- user interface
- control logic

### Templates

- Ease of maintenance
- Consistency
- Division of labor and responsibility

### Template Toolkit Example

```
class DCEtt {
  my %course_groups = (
    'groups' => ['course_groups']
  );
}
```
header and footer

header

Oracle Table

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACAD_YEAR</td>
<td>NUMBER</td>
</tr>
<tr>
<td>TERM_ID</td>
<td>NUMBER</td>
</tr>
<tr>
<td>CRN</td>
<td>NUMBER</td>
</tr>
<tr>
<td>TERM_NAME</td>
<td>VARCHAR</td>
</tr>
<tr>
<td>COURSE_GROUP</td>
<td>VARCHAR</td>
</tr>
<tr>
<td>COURSE_NUM</td>
<td>VARCHAR</td>
</tr>
<tr>
<td>TITLE</td>
<td>VARCHAR</td>
</tr>
<tr>
<td>MEETING_DAYS</td>
<td>VARCHAR</td>
</tr>
<tr>
<td>MEETING_BEGIN</td>
<td>NUMBER</td>
</tr>
<tr>
<td>MEETING_END</td>
<td>NUMBER</td>
</tr>
<tr>
<td>LOCATION</td>
<td>VARCHAR</td>
</tr>
<tr>
<td>DESCRIPTION</td>
<td>VARCHAR</td>
</tr>
<tr>
<td>COURSE_GROUP_NAME</td>
<td>VARCHAR</td>
</tr>
<tr>
<td>COURSE_TYPE</td>
<td>VARCHAR</td>
</tr>
<tr>
<td>COURSE_HEAD1_HU_ID</td>
<td>CHAR</td>
</tr>
<tr>
<td>COURSE_HEAD1_NAME</td>
<td>VARCHAR</td>
</tr>
<tr>
<td>COURSE_HEAD1_TITLE</td>
<td>VARCHAR</td>
</tr>
<tr>
<td>COURSE_HEAD2_HU_ID</td>
<td>CHAR</td>
</tr>
<tr>
<td>COURSE_HEAD2_NAME</td>
<td>VARCHAR</td>
</tr>
<tr>
<td>COURSE_HEAD2_TITLE</td>
<td>VARCHAR</td>
</tr>
<tr>
<td>COURSE_HEAD3_HU_ID</td>
<td>CHAR</td>
</tr>
<tr>
<td>COURSE_HEAD3_NAME</td>
<td>VARCHAR</td>
</tr>
<tr>
<td>COURSE_HEAD3_TITLE</td>
<td>VARCHAR</td>
</tr>
<tr>
<td>COURSE_HEAD4_HU_ID</td>
<td>CHAR</td>
</tr>
<tr>
<td>COURSE_HEAD4_NAME</td>
<td>VARCHAR</td>
</tr>
<tr>
<td>COURSE_HEAD4_TITLE</td>
<td>VARCHAR</td>
</tr>
</tbody>
</table>

footer

Template: course_info

```
# set up Perl environment and CGI module
use lib qw(/home/c/s/cscie12/perllib /home/c/s/cscie12/perl);
use CGI::Carp qw(fatalsToBrowser);
use DCEtt;
use CGI;

my $q = new CGI;

# set up Oracle stuff
use DBI;$ENV{ORACLE_HOME} = '/oracle';
$ENV{EPC_DISABLED} = 'true';
$ENV{TWO_TASK} = 'fasdv';
my $login_info = 'xxxxx/xxxxx';

# establish database connection
$dbh = DBI->connect('DBI:Oracle:', $login_info, '')
  || die "ERROR: $DBI::errstr
"

# get search information from submitted parameters
my $term_id = $q->param('term_id');
my $acad_year = $q->param('acad_year');
my $crn = $q->param('crn')

# SQL statement
$sth = $dbh->prepare(qq{
  SELECT *
  FROM   dcero.course
  WHERE  term_id = ? and acad_year = ? and crn = ?
});

# execute SQL statement
$sth->execute($term_id, $acad_year, $crn);

# retrieve course information from Oracle into a Perl data structure
my $data = $sth->fetchrow_hashref;
my $webdata = {};
$webdata->{'course'} = $data;

print $q->header();

# process template
my $file = 'course_info';
my $template = DCEtt->template();
$template->process($file, $webdata) || die $template->error();
```
## Template Toolkit Build

```

ttemplate --accept .html
> -s src -d html -l tt
> --copy .png > --copy .gif > --copy .jpg
> --copy .css
> --pre_chomp --post_chomp --trim
> -a

ttree 2.75 (Template Toolkit version 2.13)
Source: src
Destination: html
Include Path: [ tt ]
Ignore: [ ^CVS|^RCS ]
Copy: [ .png, .gif, .jpg, .css ]
Accept: [ .html ]
Suffix: [ ]

contact.html
+ directions.html
+ hours.html
> images/building.gif
> images/building.jpg
> images/building2.jpg
> images/building3.jpg
> images/map.gif
> images/pinocchio1.jpg
> images/pinocchio_216.gif
> images/pinocchio_72.gif
> images/pinocchio_72.jpg
> images/sign.jpg
+ index.html
+ menu.html
+ sicilian_vs_regular.html
+ site.css
```

