Pretty Printing NMAP XML Files for Audits Presented to Management

Prepared by

Thomas J. Munn

December 29, 2006

Nmap the program has a useful feature, called by invoking the NMAP utility with the –oX switch. This switch saves the scan results into xml format easily viewable in any modern browser. The problem is that these XML files aren't that portable, and that they will not render as on screen when printing. The screen printed version is attractive, and more importantly, color coded. The printed version, however, is in monochrome, and not as easy to read, and are not in a format liked well by upper management.

One of the most important things in presenting a document to management is having it look 'pretty'. They like colors, and easy-to-read summaries. The on-screen presentation of NMAP xml documents meets this 'prettiness' criterion. This paper will tell you how to get NMAP xml documents to print in color, with a final output format of Adobe Acrobat format.

There are two methods described here: The 'hard' way and the 'easy' way. The 'hard' way has the additional benefit of providing you with a regular .html file that is independently transportable. The 'easy' way just gets the .pdf file created, but does not give you an independently transportable html file.

The nmap.xsl file can be found on UNIX by using the 'locate' command locate nmap.xsl. In windows, just use the 'search' functionality of your operating system to locate this file needed for the net section. Put all your scan results, and the nmap.xsl file in the SAME directory!

The "Hard" way

Step 1: Edit the nmap.xsl file and look for the 'screen' style. You will want to copy this entire section over the 'print' style section in the nmap.xsl file. This will prevent you from having to manually edit each html files so that they 'pretty print' to the printer/pdf printer. Be sure to keep the parenthesis balanced or things will be foobard when you try to print or use the nmap.xsl document! Be sure to rename the style back to 'print' when you are done pasting and editing.Step

Step2: Convert .xml file to an HTML file using Xalan.

Xalan is a utility that will convert xml and .xsl style documents into a single html file. It is available at http://xml.apache.org/xalan-c/ I recommend the "c" version as it is faster and easier to deal with than the java version. You will want to download both the 'xerces' and xalan packages once you choose the download option. You will also need to choose the appropriate binary for your system, either UNIX or Windows. You will need to copy the .dlls from the xerces binary package into the xalan directory on your computer in order to get xalan working. In Unix, just copy the .dlls to your /usr/lib directory.

Once you are done with this, the conversion assumes the following:

1. That you have completed your nmap scan, and ran nmap with the –oX option to output an .xml file of the scan results
2. that you have a copy of the nmap.xsl file in the current directory with your .xml files
3. That said files are in the same directory with the xalan binary.

Assuming that these steps are completed, the process is quite simple:

```
Xalan –o file.html scan.xml nmap.xsl
```

The first argument is the –o for output, which specifies the html file to create.
The second option is the .xml file that you created using nmap.
The third file exists on your system (assuming you have installed nmap from source) in the /usr/share/nmap/nmap.xsl directory. On windows I have no idea where this file resides. Search for it using the search functionality built into windows.

Step 3: Open the saved .html file in either firefox for windows or internet explorer (UNIX browsers don't seem to print as pretty as the Windows platform, Windows is good for something after all!), and print to a .pdf printer. I use once called 'cute pdf writer" which is free and works great.

The "easy" way

Step 1: Follow step 1 from the 'hard way'

Step 2: Edit the result.xml file stored in the SAME directory as the nmap.xsl template as follows: Change the top line of the html file to reference your locate nmap.xsl file that you have edited in the first step. Your completed line should look like this:

```
<?xml-stylesheet href="nmap.xsl"
```

Step 3: open up the .xsl file in your favorite 'Windoze' browser. Unix browsers, alas don't seem to pretty print as well as Windoze….(same disclaimer as the 'hard' way for Windoze noted!) and print to your favorite pdf creator (can be adobe or cutepdf writer).

Appendix A: Credits

Credits for the finding the xalan utility go to go to a ‘sean’ who chooses to remain anonymous.

