Pulse Connect Secure
Pulse Policy Secure

Custom Sign-In Pages Developer Reference

Product Release 8.2/5.3

Document Revision 2.0
Published: 2016-02-25
END USER LICENSE AGREEMENT

The Pulse Secure product that is the subject of this technical documentation consists of (or is intended for use with) Pulse Secure software. Use of such software is subject to the terms and conditions of the End User License Agreement (“EULA”) posted at http://www.pulsesecure.net/support/eula. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.”

Revision History
2014/12/15 – Changes for rebranding
2016/02/25 – Updated Working with the Pulse Secure Desktop Client or Standalone Network Connect
# Abbreviated Table of Contents

- Table of Contents ............................................................................................................................ ii
- List of Tables ....................................................................................................................................... viii
- About This Guide ............................................................................................................................... ix
- PART 1 Custom Sign-in Pages ........................................................................................................... 1
  - CHAPTER 1 Custom sign-in pages .................................................................................................. 2
- PART 2 Index ...................................................................................................................................... 148
- PART 2 Index ...................................................................................................................................... 149
Table of Contents

Table of Contents........................................................................................................ ii
List of Tables.................................................................................................................. viii
About This Guide.......................................................................................................... ix
  Related Documentation.............................................................................................. ix
  Document Conventions................................................................................................ x
  Requesting Technical Support .................................................................................... x
    Self-Help Online Tools and Resources.................................................................... x
    Opening a Case with PSGSC.................................................................................... x
PART 1 Custom Sign-in Pages ......................................................................................... 1
CHAPTER 1 Custom sign-in pages .................................................................................... 2
  Downloadable Zip Files............................................................................................. 10
  Adding Custom Help Files.......................................................................................... 12
  Using the Template Toolkit...................................................................................... 12
    How Connect Secure Uses the Template Toolkit.................................................... 12
    Understanding the Template Toolkit Language...................................................... 15
    Template Comments............................................................................................... 16
    Accessing and Updating Variables and Files........................................................... 16
    Creating Conditional Statements........................................................................... 21
    Creating Looping Constructs.................................................................................... 22
    Unsupported Directives............................................................................................ 23
  Using Templates from the samples.zip File............................................................... 24
    Customizing Look-and-Feel..................................................................................... 25
    Simplifying the Code in the Sample Templates....................................................... 26
    Using Cascading Style Sheets.................................................................................. 30
    Working with the Pulse Secure Desktop Client or Standalone Network Connect..... 38
  Customizing Error Handling and Navigation............................................................ 38
    Redirecting Based on Any and All Errors............................................................... 38
    Redirecting Based on Specific Errors ..................................................................... 39
    Defining Error Message Locations......................................................................... 39
    Customizing Error Messages.................................................................................. 39
  Localizing Custom Sign-In Pages.............................................................................. 40
    Creating a Localized Set of Custom Sign-In Pages............................................... 40
  Upgrade Considerations............................................................................................. 41
  Custom Templates Reference...................................................................................... 41
  Mobile Device Pages.................................................................................................. 43
  iPad Pages.................................................................................................................... 43
  <failedPolicy>........................................................................................................... 43
  <failedPolicy>.name.................................................................................................... 43
  <failedPolicy>.remediation......................................................................................... 44
  action........................................................................................................................... 44
  activex_ini................................................................................................................... 44
  AgentInstall.html...................................................................................................... 45
  AgentlessInstall.html................................................................................................. 45
  AnonymousAuthentication......................................................................................... 46
applet_ini
appstr
authenticate()
autocomplete
Cancel.thtml / Cancel-mobile-webkit.html / Cancel-ipad.html
cboxuser
ccInAcTimeout
ccRunning
cert_md5
CertificateAuthentication
changePasswordTitle
check
checkActiveX()
color
Continue
csec_key
Defender.thtml / Defender-mobile-webkit.html / Defender-ipad.html
delivery_mode
DoUnload()
ds_mobile_common.css
ExceededConcurrent.thtml / ExceededConcurrent-mobile-webkit.html / ExceededConcurrent-ipad.html
expired
fieldsArr
fallback_setup_applet_vitals
FILTER verbatim
FinishLoad()
focus
FriendlyTime
frmCasque
frmLogin
frmNextToken
frmSelectRoles
g_time
gCancelNewPinMode
gCancelNextTokenMode
geckoBrowser
GeneratePin.html / GeneratePin-mobile-webkit.html / GeneratePin-ipad.html
GetCookieValue()
grab()
GraceLoginsRemaining
GraceLoginUsed.html / GraceLoginUsed-mobile-webkit.html / GraceLoginUsed-ipad.html
guest-user-print-page.html
guest-user-print-page-batch.html
guest-user-email-page.html
HC_REMED_POLICIES_CHECK
HC_REMED_SHOW_ADV_PREFS
hcInAcTimeout
hcRunning
heading
help
help_on
HideRemed()
Home
I18N_UAC_DOWNLOAD_AND_INSTALL_64_BIT
i8
IEonWindows .......................................................... 71
ie_winxp ................................................................. 71
install ................................................................. 71
instanceId ............................................................ 72
instructions ......................................................... 72
isLinux ............................................................... 72
isMac ................................................................. 72
isSAMEnabled ....................................................... 73
isUAC ................................................................. 73
keyboard.js .......................................................... 73
labelInstructions ................................................... 74
listFailedPolicies .................................................. 74
loading ................................................................. 75
locale ................................................................. 75
loginmode ........................................................... 75
Login() ..................................................................... 76
LoginMeeting.html ................................................... 76
LoginPage.html / LoginPage-mobile-webkit.html / LoginPage-ipad.html ...................................................... 77
LoginPageErrorArgs ............................................... 78
LoginPageErrorCode ............................................... 78
LoginPageErrorMessage ......................................... 81
login_required ....................................................... 82
Logout.html / Logout-mobile-webkit.html / Logout-ipad.html ............................................................... 82
meeting_ini ........................................................... 84
MeetingAppletInstaller.html ...................................... 84
MeetingRun.html .................................................... 85
MeetingRunJava.html ............................................... 85
meetings ............................................................... 85
MeetingSelect.html ................................................ 86
MeetingTestJava.html .............................................. 86
MeetingTestJS.html ................................................ 87
MeetingTestMSJava.html ......................................... 87
MeetingTestResult.html .......................................... 87
MeetingTrouble.html ............................................... 88
message ............................................................... 88
mid ..................................................................... 89
mid_param ............................................................ 89
midStr ................................................................. 90
MinimumPasswordLength ....................................... 90
msg ................................................................... 90
nc_logging ........................................................... 91
ncp_read_timeout .................................................. 91
netacekey ............................................................. 91
netesecid_key ...................................................... 92
netesecid_loginmode ............................................. 92
netesecid_pinerr ................................................... 92
netesecid_realm .................................................... 93
netesecid_username .............................................. 93
netesecidactionCancel .......................................... 94
netesecidactionEnter ............................................ 94
Netscape4xx ........................................................ 94
NextToken.html / NextToken-mobile-webkit.html / NextToken-ipad.html .................................................. 95
occurrence ........................................................... 96
onsubmit ............................................................. 96
Table of Contents

p .................................................................................................................. 96
PageErrorArgs .......................................................................................... 96
PageErrorCode ......................................................................................... 97
params ....................................................................................................... 98
password ................................................................................................. 98
password .................................................................................................... 99
password2 ................................................................................................. 99
PasswordChange.html / PasswordChange-mobile-webkit.html / PasswordChange-ipad.html .......................................................... 100
PasswordExpiration.html / PasswordExpiration-mobile-webkit.html / PasswordExpiration-ipad.html .............................................. 101
PasswordExpirationFriendlyTime ................................................................. 101
PasswordExpirationSeconds ..................................................................... 102
PasswordHistoryLength .......................................................................... 102
pleasewait ............................................................................................... 103
PleaseWait-mtg.html ................................................................................. 103
PleaseWait.html / PleaseWait-mobile-webkit.html / PleaseWait-ipad.html .................................................................................. 103
pleasewait-ipad.html ................................................................................ 104
pleasewaitLogoutJSCode ......................................................................... 104
PleaseWaitObjectCC ................................................................................ 105
pleaseWaitObjectEP ................................................................................ 106
PleaseWaitObjectHC ................................................................................ 106
pleasewaitObjectJAM .............................................................................. 106
pleasewaitTO_msec .................................................................................. 107
pleasewaitWin32 ....................................................................................... 107
plugintype ............................................................................................... 108
portal ......................................................................................................... 108
postAuthSNDelay .................................................................................... 108
postAuthSNProceed ................................................................................ 109
postAuthSNText ...................................................................................... 109
postAuthSNDelay .................................................................................... 109
preAuthSNDelay ...................................................................................... 109
preAuthSNProceed .................................................................................. 109
preAuthSNText ....................................................................................... 110
preAuthSNTitle ....................................................................................... 110
prompts .................................................................................................... 110
realm ......................................................................................................... 111
realm ......................................................................................................... 111
RealmList ................................................................................................ 112
recallLastRealmUsed() .......................................................................... 112
redirect() ............................................................................................... 113
Remediate.html / Remediate-mobile-webkit.html / Remediate-ipad.html .................................................................................. 113
remedy1 ................................................................................................... 114
remedy2 ................................................................................................... 114
RoleList .................................................................................................... 115
rows ......................................................................................................... 115
runOnLoad ............................................................................................... 115
safari ......................................................................................................... 116
secid_challenge ....................................................................................... 116
secidcontextid ......................................................................................... 117
secid_contextid ....................................................................................... 117
secid_loginmode ..................................................................................... 117
secid_pinerr ............................................................................................ 118
secid_pinformat ....................................................................................... 118
secid_pinselectmode .............................................................................. 118
<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>textContinue</td>
<td>139</td>
</tr>
<tr>
<td>textRemedOption</td>
<td>140</td>
</tr>
<tr>
<td>textTryAgain</td>
<td>140</td>
</tr>
<tr>
<td>thankyou()</td>
<td>141</td>
</tr>
<tr>
<td>troubleshooting</td>
<td>141</td>
</tr>
<tr>
<td>TryAgain</td>
<td>141</td>
</tr>
<tr>
<td>type</td>
<td>142</td>
</tr>
<tr>
<td>tz_offset</td>
<td>142</td>
</tr>
<tr>
<td>uninstall_action</td>
<td>143</td>
</tr>
<tr>
<td>Uninstall_applet()</td>
<td>143</td>
</tr>
<tr>
<td>Uninstall_onclick()</td>
<td>144</td>
</tr>
<tr>
<td>uninstallReDir</td>
<td>144</td>
</tr>
<tr>
<td>UninstallCC</td>
<td>144</td>
</tr>
<tr>
<td>upAndRunning()</td>
<td>145</td>
</tr>
<tr>
<td>UserAdmin-add-user.html</td>
<td>145</td>
</tr>
<tr>
<td>UserAdmin-add-bulk-users.html</td>
<td>145</td>
</tr>
<tr>
<td>UserAdmin-update-user.html</td>
<td>146</td>
</tr>
<tr>
<td>UsersPrintInfo</td>
<td>146</td>
</tr>
<tr>
<td>userAgentTypeAndroid</td>
<td>146</td>
</tr>
<tr>
<td>username</td>
<td>147</td>
</tr>
<tr>
<td>warnings</td>
<td>147</td>
</tr>
<tr>
<td>welcome</td>
<td>147</td>
</tr>
<tr>
<td>win32</td>
<td>148</td>
</tr>
</tbody>
</table>

**PART 2 Index**

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>149</td>
</tr>
<tr>
<td>B</td>
<td>149</td>
</tr>
<tr>
<td>C</td>
<td>149</td>
</tr>
<tr>
<td>D</td>
<td>149</td>
</tr>
<tr>
<td>E</td>
<td>149</td>
</tr>
<tr>
<td>F</td>
<td>149</td>
</tr>
<tr>
<td>G</td>
<td>149</td>
</tr>
<tr>
<td>H</td>
<td>149</td>
</tr>
<tr>
<td>I</td>
<td>149</td>
</tr>
<tr>
<td>J</td>
<td>149</td>
</tr>
<tr>
<td>K</td>
<td>149</td>
</tr>
<tr>
<td>L</td>
<td>149</td>
</tr>
<tr>
<td>M</td>
<td>150</td>
</tr>
<tr>
<td>N</td>
<td>150</td>
</tr>
<tr>
<td>P</td>
<td>150</td>
</tr>
<tr>
<td>R</td>
<td>150</td>
</tr>
<tr>
<td>S</td>
<td>150</td>
</tr>
<tr>
<td>T</td>
<td>150</td>
</tr>
<tr>
<td>U</td>
<td>150</td>
</tr>
<tr>
<td>W</td>
<td>150</td>
</tr>
</tbody>
</table>
List of Tables

Table 1: Notice Icons ................................................................................................................................. ix
Table 2: Required Templates in Each Zip ...................................................................................................... 10
Table 3: English Values of Common Variables ............................................................................................ 17
Table 4: Conditional Operators ................................................................................................................. 21
Table 5: PIN usage indicators .................................................................................................................... 64
Table 6: Login Error Messages, Codes, and Related Notes ........................................................................ 79
Table 7: Additional Login Error Messages and Codes ................................................................................ 83
Table 8: PIN Error Messages ....................................................................................................................... 95
Table 9: Additional PIN Error Messages .................................................................................................... 95
Table 10: Page Error Codes ....................................................................................................................... 97
Table 11: New PIN Assignment Messages ................................................................................................ 132
About This Guide

- Related Documentation
- Document Conventions
- Requesting Technical Support

Related Documentation

- To download a PDF version of the Connect Secure Administration Guide, go to the Connect Secure/SSL VPN Product Documentation page of the Pulse Secure Customer Support Center.

- For information about the changes that Connect Secure clients make to client computers, including installed files and registry changes, and for information about the rights required to install and run Connect Secure clients, refer to the Client-side Changes Guide.

- For information on how to develop Web applications that are compliant with the Connect Secure Content Intermediation Engine, refer to the Content Intermediation Engine Best Practices Guide.

Document Conventions

Table 1 defines notice icons used in this guide.

Table1: Notice Icons

<table>
<thead>
<tr>
<th>Icon</th>
<th>Meaning</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>![i]</td>
<td>Informational note</td>
<td>Indicates important features or instructions.</td>
</tr>
<tr>
<td>![!]</td>
<td>Caution</td>
<td>Indicates a situation that might result in loss of data or hardware damage.</td>
</tr>
<tr>
<td>![^]</td>
<td>Warning</td>
<td>Alerts you to the risk of personal injury or death.</td>
</tr>
<tr>
<td>![^]</td>
<td>Laser warning</td>
<td>Alerts you to the risk of personal injury from a laser.</td>
</tr>
</tbody>
</table>
Requesting Technical Support

Technical product support is available through the Pulse Secure Global Support Center (PSGSC). If you are a customer with an active support contract, or are covered under warranty, and need post-sales technical support, you can access our tools and resources online or open a case with PSGSC.

- **Product warranties**—For product warranty information, visit [http://www.pulsesecure.net/support](http://www.pulsesecure.net/support)

Self-Help Online Tools and Resources

For quick and easy problem resolution, Pulse Secure has designed an online self-service portal called the Pulse Secure Global Support Center (PSGSC) that provides you with the following features:

- Find CSC offerings: [http://www.pulsesecure.net/support](http://www.pulsesecure.net/support)
- Search for known bugs: [http://www.pulsesecure.net/support](http://www.pulsesecure.net/support)
- Find product documentation: [http://www.pulsesecure.net/techpubs/](http://www.pulsesecure.net/techpubs/)
- Find solutions and answer questions using our Knowledge Base: [http://www.pulsesecure.net/support](http://www.pulsesecure.net/support)
- Download the latest versions of software and review release notes: [http://www.pulsesecure.net/support](http://www.pulsesecure.net/support)
- Search technical bulletins for relevant hardware and software notifications: [http://www.pulsesecure.net/support](http://www.pulsesecure.net/support)
- Join and participate in the Community Forum: [http://www.pulsesecure.net/support](http://www.pulsesecure.net/support)
- Open a case online in the CSC Case Management tool: [http://www.pulsesecure.net/support](http://www.pulsesecure.net/support)

To verify service entitlement by product serial number, use our Serial Number Entitlement (SNE) Tool: [http://www.pulsesecure.net/support](http://www.pulsesecure.net/support)

Opening a Case with PSGSC

You can open a case with PSGSC on the Web or by telephone.

- Use the Case Management tool in the CSC at [http://www.pulsesecure.net/support](http://www.pulsesecure.net/support).
- Call 1-844 751 7629 (Toll Free, US).
For international or direct-dial options in countries without toll-free numbers, see http://www.pulsesecure.net/support.
PART 1 Custom Sign-in Pages

- Custom sign-in pages
CHAPTER 1 Custom sign-in pages

The customizable sign-in pages feature enables you to personalize the look-and-feel of the pre- and post-authentication, password management, and Pulse Secure Collaboration pages that Connect Secure presents to administrators and end-users. The tasks you can accomplish include:

- Customizing the presentation style using CSS or table-based design
- Customizing error message text
- Redirecting traffic to non-Connect Secure pages based on error codes
- Localizing custom sign-in pages in supported languages
- Adding custom HTML help

Connect Secure ships with several complete sets of sample custom sign-in pages that you can modify to suit your particular look-and-feel and behavior. You can download these files from Connect Secure. This document describes the files, their uses, and how to modify them. Connect Secure contains these files in .zip file format. You can download the zip files from the Authentication > Signing In > Sign-in pages > Upload custom sign-in pages page.

You can achieve a good understanding of the process of modifying and uploading your custom sign-in pages by reading through this document completely before attempting to perform the customization. When you feel you have a comprehensive understanding of what is involved, you should consider downloading the samples.zip file and modifying one or two pages first, just to test your understanding. There is a strong likelihood that you will want to apply a similar look and feel to all of the pages, which you can accomplish fairly simply, if you follow the directions provided in this document. Other tasks, such as redirecting traffic based on errors may require more skill and time.

Custom sign-in pages can potentially require a large amount of memory and disk space. In order to provide custom sign-in pages per IVS, we recommend you customize the sample custom sign-in pages provided with Connect Secure.

You can find the following topics in this document:

- Downloadable Zip Files
- Adding Custom Help Files
- Using the Template Toolkit
- Using Templates from the samples.zip File
- Customizing Error Handling and Navigation
- Localizing Custom Sign-In Pages
- Upgrade Considerations
- Custom Templates Reference
- Mobile Device Pages
- iPad Pages
- <failedPolicy>
- <failedPolicy>.name
- <failedPolicy>.remediation
- action
- activex_ini
- AgentInstall.thtml
- AgentlessInstall.thtml
- AnonymousAuthentication
- applet_ini
- appstr
- authenticate()
- autocomplete
- cboxuser
- ccInAcTimeout
- ccRunning
- cert_md5
- CertificateAuthentication
- changePasswordTitle
- check
- checkActiveX()
- color
- Continue
- secid_key
- delivery_mode
- DoUnload()
- ds_mobile_common.css
- expired
- fieldsArr
- fallback_setup_applet_vitals
- FILTER verbatim
- FinishLoad()
focus
FriendlyTime
frmCasque
frmLogin
frmNextToken
frmSelectRoles
g_time
gCancelNewPinMode
gCancelNextTokenMode
geckoBrowser
GeneratePin.thtml / GeneratePin-mobile-webkit.thtml / GeneratePin-ipad.thtml
GetCookieValue()
grab()
GraceLoginsRemaining
GraceLoginUsed.thtml / GraceLoginUsed-mobile-webkit.thtml / GraceLoginUsed-ipad.html
guest-user-print-page.html
guest-user-print-page-batch.html
guest-user-email-page.html
HC_REMED_POLICIES_CHECK
HC_REMED_SHOW_ADV_PREFS
hcInAcTimeout
hcRunning
heading
help
help_on
HideRemed()
Home
I18N_UAC_DOWNLOAD_AND_INSTALL_64_BIT
ie
IEonWindows
ie_winxp
install
instanceld
instructions
isLinux
isMac
isSAMEnabled
- isUAC
- keyboard.js
- labelInstructions
- listFailedPolicies
- loading
- locale
- loginmode
- Login()
- LoginMeeting.thtml
- LoginPage.thtml / LoginPage-mobile-webkit.thtml / LoginPage-ipad.thtml
- LoginPageErrorArgs
- LoginPageErrorCode
- LoginPageErrorMessage
- login_required
- Logout.thtml / Logout-mobile-webkit.thtml / Logout-ipad.thtml
- meeting_ini
- MeetingAppletInstaller.thtml
- MeetingRun.thtml
- MeetingRunJava.thtml
- meetings
- MeetingSelect.thtml
- MeetingTestJava.thtml
- MeetingTestJS.thtml
- MeetingTestMSJava.thtml
- MeetingTestResult.thtml
- MeetingTrouble.thtml
- message
- mid
- mid_param
- midStr
- MinimumPasswordLength
- msg
- nc_logging
- ncp_read_timeout
- netacekey
- netesecid_key
- netesecid_loginmode
- netsecid_pinerr
- netsecid_realm
- netsecid_username
- netsecidactionCancel
- netsecidactionEnter
- Netscape4xx
- NextToken.html / NextToken-mobile-webkit.html / NextToken-ipad.html
- occurrence
- onsubmit
- p
- PageErrorArgs
- PageErrorCode
- params
- password
- password
- password
- password2
- PasswordChange.html / PasswordChange-mobile-webkit.html / PasswordChange-ipad.html
- PasswordComplexityMessage
- PasswordExpiration.html / PasswordExpiration-mobile-webkit.html / PasswordExpiration-ipad.html
- PasswordExpirationFriendlyTime
- PasswordExpirationSeconds
- PasswordHistoryLength
- pleasewait
- PleaseWait-mtg.html
- PleaseWait.html / PleaseWait-mobile-webkit.html / PleaseWait-ipad.html
- pleasewait-ipad.html
- pleasewaitLogoutJSCode
- PleaseWaitObjectCC
- pleaseWaitObjectEP
- PleaseWaitObjectHC
- pleasewaitObjectJAM
- pleasewaitTO_msec
- pleasewaitWin32
- plugintype
• portal
• postAuthSNDecline
• postAuthSNProceed
• postAuthSNText
• postAuthSNTitle
• preAuthSNDecline
• preAuthSNProceed
• preAuthSNText
• preAuthSNTitle
• prompts
• realm
• realm
• RealmList
• recallLastRealmUsed()
• redirect()
• Remediate.thtml / Remediate-mobile-webkit.thtml / Remediate-ipad.thtml
• remedy1
• remedy2
• RoleList
• rows
• runOnLoad
• safari
• secid_challenge
• secidcontextid
• secid_contextid
• secid_loginmode
• secid_pinerr
• secid_pinformat
• secid_pinselectmode
• secid_systempin
• secid_username
• secidactionSavePin
• secidpinformat
• secidpinselectmode
• SecondaryLoginPage.thtml / SecondaryLoginPage-mobile-webkit.thtml / SecondaryLoginPage-ipad.thtml
• SelectRole.thtml / SelectRole-mobile-webkit.thtml / SelectRole-ipad.thtml
- serverdetails
- setFailed()
- setFinished
- SetLastRealm()
- setStarted()
- setSucceeded()
- setthankyou()
- setup_classid
- setup_codebase
- setup_ini
- Setupappversion
- setup_codebase
- showButtons
- showChangePasswordTitle
- showClose
- showContinue
- showHeading
- showLoading
- showReinstall
- showRemedOption
- showPolicies
- ShowSystemPin.thtml / ShowSystemPin-mobile-webkit.thtml / ShowSystemPin-ipad.thtml
- showTryAgain
- signinAgain
- signinAgain
- signinAgainUrl
- SigninNotifyPostAuth.thtml / SigninNotifyPreAuth-mobile-webkit.thtml / SigninNotifyPostAuth-ipad.thtml
- SigninNotifyPreAuth.thtml / SigninNotifyPreAuth-ipad.thtml
- SM-NewUserPin.thtml
- SM-NextToken.thtml / SM-NextToken-mobile-webkit.thtml / SM-NextToken-ipad.thtml
- softid_error
- softid_time
- SSL.thtml / SSL-mobile-webkit.thtml / SSL-ipad.thtml
- Start_onclick()
- Start_onclick_java()
- start_status
- StartCC()
- StartHConfig()
- started
- startPageLink
- startSessionReDir
- stopComponents()
- SubmitClicked()
- submitFrmCasque()
- testjava.ini
- textClose
- textContinue
- textRemedOption
- textTryAgain
- thankyou()
- troubleshooting
- TryAgain
- type
- tz_offset
- uninstall_action
- Uninstall_applet()
- Uninstall_onclick()
- uninstallReDir
- UninstallCC
- upAndRunning()
- UserAdmin-add-user.html
- UserAdmin-add-bulk-users.html
- UserAdmin-update-user.html
- UsersPrintInfo
- userAgentTypeAndroid
- username
- warnings
- welcome
- win32
Downloadable Zip Files

Although each zip file contains all the templates, each zip file is for a particular set of features:

- **Sample.zip**—This zip file is for standard Connect Secure pages, including standard Connect Secure pre- and post-authentication pages, ACE pre-authentication pages, ACE pre-authentication pages for use with eTrust SiteMinder, and password management pages.

- **SoftID.zip**—This zip file is for use with the RSA Soft ID client.

- **Kiosk.zip**—This zip file is for use by kiosk users or for any system in which you want to lock out keyboard-based login.

- **Meeting.zip**—This zip file is for pre-authentication and post-authentication pages for use by Pulse Secure Collaboration attendees.

NOTE: When applicable, the zip files contain templates for mobile devices, including templates for mobile webkit and specific mobile devices, such as Windows Mobile and Apple iPad. These templates are identical to the other templates contained in the zip file, except that they are customized for a smaller viewing area. You are not required to add any of them to the zip file. The device templates use the same names as their full screen counterparts, except that mobile-webkit, ppc, and ipad are added to their filenames. For example, the full screen sign-in page template is named LoginPage.html and its Mobile counterpart is named LoginPage-ppc.html.

Certain templates are required when uploading the zip files. If you do not include these required templates, a warning message appears. “Custom sign-in pages” lists the required templates for each zip file.

Table 2: Required Templates in Each Zip

<table>
<thead>
<tr>
<th></th>
<th>Samples.zip</th>
<th>Kiosk.zip</th>
<th>SoftID.zip (ACE)</th>
<th>Meeting.zip</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cancel.html</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>confirmation.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>confirmation_opensessions.html</td>
<td>R</td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>Defender.html</td>
<td>R</td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>ds.css</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>ExceededConcurrent.html</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>GeneratePin.html</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>GraceLoginUsed.html</td>
<td>R</td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>keyboard.js</td>
<td></td>
<td>R</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>LoginMeeting.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>LoginPage.html</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
</tr>
<tr>
<td>File Name</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td></td>
</tr>
<tr>
<td>Logout.html</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MeetingAppletInstaller.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>MeetingRun.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>MeetingRunJava.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>MeetingSelect.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>MeetingTestJava.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>MeetingTestUS.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>MeetingTestMSJava.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>MeetingTestResult.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>MeetingTrouble.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>NewPin.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>NextToken.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>PasswordChange.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>PasswordExpiration.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>PleaseWait.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>PleaseWait-mtg.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>Remediate.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>SecondaryLoginPage.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>SelectRole.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>ShowSystemPin.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>SigninNotifPostAuth.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>SigninNotifPreAuth.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>SM-NewPinSelect.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>SM-NewPinSystem.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>SM-NewUserPin.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>SM-NextToken.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>SSL.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>UserAdmin-add-user.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>UserAdmin-add-bulk-users.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>UserAdmin-update-user.html</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
</tbody>
</table>

To customize Connect Secure pages, you must use the Template Toolkit language. For more information, see "Understanding the Template Toolkit Language".

