JUNIPER INTEGRATION

HOW TO USE PINSAFE TO AUTHENTICATE A JUNIPER SSL VPN

SWIVEL SECURE
VICTORIA AVENUE
HARROGATE
HG1 1EL

JUNIPER INTEGRATION NOTE

CONTENTS

HISTORY

ISSUE	DETAILS	DATE
1.0	First issue	Sep 07
2.0	Updated for Juniper Version 6	Nov 07

INTRODUCTION

This document outlines the steps required to integrate the Juniper SSL Clientless VPN with Swivel PINsafe. The Juniper SA servers are able to use external RADIUS servers for providing authentication, and PINsafe servers are able to provide RADIUS authentication, so this forms the basis for the integration approach. (The Authentication Realm configuration section below describes how to achieve the RADIUS configuration).

PINsafe users can use either PINsafe's Single Channel (Turing, Pattern) or Dual Channel (SMS, J2ME) methods to retrieve Security Strings. The security strings are then combined with the user's PIN to extract a One-Time Code (OTC) which the user enters as their authentication credential.

With the default Dual Channel methods, the user already holds one or more Security Strings on their mobile device (and can request more at any time) so with the Juniper configured to use the matching PINsafe server for RADIUS authentication, no further integration is required.

If the Dual Channel authentication is in "on-demand" mode, this may require the VPN login page to include a button that the user can press in order to request a security string to be sent to them.

With Single Channel methods, the user must be presented with a Turing or Pattern image at sign-in time (representing a single time-limited Security String), so they can extract their OTC. The Single Channel Sign-in Page section below describes how to achieve this.

The basis of the sign-in page modification is the inclusion of some javascript on the page. This script requests the image, and this script runs within the client browser. Therefore the basic solution requires PINsafe to be accessible from the internet for image delivery. This access can be protected by a proxy if required. Alternatively you can use the enhancement described in the separate Juniper Integration Enhancement document.

PREQUISITES

You require a Juniper VPN server with correct licenses installed to modify and upload the login page. A PINsafe server is required to perform authentication.

If the Single Channel Enhancement is not used to mask the PINsafe server IP address then the PINsafe server must be accessible from the internet to provide the Single Channel images.

The Juniper server used for this document was NetScreen-SA-2000 Advanced - 100 Simultaneous Users, with System Version 5.3R3.1 Release (build 10741).

INTEGRATION STEPS FOR TURING IMAGE

The following sections are a step by step guide to integrating PINsafe and a JUNIPER SSL VPN.

DOWNLOAD SAMPLE LOGIN PAGES

From the Signing-In menu select the Sign-in-Pages then Upload Custom Pages, then from the Upload Custom Sign-In Pages select sample, download the zip file.

It is important that you start with the sample pages for your version of Juniper, uploading modified pages from a version 5 Juniper to a Version 6 Juniper server.

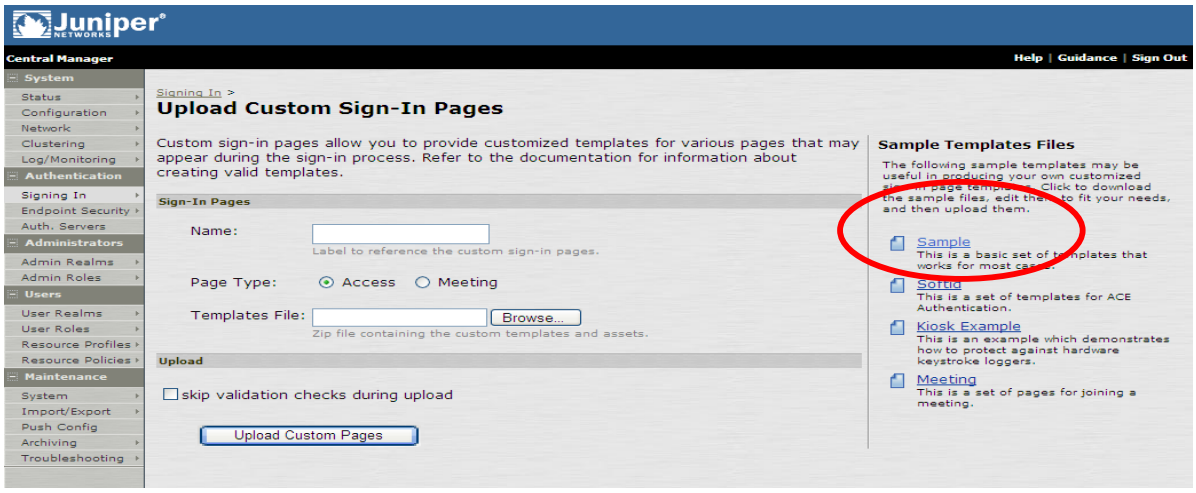

Figure 1. Download Sample Pages

MODIFYING SAMPLE PAGES TO INCLUDE PINSAFE.JS

To include the TURing image the login page (LoginPage.thtml) needs to be edited to include a section of javascript called pinsafe.js.

To do this, at the place where you want the Turing button to appear, add:

```
<SCRIPT src="pinsafe.js"></SCRIPT>
```

NOTE: This may need to be located below any tables in the authentication page. Example in the last 4 lines of the login page:

```
</table>
<SCRIPT src="pinsafe.js"></SCRIPT>
</body>
</html>
```

Or you can include in the same cell as the login button so the two buttons appear next to one another.

```
<td> </td>
      <td> </td>
        <td><script src="pinsafe.js"></script> <input type="submit"
value="<% signin %>" name="btnSubmit"> 
                                                                <% IF help_on %>
```

In addition you can add the feature whereby the TURing button is only enabled when a username has been entered. You do this by adding `onkeyup="enable_buttons();` to the form definition.

```
<blockquote><form name="frmLogin" action=login.cgi method="POST"
onkeyup="enable_buttons();" autocomplete=off onsubmit="return Login(<%
setcookies %>)">
        <input type="hidden" name="tz_offset">
```

MODIFYING PINSAFE.JS

Different versions of Juniper require different base versions of the pinsafe.js script. Different versions of the pinsafe.js file and example modified loginPage.thtml are available as .zip files from the Swivel Secure website, if you have any questions about which version to use, contact support@swivelsecure.com

The PINsafe javascript file then needs to be modified to take account of the specifics of the installation. The pinsafe.js file is described in Appendix A. The main elements that needs to be configured is the pinsafe ip address. The relevant line in the pinsafe.js is.

```
var sUrl=http://172.18.1.20:8080/pinsafe/SCImage?username=;
```

The url needs to reflect the ip address, port number and context of the PINsafe server.

You can also configure what the prompt for the one-time code entry field is by editing the following line

```
//Prompt wording...
var sOTCPrompt = "Enter your OTC:";
```

The script is designed to overwrite the standard “password” prompt with a prompt for “OTC” or whatever phrase chosen. As the page is customizable the script may need to be modified in this respect. Changes may also be required to take account in changes made by Juniper to the base version of the pages. This section explains how the script works and how to change it if required.