Appendix B: Complete modified nmap.xsl to pretty print

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
=======================================================================
==
            nmap.xsl stylesheet version 0.9b
            last change: 2006-03-04
            Benjamin Erb, http://www.benjamin-erb.de
=======================================================================
=======
    Copyright (c) 2004-2006 Benjamin Erb
    All rights reserved.

    Redistribution and use in source and binary forms, with or without
    modification, are permitted provided that the following conditions
    are met:
    1. Redistributions of source code must retain the above copyright
       notice, this list of conditions and the following disclaimer.
    2. Redistributions in binary form must reproduce the above
copyright
       notice, this list of conditions and the following disclaimer in
the
       documentation and/or other materials provided with the
distribution.
    3. The name of the author may not be used to endorse or promote
products
       derived from this software without specific prior written
permission.

    THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS'' AND ANY EXPRESS
OR
    IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
WARRANTIES
    OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED.
    IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT,
    INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
(INCLUDING, BUT
    NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS
OF USE,
    DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON
ANY
    THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
    (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE
USE OF
    THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
=======================================================================
=== -->
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:fo="http://www.w3.org/1999/XSL/Format">
<xsl:output method="html" indent="yes" encoding="UTF-8" />

<!-- global variables      -->
<!-- ............................................................ -->
```

```
<xsl:variable name="nmap_xsl_version">0.9b</xsl:variable>
<!-- ............................................................ -->
<xsl:variable name="start"><xsl:value-of select="/nmaprun/@startstr"
/></xsl:variable>
<xsl:variable name="end"><xsl:value-of
select="/nmaprun/runstats/finished/@timestr" /> </xsl:variable>
<xsl:variable name="totaltime"><xsl:value-of
select="/nmaprun/runstats/finished/@time -/nmaprun/@start"
/></xsl:variable>
<!-- ............................................................ -->


<xsl:template match="/">
      <xsl:apply-templates/>
</xsl:template>


<!-- root -->
<!-- ............................................................ -->
<xsl:template match="/nmaprun">
<html>
<head>

<xsl:comment>generated with nmap.xsl - version <xsl:value-of
select="$nmap_xsl_version" /> by Benjamin Erb - http://www.benjamin-
erb.de/nmap_xsl.php </xsl:comment>

<style type="text/css">
/* stylesheet print */
@media print
{
    body
    {
      margin: 0px;
      background-color: #FFFFFF;
      color: #000000;
      text-align: center;
    }

    #container
    {
        text-align:left;
        margin: 10px auto;
        width: 90%;
    }

    h1
    {
      font-family: Verdana, Helvetica, sans-serif;
      font-weight:bold;
      font-size: 14pt;
      color: #000000;
        background-color:#87CEFA;
        margin:10px 0px 0px 0px;
        padding:5px 4px 5px 4px;
        width: 100%;
        border:1px solid black;
        text-align: left;
    }

    h2
    {
        font-family: Verdana, Helvetica, sans-serif;
        font-weight:bold;
        font-size: 11pt;
        color: #000000;
        margin:30px 0px 0px 0px;
        padding:4px;
        width: 100%;
        border:1px solid black;
        background-color:#F0F8FF;
        text-align: left;
    }

    h2.green
    {
        color: #000000;
        background-color:#CCFFCC;
        border-color:#006400;
    }

    h2.red
    {
        color: #000000;
        background-color:#FFCCCC;
        border-color:#8B0000;
    }

    h3
    {
        font-family: Verdana, Helvetica, sans-serif;
        font-weight:bold;
        font-size: 10pt;
        color:#000000;
        background-color: #FFFFFF;
        width: 75%;
        text-align: left;
    }

    p
    {
        font-family: Verdana, Helvetica, sans-serif;
        font-size: 8pt;
        color:#000000;
        background-color: #FFFFFF;
        width: 75%;
        text-align: left;
    }

    p i
    {
        font-family: "Courier New", Courier, mono;
        font-size: 8pt;
        color:#000000;
        background-color: #CCCCCC;
    }