For information on how to internationalize custom sign-in pages, see "Localizing Custom Sign-In Pages".
Adding Custom Help Files

You can add custom help files by creating your own HTML help files and linking them to the custom sign-in pages. You can easily add links to the basic HTML in the custom sign-in pages.

If you add customized online Help to your custom sign-in pages, you must set the destination URL to point to a .html file, not a .thtml file. For example, in the following input statement, the help file, help.html, should not have a .thtml extension:

```html
<input type='submit' name='help' value="Help"
onclick='window.open("% Home%/imgs/help.html",
"wndHelp","height=400,width=500,resizeable=yes,scrollbars=yes");
return false;'/>
```

If you use the .thtml extension, the Help link on the custom sign-in page returns an error.

Using the Template Toolkit

The Template Toolkit is an open source software package written in Perl, that enables you to build and replicate Web pages using a special-purpose language. The Template Toolkit language is simple and easy to learn.

How Connect Secure Uses the Template Toolkit

Connect Secure system software recognizes the Template Toolkit directives and is pre-configured to process templates. Connect Secure processes templates into .html files, which it then displays as administrator Web console pages or as end-user Web pages. You cannot change all of the generated Web pages, but you can customize the ones noted in this topic.

You can also modify a variety of presentation items on the UI Options page in the administrative Web console.

**NOTE:** The html files and many .css and .js files in the .thtml files include `<%DANA_VERSION_HASH%>` in the filename. This placeholder is replaced by a hash value during runtime to verify the files have not been modified.

The Template Toolkit processing utilities transform template source files into complete HTML files. The source files are a combination of HTML and Template Toolkit directives. The source files carry a user-defined extension, but Connect Secure recognizes the file extension of .thtml. Processed files typically carry a .html file extension.

**NOTE:** When creating customized sign-in pages, save them as UTF-i.

Testing Your Custom Pages

There are two ways to test your custom pages:

- Using the Template Toolkit utilities you have installed locally.
- Uploading customized files to Connect Secure.
If you are customizing just one or two of the sample template files, you might find it easiest to upload the files to Connect Secure. If you do not have access to a test system, or if you are customizing many of the sample template files or making significant changes to those files, you may want to install and configure the Template Toolkit on your local machine.

**Setting up the Template Toolkit Locally**

You can install the Template Toolkit on your local system. Because it is a Perl application, you need a Perl distribution on your system before you install the Template Toolkit. If you run a Linux or Mac OS X system, you probably already have a Perl distribution on your system. If you run a Windows system, you may need to install the ActiveState ActivePerl distribution, which you can find at [http://www.activestate.com/](http://www.activestate.com/).

If you prefer, you can install Cygwin, which allows you to run UNIX programs in a virtual environment on your Windows system. You can download Cygwin from [http://sourceware.org/cygwin/](http://sourceware.org/cygwin/).

It is likely that you will need to download the latest version of the Perl AppConfig module, which allows your Perl applications to read and process configuration files. You can find the latest version of AppConfig at [http://www.cpan.org/modules/by-module/AppConfig/](http://www.cpan.org/modules/by-module/AppConfig/).

You can find a gzip-tar archive of the latest version of the Template Toolkit at [http://www.cpan.org/modules/by-module/Template/](http://www.cpan.org/modules/by-module/Template/).

Follow these steps:

1. Download, install, and configure Perl, ActivePerl, or Cygwin, depending on the type of operating system you are running, and your personal requirements.

2. Download, install, and configure AppConfig.

3. Download, install, and configure the Template Toolkit.

**NOTE:** The Template Toolkit distribution includes a number of useful documents. You should print the INSTALL document and read it closely. You must perform a number of configuration steps as detailed in the INSTALL document. If you try to perform the steps as described in INSTALL, but have not performed the requisite configuration, the installation fails. Fortunately, you can go back and perform the configuration steps you missed or performed incorrectly, and try to install again, until the installation succeeds.

After you have successfully installed and configured the Template Toolkit, you can generate HTML files from the sample templates that are available on Connect Secure. Connect Secure is pre-configured to process Template Toolkit files and many of the Connect Secure components are not available on your local system. Therefore, you are unable to run the Template Toolkit in exactly the same way or to obtain the same exact results. However, you can simulate the processing closely enough to test the presentation. You will encounter the following limitations:

- You can only test the presentation of the sample templates on your local system. You cannot test the actual pre-authentication logic unless you upload the files to a Connect Secure system.
- You must do one of the following:
Use the Template Toolkit [% TAGS %] directive to change any new directives so that they use the angle bracket tokens (<% %>) used in the Pulse Secure sample templates.

Use the standard Template Toolkit tokens ( [% ] ) and when satisfied with the presentation, modify them to angle bracket tokens (<% %>) before uploading to Connect Secure.

The Template Toolkit provides two command-line utilities for processing template files: `tpage`, which enables you to process a single template at a time, and `ttree`, which enables you to process an entire directory of template files. Although you can use the `ttree` utility, Connect Secure uses a proprietary process that simulates the `tpage` utility, and so, you may find your final results more effective by using `tpage` only.

**Downloading Sample Templates and Uploading Files to Connect Secure**

Your other option for testing customized template files is to upload the files to Connect Secure. Although this may be a bit tedious, if you are only customizing a handful of pages, you might find it easier than setting up the Template Toolkit environment on your local system.

**NOTE:** The total combined size of all uploaded customizable zip files cannot exceed 12MB.

To make sure you can view the customized pages, you must perform these procedures.

To download the sample templates:

1. Sign in to Connect Secure as an administrator.
2. Select Authentication > Signing In > Sign-in Pages.
3. Click Upload Custom Pages.
4. Download one or more of the sample template files.
5. Unzip the files on your local system.

To customize and upload the new templates:

1. Customize the files, as you like.
2. Add the files to a new zip file.

**NOTE:** When creating your zip file, zip only the template files. Do not include the folder that contains the templates in the zip file.

3. On the Upload Custom Sign-In Pages page, enter a name for the template files. This is a label that is used to identify the pages when you associate them with a specific URL.
4. Select the page type.
5. Click Browse to locate the new zip file containing your customized files.
6. Click Upload Custom Pages.

If you want your end-users to be able to see the new templates, you need to associate the templates, using the label you assigned to them, with a URL that you specify for a given role.

To associate the templates with a role-specific URL:
1. Select Authentication > Signing In > Sign-in Policies.

2. Click New URL.

3. Configure the new policy and URL, using the sign-in policies instructions in the Pulse Policy Secure Administration Guide. Select the label you assigned to the templates from the Sign-in page drop down menu.

4. Click Save Changes.

To test your customized pages iteratively, you need to cycle through the preceding procedures to upload and associate a URL. You only need to download the templates once.

Understanding the Template Toolkit Language

This section includes an abbreviated description of the Template Toolkit language for Connect Secure users. The section describes the most common directives and operators that designers may want to use when creating a customized page for Connect Secure. Although the Template Toolkit supports many powerful constructs, such as simple and complex datatypes, virtual methods, and interfaces to database systems, the custom sign-in pages use only a limited set of the Toolkit capabilities.

You can use the Template Toolkit for many applications that are far beyond the scope of this discussion. For complete information about the Template Toolkit, go to http://www.template-toolkit.org.

A Web page created with the Template Toolkit resembles a standard Web page. It can include HTML, XML, and JavaScript. Unlike a standard Web page, however, it can also include Template Toolkit directives, which you can use to add dynamic behavior to your pages.

A directive is a simple instruction that tells the template processor to perform an action and substitute the result into the document in place of the original directive. You can use directives for many purposes, such as iterating through a list of values (FOREACH), creating conditional statements (IF/UNLESS/ELSE), or including and processing another template file (PROCESS/INCLUDE).

When using directives in your code, note that:

- Directives are case-sensitive and all UPPERCASE.
- You must insert directives within the markup tags '<%' and '%>'.

**NOTE:** The Template Toolkit documentation includes examples of using square bracket tags ([% %]) to mark directives. Connect Secure only supports angle bracket markup tags (<% %>).

- You can embed directives anywhere in a line of text.
- You can split directives across several lines.
- You can use the # character to indicate comments within a directive. The Template Toolkit language ignores characters after the # character. For more information on different types of comments, see “Template Comments”.

© 2016 by Pulse Secure, LLC. All rights reserved
The Template Toolkit language generally ignores insignificant white space within the directive.

In addition to directives, you can also use the Template Toolkit language to include loops, conditionals, variable substitutions, and other template files in your page.

The following sections describe common Template Toolkit tasks, directives, and operations you may want to use in your customized pages:

- “Accessing and Updating Variables and Files”
  - “GET Directive”
  - “SET Directive”
- “Creating Conditional Statements”
  - “Conditional Operators”
  - “IF and ELSIF Directives”
  - “SWITCH and CASE Directives”
- “Creating Looping Constructs”
- “Unsupported Directives”

For information about additional directives that you may use within the Template, Toolkit language, such as PROCESS, INCLUDE, and INSERT see the Template Toolkit documentation, available at http://www.template-toolkit.org.

Template Comments

You can include template comments by adding a # sign to the directive tag, as follows:

```
<%# This is a comment
   that comments out all
   of the lines that follow.%
```

You can include a single-line comment by leaving a space between the tag and the # sign, as follows:

```
<% # This is the first line comment, but the next line is not commented.
   IF foo...
   # This is another single-line comment.%
```

Accessing and Updating Variables and Files

GET Directive

The GET directive retrieves and outputs the value of the named variable.
The `GET` keyword is optional. A variable can be specified in a directive tag by itself.

```
<% GET foo %>
```

```
<% foo %>
```

**SET Directive**

The `SET` directive allows you to assign a value to existing variables or create new temporary variables.

```
<% SET title = 'Hello World' %>
```

The `SET` keyword is optional.

```
<% title = 'Hello World' %>
```

**getText Directive**

Used for multi-lingual support, the `getText` directive retrieves and outputs the localized value of the named variable.

```
<% getText foo %>
```

**Table 3** lists some of common variables used in the templates and their English values.

**Table 3: English Values of Common Variables**

<table>
<thead>
<tr>
<th>Variable</th>
<th>English Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>I18N_ACE_CONFIRM_PIN_COLON</td>
<td>Confirm PIN:</td>
</tr>
<tr>
<td>I18N_ACE_GEN_NEW_PIN</td>
<td>Generate New PIN?</td>
</tr>
<tr>
<td>I18N_ACE_GEN_PIN</td>
<td>Generate PIN</td>
</tr>
<tr>
<td>I18N_ACE_GEN_PIN_MESSAGE1</td>
<td>The system will now generate a PIN for you. Make sure that no one else can see your screen and then click Generate PIN to continue.</td>
</tr>
<tr>
<td>I18N_ACE_INVALID_PIN_FORMAT</td>
<td>Invalid PIN Format</td>
</tr>
</tbody>
</table>
I18N_ACE_INVALID_PIN_LENGTH Invalid PIN Length

I18N_ACE_NEW_PIN_COLON New PIN:

I18N_ACE_NEW_PIN_GENERATED New PIN Generated

I18N_ACE_NEW_PIN_GENERATED_MESSAGE1 Your new PIN is

I18N_ACE_NEW_PIN_GENERATED_MESSAGE2 Be sure to remember it, because you need your PIN each time you sign in. When you have memorized it, click Continue to return to the sign-in page.

I18N_ACE_NEW_PIN_MESSAGE0 You must create a new Personal Identification Number (PIN) before you can sign in.

I18N_ACE_NEW_PIN_MESSAGE1 Your PIN should be

I18N_ACE_NEW_PIN_MESSAGE2 long.

I18N_ACE_NEW_PIN_MESSAGE3 Be sure to remember your PIN, because you need it to sign in.

I18N_ACE_NEW_PIN_MESSAGE4 If you prefer, the system can generate a PIN

I18N_ACE_NEW_PIN_MESSAGE5 for you. Generated PINs are typically more secure.

I18N_ACE_NEW_PIN_MESSAGE6 If you decide not to create a new PIN now, click Cancel.

I18N_ACE_NEW_PIN_REQUIRED New PIN Required

I18N_ACE_PIN_MISMATCH The Two PINs Entered Do Not Match

I18N_ACE_SAVE_PIN Save PIN

I18N_ACE_NEXT_TOKEN_MESSAGE1 Please enter an additional token code to continue.

I18N_ACE_NEXT_TOKEN_MESSAGE2 The server requires that you enter an additional token code to verify that your credentials are valid. To continue, wait for the token code to change and then enter the new code in the SecurID Token Code field.

I18N_ACE_NEXT_TOKEN_REQUIRED Token Resync Required

I18N_ACE_PIN_MISMATCH The Two PINs Entered Do Not Match

I18N_ACE_SAVE_PIN Save PIN
You must create a new Personal Identification Number (PIN) before you can sign in.

Wait for the token code to change, and then enter your current PIN* followed by the new token code in the SecurID Pass Code field. (If you want the system to generate a new PIN for you, select the System PIN checkbox.) Click Enter to advance to the new PIN assignment page.<br><br>*If your administrator tells you that you no longer have a PIN, enter only the new token code in the Pass Code field.

The system will now generate a Personal Identification Number (PIN) for you.

Wait for the token code to change, and then enter your current PIN* followed by the new token code in the SecurID Pass Code field. Make sure that no one else can see your screen and then click Generate PIN to continue.<br><br>*If your administrator tells you that you no longer have a PIN, enter only the new token code in the Pass Code field.

Create your new Personal Identification Number (PIN)

Wait for the token code to change, and then enter your current PIN* followed by the new token code in the SecurID Pass Code field. Then, enter your new PIN.<br><br>*If your administrator tells you that you no longer have a PIN, enter only the new token code in the Pass Code field.

Please enter additional token codes to continue.
The server requires that you enter additional token codes to verify that your credentials are valid. To continue, wait for the token code to change, and then enter your PIN followed by the new token code in the SecurID Pass Code field. Then, wait for the token code to change again and enter only the token code in the SecurID Token Code field.

Token Resync Required

SecurID Pass Code:

System PIN

SecurID Token Code:

New PIN entered is invalid. Please wait for the token to change and try login again.

to continue

Username:

Name of local authentication server.

Full name

Password

e-mail ID

Account start time

Account end time

Time zone

If you want to customize a page to add your own message, edit the appropriate template file and enter your custom message using HTML tags. Make sure you do not change any of the surrounding logic. You cannot edit the strings file(s) and change the value of the `I18N_*` variables. For example, suppose the template file has the following code:

```html
<TABLE border="0" cellspacing="0" cellpadding="0">
  <TR>
    <TD><IMG border='0' src='/dana-na/imgs/loginfo.gif'
```
To change the displayed message to “You must enter a new PIN number” instead of the default message “New PIN Required”, change the code as follows:

```html
<TABLE border="0" cellspacing="0" cellpadding="0">
  <TR>
    <TD><IMG border='0' src='/dana-na/imgs/loginfo.gif' alt='Info' width='21' height='20'></TD>
    <TD><B><% GetText('I18N_ACE_NEW_PIN_GENERATED') %></B></TD>
    <TD>You must enter a new PIN number</TD>
  </TR>
</TABLE>
```

Creating Conditional Statements

You can create conditional statements in the Template Toolkit language. Review the sample templates to find some example conditional statements. One of the simplest and most effective uses of conditional statements is in trapping error conditions.

For example, the following code snippet from the `LoginPage.thtml` template verifies the current user’s account status:

```template
<% IF AccountDisabled %>
  <% LoginPageErrorMessage %>
<% ELSE %>
  <% IF LoginPageErrorMessage %>
    <% LoginPageErrorMessage %>
  <% END %>
<% END %>
```

Conditional Operators

You can use the following conditional operators:

Table 4: Conditional Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>==</code></td>
<td>is equal to</td>
</tr>
<tr>
<td><code>!=</code></td>
<td>is not equal to</td>
</tr>
<tr>
<td><code>&lt;</code></td>
<td>is less than</td>
</tr>
<tr>
<td><code>&lt;=</code></td>
<td>is equal to or less than</td>
</tr>
<tr>
<td><code>&gt;</code></td>
<td>is greater than</td>
</tr>
</tbody>
</table>
Above are tables with symbols and their meanings:

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt;=</td>
<td>is equal to or greater than</td>
</tr>
<tr>
<td>&amp;&amp;</td>
<td>and</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>!</td>
<td>not</td>
</tr>
<tr>
<td>and</td>
<td>and</td>
</tr>
<tr>
<td>or</td>
<td>or</td>
</tr>
<tr>
<td>not</td>
<td>not</td>
</tr>
</tbody>
</table>

**IF and ELSIF Directives**

You can use the `IF` and `ELSIF` directives to construct conditional behavior. Note that these are block directives, which means that you must use the `END` directive to indicate where each block ends. You may nest blocks indefinitely, provided you include an end statement for each nested block.

```html
<% IF LoginPageErrorCode == 1002 %>
    <b> Your username or password is incorrect. Please reenter your credentials. </b>
<%ELSIF LoginPageErrorCode == 1006 %>
    <b> The system is undergoing maintenance. Only administrators are permitted to sign in at this time. </b>
<% END %>
```

**SWITCH and CASE Directives**

You may use the `SWITCH` and `CASE` directives to construct a multi-way conditional test. Note that these are block directives, which means that you must use the `END` directive to indicate where each block ends. You may nest blocks indefinitely, provided you include an end statement for each nested block.

```html
<% SWITCH loginPageErrorCode %>
<% CASE 1001 %>
    <b> Your session time has expired. </b>
<% CASE 1006 %>
    <b> The system is undergoing maintenance. Only administrators are permitted to sign in at this time. </b>
<% CASE %> # default ...
<% END %>
```

**Creating Looping Constructs**

You may use directives such as `FOREACH` and `WHILE` to loop through blocks of code. Note that these are block directives, which means that you must use the `END` directive to indicate where each block ends. You may nest blocks indefinitely, provided you include an end statement for each nested block.
For example, the following sign-in page code loops through all of the authentication realms and displays them in a select list. The sample also uses the `RealmList` and `realm` predefined Connect Secure values.

```html
<select size="1" name="realm">
  <% FOREACH r = RealmList %>
  <option value="<% r %>"><% r %></option>
  <% END %>
</select>
```

## Unsupported Directives

We do not support the `USE`, `INTERPOLATE`, `TAGS`, `PERL`, or `RAWPERL` directives when creating Connect Secure custom pages. Additionally, we do not recommend using the `CALL` or `FILTER` directives.

## Getting Familiar with the Template Page Structure

The sample templates were created using the common method of designing with HTML tables. One of the easiest ways to see how the layout works is to set all of the table borders to a visible state. For example, change `<TABLE border="0">` to `<TABLE border="1">` wherever you find it in the `LoginPage.thtml` template.

The table layout affords you a columnar layout. If you choose to maintain your customized design within the table layout, you may need to experiment with additions or deletions of table rows or cells. You can also change colors of cells, modify the locations of required fields, messages, or UI components.

You must always maintain the Template Toolkit directives that are included in each of the templates. These directives provide required information to Connect Secure and enable Connect Secure to communicate system state back to the end-user, appropriately. Deleting or modifying any of the provided directives can result in unexpected behavior or data corruption.

### CGI Files

Connect Secure generates a number of CGI files, which are called by the custom sign-in pages. Each file performs specific functions that are required by Connect Secure. You should not modify any of these calls nor should you attempt to create your own CGI files to take the place of the supplied CGIs. In particular, you may see a number of references in the custom sign-in pages to the three following CGIs:

- `login.cgi`—Handles authentication request between custom pages, Connect Secure, and any defined authentication servers.
- `welcome.cgi`—Handles redirection following a secondary authentication request, a password change, or new PIN generation.
- `starter.cgi`—Verifies that client applications are running and allows the user to attempt another login, typically when too many users are concurrently signed in to Connect Secure.

Typically, in standard sign-in pages and sample custom pages the order of cgi usage is `welcome.cgi`, `login.cgi`, and then `welcome.cgi` again. For Host Checker pre-authorization,
policies are evaluated in the first invocation of `welcome.cgi` and are passed to `login.cgi`. `login.cgi` takes the username and password parameters and determines the roles to assign. For Host Checker post-authorization, policies are evaluated only in the second invocation of `welcome.cgi`.

**NOTE:** When Host Checker and Cache Cleaner are configured, using `login.cgi` as the first URL to log in to Connect Secure is not supported.

### Using Templates from the `samples.zip` File

The `samples.zip` file contains the following templates which you can customize for use in your own environment. `Samples.zip` contains the most general set of templates. Some of the templates in `samples.zip` are also contained in `kiosk.zip` and `softid.zip`. All files in `meeting.zip` are specific to the Connect Secure Meeting product.

- **The environment. Connect Secure pre-authentication pages**
  - “Logout.html / Logout-mobile-webkit.html / Logout-ipad.html”
  - “SSL.html / SSL-mobile-webkit.html / SSL-ipad.html”
  - “PleaseWait.html / PleaseWait-mobile-webkit.html / PleaseWait-ipad.html”
  - “SelectRole.html / SelectRole-mobile-webkit.html / SelectRole-ipad.html”

- **ACE pre-authentication pages**
  - “NextToken.html / NextToken-mobile-webkit.html / NextToken-ipad.html”
  - “GeneratePin.html / GeneratePin-mobile-webkit.html / GeneratePin-ipad.html”
  - “ShowSystemPin.html / ShowSystemPin-mobile-webkit.html / ShowSystemPin-ipad.html”

- **ACE with eTrust pre-authentication pages**
  - “SM-NewUserPin.html”

- **Guest User Account Manager pages**
  - `UserAdmin-add-user.html`
  - `UserAdmin-add-bulk-users.html`
  - `UserAdmin-update-user.html`
Customizing Look-and-Feel

You can customize the look-and-feel of your custom sign-in pages. This enables you to add your own corporate design, including logos, colors, and layout.

**Required Components**

You must include:

- Any template directives you find in the template files. This includes any code surrounded with template tags `<% %>`.
- Required HTML tags, such as the `<html>`, `<head>`, and `<body>` tags, although these may be consolidated into a separate template file which you can include using either the PROCESS or INCLUDE directives.
- Any `<form>` definitions.
- Any `<input>` statements.
- Any `<script>` statements and included scripts, typically JavaScript.
- Any `<object>` statements.
- Any `<embed>` statements.
- Any `onload`, `onUnload`, or other event triggers in the `<body>` tag.
- Any of the required variables, directives, or other items as noted in the commented section at the beginning of each template file.

You can exclude:

- Any HTML presentation code that is not a template directive, such as table tags, images, or style tags.

**Displaying Custom Logos**

You can change the templates to display your company’s logo by following these steps:

1. Open the template in a text editor or HTML editor.
2. Search for either of the two strings:
3. Replace the string with the following:

```
src="<% Home %>/imgs/custom-logo.gif"
```

where *custom-logo.gif* is the new logo to be displayed. *custom-logo.gif* must be uploaded to Connect Secure as part of your zip file. For more information, see “Downloading Sample Templates and Uploading Files to Connect Secure”.

**NOTE:** For Remediate.thtml, use `<%signinId%>` instead. For example:

```
src="<%signinId%>/imgs/custom-logo.gif"
```

### Simplifying the Code in the Sample Templates

One of the first things you can do is simplify the code in the templates, by using the Template Toolkit PROCESS directive to extract common code into an include file. By moving common code to separate files, you can eliminate a certain amount of redundant code, thereby decreasing the size of the template files, and improving the readability of the files.

Let’s take a look at one of the template files, *LoginPage.thtml*. Almost every designer who wants to customize the custom sign-in pages will likely want to customize *LoginPage.thtml*, which is the first screen that end-users see. If you have not downloaded the *Samples.zip* file, do so now, following the instructions in “Downloading Sample Templates and Uploading Files to Connect Secure”.

Open the *LoginPage.thtml* file in a text editor or HTML editor.

#### Version Directive

Each template file starts with a version directive, similar to the following:

```
<%# NetScreen Page Version 1001 %>
```

Do not modify this directive.

#### Comment Section

Each template file then contains a set of comments that describe template variables, JavaScript, and HTML code required in that particular template. These comments can help you understand what directives and other elements of the file you must maintain, and give hints about other directives, variables, and methods that you can use to further customize the template.

**NOTE:** When you have completed your customization, you can consider removing the comment section from your files. The removal of this section decreases the size of the template files, and can possibly speed up page download times.
Header Tags

With the exception of the Remediate.html file, all of the template files contain the following HTML tags:

```html
<html>
<head>
<meta http-equiv="Content-Language">
<meta http-equiv="Content-Type" content="text/html">
<meta name="robots" content="none">
```

You can extract these tags into a separate .thtml file. If you ever need to change any of these tags, for example, to modify the meta tag definitions, you can then modify one file and the changes are propagated throughout all the files.

To create a simple header template file:

1. Open a new text file in your favorite text editor or HTML editor.
2. Cut and paste the preceding header lines into the file. You can cut these from one of the template files.
3. Save the file into the directory where the other template files are saved, naming the file customhead.html.

   **NOTE:** Do not name the file header.html. Along with config.html and footer.html, this name may already be used by the Connect Secure system software. You can call the files whatever you want, as long as they do not use these reserved names. They must end in the .html file extension.

4. Open each template file and delete the header lines.
5. In the place of the header lines, insert the following directive:

   ```html
   <% PROCESS customhead.html %>
   ```

   The command PROCESS must be in all UPPER CASE. If you name the header file something other than customhead.html, then use that name in the directive instead.
6. When you zip your template files up, make sure you include the customhead.html file.

Copyright and Footer Tags

You might want to perform a similar task with the footer tags. This topic shows how to create a copyright notice in a separate include file.

1. Open a new text file in your favorite text editor or HTML editor.
2. Copy the following directive into the file:

   ```html
   <% year = 2005;
   company = "Pulse Secure";
   ```
copyr = "Copyright $year $company" -%>

You can change the company name to your own company name.

3. Save the file into the directory where the other template files are saved, naming the file customconfig.thtml.

4. Open another new text file in your favorite text editor or HTML editor.

5. Copy the following directive and HTML to the file:

```html
<div id="footer">
  &copy; <% copyr -%>
</div>
</body>
</html>
```

6. Save the file into the directory where the other template files are saved, naming the file customfoot.thtml.

7. In your LoginPage.thtml file, add the following directive just prior to the PROCESS directive for customhead.thtml. The lines should look like:

```html
<% PROCESS customconfig.thtml %>
<% PROCESS customhead.thtml %>
```

8. In your LoginPage.thtml file, scroll to the end of the file.

9. Replace the last two lines that contain the closing </body> and closing </html> tags with the following directive:

```html
<% PROCESS customfoot.thtml %>
```

10. Save the LoginPage.thtml file.

**Page Banner**

You can also create a common banner for your pages, eliminating the repetitious use of images and HTML code in every file. The following sample code creates a banner using an H1 header for the text displayed in the banner. Note the commented image tag, which you could use instead of the H1, if you prefer to use an image, for example, a logo.

1. Open a new text file in your favorite text editor or HTML editor.

2. Copy the following HTML into the file:

```html
<div id="banner">
<h1 class="banner">Sample</h1>
<!-- <img border="0" src="/imgs/samples.gif" alt="Logo"> -->
</div>
```
3. Save the file into the directory where the other template files are saved, naming the file custombanner.thtml.

4. In your LoginPage.thtml file, add the following directive just after the <body> tag.

   <body id="body" onload="FinishLoad()">  
   <% PROCESS custombanner.thtml %>

5. Save the LoginPage.thtml file.

**Processing LoginPage.thtml**

Now you can process the template to see the results of making the changes. Keep in mind the following requirements:

- You must have successfully installed and configured the Template Toolkit on your local system.
- All of the template files, including the custom header, custom footer, and custom config files must be saved into the same directory along with LoginPage.thtml.
- If you intend to upload your files to Connect Secure, instead of processing them locally first, you can ignore this instruction. You must begin each file with the directive [% TAGS <% %] so that you can use angle brackets on any directives you add to the templates. By default, the Template Toolkit recognizes square brackets ([ ]) instead of angle brackets (<>) as delimiters for directives. Connect Secure recognizes angle brackets only. Adding the TAGS directive tells the local Template Toolkit utilities that you are using angle brackets.

