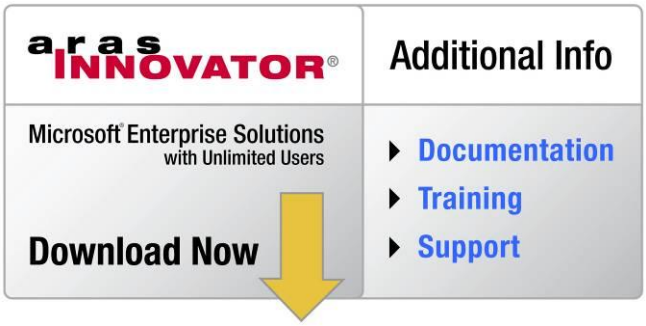
# Aras Innovator Authentication Setup


**Aras Innovator 9.0**

**Document #: 9.0.105232008**

**Last Modified:** *5/28/2008*



# ARAS CORPORATION



Aras Corporation
300 Brickstone Square
Suite 904
Andover, MA 01810

Phone: 978-691-8900
Fax: 978-794-9826

E-mail: Support@aras.com
Website: http://www.aras.com



# Table of Contents

**SEND US YOUR COMMENTS ..... 5**

**OVERVIEW ..... 6**

**1 ARAS INNOVATOR SECURITY FEATURES ..... 7**

1.1 SESSION EXPIRATION ..... 7
1.2 BLOCKING SESSIONS THAT FAIL TO AUTHENTICATE ..... 8
1.3 PASSWORD RESTRICTIONS ..... 8
*1.3.1 Password Format ..... 8*
*1.3.2 Password Expiration ..... 8*
1.4 ACCOUNT INACTIVITY REPORT ..... 9

**2 ARAS INNOVATOR LOGIN HOOKS FOR EXTERNAL AUTHENTICATION ..... 10**

2.1 ADMINISTRATIVE SETUP ..... 10
*2.1.1 Customizing the Client section of the Innovator Server ..... 10*
*2.1.2 Enabling the Logon hooks ..... 10*
*2.1.3 Configuring the Logon hooks ..... 11*
*2.1.4 Impact of Logon hooks on Workflow Activity Voting ..... 13*
2.2 EXAMPLE IMPLEMENTATION WITH ACTIVE DIRECTORY. ..... 14

**3 SECURING BUILT-IN ARAS INNOVATOR ACCOUNTS ..... 16**

**4 MIXING AUTHENTICATION METHODS ..... 17**

**5 REFERENCE DIAGRAMS ..... 19**

5.1 ARCHITECTURE ..... 19
5.2 CLIENT LOGON HOOKS AUTHENTICATION SEQUENCE, WEB SERVER MODE ..... 20
5.3 CLIENT LOGON HOOKS AUTHENTICATION SEQUENCE, PORTAL MODE ..... 21

# Send Us Your Comments

Aras Corporation welcomes your comments and suggestions on the quality and usefulness of this document. Your input is an important part of the information used for future revisions.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where and what level of detail?
- Are the examples correct? Do you need more examples?
- What features did you like most?

If you find any errors or have any other suggestions for improvement, please indicate the document title, and the chapter, section, and page number (if available).

You can send comments to us in the following ways:

- **Email:**
  Support@aras.com
  Subject: Aras Innovator Documentation

Or,

- **Postal service:**
  Aras Corporation
  300 Brickstone Square
  Suite 904
  Andover, MA 01810
  Attention: Aras Innovator Documentation

Or,

- **FAX:**
  978-794-9826
  Attn: Aras Innovator Documentation

If you would like a reply, please provide your name, email address, address, and telephone number.

If you have usage issues with the software, please visit http://www.aras.com/support/

# Overview

Aras Innovator provides the flexibility to allow administrators many options when controlling the maintenance of user logins to Aras Innovator. This document will concentrate on the Aras Innovator Security feature and the Active Directory authentication, but is not the limit of the possible configurations. Aras Innovator has many internal security features that can be enabled to maintain control of user password, and session expiration. Alternately, the Aras Innovator client logon may be customized through the use of logon hooks. These hooks provide a way to implement specialized requirements for single-sign-on, authorization control and auditing. These configurations can include standard Aras Innovator connections, leverage Web Server authentication, or use client portals for authentication. Many implementations are possible, but this document should help with the most common deployments.

Changes outlined in this document should not be made to a production instance of Aras Innovator while it is running. Please plan to implement these features only when users are not connected to the system, in a controlled deployment.

# 1 Aras Innovator Security Features

Aras Innovator Security is a set of features that allows the administrator to control actions associated with user authentication like password restrictions, session timeout, and account expiration. These features are only intended for use with users who are authenticated using Aras Innovator, and no alternate methods like Active Directory authentication. Some of these features directly conflict with other authentication methods. Also, it is recommended that logins used for purposes like the Aras Innovator Service should be excluded from these features where possible, as these users will be unable to control authentication without administrator intervention.

## 1.1 Session Expiration

Aras Innovator has the ability to require users to re-authenticate themselves after a session has timed out. By default, users never have to re-authenticate once they have logged in, but with this feature you can require all timed out sessions to do so. The changes that must be made to implement this feature only apply to the Innovator Server instance that the user is connecting to. If one database is connected to two Innovator Server instances, both must be configured, if you want both to use this feature.

First, you will need to set the session timeout to the Innovator Server. In the installation folder, edit the \Innovator\Server\web.config file. Under sessionState, set the timeout value to a positive integer in minutes. This is the number of minutes any session can go idle until timing out.

```
<sessionState
        mode="InProc"
        stateConnectionString="tcpip=127.0.0.1:42424"
        sqlConnectionString="data source=127.0.0.1;Trusted_Connection=yes"
        cookieless="false"
        timeout="480"
/>
```

Second, you will need to enable the session time-out checking on the Innovator Server. To do this, edit the InnovatorServerConfig.xml in the root of the installation folder and add the following tag:

```
  <operating_parameter key="enable_session_time_out" value="true"/>
```

If you need to disable the session time-out checking, simply set this value to “false”.

