



FlexForm Documentation

(The following graphics are screen shots from Microsoft® ISA Server 2006 which is the property of Microsoft Corp. and are included here for instructive use. Some images illustrate FlexForm, which is the property of Collective Software.)

Table of Contents

FlexForm Documentation.....	1
Single Sign-on with back end Web forms.....	2
Problems.....	2
Solution.....	2
Features.....	2
Requirements.....	2
Help is Available!.....	2
Installation of FlexForm.....	4
Install Procedure.....	4
Troubleshooting.....	4
Install rolls back (with red error message at the end).....	4
Frozen or hung install.....	4
An example published application.....	6
Configuring a listener.....	7
Publishing the application.....	8
Part 1: The main publishing rule.....	8
Part 2: The FlexForm matching rule.....	10
Paths tab.....	10
FlexForm tab.....	11
Form Field Overrides.....	12
Testing the form interaction.....	13
Filter licensing.....	15
Demo/Lab mode.....	16
Troubleshooting.....	16
Support for FlexForm.....	16

Single Sign-on with back end Web forms

- Your organization uses ISA Server 2006 in a “reverse proxy” scenario for publishing an extranet.
- For security and identification, the extranet uses ISA Form authentication.
- You publish one or more web applications that also need authentication, but they use custom HTML forms, and cannot (easily) be configured to accept Integrated or Basic authentication instead.

Problems

- Authenticating at ISA is strongly recommended for perimeter security, but you don't want your users to encounter secondary login forms for the intranet services you publish.
- Removing ISA authentication would allow anonymous traffic to reach your internal web servers, which is an extreme security risk.

Solution

FlexForm from Collective Software augments the capabilities of ISA 2006, allowing it to interact with your internal back-end web login forms and seamlessly sign in the user (provided they can use the same credentials).

Features

- Configure FlexForm with normal ISA publishing rules. You “publish” each form page with separate settings for interaction.
- Select a form on the page by name attribute or by number (e.g. find the third <form> tag on the page)
- Fill in username and password values automatically.
- Automatically scans the page to learn the form's POST URL and all default values to send.
- Set specific field values or override defaults on the page.

Requirements

- ISA Server 2006
- Extranet uses ISA Form authentication, or Basic.
- Back-end forms accept the same credentials as ISA.
- Microsoft .NET Framework version 2 should be installed on each ISA server.

Help is Available!

We are always happy to help you get our software set up and working. If you have questions or need assistance understanding/configuring/testing a Collective product,

you can get in touch with our support staff quickly and easily. For the most up-to-date information, please see our Support page at <http://www.collectivesoftware.com/Support/>

Installation of FlexForm

Install Procedure

1. Close the ISA management console if it's open.
2. Execute the FlexForm.msi file. This will stop your firewall service, install the filter and interface software, register the filter, and then re-start the firewall service.
3. If you are installing over a remote desktop session, keep in mind that when the firewall service stops and restarts your RDP connection may be frozen, dropped or timed out. If an error occurs during the installation and the firewall service cannot be restarted, you will need to access the console to troubleshoot further (see below).
4. You must run the installer on each ISA server in an array separately, so they will all have the filter files installed and registered.
5. If the installation completes with no errors, then you can proceed to the configuration section.

Troubleshooting

The installation normally completes without errors. However there are a few possible failure modes that can occur for this complex install process.

Install rolls back (with red error message at the end)

If you are presented with an error message on the final screen, then check out the application event log, which often will contain details on why the installation failed. The problem may be immediately solvable from this information, or you may need to work with Collective support for additional troubleshooting assistance.

Frozen or hung install

The installer tries to start the firewall service after it is done registering the filter components. In rare cases, everything may register properly but there could still be a problem preventing the firewall service from starting. In this situation, the installation may appear to hang on the "Starting services..." item. This is because it is trying repeatedly to start the service, and failing. In fact if you look at the application event log, you will see several errors from the firewall service as it tries to start. These messages may help identify the cause of the problem.

The install should eventually give up on starting the service, but it may take a long time. If necessary, you can expedite the rollback by going into the services control panel and setting the Microsoft Firewall service to Disabled temporarily (and applying that change). This will cause the installer to quickly give up, and it should then correctly roll back the installation while leaving the firewall service down. After this happens you can then re-enable and restart the firewall service.