To process LoginPage.thtml:

1. Open a command window.
2. Navigate to the directory in which you have saved the LoginPage.thtml file, customhead.thtml, customconfig.thtml, and customfoot.thtml. For example, /Templates/Sample/.
3. Enter the following command:

   tpage LoginPage.html > index.html

   If you get no errors and the system returns a blank directory prompt, the process completed successfully.
4. Open a browser window.
5. Select File > Open and navigate to the directory where you have saved LoginPage.thtml.

You should be able to see a copyright statement at the bottom of the page. The directive in customfoot.thtml pulled the data from customconfig.thtml and substituted the year and company name into the substitution variables $date and $company. Then, during the processing of LoginPage.thtml, customfoot.thtml was included based on the use of the PROCESS directive.
NOTE: Two warnings:

- If you received errors or if you were unable to run `tpage`, you need to check your Template Toolkit installation and documentation. You may not have the proper environment variable set to point to the Template Toolkit installation location or to your Perl distribution.

- Once you create a file, such as `index.html` in the preceding example, you must delete or move the file if you intend to create another version of the same file. The Template Toolkit `tpage` utility does not overwrite a file of the same name.

**Changing Colors in the Table-Based Layout**

Changing colors in the table-based layout is a very simple operation. All you need to do is replace color codes in the templates with the color codes representing the colors you prefer. Color codes are expressed in the hexadecimal format (#000000). If you want to change everything that is white to black, replace all occurrences of #FFFFFF to #000000.

In the **UI Options** page of a given role, you can also change banner, background, and link colors.

**Changing the Layout**

One of the simplest ways to customize the presentation of the pages is to modify the banner, colors, and table format. You can add table rows (`<tr>`) and table cells `<td>`, graphics, change the `custom-logo.gif` to your own company logo, and change the background color of one or more table cells. You can also move pre-configured objects, such as the username field, password field, realm drop down menu, and submit button to other locations within the table. We do not go into detail about how to accomplish this. You should be familiar with HTML tables and be prepared to go through a trial and error process.

If you prefer to separate the presentation from the content, consider using cascading style sheets (CSS).

**Using Cascading Style Sheets**

One of the most dramatic changes you can make to the presentation style of the templates is to convert them fully to use cascading style sheets (CSS). In some templates you can already find brief examples of CSS codes, such as the use of `<DIV>` and `<SPAN>` tags. In those cases, it is best to leave those tags in place. You can, however, replace much of the HTML code around those tags, as well as around Template Toolkit directives.

As mentioned before, the sample templates use a traditional table-based design model. In effect, the design uses HTML tables as containers for all other objects on the page. Table rows and cells contain all graphics, error messages, fields, and buttons.

Although tables can simplify some elements of HTML design, they can complicate others, not to mention that designing with tables means that you must use tables on every page, if you intend to maintain the same basic layout. This adds complexity and decreases download times, as the end-user's browser must parse each table in each page every time it loads a page.

CSS provides an alternative. CSS consists of a set of styles that can be embedded in each HTML page or that can be stored in a separate `.css` file and referenced in the HTML files with a
<link> tag. This topic is not meant to be a comprehensive CSS tutorial. It assumes you already have some familiarity with CSS. If you do not, you can find references on the Web, starting with the CSS specification at http://www.w3c.org.

For the purposes of this discussion, we will describe how to use CSS to eliminate redundant items, presentation information, and tables from your design. By doing so, you can improve the readability of your template files, and can consolidate almost all presentation instructions into one file.

**Embedding CSS**

You can embed CSS styles into an individual HTML page. If you plan on customizing only one file, you might find this approach easiest. Otherwise, you gain the most benefit by using a standalone CSS file and linking it into your HTML files.

**Attaching a Standalone CSS**

For the purposes of this discussion, we will introduce a simple CSS that will serve as an example method for replacing tables as a design element.

You can copy the following CSS into a text file and save it into the same directory with LoginPage.thtml. Save it as sample.css.

```css
body {        // Configures body margins
    margin:10px 10px 0px 10px;
    padding:0px;
}

span.error (      // Formats errors with bold and background
    font-weight: bold;
    background: silver;
}

#leftcontent {      // Styles left column
    position: absolute;
    left:10px;
    top:69px;
    width:199px;
    background:#809FB0;
    border:1px solid #000000;
    text-align:center
}

#centercontent {     // Styles center column
    background:#EFEFC9;
    margin-left: 175px;
    margin-right:11px;
    border:1px solid #000000;
    voice-family: inherit;
}```
margin-left: 199px;
margin-right: 0px;

#banner {
  // Styles the top banner
  background:#C0C0B0;
  height:60px;
  border-top:1px solid #000000;
  border-left:1px solid #000000;
  border-right:1px solid #000000;
  voice-family: ""; /*
  voice-family: inherit;
  height:59px;
  */
}

p,h1,h2,h3,pre {
  // Sets common properties of these
  margin:0px 10px 10px 10px;// paragraph elements
  font-family:Arial, Verdana, Helvetica, sans-serif;
}

p{
  // Styles paragraph element
  font-size:12px;
  text-align:left
}

h1,h2,h3 {
  // Sets common elements for
  padding-top:10px;// headings.
  font-weight:bold;
}

h1 {
  // Sets size of H1 heading
  font-size:18px;
}

h2 {
  // Sets size of H2 heading
  font-size:14px;
}

h3{
  // Sets size of H3 heading
  font-size:12px;
}

#banner h1 {
  // Sets heading selector for banner
  font-size:36px;
  padding:10px 10px 0px 10px;
  margin:0px;
  font-weight: bolder
}
To make the connection between the CSS stylesheet and the page, you must include a `<link>` tag prior to the closing head tag `</head>`, as follows:

```html
<link rel="stylesheet" type="text/css" href="sample.css" />
</head>
```

If you store the stylesheet in a directory other than the root directory where you keep your template files, make sure you express the path correctly in the `href` attribute of the `link` tag. For example, if you keep all of your stylesheets in a directory one level above the template file, you might indicate it as follows:

```html
<link rel="stylesheet" type="text/css" href="../sample.css" />
```

This CSS defines the presentation style of all of the major components of the page. Additionally, if you want to change the look-and-feel of the page, while keeping the basic structure intact, you can create another CSS that uses the same style names, but that modifies the definitions. Without making any changes to your template file, you can apply an entirely unique style.

Here is a copy of the `LoginPage.thtml` file (excluding the comments section) after it has been converted to a CSS-based page. Compare it with the template file that you download from Connect Secure. You might notice that virtually all of the presentation codes, things such as...
color settings, font settings, and border settings are missing from the following code. All of the presentation code has been extracted to the CSS stylesheet.

```html
<%# NetScreen Page Version 1001 %>

<!-- The following line, when uncommented, allows you to process the .thtml file locally using the tpage utility, without having to replace the angle brackets on all of the Connect Secure template directives. The TAGS directive tells the Template Toolkit utilities to substitute angle brackets for the default square brackets. This line should be on the first line of the file when you are testing locally. Prior to uploading to Connect Secure, move and comment the line, or remove it from the file completely. -->
<!--[% TAGS < % > %]-->

<% PROCESS customconfig.thtml %>

<% PROCESS customhead.thtml %>

<title>Instant Virtual Extranet</title>

<link rel="stylesheet" type="text/css" href="sample_two.css" />

<script>
<!--
JavaScript removed for readability. See template file for actual JavaScript.
//-->
</script>

<body id="body" onload="FinishLoad()">

<% PROCESS custombanner.thtml %>

<!-- ///////////////////////////////////////////////////////////// -->
<!-- BEGIN LEFT COLUMN -->
<!-- ///////////////////////////////////////////////////////////// -->

<div id="leftcontent">

<!-- ///////////////////////////////////////////////////////////// -->
<!-- BEGIN CUSTOM CONTENT LEFT COLUMN (Optional) -->
<!-- ///////////////////////////////////////////////////////////// -->
<p>You are not yet authorized!</p>

<% IF AccountDisabled %>

<% END %>

<% ELSE %>
<% IF LoginPageErrorMessage %>
<span class="error">% LoginPageErrorMessage %</span>
<% END %>
<% END %>

<%-- END PASSWORD MANAGEMENT DIRECTIVES -->
PART 2: Index

<!--  تمامی تصاویر غیر قابل نمایش ها -->

<!--  Optional error handling codes -->
<!--  تمامی تصاویر غیر قابل نمایش ها -->

<span class="error">
  <% IF PageErrorCode == 1020 %>
  You have successfully changed your password.
  <% END %>
</span>

<span class="error">
  <% IF PageErrorCode == 1021 %>
  Account Disabled. Please contact Administrator for help.
  <% END %>
</span>

<span class="error">
  <% IF PageErrorCode == 1023 %>
  Account Expired. Please contact Administrator for help.
  <% END %>
</span>

<!-- END optional error handling codes -->
<!--  تمامی تصاویر غیر قابل نمایش ها -->

<!-- BEGIN CUSTOM CONTENT CENTER COLUMN -->
<!--  Add any content below that you want to appear in the center column. -->
<!--  You can add a mix of headings, paragraphs, graphics or any other HTML. -->
<!--  تمامی تصاویر غیر قابل نمایش ها -->

<h1>Welcome to XYZ Company!</h1>
<h2>New Employee Benefits!</h2>
<p>Per munere latine officiis ad. Quo te elit vivendum, mea aperiri integre periculi ex, ea sea corpora consectetuer. </p>
<h2>Company Party!</h2>
<p>Dolore efficiantur vim ea, alia putant vivendo at pri. Odio cetero tibique no vis. Mea vero delenit ad. </p>
<h2>Corporate Travel Advisory</h2>
<p>Cu zzril expetenda has, ludus utinam instructor nam an, id eam fugit putent possit. Mazim epicuri contentiones mel ut, pri atqui maiestatis ei. </p>
When you manually launch Network Connect (from the Start menu) or initiate a connection to the PCS gateway from the Pulse Secure desktop client user interface, a different window appears compared to launching the client from a browser window.

To modify the standalone version of the login page, create a `LoginPage-stdaln.thtml` file (you can copy the existing `LoginPage.thtml` file), customize it according to your needs, and add it to your zip file.

Generally speaking, standalone pages follow the convention name-stdaln.thtml where name is the equivalent name.thtml file (similar to the method used for the Mobile files, name-ppc.thtml).

### Customizing Error Handling and Navigation

The Template Toolkit provides a rudimentary level of error handling capability. In most cases, errors are handled sufficiently by the included logic. However, you might want to modify the behavior of the pages, given a particular error. For example, you might want to redirect a user to your own specific custom error page on an “Account Disabled” error.

You can redirect based on specific individual errors or based on any error occurring at all. The following samples illustrate both methods.

#### Redirecting Based on Any and All Errors

If you want to redirect to your own Web page based on the occurrence of any and all errors, you can implement logic, as follows:

```html
<head>
  <% IF LoginPageErrorMessage %>
  <meta http-equiv="refresh" content="0;url=DisplayErrorPage.html" >
  <title>Redirect</title>
</head>
```
Redirecting Based on Specific Errors

If you want to redirect to your own Web page based on the occurrence of a specific error, such as the “Account Disabled” error, you can implement logic, as follows:

```html
<head>
  <% IF LoginPageErrorCode == 1021%>
  <meta http-equiv="refresh" content="0;url=AcctDisabledError.html" >
  <% ELSIF PageErrorCode == 1023 %>
  <meta http-equiv="refresh" content="0;url=AcctExpiredError.html" >
  <% END %>
<title>Redirect</title>
</head>
```

For more information on error codes, see “LoginPageErrorCode”.

Defining Error Message Locations

You can locate error messages wherever you want on the page. Keep in mind, however, that you cannot move the error handling code relative to its position within the `<form>` tags. For example, do not reverse the positions of two directives appearing within the body of `<form>` tags or move the error handling code outside of the `<form>` tags.

Customizing Error Messages

You can customize some of the error and confirmation messages your end-users might see. Connect Secure produces some error messages that you cannot change. Table 6 lists those that you can change.

Although you cannot change any messages on Connect Secure itself, you can customize error messages that will appear instead of the messages originating on Connect Secure. Using the Template Toolkit IF directive, you can test for a message based on an error’s unique error code, and expressed by the predefined template variable `PageErrorCode`. In the directive block, you include the customized message, as shown in the following example:

```html
<% IF PageErrorCode == 1020 %>
  You have successfully changed your password.
<% END %>
```

```html
<% IF PageErrorCode == 1021 %>
  Account Disabled. Please contact administrator at X5540 for help.
<% END %>
```

```html
<% IF PageErrorCode == 1023 %>
  Account Expired. Please contact administrator at X5541 for help.
```
Localizing Custom Sign-In Pages

By default, text strings are in English. Administrators supporting non-English users may need to configure the sign-in pages to provide localized text labels. This can only be done on a per-sign-in page basis. For multi-language support, Administrators must configure different sign-in pages for different locales. For further customizations, you can upload customized sign-in pages using the Template Toolkit. Contact Pulse Secure Global Support Centre for details.

You can create localized custom sign-in pages, using the downloadable sample template files. You can download the samples from Signing In > Sign-in pages > Upload Custom Pages > Upload Custom Sign-In Pages page.

You include the localized version of the template files in separate directories, one directory per language, in your uploaded zip file. Connect Secure recognizes all of the .thtml files in the root directory of the zip file as default pages. When creating the subdirectory for a set of given language files, name the subdirectory with the Accept-Language header 2-char abbreviation. Retain the default names for the localized file names.

At runtime, Connect Secure notes the default language of the browser, then searches the subdirectory corresponding to the 2-char Accept-Language abbreviation to find the specific .thtml file. If Connect Secure locates the file, Connect Secure uses that .thtml file. If Connect Secure cannot locate the file, Connect Secure uses the default page in the root directory.

NOTE: Use the Accept-Language abbreviation zh-cn if you plan to use either of the abbreviations zh-cn or zh-sg.

The directories should contain full sets of template files, as required to implement custom sign-in pages.

NOTE: For a list of acceptable Accept-Language header abbreviations, see http://www.oasis-open.org/cover/iso639a.html.

Creating a Localized Set of Custom Sign-In Pages

To create a localized set of custom sign-in pages, perform the following procedure.

1. Select Signing In > Sign in > Sign in pages.
2. Click Upload Custom Pages.
3. Click the link for the sample templates you want to localize.
4. Once downloaded, unzip the files.
5. Create a subdirectory in the sample templates directory and name it with the 2-character identifier that specifies the language into which you intend to translate the template files. For example, if you are translating into French, name the directory `fr`. Create subdirectories for each language into which you want to translate the templates, and copy all template files and the images directory into each subdirectory.

6. Copy all of the templates and the images directory (`/imgs`) and paste them into the new folder.

7. Translate the template files.

8. Change the paths in the translated files to point to the correct path. For example, you need to update the paths to images in the `/imgs` subdirectory, otherwise, they will continue to point to the default directory.

9. When translation is completed, zip the files and upload to Connect Secure.

10. To test the languages, make sure you update the language option in your Web browser, first.

**Upgrade Considerations**

Consider the following when upgrading to the latest version of the custom sign-in pages:

- You must update the version number in your templates to match the current version. If you do not update your version number, the default page appears instead of your custom page. Alternatively, you can copy your custom content to the new templates.

- Old variable names may change and new variables may be added. It is recommended that you convert old variable names to their new counterparts as the default values for the old variables may no longer exist. If you do not want to update your variable names, you can select the `Skip validation checks` during upload option in the **Upload Custom Sign-In Pages** page. If you select this option, you should review all your custom pages to ensure that they are still functioning correctly.

- Some formatting, such as text strings and color, change from prior versions. Review your page to ensure that the changes are acceptable.

**Custom Templates Reference**

The following topics describe each of the template files, JavaScript, template variables, some form variables, their uses, functionality, and customizability. The topics are organized alphabetically. Within the reference, you can find definitions for:

- Guest user access management user pages

You can customize the Policy Secure Service Guest User Account Manager user interface, which includes the Guest User Account Manager Home Page, New Local User Page (`UserAdmin-add-user.thtml`), Create Many Users Page (`UserAdmin-add-bulk-users.thtml`), Update User Account Page (`UserAdmin-update-user.thtml`), Print Details Page (`guest-user-print-page.thtml`, `guest-user-print-page-batch.thtml`).

The sample template files zip file includes the following GUAM files:
• UserAdmin-add-user.thtml
• UserAdmin-add-bulk-users.thtml
• UserAdmin-update-user.thtml
• guest-user-print-page.thtml
• guest-user-print-page-batch.thtml

• Connect Secure pre-authentication pages

You may customize a variety of standard Connect Secure pre-authentication pages, including the standard sign-in page (LoginPage.thtml), the failed authentication page (SSL.thtml), the sign out page (Logout.thtml), the sign-in warning page (ExceededConcurrent.thtml), the Host Checker and Cache Cleaner start page (PleaseWait.thtml), and the select from multiple roles page (SelectRole.thtml).

• ACE pre-authentication pages

The SoftID.zip file enables you to customize Connect Secure pages for use with the RSA Soft ID client.

When the end-user browses to the URL of the SoftID custom sign-in page, the page prompts the user to enter a PIN by way of the SecurID application on his local machine. If the application accepts the PIN, the end-user is logged in.

To configure the SoftID custom sign-in page, download the Samples.zip and SoftID.zip files. Unzip Samples.zip to a directory first, and then unzip the SoftID.zip file to the same location, overwriting any duplicates. Once you have customized the files to your liking, zip the entire set of files and upload them to Connect Secure.

To ensure that the New Pin and Next Token pages are customized for SoftID authentication, copy NewPin.thtml to GeneratePin.thtml in the SoftID.zip file and upload the modified zip file to Connect Secure for the custom sign-in page.

• ACE with eTrust pre-authentication pages

The ACE with eTrust pre-authentication pages are named with the SM prefix, for SiteMinder.

• Password management pages

These include pages such as the secondary authentication server login page (SecondaryLoginPage.thtml).

• Template variables

• Kiosk pages

The Kiosk.zip file contains templates that enable you to customize Connect Secure pages for use by Kiosk users. This sample shows how you can implement a custom UI to circumvent keystroke loggers. When you run the sample LoginPage.thtml (after it has been uploaded to Connect Secure) you see a Virtual Keyboard. The password field is disabled, so an end-user must use the Virtual Keyboard to enter a password. Because the normal keyboard is disabled, a keystroke logger cannot capture the password.

• Meeting pages
The *Meeting.zip* file enables you to customize a variety of pre-authentication and post-authentication pages used by Pulse Secure Collaboration attendees.

**NOTE:** When uploading templates to Connect Secure, you must include all of the THTML pages included in the original zip files for Connect Secure to accept the upload.

- **JavaScript**

  Many of the templates include optional JavaScript functions that you can use to perform a variety of tasks on that particular page. For example, the header in *LoginPage.thtml* contains several JavaScript functions. Most of these functions handle cases where you associate multiple authentication realms with a single sign-in URL. You can also add your own JavaScript functions to perform presentation effects, or to modify the default navigation by redirecting to your own Web pages, in some cases.

**Mobile Device Pages**

.thtml pages for mobile devices such as the Android and iPhone are identified by the *-mobile-webkit* string in the page name. For example, Cancel-mobile-webkit.thtml. Requirements for these pages are the same as their non-mobile counterparts except as noted in the following topics. For example, Cancel-mobile-webkit.thtml has the same requirements as cancel.thtml.

**iPad Pages**

.thtml pages for the iPad are identified by the *-ipad* string in the page name. For example, Cancel-ipad.thtml. Requirements for these pages are the same as their non-mobile counterparts except as noted in the following topics. For example, Cancel-ipad.thtml has the same requirements as cancel.thtml.

**<failedPolicy>**

Tag used to go through list of failed policies.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Template variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Included In</td>
<td>Remediate.thtml</td>
</tr>
<tr>
<td>Usage</td>
<td>Required. Do not modify.</td>
</tr>
<tr>
<td>Example</td>
<td>None</td>
</tr>
<tr>
<td>Related Documentation</td>
<td>Remediate.thtml / Remediate-mobile-webkit.thtml / Remediate-ipad.thtml</td>
</tr>
</tbody>
</table>

**<failedPolicy>.name**

Name of the policy that fails during remediation.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Template variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Included In</td>
<td>Remediate.thtml</td>
</tr>
</tbody>
</table>
### <failedPolicy>.remediation

Custom instructions as defined in the remediation policy.

<table>
<thead>
<tr>
<th><strong>Object Type</strong></th>
<th>Template variable</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Included In</strong></td>
<td>Remediate.thtml</td>
</tr>
<tr>
<td><strong>Usage</strong></td>
<td>Optional</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>None</td>
</tr>
</tbody>
</table>

**Related Documentation**: Remediate.thtml / Remediate-mobile-webkit.thtml / Remediate-ipad.thtml

### action

An HTML form element that provides the action that is meant to occur, usually on a submit button click or POST.

<table>
<thead>
<tr>
<th><strong>Object Type</strong></th>
<th>HTML form element</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Included In</strong></td>
<td>All templates with forms</td>
</tr>
<tr>
<td><strong>Usage</strong></td>
<td>Required</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>The following fragment from a <code>&lt;form&gt;</code> field runs the login.cgi Perl script on Connect Secure: action=&quot;login.cgi&quot;</td>
</tr>
</tbody>
</table>

**Related Documentation**: autocomplete

### activex_ini

Contains the name of an ActiveX control.

<table>
<thead>
<tr>
<th><strong>Object Type</strong></th>
<th>Template variable</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Included In</strong></td>
<td>LoginMeeting.thtml, MeetingTestJS.thtml</td>
</tr>
<tr>
<td><strong>Usage</strong></td>
<td>Required if running Internet Explorer on a Windows system. This variable contains the name of the ActiveX control that must be loaded to run Meeting in the browser.</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>The following sample checks to see if the browser is Internet Explorer. If so, it passes the name of an ActiveX object to initialize.</td>
</tr>
</tbody>
</table>

```html
<% IF ie %>
  <P>
  <OBJECT>
  <% activex_ini FILTER verbatim%>
  </OBJECT>
  </P>
```
Related Documentation

- applet_ini
- FILTER verbatim

**AgentInstall.thtml**

The page after downloading the UAC Agent.

**Object Type** Template variable  
**Usage** Optional (Valid for Policy Secure).  
**Required Variables** DSLaunchURL  
pleaseWaitObjectHC  
IEonWindows  
delivery_mode  
setup_codebase  
setup_classid  
setup_applet_vitals  
isMac  
pleaseWaitObjectJAM  
I18N_UACDOWNLOAD_AND_INSTALL_64_BIT

**AgentlessInstall.thtml**

The page after downloading the UAC Agent.

**Object Type** Template variable  
**Usage** Optional (Valid for Policy Secure).  
**Required Variables** DSLaunchURL  
pleaseWaitObjectEP  
setup_classid  
setup_codebase  
isLinux
AnonymousAuthentication

Use this variable to specify that users signing into an anonymous realm should not see the username and password fields in the sign-in page.

Object Type: Template variable

Included In: LoginPage.thtml

Usage: Optional. Connect Secure sets this value to true if the authentication realm is set to an anonymous server. If you use this variable, Connect Secure only displays the sign-in page to users when they encounter one of the errors described in "LoginPageErrorCode". If you set AnonymousAuthentication, the realm is available in RealmList.0. You need to send the realm in the form as a hidden variable.

Example: The following sample checks to see if AnonymousAuthentication and CertificateAuthentication are set to false. If the variables are set to false, the following directives generate authentication fields, setting the size of the input fields to 20 characters.

```html
<%IF !AnonymousAuthentication && !CertificateAuthentication%>
<% FOREACH prompt = prompts %>
<%NEXT IF !prompt.required %>
<% prompt.promptText %>
<input type="<% prompt.type %>" name="<% prompt.name %>" size="20">
<% END %>
```

Related Documentation

- CertificateAuthentication
- RealmList

applet_ini

Contains instructions that initialize a meeting applet.

Object Type: Template variable

Included In: MeetingAppletInstaller.thtml, MeetingTestMSJava.thtml

Usage: Required

Example: The following code fragment shows how the applet_ini variable passes an applet to the page, and how the additional code passes parameters to the applet:

```html
<% IF install %>
<P>
<APPLET
```
Related Documentation  MeetingAppletInstaller.html

**appstr**

Contains the applet tag for the Pulse Secure Collaboration client.

**Object Type**  Template variable

**Included In**  MeetingRunJava.html

**Usage**  Required

**Example**  The following example checks to see if a login is required. If not, Connect Secure loads the applet indicated in the appstr variable.

```jsp`
<% IF !login_required %>
<% appstr FILTER verbatim %>
<% END %>
```

Related Documentation  FILTER verbatim

**authenticate()**

Authenticates a user.

**Object Type**  JavaScript function

**Included In**  GeneratePin.html (ACE/SoftID), LoginPage.html (ACE/SoftID), NewPin.html (ACE/SoftID), NextToken.html (ACE/SoftID)

**Usage**  Required

**Example**  The following examples illustrate the use of the authenticate() function.

From ACE/SoftID LoginPage.html. This function checks to see whether or not the user is authenticated. If the user is not authenticated, the function displays an error from the authentication server. Otherwise, it passes the authentication credentials from the auth server to the frmLogin HTML form in LoginPage.html:

```javascript`
function authenticate()
{
    var rc = -1;
    var time = <% softid_time %>
    var user = document.frmLogin.username.value;
```
```javascript
var state = 0;

if(typeof(document.sdui.sdAuth) == "undefined")
{
    rc = -1;
} else
{
    if (error != "")
    {
        alert(error);
    }
    rc = document.sdui.sdAuth(time, user, state);
}
if (rc == 1)
{
    document.frmLogin.username.value = document.sdui.getUsername();
    document.frmLogin.password.value = document.sdui.getPasscode();
    document.frmLogin.submit();
} else
{
    document.cancelForm.submit();
}
```

From ACE/SoftID GeneratePin.thtml. The following function enables the user to generate a new PIN for signing in to an authentication server:

```javascript
function authenticate()
{
    var min = parseInt(document.minmaxForm.minlen.value);
    var max = parseInt(document.minmaxForm.maxlen.value);
    var alp = parseInt(document.minmaxForm.alphanumeric.value);
    var sys = document.minmaxForm.systempin.value;

    if(typeof(document.sdui.sdPin) == "undefined")
    {
        rc = -1;
    } else
    {
        if (error != "")
        {
            alert(error);
        }
        rc = document.sdui.sdPin(min, max, <%= secid_pinselectmode %>, alp, sys);
    }
    if (rc == 1)
    {
```
    document.frmNewPin.password.value = document.sdui.getPin();
    document.frmNewPin.password2.value =
    document.frmNewPin.password.value;
    document.frmNewPin.secidaction.value = "Save PIN";
    document.frmNewPin.secidactionSavePin.value = "Save PIN";
    document.frmNewPin.submit();
}   
else if(rc == 2)  
{    
    document.frmNewPin.password.value = "";
    document.frmNewPin.password2.value="";
    document.frmNewPin.secidactionCancel.value = "Cancel";
    document.frmNewPin.submit();
}    
else    
{    
    document.cancelForm.submit();
}

**Related Documentation**

- GeneratePin.thtml / GeneratePin-mobile-webkit.thtml / GeneratePin-ipad.thtml
- LoginPage.thtml / LoginPage-mobile-webkit.thtml / LoginPage-ipad.thtml

### autocomplete

Prevents the form from auto-filling username or authentication realm field entries using cached values.

**Object Type**
HTML form element

**Included In**
All templates with forms

**Usage**
Optional

**Example**

```html
<form name="frmLogin" action="login.cgi" method="post" autocomplete="off" onsubmit="return Login()">
```

**Related Documentation**
None

**Cancel.thtml / Cancel-mobile-webkit.thtml / Cancel-ipad.thtml**

Displays a message to the user informing him that the sign-in request has been cancelled.