Finally, restart the World Wide Web Publishing service on the server to ensure the server cache is refreshed.

## 1.2 Blocking Sessions that Fail to Authenticate

Aras Innovator has the ability to block failed attempts to authenticate. This feature is especially useful if Aras Innovator has a public URL that may be the target of automated attempts to login. When any client, identified by IP address, fails to connect in a specified number of tries, the client will be blocked from connecting for a specified number of minutes. There are two variables that must be set to enable session blocking.

**AccountLockoutThreshold_triesNum** – This defines the number of tries a client has to authenticate before being locked out. The value should be set to a positive integer. The default value is set to -1, so that this parameter is ignored until set by the administrators.

**AccountLockoutDuration_minutes** – This defines the number of minutes a client will be locked out before being allowed to attempt to connect again. The value should be set to a positive integer. The default value is set to -1, so that this parameter is ignored until set by the administrators.

## 1.3 Password Restrictions

There are several features that control password restrictions of Aras Innovator Users.

### 1.3.1 Password Format

There are two variables that control password format. To edit these Variables, select Administration\Variables from the TOC.

**User_pwd_symbols_min_number** – This variable controls the minimum number of characters a password must contain, in total. The value should be set to a positive integer. The default value is set to -1, so that this parameter is ignored until set by the administrators. This value will not be enforced on current passwords until they are changed, all new passwords will use this variable.

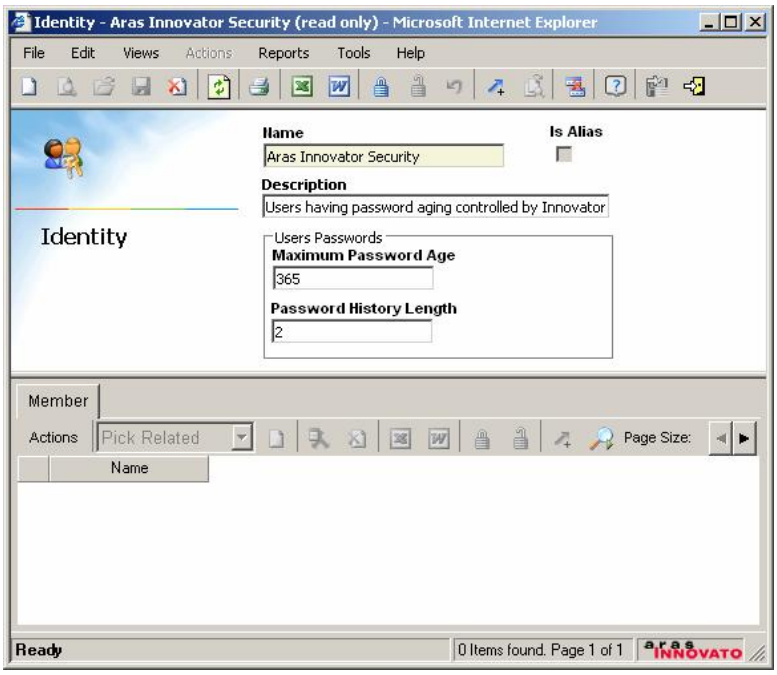
**User_pwd_digits_min_number** – This variable controls the minimum number of numerical characters a password must contain. The value should be set to a positive integer. The default value is set to -1, so that this parameter is ignored until set by the administrators. This value will not be enforced on current passwords until they are changed, all new passwords will use this variable.

As an example, if User_pwd_symbols_min_number=6 and User_pwd_digits_min_number=1, then all passwords should be at least 6 characters long and contain at least one digit. "h3lloo" would be acceptable for this set of restrictions.

### 1.3.2 Password Expiration

There are two properties that control password expiration. To edit these properties, select Administration\Identities from the TOC. Every Identity has the ability to control password expiration, but it is doubtful most installations will need this level of control. Also, administrators want to be sure to exclude any system users that might be used to allow things like the Aras Innovator Service to connect to Aras Innovator. These system users will not be able to change a password without administrative intervention. Unless administration has a specific schema in mind, Aras recommends creating an identity to

manage password aging of users, and making User Identities members of this "Aras Innovator Security" Identity to manage password aging in one location.




**Maximum Password Age** – This is the maximum number of days a user may use the same password, before they will be prompted to change their password on login. This value is blank by default, but should be set to a positive integer.

**Password History Length** – This is the number of past password the system will remember. Users may not reuse any password already in the password history. This value is blank by default, but should be set to a positive integer.

## 1.4 Account Inactivity Report

There is a report included to determine what accounts are inactive in Aras Innovator. Administrators can use this report to determine if any accounts should be disabled based on inactivity. To access this report, select Administration\Users from the TOC. Then, select Reports→Inactive Accounts from the main menu.

# 2 Aras Innovator Login Hooks for External Authentication

The Aras Innovator client logon may be customized through the use of logon hooks described in this section. These hooks provide a way to implement specialized requirements for single-sign-on, authorization control, and auditing for example. The customization is delivered as a Microsoft.NET assembly installed in the Client/bin folder. In reading this section it helps to keep in mind that this is only one possible implementation of this feature. While different features can be provided, like single-sign-on for portal mode, this section represents what Aras has had the most requests for.

The Aras.LogonHooks.WindowsAuth.dll assembly ships with the standard product and provides single-sign-on capability with Microsoft Active Directory for most customer requirements. It can be configured without programming knowledge.

## 2.1 Administrative Setup

This section outlines a series of options that the administrator can enable in an Aras Innovator instance.

### 2.1.1 Customizing the Client section of the Innovator Server

The /Client web-application portion of Aras Innovator may have its own private configuration file or it may share the configuration file with the /Server. For more detail on how to deploy distributed /Client folders, please see the *Aras Innovator – Installation Guide*.