This kind of problem should not normally occur, and will probably require additional troubleshooting by Collective support. However if you are able to fix the problem you

can re-run the install safely after completing this procedure.

An example published application



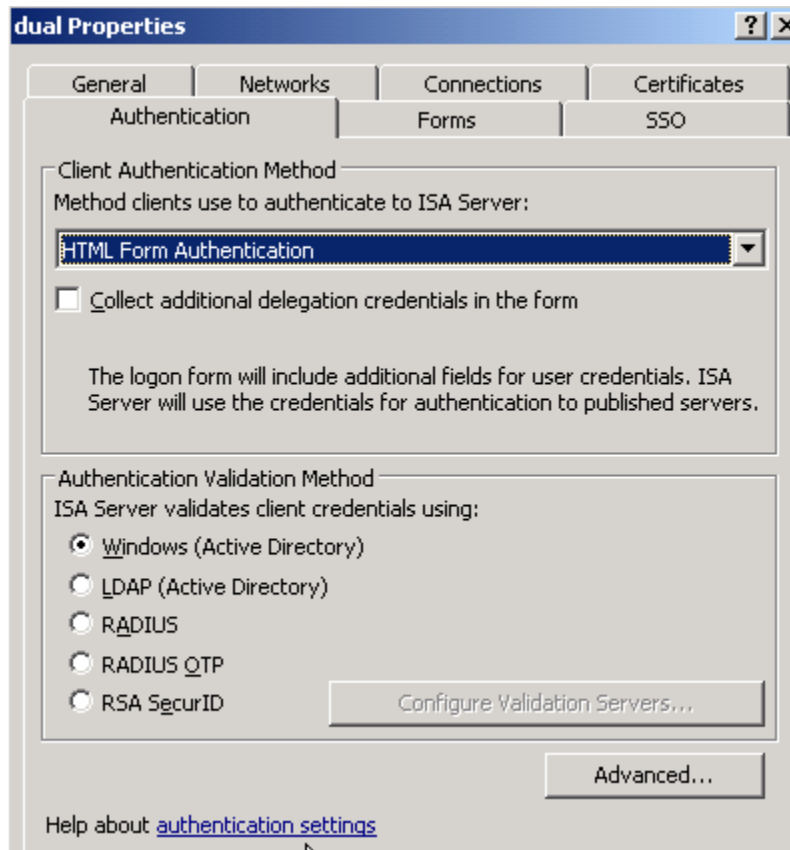
For the purposes of this document, we will pretend the public website “forums.isaserver.org” is on our own intranet. It will be the application we publish for single sign-on with ISA. This is a good selection for an example site that requires authentication and some customization of values, and you can visit the login pages yourself to better understand the example.

Notes:

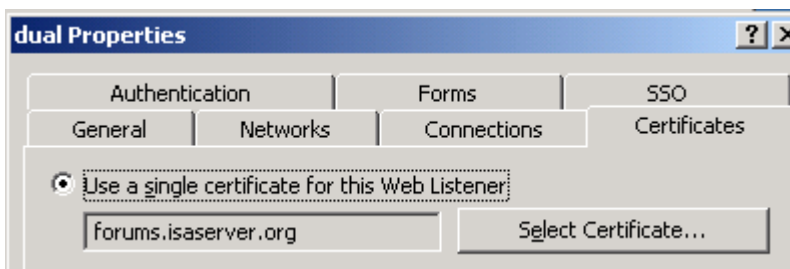
- For this example to work, a user account was created on the ISA box having the same username and password as the author's identity on the isaserver forum site. In general, this odd step would not be needed, assuming your published applications use the same user database as ISA.
- In order for the client browser to resolve the name “forums.isaserver.org” to our ISA machine, an entry was added in the hosts file. As above, this step is only needed because of the fabricated example of publishing a site that is not really on our LAN.