The script counts the number of tables on the page to determine which page it has been called from and from there works out in which cell it needs to write the one time code prompt.

```
if (tableCount == 3)
document.getElementsByTagName("td")[10].childNodes[0].nodeValue = sOTCPrompt;
else if (tableCount == 5)
   document.getElementsByTagName("td")[13].childNodes[0].nodeValue =
sOTCPrompt;
```

DETERMINING TABLE COUNT

View the page on which you want the TURing image to be displayed in a browser then view the source of the page (Page->View Source on IE7). Within the source file search for instances of the </table> tag. The number of instances of this is the tableCount for the page.

DETERMINING CELL COUNT.

Within the same file search for instances of </td>. You need to determine the value of N such that the Nth instance of the </td> tag immediately follows the current password prompt that you want to overwrite. Start at the top of the file, search for </td> and count, starting at two, how many times you need to hit find-next before you reach the cell you need.

You then need to ensure that there is a condition in the javascript that matches to table count and cell count. Therefore if the table count was 7 and the cell count was 16 you need to ensure that javascript includes the condition:

```
else if (tableCount == 7)
   document.getElementsByTagName("td")[16].childNodes[0].nodeValue =
sOTCPrompt;
```

UPLOADING MODIFIED PAGES

The modified login pages and the modified pinsafe.js file now need to be zipped into a single file. They then need to be uploaded onto the Juniper server

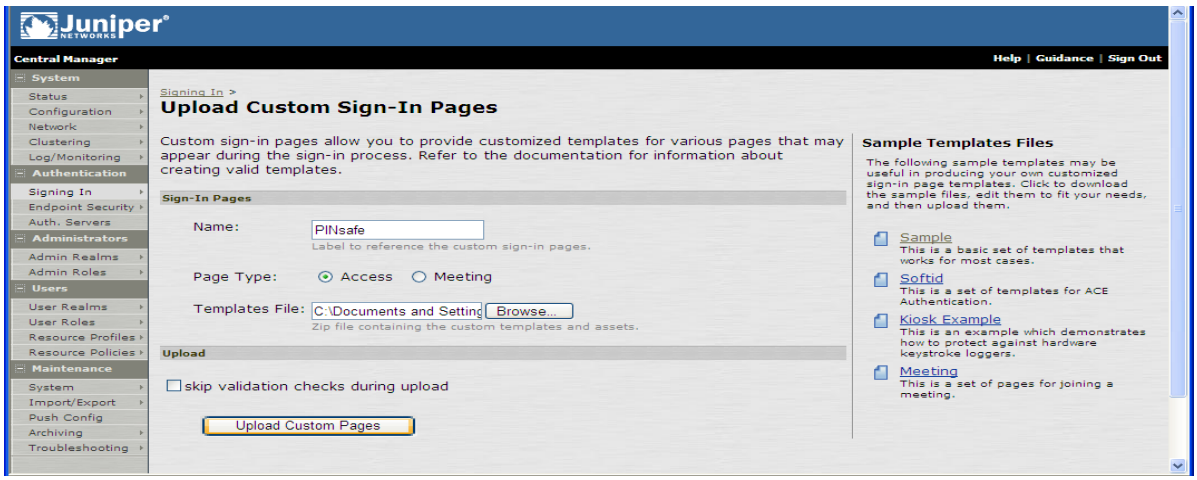

Figure 2. Uploading Customised pages

AUTHENTICATION SERVER CONFIGURATION

A new (RADIUS) authentication server needs to be created by selecting Authentication servers, with the IP address of the PINsafe server being used for the integration and the shared secret key. (A corresponding NAS entry needs to be created on the PINsafe server) The NAS-IP-Address can be used to define an IP address of the Juniper SSL to be used for RADIUS authentication, leaving it blank will use the default internal address.

Tick the box to select Users authenticate using tokens or one time passwords.