If you have a distributed Client setup, then this could have some technical architecture implications for the operation of client logon hooks that pre-fill some of the logon form input elements.

### 2.1.2 Enabling the Logon hooks

In order to enable the logon hooks, you must first include the ClientConfig tag in the InnovatorServerConfig.xml configuration file. Found in the root of your install directory, by default.

```
<ClientConfig
  AssemblyName="Aras.LogonHooks.WindowsAuth"
  AssemblyNameType="partial"
  TypeName=" Aras.LogonHooks.WindowsAuth" />
```

The AssemblyName and TypeName attributes depend on the how the customized library was developed. An example provided by Aras and described in this document is called Aras.LogonHooks.WinAuth. Other customizations should be called by other names. These names are arbitrary. For example, "AcmeOrientalRugs.InnovatorClientConfig" would be a reasonable name.

There can be more than one such assembly (dll) provided in the Client/bin folder. However, the type names should be distinct, and only one ClientConfig element should be declared in the application configuration file.

### 2.1.3 Configuring the Logon hooks

After enabling the logon hooks, the keys for these hooks must be configured based on the assembly specified in the ClientConfig tag. The standard Aras Innovator login page uses various parameters/keys for screen customization and user authorization. The assembly defined in the <ClientConfig> implements a function that returns key value to the login page when a key parameter is passed to the function. This allows to customize, for example, authentication process by providing a custom assembly. These keys can vary based on assembly, but in this section we will outline how to configure the keys for the "Aras.Login.WindowsAuth" assembly used when Microsoft Active Directory single sign-on is desired. This extension is used to leverage the Integrated Windows Authentication and Digest Authentication for Windows Domain Servers, or any other method in the web server which ends up establishing a trusted value of the server variable LOGON_USER in the form DomainName\UserName. In order to use it you must disable anonymous access to the page /Client/Scripts/login.aspx and allow only authenticated access. This method of authentication is diagrammed in the section Client Logon Hooks Authentication Sequence, Web Server Mode.

#### *2.1.3.1 The ClientLogon attributes*

The ClientLogon tag is used in the InnovatorServerConfig.xml to configure the various options available through the "Aras.Login.WindowsAuth" extension. Below is an example of various options available, as well as the purpose of each one.

```
<ClientLogon allowed_domain_names=".*"
             allowed_domain_users=".+"
             denied_domain_users="^admin$|^root$|^vadmin$"
             allowed_direct_users="^admin$|^root$"
             debugging_password="bypass1"
             shared_secret="secret for use"
             logon_user_server_variable="LOGON_USER"
             logon_user_domain_delimiter="\"
             logon_user_domain_first="true"
             empty_logon_user_allow_direct="true"
             bypass_logon_form="false"
             bypass_logon_wait="500" />
```

**allowed_domain_names** – This is a regular expression. The domain portion of the LOGON_USER must match this expression in order to be allowed into Aras Innovator. If there is a finite list of domains to recognize then it is best to use a fixed list with the or "|" operator, for example, "^europe$|^usa$|^fareast$". The "^" character in this context means to match at the start of a string, and the "$" character means to match at the end of the string. A string without these, e.g. "east" would match "FarEast" and also "EasterIsland" and any string containing the sequence "east". The match is case insensitive.

**allowed_domain_users** – This is a regular expression. Usually it is best to keep it at ".+" which means to match one or more characters. This expression must match in order for the logon to Aras Innovator to be allowed. The username portion of the LOGON_USER is matched against this. If it matches then it becomes the login_name used to log onto Aras Innovator.

**denied_domain_users** – This is matched against the username if it passes the allowed_domain_users test. If the match is true, then access to Aras Innovator is denied.

This prevents domain users from logging in as Aras Innovator users with the same username. This option should be set to a list of special purpose Aras Innovator users. The "Innovator Admin" (username=admin) user for example is often used when batch loading data or managing AML upgrades. The "Super User" (username=root) user must be used when applying database upgrade patches to the Aras Innovator database. The "Vault Admin" (username=vadmin) user is used only by the vault server in order to access the mime type database. Other denied_domain_users might include the user used by the Aras Innovator Scheduler Service, or a test user used to review upgrades in functionality.

**allowed_direct_users** – This option limits the users allowed access to the normal logon form. These users should have known passwords. If they have a "Secret Password" then it will be impossible for these users to logon to Aras Innovator. These user accounts are often disabled except during limited periods of administrative maintenance.

**debugging_password** – This is only used during debugging. It is an alternative to the computed "Secrete Password" internal to web server validated users. During debugging of the logon process you can assign specific test users this debugging password. These users could then obtain access via web server validated authentication and/or function as allowed_direct_users, if so configured. Of course, any user with a "Secret Password" will be denied access.

**shared_secret** – This is the key to the web server validation logon process. Once a user has been authenticated by the web server, and has passed all the regular expression checks, the shared_secret is used to compute an inscrutable "Secret Password" which is used as a ticket to gain access to the other Aras Innovator server functions, including the vault server. The shared_secret is shared by the Innovator /Client and /Server applications.

**logon_user_server_variable** – This is the name of the server variable that the authentication mechanism trusts to be an authenticated user. This defaults to "LOGON_USER".

**logon_user_domain_delimiter** – This is the character that separates the domain name and user name portion of the string from the domain name portion. This defaults to "\".

**logon_user_domain_first** – This should be true when the LOGON_USER is of the form Domain\Username, but it should be false if the string is of the form Username@Domain. This defaults to "true".

**empty_logon_user_allow_direct** – This can be set to true to cause the regular Innovator login box to appear when the LOGON_USER variable is blank.

**bypass_logon_form** – This controls if the login form will be shown. This key defaults to "false". If "true", then a logon form is not shown if integrated authentication information is available. The query string ?bypass_logon_form=value can over-ride a true or false value given in the configuration file. If there are multiple database choices available in the logon form then it make sense to specify a database in the query string with ?database=value, otherwise the first database in the list will be chosen.