And now look in the "html" directory:

```

```

## Simple Template Example: contact.html

```

```

```

```

```

```

```

And "header" is:

```

```

```

```

```

```

```

And now look in the "html" directory:

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```
TT: Datafiles

Menu data is:

<table>
<thead>
<tr>
<th>Item</th>
<th>Small</th>
<th>Large</th>
</tr>
</thead>
<tbody>
<tr>
<td>subs</td>
<td>Cheese</td>
<td>4.50</td>
</tr>
<tr>
<td>subs</td>
<td>Italian</td>
<td>4.75</td>
</tr>
<tr>
<td>subs</td>
<td>Ham + Cheese</td>
<td>4.75</td>
</tr>
<tr>
<td>subs</td>
<td>Meatball</td>
<td>4.75</td>
</tr>
<tr>
<td>subs</td>
<td>Tuna</td>
<td>4.75</td>
</tr>
<tr>
<td>subs</td>
<td>Turkey</td>
<td>5.50</td>
</tr>
<tr>
<td>subs</td>
<td>Chicken Parmigiana</td>
<td>5.50</td>
</tr>
<tr>
<td>subs</td>
<td>Roast Beef</td>
<td>5.50</td>
</tr>
<tr>
<td>subs</td>
<td>Eggplant Parmigiana</td>
<td>4.75</td>
</tr>
<tr>
<td>subs</td>
<td>Sausage, Peppers &amp; Onions</td>
<td>6.50</td>
</tr>
<tr>
<td>subs</td>
<td>Steak</td>
<td>5.00</td>
</tr>
<tr>
<td>subs</td>
<td>Steak + Cheese</td>
<td>5.50</td>
</tr>
<tr>
<td>subs</td>
<td>+ Mushrooms</td>
<td>+0.50</td>
</tr>
<tr>
<td>subs</td>
<td>+ Green Peppers</td>
<td>+0.50</td>
</tr>
<tr>
<td>subs</td>
<td>+ Onions</td>
<td>+0.50</td>
</tr>
<tr>
<td>subs</td>
<td>Hamburger</td>
<td>3.60</td>
</tr>
<tr>
<td>subs</td>
<td>Cheeseburger</td>
<td>4.10</td>
</tr>
<tr>
<td>subs</td>
<td>Fried Chicken</td>
<td>5.75</td>
</tr>
<tr>
<td>subs</td>
<td>Veggie</td>
<td>5.60</td>
</tr>
<tr>
<td>subs</td>
<td>Extra Cheese on any sub</td>
<td>+0.50</td>
</tr>
</tbody>
</table>

And the menu.html template is:

```
[INCLUDE header title = 'Menu']
<% USE menudata = datafile('/home/staff/dheitmey/public_html/pinocchio/data/menu.txt', delim = '|')%>
<table width="100%">
  <tbody>
    <tr>
      <td width="50%">
        <div class="callout">
          <h2>Our Menu</h2>
          <table width="100%" class="foodmenu">
            <tbody>
              <tr><th colspan="3"><a name="subs">Subs</a></th></tr>
              <tr><td></td><td>Small</td><td>Large</td></tr>
              <% FOREACH item = menudata %>
              <% IF item.type == 'subs' %>
              <tr><td style="text-align: left;" width="50%">[% item.item %]</td><td width="25%">[% item.small %]</td><td width="25%">[% item.large %]</td></tr>
              <%- END -%>
              <%- END -%>
            </tbody>
          </table>
        </div>
      </td>
    </tr>
  </tbody>
</table>
[INCLUDE footer]
```