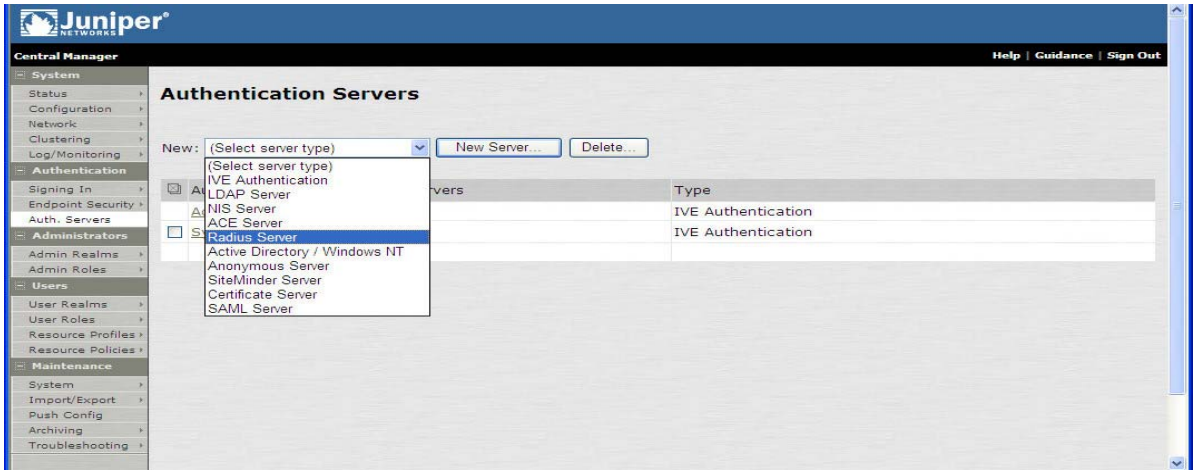

Figure 3. New Server Screen

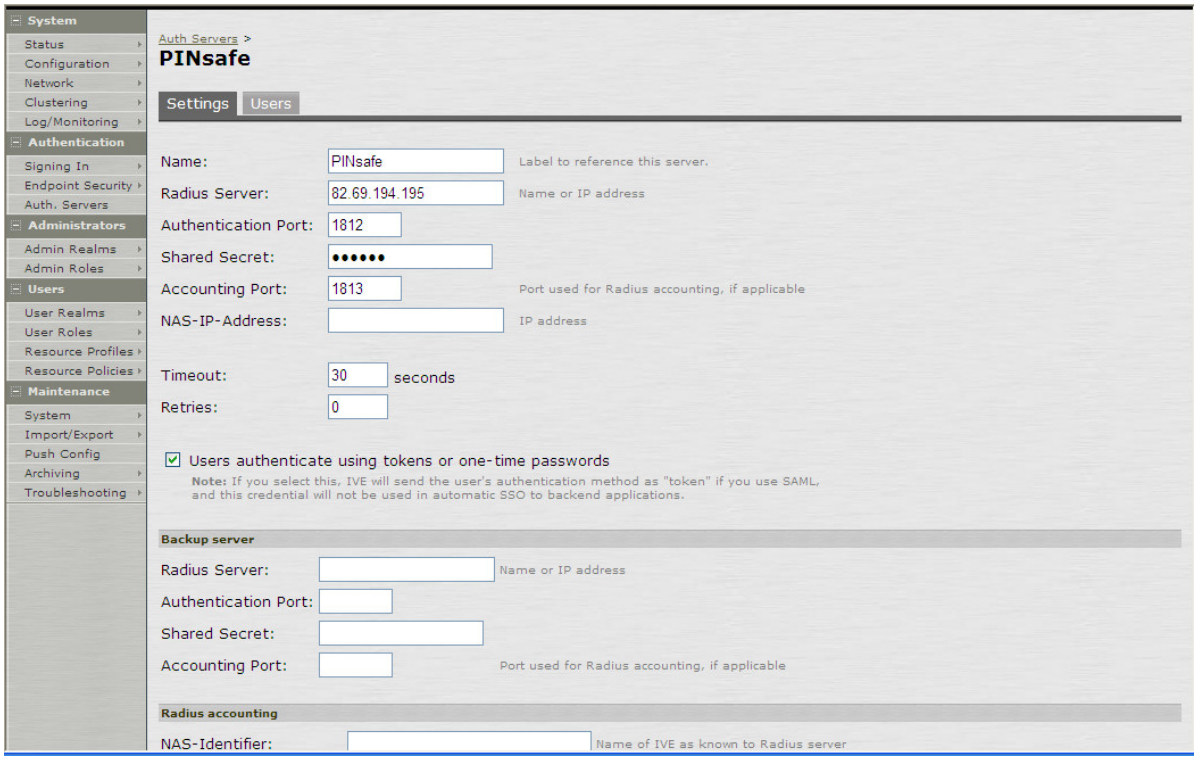

Figure 4. Configuring Authentication Server

DEFINING REALMS AND ROLES

Now that the authentication server has been configured, the SSL VPN has to be configured so it knows when to use the PINsafe server.

A new PINsafe Realm needs to be created by selecting User Realms, the New. The configured this realm to use the PINsafe Authentication Server.

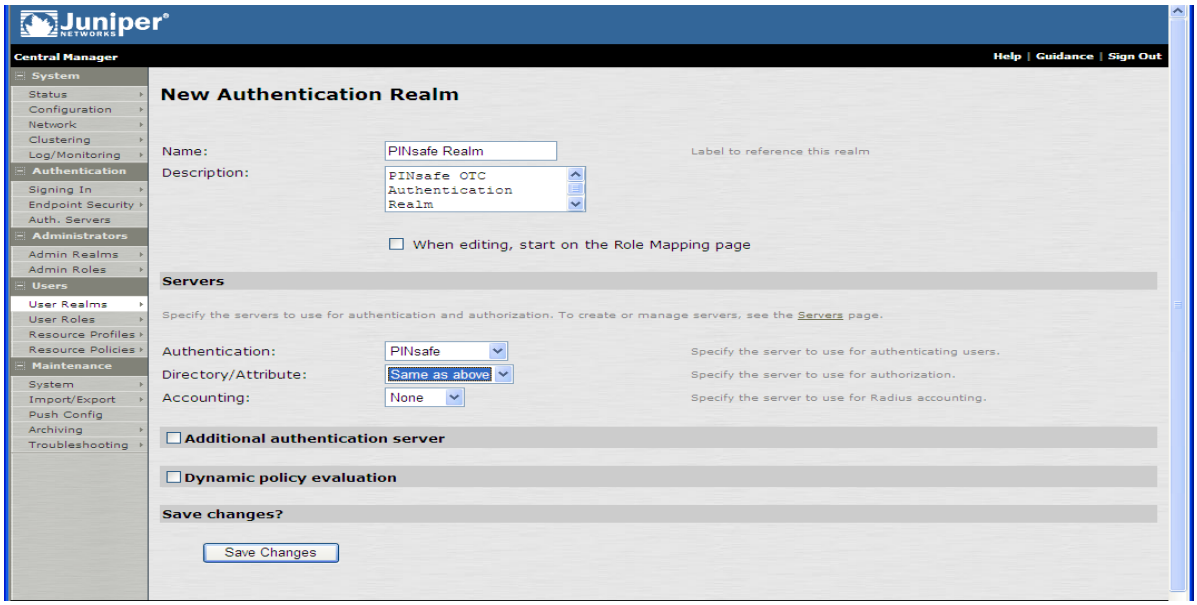

Figure 5. Realm configuration screen

Basic Role Mappings can be defined so that authentication users were assumed to be members of appropriate role group.

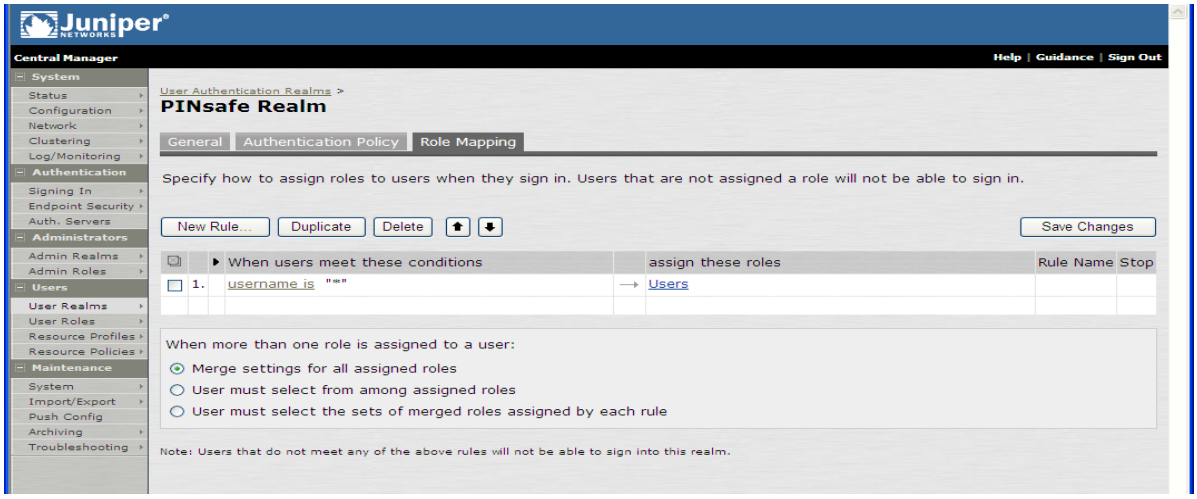

Figure 6. Realm-role mappings

Then you need to configure the SSL VPN so that it will use the PINsafe realm when users attempt to access a specific set of urls.

The diagrams below show the sign-in policy configured to use the Swivel sign-in page and realm SwivelRadius1 for urls */pinsafe. This means that the integration can be tested by browsing to https://SAipAddress/pinsafe/ while with the original configuration was still operational at https://SAipAddress/.