**Object Type**
Template

**Usage**
Required

**Required Variables**
None

**Example**
None
### cboxuser

Indicates whether or not the attendee is a Connect Secure user.

<table>
<thead>
<tr>
<th><strong>Object Type</strong></th>
<th>Template variable</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Included In</strong></td>
<td>LoginMeeting.thtml</td>
</tr>
<tr>
<td><strong>Usage</strong></td>
<td>Required</td>
</tr>
</tbody>
</table>

**Example**

The following example displays the text box and label in an HTML table.

```html
<% IF cboxuser %>
<tr><td align="right">Your Name:</td>
<td><input type=text name="username" value="<% username %>" maxlength="40" size="20"></td>
<% END %>
```

**Related Documentation**

LoginMeeting.thtml

### ccInAcTimeout

Specifies the Cache Cleaner login inactivity in minutes.

<table>
<thead>
<tr>
<th><strong>Object Type</strong></th>
<th>Template variable</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Included In</strong></td>
<td>LoginPage.thtml</td>
</tr>
<tr>
<td><strong>Usage</strong></td>
<td>Required</td>
</tr>
</tbody>
</table>

**Example**

The following code sets the timeout interval to 60000 milliseconds times the smaller value of ccInAcTimeout or hcInAcTimeout.

```javascript
<% IF hcInAcTimeout > ccInAcTimeout %>
window.setTimeout("deletepreauth();", 60000 * <% ccInAcTimeout %>);  
<% ELSE %>
window.setTimeout("deletepreauth();", 60000 * <% hcInAcTimeout %>);  
<% END %>
```

**Related Documentation**

LoginPage.thtml / LoginPage-mobile-webkit.html / LoginPage-ipad.html

### ccRunning

Specifies whether or not Cache Cleaner is currently running.

<table>
<thead>
<tr>
<th><strong>Object Type</strong></th>
<th>Template variable</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Included In</strong></td>
<td>LoginPage.thtml</td>
</tr>
</tbody>
</table>
Usage
Required

Example
The following code sets the timeout interval based on whether Host Checker and Cache Cleaner are both running or only Host Checker.

```javascript
<% IF hcRunning && ccRunning %>
  <% IF hcInAcTimeout > ccInAcTimeout %>
    window.setTimeout("deletepreauth()");, 60000 * <% ccInAcTimeout %>);
  <% ELSE %>
    window.setTimeout("deletepreauth()");, 60000 * <% hcInAcTimeout %>);
  <% END %>
  <% ELSIF hcRunning %>
    window.setTimeout("deletepreauth()");, 60000 * <% hcInAcTimeout %>);
  <% END %>
```

Related Documentation
LoginPage.html / LoginPage-mobile-webkit.html / LoginPage-ipad.html

**cert_md5**

Contains the cert-md5 statement (the MD5 fingerprint of the SSL peer's certificate) that enables the user to install an applet in a meeting session.

Object Type
Template variable

Included In
MeetingAppletInstaller.html

Usage
Required

Example
This fragment, which is contained in MeetingAppletInstaller.html, shows the use of the cert_md5 variable:

```html
<% IF install %>
  <APPLET
    <% applet_ini FILTER verbatim %>
      <PARAM NAME="Parameter0" VALUE="meeting_id=<%instanceId%>;user_name=<%username%>;cert_md5=<% cert_md5 %>;ncp_read_timeout=<%ncp_read_timeout%>"
      <PARAM NAME="StartSessionReDir" VALUE="<%startSessionReDir%>">
```

Related Documentation
MeetingAppletInstaller.html

**CertificateAuthentication**

Use this variable if you want to specify that users signing into a certificate realm should not see the username and password fields in the sign-in page.

Object Type
Template variable

Included In
LoginPage.html

Usage
Connect Secure sets this value to true if the authentication realm is set to a certificate server. If you use this variable, Connect Secure only displays the sign-in page to users when they encounter one of the errors described in " LoginPageErrorCode". (For information about hiding
this page from users authenticated against a eTrust SiteMinder server using client-side certificate authentication, see “Login()” If you set CertificateAuthentication, the realm is available in RealmList0. You need to send the realm in the form as a hidden variable.

Example
The following code tests for AnonymousAuthentication and CertificateAuthentication:

```html
<%IF !AnonymousAuthentication && !CertificateAuthentication%>
<% FOREACH prompt = prompts %>
<% NEXT IF !prompt.required %>
<% prompt.promptText %>
<%input type="<% prompt.type %>" name="<% prompt.name %>" size="20">%>
<% END %>
```

Related Documentation
- AnonymousAuthentication
- RealmList

**changePasswordTitle**

Argument to display password change title text.

**Object Type**
Template variable

**Included In**
PasswordChange.thtml

**Usage**
Optional

**Example**
The following code tests the boolean showChangePasswordTitle and, if true, passes the argument to the page, as is.

```html
<%IF showChangePasswordTitle %>
<% changePasswordTitle FILTER verbatim %>  
<%END%>
```

Related Documentation
- showChangePasswordTitle

**check**

Contains the name of the GIF file that Connect Secure displays on the MeetingTestResult page.

**Object Type**
Template variable

**Included In**
MeetingTestResult.thtml

**Usage**
Required

**Example**
The following code displays the GIF that is referenced in the check template variable:

```html
<table cellpadding="2" cellspacing="2" border="0">
<tr valign="top">
<td>
```

```html
```
Related Documentation

**MeetingTestResult.html**

**checkActiveX()**

Checks to see if ActiveX is enabled.

**Object Type**
JavaScript function

**Included In**
MeetingAppletInstaller.html, MeetingTestMSJava.html, MeetingTestJS.html, MeetingTestJava.html

**Usage**
Optional

**Example**
The following examples illustrate the verification of ActiveX.

From **MeetingTestMSJava.html**:

```javascript
function checkActiveX () {
    if (typeof(SecureMeetingApplet) == "undefined") {
        redirect();
        return true;
    }
    setTimeout('redirect()', 5000);
    return true;
}
```

From **MeetingRun.html**:

```javascript
function checkActiveX () {
    var f = document.meetingrun;
    if (typeof(f) == "undefined" ||
        typeof(f.button1) == "undefined")
    return;
    try {
        NeoterisSetup.isValid();
        f.button1.disabled = false;
    } catch (e) {
        ;
    }
}
```
From MeetingTestJS.thtml:

```javascript
function checkActiveX () {
    var win = <% win32 %>; // win32 is defined in includes.js
    try {
        NeoterisSetup.isValid();
        document.form1.submit();
        return true;
    } catch (e) {
        var url = win ? "/dana-na/meeting/<%signin%>meeting_testmsjava.cgi?<%mid_param%>" : "/dana-na/meeting/<%signin%>meeting_testjava.cgi?<%mid_param%>";
        document.location.replace(url);
    }
}
```

From MeetingTestJava.thtml:

```javascript
function checkActiveX () {
    setTimeout('redirect()', 30000);
    return true;
}
```

Related Documentation
- MeetingAppletInstaller.thtml
- MeetingRun.thtml
- MeetingTestMSJava.thtml
- MeetingTestJava.thtml
- MeetingTestJS.thtml

---

### color

Contains the color definition.

**Object Type**: Template variable  
**Default value**: None  

**Cancel.thtml**  
**NewPinSystem.thtml**  
**Usage**: Optional  
**Example**: This example sets the background color of the table cell to the value of `color`.

```html
<td bgcolor="<% color %>"><img border="0" src="welcome.cgi?p=logo" alt="Logo"></td>
```
### Continue

Submits the hidden form when an end-user clicks the **Continue** button.

**Object Type**: JavaScript function  
**Included In**: Remediate.thtml  
**Usage**: Optional  
**Example**: The following code fragment defines the **Continue** button and passes control to the Continue() JavaScript function.

```html
<input name="btnContinue" type="button" value="<% textContinue %>">
onClick="Continue()"
```

**Related Documentation**: Remediate.thtml / Remediate-mobile-webkit.thtml / Remediate-ipad.thtml

### secid_key

Contains the transaction state.

**Object Type**: Template variable  
**Included In**: Defender.thtml, Defender-mobile-webkit.thtml, Defender-ipad.thtml  
**Usage**: Required  
**Example**: The following example uses secid_key in the INPUT statement.

```html
<input type=hidden name="key" value="<% secid_key %>">
```

**Related Documentation**: Defender.thtml / Defender-mobile-webkit.thtml / Defender-ipad.thtml

### Defender.thtml / Defender-mobile-webkit.thtml / Defender-ipad.thtml

Prompts the Radius user to enter his PIN and provides the appropriate challenge values from the server.

**Object Type**: Template  
**Usage**: Password Management  
**Required Variables**: "secid_challenge"  
**Example**: None

### delivery_mode

Indicates the delivery mode of an applet, whether Java or ActiveX.

**Object Type**: Template variable
Included In
AgentInstall.thtml, Logout-ipad.thtml, Logout-mobile-webkit.thtml, Logout-ppc.thtml, Logout.thtml, PleaseWait-ppc.thtml

Usage
Required

Example
The following fragment checks to see if the delivery mode is Java when Cache Cleaner, Host Checker, or Win32 applets are running during a user logout. If Java is the delivery mode, the code indicates that the page should call the stopComponents function when loading.

```html
<% IF (pleasewaitObjectCC || pleasewaitObjectHC || pleasewaitWin32) &&
        delivery_mode == 'java'%>
        onload="javascript:stopComponents()"
<%END%>
```

Related Documentation
stopComponents()

---

DoUnload()

Forces Connect Secure to cancel the current mode.

Object Type
JavaScript function

Included In
GeneratePin.thtml, NewPin.thtml

GeneratePin.thtml
Usage
Required. You must include this function in the header of NewPin.thtml and call it from the body.

Example
The following code shows how the DoUnload() function appears in the NewPin.thtml template. The code serves a similar purpose in other templates, as well.

```javascript
function DoUnload() {
    if (gCancelNewPinMode) {
        window.open("secid_cancelpin.cgi", "newpincancel",
            "resizable=1,height=250,width=200,status=0");
    }
}
```

Related Documentation
- NewPin.thtml / NewPin-mobile-webkit.thtml / NewPin-ipad.thtml
- NextToken.thtml / NextToken-mobile-webkit.thtml / NextToken-ipad.thtml

---

ds_mobile_common.css

Cascading style sheet for .thtml mobile pages such as the Android and iPhone.

Object Type
Text file

Included In
PageHeader-mobile-webkit.thtml

Example
The following code shows how the ds_mobile_common.css style sheet is called in the PageHeader-mobile-webkit.thtml template.
ExceededConcurrent.thtml / ExceededConcurrent-mobile-webkit.thtml / ExceededConcurrent-ipad.thtml

Displays a performance warning to the user when too many concurrent users are signed into Connect Secure.

**Object Type**: Template

**Usage**: Required. You must always include this template in your zip file. This is a standard Connect Secure page that displays a performance warning to the user when too many concurrent users are signed into Connect Secure. This template does not contain any JavaScript or forms. It does contain an optional link to `starter.cgi`, however, that allows users to sign into Connect Secure as well as error message variables. For more information, see comments in the template.

**Required Variables**: None

**Example**: None

**expired**

Boolean indicating whether or not a meeting session has expired.

**Object Type**: Template variable

**Included In**: MeetingRun.thtml

**Usage**: Required

**Example**: The following code checks to see if ActiveX is enabled and that the session is not expired. If both conditions are true, the meeting is initialized.

```html
<% IF type == "activex" && !expired %>
<% meeting_ini FILTER verbatim %>
<% END %>
```

**Related Documentation**: MeetingRun.thtml

**fieldsArr**

Template Toolkit hash value of additional fields for guest user information (keys: field and fieldVal).

**Object Type**: Template variable

**Included In**: guest-user-email-page.thtml, guest-user-print-page-batch.thtml, guest-user-print-page.thtml
Usage
Optional (Required for Policy Secure)

Example
The following code displays the additional fields of guest user.

```%
FOREACH f = fieldsArr%
  f.field
  f.fieldVal
% END%
```

Related Documentation

**fallback_setup_applet_vitals**

Setup client Java applet.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Template variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Included In</td>
<td>Logout.thtml</td>
</tr>
<tr>
<td>Usage</td>
<td>Required</td>
</tr>
<tr>
<td>Example</td>
<td></td>
</tr>
</tbody>
</table>

```APPLET id=NeoterisSignoff fallback_setup_applet_vitals FILTER verbatim %>`

Related Documentation

**FILTER verbatim**

Indicates that the page should process the preceding variable raw, without performing any parsing or other actions on the variable contents.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Template directive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Included In</td>
<td>LoginMeeting.thtml, MeetingAppletInstaller.thtml, MeetingRun.thtml, MeetingRunJava.thtml, MeetingTestJava.thtml, MeetingTestJS.thtml, MeetingTestMSJava.thtml, MeetingTestResult.thtml, MeetingTrouble.thtml</td>
</tr>
<tr>
<td>Usage</td>
<td>Optional. Most often used when loading applets or ActiveX controls, which do not need to be processed by the Template Toolkit utilities in any way, but which should be passed through to the page as is. Can also be used to pass text to the page, unfiltered.</td>
</tr>
<tr>
<td>Example</td>
<td>The following code displays a label in a meeting page and loads the teleconference information contained in the template variable, regardless of its content or format:</td>
</tr>
</tbody>
</table>

```span class="cssSmall"><b>Teleconference:</b></span><% teleconference_info FILTER verbatim %></span>```
**FinishLoad()**

Use this function if you want to allow users to select from multiple realms in the sign-in page.

**Object Type**  
JavaScript function

**Included In**  

**Usage**  
Optional. If you are authenticating users through a eTrust SiteMinder server using client-side certificate authentication, you may want to hide the standard Connect Secure sign-in page from users. If you are authenticating users through an anonymous authentication server or certificate authentication server and want to bypass the standard Connect Secure sign-in page, see "AnonymousAuthentication" or "CertificateAuthentication".

**Example**  
To post data to Connect Secure without prompting users for any credentials, you may use the following modified version of the **FinishLoad()** function. Note that this function only bypasses the sign-in page if no errors occur while loading the page:

```javascript
function FinishLoad() {
    recallLastRealmUsed();
    <% IF !LoginPageErrorCode %>
    Login();
    document.frmLogin.submit();
    <% END %>
}
```

**Related Documentation**

- GeneratePin.html / GeneratePin-mobile-webkit.html / GeneratePin-ipad.html
- LoginPage.html / LoginPage-mobile-webkit.html / LoginPage-ipad.html
- SM-NewUserPin.html

**focus**

Sets the focus on given component on a form.

**Object Type**  
Template variable

**Included In**  

**Usage**  
Required
Example

The following code sets the focus (by way of a JavaScript function also called `focus()`) to the form object indicated by the template variable `focus`:

```javascript
function onLoad() {
    grab();
    document.frmLogin.<%= focus %>.focus();
}
```

Related Documentation

None

**FriendlyTime**

Argument in number of seconds.

Object Type

Template variable

Included In

PasswordChange.thtml, PasswordExpiration.thtml

Usage


Example

None

Related Documentation

- PageErrorArgs
- PageErrorCode

**frmCasque**

HTML form that enables an end-user to initiate the CASQUE RADIUS server.

Object Type

HTML form

Included In

Defender.thtml

Usage

Required

Example

The following code illustrates the use of the `frmCasque` form. If the plug-in type is the CASQUE RADIUS server plug-in, then the page loads the form:

```html
<% IF plugintype == 'CASQUE' %>
<form name="frmCasque" method="POST" autocomplete=off>
<input type=hidden name="key" value="<%= secid_key %>">
<input type=hidden name="p" value="casqueapp">
<a href="#" onClick="submitFrmCasque(); return false;">LAUNCH CASQUE PLAYER</a>
</form>
```

Related Documentation

`submitFrmCasque()`
### frmLogin

HTML form that enables the template page to trigger a call to the `Login()` function and begins the sign in process for a user.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>HTML Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>Included In</td>
<td>LoginPage.thtml, LoginMeeting.thtml</td>
</tr>
<tr>
<td>LoginPage.thtmlUsage</td>
<td>Required</td>
</tr>
<tr>
<td>Example</td>
<td>The following code is the form definition statement:</td>
</tr>
</tbody>
</table>

```html
<form name="frmLogin" action="login.cgi" method="post" autocomplete="off" onsubmit="return Login()">
</form>
```

### Related Documentation

autocomplete

### frmNextToken

Next Token HTML form.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>HTML Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>Included In</td>
<td>NextToken.thtml, SM-NextToken.thtml</td>
</tr>
<tr>
<td>Usage</td>
<td>Required</td>
</tr>
<tr>
<td>Example</td>
<td>The following code illustrates the suggested form definition:</td>
</tr>
</tbody>
</table>

```html
<FORM name="frmNextToken" action="login.cgi" method="POST" autocomplete=off onsubmit="return SubmitClicked();">
</FORM>
```

### Related Documentation

NextToken.thtml / NextToken-mobile-webkit.thtml / NextToken-ipad.thtml

### frmSelectRoles

Select Roles HTML form

<table>
<thead>
<tr>
<th>Object Type</th>
<th>HTML Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>Included In</td>
<td>SelectRole.thtml</td>
</tr>
<tr>
<td>SelectRole.thtmlUsage</td>
<td>Required</td>
</tr>
<tr>
<td>Example</td>
<td>The following code illustrates the suggested form definition:</td>
</tr>
</tbody>
</table>

```html
<FORM name="frmSelectRoles" autocomplete=off>
</FORM>
```

### Related Documentation

SelectRole.thtml / SelectRole-mobile-webkit.thtml / SelectRole-ipad.thtml
### g_time

The time when the browser first loads the page.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>JavaScript variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Included In</td>
<td>PleaseWait.thtml</td>
</tr>
<tr>
<td>Usage</td>
<td>Optional</td>
</tr>
<tr>
<td>Example</td>
<td>None</td>
</tr>
</tbody>
</table>

**Related Documentation**  
PleaseWait.thtml / PleaseWait-mobile-webkit.thtml / PleaseWait-ipad.thtml

### gCancelNewPinMode

Prevents Connect Secure from cancelling Generate PIN mode while the user is trying to save a PIN or generate a new one.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>JavaScript variable/flag</th>
</tr>
</thead>
<tbody>
<tr>
<td>Included In</td>
<td>GeneratePin.html, NewPin.html</td>
</tr>
<tr>
<td>Usage</td>
<td>Required. Works together with DoUnload() and SubmitClicked() functions to call the appropriate CGI on Connect Secure if the user chooses to close the browser window without cancelling his session first. If you do not include this code, the user may be locked out of Connect Secure when he closes his browser. In addition to including this JavaScript in your header, you must also call the function DoUnload() function from the body of GeneratePin.html.</td>
</tr>
<tr>
<td>Example</td>
<td>You include the following code in the header of the NewPin.html template. This function saves or cancels a PIN per the user’s request. Note that in this example, the gCancelNewPinMode is set to false though it was initialized to true when the page loaded.</td>
</tr>
</tbody>
</table>

```javascript
// Save / Cancel a PIN
function SubmitClicked() {
  gCancelNewPinMode = false;
  return true;
}
```

**Related Documentation**  
SubmitClicked() 

### gCancelNextTokenMode

Prevents Connect Secure from cancelling Next Token mode while the user is trying to enter his token.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>JavaScript function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Included In</td>
<td>NextToken.html</td>
</tr>
<tr>
<td>Usage</td>
<td>Required. Works together with DoUnload() and SubmitClicked() functions to call the appropriate CGI on Connect Secure if the user chooses to close the</td>
</tr>
</tbody>
</table>
browser window without cancelling his session first. If you do not include this code, the user may be locked out of Connect Secure when he closes his browser. In addition to including this JavaScript in your header, you must also call the `DoUnload()` function from the body of `NextToken.thtml`.

**Example**

You must include this flag in the header of `NextToken.thtml`:

```javascript
var gCancelNextTokenMode = true;
```

**Related Documentation**
`NextToken.thtml / NextToken-mobile-webkit.thtml / NextToken-ipad.thtml`

---

**geckoBrowser**

Boolean set to 1 if the browser is gecko-based (Firefox).

**Object Type**
Template variable

**Included In**
`Logout.thtml, PleaseWait.thtml`

**Usage**
Required

**Example**

The following code checks to see if the browser is Firefox, If yes, then the page loads the proxy form in an iFrame (inline frame).

```html
<% IF geckoBrowser %>
<IFRAME ID="proxy_frm" src="/dana-na/help/blank.html" WIDTH=0 HEIGHT=0 FRAMEBORDER=0 SCROLLING=NO></IFRAME>
<% END %>
```

**Related Documentation**
`safari`

---

**GeneratePin.thtml / GeneratePin-mobile-webkit.thtml / GeneratePin-ipad.thtml**

Enables the user to create a system-generated PIN.

**Object Type**
Template

**Usage**
Required

**Required Variables**
None

**Example**

When configuring the `secid_pinselectmode` variable (which sets the user’s PIN select mode), note that possible values include:
Table 5: PIN usage indicators

<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>User must use the system-generated PIN. He cannot enter his own PIN.</td>
</tr>
<tr>
<td>1</td>
<td>User may enter his own PIN or use the system-generated PIN.</td>
</tr>
<tr>
<td>2</td>
<td>User must enter his own PIN. He cannot use the system-generated PIN.</td>
</tr>
</tbody>
</table>

GetCookieValue()

Generic helper function that retrieves a value from a cookie, creates a string the length of the value, and populates the string with the value.

Object Type JavaScript function

Included In LoginPage.thtml, PleaseWait.thtml

Usage Use this function if you want to allow users to select from multiple realms in the sign-in page.

sName—Cookie name.

Example

```javascript
function GetCookieValue(sName) {
    var s = document.cookie;
    sName += "=";

    // where nv pair starts
    var nStart = s.indexOf(sName);
    if (nStart == -1)
        return "";
    else
        nStart += sName.length;
    // if more values, clip, otherwise just get rest of string
    var nEnd = document.cookie.indexOf(";", nStart);
    if (nEnd == -1)
        s = unescape(s.substring(nStart));
    else
        s = unescape(s.substring(nStart, nEnd));
    return s;
}
```

Related Documentation

- LoginPage.thtml / LoginPage-mobile-webkit.thtml / LoginPage-ipad.thtml
- PleaseWait.thtml / PleaseWait-mobile-webkit.thtml / PleaseWait-ipad.thtml
**grab()**

Checks for browser type and grabs a cookie when a user tries to sign in to a meeting.

**Object Type**
JavaScript function

**Included In**
LoginMeeting.thtml

**Usage**
Required

**Example**
The following code shows the `grab()` function:

```javascript
function grab() {
    var f = document.frmLogin;
    <% IF ie %>
    try {
        var result = NeoterisSetup.isValid();
        factivex_enabled.value = 1;
        if (result == 1) f.power_user.value = 1;
    } catch (e) {
        factivex_enabled.value = 0;
    }
    <% END %>
    f.java_enabled.value = navigator ? (navigator.javaEnabled() ? 1 : 0) : 0;
    <% IF document.cookie.indexOf("DSMTCookie") < 0 %> 
    alert("You must enable cookies through your Web browser in order to sign into the meeting.");
    <% END %>
}
```

**Related Documentation**
LoginMeeting.thtml

**GraceLoginsRemaining**

Number of remaining grace logins.

**Object Type**
Template variable

**Included In**
GraceLoginUsed.thtml

**Usage**

**Example**
The following code shows an example of how to display a message for the number of grace logins available to the user:

```html
<% IF LoginPageErrorCode == 2016 %>
You have <% LoginPageErrorArgs.GraceLoginsRemaining %> grace logins.
<% END %>
```

**Related Documentation**
- GraceLoginUsed.thtml / GraceLoginUsed-mobile-webkit.thtml / GraceLoginUsed-ipad.thtml
- PageErrorArgs
- **PageErrorCode**

**GraceLoginUsed.thtml / GraceLoginUsed-mobile-webkit.thtml / GraceLoginUsed-ipad.thtml**

Tells the user how many more times he can sign in using his current username and password.

**Object Type** Template

**Usage** Required

**Required Variables** "startPageLink"

**Example** None

**guest-user-print-page.thtml**

Page for printing single users by the Guest User Account Manager.

**Object Type** Template

**Usage** Optional (Valid for Policy Secure)

**Required Variables** fieldsArr

**guest-user-print-page-batch.thtml**

Page for printing batch users by the Guest User Account Manager.

**Object Type** Template

**Usage** Optional (Valid for Policy Secure)

**Required Variables** UsersPrintInfo

**guest-user-email-page.thtml**

Page for emailing user details by the Guest User Account Manager.

**Object Type** Template

**Usage** Optional (Valid for Policy Secure)

**Required Variables** fieldsArr

**HC_REMED_POLICIES_CHECK**

Default text to suggest that policies are being evaluated.

**Object Type** Template variable

**Included In** Remediate.thtml

**Usage** Optional

**Example** The following code checks to see if the page is loading. If yes, then the page displays several messages, including the default text to suggest that Connect Secure is evaluating policies:
<% IF showLoading %>
<% loading %>
<% pleasewait %>
<% HC_REMED_POLICIES_CHECK %>
<% END %>

Related Documentation

- HC_REMED_SHOW_adv_PREFS
- Remediate.html / Remediate-mobile-webkit.html / Remediate-ipad.html

**HC_REMED_SHOW_adv_PREFS**

Default text instructing how to re-enable remediation.

**Object Type**
Template variable

**Included In**
Remediate.html

**Usage**
Optional

**Example**
The following code illustrates how to call the variable to display instructions to re-enable remediation:

<% IF showRemedOption %>
<input name="btnHideRemed" type="button"
    value="<% textRemedOption %>" onClick="HideRemed();
    <% disabled %>">
<% HC_REMED_SHOW_adv_PREFS %>
<% END %>

Related Documentation

- HC_REMED_SHOW_adv_PREFS
- Remediate.html / Remediate-mobile-webkit.html / Remediate-ipad.html

**hcInAcTimeout**

Specifies the Host Checker login inactivity in minutes.

**Object Type**
Template variable

**Included In**
LoginPage.html

**Usage**
Required

**Example**
The following code sets the timeout interval to 60000 milliseconds times the smaller value of ccInAcTimeout or hcInAcTimeout.

<% IF hcInAcTimeout > ccInAcTimeout %>
    window.setTimeout("deletepreauth();", 60000 * <% ccInAcTimeout %>);
<% ELSE %>

© 2016 by Pulse Secure, LLC. All rights reserved
window.setTimeout("deletepreauth();", 60000 * <% hcInAcTimeout %>);
<% END %>

Related Documentation  
LoginPage.html / LoginPage-mobile-webkit.html / LoginPage-ipad.html

hcRunning

Specifies whether or not Host Checker is currently running.

Object Type  
Template variable

Included In  
LoginPage.html

Usage  
Required

Example  
The following code sets the timeout interval based on whether Host Checker and Cache Cleaner are both running or only Host Checker.

<% IF hcRunning && ccRunning %>
<% IF hcInAcTimeout > ccInAcTimeout %>
  window.setTimeout("deletepreauth();", 60000 * <% ccInAcTimeout %>);
<% ELSE %>
  window.setTimeout("deletepreauth();", 60000 * <% hcInAcTimeout %>);
<% END %>
<% ELSIF hcRunning  %>
  window.setTimeout("deletepreauth();", 60000 * <% hcInAcTimeout %>);
<% END %>

Related Documentation  
LoginPage.html / LoginPage-mobile-webkit.html / LoginPage-ipad.html

heading

Contains default text for the different remediation headings.

Object Type  
Template variable

Included In  
Remediate.html

Usage  
Required

Example  
The following code shows a heading and a message if the showHeading variable is true:

<% IF showHeading %>
  <% heading %>
  <% msg FILTER verbatim %>
<% END %>

Related Documentation  
showHeading

help

String to use as the button name.
**help**

Object Type: Template variable

Default value: "Help"

Included In: LoginPage.html, SecondaryLoginPage.html

LoginPage.htmlUsage: Optional

Example: The string "Help" (the default value of the help variable) is displayed on the button.

```
<input type='submit' name='help' value='<%= help %>'>
```

Related Documentation: Adding Custom Help Files

**help_on**

Flag that determines whether the help button should be displayed.