**bypass_logon_wait** – This is the number of milliseconds to display the logon form before automatically submitting the logon credentials, in the case of bypass_logon_form="true".

#### *2.1.3.2 Logon URL*

The normal URLs for accessing Aras Innovator will remain unchanged for end users. The username derived from the LOGON_USER, and is visible but not editable. The user will not be prompted a password, however, a choice of database is possible.

For user who must authenticate using the standard inputs, a logon URL in the form of /Client/default.aspx?username=X is also possible. For a limited configured number of names X this provides the normal logon form. In this case there is a password input. The password will be validated against the password in the Innovator database for the user X. (See "allowed_direct_users" attribute of <ClientLogon> tag in the section The ClientLogon attributes)

#### *2.1.3.3 Aras Innovator User Setup*

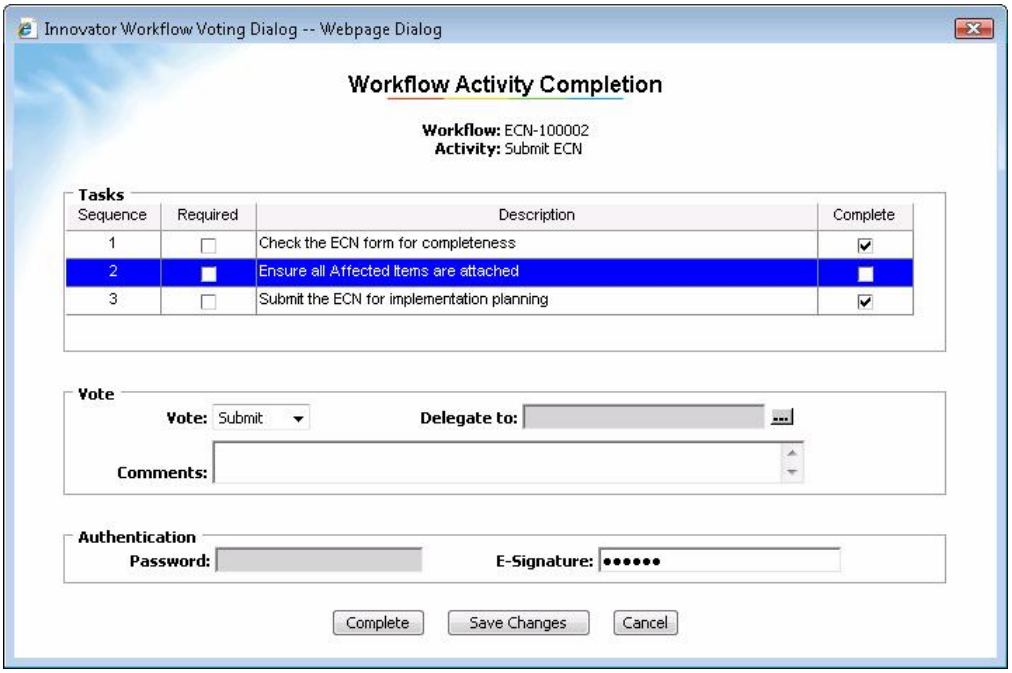
In order to use these login hook features, a user Item with the required login_name must exist in the Aras Innovator database, with logon_enabled = true. The special secret password must then be set in order for a web server authenticated LOGON_USER to gain access to Aras Innovator. If no such user exists then the Logon Form will be displayed, but upon pressing the Login button the error message "Authentication failed for X" will be seen.

After creating a user Item the administrator uses the "Reset Authentication Password" action to set the user's password.

If the shared_secret attribute in the ClientLogon configuration element is changed then the "Reset Authentication Password" action must be run or no users will be able to logon except for these configured for direct logon.

### 2.1.4 Impact of Logon hooks on Workflow Activity Voting

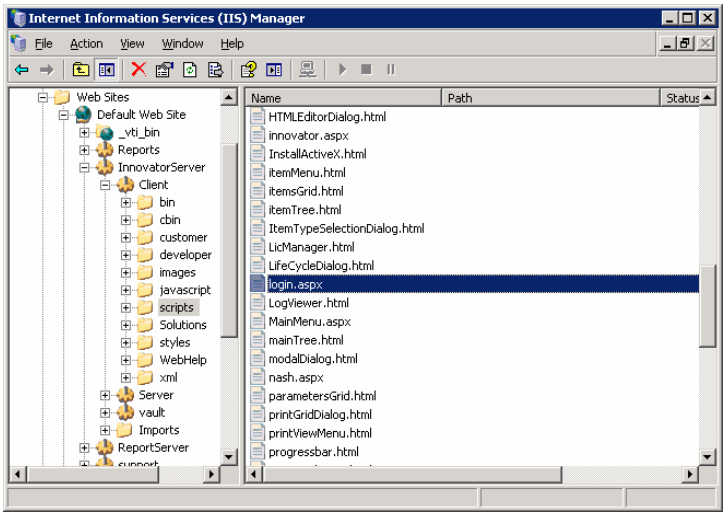
If passwords are not managed by Aras Innovator, requiring password authentication for workflow activity voting is not an option. Instead, the E-Signature option may be used. E-Signatures are a Property of the User ItemType that can be individually set by each user by selecting Tool→Change E-Signature from the main menu.