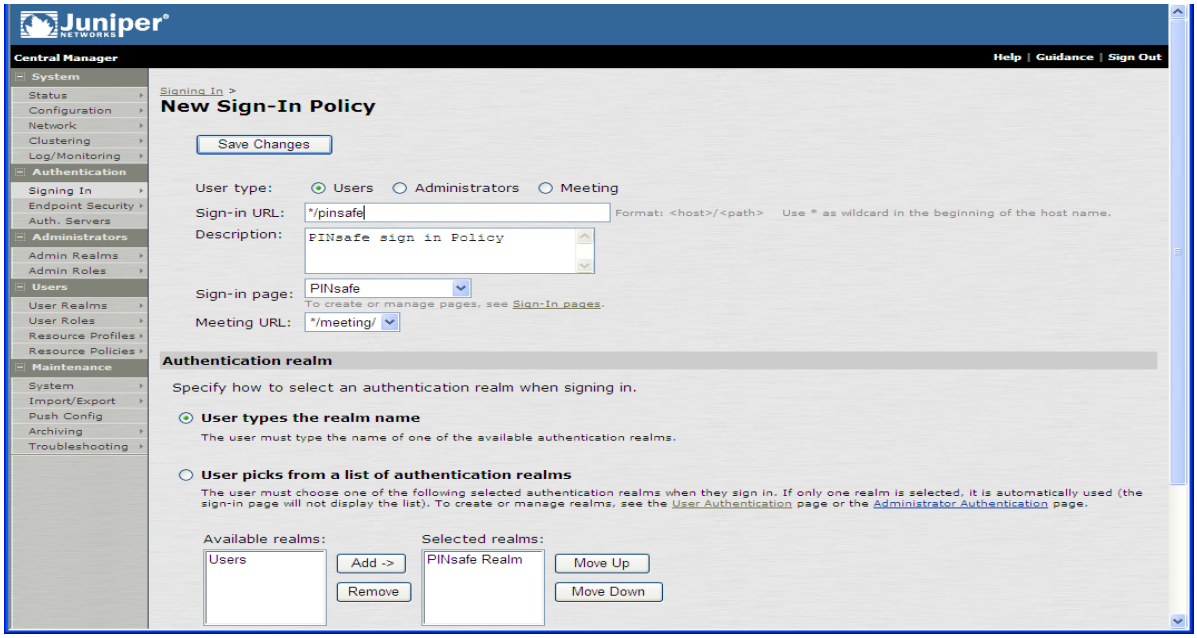

Figure 7. Sign-In Policy Settings

INTEGRATION STEPS FOR DUAL CHANNEL

For Dual Channel operation where the user is automatically sent a security string after an authentication attempt the only modification that maybe required is to change the password prompt for a one-time code prompt.

For this you only need to include edit the pinsafe.js script so that it only includes the relevant lines. For example:

```
try {
      var tableCount = document.getElementsByTagName("table").length;

      //Try and determine which page we are on from the table count and update
the appropriate
      //table cell with the OTC prompt
      if (tableCount == 3)
```

```
         document.getElementsByTagName("td")[10].childNodes[0].nodeValue =
sOTCPrompt;
      else if (tableCount == 5)
         document.getElementsByTagName("td")[13].childNodes[0].nodeValue =
sOTCPrompt;
   } catch (e){
      alert ("An error occurred in PINsafe Extensions:\n\n" + e);

   }
```

If Dual Channel is being used in “On-demand” mode then you need to add a button that allows the user to request a security string. For simplicity sake the request to send a security string to a user returns an image that indicates whether the request has been received or not by PINsafe. Therefore the pinsafe.js script is pretty much the same, with the following differences.

The button needs to be renamed to getString.

```
document.write("<input type=button name=dcMessage value=getString
onclick=ShowTuring() class='submitbutton' styleHIDDEN='visibility:hidden;
position: absolute; left:250;top:302;width:75;'>");
```

The image needs to be a Dual Channel message image, rather than a single channel image.

```
var sUrl="http://172.18.1.20:8080/pinsafe/DCMessage?username=";
```

VERIFYING INSTALLATION

Navigate to the Juniper login page (or to Juniper IP address/pinsafe if the above policy settings were used), the customised login page is visible in the addition of a **One Time Code** field and a **TURing** button. Attempting to login with a correct username and password but no one time code should result in failure. Only when a correct PINsafe one time code is entered in should the user be logged in.

TROUBLESHOOTING

The Juniper user access log (System>Log/Monitoring>User Access Log) shows entries similar to the following:

Minor	AUT21097	2004/10/08 13:47:57 - [***SAipAddress***] System - Radius Server Swivel1:	Login failed for testUserName because **host 213.152.251.227:1812 is unreachable**.

This suggests that the Juniper server is unable to reach the PINsafe server to make an authentication request, possible because of firewall restrictions.

The Juniper user access log (System>Log/Monitoring>User Access Log) shows entries similar to the following:

Info	AUT21066	2004/10/08 13:47:57 - [***SAipAddress***] testUserName (SwivelRadius1) -	Login failed from 81.157.80.225 for testUserName /SwivelRadius1 using Radius server.

This suggests the Juniper server is sending authentication requests to the PINsafe server, but they are being rejected. Examination of the PINsafe logs might reveal:

LogID	971
TimeStamp	2004-10-08 16:11:10.0
Level	0
Source	Radius Authentication
Address	
Agent	
EventID	RADIUS Log Message
EventResult	-1
Administrator	
Username	
SessionType	
Additional	<255> Access-Reject(3) LEN=51 ***SAipAddress***:12000 Access-Request by testUserName

Single Channel Image. Check the PINsafe server logs to see if a Single Channel Image message is present. If no message is present then no image request has been seen by the PINsafe server. Possible causes are network issues and the script being in the wrong position. Try in a web browser:

http://<pinsafe server URL>:8080/pinsafe/SCImage?username=demouser

If a single channel image is present check for RADIUS authentication errors.

Dual Channel and Single Channel RADIUS requests. Check the PINsafe server logs for RADIUS requests. Check name case sensitivity.

ENHANCEMENT

This section outlines the steps required to mask the PINsafe server so its IP address is not required to publicly visible when using the Single Channel Turing Image. The basis of the solution is to allow an anonymous user to access the PINsafe server during authentication. Note that this is treated as an additional session and therefore uses an additional license for process of authentication. The session will timeout and the licence released; the timeout period can be set as required, eg to five minutes.

In this way when the javascript runs in the client browser it actually requests the image via the Juniper SSL server and the SSL proxies the request. Therefore the PINsafe server only needs to be accessible from SSL server rather than the internet.