Configuring a listener

In this example, we will use an HTTPS listener using Form authentication to AD:



and an appropriate certificate:

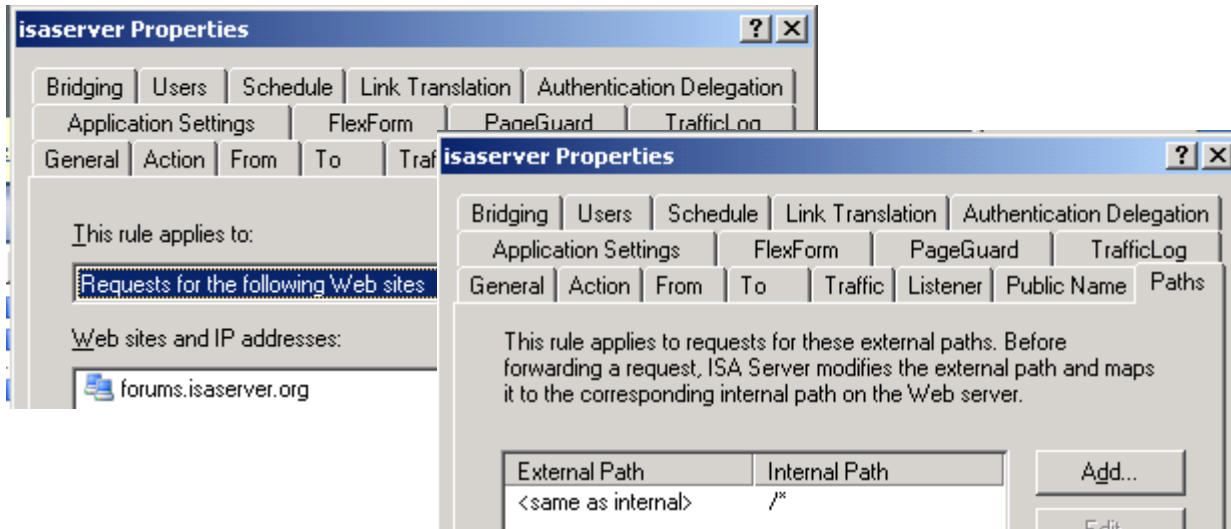
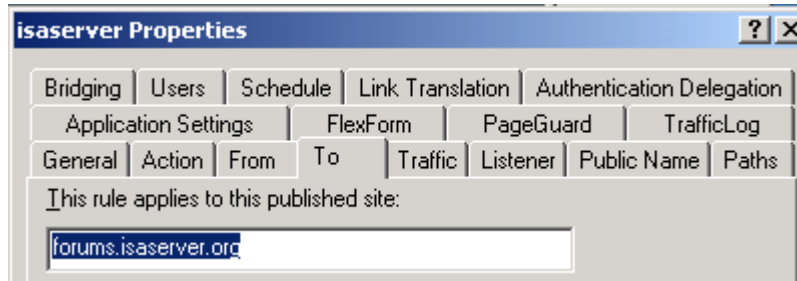


Since we do not control the domain in this example, we are just using a certificate from our own AD. In a real scenario, this certificate will generally be acquired from a trusted root source on the Internet.

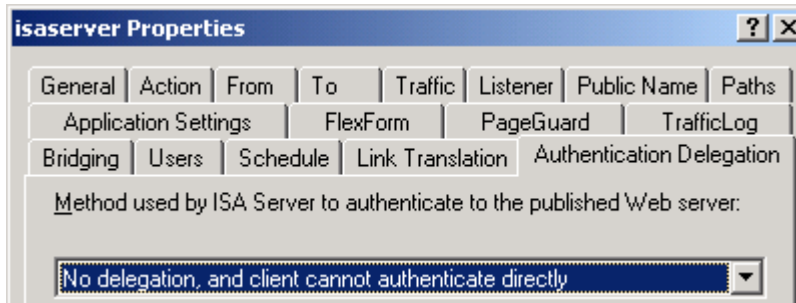
Publishing the application

Part 1: The main publishing rule

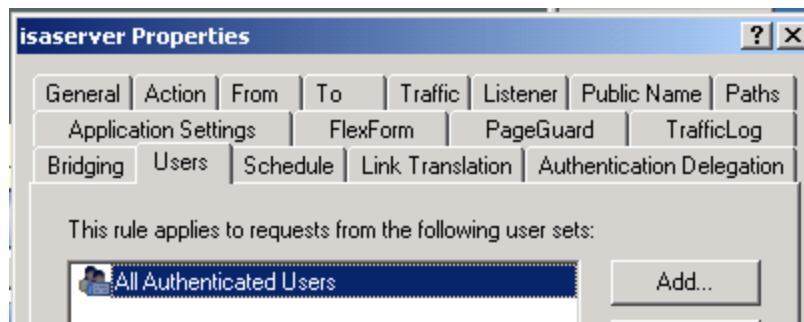
Set up a publishing rule for your application. You may already have this configured and working.