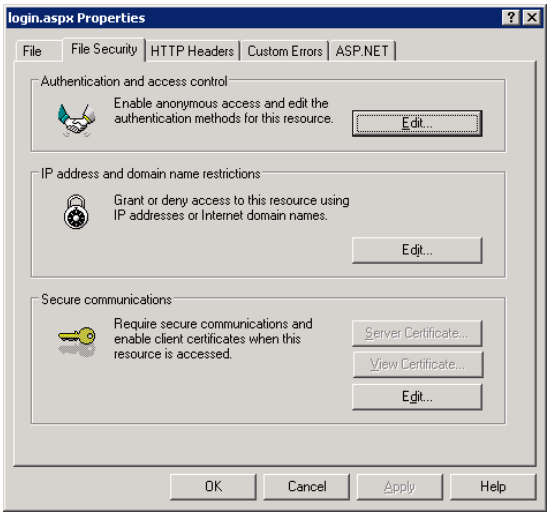


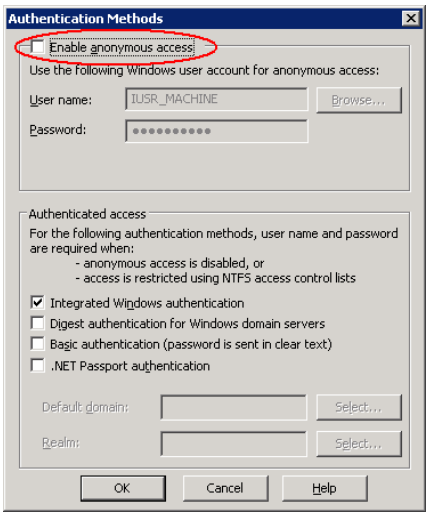
## 2.2 Example Implementation with Active Directory.

This example will walk through a basic setup for using Active Directory authentication integrated with Aras Innovator. The server must have access to the domain, and domain users must have permissions to read the /Client/scripts/login.aspx.

1) Make sure the Innovator Admin login (admin) is enabled before proceeding, just to make the process a little easier. Please remember to disable it after completion if you haven't been using it.
2) Ensure the user information is loaded into the database. (login, firstname, lastname, email, etc.)
3) Disable anonymous access to the \Client\scripts\login.aspx page in IIS




4) Set the <ClientConfig> and <ClientLogon> tags in the InnovatorServerConfig.xml file (Sample below)

```
<ClientConfig
  AssemblyName="Aras.LogonHooks.WindowsAuth"
  AssemblyNameType="partial"
  TypeName="Aras.LogonHooks.WindowsAuth" />
```

```
<ClientLogon  allowed_domain_names="^DOMAINNAMEHERE$"
              allowed_domain_users=".+"
              denied_domain_users="^admin$|^root$|^vadmin$|^PLM$"
              allowed_direct_users="^admin$|^root$"
              shared_secret="Your shared secret here"
              empty_logon_user_allow_direct="false" />
```

5) Log into innovator using a string like http://localhost/InnovatorServer/Client/default.aspx?username=admin
    a. This should display as so:


6) From the TOC select Administration\Users
7) Run search to confirm users are displayed.
8) From the main menu select Actions → Reset Authentication Passwords
9) Logout
10) Re-start IIS to flush the server cache.
11) Login using a normal URL
    a. It should display something like this:

# 3 Securing built-in Aras Innovator Accounts

The core Aras Innovator database comes with 3 built-in accounts. These are "Innovator Admin" (username=admin), "Super User" (username=root), and "Vault Admin" (username=vadmin).

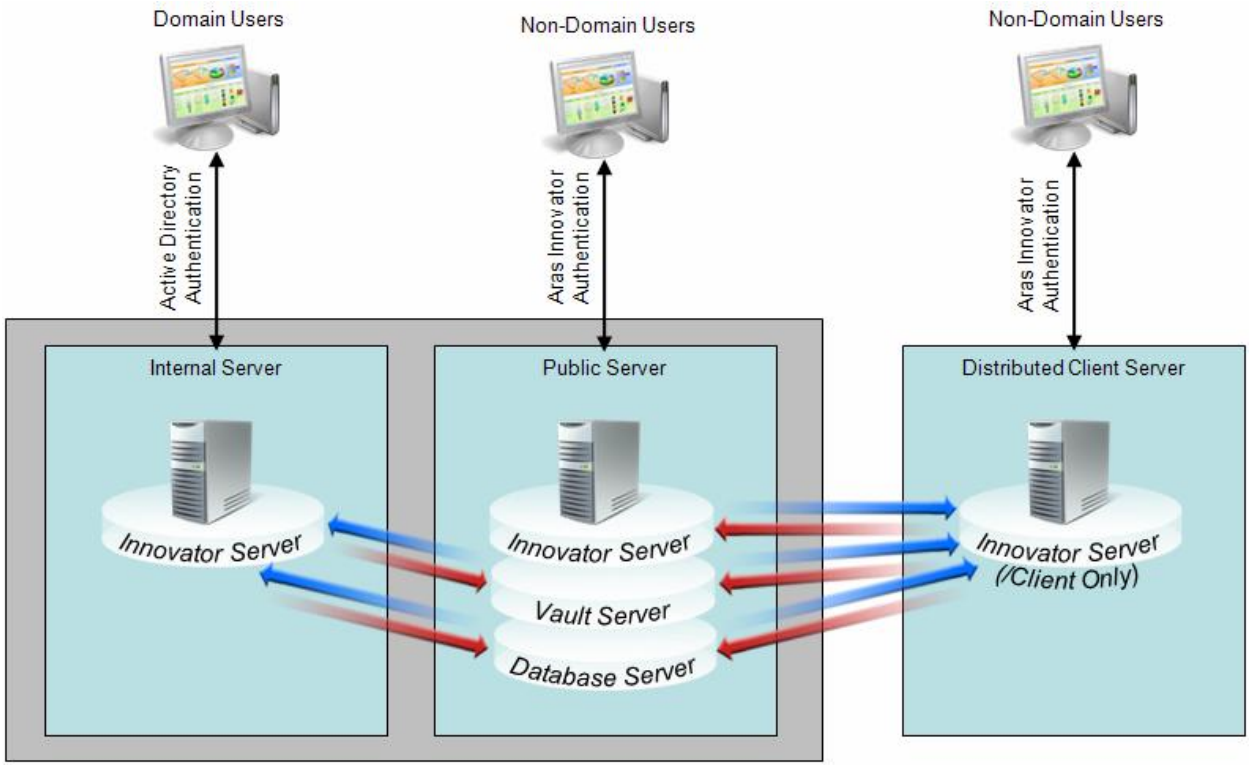
The Innovator Admin and Super User accounts should be changed to prevent them being used by persons who know something about the default values of these passwords by disabling these accounts and only enabling logon during periods controlled by strict configuration management principals. Users should be made members of the Administrators Identity to have administrative privileges assigned on their own account, rather than using the Innovator Admin or Super User accounts.