MODIFYING PINSAFE.JS

The pinsafe.js java script needs to be modified to point to access the TURing image via the Juniper server as follows:

```
//URL of radiusTuring page on the PINsafe server....

var sUrl="https://<Juniper IP hostname>/pinsafe/SCImage,DanaInfo=<IP of PINsafe
server>,Port=8080,SSO=U+?username=";
```

CREATING AN ANONYMOUS LOGIN SERVER

Select the page Authentication/Auth. Servers, from the Select New Server, set it to Anonymous Server then click on New Server, and enter a name for the server, eg PINsafeAnon

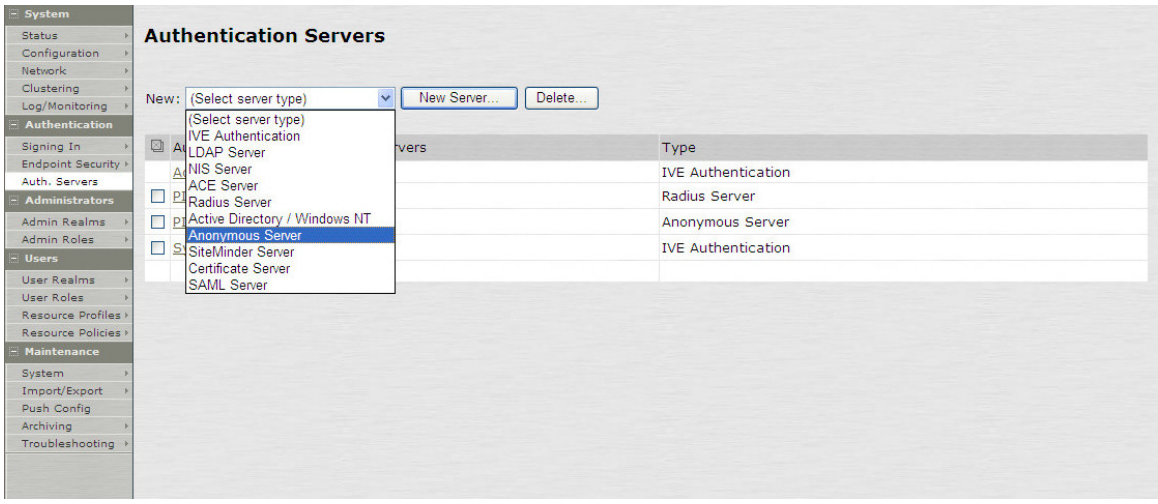

Figure 8. Creating an Anonymous Server

CREATING AN ANONYMOUS USER ROLE

Select the page Users/User Roles and from this select New Role, enter a name for the role, ensure only UI and Web are selected and then click on Save Changes.

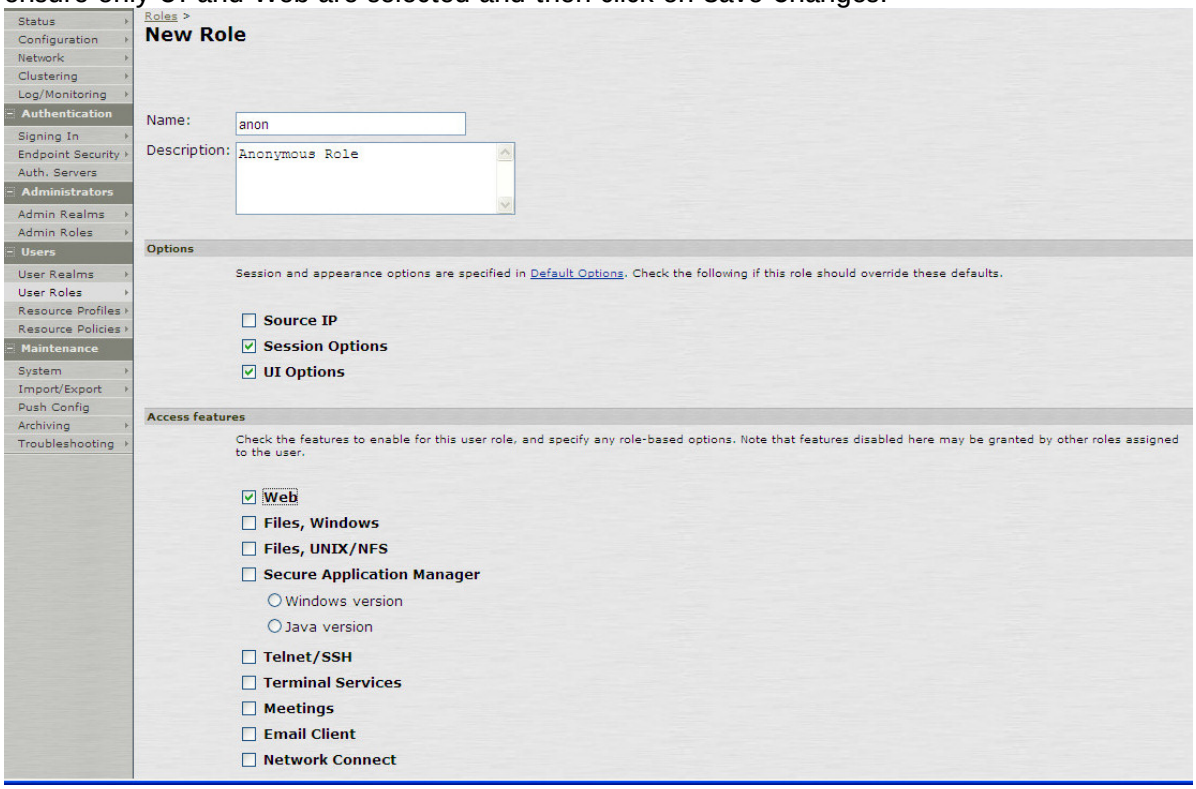

Figure 9. Creating an Anonymous Role

CREATING AN ANONYMOUS USER REALM

Select the page Users/User Realm and from this select New, enter a name for the realm, and ensure that Authentication is set to the Anonymous Authentication Server created above then click on Save Changes.

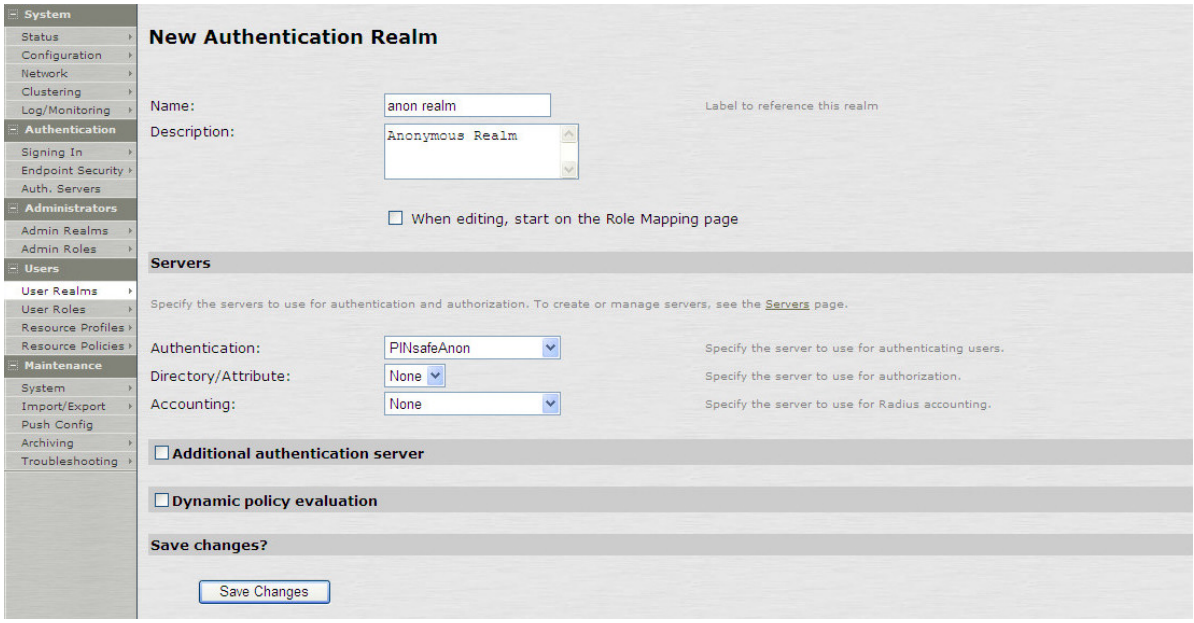

Figure 10. Creating Anonymous Realm

MAP ALL USERS TO THE ANON REALM

Select the Role Mapping page from User Realm/Anon Realm/Role Mapping, then select new. Assign Users to this realm; as shown below