    ul
    {
        font-family: Verdana, Helvetica, sans-serif;
        font-size: 8pt;
        color:#000000;
        background-color: #FFFFFF;
        width: 75%;
        text-align: left;
    }

    a
    {
        font-family: Verdana, Helvetica, sans-serif;
        text-decoration: none;
        font-size: 8pt;
        color:#000000;
        font-weight:bold;
        background-color: #FFFFFF;
        color: #000000;
    }

    li a
    {
        font-family: Verdana, Helvetica, sans-serif;
        text-decoration: none;
        font-size: 10pt;
        color:#000000;
        font-weight:bold;
        background-color: #FFFFFF;
        color: #000000;
    }

    a:hover
    {
          text-decoration: underline;
    }

    a.red
    {
        color:#8B0000;
    }
    a.green
    {
        color:#006400;
    }

    table
    {
        width: 80%;
        border:0px;
        color: #000000;
        background-color: #000000;
        margin:10px;
    }
```

```
    tr
    {
        vertical-align:top;
        font-family: Verdana, Helvetica, sans-serif;
        font-size: 8pt;
        color:#000000;
        background-color: #D1D1D1;
    }

    tr.head
    {
        background-color: #E1E1E1;
        color: #000000;
        font-weight:bold;
    }

    tr.open
    {
        background-color: #CCFFCC;
        color: #000000;
    }

    tr.filtered
    {
        background-color: #FFDDBB;
        color: #000000;
    }

    tr.closed
    {
        background-color: #FFAFAF;
        color: #000000;
    }

    td
    {
        padding:2px;
    }

    .status
    {
        display:none;
    }

    #menu li
    {
        display         : inline;
        margin          : 0;
        /*margin-right    : 10px;*/
        padding         : 0;
        list-style-type : none;
    }


}
```

```
/* stylesheet screen */
@media screen
{
    body
    {
      margin: 0px;
      background-color: #FFFFFF;
      color: #000000;
      text-align: center;
    }

    #container
    {
        text-align:left;
        margin: 10px auto;
        width: 90%;
    }

    h1
    {
      font-family: Verdana, Helvetica, sans-serif;
      font-weight:bold;
      font-size: 14pt;
      color: #000000;
        background-color:#87CEFA;
        margin:10px 0px 0px 0px;
        padding:5px 4px 5px 4px;
        width: 100%;
        border:1px solid black;
        text-align: left;
    }

    h2
    {
        font-family: Verdana, Helvetica, sans-serif;
        font-weight:bold;
        font-size: 11pt;
        color: #000000;
        margin:30px 0px 0px 0px;
        padding:4px;
        width: 100%;
        border:1px solid black;
        background-color:#F0F8FF;
        text-align: left;
    }

    h2.green
    {
        color: #000000;
        background-color:#CCFFCC;
        border-color:#006400;
    }

    h2.red
    {
        color: #000000;
        background-color:#FFCCCC;
    border-color:#8B0000;
}

h3
{
    font-family: Verdana, Helvetica, sans-serif;
    font-weight:bold;
    font-size: 10pt;
    color:#000000;
    background-color: #FFFFFF;
    width: 75%;
    text-align: left;
}

p
{
    font-family: Verdana, Helvetica, sans-serif;
    font-size: 8pt;
    color:#000000;
    background-color: #FFFFFF;
    width: 75%;
    text-align: left;
}

p i
{
    font-family: "Courier New", Courier, mono;
    font-size: 8pt;
    color:#000000;
    background-color: #CCCCCC;
}

ul
{
    font-family: Verdana, Helvetica, sans-serif;
    font-size: 8pt;
    color:#000000;
    background-color: #FFFFFF;
    width: 75%;
    text-align: left;
}

a
{
    font-family: Verdana, Helvetica, sans-serif;
    text-decoration: none;
    font-size: 8pt;
    color:#000000;
    font-weight:bold;
    background-color: #FFFFFF;
    color: #000000;
}