The Vault Admin user cannot be disabled if the VaultServer feature of Aras Innovator is being used. The best way to restrict access to this account is to generate a random, sufficiently long password as to be astronomically improbable to guess, and to store this password in encrypted form in the VaultServerConfig.xml file.

# 4 Mixing Authentication Methods

Aras Innovator allows for a flexible set of configurations for authentication, and for site structure. By combining the ability to distribute the different tiers of innovator with the different authentication modes, administrators can create a deployment that leverages more than one authentication method.

The following is an example of an existing production deployment of multi-tier mixed authentication control.




In this diagram we have three servers running Aras Innovator.

**Public Server** - This server represents the main instance of Aras Innovator. This server runs the Innovator Server, Database Server, and Vault Server tiers of Aras Innovator. The URL for this server would be used by internal and external users of Aras Innovator, and would deploy the Aras Innovator Security features. Users would authenticate against this server using standard Aras Innovator authentication methods.

**Internal Server** – This server represents a second instance of Aras Innovator on the same network as the Public Server, but does not stand alone. This server only consists of the Innovator Server tier of Aras Innovator, and would refer back to the Public Server when calling the Database Server tier or the Vault Server tier. The URL for this server would be

used by internal and external users of Aras Innovator, and would deploy the logon hooks. Users would authenticate against this server using Active Directory, and would not be subject to the session timeout restrictions of the Aras Innovator Security feature of the Public Server.

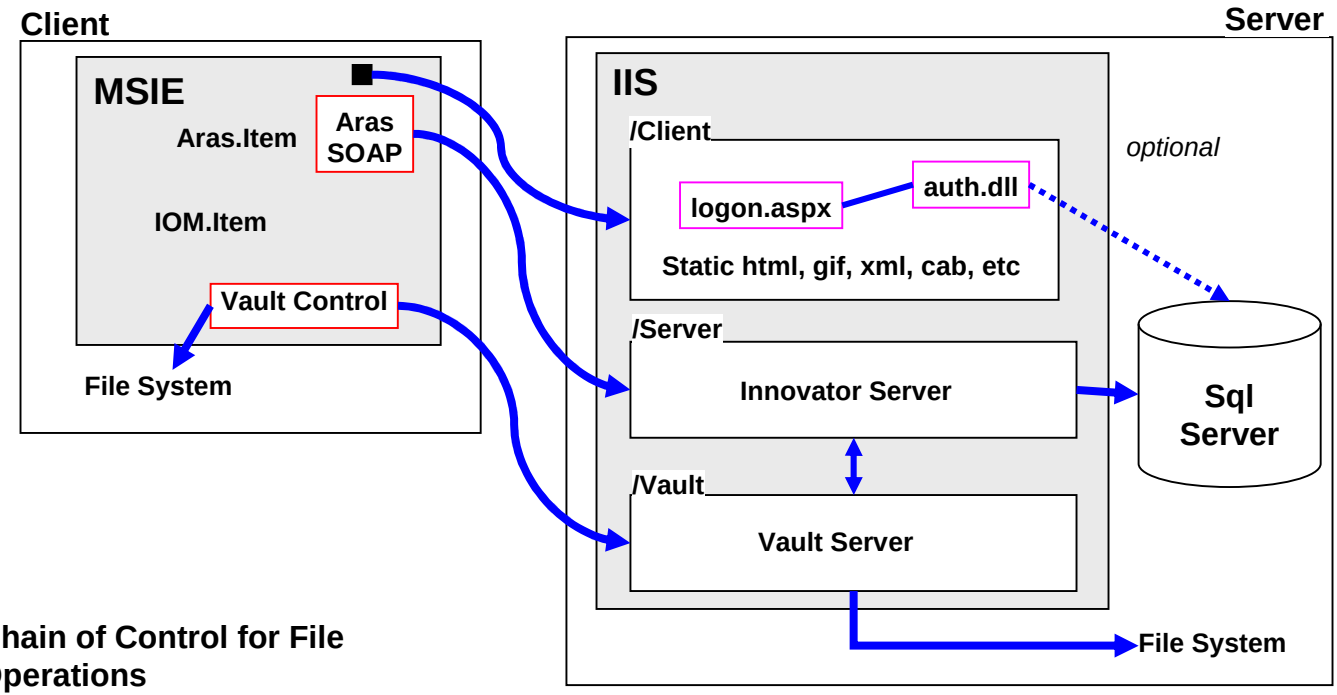
**Distributed Client Server** - This server represents an instance of Aras Innovator deployed on a different LAN than the Public Server. This server only consists of the Innovator Sever tier of Aras Innovator. Furthermore, only the /Client folder would be used from this tier. When calling the /Server folder, all requests would be redirected to the Public Server. As with the Internal Server, all requests would refer back to the Public Server when calling the Database Server tier or the Vault Server tier. This deployment is for two reasons. One, all calls to the /Client folder, like for UI images, would be done on a local network at the remote site, and would help performance over a slow WAN connection. And two, this deployment allows the Public Server to control the Authentication and session management, including Aras Innovator Security features like session timeout.

While this example only represents one configuration possibility, it does represent the flexibility of access control that can be achieved with the Aras Innovator platform.