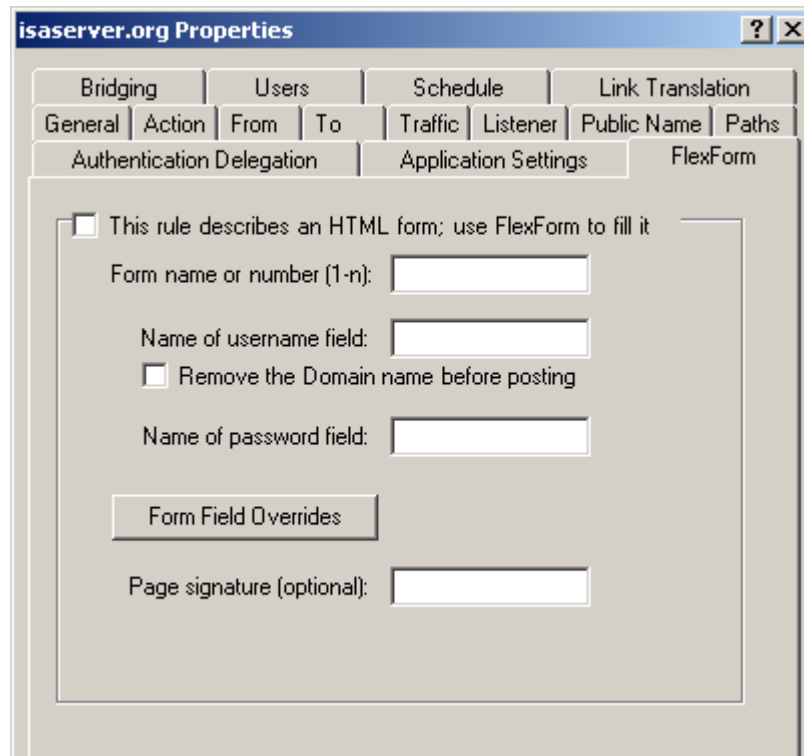
Since we will be using FlexForm to log in to our application, leave delegation disabled:



And don't forget to require authenticated users:



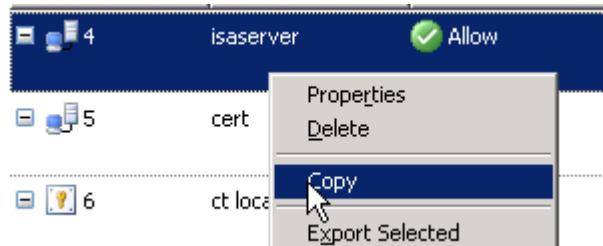
Notice that we *do not* configure any values in the FlexForm tab for this rule. That is because this rule will publish the entire application. In a moment we will configure another rule to match just the login information.



Part 2: The FlexForm matching rule

We want to treat the page containing the login form differently. Instead of sending it to the user's browser, we will send it to FlexForm for automatic processing.

Create a *new rule* by copying the rule from part 1:

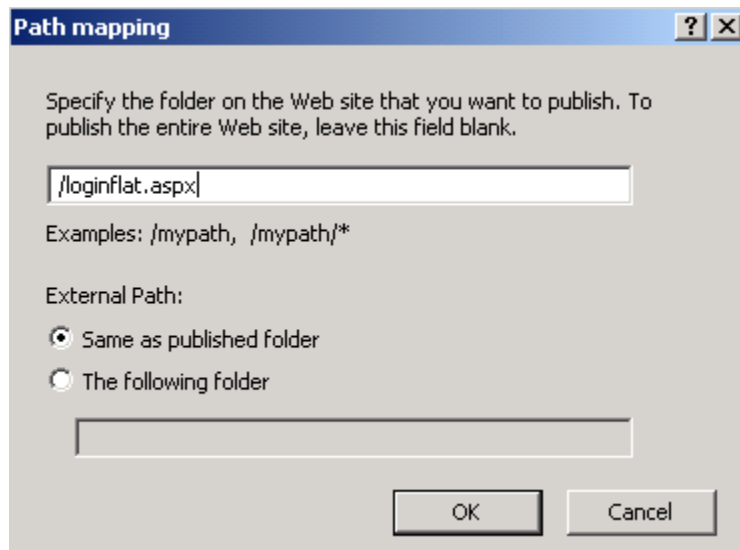


and then Paste it. The new rule should appear **above** the original. If not, then **move it up**. It is important that the FlexForm rule match first, so that FlexForm can see the login page. Otherwise it will just be served as a normal page to the browser.

The configuration of the FlexForm rule is similar to the main rule (which is why we copied it). These are the important differences:

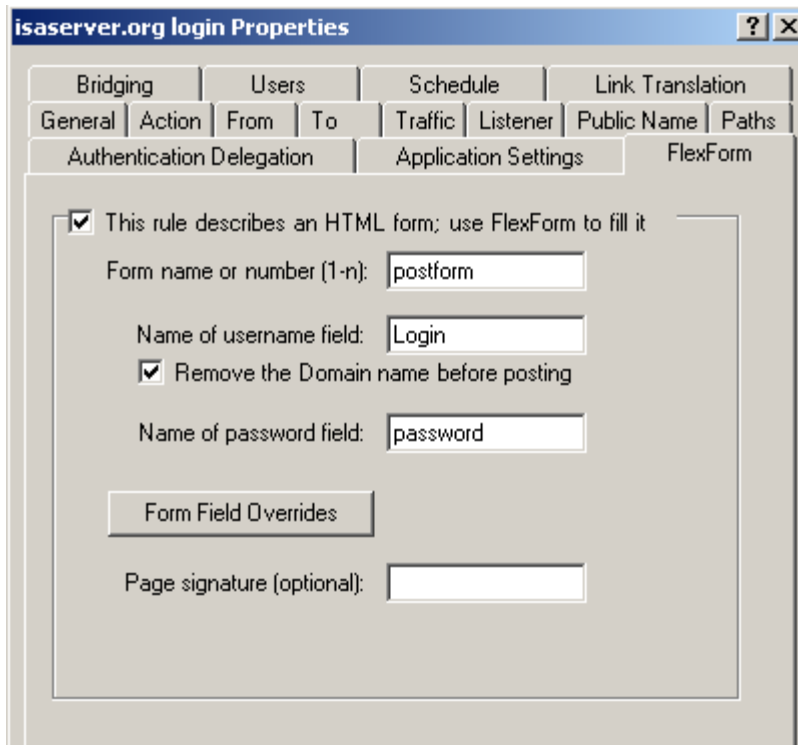
Paths tab

Instead of /*, this time we want to publish only the exact path to the login form:

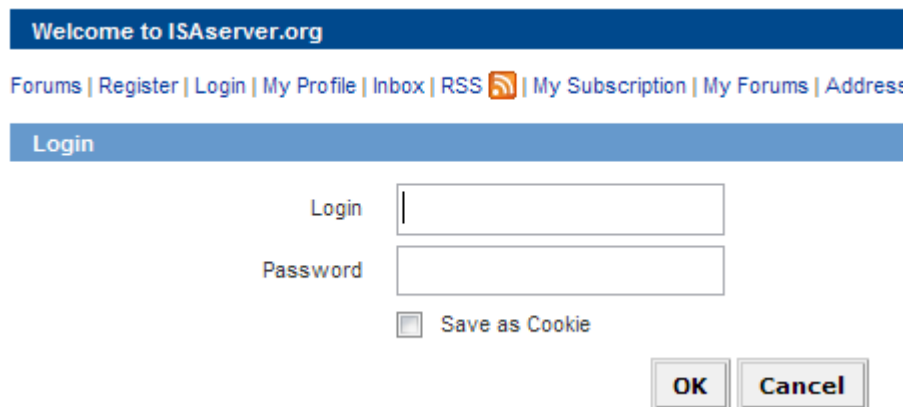


This mapping is what will be used to determine if a request is for FlexForm or not. If you specify too far general of a match here, then requests may be incorrectly matched here instead of in the rule below.