li a
{
    font-family: Verdana, Helvetica, sans-serif;
    text-decoration: none;
    font-size: 10pt;
    color:#000000;
    font-weight:bold;
    background-color: #FFFFFF;
    color: #000000;
}

a:hover
{
      text-decoration: underline;
}

a.red
{
    color:#8B0000;
}
a.green
{
    color:#006400;
}

table
{
    width: 80%;
    border:0px;
    color: #000000;
    background-color: #000000;
    margin:10px;
}

tr
{
    vertical-align:top;
    font-family: Verdana, Helvetica, sans-serif;
    font-size: 8pt;
    color:#000000;
    background-color: #D1D1D1;
}

tr.head
{
    background-color: #E1E1E1;
    color: #000000;
    font-weight:bold;
}

tr.open
{
    background-color: #CCFFCC;
    color: #000000;
}

tr.filtered
{
    background-color: #FFDDBB;
    color: #000000;
}
```

```
    tr.closed
    {
        background-color: #FFAFAF;
        color: #000000;
    }

    td
    {
        padding:2px;
    }

    .status
    {
        display:none;
    }

    #menu li
    {
        display         : inline;
        margin          : 0;
        /*margin-right    : 10px;*/
        padding         : 0;
        list-style-type : none;
    }
}
</style>
      <title>nmap report</title>
</head>

<body>
      <div id="container">
    <h1>nmap scan report - scan @ <xsl:value-of select="$start" />
    </h1>

    <ul id="menu">
      <li><a href="#scansummary">scan summary</a><xsl:text> |
</xsl:text></li>
      <li><a href="#scaninfo">scan info</a><xsl:text> |
</xsl:text></li>

          <xsl:for-each select="host">
              <xsl:sort select="substring ( address/@addr, 1, string-
length ( substring-before ( address/@addr, '.' ) ) )* (256*256*256) +
substring ( substring-after ( address/@addr, '.' ), 1, string-length (
substring-before ( substring-after ( address/@addr, '.' ), '.' ) ) )*
(256*256) + substring ( substring-after ( substring-after (
address/@addr, '.' ), '.' ), 1, string-length ( substring-before (
substring-after ( substring-after ( address/@addr, '.' ), '.' ), '.' )
) ) * 256 + substring ( substring-after ( substring-after ( substring-
after ( address/@addr, '.' ), '.' ), '.' ), 1 )" order="ascending"
data-type="number"/>
              <li>
                <xsl:element name="a">
                    <xsl:attribute name="href">#<xsl:value-of
select="translate(address/@addr, '.', '_') " /></xsl:attribute>
                    <xsl:attribute name="class">
```

```
                  <xsl:choose>
                        <xsl:when test="status/@state =
'up'">green</xsl:when>
                        <xsl:otherwise>red</xsl:otherwise>
                  </xsl:choose>
                    </xsl:attribute>
                    <xsl:value-of select="address/@addr"/>
                    <xsl:if test="count(hostnames/hostname) > 0">
                      <xsl:for-each select="hostnames/hostname">
                        <xsl:sort select="@name" order="ascending"
data-type="text"/>
                        <xsl:text> / </xsl:text><xsl:value-of
select="@name"/>
                      </xsl:for-each>
                    </xsl:if>
                </xsl:element>
               <xsl:text> | </xsl:text></li>
          </xsl:for-each>

        <li><a href="#runstats">runstats</a></li>
    </ul>

      <xsl:element name="a">
            <xsl:attribute name="name">scansummary</xsl:attribute>
      </xsl:element>
    <h2>scan summary</h2>
    <p>
      <xsl:value-of select="@scanner"/> was initiated at <xsl:value-of
select="$start" /> with these arguments:<br/>
    <i><xsl:value-of select="@args" /></i><br/>
    The process stopped at <xsl:value-of select="$end" />.
      <xsl:choose>
        <xsl:when test="debugging/@level = '0'">Debuging was disabled,
</xsl:when>
        <xsl:otherwise>Debugging was enabeld, </xsl:otherwise>
    </xsl:choose>
    the verbosity level was <xsl:value-of select="verbose/@level" />.