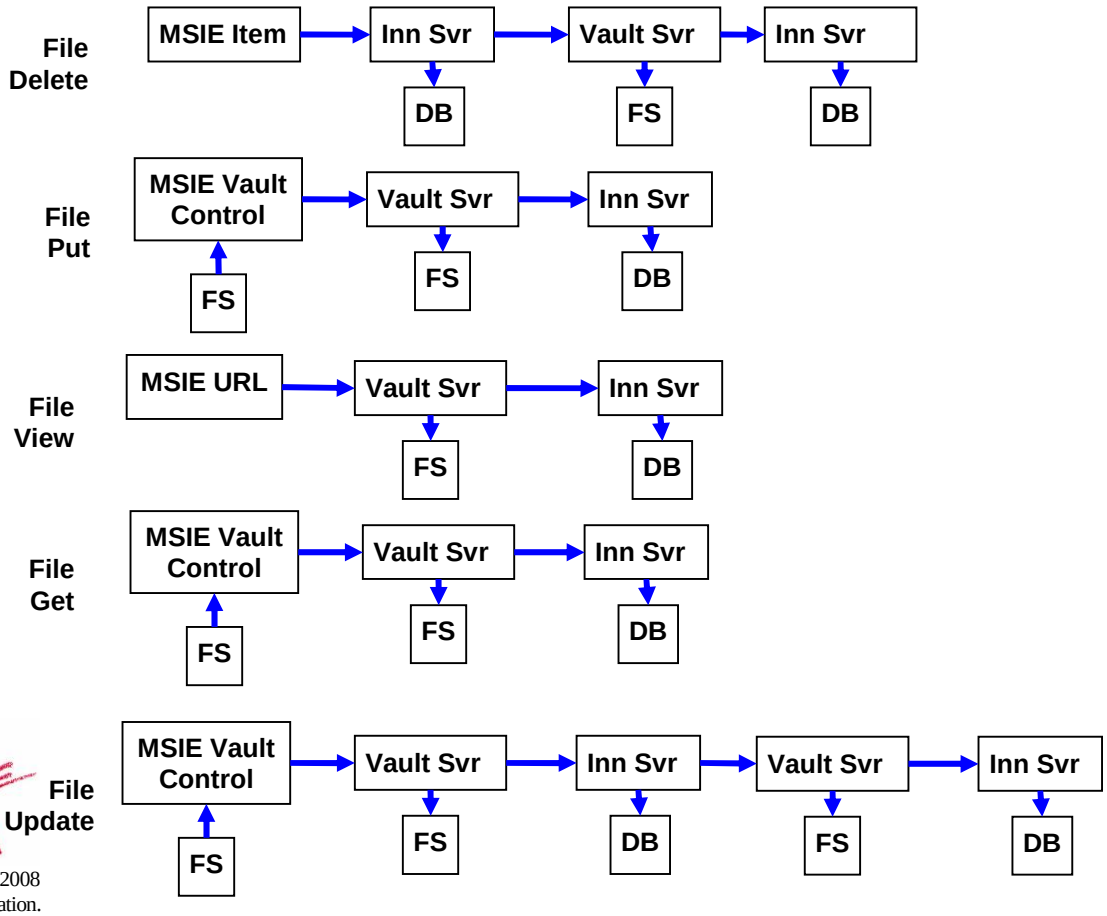
# 5 Reference Diagrams

## 5.1 Architecture

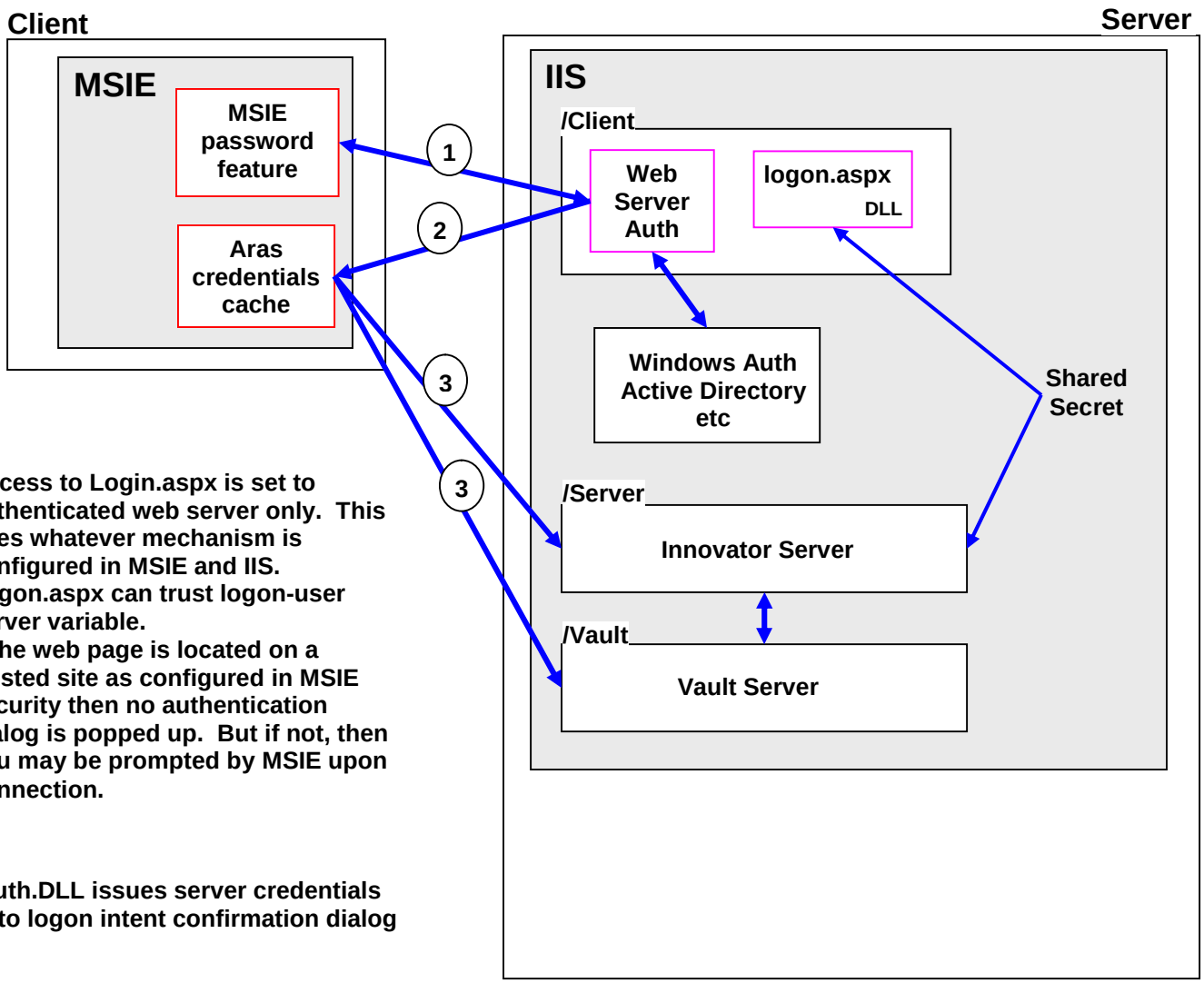
### Aras Innovator Architecture




### Chain of Control for File Operations



## 5.2 Client Logon Hooks Authentication Sequence, Web Server Mode




1

Access to Login.aspx is set to authenticated web server only. This uses whatever mechanism is configured in MSIE and IIS. Logon.aspx can trust logon-user server variable.
If the web page is located on a trusted site as configured in MSIE security then no authentication dialog is popped up. But if not, then you may be prompted by MSIE upon connection.

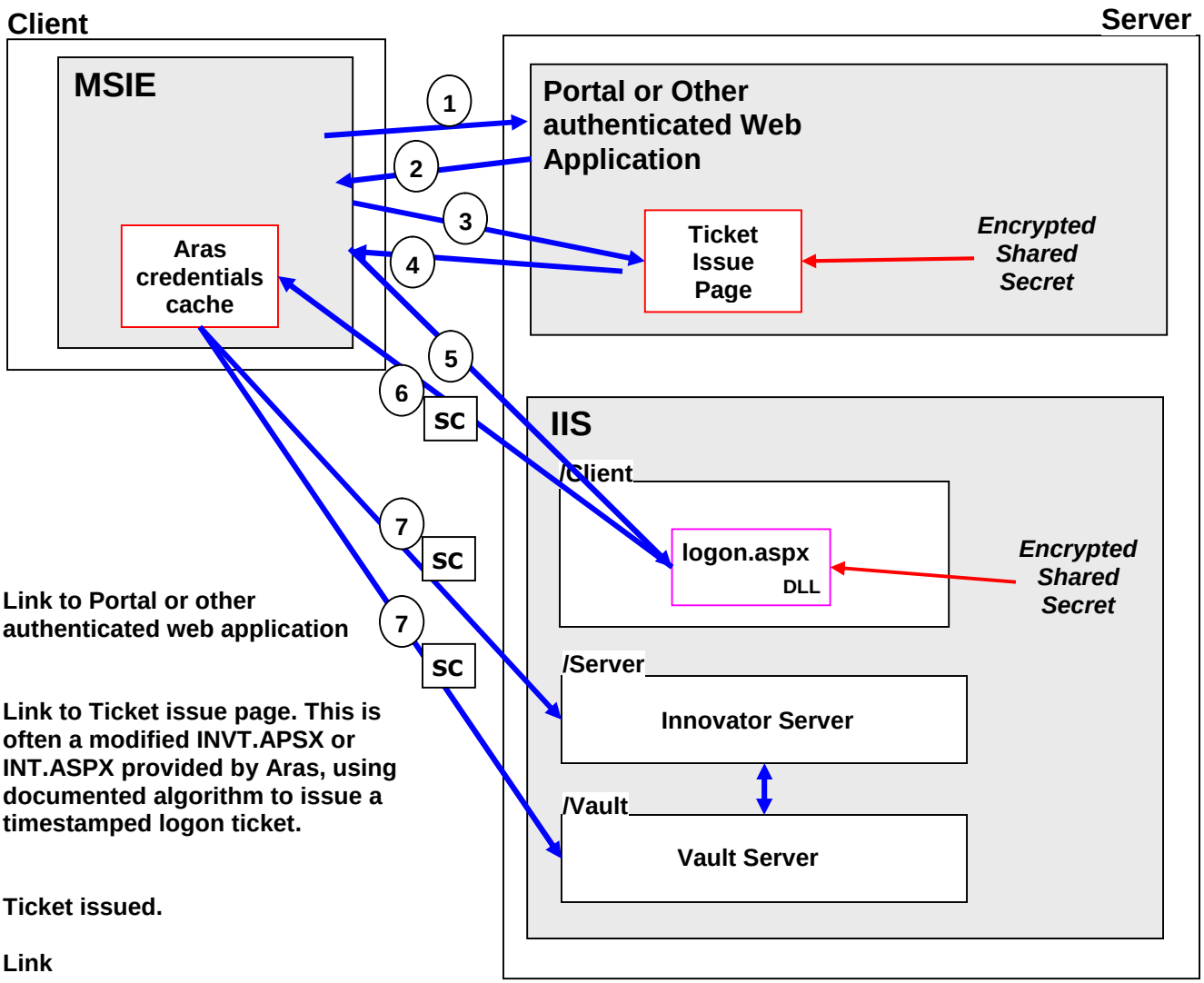
2

Auth.DLL issues server credentials into logon intent confirmation dialog

3

Servers receive credentials as needed, validated against values preset by "reset auth passwords" admin action.

## 5.3 Client Logon Hooks Authentication Sequence, Portal Mode




1 Link to Portal or other authenticated web application

2 Link to Ticket issue page. This is often a modified INVT.APSX or INT.ASPX provided by Aras, using documented algorithm to issue a timestamped logon ticket.

3 Ticket issued.

4 Link

5 Login.aspx uses a configured AUTH.DLL pulls server ticket from DB and/or creates a new server ticket for user [preventing multiple logon by same user]

SC = Server Credentials

6 Credentials issue into the logon/welcome form

7 Used when talking to the Server or Vault