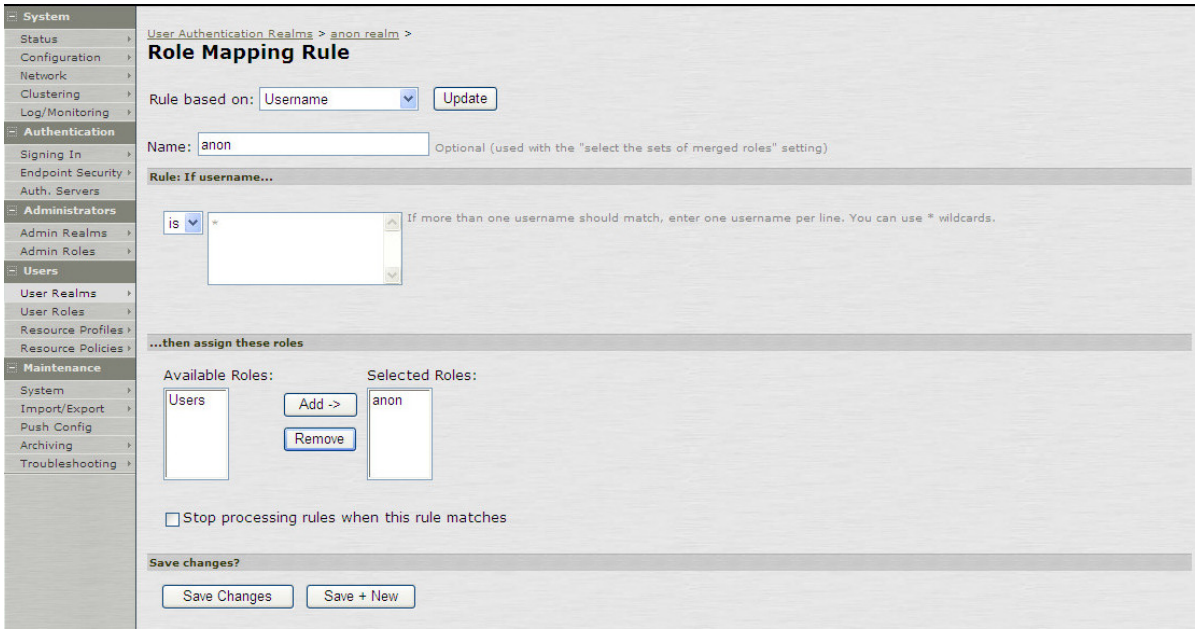

Figure 11. Mapping Users to the Anonymous Realm

CREATING A SIGN IN POLICY

Ensure that the correct signing in pages are all in place and set the sign in policy to point at the anonymous realm. Select Authentication/Signing In/Sign-in Policy, select the ./* URL or the PINsafe login page URL and set the Authentication Realm to the Anonymous realm created, then click on Save Changes.

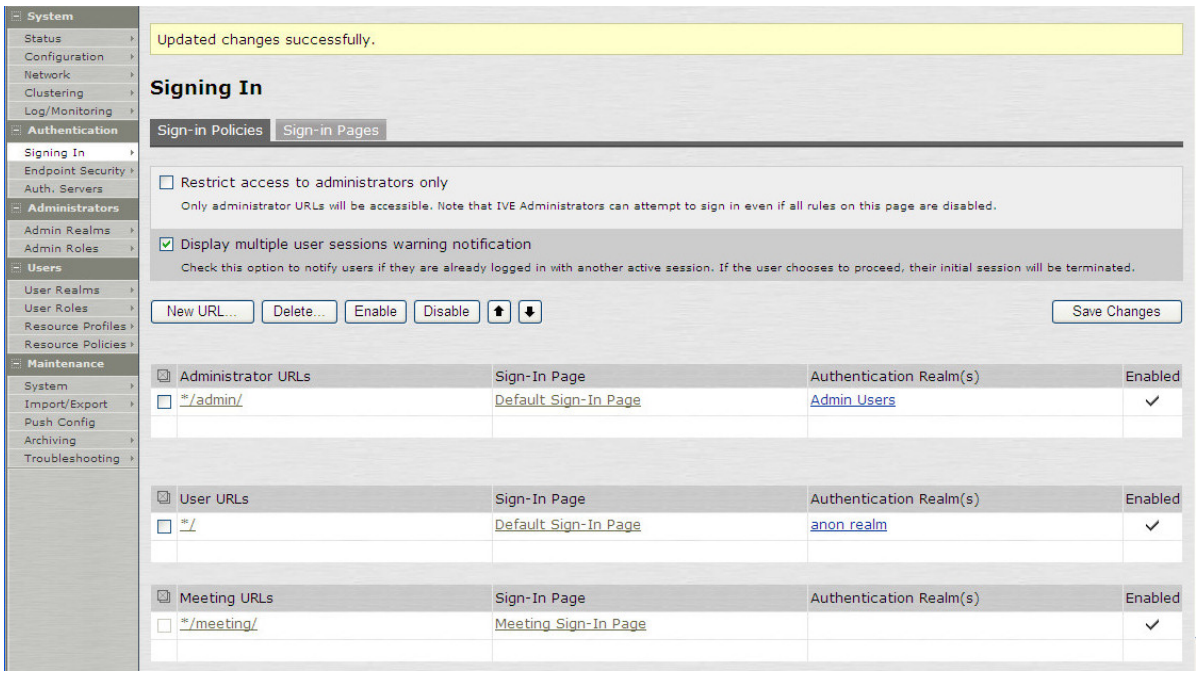

Figure 12. Setting Signing-In Policy

EDITING THE USER ROLE UI

Select the UI page from Users/User Role/anonymous role created/General/UI. On the section Start Page click on Custom page and modify the URL to represent the original sign in page which should already have been created.