    </p>
    <xsl:apply-templates select="host">
        <xsl:sort select="substring ( address/@addr, 1, string-length (
substring-before ( address/@addr, '.' ) ) )* (256*256*256) + substring
( substring-after ( address/@addr, '.' ), 1, string-length ( substring-
before ( substring-after ( address/@addr, '.' ), '.' ) ) )* (256*256) +
substring ( substring-after ( substring-after ( address/@addr, '.' ),
'.' ), 1, string-length ( substring-before ( substring-after (
substring-after ( address/@addr, '.' ), '.' ), '.' ) ) ) * 256 +
substring ( substring-after ( substring-after ( substring-after (
address/@addr, '.' ), '.' ), '.' ), 1 )" order="ascending" data-
type="number"/>
    </xsl:apply-templates>
    <xsl:apply-templates select="runstats"/>
    </div>

</body>
</html>
</xsl:template>
```

```
<!-- ............................................................ -->

<!-- scaninfo -->
<!-- ............................................................ -->
<xsl:template match="scaninfo">
      <xsl:element name="a">
            <xsl:attribute name="name">scaninfo</xsl:attribute>
      </xsl:element>

      <h2>scan info</h2>
      <ul>
        <li><xsl:value-of select="@type" />-scan</li>
        <li><xsl:value-of select="@numservices" /><xsl:text>
</xsl:text><xsl:value-of select="@protocol" /> services scanned</li>
      </ul>
      <xsl:apply-templates/>
</xsl:template>
<!-- ............................................................ -->

<!-- runstats -->
<!-- ............................................................ -->
<xsl:template match="runstats">
      <xsl:element name="a">
            <xsl:attribute name="name">runstats</xsl:attribute>
      </xsl:element>

      <h2>runstats</h2>
      <ul>
            <li><xsl:value-of select="$totaltime" /> sec. scanned</li>
        <li><xsl:value-of select="hosts/@total" /> host(s) scanned</li>
        <li><xsl:value-of select="hosts/@up" /> host(s) online</li>
        <li><xsl:value-of select="hosts/@down" /> host(s) offline</li>
      </ul>
      <ul>
            <li>nmap version: <xsl:value-of select="/nmaprun/@version"
/></li>
        <li>xml output version: <xsl:value-of
select="/nmaprun/@xmloutputversion" /></li>
        <li>nmap.xsl version: <xsl:value-of select="$nmap_xsl_version"
/></li>
      </ul>
      <xsl:apply-templates/>
</xsl:template>
<!-- ............................................................ -->

<!-- host -->
<!-- ............................................................ -->
<xsl:template match="host">
      <xsl:element name="a">
            <xsl:attribute name="name"><xsl:value-of
select="translate(address/@addr, '.', '_') " /></xsl:attribute>
      </xsl:element>

    <xsl:choose>
        <xsl:when test="status/@state = 'up'">
            <h2 class="green"><xsl:value-of select="address/@addr"/>
            <xsl:if test="count(hostnames/hostname) > 0">
```

```
              <xsl:for-each select="hostnames/hostname">
                <xsl:sort select="@name" order="ascending" data-
type="text"/>
                <xsl:text> / </xsl:text><xsl:value-of select="@name"/>
              </xsl:for-each>
            </xsl:if>
            <span class="status">(online)</span>
            </h2>
        </xsl:when>
        <xsl:otherwise>
            <h2 class="red"><xsl:value-of select="address/@addr"/>
            <xsl:if test="count(hostnames/hostname) > 0">
              <xsl:for-each select="hostnames/hostname">
                <xsl:sort select="@name" order="ascending" data-
type="text"/>
                <xsl:text> / </xsl:text><xsl:value-of select="@name"/>
              </xsl:for-each>
            </xsl:if>
            <span class="status">(offline)</span></h2>
        </xsl:otherwise>
    </xsl:choose>