FlexForm tab



To correctly fill out this tab, you need to understand the HTML of your application's login form:



Look at the View->Source and find the correct <form> tag:

```
<form class="tagv-style-form" method="post" onsubmit="return validateLogin(this)" AUTOCOMPLETE="off" name="postform">
```

```
<label>  
  <strong>Login </strong>  
  <input class="i-text" type='text' name='Login' size='25' value=''>  
</label>
```

```
<label>  
  <strong>Password</strong>  
  <input class="i-text" type='password' name='password' size='25'
```

```
value=''>
</label>
```

...

Notice the form has a name attribute of “postform”, so we will specify that. If the form had no name attribute, we could enter the number “2” into the field, since this is the second form on the html page.

Similarly, find the input tags for username and password, and enter their name attributes into the FlexForm tab.

In this case we know the application will not want to see logins in the form “domain\username” but just “username” so select the checkbox to remove the prefix.

Form Field Overrides

Note that the above web form has several other inputs as well (many “hidden” ones). FlexForm will learn all the default values for input and select tags in the form, and post the form using these values.

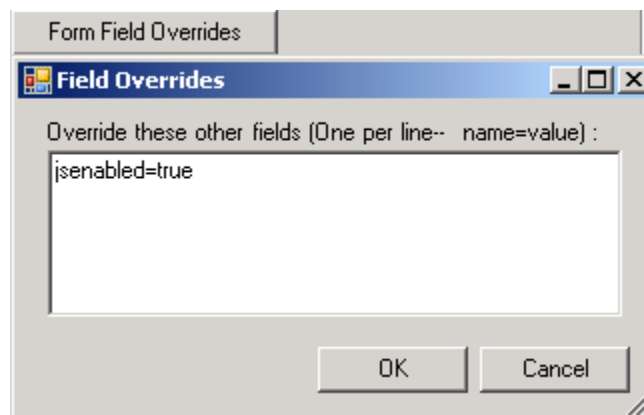
But in certain cases you may need to select a value other than what loads in the form by default. In the case of this site, it checks that javascript is enabled by having a script on the page change this input to a non-default value:

```
<input type="hidden" name="jsenabled" value="false">
```

When using FlexForm, the user's browser never sees this form, and ISA does not have a javascript interpreter of course. So if left alone, this value of “false” will be sent to the web server and it will complain to the user that script must be enabled.

This check is well-meaning but causes a problem for automation of the application's login form (remember for the example we are still pretending this is an app on our intranet).

You can force form fields to have whatever values you want by clicking the Form Field Overrides button, and filling them in:



We say “jsenabled=true” here, and this modification will be substituted into the form when it is posted. If your form needs several changes, you can list one on each line of the text entry field.

Page signature

Some web applications use the same URL for the login form as they do for other purposes. This behavior will cause FlexForm to be confused; it will try to load the form each time it encounters the URL, even if the page is not in fact displaying a login form. To mitigate this problem:

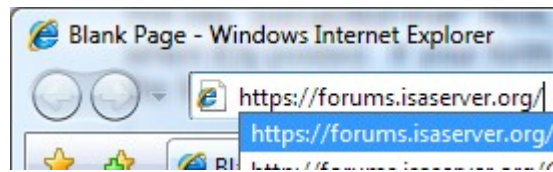
- Find some html that is only displayed when the login form is presented, and not for other uses of the URL.
- Paste the unique characters into the “Page signature” field.

An exact, case sensitive match is performed against this field, and if the page does not contain that string, it will not be treated as a logon form.