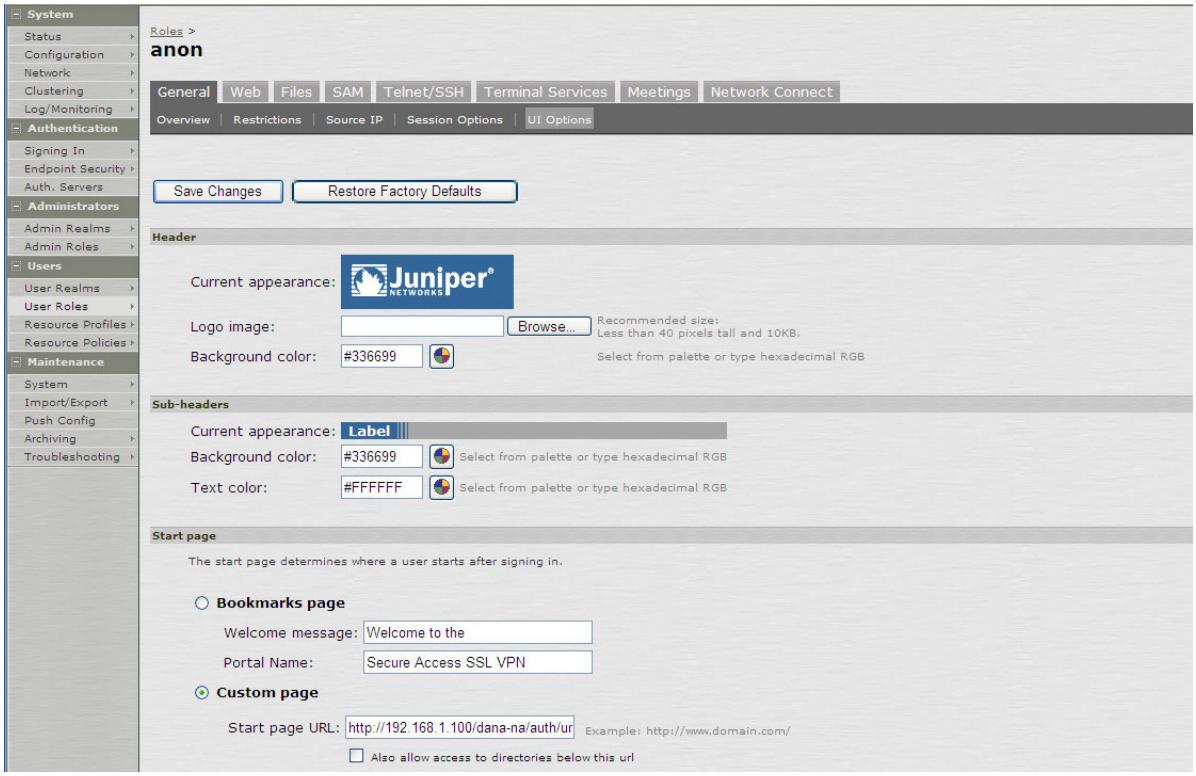

Figure 13. Setting Custom Login page

SELECTIVE REWRITING RULE

This rule prevents the address details for the server from being over-written, it must be listed before the default rewrite everything rule. Select Resource Policies/Web/Selective Rewriting and then New Policy. Enter a name for the policy and under the resources enter the SSL hostname or IP address, followed by the connection and a * for sub folders eg:

hostname:80,443/*

Select Policy Applies to Selected Roles, selecting the anonymous role, and then select the Action 'Don't rewrite content: Do not redirect to target web server', then select Save Changes.