Object Type: Template variable

Default value: 0 (integer)

Included In: LoginPage.html, SecondaryLoginPage.html

LoginPage.htmlUsage: Optional

Example: The following code displays a help button if the help_on variable is true.

```
<% IF help_on %>
  <input type='submit' name='help' value='<%= help %>'
  onclick='window.open("welcome.cgi?p=help", "wndHelp",
  "height=400,width=500,resizeable=yes,scrollbars=yes"); return false;'>
<% END %>
```

Related Documentation: Adding Custom Help Files

**HideRemed()**

Submits the hidden form when the end-user clicks the Hide Remediation button.

Object Type: JavaScript function

Included In: Remediate.html

Usage: Required

Example: The following code calls the HideRemed() function:

```
<% IF showRemedOption %>
  <input name="btnHideRemed" type="button"
  value="<%= textRemedOption %>">
  <input name="btnHideRemed" type="hidden" value="true"/>
  <input name="btnHideRemed" type="button"
  value="<%= textRemedOption %>">
  <input name="btnHideRemed" type="hidden" value="true"/>
  <% HC_REMED_SHOW_ADV_PREFS %>
<% END %>
```

### Home

References the top-level directory in the zip file.

**Object Type**
Template variable

**Included In**
All templates

**Usage**
You can use this variable or the double-dot (..) convention to reference the top-level directory.

**Example**
Both of the following references are valid:

```
<% Home %>/images/logo.gif
../images/logo.gif
```

**Related Documentation**
Using the Template Toolkit

### I18N_UAC_DOWNLOAD_AND_INSTALL_64_BIT

Contains the link to download and install Odyssey Access Client on 64-bit Windows manually.

**Object Type**
Template variable

**Included In**
AgentInstall.thtml

**Usage**
Optional (Required for Policy Secure).

**Example**
Both of the following references are valid:

```
<%IF I18N_UAC_DOWNLOAD_AND_INSTALL_64_BIT  %>
<span><% I18N_UAC_DOWNLOAD_AND_INSTALL_64_BIT FILTER verbatim %></span>
<%END%>
```

**Related Documentation**

### ie

Boolean variable that indicates whether or not the browser is the Microsoft Internet Explorer.

**Object Type**
Template variable

**Included In**
LoginMeeting.thtml

**Usage**
Required

**Example**
The following example checks to see if the browser is Internet Explorer, and if it is, enables ActiveX.

```
<% IF ie %>
<OBJECT
<% activex_ini FILTER verbatim%>
</OBJECT>
<% END %>
```
Related Documentation  ie_winxp

IEonWindows

Boolean indicating if the browser is Internet Explorer and the OS is Windows.

Object Type  Template variable

Included In  AgentInstall.html, AgentlessInstall.html

Usage  Optional. The variable value is 1 if the browser is Internet Explorer and the OS is Windows.

Example  The following example checks to see if the browser is Internet Explorer, and if it is, enables ActiveX.

    <% IF IEonWindows %> 

Related Documentation  ie_winxp

Boolean indicating if the browser is Internet Explorer and the OS is Windows XP.

Object Type  Template variable

Included In  PleaseWait.html

Usage  Optional. The variable value is 1 if the browser is Internet Explorer and the OS is Windows XP.

Example  

    <% IF ie_winxp %> 

Related Documentation

install

Boolean indicating whether or not the meeting applet should be installed.

Object Type  Template variable

Included In  MeetingAppletInstaller.html, MeetingRun.html

Usage  Required

Example  

    <% IF install %>

Related Documentation

- MeetingAppletInstaller.html
- MeetingRun.html
| **instanceld** |  
| --- | ---  
| Contains the ID of the current meeting instance. |  
| **Object Type** | Template variable  
| **Included In** | MeetingAppletInstaller.html  
| **Usage** | Required  
| **Example** |  
| `<PARAM NAME="Parameter0" VALUE="meeting_id=<% instanceld %>; 
user_name=<%username%>;cert_md5=<%cert_md5%>; 
ncp_read_timeout=<%ncp_read_timeout%>">` |  
| **Related Documentation** | MeetingAppletInstaller.html  
| **instructions** |  
| Instructional text displayed to the user on the login page. |  
| **Object Type** | Template variable  
| **Default Value** | “Please sign in to begin your secure session”  
| **Included In** | LoginPage.html  
| **Usage** | Optional  
| **Example** |  
| `<% instructions FILTER verbatim %>` |  
| **Related Documentation** | None  
| **isLinux** |  
| Boolean indicating whether or not the end-user’s operating system is a Linux system. |  
| **Object Type** | Template variable  
| **Included In** | AgentInstall.html, PleaseWait.html, Logout.html  
| **Usage** | Optional  
| **Example** |  
| `<% isLinux %>` |  
| **Related Documentation** | PleaseWait.html / PleaseWait-mobile-webkit.html / PleaseWait-ipad.html  
| **isMac** |  
| Boolean indicating whether or not the end-user’s operating system is a Macintosh system. |  
| **Object Type** | Template variable  
| **Included In** | PleaseWait.html  
| **Usage** | Required |
### isSAMEnabled

Boolean indicating whether or not JSAM application is enabled and running.

**Object Type**
Template variable

**Included In**
Logout.html

**Usage**
Optional. Allows the end-user to close the JSAM window.

**Example**

The following code checks to see if JSAM is enabled and running. If it is, then the system uses the JavaScript below to close the JSAM window:

```
<% IF isSAMEnabled %>
<!-- terminate JSAM -->
<SCRIPT LANGUAGE="JavaScript">
var win = "<%jsamwindowid %>";
win = win.replace(/[^0-9a-zA-Z]/g,"a");
w = window.open("", win, "height=1,width=1");
if (w) {
    w.bSessionClose = true;
    w.close();
}
</SCRIPT>
<% END %>
```

**Related Documentation**
Logout.html / Logout-mobile-webkit.html / Logout-ipad.html

### isUAC

Boolean indicating whether the server is for Policy Secure.

**Object Type**
Template variable

**Included In**
PleaseWait.html, Remediate.html

**Usage**
Required.

**Example**

```
<% isUAC %>
```

**Related Documentation**
Logout.html / Logout-mobile-webkit.html / Logout-ipad.html

### keyboard.js

Enables keyboard types of functions for a mouseless environment.

**Object Type**
JavaScript file
Included In
LoginPage.thtml (Kiosk)

Usage
Required in Kiosk zip. This template displays an error if the user signs out, if the user’s session
times out, or if Connect Secure uninstalls Host Checker or Cache Cleaner from the user’s
system. For more detailed information, see “Logout.html / Logout-mobile-webkit.html / Logout-ipad.html”. You must always include this template in your Kiosk.zip file.

Example
None

Related Documentation
LoginPage.html / LoginPage-mobile-webkit.html / LoginPage-ipad.html

**labelInstructions**

Instructions tag, indicating a location for instructions to appear on the page.

Object Type
Template variable

Included In
Remediate.html

Usage
Optional

Example
The following code loops through a list of failed policies, displays each policy name and
remediation instructions:

```
<% IF showPolicies %>
<% FOREACH failedPolicy = listFailedPolicies %>
<% loop.count %>
<% failedPolicy.name %>
<% labelInstructions %>
<% failedPolicy.remediation FILTER verbatim%>
<% END %>
```

Related Documentation
Remediate.html / Remediate-mobile-webkit.html / Remediate-ipad.html

**listFailedPolicies**

Template Toolkit hash value of failed policies (keys: name and remediation).

Object Type
Template variable

Included In
Remediate.html, Remediate-ppc.html, Remediate-ipad.html, Remediate-mobile-webkit.html

Usage
Required

Example
The following example checks to see if the page should show failed policies and, if so, loops
through the list of failed policies (the hash list listFailedPolicies) and displays the name of each
policy along with label instructions and remediation instructions:

```
<% IF showPolicies %>
<% FOREACH failedPolicy = listFailedPolicies %>
<% loop.count %>
<% failedPolicy.name %>
<% labelInstructions %>
<% failedPolicy.remediation FILTER verbatim%>
```
**Related Documentation**

<table>
<thead>
<tr>
<th>Loading</th>
<th>Remediate.thtml / Remediate-mobile-webkit.thtml / Remediate-ipad.thtml</th>
</tr>
</thead>
</table>

**locale**

Returns a locale that applies to an applet the page is attempting to upload.

**Object Type**

Template variable

**Included In**

Logout.thtml, MeetingAppletInstaller.thtml, MeetingTrouble.thtml

**Usage**

Required

**Example**

The following code sets the HTML form locale based on the template variable:

```html
<PARAM NAME="locale" VALUE="<% locale %>">
```

**Related Documentation**

safari

**loginmode**

Passes the user’s mode to the server using a hidden value.

**Object Type**

HTML form field

**Included In**


**Usage**

Required. Typically names a template directive that returns the user’s login mode from a server.

**Example**

The following code assigns the secure ID loginmode to the form field:
<INPUT type=hidden name="loginmode" value="<% secid_loginmode %>">

Related Documentation  secid_loginmode

Login()

Verifies that realms exist, remembers currently selected authentication realms, and establishes the user’s time zone.

Object Type  JavaScript function

Included In  LoginPage.html

Usage  Required. This function includes two actions:

- **Remember the selected authentication realm**—Use this action within the function if you want to allow users to select from multiple realms in the sign-in page.

- **Establish the user’s time zone**—Required. The function uses the tz_offset (time zone offset) variable and getTimezoneOffset() function to determine the user’s time zone.

Example

```javascript
function Login() {
    // Remember currently selected auth realm
    if (document.frmLogin.realm != null &&
        document.frmLogin.realm.type == "select-one") {
        SetLastRealm(
            document.frmLogin.realm.options
            [document.frmLogin.realm.selectedIndex].text);
    }

    if (document.frmLogin.tz_offset != null) {
        var wdate = new Date (95, 12, 1);
        var sdate = new Date (95, 6, 1);
        var winter = (-1) * wdate.getTimezoneOffset();
        var summer = (-1) * sdate.getTimezoneOffset();
        document.frmLogin.tz_offset.value = winter < summer ? winter : summer;
    }

    return true;
}
```

Related Documentation  LoginPage.html / LoginPage-mobile-webkit.html / LoginPage-ipad.html

LoginMeeting.html

Displays the user credentials form to enable a user to sign in to a meeting.

Object Type  Template file

Usage  Required in the uploaded Meeting.zip. This template provides a page on which the end-user can enter credentials to gain access to a meeting.

The form action must specify login_meeting.cgi. The template includes the JavaScript function grab(), which checks to see which type of browser the user is running, and if ActiveX, Java, and cookies are enabled, depending on the browser type.
You can modify the alert contained in the required JavaScript. The alert is:

```javascript
alert("You must enable cookies through your Web browser in order to sign into the meeting.");
```

### Required Variables

- "activex_ini"
- "cboxuser"
- "focus"
- "Home"
- "ie"
- "mid"
- "mid_param"
- "occurrence"
- "username"
- "Remember my name" (not required, but recommended)
- "Security token" (not required, but recommended)

### Example

None

### Related Documentation

- **grab()**

**LoginPage.thtml / LoginPage-mobile-webkit.thtml / LoginPage-ipad.thtml**

Calls the RSA Soft ID client, enabling the user to sign into Connect Secure using Soft ID authentication.

### Object Type

Template

### Usage

Required. You must always include this template in your zip file. You must always include **LoginPage.thtml** in your zip file, even if you are using an authentication server that does not require a username or password. This is the standard Connect Secure sign-in page that collects the user’s name, password, and authentication realm, and displays an error if authentication fails. For information about hiding this page from users authenticated against a(n):

- Anonymous server, see "AnonymousAuthentication"
- Certificate server, see "CertificateAuthentication"
- eTrust SiteMinder server using client-side certificate authentication, see "Login()"

**LoginPage.thtml** contains the following suggested form definition. In this form definition, most elements are required:

```html
<form name="frmLogin" action="login.cgi" method="post" autocomplete="off"
```
onsubmit="return login()"

**Required Variables**

- "LoginPageErrorMessage"

**Example**

None

**Related Documentation**

- Login()

## LoginPageErrorArgs

Contains the arguments for an error or message.

**Object Type**

Template variable

**Included In**

GraceLoginUsed.thtml, PasswordExpiration.thtml

**Usage**

Optional

**Example**

In the following example, GraceLoginsRemaining is the argument to error code 2016. The directive that contains LoginPageErrorArgs.GraceLoginsRemaining returns the number of remaining login attempts Connect Secure allows a user.

```html
<% IF LoginPageErrorCode == 2016 %>
Your password has expired. A grace login was used.
You have <% LoginPageErrorArgs.GraceLoginsRemaining %> grace logins remaining.
<% ELSE %>
<% LoginPageErrorMessage %>
<% END %>
```

**Related Documentation**

LoginPageErrorCode

## LoginPageErrorCode

Codes indicating errors that you can display to users.

**Object Type**

Template variable

**Included In**

ExceededConcurrent.thtml, GraceLoginUsed.thtml, LoginPage.thtml, PasswordExpiration.thtml, SSL.html

**Usage**

Optional. Although this template variable is optional, you should leave intact the template variables that are already included in the various templates. You can modify many of the error messages, but you must leave the codes as defined. Connect Secure returns these codes on various conditions. Some may be no more than confirmation messages. Others are conditions to which the user needs to respond.

**Example**

The following example shows how you can use the Template Toolkit conditional statements to define two alternate error conditions. In this example, the different error codes display appropriate HTML pages. If Connect Secure does not return either of the error codes, the user never sees the two error pages.
Table 6 lists possible errors that the user may see on this page as well as the corresponding text returned by the LoginPageErrorMessage variable:

Table6: Login Error Messages, Codes, and Related Notes

<table>
<thead>
<tr>
<th>Error message</th>
<th>Error Code</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invalid Username or Password</td>
<td>1002</td>
<td>Wrong user credentials.</td>
</tr>
<tr>
<td>This login requires a digital certificate.</td>
<td>1003</td>
<td>Client certificate not passed for authentication.</td>
</tr>
<tr>
<td>The digital certificate is invalid.</td>
<td>1004</td>
<td></td>
</tr>
<tr>
<td>Only admins are permitted to login.</td>
<td>1006</td>
<td></td>
</tr>
<tr>
<td>Logins are not permitted from this IP address.</td>
<td>1007</td>
<td>The source IP failed the IP Restriction check.</td>
</tr>
<tr>
<td>Logins are not permitted using this browser</td>
<td>1008</td>
<td>The User Agent header failed the User Agent Restriction check.</td>
</tr>
<tr>
<td>There are no auth servers defined for this user.</td>
<td>1009</td>
<td>Displayed if a realm is not sent to Connect Secure. This error is only displayed if a coding error exists in LoginPage.thtml.</td>
</tr>
<tr>
<td>Exceeded the number of concurrent users.</td>
<td>1010</td>
<td>Exceeded the number of concurrent users based on the user license plus a 10% allowance.</td>
</tr>
<tr>
<td>The IP has been blocked due too many concurrent sign-in attempts.</td>
<td>1011</td>
<td></td>
</tr>
<tr>
<td>The password is too short.</td>
<td>1012</td>
<td>The password failed the Password Restriction check. When using eTrust auto-signon, the end-user should never see this event.</td>
</tr>
<tr>
<td>Not using SSLv3</td>
<td>1015</td>
<td>This site requires Secure Sockets Layer (SSL) protocol version 3.</td>
</tr>
<tr>
<td>This site requires Strong ciphers.</td>
<td>1016</td>
<td></td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>1017</td>
<td>Too many users have logged in. The number of concurrent users signed in to the system has exceeded the system limit but is still within the 10% allowance.</td>
<td></td>
</tr>
<tr>
<td>1018</td>
<td>Sign on for administrators is disabled. You can try again in a few minutes. Displayed when a super administrator is signed in and Connect Secure is in recovery mode.</td>
<td></td>
</tr>
<tr>
<td>1019</td>
<td>Your New PIN has been saved. Please make sure to remember it. Only used with ACE authentication.</td>
<td></td>
</tr>
<tr>
<td>1020</td>
<td>Password Change Success Used with the password management feature.</td>
<td></td>
</tr>
<tr>
<td>1021</td>
<td>Account Disabled Used with the password management feature.</td>
<td></td>
</tr>
<tr>
<td>1022</td>
<td>Account Locked Out Used with the password management feature.</td>
<td></td>
</tr>
<tr>
<td>1023</td>
<td>Account Expired Used with the password management feature.</td>
<td></td>
</tr>
<tr>
<td>1024</td>
<td>Connect Secure session has been terminated. End-user explicitly logged off.</td>
<td></td>
</tr>
<tr>
<td>1025</td>
<td>You do not have permission to login. This event can be triggered when the client machine does not pass any realm restrictions (hence, no realm), or does not satisfy any endpoint security policies (HC/CC).</td>
<td></td>
</tr>
<tr>
<td>1027</td>
<td>This realm has reached the max session limit. Exceeded maximum number of users for this authentication realm.</td>
<td></td>
</tr>
<tr>
<td>1028</td>
<td>Unlicensed Feature License required to use this access method is missing.</td>
<td></td>
</tr>
<tr>
<td>1029</td>
<td>Access denied. Please try signing in again using a host name (for example, <a href="https://department.yourcompany">https://department.yourcompany</a>) instead of an IP address (such as <a href="http://10.11.12.13">http://10.11.12.13</a>) Only used with eTrust SiteMinder authentication.</td>
<td></td>
</tr>
<tr>
<td>1030</td>
<td>You do not have the correct privileges to access the system. Please contact your administrator for more information. The end-user is not allowed to sign in because the role mapping rules of the authentication realm do not map the end-user to any role.</td>
<td></td>
</tr>
<tr>
<td>1032</td>
<td>Certificate has been revoked. The client certificate has been revoked.</td>
<td></td>
</tr>
</tbody>
</table>
Cannot determine the status of your certificate. 1033

Connect Secure encountered a failure while performing an OCSP check.

You must use your source site’s Intersite Transfer Service to access this resource. 1034

Connect Secure cannot access any SAML assertion while performing SAML SSO. The user is prompted to access the source site’s Intersite Transfer Service to continue.

The number of concurrent Advanced Endpoint Defense (Malware Protection) users signed into the system has exceeded the system limit. 1035

This event will be triggered only when the feature is turned ON. The system limit is built-in 25 user license plus any purchased Advanced Endpoint Defense user license.

Related Documentation

LoginPageErrorMessage

LoginPageErrorMessage

Connect Secure error message text that you can display to users.

Object Type

Template variable

Included In


Example

The following code prompts the user to re-enter credentials or to sign in at another time. Note that the messages can be whatever you want them to be:

```html
<% IF LoginPageErrorCode == 1002 %>
<b>Your username or password is incorrect. Please reenter your credentials.</b>
<%ELSIF LoginPageErrorCode == 1006 %>
<b>The system is undergoing maintenance. Only administrators are allowed to sign in at this time.</b>
<% END %>
```

The following code tests the pleasewait condition and, if true, displays the Login page error message that occurs as a result:
if (document.getElementById('pleasewait1')) {
    document.getElementById('pleasewait1').innerHTML =
    "<% LoginPageErrorMessage FILTER verbatim%>";
}

Related Documentation  LoginPageErrorCode

### login_required

Boolean indicating whether or not login to the meeting session is required.

**Object Type**  Template variable

**Included In**  MeetingRunJava.thtml

**Usage**  Required

**Example**  The following example checks to see if login is required. If yes, then the page redirects the user back to the meeting sign-in page:

```html
<% IF login_required %>
    redirectParent();
<% END %>
```

Related Documentation  MeetingRunJava.thtml

### Logout.html / Logout-mobile-webkit.thtml / Logout-ipad.thtml

Displays an error if the user signs out, if the user’s session times out, or if Connect Secure uninstalls Host Checker or Cache Cleaner from the user’s system.

**Object Type**  Template

**Usage**  Required. You must always include `Logout.html` in your zip file.

This template contains JavaScript for detecting, stopping, installing, and uninstalling the Host Checker and Cache Cleaner components, images and text to display to users while Connect Secure performs these actions, and JavaScript that closes and opens the J-SAM window.

This template contains JavaScript for detecting, stopping, installing, and uninstalling Host Checker components, as well as images and text to display to users while the performs these actions.

This template also includes variables telling users to wait while components install, a link that users can use to sign back into Connect Secure, and error message variables.

When configuring the `LoginPageErrorMessage` variable included in this template, note that the text that Connect Secure displays to users corresponds with a code returned by `LoginPageErrorCode`. You cannot change this text on Connect Secure, but you can include code in your template to display different text based on the error code returned by the `LoginPageErrorCode` variable. For example:
<% IF LoginPageErrorCode == 1001 %>
<b> Your secure gateway session has ended due to inactivity.</b>
<% END %>

Described below are the possible errors that the LoginPageErrorCode may return as well as the corresponding text returned by the LoginPageErrorMessage variable:

Table 7: Additional Login Error Messages and Codes

<table>
<thead>
<tr>
<th>Error Message</th>
<th>Error Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Session Timeout Reached</td>
<td>1000</td>
</tr>
<tr>
<td>Idle Time Out</td>
<td>1001</td>
</tr>
</tbody>
</table>

For detailed information about the JavaScript and variables described here, see comments in the template.

Required Variables

- "delivery_mode"
- "geckoBrowser"
- "locale"
- "LoginPageErrorMessage"
- "nc_logging"
- "Netscape4xx"
- "pleasewaitLogoutJSCode"
- "PleaseWaitObjectCC"
- "pleasewaitWin32"
- "safari"
- "Setupappversion"
- "UninstallCC"
- delivery_mode
- LoginPageErrorMessage
- signinAgain
- signinAgainUrl
- pleasewaitMac
- setup_ini
- fallback_setup_applet_vitals

Example

None
meeting_ini

Contains an Active-X control.

Object Type Template variable
Usage Required if running Internet Explorer on a Windows system. This variable contains the name of the Active-X control that must be loaded to run Pulse Secure Collaboration in the browser.

Required Variables
- activex_ini
- FILTER verbatim

Example
<%meeting_ini FILTER verbatim%>

Related Documentation
- MeetingRun.thtml
- MeetingSelect.thtml

MeetingAppletInstaller.thtml

Enables Connect Secure to display list of and install meeting-related applets.

Object Type Template
Usage Required. The page displays troubleshooting messages if the applets do not install. You cannot change the text, as Connect Secure generates the troubleshooting messages. You can, however, add links if you want to provide additional meeting-related applets to your users. These applets are not covered by the certificate validation Connect Secure performs on Pulse Secure-supplied applets.

Required Variables
- "applet_ini"
- "cert_md5"
- "install"
- "instanceId"
- "locale"
- "ncp_read_timeout"
- "username"
- "startSessionReDir"
- text
- "troubleshooting"
- uninstallReDir

Example None

Related Documentation
- MeetingRun.thtml
MeetingRun.html

Tests Sun JVM compatibility.

- **Object Type**: Template
- **Usage**: Required
- **Required Variables**:
  - errors
  - "expired"
  - "Home"
  - "message"
  - "mid"
  - "occurrence"
  - title
  - "type"
  - "warnings"

- **Example**: None
- **Related Documentation**: LoginMeeting.html

MeetingRunJava.html

Launches the Meeting Java client.

- **Object Type**: Template
- **Usage**: Required
- **Required Variables**:
  - "appstr"
  - href
  - "login_required"
  - url

- **Example**: None
- **Related Documentation**: LoginMeeting.html

meetings

Returns an array of meetings that the MeetingSelect.html template uses as a way to cycle through logins to active meetings.

- **Object Type**: Template variable
<table>
<thead>
<tr>
<th>Included In</th>
<th>MeetingSelect.thtml</th>
</tr>
</thead>
<tbody>
<tr>
<td>Usage</td>
<td>Required. You should not modify or delete this variable.</td>
</tr>
<tr>
<td>Example</td>
<td>The following code returns a list of instantiated meetings:</td>
</tr>
</tbody>
</table>

```jsp
<% IF rows %>
<% FOREACH meeting = meetings %>
<input type="button" value=" &gt; " onclick="self.location='/dana-na/meeting/<%signin%>login_meeting.cgi?mid=<%mid%>&amp;occurrence=<%meeting.3%>&amp;params%'"">
<%started%>
<%meeting.0%>
<%meeting.1%>
<%meeting.2%>
<% END %>
<%END%>
```

<table>
<thead>
<tr>
<th>Related Documentation</th>
<th>rows</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MeetingSelect.thtml</strong></td>
<td>Enables user to select recurring meeting occurrences.</td>
</tr>
<tr>
<td>Object Type</td>
<td>Template</td>
</tr>
<tr>
<td>Usage</td>
<td>Required</td>
</tr>
<tr>
<td>Required Variables</td>
<td></td>
</tr>
<tr>
<td>• &quot;meetings&quot;</td>
<td></td>
</tr>
<tr>
<td>• &quot;mid&quot;</td>
<td></td>
</tr>
<tr>
<td>• &quot;params&quot;</td>
<td></td>
</tr>
<tr>
<td>• &quot;rows&quot;</td>
<td></td>
</tr>
<tr>
<td>• &quot;started&quot;</td>
<td></td>
</tr>
<tr>
<td>• text</td>
<td></td>
</tr>
<tr>
<td>• title</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Example</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Related Documentation</td>
<td>LoginMeeting.thtml</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>MeetingTestJava.thtml</strong></th>
<th>Tests Sun JVM compatibility.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object Type</td>
<td>Template</td>
</tr>
<tr>
<td>Usage</td>
<td>Required</td>
</tr>
<tr>
<td>Required Variables</td>
<td></td>
</tr>
<tr>
<td>• &quot;mid&quot;</td>
<td></td>
</tr>
</tbody>
</table>
MeetingTestJS.html

This is the first page in the platform compatibility checker sequence. Tests JavaScript compatibility.

Object Type
Template
Usage
Required. If the end-user’s browser does not support JavaScript, or if they need to enable JavaScript functionality in their browser, this template displays appropriate messages depending on the browser type and the problem, and offers hints to address the problem.

Required Variables
- "activex_ini"
- "mid"
- "mid_param"
- "win32"

Example
None

Related Documentation
LoginMeeting.html

MeetingTestMSJava.html

Tests Microsoft JVM compatibility.

Object Type
Template
Usage
Required

Required Variables
- "applet_ini"
- "mid"
- "mid_param"

Example
None

Related Documentation
LoginMeeting.html

MeetingTestResult.html

Displays Meeting test results.

Object Type
Template
Usage
Required

Required Variables
### MeetingTrouble.thtml

Template Page for the Meeting Test Result page.

**Object Type** Template

**Usage** Required

**Required Variables**
- "Home"
- "locale"
- "mid"
- "midStr"
- "occurrence"
- "startSessionReDir"
- "type"
- "uninstall_action"
- "uninstallReDir"

**Example** None

**Related Documentation**
- MeetingTestMSJava.thtml
- MeetingTestJava.thtml
- MeetingTestJS.thtml
- MeetingTestResult.thtml

---

### message

Returns a message based on a server response.

**Object Type** Template variable

**Included In** MeetingRun.thtml, MeetingTestResult.thtml, SecondaryLoginPage.thtml

**Usage** Optional

**Example**
```
<% IF message %>
```
Related Documentation

- MeetingRun.html
- MeetingTestResult.html

mid

Contains the meeting ID.

Object Type: Template variable

Included In: LoginMeeting.html, MeetingRun.html, MeetingTestJava.html, MeetingTestJS.html, MeetingTestMSJava.html, MeetingTrouble.html, PasswordChange.html

Usage: Required

Example: The following code assigns the meeting ID to a form input object:

```html
<input type="hidden" name="mid" value="<% mid %>">
```

Related Documentation

- LoginMeeting.html
- MeetingRun.html
- MeetingTestJava.html
- MeetingTestJS.html
- MeetingTestMSJava.html
- MeetingTrouble.html
- PasswordChange.html / PasswordChange-mobile-webkit.html / PasswordChange-ipad.html

mid_param

Contains the URL parameters for the platform compatibility checker page.