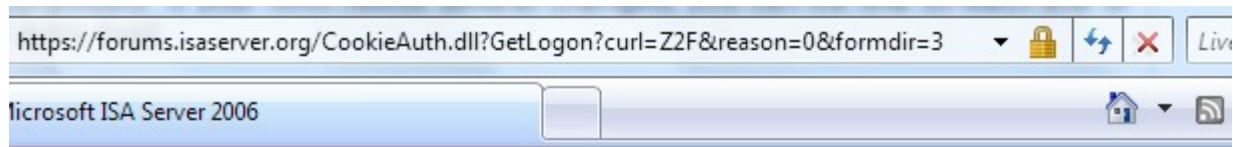
Note that this feature should not be used as a substitute for the best practice of having a separate FlexForm rule to narrow down the filter processing. When FlexForm is active on a rule, all its traffic must be cached by the filter to operate on (or test for) the login form. Running an entire site through a FlexForm rule could have a significant impact on performance, and cause unexpected results.

Testing the form interaction

We have "forums.isaserver.org" in the hosts file on the client. So entering:



we are directed to the ISA login page as expected:

A screenshot of the Microsoft Internet Security & Acceleration Server 2006 login page. The page has a blue background and contains the following elements:

- Microsoft Internet Security & Acceleration Server 2006 logo
- Security (show explanation)
- Two radio buttons: "This is a public or shared computer" (selected) and "This is a private computer"
- Domain\user name: [text input field]
- Password: [password input field]
- Log On button
- © 2006 Microsoft Corporation. All rights reserved.

We enter an account name and password that is known to ISA, and is also the same name and password used by the forum site.

We are now authenticated to ISA and so it passes us to the back end web server:

ISAserver.org

ter | [Login](#) | [My Profile](#) | [Inbox](#) | [RSS](#) | [My Subscription](#) | [My Forums](#) | [Address Book](#) | [Member List](#) | [Search](#) | [FAQ](#) | [Ticket List](#) | [Log Out](#)
398

Logged in as: Guest

[ies](#) | [Mark All Forums read](#) | [Close All Categories](#)

This server only requires authentication for certain operations, so it begins logged in as Guest.

Most applications will send you to the login page any time you try to do something that requires identity or access privilege. This site we are testing is a bit simplistic-- it requires an explicit visit to the login page, otherwise the authenticating operations just return a permission denied.

There's nothing that FlexForm could do to improve that... We could mitigate the annoyance by having our extranet link to the login page itself (loginflat.aspx) rather than the root of the site. This way the app would always log in first.

In any event, we are at the root page now and want to log in. FlexForm is properly configured to intercept the login page, so clicking LOGIN:



immediately produces the message

Login Successful!

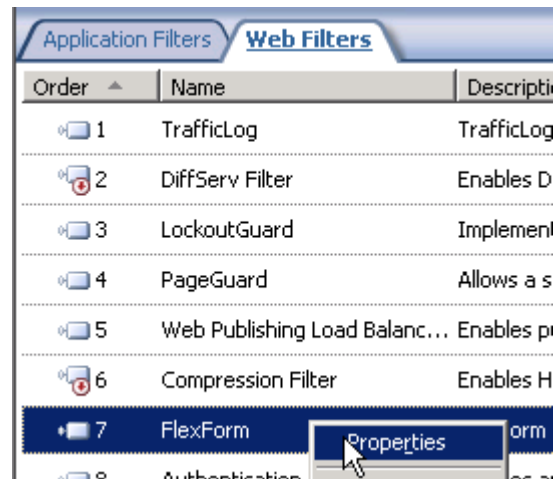
and the web server redirects us back to our destination page, now authenticated on the forum.

How has this occurred? Here are the steps taken:

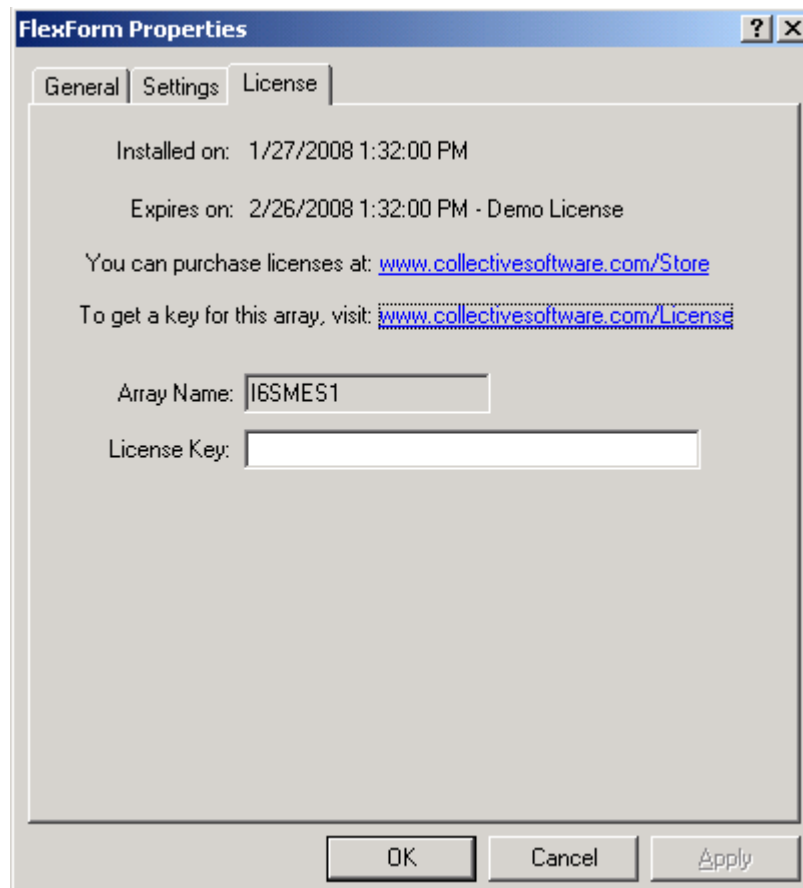
- The user logs in to the ISA form page, and FlexForm retains an encrypted copy of the user's password in memory for later use.
- The browser requests `forums.isaserver.org/loginflat.aspx`.
- The FlexForm rule (above the main publishing rule) sees this request and intercepts it.
- FlexForm gets the login page from the web server without showing it to the client, and learns the values that need to be posted.
- The username and password are entered, and any override values are added.
- The form is posted back to the web server, and the result is allowed back to the browser.

Filter licensing

To view your evaluation period or enter a key, go to Add-ins, Web Filters, and select FlexForm properties:



and select the License tab:



The License tab is used to check how long remains in the evaluation period, and to activate a permanent license.

To be eligible for a license key, you need to purchase license(s). You can do this on our [web store](#) or by [contacting us](#).

Once you have available license(s) you can request a key for your array (or single server) at our [licensing page](#). When requesting a license key, you will need to tell us the name of the ISA array, which is indicated on this dialog. The exact name is important, because it will be used to validate the key.

The license key is sensitive to the number of servers in the array. For example if you begin with only 2 servers in the array but plan to have 4, you can purchase 4 licenses and request a license key for a 4-server array. Then as you bring future servers online, they will be licensed automatically (you still need to [install the certificates](#) though.)

Warning: if you install more servers than you have licensed then the license key will be seen as invalid, and the servers will begin to operate in [demo/lab mode](#). So if you need to add more servers to a live array then you should acquire and apply your new license key in advance, so this behavior does not take place.

Demo/Lab mode

When the evaluation period expires (after 30 days) or when an invalid license key is used, the filter runs in demo/lab mode. In this mode the filter will work normally for a period of 2 hours from the starting of the Firewall Service, and then stop working after that time. This mode is meant to be useful for test labs where you don't wish to purchase licenses but still want to be able to run meaningful test setups. After 2 hours, you can restart the firewall service and the lab timer will reset again.

Troubleshooting

The first place to look if something seems to be working incorrectly is the ISA alerts tab in the Monitoring section. Often this will directly indicate the cause of the problem. This information will also be required in almost all cases if you need support.

Support for FlexForm

Collective is proud to offer support for FlexForm, whether you need help getting a configuration working, find a bug, or just have a feature question.

Support is available from our web site at <http://www.collectivesoftware.com/Support/>

- *Knowledge Base:* When our staff answers questions that will apply to the whole community, they will often create a permanent KB item to disseminate this knowledge. There is a Search feature here; you can also easily browse by topic. To get fast answers to FAQs (frequently asked questions) the knowledge base is the best place to start.
- *Support ticket:* We are always happy to help you get set up and working. If you have questions or need assistance understanding/configuring/testing a Collective product, you can get in touch with our support staff quickly and easily. For the most up-to-date information, please see our Support page.