    <xsl:if test="count(address) > 0">
        <h3>address</h3>
        <ul>
            <xsl:for-each select="address">
                <li><xsl:value-of select="@addr"/> (<xsl:value-of
select="@addrtype"/>)</li>
            </xsl:for-each>
        </ul>
    </xsl:if>

      <xsl:apply-templates/>

</xsl:template>
<!-- ............................................................ -->



<!-- hostnames -->
<!-- ............................................................ -->
<xsl:template match="hostnames">
<xsl:if test="hostname/@name != ''"><h3>hostnames</h3><ul>  <xsl:apply-
templates/></ul></xsl:if>
</xsl:template>
<!-- ............................................................ -->

<!-- hostname -->
<!-- ............................................................ -->
<xsl:template match="hostname">
<li><xsl:value-of select="@name"/> (<xsl:value-of
select="@type"/>)</li>
</xsl:template>
<!-- ............................................................ -->

<!-- ports -->
<!-- ............................................................ -->
```

```
<xsl:template match="ports">
<h3>ports</h3>
<xsl:for-each select="extraports">
    <xsl:if test="@count > 0">
          <p>The <xsl:value-of select="@count" /> ports scanned but not
shown below are in state: <b><xsl:value-of select="@state" /></b></p>
    </xsl:if>
</xsl:for-each>

<xsl:if test="count(port) > 0">
    <table cellspacing="1">
    <tr class="head">
        <td colspan="2">Port</td>
        <td>State</td>
        <td>Service</td>
        <td>Product</td>
        <td>Version</td>
        <td>Extra info</td>
    </tr>
      <xsl:apply-templates/>
      </table>
</xsl:if>
</xsl:template>
<!-- ........................................................ -->

<!-- port -->
<!-- ........................................................ -->
<xsl:template match="port">
      <xsl:choose>
            <xsl:when test="state/@state = 'open'">
            <tr class="open">
                <td><xsl:value-of select="@portid" /></td>
                <td><xsl:value-of select="@protocol" /></td>
                <td><xsl:value-of select="state/@state" /></td>
                <td><xsl:value-of select="service/@name"
/><xsl:text>&#xA0;</xsl:text></td>
                <td><xsl:value-of select="service/@product"
/><xsl:text>&#xA0;</xsl:text></td>
                <td><xsl:value-of select="service/@version"
/><xsl:text>&#xA0;</xsl:text></td>
                <td><xsl:value-of select="service/@extrainfo"
/><xsl:text>&#xA0;</xsl:text></td>
            </tr>
            </xsl:when>
            <xsl:when test="state/@state = 'filtered'">
            <tr class="filtered">
                <td><xsl:value-of select="@portid" /></td>
                <td><xsl:value-of select="@protocol" /></td>
                <td><xsl:value-of select="state/@state" /></td>
                <td><xsl:value-of select="service/@name"
/><xsl:text>&#xA0;</xsl:text></td>
                <td><xsl:value-of select="service/@product"
/><xsl:text>&#xA0;</xsl:text></td>
                <td><xsl:value-of select="service/@version"
/><xsl:text>&#xA0;</xsl:text></td>
                <td><xsl:value-of select="service/@extrainfo"
/><xsl:text>&#xA0;</xsl:text></td>
```