Object Type: Template variable

Included In: LoginMeeting.html, MeetingTestJava.html, MeetingTestJS.html, MeetingTestMSJava.html

Usage: Required

Example: The following code passes the meeting ID URL to the JavaScript compatibility CGI:

```html
<a href="/dana-na/meeting/<% signin %>meeting_testjs.cgi<% mid_param %>">
```
### Check Meeting Compatibility

**midStr**
- Contains the URL parameter for the meeting.
- **Object Type**: Template variable
- **Included In**: MeetingTrouble.thtml
- **Usage**: Required
- **Example**: The following line shows how the `midStr` variable is appended to a CGI URL, to send the user to the correct compatibility checker for the current meeting.

```
"Use the <a href="/dana-na/meeting/meeting_testjs.cgi<% midStr %>">compatibility checker</a> to see if you need to upgrade any of your components."
```

**Related Documentation**: MeetingTrouble.thtml

**MinimumPasswordLength**
- Argument for error code 2006. The argument is the minimum length of password required.
- **Object Type**: Template variable
- **Included In**: PasswordChange.thtml
- **Usage**: Optional
- **Example**: None
- **Related Documentation**: LoginPageErrorCode
  - PasswordChange.thtml / PasswordChange-mobile-webkit.thtml / PasswordChange-ipad.thtml

**msg**
- Contains any extra remediation related default instructions.
- **Object Type**: Template variable
- **Included In**: RemEDIATE.html
- **Usage**: Optional
- **Example**: The following code checks to see if a heading is to be shown. If yes, then the page displays the heading, followed by any remediation message that might be available.
### nc_loging

Initiates Network Connect logging.

**Object Type**  
Template variable

**Included In**  
Logout.html

**Usage**  
Optional. Available when the user signs out of a session from a Macintosh Safari browser. Network Connect logging is initiated and the log captured.

**Example**  
None

**Related Documentation**  
safari

### ncp_read_timeout

Determines the read timeout value over the Network Control Protocol.

**Object Type**  
Template variable

**Included In**  
MeetingAppletInstaller.html

**Usage**  
Required. Passed to MeetingAppletInstaller.html when it installs a meeting applet.

**Example**

```html
<% IF install %>
<APPLET
    <% applet_ini FILTER verbatim %>
    <PARAM NAME="Parameter0" VALUE="meeting_id=<%instanceId%>;user_name=<%username%>;cert_md5=<%cert_md5%>;ncp_read_timeout=<%ncp_read_timeout%>">
    <PARAM NAME="StartSessionReDir" VALUE="<%startSessionReDir%>">
    <PARAM NAME="UninstallReDir" VALUE="<%uninstallReDir%>">
    <PARAM NAME="locale" VALUE="<%locale%>">
</APPLET>
```

**Related Documentation**  
MeetingAppletInstaller.html

### netacekey

Passes the user's eTrust ACE key to the server using a hidden value.

**Object Type**  
HTML form field

**Included In**  

**Usage**  
Required. The form assigns the netesecid_key value to netacekey.
Example

The following example shows the use of `netacekey` in the `INPUT` statement:

```html
<INPUT type=hidden name="neteacekey" value="<% netesecid_key %>">
```

**Related Documentation**

`netesecid_key`

`netesecid_key`

Contains the user's eTrust ACE key.

**Object Type**

Template variable

**Included In**


**Usage**

Required. The value is passed by the page to the server.

**Example**

The following example shows the use of `netesecid_key` in the `INPUT` statement:

```html
<INPUT type=hidden name="neteacekey" value="<% netesecid_key %>">
```

**Related Documentation**

`netacekey`

`netesecid_loginmode`

Returns a value of Next Token mode or New PIN mode.

**Object Type**

Template variable

**Included In**


**Usage**

Required. You must send this value to the server, but if you do not want to expose it to the user, you may send it as a hidden form field.

**Example**

The following hidden `INPUT` statement sets the login mode based on the `netesecid_loginmode` variable.

```html
<INPUT type=hidden name="loginmode" value="<% netesecid_loginmode %>">
```

**Related Documentation**

`loginmode`

`netesecid_pinerr`

Error code and corresponding message that informs the user that he mis-entered his PIN.

**Object Type**

Template variable

**Included In**

`SM-NewUserPin.thtml`
Usage: Optional

Example: The following code checks the value of `netesecid_pinerr`, a boolean, to see if the PIN the user enters matches the PIN on the authentication server:

```html
<% IF netesecid_pinerr == 0 %>
  <IMG border='0' src='imgs/custom-loginfo.gif' alt='Info' width='21' height='20'>
  New PIN Assignment
<% ELSIF netesecid_pinerr == 1 %>
  The Two PINs Entered Do Not Match
<% END %>
```

Related Documentation: SM-NewUserPin.thtml

---

**netesecid_realm**

The user’s realm.

Object Type: Template variable


Usage: Optional

Example: The following code shows the configuration of the page based on various server-provided values:

```html
<FORM name="frmNewPin" action="login.cgi" method="POST"
  autocomplete=off>
  <INPUT type=hidden name="loginmode" value="<% netesecid_loginmode %>">
  <INPUT type=hidden name="realm" value="<% netesecid_realm %>">
  <INPUT type=hidden name="neteacekey" value="<% netesecid_key %>">
</FORM>
```

Related Documentation:
- SM-NewUserPin.thtml
- SM-NextToken.thtml / SM-NextToken-mobile-webkit.thtml / SM-NextToken-ipad.thtml

---

**netesecid_username**

The user’s username.

Object Type: Template variable


Usage: Optional

Example: The following `INPUT` statement assigns the `netesecid_username` to the form `username` field:

```html
<INPUT type="text" name="username" value="<% netesecid_username %>"/>
```
Related Documentation  netesecid_realm

netesecidactionCancel

Cancels a eTrust secure ID action currently in progress.

Object Type  HTML form field


Usage  Required. The value of either this field or of netesecidactionEnter must be sent to the server:

    <INPUT type="submit" value="Cancel" name="netesecidactionCancel">

Related Documentation  netesecidactionEnter

netesecidactionEnter

Initiates the submission of the current eTrust secure ID action.

Object Type  HTML form field


Usage  Required. The value of either this field or of netesecidactionCancel must be sent to the server.

Example  

    <INPUT type="submit" value="Enter" name="netesecidactionEnter">

Related Documentation  netesecidactionCancel

Netscape4xx

Checks for Netscape version 4.x browser.

Object Type  Template variable

Included In  Logout.thtml

Usage  Required. Checks to see if the user’s browser is Netscape version 4.x. If so, then, on logout, error messages are passed to the page in a compatible way.

Example  None

Related Documentation  Logout.thtml / Logout-mobile-webkit.thtml / Logout-ipad.thtml

NewPin.thtml / NewPin-mobile-webkit.thtml / NewPin-ipad.thtml

Prompts ACE users to enter a new PIN or create a system-generated PIN before signing into Connect Secure.

Object Type  Template
Usage Required. When configuring the secid_pinselectmode variable (which sets the user’s PIN select mode), note that possible values include:

Table8: PIN Error Messages

<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>User must use the system-generated PIN. He cannot enter his own PIN.</td>
</tr>
<tr>
<td>1</td>
<td>User may enter his own PIN or use the system-generated PIN.</td>
</tr>
<tr>
<td>2</td>
<td>User must enter his own PIN. He cannot use the system-generated PIN.</td>
</tr>
</tbody>
</table>

When configuring the secid_pinserr variable (which stores the error code and message that informs the user if he mis-entered his PIN), note that possible values include:

Table9: Additional PIN Error Messages

<table>
<thead>
<tr>
<th>Code</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>New PIN is required.</td>
</tr>
<tr>
<td>1</td>
<td>The two PINs entered do not match.</td>
</tr>
<tr>
<td>2</td>
<td>PIN Format is invalid.</td>
</tr>
<tr>
<td>3</td>
<td>PIN length is invalid.</td>
</tr>
</tbody>
</table>

Required Variables

- “secid_contextid”
- “secid_loginmode”

Example None

NextToken.html / NextToken-mobile-webkit.html / NextToken-ipad.html

Prompts the user to verify his credentials by entering his SecureID token code.

Object Type Template

Usage Optional. Although this template is optional, you should always include it in the zip file that you upload to Connect Secure.

Required Variables

- “secid_contextid”
- “secid_loginmode”

Example None
## occurrence

Contains the recurring meeting occurrence ID.

<table>
<thead>
<tr>
<th><strong>Object Type</strong></th>
<th>Template variable</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Included In</strong></td>
<td>LoginMeeting.thtml, MeetingRun.thtml, MeetingSelect.thtml, MeetingTrouble.thtml</td>
</tr>
<tr>
<td><strong>Usage</strong></td>
<td>Required</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>The following code shows how the occurrence variable is used in an <strong>INPUT</strong> statement:</td>
</tr>
</tbody>
</table>

```html
<input type="hidden" name="occurrence" value="<% occurrence %>">
```

### Related Documentation

mid

## onsubmit

Event trigger in a form.

<table>
<thead>
<tr>
<th><strong>Object Type</strong></th>
<th>HTML form element</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Included In</strong></td>
<td>GeneratePin.thtml, LoginPage.thtml, NewPin.thtml, NextToken.thtml, SM-NewUserPin.thtml</td>
</tr>
<tr>
<td><strong>Usage</strong></td>
<td>Required if you are using Pulse Secure Collaboration. Returns the time zone offset value set by the <code>Login()</code> function.</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>The following code shows a typical use of onsubmit:</td>
</tr>
</tbody>
</table>

```html
onsubmit="return Login()"
```

### Related Documentation

None

## p

Specifies the page name as CASQUEAPP.

<table>
<thead>
<tr>
<th><strong>Object Type</strong></th>
<th>HTML form field</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Included In</strong></td>
<td>Defender.thtml</td>
</tr>
<tr>
<td><strong>Usage</strong></td>
<td>Required</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>The following code assigns the value CASQUEAPP to the form <strong>INPUT</strong> field p:</td>
</tr>
</tbody>
</table>

```html
<input type=hidden name="p" value="casqueapp">
```

### Related Documentation

secid_challenge

## PageErrorArgs

Contains the arguments for an error or message.

<table>
<thead>
<tr>
<th><strong>Object Type</strong></th>
<th>Template variable</th>
</tr>
</thead>
</table>
Included In: PasswordChange.html

Usage: Optional

Example: The following code checks to see if a password length error occurs, then displays an informational message:

```html
<% IF PageErrorCode == 2015 %>
New password must not be longer than
<% PageErrorArgs.MaximumPasswordLength %>
characters.
</%>
```

Related Documentation:
- PageErrorCode
- PasswordChange.html / PasswordChange-mobile-webkit.html / PasswordChange-ipad.html

**PageErrorCode**

Contains the error code in case of an error or message.

Object Type: Template variable

Included In: LoginPage.html, PasswordChange.html

Usage: Optional

---

**Table 10: Page Error Codes**

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Error message</th>
</tr>
</thead>
<tbody>
<tr>
<td>1020</td>
<td>Password Change Success</td>
</tr>
<tr>
<td>1021</td>
<td>Account Disabled</td>
</tr>
<tr>
<td>1022</td>
<td>Account Locked out</td>
</tr>
<tr>
<td>1023</td>
<td>Account Expired</td>
</tr>
<tr>
<td>2001</td>
<td>Password Mismatch</td>
</tr>
<tr>
<td>2002</td>
<td>Invalid Password</td>
</tr>
<tr>
<td>2003</td>
<td>Password Change not allowed</td>
</tr>
<tr>
<td>2004</td>
<td>Password too new</td>
</tr>
<tr>
<td>2006</td>
<td>Password too short</td>
</tr>
<tr>
<td>2007</td>
<td>Password too trivial</td>
</tr>
<tr>
<td>2008</td>
<td>Password exists in password history</td>
</tr>
<tr>
<td>2009</td>
<td>Password Expired Message</td>
</tr>
<tr>
<td>2010</td>
<td>Password must change at next logon message</td>
</tr>
<tr>
<td>2011</td>
<td>Password does not meet complexity requirements</td>
</tr>
</tbody>
</table>
Could not connect to auth server to change password

Password change failed

Password successfully changed

Password will expire in some amount of time

You have some number of grace logins.

For more errors and messages, see "LoginPageErrorCode" and "LoginPageErrorMessage".

Example

You cannot change the messages that appear in Table 10, but you can create your own messages and include them in your customized pages. For example:

```html
<% IF PageErrorCode == 2009 %>
Your password has expired. Please contact your system administrator.
<% END %>
```

Related Documentation

- LoginPageErrorCode
- LoginPageErrorMessage

**params**

Contains parameters required to sign in to a meeting.

**Object Type** Template variable

**Included In** MeetingSelect.thtml

**Usage** Required.

**Example** The following code fills the form with the names of available meeting instances:

```html
<% IF rows %>
<% FOREACH meeting = meetings %>
<input type="button" value="&gt;" onclick="self.location='/dana-na/meeting/<%signin%>login_meeting.cgi?mid=<%mid%>&occurrence=<% meeting.3 %>&params='">
<%started%>
<%meeting.0%>
<%meeting.1%>
<%meeting.2%>
<% END %>
<%END%>
```

**Related Documentation** MeetingSelect.thtml

**password**

Indicates the password for an optional JavaScript function in the Defender.thtml template.

**Object Type** JavaScript variable
Included In

Defender.html

Usage

Optional

Example

The header of Defender.html contains the following optional function to automatically place the cursor in the password field of the Defender page. If you choose to include this function in the header, you must also call it from the body of Defender.html. Also note that you should not change the JavaScript function name, form name, or form field name and that the form and form fields should have the same name.

```javascript
function FinishLoad() {
    document.frmDefender.password.focus();
}
```

Related Documentation

FinishLoad()

**password**

Sets an HTML form field as a password field.

Object Type

Input form field

Included In


Usage

This field is required for all types of authentication servers except anonymous servers and certificate servers.

Example

The following code assigns the password field:

```html
<input type="password" name="password">
```

Related Documentation

- LoginPage.html / LoginPage-mobile-webkit.html / LoginPage-ipad.html
- password2

**password**

Contains the security token, which is equivalent to the meeting password. If meeting password is configured to be sent in the invitation emails, a security token is added to the meeting URL. This allows the user to join the meeting with one-click, without the requirement to enter the meeting password manually.

Object Type

Template variable

Included In

LoginMeeting.html

Related Documentation

- LoginPage.html / LoginPage-mobile-webkit.html / LoginPage-ipad.html
**password2**

Verified password.

**Object Type**
HTML form field

**Included In**

**Usage**
Required. The `password2` and `password` fields pass the password and verified password that the user enters to the server for verification.

**Example**
The value of the following fields must match:

```html
<INPUT type="password" name="password">
<INPUT type="password" name="password2">
```

---

**PasswordChange.html / PasswordChange-mobile-webkit.html / PasswordChange-ipad.html**

Notifies the user that his password is due to expire and prompts him to change it.

**Object Type**
Template

**Usage**
Optional. Note that the `changePasswordTitle` variable in this template specifies whether the changed password is for the user’s primary or secondary authentication server. When you include this variable in your template, Connect Secure adds a title to the standard old/new/confirm password text boxes specifying whether the user must change his primary or secondary password.

Optional. When you include the adds a title to the standard old/new/confirm password text boxes specifying whether the user must change his password. The `showChangePasswordTitle` variable indicates whether or not to show the change password text. The variable value is true if primary and secondary authentication both exist in the login process. The `changePasswordTitle` variable is a regular text (title) message that indicates whether the password change page is for a primary or a secondary password. Without this variable, the password change page only displays the old/new/confirm password text boxes and a message above, suggesting that the user change password. With the `changePasswordTitle` variable, the password change page displays the old/new/confirm password text boxes, a message above suggesting that the user change password, and another heading at the top of that page, stating Change Primary (or Secondary) Password.

**Required Variables**
None

**Example**
None

**Related Documentation**
- `changePasswordTitle`
- `showChangePasswordTitle`
### PasswordComplexityMessage

The password complexity message.

**Object Type**
Template variable

**Included In**
PasswordChange.thtml

**Usage**

**Example**
The following code notifies the user that the password does not meet complexity requirements:

```html
<% IF PageErrorCode == 2011 %>
New password not complex enough. Must consist of <% PageErrorArgs.PasswordComplexityMessage %> characters.
<% END %>
```

**Related Documentation**
- PageErrorArgs
- PageErrorCode

### PasswordExpiration.thtml / PasswordExpiration-mobile-webkit.html / PasswordExpiration-ipad.html

Notifies the user that his password has expired and prompts him to change it.

**Object Type**
Template

**Usage**
Optional

**Required Variables**
"startPageLink"

**Example**
None

### PasswordExpirationFriendlyTime

Message displaying how soon password expires.

**Object Type**
Template variable

**Included In**
PasswordExpiration.html

**Usage**
Argument for error code 2015. Shows time in days, hours, and minutes.

**Example**
The following code checks for error code 2015, and displays a message containing the time remaining before the password expires:

```html
<% IF LoginPageErrorCode == 2015 %>
Password expires in<br>
<% LoginPageErrorArgs.PasswordExpirationFriendlyTime %>.  
<% END %>
```
Related Documentation

- LoginPageErrorArgs
- LoginPageErrorCode
- PasswordExpirationSeconds

PasswordExpirationSeconds

The number of seconds in which the current user’s password will expire.

Object Type: Template variable

Included In: PasswordExpiration.thtml


Example: The following code indicates that the user’s password will expire in a certain number of seconds:

```html
<% IF LoginPageErrorCode == 2015 %>
Password will expire in
<% LoginPageErrorArgs.PasswordExpirationSeconds %> seconds.
<% END %>
```

Related Documentation

- LoginPageErrorArgs
- LoginPageErrorCode
- PasswordExpirationFriendlyTime

PasswordHistoryLength

The length of the list of passwords the system has saved for a given end-user.

Object Type: Template variable

Included In: PasswordChange.thtml

Usage: Optional. Argument for error code 2008. Some authentication systems save passwords and do not allow an end-user to assign the same password twice in succession, within a certain amount of elapsed time, or as long as the password still exists in the password history list.

Example: The following code shows the use of PasswordHistoryLength:

```html
<% IF PageErrorCode == 2008 %>
New password must not repeat previous
<% PageErrorArgs.PasswordHistoryLength %> passwords.
<% END %>
```

Related Documentation: PasswordComplexityMessage
**pleasewait**

Default text printed to the page when either ActiveX or an applet is loading.

**Object Type**
Template variable

**Included In**
Remediate.html

**Usage**
Required

**Example**
The following code shows a table containing two directives. The first, the loading directive, tests whether or not ActiveX or an applet is loading. If yes, then the pleasewait directive displays a message, prompting the user to wait.

```html
<% loading %>
<% pleasewait %>
```

**Related Documentation**
- loading
- Remediate.html / Remediate-mobile-webkit.html / Remediate-ipad.html

**PleaseWait-mtg.html**

Meeting client launch intermediate page.

**Object Type**
Template

**Usage**
Required

**Required Variables**
- home
- meeting_ini
- startSessionReDir

**PleaseWait.html / PleaseWait-mobile-webkit.html / PleaseWait-ipad.html**

Displays a page asking the user to wait while application components install or uninstall.

**Object Type**
Template

**Usage**
Optional. This template differs from the standard PleaseWait.html template in the types and degree of validation and status checking performed while you wait. You may choose to customize this template if you have configured Host Checker or Cache Cleaner at the realm level. Use this page to control Host Checker and Cache Cleaner starts during pre-authentication and post-authentication.

This template contains JavaScript for detecting, stopping, installing, and uninstalling the Host Checker and Cache Cleaner components as well as images and text to display to users while Connect Secure performs these actions. This template also contains JavaScript that periodically checks the user's status to determine if Host Checker or Cache Cleaner is loaded on his system and redirects him to the sign-in page (welcome.cgi) if necessary.
For detailed information about the JavaScript and variables described here, see comments in the template.

**Required Variables**

- "LoginPageErrorMessage"
- "PleaseWaitObjectCC"
- isLinux
- isMAC
- userAgentTypeApple (PleaseWait-mobile-webkit.thtml)
- userAgentTypeAndroid (PleaseWait-mobile-webkit.thtml)
- serverdetails (PleaseWait-mobile-webkit.thtml)

**Example**

None

**Related Documentation**

- Logout.html / Logout-mobile-webkit.thtml / Logout-ipad.html

**pleasewait-ipad.html**

Displays a page asking the user to wait while application components install or uninstall. This is the iPad version of PleaseWait.html.

**Object Type**

Template

**Usage**

Optional. This template differs from the standard PleaseWait.html template in the types and degree of validation and status checking performed while you wait. You may choose to customize this template if you have configured Host Checker or Cache Cleaner at the realm level. Use this page to control Host Checker start during pre-authentication and post-authentication.

**Required Variables**

- showHeading
- delivery_mode
- pleasewaitObjectHC
- setup_classid
- setup_codebase

**Example**

None

**Related Documentation**

- 

**pleasewaitLogoutJSCode**

JavaScript code for pleasewait logout.

**Object Type**

Template variable

**Included In**

Logout.html
Usage

Required

Example

The following code shows the variable passed to the page verbatim, prior to the script section of the HTML page:

```
<html>
<head>
<meta http-equiv="Content-Language">
<meta http-equiv="Content-Type" content="text/html">
<meta name=robots content="none">
<title>Instant Virtual Extranet</title>

<% pleasewaitLogoutJSCode FILTER verbatim %>
<%IF pleasewaitObjectHC OR pleasewaitObjectCC OR pleasewaitWin32%>
<%IF pleasewaitObjectHC OR pleasewaitWin32%>
<SCRIPT LANGUAGE="JavaScript">

Related Documentation

- Logout.html / Logout-mobile-webkit.html / Logout-ipad.html
- PleaseWaitObjectCC
- pleasewaitWin32

PleaseWaitObjectCC

If a Cache Cleaner object is enabled, this variable contains the Cache Cleaner object tag.

Object Type

Template variable

Included In

Logout.html, PleaseWait.html

Usage

Required

Example

The following code checks to see if the Cache Cleaner object is enabled. If so, the code displays a message:

```
<%IF pleasewaitObjectCC %>
<span><div id="cc">Cache Cleaner Component</div></span>
<%END%>

Related Documentation

- Logout.html / Logout-mobile-webkit.html / Logout-ipad.html
- PleaseWait.html / PleaseWait-mobile-webkit.html / PleaseWait-ipad.html
- PleaseWaitObjectHC
- pleasewaitWin32
pleaseWaitObjectEP

If Odyssey Access Client is enabled, this variable contains the Odyssey Access Client object tag.

Object Type: Template variable
Included In: AgentInstall.thtml
Usage: Optional (Required for Policy Secure).
Example: The following code checks to see if the Pulse Secure client is enabled. If so, the code displays a message:

```html
<%IF pleaseWaitObjectEP %>
  <span><div id="ep">OAC Object</div></span>
<%END%>
```

PleaseWaitObjectHC

If a Host Checker object is enabled, this variable contains the Host Checker object tag.

Object Type: Template variable
Usage: Required
Example: The following code checks to see if the Host Checker object is enabled. If so, the code displays a message:

```html
<%IF pleaseWaitObjectHC %>
  <span><div id="cc">Host Checker Component</div></span>
<%END%>
```

Related Documentation
- PleaseWait.html / PleaseWait-mobile-webkit.html / PleaseWait-ipad.html
- PleaseWaitObjectCC
- pleasewaitWin32

pleaseWaitObjectJAM

If Pulse Secure client is enabled, this variable contains the Pulse Secure client object tag.

Object Type: Template variable
Included In: AgentInstall.thtml, PleaseWait.html
Usage: Required
**Example**

The following code checks to see if the Pulse Secure client is enabled. If so, the code displays a message.

```html
<%IF pleasewaitObjectJAM %>
<span><div id="jam"> Pulse Object</div></span>
<%END%>
```

**Related Documentation**

- PleaseWait.html / PleaseWait-mobile-webkit.html / PleaseWait-ipad.html
- pleasewaitWin32

---

**pleasewaitTO_msec**

Timeout value for HC to be completed on an end-user system.

**Object Type**

Template variable

**Included In**

PleaseWait-ppc.html

**Usage**

Optional (Required for mobile HC clients)

**Example**

```html
<% pleasewaitTO_msec %>
```

---

**pleasewaitWin32**

Boolean indicating that you must stop one or more Win32 components from the logout page.

**Object Type**

Template variable

**Included In**

Logout.html

**Usage**

Required. Set to 1 if any components need to be stopped.

**Example**

The following code checks to see if any of the components are causing Connect Secure to display a Please Wait message and if the components are java-based. If yes, the code calls the stopComponents() JavaScript function to stop the component.

```html
<% IF (pleasewaitObjectCC || pleasewaitObjectHC || pleasewaitWin32) &&
   delivery_mode == "java"%>
   onload="javascript:stopComponents()"
<% END %>
```

**Related Documentation**

- PleaseWait.html / PleaseWait-mobile-webkit.html / PleaseWait-ipad.html
- PleaseWaitObjectCC
- PleaseWaitObjectHC
- stopComponents()
**plugintype**

Provides the information about type of defender server application.

**Object Type** Template variable

**Included In** Defender.thtml

**Usage** Optional. Valid value is CASQUEAPP. Connect Secure sets the plugintype variable to CASQUEAPP when finding a Casque RADIUS server.

**Example** The following code loads the `frmCasque` HTML form if the `plugintype` contains the value CASQUEAPP:

```html
<% IF plugintype == 'CASQUE' %>
<form name="frmCasque" method="POST" autocomplete=off>
<input type=hidden name="key" value="<% secid_key %>">
<input type=hidden name="p" value="casqueapp">
<a href="#" onClick="submitFrmCasque(); return false;">LAUNCH CASQUE PLAYER</a><BR>
</form>
<% END %>
```

**Related Documentation**

- Defender.thtml / Defender-mobile-webkit.thtml / Defender-ipad.thtml
- `frmCasque`

**portal**

User interface string that displays the sign-in page’s portal name.

**Object Type** Template variable

**Default Value** Secure SSL VPN

**Included In** LoginPage.thtml, LogoutPage.thtml, SSL.thtml, SecondaryLoginPage.thtml, Logout.html, Defender.thtml, ExceededConcurrent.thtml, GeneratePin.thtml, NewPin.thtml, NextToken.thtml, ShowSystemPin.thtml

**Usage** Optional

**Example** `<% portal FILTER verbatim %>`

**Related Documentation** None

**postAuthSNDecline**

Text for “Decline” button on sign-in notifications page.