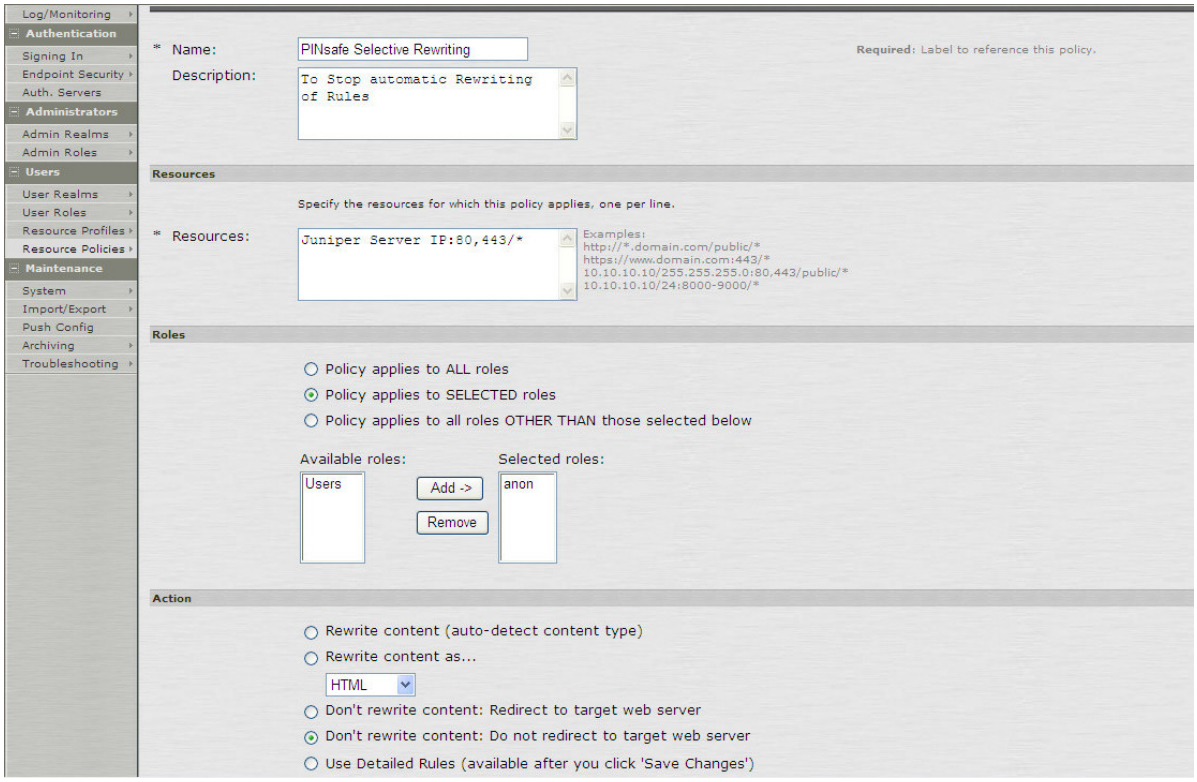

Figure 14. Selective Rewriting Configuration

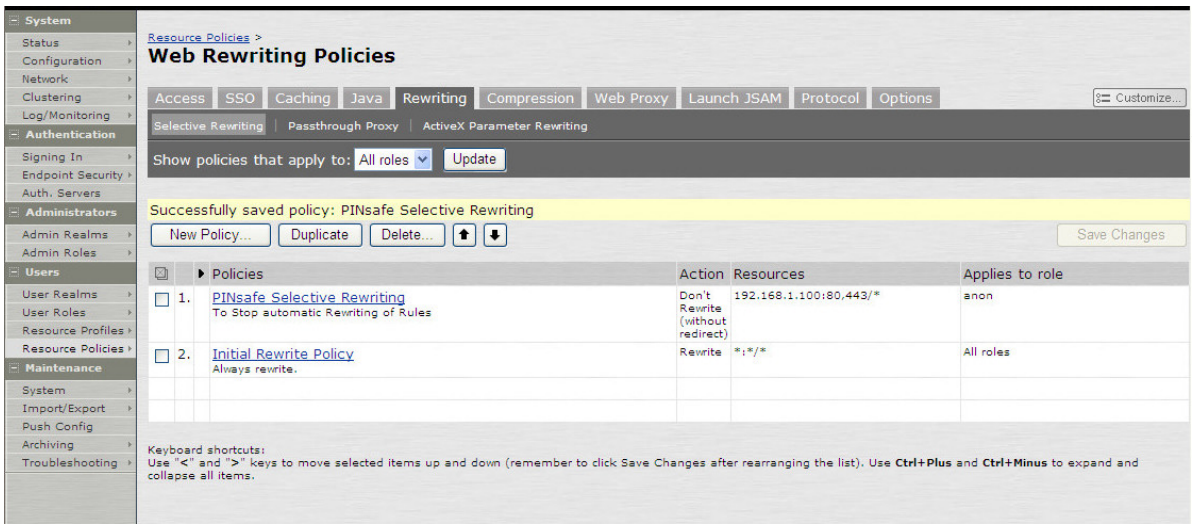

Figure 15. Showing PINsafe re-routing prior to global rewrite policy

SET IDLE TIMEOUT

Anonymous users count towards the number of concurrent sessions for each login, although they are only briefly used. To prevent the system from filling up with many concurrent anonymous users set the Idle Timeout value to a low value. Select Users/User Roles/ anonymous

role/General/Session Options. Then from the session lifetime section change the Idle Timeout to a value such as 5 minutes.

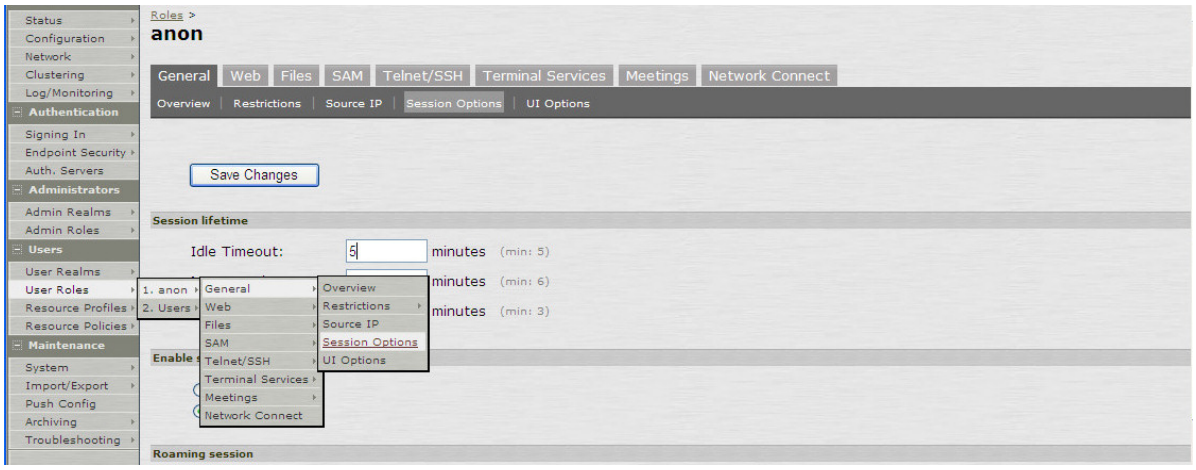

Figure 16. Setting Session Timeout

VERIFYING INSTALLATION

Navigate to the Juniper login page. The customisation is visible in the addition of a **One Time Code** field and a **TURing** button. Attempting to login with a correct username and password but no one time code should result in failure. Only when a correct PINsafe one time code is entered in should the user be logged in. The Single Channel Turing Image should not reveal the internal IP address of the PINsafe server.

The PINsafe logs will contain entries of all authentication attempts.

ADDITIONAL INFORMATION

For assistance in the PINsafe installation and configuration please contact your reseller or email Swivel Secure support at support@swivelsecure.com

This is the prompt that will appear on the login page instead of Password

```
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
//
//Configuration section.....

//Prompt wording...
var sOTCPrompt = "Enter your OTC:";

//URL of radiusTuring page on the PINsafe server....
var sUrl="http://172.18.1.20:8080/pinsafe/SCImage?username=";

//Names of the username and password texboxes in the page that's calling this
script...
var sNameOfUsernameText = "username";
var sNameOfPasswordText = "password";
```

These are the names of the fields on the login page.

```
//End configuration section.....
//
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~




//See if we're on the right 'page', ie. username field is present...
var bExists = (document.ge[illegible]Text)[0] != null);
```

Add the button, but hide it initially.
Call ShowTuring when pressed

```
document.write("<input type=button name=btnTuring value=Turing
onclick=ShowTuring() class='submitbutton' styleHIDDEN='visibility:hidden;
position: absolute; left:250;top:302;width:75;'>");
document.write("<img id=imgTuring name=imgTuring
style='visibility:hidden;position: absolute; lef[illegible]100;top:350;'>");
```

If we are on a login page, make button visible

Add the TURing image, but hide it initially

```
if (bExists){
   document.getElementsByName("btnTuring")[0].style.visibility="visible";

   try {
      var tableCount = document.getElementsByTagName("table").length;

      //Try and determine which page w[illegible]update
the appropriate
      //table cell with the OTC prompt
      if (tableCount == 3)
         document.getElementsByTagName("td")[10].childNodes[0].nodeValue =
sOTCPrompt;
      else if (tableCount == 5)
```

Workout where to write OTC prompt

```
         document.getElementsByTagName("td")[13].childNodes[0].nodeValue =
sOTCPrompt;
   } catch (e){
      alert ("An error occurred in PINsafe Extensions:\n\n" + e);

   }


}




function ShowTuring() {


if (bExists) {
   sUser=document.getElementsByName(sNameOfUsernameText)[0].value;

   if (sUser=="") {
      alert ("Please enter your username first!");
      document.getElementsByName(sNameOfUsernameText)[0].focus()
   }else{

      //Find the image using Mozilla compatible pproach...
      varImg = document.getElementById("imgTuring");

      //Set the image SRC and make it visible
      varImg.src = sUrl + sUser;
      varImg.style.visibility = "visible";

      //Alternative approach - show image in Popup
      //window.showModalDialog(sUrl +
sUser,null,"dialogWidth=305px;dialogHeight=110px;status:no;scroll:no;help:no;")

      //Set focus to the OTC input
      document.getElementsByName(sNameOfPasswordText)[0].focus()
   }

}

function enable_buttons() {
    /* Need a username to start a session */
    if (document.getElementsByName(sNameOfUsernameText)[0].value == "") {
        document.getElementById("btnTuring").disabled = true;
    } else {
        document.getElementById("btnTuring").disabled = false;
    }


}


}
```

If button pressed and username entered, retrieve image from PINsafe

Function that enables the TURing button only if the username field is not empty.