```
            </tr>
            </xsl:when>
            <xsl:when test="state/@state = 'closed'">
            <tr class="closed">
                <td><xsl:value-of select="@portid" /></td>
                <td><xsl:value-of select="@protocol" /></td>
                <td><xsl:value-of select="state/@state" /></td>
                <td><xsl:value-of select="service/@name"
/><xsl:text>&#xA0;</xsl:text></td>
                <td><xsl:value-of select="service/@product"
/><xsl:text>&#xA0;</xsl:text></td>
                <td><xsl:value-of select="service/@version"
/><xsl:text>&#xA0;</xsl:text></td>
                <td><xsl:value-of select="service/@extrainfo"
/><xsl:text>&#xA0;</xsl:text></td>
            </tr>
            </xsl:when>
            <xsl:otherwise>
            <tr>
                <td><xsl:value-of select="@portid" /></td>
                <td><xsl:value-of select="@protocol" /></td>
                <td><xsl:value-of select="state/@state" /></td>
                <td><xsl:value-of select="service/@name"
/><xsl:text>&#xA0;</xsl:text></td>
                <td><xsl:value-of select="service/@product"
/><xsl:text>&#xA0;</xsl:text></td>
                <td><xsl:value-of select="service/@version"
/><xsl:text>&#xA0;</xsl:text></td>
                <td><xsl:value-of select="service/@extrainfo"
/><xsl:text>&#xA0;</xsl:text></td>
            </tr>
            </xsl:otherwise>
      </xsl:choose>
</xsl:template>
<!-- ............................................................ -->

<!-- os -->
<!-- ............................................................ -->
<xsl:template match="os">
<xsl:if test="osmatch/@name != ''"><h3>remote operating system
guess</h3></xsl:if>
<ul>
      <xsl:apply-templates/>
</ul>
</xsl:template>
<!-- ............................................................ -->

<!-- os portused -->
<!-- ............................................................ -->
<xsl:template match="portused">
<li>used port <xsl:value-of select="@portid" />/<xsl:value-of
select="@proto" /> (<xsl:value-of select="@state" />)  </li>
</xsl:template>
<!-- ............................................................ -->

<!-- os match -->
<!-- ............................................................ -->
```

```
<xsl:template match="osmatch">
<li>os match: <b><xsl:value-of select="@name" /> </b></li>
<li>accuracy: <xsl:value-of select="@accuracy" />%</li>
<li>reference fingerprint line number: <xsl:value-of select="@line"
/></li>
</xsl:template>
<!-- ............................................................ -->

<!-- uptime -->
<!-- ............................................................ -->
<xsl:template match="uptime">
<xsl:if test="@seconds != ''"><h3>system uptime</h3></xsl:if>
<ul>
<li>uptime: <xsl:value-of select="@seconds" /> sec</li>
<li>last reboot: <xsl:value-of select="@lastboot" /></li>
</ul>
</xsl:template>
<!-- ............................................................ -->

<!-- smurf -->
<!-- ............................................................ -->
<xsl:template match="smurf">
<xsl:if test="@responses != ''"><h3>smurf responses</h3></xsl:if>
<ul>
<li><xsl:value-of select="@responses" /> responses counted</li>
</ul>
</xsl:template>
<!-- ............................................................ -->

<!-- tcpsequence -->
<!-- ............................................................ -->
<xsl:template match="tcpsequence">
<xsl:if test="@values != ''">
    <h3>tcpsequence</h3>
    <ul>
        <li>index: <xsl:value-of select="@index" /></li>
        <li>class: <xsl:value-of select="@class" /></li>
        <li>difficulty: <xsl:value-of select="@difficulty" /></li>
        <li>values: <xsl:value-of select="@values" /></li>
    </ul>
</xsl:if>
</xsl:template>
<!-- ............................................................ -->

<!-- ipidsequence -->
<!-- ............................................................ -->
<xsl:template match="ipidsequence">
<xsl:if test="@values != ''">
    <h3>ipidsequence</h3>
    <ul>
        <li>class: <xsl:value-of select="@class" /></li>
        <li>values: <xsl:value-of select="@values" /></li>
    </ul>
</xsl:if>
</xsl:template>
<!-- ............................................................ -->
```

```
<!-- tcptssequence -->
<!-- ........................................................... -->
<xsl:template match="tcptssequence">
<xsl:if test="@values != ''">
    <h3>tcptssequence</h3>
    <ul>
        <li>class: <xsl:value-of select="@class" /></li>
        <li>values: <xsl:value-of select="@values" /></li>
    </ul>
</xsl:if>
</xsl:template>
<!-- ........................................................... -->

</xsl:stylesheet>
```