**Object Type** Template variable

**Default Value** None

**Included In** SigninNotifPostAuth.thtml

**Usage** Optional
**postAuthSNProceed**

Text for "Proceed" button on sign-in notifications page.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Template variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default value</td>
<td>None</td>
</tr>
<tr>
<td>Included In</td>
<td>SigninNotifPostAuth.thtml</td>
</tr>
<tr>
<td>Usage</td>
<td>Optional</td>
</tr>
</tbody>
</table>

**postAuthSNText**

Content of sign-in notification on sign-in notifications page.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Template variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default value</td>
<td>None</td>
</tr>
<tr>
<td>Included In</td>
<td>SigninNotifPostAuth.thtml</td>
</tr>
<tr>
<td>Usage</td>
<td>Required</td>
</tr>
</tbody>
</table>

**postAuthSNTitle**

Title of sign-in notification on sign-in notifications page.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Template variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default value</td>
<td>None</td>
</tr>
<tr>
<td>Included In</td>
<td>SigninNotifPostAuth.thtml</td>
</tr>
<tr>
<td>Usage</td>
<td>Optional</td>
</tr>
</tbody>
</table>

**preAuthSNDecline**

Text for "Decline" button on sign-in notifications page.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Template variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default value</td>
<td>None</td>
</tr>
<tr>
<td>Included In</td>
<td>SigninNotifPreAuth.thtml</td>
</tr>
<tr>
<td>Usage</td>
<td>Optional</td>
</tr>
</tbody>
</table>

**preAuthSNProceed**

Text for "Proceed" button on sign-in notifications page.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Template variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default value</td>
<td>None</td>
</tr>
<tr>
<td>Included In</td>
<td>SigninNotifPreAuth.thtml</td>
</tr>
<tr>
<td><strong>Usage</strong></td>
<td>Optional</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td><strong>preAuthSNText</strong></td>
<td></td>
</tr>
<tr>
<td>Content of sign-in notification on sign-in notifications page.</td>
<td></td>
</tr>
<tr>
<td><strong>Object Type</strong></td>
<td>Template variable</td>
</tr>
<tr>
<td><strong>Default value</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Included In</strong></td>
<td>SigninNotifPreAuth.thtml</td>
</tr>
<tr>
<td><strong>Usage</strong></td>
<td>Required</td>
</tr>
<tr>
<td><strong>preAuthSNTTitle</strong></td>
<td></td>
</tr>
<tr>
<td>Title of sign-in notification on sign-in notifications page.</td>
<td></td>
</tr>
<tr>
<td><strong>Object Type</strong></td>
<td>Template variable</td>
</tr>
<tr>
<td><strong>Default value</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Included In</strong></td>
<td>SigninNotifPreAuth.thtml</td>
</tr>
<tr>
<td><strong>Usage</strong></td>
<td>Optional</td>
</tr>
<tr>
<td><strong>prompts</strong></td>
<td></td>
</tr>
<tr>
<td>Contains display information for username/password prompts.</td>
<td></td>
</tr>
<tr>
<td><strong>Object Type</strong></td>
<td>Template variable</td>
</tr>
<tr>
<td><strong>Included In</strong></td>
<td>LoginPage.thtml, NewPin.thtml, NextToken.thtml, Defender.thtml, SecondaryLoginPage.thtml, SM-NewPinSelect.thtml, SM-NewUserPin.thtml, SM-NextToken.thtml</td>
</tr>
<tr>
<td><strong>Usage</strong></td>
<td>Required when including SecondaryLoginPage.thtml template in your zip file to collect secondary authentication credentials. The prompts variable is a hash list used to display primary and/or secondary username and password prompts in the sign-in page. You can use the prompts variable instead of hardcoding username and password labels. If you want to prompt the user to enter secondary sign-in credentials, and you do not include the prompts variable in your template, Connect Secure brings up a secondary sign-in page to collect the user’s secondary credentials. If you include the prompts variable, Connect Secure collects the secondary credentials along with the primary credentials in the main sign-in page.</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>The following example is a required piece of code in the SecondaryLoginPage.thtml template:</td>
</tr>
</tbody>
</table>

```html
<% FOREACH prompt = prompts %>
<% NEXT IF !prompt.required %>
<% prompt.promptText %>
<input type="<% prompt.type %>" name="<% prompt.name %>" size="20">
<% END %>
```
realm

Returns name of the current user’s realm.

Object Type
HTML form field

Included In

Usage
Required. Login.cgi passes the realm value posted here to the appropriate authentication server. See example below.

Example
The following example appears in several template files:

```html
<% IF RealmList.size == 0 %>
  <td>LoginRealm</td><td></td><td>
  <input type="text" name="realm" size="20">
</td>
<% ELSIF RealmList.size == 1 %>
  <input type="hidden" name="realm" value="<% RealmList.0 %>">
<% ELSE %>
  <td>LoginRealm</td><td></td><td>
  <select size="1" name="realm">
  <% FOREACH r = RealmList %>
  <option value="<% r %>" ><% r %></option>
  <% END %>
  </select>
</% ELSE %>
</td><td></td><td>
</% IF RealmList.size == 0 %>

If you only associate one realm with a URL, and you want to eliminate the realm text box and realm list box altogether, you can eliminate this code with the exception of the following line:

```html
<input type="hidden" name="realm" value="<% RealmList.0 %>">
```

You must include this line. Connect Secure populates this variable with the correct value. You can use this approach if you are sure that only one realm is associated with the sign-in URL.

Related Documentation
RealmList

realm

User interface label for the string “realm”.

Object Type
Template variable

Default value
Realm

Included In
LoginPage.thtml

Usage
Optional
Example

In the following example, if no list of realms is available, display a textbox for the user to enter their realm. The realm variable is a label for the textbox.

```html
<% IF RealmList.size == 0 %>
  <input type="text" name="realm" value="" size="20">
<% END %>
```

Related Documentation

**RealmList**

List of realms available to the user.

**Object Type**
Template variable

**Included In**
LoginPage.thtml, GeneratePin.thtml, NewPin.thtml

**Usage**
Optional. If you use the RealmList variable in the LoginPage.thtml template, the following rules apply:

- If RealmList.size = 0, then allow the user to type the realm name into a text box.
- If RealmList.size => 1, then allow the user to choose the realm name from a list box.

**Example**

The following example from LoginPage.thtml shows RealmList being used as an index in a foreach loop that returns the available realms.

```html
<% FOREACH r = RealmList %>
  <option value="<% r %>"><% r %></option>
<% END %>
```

**Related Documentation**
- CertificateAuthentication
- realm

**recallLastRealmUsed()**

Use this function if you want to allow users to select from multiple realms in the sign-in page.

**Object Type**
JavaScript function

**Included In**
LoginPage.thtml

**Usage**
Optional

**Example**

The function checks that a realm exists and then gets the last selected authentication realm.

```javascript
function recallLastRealmUsed() {
  if (document.frmLogin.realm != null &&
      document.frmLogin.realm.type == "select-one") {
    // try to remember which auth realm was last used
    var sLastRealm = GetCookieValue("lastRealm");
    if (sLastRealm.length > 0) {
      var cmb = document.frmLogin.realm;
```
var nNumRealms = cmb.options.length;
for (var n=0; n < nNumRealms; n++) {
  if (cmb.options[n].text == sLastRealm) {
    cmb.selectedIndex = n;
  }
}

Related Documentation

**FinishLoad()**

**redirect()**

Redirects the page to a compatibility checking page.

**Object Type**

JavaScript function

**Included In**

MeetingRunJava.thtml, MeetingTestMSJava.thtml, MeetingTestJava.thtml

**Usage**

Optional

**Example**

The following two code samples show how the `redirect()` function redirects to a meeting page generated by a given CGI file:

From MeetingTestMSJava.thtml:

```javascript
function redirect () {
  document.location.replace("/dana-na/meeting/
  <%signin%>meeting_testjava.cgi?<%mid_param%>"
}
```

From MeetingTestJava.thtml:

```javascript
function redirect () {
  document.location.replace("/dana-na/meeting/
  <%signin%>meeting_testresult.cgi?<%mid_param%>"
}
```

**Related Documentation**

- MeetingTestMSJava.thtml
- MeetingRunJava.thtml
- MeetingTestJava.thtml
- MeetingTestJS.thtml

**Remediate.thtml / Remediate-mobile-webkit.thtml / Remediate-ipad.thtml**

Provides the user with the ability to fix the Host Checker starts during pre-authentication and post-authentication, before the Welcome page displays.
Object Type: Template

Usage: Optional. If you have enabled Host Checker remediation, this page appears before/after the sign-in page and displays messages to the user when Host Checker policies with custom instructions have failed.

Required Variables:
- "listFailedPolicies"
- "showButtons"
- "showClose"
- "showContinue"
- "showHeading"
- "showLoading"
- "showPolicies"
- "showRemedOption"
- "showTryAgain"

- showHeading (Remediate-mobile-webkit.thtml and Remediate-ipad.thtml)
- showPolicies (Remediate-mobile-webkit.thtml and Remediate-ipad.thtml)
- showButtons (Remediate-mobile-webkit.thtml and Remediate-ipad.thtml)
- showContinue (Remediate-mobile-webkit.thtml and Remediate-ipad.thtml)
- listFailedPolicies (Remediate-mobile-webkit.thtml and Remediate-ipad.thtml)

Example:

**remedy1**

Contains the first possible remedy if your system is not fully compatible.

Object Type: Template variable

Included In: MeetingTestResult.thtml

Usage: Required

Example: The following code displays the first remedy:

```html
<% IF remedy1 %>
<% remedy1 FILTER verbatim %>
<% END %>
```

Related Documentation: remedy2

**remedy2**

Contains the second possible remedy if your system is not fully compatible.
Object Type: Template variable
Included In: MeetingTestResult.thtml
Usage: Required
Example: The following code displays the second remedy:

```html
<% IF remedy2 %>
<% remedy2 FILTER verbatim %>
<% END %>
```

Related Documentation: remedy1

---

**RoleList**

List of roles available to the user.

Object Type: Template variable
Included In: SelectRole.thtml
Usage: Optional
Example: The following code uses the RoleList variable to process a loop of roles (in this case, two roles):

```html
<%FOREACH role = RoleList %>
<a href="login.cgi?loginmode=mode_selectedrole&role=<%role.0%>
&selectrolekey=<%SelectRoleKey%>&passwordExpiration=<%PasswordExpiration%>"
><%role.1%></a>
<%END%>
```

Related Documentation: SelectRole.thtml / SelectRole-mobile-webkit.thtml / SelectRole-ipad.thtml

---

**ROWS**

Contains the number of occurrences that match the search criteria.

Object Type: Template variable
Included In: MeetingSelect.thtml
Usage: Required
Example: The following code checks to see if there are any matching occurrences. If yes, the code then generates a list of meeting instances:

```html
<% IF rows %>
<%FOREACH meeting = meetings%>
<input type="button" value="">
onclick="self.location='/dana-na/meeting/
<%signin%>login_meeting.cgi?mid=<%mid%>&"</a>
<%END%>
```

© 2016 by Pulse Secure, LLC. All rights reserved 115
runOnLoad

Called when page loads the <body>.

Object Type
JavaScript function

Included In
Remediate.thtml

Usage
Optional. Useful when you need to load ActiveX or an applet as needed.

Example
None

Related Documentation
Remediate.thtml / Remediate-mobile-webkit.thtml / Remediate-ipad.thtml

safari

Boolean that indicates whether or not the user’s browser is Macintosh Safari.

Object Type
Template variable

Included In
Logout.html, MeetingTestJS.html

Usage
Required

Example
The following code fragment tests the variable to discover if the browser is the Safari browser:

<% IF safari %>

Related Documentation
geckoBrowser

secid_challenge

Challenge string obtained from a server.

Object Type
Template variable

Included In
Defender.html

Usage
Required. Used to return a challenge string from a RADIUS server in Defender.html.

Example
The following code displays the challenge to the user.

<% secid_challenge FILTER verbatim %>
Related Documentation

- Defender.thtml / Defender-mobile-webkit.thtml / Defender-ipad.thtml
- p

secidcontextid

Form field for the value represented by secid_contextid.

Object Type
HTML form field

Included In
GeneratePin.thtml, NewPin.thtml, NextToken.thtml, Defender.html

Usage
Required

Example
The following code assigns the handle of the current transaction to a form field:

```html
<input type=hidden name="secidcontextid" value="<% secid_contextid %>">
```

Related Documentation
secid_contextid

secid_contextid

Current transaction handle.

Object Type
Template variable

Included In
GeneratePin.thtml, NewPin.thtml, NextToken.thtml

Usage
Required. You must send this value to the server, but if you do not want to expose it to the user, send it as a hidden form field.

Example
The following code assigns the current transaction handle from the server to the HTML form:

```html
<input type=hidden name="secidcontextid" value="<% secid_contextid %>">
```

Related Documentation
secid_contextid

secid_loginmode

Returns a value of Next Token mode or New PIN mode.

Object Type
Template variable

Included In
GeneratePin.thtml, NewPin.thtml, NextToken.thtml, Defender.html

Usage
Required. You must send this value to the server, but if you do not want to expose it to the user, you may send it as a hidden form field.

Example
The following code assigns the mode to the HTML form:

```html
<input type=hidden name="loginmode" value="<% secid_loginmode %>">
```
**Related Documentation**

- secidcontextid
- secid_contextid

---

### secid_pinerr

Error code and corresponding message that informs the user if he has entered his PIN incorrectly.

**Object Type**
Template variable

**Included In**
GeneratePin.htmml, NewPin.htmml

**Usage**
Optional

**Example**
The following code shows four different conditions using `secid_pinerr`:

```html
<% IF secid_pinerr == 0 %>
New PIN required
<% ELSIF secid_pinerr == 1 %>
The Two PINs Entered Do Not Match
<% ELSIF secid_pinerr == 2 %>
Invalid PIN Format
<% ELSIF secid_pinerr == 3 %>
Invalid PIN Length
<% END %>
```

---

### secid_pinformat

Error message informing the end-user of the restrictions on the required PIN.

**Object Type**
Template variable

**Included In**
GeneratePin.htmml, NewPin.htmml

**Usage**
Optional

**Example**
The following fragment shows how to use this variable to display a particular restriction:

```
You must create a new Personal Identification Number (PIN) before you can sign in. Your PIN should be <% secid_pinformat %> long.
```

---

### secid_pinselectmode

Sets the user’s PIN select mode.
<table>
<thead>
<tr>
<th>Object Type</th>
<th>Template variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Included In</td>
<td>GeneratePin.html, NewPin.html</td>
</tr>
<tr>
<td>Usage</td>
<td>Optional</td>
</tr>
<tr>
<td>Example</td>
<td>The following code creates a link and sets the login mode:</td>
</tr>
</tbody>
</table>

```html
    <% IF secid_pinselectmode == 1 %>
    If you prefer, the system can
    <a href="secid_genpin.cgi?loginmode=<% secid_loginmode %>
        onclick="gCancelNewPinMode=false;">
    generate a PIN</a> for you.
    <% END %>
```

| Related Documentation | secid_pinselectmode |

### secid_systempin

Object Type | Template variable
---|---
Included In | ShowSystemPin.html, GenerateNewPin.html, NewPin.html
Usage | Optional
Example | The following code sets the user’s PIN.

```
    Your new PIN is <%= secid_systempin %>
    Be sure to remember it, because you need your PIN each time you sign in.
```

| Related Documentation | secid_actionSavePin |

### secid_username

Object Type | Template variable
---|---
Included In | GeneratePin.html, NewPin.html, NextToken.html, Defender.html, LoginPage.html
Usage | Required
Example | The following code posts the username passed into the form back to the server:

```
    <INPUT type=hidden name="username" value="<%= secid_username %>">
```

| Related Documentation | username |
secidactionSavePin

Specifies an action to be taken when a new PIN is created.

Object Type
HTML form field

Included In
NewPin.thtml, GeneratePin.thtml

Usage
Required

Example
The value of one of the following fields must be sent to the Connect Secure server:

```html
<input type="submit" name="secidactionSavePin" value="Save PIN">
<input type="submit" name="secidactionCancel" value="Cancel">
```

Related Documentation
- GeneratePin.thtml / GeneratePin-mobile-webkit.html / GeneratePin-ipad.html

secidpinformat

Error message informing the end-user of the restrictions on the required PIN.

Object Type
HTML form field

Included In
NewPin.thtml, GeneratePin.thtml

Usage
Required

Example
The following code assigns the secure ID PIN format:

```html
<input type="hidden" name="secidpinformat" value="<% secid_pinformat %>">
```

Related Documentation
secid_pinformat

secidpinselectmode

Passes the user’s PIN select mode to the server.

Object Type
HTML form field

Included In
NewPin.thtml, GeneratePin.thtml

Usage
Required

Example
The following code assigns the PIN select mode:

```html
<input type="hidden" name="secidpinselectmode" value="<% secid_pinselectmode %>">
```

Related Documentation
secid_pinselectmode

Collects the user’s name and password for a secondary authentication server.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Template</th>
</tr>
</thead>
<tbody>
<tr>
<td>Usage</td>
<td>Optional. You must include the prompts variable in LoginPage.html in order to call this template. The prompts variable provides the proper form input names and indicates whether or not the inputs are required. However, if you prefer, you can hardcode the form input names into the page and it still works. Further, the sign-in process accepts form input names user#2 and password#2 as the secondary credentials for any authentication server. The names in the prompt variables include the authentication server name in this format: user@ldap, password@ldap.</td>
</tr>
<tr>
<td>Required Variables</td>
<td>None</td>
</tr>
<tr>
<td>Example</td>
<td>None</td>
</tr>
</tbody>
</table>

**SelectRole.html / SelectRole-mobile-webkit.html / SelectRole-ipad.html**

Appears after the sign-in page and displays a list of roles from which the user can choose.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Template</th>
</tr>
</thead>
<tbody>
<tr>
<td>Usage</td>
<td>Optional. You may choose to customize this template if you have assigned your users to multiple roles, but have not permissively merged those roles.</td>
</tr>
<tr>
<td>Required Variables</td>
<td>None</td>
</tr>
<tr>
<td>Example</td>
<td>None</td>
</tr>
</tbody>
</table>

**serverdetails**

Contains some server details that are used on the end-user system.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Template variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Included In</td>
<td>PleaseWait-ipad.html, PleaseWait-mobile-webkit.html</td>
</tr>
<tr>
<td>Usage</td>
<td>Optional (Required for Pulse HC on Android or iOS end-user systems)</td>
</tr>
<tr>
<td>Required Variables</td>
<td>None</td>
</tr>
<tr>
<td>Example</td>
<td>&lt;% serverdetails %&gt;</td>
</tr>
</tbody>
</table>

**setFailed()**

Called if the component is stopped intentionally.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>JavaScript function</th>
</tr>
</thead>
</table>
Included In

Logout.html, PleaseWait.html

Usage
component signifies the component that is being checked to see if it has been stopped intentionally.

Example

The following code stops Host Checker:

```html
<%IF pleasewaitObjectHC %> setSucceeded('hc');
<%END%>
```catch (e) {
<%IF pleasewaitObjectHC %> setFailed('hc'); <%END%>
```

Related Documentation

setStarted() setFinished

setFinished

Called if the component has finished starting up.

Object Type
JavaScript function

Included In
Logout.html, PleaseWait.html

Usage
Optional

component—Valid values for component are hc for Host Checker and cc for Cache Cleaner.

Example

The following code shows how setFinished is used in the Logout.html template:

```javascript
function setFinished(component) {
    document.getElementById(component).style.fontWeight = "normal";
}
```

Related Documentation

- Logout.html / Logout-mobile-webkit.html / Logout-ipad.html
- PleaseWait.html / PleaseWait-mobile-webkit.html / PleaseWait-ipad.html

SetLastRealm()

Enables users to select from multiple realms on the sign-in page.

Object Type
JavaScript function

Included In
LoginPage.html

Usage
When the user signs in, this function retrieves the authentication realm selected by the user and sets an expiration date of 30 days.

sValue—The name of the realm the user selects.

Example

The following code shows the function:
function SetLastRealm(sValue) {
    var dtExpire = new Date();
    dtExpire.setDate(dtExpire.getDate() + 30);
    document.cookie = "lastRealm=" + escape(sValue) + "; expires=" + dtExpire.toGMTString();
}

Related Documentation
LoginPage.thtml / LoginPage-mobile-webkit.thtml / LoginPage-ipad.thtml

setStarted()

Called if the component is started in order to stop it.

Object Type
JavaScript function

Included In
Logout.thtml, PleaseWait.thtml

Usage
Required.

component—Name of the component.

Example
The following code shows the function:

    function setStarted(component)
    {
        document.getElementById(component).style.fontWeight = "bold";
    }

Related Documentation
setFailed()

setSucceeded()

Called if the component is successfully stopped.

Object Type
JavaScript function

Included In
Logout.thtml, PleaseWait.thtml

Usage
Required

Example
The following code calls the function if Host Checker has been successfully stopped:

    <%IF pleasewaitObjectHC %rangle setSucceeded('hc'); <%END%>

Related Documentation
setStarted()

setthankyou()

Sets the timeout value and calls the thankyou() function, which displays the login page after the user has signed out.
### Object Type
JavaScript function

### Included In
Logout.thtml

### Usage
Required

### Example
The following code shows the function:

```javascript
function setthankyou()
{
    setTimeout("thankyou()", 2000);
}
```

#### Related Documentation
thankyou()

---

### setup_classid

The CLSID of the ActiveX object.

### Object Type
Template variable

### Included In
Logout.thtml, Logout-ipad.thtml, Logout-mobile-webkit.thtml, Logout-ppc.thtml, PleaseWait-ppc.thtml, PleaseWait.thtml, AgentInstall.thtml, AgentlessInstall.thtml

### Usage
Required

### Example
The following example assigns the CLSID if the delivery mechanism is ActiveX.

```jsp
<%IF delivery_mode == 'activex'%>
    <OBJECT <%setup_classid FILTER verbatim%> id=NeoterisSetup
```

#### Related Documentation
- setup_codebase

---

### setup_codebase

The path and version information (CODEBASE) of the ActiveX object.

### Object Type
Template variable

### Included In
Logout.thtml, Logout-ipad.thtml, Logout-mobile-webkit.thtml, Logout-ppc.thtml, PleaseWait-ppc.thtml, PleaseWait.thtml, AgentInstall.thtml, AgentlessInstall.thtml

### Usage
Required

### Example
The following example assigns the CODEBASE if the delivery mechanism is ActiveX.

```jsp
<%IF delivery_mode == 'activex'%>
    <OBJECT <%setup_classid FILTER verbatim%> id=NeoterisSetup
        <%setup_codebase FILTER verbatim%>
        width=0 height=0 >
    </OBJECT>
```

#### Related Documentation
- setup_classid
### setup_ini

Contents of the ActiveX object.

**Object Type**: Template variable  
**Included In**: Logout.thtml, Logout-ppc.thtml  
**Usage**: Required  
**Example**: The following example assigns the CODEBASE if the delivery mechanism is ActiveX.

```html
<%IF setup_ini %>
  <OBJECT <%setup_ini FILTER verbatim%>
```

**Related Documentation**: Logout.thtml, Logout-ppc.thtml

### Setupappversion

The version of the Java installer setup application.

**Object Type**: Template variable  
**Included In**: Logout.thtml  
**Usage**: Required  
**Example**: The following example assigns the `setupappversion` after determining the delivery mode:

```html
<% IF delivery_mode == 'java' %>
  <APPLET code=dsSetupApplet.class
code=dsSetupApplet
  id=NeoterisSetup
  archive="NeoterisSetupApplet.jar"
MAYSCRIPT
width=0 height=0
codebase="/dana-cached/setup/" >
  <PARAM NAME="SetupAppVersion" VALUE="<% setupappversion %">"
```

**Related Documentation**: Logout.thtml, Logout-mobile-webkit.thtml, Logout-ipad.thtml

### setup_codebase

Codebase of Pulse Secure Neoteris Setup ActiveX control.

**Object Type**: Template variable  
**Included In**: Logout.thtml, PleaseWait.thtml  
**Usage**: Optional  
**Example**: The following code declares the `codebase`:
**showButtons**

Boolean indicating whether or not to show any of the action buttons.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Template variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Included In</td>
<td>Remediate.thtml , Remediate-ppc.thtml, Remediate-ipad.thtml, Remediate-mobile-webkit.thtml</td>
</tr>
<tr>
<td>Usage</td>
<td>Required</td>
</tr>
<tr>
<td>Example</td>
<td>The following code tests showButtons. If true, then the code continues to test other directives to determine which action buttons to display:</td>
</tr>
</tbody>
</table>

```html
<% IF showButtons %>
  <input name="btnTryAgain" type="button" value="<% textTryAgain %>">
  <input name="btnHideRemed" type="button" value="<% textRemedOption %>">
  <input name="btnContinue" type="button" value="<% textContinue %>">
  <input name="btnClose" type="button" value="<% textClose %>">
<% END %>
```

**Related Documentation**

- showContinue
- showClose
- showRemedOption
- showTryAgain
showChangePasswordTitle

Boolean indicating whether or not to show primary/secondary password change title.

Object Type Template variable

Included In PasswordChange.thtml

Usage Optional.

Example The following code tests the variable. If true, the code then displays the value contained in changePasswordTitle:

```html
<%IF showChangePasswordTitle %>
  <% changePasswordTitle FILTER verbatim %>
<%END%>
```

Related Documentation
- changePasswordTitle
- PasswordChange.thtml / PasswordChange-mobile-webkit.thtml / PasswordChange-ipad.thtml

showClose

Boolean indicating whether or not to show the Close button.

Object Type Template variable

Included In Remediate.thtml

Usage Required

Example The following code tests showClose. If true, the code constructs a button that closes the page when the user clicks it:

```html
<% IF showClose %>
  <input name="btnClose" type="button" value=" <% textClose %> " onClick="javascript:self.close()">
<% END %>
```

Related Documentation showContinue

showContinue

Boolean indicates whether or not to show the Continue button.

Object Type Template variable

Included In Remediate.thtml, Remediate-ppc.thtml, Remediate-ipad.thtml, Remediate-mobile-webkit.thtml

Usage Required

Example The following code shows a Continue button if the showContinue variable is true:
<% IF showContinue %>
<input name="btnContinue" type="button" value="<% textContinue %>"
onClick="Continue()"></td>
<% END %>

Related Documentation  Remediate.thtml / Remediate-mobile-webkit.thtml / Remediate-ipad.thtml

dshowHeading
flag to determine if we should show heading

Object Type  Template variable
Included In  Remediate.thtml, Remediate-ppc.html, Remediate-ipad.html, Remediate-mobile-webkit.html
Usage  Required
Example  The following code displays message text if the showHeading variable is true:

<% IF showHeading %>
<% msg FILTER verbatim %>
<% END %>

Related Documentation  Remediate.thtml / Remediate-mobile-webkit.thtml / Remediate-ipad.thtml

dshowLoading
Boolean indicates whether or not to show text during the loading of ActiveX or an applet.

Object Type  Template variable
Included In  Remediate.html
Usage  Required
Example  The following code displays message text if the showLoading variable is true:

<% IF showLoading %>
<% pleasewait %>
<% HC_REMED_POLICIES_CHECK %>

Related Documentation  Remediate.html / Remediate-mobile-webkit.html / Remediate-ipad.html

dshowReinstall
Boolean indicates whether or not to show the Reinstall button.

Object Type  Template variable
Included In  Remediate.html, Remediate-ppc.html
Usage  Required
Example

The following code displays message text if the showReinstall variable is true:

```html
<% IF showReinstall %>
<% pleasewait %>
<% HC_REMED_POLICIES_CHECK %>
```

Related Documentation

**showRemedOption**

Boolean indicates whether or not to show the button that displays the message: Do not show remediation for this session.

**Object Type** Template variable

**Included In** Remediate.thtml, Remediate-ppc.thtml, Remediate-ipad.thtml, Remediate-mobile-webkit.thtml

**Usage** Required

**Example** The following code tests the value of showRemedOption. If true, the code hides the remediation button:

```html
<% IF showRemedOption %>
<input name="btnHideRemed" type="button" value="<% textRemedOption %>" onClick="HideRemed(); <% disabled %>">  
<% HC_REMED_SHOW_ADV_PREFS %>
<% END %>
```

Related Documentation Remediate.thtml / Remediate-mobile-webkit.thtml / Remediate-ipad.thtml

**showPolicies**

Boolean indicating whether or not to show remediation policies.

**Object Type** Template variable

**Included In** Remediate.thtml, Remediate-ppc.thtml, Remediate-ipad.thtml, Remediate-mobile-webkit.thtml

**Usage** Required

**Example** The following code tests the values of several template variables:

```html
<% IF showHeading || showPolicies || showButtons %>
```

Related Documentation Remediate.thtml / Remediate-mobile-webkit.thtml / Remediate-ipad.thtml

**ShowSystemPin.thtml** / **ShowSystemPin-mobile-webkit.thtml** / **ShowSystemPin-ipad.thtml**

Displays the system-generated PIN to the user.
### showTryAgain

Boolean that indicates whether or not to show the Try Again button.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Template variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Included In</td>
<td>Remediate.thtml</td>
</tr>
<tr>
<td>Usage</td>
<td>Required</td>
</tr>
<tr>
<td>Example</td>
<td>The following code checks the variable and, if true, displays the Try Again button. The code also disables the button once the end-user has clicked it:</td>
</tr>
</tbody>
</table>
|               | `<% IF showTryAgain %>
|               |   `<input name="btnTryAgain" type="button" value="<% textTryAgain %>"
|               |     onClick="TryAgain()" `<% disabled %>>
|               | `<% END %>`       |

**Related Documentation**
- TryAgain

### signinAgain

Boolean that indicates whether or not to display the login page after users log out.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Template variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Included In</td>
<td>Logout-ipad.thtml, Logout-mobile-webkit.thtml, Logout-ppc.thtml, Logout.thtml</td>
</tr>
<tr>
<td>Usage</td>
<td>Optional</td>
</tr>
<tr>
<td>Example</td>
<td>The following code checks the variable and, if true, redirects the user to the sign-in page (welcome.cgi).</td>
</tr>
</tbody>
</table>
|               | `<% IF signinAgain %>
|               |   `<a href="welcome.cgi" >
|               |     `<div id=signindiv>
|               |     `<% signinAgain FILTER verbatim %>`
|               | `<% END %>`       |

**Related Documentation**
- None
Example

The following code checks the variable and, if true, redirects the user to the sign-in page (welcome.cgi).

```html
<% IF signinAgain %>
   <a href="welcome.cgi">
      <div id=signindiv>
         <% signinAgain FILTER verbatim %>
      </div>
   </a>
<% END %>
```

Related Documentation

None

signinAgainUrl

Contains the signin url to be used after the user logs out.

Object Type

Template variable

Included In

Logout-ipad.thtml, Logout-mobile-webkit.thtml, Logout.thtml

Usage

Optional.

Example

```html
<% <%# signinAgainUrl %>
```

Related Documentation

SigninNotifPostAuth.thtml / SigninNotifPreAuth-mobile-webkit.thtml / SigninNotifPostAuth-ipad.thtml

Displays Sign-in Notification to user and asks for agreement to proceed.

Object Type

Template

Usage

You can implement pre-auth sign-in notifications, post-auth sign-in notifications, or both.

Required Variables

postAuthSNText

Example

None

SigninNotifPreAuth.thtml / SigninNotifPreAuth-ipad.thtml

Displays Sign-in Notification to user and asks for agreement to proceed.

Object Type

Template

Usage

You can implement pre-auth sign-in notifications, post-auth sign-in notifications, or both.

Required Variables

preAuthSNText

Example

None


Prompts the user to enter a new PIN or create a system-generated PIN before signing into Connect Secure.

Object Type

Template

Usage

Required for SiteMinder authentication.
**Required Variables**  
"secid_loginmode"

**Example**  
None


Enables the user to create a system-generated PIN if the user selects the System PIN option in the SM-NewPinSelect.thtml page.

**Object Type**  
Template

**Usage**  
Optional

**Required Variables**  
"secid_loginmode"

**Example**  
None

**Related Documentation**  

**SM-NewUserPin.thtml**

Prompts the user to create a new PIN if he selects the Enter PIN option in the SM-NewPinSelect.thtml page.

**Object Type**  
Template

**Usage**  
Optional. Also determines if the two PINs entered by the user match and alerts the user if necessary. When configuring the secid_pinserr variable (which stores the error code and message that informs the user if he mis-entered his PIN), note that possible values include:

**Table11: New PIN Assignment Messages**

<table>
<thead>
<tr>
<th>Code</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>New PIN Assignment</td>
</tr>
<tr>
<td>1</td>
<td>The Two PINs Entered Do Not Match</td>
</tr>
</tbody>
</table>

**Required Variables**  
"secid_loginmode"

**Example**  
None

**Related Documentation**  

**SM-NextToken.thtml / SM-NextToken-mobile-webkit.thtml / SM-NextToken-ipad.thtml**

Prompts the user to verify his credentials by entering his SecureID token code.

**Object Type**  
Template

**Usage**  
Optional. For information the JavaScript, form definitions, form fields, and variables contained in these templates, see comments in the templates.
NOTE: If you prohibit the use of System-generated PINS, you do not need to include SM-NewPinSelect.html and SM-NewPinSystem.html in the zip file that you upload to Connect Secure.

Required Variables

- "secid_loginmode"

Example

None

Related Documentation


**softid_error**

Set if the ACE/SoftID authentication server returns errors.

Object Type

Template variable

Included In

- GeneratePin.html (ACE/softID)
- LoginPage.html (ACE/softID)
- NewPin.html (ACE/softID)
- NextToken.html (ACE/softID)

Usage

Required

Example

The following code assigns any SoftID errors to a JavaScript variable:

```javascript
var error = "<% softid_error %>";
```

Related Documentation

- softid_time

**softid_time**

Contains the ACE/SoftID authentication server time.

Object Type

Template variable

Included In

- LoginPage.html (ACE/softID)

Usage

Required by authenticate() function.

Example

The following code assigns the ACE/SoftID time to a time variable:

```javascript
var time = <% softid_time %> ;
```

Related Documentation

- LoginPage.html / LoginPage-mobile-webkit.html / LoginPage-ipad.html
- SSL.html / SSL-mobile-webkit.html / SSL-ipad.html

SSL.html / SSL-mobile-webkit.html / SSL-ipad.html

Displays an error if authentication fails and the user is not allowed to sign into Connect Secure.

Object Type

Template
Usage

You must always include `SSL.html` in your zip file. This is a standard Connect Secure page that displays an error if authentication fails or realm-level checks fail and the user is not allowed to sign into Connect Secure. This template does not contain any JavaScript or forms. However, it does contain error message variables. For more information, see comments in the template.

Required Variables

None

Example

None

**Start_onclick()**

Initiates the proper applets when beginning a meeting session.

Object Type

JavaScript function

Included In

MeetingRun.html

Usage

Required

Example

The following code shows the `Start_onclick()` function:

```javascript
function Start_onclick() {
    if (typeof(NeoterisSetup) != "undefined") {
        try {
            NeoterisSetup.startSession();
            return 0;
        } catch (e) {
            return Start_onclick_applet();
        }
    } else {
        return Start_onclick_applet();
    }
}
```

Related Documentation

MeetingRun.html

**Start_onclick_java()**

Configures the session manager.

Object Type

JavaScript function

Included In

MeetingRun.html

Usage

Required. Called by various pages from which the user starts and stops meeting sessions.

Example

The following code illustrates how the function begins to configure the session manager:

```javascript
function Start_onclick_java() {
    var sOptions = "resizable=no,scrollbars=no,status,width=250,height=150";
    var x,y;
    if (screen) {
        if ((screen.availWidth) && (screen.availHeight)) {
            x = screen.availWidth - 290; // 40 pixel slop
```
y = screen.availHeight - 190; // 40 pixel slop
sOptions += ",left=" + x + ",screenX=" + x + ",top=" + y + ",screenY=" + y;
}

Related Documentation  MeetingRun.html

**start_status**

Status of the started Host Checker or Cache Cleaner instance.

**Object Type**  JavaScript variable

**Included In**  PleaseWait.html

**Usage**  Required. The return status equals 1 if a test of this variable fails.

**Example**  The following code shows how the `getStatus` function uses the `start_status` variable:

```javascript
function getStatus()
{
    return start_status;
}
```

Related Documentation  PleaseWait.html / PleaseWait-mobile-webkit.html / PleaseWait-ipad.html

**StartCC()**

Enables the current user to start Cache Cleaner.

**Object Type**  JavaScript function

**Included In**  PleaseWait.html

**Usage**  Optional

**Example**  None

Related Documentation  PleaseWait.html / PleaseWait-mobile-webkit.html / PleaseWait-ipad.html

**StartHC()**

Enables the current user to start Host Checker.

**Object Type**  JavaScript function

**Included In**  PleaseWait.html

**Usage**  Optional

**Example**  None

Related Documentation  PleaseWait.html / PleaseWait-mobile-webkit.html / PleaseWait-ipad.html
• StartCC()

**started**

Boolean indicating that a meeting instance has started.

**Object Type**  
Template variable

**Included In**  
MeetingSelect.thtml

**Usage**  
Required

**Example**  
The following code checks to see if meetings have started, and if so, populates the page with sign-in components corresponding to each meeting.

```html
<% IF rows %>
<%FOREACH meeting = meetings%>
<input type="button" value=" &gt; "onclick="self.location='/dana-na/meeting/%signin%>login_meeting.cgi?mid=<%mid%>
&occurrence=<%meeting.3%>&%params%">
<% started %>
<%meeting.0%>
<%meeting.1%>
<%meeting.2%>
<% END %>
<%END%>
```

**Related Documentation**  
MeetingSelect.thtml

**startPageLink**

Page Link for the Welcome Page.

**Object Type**  
Template variable

**Included In**  
GraceLoginUsed.thtml, PasswordExpiration.thtml

**Usage**  
Required. Must be added as an href link. Provides a link in the page, allowing user to navigate to the Welcome page.

**Example**  
The following code shows this variable used as an href link:

```html
<a href="<% startPageLink %>">Click here to go to Welcome Page</a>
```

**Related Documentation**

- GraceLoginUsed.thtml / GraceLoginUsed-mobile-webkit.thtml / GraceLoginUsed-ipad.thtml
- PasswordExpiration.thtml / PasswordExpiration-mobile-webkit.thtml / PasswordExpiration-ipad.thtml
**startSessionReDir**

Initiates a redirect on finding that a meeting applet needs to be installed.

**Object Type** Template variable

**Included In** `MeetingAppletInstaller.thtml, MeetingTrouble.thtml, PleaseWait-mtg.thtml`

**Usage** Required

**Example**

```html
<PARAM NAME="StartSessionReDir" VALUE="<% startSessionReDir %>">
```

**Related Documentation**

- `MeetingAppletInstaller.thtml`
- `MeetingTrouble.thtml`

**stopComponents()**

Performs a variety of shutdown checks and operations, shutting down Host Checker, Cache Cleaner, and Win32 applets, if necessary, when the end-user signs out.

**Object Type** JavaScript function

**Included In** `Logout.thtml`

**Usage** Required

**Example**

```html
<% IF (pleasewaitObjectCC || pleasewaitObjectHC || pleasewaitWin32) && delivery_mode == 'java'%>
onload="javascript:stopComponents()"
<%END%>
```

**Related Documentation** `delivery_mode`
From **SM-NewUserPin.html**:

```javascript
function SubmitClicked() {
    if (document.frmNewPin.password2.value !=
        document.frmNewPin.password3.value) {
        alert("The Two PINs Entered Do Not Match");
        document.frmNewPin.password2.focus();
        return false;
    }
}
```

**Related Documentation**

- GeneratePin.html / GeneratePin-mobile-webkit.html / GeneratePin-ipad.html
- NextToken.html / NextToken-mobile-webkit.html / NextToken-ipad.html
- SM-NewUserPin.html

**submitFrmCasque()**

Initiates the download of the CASQUE RADIUS server.

**Object Type**

JavaScript function

**Included In**

Defender.html

**Usage**

Required if CASQUE RADIUS server is needed.

**Example**

The following code shows the action to initiate the CASQUE server:

```javascript
function submitFrmCasque() {
    document.frmCasque.action = "/dana-na/download/x.casque?
    url=/dana-na/auth/welcome.cgi";
    document.frmCasque.submit();
}
```

**Related Documentation**

- frmCasque

**testjava.ini**

Boolean indicating whether or not Connect Secure needs to perform a compatibility check for Java.

**Object Type**

Template variable

**Included In**

MeetingTestJava.html

**Usage**

Required
Example

```html
<APPLET
<% testjava_ini FILTER verbatim %>
<PARAM NAME="TestJavaApplet" VALUE="true">
<PARAM NAME="RedirectUrl" VALUE="/danana/meeting/<%signin%>meeting_testresult.cgi
?java=1<%mid_param%>">
</APPLET>
```

Related Documentation  
MeetingTestJava.thtml

textClose

Default text for the Close button.

Object Type  
Template variable

Included In  
Remediate.thtml

Usage  
Required. Used by Connect Secure to dynamically update the button name, depending upon the user’s activity.

Example  
The following code changes the button to a Close button:

```html
<% IF showClose %>
<input name="btnClose" type="button" value="<% textClose %>
 onClick="javascript:self.close()">
<% END %>
```

Related Documentation

- textContinue
- textRemedOption
- textTryAgain

textContinue

Default text for the Continue button.

Object Type  
Template variable

Included In  
Remediate.thtml

Usage  
Required. Used by Connect Secure to dynamically update the button name, depending upon the user’s activity.

Example  
The following code changes the button to a Continue button:

```html
<% IF showContinue %>
<input name="btnContinue" type="button" value="<% textContinue %>
 onClick="Continue()">
<% END %>
```

Related Documentation
textRemedOption

Default text for the Do not show remediation for this session button.

Object Type: Template variable

Included In: Remediate.thtml

Usage: Required. Used by Connect Secure to dynamically update the button name, depending upon the user’s activity.

Example:
```html
<% IF showRemedOption %>
<input name="btnHideRemed" type="button" value="<% textRemedOption %>
onClick="HideRemed(); <% disabled %>">
<% HC_REMED_SHOW_ADV_PREFS %>
<% END %>
```

Related Documentation:
- textClose
- textContinue
- textTryAgain

textTryAgain

Default text for the Try Again button.

Object Type: Template variable

Included In: Remediate.thtml

Usage: Required. Used by Connect Secure to dynamically update the button name, depending upon the user’s activity.

Example:
```html
<% IF showTryAgain %>
<input name="btnTryAgain" type="button" value="<% textTryAgain %>
onClick="TryAgain()" <% disabled %>>
<% END %>
```

Related Documentation:
- textClose
- textContinue
- textRemedOption
thankyou()

Removes components and displays the login page after an end-user signs out.

**Object Type**  JavaScript function  
**Included In**  Logout.html  
**Usage**  Optional  
**Example**  The function checks to see which application is being stopped and removed, then displays either an error or the sign-in page.

```javascript
function thankyou()
{
    <%IF pleasewaitObjectHC %>
    removeBulb('hc');
    removeComponent('hc');
    <%END%>
    <%IF pleasewaitObjectCC %>
    removeBulb('cc');
    removeComponent('cc');
    <%END%>
    document.getElementById('pleasewait1').innerHTML = "<% LoginPageErrorMessage verbatim%>";
    document.getElementById('signindiv').innerHTML = "Click here to sign in again";
}
```

**Related Documentation**  setthankyou()

**troubleshooting**

Contains the troubleshooting text string.

**Object Type**  Template variable  
**Included In**  MeetingAppletInstaller.html, MeetingRunJava.html  
**Usage**  Required  
**Example**  The following code checks to see if there is a troubleshooting string. If yes, then it displays the text string, as is:

```html
<% IF troubleshooting %>
<% troubleshooting FILTER verbatim %>
<% END %>
```

**Related Documentation**  
- MeetingAppletInstaller.html  
- MeetingRunJava.html
TryAgain

Submits the hidden form when the user clicks the Try Again button.

**Object Type**
JavaScript function

**Included In**
Remediate.thtml

**Usage**
Required

**Example**
```
<% IF showTryAgain %>
  <input name="btnTryAgain" type="button" value="<% textTryAgain %>">
  onClick="TryAgain()" enabled
<% END %>
```

**Related Documentation**
showTryAgain

type

Contains the type of client to use when running Meeting.

**Object Type**
Template variable

**Included In**
MeetingTrouble.thtml

**Usage**
Required. Allowable values are:

- activex
- java
- applet

**Example**
The following code checks to see if the type of client is ActiveX:
```
<% IF type == 'activex' %>
  <OBJECT><% ini FILTER verbatim %>
  <PARAM NAME="StartSessionReDir" VALUE="<%startSessionReDir%>">
  <PARAM NAME="UninstallReDir" VALUE="<%uninstallReDir%>">
  <PARAM NAME="locale" VALUE="<%locale%>">
  </OBJECT>
<% END %>
```

**Related Documentation**
MeetingTrouble.thtml

tz_offset

Contains the time zone offset.

**Object Type**
HTML form field

**Included In**
LoginPage.thtml
Usage  Required. The `Login()` function uses the `tz_offset` (time zone offset) value to help determine the user's time zone.

Example  The following code shows how the `tz_offset` field is used in an HTML form.

```html
<input type="hidden" name="tz_offset">
```

Related Documentation  `LoginPage.thtml` / `LoginPage-mobile-webkit.thtml` / `LoginPage-ipad.thtml`

### uninstall_action

Indicates the type of action to take when having trouble launching a meeting instance.

**Object Type**  Template variable

**Included In**  `MeetingTrouble.thtml`

**Usage**  Required

**Example**  The following code illustrates the use of the `uninstall_action` variable:

```html
<% IF type == 'activex' || type == 'applet' %>
If you encounter problems launching Pulse Secure Collaboration, uninstall the version that is currently on your system.
<input type="button" value="Uninstall" id=Uninstall name=button2 LANGUAGE=javascript onclick="<% uninstall_action %>">
After uninstalling, return to the "Join Meeting" page and click "Start Meeting" or "Join Meeting" to reinstall the latest Pulse Secure Collaboration application.<br>
<% END %>
```

**Related Documentation**  `Uninstall_applet()`

### Uninstall_applet()

Initiates the uninstallation of an applet.

**Object Type**  JavaScript function

**Included In**  `MeetingTrouble.thtml`

**Usage**  Required. Note that the first character is upper cased.

**Example**  The following code is the `uninstall_applet()` function:

```javascript
Uninstall_applet() {
    window.location = "/dana-na/meeting/meeting_appletuninstall.cgi?mid=<%mid%>&occurrence=<%occurrence%>&type=<%type%>";
    return true;
}
```

**Related Documentation**  `MeetingTrouble.thtml`
Uninstall_onclick()

Initiates the uninstall script.

Object Type
JavaScript function

Included In
MeetingTrouble.thtml

Usage
Required. Note that the first character is upper cased.

Example
The following code is the uninstall_onclick() function:

```javascript
Uninstall_onclick() {
    if ( typeof(NeoterisSetup) != "undefined" ) {
        NeoterisSetup.uninstall();
        return 1;
    }
    else {
        return 0;
    }
}
```

Related Documentation
MeetingTrouble.thtml

uninstallReDir

Contains the location of the page to go to on an uninstall.

Object Type
Template variable

Included In
MeetingAppletInstaller.thtml, MeetingTrouble.thtml

Usage
Required

Example
The following code assigns the uninstall redirect page to a form parameter:

```html
<PARAM NAME="UninstallReDir" VALUE="<% uninstallReDir %>">
```

Related Documentation
- MeetingAppletInstaller.thtml
- MeetingTrouble.thtml

UninstallCC

Indicates whether or not Cache Cleaner uninstall is enabled.

Object Type
Template variable

Included In
Logout.thtml

Usage
Required. Not to be confused with JavaScript variable uninstallcc.
### Example

The following code checks to see that the Cache Cleaner uninstall is enabled. If yes, then the code sets the JavaScript `uninstallcc` variable to 1, which indicates that the system should uninstall if the JavaScript function `stopCC()` is running:

```<%IF uninstallCC %>
uninstallcc = 1;
<%END%>```

### Related Documentation

- **StartCC()**
- **upAndRunning()**

`upAndRunning()` indicates whether or not Host Checker or Cache Cleaner are up and running after they are started.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>JavaScript function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Included In</td>
<td>PleaseWait.html</td>
</tr>
<tr>
<td>Usage</td>
<td>Optional</td>
</tr>
<tr>
<td>Example</td>
<td>None</td>
</tr>
<tr>
<td>Related Documentation</td>
<td>PleaseWait.html / PleaseWait-mobile-webkit.html / PleaseWait-ipad.html</td>
</tr>
</tbody>
</table>

#### UserAdmin-add-user.html

The page for creating new guest user by the Guest User Account Manager.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Template variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Usage</td>
<td>Required (Valid for Policy Secure). You must always include UserAdmin-add-user.html in your sample.zip file</td>
</tr>
</tbody>
</table>

You can edit the HTML markup to modify the look and feel. You can add, modify, or delete JavaScript functions and variables to customize the functionality presented on this page.

To customize this page, see Customizing the Guest User Account Manager Pages.

<table>
<thead>
<tr>
<th>Required Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>authServer</td>
</tr>
<tr>
<td>errorMsg</td>
</tr>
</tbody>
</table>

#### UserAdmin-add-bulk-users.html

The page for creating bulk guest user by the Guest User Account Manager.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Template variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Usage</td>
<td>Required (Valid for Policy Secure). You must always include UserAdmin-add-bulk-users.html in your sample.zip file</td>
</tr>
</tbody>
</table>

You can edit the HTML markup to modify the look and feel. You can add, modify, or delete JavaScript functions and variables to customize the functionality presented on this page.
To customize this page, see Customizing the Guest User Account Manager Pages.

**Required Variables**

- authServer
- errorMsg

### UserAdmin-update-user.thtml

The page for updating existing guest user by the Guest User Account Manager.

#### Object Type
Template variable

#### Usage
Required (Valid for Policy Secure). You must always include UserAdmin-update-user.thtml in your sample.zip file

You can edit the HTML markup to modify the look and feel. You can add, modify, or delete JavaScript functions and variables to customize the functionality presented on this page.

To customize this page, see Customizing the Guest User Account Manager Pages.

**Required Variables**

- authServer
- errorMsg

### UsersPrintInfo

Template Toolkit hash value of user information (keys: name, passwd, login, startTimeDisplay, expirationDisplay and pageBreak).

#### Object Type
Template variable

#### Included In
- guest-user-email-page.html, guest-user-print-page-batch.html

#### Usage
Optional (Required for Policy Secure).

#### Example
The following example displays the user information:

```html
<% FOREACH User = UsersPrintInfo %>
  <% User.login %>
  <% User.name %>
  <% User.passwd %>
  <% User.startTimeDisplay %>
  <% END %>
```

### Related Documentation

#### userAgentTypeAndroid

Boolean indicating whether or not the end-user’s operating system is an Android or iOS system.

#### Object Type
Template variable

#### Included In
- PleaseWait-ipad.html, PleaseWait-mobile-webkit.html

#### Usage
Optional (Required for Pulse HC on Android or iOS end systems)
Example

```% userAgentTypeAndroid %>
```

Related Documentation

**username**

This field is required for all types of authentication servers except anonymous servers and certificate servers.

Object Type: Template variable

Included In: LoginMeeting.thtml, MeetingAppletInstaller.thtml

Usage: Required. As an example of username’s use, **Login.cgi** (a Perl script on Connect Secure) passes the `username` value to the appropriate authentication server.

Example: The following example shows the use of the username variable in a meeting sign-in sequence:

```<PARAM NAME="Parameter0"
VALUE="meeting_id=<%instanceId%>;user_name=<% username %>;cert_md5=<%cert_md5%>;ncp_read_timeout=<%ncp_read_timeout%>">
```

Related Documentation

**warnings**

Contains meeting warnings.

Object Type: Template variable

Included In: MeetingRun.thtml

Usage: Required

Example: The following code is included in **MeetingRun.thtml**. You should not modify the code:

```<% warnings FILTER verbatim %>
```

Related Documentation: **MeetingRun.thtml**

**welcome**

Contains the welcome message string.

Object Type: Template variable

Default Value: “Welcome to the”

Usage

<table>
<thead>
<tr>
<th>Description</th>
<th>win32</th>
</tr>
</thead>
<tbody>
<tr>
<td>Indicates whether or not the client operating system is Windows.</td>
<td></td>
</tr>
<tr>
<td>Object Type</td>
<td>Template variable</td>
</tr>
<tr>
<td>Included In</td>
<td>Logout.html, MeetingTestJS.html</td>
</tr>
<tr>
<td>Logout.html Usage</td>
<td>Required</td>
</tr>
</tbody>
</table>

Example

The following code checks to see if the operating system is Windows. If yes, the code loads redirects the page to a page to test the compatibility of Microsoft Java:

```javascript
function checkActiveX () {
    var win = <% win32 %>; 
    try {
        NeoterisSetup.isValid();
        document.form1.submit(); 
        return true;
    } catch (e) {
        var url = win ? "/dana-na/meeting/
        <%signin%>meeting_testmsjava.cgi?
        <%mid_param%>" : "/dana-na/meeting/<%signin%>
        meeting_testjava.cgi?<%mid_param%>"; 
        document.location.replace(url);
    }
}
```

Related Documentation

MeetingTestMSJava.html

PART 2 Index

- Index
PART 2 Index

A

Accept-Language
Accept-Language header abbreviations
ActivePerl
AnonymousAuthentication variable, discussed
AppConfig
arithmetic operators, defined
authentication servers
  SiteMinder
    custom sign in pages ...

B

block directives, defined......

C

Cache Cleaner
  custom pages .................
CALL directive, restriction ....
Cancel.thtml, discussed ........
CASE directive, discussed ...
CertificateAuthentication variable, discussed
conditional operators, defined
conditional test, creating ......
custom help files ..............
custom sign-in pages ...........
  localizing ...................
customer support ..............
  contacting PSGSC ...........
Cygwin ................................

D

Defender.thtml, discussed....
directive, defined...............  

E

ELSIF directives, discussed .
END directive, discussed ..... 
errors
  modifying ......................
ExceededConcurrent.thtml, discussed

F

FILTER directive, restriction
FinishLoad() function, discussed
FOREACH loop, discussed .

G

GeneratePin.thtml, discussed
GET directive, discussed ..... 
getTimezoneOffset() function, discussed
GraceLoginUsed.thtml, discussed

H

Home variable, discussed....
Host Checker
  custom pages ................

I

IF directive, discussed ........
INCLUDE ........................
INSERT ...........................
internationalization.......... 
INTERPOLATE directive, restriction

J

J-SAM See  Secure Application Manager, J-SAM

K

keyboard.js  .....................
Kiosk.zip .........................
Kiosk.zip, discussed ..........  

L

languages ....................... 
Linux ...............................
localization ........................
Login() function, discussed ..
Login.cgi, discussed ...........
LoginPage.html, customizing
LoginPage.thtml, discussed. 
LoginPageErrorCode .......... 
LoginPageErrorMessage variable, discussed
Logout.thtml, discussed ......
M
Mac OS X .........................
mathematical operators, defined
Meeting.zip ......................

N
NewPin.thtml, discussed ......
NextToken.thtml, discussed .

P
PasswordChange.thtml, discussed
PasswordExpiration.thtml, discussed
Perl AppConfig...................
PERL directive, restriction....
    custom sign in pages......
PROCESS ........................

R
RAWPERL directive, restriction
RealmList variable, discussed
recallLastRealmUsed() function, discussed
RSA Soft ID client ............
RSA SoftID client, custom pages

S
sample templates
    downloading ................
Sample.zip.....................
samples.zip, discussed .......
secid_pinselectmode variable, discussed
secid_pinserr variable, discussed
Secure Application Manager
    J-SAM
        custom pages ..........
SET directive, discussed......
SetLastRealm(sValue) function, discussed
ShowSystemPin.thtml, discussed
sign-in page, customizing.....
SigninNotifPostAuth.thtml, discussed
SigninNotifPreAuth.thtml, discussed
SM-NewPinSelect.thtml, discussed
SM-NewPinSystem.thtml, discussed
SM-NewUserPin.thtml, discussed
SM-NextToken.thtml, discussed
SoftID.zip ........................
SoftID.zip, discussed ........
SSL.thtml, discussed ..........
support, technical See technical support

SWITCH directive, discussed

T
TAGS directive, restriction...
technical support
    contacting PSGSC...........
template comments ...........
Template Toolkit.............
Template Toolkit documentation
template toolkit language, discussed
template variable
    LoginPageErrorCode ......
    thtml files .................
tpage ........................
ttree ........................
tz_offset variable, discussed
tz_offset variable, discussed

U
USE directive, restriction .....